ANU PHYS2020 Thermodynamics Course Arduino Quick Start Guide

Tony.Yan @anu.edu.au

March 2024

This is a guide for setting up an Arduino for data logging.
Source code and this document are available at https://github.com/TonyXTYan/PHYS2020-Arduino-Quick-Start
The sample code performs basic data logging of provided components. We encourage you to extend and customise it to your project-specific needs.

Begin by completing the Connecting to Arduino, followed by the section about your specific sensors. Then you could proceed to Recording Data to SD Card.

Note: the sensor sections from this guide are TLDR versions from Last Minute Engineers, with some pin changes to integrate the SD card module. You're of course welcome to explore more.

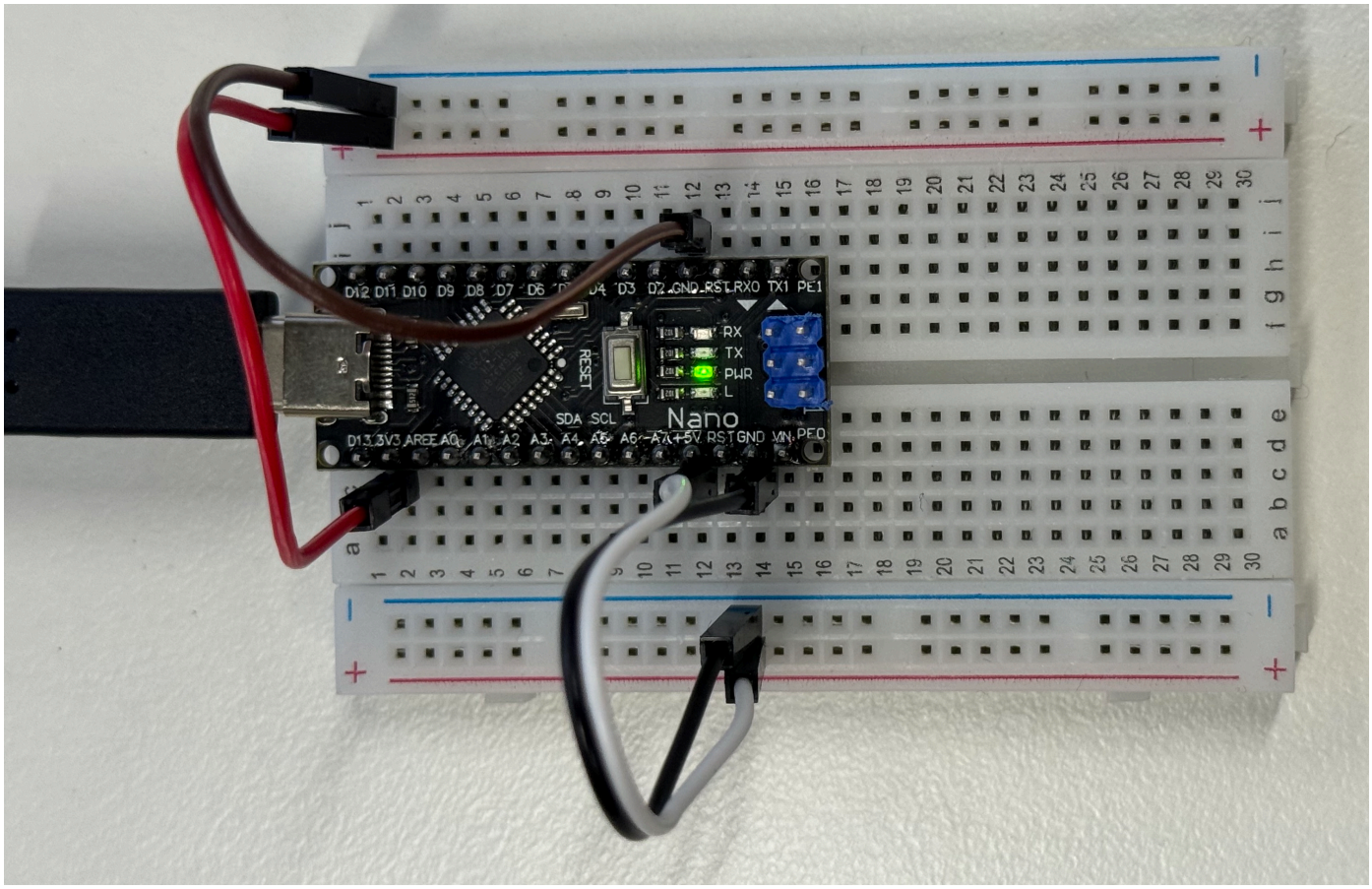
> *For reference: I'm using macOS 14.4, Apple Silicon, Arduino IDE 2.3.2, Arduino Nano and various modules from Adrian.*

Connecting to Arduino

First, download Arduino IDE (integrated development environment) from https://www.arduino.cc/en/software.

Once you have the Arduino IDE installed, we need to verify that the Arduino works by connecting it to your computer and the IDE.

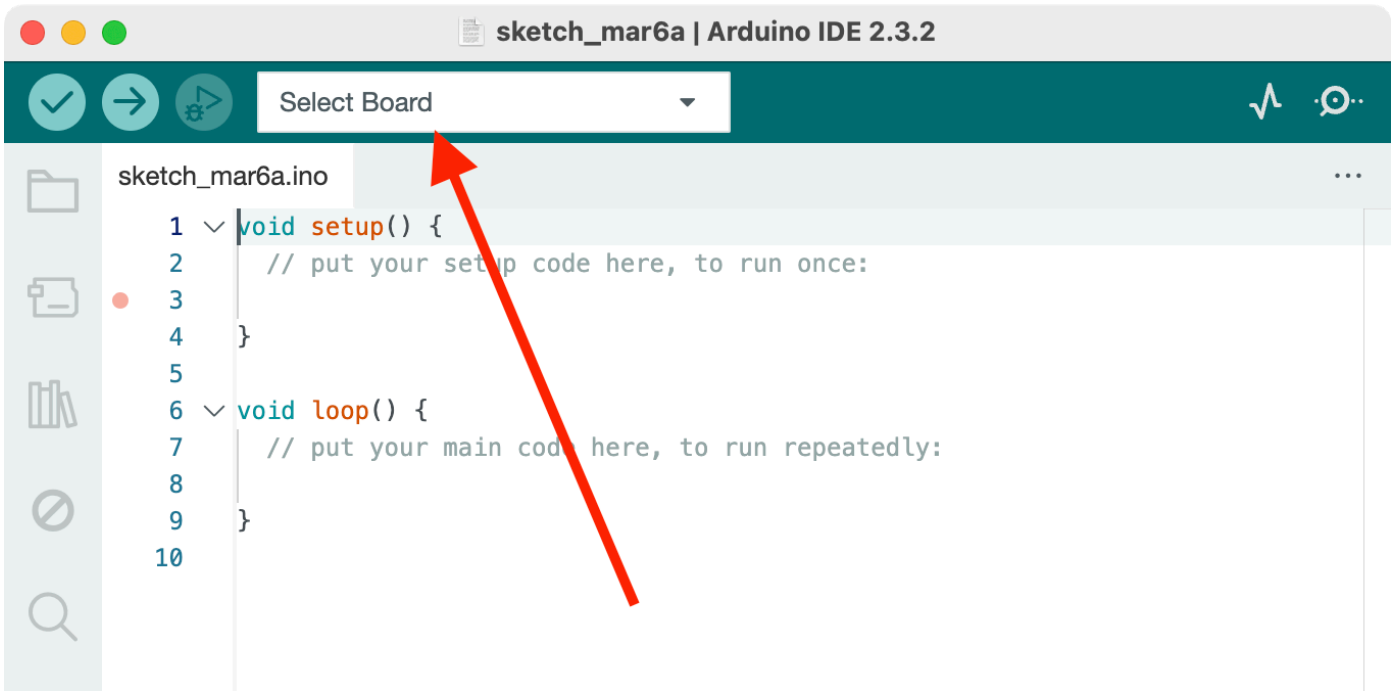
As illustrated in the figure below, I also strongly recommend mounting your Arduino Nano onto a breadboard and connecting the 5V and 3.3V to each power rail on the breadboard. This makes connecting various modules much simpler later.



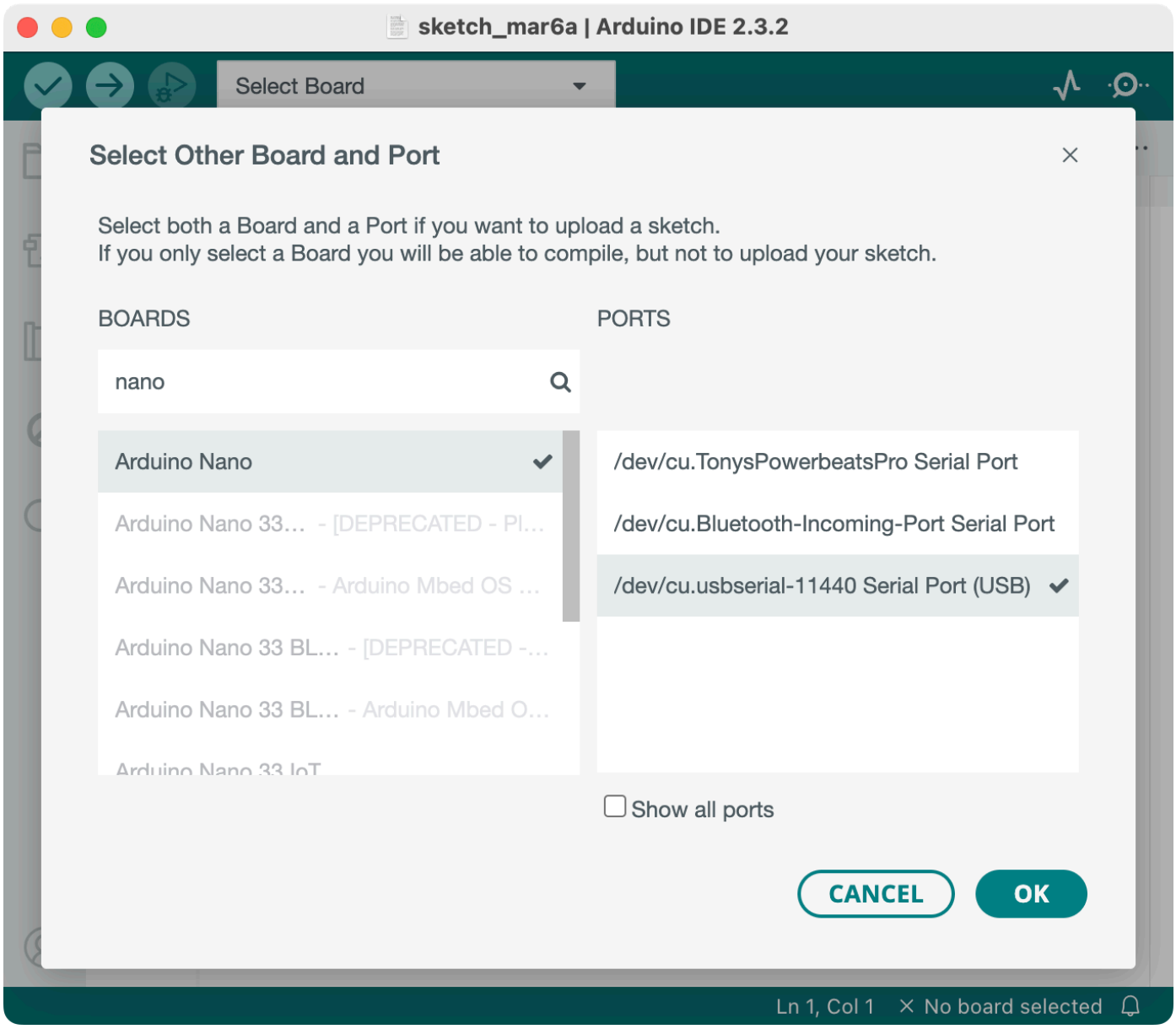
Our Arduino Nano uses a USB Type-C connector; you can connect it to your computer with any Type-C cable and Arduino can be powered from Type-C.

The Arduino IDE will preload a default BareMinimum script (`Menu -> File -> Examples -> Basic -> BareMinimum`) so you can test your connections to your Arduino.

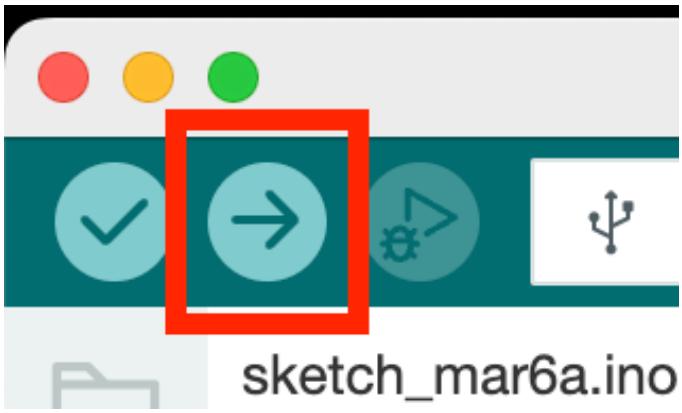
Configure the IDE to our specific Arduino by clicking `Select Board`



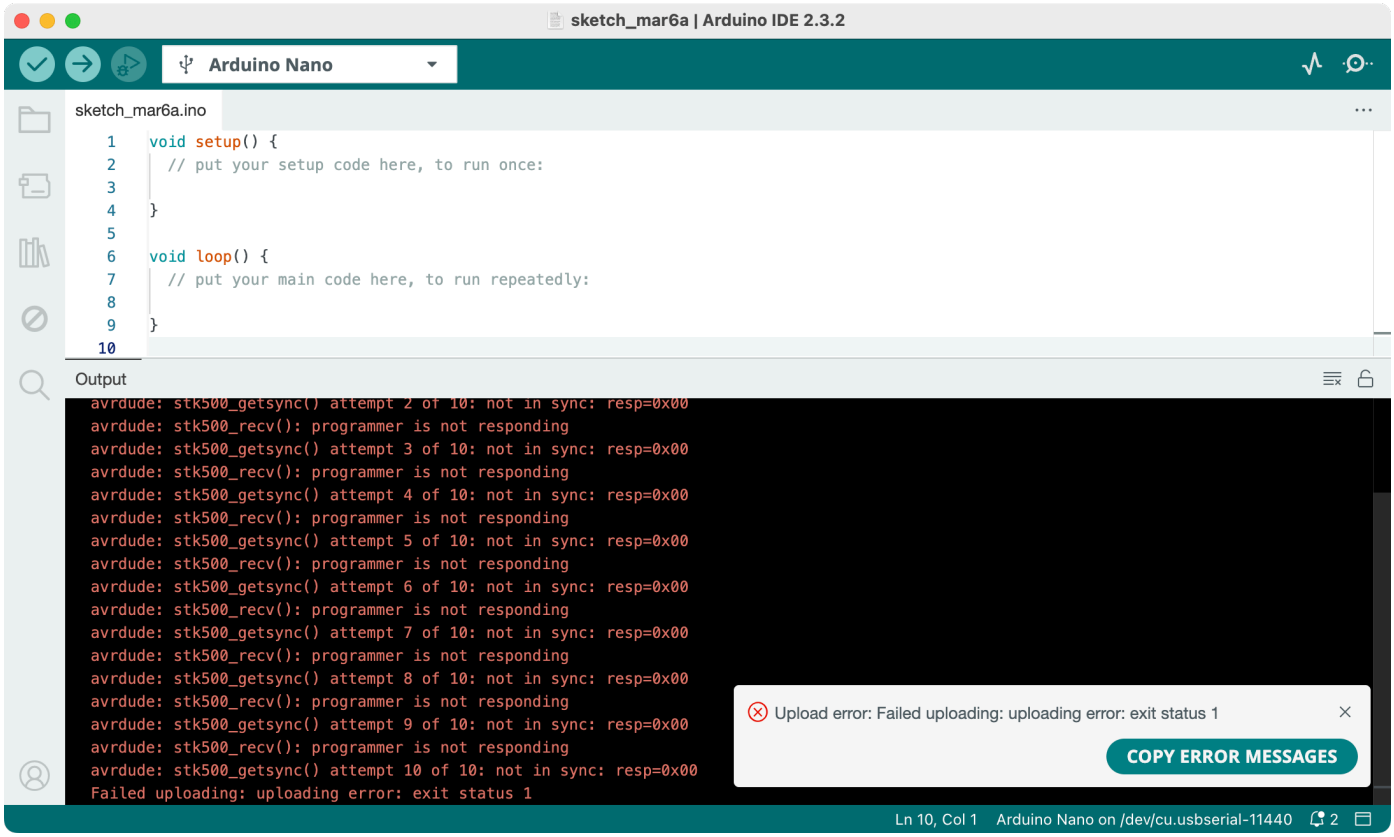
Select `Arduino Nano` and its `USB Serial Port`.



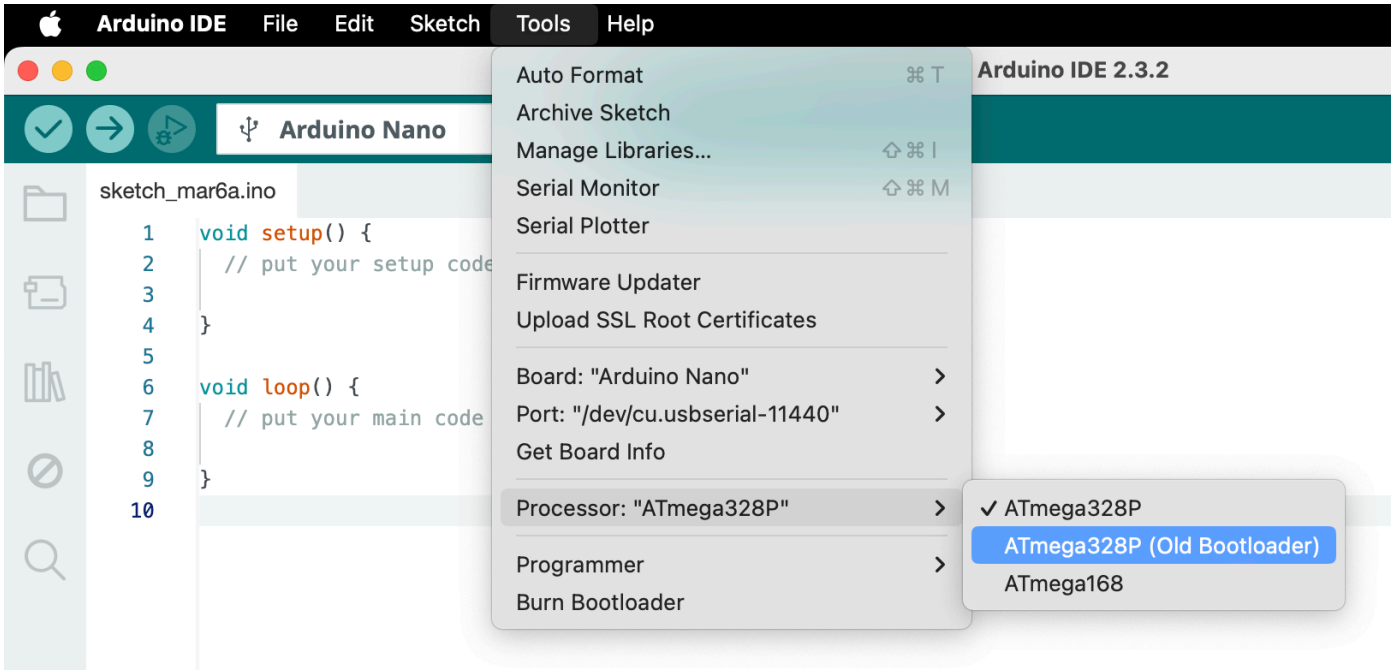
Then, you can try uploading the BareMinimum script onto your Arduino by pressing the `Upload` icon (or `Menu -> Sketch -> Upload` or `Cmd-U`).



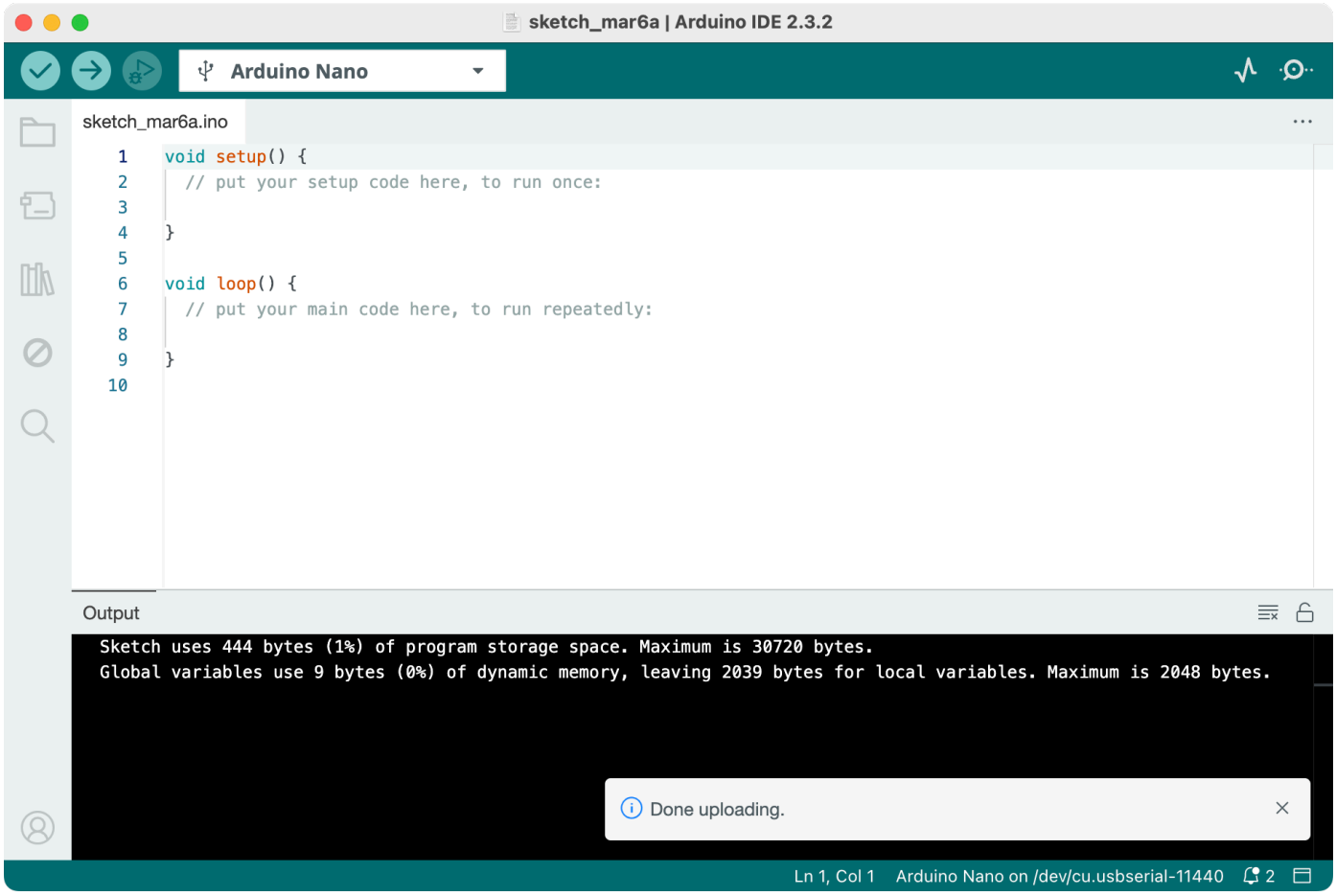
You might experience the following error saying the programmer is not in sync



In that case, you need to go to Menu -> Tools -> Processor and select: **ATmega328P (Old Bootloader)**



Then try upload again and it should upload without error.



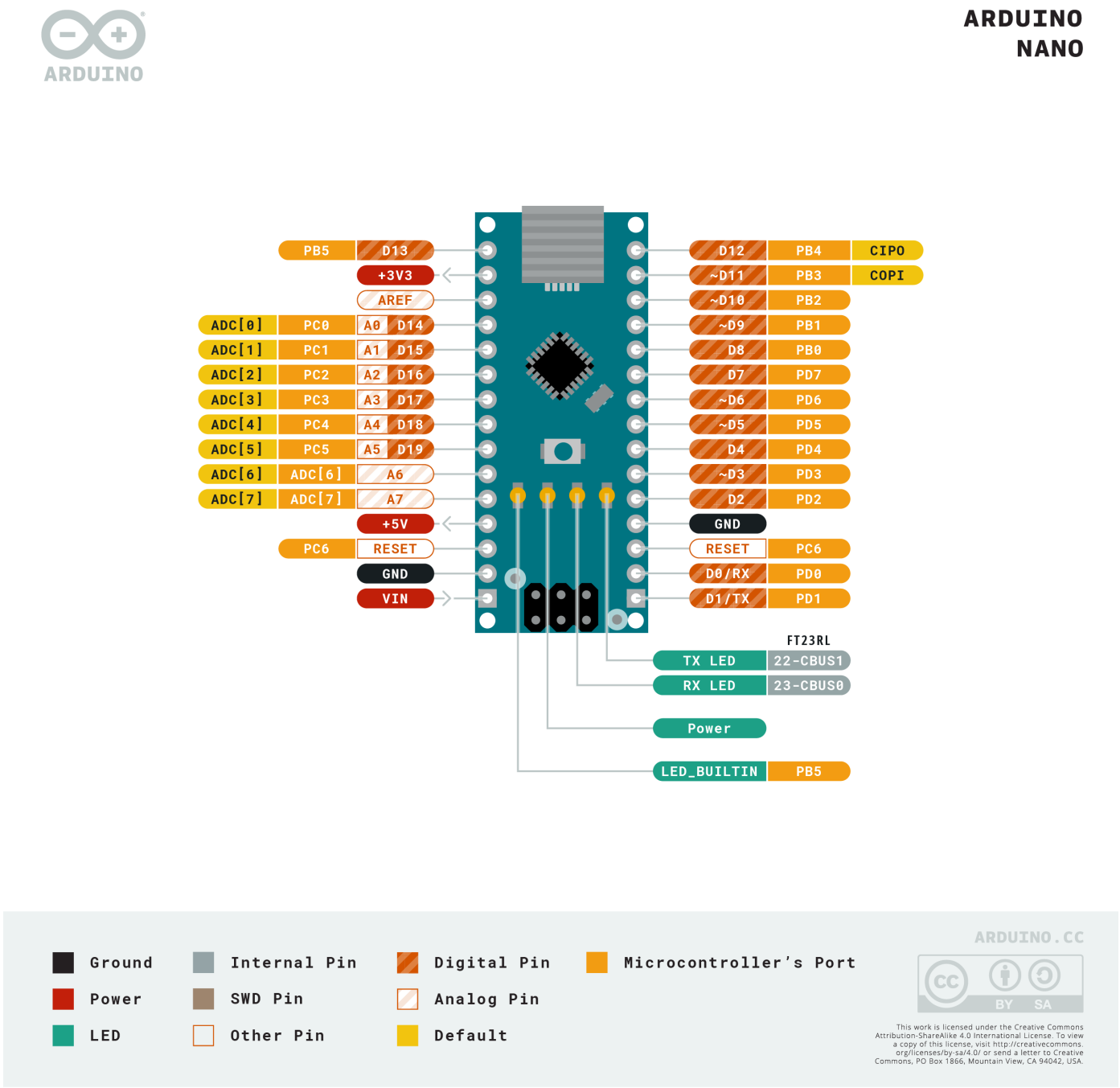
Other Common Issues/Troubleshooting:

- Try different USB cable
- Try different USB ports, avoid using adapters
- Try peer's Arduino

If there are other Arduino setting-up issues, feel free to contact us.

General Info and Tips about using Arduino in PHYS2020 Project

You might find the following Arduino Pin layout helpful (available at https://content.arduino.cc/assets/Pinout-NANO_latest.pdf).



- If you want to power the Arduino without connecting it to a computer, you can use either
  - 6-20V unregulated external power supply (pin 30 VIN), e.g. a 9V battery.
  - 5V regulated external power supply (pin 27 +5V). e.g. provided breadboard power supply, you might prefer this if you want to use some high current modules such as a display.
  - The power source is automatically selected to the highest voltage source.

- In general, disconnect all unused components from the Arduino pin. Some Arduino libraries will hardcode specific pins and send current through without warning.
- Be careful about which voltage rail you connect the modules to! Some modules come with a built-in voltage regulator, so you can connect them to either 3.3V or 5V, while some don't and have to be connected to 3.3V; otherwise, they could get permanently damaged.
- You should probably test each of your sensors and make sure they meet the manufacturer's claims before quoting their measurements scientifically, e.g., knowing their precision, measurement range, systematic error, and response time.
- ChatGPT is a fantastic place to get some starter code and Arduino Q&A.
- Arduino Official Documentation and Arduino Offical Forum are also good sources to look up for any issues.
- https://lastminuteengineers.com/electronics/arduino-projects/ provides extensive and detailed guides on Arduino modules. I strongly recommend reading them to understand the module you are using, especially if you want to go beyond the sample codes. (I've also stolen some figures from their website for educational purposes.)

MAX6675 (or MAX31855) Temperature Sensor

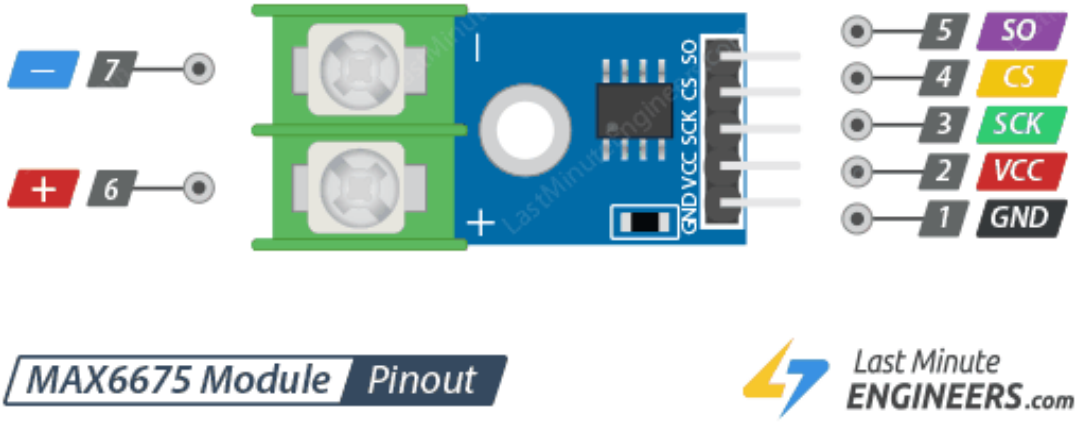
For a more detailed guide on using the MAX6675 module, please read https://lastminuteengineers.com/max6675-thermocouple-arduino-tutorial/

The MAX31855 is a newer version MAX6675, but they work basically the same.

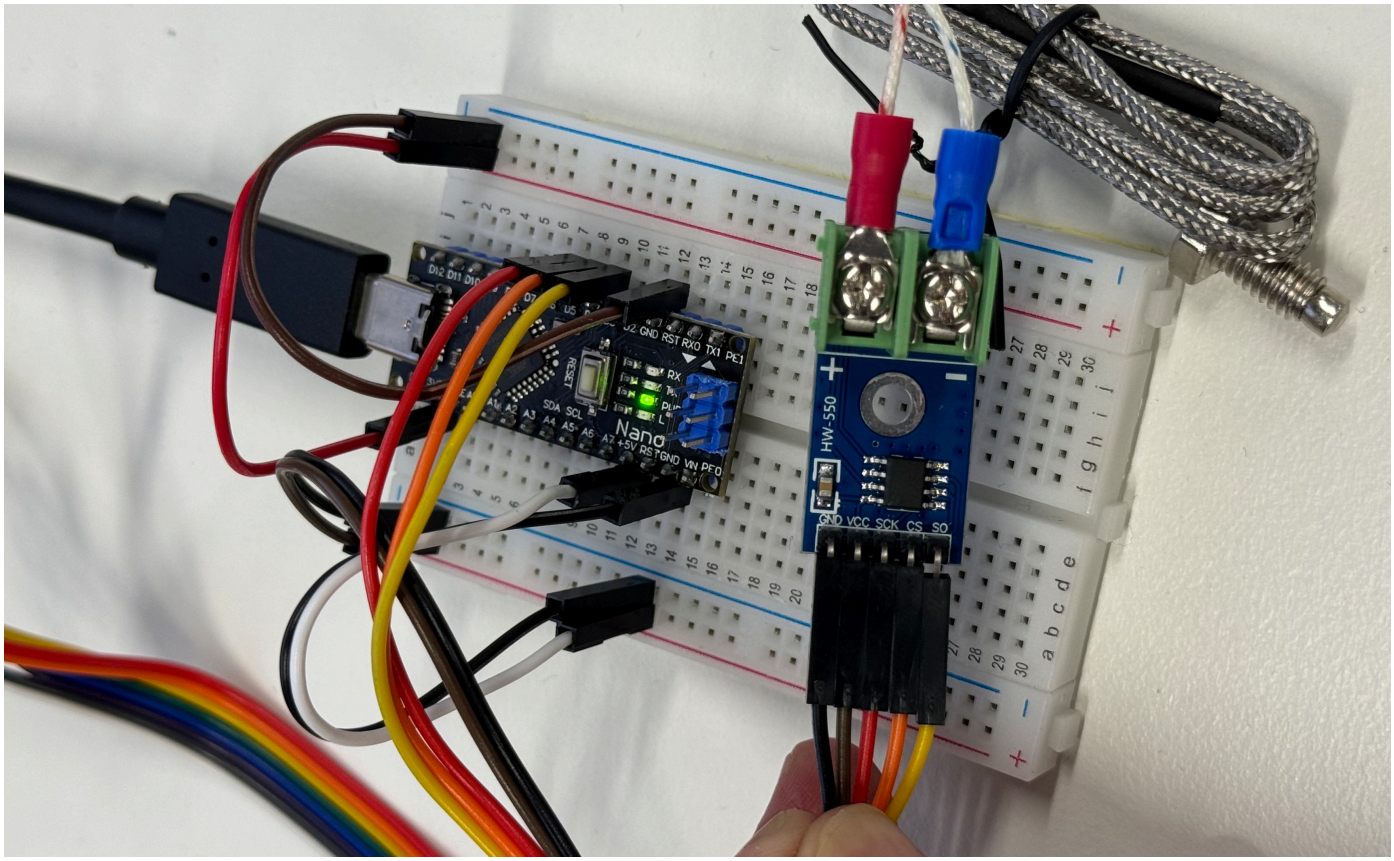
The kit includes

- Type-K thermocouple probe
  - M6 threads
  - measurement range 0-80ºC. TBC!
- MAX6675 breakout board,
  - 12-bit ADC (analog to digital converter),
  - temperature range 0-1024ºC with resolution of 0.25ºC (12-bit)
  - accuracy ±3ºC (however, in my experience, I got ±10ºC errors, so please check your probe is not faulty).

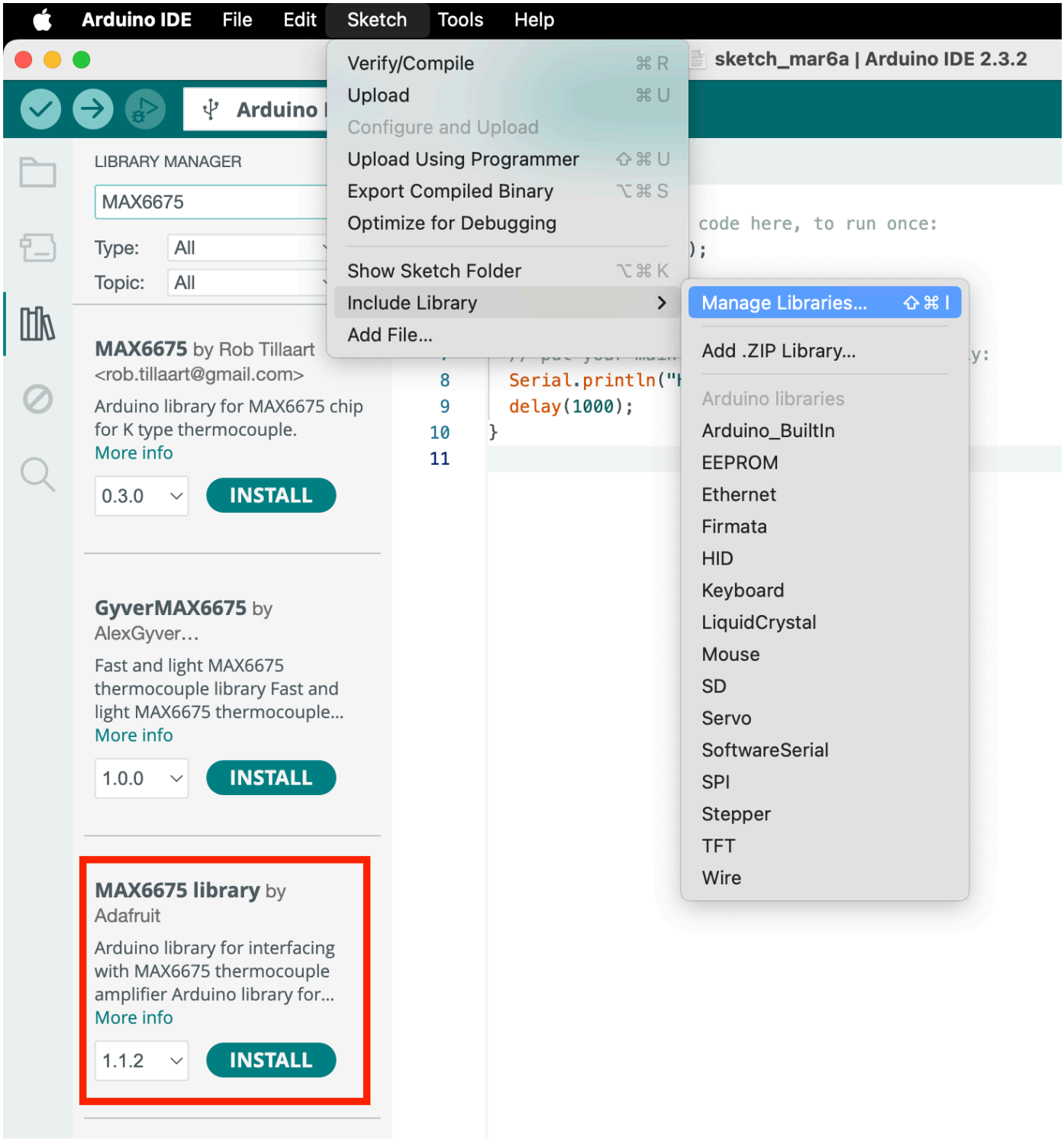
The following table shows the connections used in the sample code:



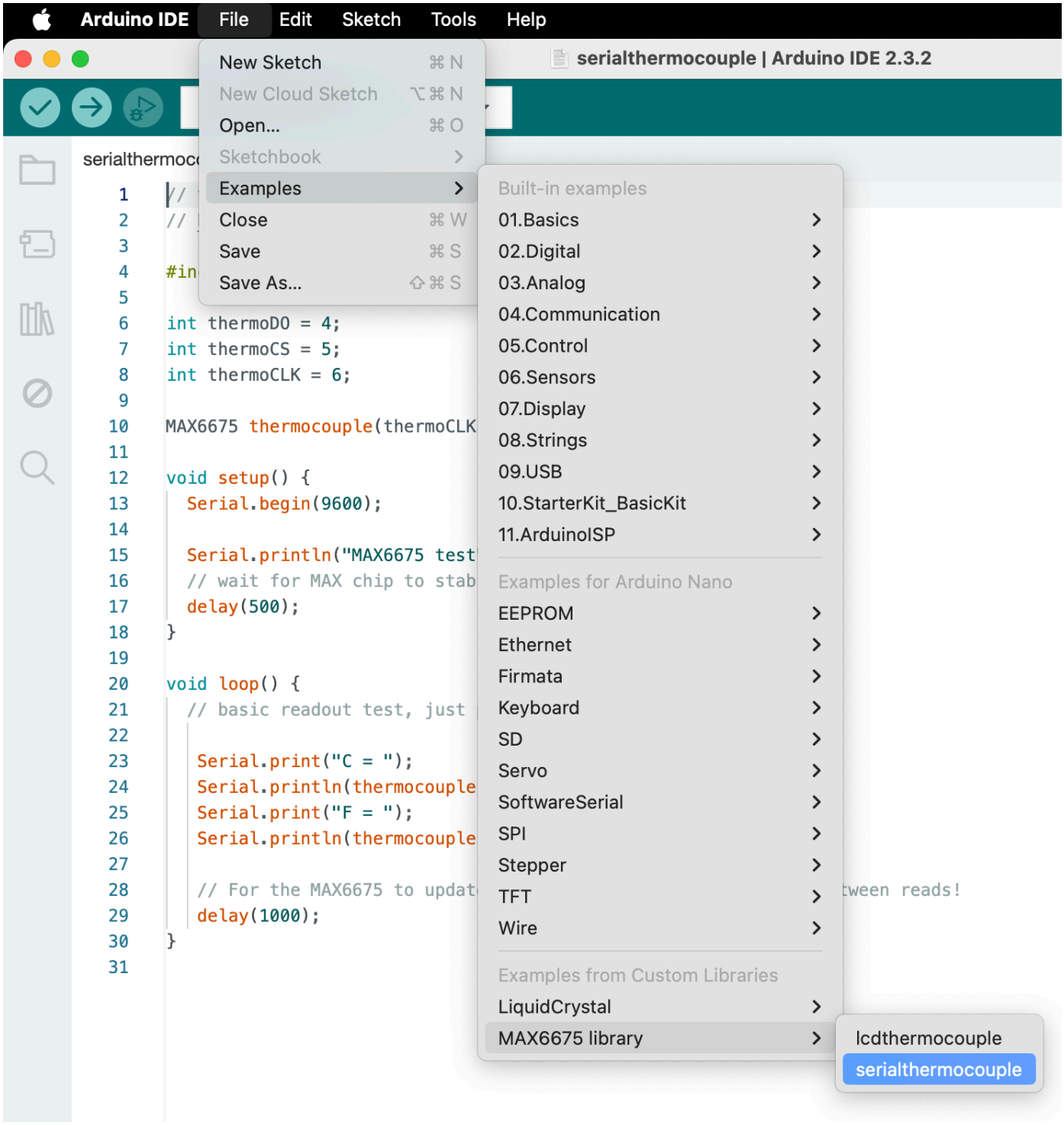
MAX6675	Arduino
VCC	3.3V or 5V
GND	GND
SCK (Serial Clock for synchronising data transmission)	D6
CS (Chip Select pin)	D5
SO (Serial data Out)	D4



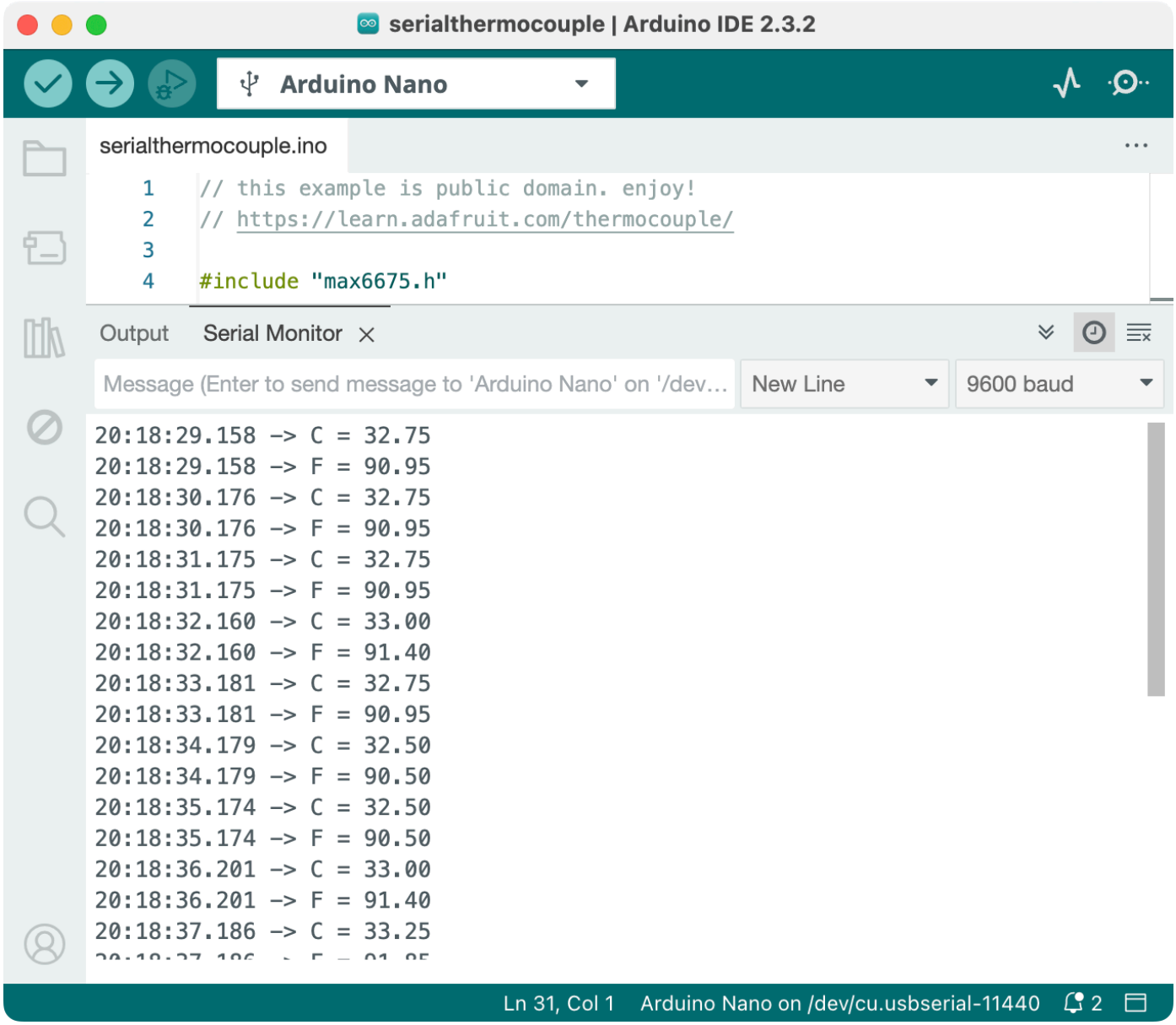
You also need to grab a code library to use the MAX6675 module. Go to `Menu -> Sketch -> Include Library -> Manage Libraries` (or `Shift-Cmd-I` or click on the Library icon in the left sidebar) and search for `MAX6675`.
I used the official library from Adafruit, but you're of course welcome to experiment with other ones.



Once the library is installed, open the provided sample code at `Menu -> File -> Examples -> MAX6675 library -> serialthermocouple`; there's also a copy of it named `MAX6675SerialLogger.ino` in this repository.



Try uploading their sample script and opening the serial monitor (`Menu -> Tools -> Serial Monitor` or `Shift-Cmd-M` or the top right Magnifying glass icon). You should see the thermocouple printing the temperatures onto the Serial Monitor. You could also show their timestamps by toggling the clock icon on the right.



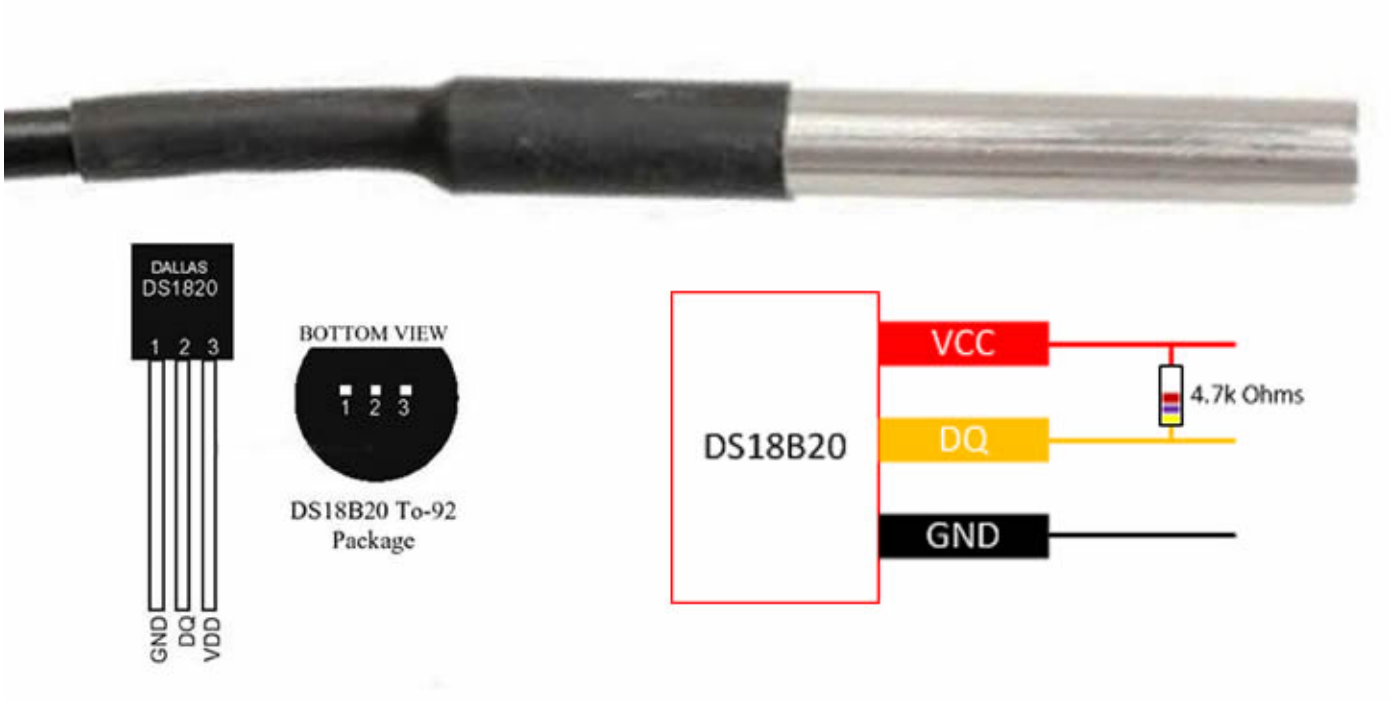
Tips & Tricks

- If you want to use multiple MAX6675 (or MAX31855), you can (should) let them share the same SO and CLK pins and only connect different CS pins to the Arduino.

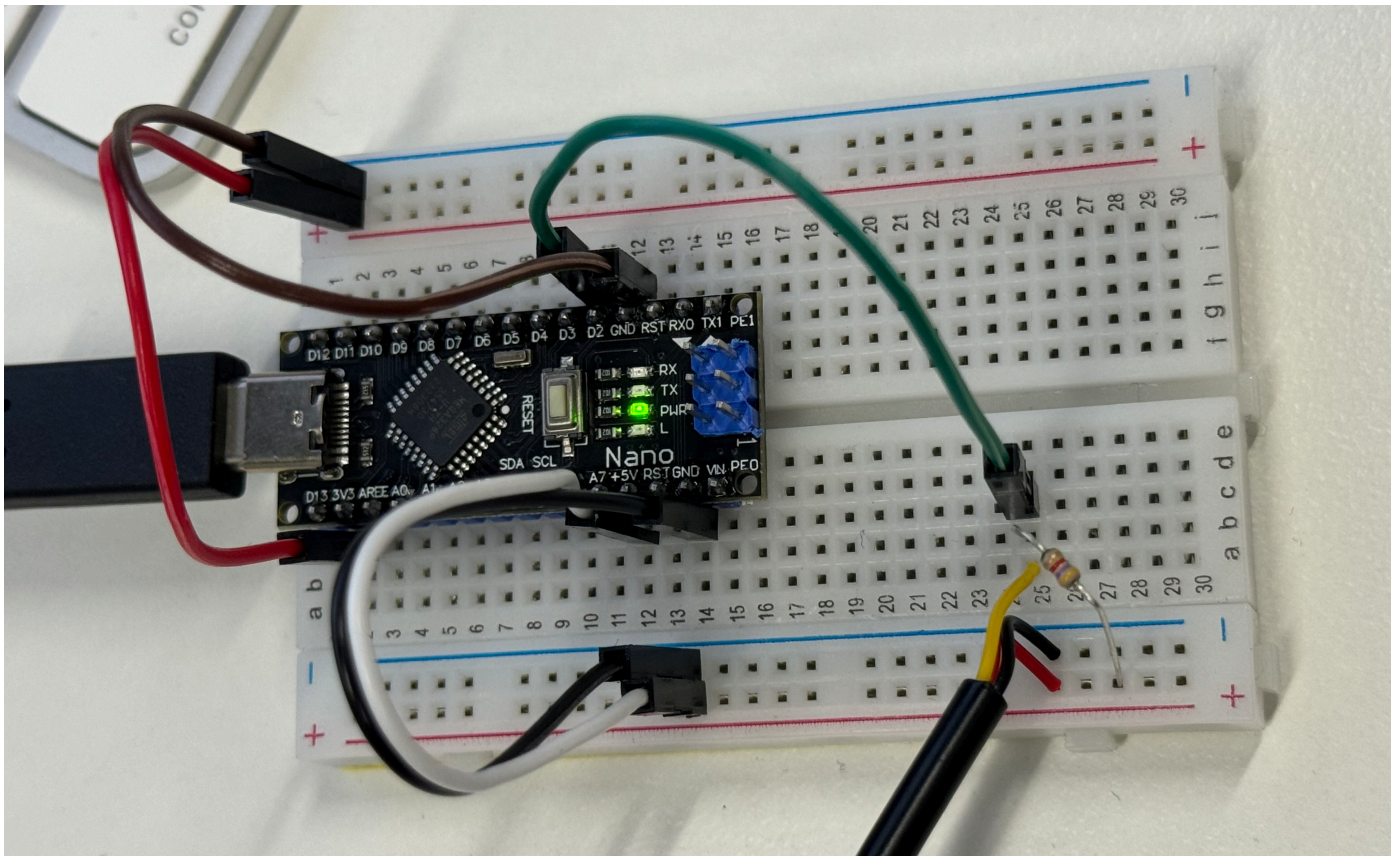
DS18B20 Temperature Sensor

For a more detailed guide on using the MAX6675 module, please read https://lastminuteengineers.com/ds18b20-arduino-tutorial/ .

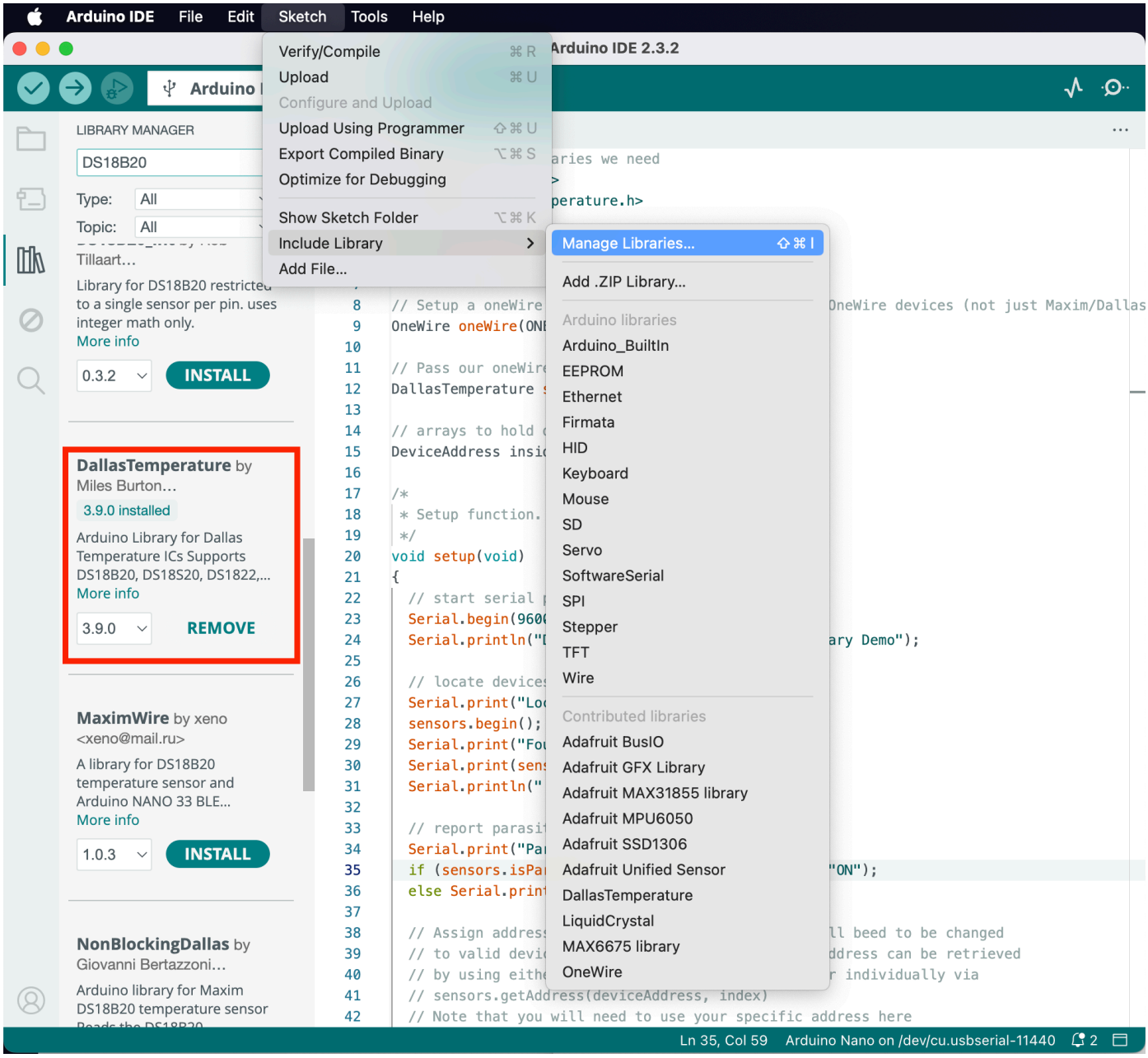
We provide the DS18B20 sensor that comes in the waterproof probe. You also need a 4.7kΩ resistor between VCC and DQ.



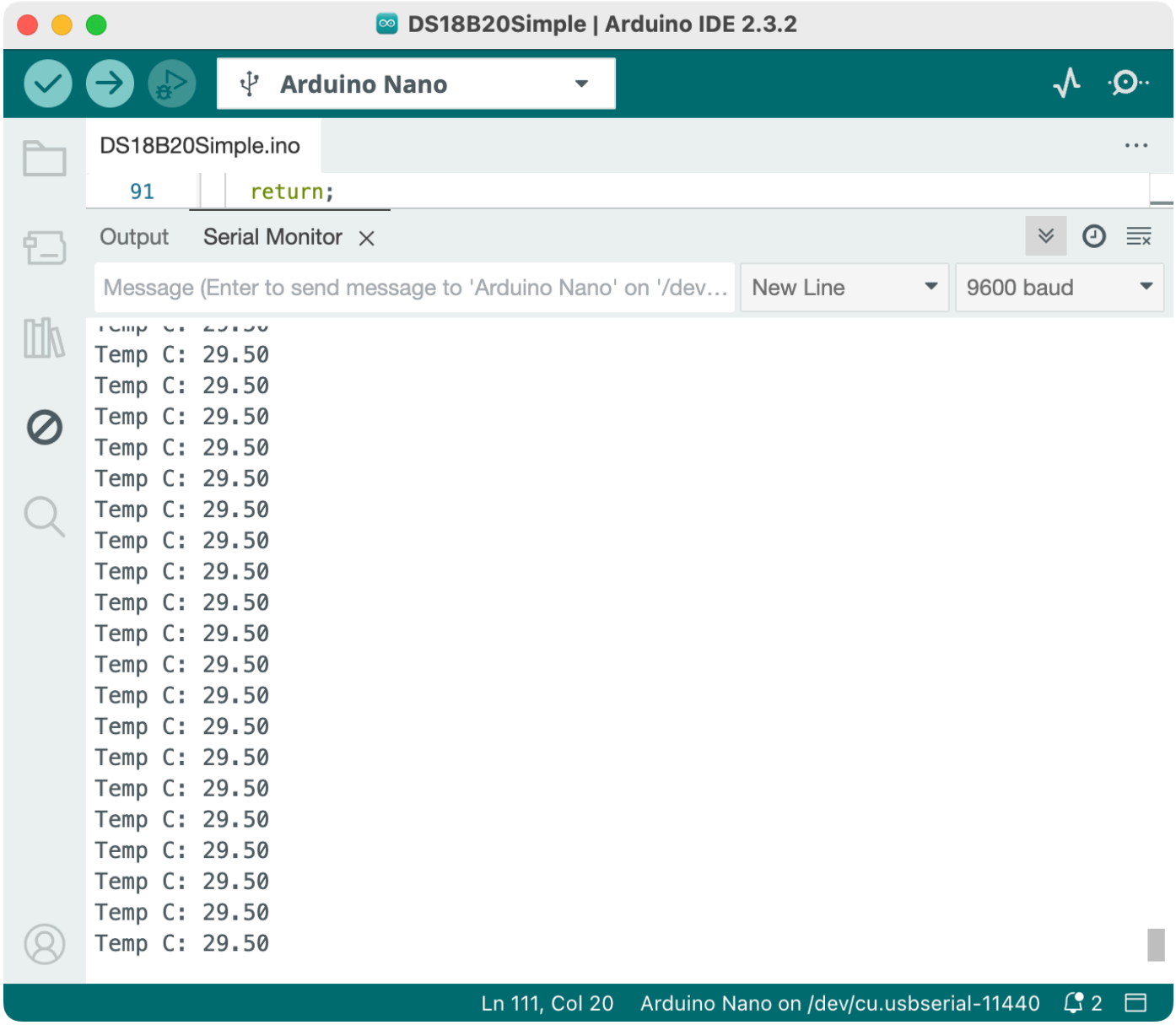
DS18B20	Arduino
VCC	3.3V or 5V
GND	GND
DQ (Data Queue)	D2



You also need to grab a code library to use the DS18B20 module. Go to `Menu -> Sketch -> Include Library -> Manage Libraries` (or `Shift-Cmd-I` or click on the Library icon in the left sidebar) and search for `DS18B20`.
I used the official library from `DallasTemperature`, but you're of course welcome to experiment with other ones.



Once the library is installed, open the provided sample code at `Menu -> File -> Examples -> DallasTemperatures -> Single`; there's also a copy of it named `DS18B20Simple.ino` in this repository.



Try uploading their sample script and opening the serial monitor (`Menu -> Tools -> Serial Monitor` or `Shift-Cmd-M` or the top right Magnifying glass icon). You should see the thermometer printing the temperatures onto the Serial Monitor. You could also show their timestamps by toggling the clock icon on the right.

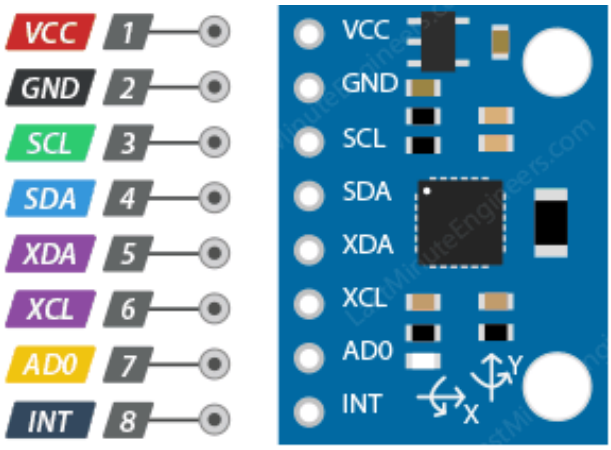
DS18B20 usage notes

- Since this sensor only requires a single-pin connection to Arduino, it is straightforward to add multiple sensors; for more info, see `Menu -> File -> Examples -> DallasTemperatures -> TwoPin_DS18B29`.
- The sensor wires could easily come loose on a breadboard; you should come up with a more permanent solution.

MPU6050 Accelerometer and Gyroscope

For a more detailed guide on using the MPU6050 module, please read https://lastminuteengineers.com/mpu6050-accel-gyro-arduino-tutorial/.

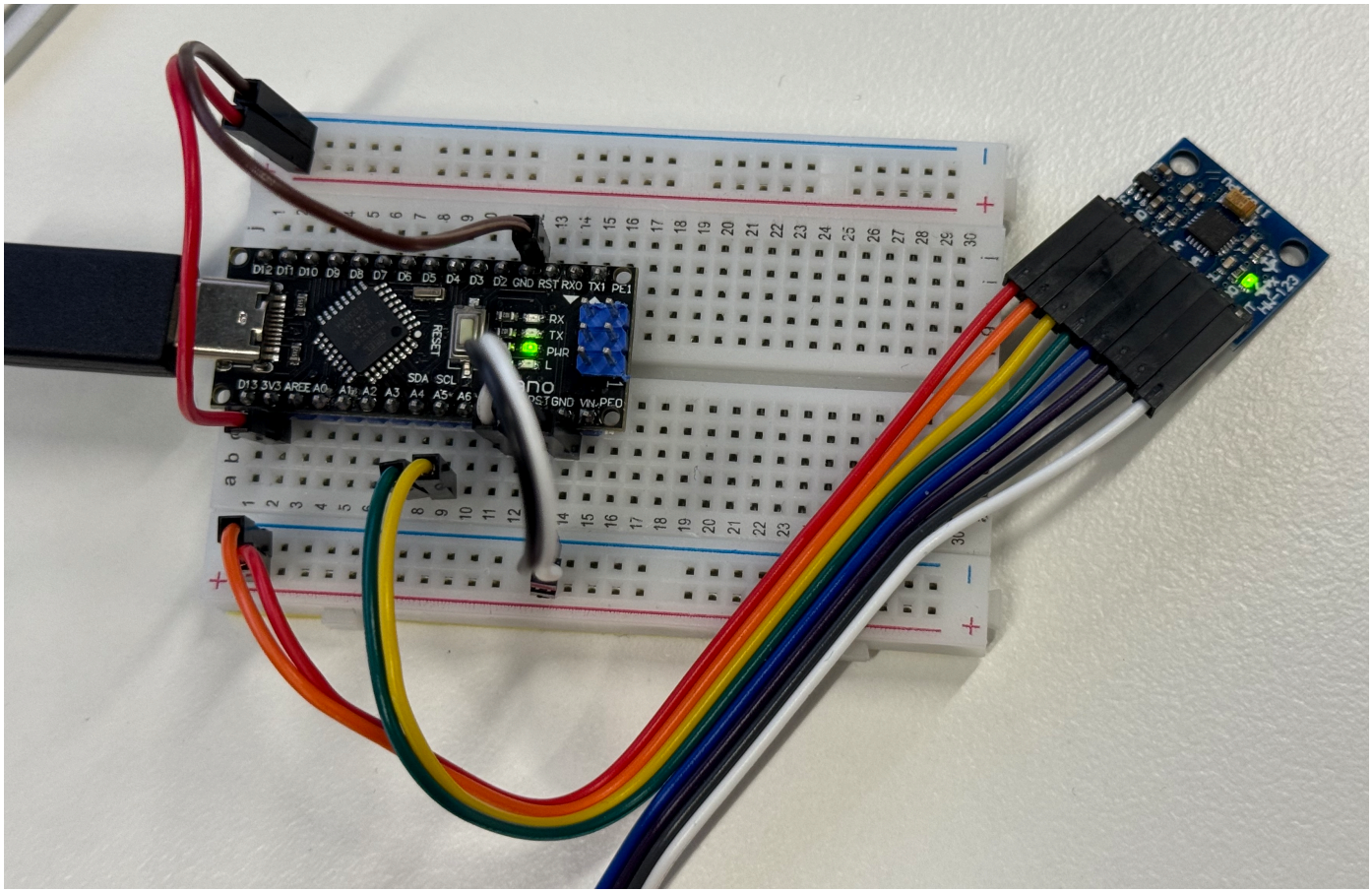
You might need soldering skills to make the pins.



MPU6050 Module Pinout

Last Minute ENGINEERS.com

MPU6050	Arduino
VCC	3.3V or 5V
GND	GND
SCL (Serial Clock Line)	A5 (SCL)
SDA (Serial Data Line)	A4 (SDA)
Other pins not connected.	



You also need to grab a code library to use the DS18B20 module, go to `Menu -> Sketch -> Include Library -> Manage Libraries` (or `Shift-Cmd-I` or click on the Library icon in the left sidebar), and search up `MPU6050`.

I used the official library of `Adafruit`, but you're of course welcome to experiment with other ones.

Once the library is installed, open the provided sample code at `Menu -> File -> Examples -> Adafruit MPU6050 -> basic readings`; there's also a copy of it named `MPU6050Basic.ino` in this repository.

Try uploading their sample script and opening the serial monitor (`Menu -> Tools -> Serial Monitor` or `Shift-Cmd-M` or the top right Magnifying glass icon). The sample codes use 115200 serial baud rate (instead of the default 9600), so you need to manually select the baud rate in the serial monitor. Then, you should see it printing the sensors' values onto the Serial Monitor. You could also show their timestamps by toggling the clock icon on the right.

You could also try the script in `Menu -> File -> Examples -> Adafruit MPU6050 -> plotter` and use the Serial Plotter (top right oscilloscope icon) to get a realtime plot from the accelerometer and gyroscope.

Pressure Sensor

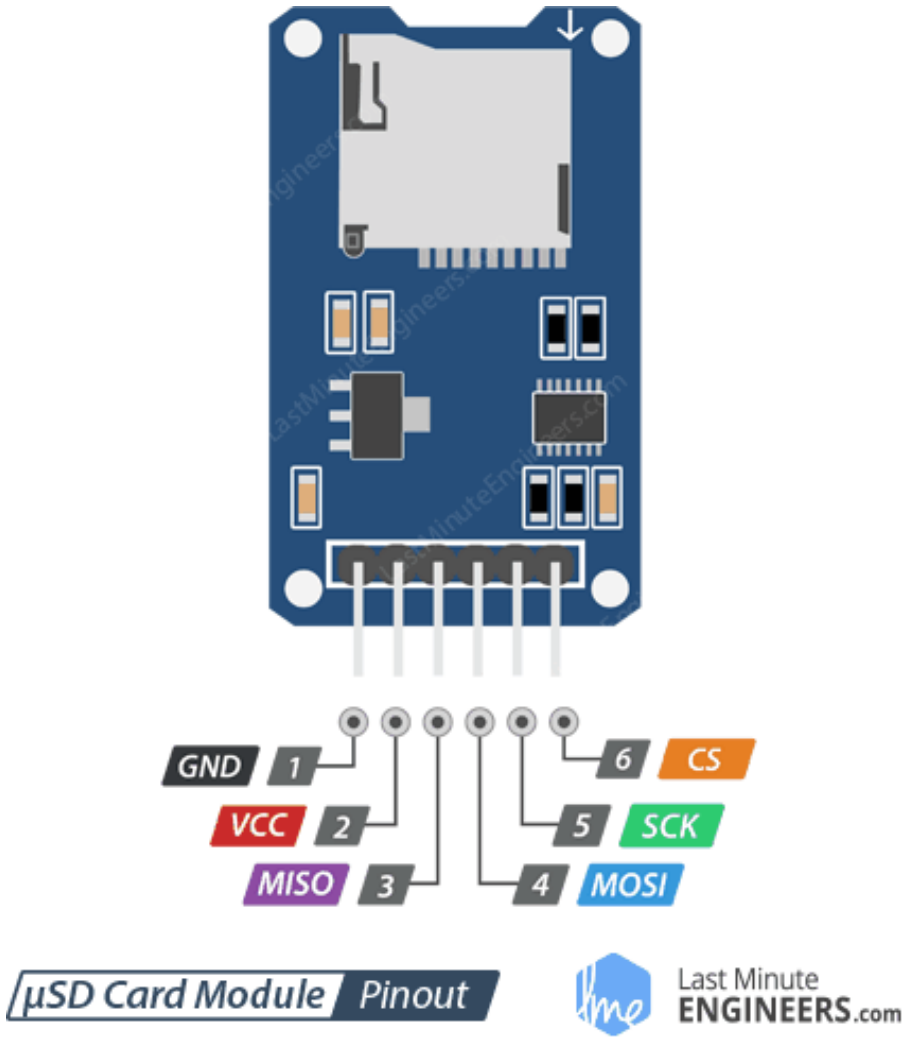
Depending on your sensor type, usually, they only need ground and 5V, then the sensor output 0-5V depending on their spec. Then any Arduino's analog pin can read that 0-5V in 1024 steps using `analogRead(pinNo)`.

- Thus, if for example your pressure sensor range is 0-5PSI, then 0-5Psi will correspond to 0-5V output, and your precision is 0.001 PSI.
- You would need to test your pressure sensors carefully and make sure they are up to spec.
- The pressure sensor in the template data logger (`SDCardDataLogger.ino`) is connected to analog pin A1.

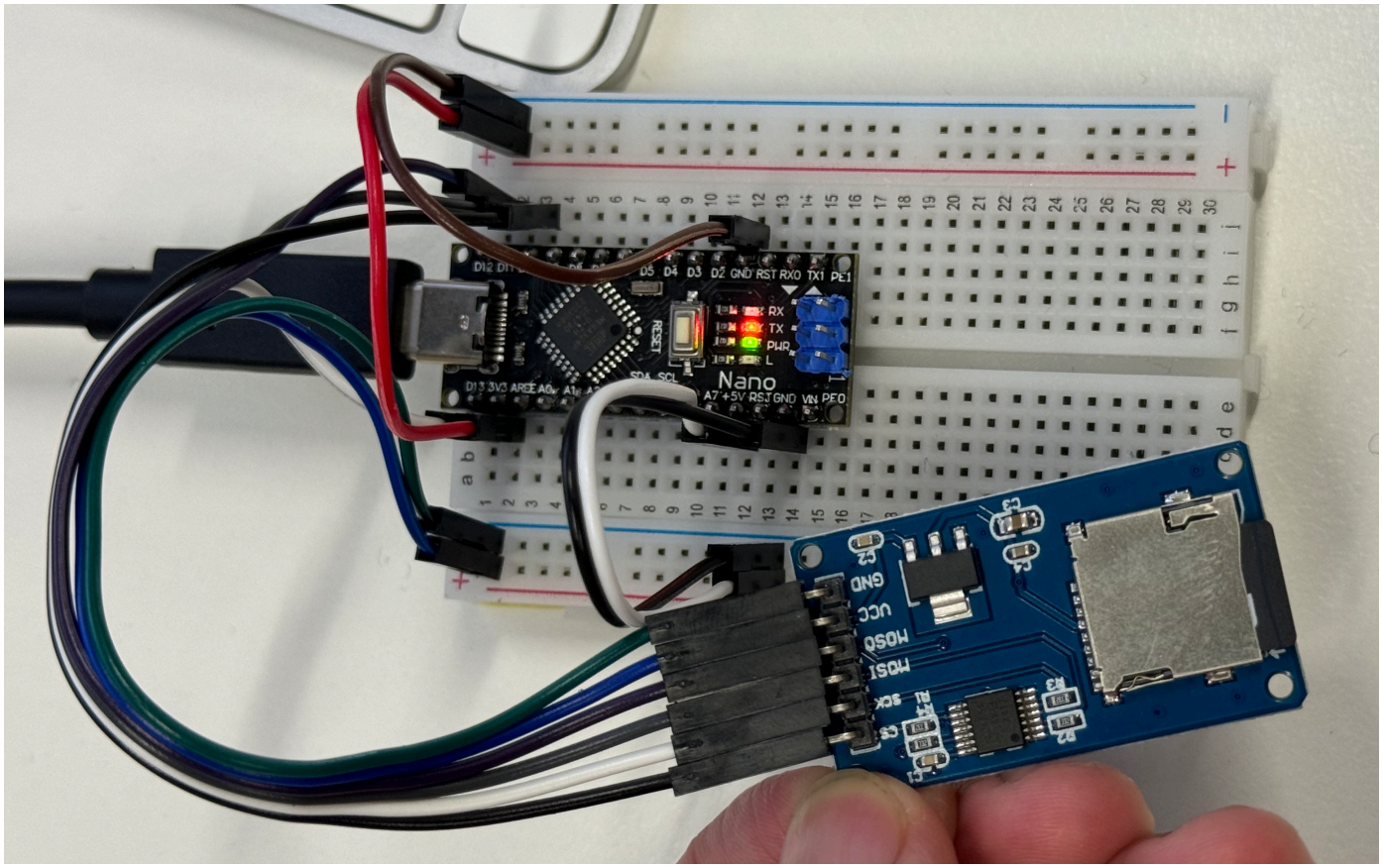
Recording Data to SD Card

For a more detailed guide on using the SD card module, please read https://lastminuteengineers.com/arduino-micro-sd-card-module-tutorial/

For the SD card module to work with the sample code (`SDCardDataLogger.ino`), connect the pins as follows,

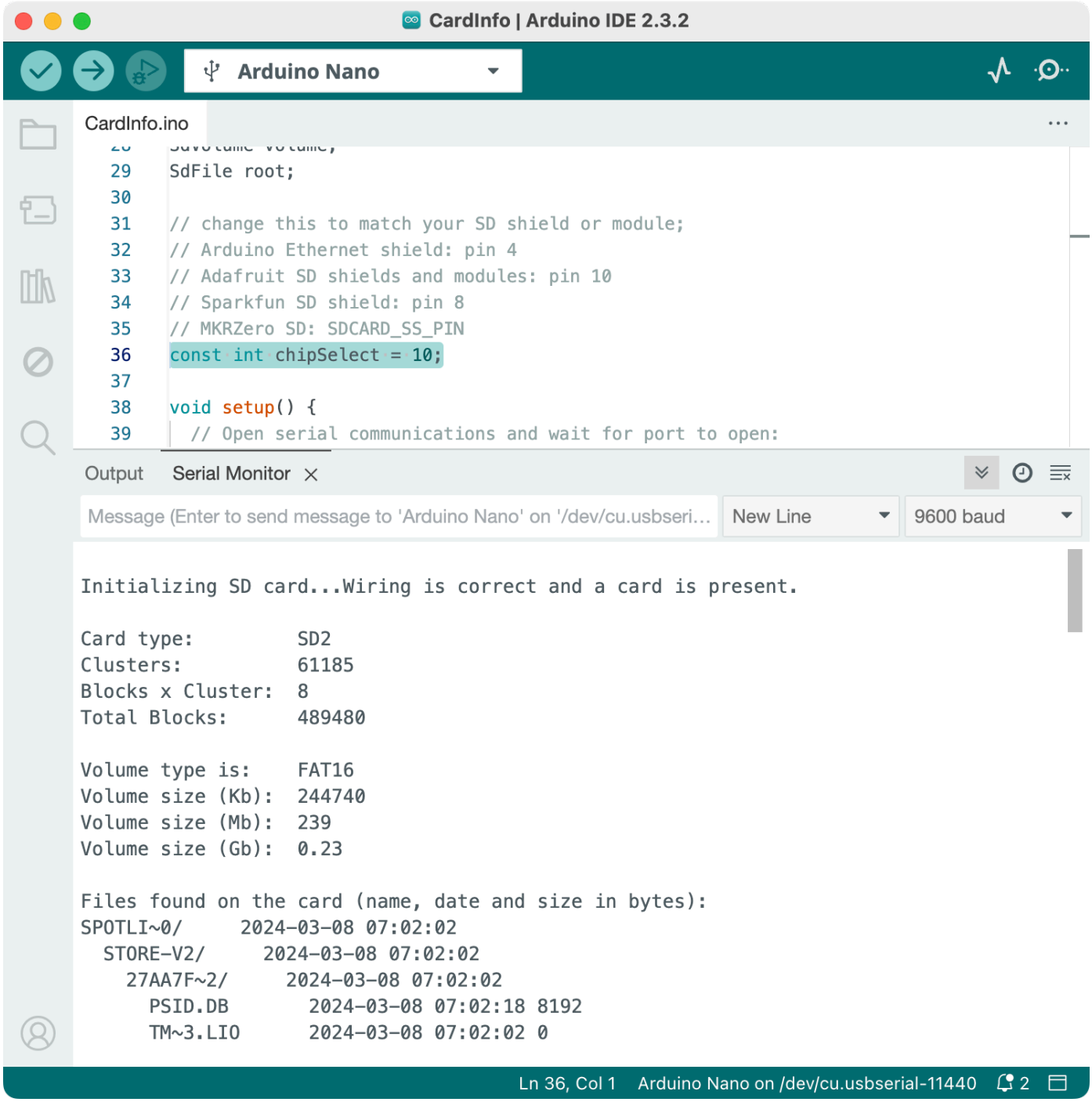


microSD Card Module	Arduino
VCC	3.3V or 5V
GND	GND
MISO (Serial Peripheral Interface Output)	12
MOSI (Serial Peripheral Interface Input)	11
SCK (Serial Clock)	13
CS (Chip Select)	10



Verify your SD card and reader module works

Open the built-in sample code at `Menu -> File -> Examples -> SD -> CardInfo`, and change the value of `chipSelect` at `line 36` to `10`, then upload the code onto Arduino, you should get the following info from the provided SD card (your's probably will not show any files, I have some files from macOS filesystem crap).



Using the sample code

Unless you installed all the sensors mentioned in the exact pins, the sample code `SDCardDataLogger.ino` would most likely throw some errors and complain about missing sensors.

If you do have all sensors installed, then `SDCardDataLogger.ino` would log data to both the SD card and Serial Monitor, and it could also log data to the SD card without connecting it to a computer. The Serial monitor output would look something like this:

```
MPU6050 Initialised.
Initialising SD card...done.
Logging to: log20.csv
Tms,423,Acc,-3.10,3.44,-8.87,Rot,-0.08,-0.01,0.02,V1,0.01,TC,0.00,TS,25.00,
Tms,557,Acc,-3.09,3.45,-8.88,Rot,-0.08,-0.02,0.02,V1,0.01,TC,0.00,TS,25.00,
Tms,690,Acc,-3.11,3.47,-8.85,Rot,-0.08,-0.02,0.02,V1,0.00,TC,0.00,TS,25.00,
...
...
```

The first three lines are just checking setup; from the fourth line onwards, they show the exact data logged to the SD card.

- `Tms` is milliseconds since Arduino boot.
- `Acc` shows x,y,z acceleration (m/s^2) and `Rot` shows xyz rotation (rad/s) using MPU6050
- `V1` is the voltage from analog pin A1 for the pressure sensor. Depending on your pressure gauge spec, you need to convert this value to pressure.
- `TC` is MAX6675 thermocouple reading in C. (It's showing zero cuz I might have accidentally blown mine)
- `TS` is DS18B20 temperature sensor in C.

The data are logged in CSV (comma separated value) format which can be easily read by Mathematica, Python, Excel or your preferred stats tool.

You should comment out (`Cmd-/`) any code about the sensors that you are not using.

- Codes about MAX6675:

```
// Line 13-14
// MAX6675 Thermocouple
#include "max6675.h"
MAX6675 thermocouple(6, 5, 4); // SCK - pin 6, CS - pin 5, SO - pin 4.
...
// Line 100-101
// MAX6675
logToSerialAndSD("TC");
logToSerialAndSD(thermocouple.readCelsius());
```

- Codes about DS18B20

```
// Line 18-21
// DS18B20 Temperature Sensor
#include <OneWire.h>
#include <DallasTemperature.h>
OneWire oneWire(2);         // setup a oneWire with DS18B20 DQ connected to pin 2
DallasTemperature tempSensor(&oneWire); // pass oneWire to DallasTemperature library
...
// Line 104-106
// DS18B20
tempSensor.requestTemperatures(); // send the command to get temperatures
logToSerialAndSD("TS");
logToSerialAndSD(tempSensor.getTempCByIndex(0));
```

- Codes about MPU6050

```
// Line 25
// MPU6050 Accelerometer and Gyroscope
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>
Adafruit_MPU6050 mpu;
...
// Lines 83-92
// MPU6050
sensors_event_t a, g, temp; // Get new sensor events with the readings
mpu.getEvent(&a, &g, &temp);
logToSerialAndSD("Acc");
logToSerialAndSD(a.acceleration.x);
logToSerialAndSD(a.acceleration.y);
logToSerialAndSD(a.acceleration.z);
logToSerialAndSD("Rot");
logToSerialAndSD(g.gyro.x);
logToSerialAndSD(g.gyro.y);
logToSerialAndSD(g.gyro.z);
```

- Codes about pressure sensor

```
// Lines 96-97
// Pressure Sensor at Analog pin A1
logToSerialAndSD("V1");
logToSerialAndSD(analogRead(A1)*5.0/1024.0);
```

Other things to consider

- `SDCardDataLogger.ino` will dump data onto SD immediately after the Arduino is powered, and the only way to stop is to power down the Arduino. Consider adding a push button to start and stop recording.

- Arduino doesn't have a built-in real-time clock, only an internal timer to count the time from boot. So the only labels for which recording are their file names. You should think about how to organise your data files.
- My quick test shows this logs data at about 40kB/min, which a 256MB SD card will take a few days to fill up.
- Arduino Nano has a built-in reset button. If you press it once, Arduino will restart and automatically run `SDCardDataLogger.ino`, creating a new file and recording.