17.383 Microprocessors A
Lab 1
Fall 2010

Introduction to MPLAB IDE, the PIC16F684 and PICC Compilers

Lab Report

See separate report form located on the course webpage. This form should be completed during the performance of this lab.

Objectives

1) To use MPLAB IDE and PICC Lite or the HI-TECH C PRO for the PIC10/12/16 MCU family (Lite) Compilers to demonstrate the following:

   a. Selecting a Device
   b. Creating a Project
   c. Setting Up Language Tools
   d. Naming a Project
   e. Adding Files to the Project
   f. Building a Project
   g. Creating Code
   h. Using the simulator
   i. Program the PIC16F684 using the PICKit 1
   j. Flashing the D0 LED on the PICKit 1
   k. Flashing additional LEDs on the PICKit 1 board

Materials

□	1	Computer (PC) with an available USB Port
□		MPLAB IDE and PICC Lite or PICC PRO (Lite Mode) software installed on PC
□	1	PICkit ™ 1 with USB cable
□	1	PIC16F684
□	1	14 pin Machine Contact Dual In-Line Female Socket

WARNINGS AND PRECAUTIONS

1) Never remove the PIC16F684 from an energized circuit

2) Do not construct circuits while energized

3) Follow electrical safety precautions

17.383 Microprocessors A
Lab 1
Fall 2010

Source File Locations

1. HI-TECH C PRO Toolsuite

    - HI-TECH ANSI C Compiler

        C:\Program Files\HI-TECH Software\PICC\PRO\9.70\bin\picc.exe

2. Microchip MPLASM Toolsuite

    - MPASM Assembler (mpasmwin.exe)

        C:\Program Files\Microchip\MPASM Suite\mpasmwin.exe

    - MPLINK Linker (mplink.exe)

        C:\Program Files\Microchip\MPASM Suite\mplink.exe

    - MPLIB Librarian (mplib.exe)

        C:\Program Files\Microchip\MPASM Suite\mplib.exe

Background Information

MPLAB Integrated Development Environment (IDE) is a comprehensive editor, project manager and design desktop for application development of embedded designs using Microchip PICmicro MCUs and dsPIC DSCs. The basic concepts of the Project Manager, Editor and Debugger will be introduced.

HT-Soft PICC Lite or the HI-TECH C PRO (Lite) compilers are a third party software that used to build/compile Code for the PIC16F684 microcontroller all within the MPLAB IDE.

The PICkit 1 is used as the programmer as well as for testing utilizing the components that have been pre-populated.

In order to create code that is executable by the target PICmicro MCU, source files need to be put into a project. The code can then be built into executable code using selected language tools (assemblers, compilers, linkers, etc.). In MPLAB IDE, the project manager controls this process.

17.383 Microprocessors A
Lab 1
Fall 2010

All projects will have these basic steps:

. Select Device
The capabilities of MPLAB IDE vary according to which device is selected. Device selection should be completed before starting a project.

. Create Project
MPLAB IDE Project Wizard will be used to Create a Project.

. Select Language Tools
In the Project Wizard the language tools will be selected. For this tutorial, the built-in assembler and linker will be used. For other projects, one of the Microchip compilers or other third party tools might be selected.

. Put Files in the Project
Two files will be put into the project, a template file and a linker script. Both of these exist in sub-folders within the MPLAB IDE folder. It is easy to get started using these two files.

. Create Code
Some code will be added to the template file to send an incrementing value out an I/O port.

. Build Project
The project will be built, causing the source files to be assembled and linked into machine code that can run on the selected PICmicro MCU.

. Test Code with Simulator
The code will be tested with the simulator.

. Program the PIC16F684 using the PICKit 1
Finally, the code will be burnt into the microcontroller.

Pre-Lab Preparation

1. Download Lab # 1 from the course website. Read and understand the lab.
2. Part 2 of this lab will be performed during the next class. Develop the required code prior to the next class.
3. Read the various course reference materials (PIC16F684 data sheet, HI-TECH C PRO manuals, PICkit 1 material, etc).

17.383 Microprocessors A
Lab 1
Fall 2010

Procedure

Experiment 1. INTRODUCTION

Step 1. START MPLAB IDE

a. Start MPLAB IDE by double clicking on the icon installed on the desktop or select *Start>All Programs>Microchip>MPLAB IDE vx.xx>MPLAB IDE*. A screen will display the MPLAB IDE logo followed by the MPLAB IDE desktop.

Step 2. CODE DEVELOPMENT

a. Open the text editor by selecting:

File>New

PROGRAM LISTING 1 (located at the end of this lab) will now be typed into the text editor window that was just opened. Ensure that ...

a. The code is entered correctly as most problems are the result of incorrect translation of the code from the listing to the editor.

b. The Code listing has comments throughout the program. The structure of the comments will be consistent during this semester. Ensure that you enter your name, the semester, date as well as ensure that the Program Title and Program File Name change as modifications are made to your code.

Note: Line numbers may be toggled on/off by right clicking in the editor window, selecting *Properties* and then checking/unchecking Line Numbers on the **File Type** tab of the Editor Options dialog. Click Apply, then click OK.

b. Save the program as "c:\ Student Data Area Drive C\Lab_1A.c" by doing one of the following:

File>Save

Click the Save File icon on the Project toolbar.

c. Note that different parts of the program are displayed using different colors.

d. Close the window

Step 3. NAMING THE PROJECT

a. Name the project by doing one of the following:

Project>New …

Click the New Project icon on the Project toolbar

b. Project Name: Lab_One

c. Project Directory: c:\ Student Data Area Drive C

Step 4. SETTING UP LANGUAGE TOOLS

a. The application is written in the C programming language for the HI-TECH C PRO.

b. To specify the HI-TECH C PRO Compiler do the following:

1. *Project>Select Language Toolsuite …*
2. Scroll through the Active Toolsuite to find …

   HI-TECH Universal Toolsuite
3. Verify that the location of picc.exe for the PICC PRO Compiler is correct (see source file locations at the beginning of this lab)
4. Click … **OK**>

c. To specify the HI-TECH PICC Lite Compiler do the following:

1. *Project>Select Language Toolsuite …*

2. Scroll through the Active Toolsuite to find …

   HI-TECH PICC Toolsuite

3. Verify that the location of picc.exe for the HI-TECH C PRO (Lite) is correct (see source file locations at the beginning of this lab)

4. Click … **OK**>

We will not be using the Microchip MPLASM Toolsuite for this lab, therefore only select the HI-TECH PICC Toolsuite as indicated above

Step 5. ADDING FILES TO THE PROJECT

a. Open the Lab_One.mcw window by selecting …

View

… then click on Project. The window will appear.

b. Right click on Source Files in the Lab_One.mcw window. Select

Add Files …

c. Locate the Lab_1A.c file previously created and saved above

d. Click **Open>**

This will associate Lab_1A.c with the project that was just created

TIPS:
1. Files can be added and projects saved by using the right mouse button in the project window.
2. In case of error, files can be manually deleted by selecting them and using the right mouse click menu.
3. Files can also be added by selecting *Project>Add Files to Project …* and performing c and d above.

Step 6. SELECTING THE PIC16F684 MCU

a. Click *Configure>Select Device …*

b. Find the **PIC16F684** in the list

The lights indicate which MPLAB IDE components support this device.

A green light indicates full support.

A yellow light indicates preliminary support for an upcoming part by the particular MPLAB IDE tool component Components with a yellow light instead of a green light are often intended for early adopters of new parts who need quick support and understand that some operations or functions may not be available.

c. A red light indicates no support for this device. Support may be forthcoming or inappropriate for the tool, e.g., dsPIC DSC devices cannot be supported on MPLAB ICE 2000.

d. Click **OK>**

Step 7. BUILDING THE PROJECT

a. To build the project, select either:

Project>Build All

Right click on the project name in the project window and select *Build All*

Click the ***Build All*** icon on the Project toolbar.

The Output window shows the result of the build process. There should be no errors on any step. If any errors occur, go over the code keyed in and compare it to the listing. This is the most likely source of the problem.

The Warnings/Advisories listed will not interfere with the operation of the lab program. They are merely identifying a directive that has been deprecated, i.e., is being discontinued in favor of another.

b. CAPTURE YOUR OUTPUT WINDOW FOR SUBMISSION by:

1. Right click in the Output window and select *Select All>*

2. Right click in the Output window and select *Copy>*

You can also use Window capture ...

3. Window capture is simply ALT + Print Scrn, then paste into a document.

4. Paste the information in your lab results file

Step 8. TESTING CODE WITH THE SIMULATOR

a. Select the simulator as the debug execution tool by:

Selecting *Debugger>Select Tool>MPLAB SIM*

b. After selecting MPLAB SIM, the status bar on the bottom of the MPLAB IDE window will change to MPLAB SIM. In addition, the following changes also occur:

1. The status bar on the bottom of the MPLAB IDE window should change to MPLAB SIM.

2. Additional menu items now appear in the Debugger menu.

3. Debug Tool Bar appears.

TIP: Position the mouse cursor over a toolbar button to see a brief description of the button's function.

4. An MPLAB SIM tab is added to the Output window.

5. Ensure that Trace is not selected by

Selecting *Debugger>Settings...*

Under **Trace Options**, ensure that **Trace All** is NOT selected.

6. There are shortcuts (Table 1) for commonly used functions in the Debug Tool Bar.

TABLE 1: DEBUG SHORT CUT ICONS

Debugger Menu	Toolbar Buttons	Hot Key
Run		F9
Halt		F5
Animate		
Step Into		F7
Step Over		F8
Step Out		
Reset		F6

TIP: Click on the appropriate icon on the toolbar or use the hot key shown next to the menu item. This is usually the best method for repeated stepping.

c. Run the simulated program at full speed by doing one of the following:

Selecting *Debugger>Run*

Click the ***Run*** icon on the Project toolbar

Press **F9** on the keyboard

At the bottom left of the Project Window ***Running ...*** with a green indicator bar will appear.

NOTE: If the ***Running ...*** and green indicator bar only show for an instance, you have the Trace Option selected. Deselect this Option as outlined above.

Make note as to what other windows or features have changed, if any.

d. Experiment with the following other features of the MPLAB simulator and make note as to what they do:

Halt
Animate
Reset
Setting Breakpoints
Watch Window ... *View>Watch*
Stopwatch (see next step for details)

e. When calculating delay times, the data book can be used to determine how long each instruction will take in the delay loop, this will provide a fairly accurate number. You can also use the MPLAB IDE StopWatch to measure the delay. To measure the delay_routine time, do the following:

1. Change the delay_routine by adding two ***NOP();*** statements as indicated in the Table below:

```
/*******************************************************
* Function: delay_routine
*
* Description:  Causes a delay in program execution
*
* Notes:
*
* Delay was determined through trial and error
*
* Returns:  None
*
*******************************************************/
void delay_routine(void)
{
      int i, j;
            NOP();
            for (i = 0; i < 255; i++)
            for (j = 0; j < 140; j++);
            NOP();
      return;

}
/******** END OF delay_routine ************************/
```

2. Re-build your Code.

3. Set a break point at each ***NOP();*** statement

4. Verify that the Processor Frequency is set for 4 MHz by *Debugger>Settings>OSC/Trace Tab>Processor Frequency.* Set the value to 4 MHz.

5. Use *Debugger>StopWatch* to bring up the StopWatch dialog.

6. Press *Debugger>Reset* and then *Debugger>Run* to halt at the first ***NOP();*** instruction. With the default processor frequency of 4 MHz, the StopWatch should show that it took 21.000000 microseconds to reach the first breakpoint.

7. In the Stopwatch window, press Zero to rest the stopwatch

8. Execute Run again to get to the next ***NOP();*** instruction, and note that the StopWatch shows that it took about 914.178000 milliseconds. To change this, you can change the values in the delay loop.

f. Stop the simulator from executing by doing one of the following:

Selecting *Debugger>Halt*

Click the ***Halt*** icon on the Project toolbar

Press **F5** on the keyboard

Step 9. TESTING CODE WITH THE PICkit 1

WARNING: Do not connect the PICkit 1 starter kit to the computer using the USB cable until directed.

a. Turn off the simulator and return to the MPLAB Project window by

Selecting *Debugger>Select Tool>None*

MPLAB Sim should disappear from the lower left of the Project Window

b. If needed, close all Watch, Special Function Register, Stop Watch, etc. **LEAVE OPEN** the MPLAB Desktop, Lab_One.mcw and Output windows.

c. Verify that the language tool and source files are the same as Step 4 above. If they are not, restore to those values.

d. Remove the ***NOP();*** instructions previously placed in the code.

e. Build the project.

f. Install the 14 pin Machine Contact Dual In-Line Female Socket in the PICkit 1 evaluation socket. **BE CAREFUL** not to bend any of the pins.

g. Install the PIC16F684 into the 14 pin Machine Contact Dual In-Line Female Socket that was previously installed in the evaluation socket. **BE CAREFUL** not to bend any of the pins.

h. With the program compiled, plug your PICkit 1 starter kit into your computers USB port (there should be a USB cable readily assessable on your lab bench that you can plug into). The following window should appear:

An Output window appears with the Firmware Version displayed. IF it does not display, do the following:

Select *Programmer>Select Programmer>PICkit 1*

The Output window with the Firmware Version should now be displayed.

i. Program the device by doing one of the following:

Select *Programmer>Program Device*

Click the ***Program*** icon on the Project toolbar

The programming operation will take a few moments. The operation is indicated by a growing bar at the bottom left corner of the desktop.

When programming has been complete, a "Program Succeeded" message will appear in the status window.

Once the programming has completed, the D0 LED of the PICkit 1 starter kit will start to flash.

If it does not flash, review your source code and repeat the process above for creating the project and programming the device.

j. When the PICkit 1 is operating, notify the instructor so it can be observed.

k. Submit a copy of your code with your lab report.

Experiment 2. MODIFYING THE CODE – LIGHTING MULTIPLE LEDs

Step 1. TURN ON TWO LEDs WITH THE PICkit 1

a. Modify the LEDFlasher code as follows:

Light D1 for approximately 0.5 seconds
Turn D1 off for approximately 0.5 seconds
Turn on another LED (D2 through D7 … pick one) for approximately 0.5 seconds
Turn off the chosen LED for approximately 0.5 seconds
Repeat the process (a continuous loop)

HINT: You will need to review the schematic, figure 1, for the PICkit 1 to understand what controls each LED. It is best that only one LED is turned on at a time. This is due to the way the circuit connecting the LEDs was created. Question 1 creates a table that will aid you in identifying the connections and code required to light each LED. This chart will be useful throughout the semester.

b. Create a new Project. Run the Project using the simulator. Once the code runs on the simulator, Program the device (PIC16F684) and verify that it does what was requested. Filename for the new Code will be Lab_1B.c.

c. When the PICkit 1 is operating as designed, notify the instructor so it can be observed.

d. Submit an electronic copy of your modified code via email as well as a printed copy with your lab report.

Experiment 3. SOFTWARE STRUCTURE

Step 1. Rewrite (new filename for the new Code will be Lab_1C.c.) and verify by downloading to the PICkit 1 the software listed in Program Listing 1 such that the main() Function is the first Function of the program. HINT … use the Function Prototype.

a. When the PICkit 1 is operating as designed, notify the instructor so it can be observed.

b. Submit a copy of your modified code with your lab report.

17.383 Microprocessors A
Lab 1
Fall 2010

SUMMARY:

You have performed the major steps for creating, building and testing a simple project utilizing MPLAB IDE and HI-TECH PICC Lite or the HI-TECH C PRO (Lite) compilers. Tasks completed include:

Code development
Naming the Project
Setting up the language tool
Adding files to the project
Selecting the device
Building the project
Testing with the simulator
Using the PICkit 1 to "burn" the code into the PIC16F684
Using the schematic and programming techniques, you were able to modify the code to flash two LEDs in sequence

These are the essential steps for getting started with MPLAB IDE and the PIC microcontroller.

General Lab Report Requirements:

- Using the windows Print Screen feature, ensure that all the identified windows have been copied and recorded in a file for your Lab Results report submission.

- Ensure that you have recorded all the data requested during the lab in your lab notebook as well as your lab report.

Question:

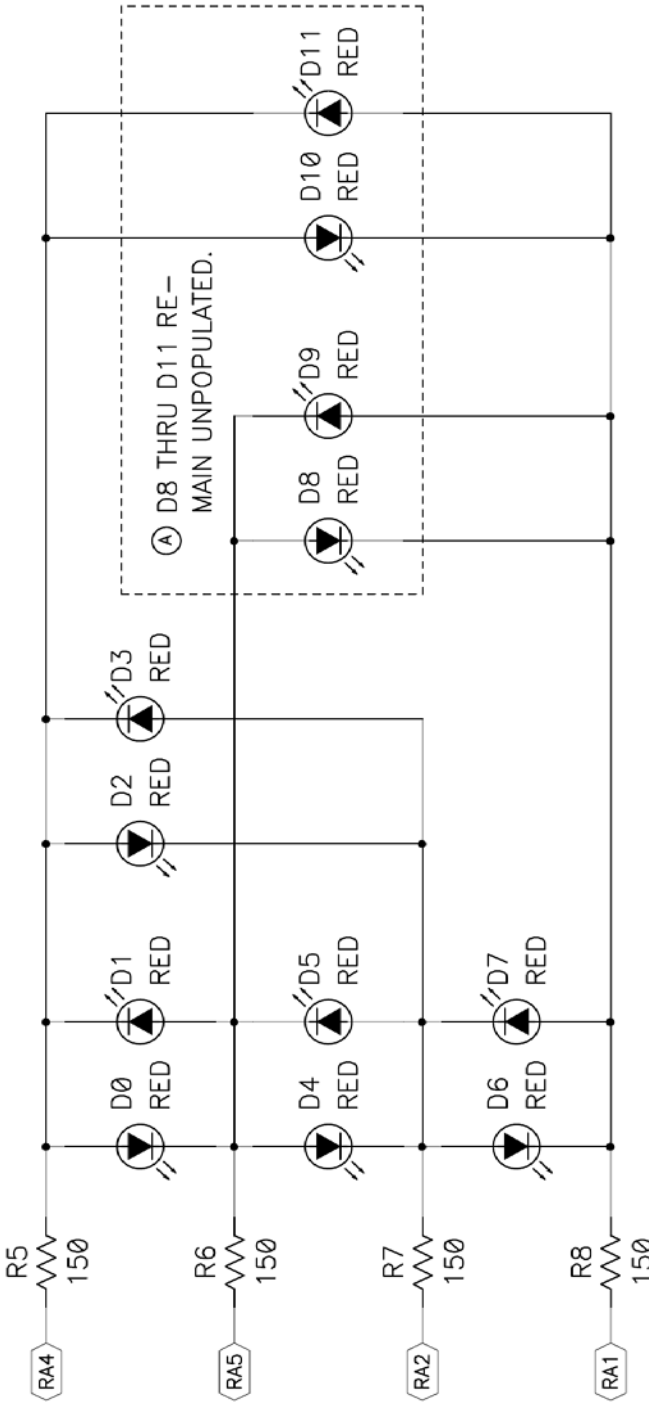
1. Fill out Table 2 for each LED connection listed in Figure 1.

2. What are the basic steps in creating a MPLAB project?

3. Why is it best that only one LED on the PICkit 1 be turn on at a time? Is it possible to turn on multiple LEDs on the PICkit 1? If yes, can they all be turned on at the same time, why? If no, why?

4. Why do different parts of the program change colors when the file is saved?

Figure 1 - PICkit 1 Starter Kit LED Layout



17.383 Microprocessors A
Lab 1
Fall 2010

PROGRAM LISTING 1

```
#include <pic.h>

/*******************************************************
*
* Program Title:     Flash D0
*
* Program File Name: Lab_1A.c
*
* Microprocessors A 17.383
*
* xxxxxxxx - Put in Semester (i.e. Fall 2010) here
*
* xxxxxxxx - Put in your name here
*
* xx/xx/xx - Put date here
*
*******************************************************/

/*******************************************************
* Function: PORT_init
*
* Description:  Initializes PORTA to a known condition
*
* Notes:    None
*
* Returns:  None
*
*******************************************************/
void PORTA_init(void)
{
      PORTA = 0;           //  All PORTA Pins are low
      CMCON0 = 7;          //  Turn off Comparators
      ANSEL = 0;           //  Turn off ADC
      TRISA = 0b001111;    //  RA4 and 5 are outputs; RA0,1,2, and 3 are input

      return;

}
/******** END OF PORTA_init ***************************/

/*******************************************************
* Function: delay_routine
*
* Description:  Causes a delay in program execution
*
* Notes:
*
* Delay was determined through trial and error
*
* Returns:  None
*
*******************************************************/
void delay_routine(void)
{
      int i, j;

            for (i = 0; i < 255; i++)
            for (j = 0; j < 255; j++);

      return;
```

PROGRAM LISTING 1 Con't
Next page

PROGRAM LISTING 1 Con't

```
}
/******** END OF delay_routine ************************/



/*******************************************************
* Function: main
*
* Description:  D0 on PICkit 1 will Flash on and off
*
* Notes:
*
* RA4 - Positive LED Connection for D0
* RA5 - Negative LED Connection for D0
*
* Returns:  This routine contains an infinite loop
*
*******************************************************/

/* Configuration Word */

__CONFIG(INTIO & WDTDIS & PWRTEN & MCLRDIS & UNPROTECT \
  & UNPROTECT & BORDIS & IESODIS & FCMDIS);

main()
{

      PORTA_init();

      while(1 == 1)                 //  Loop Forever
      {

            delay_routine();

            RA4 = 1;                //  D0 LED On by making RA4 high

            delay_routine();

            RA4 = 0;                //  D0 LED Off by making RA4 low

      }                             // *** END OF While (1 == 1) LOOP

      return;

}

/******** END OF main ROUTINE *************************/
```

Table 2 - PICkit 1 Starter Kit LED Chart

PICkit 1 LEDs

To Light LED Dx, RAx values

	RA5	RA4	RA3	RA2	RA1	RA0
D0	0	1				
D1						
D2						
D3						
D4						
D5						
D6						
D7						

PORTA

	RA5	RA4	RA3	RA2	RA1	RA0	PORTA (binary)	PORTA (HEX)
D0	0	1					0b010000	0x10
D1								
D2								
D3								
D4								
D5								
D6								
D7								

TRISAx - 0 is an output; 1 is an input;

	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0	TRISA (binary)	TRISA (HEX)
D0	0	0	1	1	1	1	0b001111	0x0f
D1								
D2								
D3								
D4								
D5								
D6								
D7								