Sensing for Science

Level 6

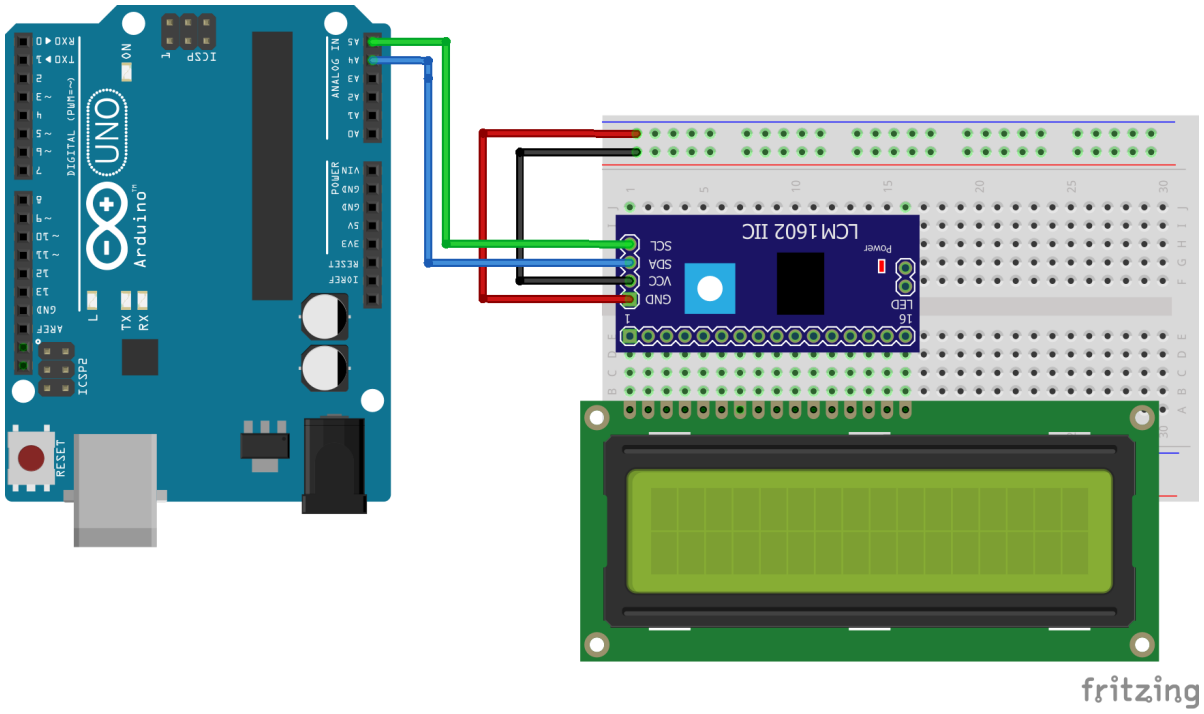
This material is based upon work supported in part by the National Science Foundation EPSCoR Cooperative Agreement OIA-1757351. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

The Final Project

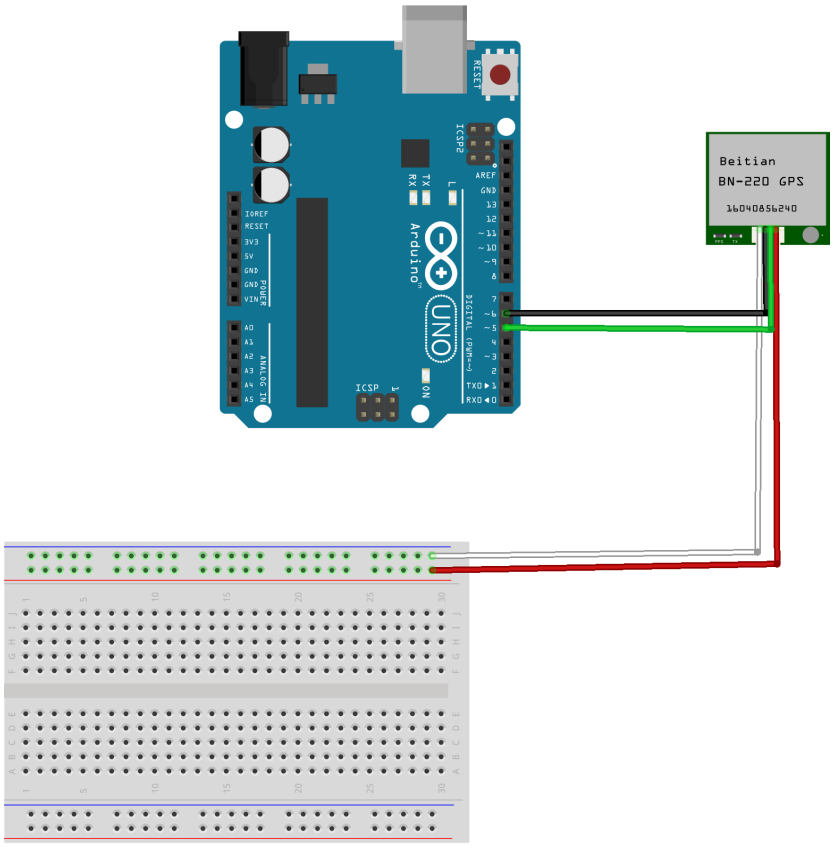
In this level we are going to create a circuit that will test the total dissolved solids (TDS) in water using the GravityTDS sensor from DFRobot. It will also collect the GPS location and save the data to a SD card. We will have a LCD display too. There is a lot going on here with hardware and software, so take time to tape wires and do some organizing. It will help with troubleshooting when out in the field. There are many ways this circuit can be constructed. I will show you the way I did it, but you can build it in the way you think is good.

Here is how I wired up this circuit. I will do a schematic for each device connected to the Arduino board. Take note, I will use the power rails on the breadboard for all 5v and GND connections.

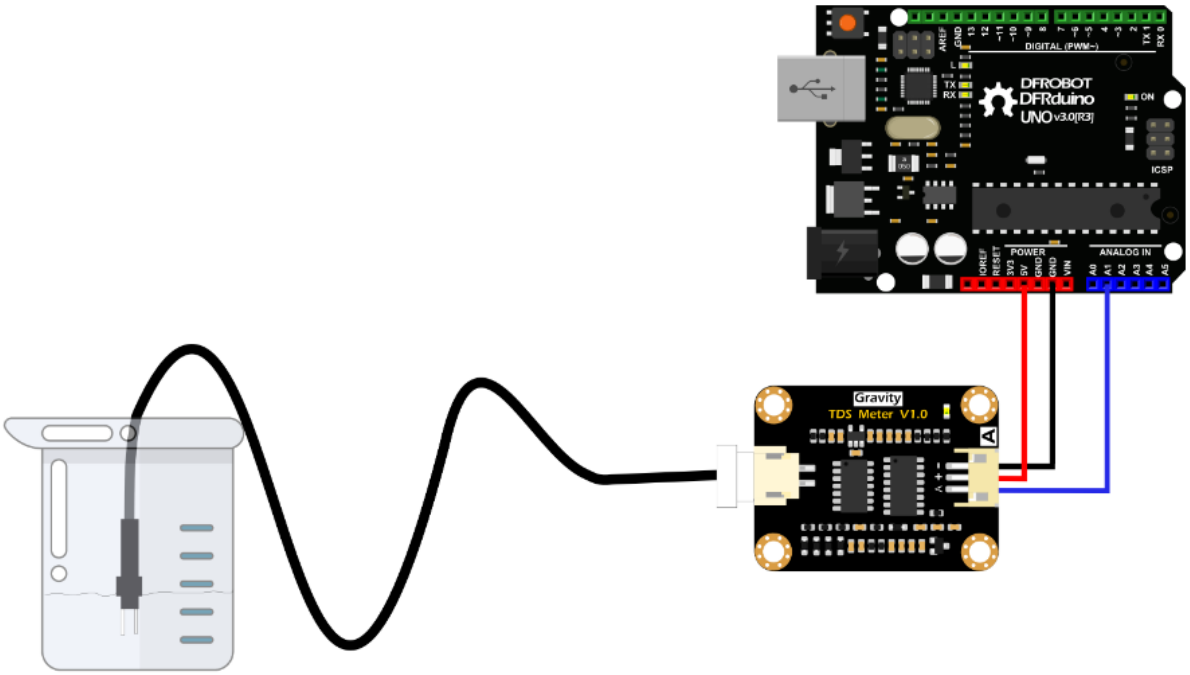
LCD hook with I2C adaptor.



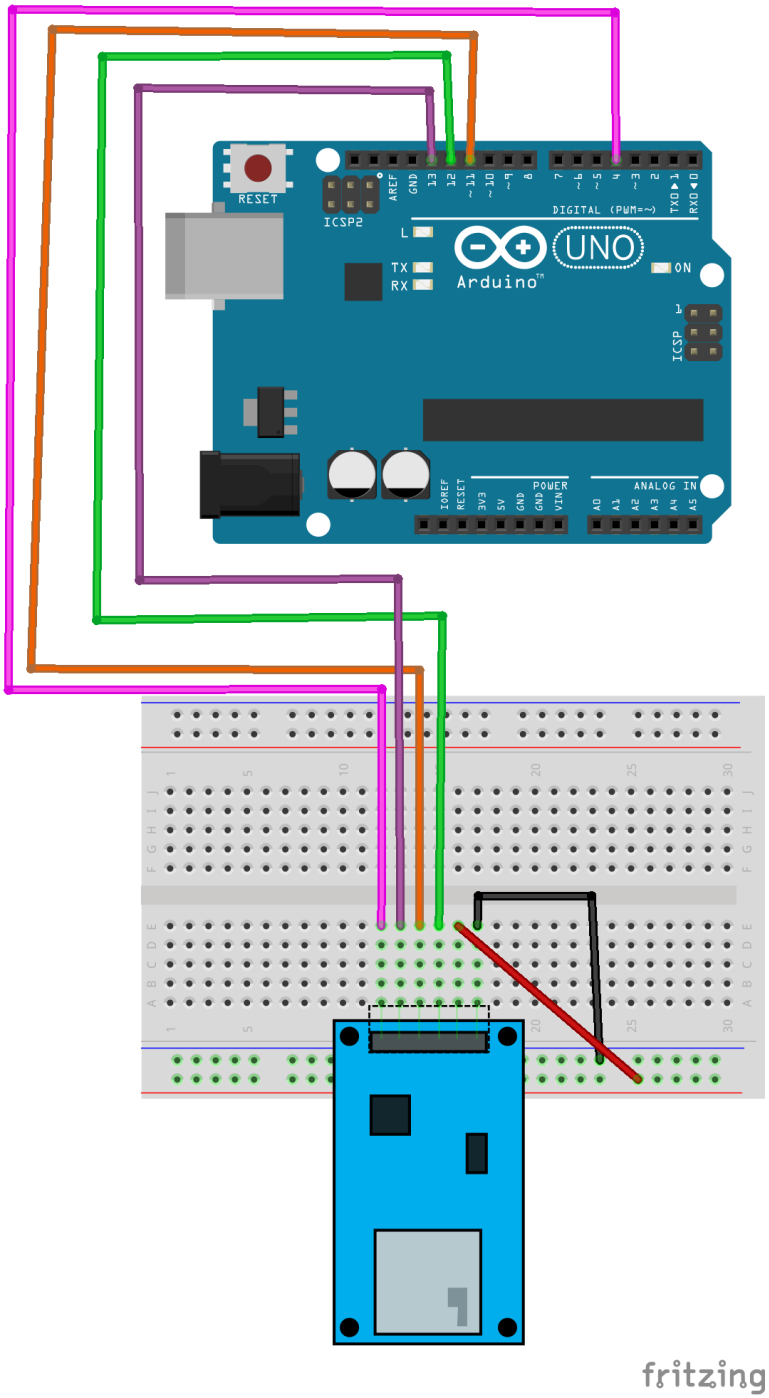
GPS Hook up (Pin 1 = GND, Pin 2 = 6, Pin 3 = 7, Pin 4 = 5V)



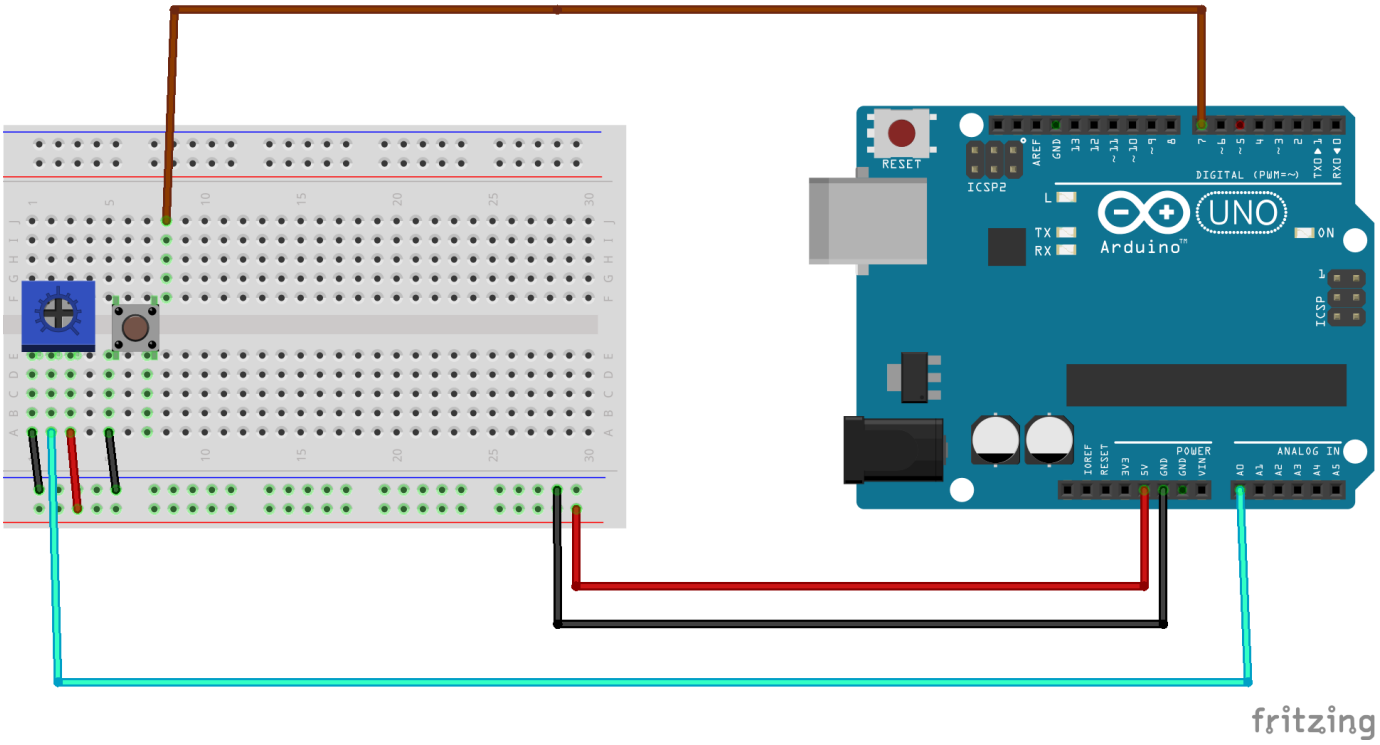
TDS hook up (Image is from DFrobot's wiki page)



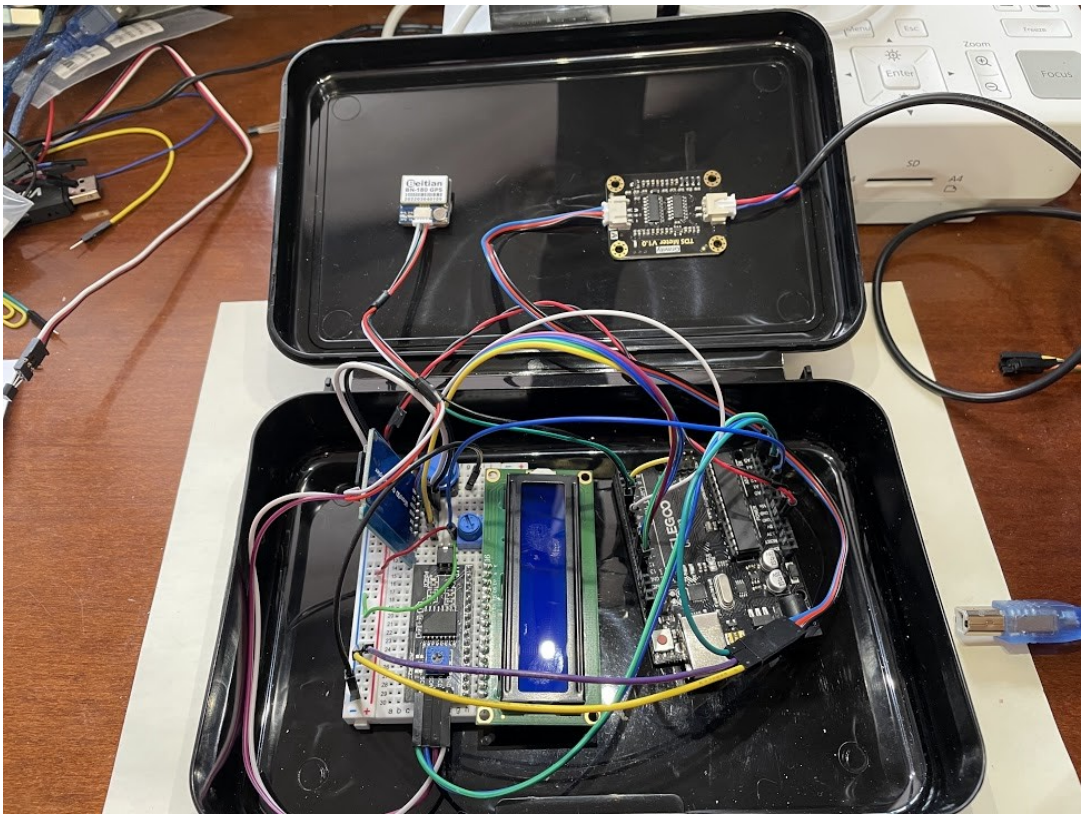
SD car reader Hook Up



Button and Potentiometer hook up (note you can replace the potentiometer with a temperature sensor if you want, just make sure it is waterproof.)



Here is what mine looks like as a prototype. I used tape to organize wires and tape down different modules.



Its now time to code this circuit. You can copy it from the worksheets or download it from the Sensing for Science website. You will need the TDS library from DFRobot which can be found here:

https://wiki.dfrobot.com/Gravity__Analog_TDS_Sensor___Meter_For_Arduino_SKU__SEN0244

```
#include <LiquidCrystal_I2C.h>
#include <SPI.h>
#include <SD.h>
#include <TinyGPS++.h>
#include <SoftwareSerial.h>
#include <EEPROM.h>
#include "GravityTDS.h"

// String to hold GPS data
String gpstext;

// GPS Connections
static const int RXPin = 6, TXPin = 5;

// GPS Baud rate (change if required)
static const uint32_t GPSBaud = 9600;

// TinyGPS++ object
TinyGPSPlus gps;

// SoftwareSerial connection to the GPS device
SoftwareSerial ss(RXPin, TXPin);

//I2C LCD
const int chipSelect = 4;

LiquidCrystal_I2C lcd(0x27,16,2);  // set the LCD address to 0x3F for a 16 chars and 2 line display

//Button Push
int run;
int buttonPin;

//TDS
#define TdsSensorPin A1
GravityTDS gravityTds;
float temperature = 25,tdsValue = 0;

//TMP
float degreesC = 0;                          //the temperature in Celsius, calculated from the voltage



void setup() {

// Start Serial Monitor for debugging
  Serial.begin(115200);

 // Start SoftwareSerial
   ss.begin(GPSBaud);

 //Prep for Button Push
  run = 0; //starts stopped
  buttonPin = 7; //whatever pin your button is plugged into
  pinMode(buttonPin, INPUT_PULLUP);
  pinMode(2,INPUT);

 //TDS startup
 gravityTds.setPin(TdsSensorPin);
 gravityTds.setAref(5.0);  //reference voltage on ADC, default 5.0V on Arduino UNO
 gravityTds.setAdcRange(1024);  //1024 for 10bit ADC;4096 for 12bit ADC
 gravityTds.begin();  //initialization
```

```
//LCD startup and SD Check
  lcd.init();
  lcd.clear();
  lcd.backlight();      // Make sure backlight is on

  if (!SD.begin(chipSelect)) {
   lcd.clear();
   lcd.setCursor(0,0);
   lcd.print("Card failed, or not present");
    // don't do anything more:
    while (1);
  }

  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("card initialized.");
  delay(1000);
  lcd.clear();
  lcd.print("Finding Sats");
}

void loop() {

while (ss.available() > 0)
    if (gps.encode(ss.read()))

      // See if we have a complete GPS data string
      if (displayInfo() != "0")
      {

        // Get GPS string
        gpstext = displayInfo();
Serial.println(gpstext);


degreesC = map(analogRead(A0),0,1023,0,100);

gravityTds.setTemperature(degreesC);  // set the temperature and execute temperature compensation
gravityTds.update();  //sample and calculate
tdsValue = gravityTds.getTdsValue();  // then get the value
Serial.print(tdsValue,0);
Serial.println("ppm");

 lcd.clear();
 lcd.setCursor(0, 0);                    //set the cursor to the top left position
 lcd.print("Degrees C: ");               //print a label for the data
 lcd.print(degreesC);                    //print the degrees Celsius

 lcd.setCursor(0, 1);                    //set the cursor to the lower left position
 lcd.print(tdsValue,0);                //Print a label for the data
 lcd.print("ppm");                       //print the degrees Fahrenheit
```

```
 //check button pressed, if so enter program condition (inside if statement)
 if(digitalRead(buttonPin) == LOW) //functions based off of button pulling input pin LOW
    {


 File dataFile = SD.open("datalog.txt", FILE_WRITE);

  //if the file is available, write to it:
  if (dataFile) {
    dataFile.print("Degrees C: ");
    dataFile.print(degreesC);
    dataFile.print(",");
    dataFile.print(tdsValue,0);
    dataFile.print("ppm");
    dataFile.print(",");
    dataFile.println(gpstext);
    dataFile.close();
 lcd.clear();
 lcd.setCursor(0,0);
 lcd.print("info saved");
 delay(1000);
  }
 // if the file isn't open, pop up an error:
  else {
lcd.clear();
lcd.setCursor(0,0);
lcd.print("error SD");
delay(1000);
}}



    delay(1000);
  }}


String displayInfo()
{
  // Define empty string to hold output
  gps.encode(ss.read());
  String gpsdata = "";

  // Get latitude and longitude
  if (gps.location.isValid())
  {
    gpsdata = String(gps.location.lat(), 6);
    gpsdata += (",");
    gpsdata += String(gps.location.lng(), 6);
    gpsdata += (",");
  }
  else
  {
    return "0";
  }

  // Return completed string
  return gpsdata;
}
```