EC PROJECT

SIGN LANGUAGE DETECTION USING FLEX SENSORS

Section: A
Table: 3

Project Members:
Aravind Seshadri (210180)
Amay Raj (210116)
Raghav Shukla (210800)

OBJECTIVE:

To design a sign language detection module that is attached to a glove and report the corresponding sign using Bluetooth on a phone.

PROJECT SCOPE AND EXECUTION PLAN:

WHAT PROBLEM STATEMENT ARE YOU TRYING TO SOLVE, AND WHY IS IT IMPORTANT / INTERESTING?

Our project aims to address the communication barriers faced by individuals with hearing impairments by creating a sign language detection module using gloves. Sign language serves as a crucial means of communication for the deaf and hard of hearing community, allowing them to express themselves and interact with others effectively. However, there is often a lack of accessibility to sign language interpreters and resources, hindering communication in various settings such as educational institutions, workplaces, and public spaces. By developing a portable and user-friendly sign language detection module, we aim to empower individuals with hearing impairments to communicate more seamlessly and independently in their daily lives. This project is important as it promotes inclusivity, accessibility, and equal participation for individuals with disabilities, contributing to a more inclusive society.

WHAT ARE THE EXISTING SOLUTIONS? DESCRIBE A FEW OF THEM AND LIST ANY SHORTCOMINGS IN THEM. IS YOUR SOLUTION APPROACH UNIQUE IN SOME WAY?

Existing solutions to facilitate communication for individuals with hearing impairments include manual sign language interpretation by human interpreters, assistive devices such as video relay services and text-to-speech applications, and wearable technologies like smart gloves and

wristbands. While manual interpretation by human interpreters is accurate, it may not always be available or feasible due to cost and accessibility constraints. Assistive devices and wearable technologies offer promising solutions but may have limitations such as limited vocabulary recognition, complex setup procedures, and high costs.

Our solution approach with the sign language detection module using gloves is unique in its simplicity, portability, and real-time interaction capabilities. By leveraging flex sensors and Bluetooth technology, we have developed a cost-effective and user-friendly solution that allows individuals to communicate through sign language gestures detected by the glove and transmitted wirelessly to a smartphone. The module offers real-time feedback, enabling seamless communication without the need for complex setup or external devices. Additionally, our solution prioritizes user experience and accessibility, making it suitable for a wide range of applications and environments.

WHAT RESOURCES DO YOU REQUIRE TO COMPLETE THE PROJECT?

RESOURCES REQUIRED:

HARDWARE COMPONENTS:

1. Flex sensors x 5
2. Resistors x 5 (The value of resistances can range between 40k to 60k ohms ideally)
3. HC05 Bluetooth module x 1
4. Arduino Nano x 1
5. Breadboard x 1
6. Printed Circuit Board (PCB) x 1
7. Soldering kit
8. Glove
9. Single sided tape
10. Double sided tape
11. Phone with Serial Bluetooth Communication app
12. Wires

SOFTWARE COMPONENTS:

i. Arduino IDE for coding and uploading firmware.
ii. Serial Bluetooth Communication app for smartphone interface

GIVE A BREAKUP OF TASKS THAT YOU NEED TO ACCOMPLISH WEEK BY WEEK TO COMPLETE THE PROJECT.

WEEK 1: TESTING OF ARDUINO & BLUETOOTH SENSORS

- Testing the Arduino kit with a basic code and blinking LED
- Connecting the HC05 Bluetooth sensor to the circuit and testing with the serial Bluetooth communication app

WEEK 2: CALIBRATION OF FLEX SENSORS

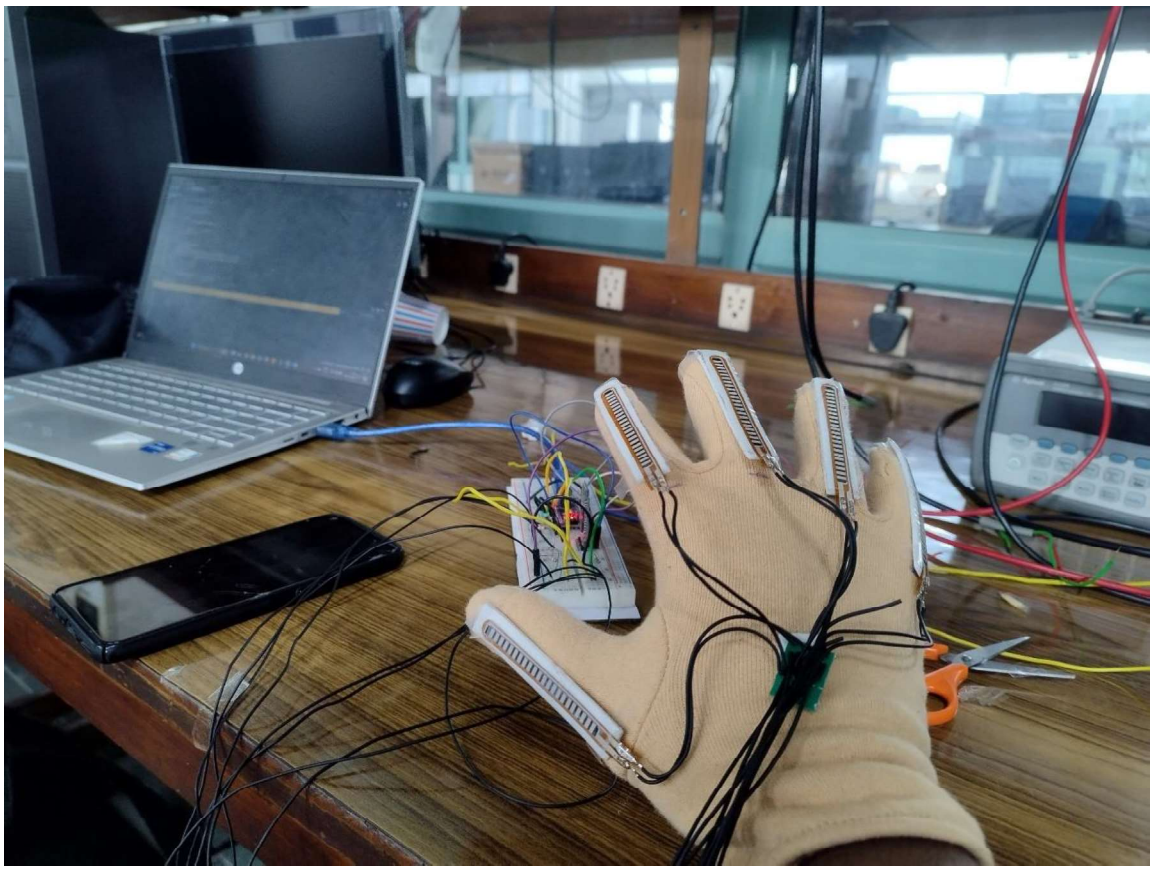
- Resistance testing of flex sensors and appropriate choice of resistance for each sensor.
- Calibration of the flex sensors and the initial code needed for the calibration.

WEEK 3: PROJECT SETUP & ROBUSTNESS

- Soldering of the flex sensors and final assembly of the project
- Modifying and finalizing the code to achieve the robustness needed
- Thresholding of the data and final testing for different edge cases

DESCRIPTION OF EACH COMPONENT:

PROJECT SETUP:



FLEX SENSOR:

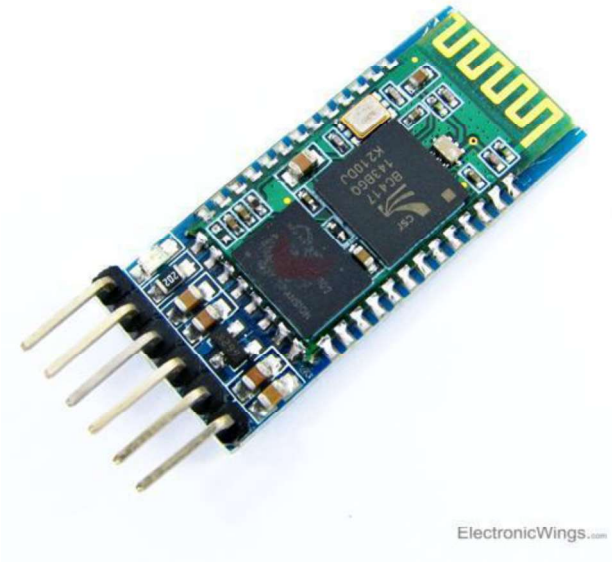
A flex sensor is used to measure the amount of bending. Its design typically uses carbon and plastic. The carbon surface is attached to a plastic strip. As the sensor is bent, the amount of resistance offered by it varies. Thus, this is very useful in applications involving the bending of any part such as in robotics, Virtual motion, Musical instruments etc. In this case, bending the fingers can give us data about the sign being used.

SPECIFICATIONS & FEATURES

- Operating voltage of this sensor ranges from 0V to 5V
- It can function on low-voltages.
- Power rating is 1 Watt for peak & 0.5Watt for continuous.
- Operating temperature ranges from -45ºC to +80ºC
- Flat resistance is 25K Ω
- The tolerance of resistance will be ±30%
- The range of bend resistance will range from45K -125K Ohms

HC05 BLUETOOTH MODULE

HC-05 is a module that is designed for wireless communication through Bluetooth. It has many applications, including wireless headsets, keyboards, and other consumer applications. Further, it can be used to send or receive data from a user when connected to an Arduino.



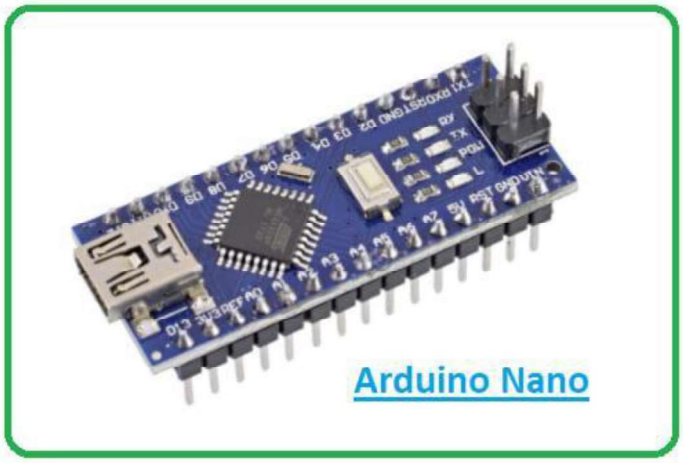
SPECIFICATION OF HC-05 BLUETOOTH MODULE

- Bluetooth version: 2.0 + EDR (Enhanced Data Rate)
- Frequency: 2.4 GHz ISM band
- Modulation: GFSK (Gaussian Frequency Shift Keying)
- Transmit power: Class 2 (up to 4 dBm)
- Sensitivity: -80 dBm typical

- Range: approximately 10 meters (or 33 feet) in open air
- Operating voltage: 3.3V to 5V DC
- Operating temperature: -20°C to 75°C (-4°F to 167°F)

ARDUINO NANO:

Arduino Nano is a small open-source electronic development board based on AVR microcontroller. It can perform similar functions to other boards but is smaller and more compact. It can be used for standalone applications in medical instruments, embedded systems, industrial automation etc.



SOME TECHNICAL SPECIFICATIONS:

- Microcontroller: Microchip ATmega328P
- Operating voltage: 5 volts
- Input voltage: 5 to 20 volts
- Digital I/O pins: 14 (6 optional PWM outputs)
- Analog input pins: 8

RESISTANCES:

Resistors of different values were used for each flex sensor. They were selected in the following manner:

1. The minimum and maximum value of the resistance of the flex sensor is measured with the help of a multimeter.
2. The equation to measure the voltage of the voltage divider is:

$$V_{voltage-divider} = \frac{V_{resistor}}{V_{flex} + V_{resistor}} \times (5V)$$

The resistance of the flex sensor increases when it is bent. Hence, the value of the voltage of the voltage divider decreases when it is flexed.

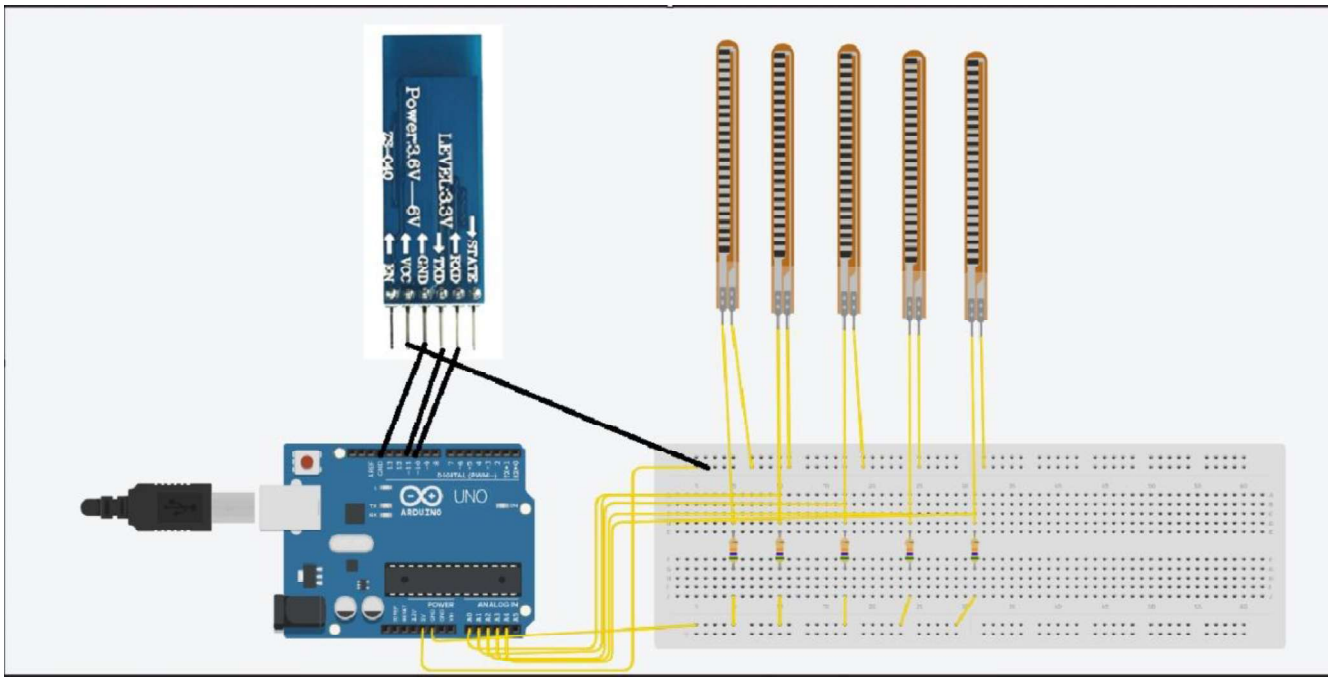
3. Maximum and minimum value of the voltage divider:
   a. $V_{voltage-divider,max} = \frac{V_{resistor}}{V_{flex,min}+V_{resistor}} \times (5V)$

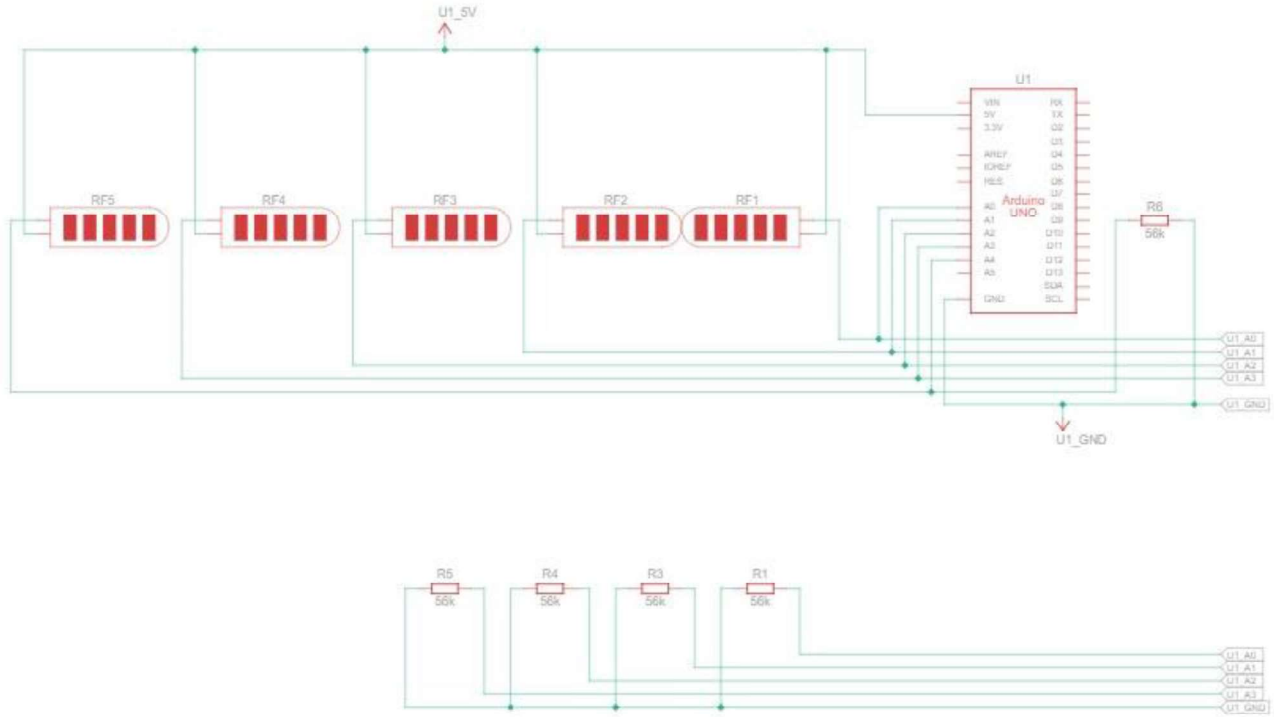
   b. $V_{voltage-divider,min} = \frac{V_{resistor}}{V_{flex,max}+V_{resistor}} \times (5V)$

4. Sensitivity of device is $V_{voltage\text{-}divider,max}$ - $V_{voltage\text{-}divider,min}$. We used the DESMOS graph plotter to plot sensitivity versus $V_{resistor}$ and found the value of $V_{resistor}$ sensitivity is maximum. This resistor value was used.

This process was repeated for all 5 flex sensors.

CIRCUIT DIAGRAM:





(Schematic without bluetooth)

CIRCUIT ANALYSIS:

We have 5 flex sensors attached to each finger on a glove. 1 terminal from each sensor is soldered together on a PCB to act as the common 5V supply. The PCB takes 10 wires as the input and 6 wires as the output. The common terminal is connected to the 5V supply on the breadboard. The other individual terminals are each connected to one end of a resistor whose other end is grounded.

The point of connection between the resistor and the sensor is used as the data. We read the data from each sensor through analog pins A0 to A4 of the Arduino.

Further, a Bluetooth module HC05 is connected to the Arduino. Its transmitting and receiving pins are connected to the Arduino's digital pins 10 and 11.

A Bluetooth serial communication app connects the phone to the Arduino.

CODE:

```
#include <SoftwareSerial.h>// import the serial library


SoftwareSerial bt(11, 10); // RX, TX


int hand[]={0,0,0,0,0};
float low[]={0,0,0,0,0};
float high[]={0,0,0,0,0};
float min[]={1023,1023,1023,1023,1023};
float max[]={0,0,0,0,0};
float val[]={0,0,0,0,0};
long NUMITERATIONS=1000;
long AVG=0;



// Define a hashmap-like structure to map arrays to strings
struct ArrayToStringMap {
  int key[5];
  String value;
};


ArrayToStringMap hi[]={
 {{0,1,0,0,0},"One"},
 {{0,1,1,0,0},"Two"},
 {{0,1,1,1,0},"Three"},
 {{0,1,1,1,1},"Four"},
 {{1,1,1,1,1},"Five"},
 {{1,0,0,0,0},"Thumbs-Up"},
 {{1,1,1,0,0},"Gun"},
 {{1,1,0,0,1},"Spiderman"},
 {{0,0,1,1,1},"OK"},
 {{1,0,0,0,1},"Call"},
 {{0,0,0,1,1},"Y"},
 {{0,0,0,0,1},"I"}
};


bool arraysAreEqual(int arr1[], int arr2[], int size) {
  for (int i = 0; i < size; i++) {
    if (arr1[i] != arr2[i]) {
      return false; // Arrays are not equal
```

```
    }
  }
  return true; // Arrays are equal
}


String getValue(int arr[5]) {
  for (int i = 0; i < 12; i++) {

    if (arraysAreEqual(arr,hi[i].key,5))
    {
      return hi[i].value;
    }
  }
  // Key not found, return an empty string
  return "None";
}


unsigned long interval=15000;
void setup() {
 // put your setup code here, to run once:
 bt.begin(9600);
 bt.println("Bluetooth On please wait...");
}


void loop()
{
 // put your main code here, to run repeatedly:
 unsigned long currentMillis = millis();
 for(int pin=0;pin<5;pin++)
 val[pin]=analogRead(pin);

 if(currentMillis<5000)
 {
  bt.println("Relax");
  for(int pin=0;pin<5;pin++)
  {
   AVG=0;
   for(int i=0;i<NUMITERATIONS;i++)
   {
    val[pin]=analogRead(pin);
    AVG=AVG+val[pin];
   }

   AVG/=NUMITERATIONS;
   max[pin]=AVG;
  }
```

```
 }
 else if(currentMillis<10000)
 {
  bt.println("Bend");
  for(int pin=0;pin<5;pin++)
  {
    AVG=0;
    for(int i=0;i<NUMITERATIONS;i++)
    {
      val[pin]=analogRead(pin);
      AVG=AVG+val[pin];
    }

    AVG/=NUMITERATIONS;
    min[pin]=AVG;
    low[pin]=((max[pin]+min[pin])/2)-((max[pin]-min[pin])/8);
    high[pin]=((max[pin]+min[pin])/2)+((max[pin]-min[pin])/8);
  }
 }
 else
 {
  for(int pin=0;pin<5;pin++)
  {
    AVG=0;
    for(int i=0;i<NUMITERATIONS;i++)
    {
      val[pin]=analogRead(pin);
      AVG=AVG+val[pin];
    }

    AVG/=NUMITERATIONS;
    if(AVG<low[pin])
    hand[pin]=0;

    if(AVG>high[pin])
    hand[pin]=1;


    // bt.print(hand[pin]);
    // bt.print(" ");
  }
  bt.println(getValue(hand));
  // bt.println("");
  // delay(100);
 }
}
```

CODE EXPLANATION

- The code begins by including necessary libraries, defining global variables and arrays to store sensor readings and hand gestures.
- A hashmap-like structure is defined to map sensor readings to corresponding hand gestures.
- Helper functions:

```
bool arraysAreEqual(int arr1[], int arr2[], int size)
String getValue(int arr[5])
```

  These are defined to check array equality and retrieve values from the hashmap.

- In the setup function, Bluetooth communication is initialized and a message is sent to the serial monitor.
- The loop function contains two phases: calibration and detection.
- During the calibration phase, the maximum and minimum sensor values are recorded. The resistance of the flex sensor increases when it is bent. Hence, the value of the voltage of the voltage divider as measured by the Arduino decreases when we flex our fingers.
  - Initially, the wielder of the glove must keep his/her fingers relaxed. During this time, the code outputs 'Relax' in the serial monitor to indicate this. A for loop is used to measure the average value of the voltage value(using AnalogRead) of the voltage divider for 5000 milliseconds(ms)=5 seconds(s). This is done for all 5 pins. This will yield the average maximum value of the voltage. Therefore, max[pin] stores the maximum value of the voltage measured for the corresponding pin. We use average instead of absolute maximum to avoid jittery values during measurement that might be unnecessarily high and make detection difficult.
  - Subsequently, the wielder of the glove must flex his/her fingers . During this time, the code outputs 'Bend' in the serial monitor to indicate this. Again, a for loop is used to measure the average value of the voltage value(using AnalogRead) of the voltage divider for 5000 milliseconds(ms)=5 seconds(s). This is done for all 5 pins. This will yield the average minimum value of the voltage. Therefore, min[pin] stores the maximum value of the voltage measured for the corresponding pin. We use average instead of absolute minimum to avoid jittery values during measurement that might be unnecessarily low and make detection difficult.

- During the detection phase, sensor values are compared to threshold values to classify hand gestures, and the detected gestures are sent via Bluetooth. The threshold values are defined as:
  - $Upper\ Threshold\ high[pin] = (\frac{\max[pin]+\min[pin]}{2}) + (\frac{\max[pin]-\min[pin]}{8})$

If the value by Arduino through AnalogRead measured is greater than this value, then the value assigned to the array 'hand' is 1.

- $Lower\ Threshold\ low[pin] = \left(\frac{\max[pin]+\min[pin]}{2}\right) - \left(\frac{\max[pin]-\min[pin]}{8}\right)$

  If the value by Arduino through AnalogRead measured is lesser than this value, then the value assigned to the array 'hand' is 0.

The range between these thresholds is defined as the forbidden range. After detection of these values, the getValue function outputs the value of the corresponding hand sign.

ROBUSTNESS OF THE CODE

The designed code is designed in such a way that it can accommodate changes in resistance values of the resistors and the flex sensors during each calibration, without the need of changing anything in the code. During each calibration phase, the code defines new thresholds and hence a new forbidden range as well.

TESTING AND OBSERVATIONS:

- Once the code is loaded, in the first 5 seconds, the glove is kept in a relaxed state
- For the next 5 seconds, the glove's fingers are to be bent.
- Once the calibration is over, each sign made by the hand reflects on the phone screen with a max delay of 2 seconds.

CONCLUSIONS:

In conclusion, the project of making a sign with gloves and sends the corresponding signal to our phone was achieved using

- Arduino nano
- Flex sensors
- HC-05 module

Further advancements may include adding a machine-learning model that reads a couple of required signs and outputs a proper sentence. Additionally, the speed can also be improved.