# Self-Sustainable IoT Biosensors with Long-Range Wireless Communication

Calvin Condo - PCB design and power
Qin Xia - Sensors & Testing
Chuxin Chen - Arduino / Sensors
Lun Zhang - LoRa Wireless module/Arduino
Yuchen Zhao - LoRa Wireless module/Arduino
Luke Healy- Sensors & Testing

Client/Advisors: Dr. Meng Lu and Dr. Cheng Huang

Luke

## Problem Statement

Dr. Lu would like a device that measures the concentration of organisms (bacteria) in soils. Dr. Lu plans to pour certain chemicals into the soil that react with the bacteria and produce light. Our project is to create a device that will measure the light produced by the bacteria to get an estimation on their concentration. The device would also measure the temperature for information on the conditions the bacteria lives in. All this data will be sent wirelessly to a receiver over a 300 meter range.

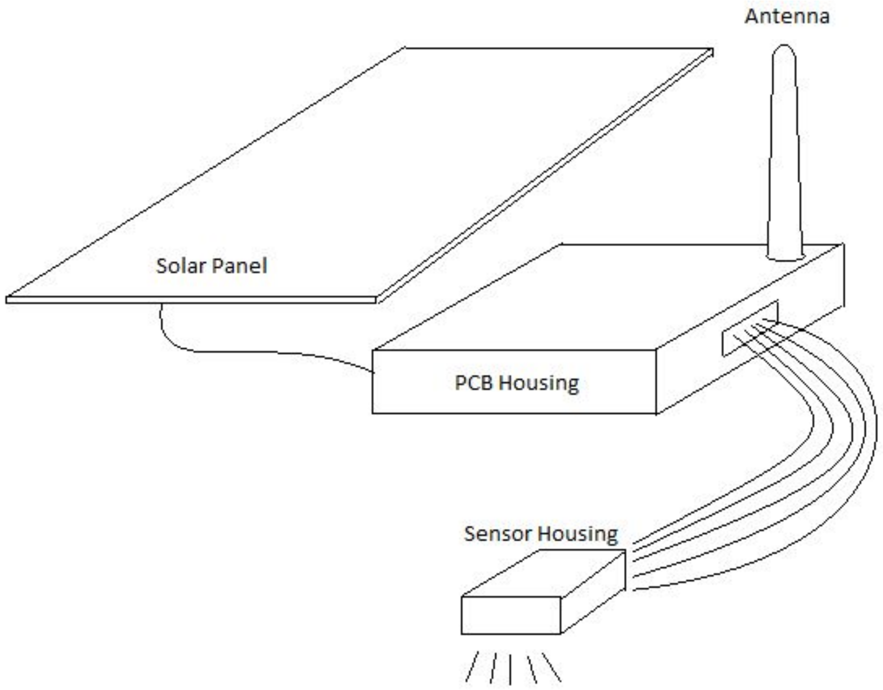
Luke

## Project Overview

- The device will be purely for Dr. Lu’s research purposes.
- The device will:
  - Be completely self-sustainable using solar power.
  - Accurately measure light (lux) and temperature.
  - Consistently send and receive complete and correct data transmissions.
  - Pocket sized and be able to survive in outdoor weather conditions.
- The total cost for the components are below $50 and the PCB is $26.95 (OSHpark).

Luke

## Conceptual Sketch



- The ‘PCB Housing’ holds main PCB with the LoRa module and circuitry for power and sensors.
- The ‘Sensor Housing’ holds the sensor PCB with the light sensor and temperature sensor. It is connected to the  main PCB via ribbon cable.

Luke

## Functional Requirements

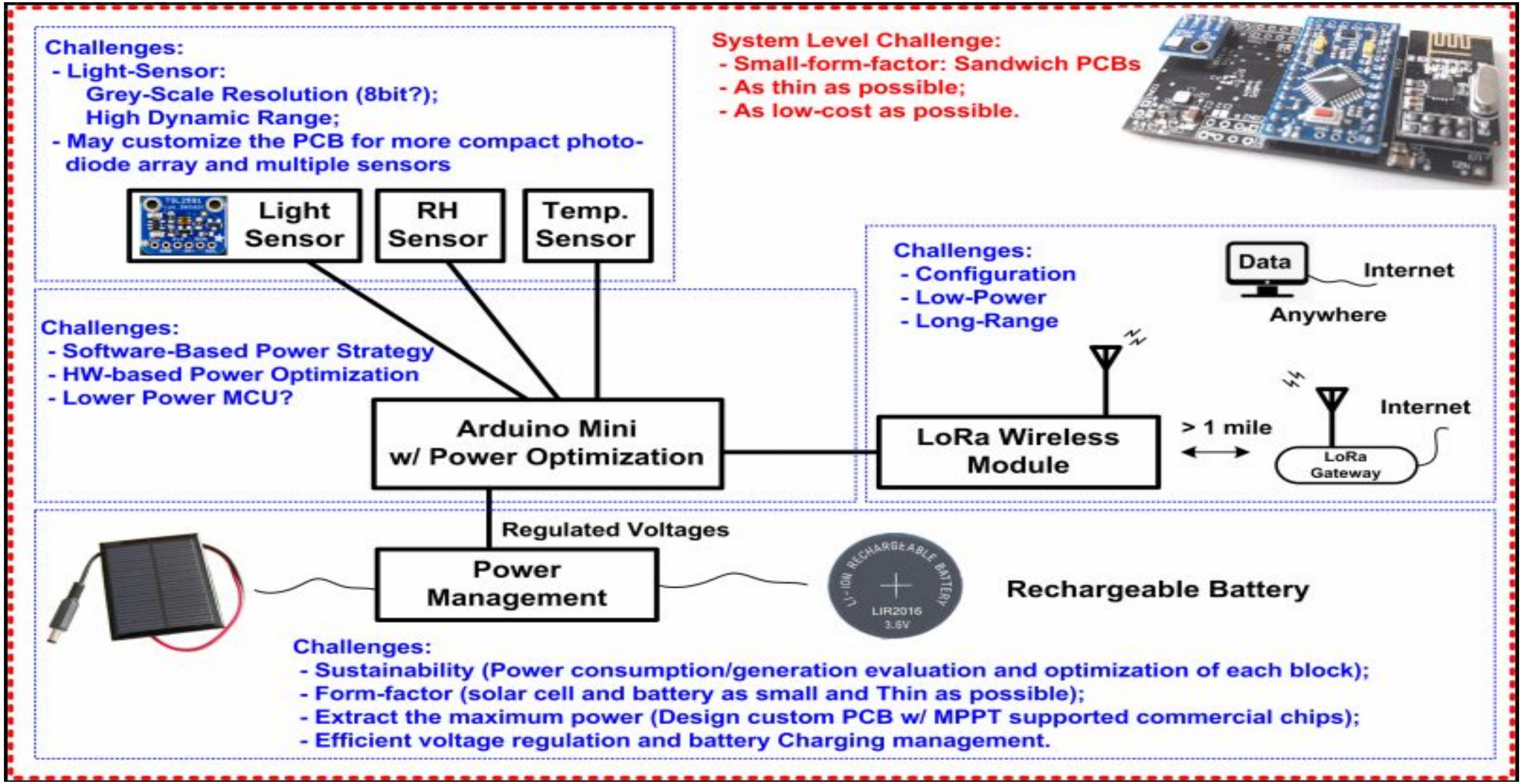
- Gather correct information from all sensors
- Use sensors that are sensitive enough to pick up small amounts of light
- Send data wirelessly over a long range
- Entire product must be self-sustainable (solar power)
- Power consumption must be very low

## Non-Functional Requirements

- Pocket sized
- Left out in field for long time
- Waterproof/withstand general weather

# Conceptual Design Diagram

Lun



Challenges:
- Light-Sensor:
  Grey-Scale Resolution (8bit?);
  High Dynamic Range;
- May customize the PCB for more compact photo-diode array and multiple sensors

System Level Challenge:
- Small-form-factor: Sandwich PCBs
- As thin as possible;
- As low-cost as possible.

Challenges:
- Software-Based Power Strategy
- HW-based Power Optimization
- Lower Power MCU?

Challenges:
- Configuration
- Low-Power
- Long-Range

Challenges:
- Sustainability (Power consumption/generation evaluation and optimization of each block);
- Form-factor (solar cell and battery as small and Thin as possible);
- Extract the maximum power (Design custom PCB w/ MPPT supported commercial chips);
- Efficient voltage regulation and battery Charging management.

Lun

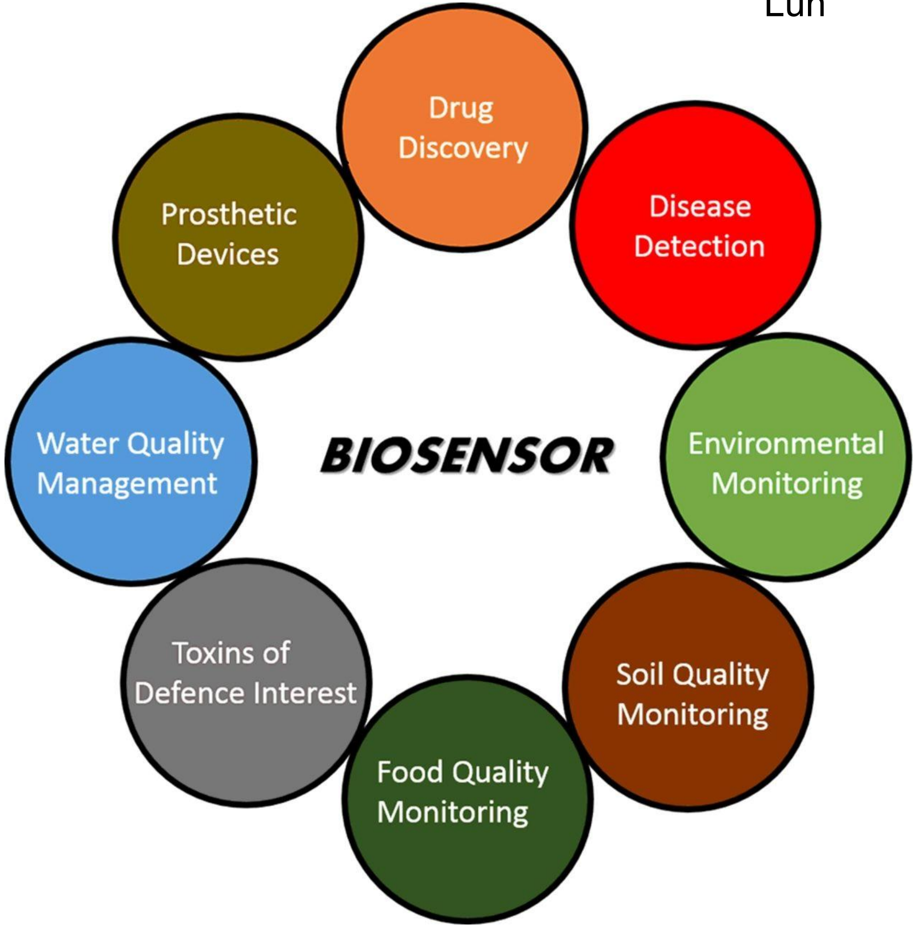
## Engineering Standards

- Organized and concise code.
- Clean circuit and PCB design.
- Organized and dated documentation
- Follow the IEEE code of ethics and environmental standards.

Lun

## Market Survey

- There are numerous IoT devices on the market.
- Ours will be:
- Self-sustainable (Solar powered)
- Pocket-sized
- Weather-resistant
- Biosensor
- Extremely specific usage (1 client)

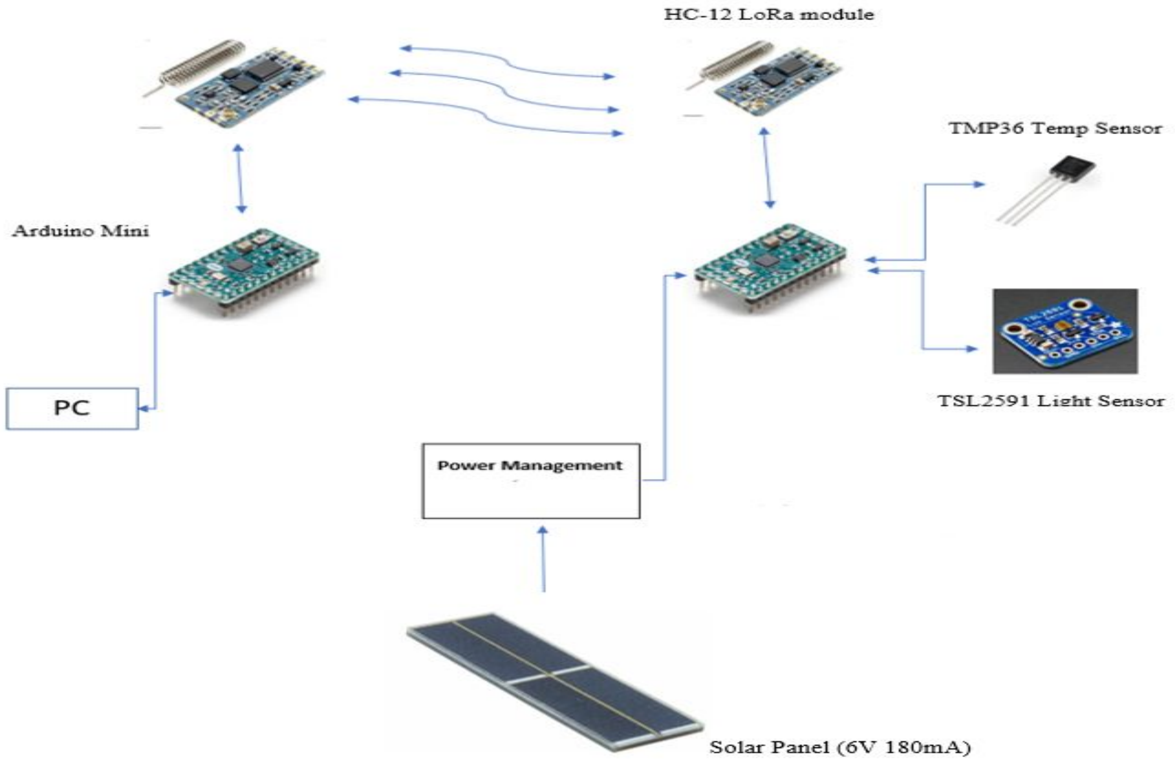


Lun

## Risks and Mitigation Plan

- Power: system relies on sunlight to charge and keep device functional.
  - Mitigation: we assume the longest the device will be without light is 24 hours. With that assumption, we can design the system to charge faster than the system consumes.
- Housing: we want our housing to be able to withstand outdoor weather and still operate properly.
  - Mitigation: a membrane (or seal) will cover the sensors to protect them and still operate normally.

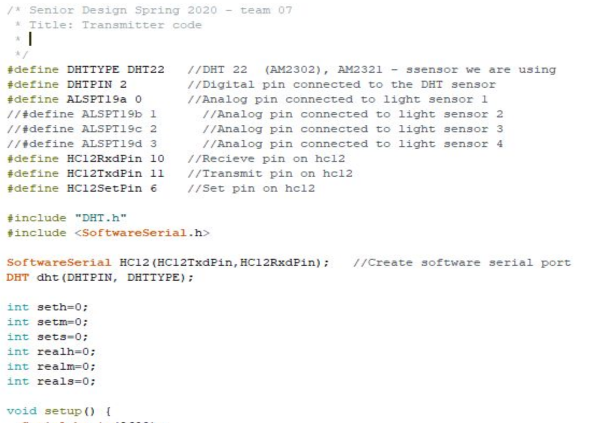
## Design Overview



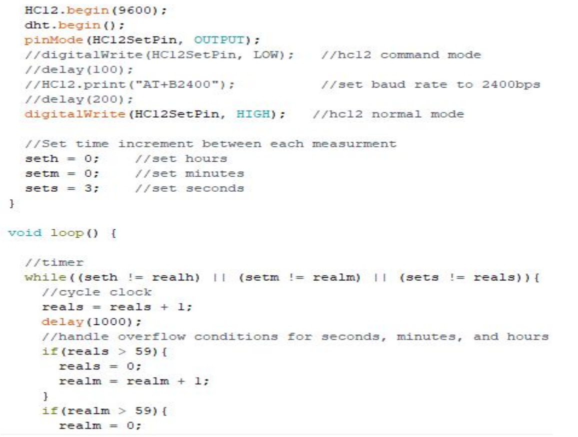
# Software Arduino

Qin

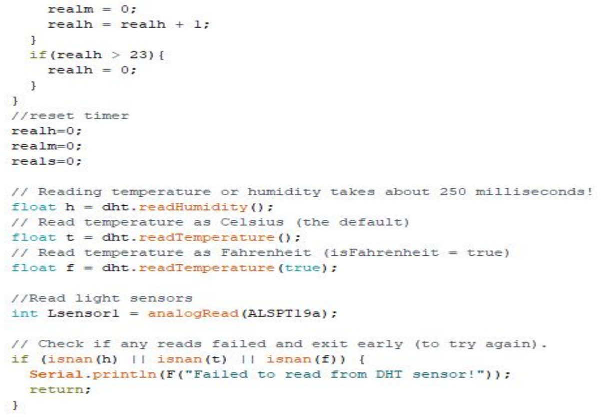
*Transmitter code 1 to 5 :*



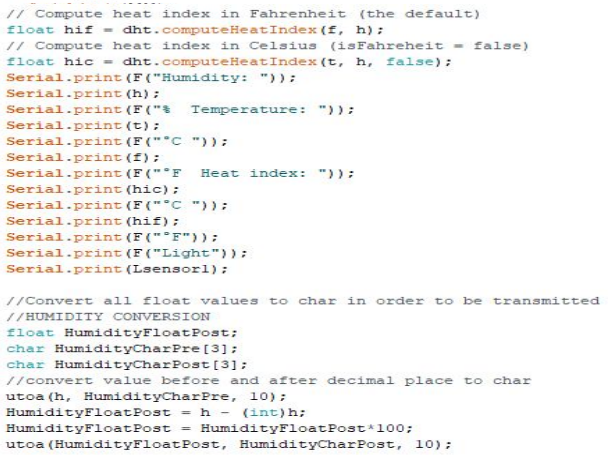
```
/* Senior Design Spring 2020 - team 07
 * Title: Transmitter code
 * |
 */
#define DHTTYPE DHT22   //DHT 22  (AM2302), AM2321 - ssensor we are using
#define DHTPIN 2        //Digital pin connected to the DHT sensor
#define ALSPT19a 0      //Analog pin connected to light sensor 1
//#define ALSPT19b 1      //Analog pin connected to light sensor 2
//#define ALSPT19c 2      //Analog pin connected to light sensor 3
//#define ALSPT19d 3      //Analog pin connected to light sensor 4
#define HC12RxdPin 10   //Recieve pin on hc12
#define HC12TxdPin 11   //Transmit pin on hc12
#define HC12SetPin 6    //Set pin on hc12

#include "DHT.h"
#include <SoftwareSerial.h>

SoftwareSerial HC12(HC12TxdPin,HC12RxdPin);   //Create software serial port
DHT dht(DHTPIN, DHTTYPE);

int seth=0;
int setm=0;
int sets=0;
int realh=0;
int realm=0;
int reals=0;

void setup() {
```



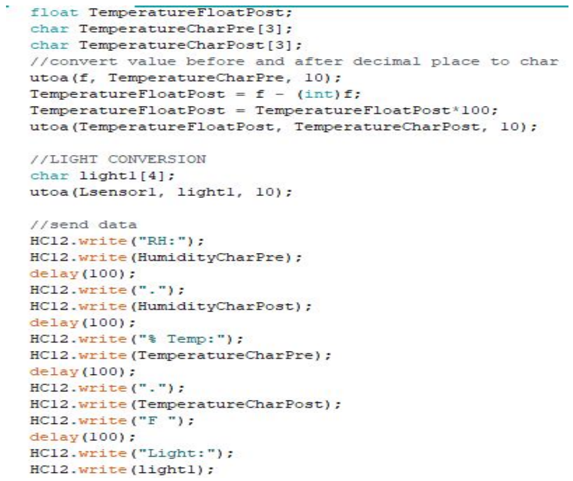
```
  HC12.begin(9600);
  dht.begin();
  pinMode(HC12SetPin, OUTPUT);
  //digitalWrite(HC12SetPin, LOW);   //hc12 command mode
  //delay(100);
  //HC12.print("AT+B2400");          //set baud rate to 2400bps
  //delay(200);
  digitalWrite(HC12SetPin, HIGH);   //hc12 normal mode

  //Set time increment between each measurment
  seth = 0;    //set hours
  setm = 0;    //set minutes
  sets = 3;    //set seconds
}

void loop() {

  //timer
  while((seth != realh) || (setm != realm) || (sets != reals)){
    //cycle clock
    reals = reals + 1;
    delay(1000);
    //handle overflow conditions for seconds, minutes, and hours
    if(reals > 59){
      reals = 0;
      realm = realm + 1;
    }
    if(realm > 59){
      realm = 0;
```



```
      realm = 0;
      realh = realh + 1;
    }
    if(realh > 23){
      realh = 0;
    }
  }
  //reset timer
  realh=0;
  realm=0;
  reals=0;

  // Reading temperature or humidity takes about 250 milliseconds!
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  //Read light sensors
  int Lsensor1 = analogRead(ALSPT19a);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
```

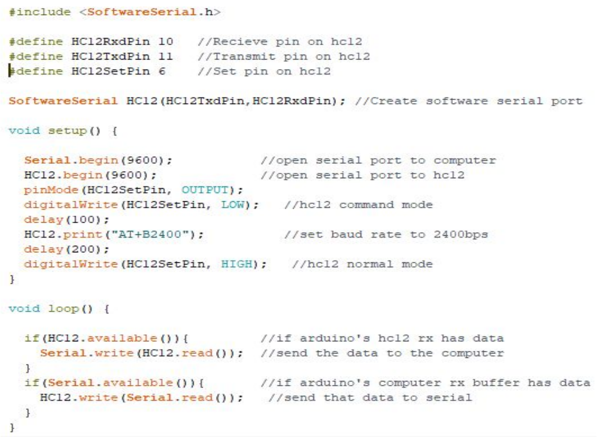


```
// Compute heat index in Fahrenheit (the default)
float hif = dht.computeHeatIndex(f, h);
// Compute heat index in Celsius (isFahreheit = false)
float hic = dht.computeHeatIndex(t, h, false);
Serial.print(F("Humidity: "));
Serial.print(h);
Serial.print(F("%  Temperature: "));
Serial.print(t);
Serial.print(F("°C "));
Serial.print(f);
Serial.print(F("°F  Heat index: "));
Serial.print(hic);
Serial.print(F("°C "));
Serial.print(hif);
Serial.print(F("°F"));
Serial.print(F("Light"));
Serial.print(Lsensor1);

//Convert all float values to char in order to be transmitted
//HUMIDITY CONVERSION
float HumidityFloatPost;
char HumidityCharPre[3];
char HumidityCharPost[3];
//convert value before and after decimal place to char
utoa(h, HumidityCharPre, 10);
HumidityFloatPost = h - (int)h;
HumidityFloatPost = HumidityFloatPost*100;
utoa(HumidityFloatPost, HumidityCharPost, 10);
```



```
float TemperatureFloatPost;
char TemperatureCharPre[3];
char TemperatureCharPost[3];
//convert value before and after decimal place to char
utoa(f, TemperatureCharPre, 10);
TemperatureFloatPost = f - (int)f;
TemperatureFloatPost = TemperatureFloatPost*100;
utoa(TemperatureFloatPost, TemperatureCharPost, 10);

//LIGHT CONVERSION
char light1[4];
utoa(Lsensor1, light1, 10);

//send data
HC12.write("RH:");
HC12.write(HumidityCharPre);
delay(100);
HC12.write(".");
HC12.write(HumidityCharPost);
delay(100);
HC12.write("% Temp:");
HC12.write(TemperatureCharPre);
delay(100);
HC12.write(".");
HC12.write(TemperatureCharPost);
HC12.write("F ");
delay(100);
HC12.write("Light:");
HC12.write(light1);
```

*Receiver cod:.*



```
#include <SoftwareSerial.h>

#define HC12RxdPin 10   //Recieve pin on hc12
#define HC12TxdPin 11   //Transmit pin on hc12
#define HC12SetPin 6    //Set pin on hc12

SoftwareSerial HC12(HC12TxdPin,HC12RxdPin); //Create software serial port

void setup() {

  Serial.begin(9600);             //open serial port to computer
  HC12.begin(9600);               //open serial port to hc12
  pinMode(HC12SetPin, OUTPUT);
  digitalWrite(HC12SetPin, LOW);   //hc12 command mode
  delay(100);
  HC12.print("AT+B2400");          //set baud rate to 2400bps
  delay(200);
  digitalWrite(HC12SetPin, HIGH);   //hc12 normal mode
}

void loop() {

  if(HC12.available()){           //if arduino's hc12 rx has data
    Serial.write(HC12.read());    //send the data to the computer
  }
  if(Serial.available()){         //if arduino's computer rx buffer has data
    HC12.write(Serial.read());     //send that data to serial
  }
}
```

## Hardware Overview

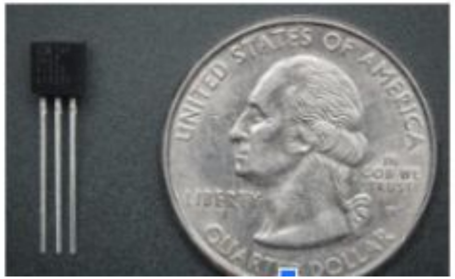
Qin

- Type: TSL2591
- Size: 4.72 x 2.95 x 0.39 inches
- Weight: 0.16 ounces
- Price: US$8.78

- Type: HC-12
- Size: 27.8 x 14.4 mm
- Low power
- Maximum communication distance: 1000m
- Price: US$4.81
- Antenna (430- 435MHz)

Arduino Mini Pro

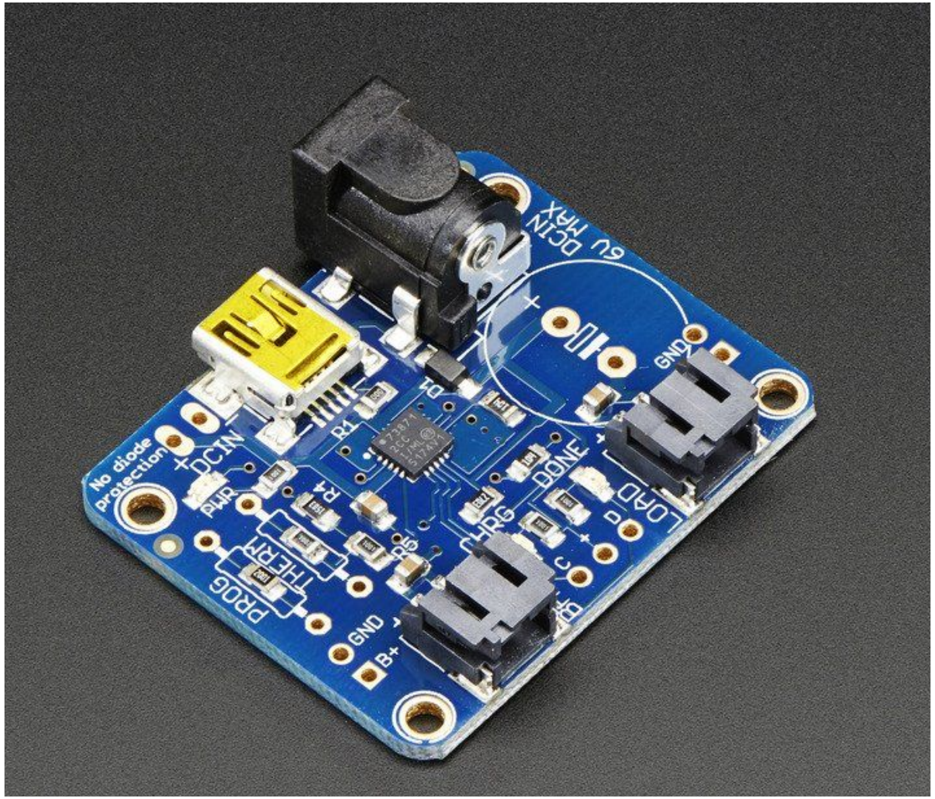
TMP36 Temperature Sensor



Calvin

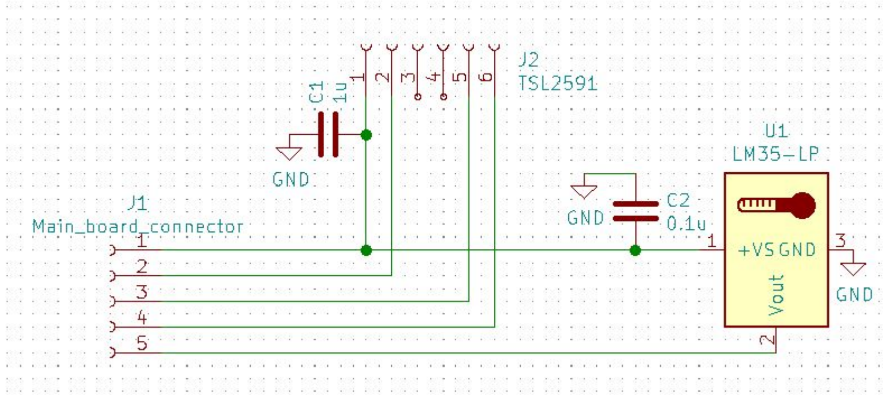
## Power

- Solar panel
  - 6V at 180mA
  - Dimensions: 89x113mm
- Batteries and charger
  - 2032 coin cell batteries
  - Li-Ion battery charger from Adafruit
  - Management similar to MPPT
- Regulators
  - MCP1700 linear regulator, 3.3V
  - MCP1801 linear regulator, 5V

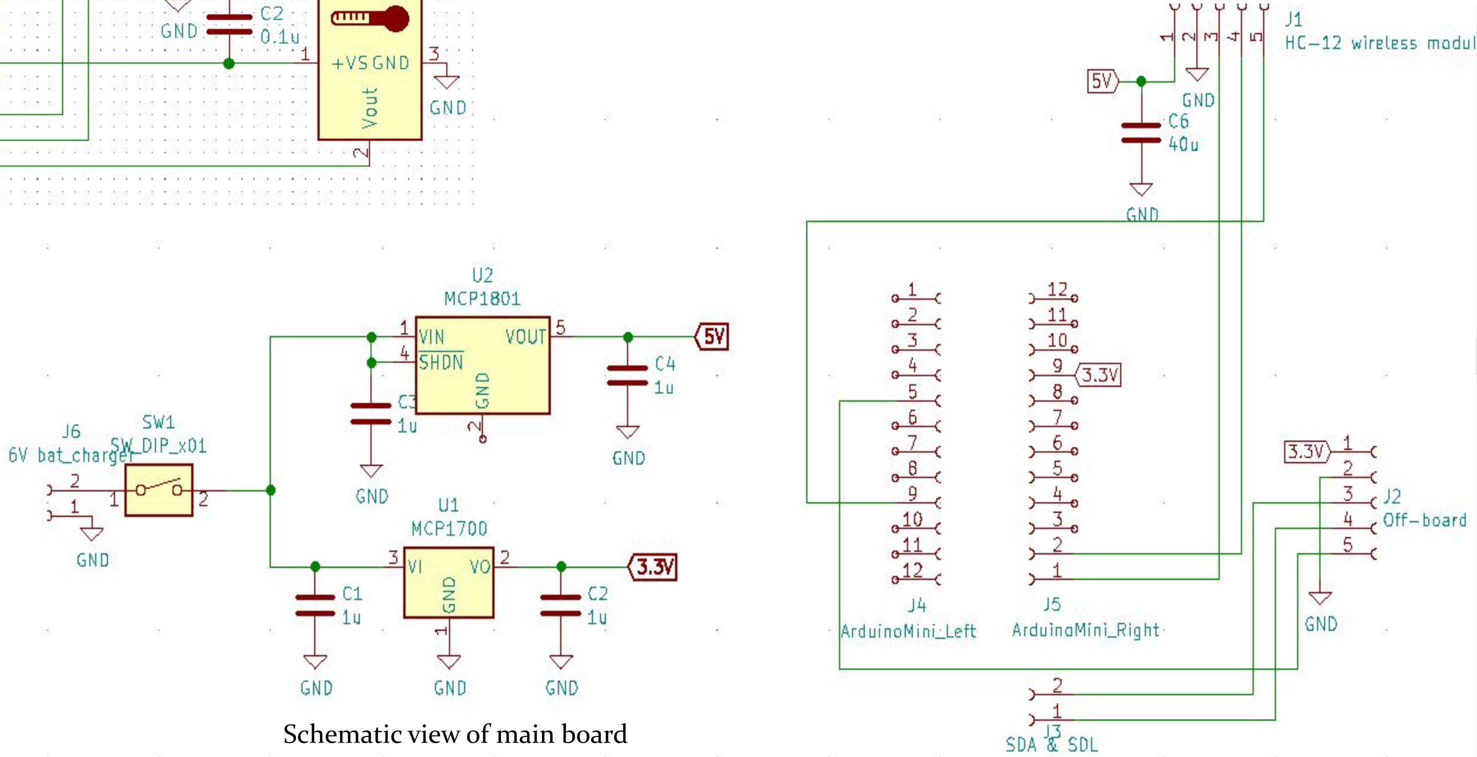


Calvin

# Schematic View



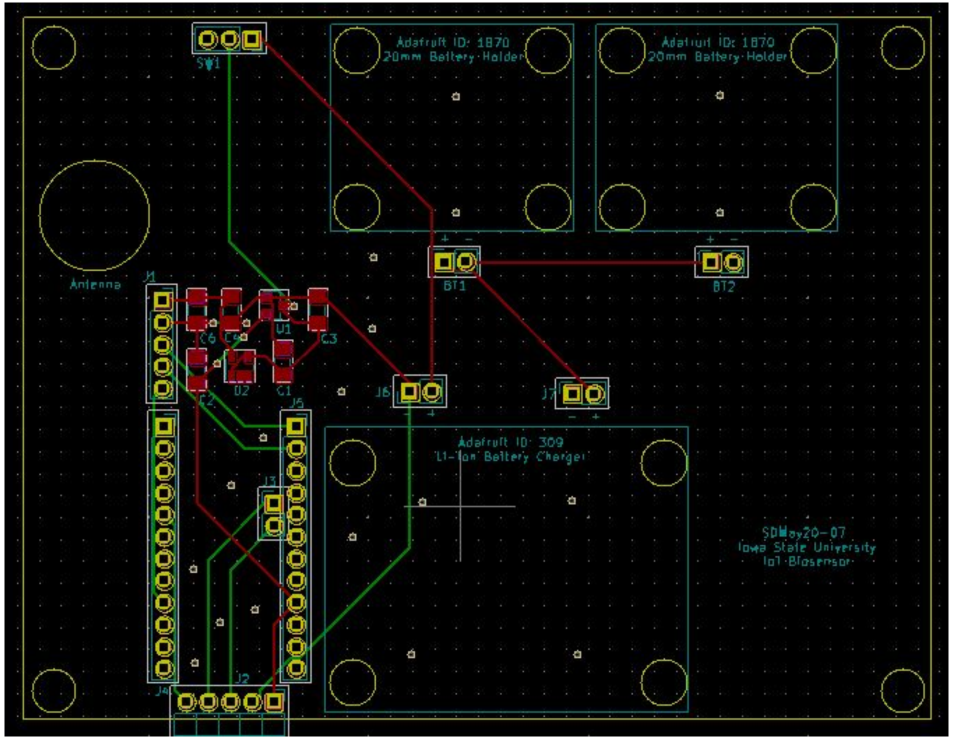
*Schematic view of sensor board.*



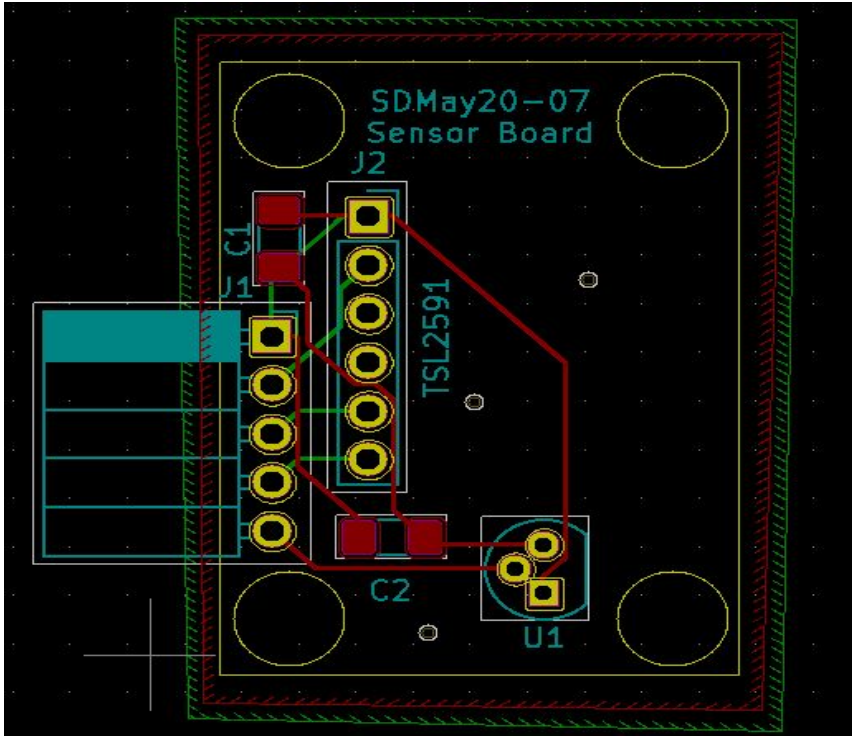
Schematic view of main board

# PCB View

Calvin



*Main PCB*

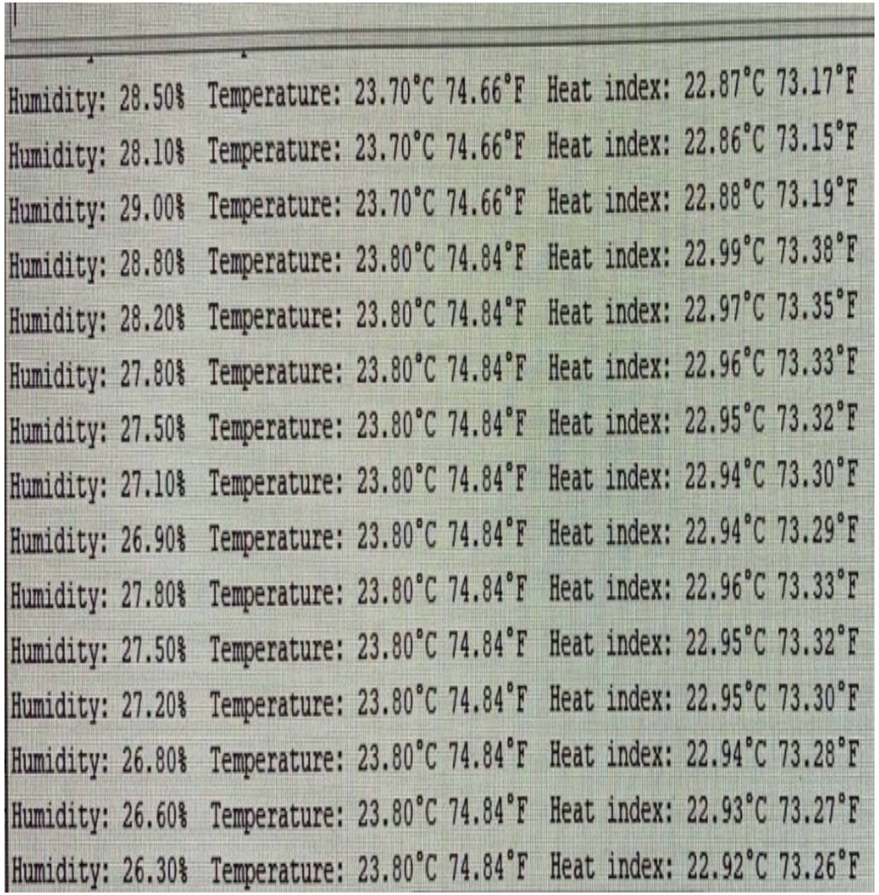


*Sensor PCB*

Yuchen

## Temperature Sensor Testing

- Wrote code using Arduino IDE to get frequent temperature readings (every 3 seconds).
- Compared sensor temperature readings with market thermometer for accuracy.
- Tested in multiple temperature ranges.
  - Room temperature.
  - Cold, outside in October.
  - Warm, using a air desolderer from  a distance.

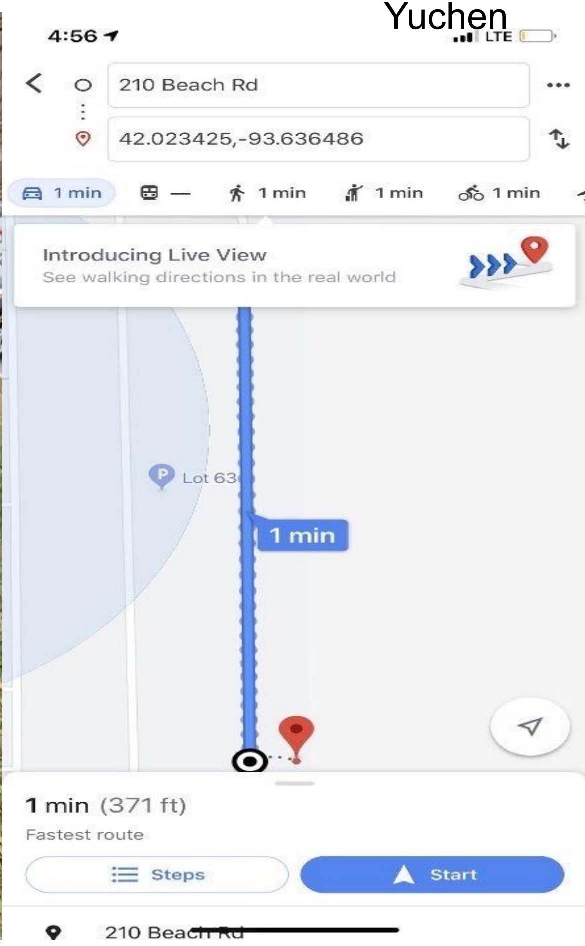


```
Humidity: 28.50%  Temperature: 23.70°C 74.66°F  Heat index: 22.87°C 73.17°F
Humidity: 28.10%  Temperature: 23.70°C 74.66°F  Heat index: 22.86°C 73.15°F
Humidity: 29.00%  Temperature: 23.70°C 74.66°F  Heat index: 22.88°C 73.19°F
Humidity: 28.80%  Temperature: 23.80°C 74.84°F  Heat index: 22.99°C 73.38°F
Humidity: 28.20%  Temperature: 23.80°C 74.84°F  Heat index: 22.97°C 73.35°F
Humidity: 27.80%  Temperature: 23.80°C 74.84°F  Heat index: 22.96°C 73.33°F
Humidity: 27.50%  Temperature: 23.80°C 74.84°F  Heat index: 22.95°C 73.32°F
Humidity: 27.10%  Temperature: 23.80°C 74.84°F  Heat index: 22.94°C 73.30°F
Humidity: 26.90%  Temperature: 23.80°C 74.84°F  Heat index: 22.94°C 73.29°F
Humidity: 27.80%  Temperature: 23.80°C 74.84°F  Heat index: 22.96°C 73.33°F
Humidity: 27.50%  Temperature: 23.80°C 74.84°F  Heat index: 22.95°C 73.32°F
Humidity: 27.20%  Temperature: 23.80°C 74.84°F  Heat index: 22.95°C 73.30°F
Humidity: 26.80%  Temperature: 23.80°C 74.84°F  Heat index: 22.94°C 73.28°F
Humidity: 26.60%  Temperature: 23.80°C 74.84°F  Heat index: 22.93°C 73.27°F
Humidity: 26.30%  Temperature: 23.80°C 74.84°F  Heat index: 22.92°C 73.26°F
```

# LoRa Testing

Yuchen

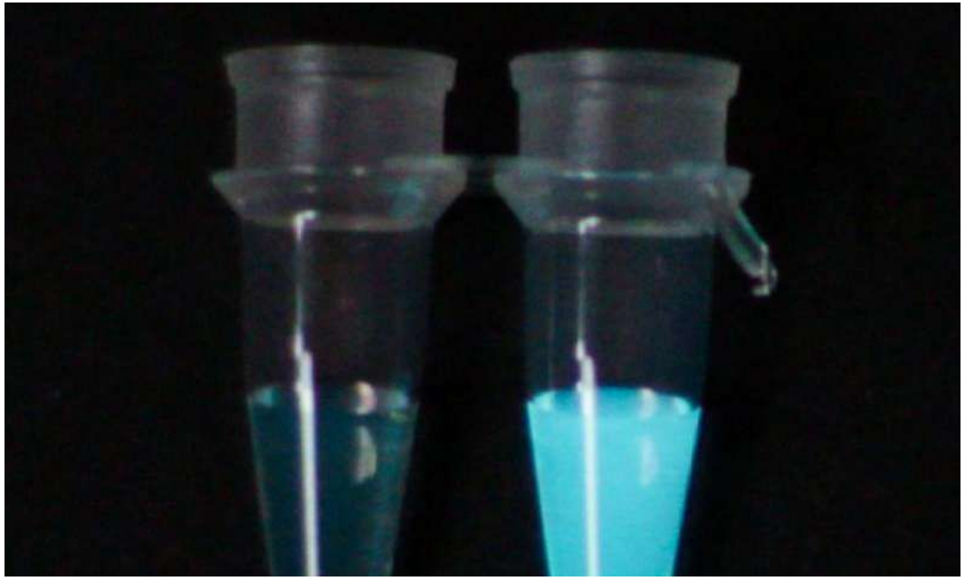
- Similarly, write code to transmit and receive values frequently.
- Set up the receiver in one location and move the transmitter away. Keep track of the distance using cell phone navigation app.
- Install delays in code to ensure complete data transmission.



Yuchen

## Light Sensor Testing

- Tested 2 different light sensors in Dr. Lu's biosensor lab.
- Used Enhanced Chemiluminescence (ECL) to create light that mimics that of bacteria.
- Light sensor was placed on top of a shoebox with an opening. The reaction was created using small capsules and then quickly placed in the box with the lights shut off.
- Experiment resulted in fluorescence of 0.054 lux.



Calvin

## Power Testing and Final Design Debugging

- Due to COVID-19, a few of our tasks were cut short.
    - For power, we relied on data provided from datasheets to find quiescent and functioning current values. From that data, we were able to estimate appropriate power component options.
    - For the final prototype, we wanted to test the power components before ordering the custom PCB. Instead, we chose skip the testing and order the PCB and hope our calculations were correct.
    - The final components and PCB have yet to arrive due to the slow down in processing mail.

Chuxin

## Team Contributions Summary

Calvin Condo

Process development with: light sensor, temperature sensor, LoRa module, and power management. Also worked on the PCB design and documentation.

Luke Healy

Light and Temperature sensor development and testing. Arduino functionality. Documentation

Qin Xia

Light sensor, temperature sensor, LoRa module testing. Arduino functionality.

Lun Zhang

Light and Temperature sensor testing. LoRa module testing. Final poster.

Yuchen Zhao

light sensor, LoRa module testing. Arduino functionality. Documentation

Chuxin Chen

Light sensor, temperature sensor, LoRa module testing. Final poster.

Chuxin

## Lessons Learned

- Wireless communication can be unpredictable, so you should implement extra measures in the software to insure complete transmissions.
- There are a lot of available components in the market - choosing the right one for the project can be difficult. Learn from understanding all the options and from experience.
- Testing in conditions as close to actual function is essential. If we had not done light sensor test, our original light sensor choice would not have worked.

Chuxin

## Possible Improvements

- Implement a gateway to access data from a website or mobile app.
- Optimize the device from breakout boards to use only components that are needed and make the device smaller.
- Better power optimization.

Chuxin

## Conclusion

We have successfully designed a self-sustainable IoT biosensor with wireless communication for Dr. Lu’s research. The biosensor can measure temperature and light and transmit the data to a receiver. The system stays powered by using a solar panel and optimizing charging two Li-Ion batteries. Due to the current situation, we are not be able complete or test the physical final prototype. However, we are proud for what we have accomplished and will continue to finish the project once things go back to normal!

Thank you for your attention!

Questions?