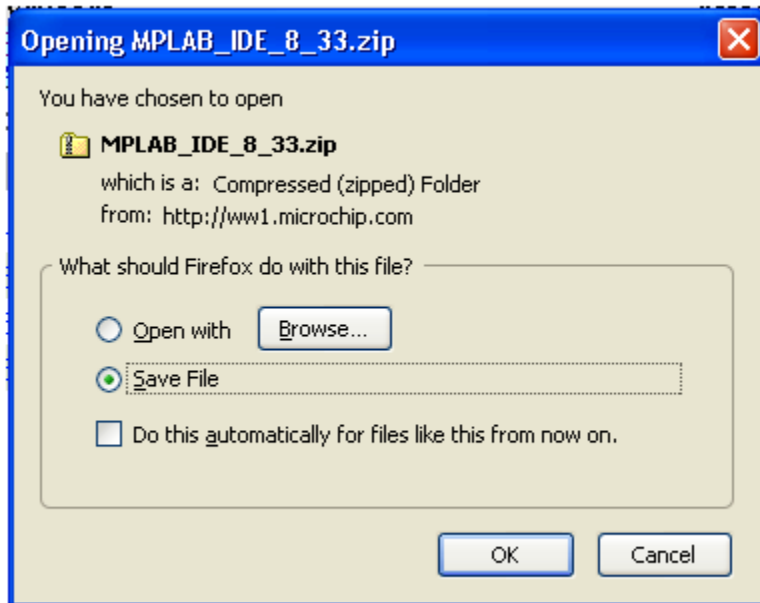
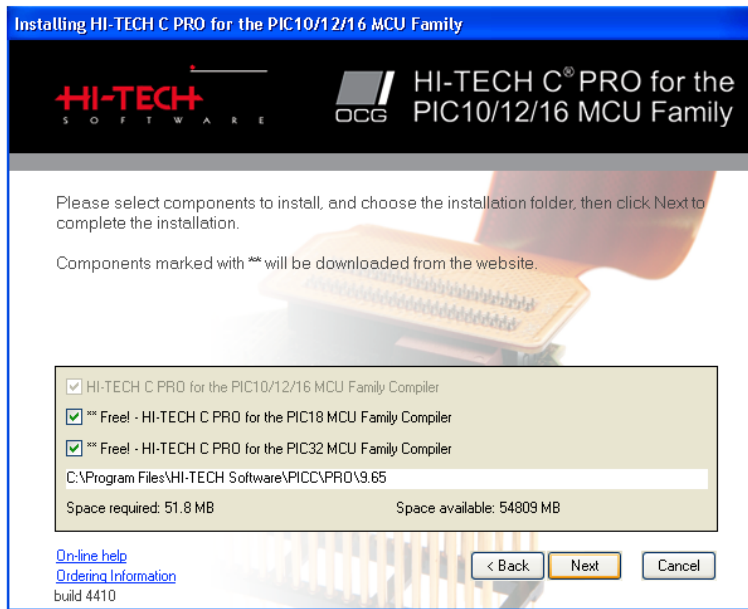


Downloading and Installing MPLAB IDE v8.33

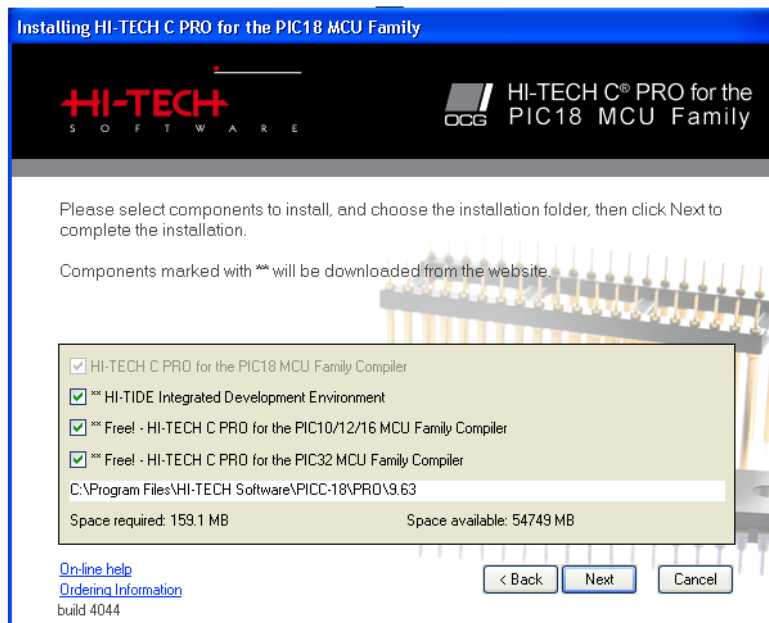
1. The free download can be found on the MicroChip website: <http://www.microchip.com/mplab>
2. Save the zip file somewhere you can easily access it, like the desktop



3. When the file is done downloading, select the “Open” option, which will direct you to extract the files from the zipped folder.
4. Select **Next**, and then **Next** again until all files are extracted.
5. Click on the box to show extracted files, and select **Finish**.
6. When the MPLAB IDE 8.33 folder opens, double click on the Install_MPLAB_8_33 icon.
7. If prompted for a security warning, select Run which will begin the InstallShield Wizard.
8. When the MPLAB Tools window appears, select **Next**, making sure that all previous versions of MPLAB are uninstalled, all applications are closed down, and any anti-virus software is disabled for the time being.
9. Select “I accept the terms of the license agreement” and click **Next**.
10. For the sake of this class, select the Complete setup type to ensure all components of the program are installed. Select **Next**.
11. When the Application Maestro License window appears, select “I accept the terms of the license agreement” and select **Next**. Repeat for the MPLAB C32 License window, and select **Next**.
12. If the HI-TECH C PRO for the PIC10/12/16 MCU Family Installer window appears, select **Yes**, then **Next**, and select “I accept the terms of the license agreement,” then **Next** again.
13. When the Component Installer window appears, select all available compilers and select **Next**.



14. You will be prompted to select a language (English). Select “Add to environment path” and click **Next**, and then **Next** again when the program installer window for the PIC18 MCU Family appears.
15. Select “Operate in Lite mode (all other options are only evaluation packages, or require a serial number from a new PIC).
Repeat these steps for the PIC18 MCU Family:



16. When the HI-TIDE window appears, select **Next**, then “I accept the terms of the license agreement,” and **Next** again.
17. In the next window, select the HI-TECH CJTAG Debug-Interface Driver and select **Next**. Select English as the primary language, and add it to the environment path, then select **Next**.

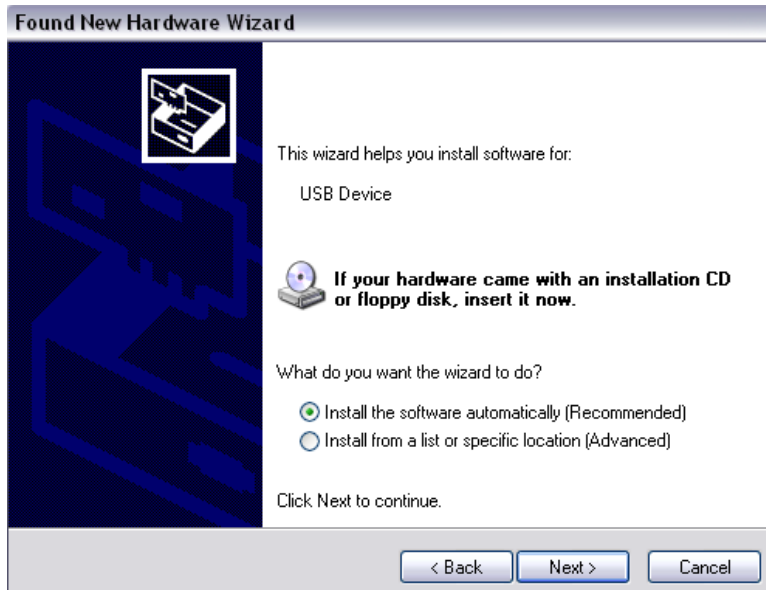
18. In the final window, select the **Run Java Runtime Environment Installer**, and select **Finish**.
19. The J2SE Runtime Environment 5.0 Update 12 window will appear where you should select “Typical setup” and then “Accept.” When this finishes, select **Finish**.
20. Select **Next** to install the HI-TECH C PRO for the PIC10/12/16 MCU Family, then “Operate in Lite mode, and **Next**. After accepting the terms of the license agreement, select **Next**, and select all of the boxes before clicking **Next**.
21. Select **Next** to install the HI-TIDE version 3.15PL2, Choose English, and add it to the environment path. Select the HI-TECH CJTAG Debug-Interface Driver and select **Next**.
22. Select “Run HI_TIDE now” and then **Finish**. Repeat these steps for the PIC18 MCU Family

Installing the Right Drivers

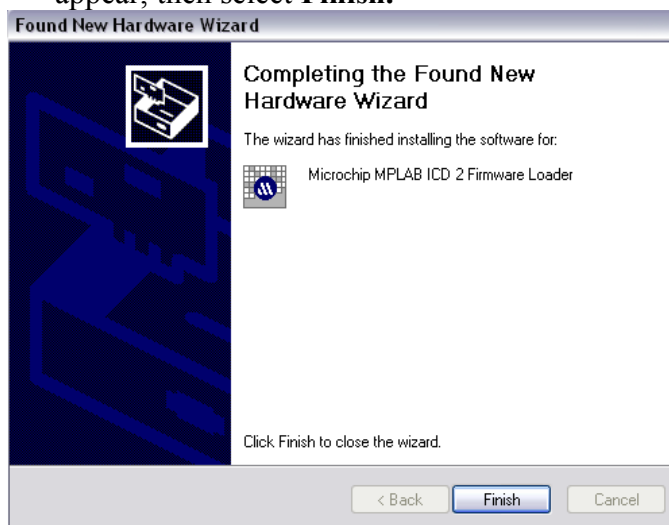
1. After downloading the MPLAB IDE 8.33 on a new computer, a project may not be able to compile until the right drivers are downloaded accordingly to that computer.
2. Go to Control Panel, then Add Hardware, and select **Next**.
3. Welcome to the Add Hardware will appear, select **Next** again.
4. Choose “No, not this time.” Then select **Next**.



5. Select the first choice given.

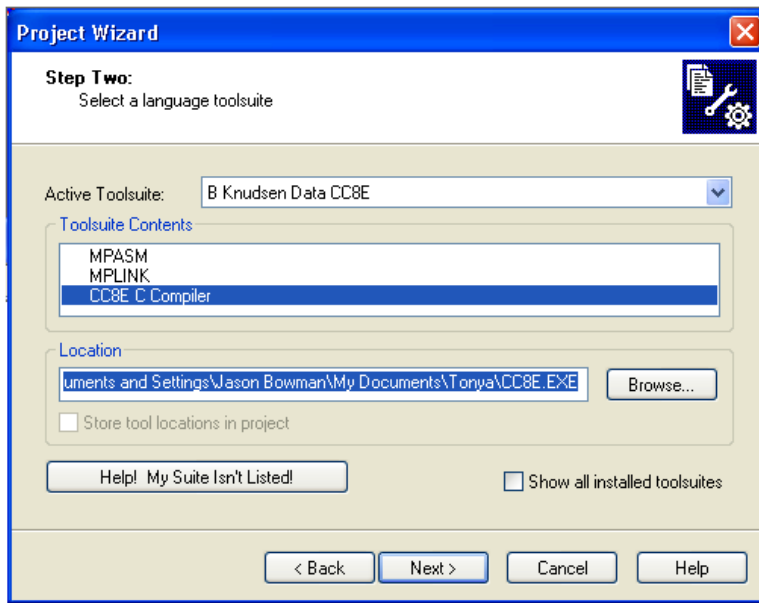


5. The next window may ask to add the correct Microchip MPLAB ICD 2 Firmware Loader, choose the correct one for your computer. Then select **Next**.
6. There should be two folder transporting material, it may take a few minutes, a message will appear, then select **Finish**.



Creating a New Project in MPLAB

1. Double-click on the MPLAB icon on the desktop. When it has finished downloading, go to Project, Project Wizard, and click **Next**.
2. Make sure that the correct PIC is selected depending on the one you are working with and select **Next**.
3. In the next window, select the correct Active Toolsuite (in this case B Knudsen Data CC8E) and make sure that all of the toolsuite contents are made available, meaning none of them have a red "X" next to them:



If any/all of the toolsuite contents are not available, you must select **Browse** to search for their locations. If you are still having trouble finding their locations, the following locations can be used for the MPASM, MPLINK, and CC8E C Compiler:

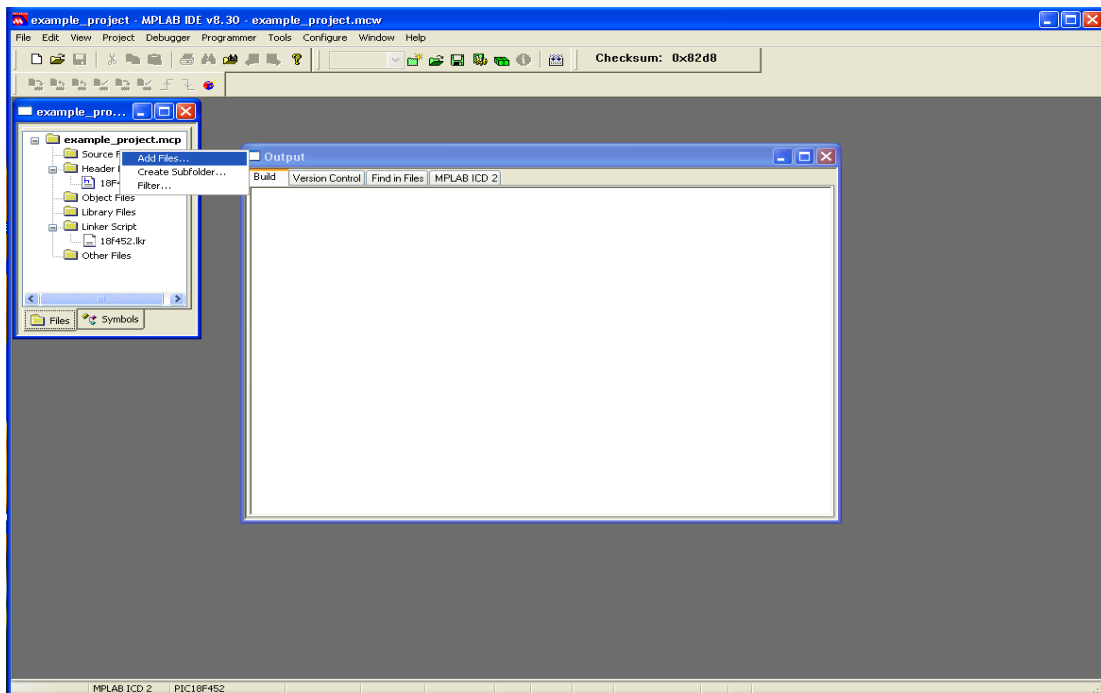
MPASM: MPASMWIN.exe
 MPLINK: mplink.exe
 CC8E C Compiler: CC8E.EXE

*Note: It is helpful to copy these folders into one folder so that each time you start a new project they can be easily retrieved.

4. After clicking **Next**, select **Browse** to choose the folder where your new project will be stored, as well as the name of your new project itself.
5. When prompted to add certain files to the project, make sure to include the header file "18F452.H" and the linker file "18f452.lkr." Select **Next**. A summary window will appear, and after reading it over to check for any errors, select **Finish**.
6. A new workspace has been created, and your new project has been added to it.

Working Within the Workspace

1. If you see a blank grey workspace, go to the View menu and select both **Project** and **Output**. In the **Project** window, you should be able to see both your header and linker files.
2. Select **Programmer >> Select Programmer** and check off the MPLAB ICD2 option.
3. To add your source file (C file) to the project, right click on **Source File**, then add the appropriate file.



Try to store all your related header files, linker files, source files, and toolsuite contents in the same folder so as to avoid confusing the programmer.

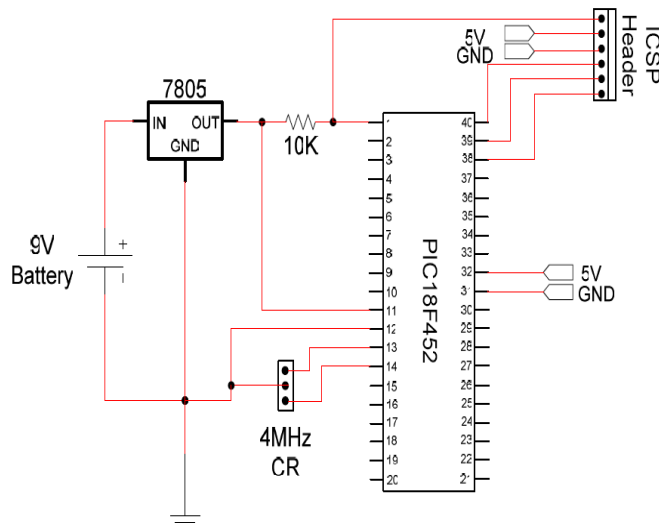
4. Go to the Configure menu and select **Configure Bits**. This is important to do for every single new project that you build, and failing to do so will cause most of the errors users encounter when working on a project. The bits should be configured as follows:

Configuration Bits				
<input checked="" type="checkbox"/> Configuration Bits set in code.				
Address	Value	Category	Setting	
300001	21	Oscillator	XT	
		Osc. Switch Enab	Disabled	
300002	0F	Power Up Timer	Disabled	
		Brown Out Detect	Enabled	
		Brown Out Volta	2.0V	
300003	0E	Watchdog Timer	Disabled-Conti	
		Watchdog Postsc	1:128	
300005	01	CCP2 Mux	RC1	
300006	81	Stack Overflow	Enabled	
		Low Voltage Pro	Disabled	
300008	0F	Code Protect 00	Disabled	
		Code Protect 02	Disabled	
		Code Protect 04	Disabled	
		Code Protect 06	Disabled	
300009	C0	Code Protect Bo	Disabled	
		Data EE Read Pr	Disabled	
30000A	0F	Table Write Pro	Disabled	
		Table Write Pro	Disabled	
		Table Write Pro	Disabled	
		Table Write Pro	Disabled	
30000B	E0	Config. Write P	Disabled	
		Table Write Pro	Disabled	
		Data EE Write P	Disabled	
30000C	0F	Table Read Prot	Disabled	
		Table Read Prot	Disabled	
		Table Read Prot	Disabled	
		Table Read Prot	Disabled	
30000D	40	Table Read Prot	Disabled	

Make sure to check “Configure Bits set in code” so that this configuration remains constant throughout the use of your project.

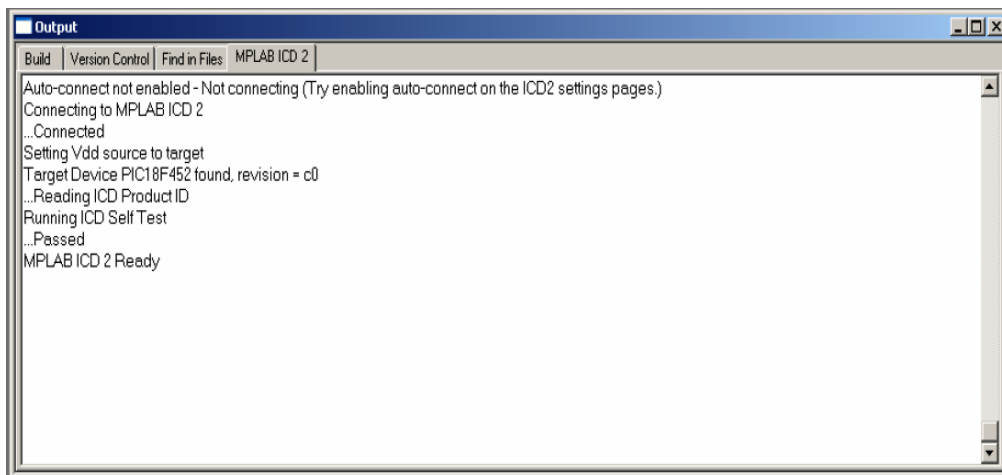
Constructing your Circuit and Connecting the Programmer

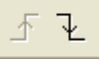

A schematic for the circuit is shown below:




*Note: Before connecting the PIC, plug in the battery and test to make sure the correct voltages are visible at the appropriate locations. A clear sign that the wrong voltage is being supplied to the PIC is the D/A converter will heat up very rapidly. Supplying the wrong voltage to the PIC can permanently damage it, so continually check voltages throughout your use to prevent this from happening.

Assuming you have built the circuit correctly, connect your 9V battery again. You can now connect the programmer to the circuit. While in the workspace check to see that in the Output window the MPLAB ICD 2 has successfully connected:



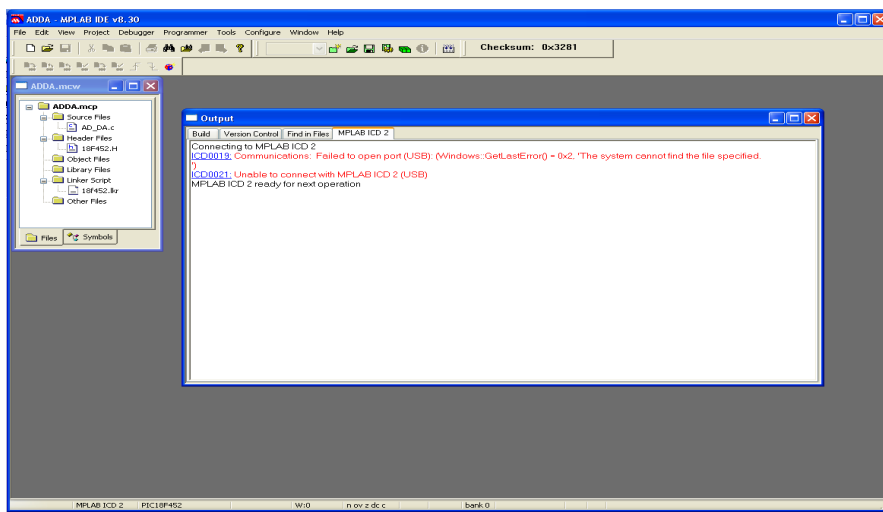
When you are ready to program the PIC microcontroller, connect your 9V battery to the circuit and save your workspace. Then select the down-latch  from the various tabs located at the top of the workspace, and then **Build** . If programming has succeeded, the output window will validate it by saying “BUILD SUCCEEDED.”

To disconnect the programmer using the workspace, select the up-latch  from the various tabs located at the top of the workspace. You can now disconnect the battery, and unplug the MPLAB ICD 2 Programmer from the circuit.

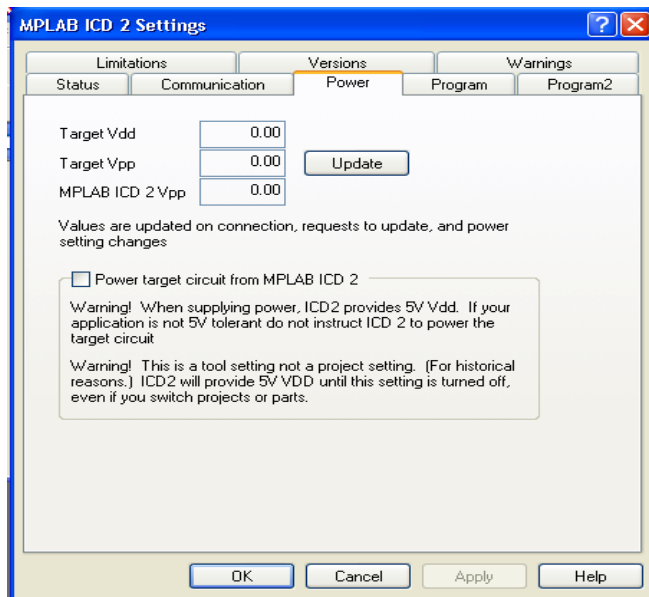
Troubleshooting Programmer

Here are a few tips to help you if you encounter problems in programming your PIC, or even just compiling your source code:

You cannot connect to the MPLAB ICD 2 (USB):

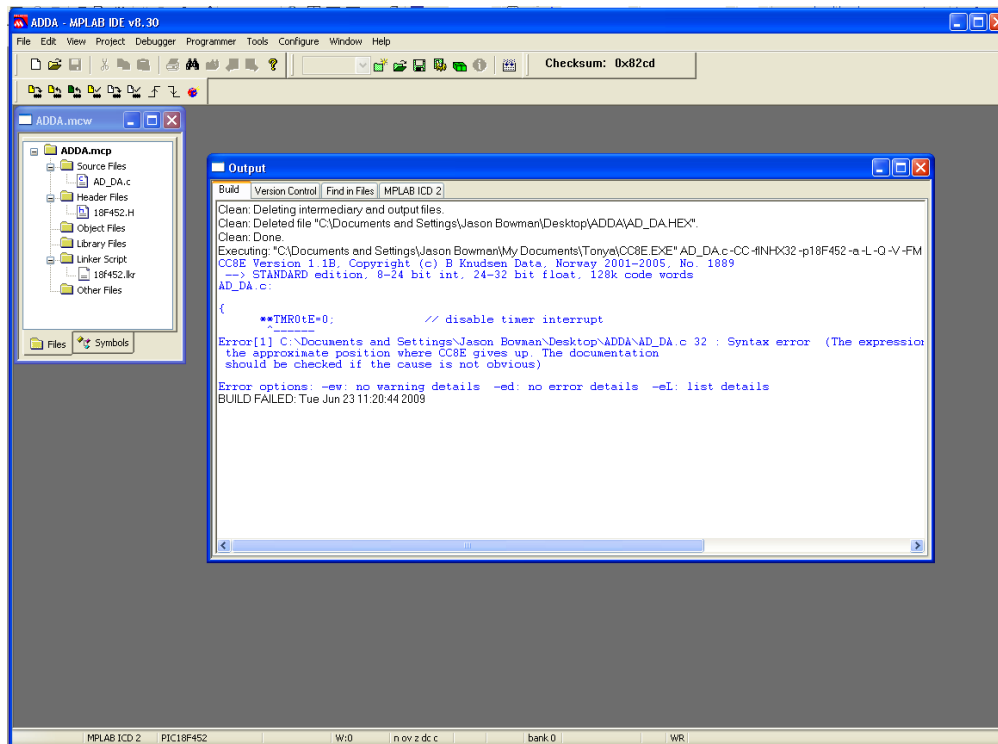


1. Make sure your battery is plugged in correctly, and has at least 8.2V.
2. Check to see that the connector pins are oriented in the correct direction (test this by reversing orientation of the connector pins).
3. Unplug the USB extension from the computer and reinsert, try connecting again.
4. Go to **Programmer >> Settings >> Power**, and make sure that the power target circuit is coming from the battery, not the programmer itself by unselecting the box:



5. In **Programmer >> Settings >> Communication**, make sure that the USB device is selected for the Communication Port.

You are connected to the device, but are unable to successfully compile your source code:



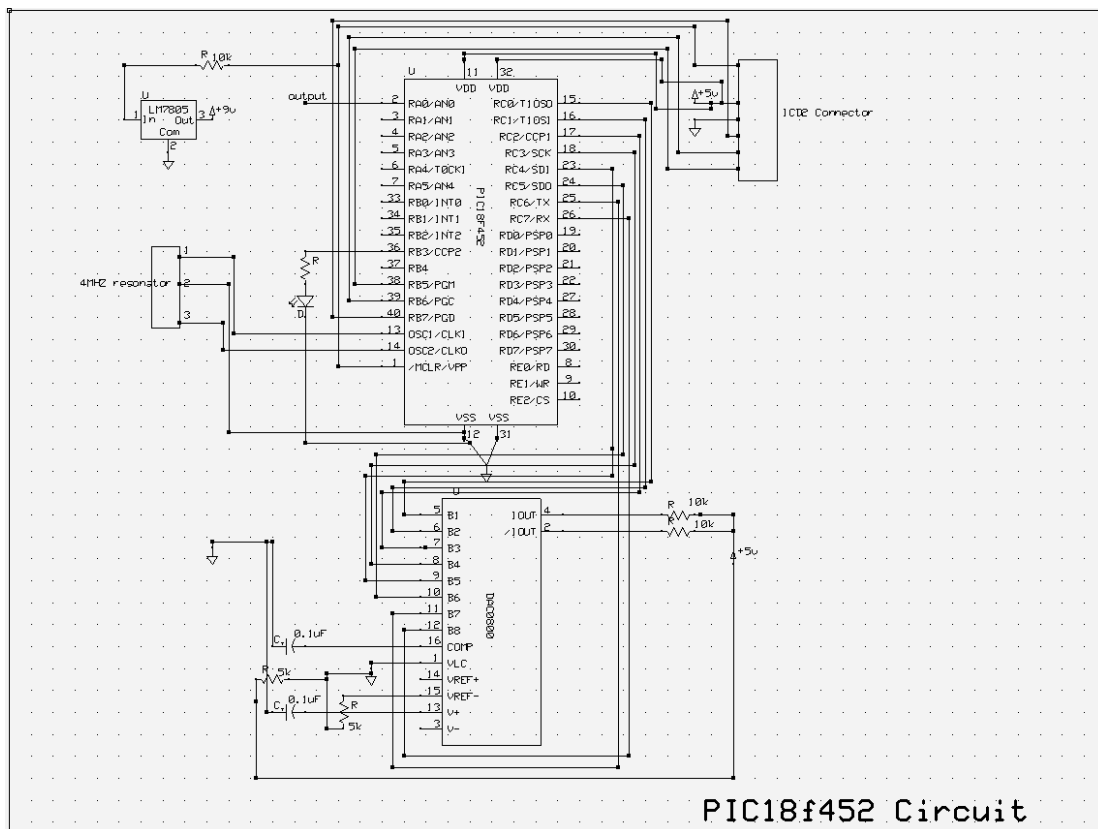
1. In the output window, there will be at least one Error Message (blue text). You can double-click on the word “Error” which will direct you to the exact line where the error occurs. After fixing each error, re-compile and debug until no more errors are shown.
2. If this does not work, save the C-file, remove it from the project window, and add it again to the project. When you double-click on the file, make sure the correct version of your source file

opens up in a new window. If not, an older version of the source file has been trying to compile while you've been making changes to the newer version.

You are connected, and the source code compiles, but programming still fails:

1. Check to make sure your PIC is not reversed on your breadboard, if it is, immediately disconnect the battery to prevent damaging the PIC and/or your breadboard.
2. Try replacing the PIC with a new one. In the case that this works, the PIC you were using is damaged.
3. Review your breadboard for loose connections, short circuits, broken components, and overheating.
4. Try replacing components with new ones in the case that some are broken or are not working properly. This is especially relevant to the 7805 5V regulator because if this component does not work properly, you can potentially damage your PIC and breadboard when it overheats.
5. As a last resort, rewire your breadboard in a new location (i.e. lower right-hand corner) to avoid defects or bad connections.

In order to view the results of your program on an oscilloscope, you must connect a D/A (Digital to Analog) converter chip. Notice that pin 2 of the PIC is the output pin, and can be directly connected to the oscilloscope. The schematic for the circuit is shown below:



Your circuit programmed successfully, but you are not receiving the correct (or any) response from the output pin:

1. Check to make sure your battery is plugged in and has enough voltage (at least 8.2V). If you notice that your battery's voltage is dropping quickly during use, check your breadboard connections to find the source of the battery's drainage.
2. Make sure that the up-latch in the MPLAB workspace is selected, which will disconnect the ICD 2 connector from the circuit and allow the output to be seen.
3. Disconnect and reconnect the cables from the oscilloscope to ensure they are connected properly, and perform a probe calibration which is located on the oscilloscope.
4. You should also make sure that the probe tips are not damaged or oxidized, and replace them if they are.
5. Check and recheck your source code. Even if compilation and programming succeed, your code may not work as expected. Review for infinite loops, faulty switch-statements, and unused variables. Just because your program compiles does not mean it will do what you want it to do.