# Wireless Communication for Greenhouse Environment Monitoring

Chris Lambert, Dean Lockman, Ian Nielsen

Advisor: Dean Arakaki

June 2017

# Contents

# List of Figures

# List of Tables

# 1 Acknowledgements

First and foremost we would like to thank our advisor, Dr. Arakaki, for his support and wisdom throughout the project design. He was instrumental in explaining the antenna intricacies we knew little about, such as balun physics and antenna construction. We would also like to thank Dr. Derickson for his help in creating the fund for senior project groups as well as facilitating any questions for paperwork submissions. Without their help we would not have been able to complete the project so smoothly and completely.

# 2 Introduction

Modern day population growth and global climate change requires high-yield agricultural systems that can function in harsh environments. Greenhouses can maintain temperature and humidity, while serving as a protective, nurturing alternative to vulnerable farm environments. With weather effects eliminated, greenhouses provide productive environments for experimental agriculture techniques such as hydroponics and aquaculture. These methods of agriculture, specifically aquaculture, require constant monitoring. The failure of electronic devices such as an air pump prove fatal to a fish population in a short time frame. In developing countries, agricultural locations may be a significant distance from a farmer's home, with no internet or utility connections. Remotely located greenhouses utilizing aquaculture methods would benefit from long range, wireless monitoring systems that alert farmers to detrimental growth conditions including unstable pH, low dissolved oxygen content, and extreme temperatures. This project focuses on creating a dual end monitoring system with multiple functions. Sensor-end module functions include:

1. Recording greenhouse environment metrics including humidity and air and water temperatures.
2. Monitoring the greenhouse power line and off grid renewable sources alerting the user of complications.
3. Relaying data to a server-end module through internet connection.

This system must perform without error at 1 km range over a largely agricultural region with variable topography, and in rainy and arid weather conditions. This is achieved using LoRa, a low power radio based wireless platform. System function is verified using:

1. Two LoRa-capable microcontrollers
2. Two computers with virtual USB serial connections
3. Temperature and humidity probes
4. Voltage sensor module
5. Multimeter and oscilloscope

6. Design, construction, and optimization of RF antennas are performed to ensure functionality over communication ranges of 1-5 km.

# 3 System Description

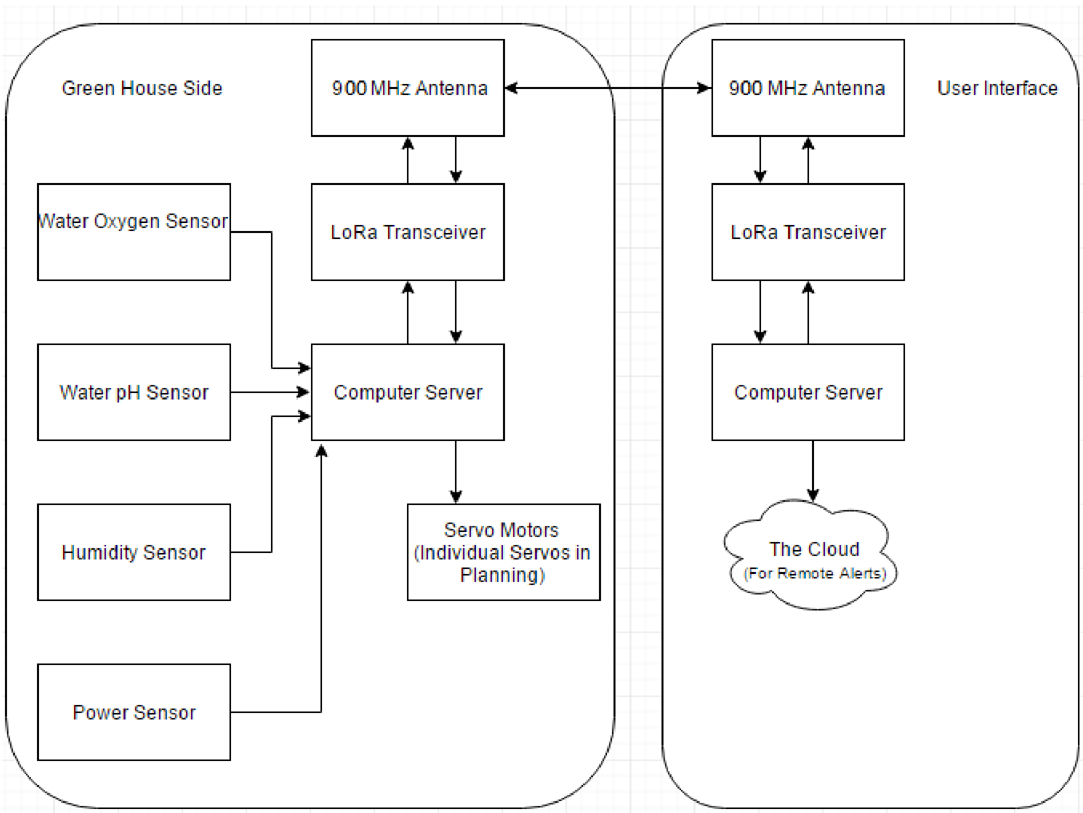


Figure 1: *PolyPonics Wireless System Top-Level Block Diagram*

# 4 Antenna Development

Two initial antenna designs are simulated to determine the most cost-effective and best-range option: patch and Yagi-Uda. The proposed antenna location is on Building 11, the Agricultural Sciences building, for its line of sight view to the greenhouse. Building 11 and the greenhouse are 1.0 km apart. This is the range for inter-station communications. The design was developed using HFSS: High Frequency Simulation Software.[1]

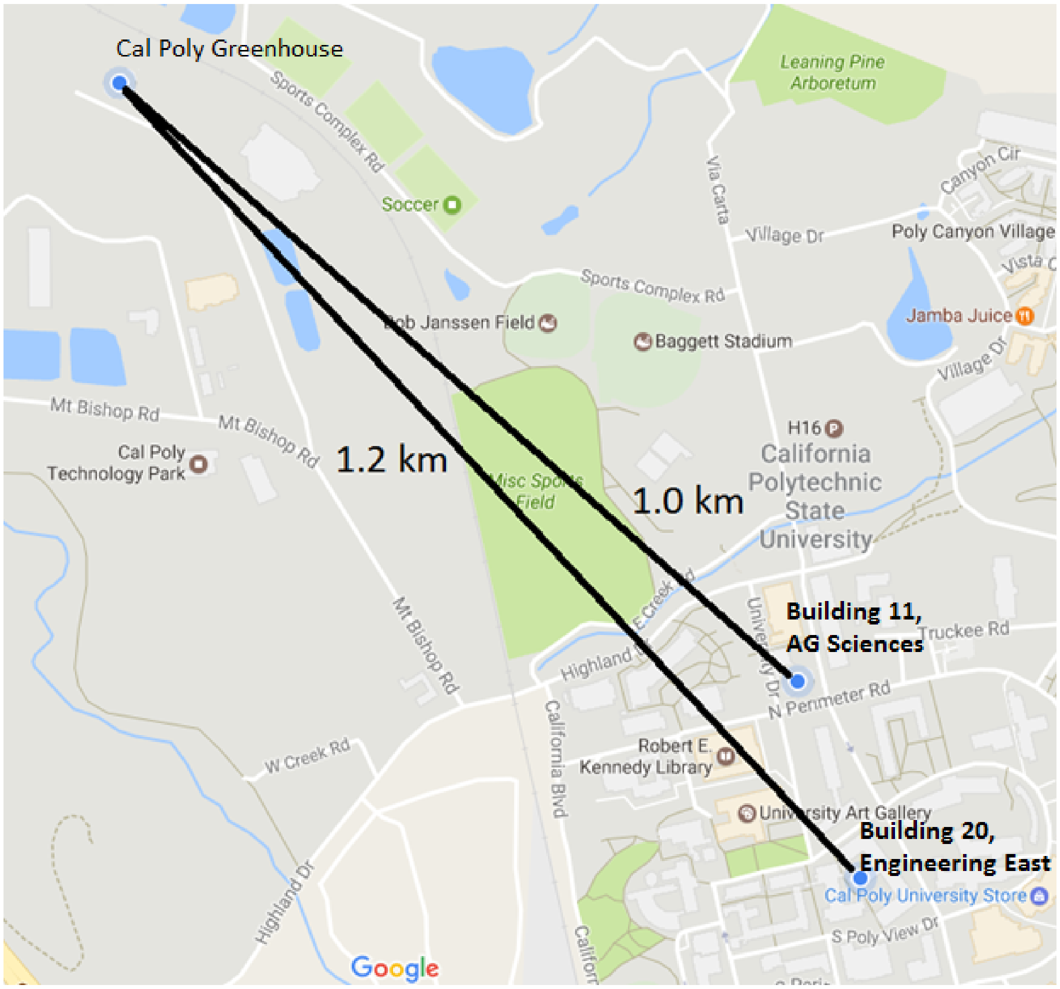


Figure 2: *Campus Map: Cal Poly's Greenhouse, Agriculture Science Building (11), Electrical Engineering Building (20)*

## 4.1 Patch Antenna Design

The design frequency for all antennas is 900 MHz. Patch length, width, and feed length are calculated using the following equations.

$$f_c = \frac{c}{2L\sqrt{\epsilon_r}} \tag{1}$$

$$B \approx \frac{(\epsilon_r - 1)W}{\epsilon_r^2 L} h \tag{2}$$

B, L, W, h, and $\epsilon$ are bandwidth, patch length and width, substrate thickness, and dielectric constant, respectively. From Equations 1 and 2, with a dielectric constant 4.4 for FR-4 board material, design substrate thickness 1.6 mm, and 900 MHz operating frequency, the patch design length L is 78.1 mm. The LoRa modules require a bandwidth of 500 kHz for standard transmission, hence the design width W is 151 mm. The microstrip inset feed length is a quarter wavelength at 900MHz. To improve $|S_{11}|$ at 900 MHz to less than -10 dB, patch width and length is tuned to 104.8 mm and 76.9 mm respectively. Measurements recorded in Table I, HFSS model in Figure 3.

Table I: *900 MHZ PATCH SIMULATION PARAMETERS*

| Antenna Specific Measurement | Length (mm) |
|---|---|
| Width | 104.8 |
| Length | 76.9 |
| Substrate Thickness | 1.6 |

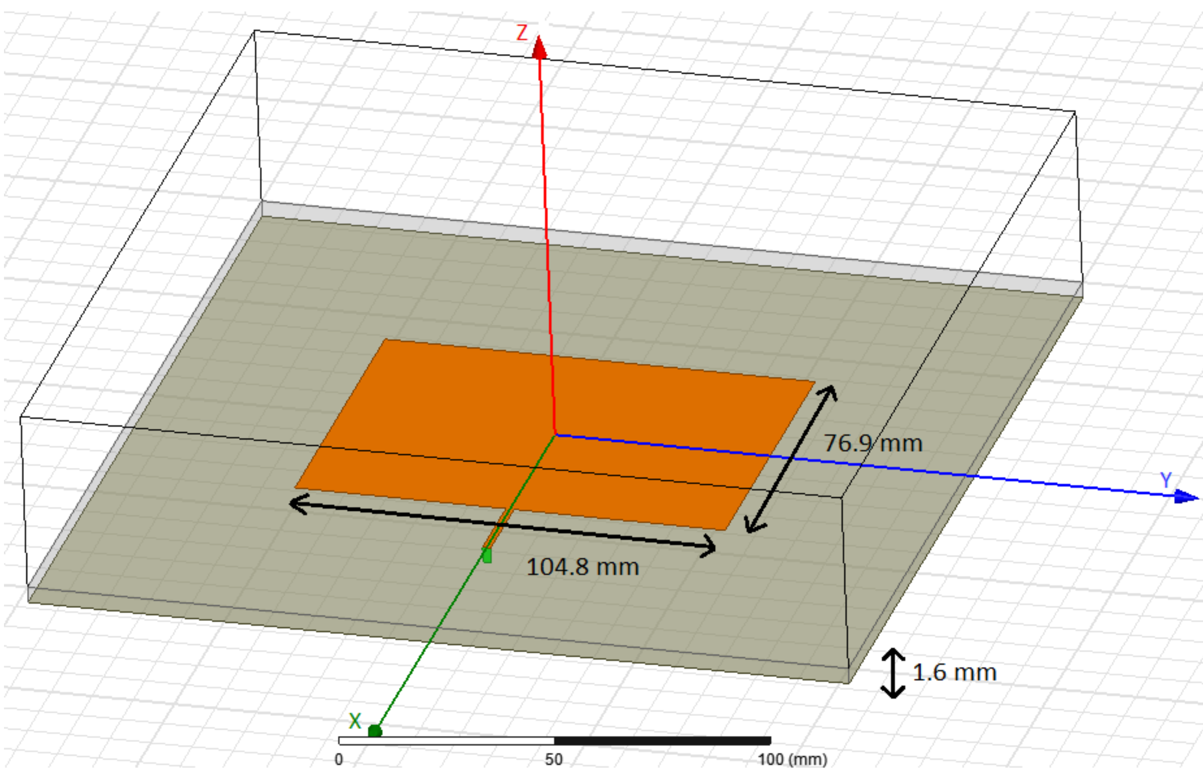


Figure 3: *HFSS 3D Model, 900 MHz Patch Antenna, Microstrip Feed*

## 4.2 Conclusions, Patch Antenna Design

Figure 4 shows -18.2 dB at 900 MHz. From Figure 5, the patch antenna has 1.9 dBi gain and is not selected since it does not meet the minimum 4.0 dBi gain requirement for the 1 km range.

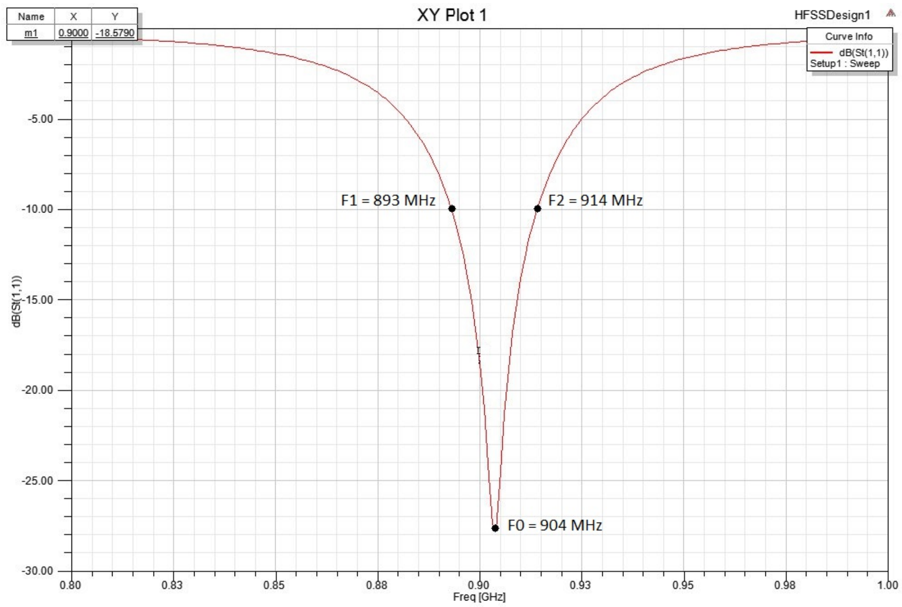


Figure 4: $|S_{11}|$ *vs. Frequency, 900 MHz Patch Antenna, 0.8-1.0 GHz, Nodal Excitation*

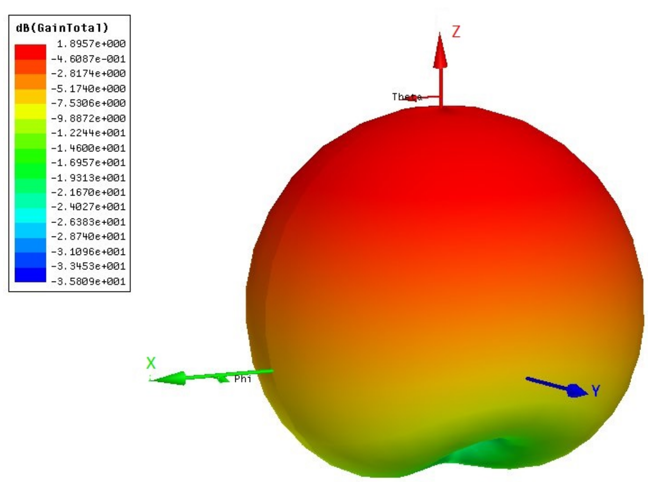


Figure 5: *3D Gain, 900 MHz Patch Antenna. CW 900 MHz, Nodal Excitation*

## 4.3 Yagi-Uda Design

The Yagi-Uda antenna exhibits considerable directional gain with affordable implementation. Table II defines expected Yagi-Uda antenna gain performance.

Table II: *APPROXIMATE YAGI-UDA ANTENNA GAIN LEVELS*[2]

| Number of Elements | Anticipated Gain dBi |
|---|---|
| 2 | 5.0 |
| 3 | 7.5 |
| 4 | 8.5 |
| 5 | 9.5 |
| 6 | 10.5 |
| 7 | 11.5 |

A four element design is chosen with a half wave dipole antenna as the driven element with one reflector and two directors as parasitic elements. Dipole antennas are preferable to use in Yagi-Uda designs due to their toroidal radiation pattern. The selected dipole type is based on a center-fed half-wave dipole design, due to ease of construction and well-known efficiency. This dipole design is efficient due to current distribution along the two quarter wavelength elements. Using Equation (3), the antenna wavelength as a function of frequency is 333 mm.

$$\lambda = \frac{c}{f} \tag{3}$$

Dimensions are chosen and modeled for the design based on guidelines outlined in the ARRL handbook[3], see Table III for dimensions, Figure 6 for model.

R = Universal Rod Radius = 3.175 mm

$\lambda$ = Wavelength = 327.868 mm

Table III: *900 MHZ YAGI-UDA CONSTRUCTION PARAMETERS*

| Antenna Specific Measurement | Reference Designator | Length (mm) | Electrical Length ($\lambda$) |
|---|---|---|---|
| Reflector | L1 | 153.0 | 0.466 |
| Driven Element | L2 | 142.0 | 0.460 |
| 1st Director | L3 | 135.0 | 0.412 |
| 2nd Director | L4 | 135.0 | 0.412 |
| Reflector to Driven Element | S1 | 45.3 | 0.138 |
| Driven Element to 1st Director | S2 | 78.4 | 0.239 |
| 1st to 2nd Director Distance | S3 | 57.7 | 0.176 |
| Universal Rod Diameter | R | 1.2 | 0.176 |

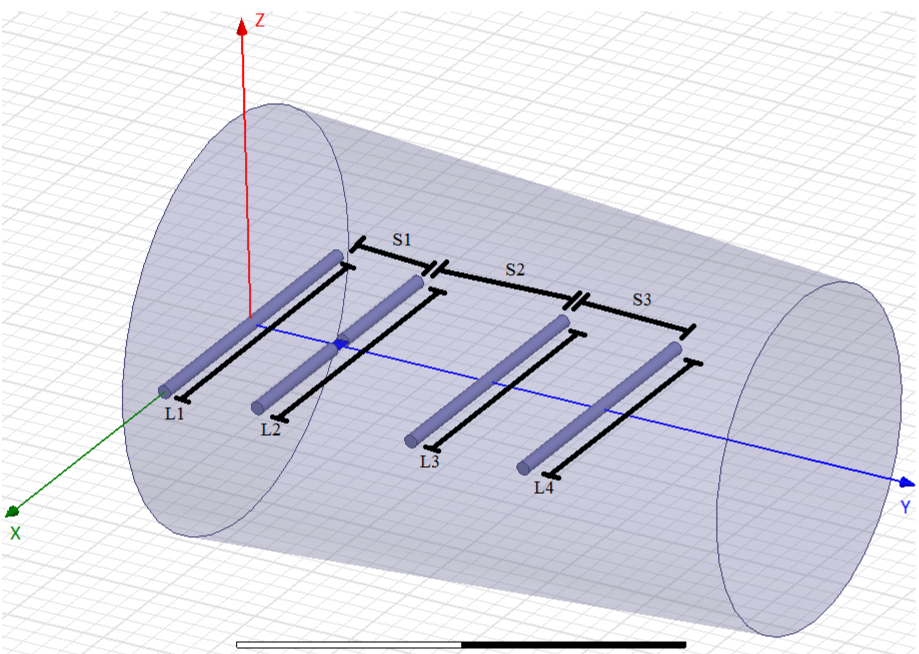


Figure 6: *HFSS Generated 3D Model of 900 MHz Yagi-Uda*

Frequency and gain plots are compiled from HFSS, see Figures 7 and 8 respectively.

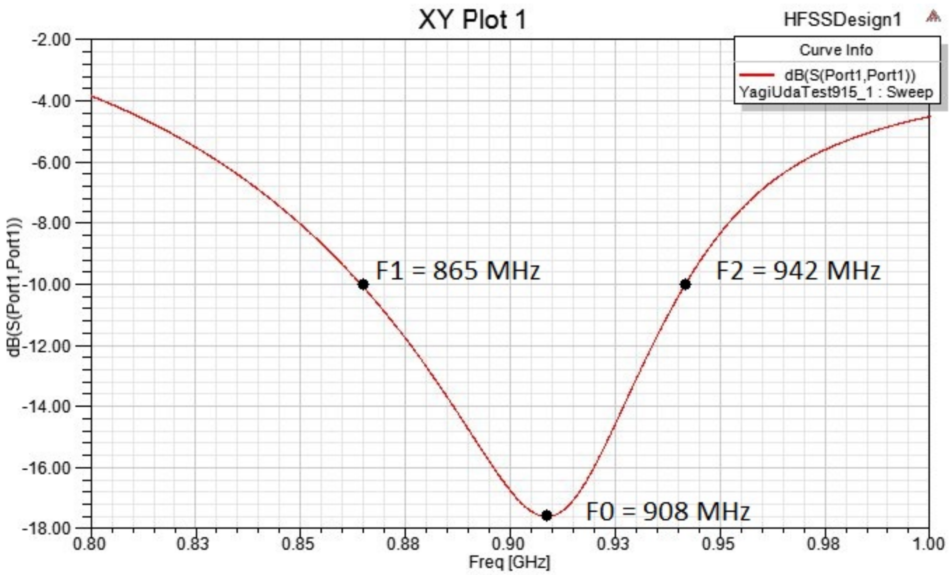


Figure 7: $|S_{11}|$ *vs Frequency (GHz), Yagi-Uda Prototype*

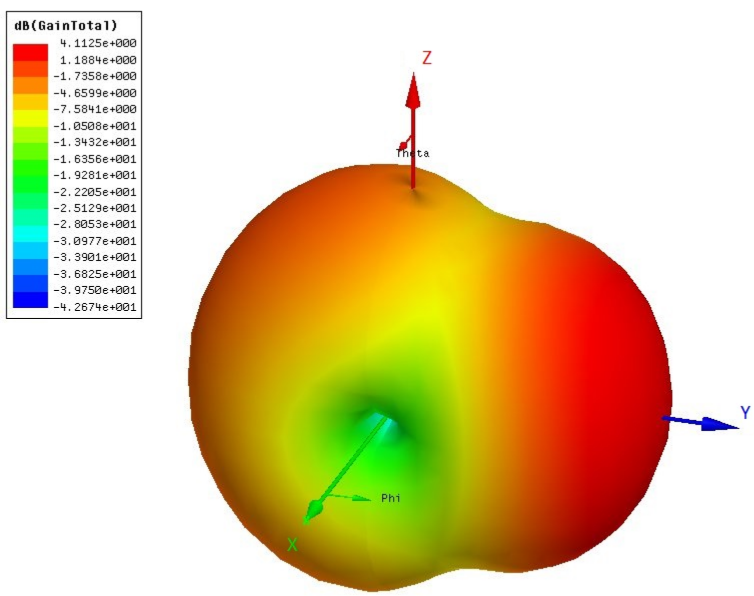


Figure 8: *3D Far Field Pattern, 900 MHz Yagi-Uda*

## 4.4 Conclusions, Yagi-Uda Design

Based on HFSS results in Figure 8, a gain of 4.1 dB is adequate for the project specifications. The Yagi-Uda is chosen as the prototype antenna.

# 5 Construction And Design Tradeoffs

## 5.1 LoRa Module Development

LoRa (Long Range Radio Antenna) modules were purchased for testing. The LoRa radios selected for this project are developed on a Feather M0 board[4] which provides an interface to the radio by adding a programmable microcontroller. Complete specifications are defined in the appendix. Code is created using functions from RadioHead's library, included in Appendix A - (open source libraries for LoRa radio development)[5] and the two modules are tested with quarter-wave whip antennas, see Figure 9. Code is included in the appendix for both the transmitter and the receiver. Communication field testing determines device range with quarter-wave antennas, see Figure 9.

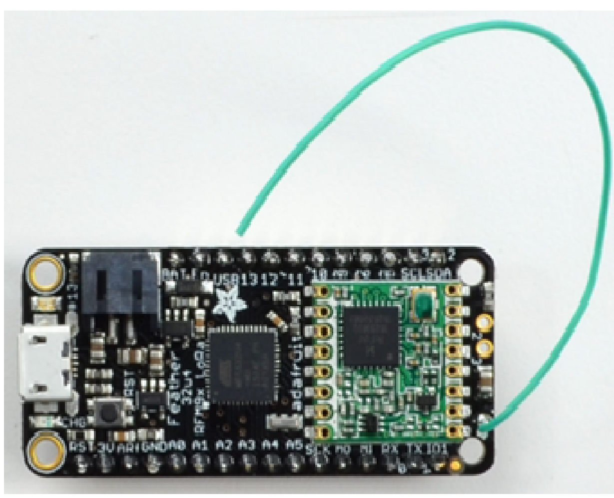

Figure 9: *LoRa 900 MHz Radio Module on Feather M0 Board with Quarter-Wave Whip*

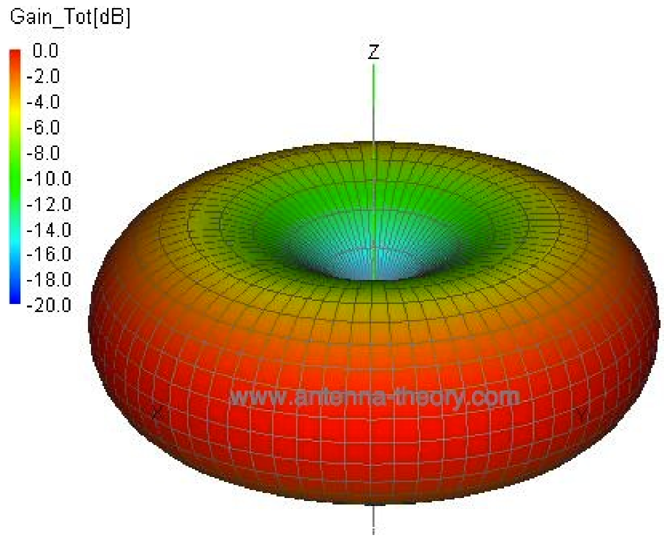


Figure 10: *3D Gain, Dipole Antenna*[6]

A quarter-wave antenna radiates uniformly around the central axis, shown in Figure 10. Quarter-wave antennas can reach the 1.0 km distance between Cal Poly's greenhouse and Cal Poly's Building 11 with an RSSI reading of -110-unitless (-100dBm), according to the LoRa modules datasheet[4]. According to the Friis equation[7] with a radio set to 23 dBm transmit power, the gain of the quarter-wave whip antennas is 4.6 dB.

$$P_r = P_t + G_t + G_r + 20log_{10}(\frac{\lambda}{4\pi R}) \tag{4}$$

Therefore the transmit and receive gain of each half-wave dipole antenna at 900 MHz is 2.3 dBi. Gain is assumed equal for both antennas since dimensions and wire thicknesses are identical. The received power at maximum range is -110 dBm. A directional antenna with 4 dB gain in the desired direction results in a received signal strength at the greenhouse of -64 dBm. Yagi-Uda simulation data predicts 4.7 dBi.

## 5.2 Balun Construction

In order to match a balanced load to an unbalanced input signal, baluns are constructed from brass tubing. The balun increases antenna gain by reducing the current on the coaxial cable's outer conductor thus increasing radiation in desired directions.

## 5.3 Split Balun

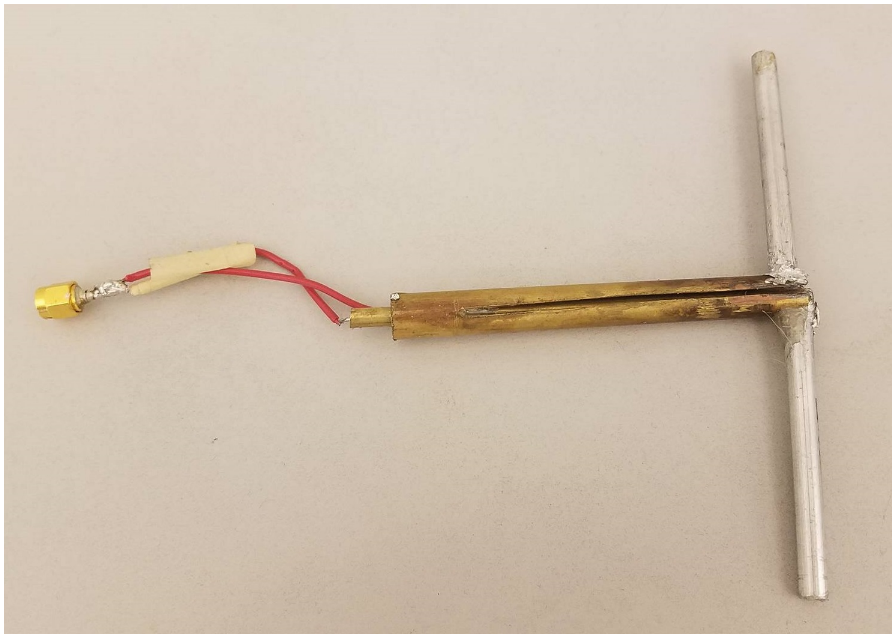

Figure 11: *Split Baluns with Active Dipoles*

In order to mechanically and electrically connect the brass balun to the aluminum dipole, Super Alloy 1[8] solder is required. Cal Poly's Industrial and Manufacturing Engineering Department provided the solder. The IME department advised using low heat, around 200°C typical of soldering irons, to avoid melting the hollow brass rod. A balun and dipole are constructed: Figure 11. Tests are conducted on the completed antenna in the anechoic chamber.

Figure 12 shows the measured reflections over the 800 MHz to 1 GHz range. There is a slight match at 900 MHz, but it does not meet the -10 dB reflection requirement.

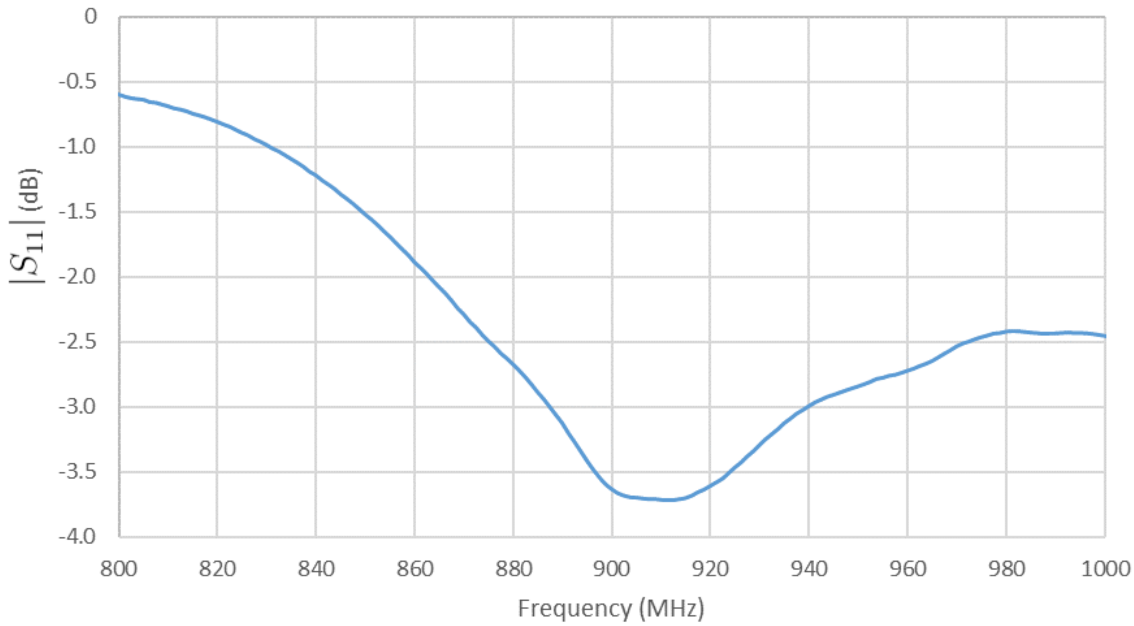


Figure 12: $|S_{11}|$ *vs. Frequency (MHz), Split Balun*

## 5.4 Sleeve Balun

Since the split balun was unable to reach the -10 dB reflection requirement, a sleeve balun was constructed, shown in Figure 13.

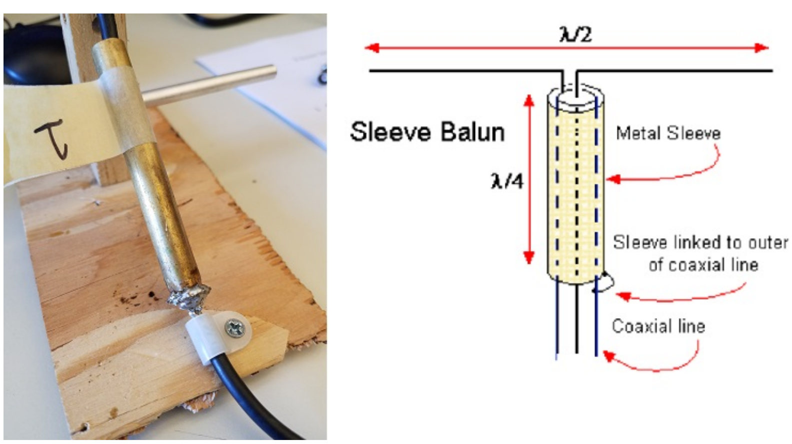


Figure 13: *Constructed Sleeve Balun and Current Diagram*[9]*, 900 MHz*

Sleeve balun test results in Figure 14 show $|S_{11}|$ greater than -10 dB at 900 MHz. The design goal of -10 dB is met.

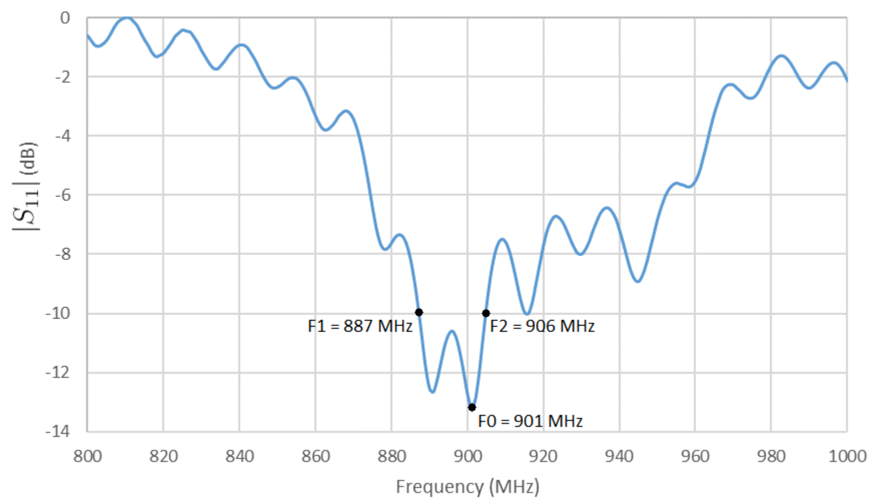


Figure 14: $|S_{11}|$ *vs. Frequency (MHz), Sleeve Balun*

# 6 Test Results

Final constructed antenna with sleeve balun shown in Figure 15. Tests were performed inside Cal Poly's Electrical Engineering Department's Anechoic Chamber. A calibrated vector network analyzer is used for all reflection measurements.

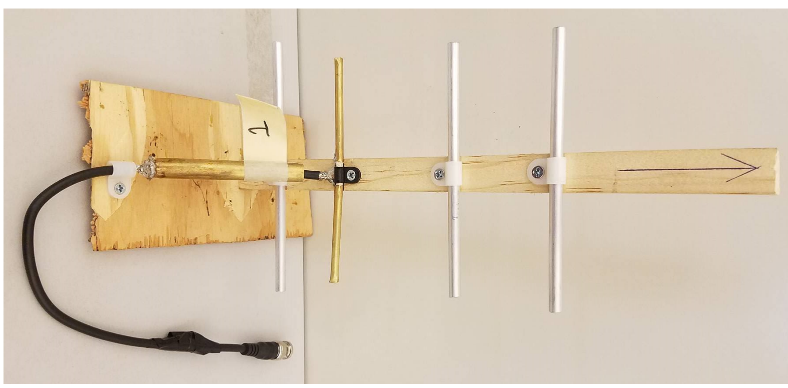

Figure 15: *Final Constructed Yagi-Uda, 900 MHz*

## 6.1 Antenna Reflection Coefficient Comparison

The following $|S_{11}|$ data is recorded for both antennas; labeled "1" and "0". Figure 16 (Antenna "0") shows a reflection of -10 dB at 900 MHz. Matched responses also occur at resonant frequencies 800 MHz and 1 GHz.

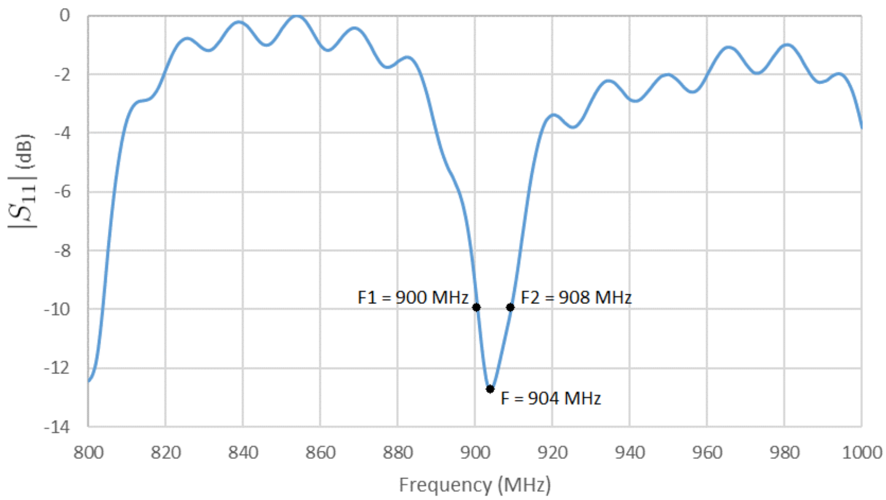


Figure 16: $|S_{11}|$ *vs. Frequency (MHz), Antenna "0"*

Figure 17 (Antenna "1") shows a reflection of -13 dB at 900 MHz. Matching also occurs at resonant frequencies 700 MHz and 1.1 GHz.

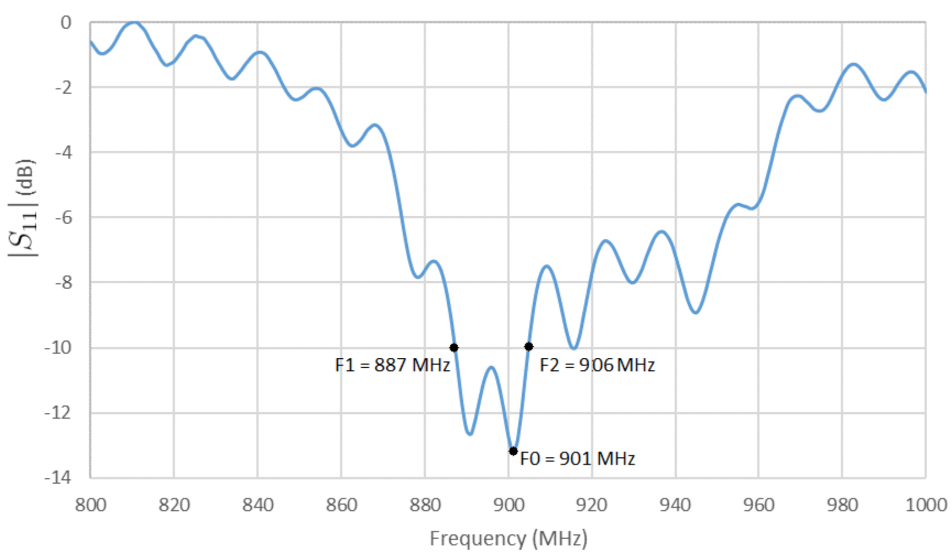


Figure 17: $|S_{11}|$ *vs. Frequency (MHz), Antenna "1"*

The design is optimized in order to obtain the greatest amplitude reflections at the design frequency. $|S_{11}|$ and bandwidth measurements are recorded and compared to simulation in Table IV and Table V.

Table IV: *MEASURED AND SIMULATED* $|S_{11}|$ *COMPARISON*

| Antenna | Frequency (MHz) | Measured $\|S_{11}\|$ (dB) | Simulated $\|S_{11}\|$ (dB) | Difference (dB) |
|---|---|---|---|---|
| "0" | 900 | -10.2 | -18.2 | 8.2 |
| "1" | 900 | -13.0 | -18.2 | 5.2 |

Table V: *MEASURED AND SIMULATED BANDWIDTH COMPARISON*

| Antenna | Frequency (MHz) | Measured Bandwidth (%) | Simulated Bandwidth (%) | Difference (%) |
|---|---|---|---|---|
| "0" | 900 | 0.88 | 2.29 | 1.41 |
| "1" | 900 | 3.13 | 2.29 | 0.83 |

## 6.2 Antenna Gain Comparison

The antennas were set up inside the chamber; the vector network analyzer transmitting 900 MHz at 0 dBm through port 1. Output from the vector network analyzer is measured at -11.04 dBm. The signal at the receive antenna is -32.22 dBm. The measured distance inside the anechoic chamber from antenna "1" to antenna "0" is 1.5 m. Assuming identical gain values, the gain is calculated using Equation (4). Therefore the two antennas have approximately 6.9 dB gain each. The $|S_{11}|$ plot of antenna "1" shows that it is a better matched antenna and therefore has a higher gain. Antenna "1" is used for the transmission side of the greenhouse base.

## 6.3 Antenna Radiation Comparison

The co-pol radiation pattern for the E planes of the simulated antenna and antenna "1" are displayed in Figures 18 and 19. The half power is calculated as the difference in angles between the two half power points from the main lobe.

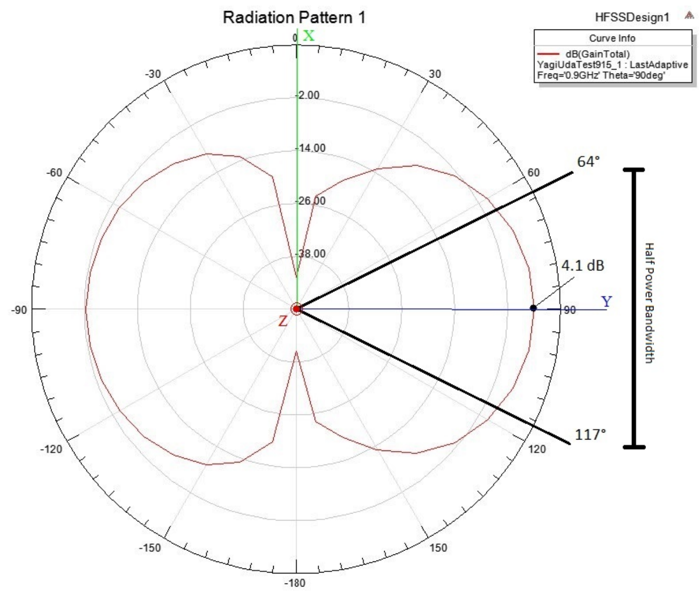


Figure 18: *E-Co Radiation Pattern, Simulated Yagi-Uda, 900 MHz*

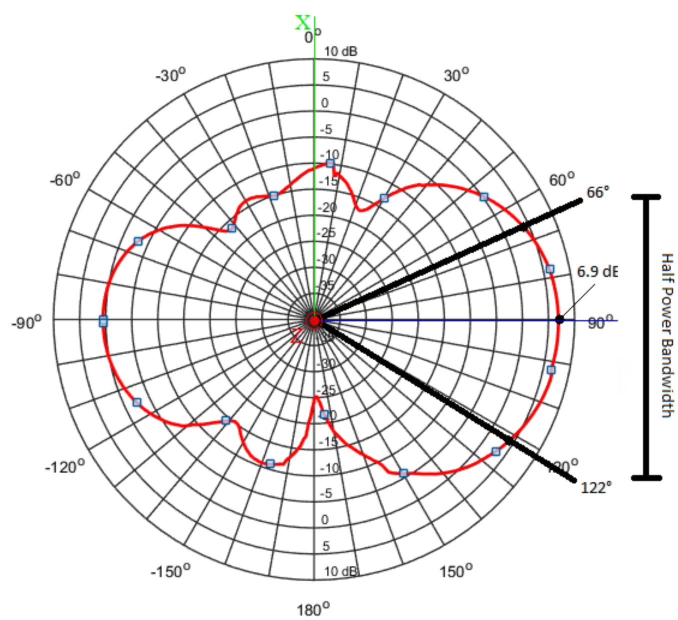


Figure 19: *E-Co Radiation Pattern, Constructed Yagi-Uda, 900 MHz*

Table VI shows the half power beamwidth of simulated and constructed antennas. The beamwidth allows for error in the project because there is some imprecision in lining up the two antennas at a 1 km distance.

Table VI: *MEASURED AND SIMULATED RADIATION PATTERN COMPARISON*

| Antenna | Frequency (MHz) | Half-Power Beamwidth Measured (°) | Half-Power Beamwidth Simulated (°) | Difference (°) |
|---|---|---|---|---|
| "1" | 900 | 56 | 53 | 5 |

## 6.4 Losses

To measure transmit to receive loss, the following experiment was configured. A LoRa radio set to +23 dBm transmit power through antenna "1" is mounted on Cal Poly's Greenhouse. The second LoRa Radio with antenna "0" is 1.5 km away at Cal Poly's EE Department's Microwave Laboratory. Transmitter signal strength of the LoRa module was measured at +22.7 dBm. Receiver signal strength was measured at -103 dBm. Calculated losses recorded in Table VII.

Spreading Loss:

$$20log_{10}(\frac{\lambda}{4\pi R}) \tag{5}$$

External Loss:

$$P_r - P_t - G_t - G_r - 20log_{10}(\frac{\lambda}{4\pi R}) \tag{6}$$

Table VII: *LOSS ANALYSIS*

| Transmit Power (dBm) | Received Power (dBm) | Spreading Loss (dB) | Antenna Gain (dB) | External Loss (dB) |
|---|---|---|---|---|
| 23 | -103 | 95.13 | 6.9 | 44.57 |

These losses are attributed to polarization mismatch and line-of-sight building interference loss between the Experimental Farm and the Electrical Engineering Building.

# 7 Conclusion

The working system will be implemented at the greenhouse to monitor power system status. The receive system will be set up outside the microwave lab for others to use and expand upon. The potentials of using directional antennas for communication in remote areas are boundless. Aquaponics is used in developing nations to grow food and raise livestock. This alert system is invaluable to preserving ideal conditions for crops. Alerting farmers of problematic conditions will prevent large external costs of replacing perished livestock.

# 8 Further Development

The project will be presented to the Aquaponics Director. A possible future integration is the Solar Ventilation Project that requires a microcontroller to regulate greenhouse temperature. This project accommodates Solar Ventilation project performance requirements; 97% of the LoRa radio bandwidth is available for implementation.

# 9 Appendices

## 9.1 Appendix A

LoRa Radio Code

Greenhouse Transmitter

---

```
// Feather9x_RX
// -*- mode: C++ -*-
// Fish Saving Receiver Sensor Setup

#include <Wire.h>
#include <SPI.h>
#include <RH_RF95.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_Si7021.h>

#define RFM95_INT 3
#define RFM95_RST 4
#define RFM95_CS 8

#define POWERDETECT A0
#define NORMAL A1
#define DANGER A2
#define LIGHT A3
#define DOOR A4

// Change to 434.0 or other frequency,
    must match RX's freq!
#define RF95_FREQ 900.0

// Singleton instance of the radio
    driver
RH_RF95 rf95(RFM95_CS, RFM95_INT);

/*#if defined(ARDUINO_SAMD_ZERO) &&
    defined(SERIAL_PORT_USBVIRTUAL)
// Required for Serial on Zero based
    boards
#define Serial SERIAL_PORT_USBVIRTUAL
#endif*/

Adafruit_Si7021 sensor =
    Adafruit_Si7021();

void setup()
{
  pinMode(RFM95_RST, OUTPUT);
  pinMode(POWERDETECT, INPUT);
  pinMode(NORMAL, OUTPUT);
  pinMode(DANGER, OUTPUT);
```

```
pinMode(LIGHT, INPUT);
pinMode(DOOR, INPUT);
digitalWrite(RFM95_RST, HIGH);

//Serial.begin(9600);

//Serial.println(F("Si7021 test"));

while(!sensor.begin())
{
  //Serial.println(F("Could not find
      a valid Si7021 sensor, check
      wiring!"));
  delay(1000);
}

//Serial.println("Feather LoRa RX
    Test!");

// manual reset
digitalWrite(RFM95_RST, LOW);
delay(10);
digitalWrite(RFM95_RST, HIGH);
delay(10);

while (!rf95.init()) {
  //Serial.println("LoRa radio init
      failed");
  while (1);
}
//Serial.println("LoRa radio init
    OK!");

// Defaults after init are 434.0MHz,
    modulation GFSK_Rb250Fd250, +13dbM
if (!rf95.setFrequency(RF95_FREQ)) {
  //Serial.println("setFrequency
      failed");
  while (1);
}
//Serial.print("Set Freq to: ");
    Serial.println(RF95_FREQ);

// Defaults after init are 434.0MHz,
    13dBm, Bw = 125 kHz, Cr = 4/5, Sf
    = 128chips/symbol, CRC on

// The default transmitter power is
    13dBm, using PA_BOOST.
// If you are using RFM95/96/97/98
    modules which uses the PA_BOOST
    transmitter pin, then
// you can set transmitter powers
    from 5 to 23 dBm:
rf95.setTxPower(23, false);
}

void loop()
```

```
{
  // Constants for Program Use
  static uint8_t packet[128];
  memset(packet, 0, sizeof(packet));
  uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
  uint8_t len = sizeof(buf);
  static float temperature = 0;
  static float temperatureF = 0;
  static float humidity = 0;
  static float voltage = 0;
  static float powercheck = 0;
  static int numberBytes = 0;
  static float light = 0;
  static float lightIntensity = 0;
  static int door = 0;
  //static boolean ajar = false;

  //Check System Voltage
  voltage = analogRead(POWERDETECT);
  voltage = voltage*6/1023;
  light = analogRead(A3);
  door = analogRead(A4)-100;
  lightIntensity = ((light/1023)*100);

  if(voltage>4.5)
  {
    digitalWrite(NORMAL, HIGH);
    //Modify the data to be sent into
        byte form
    temperature =
        sensor.readTemperature();
    temperatureF = temperature*9/5+32;
    //Send Temperature
    htonf(temperature, packet+1,
        &numberBytes);
    humidity = sensor.readHumidity();
    //Send Humidity
    htonf(humidity, packet+5,
        &numberBytes);
    //Send Light Intensity
    htonf(lightIntensity, packet+9,
        &numberBytes);
    //Send Door Value 0 is open, 1 is
        closed
    if(door > light+100)
    {
      //ajar = false;
      //Serial.println("Door is
          closed");
      htonf(1, packet+13, &numberBytes);
    }
    else
    {
      //ajar = true;
      //Serial.println("Door is open");
      htonf(0, packet+13, &numberBytes);
    }
    //altitude =
```

```
    bmp.readAltitude(1012.19);
//htonf(altitude, packet+8);
//Transmit Temperature, Pressure,
    Altitude
rf95.send(packet, numberBytes+1);
rf95.waitPacketSent();

//For Debugging Purposes comment
    out if not used
/*Serial.print("Number of Bytes
    Transmitted: ");
Serial.println(numberBytes);
Serial.print("Voltage: ");
Serial.println(voltage);
Serial.print("Temp: ");
Serial.println(temperatureF);
Serial.print("Humidity: ");
Serial.println(humidity);
Serial.print("Light Intensity: ");
Serial.print(lightIntensity);
Serial.println("%");
Serial.print("Is the door open?: ");
Serial.println(ajar);
Serial.print("Bytes Transmitted: ");
for(int i = 0; i<numberBytes;i++)
{
  Serial.print(packet[i], HEX);
  Serial.print(" ");
}
  Serial.println(buf[numberBytes+1],
      HEX);
  Serial.println();*/
  numberBytes = 0;
}
else
{
  digitalWrite(NORMAL, LOW);
  digitalWrite(DANGER, HIGH);
  delay(500);
  while(voltage<4.5)
  {
    digitalWrite(DANGER, LOW);
    powercheck =
        analogRead(POWERDETECT);
    voltage = powercheck*6/1023;
    packet[0] = 255;
    rf95.send(packet, sizeof(packet));
    rf95.waitPacketSent();
    delay(500);
    digitalWrite(DANGER, HIGH);
    delay(500);
  }
  digitalWrite(DANGER, LOW);
  digitalWrite(NORMAL, HIGH);
  packet[0] = 0;
}
}
```

```
//Function for modifying floats into
   byte form
void htonf(float message, uint8_t
   *packet, int *numberBytes)
{
  int32_t pelican =
     (int32_t)(message*100);
  *packet = pelican&255;
  pelican>>=8;
  *(packet+1) = pelican&255;
  pelican>>=8;
  *(packet+2) = pelican&255;
  pelican>>=8;
  *(packet+3) = pelican&255;
  *numberBytes = *numberBytes+4;
}
```

Campus Base Receiver

```
// -*- mode: C++ -*-
//Design for PolyPonics Transmitter
   for Fish Saving
#include <SPI.h>
#include <RH_RF95.h>
#define RFM95_CS 8
#define RFM95_RST 4
#define RFM95_INT 3
// Change to 434.0 or other frequency,
   must match RX's freq!
#define RF95_FREQ 900.0
// Singleton instance of the radio
   driver
RH_RF95 rf95(RFM95_CS, RFM95_INT);

#if defined(ARDUINO_SAMD_ZERO) &&
   defined(SERIAL_PORT_USBVIRTUAL)
  // Required for Serial on Zero based
     boards
  #define Serial SERIAL_PORT_USBVIRTUAL
#endif

#define PACKETSIZE 128
void setup()
{
  pinMode(RFM95_RST, OUTPUT);
  digitalWrite(RFM95_RST, HIGH);
//while (!Serial);
Serial.begin(9600);
//Serial1.begin(9600);
delay(100);

Serial.println("Feather LoRa TX
   Test!");

// manual reset
digitalWrite(RFM95_RST, LOW);
delay(10);
digitalWrite(RFM95_RST, HIGH);
delay(10);

while (!rf95.init()) {
  Serial.println("LoRa radio init
     failed");
  while (1);
}
Serial.println("LoRa radio init OK!");

// Defaults after init are 434.0MHz,
   modulation GFSK_Rb250Fd250, +13dbM
if (!rf95.setFrequency(RF95_FREQ)) {
  Serial.println("setFrequency
     failed");
```

```
    while (1);
  }
  Serial.print("Set Freq to: ");
    Serial.println(RF95_FREQ);
  // Defaults after init are 434.0MHz,
     13dBm, Bw = 125 kHz, Cr = 4/5, Sf
     = 128chips/symbol, CRC on
  // The default transmitter power is
     13dBm, using PA_BOOST.
  // If you are using RFM95/96/97/98
     modules which uses the PA_BOOST
     transmitter pin, then
  // you can set transmitter powers
     from 5 to 23 dBm:
  rf95.setTxPower(23, false);
}

void loop()
{
  static uint8_t packet[128];
  static int counter = 0;
  static float temperature = 0;
  static float humidity = 0;
  static float lightIntensity = 0;
  static boolean doorOpen = false;
  static int rssi = 0;

  uint8_t buf[RH_RF95_MAX_MESSAGE_LEN];
  uint8_t len = sizeof(buf);
```

```
if (rf95.waitAvailableTimeout(5000))
{
  // Should be a reply message for us
     now
 if (rf95.recv(buf, &len))
 {
   if(buf[0]!=255)
   {
     /*Serial.print("Bytes Received:
        ");
     for(int i = 0; i<len; i++)
     {
       Serial.print(buf[i]);
       Serial.print(" ");
     }
     Serial.println(buf[12]);*/
     Serial.print("Rssi: ");
     rssi = rf95.lastRssi();
     Serial.println(rssi);
     temperature =
        ntohf(buf+1)*9/5+32;
     Serial.print("Temp: ");
     Serial.println(temperature);
     humidity = ntohf(buf+5);
     Serial.print("Pressure: ");
     Serial.println(humidity);
     lightIntensity = ntohf(buf+9);
     Serial.print("Light Intensity:
```

```
    ");
  Serial.print((int)lightIntensity);
  Serial.println("%");
  doorOpen = ntohf(buf+13);
  Serial.print("Is the door
    closed?: ");
  if(doorOpen)
  {
    Serial.println("Yes");
  }
  else
  {
    Serial.println("No");
  }
  Serial.println();
  Serial.println();
}
else
{
  Serial.print("Rssi: ");
  Serial.println(rssi);
  Serial.println("Emergency
    Status, Power Failure
    Detected, Alerting Help");
  Serial.println();
}
    }
    else
    {
      Serial.println("Receive failed");
    }
  }
  else
  {
    Serial.println("No reply, is there
      a listener around?");
  }
}

float ntohf(uint8_t *packet)
{
  int32_t fish = 0;
  fish|=*(packet+3);
  fish<<=8;
  fish|=*(packet+2);
  fish<<=8;
  fish|=*(packet+1);
  fish<<=8;
  fish|=*packet;
  return (float)fish/100;
}
```

### 9.2 Appendix B

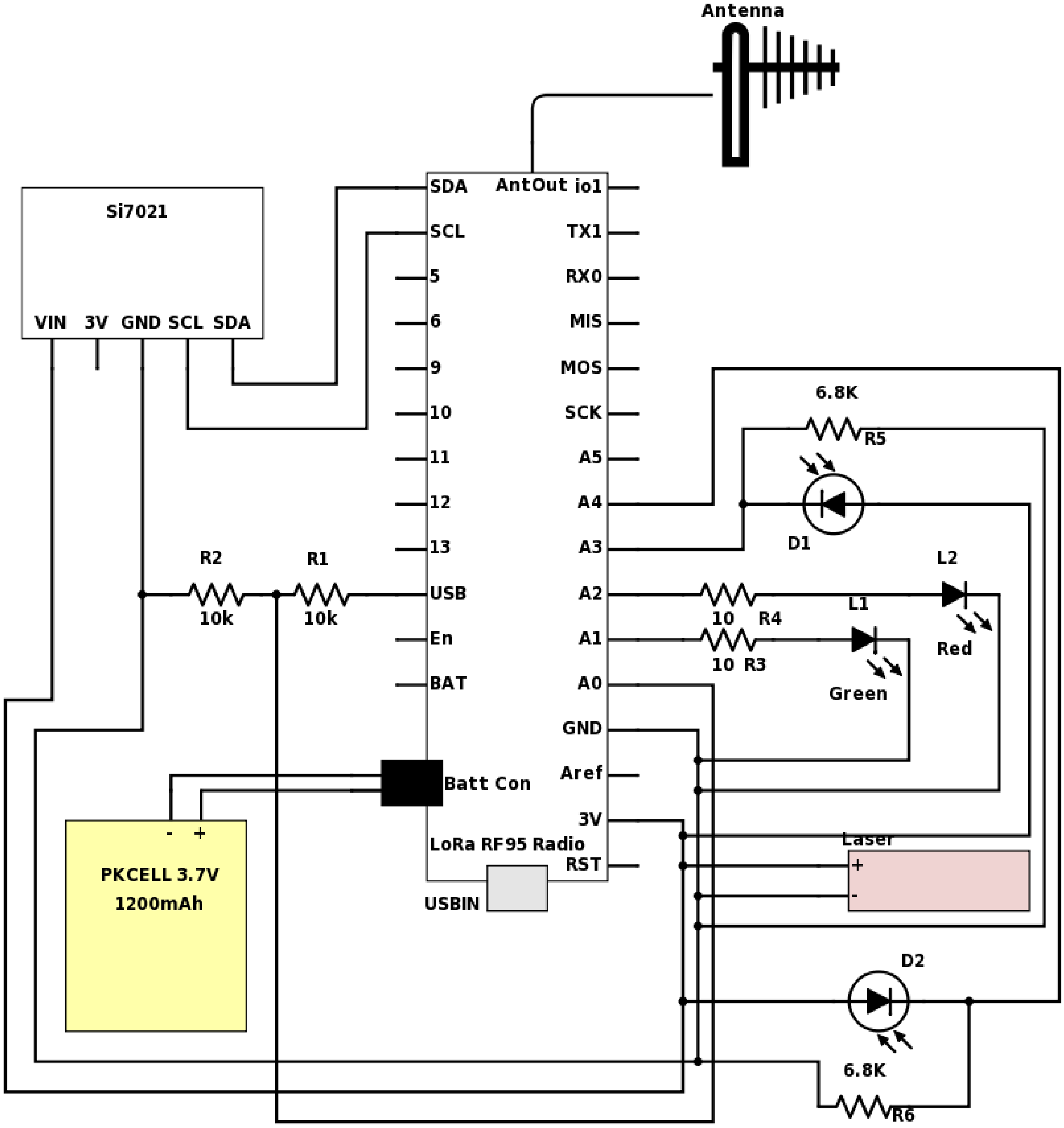


Figure 20: *Greenhouse Base Transmitter*

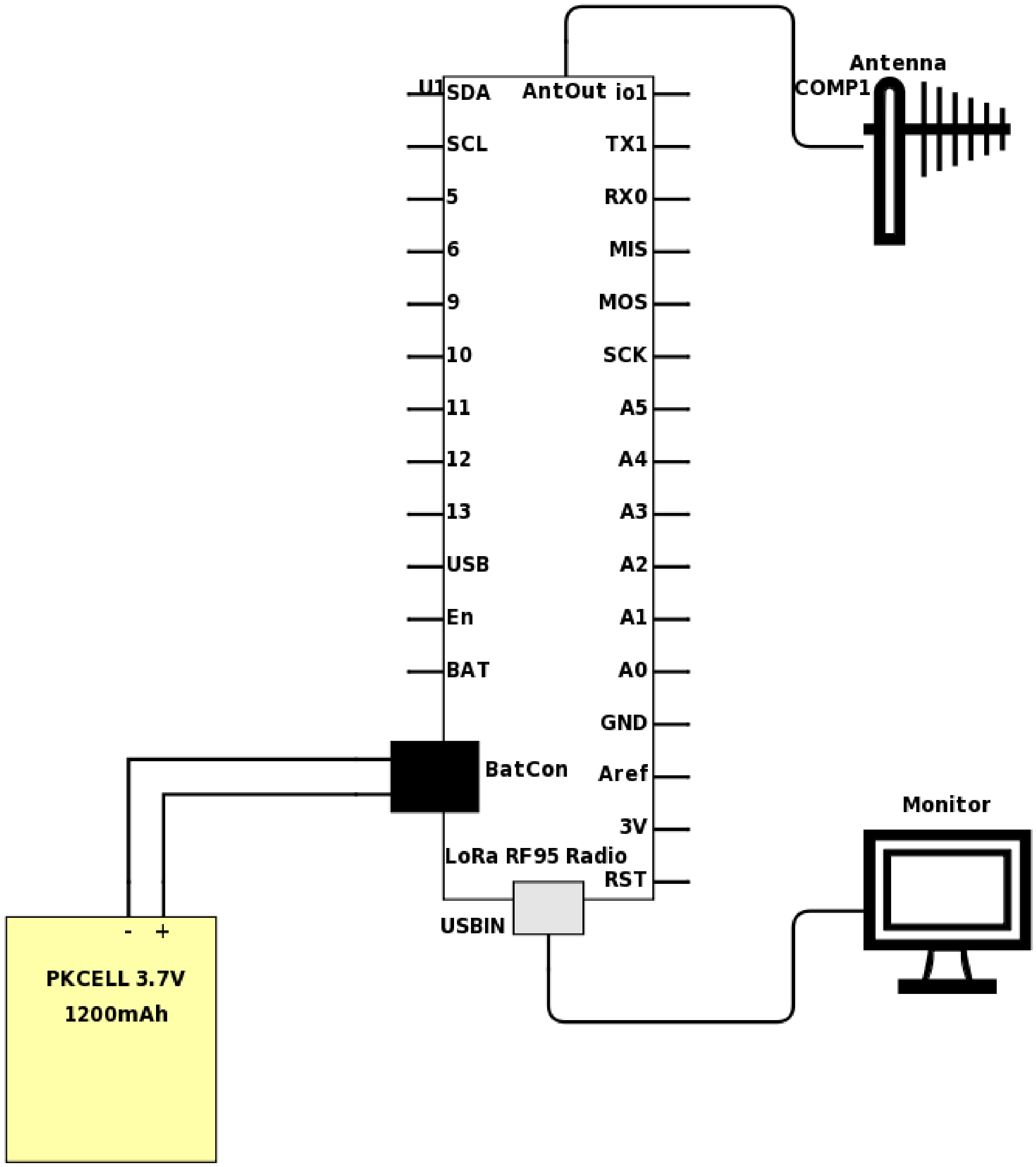


Figure 21: *Campus Base Receiver*

## 9.3 Appendix C

Table VIII: *BILL OF MATERIALS*

| Name | Part # | Vendors | Dimensions | Qty | Cost $ |
|---|---|---|---|---|---|
| RFM95 LoRa Radio | 3178 | Adafruit | 1” x2” | 2 | 34.50 |
| RG174 SMA Coaxial Cable | CO-174SMAX200-005 | Amazon | 5’ | 2 | 10.92 |
| SMA F to RP-SMA M | AD078 | DHT Electronics | 0.3” x 0.3” x 0.2” | 2 | 4.80 |
| Aluminum Rod | 204273994 | Everbuilt | 0.25” Dia x 3’ L | 2 | 4.21 |
| Brass Rod | 5024161 | K & S | 0.25” Dia x 1’ L | 1 | 2.59 |
| Brass Rod | 5024245 | K & S | 0.5” Dia x 1’ L | 1 | 3.79 |
| 10ΩResistor | CF14JT10R0 | Digikey | 0.091” Dia x 0.236” L | 1 | 0.91 |
| 100ΩResistor | CF18JT100R | Digikey | 0.091” Dia x 0.236” L | 1 | 0.10 |
| 6.8kΩResistor | CF14JT6K80 | Digikey | 0.091” Dia x 0.236” L | 1 | 0.10 |
| 10kΩResistor | CF14JT10K0 | Digikey | 0.091” Dia x 0.236” L | 2 | 0.10 |
| Green LED | 298 | Adafruit | 5mm | 1 | 0.16 |
| Yellow LED | 2700 | Adafruit | 5mm | 1 | 0.16 |
| Laser Diode | 1054 | Adafruit | 0.4” x 1.2” | 1 | 5.95 |
| Photo Diode | 2831 | Adafruit | 0.2” x 0.21” | 2 | 0.95 |
| Battery | LP-503562 | Adafruit | 1.3” x 2.4” x 0.2” | 2 | 9.95 |
| Temp Sensor | Si7021 | Adafuit | 0.7” x 0.6” x 0.1” | 1 | 6.95 |
| 100x M-M Breadboard Wires | B01L5UJ36U | GenBasic | 4” and 8” | 1 | 5.99 |
| Breadboard | B0040Z1ERO | BusBoard Prototype Systems | 3.3” x 2.1” x 0.3” | 2 | 4.75 |
| Wooden Box | NA | Michael’s | 8.4” x 5.3” x 3.4” | 2 | 5.99 |
| | | | | | Total |
| | | | | | 179.04 |

# 10 References


[1] ANSYS® Academic Research, Release 15.0

[2] Radio-Electronics. *Yagi Antenna Gain and Directivity*, Web. 13 Jun. 2017

[3] *ARRL Handbook.* The American Radio Relay League, Inc., Sept. 2016.

[4] AdaFruit Learning System. LoRa Feather M0 Board Datasheet. Web. 13 Jun. 2017

[5] RadioHead. Web. 13 Jun. 2017.

[6] *The Dipole Antenna.* Antenna Theory, 2016. Web. 13 Jun. 2017.

[7] Shaw, Joseph A. *Radiometry and the Friis Transmission Equation.* American Journal of Physics, Sept. 2012. Web. 13 Jun. 2017.

[8] Muggy Weld LLC. Web. 13 Jun. 2017.

[9] Bredhurst Receiving and Transmitting Society, Web. 19 Jun. 2017.