DEAF / DUMB PEOPLE HOSPITAL SUPPORT SYSTEM

A Project report submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

B. Sai Babu (319126512134)	G. Sai Devi (319126512145)
M. Vamsi (319126512162)	B. Mounica (319121512138)

Under the guidance of

Mr. Srinivasa Naidu Nalla

Assistant Professor

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES

(UGC AUTONOMOUS)

(Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC)
Sangivalasa , bheemili mandal, Visakhapatnam dist.(A.P) (2022-2023)

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

ANIL NEERUKONDA INSTITUTE OF TECHNOLOGY AND SCIENCES

(UGC AUTONOMOUS)

(Permanently Affiliated to AU, Approved by AICTE and Accredited by NBA & NAAC)

Sangivalasa, Bheemili Mandal, Visakhapatnam dist.(A.P)

PRAGNANAM BRAHMA

ANITS

CERTIFICATE

This is to certify that the project report entitled "**DEAF/DUMB PEOPLE HOSPITAL SUPPORT SYSTEM**" submitted by B.SAI BABU (319126512134), G.SAIDEVI (319126512145), M.VAMSI (319126512162), B.MOUNICA (319126512138) in partial fulfillment of the requirements for the award the degree of Bachelor of Technology in Electronics and Communication Engineering of Andhra University, Visakhapatnam is a record of bonified work carried out under my guidance and supervision.

[illegible] 21/4/23	[illegible]
Project Guide	**Head of the Department**
Mr. Srinivasa Naidu Nalla	**Dr. B. Jagadeesh**
Assistant Professor	Professor & HOD
Department of E.C.E	Department of E.C.E
ANITS	ANITS
Assistant Professor Department of E.C.E. Anil Neerukonda Institute of Technology & Sciences Sangivalasa, Visakhapatnam-531 162	Head of the Department Department of E C E Anil Neerukonda Institute of Technology & Sciences Sangivalasa - 531 162

ACKNOWLEDGEMENT

We want to convey our sincere thanks to **Mr. Srinivasa Naidu Nalla**, Assistant Professor, Department of Electronics and Communication Engineering, ANITS, for leading us through this project with his/her unmatched expertise and tremendous support.

We are grateful to **Dr. B. Jagadeesh**, Head of the Department of Electronics and Communication Engineering, gave us the resources we needed to finish the project.

For their assistance and cooperation in this study, the **Principal and Management of ANITS, Sangivalasa**, have our sincere gratitude. We want to express our gratitude to the **teaching personnel** of department ECE ,whose suggestions during reviews helped us in accomplishment of our project.

We would like to thank **all non-teaching personnel** for their invaluable assistance in getting our project done. We would want to express our gratitude to our parents, friends, and co-workers who helped us during this study. Lastly, we would want to express our gratitude to everyone who helped make this initiative a success, both directly and indirectly.

Project Students

B.SAI BABU(319126512134)

G.SAI DEVI(319126512145)

M.VAMSI(319126512162)

B.MOUNICA(319126512138)

ABSTRACT

From our ancestors to present generation human beings convey their feelings through communication. Communication plays a vital role in human life. What if the person doesn’t possess the power of speech? In this case the only way to covey with others is through sign language. In this regard we conducted a survey and identified a serious communication lapse exiting between deaf/dumb patients and doctors. To address this problem we want to develop a hand glove which will establish a communication between deaf/dumb patients and doctors by converting hand gestures into voice. With the help of this hand glove, they can express their inconvenience to the doctor with ease. The Sensors in the glove will be designed to read the pre-determined signs to decode the sign and intimate as through any electronic gadget connected to it (mobile). It leads to the removal of communication.

Keywords: Smart hand glove, Internet of things, Arduino UNO, Flex Sensors, Accelerometer

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

The World Health Organisation (WHO) also estimates that 466 million people, or 5% of the world's population (432 million adults and 34 million children), are deaf. These people are unable to communicate with others because they are unable to hear or speak.

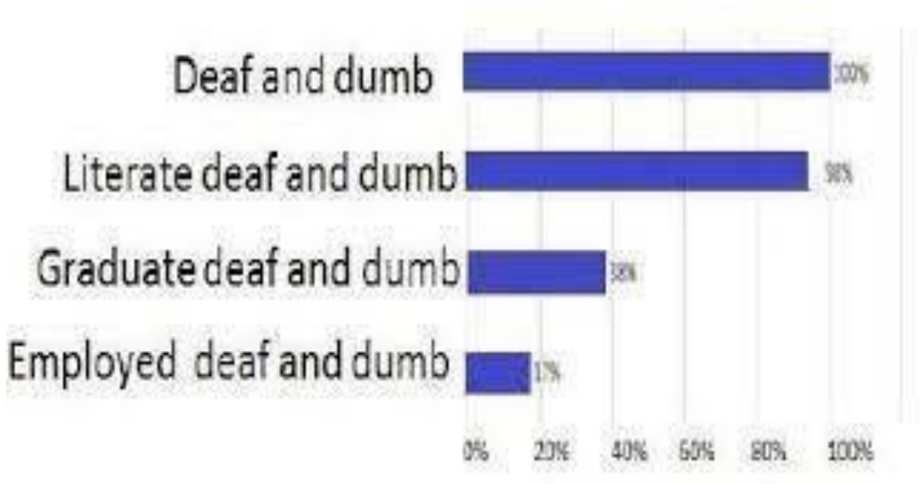


Fig. 1.1**:** Survey on Deaf and Dumb people

Communication is the only medium where we can share our thoughts and send messages, but language and speech cannot compete with normal people in order to communicate with normal people for the disabled, and with normal people for the hearing impaired. It is visual, not auditory. Idiots often use language to communicate, but have a hard time communicating with people who don't understand the language. Therefore, there is a communication barrier between the two communities. To overcome this problem, we present a smart glove that can translate sign language into the English word I used. Whenever a deaf person sees sign, this glove will translate the sign into relevant English in a way that ordinary people can understand. In this device, the switch attached to the finger plays an important role and when the finger is bent it changes the resistance according to the bending value of the sensor. We also use different sensors such as accelerometers to detect movements. We use bluetooth module to send data from smart glove to mobile phone and Arduino bluetooth script for request to receive message.

The main purpose of this publication is to communicate between doctors and the hearing impaired. Hand motions may be translated into electrical signals by the gadget presented in this project, which can then be sent to a microcontroller, which can then translate the signals into letters or commands. By limiting contact between the community and physicians, this would be very advantageous to these deaf individuals. It is affordable, small, adaptable, and uses less electricity to run.

1.1 PROJECT OBJECTIVE:

Once completed, the project will become a medium for the deaf to communicate with doctor and ordinary people. The words of the deaf cannot be understood by normal people. Thus, the device translates sign language into speech or voice that ordinary people can easily understand.

1.2 OUTCOME AND EDUCATION IMPLICATIONS:

The curriculum offers an excellent method for engaging students and promoting the growth of key 21st century skills like creativity, problem-solving, critical thinking, and presentation skills. The SOLE concept has given them the tools they need to understand how to overcome obstacles via efficient teamwork and a positive outlook.

1.3 METHODOLOGY:

To carry it out, we contemplated applying the design thinking methodology. This includes:

1.Brainstorming
2.Idea and research gathering
3.Prototype design.
4. An Arduino design and programming model
5. Model measurement and testing

1.4 SCOPE OF THE PROJECT:

Smart gloves make everyone equal. Dumb people can't communicate, these smart gloves can help in reducing communication barrier by allowing users to manually enter texts and pre-written messages.

1.5 THESIS OUTLINE:

During the course of the final six chapters, this thesis is stated. The summary of the Domain Description is presented in Chapter 2. The project's numerous parts and sensors are described in detail in Chapter 3. An overview of sign language description is given in Chapter 4. The detailed overview of Arduino programming is provided in Chapter 5. The hardware results and comments are covered in Chapter 6. Conclusions are discussed in Chapter 7.

CHAPTER 2
DOMAIN SPECIFICATION

This project is related to Embedded System domain, as we are interfacing flex sensors and accelerometer with Arduino UNO (micro controller).When Specialised hardware or electronics are combined with a computer processor, computer memory, and input/output peripherals, the result is a computer system, which is an embedded system. It frequently functions as part of a complex apparatus that also has mechanical and electrical parts. The embedded system typically has time constraints since it frequently controls the actual operations of the machine in which it is installed. Embedded systems power a lot of contemporary products.

Fig. 2.1: An embedded system with peripherals

Most modern embedded systems (i.e., microprocessors with embedded controllers) are built on microcontrollers. Ordinary microprocessors are also available, especially in more complex system contexts (where they use separate chips for memory and peripheral interface circuits). In both situations, the type of processor utilised can be general-purpose, specialised for a certain class of computing, or even a processor created especially for the application in question.

The size of embedded systems ranges from portable gadgets like digital clocks players to large systems such as household appliances, assembly lines, robots, vehicles, lighting control a medical imaging. They often form subsystems of other systems, such as avionics in aircraft an Astro electronics in spaceships. Large facilities such as factories, pipelines and power lines rely on many interconnected systems.

Embedded systems such as generalized, programmable logic controllers with software customization often have their own functionality.

Embedded systems can range from low complexity systems to high complexity systems with numerous components, peripherals, and networks spanning broad regions that can be organised in racks or connected via long-term connection by lines.

2.1CHARACTERISTICS OF EMBEDED SYSTEMS:

Unlike general-purpose computers, embedded systems are designed to perform a small number of highly specialised tasks. Some may only have very low or no performance requirements, which makes the hardware more adaptable and affordable. Others might have real-time performance requirements that must be met for usability reasons. Not every embedded system works alone. Tiny components that are located inside larger, more significant embedded systems make up many of them. Firmware, which is read-only or stored on flash memory chips, is the term for instructions that are written to the computer. They use extremely little computer hardware to function, including a keyboard, monitor, and memory.

2.2 APPLICATIONS OF EMBEDDED SYSTEMS:

Consumer, commercial, automotive, appliance, medical, telecommunications, industrial, aerospace, and military applications all often use embedded systems. The network switch and user's mobile phone are only two examples of the numerous embedded systems used in telecommunications networks. Data links and specialised routers are used by computers. Electronic devices include things like MP3 players, TVs, smartphones, gaming consoles, digital cameras, GPS receivers, and printers. Microwave ovens, washing machines, and functioning are examples of household gadgets that provide comfort and utility. Temperature sensors are used in modern heating, ventilation, and air conditioning (HVAC) systems to precisely connect and control temperatures that vary throughout the year.

CHAPTER 3
DESCRIPTION OF COMPONENTS

3.1 FLEX Sensors:

A flex sensor, often referred to as a flexural sensor, is a sensor that measures the amount of deflection or bending. The sensor is typically attached to something, and by bending the ground, the resistance of the primary sensor is changed. These sensors can be created using materials like carbon and plastic. The sensor's resistance varies as a carbon monoxide-filled plastic strip is bent.

3.1.1 Flex Sensor Overview:

It is also called bending sensor as it changes with bending. Since it can be adjusted according to it functions as a potentiometer according to the number of turns. These sensors are divided into two according to their size as 2.2inch flexible sensor and 4.5inch flexible sensor. Besides the principle of operation, the sensors differ in size and resistance. So, you can choose the size as you like.
It has two terminals with terminals like p1 and p2. This sensor has no polarized terminals like a diode or capacitor, i.e., no positive or negative. The sensor requires a voltage range of 3.3 V - 5 V DC to activate the sensor and can be received by any interface.

Fig. 3.1: flex sensor

- **Pin P1:** This pin is typically fastened to the positive terminal of the power supply.
- **Pin P2:** This pin frequently connects to the GND pin of the power supply.

3.1.2 Working principle:

The sensor is based on the idea that as a curved strip is bent, it changes. This can be measured with any controller. The sensor acts like a variable resistor when bent, changing resistance. The resistance varies when the surface is horizontal, hence the linearity of the surface will also have an impact on the fluctuation.
When the sensor is turned 45 degrees, the resistance will change. Also, when this meter is rotated 90 degrees, the resistance will be different. These are the convolutions of the three variants.

According to these three states, the resistance of the first state is the same, the resistance of the first state is twice, and the resistance of the first state is four times. That is, as the angle increases, the drag increases.

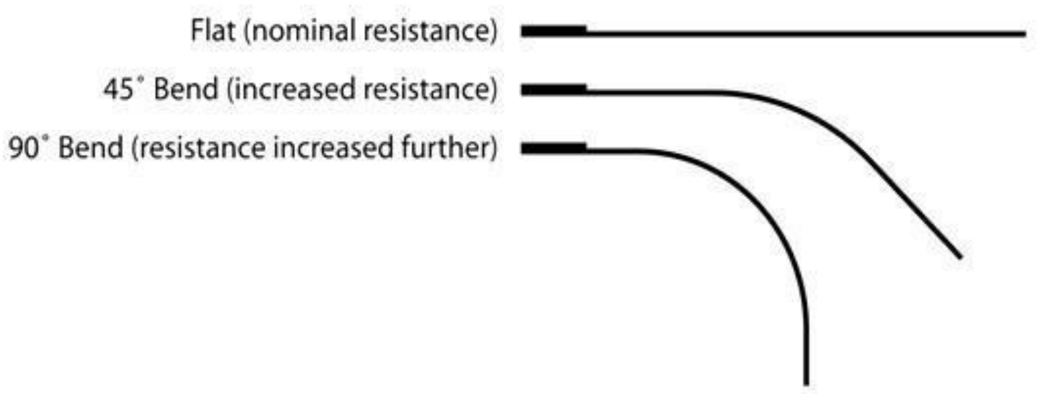

Flat (nominal resistance)
45° Bend (increased resistance)
90° Bend (resistance increased further)

Fig. 3.2: working of flex sensor

3.1.3 Interfacing of flex sensor:

3.1.3.1 Connection of flex sensor:

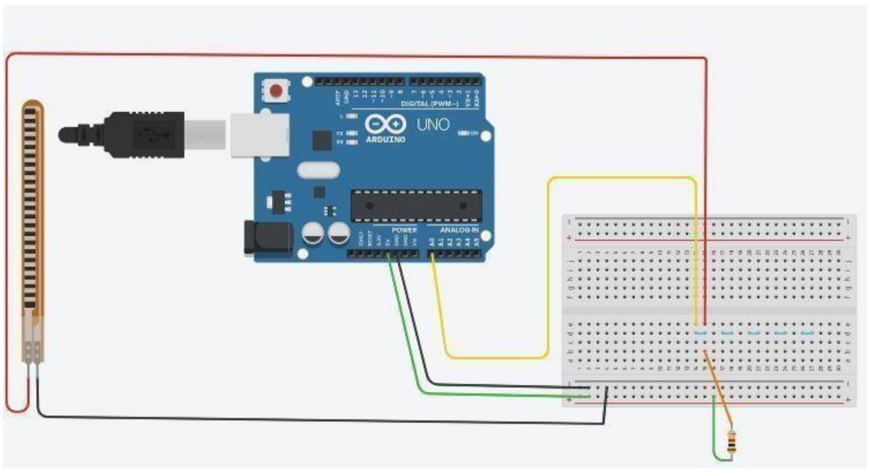
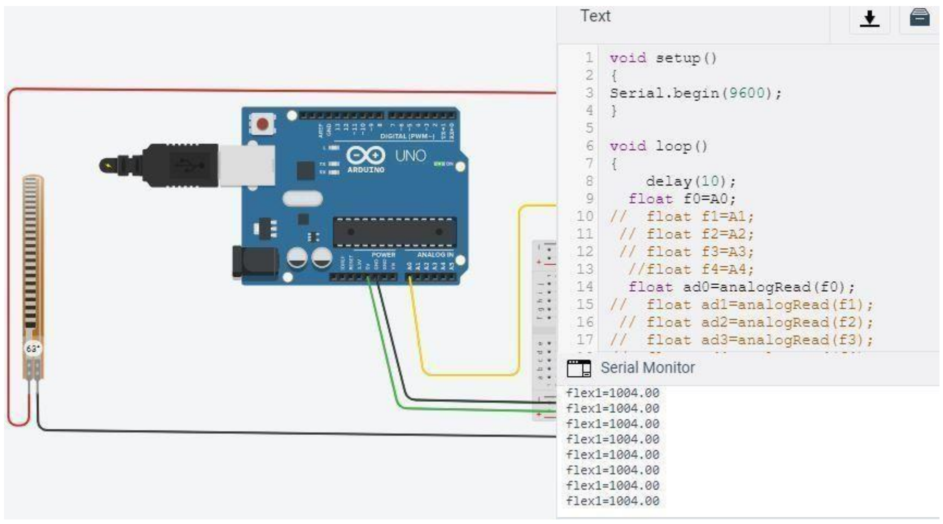


Fig. 3.3: connection of flex sensor on bread board

Pin 1 is connected to the ground pin of Arduino and pin 2 is connected to data pin(A0) and power supply(vcc) via 4.7kohm resistor.

3.1.3.2 Results of Flex sensor:



```
void setup()
{
Serial.begin(9600);
}

void loop()
{
    delay(10);
  float f0=A0;
//  float f1=A1;
 // float f2=A2;
 // float f3=A3;
  //float f4=A4;
  float ad0=analogRead(f0);
//  float ad1=analogRead(f1);
 // float ad2=analogRead(f2);
//  float ad3=analogRead(f3);
```

```
flex1=1004.00
flex1=1004.00
flex1=1004.00
flex1=1004.00
flex1=1004.00
flex1=1004.00
flex1=1004.00
flex1=1004.00
```

Fig. 3.4: Results of flex sensor

The readings are varying from 600 to 1100 based on the different types of bend angles.

3.2 ACCELEROMETER SENSOR (MPU 6050):

The MPU6050 sensor module offers a complete 6-axis movement tracking system. It comes with a 3-axis gyroscope, 3-axis accelerometer, and a digital motion processor in a small size. Additionally, a temperature sensor is integrated inside the semiconductor itself. It contains an I2C bus interface for communication with microcontrollers, as well as a pressure sensor, a three-axis magnetometer, etc. It includes a further I2C bus for attaching to various sensor devices, including The gyroscope and accelerometer on the MPU6050 can be used to monitor rotation in three axes, static acceleration caused by gravity, and dynamic acceleration caused by force, shock, or vibration.

3.2.1 Accelerometer sensor (MPU 6050) overview:

Utilising microelectromechanical systems (MEMS) technology, the MPU6050 incorporates a 3axis gyroscope. It is employed to regulate the X, Y, and Z axis rotational speed. Microelectromechanical (MEM) technology is used in the 3-axis accelerometer of the MPU6050. Along the X, Y, and Z axes, it is used to specify angles.

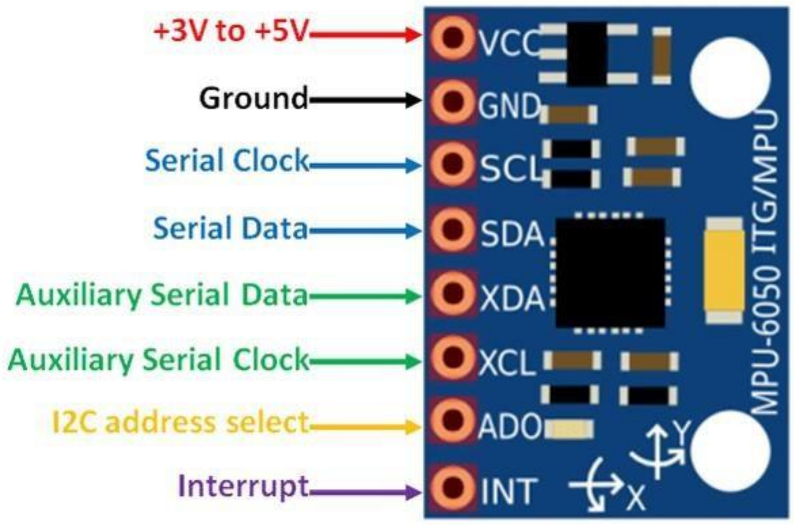


Fig. 3.5: MPU 6050 module pinout

- The module receives its power from VCC. Connect it to the 5V output of the Arduino.
- The ground pin on the Arduino must be connected.
- SCL is the I2C clock pin. The Bus Grip gadget delivered that time indication. connect to the SCL pin on the Arduino.
- SDA pin for I2C statistics. This line is simultaneously used for transmission and reception. connect to the SDA pin on the Arduino.
- XDA stands for the external I2C data line. The I2C bus is used to connect external sensors.
- The outer clock line, or XCL, of I2C is utilised.
- AD0 can be used to modify the internal I2C address of the MPU6050 module. If the module is incompatible with any other I2C device, two MPU6050s may be used on the same I2C bus.
- If you leave the ADO pin unconnected, the I2C deal with is 0x68HEX by default, and it changes to 0x69HEX when you connect it to 3.3V.
- The prefix INT designates the interrupt output. When the MPU6050 detects motions, tremors, panning, zooming, or scrolling, it can be set to raise the interrupt level.

3.2.2 Working of MPU 6050:

1.Measuring Acceleration:

The MPU6050 can track acceleration by using an accelerometer with four customizable sums of 2 g, 4 g, 8 g, and 16 g. The MPU6050 supports the movement of a model along three axes, including the X, Y, and Z axes, using three 16-bit analog to digital converters.

2.Measuring Rotation:

The MPU6050 can track angular rotation using its on-chip gyroscope, which has four programmable ranges: 250°/s, 500°/s, 1000°/s, and 2000°/s. On the MPU6050, additional 16-bit analog-to-digital converters simultaneously sample three rotational axes (the X, Y, and Z axes). A sampling frequency between 3.9 and 8000 samples per second is available for selection.

3.Measuring Temperature:

The built-in temperature sensor in the MPU6050 can monitor temperatures between -40 and 85°C with a 1°C precision. Please take note that the silicon die itself, not the ambient air, is what this temperature reading relates to. Such measurements are commonly made to adjust for accelerometer and gyroscope calibration or to error on the side of temperature modifications rather than measuring absolute temperatures.

3.2.3 Interfacing of accelerometer:

3.2.3.1 Connection of accelerometer:

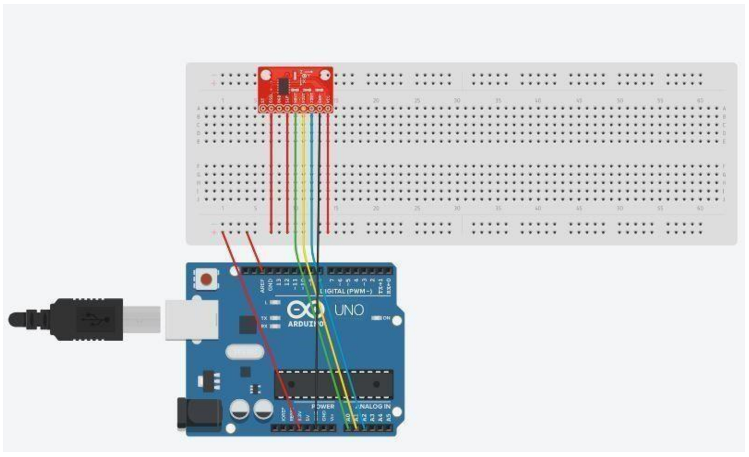
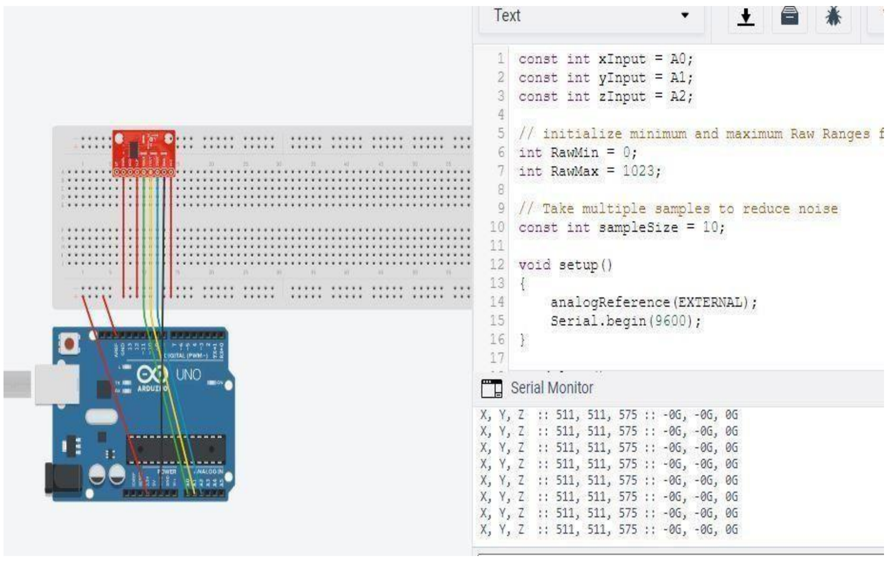


Fig. 3.6: Connection of accelerometer on bread board

Vcc of accelerometer is connected to vcc of Arduino and gnd pin of accelerometer is connected to gnd of Arduino, SDA of accelerometer is connected to SDA of Arduino, and SCL of accelerometer is connected to SCL of Arduino. A0, A1 and A2 are connected to X, Y, Z respectively.

3.2.3.2 Results of accelerometer:



```
const int xInput = A0;
const int yInput = A1;
const int zInput = A2;

// initialize minimum and maximum Raw Ranges f
int RawMin = 0;
int RawMax = 1023;

// Take multiple samples to reduce noise
const int sampleSize = 10;

void setup()
{
    analogReference(EXTERNAL);
    Serial.begin(9600);
}

```

```
X, Y, Z  :: 511, 511, 575 :: -0G, -0G, 0G
X, Y, Z  :: 511, 511, 575 :: -0G, -0G, 0G
X, Y, Z  :: 511, 511, 575 :: -0G, -0G, 0G
X, Y, Z  :: 511, 511, 575 :: -0G, -0G, 0G
X, Y, Z  :: 511, 511, 575 :: -0G, -0G, 0G
X, Y, Z  :: 511, 511, 575 :: -0G, -0G, 0G
X, Y, Z  :: 511, 511, 575 :: -0G, -0G, 0G
X, Y, Z  :: 511, 511, 575 :: -0G, -0G, 0G
```

Fig. 3.7: Results of accelerometer

The values of X, Y and Z axes are varying from 100 to 600 based on the motion of our wrist.

3.3 HC-05 BLUETOOTH MODULE:

For open wireless communication, the HC05 Class 2 Bluetooth module was developed. Already configured to function as a Bluetooth slave device. When connected to a Bluetooth host device like PCs, cell phones, or tablets, users can simply comprehend how it operates. Wireless communication of every bit of information gathered from the serial input in real time. If the module doesn't receive data before sending over the serial port, it won't. Every user microcontroller software does not require a user code unique to the Bluetooth module.

3.3.1 HC-05 Bluetooth Module overview:

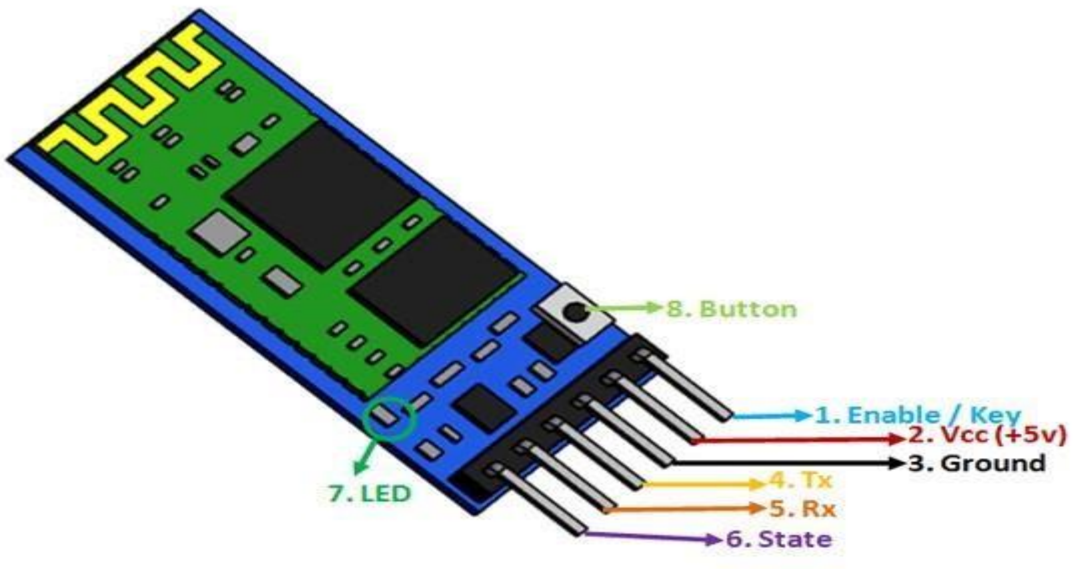


Fig. 3.8: HC-05 Bluetooth Module

- Activate / Key: Toggle between records Mode (set low) and AT command using this pin. Toggle between records Mode (low setting) and AT command mode (high setting) using this pin. It is currently in information mode by default.
- Vcc: energises the module. connect to a +5V power supply.
- Ground: Connect the module's ground pin to the device's floor.
- TXD–Transmitter: Serial records are transmitted. This pin will distribute anything received via Bluetooth in the form of serial data.
- RXD–Receiver: Gain access to Serial statistics. Each serial information provided to this pin may be transmitted over Bluetooth.
- State: The state pin can be used to test whether Bluetooth is functioning properly because it is connected to an on-board LED.
- LED: hints at Module's repute
- The module has entered command mode; blink once every two seconds.

•Repeated Blinking: anticipating connection in statistics Mode

•Blink twice in 1 sec: Connection successful in statistics Mode. • Button: Used to manipulate the key/permit pin to toggle among facts and command.

3.3.2 Working of Bluetooth module:

- Either the command mode or the data mode can be used by the HC05. The HC05 may change between operational modes by pressing the built-in button. Pressing the button will activate the HC05's command mode. Using a serial to TTL converter and terminal software running on the host, the user can change settings (such pin code and baud rate) in command mode. All system parameter changes will be kept even after the device is turned off. Resetting the HC05 will put it back in file mode. For transparent UART data transmission to and from remote connections, only files are accepted.
- The HC05 can be user-configured to work as a master Bluetooth device via a series of AT commands. When configured as a master, it can be coupled with an HC05 or HC06 module in a pre-slave configuration to enable peer-to-peer communication.
- The HC-05 can function with a supply voltage range of 3.6 to 6 volts, although the logic level of the RXD pin is only 3.3 volts. It is not typically tolerant to 5 volts. It is recommended to use a common-sense degree converter to isolate the sensor if you attach it to a 5V device (such as an Arduino Uno or Mega). The HC-05's power will be turned off if the "EN" pin is fully depressed to common sense zero.

3.4 ARDUINO UNO:

The Arduino UNO microcontroller board, which is ATmega328P-based. It has a USB connection, a power input, an ICSP header, a reset button, a 16 MHz ceramic resonator, 6 analogue inputs, 14 input/output pins, 6 of which can be used as PWM outputs. Contains everything needed to support the microcontroller; all you need to do is connect it to your computer with a USB cable, or power it using an ACDC converter or a battery. The Italian word "uno," which translates to "one," was chosen to denote a fresh Arduino hardware and software improvement.

3.4.1 Arduino UNO Overview:

A USB cable, which functions as both a serial port and a power source, can be used to rapidly connect the board to the computer.

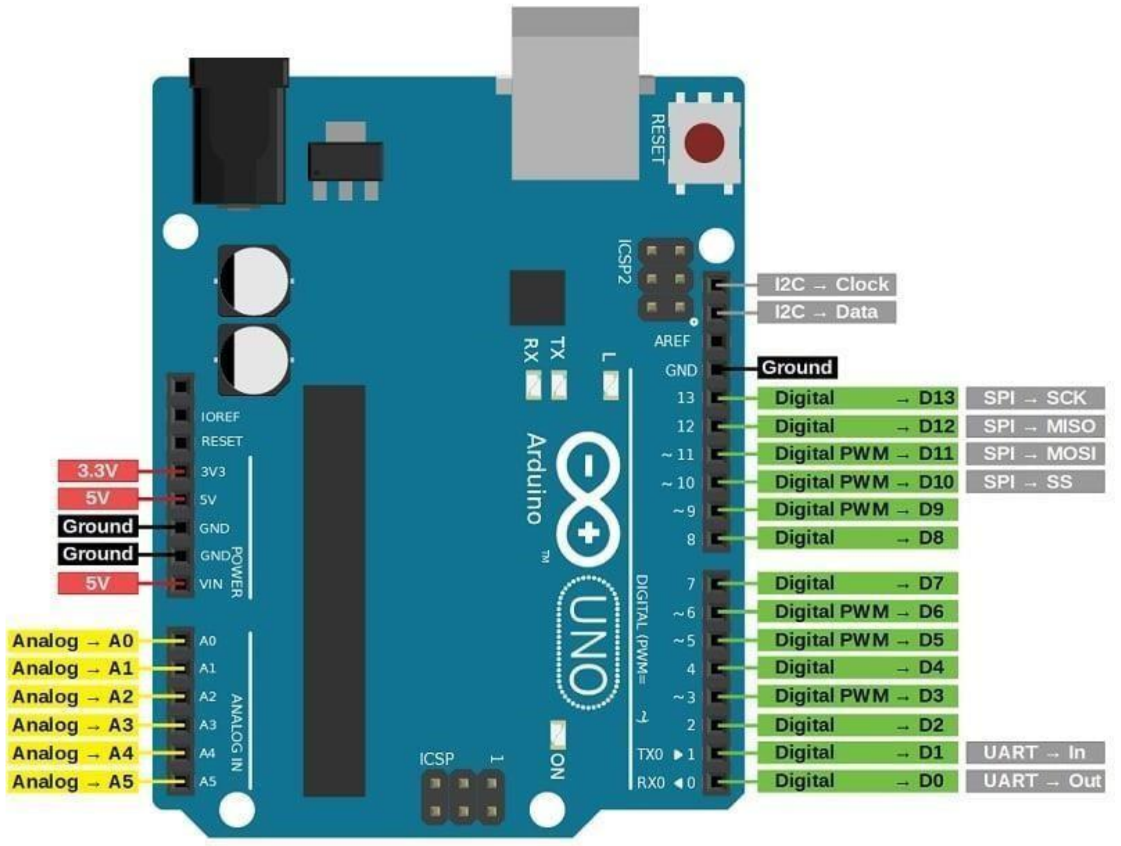


Fig. 3.9: Arduino uno

- Vin: This is the Arduino board's input voltage pin used to provide input supply from an external power source.
- 5V: This Arduino board pin serves as a regulated power supply voltage and is utilised to supply power to the board as well as onboard components.
- 3V: A voltage regulator on the board generates 3.3V at this pin, which is utilised to provide that voltage. • GND: The Arduino board is grounded using this pin on the board.
- Reset: The microcontroller is reset using this board pin. The microcontroller is reset using it frequently. • Analog Pins: The analogue input pins (A0 to A5) are located inside the• digital Pins: The pins zero to thirteen are used as a digital enter or output for the Arduino board. • Serial pins are also referred to as UART pins. It is currently used to communicate with other devices, such as laptops, in addition to the Arduino board. To broadcast and receive the information, respectively, the transmitter pin number one and receiver pin quantity 0 are employed.
- External Interrupt Pins: This pin on the Arduino board is utilised to supply the external interrupt, and pins 2 and 3 are used to accomplish this.

- PWM Pins: By varying the pulse width, these board pins are utilised to convert a virtual signal into an analogue one. As a PWM pin, the digits 3, 5, 6, 9, 10, and 11 are employed.
- Serial Peripheral Interface pins, also known as SPI pins, are used to maintain SPI verbal exchange with the aid of the SPI library. SPI pins include:
- SS: Slave choice is made via pin variation 10
- MOSI: Pin 11 is used as an Out Slave In grab.
- MISO: Pin 12 is utilised as a master in a slave out configuration.
- SCK: A serial clock is utilised with pin range 13
- LED Pin: The board uses digital pin 13 to power an integrated LED. The digital pin should be high for the LED to light optimally.
- AREF Pin: This is the Arduino board's analogue reference pin. It is frequently employed to provide a reference voltage from an external energy source.

3.4.2Working of Arduino uno:

With the help of the communication ports and input/output ports on the Arduino microcontroller, we may connect several types of peripherals to the board. The microcontroller may receive the data from the peripherals you connect so that it may process the data that comes via them more quickly.

On the other hand, Arduino gives us software like an integrated development environment (IDE) that uses the Arduino programming language, the means of transferring firmware to the microcontroller, and a bootloader that is implemented on the board. The simplicity and usability of the software programme and programming language are its key features.

Arduino promises to be an easy way for everyone to carry out interactive projects. The process for someone who wants to complete a project is to download and set up the IDE, conduct some internet research, and genuinely "clip and paste" the code that amuses us into our HW. After the peripherals are wired appropriately, the programme is already communicating with the hardware. For just the price of the Arduino and its accessories, all of this is possible.

As Arduino is a project rather than a single type of board, you can find a variety of forums with their own unique aesthetics that share its straightforward design. For the demands of the project, you're working on, they come in a variety of shapes, sizes, and colours. Easy or advanced features, Arduinos focused on the internet of things or threeD printing, and, of course, varying price ranges depending on those characteristics.

3.4.3 History of Arduino UNO:

The Interplay Design Institute (IDII), located in Ivrea, Italy, was where the Arduino project first began. At the time, the children were utilising a $50 basic Stamp microcontroller. During his master's thesis research at IDII in 2003, Hernando Barragán created the improvement platform Wiring with the help of Massimo Bansi and Casey Reas. Casey Reas and Ben Fry collaborated to design the Processing development environment. The project's new goal was to create straightforward, reasonably priced tools that non-engineers could use to build virtual projects. The Wiring platform contained a Processing-only integrated development environment (IDE), software libraries for the microcontroller, and an ATmega128 microcontroller on a printed circuit board (PCB). Together with David Mellis, Massimo Banzi enhanced Wiring in 2005 by including instructions for the less expensive ATmega8 microcontroller, David Cuartielles and any further IDII students. The new task that separated from Wiring was called Arduino. The original Arduino centre team included Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis. The open-supply community was given access to lighter, much less expensive alternatives once the platform was completed. Over 300,000 official Arduinos were expected to be produced commercially by the middle of 2011, and 700,000 reputable forums were expected to be in users' hands by 2013.

3.4.4 Technical Specifications of Arduino UNO:

- Atmel ATmega328P microcontroller.
- The working voltage is 5 volts.
- 7–20 volts as the input voltage.
- There are 14 digital I/O pins, 6 of which may output PWM.
- UART: 1
- DC Current per I/O Pin: 20 mA
- I2C: 1
- SPPI: 1
- Analogue Input Pins: 6
- The 3.3V pin's DC current is 50 mA.
- Flash Memory: 32 KB, of which the bootloader uses 0.5 KB
- SRAM: 2 KB
- EEPROM: 1 KB
- Clock Speed: 16 MHz
- Measures 68.6 mm in length
- 53.4 mm in width

CHAPTER 4
SIGN LANGUAGE DESCRIPTIPON

The majority of sign language users for physical communication are also deaf. Since sign language is not standardised, deaf individuals all over the world use different sign languages to communicate. The movements or symbols are grouped linguistically in sign language. A signal is referred to as every single gesture. Wherever there are deaf groups, signal languages have evolved as practical ways of communication and have influenced the fundamentals of local Deaf cultures. Although the deaf and hard of hearing use signing more frequently than hearing people, hearing people who are unable to physically communicate, have trouble utilising oral language due to a condition or handicap, or have deaf family members, including children of deaf parents, also use signing. Rarely is the precise number of signal languages spoken worldwide known. However, some of us of a have more than one. Most of us of a have our own native sign language. The 2021 version of Ethnologue only lists 150 sign languages, compared to the signal-HUB Atlas of sign Language systems, which lists over 200 sign languages and highlights that there are still further sign languages that have not been found or documented. The Indo sign language is the most extensively used sign language in the entire world, according to Ethnologue, which ranks it as the 151st most "spoken" language in the world as of 2021. In prisons, several sign languages have become somewhat more common.

4.1 DIFFERENT SIGN LANGUAGES:

Sign Language Alphabets from Around the World:

- American Sign Language (ASL)
- British, Australian and New Zealand Sign Language (BANZSL)
- Chinese Sign Language (CSL)
- French Sign Language (LSF)
- Japanese Sign Language (JSL) Syllabary
- Arabic Sign Language
- Spanish Sign Language (LSE)
- Mexican Sign Language (LSM)

American Sign Language(ASL) :

Despite using the same alphabet, ASL is not a subset of English. American sign language was independently created and has a distinctive linguistic makeup. It actually derives directly from the first forms of French sign language. Additionally, symptoms no longer follow the same word order as English. It is a result of the specific grammar and visual structure of sign language. Over 500,000 people use American Sign Language globally. At the American School for the Deaf (ASD) in West Hartford, Connecticut, in the early 19th century, a condition of language contact led to the formation of ASL. Since that time, ASL usage has been vigorously promoted by deaf organisations and institutions. Despite being widely used, no precise count of has been done

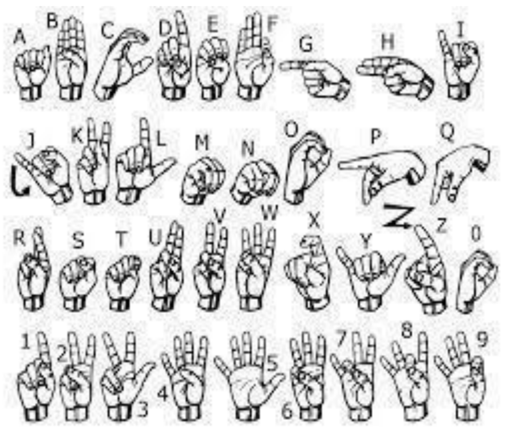


Fig. 4.1: American Sign Language(ASL)

British, Australian and New Zealand Sign Language (BANZSL):

All three sign languages share the same alphabet, which is used by British sign language, Australian sign language (Auslan), and New Zealand sign language. In contrast to ASL, these alphabets use two arms rather than one. These three languages could be thought of as dialects of a single language (BANZSL) because they share a common alphabet, have a similar syntax , and have a large amount of lexical overlap. The words "BANZSL" were created by Adam Schembri and Trevor Johnston.

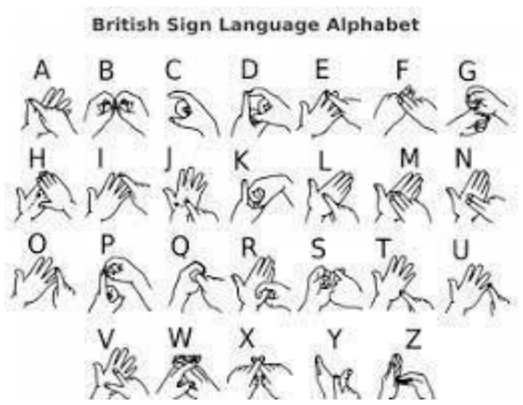


Fig. 4.2**:** BANZ Sign Language

Chinese Sign Language (CSL):

Chinese characters are written using a single basic alphabet, and the signs used in Chinese Sign Language (CSL) are visual representations of those letters. Despite the fact that there are additional CSL dialects, the Shanghai dialect is the most common. The Chinese National Association of the Deaf is working hard to promote language use across the United States of America and raise awareness of it. Since the late 1950s, the language has evolved**.**

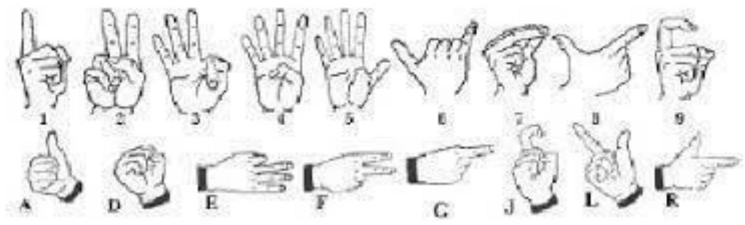


Fig. 4.3: Chinese Sign Language (CSL)

French Sign Language (LSF):

The ancestor of ASL, there are some minor differences between French Sign Language and ASL. Additionally, LSF has a fascinating past.

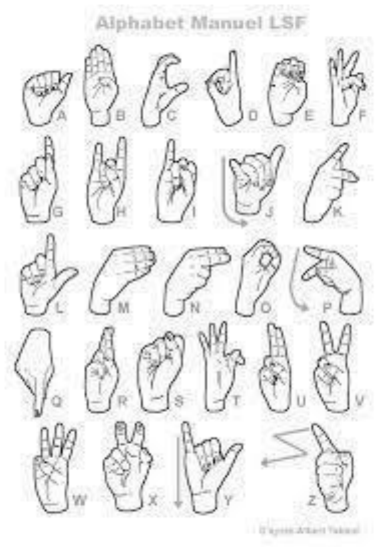


Fig. 4.4**:** French Sign Language (LSF)

Japanese Sign Language (JSL) Syllabary:

The Japanese alphabet, which is made up of phonetic syllables, serves as the foundation for the Syllabary of Japanese Sign Language (JSL).

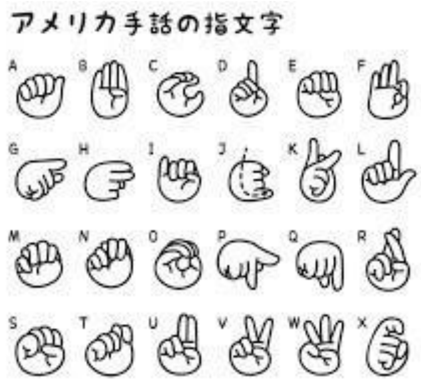


Fig. 4.5: Japanese Sign Language (JSL)

Arabic Sign Language:

A group of sign languages from the Arab Middle form the Arab sign-language own family. There are very few facts about those languages, however some have been noted, such as Levantine Arabic sign language.

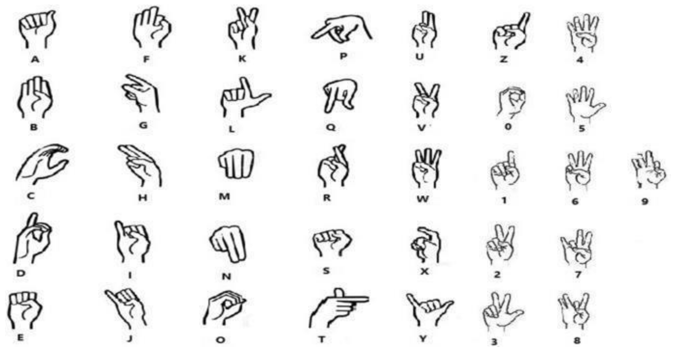


Fig. 4.6: Arabic Sign Language

Spanish Sign Language (LSE):

Spanish is the official Signal Language according to the Spanish government. It is far closer to Spain than any other region outside Catalonia and Valencia. Spanish sign language is no longer used in many Spanish-speaking countries! (See below for an example of Mexican Sign Language.) Particularly in Spain, where there are allegedly 100,000 SSL signers, SSL is used. SSL is entirely distinct from ASL, much like how English is different from Spanish. SSL is used throughout all of Spain, with the exception of Catalonia and Valencia, which employ the Catalan and Valencian signal languages, respectively.

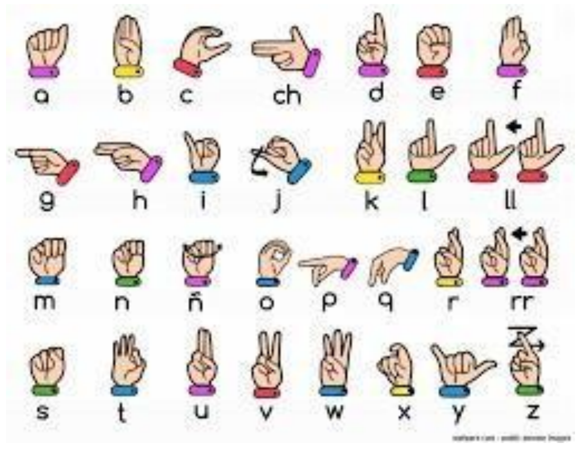


Fig. 4.7: Spanish Sign Language (LSE)

Mexican Sign Language (LSM):

Mexican sign language (also known as LSM) differs from Spanish in that it uses unique verb tenses and sentence structures. The majority of users of Mexican sign language reside in Monterrey, Guadalajara, and Mexico City. The use of this terminology is prevalent among people of all ages and religious backgrounds.

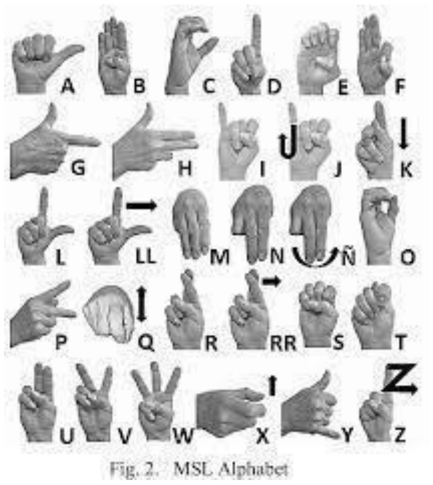


Fig. 4.8**:** Mexican Sign Language

4.2 STANDARD SIGNS:

There is no common signalling language. Unique countries or areas use distinctive sign languages. Those who are familiar with ASL may not be able to understand British sign language (BSL), which is a distinct language from ASL. ASL is used in several foreign countries to perform sign language functions.

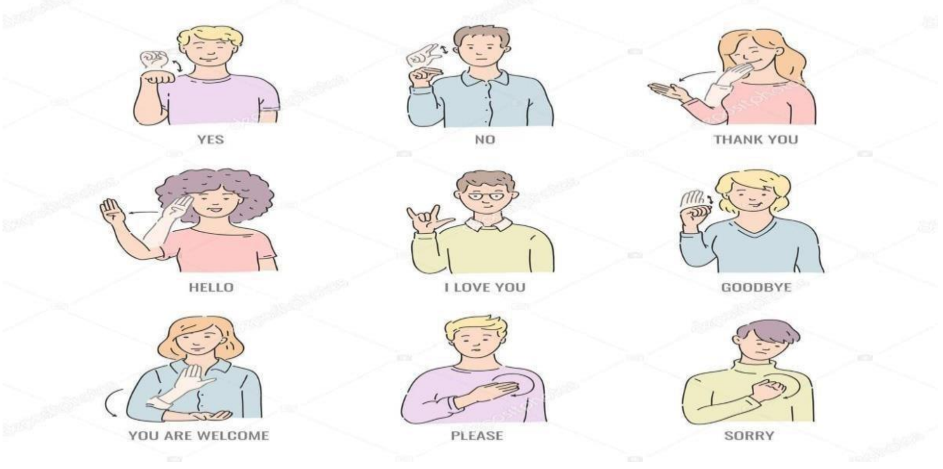


Fig. 4.9**:** Standard Signs

4.3 Advantages of Sign Languages:

1. It lessens irritation.
2. It boosts confidence.
3. It improves listening skills and language abilities.
4. It improves connections.
5. It offers a glimpse into the life of your toddler.

Their Intellect rises as a result.

4.4 Signs we used:

GESTURES:

RESISTANCE VALUES

Table 4.1: Yes values

Yes

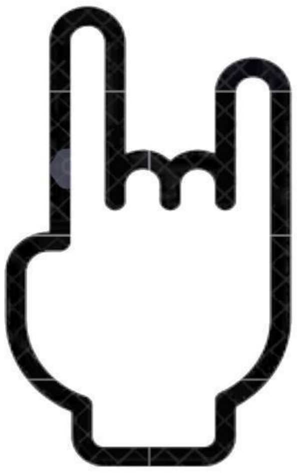
	Set 1	Set 2	Set 3	Set 4
Flex1	832	829	833	831
Flex2	833	839	840	838
Flex3	824	829	828	829
Flex4	986	990	990	986
Flex5	765	766	766	769
X	29	26	31	26
Y	195	190	195	190
Z	121	126	144	136

Table 4.2: No values

	Set 1	Set 2	Set 3	Set 4
Flex1	850	850	847	850
Flex2	838	837	840	838
Flex3	824	826	824	822
Flex4	986	989	986	986
Flex5	765	765	767	769
X	219	246	236	228
Y	110	101	120	126
Z	215	175	192	205

No

Table 4.3: Thank you values



Thank You

	Set 1	Set 2	Set 3	Set 4
Flex 1	828	833	831	829
Flex 2	863	868	871	871
Flex 3	833	831	826	823
Flex 4	933	990	928	929
Flex 5	767	766	768	771
X	139	144	148	146
Y	244	220	233	223
Z	186	204	186	201

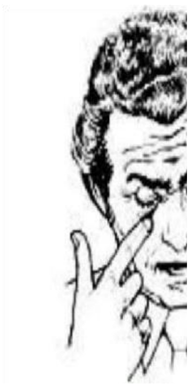
Table 4.4: Stop values

	Set 1	Set 2	Set 3	Set 4
Flex1	828	833	831	829
Flex2	863	868	871	871
Flex3	833	831	826	823
Flex4	933	990	928	929

Stop

Flex5	767	766	768	771
X	139	144	148	146
Y	244	220	233	223
Z	186	204	186	201

Table 4.5: Pain values



Pain

	Set 1	Set 2	Set 3	Set 4
Flex 1	858	845	847	861
Flex 2	893	889	876	882
Flex 3	823	817	815	818
Flex 4	929	985	925	925
Flex 5	769	767	766	767
X	114	122	134	139
Y	221	205	212	206
Z	203	222	215	221

CHAPTER 5
ARDUINO PROGRAMMING

Arduino is a free and open-source hardware and software prototyping platform. It comprises of a circuit board with Arduino ready-made software and a programmable microcontroller.

Using the IDE (integrated development environment), computer code can be added to the physical board. The widely used form factor made available by Arduino compresses the functions of the microcontroller into a more user-friendly small.Key characteristics include:

•Arduino forums can read analogue or virtual input warnings from special sensors and convert them to an output. They entail turning on a motor, turning on and off LEDs, connecting to the cloud, and carrying out 65 different tasks.

• The board's microcontroller can be hard coded with commands using the Arduino IDE to manage its operations.

5.1 ARDUINO BOARD TYPES:

A list of available Arduino boards is provided below:

Different varieties of Arduino boards are available depending on the specific microcontrollers used. All of the forums are, however, programmed using the Arduino IDE, which is a regular issue. The primary differences are to the quantity of inputs and outputs (the number of sensors, LEDs, and buttons you can use on a single board), velocity, working voltage, form factor, and a variety of other aspects. You might wish to purchase certain forums one at a time because they are designed to be embedded and don't have a hardware programming interface. While some can function right away with a 3.7V battery, others need at least 5V. Numerous Arduino models, including the Arduino UNO, Arduino Nano, and Arduino lilypad.

Table 5.1 Arduino boards based on ATMEGA32u4 micro controller

Board name	**Operating voltage**	**Clock speed**	**Digita l i/o**	**Analog Inputs**	**PWM**	**UAR T**	**Programmin g Interface**
Arduino Leonardo	5V	16MH z	20	12	7	1	Native USB
Pro micro 5V/16MH z	5V	16MH z	14	6	6	1	Native USB
Pro micro 5V/16MH z	3.3V	8MHz	14	6	6	1	Native USB
Lilypad Arduino USB	5V	16MH z	14	6	6	1	Native USB

Table 5.2 Arduino boards based on AT91SAM3X8E micro controller

Board name	**Operating voltage**	**Clock speed**	**Digital i/o**	**Analog Inputs**	**PWM**	**UART**	**Programming Interface**
Arduino Due	3.3V	84MHz	54	12	12	4	USB native

Table 5.3 Arduino boards based on ATMEGA2560 microcontroller

Board name	**Operating voltage**	**Clock speed**	**Digital i/o**	**Analog Inputs**	**PWM**	**UART**	**Programming Interface**
Arduino Mega 2560 R3	5V	16MHz	54	16	14	4	USB via ATMega16U2
Mega Pro 3.3V	3.3V	8MHz	54	16	14	4	FTDI Compatible Header
Mega Pro 5V	5V	16MHz	54	16	14	4	FTDI Compatible Header
Mega Pro Mini 3.3V	3.3V	8MHz	54	16	14	4	FTDI Compatible Header

Table 5.4 Arduino boards based on ATMEGA328 microcontroller

Board name	**Operating voltage**	**Clock speed**	**Digital i/o**	**Analog Inputs**	**PWM**	**UART**	**Programming Interface**
Arduino Uno R3	5V	16MHz	14	6	6	1	USB via ATMega16U2
Arduino Uno R3 SMD	5V	16MHz	14	6	6	1	USB via ATMega16U2
Red Board	5V	16MHz	14	6	6	1	USB via FTDI
Arduino Pro 3.3v/8 MHz	3.3V	8 MHz	14	6	6	1	FTDI Compatible Header
Arduino Pro 5v/16 MHz	5V	16 MHz	14	6	6	1	FTDI Compatible Header

Arduino mini 05	5V	16 MHz	14	8	6	1	FTDI Compatible Header
Arduino Pro mini 3.3v/8 MHz	3.3V	8 MHz	14	8	6	1	FTDI Compatible Header
Arduino Pro mini5v/16 MHz	5V	16 MHz	14	8	6	1	FTDI Compatible Header

We are using ARDUINO UNO ATmega328P board in this project, because it is an open source and easy to learn and use. It is cost effective also.

5.2 ARDUINO UNO DESCRIPTION:

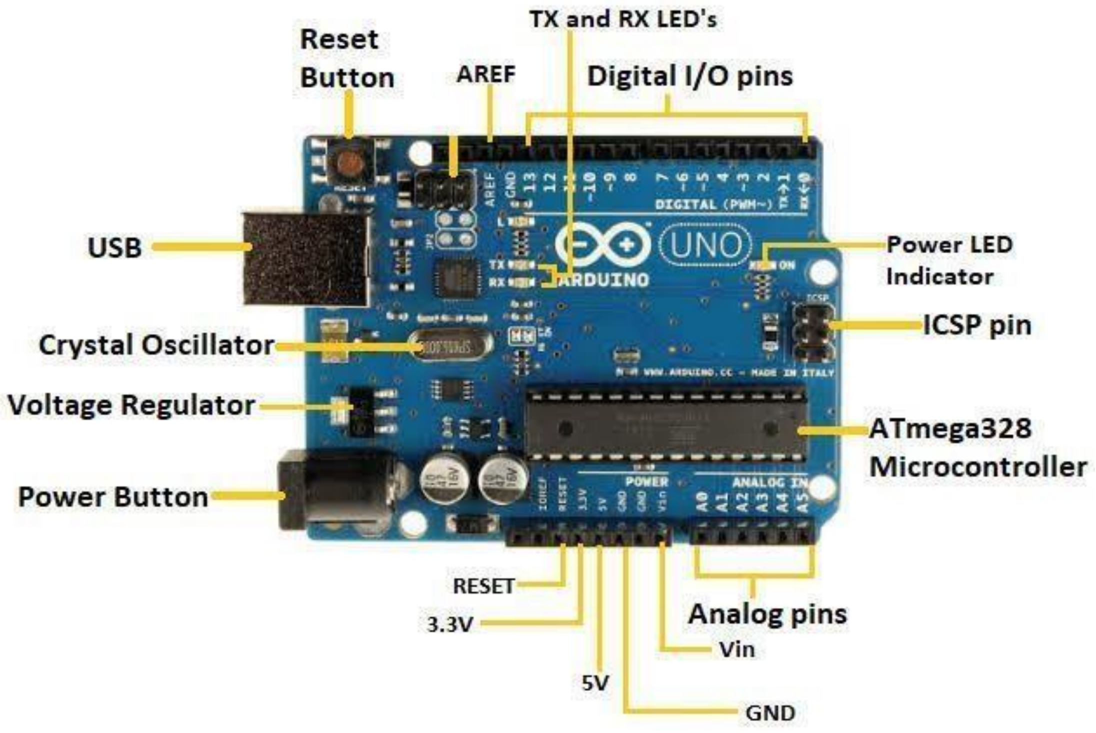


Fig. 5.1: ARDUINO UNO Board

1. Power (Barrel Jack): Arduino boards can be powered directly from the AC mains energy source by connecting to the Barrel Jack.
2. Voltage Regulator: The Arduino board can be powered directly from the AC mains power supply by connecting it to the Barrel Jack.
3. Crystal Oscillator: The crystal oscillator helps Arduino deal with time-related problems. What is the mechanism of the Arduino clock? The solution to this issue is the crystal oscillator. The top of the Arduino crystal shows the broad diversity, which is 16.000H9H. The frequency is stated to be 16 MHz, or 16,000,000 Hertz.
4. Restarting your Arduino board will allow you to start anew with your programme. There are various ways to reset the UNO board. starting with the reset button (17) on the board. To the RESET pin, you can 8 attach a reset button that is located externally to the Arduino.
5. Pins (three, five, GND, Vin): Provide three volts.3.3 output volts .
6. Analogue inputs: The Arduino UNO board has 5 analogue input pins, labelled A0 through A5. These pins can read the signal from an analogue sensor, such as a

temperature or humidity sensor, and convert it into a digital cost that the computer can examine.

7. microcontroller: There is a distinct microcontroller on each Arduino board (eleven). Your board is focused on it, therefore you can rely on it. From board to board, the principal IC (including 53 circuit) of the Arduino differs slightly. The ATMEL company typically manufactures microcontrollers. You need to know what ICs are on your board before installing a new programme from the Arduino IDE. You may find this information above the IC. For further information about the IC creation and its potential, refer to the information page.
8. .ICSP pin: The MOSI, MISO, SCK, RESET, VCC, and GND pins on the Arduino's ICSP (12) programming header are typically AVR pins. SPI (Serial Peripheral Interface) is a term that is occasionally used to refer to it, which is analogous to "expanding" the output.
9. electricity LED indicator: This LED must light up when you plug your Arduino into a power source to show that your board is actually powered on. If this light is no longer working, there is a connectivity issue.
10. LEDs for TX and RX: The letters TX (for transmit) and RX (for receive) can be found on your board. They can be viewed on the Arduino UNO board in particular locations. Let's start by concentrating on digital pins 0 and 1, which are the pins in charge of serial communication. After that, the TX and RX assumed control (13).

The TX led blinks at a specific frequency while transmitting the serial records.

11. Digital I/O: There are 15 virtual I/O pins altogether on the Arduino UNO board, of which 6 are used to produce PWM (Pulse Width Modulation). These pins can be set up to work as virtual input pins for researching good judgements 14 values (0 or

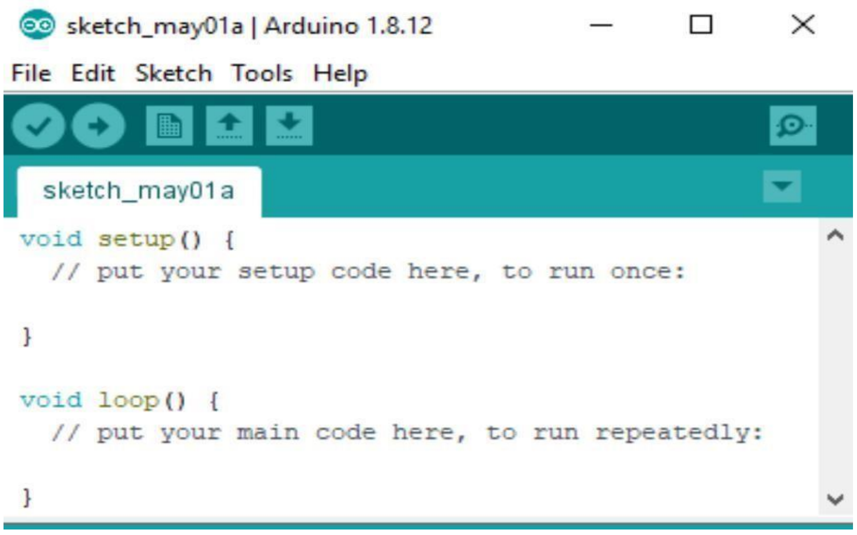
1) or as digital output pins for driving various modules like LEDs, relays, etc. These pins can be used to generate PWM.

12. AREF: The full name of AREF is Analogue Reference. On rare occasions, it is used to adjust the analogue input pins' top limit to an external voltage (between 0 and 5 volts).

5.3 PROGRAMMING:

5.3.1 Basics:

Coding screen: The "setup" and "loop" blocks make up the coding display. While the Loop is regarded as a 1 Execution block, the setup is treated as a coaching block. It is depicted as follows:



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Fig. 5.2: Coding screen

Curly brackets enclose the group of statements in the setup and loop blocks. Depending on the coding requirements of the chosen work, we will write several statements.

installation (): It includes the first task to be completed in the code. The setup process includes the initialization of pin modes, libraries, variables, and many other things. It is often only finished once, during programme uploading, and after the Arduino board has been reset or powered on. setup () is located at the very top of each cartoon. The code enclosed in a bracket is finished in the setup as soon as the programme begins to run, and it only executes once.
Loop (): Statements that are finished repeatedly make up the loop. Depending on the cost of variables, the code segment enclosed in curly brackets is repeated.

Pin Mode (): The unique pin variety is set up as the entrance or OUTPUT under the pin Mode () function. Pin Mode (pin, mode) is the syntax, and
•pin: it's miles the pin wide variety. we are able to choose the pin range consistent with the requirements.

•Mode: Depending on the appropriate pin type, we can adjust the mode to either enter or output.

In the OUTPUT mode of a specific pin number, a sizable quantity of current is supplied to other circuits, sufficient to power an LED or operate a sensor. The output country of a pin is thought to be the low-impedance kingdom.

The Atmel chip may be harmed by a pin's high contemporary and rapid circuit. Setting the mode to OUTPUT is therefore recommended.

Digital Write (): To set a pin's fee as HIGH or LOW, use the digital Write() function.
High: It sets the fee of the voltage. For the SV board, it will set the value of 5V, at the same time as for 3.3V, it will set the cost of three.3V.
LOW: The cost is set to zero (GND).
The LED might also shine dimly if the pin's Mode is no longer set to OUTPUT.
As for the syntax: pin, cost excessive/LOW, digitalWrite.
The high/low price of the digital pin is determined using the digital Write () function, and the high/low price of the virtual pin is analysed using the digital Inspect() functionality. pin: we can specify the pin range or the declared variable. put off (): The put off () feature is a blocking feature that pauses an application from performing a task for the specified amount of time in milliseconds. as an example, - put off (2000) in which, I sec = 1000millisecond consequently, it's going to provide a put off of two seconds.

5.3.2SYNTAX AND PROGRAMME FLOW:

Syntax:

Arduino syntax suggests that particular guidelines must be followed in order for the Arduino application to be successfully uploaded to the board. The syntax of Arduino is the same as the grammar of English. The instructions must be adhered to in order for our code to be assembled and executed properly. Our computer application might still come together and work if we break those principles, but it might do so with a few flaws.

Functions:

- A large number of lines of code are combined into one by the features of an Arduino.
- The features often return a fee after execution is complete. But, because there is void present in this instance, the characteristic does not return any value.
- The void keyword is present in front of the function call for the setup and loop functions.
- Curly brackets are used to denote the multiple lines of code that a characteristic encompasses.
- Each outlet curly bracket in the code must match the corresponding ultimate curly bracket.

- If you would like to discuss this educational later, we may also write about our personal functions. spaces:

White spaces and tabs are ignored by Arduino before any coding statements.

- The coding statements in the code are what the clean analysing is motivated by (see the empty gap at the beginning).
- I motive = 2 spaces within the characteristic definition, loop, and conditional phrases.
- The spaces in parentheses, commas, clean strains, etc. are likewise ignored by the Arduino compiler. Tools Tab:
- The code is only compiled when the verify icon is given at the tool tab. It is a quick technique to determine whether or not the software's syntax is correct.
- We must click the upload button to build, run, and add the code to the board. Uses for the parenthesis symbol (): • It indicates properties similar to those of void setup () and void loop().
- The brackets are used to change the sequence of operations in mathematical operations, and the parameter's inputs to the function are surrounded within them.

  SemiColon(;) : • In both C and C++, it is the declaration terminator.
- An assertion is a directive that tells the Arduino to perform some sort of action. The terminator is therefore essential to indicate the end of an announcement.
- We have the ability to write one or more statements on a single line, but each statement must end with a semicolon.

- If any of the statements don't contain a semicolon, the compiler will indicate an error. • Every declaration should be written with a semicolon on a separate line, as this makes the code easier to read and understand.
- A semicolon is not necessary to follow the setup and loop function's curly braces. Arduino approaches every announcement sequentially. It executes one announcement at a time earlier than moving to the following declaration.

Program Flow:

The Arduino programme flow is comparable to flowcharts. It symbolises the orderly execution of a programme.

The flowchart represents the Arduino coding process is shown below:

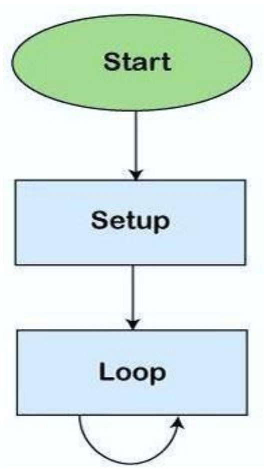


Fig 5.3 : Program Flowchart

5.3.3 Serial Functions:

1.Serial.begin(): The baud rate for serial data connection is specified via the serial.begin() function. The baud rate is a measure of how much data costs in bits per second. The default baud rate for Arduino is 9600 bps (bits per 2d). Additionally, we have the option to define several baud rates, such 4800, 14400, 38400, 28800, and others. The Serial.begin() is declared in two formats, which are shown below:

•begin(speed)

•begin(speed, config) where, serial: It denotes the object

attached to a serial port.

speed: It denotes the bit-per-second (bps) rate or baud rate. It supports extended data types. The stop, parity, and data bits are configured.

2.Serial.print: The facts are printed to the serial port by Arduino's serial.print() function. The broadcast data is saved in the ASCII (American Typical Code for Information Interchange) format, which is a text format that can be read by humans. The use of the ASCII characters is described for each digit of more than a few. The serial display, located in the right corner of the toolbar, allowed users to view the printed records. The Serial.print() is declared in two formats, which are shown below:

•print(value)

•print(value, format)

Serial: It refers to the object on the serial port.

Print: The print () command returns the written bytes in the requested amount.

Value: This word denotes the value to be printed, which can be a value of any data type. For integral data types, the format consists of number bases such as OCT (Octal), BIN (Binary), HEX (Hexadecimal), etc. Moreover, the number of decimal places is specified.

5.3.4 AnalogRead():

- The analogRead() function on a particular Arduino board retrieves a value from an analogue pin.
- The ADC (Analogue to Digital Converter) on the Arduino board is a multichannel converter. It maps the working voltage and entry voltage between 0 and 1023. Options for operating voltage include 3.3V and 5V.
- The integer values range from 0 to 1023. It can also be expressed as 0 to (210)

-1. • One hundred microseconds, or 0.0001 second, is the allotted amount of time for studying an analogue enter indication at the UNO, Mega, Mini, and Nano boards. • The Arduino UNO, Mini, Mega, Nano, Leonardo, and Micro have a 5V operating voltage and a 10 bit resolution. In accordance with the working voltage of 2d.permit and the determination of a few Arduino boards, the maximal reading charge of analogue input is therefore around ten thousand times.

- The working voltage for the Arduino Due, 0 and MKR circle of linked forums is three V, and their resolution is 12 bits.

5.3.5 Arduino Data Types:

In order to learn about record types and aspects linked to maintaining statistics, record types are used. Asserting capabilities and variables determine the bit sample and storage area. The many statistics types that the Arduino supports are listed in the table below: void Data Type

•int Data Type

•Char Data Type

•Float Data Type

•Double Data Type

•Unsigned int Data Type

•Short Data Type

•Long Data Type

•Unsigned long Data Type

•Byte Data Type

•Word Data Type

5.3.6 Arduino Variables:

The place where values and data are stored determines the variables. It has a cost, a sort, and a call. Any statistical type, such as int, float, char, and others, may be used for the variables. Consider the website that sorts information for specified data using Arduino. Ex: int pin=8

Here, a variable called pin that stores the fee eight is made using the int statistics type. Additionally, it indicates that the variable pin has been initialised with the value eight. The variable's name can be changed to suit our needs.

5.3.7 Arduino Operators:

All levels of Arduino programming, from beginner to advanced, primarily rely on the operators. It is the foundation of every programming concept, including C, C++, Java, and many more. Use the operators to resolve logical and mathematical problems. As an example, rely on some analogue voltage to determine the temperature provided by the sensor.

The following are the many groups of operators in Arduino:

1.Arithmetic Operators

2.Boolean Operators

3.Comparison Operators

4.Bitwise Operators

1. Arithmetic Operators: The six principal operators that cause mathematical operations to emerge in Arduino are listed below.

- Assignment Operator (=): The assignment operator on Arduino is used to change a variable's value. It is very different from the equal symbol (=), which is frequently used in mathematics.
- Addition (+): The addition operator is used to combine two numbers. P+Q serves as an example.
- Subtraction (-): Separating two prices using subtraction is a mathematical process. P-Q, for example.
- Multiplication (*): Multiplication is the process of multiplying two numbers. P* Q, for example.

Division (/): The result of multiplying two ranges by one another is determined by the department.

Take P/Q as an example.

- Modulo (%): After dividing a range by one, the remainder is calculated using the modulo operators.

2. Boolean Operators:

Not (!), Logical AND (& &), and Logical OR are the three Boolean operators (eleven). the following operators in more detail:

- And (&&) logic: The conclusion of the situation is true if all of the operands in the condition are true.
- Logical OR (11): The condition's outcome is true if both of the situation's variables are true.
- not (!): it is currently employed to reverse the logical order of the operands.

3. Comparision Operators:

Using comparison operators, the cost of one variable is contrasted with the cost of the opposing variable.
The assessment operators are included in the following index:
Reduced from () The less than operator confirms that the value of the left operand is less than the value of the right operand. The claim is accurate if the condition is satisfied.

- More than (>):The less than operator checks to see if the left side of a declaration costs more than the right side. The announcement is true if everything is happy. • equivalent to (==): Scanned with a carbon scanner The values of two operands are assessed. The values must be equal for the condition to be true.
- Not the same as (!=) It assesses the cost of many variables. The criterion might be correct and gratifying if the data don't match.
- Less than or equal to (=): This operator evaluates a statement to see if the value on the left is less than or equal to the value on the right. The statement is true if one of the two conditions is true.
- Greater than or equal to (>=): This operator determines if the value on the left side of a statement is greater than or equal to the value on the right side. The statement is accurate if everything is joyful.

4.Bitwise Operators:
The Bitwise operators operate in the binary stage. Utilising these operations is rather easy.
There are numerous types of bitwise operators. Some of the well-known operators are listed below:

- Bitwise not: Bit reversal is complemented by the bitwise not operator.
- Bitwise XOR (): If both inputs are the same, the result is 0; otherwise, it is 1, indicating that the two input bits are different.
- Bitwise OR (1): If all of the inputs to the OR operation are zero, the result is 0. The two input styles both have four bits, therefore in any other scenario the output is 1. • Bitwise AND (&): If all of the inputs to the AND operation are 1, the result is 1. Otherwise, there is no output. The two input formats each use four bits.
- Bitwise left shift (); The number of bits specified by the right operator is used to shift the left operator.
- Bitwise right shift (>>): The number of bits specified by the left operator is used to shift the proper operator.

5.3.8 Arduino IF statement:
1.The if ():
The cornerstone of all computer languages is the conditional statement, also known as the "if()" announcement. If the circumstance in the code is actual, this is how the linked task or function gets finished. The software returns a single cost if the conditions are true. It also returns any other value if the scenario is false. The if ()

statement checks the situation in this way before executing a statement or group of statements.

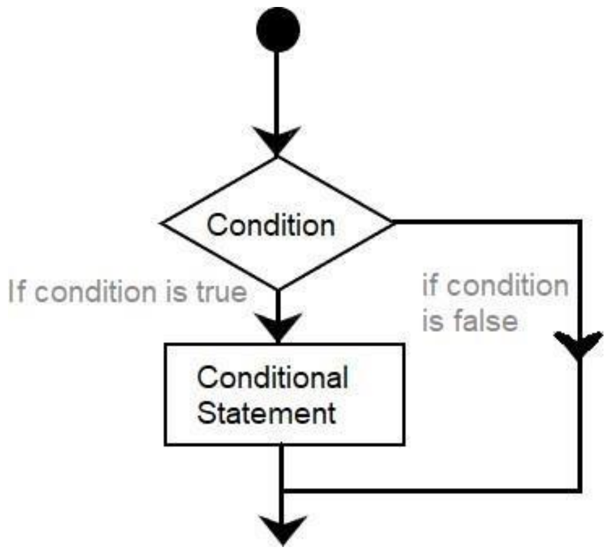


Fig 5.4: Flow chart of IF Statement

2.Arduino If-Else:

The if-else statement and the else statement make up the if-else situation. If the outcome of the If () announcement is false, the condition in the else statement is satisfied.

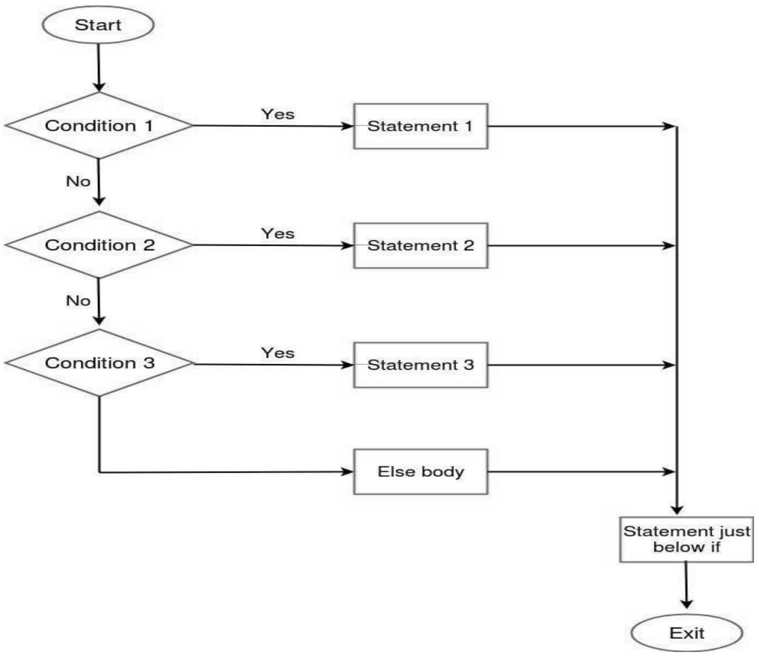


Fig. 5.5: flow chart of If-Else

Until the genuine claim can be observed, each of the statements will be completed. The code will run the relevant blocks of code while going through all of the if and else lines when the appropriate announcement is made.
3.Else-if:
The else () announcement can be used either with or without the else if claim. There will be a lot more if statements in a programme. The gliding will come to an end when the else if () phrase is correctly executed.

5.4 ARDUINO SENSORS:

A sensor is a device, element, or tool that detects changes in the environment. The sensors convert these alterations into a signal, which is then sent to the electronic apparatus.
Sensors and electronic devices are continually interacting. The output signal is simple for users to understand. Sensors are a common component of modern living. lowering the light by, say, lightly touching the base.
The use of sensors is growing as a result of contemporary technologies. The sensors are used to monitor a variety of physical properties, including pressure, temperature, sound, humidity, and light.
Since the Hearth Alarm has a heat- or smoke-detection device, it is a form of sensor. The signal that the detector emits is captured by the alerting device.
The list of several sensor types in Arduino is provided below:

- lighting sensor: The light sensor is used to change the lighting. It is currently used in Arduino in conjunction with the LDR (light established Resistor).
- Ultrasonic sensor: SONAR determines an object's distance using an ultrasonic sensor.
- Temperature sensor: This device measures the temperature of the surrounding air.
- Knock Sensor: The knock sensor is used to choose the vibrations of the banging. It belongs to a class of vibration sensor that is common.
- One object-detection sensor: It is used to detect the item by using infrared radiations that are emitted and returned by the object.

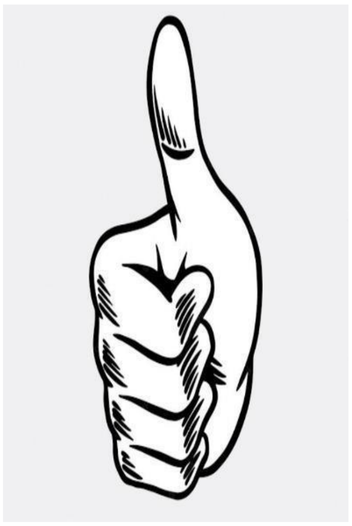
CHAPTER 6
RESULTS AND DISCUSSIONS

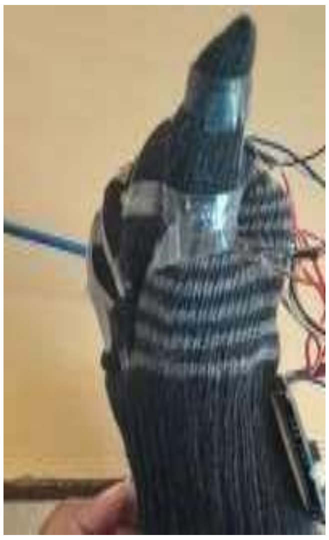
6.1 RESULTS:

All of the signs and often used phrases were protected inside the dataset. The gadget turned into skilled and a version become constructed the use of the ones datasets. Then, so one can educate the device with less version and to make sure that the version stays fashionable and overfits much less, the datasets for all of the signs were pooled and shuffled. the closest cost expected turned into played and shown after the skilled pc had processed the input for the phrase's signal language. The correctness of the version was then anticipated by means of comparing the correlation plot of each signal with the other signal to decide how closely associated the 2 symptoms were to one another. In phrases of signs and symptoms, our version's accuracy changed into observed to be 85% for signs and symptoms because the facts have been distinctly correlated with every different because of that some of the expected output have been not accurate.

In keeping with the quantity of facts used to teach the machine, a excessive degree of accuracy for gesture reputation turned into attained signs.. Close to the discovered dataset, system learning became used to anticipate the gesture. The mission of smart gloves for communities of the deaf and the dumb has been covered on this essay. Their pleasant of life may be progressed with the aid of this machine. The closeness of the dataset of numerous signals and phrases to one another is a chief flaw on this version. This could were prevented with the aid of increasing the data series. Not withstanding the problems described, the advanced glove may help to a certain volume in closing the conversation gap between the deaf and dumb and the general population.

Jumper wires are used to attach every of the 5 sensors to the Arduino Uno board. After the connections are wonderful, Arduino gets input from 3 sensors (Flex sensor, accelerometer). Flex sensors are connected to the palms and display how some distance the hands bend in response to a glove gesture. inside the palm, an accelerometer is placed to take X, Y, and Z axes measurements of the hand's position. at the beginning, simply flex sensors are carried out on this signal language transition. Some hand gestures resemble other gestures in many ways. An second sensor called an accelerometer is also used to distinguish between these different gesture types. When two signals have the same bend in the hands but different bends in the palm, this is crucial in separating them.The designed circuit has been linked to and tested using a variety of hand gestures, and for all of the gestures, the voice was audible, as seen in the following examples:





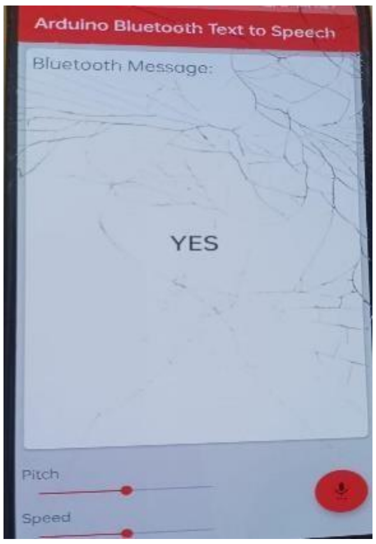
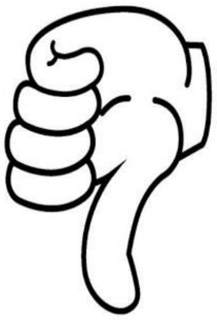
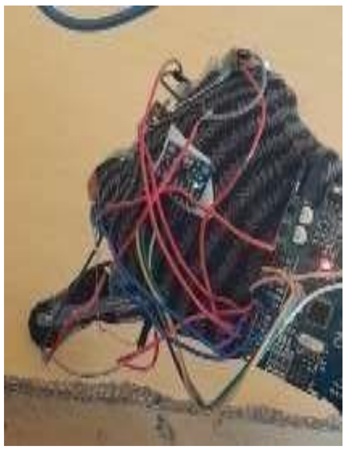


Figure 6.1: yes gesture

Figure 6.2: Output for yes gesture





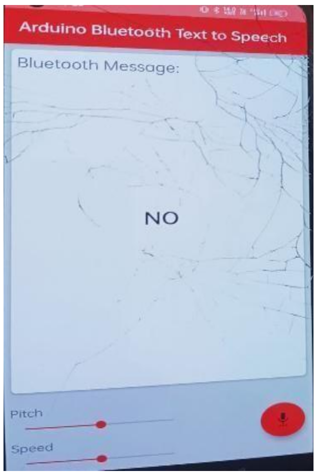
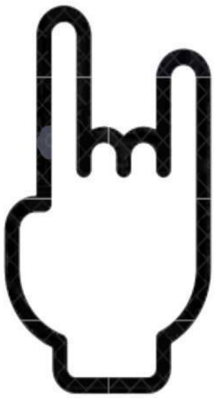
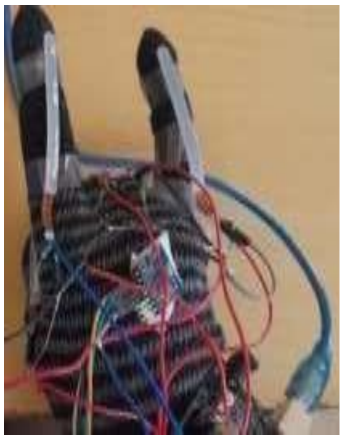


Figure 6.3: No gesture

Figure 6.4: Output for No gesture





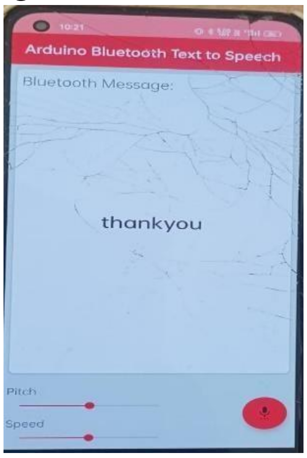


Figure 6.5: Thank you gesture

Figure 6.6: Output for Thank you

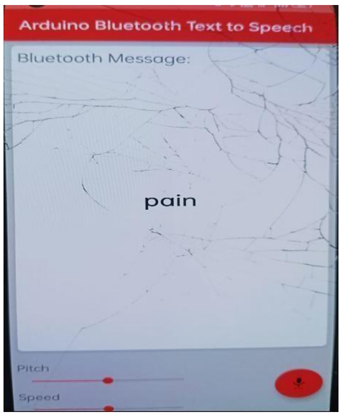


Figure 6.7: Pain gesture

Figure 6.8: Output for Pain

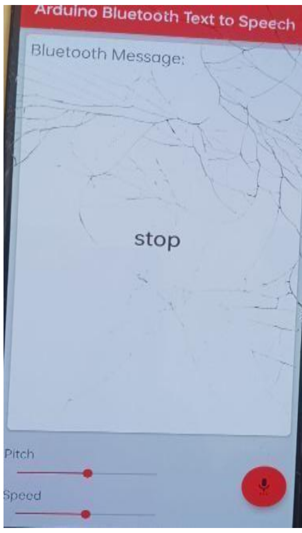


Figure 6.9: Stop gesture

Figure 6.10: output for Stop

6.2 DISCUSSIONS:

The principle component of the popularity gadget is the gesture supervisor. It has information to compare to incoming statistics. The machine makes an effort to align incoming information with the cutting-edge posture. the space to the contemporary data is determined the usage of the finger bend values and every posture specification.

A consumer can interact with others more without problems and without problems due to the fact to the benefit of simple flex sensors. This allows the person to interact no longer best with their network but additionally with others and feature a ordinary lifestyles. The finished product will function a low-fee, trustworthy design that makes it simple for customers to engage with. The device can pick out warning signs extra quicker now. real-time reputation is another ratio of almost ninety-nine% can be effortlessly finished.

With the assist of the clever gloves, we were capable of understand gestures with a excessive diploma of accuracy (ninety six%). The person can define his personal symbols according along with his comfort way to the system learning approach utilised. the usage of the advantages of each analogue and digital sensors allows short machine debugging. The gadget's production fee was reduced by using inexpensive sensors and simple digital additives to create the gloves. A huge gain to society and people with disabilities ought to come from such low-priced but efficient answers. additionally, we have been able to signify some upgrades to the gloves so that they may be utilised as a dressmaker's device based on our know-how of the findings to this point.

CHAPTER 7
CONCLUSION

In this challenge, we created a smart Glove to assist individuals who are dumb or deaf and face issues in speaking with regular humans. An Android smartphone can be connected to the smart-Glove to allow message sharing. Arduino Bluetooth app is used to ship and acquire messages, this app related via Bluetooth module. The cleverGlove is risk-loose, transportable, reasonably priced, and smooth to use. We believe the initiative is a hit and extraordinarily wi-fi wireless for deaf-dumb people if this signs are taught to them so that it will communicate with their households and people round them. machine can interact in wi-fi of methods, including via , which enables a faster connection. The maximum not unusual cellular era for sending mobile facts offerings is GSM, which is also the most extensively utilised. The device can be increased inside the assignment's future to handle extra signs, and its obvious wi-fit is that it's miles used extensively over the arena.

REFERENCES

- International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 3, March 2016 Copyright to IJARCCE DOI 10.17148/IJARCCE.2016.5389 369 Gesture Based Vocalizer for Deaf and Dumb Supriya Shevate1 , Nikita Chorage2 , Siddhee Walunj3, Moresh M. Mukhedkar4 Electronic and Telecommunication, Dr.D.Y.Patil College Of Engineering Ambi, Talegaon Dabhade, Pune, India1,2,3,4

- Smart Glove Based Gesture Vocalizer for Deaf and Dumb Karibasappa R and Choodarathnakara A L Government Engineering College, Kushalnagar, Karnataka, INDIA Ishana M E, Thanuja C, Basavaraju M K and Sunitha C Government Engineering College, Kushalnagar, Karnataka, INDIA

- INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY GESTURE VOCALIZER FOR DEAF AND
  DUMB Kshirasagar Snehal P.*, Shaikh Mohammad Hussain, Malge Swati S., Gholap Shraddha S., Mr. Swapnil Tambatkar, Asst. Prof. Department of Electronics And Telecommunication, Shivaji University, Kolhapur, India
- Manandhar, Sanish, Sushana Bajracharya, Sanjeev Karki, and Ashish Kumar Jha. "Hand Gesture Vocalizer for Dumb and Deaf People." SCITECH Nepal 14, no. 1 (2019): 22-29.

- Mali Pooja Dadaram, Gosavi Deepali Balu, Sonawale Rutuja Ramesh, Prof. S.N.Wangikar.” Sign Language to Speech Conversion Gloves using Arduino and Flex Sensors”. International Research Journal of Engineering and Technology (IRJET) Volume: 07 Issue: 04 | Apr 2020