Analytical Analysis: Wireless Communication

Prosthetic Hand

Allison Cutler

Team 18S12 Active Prosthetic

ME 476CH Mechanical Engineering Design

March 1, 2019

Introduction

A large aim of the prosthetic hand project is to allow the client to use sensors within a shoe insole to control finger motion on the prosthetic hand. In order to avoid a tangled and potentially dangerous mess of wires, the best communication between these sensors on the foot and the motors in the hand is through wireless communication. Using arduino to control the motors and XBee to relay the data from the sensors in an apparatus shown in figure 1, the code for wireless communication needs to successfully show transmission of data. Before data can be transmitted, the XBee and Arduino needed to communicate with each other. This individual analysis involved deriving a code using Arduino v1.8.8 and XCTU software to show that the XBee and Arduino are transmitting and receiving words.

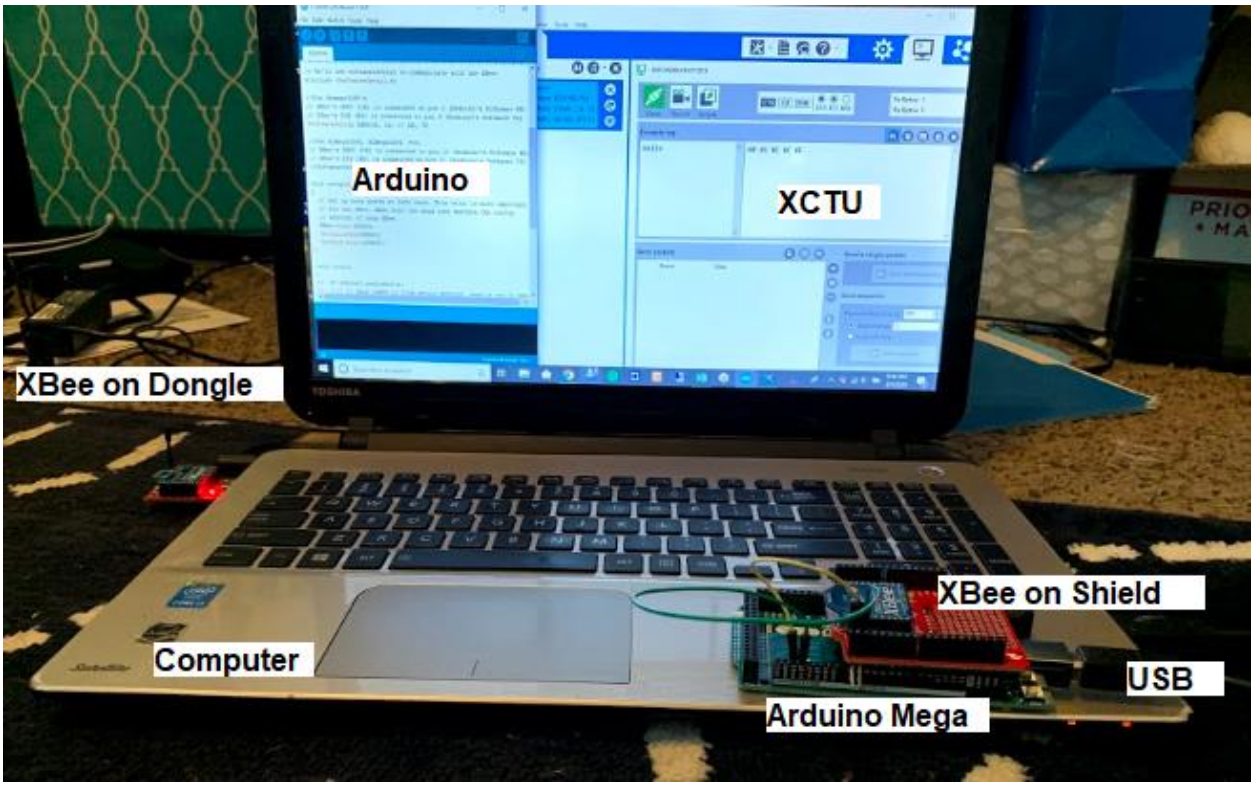

Figure 1: Wireless Communication Apparatus

The materials used to conduct the analysis were two XBee Series 1, an Arduino Mega, a dongle device, an XBee shield, an USB cable, and a computer with Arduino v1.8.8 and XCTU software. XBee was determined to be the most appropriate form of wireless communication due to the materials being on hand for testing, the materials being cheaper, and the ability to send data across using Arduino to be more reliable.

Configuration

Before code could be successfully passed between the devices, each XBee had to be configured. The configuration for each is shown in figure 2.

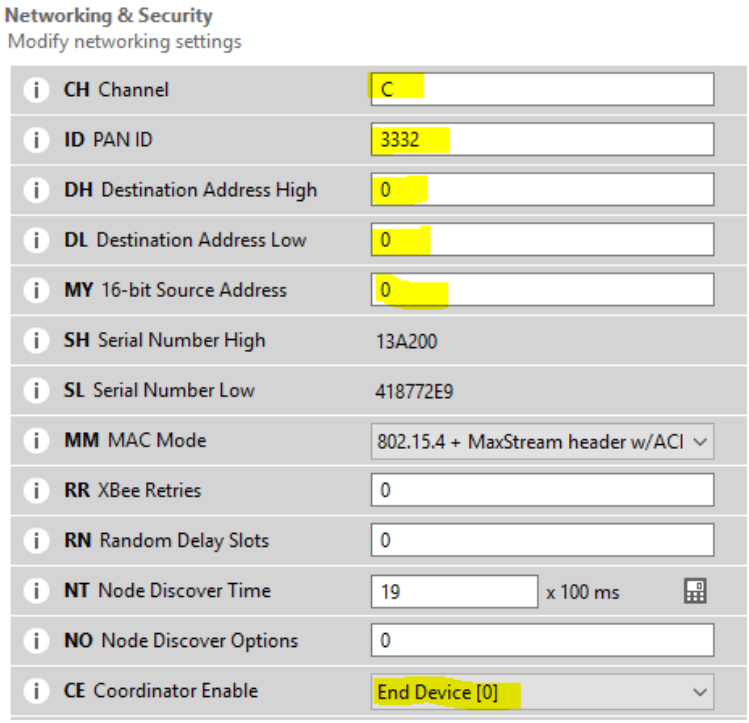

Figure 2: XBee Configuration

This configuration was recommended by the SparkFun guide to XBee control [1]. The zero values in the configuration allow for the receiving and transmitting ports of the device to start at a base value. Both XBees needed to be an End Device in order to allow both-way communication.

Code Break Down

The code written allows the XBee and Arduino to transmit and receive words between the devices. The code in its complete form is shown in the Appendix. The receiving and transmitting ports on the XBee needed to be set, and was done so by the following line, where port 2 was the receiving port and 3 was the transmitting port:

```
SoftwareSerial XBee(2, 3);
```

The XBee and arduino needed to be set to a baud of 9600 for the most effective communication [2]. The baud was set by:

```
XBee.begin(9600);
 Serial.begin(9600);
 Serial1.begin(9600);
```

A loop needed to be started so that the communication could occur indefinitely until the software was shut down or the devices unplugged from the computer. “Serial” is the middle ground of the shield that is attached to the arduino. “Serial 1” is the XBee device. The code reads that if the

shield is present and available, the XBee device will write and the arduino will read what is written. This section of the code is:

```
void loop()
{
if (Serial.available())
  { // If data comes in from serial monitor, send it out to XBee
    Serial1.write(Serial.read());
  }
}
```

The communication needed to go both ways, so another if statement was written for if the XBee wrote something, then the arduino would write it. This section of the code is:

```
 if (Serial1.available())
  { // If data comes in from XBee, send it out to serial monitor
    Serial.write(Serial1.read());
}
```

Finally, brackets were used to end the void loop.

Results

Successful results occur when words written in one program (either XCTU or Arduino) appear in the opposite program. This is shown in figures 3 through 6.

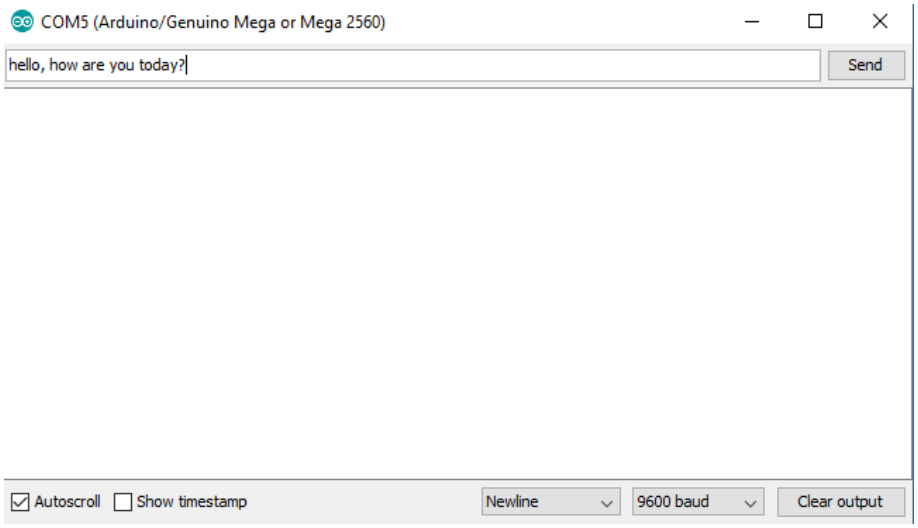

Figure 3: Typing in Arduino

In figure 3, the phrase “hello, how are you today?” was typed in Arduino. Once enter was pressed, the same exact words appeared in XCTU in the color red (figure 4), and the number of Rx bytes increased, meaning that something was received.

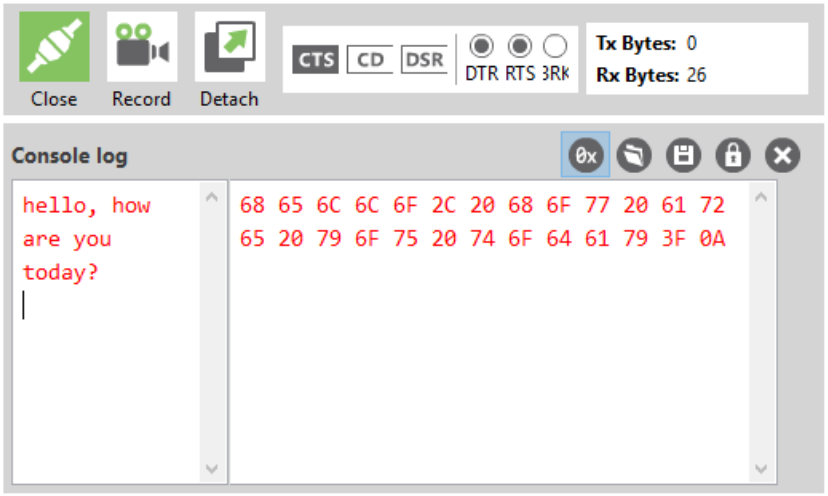

Figure 4: Response in XCTU

In figure 5, the phrase “I am doing well, thank you” was typed into XCTU. The color is blue and the Tx bytes increased, meaning something was being transmitted. Over in arduino, shown in figure 6, the same phrase simultaneously appeared in the output box.

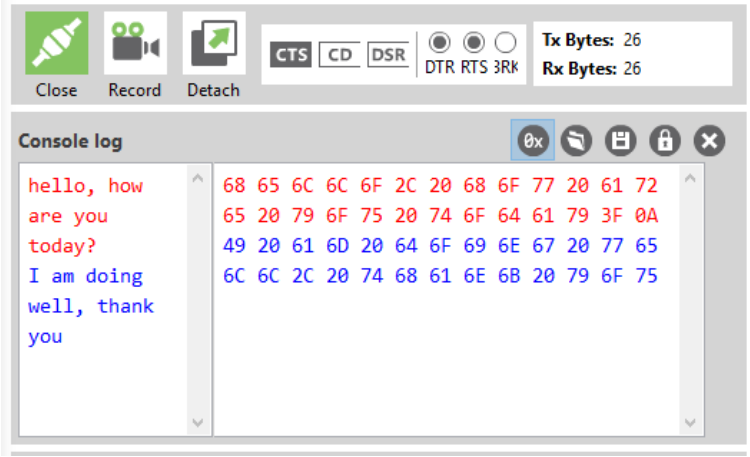

Figure 5: Typing in XCTU

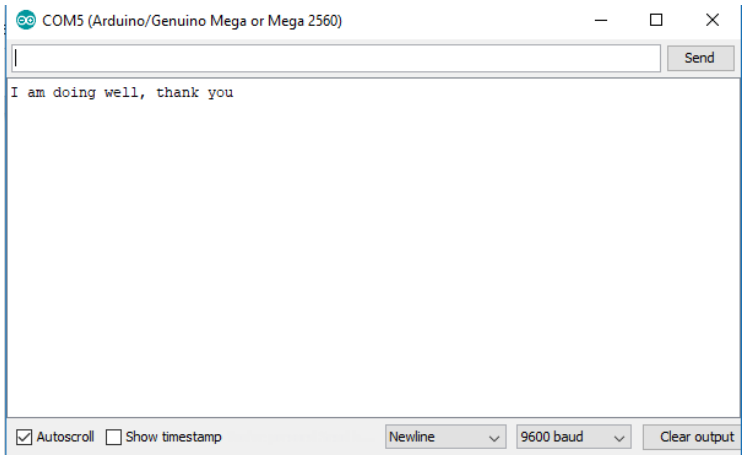

Figure 6: Response in Arduino

Other proof of successful transmitting and receiving was demonstrated in the physical shield attached to the arduino. When words were transmitted from arduino to XCTU, the “Dout” light would light up red, shown in figure 7. When words were received by the arduino, the “Din” light world light up green, faintly shown in figure 8. These lights show that the XBee attached to the arduino was successfully receiving information and transmitting information, meaning the wireless communication was complete.

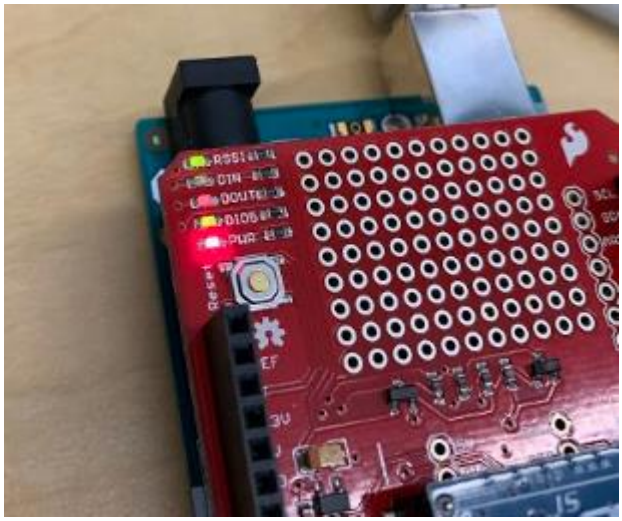

Figure 7: Dout Lights Up

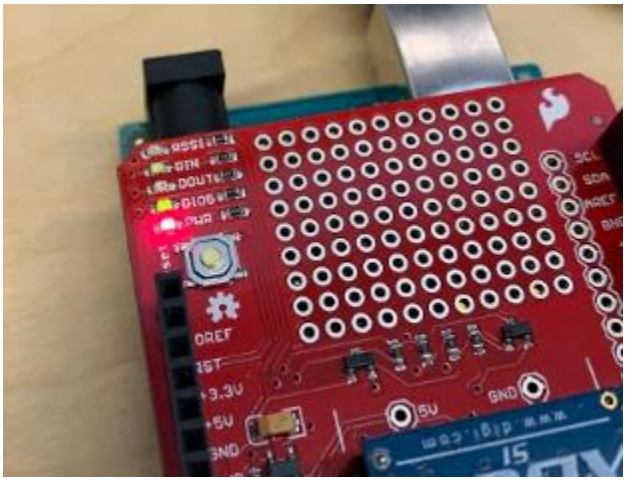

Figure 8: Din Lights Up

Conclusion

The code written for this individual analysis is used for wireless communication between arduino and XBee. This communication is important to establish before the code for data transfer can be written because it proves that a connection can be made from the foot to the arm of the client. This analysis used an Arduino Mega as the arduino device due to the number of transmitting and receiving ports on the arduino, however a Mega is a larger arduino than desired for the prosthetic design. The team will work with the EE capstone team to update the code so that it

will work for an Arduino Uno, which is smaller but has fewer ports, making the current code unsuccessful. However, due to time constraints and a late realization that a Mega was too large of a device, the code could not be constructed for an Uno before the analysis submission date. This code benefits the capstone team because it is the foundation for data transfer between XBee and Arduino. Setting up a base communication allows the team to begin testing of distance for placement of the devices, as well as start altering the code to include data and commands.

Appendix

```
#include <Servo.h>
#include <SoftwareSerial.h>

SoftwareSerial XBee(2, 3); // RX, TX

void setup()
{
  // Set up both ports at 9600 baud.
  XBee.begin(9600);
  Serial.begin(9600);
  Serial1.begin(9600);
}

void loop()
{
if (Serial.available())
  { // If data comes in from serial monitor, send it out to XBee
    Serial1.write(Serial.read());
  }
  if (Serial1.available())
  { // If data comes in from XBee, send it out to serial monitor
    Serial.write(Serial1.read());
  }
}
```

References

[1] Jimblom, Toni. K, “Exploring XBees and XCTU,” *SparkFun.* [Online]. 2015. Available: https://learn.sparkfun.com/tutorials/exploring-xbees-and-xctu [Accessed Feb. 23rd 2019]

[2] Jimblom, “XBEE Shield Hookup Guide,” *SparkFun.* [Online]. 2014. Available: https://learn.sparkfun.com/tutorials/xbee-shield-hookup-guide/all [Accessed Feb. 23rd 2019]