Arduino Thermistor base DAQ (Data Acquisition)

**Print or View this document in COLOR

Overview

A thermistor is a resistor whose resistance (ohms) is dependent on temperature. An Arduino is a microcontroller that can be programed in C++ and run independent or with a serial connection to a personal computer. Both are inexpensive and fit the testing requirements.

A little more detail about Thermistors and Arduino

There are two types of thermistors, NTC where the resistance drops as temperatures increase, and PTC where the resistance increases as temperature increases. Thermistors come in many materials depending on the use case. A thermistor is characterized by its rated value, in our case 100KOhm, and its Beta value, ours is 3950. The resistance vs temperature curve is NON-linear, to account for this the temperature is calculated from the resistance and the given parameters via the Beta Parameter Equation;

$$\text{where } r_\infty = R_0 e^{-B/T_0}.$$

$$T = \frac{B}{\ln(R/r_\infty)}$$

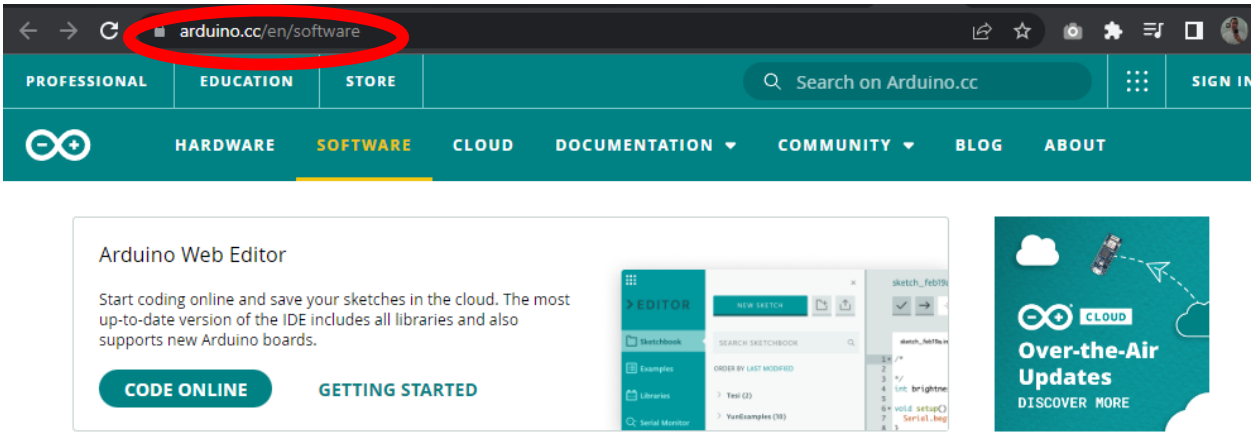
A “Mega” is a variation based on the Arduino platform that has 16 analog inputs available. The thermistor is passed 5V from the Mega, a matching resistor completes the circuit to ground. The analog input measures the voltage and returns a 10-bit value relative to the 5V input. The resistance is then calculated utilizing the voltage divider circuit:

$$V_{(x)} = \frac{R_{(x)}}{R_T} V_S$$

Requirements

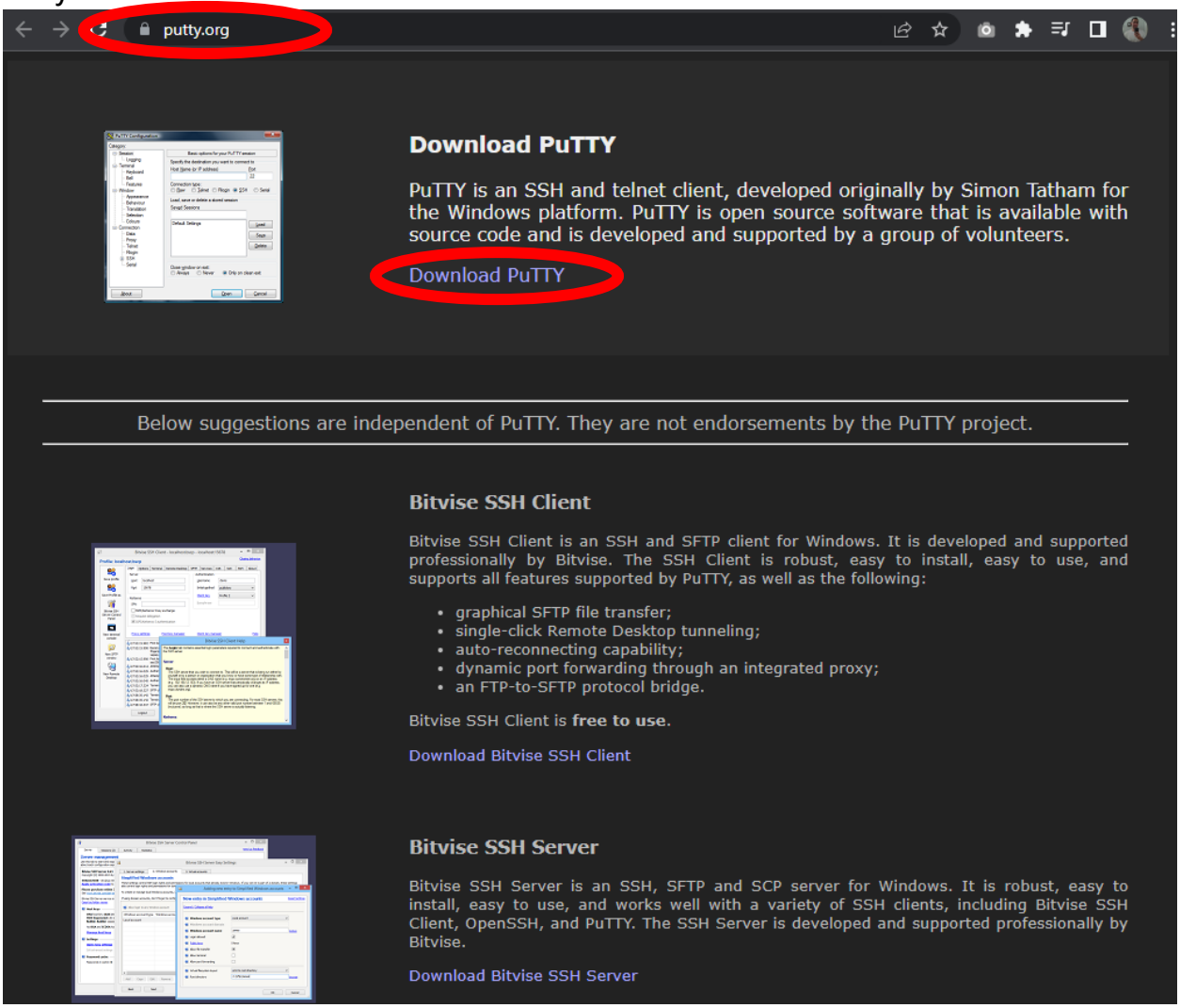
- Windows PC with USB port and cable (you can use Linux or Mac if the other requirements are available for it, but these instructions are based on Windows)

- Arduino Sketch:



Downloads

- Putty



- Python (3.8) or Mat-lab (2022a) [Other versions likely to work as well] This is for data analysis which is an art left for the user.

High Level View Operation

1. Connect the Arduino Mega to the PC with the USB cable
2. Open Arduino Sketch on PC
3. Select the Arduino Mega, connected COM port and baud rate (wait for walkthrough below)
4. Open the plotter and make sure thermistor readings are consistent with one another (no outliers)
5. Open the monitor and make sure the serial output is in the format you want (tx,T1,T2, … T16), rows of time, temperature readings (in C) separated by commas
6. Close Sketch

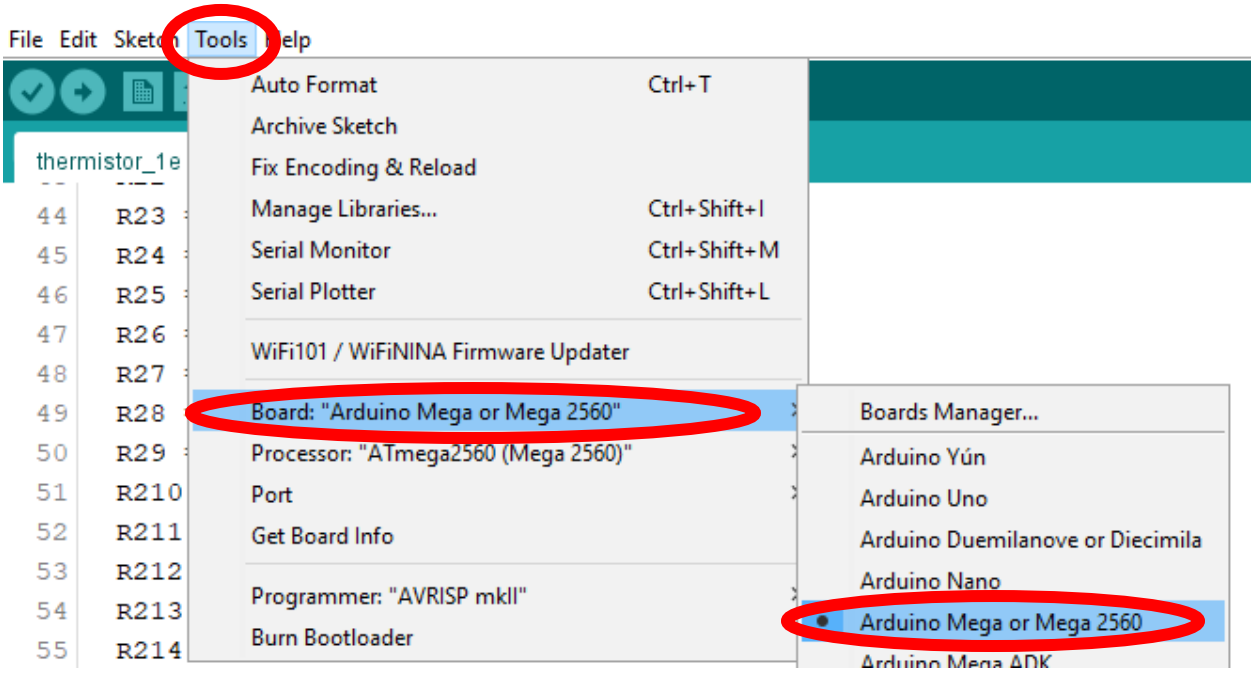
7. Open Putty
8. Configure Putty or load saved configuration (how to configure is below)
9. When test is ready, select “open” in Putty and press reset on Arduino Mega
10. You are now running the test. When the test is complete close the serial monitor Putty started and select again to end session.
11. RETREIVE YOUR DATA from the desktop, the putty.log file
12. You may unplug device to end testing.

Walkthrough

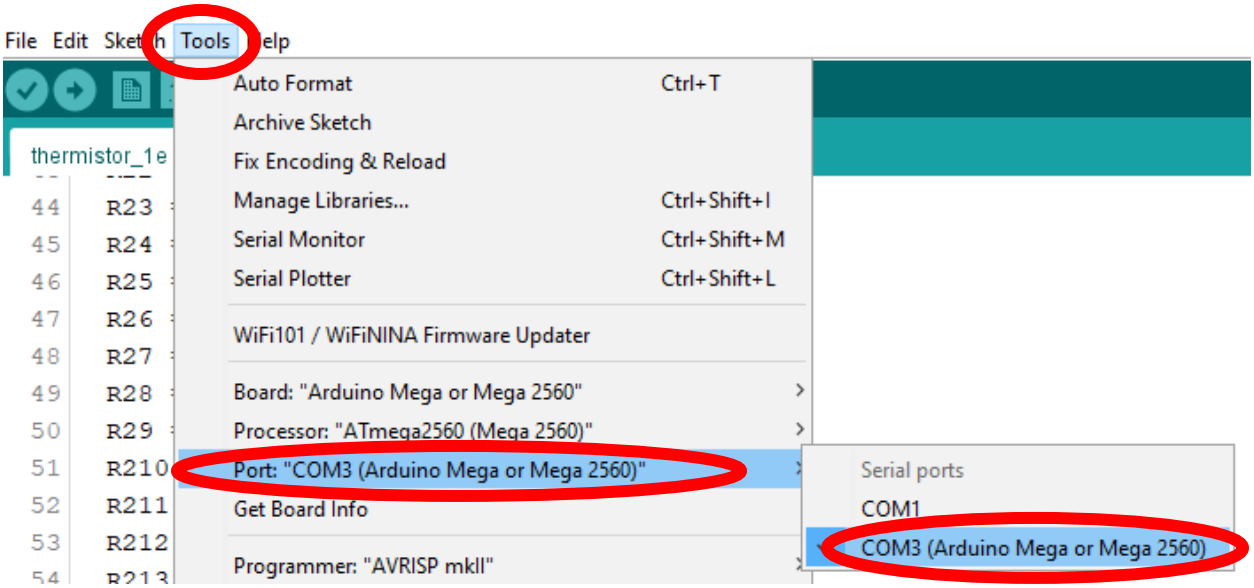
At this point you have installed Sketch and Putty, you have your computer and the Arduino with attached thermistors ready for testing. Plug the USB cable into your Arduino Mega and computer.

Open Sketch:

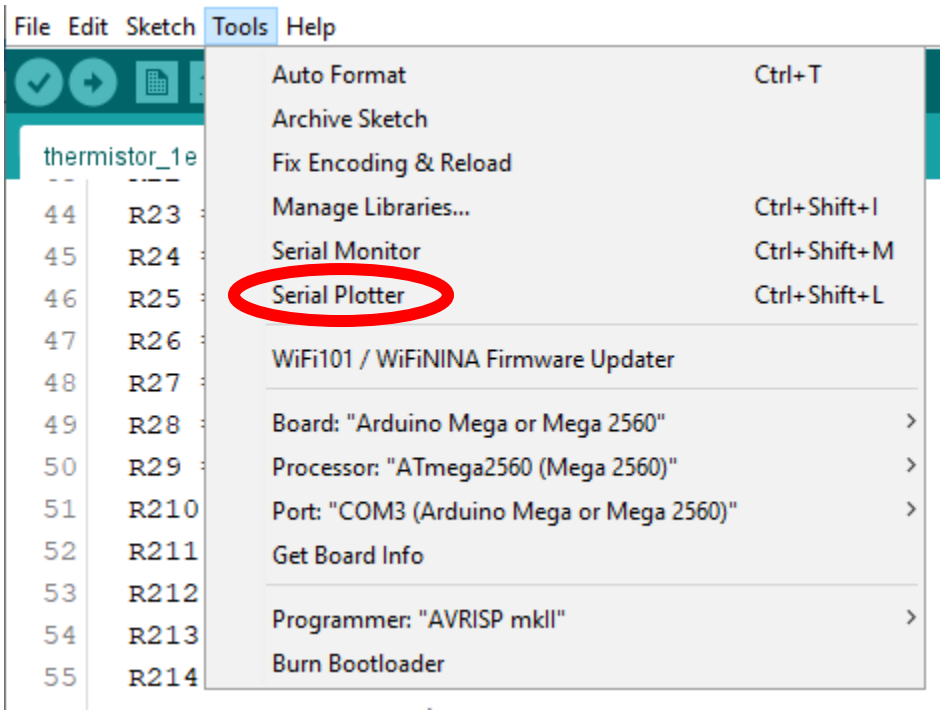
In Tools select “Board” and choose “Arduino Mega or Mega 2560”:



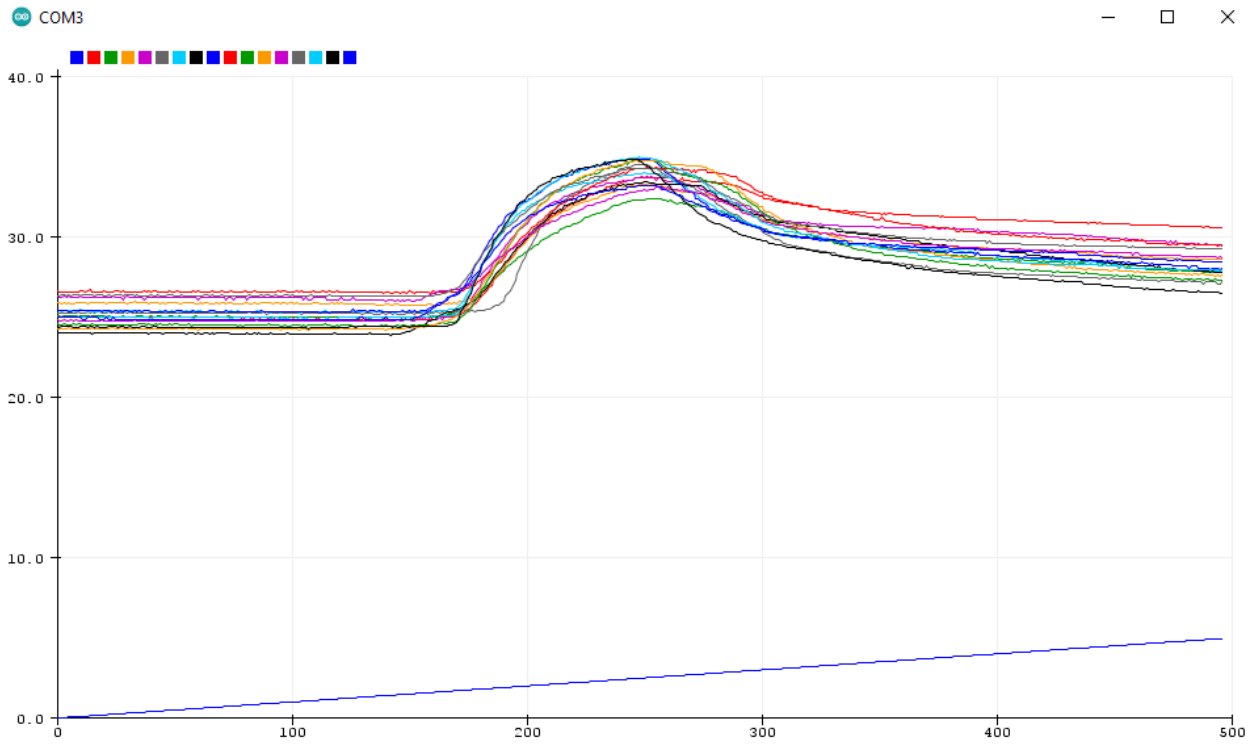
In Tools select “Port” select the COM listed with your Mega (Mine is COM3 this time):



In Tools select “Serial Plotter”:

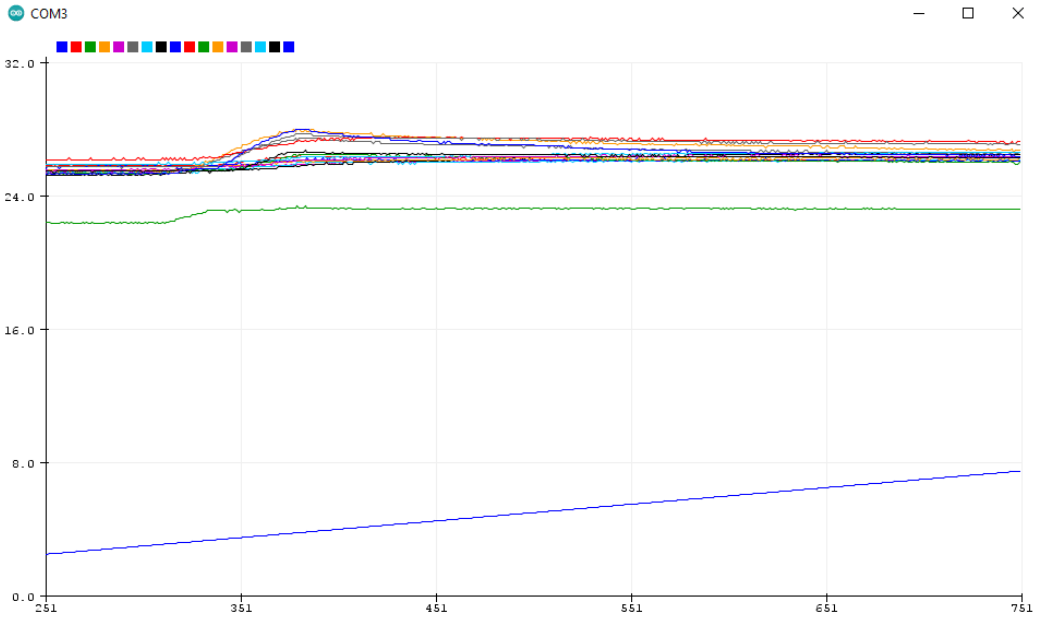


Your Plotter should look like this:



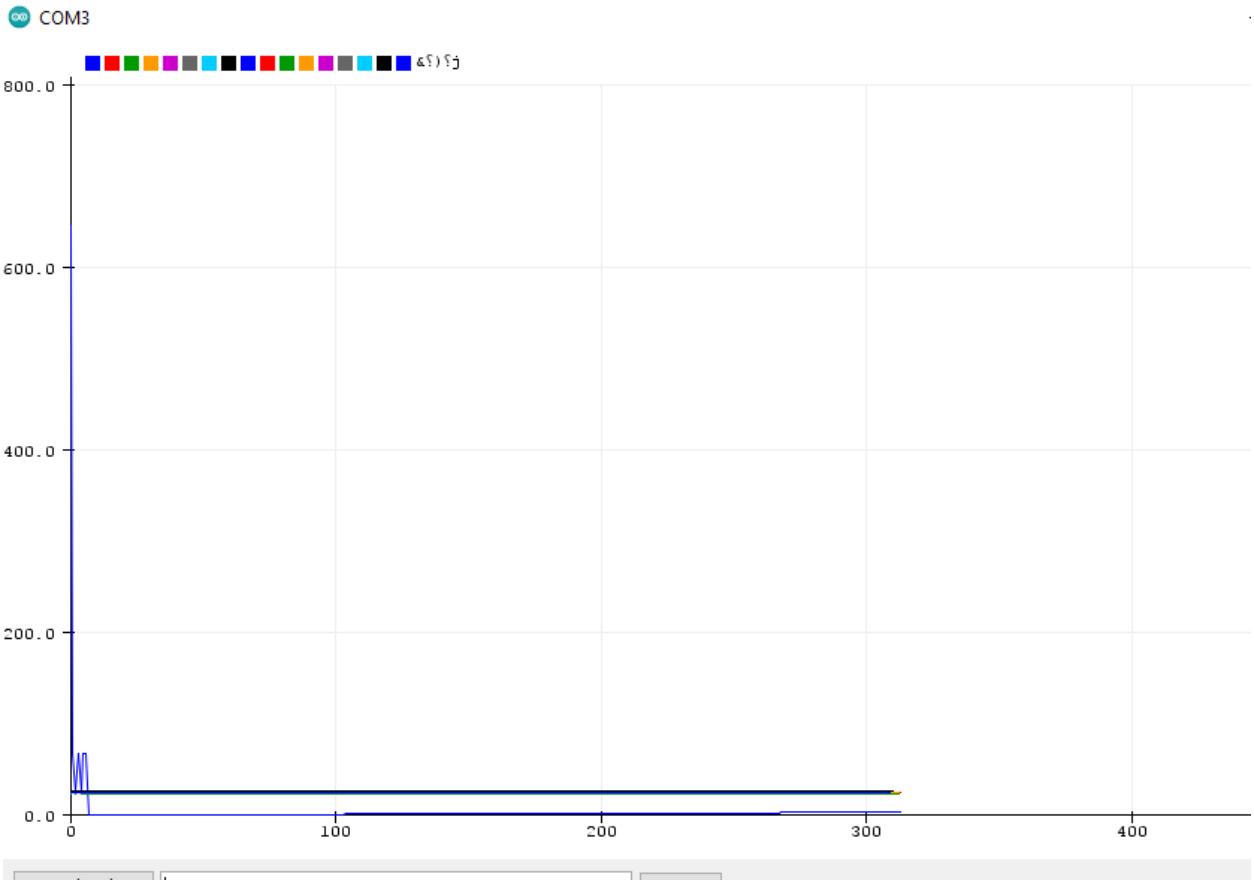
The blue line is the time stamp. Pick up the thermistors and cup them together and blow on them to test that they respond together.

Why does my plotter look like:



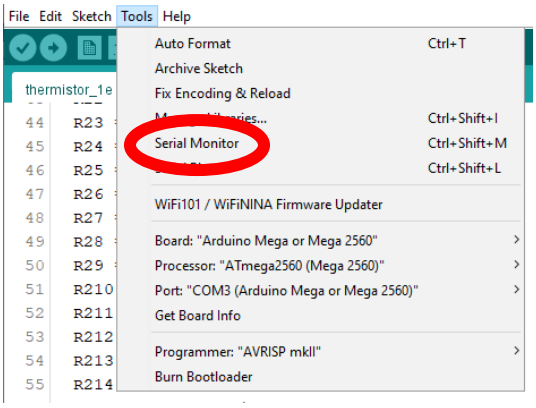
Here is the outlier scenario caused by a loose thermistor plug. Lightly press on the back of the connectors to fix and try not to move Arduino Mega after. Check plotter again before

data collection after nothing must be moved around again. The blue line at the bottom is the time stamp. The problematic thermistor plugs are typically the ones on edge by the power jumper on the inside as of this writing.



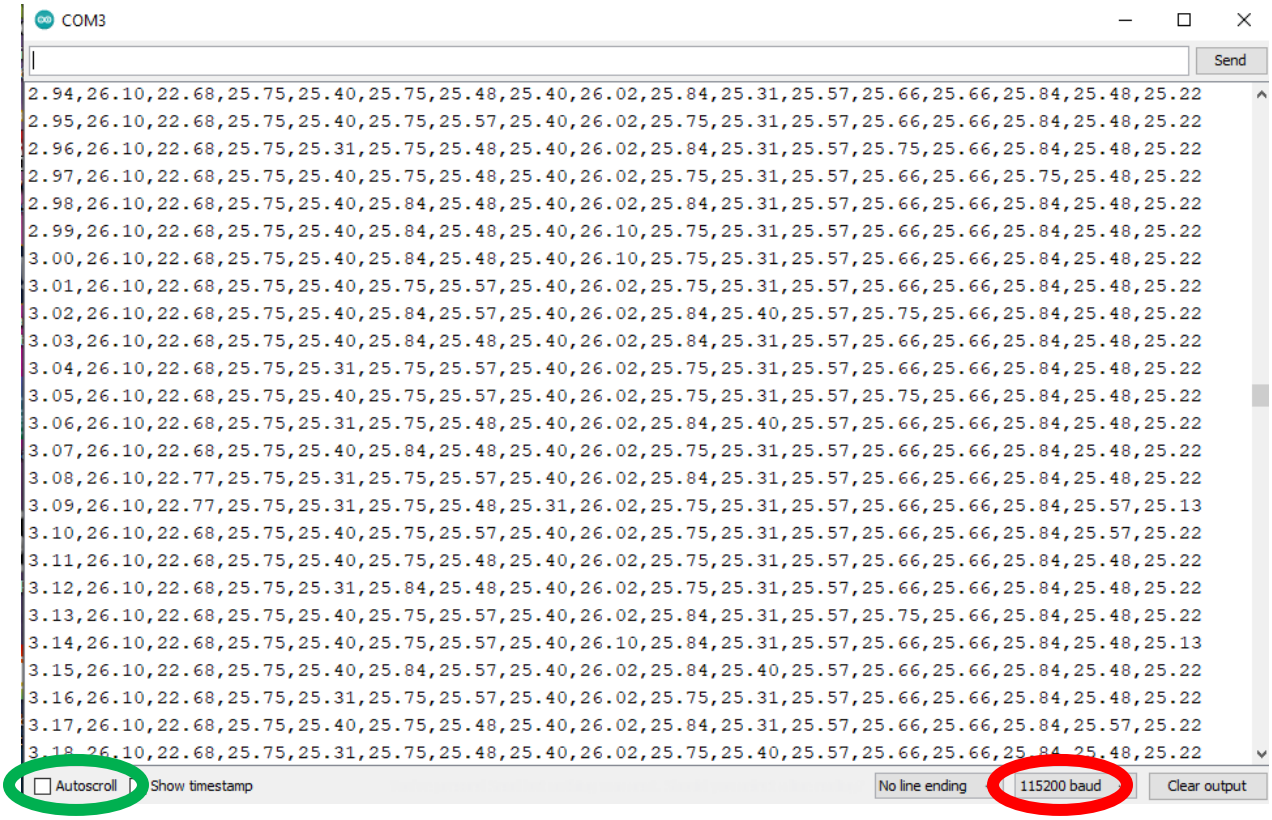
Notice the scaling range is exceptionally large. Close the plotter, unplug the Arduino Mega and plug back in. Re open the plotter.

In Tools select “Serial Monitor”:



Your monitor should open and look like this:

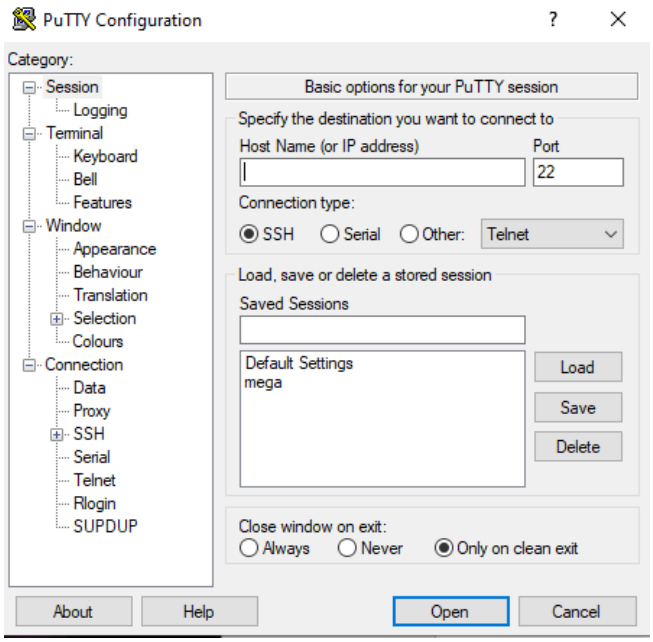
Note that Autoscroll should be checked (it was unchecked for the screen capture) and ensure the baud rate is 115200.



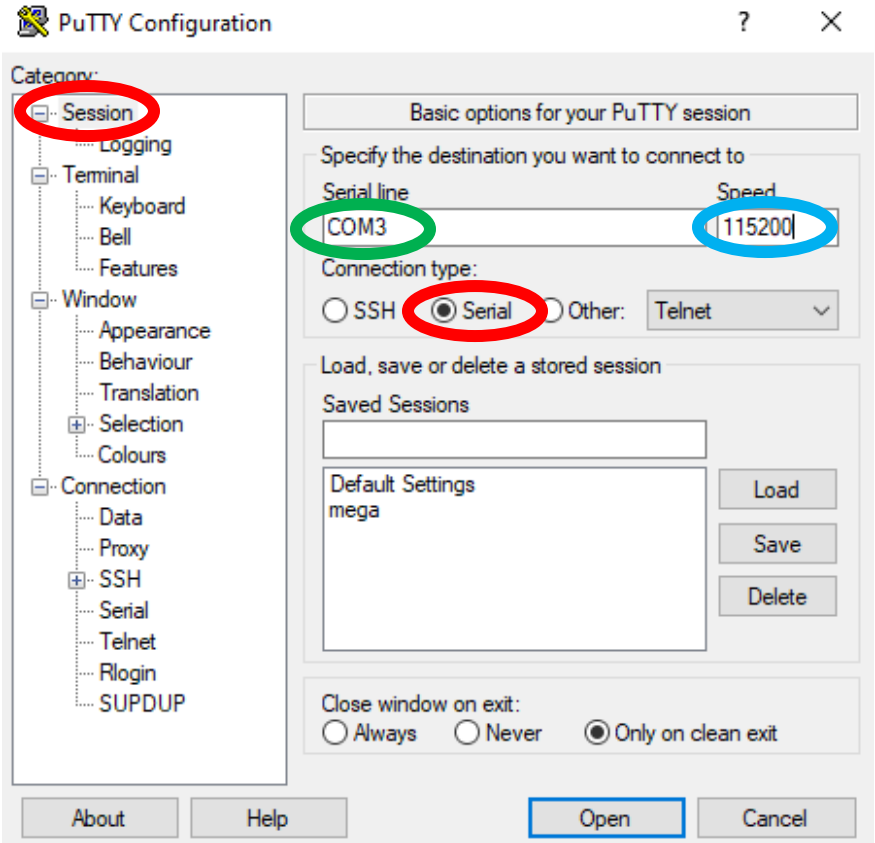
Comma separated rows. The first column is the time series (not actual time but aggregated delay). When plotting you may multiply the time stamps by the actual test time to produce a time correction coefficient.

Close Sketch, the plotter and monitor if still open and open Putty:

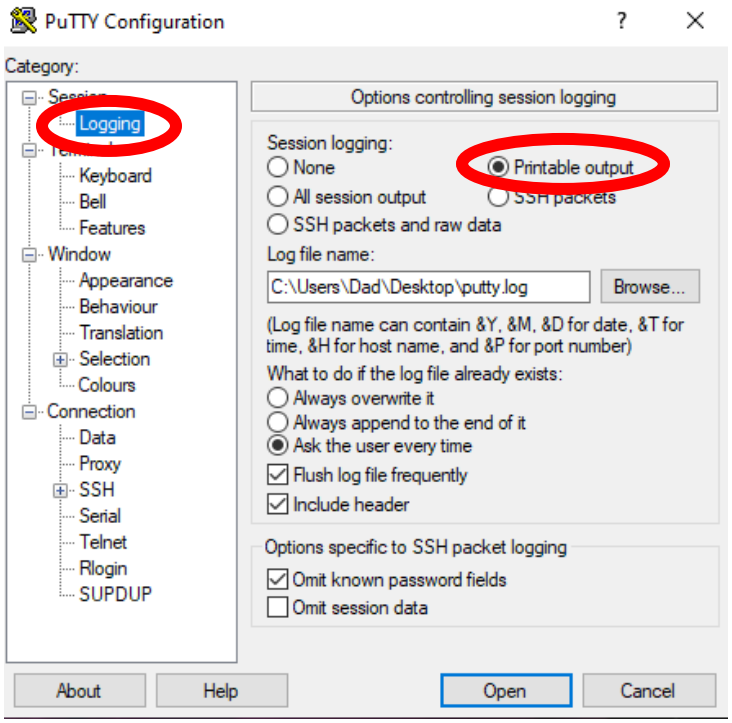
The PuTTY configuration window should open:



Select Session, then Serial, then enter the COM port you got earlier and change the speed (baud rate) to 115200.

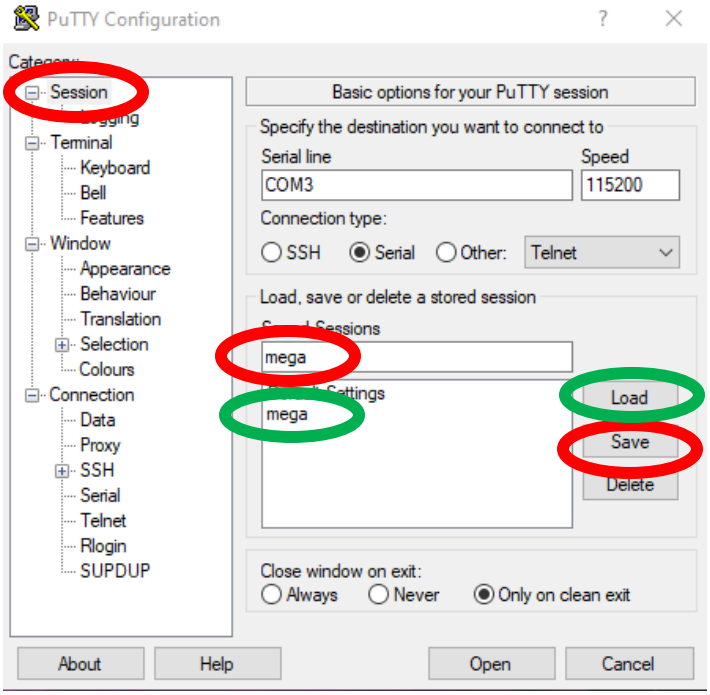


Select “Logging” and “Printable output”:

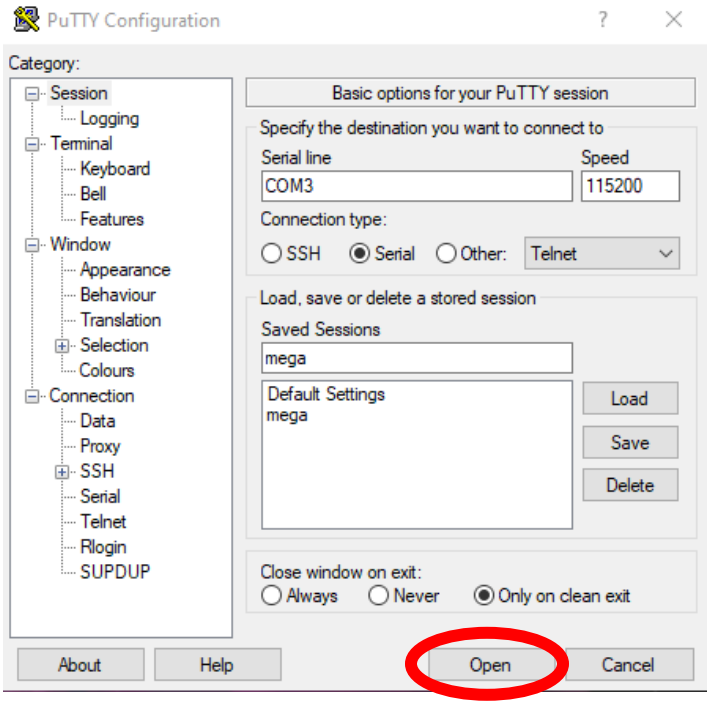


ADVISED BUT NOT NESSESSARY

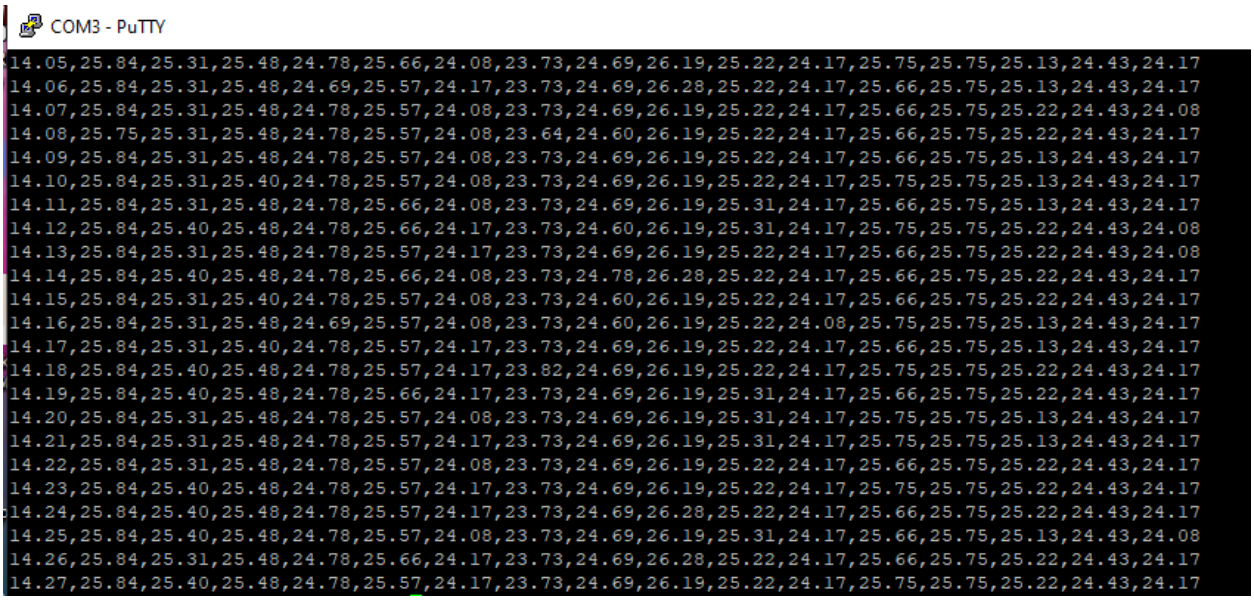
Go back to sessions, enter a name in “Saved Session” (Here I entered “mega”) and select “save”. When needed in the future select mega from the list and then press the load button to restore the saved settings.



When ready to begin data capture select open:

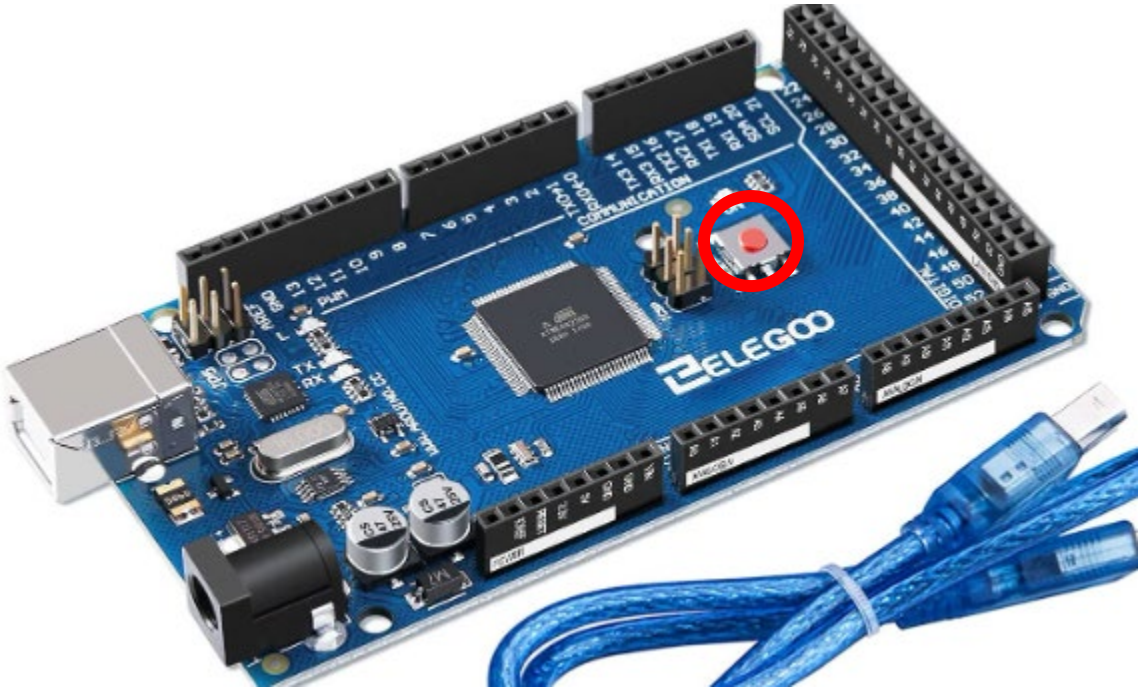


The PuTTY serial monitor should start and look like this:

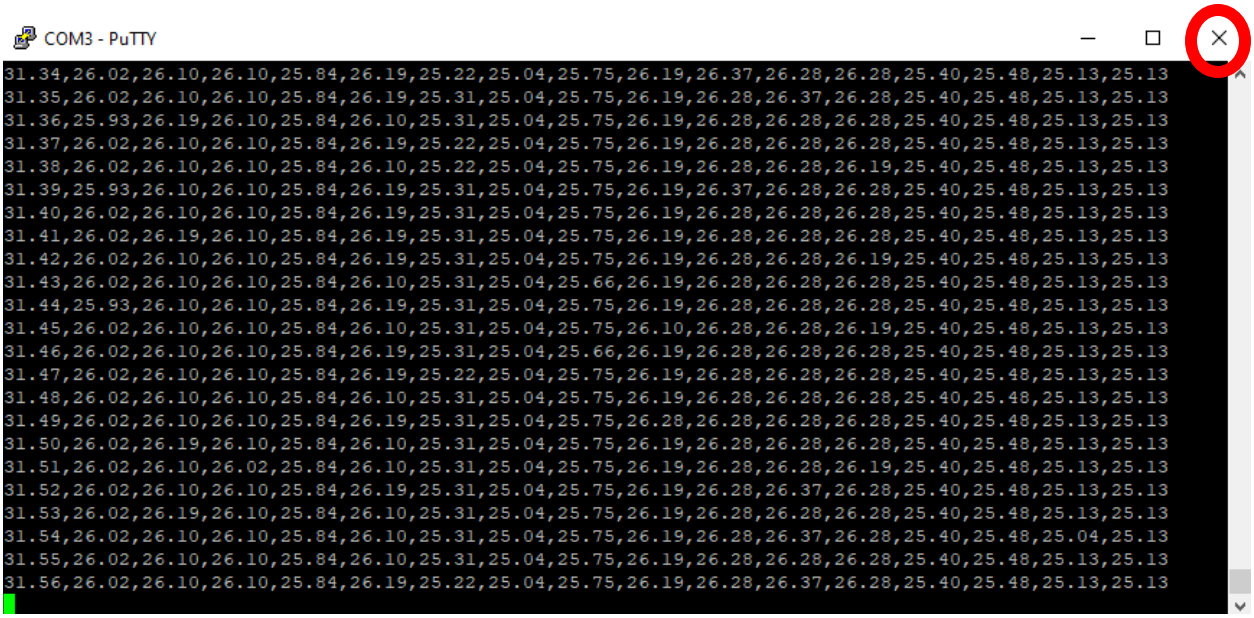


You may need to resize the window (drag it larger from the corner) for the rows to display correctly.

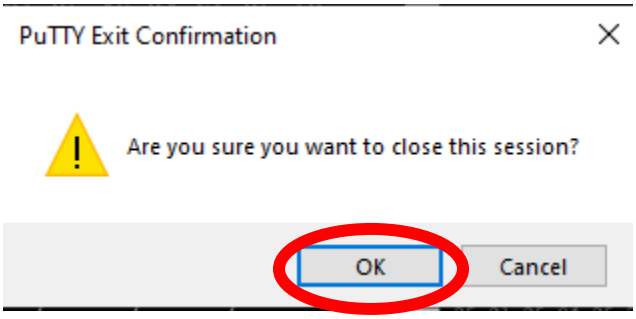
Now reset the Arduino by pressing the reset button to begin the test. The data collection will restart and the earlier data removed and discarded in data processing:



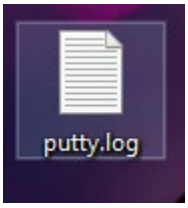
When the test is over, close the PuTTY monitor:



Then confirm selection:

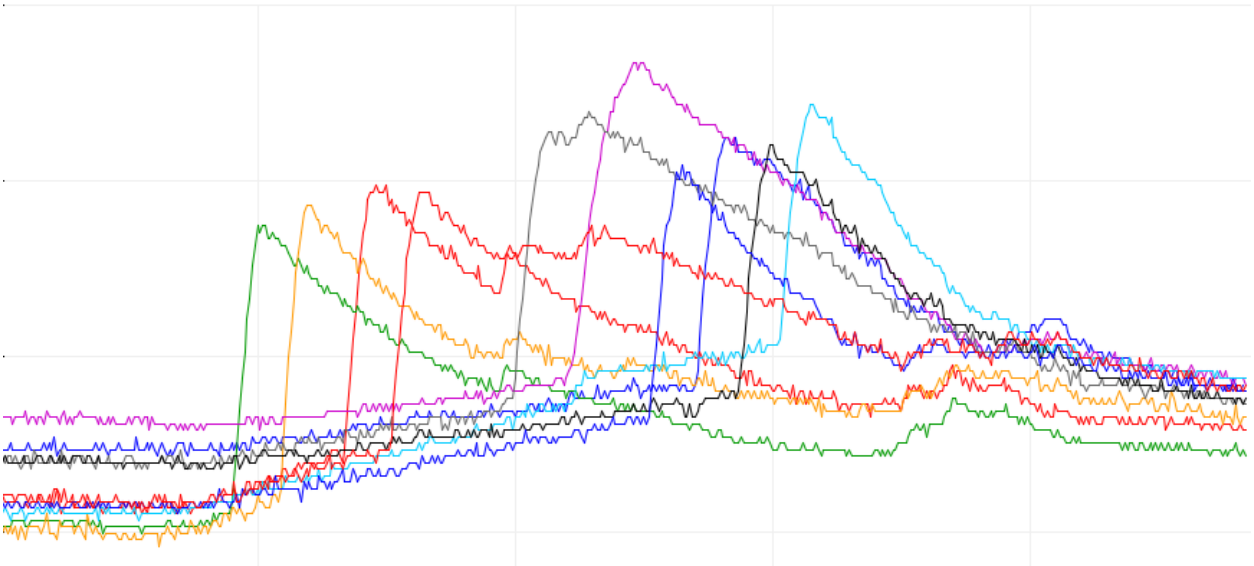


COLLECT YOUR DATA LOG FROM THE DESKTOP AND COPY TO DESIRED LOCATION. This will be overwritten or interfere with the next test if not done and could cause you to lose your testing data!



Time Savers

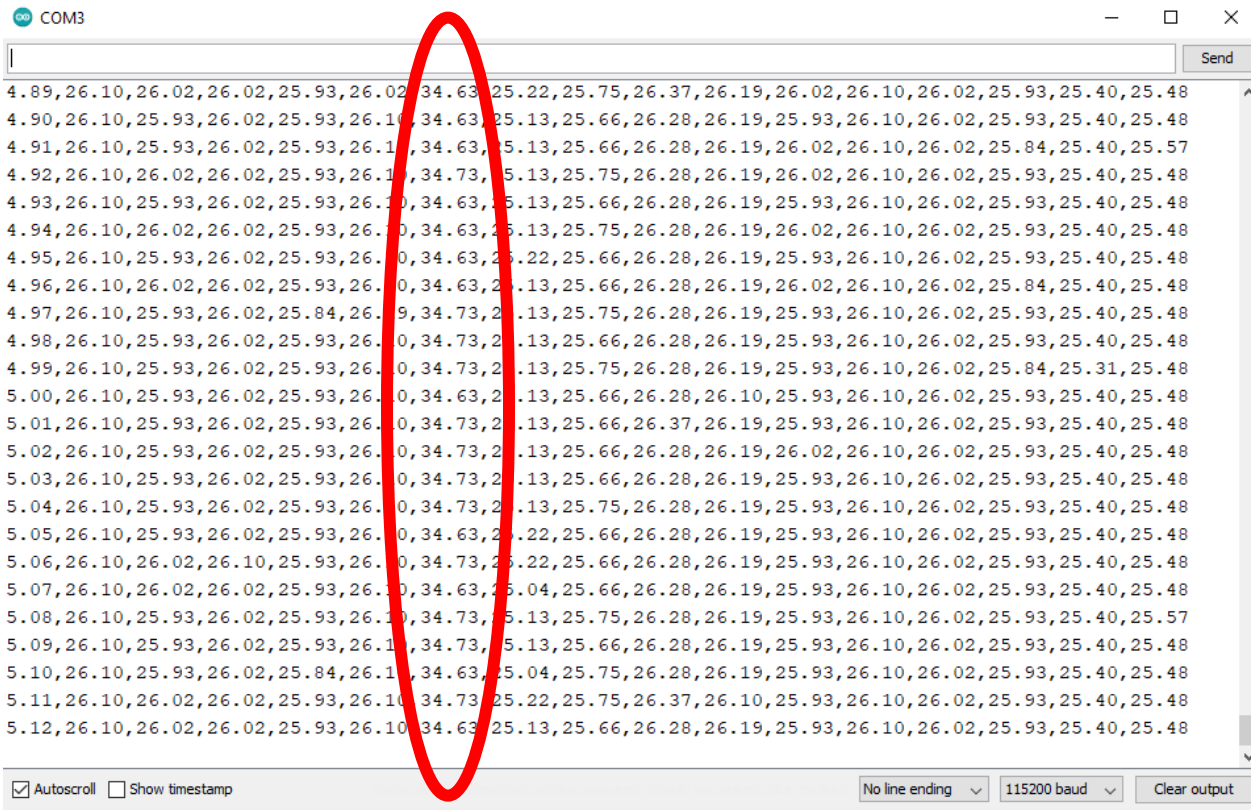
If the thermistors are not numbered you may pinch the ends and identify which it is by the temperature change from your finger as they are very responsive:



In the monitor you would have to watch for the increasing value and count the position from the left knowing the data output scheme, for example:

Time,T1,T2,T3,T4,T5,T6,T7,T8,T9,T10,T11,T12,T13,T14,T15

Here we can see the thermistor I am grabbing is warmer than the others and counting from the left we find it is T6. Remember that the first column is time, not a temperature.



Here is an early test against ice, a known temperature: