**Arduino Firmata**

***A Quick Guide*** ***To Using Firmata*** ***On*** ***An Arduino***

Note: This is copy-pasted from a document that lives on Assembla. I don’t really want to maintain two different formatting structures, so if you suspect this document is out of date and want an up-to-date document, check the link [here](https://app.assembla.com/spaces/airmarus/subversion-2/source/HEAD/TestEng/Python3/trunk/Test_Equipment/Arduino_Firmata.md).

While Arduino compatible hardware is underpowered at best for taking analog measurements or generating high-fidelity signals, it still has a valuable place as a familiar hardware set for students and engineers, and logic level IO is always incredibly useful for toggling relays, or driving FETs/h-bridges/etc.

[Firmata](https://www.arduino.cc/reference/en/libraries/firmata/) is an intermediate protocol that connects an embedded system to a host computer. Effectively, this allows you to interface directly with any IO on the microcontroller through a communication medium of your choosing (USB, Wifi, ethernet, Bluetooth). Using Firmata, we can quickly create USB controlled IO devices out of cheap, commonly available, off-the-shelf Arduino boards, allowing us to create reliable hardware control interfaces for test fixturing.

**Getting Started**

To get started, you'll need a few things:

* Arduino: You'll need an Arduino board; any official board or clone will do.
* Arduino IDE: You'll need the [Arduino IDE](https://www.arduino.cc/en/software).

You'll also need to figure out how you want to talk to your device. Pick one of the following methods to get started:

* [Windows Remote Arduino Experience](https://www.microsoft.com/en-us/p/windows-remote-arduino-experience/9nblggh2041m?activetab=pivot:overviewtab): This is a Windows Store app that has a nice GUI and gives you an overview of what to expect to be able to do with Firmata. If you just want to test capabilities and play around, I highly recommend this route. Click the link to install and follow the prompts.
* [pyFirmata](https://github.com/tino/pyFirmata): pyFirmata is a Python library for interfacing with any Firmata board. The examples and syntax are incredibly simple, and for basic IO operations (like toggling relay IO on Silver Box) you can't ask for much more. To install, use pip and run the following command: pip install pyfirmata.

**Installing Firmata**

To install Firmata on your Arduino, do the following:

* Start the Arduino IDE.
* Plug in your Arduino compatible board.
* In the Arduino IDE, go to Tools, Port, and select the com port your device is connected to.
* Go to Tools, Board: and select the device you have connected.
* Go to File, Examples, Firmata and select StandardFirmata
* Click the Upload button (it looks like a rightward pointing arrow).
* Wait for installation to complete.

At this point, your Arduino board should have Firmata installed.

**Windows Remote Arduino Experience**

To use Windows Remote Arduino Experience to control your Firmata Arduino, do the following:

* Open the Windows Start menu.
* Navigate to Windows Remote Arduino Experience and launch the application.
* Select the communication medium to talk to your device with the Connection dropdown.
* Your device should show up under Devices discovered.
* Select your device and press the Connect button.
* Use the Digital, Analog, and PWM tabs at the top to control all the different IO functionality of each pin.

**pyFirmata**

To use pyFirmata to control your Firmata Arduino, do the following:

* Open your Python editor of choice.
* Copy and paste the following code into your editor (remove the backtick marks if you're viewing this as a .txt doc):

#!/usr/bin/env python3

import pyfirmata

import time

if \_\_name\_\_ == '\_\_main\_\_':

board = pyfirmata.Arduino('YOUR\_PORT\_HERE')

print("Communication Successfully started")

while True:

board.digital[13].write(1)

time.sleep(1)

board.digital[13].write(0)

time.sleep(1)

* Open your Device Manager or Arduino IDE.
* Find what com port your Arduino is registered as.
* Replace 'YOUR PORT HERE' in the example code with your Arduino's com port (on Windows this should look like “COM1”).
* Running the Python script will toggle pin 13 on your Arduino (usually the onboard LED) high for 1 second, low for 1 second. And repeat.