FirmataPlus-setup-2019JAN31

March 20, 2019

1 Connecting desktop Python to the Arduino Uno running Firmata-Plus

Last updated: 2019-Jan-31

1.1 Objective

The goal of this setup procedure is to get Python programs within JupyterLab to be able to remote control your Arduino Uno board stack over the USB connection. Both sides of this setup (the PC, and the Arduino) will be using the FirmataPlus protocol to interact with each other. To do so, both sides must be configured with the FirmataPlus software. First, a program needs to be loaded onto your Arduino so that it understands what the PC (running Python) is telling it what to do. Second, because we are using Anaconda/JupyterLab on the PC, some configuration updates and additions are needed to enable this communication.

1.2 Step 1: Download config, library, test installation files

Using the "FirmataPlus installation data files" link on the me30.org website, download the ZIP file. Extract the contents of this ZIP file into whatever directory in which you like to keep your ME30 homework and/or other ME30 labs.

When unpacked, the name of the directory that it will create and populate will be called `ME30_FirmatatPlus_files`. Remember where you put these unpacked files because you'll need to refer to them in a few steps. From now on, we'll call this your *ME30_FirmataPlus_files* directory.

1.3 Step 2a: Download and install Arduino IDE software

You need to install the Arduino IDE software both to get the Arduino USB driver installed, as well as to program your Arduino with the FirmataPlus software.

Download the software installer from https://www.arduino.cc/en/Main/Software in the section entitled "Download the Arduino IDE". For Windows, click on the "Windows Installer" option, whereas MacOS users should select the (only) link for MacOS. You will be sent to the donation page, where you next click on "Just Download".

Run the software installer executable, accepting the default settings when asked. The process for MacOS should be similar to Windows'.

1.4 Step 2b: Install FirmataPlus Arduino files

The Arduino IDE defines a standard location for users' Arduino code and support libraries, which is located in the *Arduino* subdirectory of your home directory. Within this directory is the directory *libraries*, in which code is placed for building more-than-basic programs.

Unzip the file *ME30-FirmataPlus-libraries.zip* from your *ME30_FirmataPlus_files* directory and choose your home directory's *Arduino* directory as the destination for the unzipped files. If you do this correctly, a *libraries* directory should have been created in your *Arduino* directory, with four (or so) of its own library subdirectories.

1.5 Step 3: Anaconda configuration

1.5.1 Opening the correct type of command window

You should already have installed the Anaconda distribution of JupyterLab running on your own laptop, and it must be known-working before proceeding. The instructions for Anaconda installation was part of Lab 1. To proceed you must also have a connection to the Internet because software will be downloaded.

On MacOS, if you haven't started the Anaconda Launcher already, do so. Then, from within the Anaconda Launcher, select `New->Console`.

On Windows, you *must not* be currently running a copy of Anaconda or JupyterLab. If you are, the following update steps will not be successful. Be warned that sometimes copies of JupyterLab that have become detached from Anaconda are still running in the background on your machine, which is the real problem. Unless you're confident that you can kill all of the pices of JupyterLab running in the background, it's safest to just reboot your machine and run the following steps after the restart.

First, run Anaconda Prompt ***as Administrator***, as shown below. You do this by pressing the Windows key (or icon) and *right*-clicking on *Anaconda Prompt* under the Anaconda program group. From there select either *Run as Administrator* or *More->Run as Administrator* to start it. If you don't run it in admin mode, you will probably get errors during later steps, and you may have to come back and do everything all over again.

During each of the following steps, it is *imperative* that you look at the output of each command and ensure that it doesn't tell you that anything "failed" or had errors. You should be able to safely ignore messages that say "DEBUG" in them, so don't worry about them. But ALWAYS keep your eyes out for failures/errors before going on to the next step.

1.5.2 conda update -n root conda

In the command window, start by entering `conda update -n root conda`. You will be asked to proceed after being shown what it is planning to update. Answer `y`. This may take a few minutes to complete.

1.5.3 conda update --all

Enter `conda update --all`. Again, answer `y` when it asks you if you should proceed.

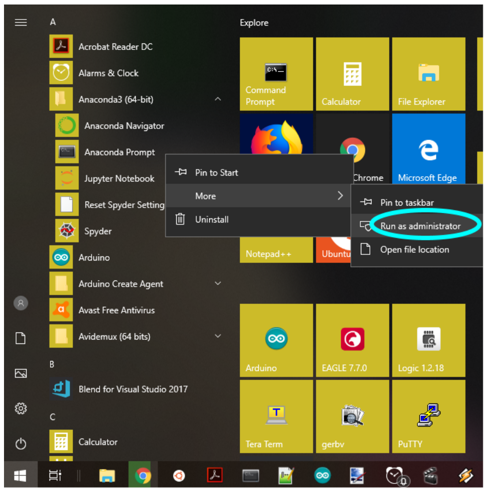
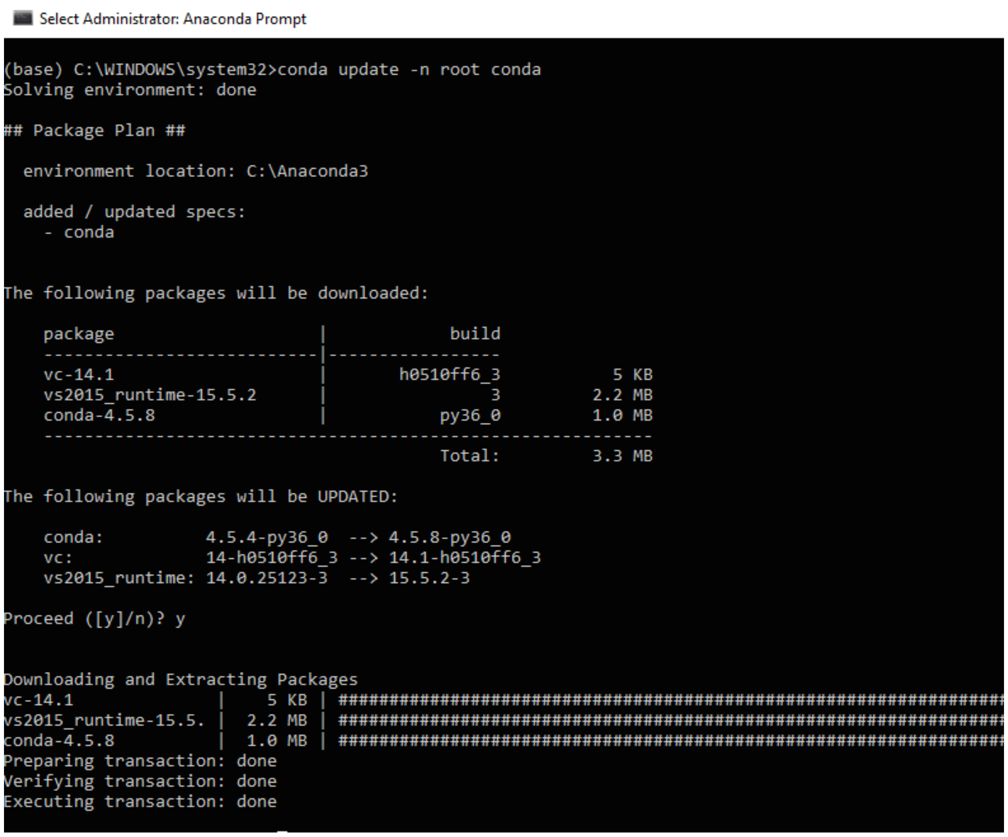


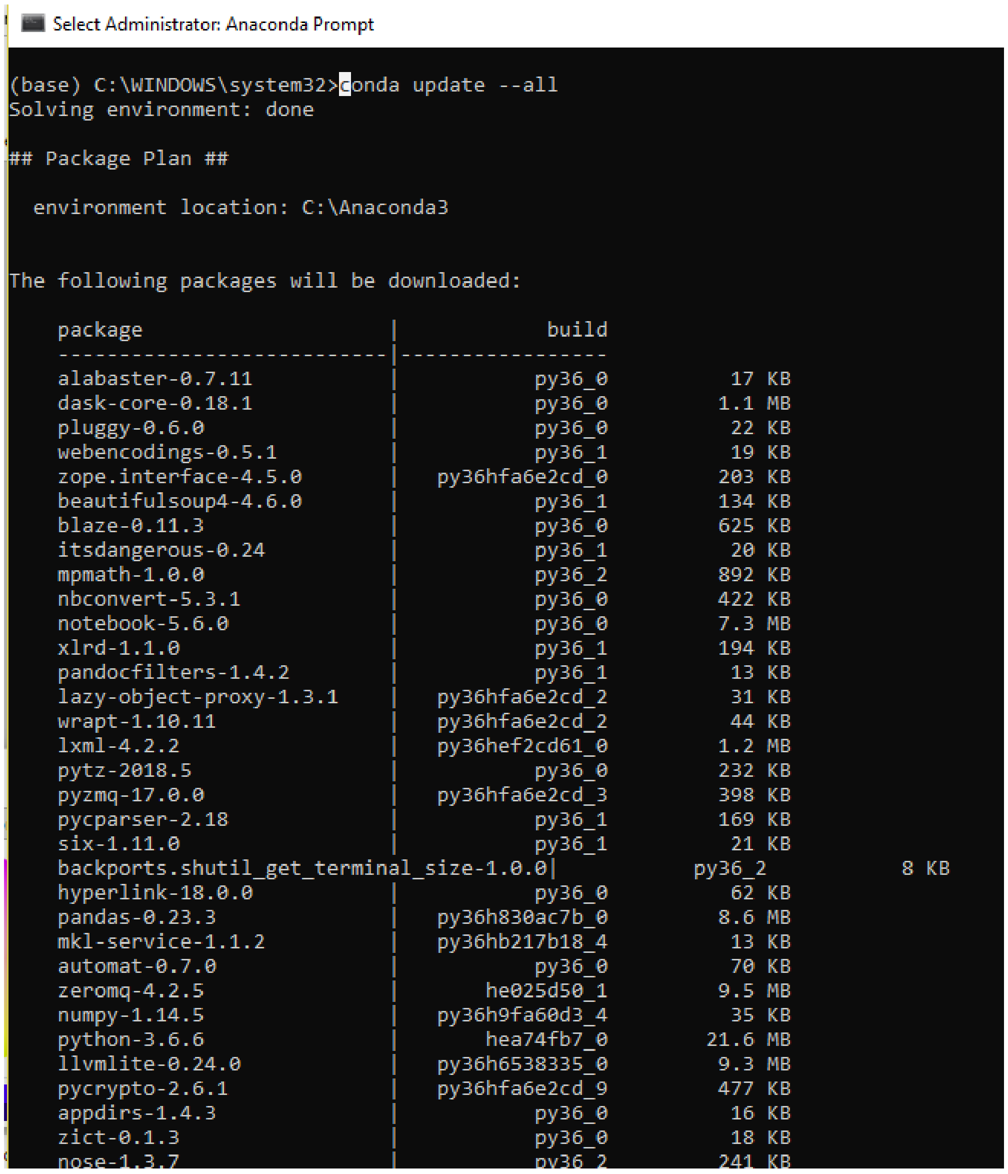
Figure 7: Running Anaconda Prompt as Administrator.



Select Administrator: Anaconda Prompt

```
(base) C:\WINDOWS\system32>conda update -n root conda
Solving environment: done

## Package Plan ##

  environment location: C:\Anaconda3

  added / updated specs:
    - conda


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    vc-14.1                    |       h0510ff6_3           5 KB
    vs2015_runtime-15.5.2      |                3         2.2 MB
    conda-4.5.8                |           py36_0         1.0 MB
    ------------------------------------------------------------
                                           Total:         3.3 MB

The following packages will be UPDATED:

    conda:          4.5.4-py36_0  --> 4.5.8-py36_0
    vc:             14-h0510ff6_3 --> 14.1-h0510ff6_3
    vs2015_runtime: 14.0.25123-3  --> 15.5.2-3

Proceed ([y]/n)? y


Downloading and Extracting Packages
vc-14.1              |    5 KB | ##################################################################
vs2015_runtime-15.5. |  2.2 MB | ##################################################################
conda-4.5.8          |  1.0 MB | ##################################################################
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

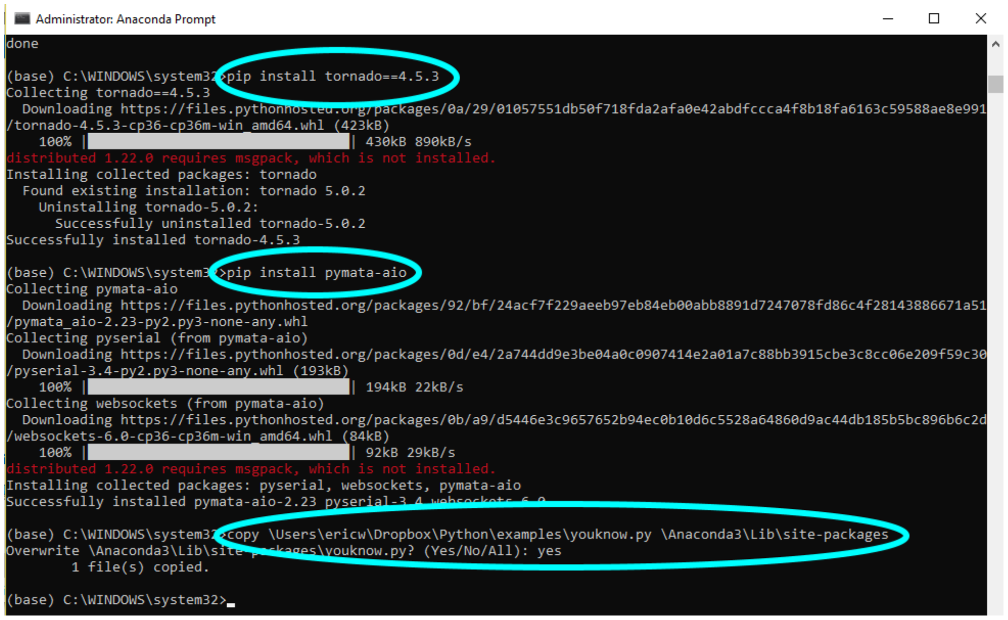
Figure 8: Updating conda.



Select Administrator: Anaconda Prompt

```
(base) C:\WINDOWS\system32>conda update --all
Solving environment: done

## Package Plan ##

  environment location: C:\Anaconda3


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    alabaster-0.7.11           |           py36_0          17 KB
    dask-core-0.18.1           |           py36_0         1.1 MB
    pluggy-0.6.0               |           py36_0          22 KB
    webencodings-0.5.1         |           py36_1          19 KB
    zope.interface-4.5.0       |   py36hfa6e2cd_0         203 KB
    beautifulsoup4-4.6.0       |           py36_1         134 KB
    blaze-0.11.3               |           py36_0         625 KB
    itsdangerous-0.24          |           py36_1          20 KB
    mpmath-1.0.0               |           py36_2         892 KB
    nbconvert-5.3.1            |           py36_0         422 KB
    notebook-5.6.0             |           py36_0         7.3 MB
    xlrd-1.1.0                 |           py36_1         194 KB
    pandocfilters-1.4.2        |           py36_1          13 KB
    lazy-object-proxy-1.3.1    |   py36hfa6e2cd_2          31 KB
    wrapt-1.10.11              |   py36hfa6e2cd_2          44 KB
    lxml-4.2.2                 |   py36hef2cd61_0         1.2 MB
    pytz-2018.5                |           py36_0         232 KB
    pyzmq-17.0.0               |   py36hfa6e2cd_3         398 KB
    pycparser-2.18             |           py36_1         169 KB
    six-1.11.0                 |           py36_1          21 KB
    backports.shutil_get_terminal_size-1.0.0|           py36_2           8 KB
    hyperlink-18.0.0           |           py36_0          62 KB
    pandas-0.23.3              |   py36h830ac7b_0         8.6 MB
    mkl-service-1.1.2          |   py36hb217b18_4          13 KB
    automat-0.7.0              |           py36_0          70 KB
    zeromq-4.2.5               |       he025d50_1         9.5 MB
    numpy-1.14.5               |   py36h9fa60d3_4          35 KB
    python-3.6.6               |       hea74fb7_0        21.6 MB
    llvmlite-0.24.0            |   py36h6538335_0         9.3 MB
    pycrypto-2.6.1             |   py36hfa6e2cd_9         477 KB
    appdirs-1.4.3              |           py36_0          16 KB
    zict-0.1.3                 |           py36_0          18 KB
    nose-1.3.7                 |           py36_2         241 KB
```

Figure 9: Updating Anaconda with conda.



```
done

(base) C:\WINDOWS\system32>pip install tornado==4.5.3
Collecting tornado==4.5.3
  Downloading https://files.pythonhosted.org/packages/0a/29/01057551db50f718fda2afa0e42abdfccca4f8b18fa6163c59588ae8e991
/tornado-4.5.3-cp36-cp36m-win_amd64.whl (423kB)
    100% |████████████████████████████████| 430kB 890kB/s
distributed 1.22.0 requires msgpack, which is not installed.
Installing collected packages: tornado
  Found existing installation: tornado 5.0.2
    Uninstalling tornado-5.0.2:
      Successfully uninstalled tornado-5.0.2
Successfully installed tornado-4.5.3

(base) C:\WINDOWS\system32>pip install pymata-aio
Collecting pymata-aio
  Downloading https://files.pythonhosted.org/packages/92/bf/24acf7f229aeeb97eb84eb00abb8891d7247078fd86c4f28143886671a51
/pymata_aio-2.23-py2.py3-none-any.whl
Collecting pyserial (from pymata-aio)
  Downloading https://files.pythonhosted.org/packages/0d/e4/2a744dd9e3be04a0c0907414e2a01a7c88bb3915cbe3c8cc06e209f59c30
/pyserial-3.4-py2.py3-none-any.whl (193kB)
    100% |████████████████████████████████| 194kB 22kB/s
Collecting websockets (from pymata-aio)
  Downloading https://files.pythonhosted.org/packages/0b/a9/d5446e3c9657652b94ec0b10d6c5528a64860d9ac44db185b5bc896b6c2d
/websockets-6.0-cp36-cp36m-win_amd64.whl (84kB)
    100% |████████████████████████████████| 92kB 29kB/s
distributed 1.22.0 requires msgpack, which is not installed.
Installing collected packages: pyserial, websockets, pymata-aio
Successfully installed pymata-aio-2.23 pyserial-3.4 websockets-6.0

(base) C:\WINDOWS\system32>copy \Users\ericw\Dropbox\Python\examples\youknow.py \Anaconda3\Lib\site-packages
Overwrite \Anaconda3\Lib\site-packages\youknow.py? (Yes/No/All): yes
        1 file(s) copied.

(base) C:\WINDOWS\system32>_
```

Figure 10: Tornado version regression, pymata-aio installation and youknow.py copying.

1.5.4 pip install tornado==4.5.3

Enter `pip install tornado==4.5.3`. After you run this command you may get a message about your version of "pip" being out-of-date. If so, enter the command to update your pip, then re-enter the tornado install command.

1.5.5 (MacOS only) conda install pyserial

On MacOS only, you need to `conda install pyserial`. Windows users don't have to do anything special here because this is taken care of within the next command.

1.5.6 pymata-aio

Next, enter `pip install pymata-aio`.

If you are on MacOS and this command does *not* work for you, then you need to copy the `pymata_aio` directory from the *FirmataPlus_install_files* directory into your `lib/python3.<n>` directory that's under your `Anaconda3` installation directory. This directory name may or may not be capitalized and may or may not be located in your home directory (for example `/Users/<your-name>/anaconda3`) depending on how exactly you installed Anaconda.

1.5.7 youknow.py

In *FirmataPlus_install_files* there is a file named `youknow.py` which contains helper classes/functions that you'll need for developing your programs. We are going to put this Python source file in a central location where Anaconda will always find it when you refer to it in your programs. If you don't do this, the file will have to be present in every directory in which you write programs for the rest of the semester, and you *really* want to avoid having to do that if you can.

To do this, copy `youknow.py` into your Anaconda installation's global packages directory. On Windows, this directory appears to be in `lib\site-packages`, and on MacOS it seems to be `Anaconda3/lib/python3.<n>`. Your Anaconda install directory may be in either your home directory (in my case `C:\Users\eric\Anaconda3`) or `C:\Anaconda3` on Windows, or on MacOS in your home directory (like `/Users/<your name>/anaconda3`) or unfortunately perhaps elsewhere.

On Windows, depending on how you originally installed Anaconda (for "All Users" or "Just Me"), the `site-packages` directory is usually in one of the following locations: - `C:\Users\<your-username>\Anaconda3\Lib\site-packages` - `C:\Anaconda3\Lib\site-packages`

1.6 Step 4: Testing the PC-to-Arduino communication

If you're on MacOS, you've already launched JupyterLab but if you're on Windows then launch JupyterLab from the Anaconda launcher if you don't already have it running.

In *ME30_Lab7_files* directory there is a test program named "blink4.py". Create a new JupyterLab notebook and create a code cell into which you copy-and-paste the contents of this file. This program blinks all four red LEDs on the YouKnow board, as long as the YouKnow board's slide switch is in the "RED" position (otherwise it'll blink the RGB LED white)

At this point you should plug in your board(s) into a USB port and at least one LED on the *bottom* board should be lit, showing you that it's getting power from your laptop.

You should now be able to Shift-Return to execute this code in your and after a few seconds, all four LEDs should start to blink. You will (OK, should!) also see some messages in an Output cell in your notebook with the following contents:

```
pymata_aio Version 2.25 Copyright (c) 2015-2018 Alan Yorinks All rights reserved.

Using COM Port:com3

Initializing Arduino - Please wait...
Arduino Firmware ID: 2.5 FirmataPlusMrYsLab.ino
Auto-discovery complete. Found 20 Digital Pins and 6 Analog Pins
```

If you receive any error message(s), ensure that your Arduino is plugged into your laptop and is getting power. In Anaconda, select `Kernel->Restart Kernel and Clear All Outputs`. This clears the previous output window and resets the connection to the board, after which you will hopefully be able to execute the code cell successfully.

Unfortunately, when running any new/modified/restarted program using this software within Anaconda, you always have to do a kernel restart before executing a code cell. Using the "Clear All Outputs" option is nice because it erases the error messages from the last run, and this makes it clear these errors aren't from the most recent run of the Python program. Whenever in doubt, reset your kernel and re-run.

If your board is blinking, you're ready to continue on to the programming portion of this lab. If not, you'll need help from you TA to try to figure out what's not working for you, based on whatever errors show up in the Output cell.