PIC PROGRAMMING PROCEDURE:

1. Open MicroCode Studio

Double click on the MicroCode Studio desktop icon

or select from the *Start* menu:

Programs | *MicroCode Studio (MCSX)* | *MicroCode Studio (MCSX).*

2. Create or Open Your PicBasic Pro Program

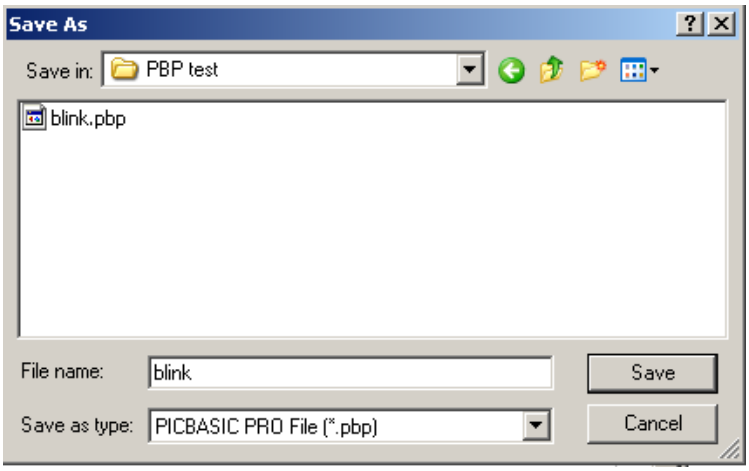
If you are starting a new project, either edit the file that comes up by default, use the code template available on the Lab website, or select *File* | *New* to start from scratch. We recommend always starting the code template when using the PIC16F88.

If you want to edit an existing project, select *File* | *Open* and browse to your code file. The file can be created initially in any text editor (e.g., Windows NotePad or Microsoft Word, saving the file as "Plain Text: *.txt").

Note - To disable Microcode Studio's command case changing, select *View* | *Editor Options ...*, click on the *Highlighter* tab, and under *Reserved word formatting*, select *Default*.

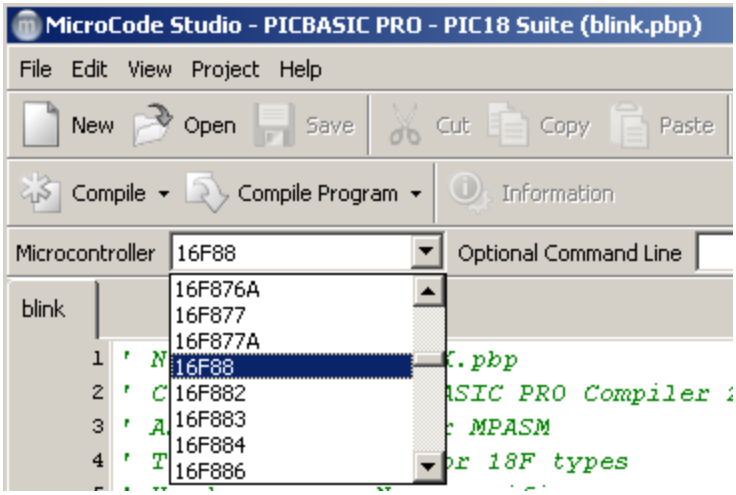
3. Save and Name Your Project File

Save the file to the folder where you want to store your project. Make sure you select the appropriate drive (e.g., your U-drive) in the *Save in* pull-down box. Use either PICBASIC PRO file (*.pbp) or BASIC file (*.bas) as the file type. **NOTE - Do not use periods in your file name.**



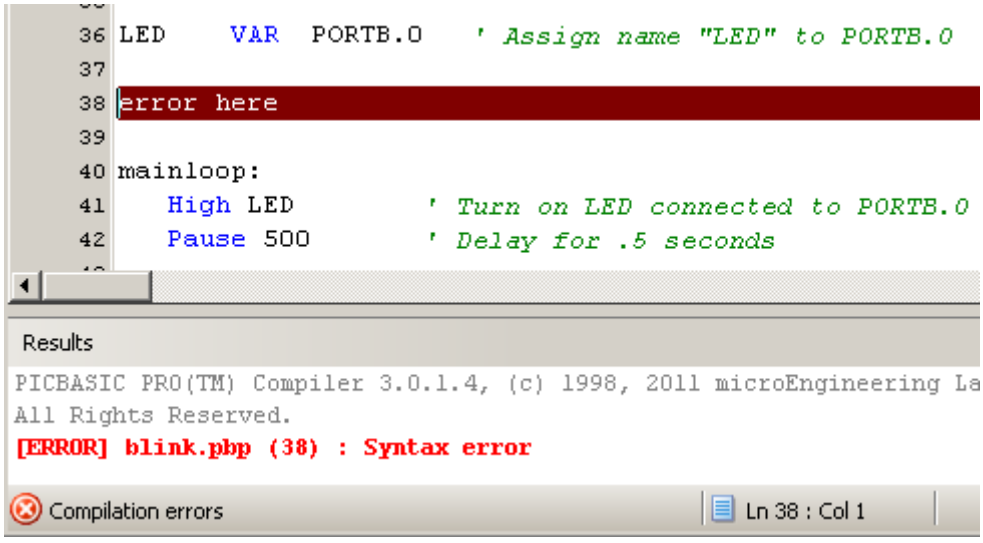
4. Choose the PIC Device You are Using

Select the appropriate PIC microcontroller (usually the 16F88) from the pull-down box in the *Microcontroller* (MCU) toolbar. MicroCode Studio and the U2 Programmer support only the devices listed.



5. Check For Errors

To make sure there are no errors in your code, click on the *Compile* button on the *Compile and Program Toolbar*. If there are any errors, MicroCode Studio will identify and locate them. Here's an example:



To have the line #'s appear in the editor window (if they aren't there already), select *View* | *Editor Options* ... and check the *Show line numbers in left gutter* box.

Correct any errors found in the code and *Compile* again until there are no more errors. After a successful compile, the status line at the bottom of the window will read "Success" and indicate how much memory your program is using on the PIC.

6. Prepare the PIC for Programming

Make sure the USB cable is plugged into the U2 Programmer. The green LED in the device should be on.

Make sure the metal lever on the U2 Programmer ZIF socket is in the up position.

NOTE - Always support the programmer socket with your spare hand while pivoting the lever up or down.

Insert your PIC into the socket with pin 1 in the position indicated on the socket board. **Make sure the PIC is in the correct orientation.**

NOTE - The "Pin 1" position is different depending on the # of pins on your PIC, as indicated on the green U2 socket board. The required ribbon cable connector position is also different.

Pivot the socket lever down to lock the PIC in place.

Make sure the PIC is positioned and oriented in the programmer properly before continuing.

7. Prepare the Code For Download Onto the PIC

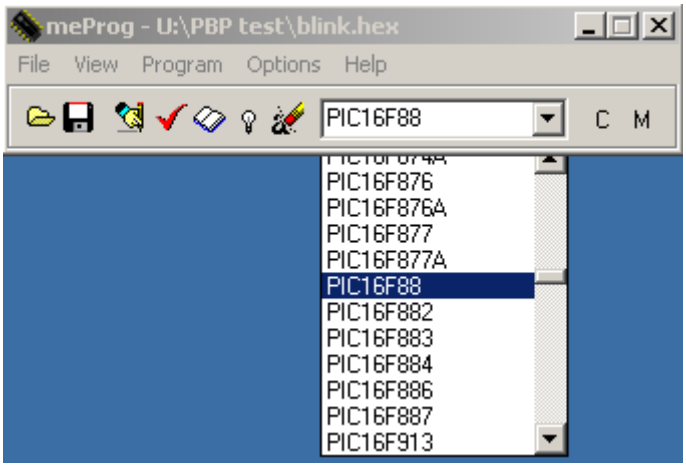
Click on the *Compile Program* button to compile the code and generate the files needed for programming the PIC.

This will launch the meProg utility that allows you to store the code on the PIC. The following window will appear:

NOTE - The window may take a while to appear, especially the first time you compile, while the software generates the files and searches for the U2 hardware, so be patient.

8. Identify the PIC Model Number

The PIC device number should transfer from Microcode Studio, but you should still verify this and change it if necessary in the meProg window pull-down list.



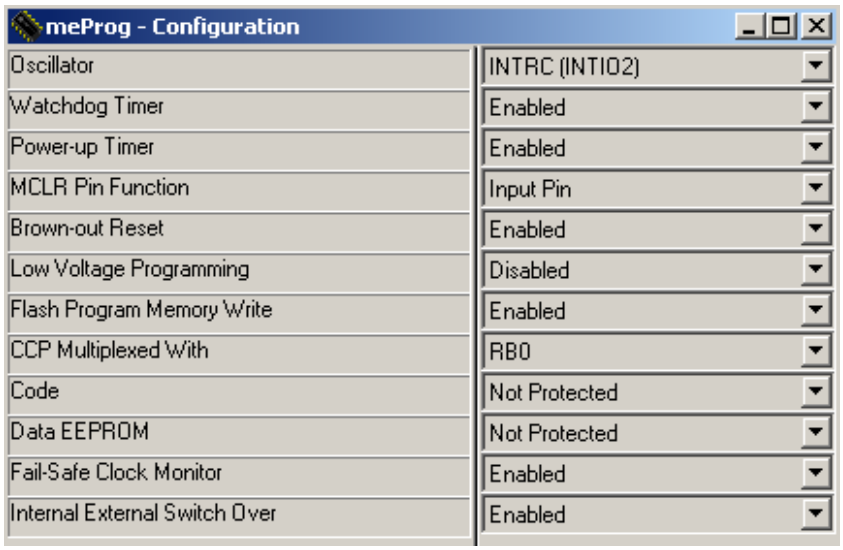
9. Select the Appropriate Configuration Bit Settings

Again in the meProg window, Select *View* | *Configuration* (or click on the "C" on the toolbar) to display the Configuration window (if it isn't visible already).

Click on the down-arrows to select the desired or appropriate choice for each feature listed.

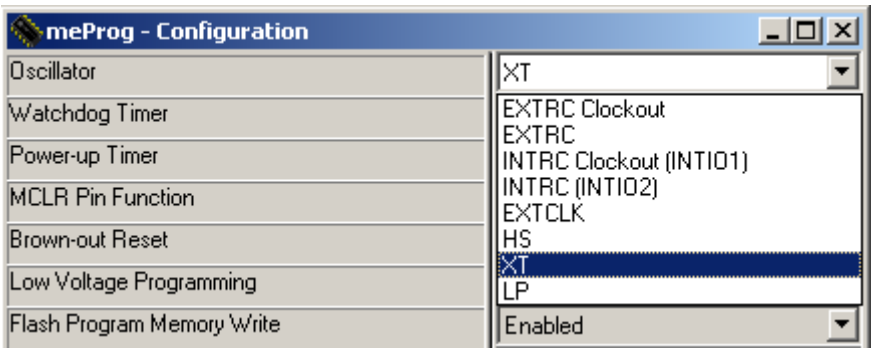
NOTE - The configuration choices need to be set to the desired values every time you recompile your code, unless you define them in your code, as described in the next section.

Typical choices (e.g., for the PIC16F88) are shown below.



With many PICs, some pins offer multiple functions, and you indicate the desired function with the configuration setting. For example, the MCLR pin can be used to activate a reset of the PIC, but it can also be used as an additional I/O pin (RA5):

And many PICs offer many options for the type of oscillator used. For example, if you wanted to use a more-accurate external crystal oscillator, or if you were using a PIC that did not have an internal oscillator, you would want to select the XT option:



To learn about the different features and choices listed in the Configuration window, refer to appropriate sections in the datasheet for the specific PIC you are using.

NOTE - Depending on how multi-function pins are being used, bits in certain registers (e.g., OSCCON, ANSEL, and ADCON) must also be set in your code to have the functions operate as desired. For example, with the PIC16F88, to use PORTA pins for digital I/O, the ANSEL bits must be set to 0. See the relevant sections in the PIC datasheet for more information.

10. Changing Configuration Settings in Code

An alternative to setting the configuration bits manually, as described in the previous section, is to set them within your program.

You only need to add code for the settings for which the default values are different from what you want.

For example, you can automatically achieve the settings shown in the previous section for the PIC16F88, by adding the following code to your program (or by using the code template on the Lab book website that already contains the code):

```
#CONFIG
    __CONFIG _CONFIG1, _INTRC_IO & _PWRTE_ON & _MCLR_OFF & _LVP_OFF
#ENDCONFIG
```

Note that there are two underscores in front of the "CONFIG" and only one underscore in front of the "CONFIG1." There is also a comma between "CONFIG1" and the settings. All settings, including any that might be added, are separated by the bitwise AND operator (&).

The settings available for a given PIC can be found in the appropriate *.INFO file for the device. These files can be found in: *C:\PBP3\DEVICE_REFERENCE*.

11. Download Your Code Onto the PIC

After all of the configuration choices have been set to the desired values, click on the Program icon

or select *Program* from the *Program* menu in the meProg window.

The U2 programmer LED will glow red while the code is being downloaded, and it should glow green again when the process is completed.

After the program is written and verified, a *Program Verify complete* dialog box should appear, indicating that everything worked properly. Click on *OK*.

NOTE - Never insert or remove a PIC when the LED glows red. This can cause damage to the chip and/or the programmer.

12. Remove and Test the Programmed PIC

Lift the lever on the programmer to release the pin clamp. Then remove the PIC from the socket and insert it into your circuit for testing.

13. Shutdown the Software and Logoff

Close (Exit) the MicroCode Studio application. The programmer and configuration windows will close automatically with MicroCode Studio.

Be sure to log off your session on the PC so others won't use (and/or abuse) your account.