Microcontrollers and Sensors: Electronic Measurement

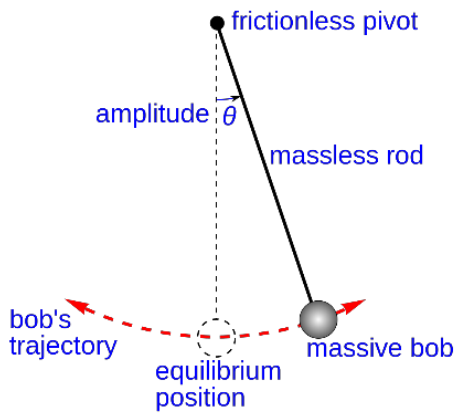
Learning outcomes

1. Use the Arduino to display data from a light sensor
2. Use the Arduino to capture data from a light sensor and write this to a file on an SD Card.
3. Use this captured data to calculate the period of a pendulum
4. *Additional (optional): use the light sensor to build a water-quality (turbidity) monitoring station*

Background

Huygens's Law for determining the period of the motion of a pendulum

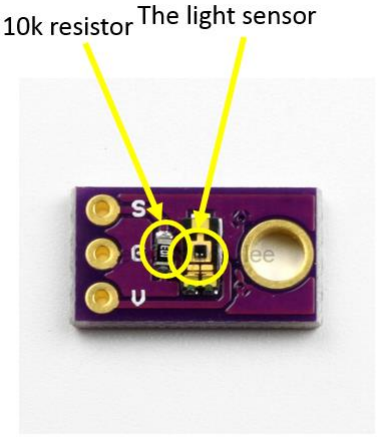
In this experiment, students will gain their first experience of using sensors with the Arduino system using a simple set-up consisting of a **light sensor connected to the Arduino** and a **pendulum**. Students will use the light sensor to measure the **period** of the motion of the pendulum and confirm that this motion obeys **Huygens's Law.**



A pendulum given a small initial push will swing backwards and forwards. An ideal pendulum, experiencing no friction at the pivot, will continue to swing with constant amplitude with a period, T, that depends only on the length of the pendulum, L, and the strength of gravity, g, such that:

$$T = 2\pi\sqrt{\frac{L}{g}}$$

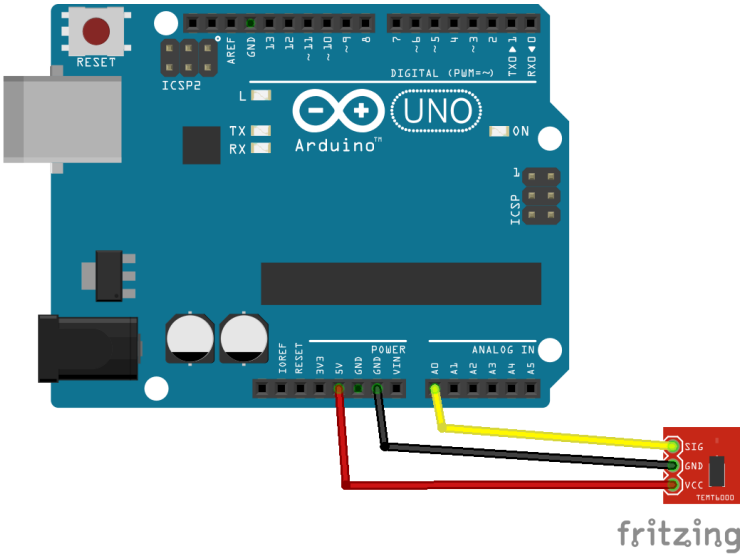
T is the time the pendulum takes to swing all the way from one side to the other <u>and</u> <u>back again</u>. This motion is independent of the mass of the bob, and largely independent of the amplitude of the swing (how large the angle, θ, is). In practice, friction at the pivot causes the motion to diminish over time, but, at least initially, this relationship, the so-called **Huygens's Law of the period of a pendulum** holds true (https://en.wikipedia.org/wiki/Pendulum).

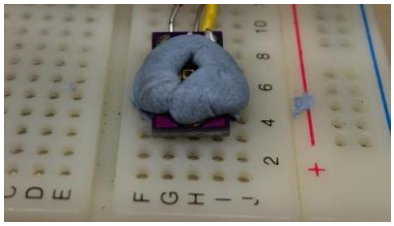


In order to do this, students will make use of the TEMT6000 ambient light sensor provided. The light sensor is mounted on a board with a 10k resistor. There are three connections:
G – GND (ground)
V - 5V from the Arduino
S – connection to an Arduino sensing pin (A0 in this instance).

1. Use the Arduino to display data from a light sensor

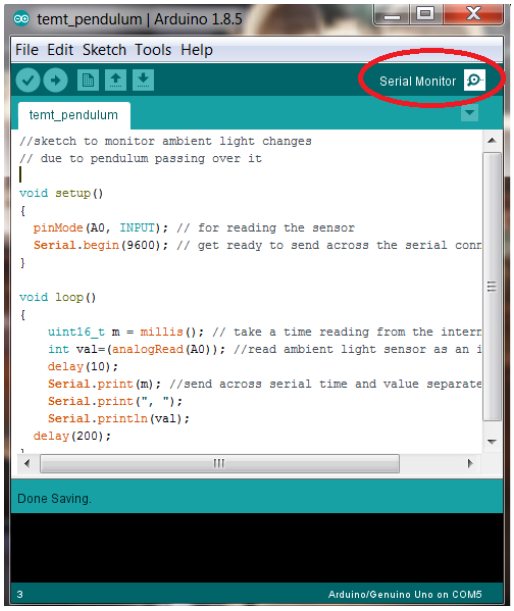
For the first iteration of the experiment, the set-up is very straightforward – the sensor can be connected directly to the Arduino as below. Make use of the Blu Tac provided to ensure that the sensor sits flat on the bench. You may also find it useful to build up a Blu Tac surround around the sensor as illustrated below. This will help restrict the field of view of the sensor. (***Note***: you don't need to have the sensor mounted on a breadboard as in the photo, mounted on the bench is good enough.)





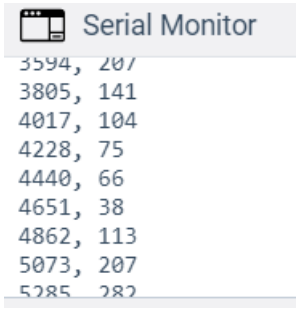
The Arduino sketch for this first iteration is very simple, and can be seen below (you can copy-paste this into the Arduino IDE) **(remember to check you are reading from the correct COM port):**

```
//sketch to monitor ambient light changes
// due to pendulum passing over it

void setup()
{
  pinMode(A0, INPUT); // for reading the sensor
  Serial.begin(9600); // get ready to send across the serial connection
}

void loop()
{
    uint16_t m = millis(); // take a time reading from the internal clock
    int val=(analogRead(A0)); //read ambient light sensor and store as integer
    delay(10);
    Serial.print(m); //send across serial time and value separated by comma
    Serial.print(", ");
    Serial.println(val);
  delay(20); // controls the rate at which we take readings
}
```

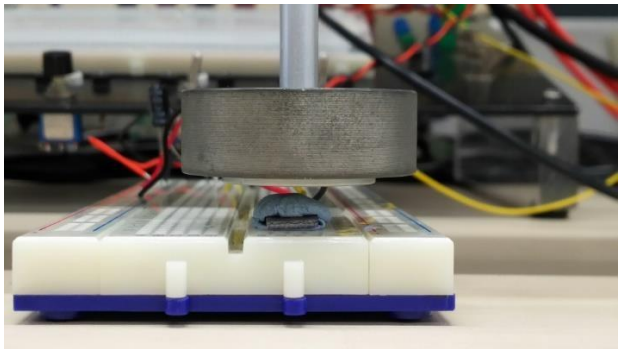


As it can be seen, the sketch uses the serial connection between the Arduino and the PC for sending data from the sensor, via the Arduino, to the PC. This can be monitored using the **Serial Monitor** of the Arduino. The Serial Monitor is launched by clicking on the magnifying glass symbol, top right on the Arduino IDE window. Note: make sure the Baud Rate (rate of data transfer) of the serial monitor is set to same rate as in the sketch (9600).

As we are going to measure the period of a pendulum, we are also recording the time of the measurements, using the internal Arduino clock, and assigning this as a "**uint16_t**" – an unsigned 16-bit integer – we've called "m". We will print this time and the reading of the light sensor on the serial monitor separated by a comma (Serial.print(", ");).



Launch the serial monitor and upload the sketch to the Arduino. Pass you hand over the sensor a few times, casting a shadow over it (**do not touch the sensor**). If everything is working correctly, you should see the reading on the monitor change as the shadow of your hand falls on the sensor, something like left (note: the actual reading will depend on how much light there is in the room where you are).



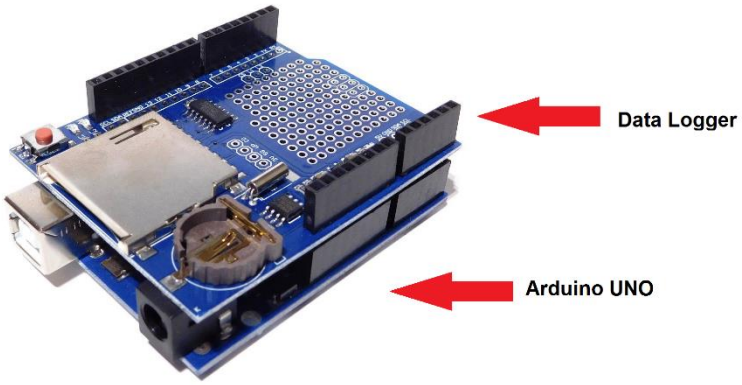
Try this experiment again swinging the pendulum with which you have been provided. They key is to have the pendulum pass over the sensor as close to the sensor as possible without touching it, as left. (***Note***: you don't need to have the sensor mounted on a breadboard as in the photo, mounted on the bench is good enough.)

This is all well and good. But in order to measure the period, we are going to have to **capture the data and process it**. We will do this by writing the data to a file on an **SD card.**

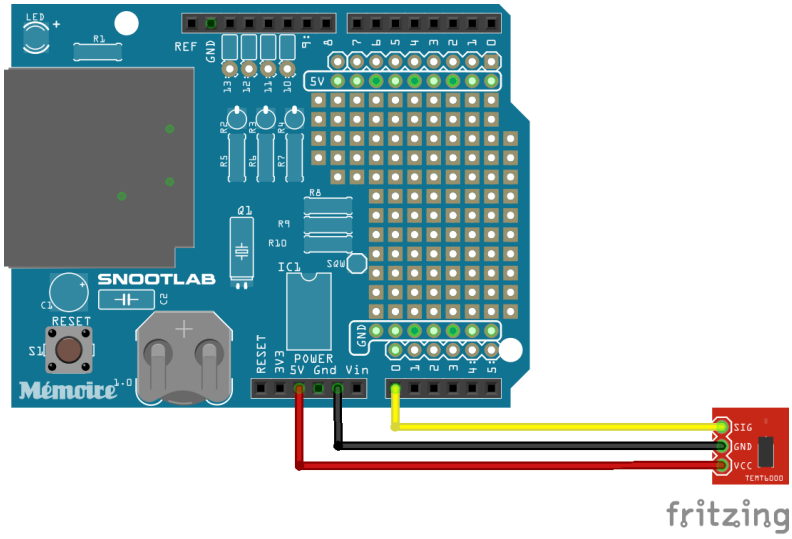
2. Use the Arduino to capture data from a light sensor and write this to a file on an SD Card.

In order to write data from the Arduino to a file on an SD Card, we will make use of the **Data Logger Shield.** This is an additional piece of hardware that we will attach to the Arduino.

Disconnect the TEMT6000 sensor from the Arduino. Add the shield to the Arduino as below, ***taking a great deal of care to correctly align the pins before pressing down.***



You can now connect the TEMT6000 to the data logger as below.



In the first instance, simply check that everything still works as before, and you can monitor the readings from the sensor on the serial monitor using the same sketch as in section 1.

If everything is okay, next insert the SD card into the reader/writer. The card is already formatted. We will first check that the system can read the card. This is done using a sketch that can be found in the example sketches in the Arduino IDE "CardInfo" (file > examples > SD > CardInfo). We need to tell the sketch which data logger we are using. We are using a copy of the "Adafruit SD…". Therefore we need to change the line **const int chipSelect = 4;** to **const int chipSelect = 10;**

Once you have changed this line, upload the sketch to the Arduino and open the serial monitor. If everything is working correctly, you should see the message below appear:

```
Initializing SD card...Wiring is correct and a card is present.

Card type: SDHC

Volume type is FAT32

Volume size (Kbytes): 15542720
Volume size (Mbytes): 15178
```

You are now ready to create a new version of the sketch that takes light level readings. In this sketch, in addition to sending the time-stamp and light level reading to the PC via the serial connection, you will create a CSV (comma-separated values) file on the SD card we will call “data.csv”, and then write the time-stamp and light level reading to this file. The sketch can be found on the next page.

You can see that this sketch takes the sketch from section 1 and adds new lines associated with writing to the SD card. The syntax of the card operations using definitions contained in the two libraries needed for serial communication with the SD card SD.h and SPI.h and these must be included at the start of the sketch.

```
// DLogger_Light made for using with the TEMT6000 light sensor and FAT32 formatted 16 GB SD Card
//
// adapted from https://maker.pro/arduino/tutorial/how-to-make-an-arduino-sd-card-data-logger-for-temperature-sensor-data
// and http://www.toptechboy.com/arduino/arduino-lesson-21-log-sensor-data-to-an-sd-card/
// and https://learn.adafruit.com/adafruit-data-logger-shield/using-the-real-time-clock-3
//
// Jamie Barras June 2019


#include <SD.h> // the two libraries we need for the SD card
#include <SPI.h>

File sdcard_file;

void setup() {
  Serial.begin(9600);
  pinMode(A0,INPUT);
  pinMode(10, OUTPUT); // reserve Pin 10 on Arduino Uno for data logger

  // SD Card Initialization
  if (SD.begin()) // if the system detects the card reader
  {
    Serial.println("SD card is ready to use.");
  } else {
    Serial.println("SD card initialization failed");
    return;
  }
 }

void loop() {

    uint16_t m = millis(); // take a time reading from the internal clock
    int val=(analogRead(A0)); //read ambient light sensor as an integer
    delay(10);
    Serial.print(m); //send across serial time and value separated by comma
    Serial.print(", ");
    Serial.println(val);

  sdcard_file = SD.open("data.csv", FILE_WRITE);
  if (sdcard_file)
  {
  sdcard_file.print(m);          // milliseconds since start
  sdcard_file.print(", ");      //write a comma
  sdcard_file.print(val);                          //write light data to card
  sdcard_file.println(", ");                          //write a comma
  sdcard_file.close();                           //close the file
  }
  // if the file didn't open, print an error:
  else {
  Serial.println("error opening file");
  }
  delay(20); //wait before reading again
}
```

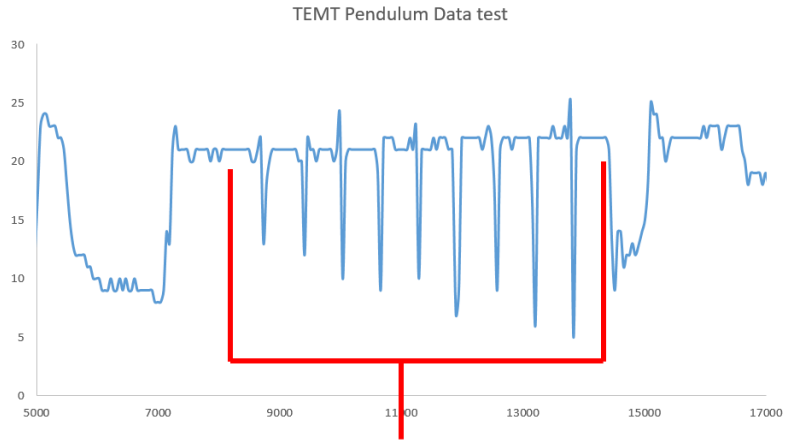
Copy-paste the above into the Arduino IDE. Upload the sketch, run the serial monitor. If everything is working correctly, you should see the message “SD card is ready to use” appear and then the time and light level readings should appear as before.

Repeat the pendulum experiment as before, making sure by checking the serial monitor that you can see the effect of the shadow of the pendulum falling on the sensor in the readings. Do this for a series of passes (say 10). Then stop and disconnect the Arduino.

3. Use the captured data to estimate the period of the pendulum

Transfer the SD card to one of the USB readers provided. This will allow you to review the file you have created on the PC.

Rename the data.csv file “data001.csv”. Open the renamed file in excel (double-clicking on the file should do). Check that you have two columns of data, the time in milliseconds, and the light level reading. Now create a graph of the data using the insert scatter plot and selecting the two columns. You should see something like this:



Pendulum passing over the sensor

Use this graph to estimate the period of your pendulum (reread the background material at the start of these notes to work out the relationship between what you see in the graph and the period). **(Note: the period will depend on the length of the string, so don’t expect your value to come out the same as that of other groups – unless you use the same length of string.)**

You have now taken and captured readings from a sensor, saved these to a file and processed the data.

Additional (optional) out of the lab: can you think of an algorithm to automate the process of determining the period? Apply Computational Thinking to this problem.

Note: if you have problems with the hardware, the test data illustrated above is available on the KEATS page for you to process.

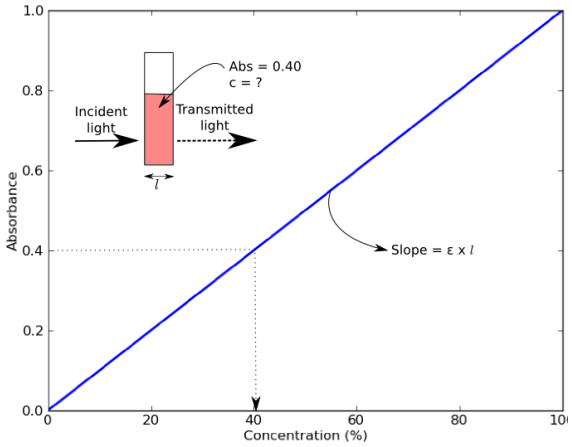
4. (Additional Optional) Water Quality Monitoring Station

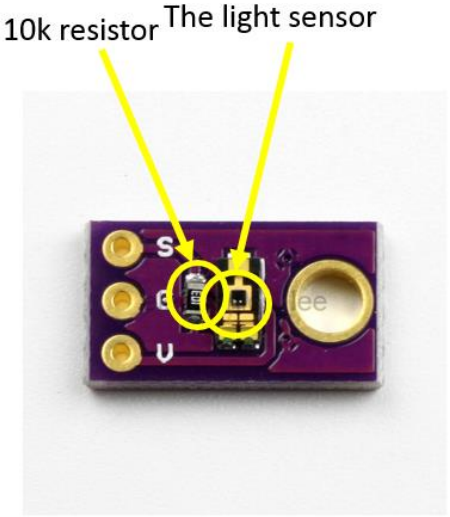
Background

WHO/UNICEF estimate that 2.1 billion people lack access to safe drinking water (http://www.un.org/en/sections/issues-depth/water/).

The optimum strategy for water quality monitoring (WQM) is real-time continuous monitoring at multiple sample points, something that is made possible with modern technologies, but is not widely adopted due to high cost. This has led researchers to exploit the possibilities of Open-Source Hardware (OSHW) systems like the Arduino microcontroller (e.g. Rao, Aravinda & Marshall, Stephen & Gubbi, Jayavardhana & Palaniswami, Marimuthu & Sinnott, Richard & Pettigrovet, Vincent. (2013). Design of low-cost autonomous water quality monitoring system. Proceedings of the 2013 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2013. 14-19.)

In this experiment, students will adapt the light sensor build with switch control to produce a simple water quality monitoring system that exploits **BEER'S LAW** (the darker a solution the more light is absorbed) to use light level measurements to detect the level of ***turbidity*** – cloudiness – in a water sample and alarm if it is too high.

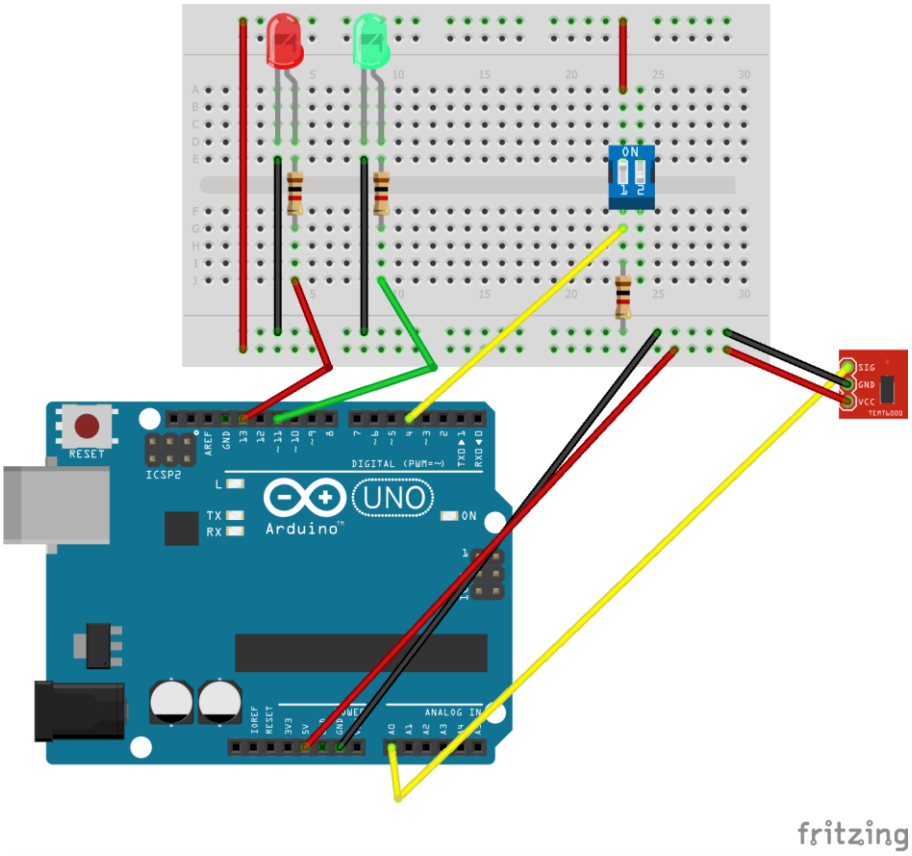




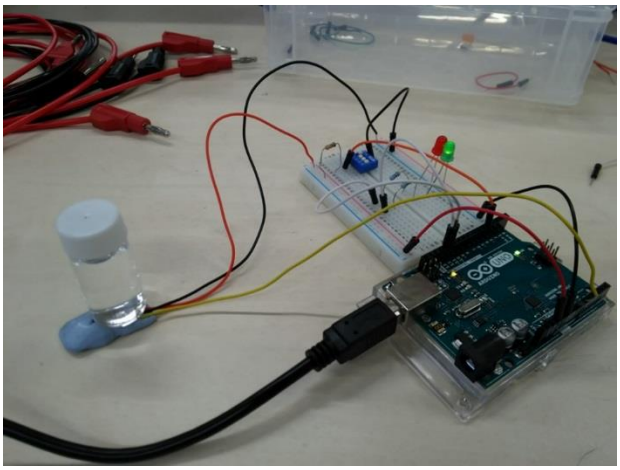
In order to do this, students will again make use of the TEMT6000 ambient light sensor provided. The light sensor is mounted on a board with a 10k resistor. There are three connections **G** – GND (ground)
V - 5V from the Arduino
S – connection to an Arduino sensing pin (A0 in this instance).

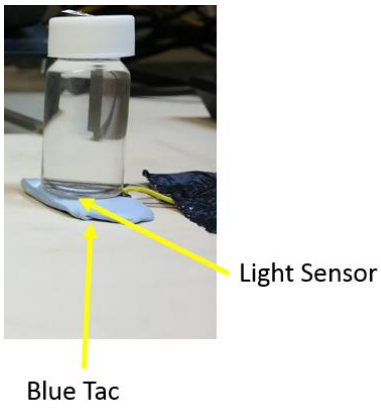
Method

Modify the light sensor build as below, adding two LEDs and a DIP switch (use the 4 pin DIP switch in the lab, not the 2 pin as shown below) and adding the TEMT6000 sensor connected to A0, 5V and GND. (Note the illustration below shows just the Arduino, but you can do this with the Data logger in place. The pin assignments are the same.).

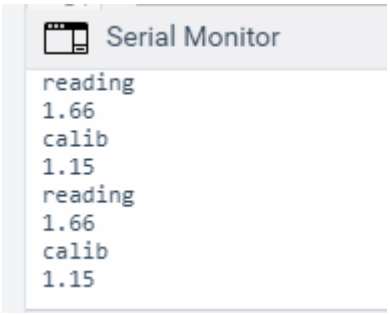


Below is a photo showing a complete build.





The leads on the sensor allow the sensor itself to be secured to the bench using the blue tac provided as left.



The purpose of the switch in this instance is to allow us to run the system in CALIBRATION or READING mode. The CALIBRATION mode sets up a reading on a clear solution that can then be used for comparison with readings taken from the bottle containing coloured solution. In the program, the calibration reading is written to the **EEPROM** on the microcontroller and recalled when needed. This can be seen from the code provided in the "***light sensor sketch***" file on the KEATS page.

NOTE: water and electronics don't mix, so work only with the sealed bottles

1. **Complete the circuit as above and place the clear solution on top of the sensor. Upload the light sensor sketch into the IDE and bring up the serial monitor.**
2. **Carry out a CALIBRATION with the switch in one position and then switch to READING. Test the system with the coloured solution. If the calibration has been successful the red LED will light up. You may find it useful to try different values for "nmeas" the number of readings the system averages in order to improve stability.**
3. **Now do a CALIBRATION with the coloured solution. If this is successful, when you switch to READING and place the clear solution on the sensor, the green LED will light.**

Think about ways to improve this very basic system.

Equipment

1 x Arduino Uno & USB cable
1 x Data Logger Shield
1 x SD Card
3 x LED
3 x 470 Ohm resistors
1 x 1 kOhm resistor
Switch
1 x TEMT6000 light sensor
PC with Arduino Integrated Development Environment (IDE)