EE380 Project 3 Weather Station

Team 3: Abdullah, Austin, Quentin, Sayed, Zenawi

4/25/2019

Project Description:

Design an instrumentation package that will be placed in a remote location for up to seven days to sample relevant data to support a hypothesis.

Hypothesis:

As UV light increases, ground moisture, and air humidity will both decrease.

Hardware Design:

Figure 1 Hardware Schematic:

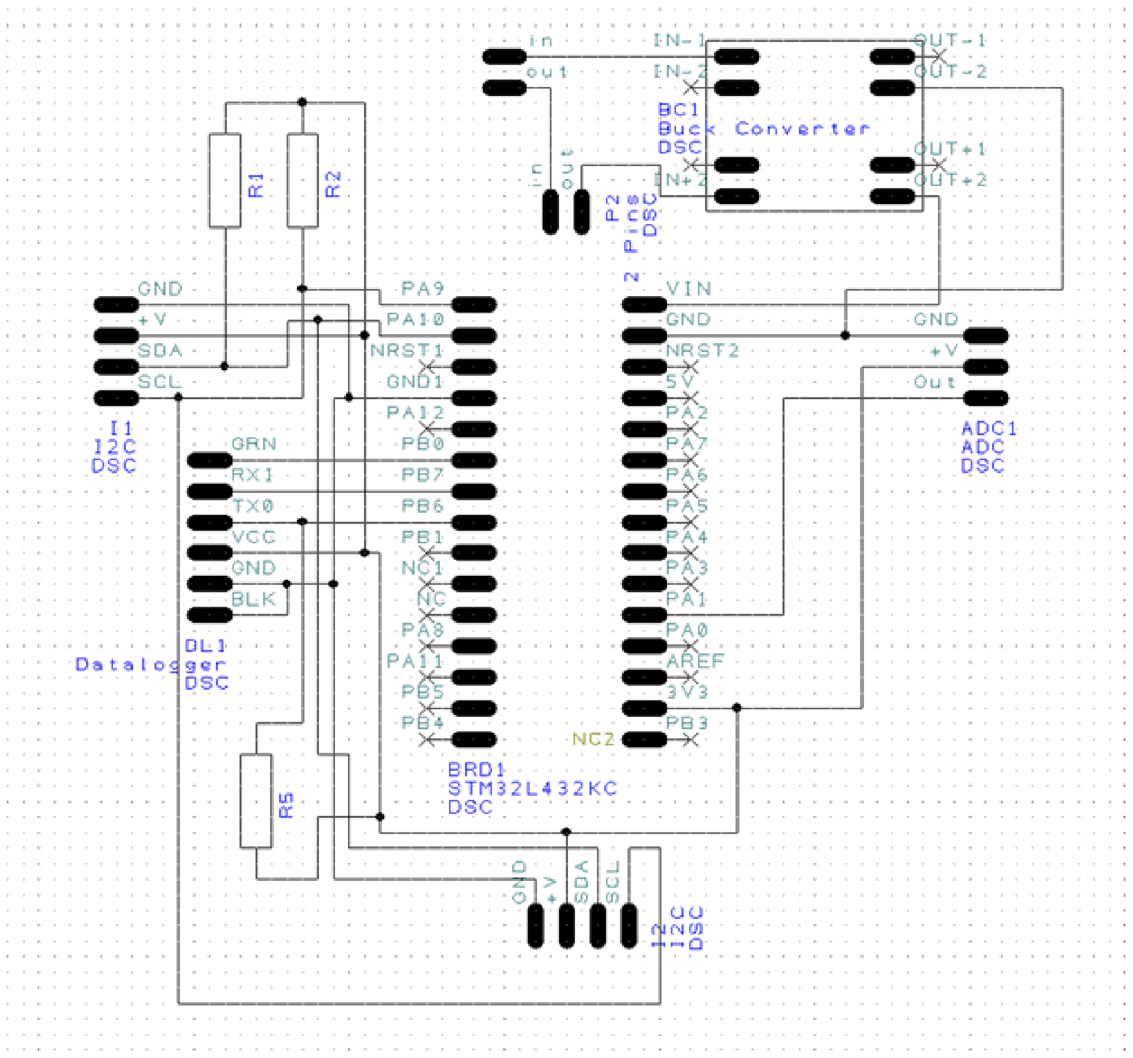
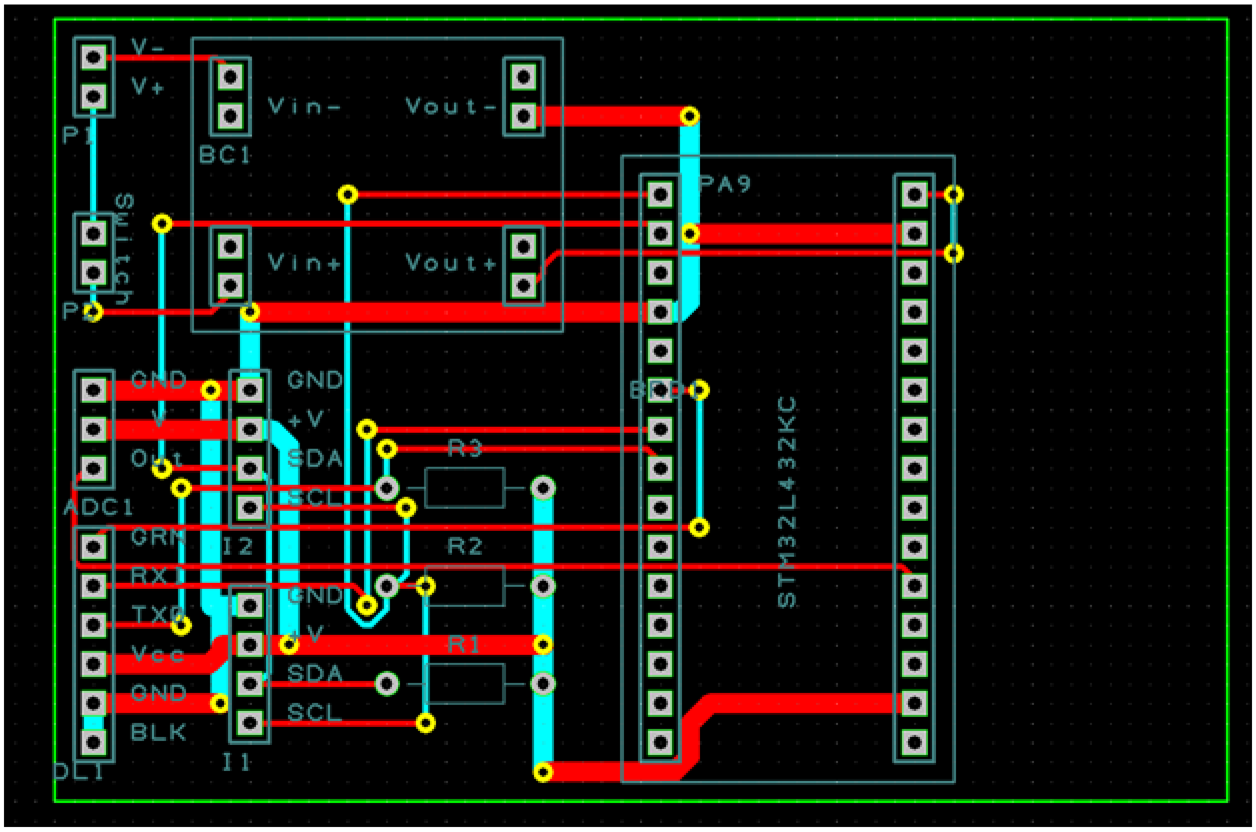


Figure 1 shows the complete hardware schematic. This was designed by first building all the components within PCB Artist. Once this was done, there needed to be communication between the programmers and the hardware designer to figure out which pins would be needed on the microcontroller. Specifically, the ADC and I2C sensors would need to be connected to the pins associated with the according channels for ADC and I2C. Additionally, I2C requires a pull up resistor on the SDA and SCL lines and thus needed to be accounted for. Finally, a buck converter was added so that 9V batteries could be stepped down to 5V to be compatible with the board.

Figure 2 PCB Design:



Once the schematic was complete, it needed to be transferred over to PCB format. This was done by using the auto route button on PCB artist. Special consideration was put towards making sure that all the pins were properly spaced so that the microcontroller and buck converter specifically would fit. The final product is seen in Figure 2.

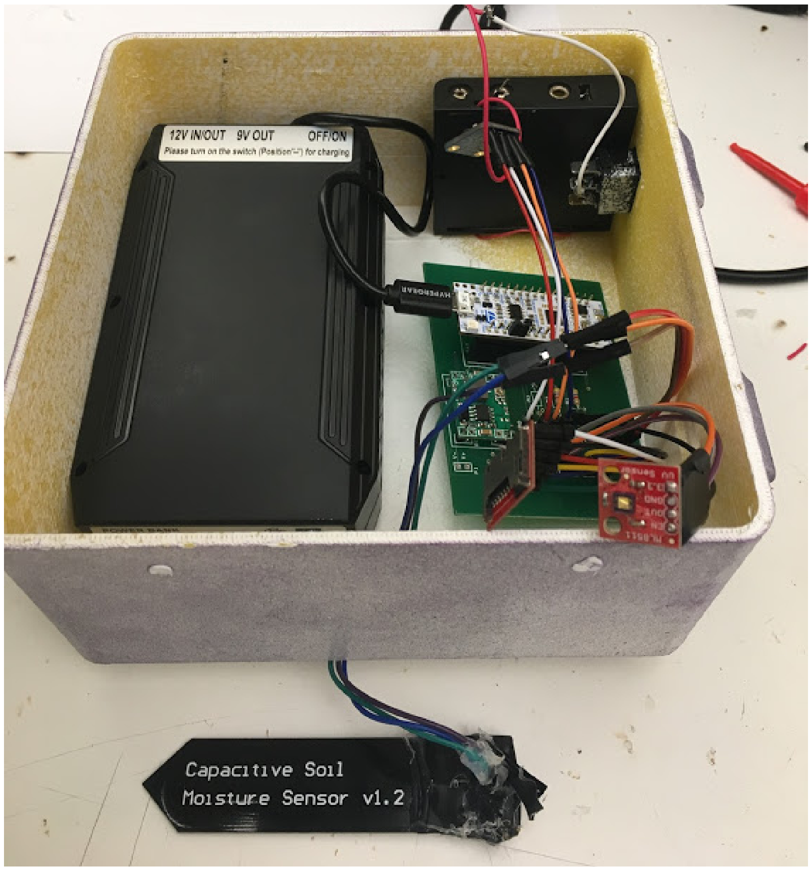
NOTE: While this is the design that was used for our PCB, it is not what was really used in the final product. Due to changes of plans in power management and a sensor not working properly we had to adjust as necessary. Specifically, the new sensor had to be directly attached to the microcontroller via jumper wires and the buck converter was no longer used due to us using a power bank that had a USB output.

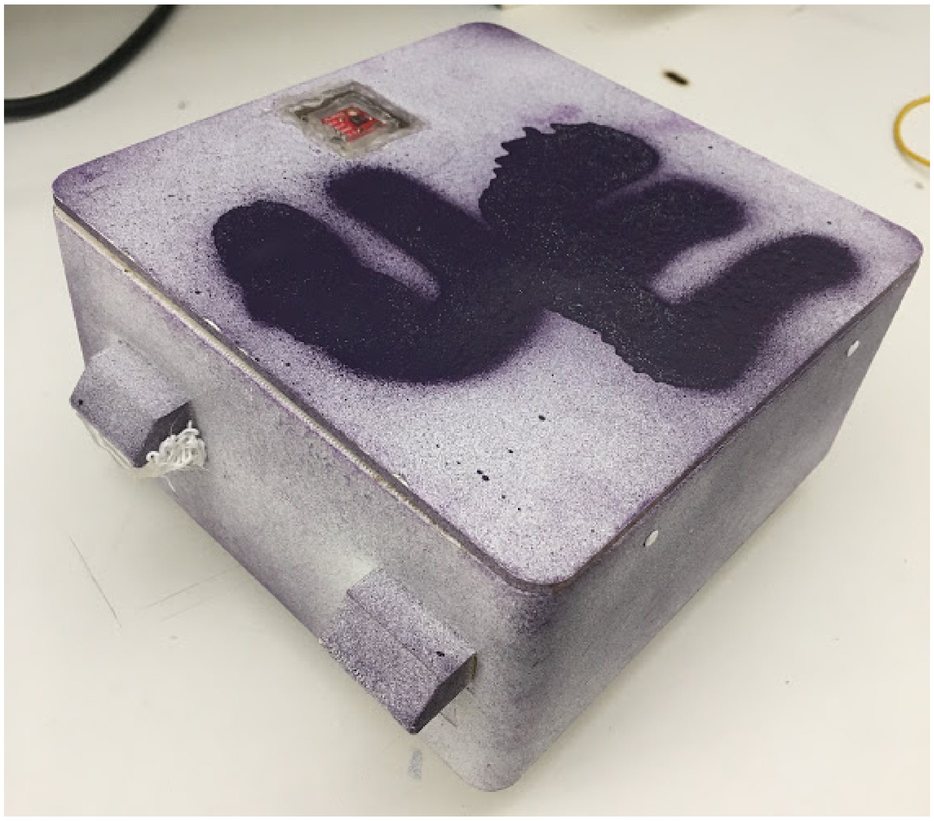
Components used:

1) STM32L432KC Microcontroller - x1
2) 4700 ohms resistor -x2
3) ML8511 UV light sensor -x1
4) Capacitive Soil Moisture Sensor v1.2 -x1
5) Adafruit HTU21D-F Humidity sensor -x1
6) Sparkfun Openlog Data Logger -x1
7) TalentCell Rechargeable 20000mAh Power Bank -x1

Packaging design:

For the packaging design, we decided to use Autodesk Inventor to create a box of sorts that would be later 3D printed. We understood that the plastic that the printer would be using was not completely waterproof, as there would be many small cracks and crevices that water could seep into over time. With that in mind, we decided to use spray-on flex seal in order to waterproof the outside of the box completely. There was also the consideration of our humidity sensor that had to be exposed to the outside air in order to get an accurate reading. We had a ‘hood’ designed to go over top of the hole in the side of the box that was used to expose the sensor. We later were concerned with the amount of water that could splash up onto the sensor from under the hood if it rained hard, so we used a 3D printing pen to create an upside-down hood essentially to help prevent that. Right before deployment, we sealed the lid of the box with tape around the outside, so that water would not seep in through the lid. The ground moisture sensor had to be external to the box as it had to be stuck into the ground to collect data. With this in mind, we created a compartment on the underside of the box to put the moisture sensor in when it was not being used. The top of the lid contained a square cut-out with a piece of clear plastic to cover it, so that our UV light sensor had access to the sunlight throughout the day.

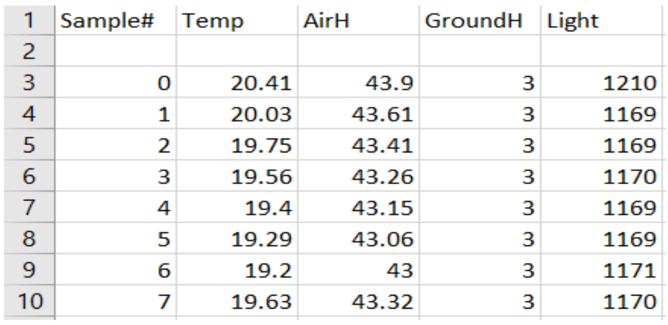

12V IN/OUT 9V OUT OFF/ON
Please turn on the switch (Position'–') for charging
UV Sensor
ML8511
Capacitive Soil
Moisture Sensor v1.2



Software Design:

Measurements were made every 10 minutes, and timer2 channel 3 was used as a delay between readings. Following the delay, readings were taken from sensors one at a time, stored in variables, and sent all at once to the data logger. Since the program was not computationally heavy and speed was not crucial, interrupts were not needed for UART or ADC communication. Instead, polling was used. To avoid the program polling indefinitely in the case an issue happens with the hardware, all while loops were exited after a certain number of iterations.

Communication with the data logger was done using UART. Since we could only send one byte at a time, numbers bigger than 9 could not be sent in one message. Thus, data points (floats) were converted to strings and sent one character at a time. The data logger could be set in three different modes: new log mode, sequential log mode, and command mode. Since many data points were needed, the data logger was set in sequential mode where new data was appended to the same file. The data was organized in a table for easy extraction, as shown in the figure below.



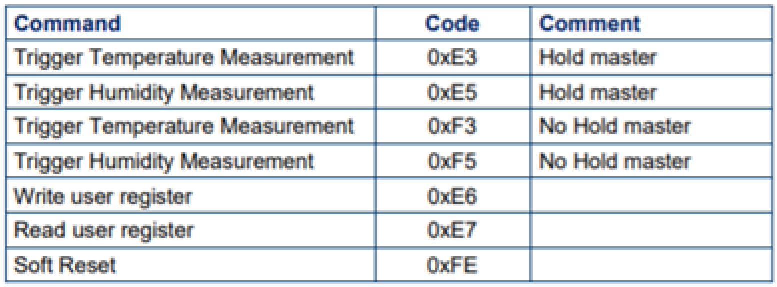
1	Sample#	Temp	AirH	GroundH	Light
2					
3	0	20.41	43.9	3	1210
4	1	20.03	43.61	3	1169
5	2	19.75	43.41	3	1169
6	3	19.56	43.26	3	1170
7	4	19.4	43.15	3	1169
8	5	19.29	43.06	3	1169
9	6	19.2	43	3	1171
10	7	19.63	43.32	3	1170

Sensors:

Humidity/Temperature Sensor:

The HTU21D(F) is a new digital humidity sensor with temperature that sets new standards in terms of size and intelligence, it is embedded in a reflow solderable Dual Flat No leads (DFN) package with a small 3 x 3 x 0.9 mm footprint. This sensor provides calibrated, linearized signals in digital, I²C format. This is a low power sensor that is designed for for high volume and cost sensitive applications with tight space constraints.

SCL is used to synchronize the communication between microcontroller and HTU21D(F) sensor. Since the interface consists of fully static logic there is no minimum SCL frequency. The DATA pin is used to transfer data in and out of the device. For sending a command to the HTU21D(F) sensor, DATA is valid on the rising edge of SCL and must remain stable while SCL is high. The HTU21D(F) sensor requires a voltage supply between 1.5V and 3.6V. After power up, the device needs at most 15ms while SCL is high for reaching idle state (sleep mode), i.e to be ready accepting commands from the master. To initiate transmission, a start bit has to be issued. It consists of a lowering of the DATA line while SCL is high followed by lowering SCL. After sending the start condition, the subsequent I²C header consist of a 7-bit I²C device address 0x40 and a DATA direction bit (‘0’ for Write access: 0x80). The HTU21D(F) sensor indicates the proper reception of a byte by pulling the DATA pin low (ACK bit) after the falling edge of the 8th SCL clock. After the issue of a measurement command (0xE3 for temperature, 0xE5 for relative humidity), the Master must wait for the measurement to complete. The basic commands are given in the table below:



Command	Code	Comment
Trigger Temperature Measurement	0xE3	Hold master
Trigger Humidity Measurement	0xE5	Hold master
Trigger Temperature Measurement	0xF3	No Hold master
Trigger Humidity Measurement	0xF5	No Hold master
Write user register	0xE6	
Read user register	0xE7	
Soft Reset	0xFE	

The way did the communication in my code is setting up a master (the microcontroller) that reads data from the slave (The sensor). The master sends a 0xE3 to the salve to trigger the temperature sensor then reads the temperature in Celsius using that equation:
Temp = -46.85 + 175.72 x Stemp / (2^16)
Then, the master sends 0xE5 to slave tot trigger the humidity sensor then I used this equation to calculate the percentage of air humidity:
RH = -6 + 125 x (SRH/2^16)

<ins>Ground Moisture Sensor & UV Light Sensor:</ins>
The ground moisture sensor used an analog output to send data to the microcontroller just like the UV light sensor did, so we had to set up 2 different ADC channels for the project. We decided to use a non-continuous method for changing the channels and reading in the data from the Data Register. We did this by manually changing the channel for first conversion after the previous measurement had been taken and loaded into the correct variable.

<u>Power Requirements:</u>

<u>Considerations:</u>

At first, we were planning on using batteries to supply power to our circuit. We calculated the total current the circuit draws by looking at the datasheets of the three sensors we used and the data logger. To measure the current the microcontroller draws we used a multimeter, and then the current the devices draw was listed on the table below.
In the end, we decided to use a power bank to ensure we would have enough power to last us the entire time. We also made sure that once the power bank is connected to the microcontroller, it stays on until it needs to be charged again because some power banks auto have turn on/off. Also, the power bank has the advantage of being rechargeable. The power bank is also neater and easier to use because we could use a USB 5v instead of having to use wires from batteries and then use a buck converter to convert from 9v to 5v.

Other things I have looked into to minimize the operating current of our circuits:
1) Low power mode
2) Reducing clock speed
3) Turning off on-board LEDs of the microcontroller

<u>Power Consumption:</u>

Component	**Current**	**Voltage**	**Power (P=VI)**
Microcontroller	60mA	5	300 mW
Moisture sensor	5mA	3.3	16.5 mW
Humidity sensor	500uA	3.3	1.65 mW
Light sensor	500uA	3.3	1.65 mW
Data logger	3mA (20mA when active)	3.3	11 mW

Total power consumption = 330.8 mW

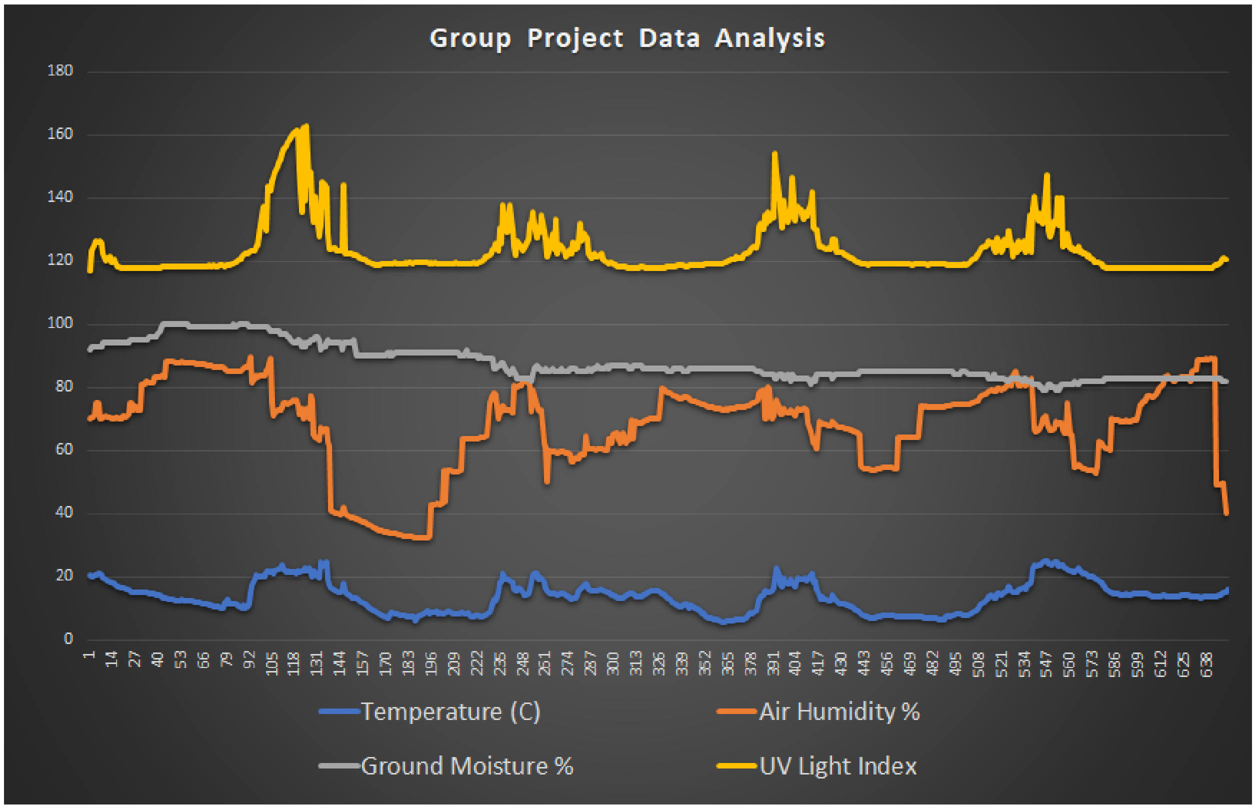
The power bank has capacity of 20000mAh for 5v.
- 20000 * 5 = 100000 mWh/330.8 mW = 302.3 hours / 24 = 12.5 days

- ○ We could have run the device for roughly 12 days on the power bank.

Novel Features:

- Temperature readings were collected.

Analysis:



In this graph, the orange line represents the air humidity percentage, the gray line represents the percentage of ground moisture, and then finally, those two are compared against the yellow line which represents the UV light index. After analyzing this graph it has come to our conclusion that our hypothesis, both ground moisture and air humidity would decrease as the UV light increased, was proven to be half correct. The ground moisture actually followed our hypothesis. It is hard to tell from the data, as it was raining most of the time during data collection and the ground was almost completely saturated because of this, but you can see dips in the ground moisture data at the periods where the UV light increases. The air humidity data also appears to drop shortly after the spikes in the UV light data. So essentially, the humidity appeared to decrease in the evenings. It is hard to tell if there is a direct relationship with the UV light however, because the humidity data does make spikes that do not completely correlate to the UV light data. The air humidity sensor we used also collected temperature data, which is the bottom blue line on the graph. An extra observation we made using this temperature data was the close direct relationship between the temperature and the UV light. This makes sense as the UV light increases during the day, and the temperature is typically warmer when the sun is out.