American Sign Language Assistant (A.L.S.A.)

Isabel Atanacio, Colin Fox, Charlie Munoz, and Daniel Sarmiento

Dept. of Electrical Engineering and Computer Science, University of Central Florida, Orlando, Florida, 32816-2450

***Abstract* — A.L.S.A. is a glove that converts American Sign Language to computer generated speech to help those who are hearing-impaired to communicate in everyday life. Connecting via bluetooth to a computer system to predict and output the signed word/phrase. Using a microcontroller to interact and communicate with various sensors. Transmitting this data to a host computer which would then, using a trained machine learning model, would predict which word/phrase the user is signing.**

***Index Terms* — Assistive devices, bluetooth, deafness, machine learning, speech synthesis, supervised learning, support vector machines.**

I. Introduction

Designed and engineered to help the communication of those who are hearing-impaired with anyone else. This glove uses different sensors that track the movement and position of the hand to allow the translation of different hand gestures. The project is implemented using a bluetooth wireless technology for fast data transfer. This project was executed on a regular handglove that houses a flex sensor on each finger as well as an IMU on the index finger for better accuracy. Also, to be able to differentiate between some letters, we added two contact sensors on the index finger and the middle finger. The glove contains the PCB which accommodates a second IMU as well as the microcontroller, bluetooth module, voltage regulator, and voltage sensor. The PCB was specifically designed to fit on the top part of the hand to allow for easy hand gestures. All of the different sensors are connected to the PCB which then uses the microcontroller to process all of the sensor data to be ready to send to the compute via bluetooth. After this data is transferred to the computer, it is then processed using a machine learning model which teaches the computer each letter using different gestures. The glove will be able to recognize every letter in the alphabet. Ultimately, these letters will be generated to speech.

II. System Components

This project was developed with some key physical system components, whether they were purchased or simply designed by our own hardware team. Below is a short technical description of these system components:

A. Microcontrol Unit (MCU)

The core of this project is an Atmel low power microcontroller chip, the ATmega328P. This microcontroller was chosen due to its large online community support and because it includes fourteen digital input-output pins which was necessary for this project. This chip controls the flow of data from the sensors to the computer.. Since some of the sensor data is analog, this chip is required to convert from analog to digital so the data can be transmitted to the computer

B. Flex Sensors

The flex sensors are manufactured by Spectra Symbol, one of the best flex sensors available on the market. These are in charge of measuring angle displacement, bends and flexes generated by each hand gesture. The way they work is by increasing the resistance as the flex sensor is bend, so it basically acts as a variable resistor. It has a life cycle of more than a million flexes and a power rating of 0.5 W continuous. These flex sensors come in different sizes, we chose two different sizes. One of them is 2.2” which was appropriate for the thumb, and the other 4.5” size to the rest of the fingers. Although, with the flex sensors alone, most all the letters in the alphabet are able to be translated, we decided to go further and increase the accuracy by adding other sensors such as IMUs, and contact sensors. These are described next.

C. Inertial Measurement Unit (IMU)

The inertial measurement unit or IMU was chosen to have more precise readings in order to differentiate

between very close signs and to allow better readings by tracking the position of the hand. We chose the LMS9DS1 IMU by sparkfun. This is a very unique and versatile motion-sensing system in a chip. It houses a 3-axis accelerometer, 3-axis gyroscope and a 3-axis magnetometer. This IMU is capable of measuring angular velocity, acceleration, and heading. For our design we were only required to use two of the measuring properties which are the gyroscope and accelerometer for a total of a 6-dimension orientation. This sensor takes care of the position of the hand and orientation. In this project we used two of these IMU's for more precise readings.

D. Contact Sensors

To help differentiate between similar signs like 'R', 'U' or 'V' , a set of two contact sensors were developed by the hardware team. We simply soldered the end of the wires to a copper tape to act as act sensor. These basically act like switches, when they are closed, they act as a closed circuit which creates a high in the digital pin. For this project they are located on the index and middle finger. This design for the contact sensors uses two digital pins which was primarily created for the three letters mentioned before. Therefore, the system will produce two different high signals depending on the contact sensor being closed. These signals along with the other data collected from the other sensors will generate enough data to translate these similar hand gestures.

E. Bluetooth

The wireless communication was executed by a BlueSMIRF Gold bluetooth module assembled by sparkfun. This is bluetooth uses 3.3 V or 5V for power and has a range of 100 meters, which was more than necessary for the purpose of the project. Since the data rate is very important for the project due to the machine learning implementation, this bluetooth module uses a RN-41 chip which is capable of 2400-115200 bps baud rate. This project requires lots of data being transferred wirelessly and constantly which made this new bluetooth module ideal for the design and purpose. This RN-41 chip that is integrated into the printed circuit board design. This chip also contains the bluetooth antenna.

F. LiPo Fuel Gauge Voltage Monitor

The Maxim MAX17043G+U was chosen to be able to monitor the battery. This piece of equipment was needed in order to find out if the battery is running low and if it needs to be charged. This is key because the project cannot die during the presentation. This liPo voltage monitor operates at a 2.5 -5 Voltage. It has a 30 mV accuracy up to 10 Volts with a typical current draw of 50 uA.

G. Battery Charger and Voltage Regulator

Since this is a low power project and all the components can run on 3.3 volts. There was a need to step down the voltage from 3.7 volts when the battery is fully charge to a 3.3 voltage in order to prevent any damages to the circuit components. Also, it was important to charge the battery directly from the circuit board while connected to the project simultaneously. For this reason, we chose the Microchip MCP73831/2 for charging purposes. For regulating purposes, we chose the Texas Instruments TPS61200 chip which has a low voltage boost converter.

III. System Concept

The system initiates once power is applied and all the sensors have connected to the microcontrol unit (MCU) properly. Once all the units have connected properly the MCU waits to send data until it senses a large enough change in the data collected from the IMUs. A large enough motion of the hand, like a wave, would prompt the MCU to begin sending signals via bluetooth to the host computer where the data recognition and sign interpretations would be performed.

The data is sent in an uniform fashion, it is important that the data stays in a consistent order to ensure the machine learning will function properly and be able to learn and understand signs. The machine learning will learn to recognize patterns in all of the data and associate this data with signs. Once the program translates the data into text it will run through a text-to-speech library and generate the appropriate word or phase.

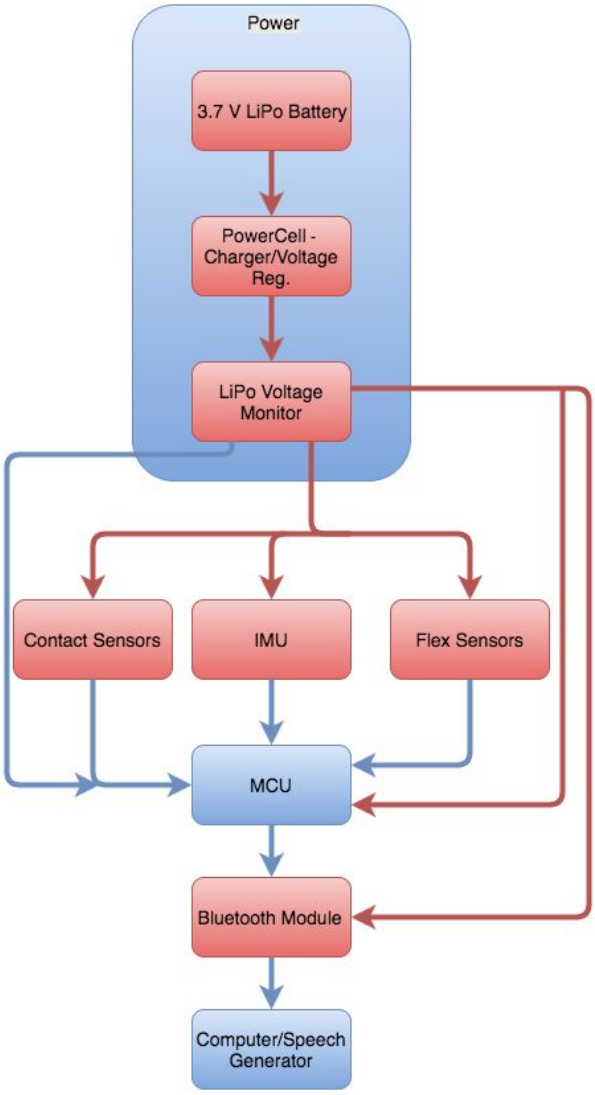

Fig. A. Complete system flowchart illustrating the flow of data and power.

IV. Hardware Detail

The ASLA sign language glove has many different components and parts. A big part of what makes the glove so unique is the component aspect and which specific hardware was used. A lot of time was put into choosing the brand, specifications, and performance output to be used in conjunction for the final design.

A. Microcontrol Unit (MCU)

Due to size and portability of the project, it was important to find a MCU that was both compact and energy efficient. Atmel's ATmega328P was one that fit both profiles. The chips typical current draw when operating at 3.3 volts is 7 mA. The chip is 7x7x1.2 mm in size and this worked for the size of the printed circuit board design. The ATmega can run at a 16 MHz frequency which is a suitable speed for the amount of data that will be pushed through. It also has on-chip dedicated UART hardware which is used for the serial communication. The specific package of the ATmega328P used in the project allowed for enough analog signals to be taken in from the sensors and converted to a digital signal by utilizing the on-chip analog-to-digital converter.

B. Flex Sensors

The flex sensor performs with resistance ranges of 10k ohms at rest to 20k ohms at max bend for the 4.5" flex sensor, and 25k ohm at rest to 125k ohms max bend for the 2.2" flex sensor. The machine itself doesn't need to know the angle it is making, it just needs to recognize change in values from the sensors. The machine learning works by seeing repetitive data points so as along as the sensors are calibrated properly, it should be able to recognize different sets of data points. Each flex sensors have two pins. One pin connects in parallel with a pullup resistor and to its own analog pin. The other pin connects to VCC for power input. The analog data received gets converted to a digital signal within the MCU using internal ADC.

C. Contact Sensors

The contact sensor used in the glove is connected to a digital pin. The glove will use 2 contact sensors which have 3 terminals. One terminal is located in the index finger which is connected to power, while the other two terminals are ground terminals that connect in parallel with a pull-down resistor to digital pins. Once the power terminal touches either of the other terminals, the closed connection drives the corresponding digital pin to higher levels, where the MCU will recognize the change. This acts like a digital switch. This data will be combined with other sensor data and be relayed forward.

D. Inertial Measurement Unit (IMU)

Two IMUs are going to be used for the glove. Each IMU contains both an accelerometer and gyroscope, which allows for 6 degrees of freedom. One IMU is located on the printed circuit board (PCB) and the other IMU will be

located on the back of the middle finger. The inclusion of these two units creates better overall orientation, as well as easier recognition of words and phrases when signing.

Generally, the IMU's require specific orientations when placed on the PCB, this is due to the fact they record X, Y and Z direction. Without being oriented properly, the data being measured by the IMUs would be incorrect in normal applications. However due to the nature of the machine learning, the initial orientation of the IMU's placement on the PCB wasn't as critical. The IMU uses the I2C line, which offers a two-wire serial interface that will allow the two IMU's to communicate to each other. They are connected through the same two lines, SDA and SC, as you can see in Fig.B. Each IMU uses a separate address, which helps differentiate between the two. The IMU's can only be run at 3.3v, not the traditional 5v.

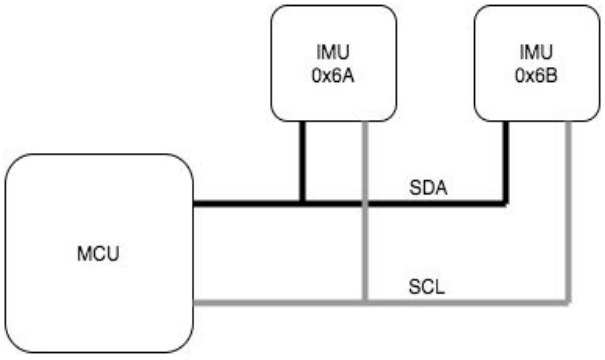

Fig. B. Illustration of the two IMU connections using the I^2C protocol; the two IMUs must have different addresses.

E. Bluetooth

The connection for the BlueSMiRF Gold Bluetooth device is simple, the grounds are connected to each other, the VCC is connected to the 3.3 volt power of the MCU. The transmitter port TX-0 of the Bluetooth is connected to one of the digital pins of the ATmega, while the receiver port RX-I of the bluetooth is connected to another digital port. It is important to highlight that the digital pins of the MCU are designed to be capable of supporting software serial UART, meaning the RX and TX connections of the bluetooth module can be connected to any of the digital pins of the MCU. This means the hardware UART of the MCU can still be used for uploading codes as it is now.

The serial data connections will be hooked up to the ATmega to transfer all the data collected from the sensors on the glove. Then the data received by the Bluetooth module from the MCU and sensors will be passed through the Bluetooth module and directed to the antenna of the module to then be sent to the computer to update the database or to just compare the data from the glove with the one on the already existing database.

F. LiPo Fuel Gauge Voltage Monitor

Maxim's MAX17043G+U was acquired once it had been decided the project was going to be portable. It too, like many of the other components, has a very small profile and works well with the spacing requirements of the circuit board. The chip has an operating voltage that works at the 3.3 volt level that is used to power the system. When powered at 3.3 volts the voltage monitor ha a typical current draw of 50μA. It has a very accurate reading with ±30mV of the actual voltage left in the battery. This chip, like the IMUs, utilizes the I^2C communication protocol to send the data to the MCU about the current voltage levels. There is also a output pin that is used to light up an LED indicator light as well.

G. Battery Charger and Voltage Regulator

The lithium ion polymer battery used for the project requires a charging circuit since we plan on keeping the unit wireless and all the components on a single board. The module used for the charge regulation of the battery is a Microchip MCP73831/2. This chip only contains one cell and can accept an voltage input ranging from 3.75 to 6 volts. It has a regulation accuracy of 0.75 % which is very important when working with the batteries. The output voltage can be set to one of 4 options, and there is also a charge status output pin that can drive LEDs to show whether the battery is charging or not. An external circuit that was created on the output of the module that when power is applied for charging it disconnects power from the rest of the unit such that the battery charges faster.

Since the battery being used can supply a voltage higher than that wanted for the operation of the unit it was necessary to use a voltage regulator. The regulator used was a Texas Instruments TPS61200, which has an operating voltage of 0.3-5.5 volts. Since the project runs at 3.3 volts the typical current draw is about 300mA. This is good since it means the voltage regulator is very efficient

when stepping down the voltage resulting in very little power loss, which in turn means longer battery life and operating time.

H. Battery

After considering different types of batteries. We decided to use a 2000mAh Lithium Ion Polymer Battery. The size of the battery (54 x 60mm) is small enough where it won't make the glove bulky. The battery was chosen based on estimated power consumption. It has an estimated 6.5 hours of runtime. Its temperature functionality is also impressive, being able to withstand temperatures of -25°C to 60°C. There will be an LED in the PCB that will turn on when the battery is fully charged, which is helpful so the battery doesn't overcharge and get damaged over time.

V. Software Detail

Overall software for the project is split up into three main objectives: Embedded System, Learning, and Translating. Embedded System is the entirety of the glove programming and its components. Learning and Translating is part of a bigger program that is hosted on a computer running Python 2.7 along with all the necessary libraries, which are in no particular order: SciPy, NumPy, scikit-learn, Serial, and PyTTSx.

A. Embedded System Software

The system which controls the embedded system portion of the project is written in C++. It will follow standard device layout of a setup section and a continuous loop section. Setup happens once, when the device is initially powered on, or a reset has been initiated. After which the software will enter an infinite loop for the duration of its life.

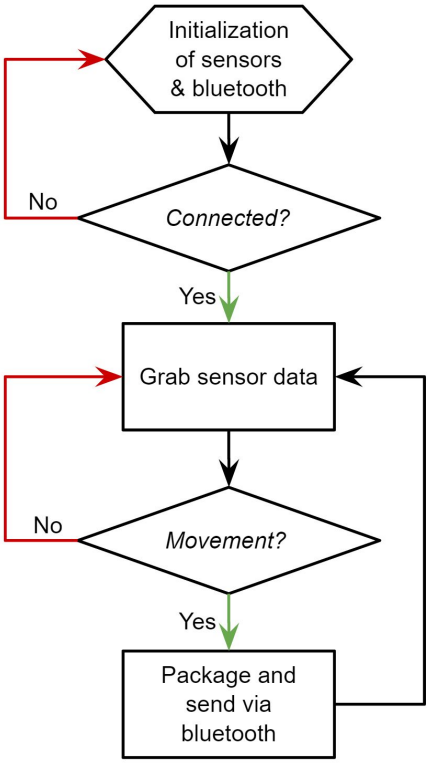

Fig. C. Embedded software flow diagram

As seen in Figure C we can see where the startup code and continuous loop code splits. We first initialize each of the IMU sensors and each flex sensor. If there is a problem connecting to the IMU sensors we then restart the initialization process along with telling the user that there has been a problem via a status LED on the PCB. Assuming there is no communication problem the embedded software then enters an endless loop of what is the main function of the system. It then communicates to two IMU sensors using the I^2C communication protocol.

We have decided to use the I^2C protocol in favor of other communication because it allows for multiple slave devices to one or many master devices. This is achieved with only two connections, a bidirectional data line and a clock line. Each slave device has a set address so only one device can communicate on the I^2C line at a time [1].

Using UART we connect to the bluetooth module, which was previously set up. UART also requires two connections but each connection is a one way data line. The communication speed must be previously determined during the development of the project. After setting up the

communication speed in the software we then move on to the main loop of the embedded software.

During the main loop we grab all of the sensor data, five flex sensors, two IMUs, and two contact sensors for a total of thirteen individual values. We then filter this data to make sure we only grab the necessary data for sign and word recognition. For filtering we are experimenting with a number of techniques. First of is detecting if any of the sensors experience a drastic change in values, this would indicate there is some form of movement from the user. Another way is extracting key movements and creating a set code sequence for each of the movements we register. We are currently using the first way discussed and depending on how experimentation goes we will revisit this choice. After filtering the data we then package it up into a comma separated value (CSV) and transmit it through bluetooth to the host computer.

B. Training Software

Before we start using the translating function of the program, first we must train the machine learning model, using support vector machines (SVM), which is a classification supervised learning algorithm. We first train the entire alphabet with the user. After of which we start creating a library for the program with common words and phrases used by the current user.

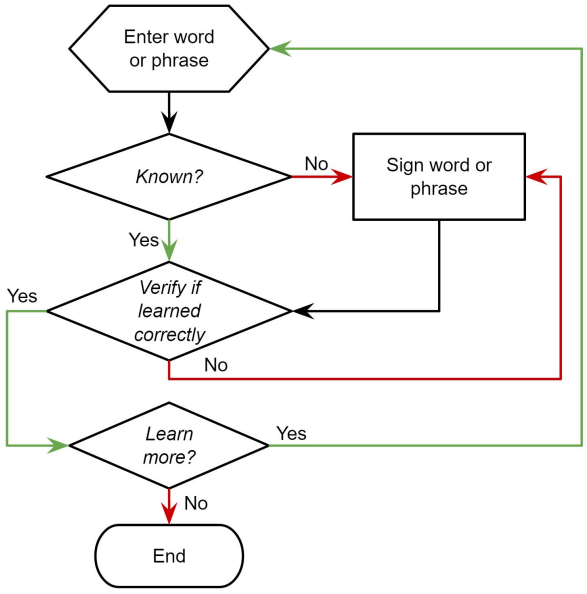

Fig. D. Training software flow diagram

As seen in Figure D we first check to see if this was a previously known word or phrase. If it was not known we then ask the user to sign the word or phrase for input into the library. If it was known or if we enter a new word or phrase for the library we then verify if it was correctly learned. After we learn the new word or phrase we ask the user if he wants to learn a new one, if not we then terminate to the main menu.

For each word or phrase a large enough data sample set needs to be acquired for SVM to properly fit all of the data. As of current testing we are asking users to input the word or phrase five times.

When the user has inputed enough words or phrases into the library the SVM model will then create a fitting to be able to predict signs during translation. SVM handles fitting of high dimensional data very well, which is needed for our project seeing how we have a time series data for thirteen sensor values.

C. Translating Software

Now we have learned enough for the user to start using it in a lecture setting, or even every day life, we now must start translating. The program enters a continuous loop that terminates after an user specified timeout or the user terminates the translating portion of the software as seen in Figure E.

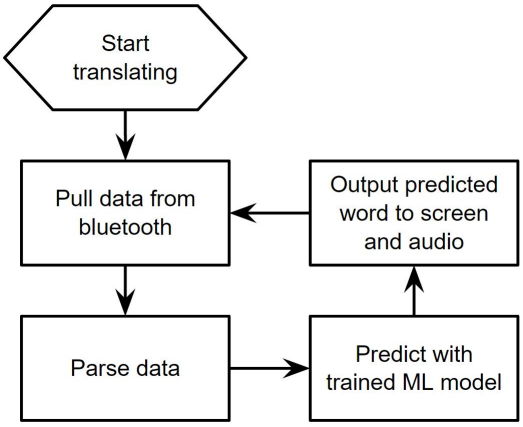

Fig. E. Translating software flow diagram

The software first pulls previously filtered data from the bluetooth connection. It then parses it from a CSV string into an array of numbers. It then uses that array along with the trained SVM model to predict what the user is signing.

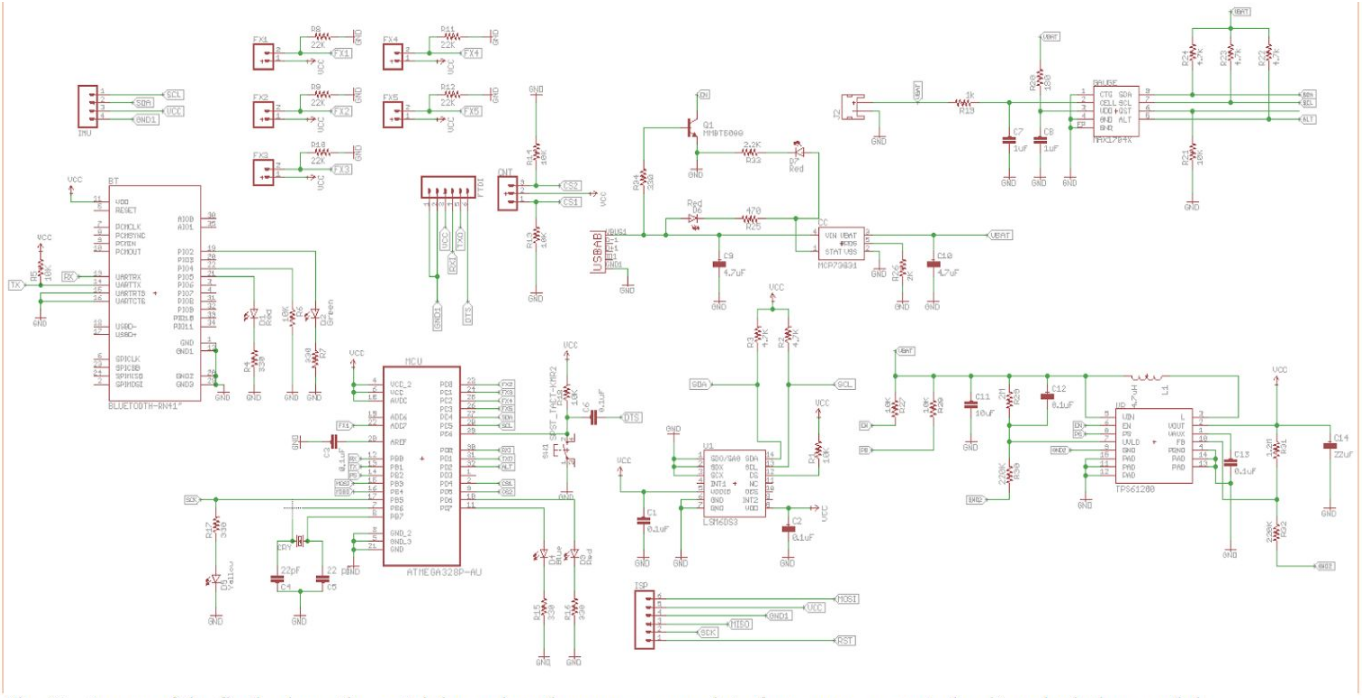

Fig. G. Image of the final schematic containing onboard sensors, sensor interfaces, power, control unit, and wireless module.

Even if it is correct or incorrect the SVM model will predict to the best of its ability and output the word on to the screen as text and the vocalization through the speakers. The cycle then repeats until the timeout with no received data from the bluetooth hits orthe user terminates the program.

VI. Board Design

A schematic was designed that included connections for power, sensors, bluetooth and MCU all in one. It was important to label nets with appropriate names so when errors arose debugging became much easier. Also keeping different functional groups of the overall system separated, as seen in Figure G above, made it much easier to see that all the components and connections were made properly.

Trying to keep the components on a single side of the board was desired, but it was determined that some of the components needed to be placed on the bottom layer in order for the design to fit properly. This gave us more room to make simpler traces. A ground plane was used on both sides of the board to ground all the components further eliminating crossing traces and cleaning up the board design. The ground pour also help reduce signal interference. Vertical traces were mostly used in the top layer of the PCB, while Horizontal traces were mostly used in the bottom layer. The PCB design can be seen down below in Fig. F. This method of laying traces was recommended to minimize routing interference.

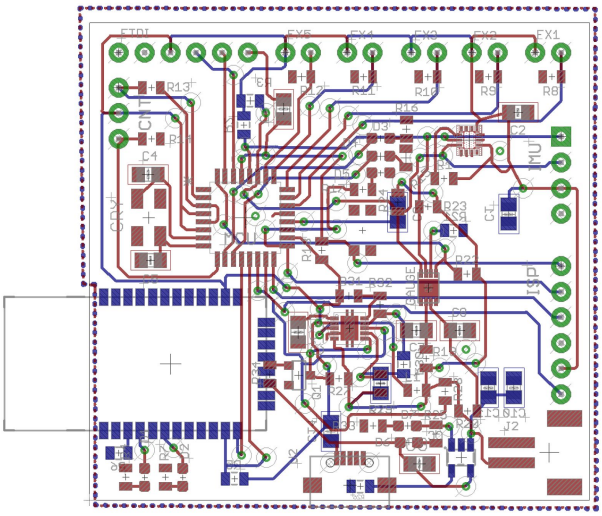

Fig. G. Both layers of the final PCB design and all the routing and interconnects of the components.

The flex sensors pins are located on the top of the board to make the shorter connections to the flex sensors. The bluetooth was placed on the bottom layer because it took up a lot of space and was not necessary to keep on the top. The antenna part of bluetooth projects out from the board because it can not have any copper pour or traces underneath it due to potential interference. The JST connector used for the battery was specifically placed in the bottom right corner to allow for easier connection of the battery. The final PCB dimensions came out to be 1.8 x 1.9 in, which is an ideal size to fit in the back of the glove.

The Engineers

Isabel Atanacio is a 22-year old graduating Electrical Engineering student. Isabel Atanacio wants to focus on systems integration.

Colin Fox is a 22-year old graduating Electrical engineering student. Colin's fields of study are applications engineering and systems integration.

Charlie Munoz is a 24-year old graduating Electrical engineering student. Charlie's fields of study are signal processing and systems engineering.

Daniel Sarmiento is a 23-year old graduating Computer Engineering student. Daniel hopes to pursue a career in embedded software development within the space industry.

Acknowledgement

References

[1] https://learn.sparkfun.com/tutorials/i2c