Dwight Look College of
ENGINEERING
TEXAS A&M UNIVERSITY

ECEN 404 Final Presentation
Team 14: RF Triangulation

Josh Broyles, Jack Parkinson,
Kathleen Hutchinson, & Brandon Stokes
TA: Souryendu Das
Sponsor: Max Lesser

Project Summary

Biologists have struggled being able to collect data on wildlife's location, habitat use, and breeding patterns without getting directly involved with the animal they're tracking.

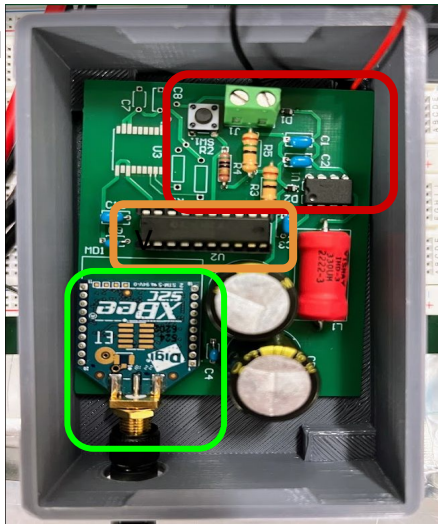
Radio Frequency Triangulation allows a user to track a **known frequency** (such as a previously tagged animal) within the **triangulated area of three antennas** by using a motor to successfully **pinpoint the strongest signal.**

Helpful to study both invasive and threatened wildlife, our RF Triangulation system will focus on being able to accurately **track a transmitter within a 100 meter radius with >10% error.**

Dwight Look College of ENGINEERING
TEXAS A&M UNIVERSITY

Integrated Project Diagram

1. **GUI** connected to database through WiFi
2. **Receiver** gathering and sending data through WiFi to update database
3. **Directional Antenna**, receiving signal at different signal strengths
4. **Stepper Motor**, spinning receiver to scan the area for the transmitter
5. **Transmitter**, constantly outputting signal on a known frequency so it can be found



Fully Implemented Transmitter

Power Supply

MCU

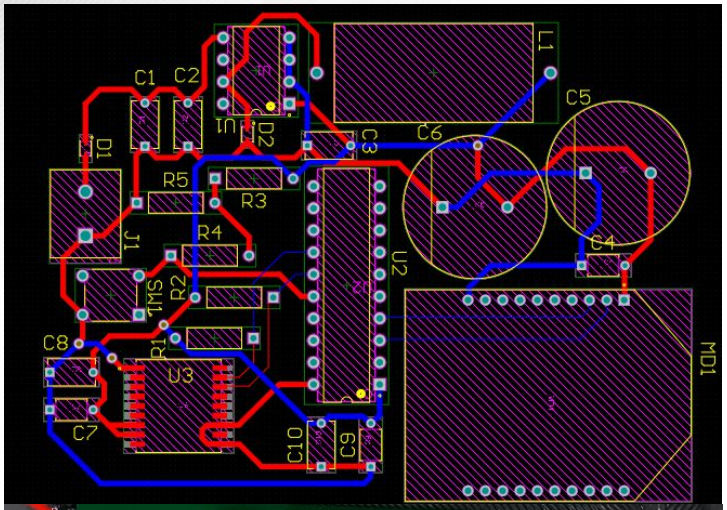
Radio Module

Josh Broyles Accomplishments

Designed	Tested	Challenges	Solutions

PCB Designed:
• MSP430
• Xbee
• Ublox
 	
Coded
• MSP430
  ○ Transmit
  ○ Echo
• Xbee
	PCB:
	• Power Supply
	  ○ Line Regulation
	  ○ Load Regulation
	
	Code:
	• MSP430 Transmit message to Xbee
	• MSP430 Echo message back through Xbee
	
	Other:
	• Transmitter Range Test
		• MSP430 unresponsive I2C commands
		• MSP430 debugging without terminal
		• MSP430 reset does not rerun program
			• GPS coordinates can be hardcoded in
			• MSP can output messages to Xbee, which can be used to debug
			• Turning the system off then back on will rerun the program

Josh Broyles
Designed / Coded




```c
#include <msp430g2553.h>
#define TXD BIT2
#define RXD BIT1

void String_TX(char * data);
int main(void)
{
   WDTCTL = WDTPW + WDTHOLD;                 //
   DCOCTL = 0;                               //
   BCSCTL1 = CALBC1_1MHZ;                    //
   DCOCTL = CALDCO_1MHZ;
   P1SEL |= RXD + TXD ;                      //
   P1SEL2 |= RXD + TXD ;                     //
   UCA0CTL1 |= UCSSEL_2;                     //
   UCA0BR0 = 0x68;                           //
   UCA0BR1 = 0x00;                           //
   UCA0MCTL = UCBRS0;                        //
   UCA0CTL1 &= ~UCSWRST;                     //
   ///Start looping///
   while(1)
   {
           String_TX("30.620995,-96.340089\r");
           __delay_cycles(1000);          //Wait
   }
}

void String_TX(char * data)                  //
   {
       unsigned int i=0;
       while(data[i])                        //
       {
           while (!(IFG2&UCA0TXIFG));      // W
           UCA0TXBUF = data[i];              //
           i++; // Increment variable for arra
       }
   }
```

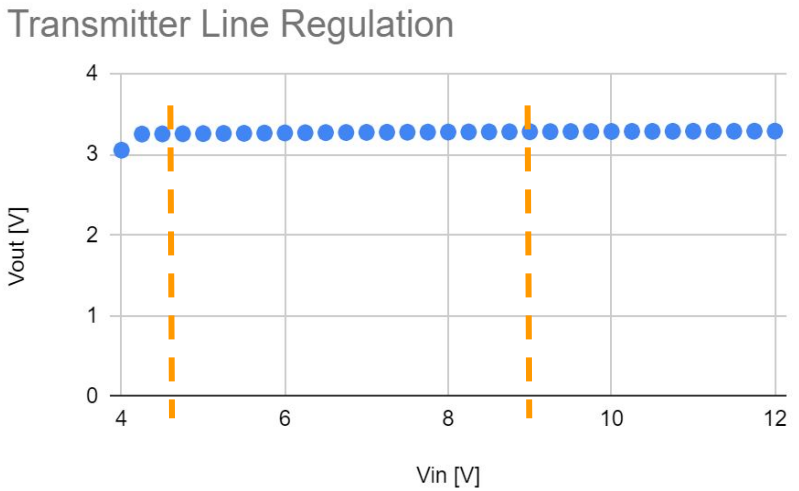
Repeat Message code

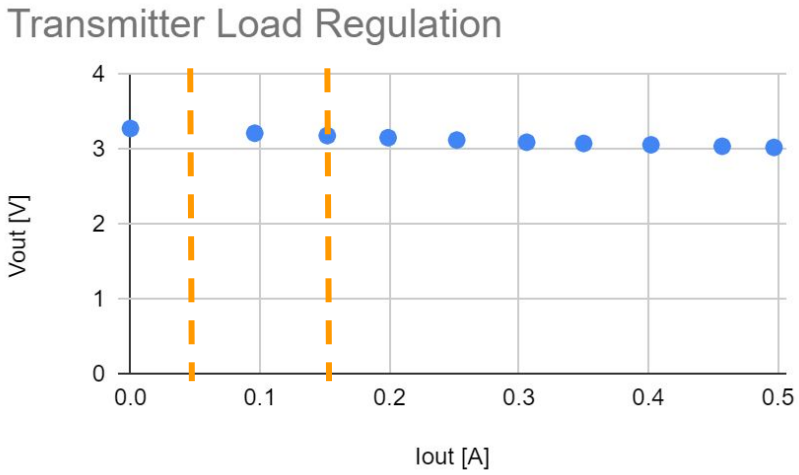
```c
#include <msp430.h>
#define TXD BIT2
#define RXD BIT1

int main(void)
{
  WDTCTL = WDTPW + WDTHOLD;
  if (CALBC1_1MHZ==0xFF)
  {
    while(1);
  }
  DCOCTL = 0;
  BCSCTL1 = CALBC1_1MHZ;
  DCOCTL = CALDCO_1MHZ;
  P2DIR |= 0xFF;
  P2OUT &= 0x00;
  IFG2 &= ~UCA0TXIFG;          //clear the TX

  UCA0IRRCTL |= UCIRRXPL;

  P1SEL = TXD | RXD;
  P1SEL2 = TXD | RXD;
  UCA0CTL1 = UCSWRST;
  UCA0CTL1 |= UCSSEL_2;
  UCA0BR0 = 104;
  UCA0BR1 = 0;
  UCA0MCTL = UCBRS1;
  UCA0CTL1 &= ~UCSWRST;
  IE2 |= UCA0RXIE;
  __bis_SR_register(LPM0_bits + GIE);
}
//  Echo back RXed character, confirm TX buf
#pragma vector=USCIAB0RX_VECTOR
__interrupt void USCI0RX_ISR(void)
{
  while(!(IFG2&UCA0TXIFG));
  UCA0TXBUF = UCA0RXBUF;   // Send TX to RX
}
```

Echo Code

Josh Broyles
Validation

Requirement	Validated
Vout = 3V - 3.3V with Vin between 4.5V - 9V	Vin = 9V, Vout = 3.281V Vin = 4.5V, Vout = 3.25V
3V - 3.3V at 0.15A for MCU and other modules	3.182V at 0.152A

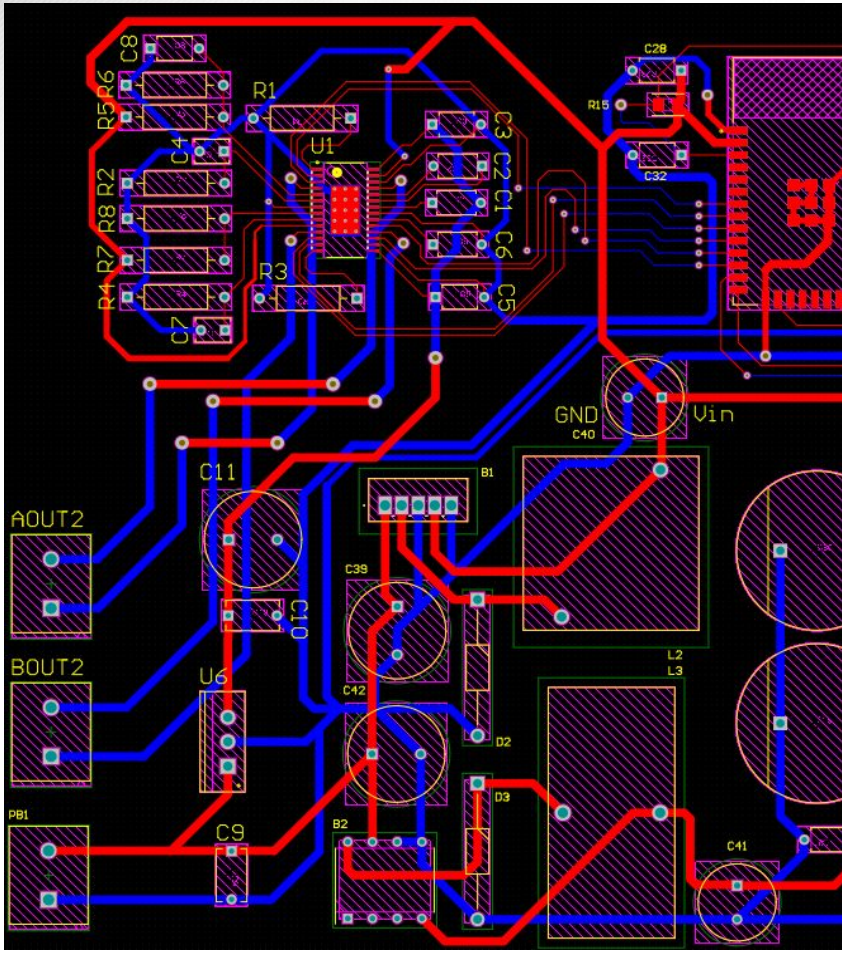


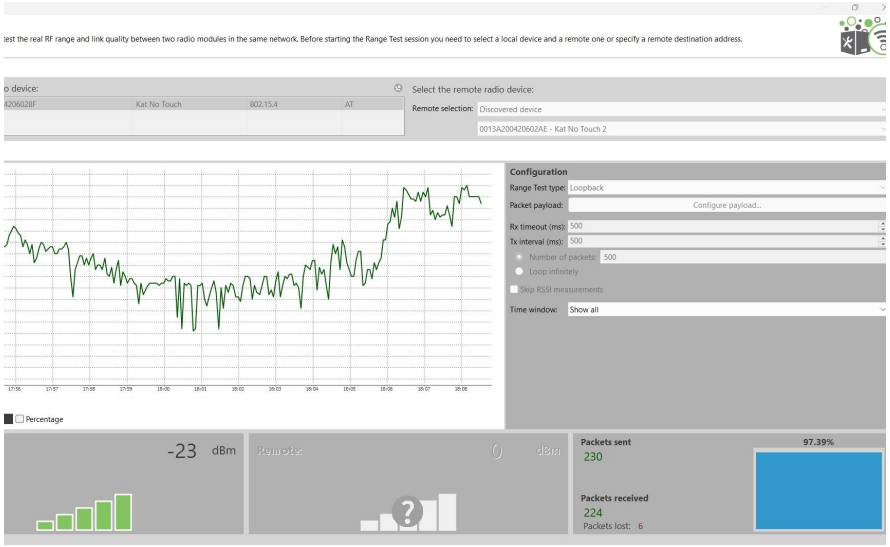


Jack Parkinson Accomplishments

Designed	Tested	Challenges	Solutions
PCB Design: • Stepper Motor Driver • 12V regulator Directional Antenna: • Patch Antenna • Yagi Antenna ESP32 Stepper Motor Code	PCB: • 12V regulator Directional Antenna: • directionality • range Stepper Motor: • speed • step size • step direction • motor torque	• ESP32 flash capabilities • Motor Driver parts too small • 12V regulator dropout voltage too large • Patch Antenna simulation • Yagi antenna manufacturing	• Redesign PCB • Buy new parts • Buy an antenna

Jack Parkinson
Tested

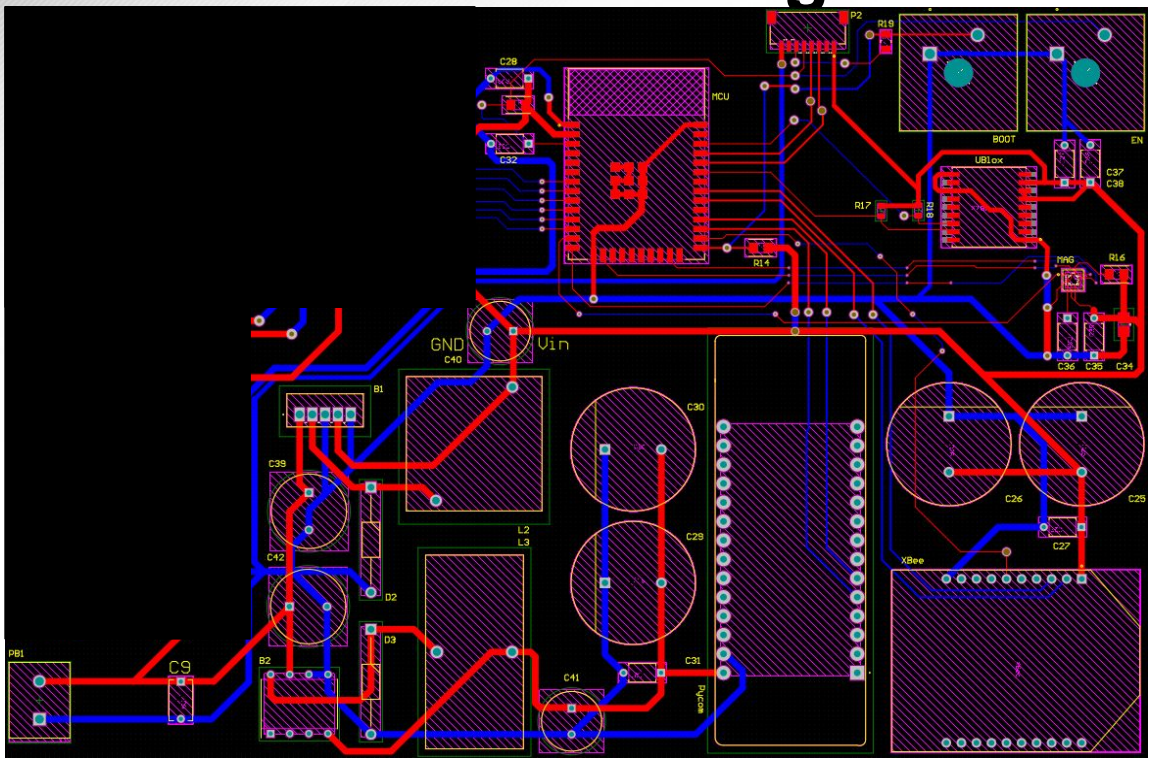




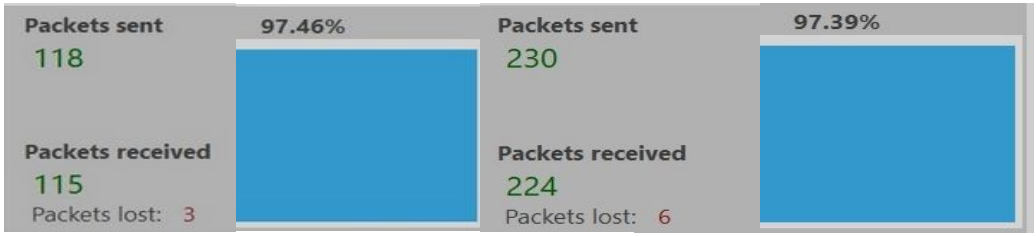
Kathleen Hutchinson Accomplishments

Designed	Tested	Challenges	Solutions
PCB Designed: • ESP32 • XBee • GPS • Magnetometer • Pycom Coded • ESP32 • XBee • Pycom	PCB: • Power Supply • ESP32 Tx0/Rx0 Code: • ESP32 reading XBee data • ESP32 reading and transmitting Pycom data Other: • XBee Accuracy • Transmission Time	• SPI and I2C with small, surface mount modules • Circuit/PCB design • XBee RSSI value • XBee accuracy (packets) drop after 50 meters • Power Supply (inductors, heating problems)	GPS Coordinates can be manually set since phone hotspots will be used Power supply and connections are updated on new PCB

Kathleen Hutchinson
Designed / Coded



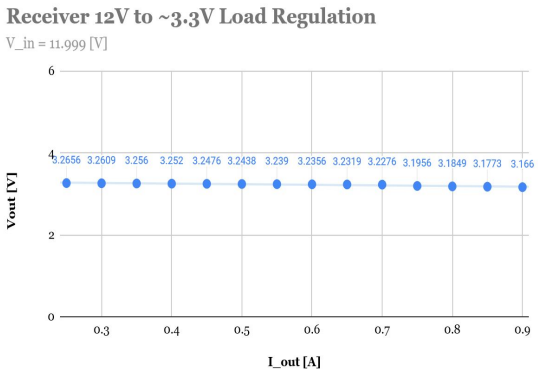

```
void loop() {
  // Looking for "start signal", set to "esp32 check "
  PycomReceived1 = MySerial1.readStringUntil('\r');
  Serial.println(PycomReceived1);
  // System does not start until "start signal" is sent
  if (PycomReceived1 == "esp32 check "){
    // While loop stating to read in when XBee sends string
    while((MySerial2.available() > 0) || (MySerial1.available() > 0)) {
      // Reading until end line
      PycomReceived2 = MySerial1.readStringUntil('\r');
      XBeeReceived = MySerial2.readStringUntil('\r');
      // Print Pycom to Serial Monitor
      Serial.print("Printing Pycom Message UART1: ");
      Serial.println(PycomReceived2);
      // Print XBee to Serial Monitor
      Serial.print("Printing XBee Message UART2: ");
      Serial.println(XBeeReceived);
      // Sending/Writing GPS Coordinates to Pycom
      MySerial1.print(XBeeReceived);
    }
    // 1 second delay
    delay(1000);
  }
}
```

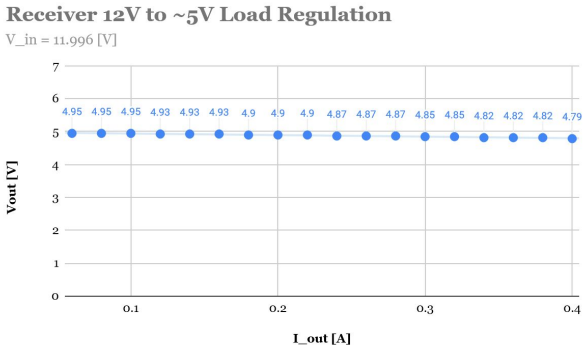


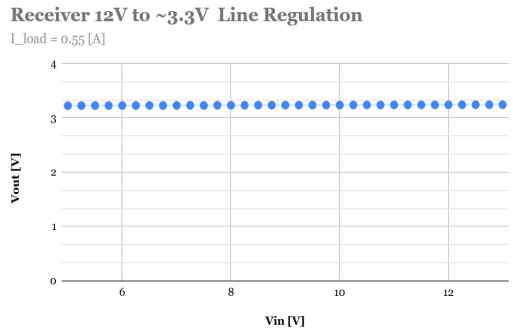
Dwight Look College of ENGINEERING
TEXAS A&M UNIVERSITY

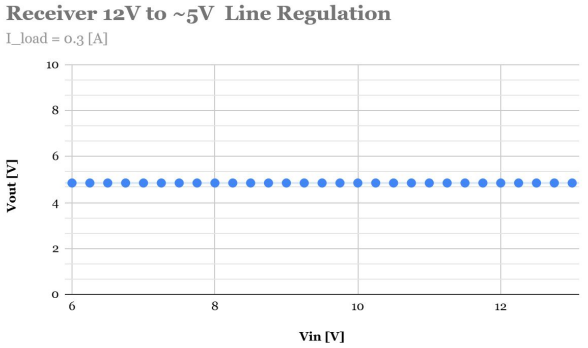
Kathleen Hutchinson Validation

Requirement	Validated
3V at 1A for modules & MCU	3.1V at 1A
3.5V at 0.5A for Pycom	4.7V at 0.5A
30 sec transmission time	~14 sec transmission time









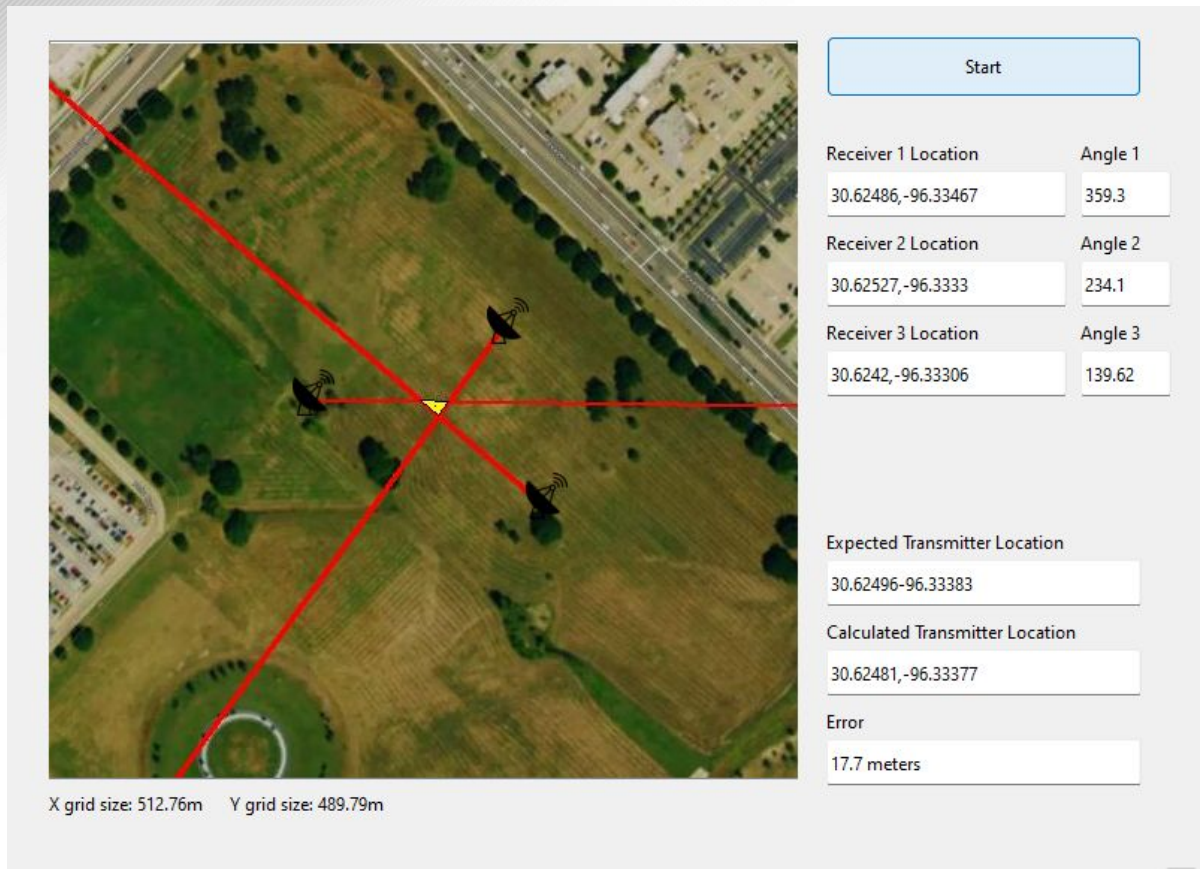
Dwight Look College of ENGINEERING TEXAS A&M UNIVERSITY

Brandon Stokes Accomplishments

Designed	Tested	Challenges	Solutions
GUI Designed: • Display locations based on GPS • Satellite map view of the area • Error between the calculated and actual transmitter Database: • Hosted on AWS that stores all the relevant information Pycom: • Coded communication between GUI, Database and MCU	WiFi: • WiFi connection will be able to run continuously for 20 minutes and be able to reconnect Code: • Gives errors for wrong values(i.e given a wrong angle will tell you which receiver it came from • How much RAM the program takes up on Users computer Other: • Database integrity to not allow duplicate values for receiver data	• LTE	Solution to not having LTE was to use WiFi hotspots as the connection for each of the receivers

Dwight Look College of ENGINEERING
TEXAS A&M UNIVERSITY

Brandon Stokes Design



	recname [PK] smallint	rec_lat double precision	rec_long double precision	trans_lat double precision	trans_long double precision	sig_angle double precision	time_last_updated text
1	1	30.62486	-96.33467	30.62496	-96.33383	359.3	(2023, 4, 10, 10, 44, 0, 155130, None)
2	2	30.62527	-96.3333	30.62496	-96.33383	234.1	(2023, 4, 10, 10, 44, 0, 182234, None)
3	3	30.6242	-96.33306	30.62496	-96.33383	139.62	(2023, 4, 10, 10, 44, 0, 139972, None)

Dwight Look College of ENGINEERING
TEXAS A&M UNIVERSITY

Brandon Stokes Validated

Requirement
The wifi hotspot does not drop more than 1 time per 5 minutes and shall reconnect within 20s. Total Pycom connection time max = 20 min
The time it takes to connect and transmit data between the GUI, transmitter, and receiver shall be 30s.
The GUI shall not take up more than 1 GB RAM on user's computer.
The Database should not be able to take in duplicate values or values that do not match the expected criteria

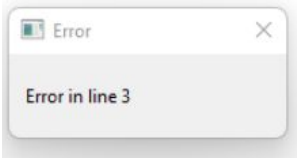
```
Connected for 6 minute(s).
Connected for 7 minute(s).
Connected for 8 minute(s).
Connected for 9 minute(s).
Connected for 10 minute(s).
Disconnected at 600.0463 seconds.
Attempting to reconnect...
Reconnected successfully in 4.00 seconds.
Connected for 11 minute(s).
Connected for 12 minute(s).
```

```
Connected for 15 minute(s).
Connected for 16 minute(s).
Connected for 17 minute(s).
Connected for 18 minute(s).
Connected for 19 minute(s).
Connected for 20 minute(s).
```

```
Network found!
Attempting Connection....
Attempting Connection....
WLAN connection succeeded!
Ready for Start to be pressed
------------Start Button has been pressed------------
Completed update for Receiver 1
Completed update for Receiver 2
Completed update for Receiver 3
Elasped time:  14.85516 s
```

```
Time taken for GUI to load:  2.237
RAM used in process:  105.211  Mb
Memory percent is :  0.644 %
```

```
Time taken for GUI to load:  1.996
RAM used in process:  106.277  Mb
Memory percent is :  0.651 %
```




```
ERROR:  duplicate key value violates unique constraint "recdata_pkey"
DETAIL:  Key (recname)=(1) already exists.
SQL state: 23505
```

Integrated System Results

The **connection** between the transmitting XBEE and the receiver's XBEE, ESP32, Pycom, to database **transmits in about 14 seconds**, as previously stated in Brandon Stokes's slide.

When within about **50 meters of each other,** the XBEE's have an **accuracy of 97%**; however, our project requires a distance of 100 meters from each antenna - thus an updated antenna is required.

Performance will be improved with a redesigned PCB.

Success in this area was described as successfully transmitting data, failure would be a lack of communication.

Further system success criteria requires the motor and redesigned PCB to successfully move to find the transmitter.

```
Printing Pycom Message UART1:
Printing XBee Message UART2: 30.624961, -96.333823
Printing Pycom Message UART1:
Printing XBee Message UART2: 30.624961, -96.333823
Printing Pycom Message UART1:
Printing XBee Message UART2: 30.624961, -96.333823
Printing Pycom Message UART1:
Printing XBee Message UART2: 30.624961, -96.333823
```

```
Network found!
not connected
not connected
WLAN connection succeeded!
running
30.624961, -96.333823
trans_lat: 30.6249595
trans_long: -96.3338280
running
30.624961, -96.333823823
trans_lat: 30.6249595
trans_long: -96.3338280
running
```

Conclusions

Some parts of the project were changed due to changes in the modules. Specifically, the major changes resulted in the project moving from:

- Automatically reading in GPS coordinates from the UBLOX module to **manually entering the coordinates** because the team was unsuccessful in reading the data.
- Using the magnetometer to move system to North to **manually setting the receivers facing North** when placed in the field.
- Using LTE through the Pycom module because the module refused to connect with multiple sim cards, thus the team **switched to WiFi hotspots**.

Current Status	
Transmitter	Fully Tested/Validated
Antenna and ESP32+Modules	Repeating Testing for Replacement PCB
Motor and ESP32+Modules	Repeating Testing for Replacement PCB
Pycom and ESP32+Modules	Repeating Testing for Replacement PCB
GUI + Database and Pycom	Fully Integrated/Tested/Validated