Contents

[1. Programming a PIC (and how to use I2C) 1](#_Toc39932876)

[2. Available Controllers for the pressure regulators 7](#_Toc39932877)

Last edited 2020-05-09 by Dylan Shah.

# Programming a PIC (and how to use I2C)

General Notes:

* Steps 1-8 describe MPLab and circuit setup. Step 9 describes communication methods.
* This tutorial is written for the capacitor board, but is mostly applicable to all Faboratory circuit boards with a PIC microcontroller

1. Power circuit board with an external power supply of 5.00 volts, at least 10 mA. The Arduino 5V output line is insufficient. (Limit the circuit to 100 mA or less, to avoid frying the board in case of short-circuit. If four sensors are plugged in and one is being programmed by the PICkit3, <0.013 A will be used.) Plug PICkit 3 into PC.

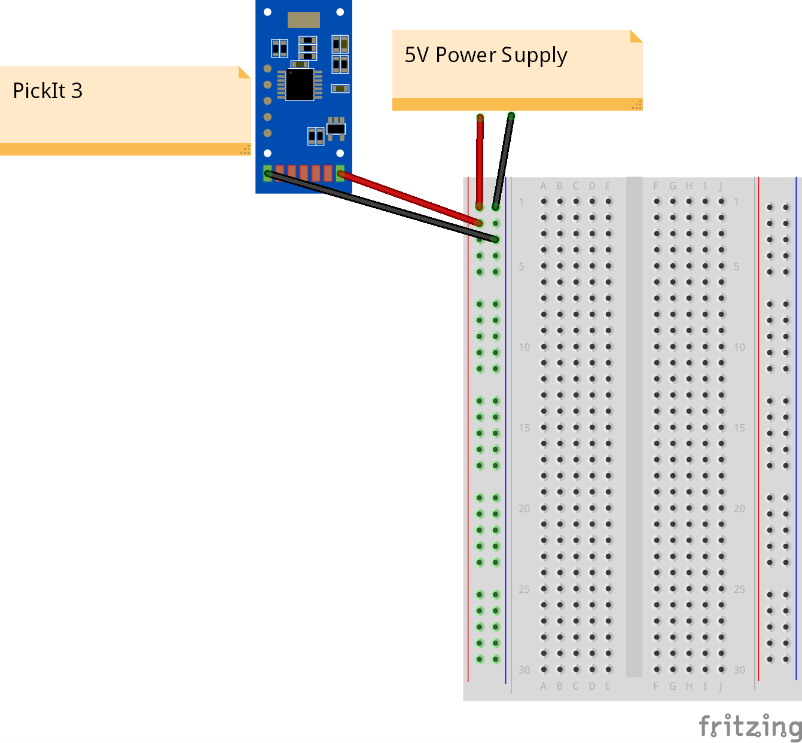


Figure 1. A suggested circuit. The blue PCB is the capacitor board or a pressure regulator. The white board is a breadboard.

1. Open Pic project <pic\_i2c\_project > in MPLab X IDE
2. Double-click the relevant “C” (.c) file to open it. In line 77, edit i2c\_address to your desired address. In the photo below, the Arduino will communicate with address 99. It is stored in the PIC as address 198, since the Arduino automatically “shifts left” the address bits, i.e. it multiplies by 1 binary bit = value of 2.



Figure 2. Edit the I2C\_ADDRESS variable.

1. Make sure there aren’t any hard-coded filepaths in the files, by right-clicking the project and doing “Clean and Build”

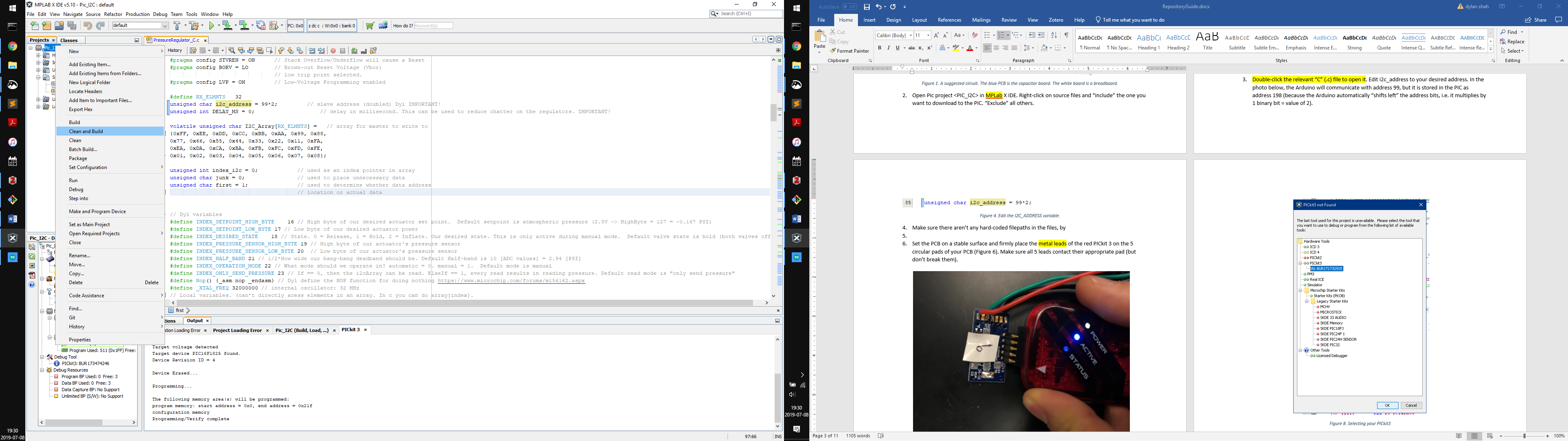


Figure . Prepare the project with “Clean and Build”

1. Set the PCB on a stable surface and firmly place the metal leads of the red PICkit 3 on the 5 circular pads of your PCB (Figure 4). Make sure all 5 leads contact their appropriate pad (but don’t break them).



Figure 4. Connecting the PICit3 to the PCB.

1. Keeping the pads pressed, download to device using "Make and Debug Program Device".

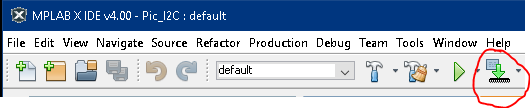


Figure 5. Make and Debug Program Device

* 1. If you didn’t already plug in the PICit3, a window will pop up. Plug in your PICit3, select it in the menu, and click “OK”.

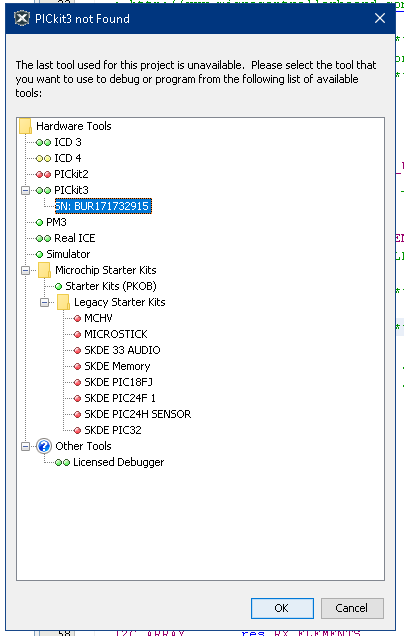


Figure 6. Selecting your PICkit3

* 1. The console (probably at the bottom of your screen) will give you a message like “BUILD FAILED” or “BUILD SUCCESSFUL”. If the build failed , right-click your Project, select “clean and build”, and repeat this step.
     1. Other troubleshooting steps: 1. make sure you excluded all unnecessary files. 2. Make sure your board is powered by an external power supply, not the Arduino. 3. “Advanced” users: debug the firmware code.

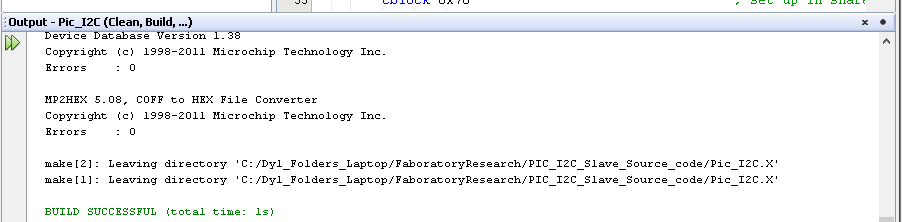


Figure 7. A “build successful” message.

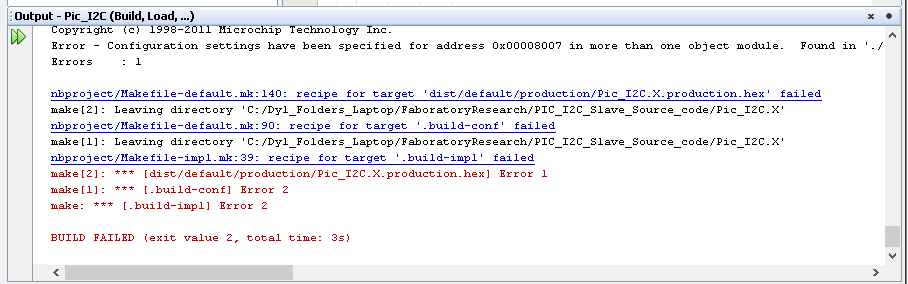


Figure 8. A "build failed” message.

1. Once it shows “build successful”, the console will clear and begin showing things like “Connecting to MPLAB PICkit 3”.

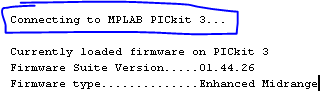


Figure . Connecting to PICkit 3

1. Some errors might show up.
   1. Basically , return to Step 4. Read the rest of bullet-point 8’s sub-bullets for additional tips.
   2. Shows red “Target device was not found…” Return to Step 4 (i.e, connect PIC and PICkit3, Make and Download…). Make sure to have good contact between each PICkit3 metal lead and the PCB’s corresponding circular contact.

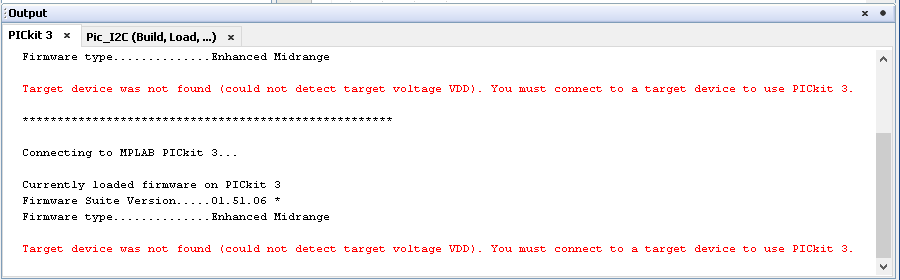


Figure . Target device not found.

* 1. Popup similar to “Target Device ID (0x0) is an Invalid Device ID…”. DO NOT continue programming. Click Cancel, and return to step 4. Continuing can sometimes make the PIC un-programmable, but “when and why” is not well docmumented.

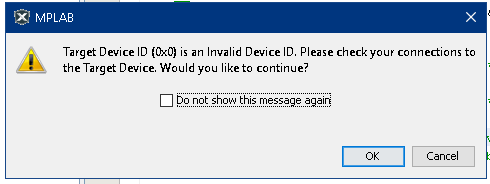


Figure . Target Device ID Invalid

1. Programming is complete when console displays “verify/program complete”.

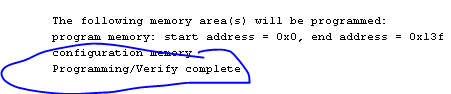


Figure . Program/Verify complete

1. Optional: lift leads off of pads; power-cycle circuit. (I.e. turn the power supply off then on.) Sometimes the PIC will be put in a harmless but nonresponsive state during programming.
2. Connect the appropriate pins to an I2C circuit (“bus”). Arduino: A4 (SDA)=blue, A5(SCL) = green. Optionally, adjust so the PIC is powered by the Arduino. An example circuit is shown below. Typically I use 3.3KOhm [pull-up resistors](https://electronics.stackexchange.com/questions/1849/is-there-a-correct-resistance-value-for-i2c-pull-up-resistors).

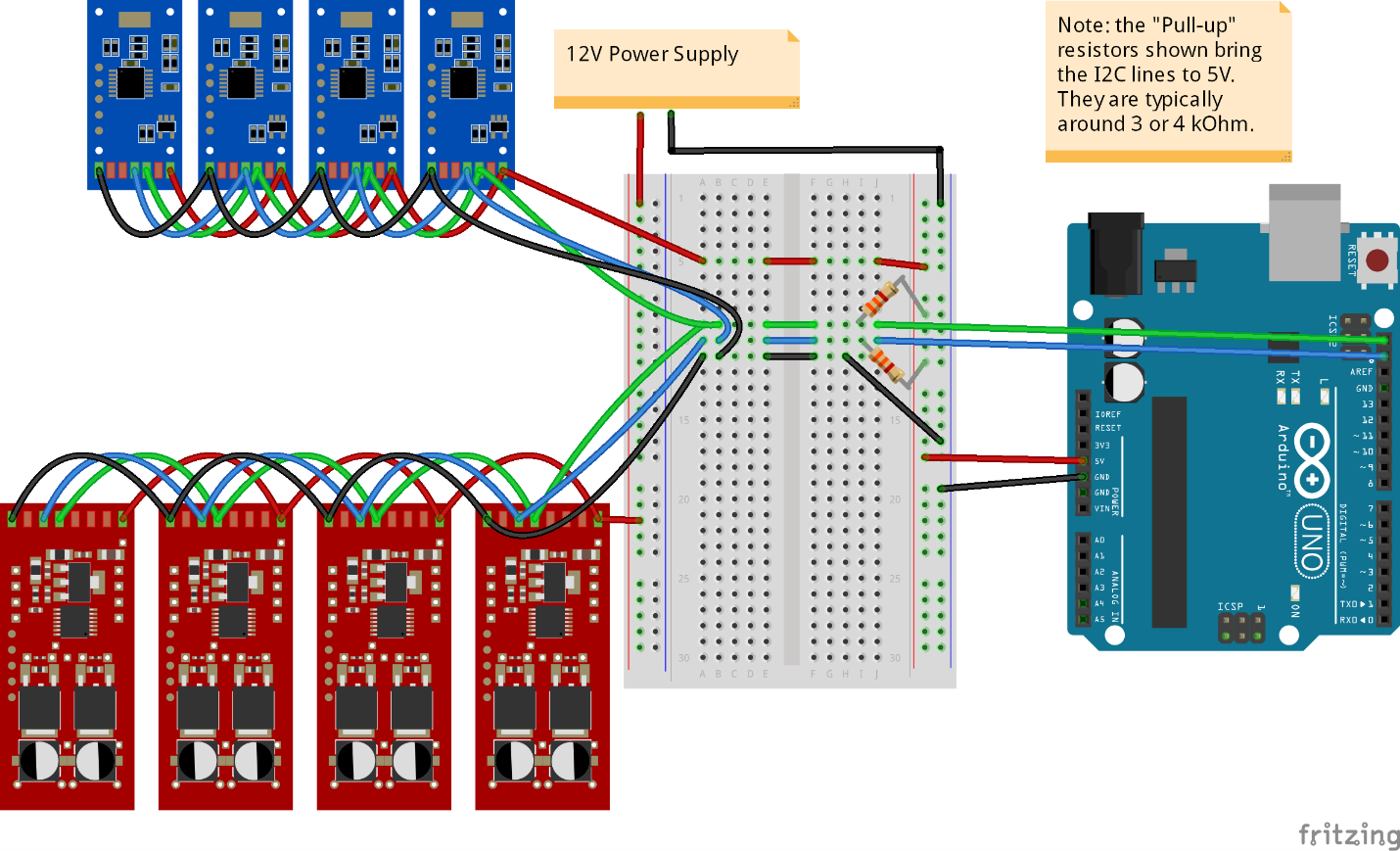


Figure 13. An example circuit.

1. Ready to communicate! Check the PIC firmware or repository README for an explanation of available interfaces/commands. Options:
   1. Arduino 🡪 PIC. Use the Arduino driver for your PIC firmware (in the Faboratory GitHub repository “Sensory-Active-Skin-Controller”) , or the sample Arduino Wire library code.
   2. PC 🡪 Arduino 🡪 PIC. Use your favorite Python, MATLAB, or terminal to interface with an arduino over Serial, which will communicate with the PIC. Talk to labmates.

# Available Controllers for the pressure regulators

The pressure regulators can be controlled manually or “automatically” (Figure 15). Manually allows you to set the regulator to “Inflate”, “hold”, or “release”, while automatic mode implements a three-state bang-bang controller (Figure 14). We only switch from one state (green ellipsoid) to another if the condition along the transition arrow is met. Basically, the system tries to bound the error to be lower in magnitude than half of a “band width”. Abbreviations:

S = sensor data. SP = set point. BW = band width. UB = upper bound = SP + BW/2. LB = lower bound = SP – BW/2

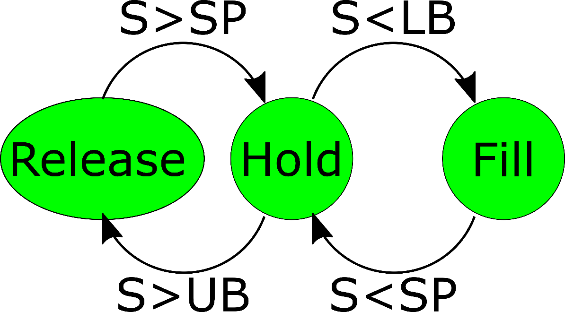


Figure . Automatic mode state diagram

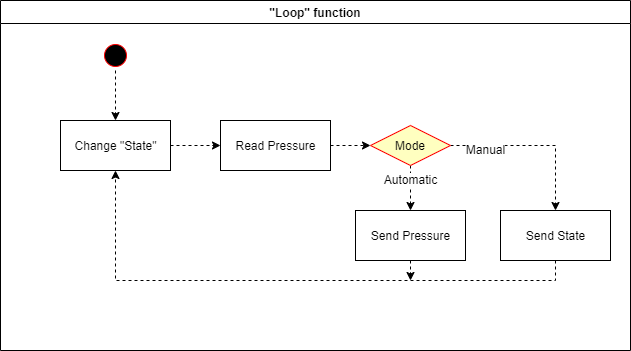


Figure . Control modes flowchart