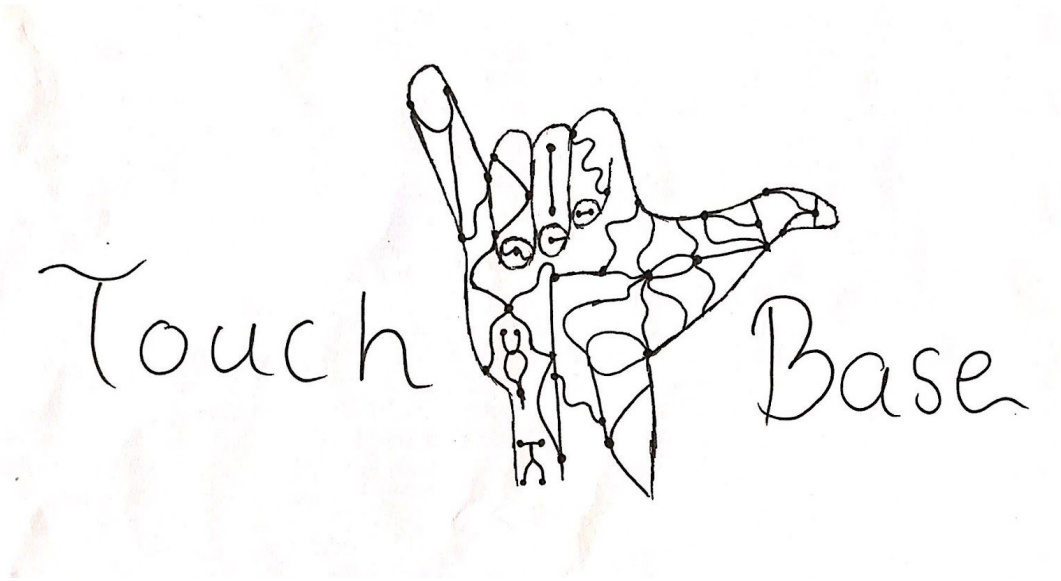
An Active Prosthetic Device



Aseel Yousef
Lihua Diao
Xiaohan Liu
Ethan Gage
Client: Dr. Kyle Winfree

Table of Contents

1. Introduction

We hope you like the prosthesis we developed for you. This is a low-cost and active control device which will help to improve the way children interact with their surroundings. We have carefully investigated the existing prosthesis, and found that they are too expensive such that a lot of families cannot afford them. Furthermore, the existing prosthesis does not have active control system, which does not enable the user to feel the sense of touch. Based on these defects, we worked with a Mechanical Engineering team to develop a more economical and user-friendly prosthesis, including an active control and feedback system.

First, we are “Touch Base” and are very honored to be chosen to work on this project as our capstone. There is a strong need for the active prosthesis, as evidenced by above, we are providing a powerful system for control fingers’ movement and feeling touch that has been custom-designed to meet your needs. Some of the key highlights include:

1. The device was developed with aspects of haptics feedback to mimic the functionality of the touch sensation.
2. The device is light enough which makes it convenient be used on a daily basis.
3. The components used in the device are cheap which makes the device affordable.
4. The batteries used to power the electrical components are rechargeable and last for more than 8 hours.
5. All design files are uploaded as open sources for future development.

The purpose of this user manual is to help you, the client, successfully use and maintain the prosthesis product in your actual business context going forward. Our aim is to make sure that you are able to profit from our product for many years to come!

1.1 Project Problems

First, our project is for children with amputated, or otherwise missing an arm below the elbow to easier manage their day to day activities, so our device should have the basic function of prosthesis control four prosthetic fingers and let them move like real fingers.

Second, most of the prosthetic devices are hardly actively operated which makes the user experience less perfect, because in real world, hand can not only be controlled movement by our brains, but also feedback the touch sense to us. In order to give the user a closer experience with the real arm, we planned to add feedback subsystem in our device.

Third, we need motors and other components to act out finger’s movement. Because this motor is mounted on child’s arm, and children cannot handle heavy weighted devices, which makes them feel tired and uncomfortable, the components should be light. Besides, our goal it to make an economy device that every family can afford, so the components on our devices should be light and cheap.

1.2 Project Subsystems

1.2.1 Foot-To-Hand

In order the resolve the first problem, being able to control the device, we added what we call the Foot-to-Hand subsystem, which would allow the user to control fingers wirelessly with pressure sensors beneath the users toes. There are in total 5 pressure sensors under the toes, three on one foot, two on the other, these are connected to Xbee wireless control modules, one of which, the one with three sensors, has an arduino uno to assist with communication. The transmitter module starts by reading its sensor values, and transmitting them to the second transmitter module, the one with the arduino, at 50hz. The Arduino Uno, then reads the transmitted data, reads its own sensor values, and repackages all five sensor values to be sent to the master receiver. During proper operation it enables the device to act such that when the user presses on the toe sensors, the sensors can detect the pressure and send data to motor on the arm through Arduino, Xbee, then the servo motor controls the movement of fingers. This subsystem achieves the control fingers function of the prosthesis.

1.2.2 Pressure Feedback

In order to give sense of touch to our client, we added pressure sensor on the hand and mount a constriction band encircling the user's arm. The band will constrict or expand based on the grip. In this way, we simulate the touch sense, and feedback to user.

1.2 3 Haptics Feedback

The subsystem have same goal with Pressure Feedback subsystem. There is a vibration motor on the arm, and some pressure sensors at the hand. The sensor can detect pressure and control vibration motor on the arm. It allows to adjust vibration pattern and intensity. As a secondary feedback system, the subsystem enrich the sense of touch of the user.

1.3 Design Process

To choose the best solutions to the project problem, we brainstormed and created a mind map of possible systems to solve the project problems. We picked a few of the systems from our mind map, outlined in the document as our systems for their low cost, ease of implementation, and simplicity. Then we brought our solutions to meet with client, and discussed together to choose the best solutions. Based on the solutions, we divided our project to subsystems and established the initial prototypes. Next, we discussed with out mentor Dr. Winfree, GTA Demetria, and ME team and got more suitable advices. Then we modified and refined the prototypes into the current subsystems.

We ended up with the three main subsystems outlined, the Foot-to-Hand subsystem to allow the user to control the device, and the two feedback systems, the pressure and haptics subsystems, which give the user the sensation of touch in the device. Combining these subsystems with the 3d-printed arm, we are able to accomplish all of our original project requirements. With the 3d printed components, we are able to keep the cost down to make sure many families can afford the device. The batteries on each of the modules contain enough power to power the device for 8 hours, or a full working day. Finally the subsystems give back the functionality, and the sensation of a human hand back to the user.

2. Installation

2.1 Introduction

This section goes over the device installation, including programming requirements, and customer-specific setup of the device. To install the given program files, the user will need to download the Arduino IDE, available: https://www.arduino.cc/en/Main/Software. Furthermore, the user will need to download two libraries into the Arduino IDE, this is done with *tools -> Manage Libraries* and search for redbot, and xbee separately, make sure to download the latest versions.

2.2 Programming

2.2.1 RedBot Mainboard

Take the file named “Receiver.ino” and, using a usb mini cable, upload it to the RedBot Mainboard, figure A2, with the Arduino IDE, board type Arduino/Genuino uno.

2.2.2 Arduino Uno

Similar to above, take the file named “Transmitter2.ino” and upload the program to your Arduino Uno using the standard arduino cable.

2.3 Feedback System Install

2.3.1 Pressure Feedback

The band used in this subsystem has a velcro material at both ends to make it adjustable to the user. The size of the band will be taken by measuring the area above the elbow while the muscles are flexed and unflexed to prevent the band from interfering with the range of motion of the user. A picture of the constricting band setup is shown in Figure D6 & Figure D7.

2.3.2 Haptics Feedback

The size of pressure sensor is determined by the area of the prosthetic arm fingertip. There is no limit to the position of vibration mini motor as long as it can be in touch with user’s muscle within the length of the cable.

2.4 Wireless Modules

2.4.1 Transmitter 1

This module consists of a lone XBee module on an Xbee explorer regulated board, figure A1, with 2 sensors attached to its DIO0 and DIO1 ports, figure B1. The user will start by soldering leads from the + and - terminals on the Charger/booster figure A5, to the 5v and GND pins on the Xbee explorer regulated board, as well as wiring the a connection between the 5v and the RES pin on the xbee explorer regulated, this is used as a voltage reference to compare the sensor values to. The sensors need to be hooked up in such a way that power, 5v, goes to one of the leads of the sensor, and the 3.3kOhm resistor connects to other side of the sensor to ground. The user needs to send a wire from between the resistor and the sensor to the DIO ports to ensure proper sensor reading. Once mounted in the 3D printed housing it should look similar to figure D4.

2.4.2 Transmitter 2

This module consists on an xbee wireless module connected to an arduino with an XBee shield, figure A3, as well as three pressure sensors connected to the arduino's analog inputs a0, a1, and a2, figure B2. The user will start by soldering leads from the + and - terminals on the charger booster board to the 5v and gnd pins on the arduino uno respectively. Next, using the perf board on the Xbee shield, build the resistor network for the three sensors, using the same method as listed above, connecting the sensors to the appropriate analog input ports.Once soldering has completed it should look similar to figure D5.

2.4.3 Receiver

This circuit consists of the Redbot Mainboard, figure A2, with the feedback systems, and hand motors attached, figure B3. As always start by soldering leads from the + and - terminals of the charge booster board to the V_IN + and - terminals on the RedBot Mainboard. Now connect the servos, one to each of the four built in servo ports on the redbot main board. The fifth servo has the signal, yellow, wire attached to the the Left Motor signal pin, the unlabeled pin beneath the RED/BLACK labels on the left motor port, and the 5v and Gnd side rail headers. Finally attach the haptic motor, figure A7, to the Red and Black inputs of the left Motor.

2.5 Turning on Device

In order, flip the on switches to the first transmitter circuit, then to second transmitter circuit, and then the on switch to the redbot board. Assuming proper installation, and charged batteries, the device should turn on, signified with the diagnostic LED's on each of the circuits lighting up.

3. Configuration and Use

3.1 Introduction

After the code has been successfully compiled and uploaded to the circuit boards, the next task to complete is configuring the device for the user. For the main configuration of the device and the XBee configuration software XCTU, available: https://www.digi.com/products/iot-platform/xctu. We will go over what the user needs to do with their XBees to ensure proper wireless communication, as well as describing the location, and uses of the sensors in the device. Finishing with a description of how the user would control the device after final assembly and configuration.

3.2 Foot to Hand Subsystem

3.2.1 XBee XCTU Configuration

First open XCTU and plug an XBee module into your computer using the XBee explorer, figure A4 . You should see what looks like the image below:

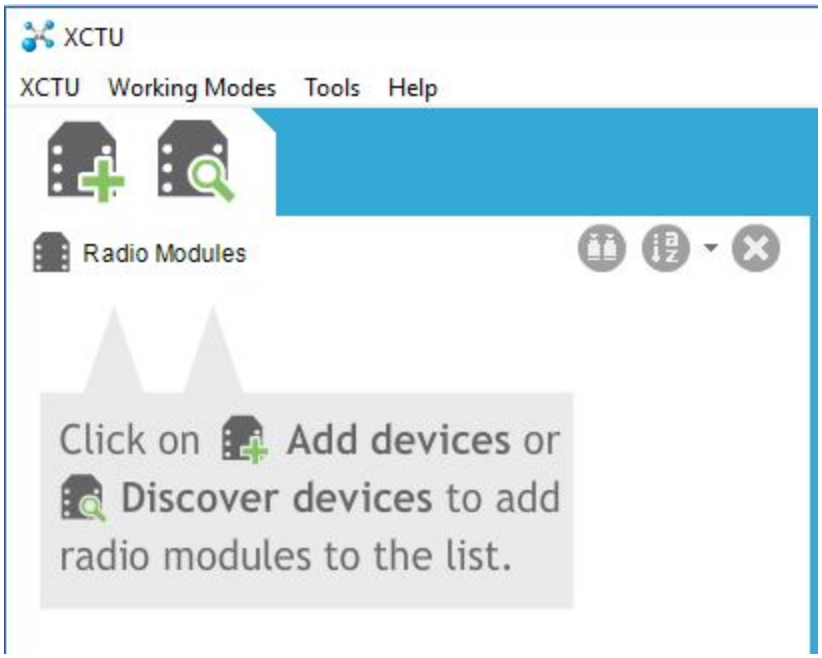


Figure 1: XCTU startup page

Now Click the ‘Add devices Icon and you will see the following pop-up:

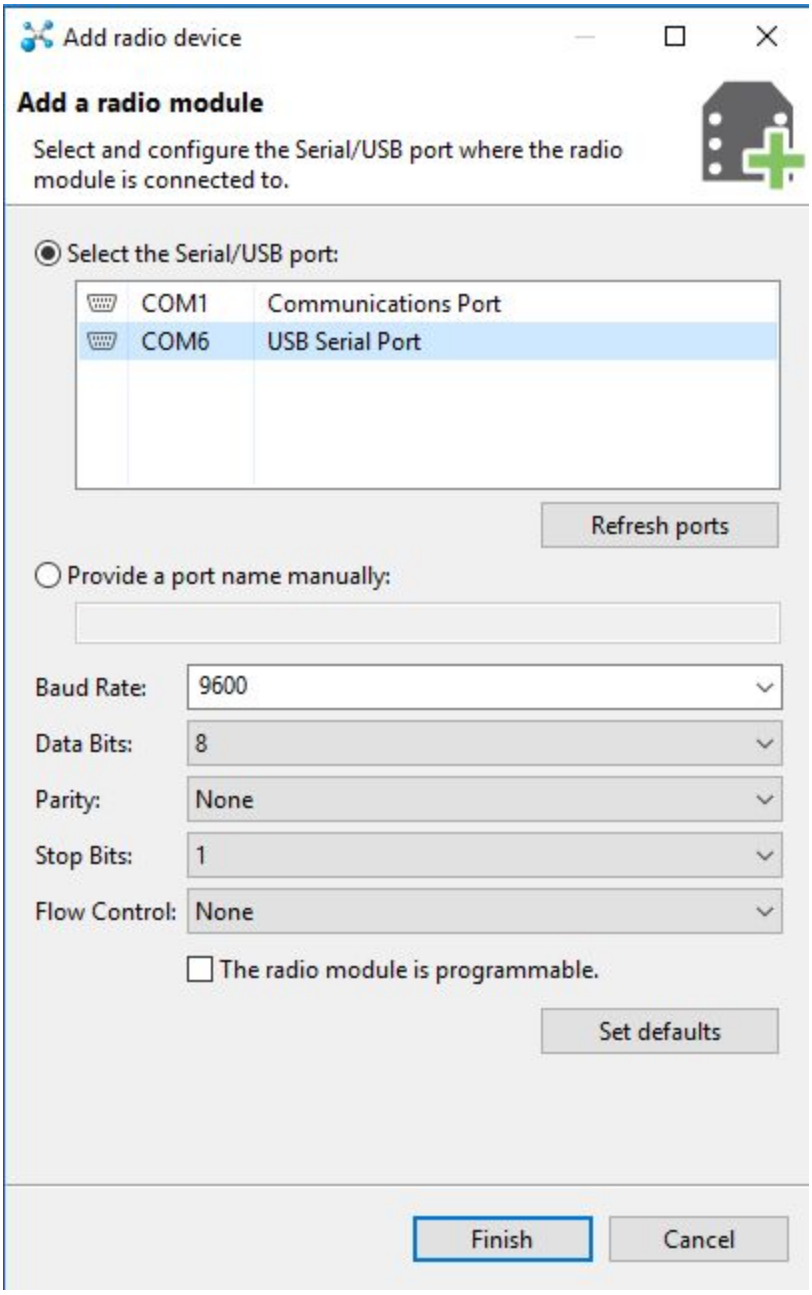


Figure 2: XCTU add radio device pop-up

Select the appropriate com port and click finish, and the right side of the application will update. In this project there are three xbee modules per individual. The user will only be editing a few boxes for each Xbee module, below show those options.

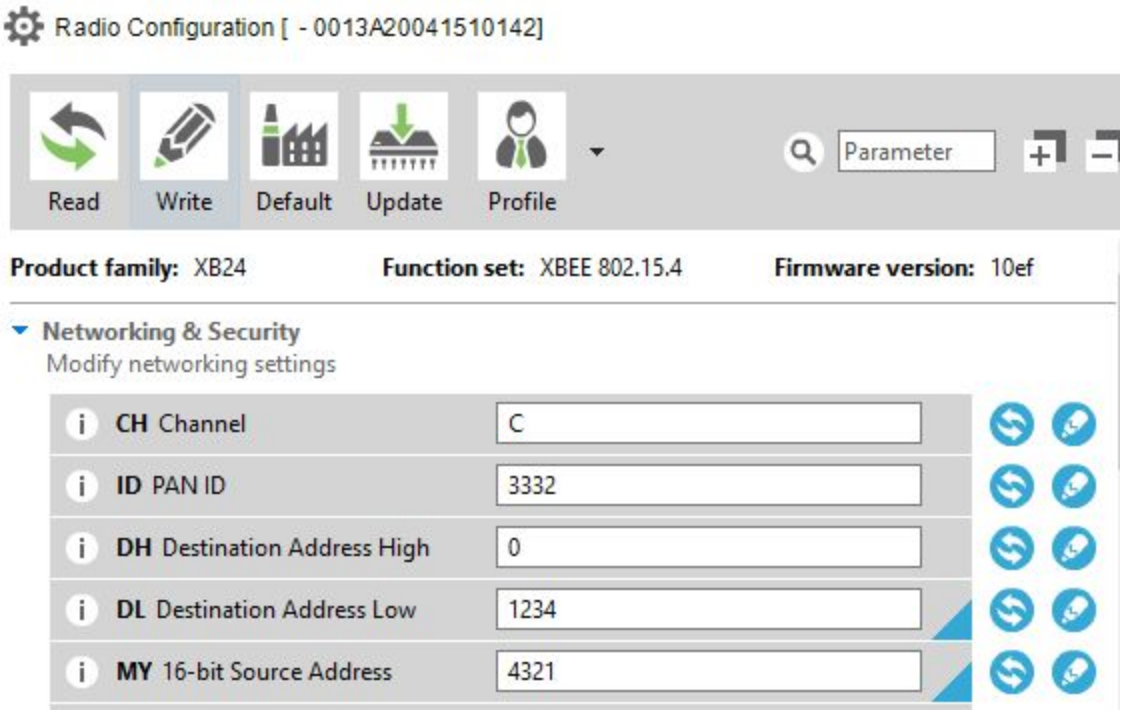


Figure 3: XCTU addressing configuration

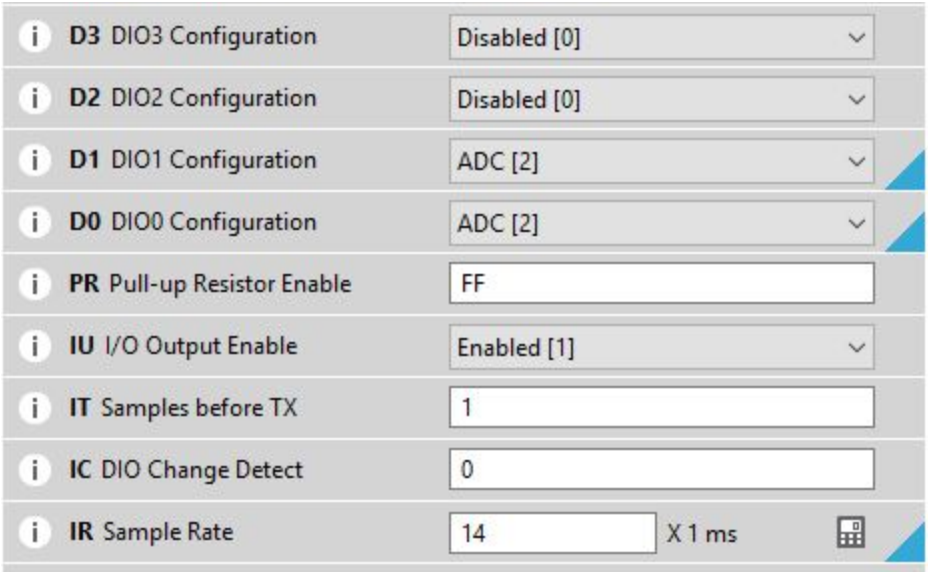


Figure 4: XCTU DIO configuration

For each individual one should generate a unique pan id, ID, a four unit hexadecimal number 0000 to FFFE, this is the unique number that is linked to the xbees directly talking to each other. So to avoid errors in wireless communication, this number should be unique per individual with the arm. For this project there are three Xbee modules configured as follows:

Table 1: XBee XCTU configuration

XBee 1 - Transmitter 1 DL - 1234 MY - 4321 DIO0 - ADC[2] DIO1 - ADC[2] IR - 14	XBee 2 - Transmitter 2 DL - 5555 MY - 1234	Xbee 3 - Receiver DL - 1234 MY - 5555

For transmitter 1, DIO0 and DIO1 have to be set to ADC[2], with the IR set to 14, this means the XBee will read the values from the sensors connected to its DIO0 and DIO1 pins at 20 times a second, 14 in hex is equal o. Though the rest of the values for the XBee can be changed, so long as the DL of transmitter 1 equals the MY of transmitter 2, and the DL of transmitter 2 equals the MY of the receiver.

After configuring each of the XBee, you can replace them into their appropriate circuits, and verify communication with the Arduino Serial Plotter from the receiver circuit

3.2.2 Arduino Code

In the transmitter 2 code, there is one line that will need alteration depending on how the XBee configuration was completed. If you followed the directions from the example, ignore this section. If you changed the above numbers at all, you will need to update the “0x5555” in the line below to whatever your MY value of your receiver is. Beyond that we describe how to make minor alterations to the code in other sections of this manual.

```
    tx16 = Tx16Request(0x5555, payload, sizeof(payload));
    node.send(tx16);
}
```

3.2.3 Hardware Setup

This subsystem contains the five total sensors from the two transmitters, writing values to five servo motors in the arm, controlling the fingers. The user will need to define which foot will be their dominant foot to determine which transmitter module goes on which foot. For each foot, there will be a sensor mounted under the big toe, and the ball, just behind the pinkie toe, the fifth sensor goes on the dominant foot underneath the ball behind the ‘pointer’ and ‘middle’ toes. It should be noted these are positions we determined with minimal testing, the user should have ultimate decision over where they like the sensors to be mounted.

3.2.4 Operation

As mentioned before with the receiver, RedBot, circuit plugged into a computer one can verify communication by opening the serial plotter *Tools -> Serial Plotter.* The receiver code is written to graph the five sensors values to the serial plotter to verify proper communication and troubleshooting.

The code to wirelessly receive and remap the sensor values to the servos is shown in figure C1, in the first part sensorVal1 and sensorVal2 represent the two sensors on the recessive foot/transmitter 1 module, where the vals 3 -> 5 represent the 3 sensors of the dominant foot/transmitter 2 module. If the user does not like which sensors control which servos by rewriting that shown block of code in the “Receiver.ino” file. For example, in the lines where it is written “servoVal1 = map(testVal, ……”, figure C1, swapping the servoVal1 with another servoValX from another line would swap those two servo/sensor pairs.

3.3 Feedback Systems

There is 2 different Feedback Subsystems implemented in the device; The Pressure Feedback Subsystem indicates the amount of pressure being applied on the hand, and The Haptics Feedback Subsystem indicates the functionality of the sense of touch.

3.3.1 Pressure Feedback Hardware Setup

This subsystem contains one pressure sensor and a servo motor that is connected to a constricting band. To setup; place the pressure sensor on the palm of the device, and wrap the band above the elbow. Pressure Feedback schematic and circuit are shown in Appendix D.

3.3.2 Pressure Feedback Software

Use an Arduino code to enable the pressure sensor to control the servo motor. Whereas; when pressure is high on the hand of the device, the band constricts around the area above the elbow, and when pressure is low, the band expands. Additionally, the pressure sensor should be able to read different levels of pressure and control the servo motor based on the readings. An example code is shown in Figure C2.

3.3.3 Haptics Feedback Hardware Setup

The haptic feedback consists of pressure sensors and vibration mini motor for mechanical part and arduino for software part. The pressure sensor is located on the tip of the finger in order to get in contact with objects and get pressure. The vibration mini motor is placed on the cuff in order to close the tricep tightly and let the user feel the sensation. The specific position on the cuff is not regulated strictly and it can be adjusted according to the user’s comfort. The arduino we used is the mainboard which integrated all arduino code and is placed on cuff motor assembly box. Haptic feedback schematic is shown in figure B3 and test circuit is shown in figure D3.

3.3.4 Haptics Feedback Software

Use the Arduino code to make the pressure sensor control the vibration mini motor. When the finger of the prosthetic arm touches the object, the object gives pressure to the pressure sensor. Arduino collects the pressure value and judges whether it touches the object

according to the value. If there is pressure, control the vibration of the vibration mini motor. If there is no pressure, the vibration mini motor will not vibrate. An example code is shown in Appendix C figure C3.

4. Maintenance

4.1 General

To maintain usage of the device, It is recommended to charge each of the batteries on a daily basis, as a single dead battery can greatly affect the capability of the device.

4.2 Foot to Hand

This subsystem uses pressure sensors in constant direct contact to the human individual when in use. As such we can expect that these sensors will wear over time, and will need to be replaced. This can be done simply by purchasing, and re-crimping a sensor to connect to the wires coming from each wireless module.

Since this system uses batteries that will be constantly charged and discharged, their lifetime will be limited and need to be replaced after some time. To do so, one simply needs to open either wireless module, and remove the battery pack from the charge booster on the device (these connections are stiff and may take more force than one expects to remove.

Similarly with battery life, the servos will have a life expectancy where eventually they will not be able to maintain a stable position, or provide enough force to be considered usable. When this happens, the user will need to open the forearm of the device, and swap out the servo in question, simply removing the screws holding it in place, and then swapping over the head of the sevo, the 3d-printed component that the cable attaches to.

4.3 Pressure Feedback System

To maintain a proper usage of this subsystem, the band should be functioning properly, since the band could lose its tension and become loose, it is recommended to replace the band over time. Additionally, the pressure sensor used in this subsystem will be exposed on the palm of the hand, it could wear out from constant use and friction, therefore it will also need to be replaced over time. The battery that powers this subsystem should also be changed constantly to avoid running out of battery resulting in the system dysfunctioning. Finally, The servo motor used in this subsystem is also subjected to wear out, since it expected to lose its steady rotations, and it might burn out therefore, to maintain functionality of this subsystem the servo motor should be replaced regularly.

4.4 Haptics Feedback System

In order to maintain correct use of the subsystem and the normal operation of the vibrating mini motor, we need to maintain the components of the system. The pressure sensors used in this subsystem are placed on the fingertips and wear out as the user uses them, so they need to be replaced over time. In addition, the batteries used in the system should be fully charged, so they need to be recharged during idle time to avoid battery depletion and system malfunction. Finally, the parts used in the system are very fragile. For example, the vibration motor has a very thin connecting line and the pressure sensor is very small. When the user uses it, please try not to fold or bend these parts.

5. Troubleshooting Operation

5.1 Foot to Hand

For this subsystem there were a number of issues during development that could arise during standard operation. First is communication errors, where the wireless communication fails for some reason or another. Next was servo errors, where the servos either were not writing the appropriate values from the sensors, or were otherwise inoperable.

For communication errors, double check that each XBee in your system is properly configured, meaning each has the appropriate MY and DL addresses, that they are all on the same Pan ID, and the first transmitter has the appropriate DIO0, DIO1, and IR rates specified. To determine if the signal is transmitting, the user can plug the receiver circuit into a computer using a mini usb cable, and with the Arduino IDE, open the serial plotter, *Tools -> Serial Plotter*, the code is written to graph the five sensor values, these should appear as different colored lines moving in response to the pressure sensors being activated. They should be values between 0-255, if you are seeing a dip in the max range of the sensor values, check the voltage of the batteries with a multimeter, they should be between 3.4 and 3.7 volts, much below then and it will affect device capability.

Because the servos operate using a PWM line as their control, there are a few issues than can occur during the operation of the device. If the PWM is not able to maintain consistent duty ratio, or if the voltage dips too low, one will see a jittery motor, or one that won't operate at all. To fix, make sure that all connections between the main board and the servos are secure, and that the batteries are charged. If this does not fix the problem, it may be a faulty motor, and will need further inspection and possible replacement.

5.2 Pressure Feedback System

For this subsystem, the velcro material used on the band could detach due to constant use; this problem could be solved by simply applying adhesive to reattach it. This is also

applicable to the servo and band connection, that could break, applying adhesive to reattach them should work as well.

5.3 Haptics Feedback System

For this subsystem, some errors may occur during use. If the vibration mini motor does not vibrate, most of reasons are hardware parts such as the connection line. The pin line of the vibration mini motor is very thin, please check if it is connect problem due to breakage. If it is, the user could choose replace a new vibration mini motor or reconnect the line by soldering or other thing.

6. Conclusion

To sum up, we successfully completed the prosthesis and the device meets the client requirements. This device is similar to a real arm and using cheap and lightweight components, which enable more family to benefit from it. We are very happy to help those in need, and best wish the client many happy years of productive use of the product from your [Touch Base] prosthesis developers: Ethan Gage (eog6@nau.edu), Aseel Yousef (ay339@nau.edu), Lihua Diao(ld549@nau.edu), Xiaohan Liu(xl225@nau.edu). Even though we are all moving on to professional careers, we would be happy to answer short questions in the coming months to help you get the product deployment and operating optimally.

7. Appendices

Appendix A

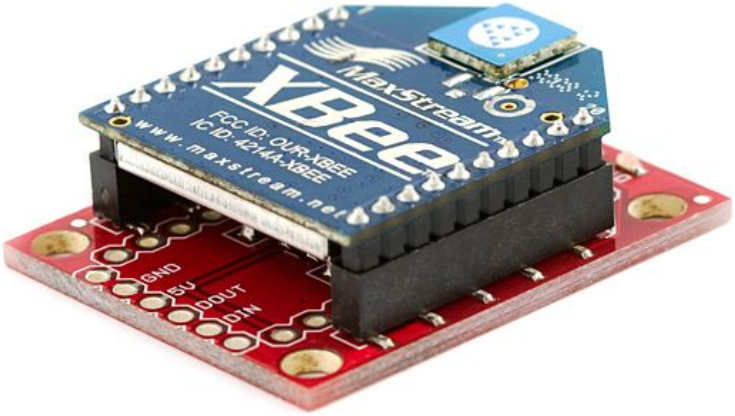


Figure A1: Xbee Explorer Regulated w/ Mounted Xbee.
https://www.sparkfun.com/products/11373

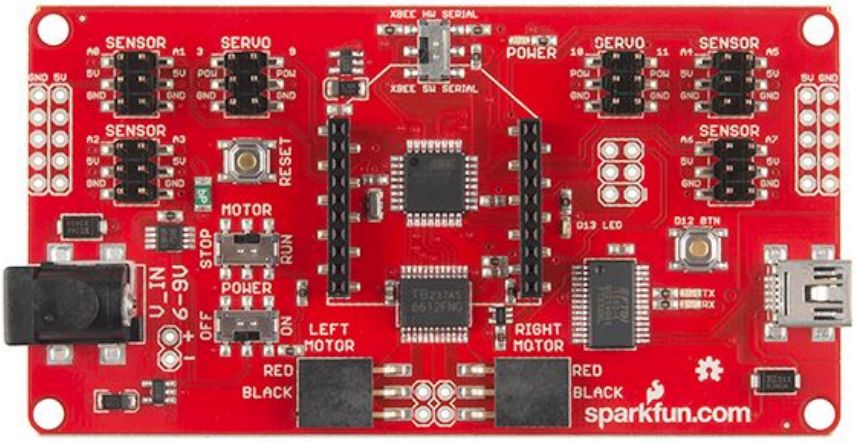


Figure A2: RedBot Mainboard.
Used as the main programmable circuit board for the device
Referred to as Receiver in Foot to Hand subsystem. Note the power and Motor switches, these need to be in the ON and RUN states for proper operation. As well the switch in the top middle should be switched to XBEE SW SERIAL for the communication to properly work. The Haptic motor was wired into the right motor slot, with wiring: red to red, blue to black. The 5th hand servo will be connected to the left motor port with the signal pin, yellow wire on servo, connecting to the 3rd pin on the left motor port, and red/black servo wires connected to the 5V and ground header rails on the sides on the board.

https://www.sparkfun.com/products/12097

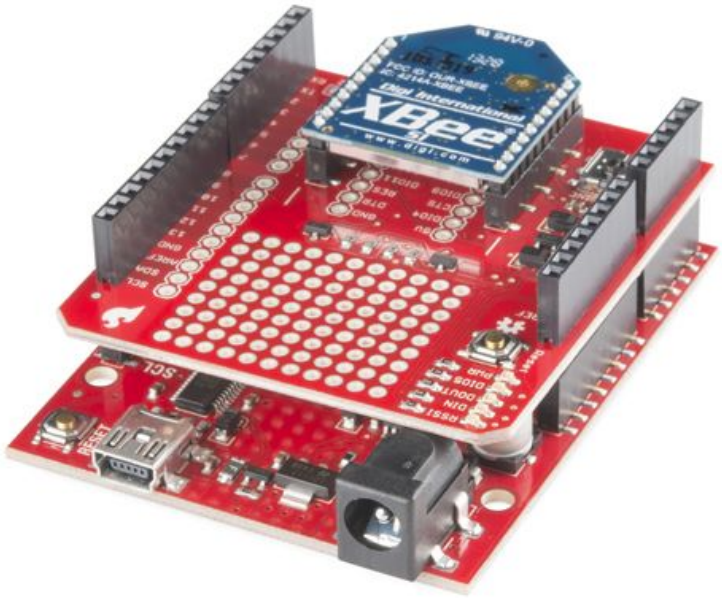


Figure A3: XBee Arduino Uno Shield with mounted Xbee, attached to arduino Uno.
https://www.sparkfun.com/products/12847

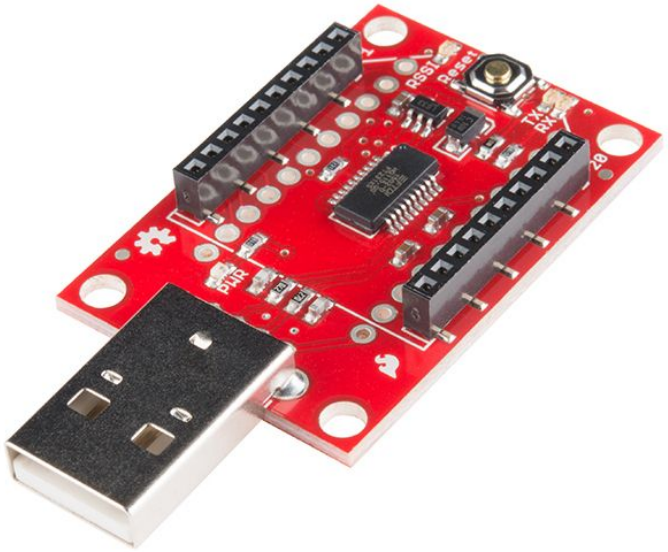


Figure A4: XBee USB Explorer Board.
Used to connect the XBee wireless communication modules to a computer for configuration.
https://www.sparkfun.com/products/11697

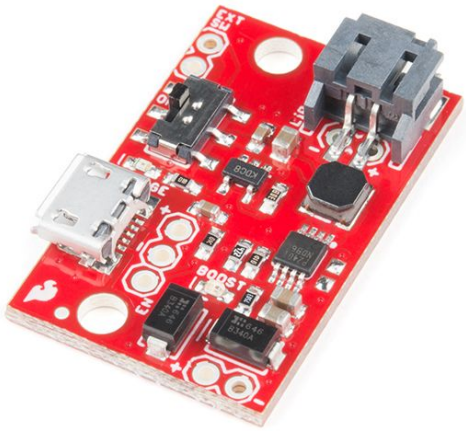


Figure A5: LiPo Charger/Booster 5V/1A.
https://www.sparkfun.com/products/14411

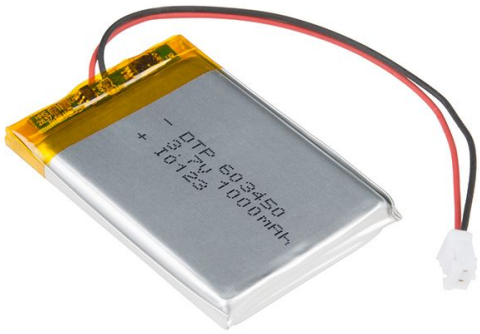


Figure A6: LiPo Battery.
1 Ah: https://www.sparkfun.com/products/13813
2 Ah: https://www.sparkfun.com/products/13855

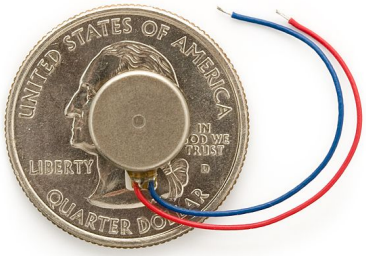


Figure A7: Haptic Feedback Motor.
https://www.sparkfun.com/products/8449

Appendix B

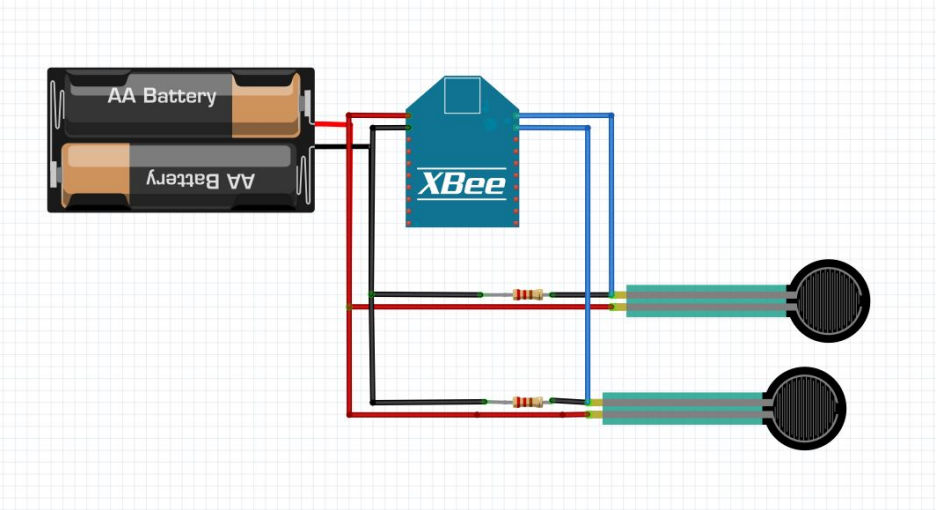


Figure B1: Transmitter 1 Circuit.
Consists of an xbee module mounted on a Xbee explorer regulated board like in Figure A1.

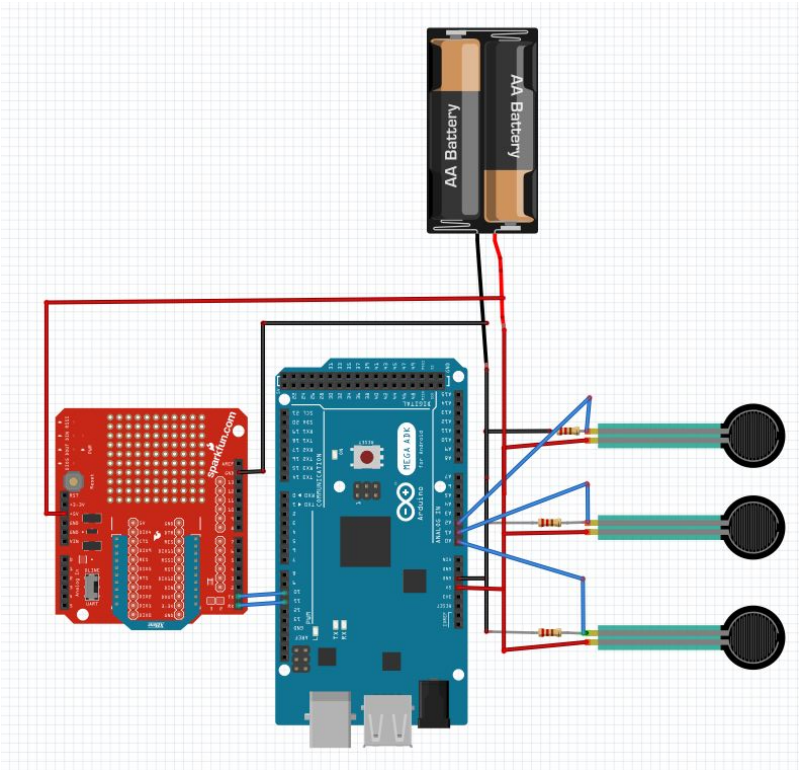


Figure B2: Transmitter 2 Circuit.
This will look like figure A3 when assembled.

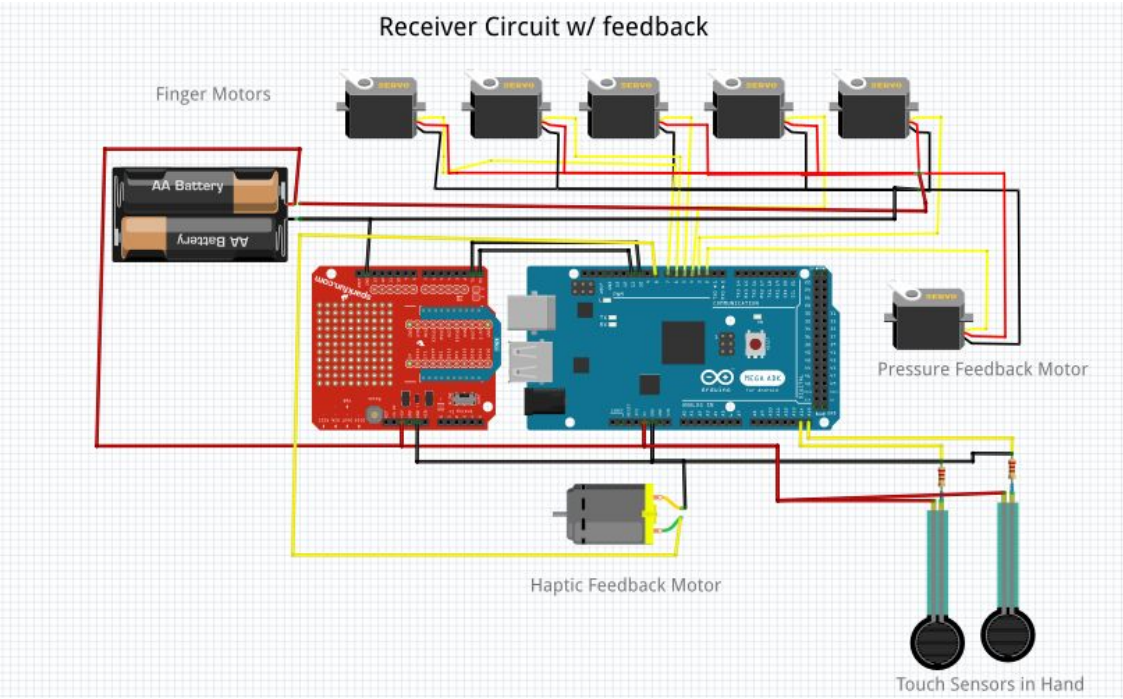


Figure B3: Receiver Circuit with Both feedback systems installed
Image shows Arduino Mega with XBee shield. Our completed system used the redbot mainboard instead, and had only the hand servos, and one touch sensor in hand, but included the haptic feedback motor. The servo signal pins(yellow/white) will be connected to the digital PWM servo pins on the Redbot board number 3, 9, 10, and 11, while the power (red/orange) and ground(black/brown) wires are connected to the 5v and GND rails on the sides of the board.

Appendix C

```
sensorVal1 = String(rx16.getData(0), HEX);
sensorVal1 += String(rx16.getData(1), HEX);
sensorVal2 = String(rx16.getData(2), HEX);
sensorVal2 += String(rx16.getData(3), HEX);
sensorVal3 = String(rx16.getData(4), HEX);
sensorVal3 += String(rx16.getData(5), HEX);
sensorVal4 = String(rx16.getData(6), HEX);
sensorVal4 += String(rx16.getData(7), HEX);
sensorVal5 = String(rx16.getData(8), HEX);
sensorVal5 += String(rx16.getData(9), HEX);

sensorVal1.toCharArray(testChar, 8);
testVal = StrToHex(testChar);
servoVal1 = map(testVal, 0, 1023, 0, 180);

sensorVal2.toCharArray(testChar, 8);
testVal = StrToHex(testChar);
servoVal2 = map(testVal, 0, 1023, 0, 180);

sensorVal3.toCharArray(testChar, 8);
testVal = StrToHex(testChar);
servoVal3 = map(testVal, 0, 1023, 0, 180);

sensorVal4.toCharArray(testChar, 8);
testVal = StrToHex(testChar);
servoVal4 = map(testVal, 0, 1023, 0, 180);

sensorVal5.toCharArray(testChar, 8);
testVal = StrToHex(testChar);
servoVal5 = map(testVal, 0, 1023, 0, 180);
```

Figure C1: Arduino Code for receiver module to decode the received packets.
In the event the user does not like which sensors map to which servos, this is block of code to alter.

```
servo.write(servo_value);
Serial.print("sensor: ");
Serial.print(sensor_value);
Serial.print(", servo: ");
Serial.println(servo_value);
```

Figure C2: An example of Arduino code for the Pressure Feedback system.

```
void setup() {
  Serial.begin(9600);
  pinMode(3, OUTPUT);
  digitalWrite(3, LOW);
}

void loop() {
  sensor_value = analogRead(0);
  if (sensor_value && !run_flag) {
    digitalWrite(3, HIGH);
```

Figure C3: an example Arduino code of the the Haptic Feedback system.

Appendix D

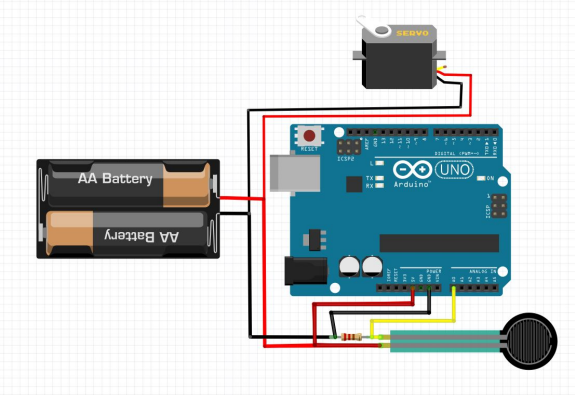


Figure D1: Pressure feedback schematic.

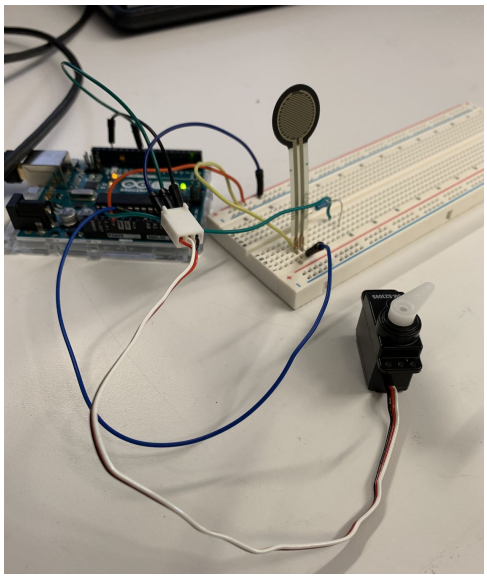


Figure D2: Pressure Feedback testing circuit.

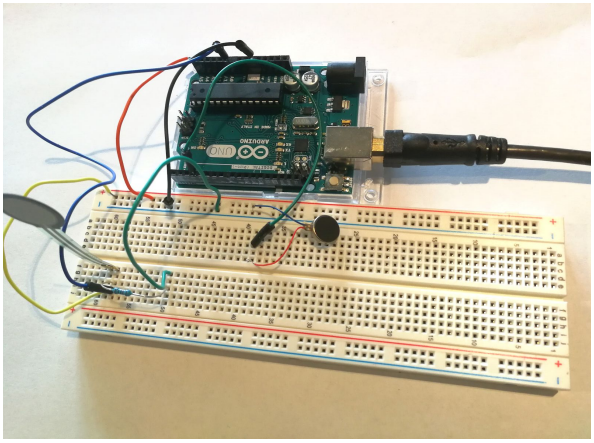


Figure D3: Haptic Feedback testing circuit.

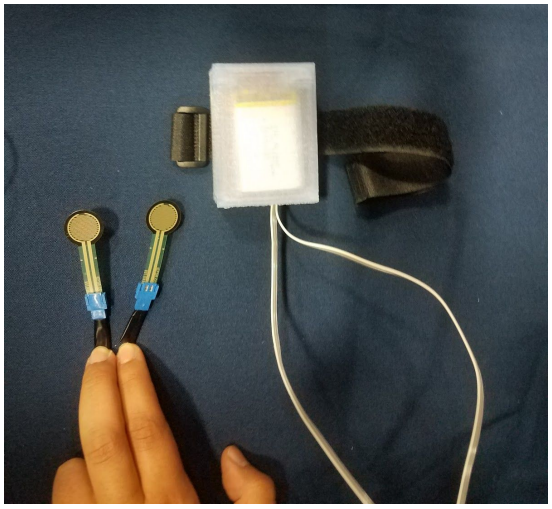


Figure D4: Completed Transmitter 1 Module.

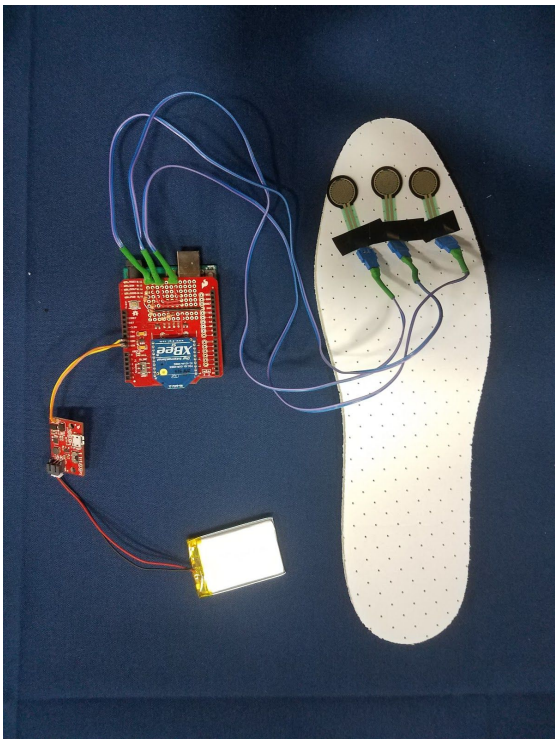


Figure D5: Completed Transmitter 2 Circuit with example placement of sensors in insole.

Figure D6: Constricting band with the velcro shown.

Figure D7: Constricting band connected to the tip of the servo motor.