Sudan University of Science and Technology

College of Engineering

School of Electronics Engineering
A Research Submitted In Partial Fulfilment for The Requirement of B.Sc. (Honors)in Electronic Engineering

Sign Language Glove (Arabic Sign Language)

Written by:

1. **Amin Galal Elhag Mohamed.**
2. **Hamid Mohamed Hamid Mohamed.**
3. **Mohamed Osama Omer Abuzaid.**
4. **Tahleel Mahmoud Idrees Abd-Alhafeez.**

Supervised by:

PhD. Mohammed Elnoor Abdallah

March 2022

الأية

قال تعالى: (يَٰٓأَيُّهَا ٱلنَّاسُ إِنَّا خَلَقۡنَٰكُم مِّن ذَكَرٖ وَأُنثَىٰ وَجَعَلۡنَٰكُمۡ شُعُوبٗا وَقَبَآئِلَ لِتَعَارَفُوٓاْۚ إِنَّ أَكۡرَمَكُمۡ عِندَ ٱللَّهِ أَتۡقَىٰكُمۡۚ إِنَّ ٱللَّهَ عَلِيمٌ خَبِيرٞ)

Abstract

Communication between people is one of the important things that the society is built on and we cannot live without it. It connects people together. We are trying to tear down the communication barrier between Arabic spoken/written language users and signed Arabic language users. The communication problem here is that people don’t understand sign language. And the communication process cannot be completed without all the involved parties being able to understand each other. Our Sign Language Glove utilizes the power of software aided hardware to translate Arabic sign language widely used in Sudan into written letters The hardware uses the sensory data fed from Flex sensors and an accelerometer to interpolate the corresponding letter of the sign just done. Then the output is chosen from our data base of the Arabic sign language letters. This data base was collected and formulated on the bases of a code.

التجريد

يعتبر التواصل بين البشر إحدى أهم ألاشياء التي لا يمكننا العيش بدونها. إذ أن التواصل يجمع الناس سوية. نحن نحاول أن نسهل هذا التواصل عبر إيجاد حل لمشكلة تواصل مستخدمي لغة الإشارة العربية مع مستخدمي اللغة العربية المقولة\المقروءة، لعدم إستطاعة الأخير فهم لغة الأول. ولا يمكن لعملية التواصل أن تتم بدون فهم أي طرف للطرف الأخر. يتيح قُفَازنا إستخدام الأجهزة بمساعدة البرامج لترجمة لغة الإشارة العربية المستخدمة في السودان إلى لغة عربية مكتوبة. إذ يستخدم الجهاز البيانات الأتية من الحساسات لإعطاء الحرف المقابل للإشارة. يتم إختيار الأخير من قاعدة البيانات المعدة بما يناسب حسب ما تم من مقابلات مع بعض الأشخاص الصُم.

Acknowledgment

To the ones who stood with us during the creation of this work, you did support us the most, offered your help to the maximum extent, even in those of times which are the hardest for you. You were Team A for us.

We wish to thank:

1. Ayman Al-Nour Haj.
2. Mohamed Salah Al-Deen Mohamed.
3. Yassin Ahmed Hassan.
4. Ali Ibraheam.

Dedecation

To father, mother and friends.

Contents

List of figures:

Chapter One : Introduction

1.1 Preface

A person who have hearing disability is one who cannot hear ranges between 20 dB to 120-130 dB or better in both ears. They often use sign language for communication. According to the World Health Organization globally, 1.5 billion people live with some degrees of hearing loss out of which around 430 million people require rehabilitation services for their hearing loss. By 2050 nearly 2.5 billion people are projected to have some degree of hearing loss and at least 700 will require hearing rehabilitation. Over 1 billion young adults are at risk of permanent, avoidable hearing loss due to unsafe listening practices. Unaddressed hearing loss is expensive to communities worldwide and costs governments US$ 980 billion annually. Interventions to prevent, identify and address hearing loss are cost-effective and can bring great benefit to individuals. Of those who could benefit with the use of a hearing aid, only 17% actually use one. The gap is consistently high in all parts of the world, ranging from 77% to 83% across WHO regions, and from 74% to 90% across income levels. An annual additional investment of less than US$ 1.40 per person is needed to scale up ear and hearing care services globally. In over a 10-year period, this promises a return of nearly US$ 16 for every US dollar invested.

A Sign Language glove is an electronic device which attempts to convert the motions of a Sign Language into written or spoken words.

The potential of such gloves to do this is commonly overstated or totally misunderstood, because sign languages have a complex grammar that includes use of the sign space and facial expressions (non-manual elements). The wearable device contains sensors that run along the five fingers to identify each word, phrase or letter as it is made in Sign Language. Those signals are then sent wirelessly to a smartphone, which translates them into spoken words at a rate of one word per second. Scientists at UCLA, where one the many projects was developed, believe the innovation could allow for easier communication for deaf people. "Our hope is that this opens up an easy way for people who use sign language to communicate directly with non-signers without needing someone else to translate for them" said lead researcher Jun Chen. The researchers also added adhesive sensors to the faces of people used to test the device -between their eyebrows and on one side of their mouths- to capture facial expressions that are a part of Sign Language.

1.2 Problem statement

1. What are the obstacles affecting the communication between the sign languages users and the users of spoken/heard languages?
    - A. What is the effect of the proposed solution on the communication between the sign language users and the users of spoken/heard language?
    - B. Is there a translation problem affecting the proposed solution?
    - C. How do the different components of the project as a proposed solution communicate with each other to make the communication process successful?

1.3 Proposed Solution

1. Creating a device to translate Arabic sign language into written Arabic sign language.
   A. Narrow down the number need in Arabic sign language translators while increasing numbers of learners of this sign language.
   B. There is some dissimilarity noted between the signs in among the same language standards of the same sign language.
   C. This project consists of three key components, the sensors, the microprocessor and the presentation addition. The way those components communicate with each other is through the microprocessor as a medium between the other two. The microprocessor receives signals given off to him by the sensors and in accordance to his program orders the display unit to show characters.

1.4 Methodology

By the use of the experimental approach, we synthesized a project that sense displacement in fingers' position and correlate it to the available data base of the sign language available. This is done by means of software aided hardware.

1.5 Thesis Layout

Chapter one discuss on the background of the research done, objectives, problem statement, methodology and the thesis outlines.

Chapter two focuses on literature reviews on the project. Based on journals and other references.

Chapter three mainly discuss on the methodology utilized in sign language glove.

Chapter four mainly discuss the scenarios faced leading to the using the better approach on them.

Chapter five concludes the overall about the research done and the recommendations to the sign language gloves.

Chapter Two : Literature Review

2.1 Overview

A Sign Language glove is an electronic device which attempts to convert the motions of a Sign Language into written or spoken words. The potential of such gloves to do this is commonly overstated or totally misunderstood, because sign languages have a complex grammar that includes use of the sign space and facial expressions (non-manual elements).

The wearable device contains sensors that run along the five fingers to identify each word, phrase or letter as it is made in Sign Language. Those signals are then sent wirelessly to a smartphone, which translates them into spoken words at a rate of one word per second. [1]

2.1.1 Arabic sign Languages (ArLS)

Arabic sign languages (ArLS) are still in their developmental stages. Only in recent years has there been an awareness of the existence of communities consisting of individuals with disabilities; the Deaf are not an exception. Arab Deaf communities are almost closed ones. Interaction between a Deaf community and a hearing one is minimal and is basically concentrated around families with deaf members, relatives of the deaf, and sometimes play friends and professionals. As in other communities, communication with a deaf person is polarized within such circles. This situation has led to the emergence of many local means of sign communication. Until recently, such signs have not been gathered or codified. Signs are starting to spread, forming acknowledged sign languages.

By and large, the view held Vis-a`-Vis disability, including hearing, in the Arab society is still one of accommodation rather than assimilate [2].

2.1.2 Microsoft's Visual Studio Code

Visual Studio Code is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

first announced on April 29, 2015, by Microsoft at the 2015 Build conference. A preview build was released shortly thereafter.

It can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python and C++.It is based on the Electron framework, which is used to develop Node.js Web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services). [3].

2.1.3 Flex sensor

A Flex sensor, also known as Bend Sensor, measures the amount of deflection caused by bending the sensor. There are three kinds of Flex sensors. Initially optical Flex sensors were created and later conductive ink-based Flex sensors and capacitive Flex sensors were developed as alternates to prior, by different people. Although used for sensing "deflection", each of the type of Flex sensor is different in both construction and working principle. As of an example, Optical Flex sensor consists of a flexible tube having two ends, a reflective interior wall within the flexible tube and a light source placed within one end and a photosensitive detector placed within the other end of the flexible tube to detect a combination of direct light rays when the flexible tube is bend. [4]

2.1.4 ITG/MPU6050

The MPU-6050 is the world's first and only 6-axis motion tracking devices designed for the low power, low cost, and high performance requirements of smartphones, tablets and wearable sensors.

Having a Digital Motion Processor (DMP), a 16-bit analogue to digital converter hardware, a I2C module for interfacing with Arduino. [5]

2.1.5 Fixed Resistors

Fixed resistors are the most frequently used resistors in the electronic circuits. These resistors have the fixed resistance value. Hence, it is not possible to vary the resistance of the fixed resistor.

Resistance is defined as the process of restricting the flow of electric current to a certain level but not varying or controlling the flow of electric current.

Fixed resistor other than not changing its resistances, they are persistence to the change in voltage or temperature. And are available in various shapes and sizes. [6]

2.1.6 LM358

The LM358 is an operational amplifier commonly used with the flex sensor and is combined with it in various schemes to get multiple usage like an Adjustable Buffer or a variable Deflection Threshold Switch.

2.2 Related work

[7] examines the possibility of recognizing sign language gestures using sensor gloves. Previously sensor gloves are used in games or in applications with custom gestures. The paper explores their use in Sign Language recognition. This is done by implementing a project called "Talking Hands", and studying the results. The project uses a sensor glove to capture the signs of Arabic sign languages performed by a user and translates them into sentences of English language. Artificial neural networks are used to recognize the sensor values coming from the sensor glove. These values are then categorized in 24 alphabets of English language and two punctuation symbols introduced by the author. So, mute people can write complete sentences using this application.

The system is aimed at maximum recognition of gesture without any training. This makes the system usable at public places where there is no room for long training sessions. The speed of gesture capturing and recognition can be adjusted in the application to incorporate both the slow and fast performers of ARLS. Since a glove can only capture the shape of the hand and not the shape or motion of other parts of the body, e.g. arms, elbows, face, etc. so only postures are taken in this project. Two custom signs have been added to the input set. One is for space between words and the other is for full stop. These are not part of the sign language, but have been added to facilitate in writing the English equivalent of the sentence being performed.

The accuracy rate of the software was found to be 88%. This figure is lower due to the fact that training was done on the samples of people who did not know sign language and were given a handout to perform the signs by reading from it. So, there was great deal of variation in the samples. Some samples even gave completely wrong readings of the sensors

The problems they faced in the project was that some of the alphabets involved dynamic gestures. These may not be recognized using this glove. So these were left out from the domain of the project. Also, some gestures require use of both hands. This requires two sensor gloves.

In [8], They developed a wearable wireless gesture decoder module that can translate the fundamental set of the sign language gestures into corresponding alphabets and words and it utilizes a glove that houses a series of Flex sensors on the metacarpal and inter-phalangeal joints of the fingers to detect the bending of fingers through piezo resistive effect. The project has the following advantages: Portability, compactness, cost-effectiveness and a user-friendly android application gives this system an edge. From the research they done on international sign languages they learnt that there is no universal sign language in the world Language. The database of alphabets and words, embedded with various sensors can be used using only the formal part of the sign language consisting of 15 words and 26 letters. Recognition happens in 3 phases: recognizing sensors' values, processing and displaying/listening the sign. Sensors where one Flex sensor for thumb and pinkie, and a pair for rest of the fingers of the hand. The primary input to Arduino Leonard is the values of Flex sensors and accelerometer.

Chapter Three : System Design and operation

3.1 Utility

The system proposed here aims to know the shape of the hand through the sensors and send the signal to the controller that processes it and runs it through a database of the data set created for matches, then the program transmits it to a display screen for display.

3.2 How it works

An accelerometer is used to measure the tilt in the palm. Five flex-sensors are sued onto the surface of the glove, One for each finger. These sensors measure the bend in the fingers and according to the bend angle value the Arduino Uno microcontroller understands which set of value represent which symbol and transfer the appropriate outcome value to the Visual Studio Code's displays.

3.3 Components used

3.3.1 Arduino Uno

We selected the Arduino Uno is because gives a processing speed of 16 MHz so it is optimum to process the Flex sensor and the accelerometer input in real-time. Also, the Arduino has a flash memory of 32K bytes in capacity and this space is considered more than enough.

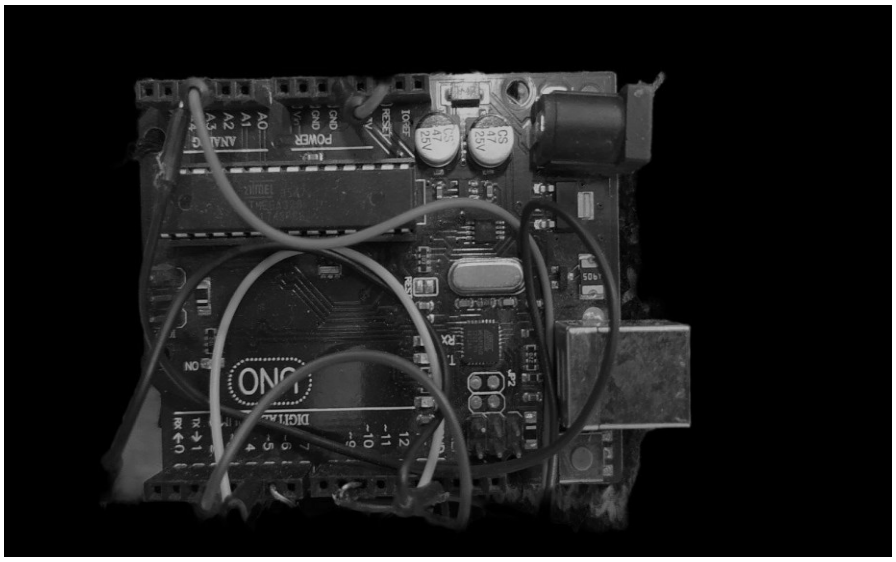


Figure 3-1: Arduino Board

3.3.2 The Flex sensor

The Flex sensor is to be considered as one of the main components in the project, it is the main source of the fingers movement sensing and without its presence, the sign language will not be translated at all. The type of movement it sense is the bendiness of the fingers. Hence, it gives a different value to the Arduino for any bending angle of the fingers.

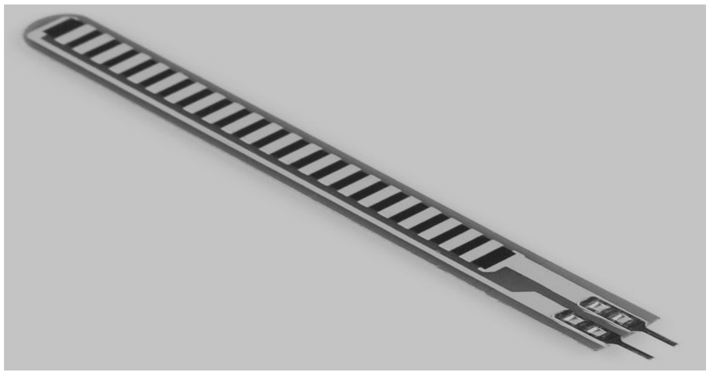


Figure 3-2: Flex sensor

3.3.3 The Accelerometer chip

ITG/MPU6050, commonly known as The Accelerometer chip is used for motion sensing of the hand movements in the dimensional space defined by the X-axis and Y-axis. This information is highly useful as some alphabets in the Arabic sign language are typically the same but differ as there is some movement in the hand to distinguish one alphabet from another similar alphabet that does not contain a hand movement.

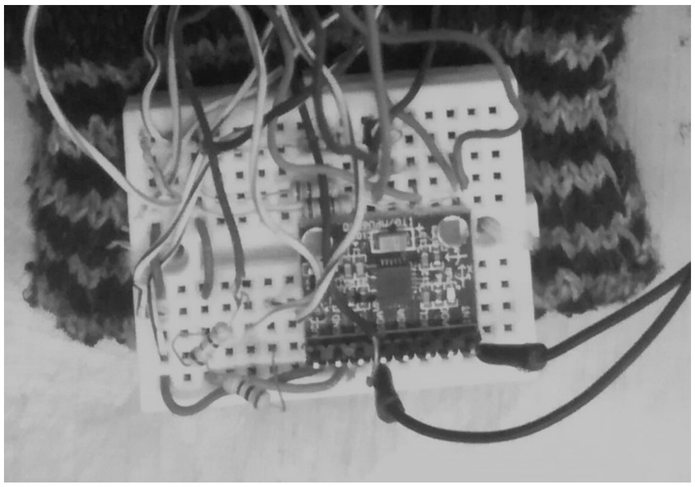


Figure 3-3: Accelerometer

3.3.4 Resistors

Resistors in the scheme are used to protect the Flex sensor against over the limit current flows, they act as a current throttling device and allow a certain percentage of the incoming current to flow into the Flex sensor.

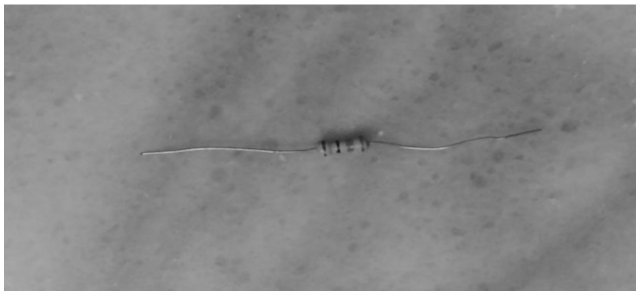


Figure 3-4: Resistor

3.3.5 Wires and jumpers

Jumpers and Wires is what connects all the components together to form the project scheme and allow them to share current and voltage given by the main power source.

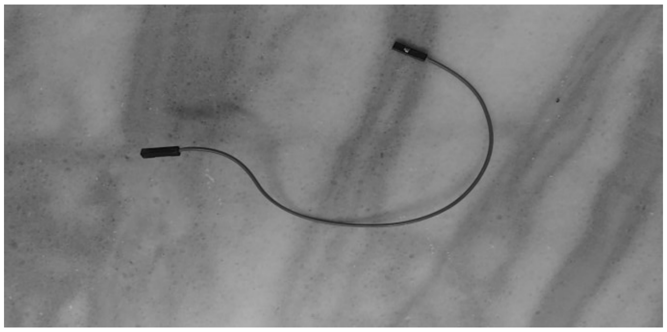


Figure 3-5: Wires and jumbers

3.4 Composition

The circuit started off with having the bread board prepared as it was dusty from it is long life on the shelf of the storage, then laid down to dry up as it was washed with water and soap for it to be clean and impurities free.

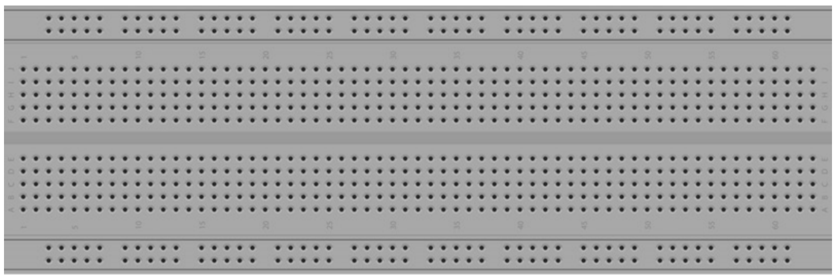


Figure 3-6: Bread board

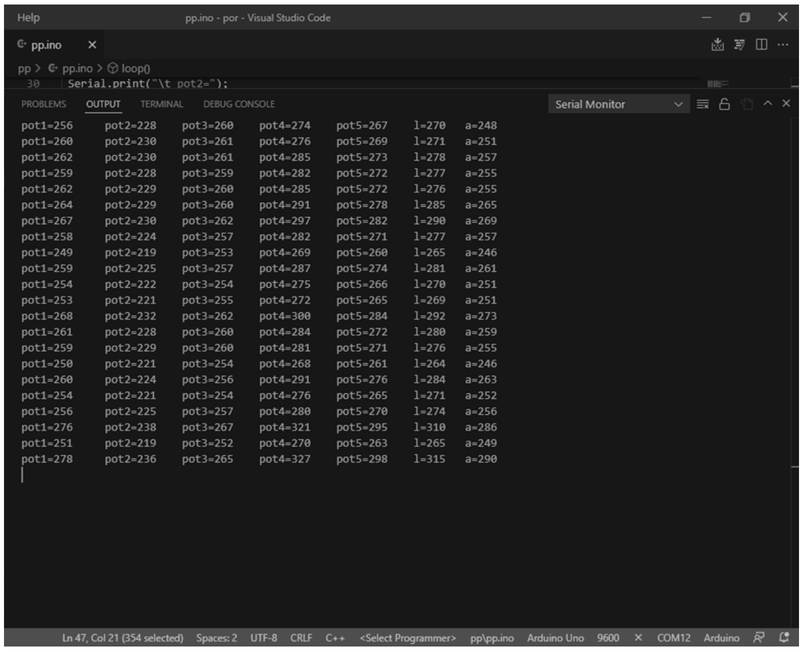
Collecting the components required for the hardware part of the project is the first step. The project's circuitry consists of resistors, Flex sensors, wires, an Accelerometer, an Arduino Uno board and a Bread Board. This combination of hardware however was selected from a variety of schemes and circuitries that were all read about in researches done by other, some of which are found in Related Work section. One of the main reasons to select them is to reduce the complexity level of the hardware while maintaining a sustainable level of accuracy.

The next part was to check the market for our chosen set of components and mark the stores' locations of the where the components are to be bought from. As we searched, the standard was to know where they are and at what price, the lesser in price the better. Fortunately, the majority of circuitry parts were found as expected. However, there was two components that we were forced to buy from out of Sudan, so there was a short delay as we waited for them to arrive.

First, the critical components had to be tested out to make sure they are working properly, the first to be put into test was the Arduino Uno, after researching, we found that the way to make sure it was good and running was to lookout for any defect by an eye examining for physical damage. After making sure no physical damage was previously inflected to the Arduino, we test the software part of the Arduino, this is done through a procedure of the steps by wiring the VCC or 5V pin to the GND pin and the TX pin to Rx pin, and then upload an empty program to the Arduino followed by opening the serial monitor of the Visual Studio Code application, as the webpage indicate, if and when writing something in the serial monitor, it will be copied instantly to the massage output in it, then your Arduino is functioning properly, else you should change the Arduino by mean of returning it to it is store.

Next to check was the Flex sensor, this was done easily as we used the Multimeter to measure the resistance given by it at it is near bending point and ran the values we measured against the data sheet of the component.

Next to check was the Accelerometer, we first connected the chip to the Arduino and then placed the chip on a surface that is flat and not sloped at any angle. The next step is to upload the Arduino with the appropriate code, this is done by navigating to File> Examples> Adafruit> lib> sensor test in the interface of the Arduino IDE, copy and paste the code onto the Visual Studio Code's developing environment and then upload the code to the Arduino board. Next we observe the output given by the sensor throughout the serial monitor. As the sensor is placed on a flat and not angled surface, the readings should be zeros or approximately zeros.


Help
pp.ino - por - Visual Studio Code
pp.ino
pp > pp.ino > loop()
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
Serial Monitor
Ln 47, Col 21 (354 selected)   Spaces: 2   UTF-8   CRLF   C++   <Select Programmer>   pp\pp.ino   Arduino Uno   9600   COM12   Arduino

```
Serial.print("\t pot2=");
```

```
pot1=256     pot2=228     pot3=260     pot4=274     pot5=267     l=270    a=248
pot1=260     pot2=230     pot3=261     pot4=276     pot5=269     l=271    a=251
pot1=262     pot2=230     pot3=261     pot4=285     pot5=273     l=278    a=257
pot1=259     pot2=228     pot3=259     pot4=282     pot5=272     l=277    a=255
pot1=262     pot2=229     pot3=260     pot4=285     pot5=272     l=276    a=255
pot1=264     pot2=229     pot3=260     pot4=291     pot5=278     l=285    a=265
pot1=267     pot2=230     pot3=262     pot4=297     pot5=282     l=290    a=269
pot1=258     pot2=224     pot3=257     pot4=282     pot5=271     l=277    a=257
pot1=249     pot2=219     pot3=253     pot4=269     pot5=260     l=265    a=246
pot1=259     pot2=225     pot3=257     pot4=287     pot5=274     l=281    a=261
pot1=254     pot2=222     pot3=254     pot4=275     pot5=266     l=270    a=251
pot1=253     pot2=221     pot3=255     pot4=272     pot5=265     l=269    a=251
pot1=268     pot2=232     pot3=262     pot4=300     pot5=284     l=292    a=273
pot1=261     pot2=228     pot3=260     pot4=284     pot5=272     l=280    a=259
pot1=259     pot2=229     pot3=260     pot4=281     pot5=271     l=276    a=255
pot1=250     pot2=221     pot3=254     pot4=268     pot5=261     l=264    a=246
pot1=260     pot2=224     pot3=256     pot4=291     pot5=276     l=284    a=263
pot1=254     pot2=221     pot3=254     pot4=276     pot5=265     l=271    a=252
pot1=256     pot2=225     pot3=257     pot4=280     pot5=270     l=274    a=256
pot1=276     pot2=238     pot3=267     pot4=321     pot5=295     l=310    a=286
pot1=251     pot2=219     pot3=252     pot4=270     pot5=263     l=265    a=249
pot1=278     pot2=236     pot3=265     pot4=327     pot5=298     l=315    a=290
```

Figure 3-7: Serial Monitor of Visual Studio Code

With the most important parts tested out, we then test the remaining of our components, which is easy as it is all done using the different measures covered by a Multimeter.

Next to do after gathering the components and testing it to make sure that they are in a great condition, is to build the circuitry of our system.

For our first step, the Flex sensor is connected to the bread board. This is done by connecting the negative pin of the Flex sensor to a resistor and then to the ground (GND pin of the Arduino). And to wire the positive pin of the Flex sensor to either a power source of 5 volts or the 5V pin of the Arduino.

And after which a reading phase was did to validate it is values are read throughout the Flex-Arduino interface, a program is loaded to the Arduino. As the photo below indicate, the input is fed to the Arduino to be read by the program by wiring one digital pin to a point between the negative pin of the Flex sensor and the resistance.

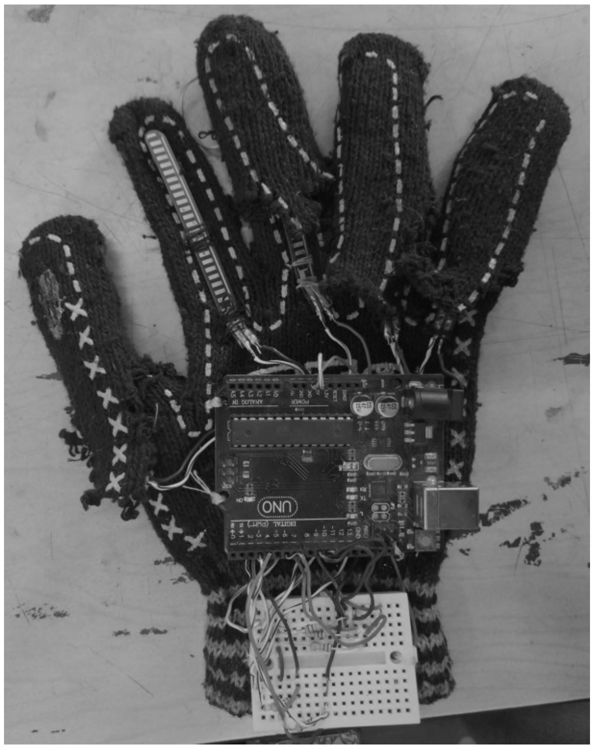


Figure 3-8: Flex sensor connection with Arduino Board

On the next step we connected the accelerometer chip to our Arduino. It was first done to make sure the chip itself is good and running and then

Connecting it to composite the different part of the system. Nevertheless, both incidents have the same connection wiring to the Arduino.

We then checked to see the wiring scheme on the internet, And the result is as follows: wire SCL of the chip to Arduino's A5 pin, SDA from the chip to A4 of the Arduino, VCC of the chip to 5V of the Arduino and lastly the GND of the chip to the GND of the Arduino. Take hint that this is only two axis connection and it was that we only needed two axes. If, however, three are needed you simply connect SDO to A3. [5]

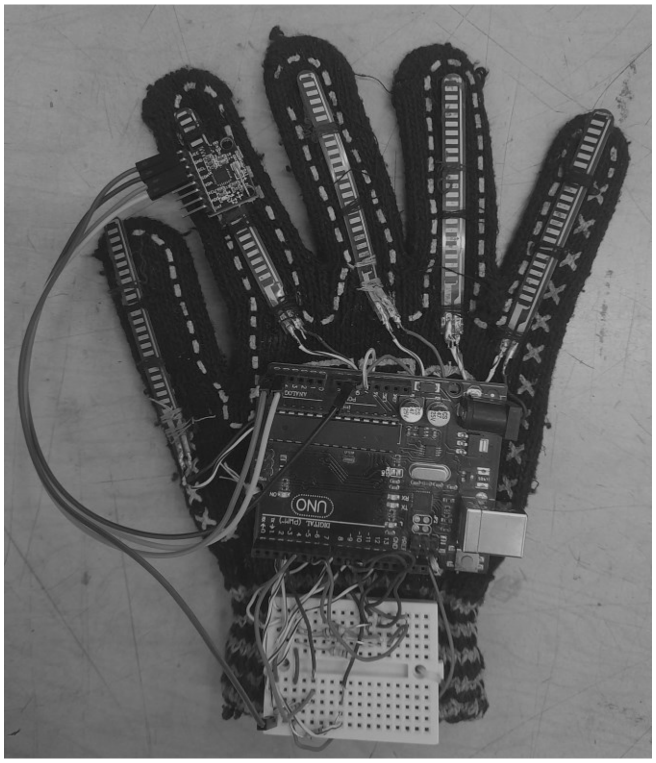


Figure 3-9: Accelerometer Connections to the Arduino Board

We put together five Flex sensors with the Arduino and to test their readability the same way we did with one Flex sensor. Using the same connection layout with difference in the part that each one has it is own unique pin of output data to the Arduino.

As the number of Flex sensors increased, came the need to make sure of their readings and hence a reading code was written to ensure the reliability and to know what sort of output each of the five is giving.

The whole project look was expected to be a glove with an exposed wires and sensors, but on a satisfactory level it looked unorganized with wires going at every direction. “it looks scary” said one of the test subjects we viewed the project to [9].

A work around was in order to change how the project look and it was decided to act on in against the wires first then the overall view. The wires were trimmed to get rid of the excess length, or either replaced with a good lengthen version of it when trimming was out of options.

When it came to resizing the overall dimensions of the circuit to allocate as much space on the glove as optimum, we decided to replace the normal sized bread board with a smaller one.

Some effort was also put into covering the exposed parts of the Flex sensors, by having the other hand glove –they came in pairs to cover both hands- cut off to take out the fingers from it. Afterwards the cut fingers were used to cover up the Flex sensors by waring it on the Flex sensors as it they were wearing a glove of their own.

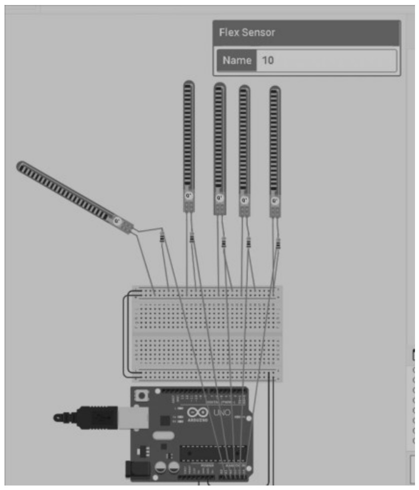


Figure 3-10: Circuite layout

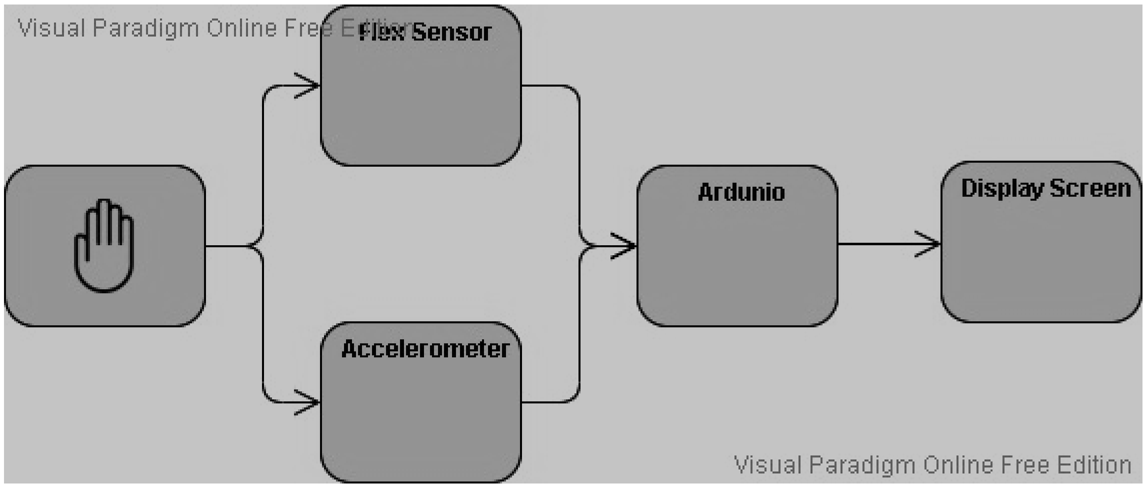

Visual Paradigm Online Free Edition
Flex Sensor
Accelerometer
Ardunio
Display Screen
Visual Paradigm Online Free Edition

Figure 3-11: Block diagram

3.5 Data acquisition

As the Flex sensors were fixed in the fingers of the glove, we started the data input phase of the project. And hence the building of the data basethat is going to be installed in the system and will be the main sign language for the device to translate into letters\words.

The choosing of our language is built on the mother tongue in the country we live in, Sudan, and that was the Arabic language. Hence, we are selecting the Arabic sign language, abbreviated as ArSL, to build the data base from.

After having acquired enough data from the interviews we did, we build up the conclusion that the sign language of matter includes 26 letters and zero words.

The starting point was to build the letters as a set of data, A simple and straight forward way of approach. The glove is to be worn and the letter is assisted by it is shape of the hand. The letter shape is kept for as long as it takes for the input to be read from the sensors in the glove 30 times.

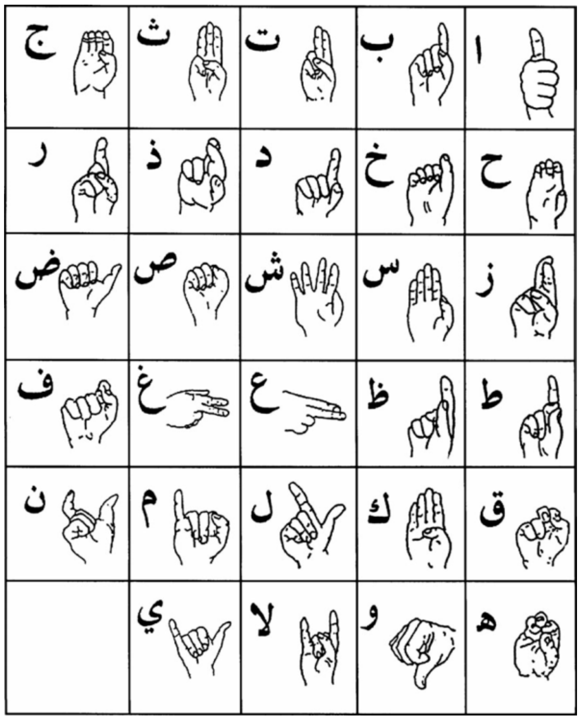


Figure 3-12: Aabic sign language

This is done to make the data base resilient when it comes to errors and verity of hand level of grip measure. We take the level of grip measure into account first and as a source of error. Secondly because not all people represent the same letter sign with the same level of bendiness each time, some has a tight hand that is able to make the letter roughly and other personnel make a soft sign of the letter.

The purpose behind taking several input for the same letter is also due to the need to make a list of all the possible inputs of the later which add to the level of reliability of the system, and from this list we make a range of values of the letter’s incomes.

3.6 Software

We wrote a program to enter the value we want to represent, below is a flowchart representing it.

We included the values into the database and performed a statistic representation of the values with the exclusion of the anomalous results and their representation in graphs.

And then we entered the results into a program that we wrote in order to display the letters translated from the sign language.

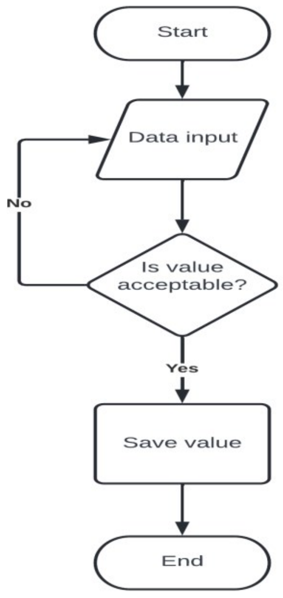


Figure 3-13: Getting Values of Letters Algorithm

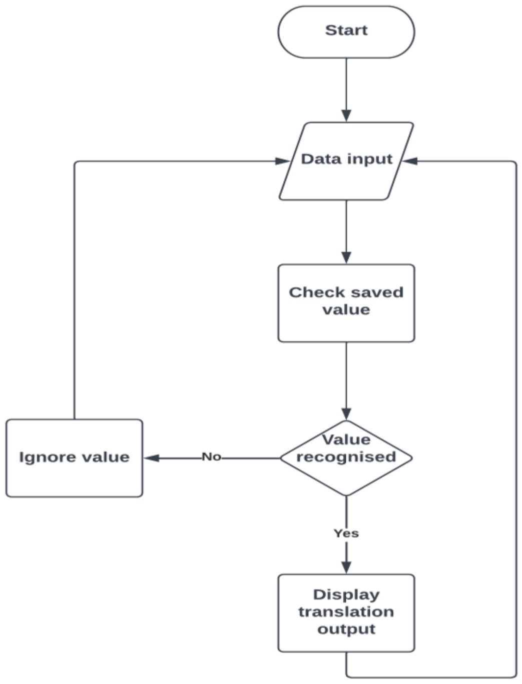


Figure 3-14: Translation Program

Chapter Four : Results and discussions

4.1 Notations and scenarios

4.1.1 Similarity in shape but difference in gesture

As the alphabets are listed in our language data base, there is some similarities between some letters, the alphabet (س) and (ش) are taken as an example of the methods we used to overcome it.

It is commonly known that any sign language, from any country, there are some similarities between assigned hand shape of different letters of the language. Take as an example (I) and (J) of the German sign language or (R) and (RR) of the Mexican sign language. All are illustrated bellow.

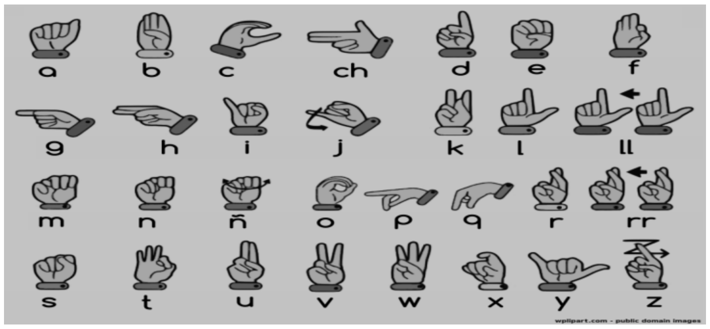


Figure 4-1: German sign language

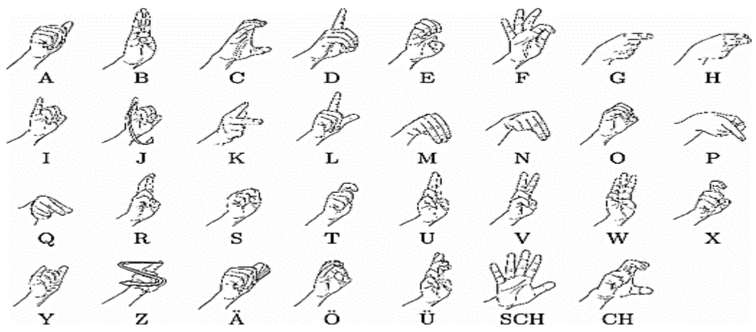


Figure 4-2: Mexican sign language

Almost in all the languages, written language as to say, two letters differ very much in their written shape. When we process them, listening-wise, the letters also can be distinguished from each other seamlessly as the sound produced by each when spoken are different.

When it comes to sign language, it is sometimes hard to tell them apart without having to focus on the gesture or the shape of the hand. Our system aims to lower down how much focus you need to put-up in order to tell them apart.

Here we take the possibility of the same hand shape but with a gesture involved. The two letters are exact -in hand shape- but are told apart by using a movement along with the shape of the hand. The accelerometer is used here to sense the movement of the hand, it can tell the difference of the hand position in the X-axis and the Y-axis and is available as one of the input that the Arduino Uno process to give the corresponding output.

4.1.2 Translation order

The proposed system translates sign language in a letter by letter progression, to form any word we spell the word by sign language, as the sign language only support signs of letters and very little words. The reason behind this, is that there are simply too many words to figure then assign a sign for.

4.1.3 Display method

the output methods we took into considerations and then used are two distinguishable methods. One is thought as a hardware approach and the other is the software approach.

On the hardware approach we are using an LCD as our way of display. This has the advantage of being an ease of use to both parties using the project. However, it will add to the overall cost of making the project, together as a unit or if it was made on an industrial scale.

On the software approach there was using Microsoft's Visual Studio Code. The application provides you with much power throughout coding or using extensions. A helpful extension that is used here is the Arduino extension providing a serial monitor. The extension provides ArduinoC language support and can compile as if it was an Arduino IDE, through this the extensions serves as to display letters translated by the project and display it on screen. The reason behind taking the initiate to move from the Arduino IDE to Visual Studio Code, is the support of UTF-8 encoding that is available in Microsoft's Visual Studio Code.

4.1.4 Software display method

On the software approach of the display method we had also two alternatives. The Android approach and the Visual Studio approach.

The android phone needs to have only two things for this method to work, a build-in Bluetooth module and an android application build to translate received data into letters.

The project would have had a Bluetooth module connected to the circuitry and will establish a connection with the phone's Bluetooth. The application is opened by the user and the translation output will be displayed on the phone. As the Arduino sends the output through the Bluetooth connection made available by the Bluetooth module.

The Microsoft's Visual Studio is the other software method. It was our choice in the project, and is discussed on section 4.1.3.

4.1.5 Simulating the project schematics

The project circuitry is done on a simulation software found on the web, named TINKERCAD . The choice of this particular software is due to its ability to simulate the flex sensor moving seamlessly and with a 3D movement [10].

It is found that the flex sensors used in the project have failed to work because the they are burned up. Hence, an alternative was needed to verify whether the working principle of the project is working. The flex sensors could not be replaced for they are not found on Sudan and if ordered will not be viable both time-wise and money-wise.

Chapter Five : Conclusion and recommendations

5.1. conclusion

It is concluded that the project we build wielded some good results at translating the Arabic sign language into written Arabic language and is working as supposed to. it also did the translation with a good accuracy as expected.

One might also note that there have been some alternations from the original plan, noted throughout the documentation -on the previous chapter- being read now, those alternations are manly highlighted in the fourth chapter of the project research.

5.2. Recommendations

It is highly recommended that the building of the data base starts by checking the mother tongue of the region you live in. For that is valuable in the sense of easing the hustle of looking up through language and it also provide one with the ability to test locally. And it does provide one with more insight on the language.

When applying in the output method of choice, always check for compatibility of the output method's characteristics as some of them has no formality for the language translation. in other words, the output way of choice might not display the language of translation correctly whether in order or in the shape of the characters.

The consideration to add multi-languages translation ability, to translate in-between multiple written language, so that the device holds the advantage to translate to non-Arabic language speakers.

References

[1] M. Erard, “The Atlantic,” 2021. [Online]. Available: https://www.theatlantic.com/technology/archive/2017/11/why-sign-language-gloves-dont-help-deaf-people/545441/.

[2] M. A. Abdel-Fattah, “Arabic Sign Language: A Perspective,” *Department of Languages and Translation, Birzeit University.*

[3] Wikipedia, “Visual Studio Code - Wikipedia,” 2022. [Online]. Available: https://en.wikipedia.org/wiki/Visual_Studio_Code. [Accessed 15 February 2022].

[4] S. Alapati and S. Yeole, “A Review on Applications of Flex Sensors,” *Development of Flex sensor array to Identify Damage to Sheet Metal,* July 2017.

[5] Cifer, “What is MPU6050 - Arduio Projects,” 2019. [Online]. Available: https://create.arduino.cc/projecthub/CiferTech/what-is-mpu6050-b3b178. [Accessed 15 October 2022].

[6] Krammer, “Fixed Resistors,” p. 4, 2001.

[7] Yasir Niaz Khan and Syed Atif Mehdi, “Sign language recognition using sensor gloves,” in *Neural Information Processing, 2002. ICONIP '02. Proceedings of the 9th*, December 2002.

[8] Shaheer Bin Rizwan, Muhammad Saad Zahid Khan and Muhammad Imran, “American Sign Language Translation via Smart,” Center for Advanced Studies in Engineering, Pakistan.

[9] T. M. Idrees, Interviewee, *Beta Viewing.* [Interview]. September 2021.

[10] AUTODESK, “TINKERCAD,” 2022. [Online]. Available: https://www.tinkercad.com. [Accessed March 2022].

[11] “Fritzing,” [Online]. Available: https://cdn.sparkfun.com/assets/learn_tutorials/5/1/1/example_circuit_bb.png. [Accessed 9th September 2021].

[12] “Circuit Design,” [Online]. Available: https://circuitdigest.com/sites/default/files/circuitdiagram_mic/ADXL345-Accelerometer-Interface-with-Arduino-Circuit-Diagram.png. [Accessed 15th September 2021].

[13] NIDCD, “American Sign Language,” NIH Publication No. 11-4756, 2019.

Appendixes

1. Arduino code:

```
        -------PIN
CONFIGURATION----------------
        A0-A4 : FLEX SENSOR
        D4&D5         :         FOR
BLUETOOTH RX AND TX
        A5&A6 : XPIN AND YPIN
FOR ACCELROMETER
        */
        #include
<SoftwareSerial.h>

        SoftwareSerial
mySerial(5,4);

        char temp = '0';

        //variable initializtion
        int xpin = A5;
        int xadc = 0;
        int xmax = 0;
        int xmin = 1023;

        int ypin = A6;
int yadc = 0;
int ymax = 0;
int ymin = 1023;

int FLEX_PIN1 = A0;
int flexADC1 = 0;
int sensorMin1 = 1023;
int sensorMax1 = 0;

int FLEX_PIN2 = A1;
int flexADC2 = 0;
int sensorMin2 = 1023;
int sensorMax2 = 0;

int FLEX_PIN3 = A2;
int flexADC3 = 0;
int sensorMin3 = 1023;
int sensorMax3 = 0;

int FLEX_PIN4 = A3;
int flexADC4 = 0;
int sensorMin4 = 1023;
int sensorMax4 = 0;
        int FLEX_PIN5 = A4;
        int flexADC5 = 0;
        int sensorMin5 = 1023;
        int sensorMax5 = 0;

        void setup()
        {
        mySerial.begin(9600);
        while (!Serial)
        {
        ; // wait for serial port to connect. Needed for native USB port only
        }
        // callibrating the sensors for adaptivity with different bends
        while(millis()<15000)
        {
        if(digitalRead(7)==HIGH)
        {
        float flexADC1 = analogRead(FLEX_PIN1);
        float flexADC2 = analogRead(FLEX_PIN2);
        float flexADC3 = analogRead(FLEX_PIN3);
        float flexADC4 = analogRead(FLEX_PIN4);
        float flexADC5 = analogRead(FLEX_PIN5);

if(flexADC1<sensorMin1)
{
sensorMin1=flexADC1;
}
if(flexADC1>sensorMax1)
{
sensorMax1=flexADC1;
}

if(flexADC2<sensorMin2)
{
sensorMin2=flexADC2;
}
if(flexADC2>sensorMax2)
{
sensorMax2=flexADC2;
}

if(flexADC3<sensorMin3)
{
sensorMin3=flexADC3;
}
        if(flexADC3>sensorMax3)
        {
        sensorMax4=flexADC4;
        }

        if(flexADC5<sensorMin5)
        {
        sensorMin5=flexADC5;
        }
        if(flexADC5>sensorMax5)
        {
        sensorMax5=flexADC5;
        }

        if(flexADC4<sensorMin4)
        {
        sensorMin4=flexADC4;
        }
        if(flexADC4>sensorMax4)
        {
        sensorMax4=flexADC4;
        }
        }
        }
        }
```

```
void printfun(char cp) //to avoid printing repeating symbols
{
if(cp!=temp)
{
mySerial.print(cp);
temp=cp;
}
}

void loop()
{
// reading sensor value
float flexADC1 = analogRead(FLEX_PIN1);
float flexADC2 = analogRead(FLEX_PIN2);
float flexADC3 = analogRead(FLEX_PIN3);
float flexADC4 = analogRead(FLEX_PIN4);
float flexADC5 = analogRead(FLEX_PIN5);

flexADC1 = constrain(flexADC1,sensorMin1, sensorMax1);
```

```
        flexADC2                          =
constrain(flexADC2,sensorMin2,
sensorMax2);
        flexADC3                          =
constrain(flexADC3,sensorMin3,
sensorMax3);
        flexADC4                          =
constrain(flexADC4,sensorMin4,
sensorMax4);
        flexADC5                          =
constrain(flexADC5,sensorMin5,
sensorMax5);


        float                        angle1=
map(flexADC1,          sensorMin1,
sensorMax1, 0, 90);
        float                        angle2=
map(flexADC2,          sensorMin2,
sensorMax2, 0, 90);
        float                        angle3=
map(flexADC3,          sensorMin3,
sensorMax3, 0, 90);
        float                        angle4=
map(flexADC4,          sensorMin4,
sensorMax4, 0, 90);
        float                        angle5=
map(flexADC5,          sensorMin5,
sensorMax5, 0, 90);

        xadc = analogRead(xpin);
        yadc = analogRead(ypin);

        if(((angle1>=70)&&(angle1
<=82))&&((angle2>=77)&&(angl
e2<=95))&&((angle3>=70)&&(an
gle3<=86))&&((angle4>=73)&&(
angle4<=85))&&((angle5>=0)&&
(angle5<=45)))
        printfun('A');
        if(((angle1>=0)&&(angle1<
=10))&&((angle2>=0)&&(angle2
<=10))&&((angle3>=0)&&(angle
3<=12))&&((angle4>=0)&&(angl
e4<=10))&&((angle5>=65)&&(an
gle5<=80)))
        printfun('B');
        if(((angle1>=40)&&(angle1
<=72))&&((angle2>=50)&&(angl
e2<=90))&&((angle3>=51)&&(an
gle3<=75))&&((angle4>=42)&&(
angle4<=66))&&((angle5>=34)&
&(angle5<=50)))
```

```
        printfun('C');
        if(((angle1>=50)&&(angle1
<=72))&&((angle2>=45)&&(angl
e2<=90))&&((angle3>=35)&&(an
gle3<=75))&&((angle4>=0)&&(a
ngle4<=10))&&((angle5>=45)&&
(angle5<=80))&&!(((xadc>=412)
&&(xadc<=418))&&((yadc>=340
)&&(yadc<=360))))
        printfun('D');
        if(((angle1>=68)&&(angle1
<=88))&&((angle2>=68)&&(angl
e2<=90))&&((angle3>=50)&&(an
gle3<=80))&&((angle4>=54)&&(
angle4<=80))&&((angle5>=58)&
&(angle5<=88)))
        printfun('E');
        if(((angle1>=0)&&(angle1<
=10))&&((angle2>=0)&&(angle2
<=10))&&((angle3>=0)&&(angle
3<=10))&&((angle4>=15)&&(ang
le4<=45))&&((angle5>=34)&&(a
ngle5<=65)))
        printfun('F');
        if(((angle1>=75)&&(angle1
<=90))&&((angle2>=75)&&(angl
e2<=90))&&((angle3>=65)&&(an
gle3<=90))&&((angle4>=0)&&(a
ngle4<=15))&&((angle5>=0)&&(
angle5<=30))&&(((xadc>=400)&
&(xadc<=420))&&((yadc>=340)
&&(yadc<=360))))
        printfun('G');
        if(((angle1>=70)&&(angle1
<=85))&&((angle2>=75)&&(angl
e2<=90))&&((angle3>=0)&&(ang
le3<=10))&&((angle4>=0)&&(an
gle4<=10))&&((angle5>=50)&&(
angle5<=65))&&!(((xadc>=410)&
&(xadc<=420))&&((yadc>=368)
&&(yadc<=380))))
        printfun('H');
        if(((angle1>=0)&&(angle1<
=10))&&((angle2>=50)&&(angle
2<=70))&&((angle3>=50)&&(ang
le3<=70))&&((angle4>=50)&&(a
ngle4<=70))&&((angle5>=50)&&
(angle5<=85)&&((xadc>=410)&
&(xadc<=420))&&((yadc>=330)
&&(yadc<=370))))
        printfun('I');
        if(((angle1>=0)&&(angle1<
=10))&&((angle2>=50)&&(angle
2<=70))&&((angle3>=50)&&(ang
```

```
le3<=70))&&((angle4>=50)&&(a
ngle4<=70))&&((angle5>=50)&&
(angle5<=85))&&(!((xadc>=410)
&&(xadc<=420))&&((yadc>=355
)&&(yadc<=370))))
        printfun('J');
        if(((angle1>=60)&&(angle1
<=75))&&((angle2>=60)&&(angl
e2<=85))&&((angle3>=0)&&(ang
le3<=10))&&((angle4>=0)&&(an
gle4<=15))&&((angle5>=30)&&(
angle5<=55))&&(((xadc>=404)&
&(xadc<=415))&&((yadc>=368)
&&(yadc<=380))))
        printfun('K');
        if(((angle1>=75)&&(angle1
<=90))&&((angle2>=75)&&(angl
e2<=90))&&((angle3>=70)&&(an
gle3<=90))&&((angle4>=0)&&(a
ngle4<=15))&&((angle5>=0)&&(
angle5<=30))&&(((xadc>=390)&
&(xadc<=405))&&((yadc>=360)
&&(yadc<=380)))&&!((xadc>=27
0)&&(xadc<=300))&&((yadc>=3
60)&&(yadc<=390)))
        printfun('L');
        if(((angle1>=40)&&(angle1
<=61))&&((angle2>=72)&&(angl
e2<=84))&&((angle3>=45)&&(an
gle3<=65))&&((angle4>=62)&&(
angle4<=75))&&((angle5>=65)&
&(angle5<=86)))
        printfun('M');
        if(((angle1>=54)&&(angle1
<=70))&&((angle2>=50)&&(angl
e2<=61))&&((angle3>=48)&&(an
gle3<=66))&&((angle4>=60)&&(
angle4<=76))&&((angle5>=50)&
&(angle5<=65))&&(((xadc>=400)
&&(xadc<=435))&&((yadc>=350
)&&(yadc<=390))))
        printfun('N');
        if(((angle1>=68)&&(angle1
<=88))&&((angle2>=68)&&(angl
e2<=90))&&((angle3>=50)&&(an
gle3<=80))&&((angle4>=54)&&(
angle4<=80))&&((angle5>=0)&&
(angle5<=30)))
        printfun('O');
        if(((angle1>=60)&&(angle1
<=75))&&((angle2>=60)&&(angl
e2<=85))&&((angle3>=0)&&(ang
le3<=10))&&((angle4>=0)&&(an
```

```
gle4<=15))&&((angle5>=30)&&(
angle5<=55))&&(((xadc>=270)&
&(xadc<=290))&&((yadc>=360)
&&(yadc<=380))))
        printfun('P');
        if(((angle1>=75)&&(angle1
<=90))&&((angle2>=75)&&(angl
e2<=90))&&((angle3>=65)&&(an
gle3<=90))&&((angle4>=0)&&(a
ngle4<=15))&&((angle5>=0)&&(
angle5<=30))&&(((xadc>=270)&
&(xadc<=300))&&((yadc>=360)
&&(yadc<=390))))
        printfun('Q');
        if(((angle1>=40)&&(angle1
<=72))&&((angle2>=45)&&(angl
e2<=90))&&((angle3>=20)&&(an
gle3<=45))&&((angle4>=0)&&(a
ngle4<=10))&&((angle5>=45)&&
(angle5<=80))&&(((xadc>=412)&
&(xadc<=418))&&((yadc>=340)
&&(yadc<=360))))
        printfun('R');
        if(((angle1>=70)&&(angle1
<=90))&&((angle2>=80)&&(angl
e2<=90))&&((angle3>=80)&&(an
gle3<=90))&&((angle4>=80)&&(
angle4<=90))&&((angle5>=60)&
&(angle5<=80)))
        printfun('S');
        if(((angle1>=40)&&(angle1
<=61))&&((angle2>=72)&&(angl
e2<=84))&&((angle3>=45)&&(an
gle3<=65))&&((angle4>=44)&&(
angle4<=63))&&((angle5>=65)&
&(angle5<=86))&&(digitalRead(6
)==HIGH))
        printfun('T');
        if(((angle1>=70)&&(angle1
<=90))&&((angle2>=80)&&(angl
e2<=90))&&((angle3>=0)&&(ang
le3<=10))&&((angle4>=0)&&(an
gle4<=10))&&((angle5>=60)&&(
angle5<=80)))
        printfun('U');
        if(((angle1>=70)&&(angle1
<=90))&&((angle2>=80)&&(angl
e2<=90))&&((angle3>=0)&&(ang
le3<=10))&&((angle4>=0)&&(an
gle4<=10))&&((angle5>=60)&&(
angle5<=80))&&(digitalRead(6)=
=HIGH))
        printfun('V');
        if(((angle1>=70)&&(angle1
<=90))&&((angle2>=0)&&(angle
2<=10))&&((angle3>=0)&&(angl
e3<=10))&&((angle4>=0)&&(ang
le4<=10))&&((angle5>=60)&&(a
ngle5<=80)))
        printfun('W');
        if(((angle1>=50)&&(angle1
<=72))&&((angle2>=45)&&(angl
e2<=90))&&((angle3>=35)&&(an
gle3<=75))&&((angle4>=80)&&(
angle4<=89))&&((angle5>=45)&
&(angle5<=80)))//&&!(((xadc>=4
12)&&(xadc<=418))&&((yadc>=
340)&&(yadc<=360))))
        printfun('X');
        if(((angle1>=0)&&(angle1<
=10))&&((angle2>=70)&&(angle
2<=90))&&((angle3>=60)&&(ang
le3<=80))&&((angle4>=80)&&(a
ngle4<=90))&&((angle5>=15)&&
(angle5<=35)))
        printfun('Y');
        if(((angle1>=50)&&(angle1
<=72))&&((angle2>=45)&&(angl
e2<=90))&&((angle3>=35)&&(an
gle3<=75))&&((angle4>=0)&&(a
ngle4<=10))&&((angle5>=45)&&
(angle5<=80))&&(((xadc>=412)&
&(xadc<=418))&&((yadc>=340)
&&(yadc<=360))))
        printfun('Z');
        else printfun('?');

        delay(200);


        }
        //----------------------END----
------------------------
```