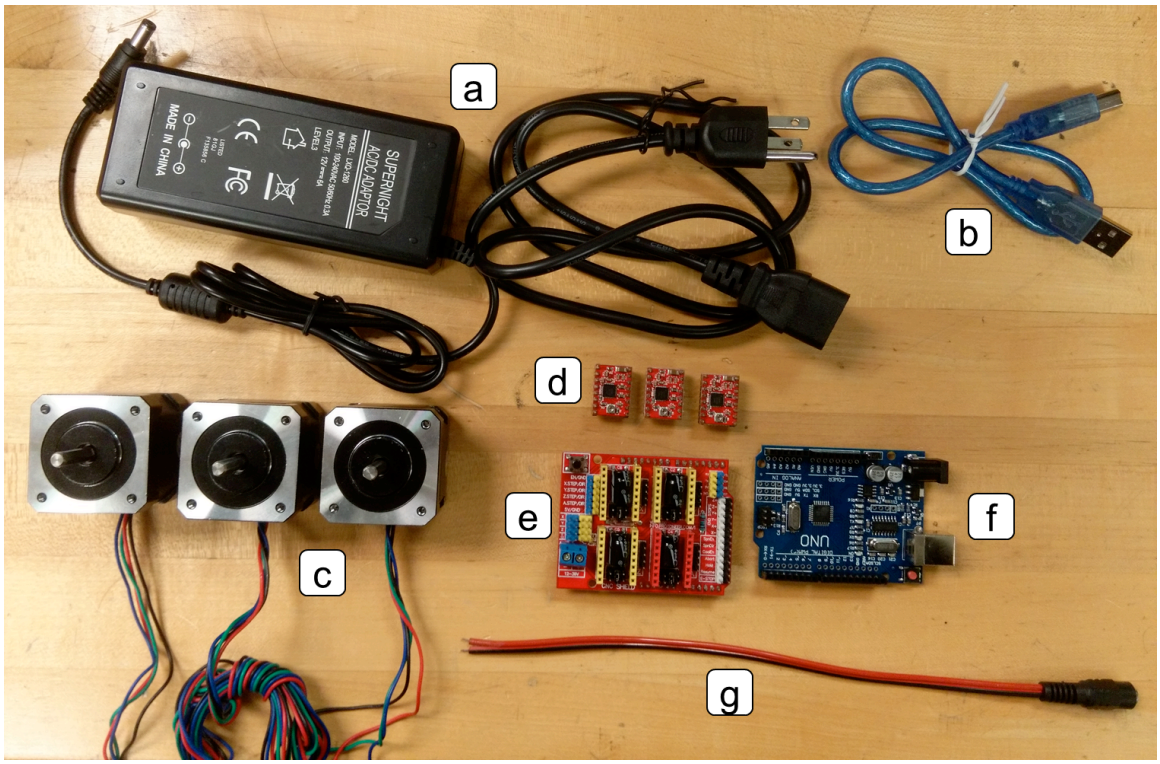
2.70 Hardware Set-up Instructions

Preparing Your Hardware

1. Check to make sure your hardware kit has all its parts



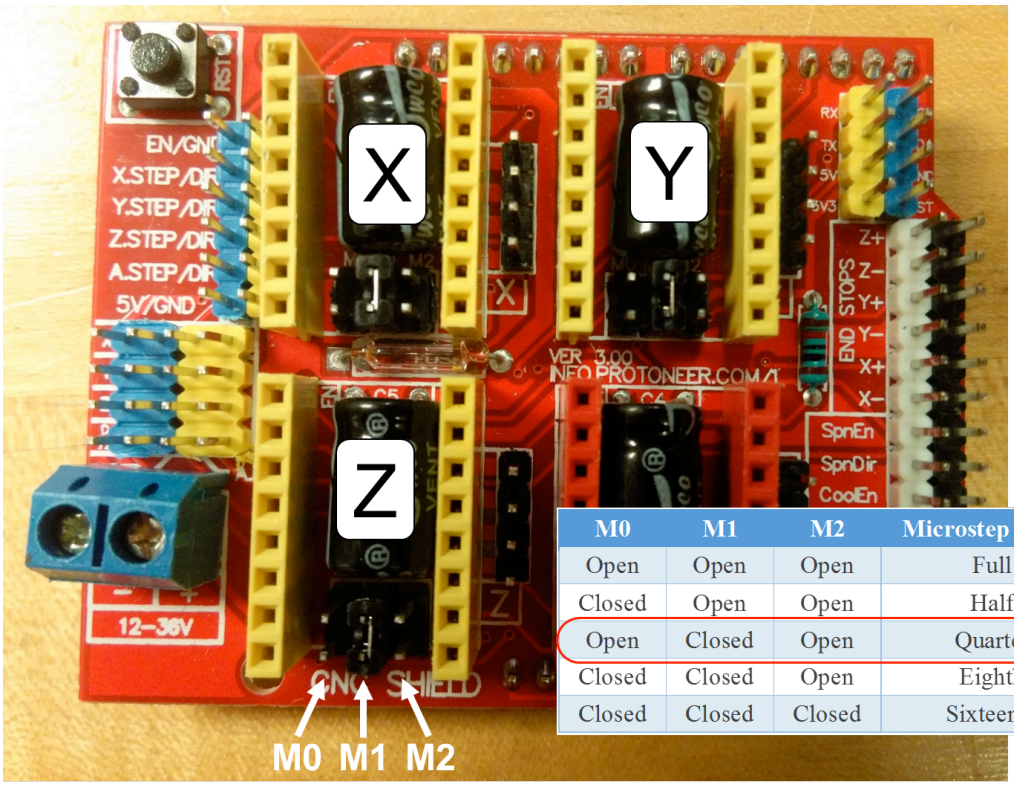
   a. 12V/6A power supply
   b. Blue USB A-B cable
   c. (3x) NEMA 17 stepper motors
   d. (3x) Stepper Driver boards
   e. Protoneer CNC Shield
   f. Arduino UNO
   g. Barrel jack to bare leads cable

2. Unplug the CNC Shield from the Arduino board
   a. Be careful to avoid bending the pins (slowly rocking the board back and forth as you gradually remove it is a good way to avoid pin bending)
3. Unplug the three Stepper Driver boards from the CNC Shield
   a. Be careful to avoid bending the pins (slowly rocking the board back and forth as you gradually remove it is a good way to avoid pin bending)

4. Adjust the microstepping jumpers

    a. The Stepper Driver boards we are using use an Allegro A4988 IC, which allows for full, 1/2, 1/4, 1/8, and 1/16 microstepping modes. Microstepping allows smoother and finer movement, but at the cost of a reduced maximum rotation speed, and potentially loss of step accuracy (particularly at larger microstepping values).

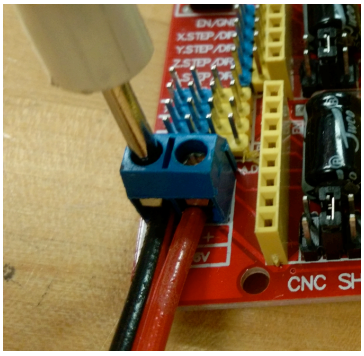
    b. The microstepping mode for each Stepper Driver board is set using the M0, M1, and M2 jumpers. The table below shows which jumper settings correspond to which mode. A reasonable place to start is 1/4 microstepping for all three drivers.



M0	M1	M2	Microstep Resolution
Open	Open	Open	Full step
Closed	Open	Open	Half step
Open	Closed	Open	Quarter step
Closed	Closed	Open	Eighth step
Closed	Closed	Closed	Sixteenth step

5. Install the barrel jack to bare leads cable

    a. The bare ends of the cable go into the 2-position blue terminal block on the CNC Shield. The **red** wire goes into to the positive (+) terminal, and the **black** wire goes to the negative (-) terminal. DON'T GET THESE MIXED UP!



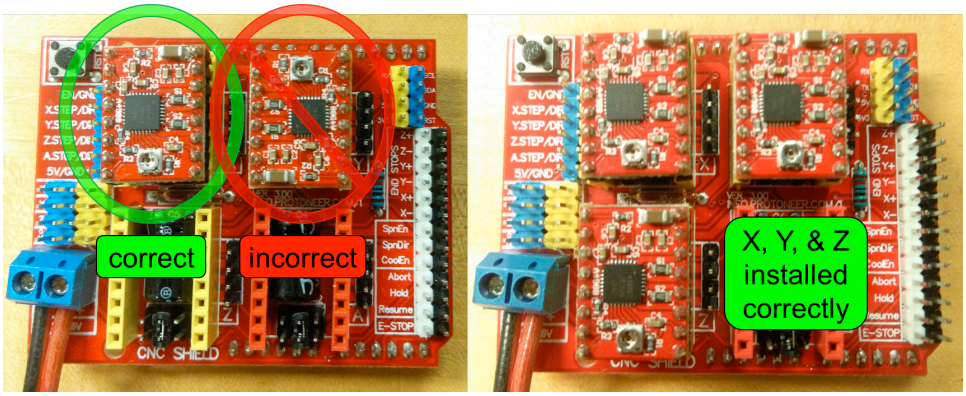
b. You may need to loosen the screws on the terminal block to fully insert the bare leads (and you might also need to further strip and/or split the two wires).

c. After you've inserted the wires in the correct orientation, tighten the screws on the terminal block to retain the wire. You can use a multimeter to confirm you have a good connection between the barrel jack (the center pin is positive) and the terminal block.

6. Re-plug in the Stepper Driver boards to the X, Y, and Z sockets on the CNC Shield.

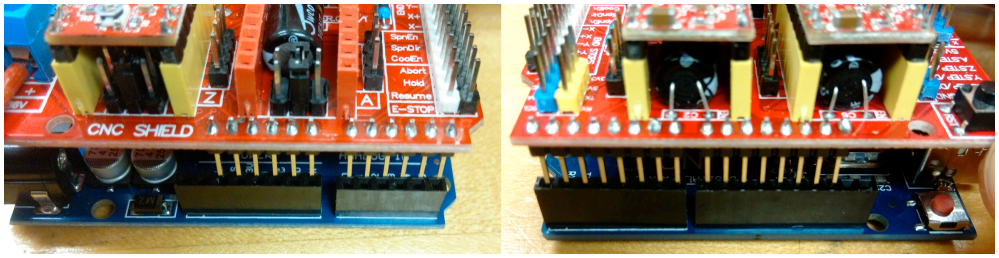
   a. PAY ATTENTION TO THE ORIENTATION OF THE BOARDS! The ENABLE pin on the Stepper Driver boards should be matched to the EN header on the CNC Shield. Another orientation check is to make sure the potentiometer on the Stepper Driver board sits above the jumper pins.

   b. If you plug the Stepper Driver boards in backwards there's a good chance you'll fry your boards and maybe your Arduino.

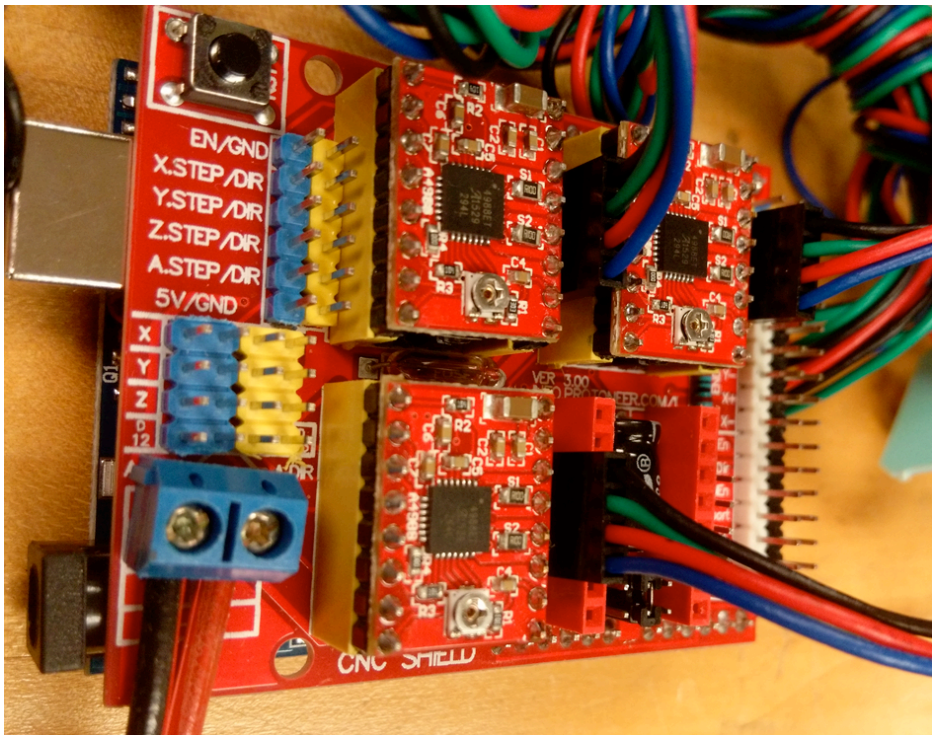


7. Re-plug in the CNC Shield to the Arduino

   a. MAKE SURE YOU MATCH THE PINS UP APPROPRIATELY!

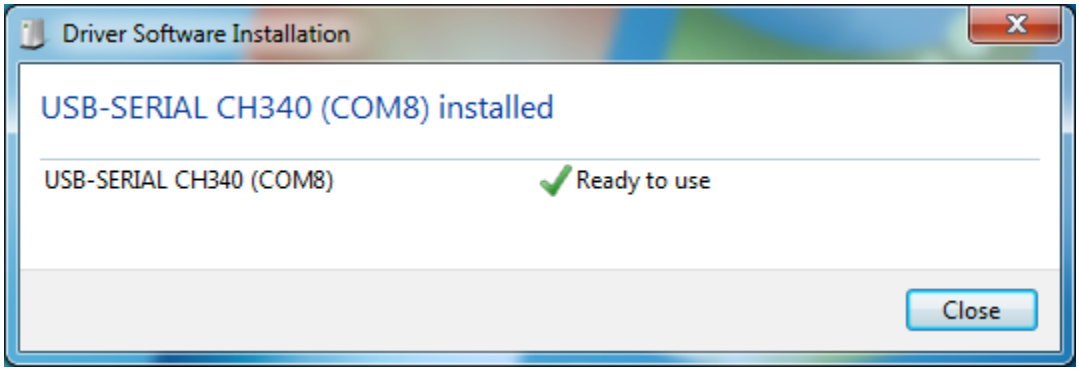


8. Plug the three stepper motors into the CNC Shield
    a. If you plug all three motors in with the blue wires to the bottom (see picture), the motor directions should match the right hand coordinate system of a CNC mill with the motors located at +X, -Y, and +Z.



    b. The rotational direction convention for the stepper can be reversed by flipping the stepper connector on the 4 pins of the CNC shield, or it can be reversed via the Grbl software interface.

9. Plug the Arduino (with the connected peripherals) into your computer using the USB A-B cable
    a. The little red “ON” LED on the Arduino board should illuminate.
    b. **(Windows)** If you allow Windows (tested on 7 and 10) to use Windows Update to look for driver software, it should recognize the board as something like a “USB-SERIAL CH340” or something similar, and automatically install the correct driver.



c. If the driver doesn't install automatically or correctly, unplug and replug the USB cable. If that doesn't fix it, you'll need to manually install the drivers for the CH430G USB-to-serial chip

   i. Windows 7 CH430G Driver - http://catalog.update.microsoft.com/v7/site/ScopedViewRedirect.aspx?updateid=032a878e-8ca0-40d2-b7b1-936640b0eecb

   ii. Windows 8/10 CH430G Driver - http://www.arduined.eu/ch340-windows-8-driver-download/

d. **(Mac OS X)** Mac will most likely not recognize the board and install any drivers. Therefore, you will need to manually install the drivers.

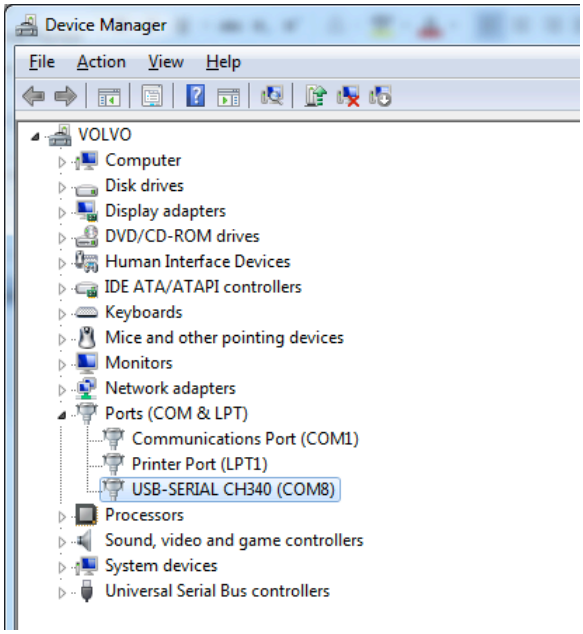
   i. Mac OS X El Capitan Driver - http://kiguino.moos.io/downloads/CH34x_Install.zip

10. Verify the serial port assignment for your Arduino board

a. If a serial port assignment (COM# in Windows, and /dev/tty… in Mac OS X and Linux) wasn't given during the driver install process (e.g. COM8 for the example picture in **Step 9b**), we need to figure out which serial port identifier your computer assigned to the board. This is computer-specific, and will likely change if you use the board on a different computer.

b. **(Windows)** To check this, open Device Manager and expand the Ports (COM & LPT) tab. What we're looking for is a USB-SERIAL CH430 entry (or something similar). In the example picture below, our board has been assigned COM8.

   i. If you have a lot of similar entries shown, an easy way to check which one corresponds to your board is to unplug the USB cord and see which port disappears (and vice-versa, which port appears when you re-plug the cord.



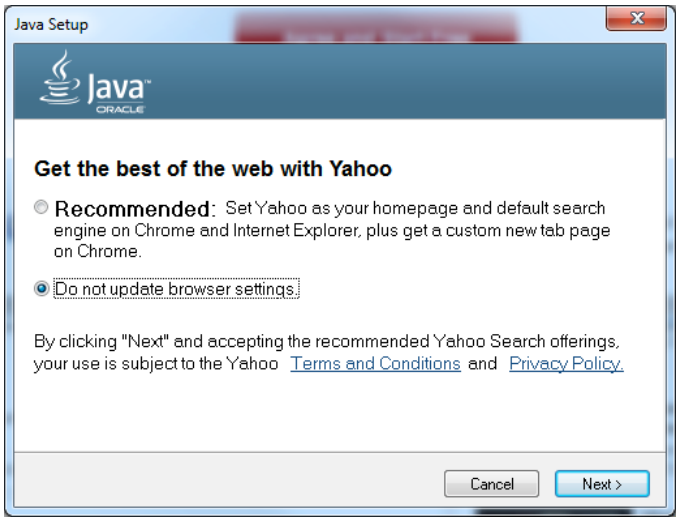
c. **(Mac OS X)** If the drivers were installed correctly, you should find a serial device in the device list (/dev/tty.wchusbserial*). To check this, open system report -> Hardware -> USB and on the right side under the "USB Device Tree" you should find a device named "Vendor-Specific Device". If you cannot find it, try to reinstall the drivers again.

    i. **Note:** You may need to go into System Preferences -> Security & Privacy -> General and Allow apps downloaded from Mac App Store and identified developers to ensure the driver will work properly.

Test G-code Interpretation

The Arduino board you received with your hardware kit has already been flashed with the latest version of Grbl (as of 1/26/2016, version 0.9j). If you need to reflash the firmware, see the section at the end of this document ("Uploading the Grbl Firmware").

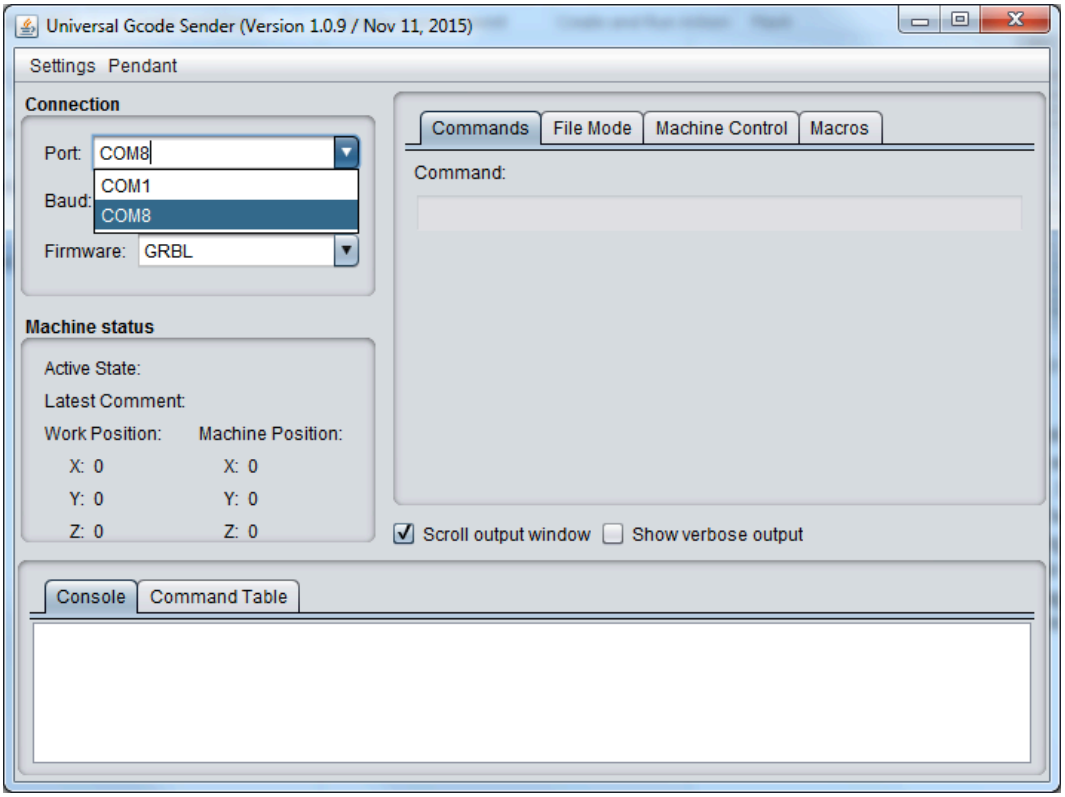
1. For this test, you should have your Arduino, CNC Shield, Stepper Driver boards, and stepper motors all connected together appropriately (as described in the "Preparing Your Hardware" section). Plug the Arduino into your computer via the USB cable, but leave the 12V power supply **unplugged** for now.
2. You'll also need to have downloaded the "2.70_start_files.zip" file from the course website.
    a. There are four folders inside this zip folder. We'll be using the "Gcode_interpreter" and the "Test_gcode_files" folders in this section.
3. Unzip the files to somewhere convenient for you and open the UniversalGcodeSender.jar file inside the Gcode_interpreter folder.
    a. You need Java (just the basic JRE) to run this program (as of 1/26/2016, the current version is 8 update 71). It's reasonably likely that you already have Java installed, but if you don't, you can download it from <https://www.java.com/en/download/>. This page should auto-detect your operating system and download the appropriate installer. During the installation process, for some ridiculous reason, it will "recommend" that you switch your default homepage and search engine to Yahoo, so be sure to avoid that (unless you're really a Yahoo fan).

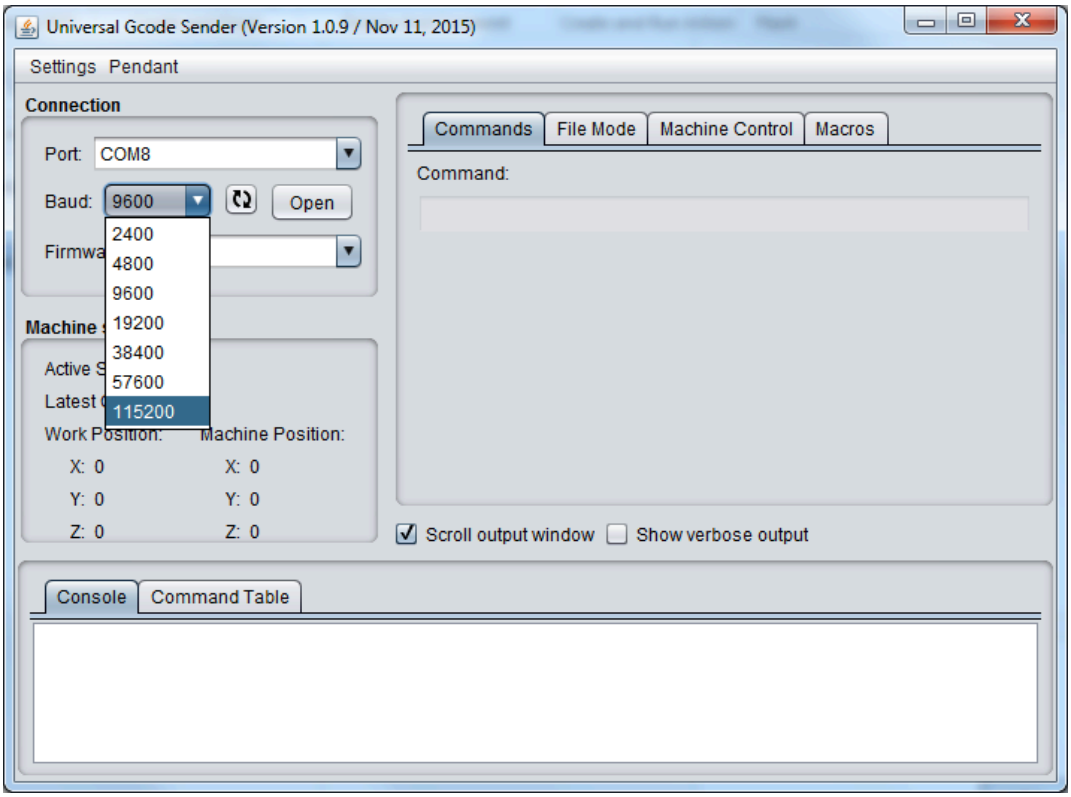


b. **(MAC OS X)** Make sure to use the Gcode Interpreter – Mac folder. This is an older version that seems to be more stable. You will need to create "/var/lock" directory on OSX to fix a bug in the serial library. To do this open the Terminal application and run the following two commands:

   sudo mkdir /var/lock
   sudo chmod 777 /var/lock

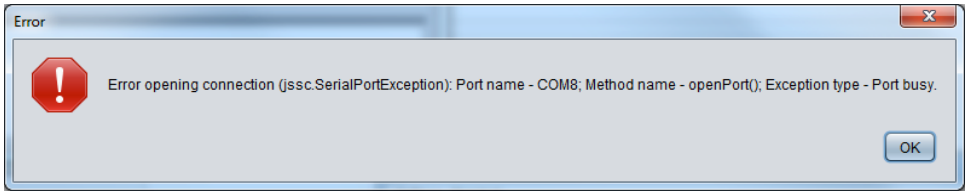
4. Once the .jar file opens, change COM port to the appropriate number for your system, and the baud rate to 115200



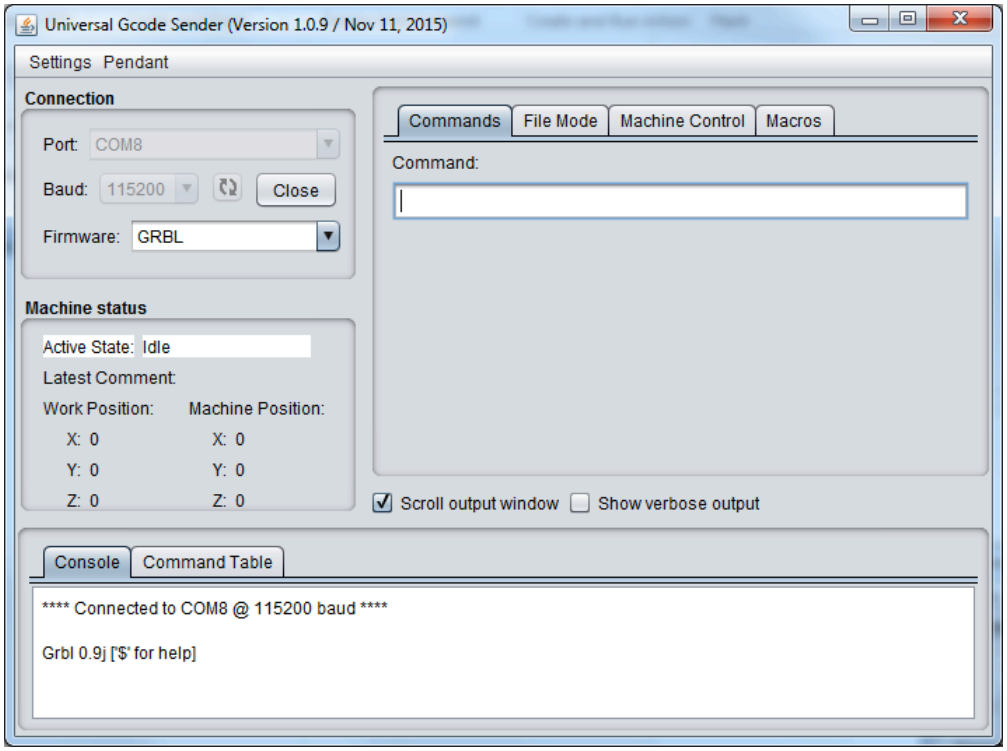


5. Click the “Open” button to open the serial port connection to your Arduino board.

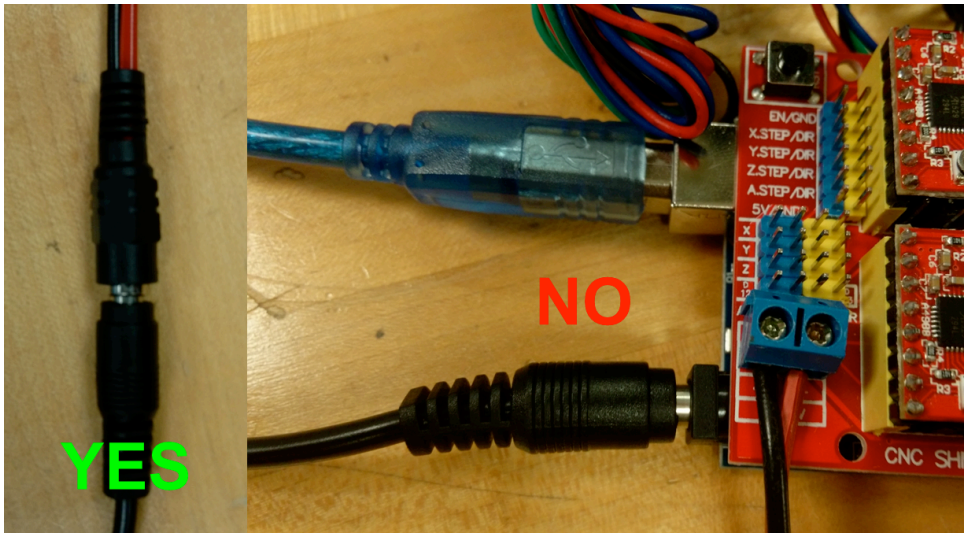
a. If you get an error like the one below, make sure any other serial connections (like the Serial Monitor from the Arduino IDE) are closed.



b. The console should show the Grbl startup message (“Grbl 0.9j [‘$’ for help]”) once the connection is successfully opened.



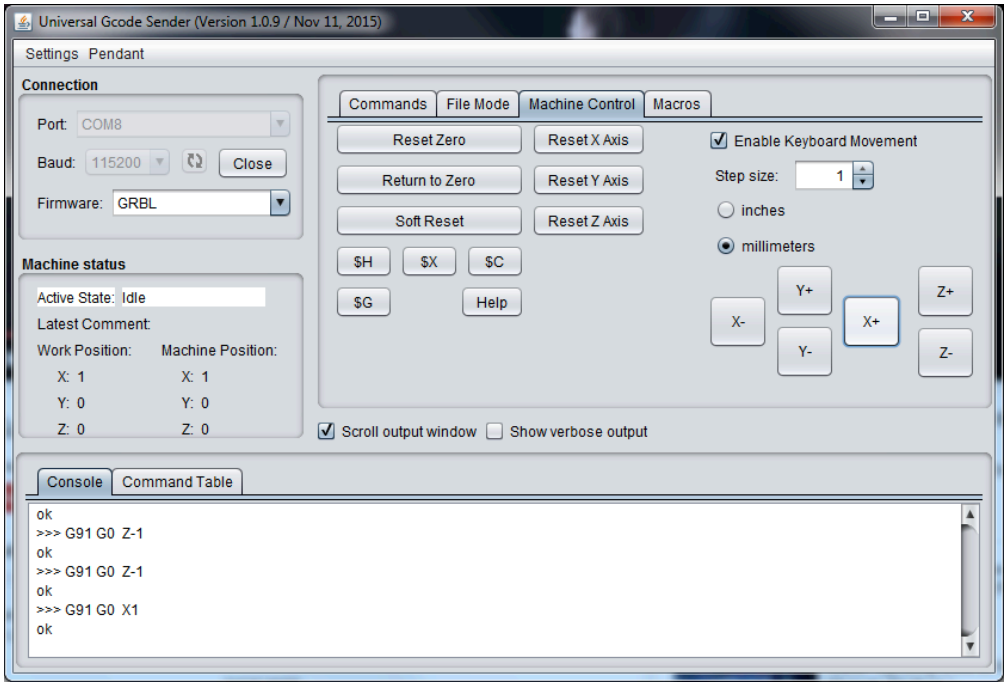
6. Plug in the 12V power supply to the barrel jack cable (**not** the barrel jack adapter on the Arduino)



   a. Now the stepper motors and the Stepper Driver boards are properly powered. If you hear any whining, or any of motors start moving, quickly unplug the power supply and check all of your connections.

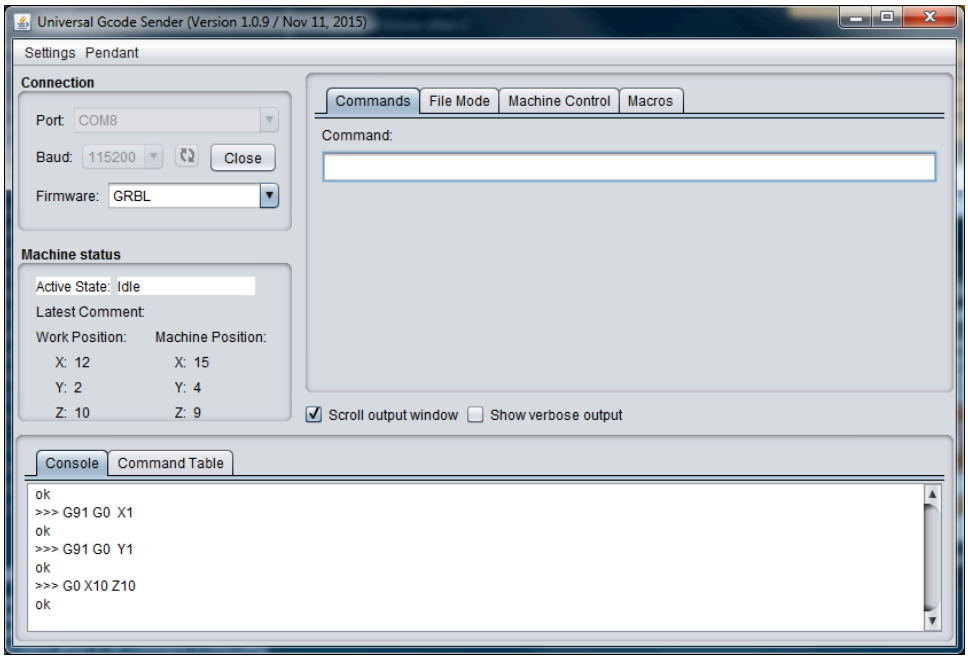
7. Use the “Machine Control” tab to jog your stepper motors

   a. The interface here is reasonably intuitive. You can play around with the different buttons to see what they do. The big thing is to confirm that you can move each of your stepper motors clockwise and counter-clockwise.

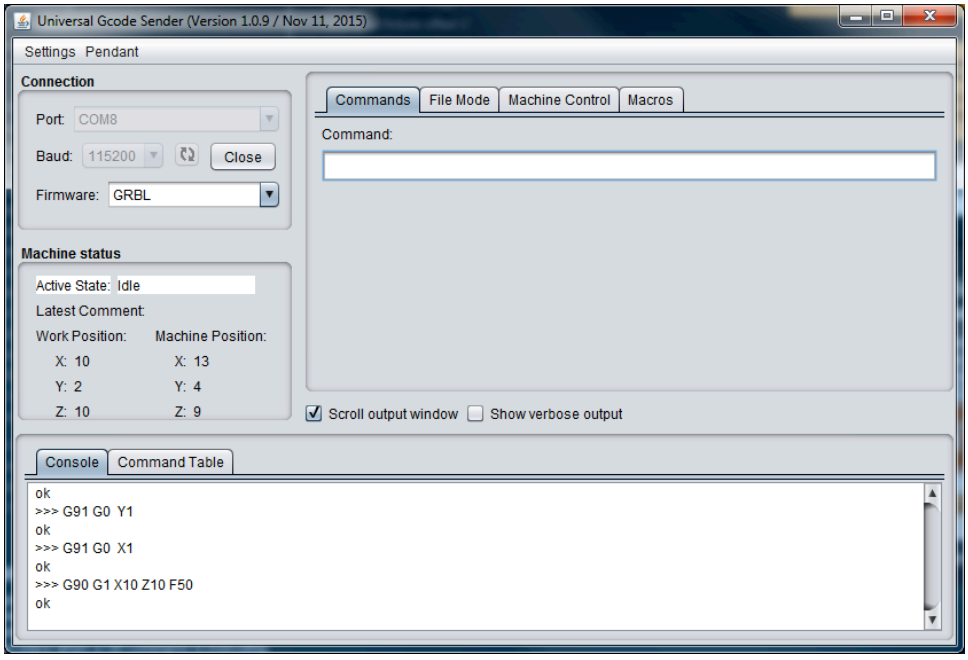


8. If you are familiar with Gcode, you can send some test Gcode commands using the “Commands” tab.

    a. Example #1: from a Work Position of (2,2,0), the command “G0 X10 Z10” will perform a rapid, incremental move of the X and Z steppers to a Work Position of (12,2,10)



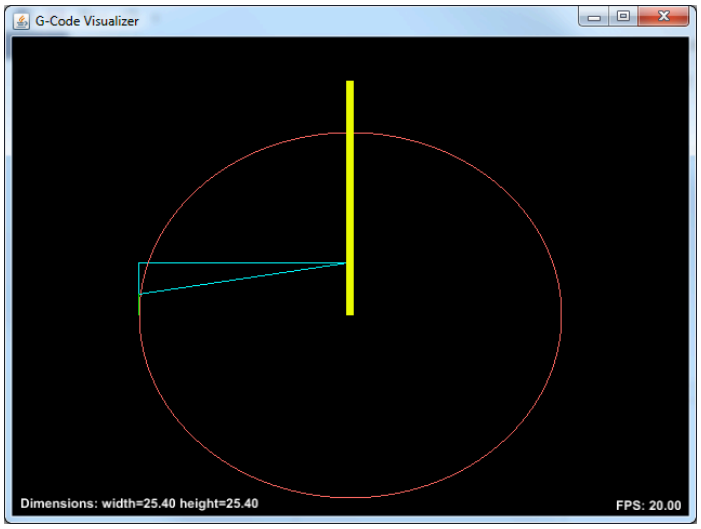
    b. Example #2: from a Work Position of (2,2,0), the command “G90 G1 X10 Z10 F50” will peform a linear interpolation move to a Work Position of (10,2,10) at a speed of 50 units/min.



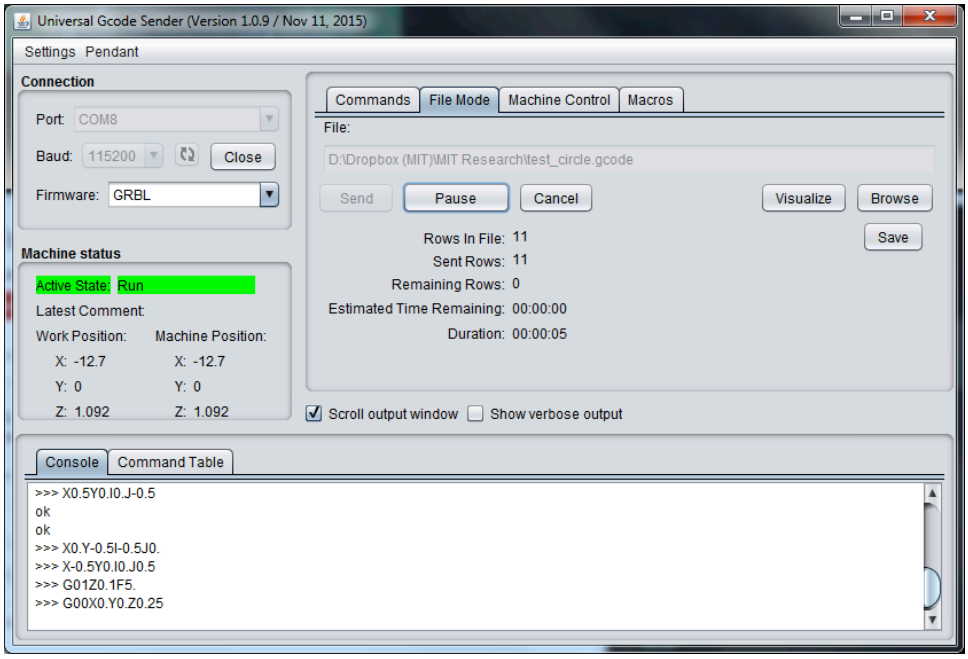
9. Universal Gcode Sender can also send and graphically represent full CAM files.

    a. To load a CAM file (aka a Gcode file), select the “File Mode” tab and click the “Browse” button. Select a gcode file you’d like to use (the file “test_circle.gcode”, which is included in the “Test_gcode_files” folder of the “2.70_start_files.zip” file is a good one to start with).

    b. To see if your file has been properly interpreted, you can click the “Visualize” button, which will open up a window with a graphical representation of the tool path defined by your Gcode.



    c. To run the Gcode file, click the “Send” button, which will begin streaming the commands to the Arduino and motors. There may be a small lag between what the GUI displays and what the motors are actually doing (in both the “Machine Status” box and the “G-Code Visualizer” window).

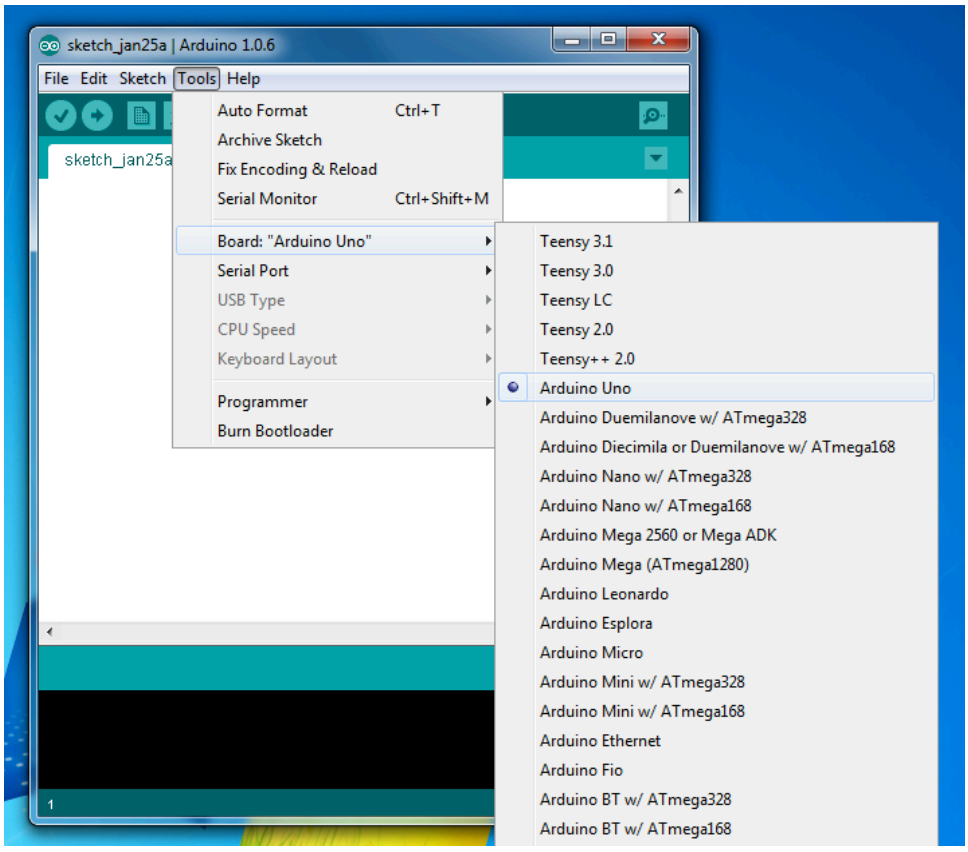


10. There are many other Grbl-compatible G-code sender/visualizers. A good starting list can be found here <https://github.com/grbl/grbl/wiki/Using-Grbl>.
11. The Grbl wiki page <https://github.com/grbl/grbl/wiki> (and particularly <https://github.com/grbl/grbl/wiki/Configuring-Grbl-v0.9>) is great resource for learning about the different settings, configurations, etc. that can be adjusted to fit your particular use case.

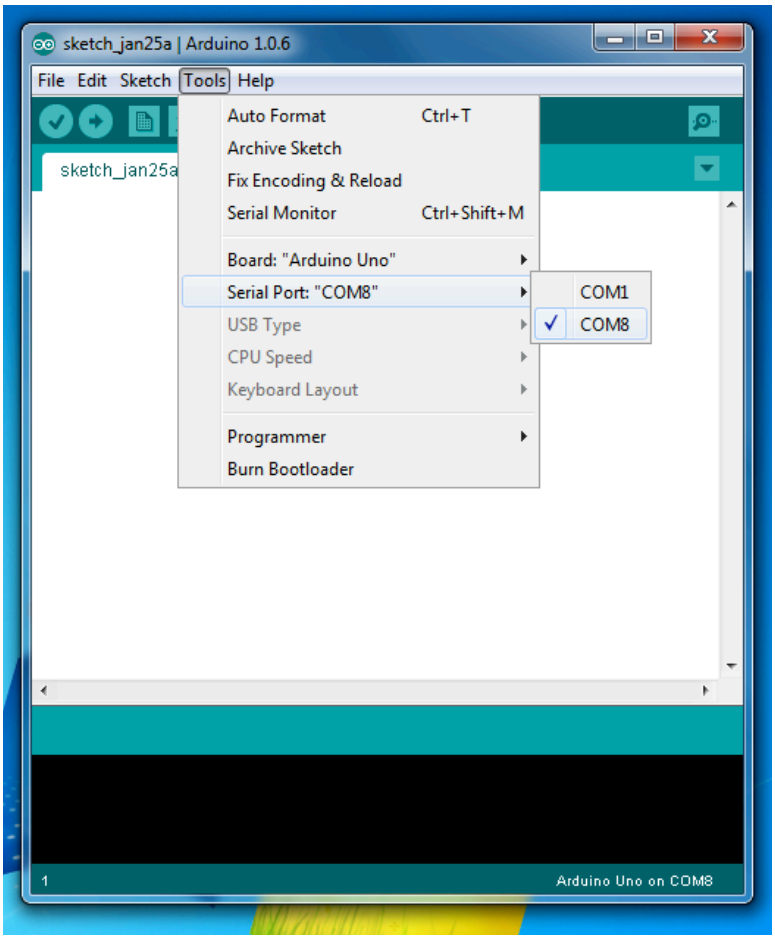
Uploading the Grbl Firmware

Unless you are having problems with your Arduino setup, this sequence of steps should be unnecessary (the Arduino boards included with the hardware kits were already flashed with Grbl 0.9j).

1. Install Arduino software (arduino.cc)
    a. The classic (1.0.6) or the latest (1.6.7) versions both work.
        i. *This tutorial was done using the classic 1.0.6 Arduino IDE.*
    b. Make sure to pick the correct version for your operating system.
2. Open Arduino
3. Check that your Arduino board is being properly recognized by the software
    a. Tools → Board → select Arduino UNO (or something similar like Genuino UNO)



    b. Tools → (Serial) Port → select the appropriate serial port identifier for your Arduino (which you found at the end of the “Preparing Your Hardware” section above).



   c. If it isn't listed, make sure your Arduino board is plugged in and the drivers are properly installed.

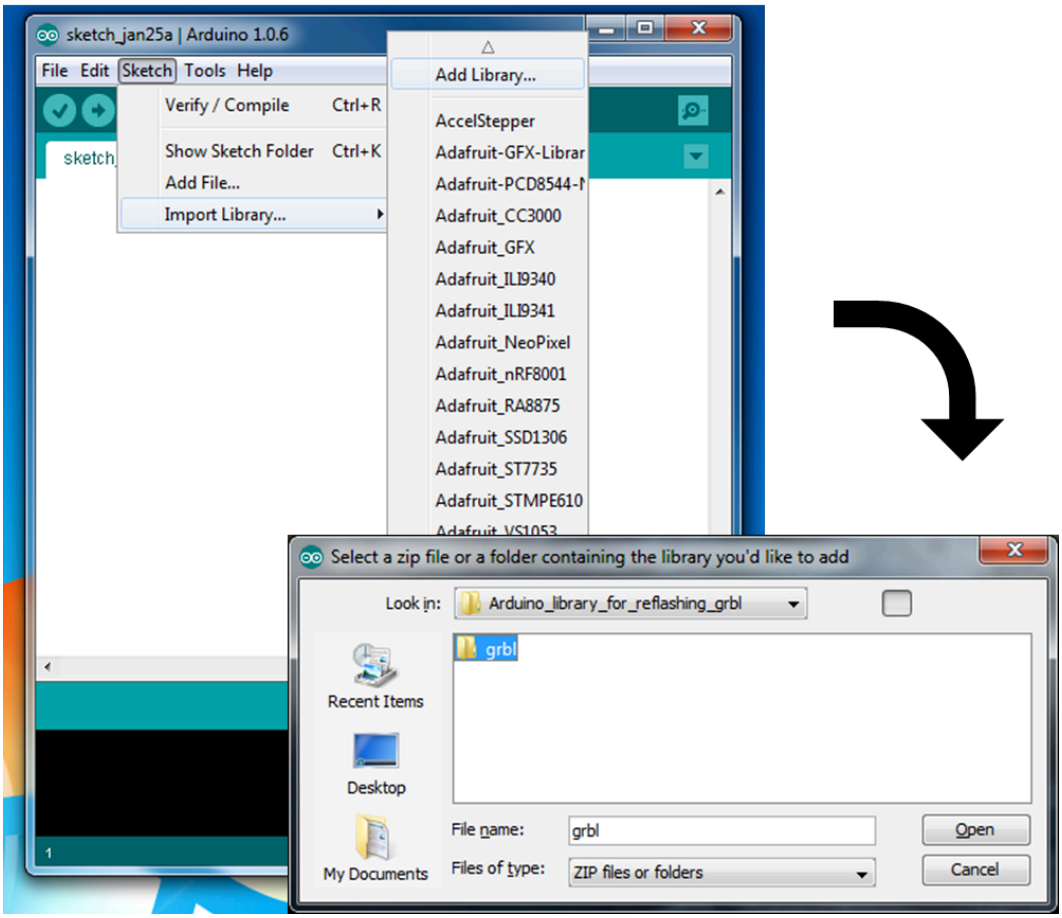
4. Locate the "grbl" folder inside the "Arduino_library_for_reflashing_grbl" folder from the "2.70_start_files.zip" file.

   a. This folder contains Grbl version 0.9j (the current version as of 01/26/2016)

5. Add the Grbl "library"

   a. (Arduino 1.0.6) Sketch → Add Library → select the "grbl" folder

   b. (Arduino 1.6.7) Sketch → Include Library → Add .ZIP Library → select the "grbl" folder

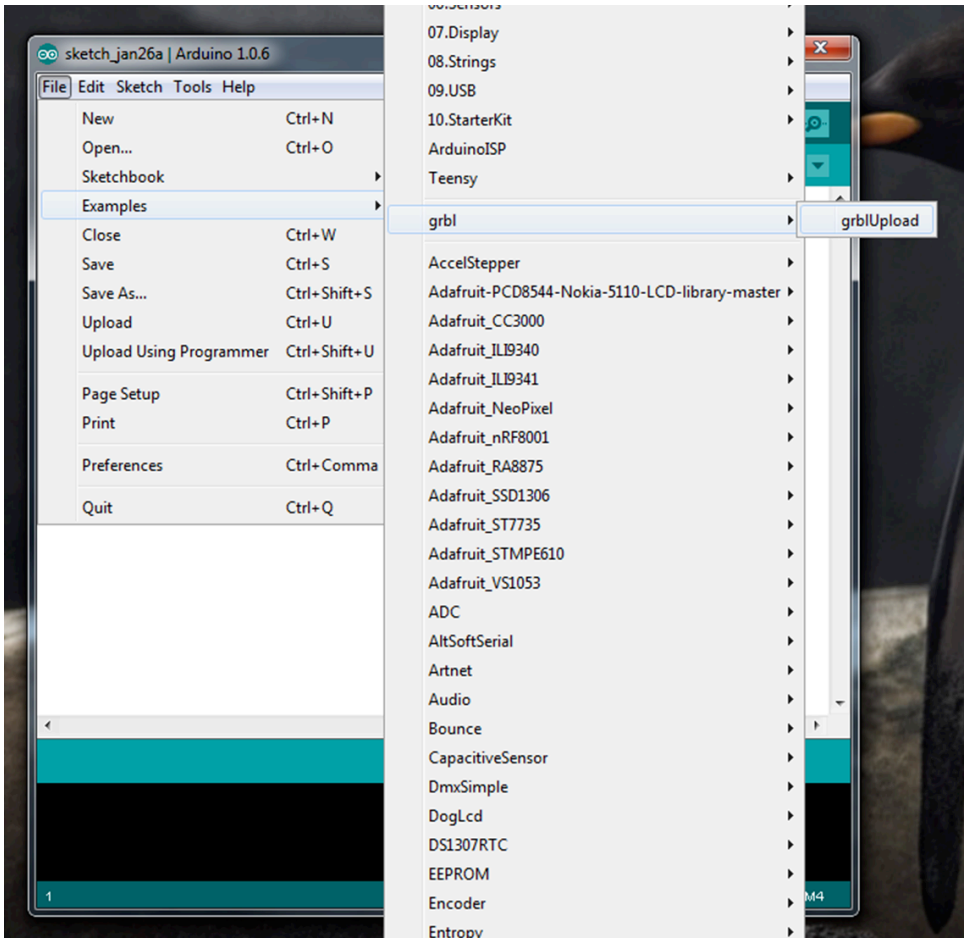


c. A message should display in the Arduino IDE indicating that the library import was successful.

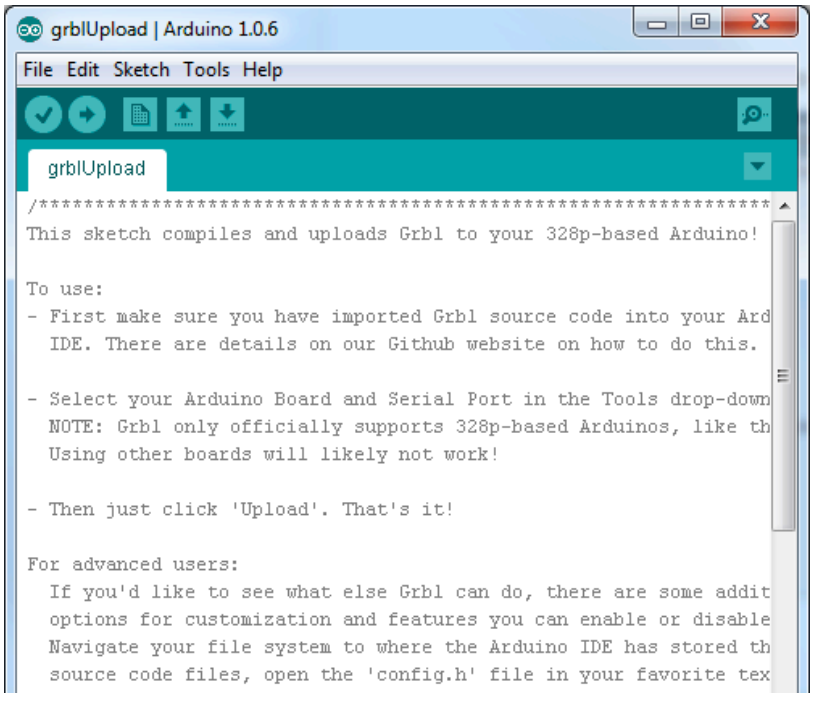
    i. If you get an error, make sure you are selecting the “grbl” folder, not anything in a higher or lower directory.

6. Open the “grblUpload” sketch

    a. File → Examples → grbl → grblUpload



b. An Arduino window with the grblUpload sketch should open



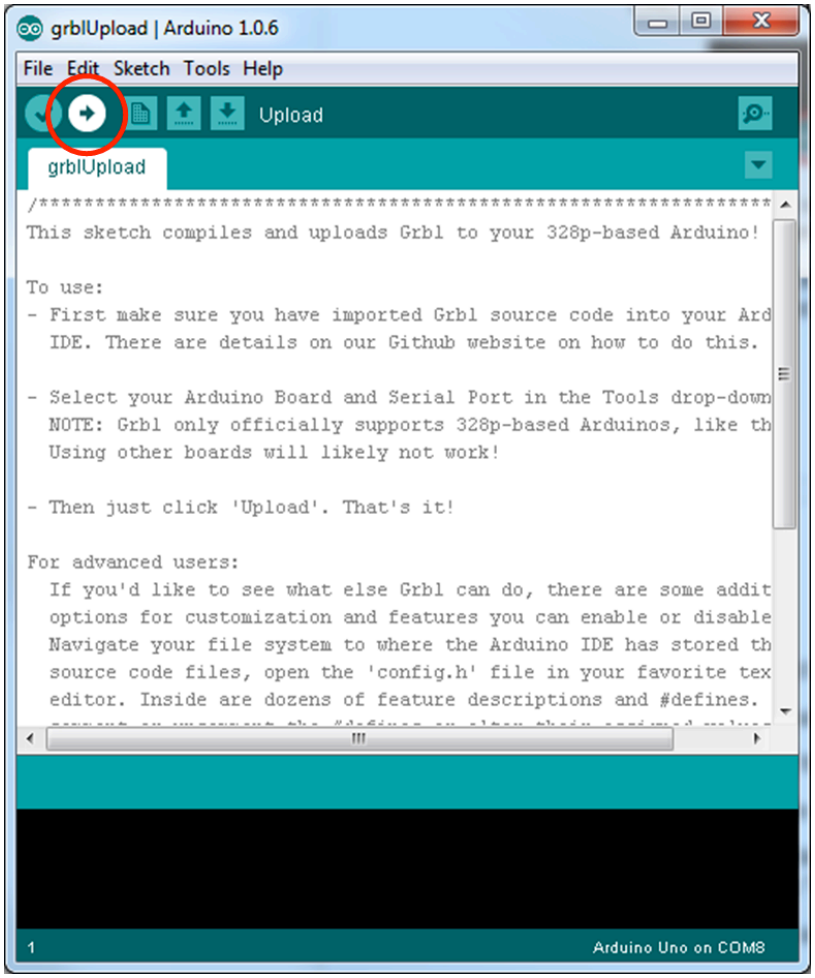
7. Upload the sketch to your Arduino board

    a. Verify that at the bottom right of the software window, something like “Arduino UNO on COM#” or “Arduino UNO on /dev/tty….” is displayed.

        i. If it says something like Arduino MEGA or Arduino Duemilanove, check your board settings via Step 9a of the “Preparing Your Hardware” section.

        ii. If the serial port identifier doesn’t match up with what you found in Step 4, reselect the correct port via Step 9b of the “Preparing Your Hardware” section.

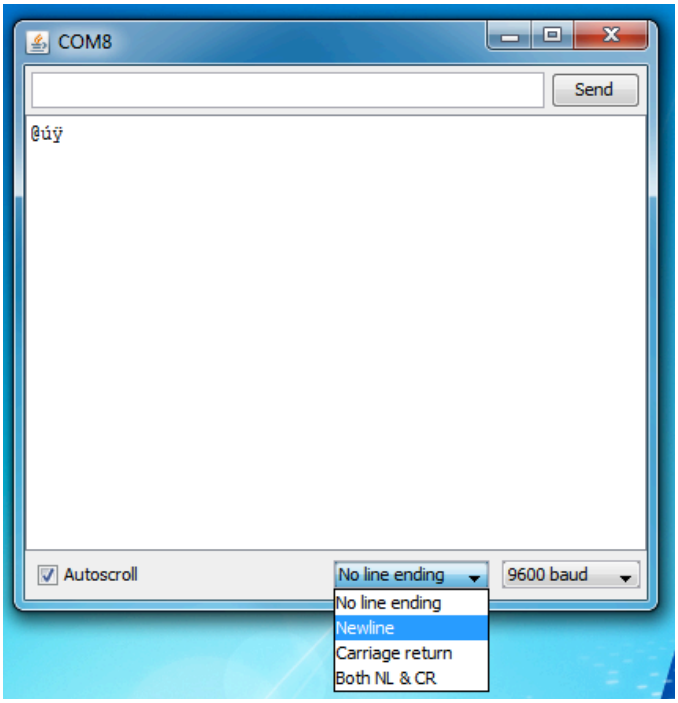
    b. Click the Upload button

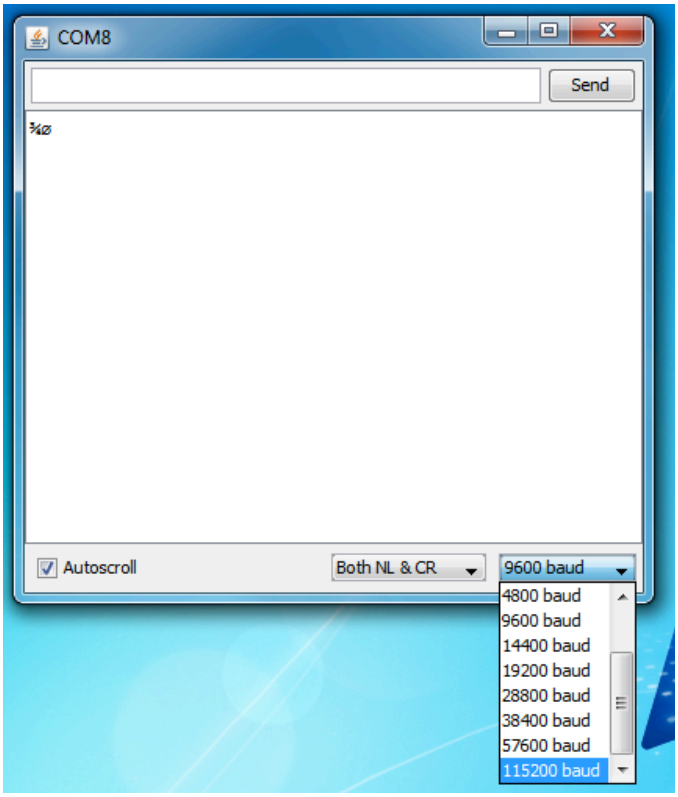


        i. This should take a little while (but still probably <60s).

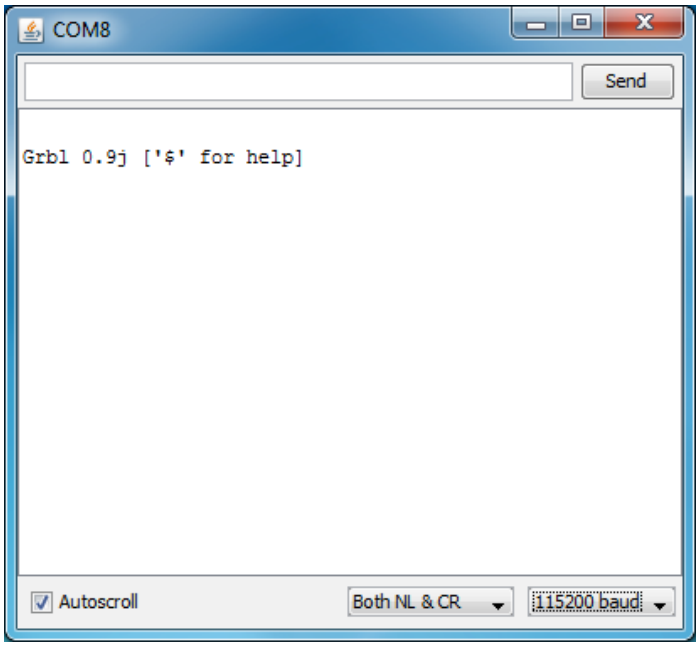
        ii. Some of the LEDs on the board will flicker

8. Test the Grbl firmware upload
    a. Click the Serial Monitor button and adjust the dropdown boxes from “No line ending” to “Newline” and “9600 baud” to “115200 baud”.
        i. Note: Carriage return is also a valid command terminator for Grbl.





b. You should see a message “Grbl 0.9j [‘$’ for help]”



c. You can send commands to the board using the text box and the send button (or hit enter). $, ?, and $$ are good test commands to send.

   i. If you are seeing readable information returning on the serial monitor, the Grbl firmware is correctly set up.

   ii. If you are only receiving garbage characters back, check your baud rate and line ending settings.

   iii. Here is the response to a three “?” (current status) commands.