JSS MAHAVIDYAPEETA

JSS SCIENCE AND TECHNOLOGY UNIVERSITY

SRI JAYACHAMARAJENDRA COLLEGE OF ENGINEERING

MYSORE-570006

SUBJECT NAME: SIGNALS AND SYSTEMS (EC430)
REPORT SUBMITTED AS FULFILLMENT OF EVENT – 2

Submitted by

Sl. No.	USN	NAME
1.	01JST19EC005	AJAY M
2.	01JST19EC045	MD ZAID SALMAN
3.	01JST19EC057	P SUDARSHAN YADAV
4.	01JST19ECO89	SURYA MS

Submitted to

Prof. Anitha S Prasad
Assistant Professor
Department of E & C
JSS S&TU, SJCE, Mysore

DEPARTMENT OF ELECTRONICS AND COMMUNICATION
MYSORE-570006
2020-2021

Table of Contents:

ABSTRACT

This paper presents a sign to speech converter for dumb people. In the present world it is very difficult for the dumb people to talk with the ordinary people. So it becomes impossible for them to communicate with ordinary people unless and until ordinary people like us learn sign language for the purpose of communication. The sign language of dumb is quite difficult to learn and it is not possible for everybody to learn that language. So every person cannot come and share their thoughts with these physically impaired people. So here is a system which would enable the dumb people to communicate with each and every one. In this system, we have used hand gestures as a method of communication for dumb people. Each unique hand gesture is coded to produce a particular audio message through the speaker. We have used flex sensor and accelerometer to capture hand gestures which are sent as inputs to the microcontroller(arduino mega in our project).

Chapter-1:

PROJECT OVERVIEW

Section 1.1: Introduction

Fig 1.1 Sign language

It is very difficult for mute people to convey their message to regular people. Since regular people are not trained in hand sign language, communication becomes very difficult. In emergencies or other times when a mute person travelling or among new people communication with nearby people or conveying a message becomes very difficult. Here we propose a smart speaking system that helps mute people in conveying their message to regular people using hand motions and gestures. The system makes use of a hand motion reading system equipped with motion and flex sensors along with a speaker unit. This system is powered by a battery powered circuitry to run it. An Arduino mega is used for processing the data and operating the system. The system consists of around 10 stored messages like "need help", "where is the toilet/washroom" and so on that help mute people convey basic messages. The system reads persons hand motions for different variations of hand movement. The system uses accelerometer and flex sensors as input sensors. The atmega 2560 processor constantly receives input sensor values and then processes it. Now it searches for matching messages for the set of sensor values. Once it is found in memory this message is retrieved and is spoken out using text to speech processing through the interfaced speaker. Thus we have a fully functional smart speaking system to help mute people communicate with regular people using a simple wearable system.

Section 1.2: Problem statement

Speech is one of the most effective ways to communicate with people and easy exchange of information can take place. But there are people who are deprived of such creation. These people often called as dumb, do use sign language to communicate with people. but the hard part is not everyone knows sign language unless he is also dumb. This makes it difficult for them to communicate with general people. Also in emergency situations where they have to reach out to people from a distance, it is almost impossible to establish communication in such situations.

Section 1.3: Objective of the project

1) To make communication for dumb people more effective using hand gestures.
2) To make the system as compact as possible.
3) To learn the usage of microcontrollers and transducers.
4) To reduce cost(by using arduino) and still make it effective.
5) To make it applicable in real world as much as possible

Section 1.4: Literature survey

Deaf Mute Communication Interpreter- A Review [1] : This paper aims to cover the various prevailing methods of deaf-mute communication interpreter system. The two broad classification of the communication methodologies used by the deaf –mute people are - Wearable Communication Device and Online Learning System. Under Wearable communication method, there are Glove based system, Keypad method and Handicom Touch-screen. All the above mentioned three sub-divided methods make use of various sensors, accelerometer, a suitable micro-controller, a text to speech conversion module, a keypad and a touch-screen. The need for an external device to interpret the message between a deaf –mute and non-deaf-mute people can be overcome by the second method i.e online learning system. The Online Learning System has different methods. The five subdivided methods are- SLIM module, TESSA, Wi-See Technology, SWI_PELE System and Web-Sign Technology. An Efficient Framework for Indian Sign Language Recognition Using Wavelet Transform [2]:The proposed ISLR system is considered as a pattern recognition technique that has two important modules: feature extraction and classification. The joint use of Discrete Wavelet Transform (DWT) based feature extraction and nearest neighbour classifier is used to recognize the sign language. The experimental results show that the proposed hand gesture recognition system achieves maximum 99.23% classification accuracy while using cosine distance classifier.Hand Gesture Recognition Using PCA in [3]: In this paper authors presented a scheme using a databasedriven hand gesture recognition based upon skin color model approach and thresholding approach along with an effective template matching with can be effectively used for human robotics applications and similar other applications.. Initially, hand region is segmented by applying skin color model in YCbCr color space. In the next stage thresholding is applied to separate foreground and background. Finally, template based matching technique is developed using Principal Component Analysis (PCA) for recognition. Hand Gesture Recognition System For Dumb People [4]: Authors presented the static hand gesture recognition system using digital image processing. For hand gesture feature vector SIFT algorithm is used. The SIFT features have been computed at the edges which are invariant to scaling, rotation, addition of noise.

Chapter-2:

SOFTWARES USED FOR IMPLEMENTATION

Section 2.1: Arduino IDE

Fig 2.1 Logo of arduino

The Arduino Integrated Development Environment (IDE) is a cross-platform application that is written in functions from C and C++. It is used to write and upload programs to Arduino boards, but also, with the help of third-party cores, other vendor development boards. It contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino compatible hardware to upload programs and communicate with them.

Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension.ino. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text. output by the Arduino Software (IDE), including complete error messages and other information.

The program for arduino uno used in our project was coded using this software. We have included libraries dedicated for serial communication and audio decoding which are essential for our project.

Section 2.2: Online audio convert

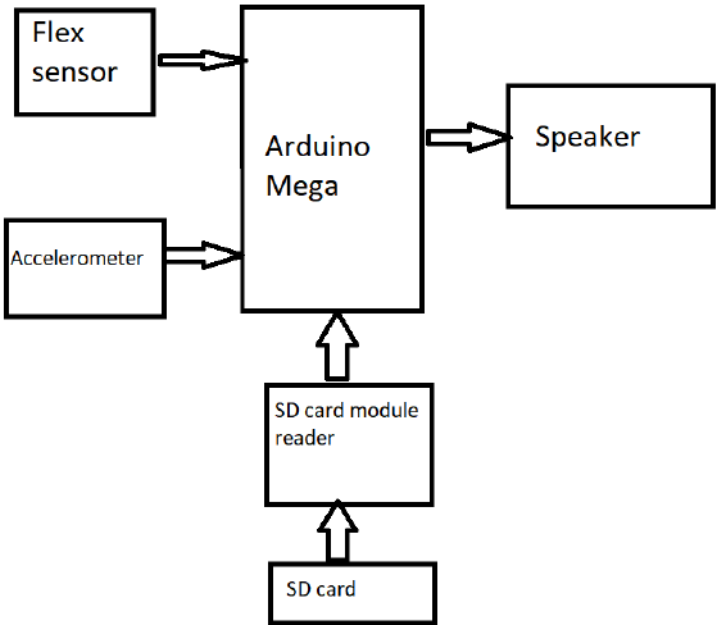
Fig 2.2 Logo of online converter software

Online audio converters can convert music/audio files into other formats. This can be useful to convert music-software or music-devices that only support certain file formats. We can also convert audio from video. Popular conversions are mp4 to mp3, wav to mp3, m4a to mp3, mp3 to wav, flac to mp3, ogg to mp3. We have used this software to convert .mp3 to .wav and also to change the bit resolution to 8-bit, sampling rate to 8000Hz and audio channel to mono.

Chapter-3:

HARDWARE DETAILS:

Section 3.1: Block diagram



Section 3.2: Components used

3.2.1: Arduino mega

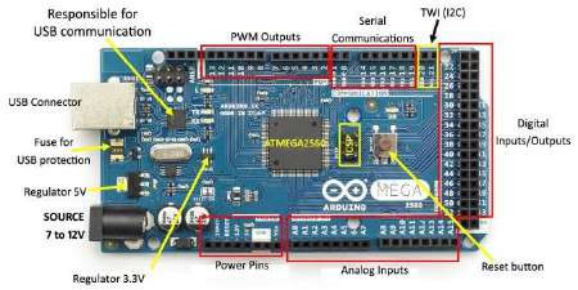


Fig 3.1 Arduino mega board

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal

oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller
Flash - 256k bytes (of which 8k is used for the bootloader)
SRAM - 8k bytes
EEPROM - 4k byte

3.2.2: Flex sensor

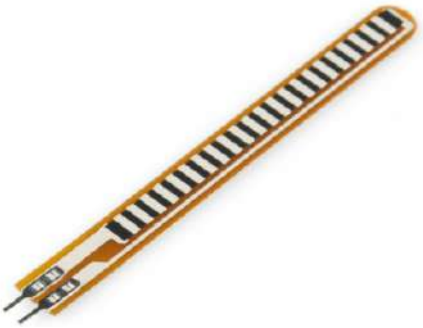


Fig 3.2 Flex sensor

This flex sensor is a variable resistor like no other. The resistance of the flex sensor increases as the body of the component bends. Sensors like these were used in the Nintendo Power Glove. They can also be used as door sensors, robot whisker sensors, or a primary component in creating sentient stuffed animals.

Flex sensors are available in two sizes: one 2.2" (5.588cm) long and another coming in at 4.5" (11.43cm) long. Left flat, these sensors will look like a 30kΩ resistor. As it bends, the resistance between the two terminals will increase to as much as 120kΩ at a 90° angle.

3.2.3: ADXL-345 Accelerometer

Fig 3.3 ADXL-345 Accelerometer

A basic accelerometer consists of multiple axes, two to determine most two-dimensional movement with the option of a third for 3D positioning. Many smartphones and cameras these days use three-axis models, whereas cars basically use two-axis to determine the moment of impact. Accelerometers are sensitive, they even manage to measure even the tiniest of the changes in acceleration. Accelerometer senses static or dynamic forces

of acceleration.i.e static forces such as gravity and dynamic forces such as vibrations and movement.

Nowadays accelerometers can measure multiple axes as many as 3 axes. Basically, an accelerometer consists of internal capacitive plates, some are fixed while others are attached to minuscule springs that move internally as acceleration force acts upon the sensor. As plates will move the acceleration can be determined by changing capacitance.

_____This sensor uses the I2C protocol for communication with the Arduino. Well each device has a preset ID or a unique device address so the master can choose with which devices will be communicating. The two wires, or lines which help in achieving this are called Serial Clock (or SCL) and Serial Data (or SDA). The SCL line is the clock signal which synchronizes the data transfer between the devices on the I2C bus and it's generated by the master device. The other line is the SDA line which carries the data. We need only two wires for connecting it's SCL and SDA pins to the same pins found on arduino mega, plus the two wires for powering it.

3.2.4: Resistors

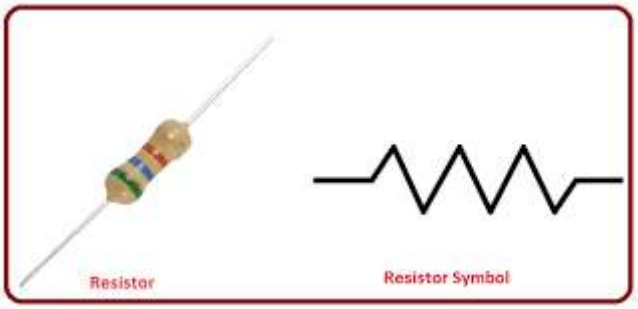


Fig 3.4 Resistor

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses. High-power resistors that can dissipate many watts of electrical power as heat, may be used as part of motor controls, in power distribution systems, or as test loads for generators. Fixed resistors have resistances that only change slightly with temperature, time or operating voltage. Variable resistors can be used to adjust circuit elements (such as a volume control or a lamp dimmer), or as sensing devices for heat, light, humidity, force, or chemical activity.

3.2.5: Speaker

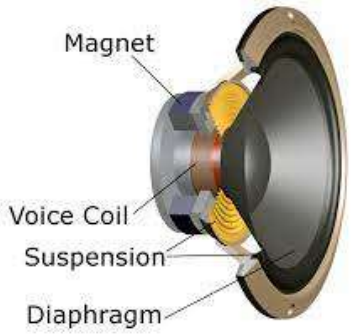


Fig 3.5 Speaker

A loudspeaker is an electroacoustic transducer, a device which converts an electrical audio signal into a corresponding sound. The most widely used type of speaker is the dynamic speaker. The sound source (e.g., a sound recording or a microphone) must be amplified or strengthened with an audio power amplifier before the signal is sent to the speaker.

The most common type of driver, commonly called a dynamic loudspeaker, uses a lightweight diaphragm, or cone, connected to a rigid basket, or frame, via a flexible suspension, commonly called a spider, that constrains a voice coil to move axially through a cylindrical magnetic gap. A protective cap glued in the cone's center prevents dust, especially iron filings, from entering the gap. When an electrical signal is applied to the voice coil, a magnetic field is created by the electric current in the voice coil, making it a variable electromagnet. The coil and the driver's magnetic system interact, generating a mechanical force that causes the coil (and thus, the attached cone) to move back and forth, accelerating and reproducing sound under the control of the applied electrical signal coming from the amplifier.

3.2.6: Micro SD card module reader

Storing data is one of the most important parts of every project. There are several ways to store data according to the data type and size. SD and micro SD cards are one of the most practical ones among the storage devices, which are used in devices such as mobile phones, minicomputers, etc. The SD and micro SD card modules allow you to communicate with the memory card and write or read the information on them. The module interfaces in the SPI protocol.

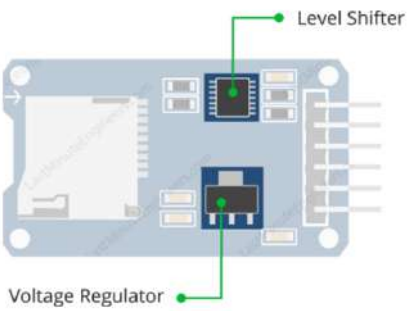


Fig 3.6 Hardware overview

- The operating voltage of any standard micro SD Cards is 3.3 V. So we cannot directly connect it to circuits that use 5V logic. In fact, any voltages exceeding 3.6V will permanently damage the micro SD card. That's why; the module has an onboard ultra-low dropout regulator that will convert voltages from 3.3V – 6V down to ~3.3V.
- There's also a 74LVC125A chip on the module which converts the interface logic from 3.3V-5V to 3.3V. This is called logic level shifting. That means you can use this board to interact with both 3.3V and 5V microcontrollers like Arduino.

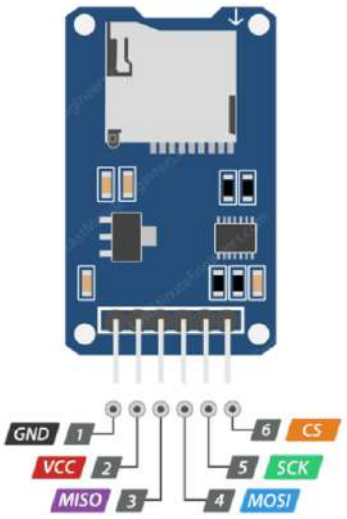


Fig 3.7 Pinout

- The VCC pin supplies power for the module and should be connected to a 5V pin on the Arduino.
- GND should be conn0ected to the ground of the Arduino.
- MISO (Master In Slave Out) is SPI output from the Micro SD Card Module.
- MOSI (Master Out Slave In) is SPI input to the Micro SD Card Module.
- SCK (Serial Clock) pin accepts clock pulses which synchronize data transmission generated by Arduino.

- SS (Slave Select) pin is used by Arduino(Master) to enable and disable specific devices on the SPI bus.

3.2.7: SD card

Fig 3.8 Micro SD card

Pin Number	Pin Name	In SD Mode	In SPI Mode
1	DAT2/X	Connector Data line 2	No use
2	DAT3/CS	Connector Data line 3	Chip Select
3	CMD/DI	Command / Response Line	Data Input
4	VDD/VDD	Power supply (+3.3V)	Power supply (+3.3V)
5	CLK/SCLK	Clock	Serial Clock
6	VSS/VSS	Ground	Ground

7	DAT0/D0	Connector Data line 0	Data Out
8	DAT1/X	Connector Data line 1	No use

Fig 3.9 Pin description

SD Cards are the most commonly used Storage devices in embedded applications. Almost all microcontrollers have a limited Flash memory for programming and a limited EEPROM memory to store important data. However for projects involving Data logging or Pictures or other heavy graphics the programmer might have to save huge piece of data in terms of Mega bytes. In that situation an SD card is employed.

The SD cards can work in two operating modes, one is using the SD mode commands and the other is SPI mode. Most of the Digital cameras and mobile phones will use the SD mode to communicate with the SD card, but only the SPI mode can be used to communicate between an SD card and a Microcontroller like Arduino (ATmel), PIC, AVR etc.. A simple connection diagram for SPI with a microcontroller shown below:

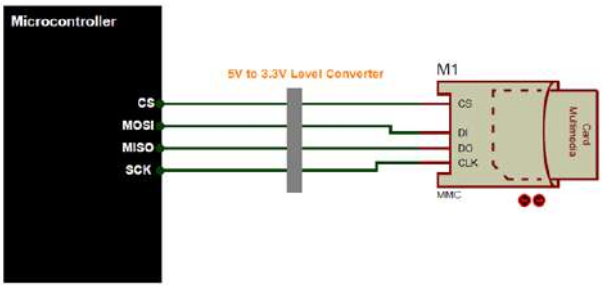


Fig 3.10 Connection of SD card to microcontroller

The SPI communication requires only four wires and is vastly supported by most of the microcontrollers. However the SD card operates with a voltage of 3.3V and all its pins speak with only 3.3V, the Microcontroller on the other hand might work with +5V in those cases a bi-directional logic level shifter that can convert the 5V signals to 3.3V is recommended. Once you have interfaced the Card you would have to initialize it and then start communicating with it through the SPI commands.

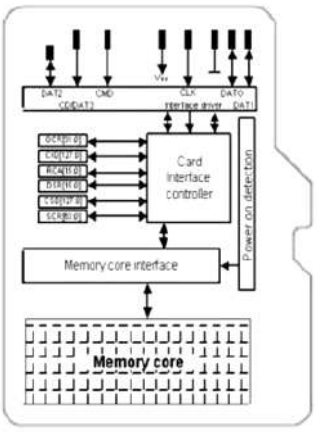


Fig 3.11 Architecture of SD card

3.2.7.1:What is SPI?

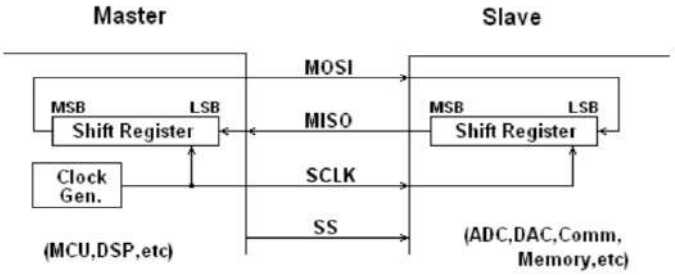


Fig 3.12 SPI communication method

Serial Peripheral Interface (SPI) communication was used to connect devices such as printers, cameras, scanners, etc. to a desktop computer; but it has largely been replaced by USB. SPI is still utilized as a communication means for some applications using displays, memory cards, sensors, etc. SPI runs using a master/slave set-up and can run in full duplex mode (i.e., signals can be transmitted between the master and the slave simultaneously). When multiple slaves are present, SPI requires no addressing to differentiate between these slaves. There is no standard communication protocol for SPI.

SPI is used to control peripheral devices and has some advantages over I2C. Because of its simplicity and generality, it is being incorporated in various peripheral ICs. The number of signals of SPI, three or four wires, is larger than I2C's two wires, but the transfer rate can rise up to 20 Mbps or higher depending on device's ability (5 - 50 times faster than I2C). Therefore, it is often used in applications (ADC, DAC or communication between ICs) that require high data transfer rates.

The SPI communication method is illustrated in fig. The master IC and the slave IC are tied with three signal lines, SCLK (Serial Clock), MISO (Master-In Slave-Out) and MOSI (Master-Out Slave-In). The contents of both 8-bit shift registers are exchanged with the shift clock driven by the master IC. An additional fourth signal, SS (Slave Select), is utilized to synchronize the start of packet or byte boundary and to facilitate working with multiple slave devices simultaneously. Most slave ICs utilize different pin names (e.g., DI, DO, SCK and CS) to the SPI interface.

For one-way transfer devices, such as DAC and single channel ADC, either of data lines may be omitted. The data bits are shifted in MSB first.

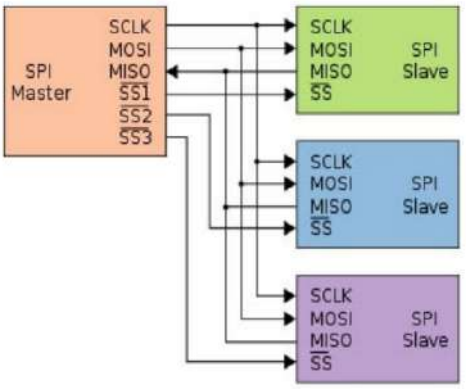


Fig 3.13 Single master multiple slave configuration

When additional slaves are attached to the SPI bus, they are attached in parallel and an individual SS signal must be connected from the master to each of the slaves (as shown in Fig.2). The data output of the slave IC 2 is enabled when the corresponding SS signal is set; the data output is disconnected from the MISO line when the slave device is deselected.

3.2.8: Jumper wires

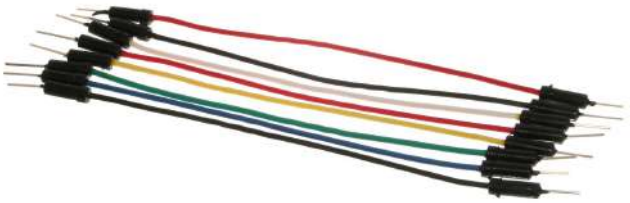


Fig 3.15 Jumper wires

A jump wire is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them), which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally or with other equipment or components, without soldering. Individual jump wires are fitted by inserting their "end connectors" into the slots provided in a breadboard, the header connector of a circuit board, or a piece of test equipment.

3.2.9: Bread board

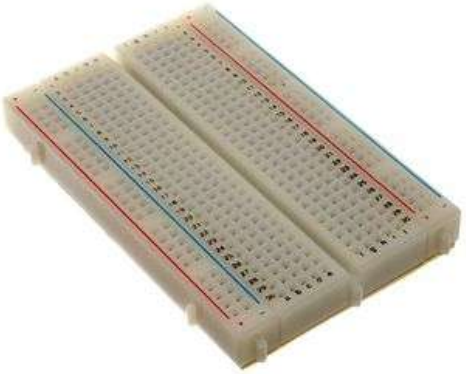


Fig 3.16 Bread board

A breadboard, or protoboard, is a construction base for prototyping of electronics. Because the solderless breadboard does not require soldering, it is reusable. This makes it easy to use for creating temporary prototypes and experimenting with circuit design. For this reason, solderless breadboards are also popular with students and in technological education. Compared to more permanent circuit connection methods, modern breadboards have high parasitic capacitance, relatively high resistance, and less reliable connections, which are subject to jostle and physical degradation. Signaling is limited to about 10 MHz, and not everything works properly even well below that frequency.

Section 3.3: Circuit design and working:

We have used arduino mega 2560 as the heart of the project. Micro SD card reader module is connected to arduino and communicates with it using SPI communication protocol. Audios are preloaded into the micro sd card and fitted to the SD card reader module.

3.3.1 Using flex sensors:

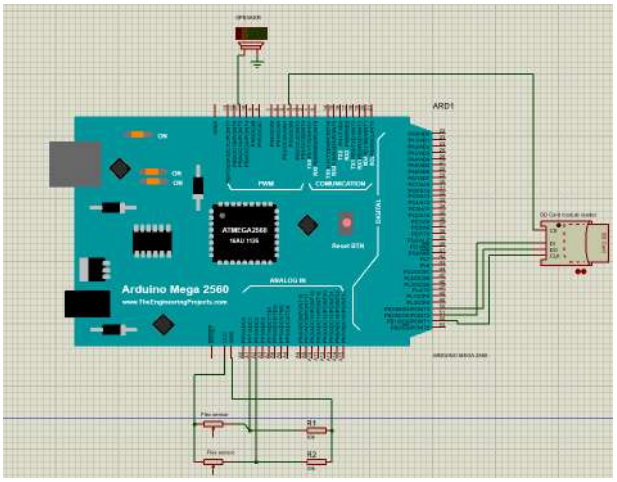


Fig 3.17 Circuit diagram

Two flex sensors are mounted on the first two finger slots of gloves and are connected in series with a resistor and the voltage change in the resistor(using voltage divider rule) is taken as input whenever the flex sensor is bent. Whenever the finger is bent, unique audio is produced.

3.3.2 Using accelerometer.

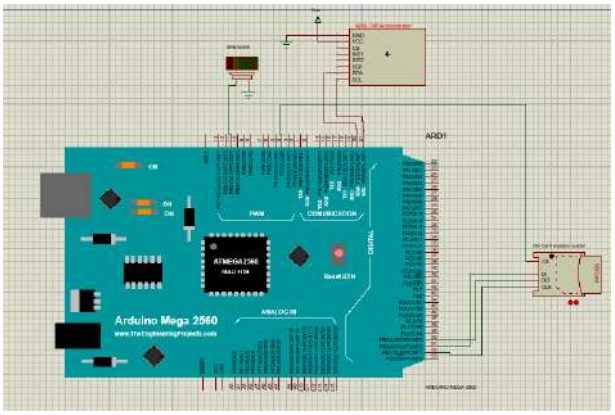


Fig 3.18 Circuit diagram

The ADXL-345 accelerometer is used to record different hand positions. This is achieved by recording acceleration data in 3-axes and sending it to arduino using I2C serial communication protocol. This acceleration data is used to decode a unique audio from the SD card and is played through the speaker.

Chapter-4:

PROGRAMMING DETAILS

Section 4.1: Code

1. **Producing sound using flex sensors:**

```
#include <SD.h>                          //include SD module library
#include <TMRpcm.h>                      //include speaker control library

#define SD_ChipSelectPin 4               //define CS pin
#define flex1 A1
#define flex2 A2

TMRpcm tmrpcm;                           //create an object for speaker library

void setup()
{
  pinMode(flex1, INPUT);
  pinMode(flex2, INPUT);

  tmrpcm.speakerPin = 11;                //define speaker pin.

  if (!SD.begin(SD_ChipSelectPin))       //see if the card is present and can be
                                           //initialized

  {
  return;                          //don't do anything more if not
  }

  tmrpcm.setVolume(7);                   //0 to 7. Set volume level
}


int val1 = 0, val2 = 0;

void loop()
{
  int avg(int);
  val1 = avg(flex1);
  val2 = avg(flex2);

  if((val1>550)&&(val2>370))
  {
```

```
    tmrpcm.play("1.wav");
    delay(3100);
  }

  else if((val1>550)&&(val2<370))
  {
    tmrpcm.play("2.wav");
    delay(3100);
  }

  else if(((val1>460)&&(val1<530))&&(val2>370))
  {
    tmrpcm.play("3.wav");
    delay(3100);
  }

  else if(((val1>460)&&(val1<530))&&(val2<370))
  {
    tmrpcm.play("4.wav");
    delay(4000);
  }

  else if((val1<460)&&(val2>370))
  {
    tmrpcm.play("5.wav");
    delay(3100);
  }

  else
  {
    tmrpcm.play("6.wav")
    delay(3100);
  }
}

int avg(int flex)
{
  long int sum=0;
  for(int i=0;i<200;i++)
  {
    sum=sum+analogRead(flex);
  }
  return (sum/200);
}
```

2. **Producing sound using accelerometer:**

```
#include <Wire.h>                    // Wire library - used for I2C communication
#include <SD.h>                      //include SD module library
#include <TMRpcm.h>                  //include speaker control library

#define SD_ChipSelectPin 4           //define CS pin
int ADXL345 = 0x53;                  // The ADXL345 sensor I2C address
float X_out, Y_out, Z_out;           // Outputs
int del = 4000;

TMRpcm tmrpcm;                       //create an object for speaker library

void setup()
{
  Wire.begin();                                  // Initiate the Wire library
  Wire.beginTransmission(ADXL345);    // Start communicating with the device
  Wire.write(0x2D);                              // Access/ talk to POWER_CTL
                                                    Register - 0x2D
                          // Enable measurement
  Wire.write(8);                                 // (8dec -> 0000 1000 binary) Bit D3
                                                    High for measuring enable
  Wire.endTransmission();
  delay(10);


  if (!SD.begin(SD_ChipSelectPin))
  {                                 //see if the card is present and can be initialized
    return;                           //don't do anything more if not
  }

  tmrpcm.speakerPin = 11;
  tmrpcm.setVolume(7);

}

void loop()
{
  Wire.beginTransmission(ADXL345);
  Wire.write(0x32);                              // Start with register 0x32
                                                    (ACCEL_XOUT_H)
  Wire.endTransmission(false);
  Wire.requestFrom(ADXL345, 6, true); // Read 6 registers total, each axis
                                                    value is stored in 2 registers
  X_out = ( Wire.read()| Wire.read() << 8); // X-axis value
```

```
  Y_out = ( Wire.read()| Wire.read() << 8); // Y-axis value

  Z_out = ( Wire.read()| Wire.read() << 8); // Z-axis value

  if(X_out>150)                //Forward
  {
    tmrpcm.play("1.wav");   //Hello good morning
    delay(del);
  }


  else if(X_out<-150)           //Backward
  {
    tmrpcm.play("2.wav");   //Can you please do me a favour
    delay(del);
  }

    else if(Y_out>150)          //Right
  {
    tmrpcm.play("3.wav");   //Please help me
    delay(del);
  }

    else if(Y_out<-150)           //Left
  {
    tmrpcm.play("4.wav");  //Can you please help me cross the road
    delay(del);
  }
 }
```

Section 4.2: Explanation of libraries and its functions used

1. **Wire.h library:**
   This library allows you to communicate with I2C / TWI devices. The Wire library implementation uses a 32 byte buffer, therefore any communication should be within this limit. Exceeding bytes in a single transmission will just be dropped. We have used the following functions of this library:

   Wire.begin(); The Wire.begin() function will initiate the Wire library.
   Wire.beginTransmission(): In the loop() we will start with the Wire.beginTransmission() function which will begin the transmission to the particular sensor, the 3 Axis Accelerometer in our case.

Wire.write(): Then with the Wire.write() function we will ask for the particular data from the two registers of the X axis, Y axis and Z axis.
Wire.endTransmission(): The Wire.endTransmission() will end the transmission and transmit the data from the registers.
Wire.requestFrom(): the Wire.requestFrom() function will request the transmitted data or the two bytes from the two registers.
Wire.available(): The Wire.available() function will return the number of bytes available for retrieval.
Wire.read(): The Wire.available() function will return the number of bytes available for retrieval and if that number match with our requested bytes, in our case 2 bytes, using the Wire.read() function we will read the bytes from the two registers of the X axis, Y axis and Z axis

2. **SD.h library:** The SD library allows for reading from and writing to SD cards. The library supports FAT16 and FAT32 file systems on standard SD cards and SDHC cards.
   SD_ChipSelectPin: This variable is used to define the chip select pin of the arduino to establish SPI type of serial communication.
   SD.begin(): Initializes the SD library and card. This begins use of the SPI bus (digital pins 50, 51, and 52 on the Mega) and the chip select pin, which defaults to the hardware SS pin.

3. **TMRPCM.h library:** It is a library for asynchronous transmission of PCM/WAV files direct from SD card. It utilises standard arduino SD library, SD card and output device(speaker, headphones, amplifier).
   objectname.speakerPin: This variable is used to set the digital pins of arduino to produce audio.
   objectname.setvolume(0-7): This function is used to set the volume of the output device.
   objectname.play(filename.wav): This function is used to play the file(present in SD card) mentioned in it as an argument.

Section 4.3: Working of code

1. **Producing sound using flex sensors:** The SD.h library is included to establish communication with SD card module reader in SPI serial communication protocol. TMRpcm library is used for producing audio using PWM functionality of arduino in digital pin 11. The 2 variables flex1 and flex2 are used to store the digitally converted data corresponding to the bent position of 2 flex sensors.

   In the setup section, we have created an object of type TMRPCM. Later we have defined pin 11 to produce audio output and set the volume to max(we can set volume from 0-7).

In the loop section, we have called the user-defined function avg() which takes 200 readings of the flex sensor position and returns an average value back to the main function which is used to decode a specific audio message. We have encoded 6 different audio messages for 6 different combinations of flex sensor positions.

2. Producing sound using accelerometers:

The wire.h() library is used for I2 C communication. Each device that uses the I2C communication has a unique I2C address, and this address can be found in the datasheet of the sensor. The I2C address of the accelerometer is 0x53 as found from the data sheet. In the setup section, we have to set the acceleration in measuring mode. To do that, we have to set the D3 bit of the power control register(address - 0x2D).

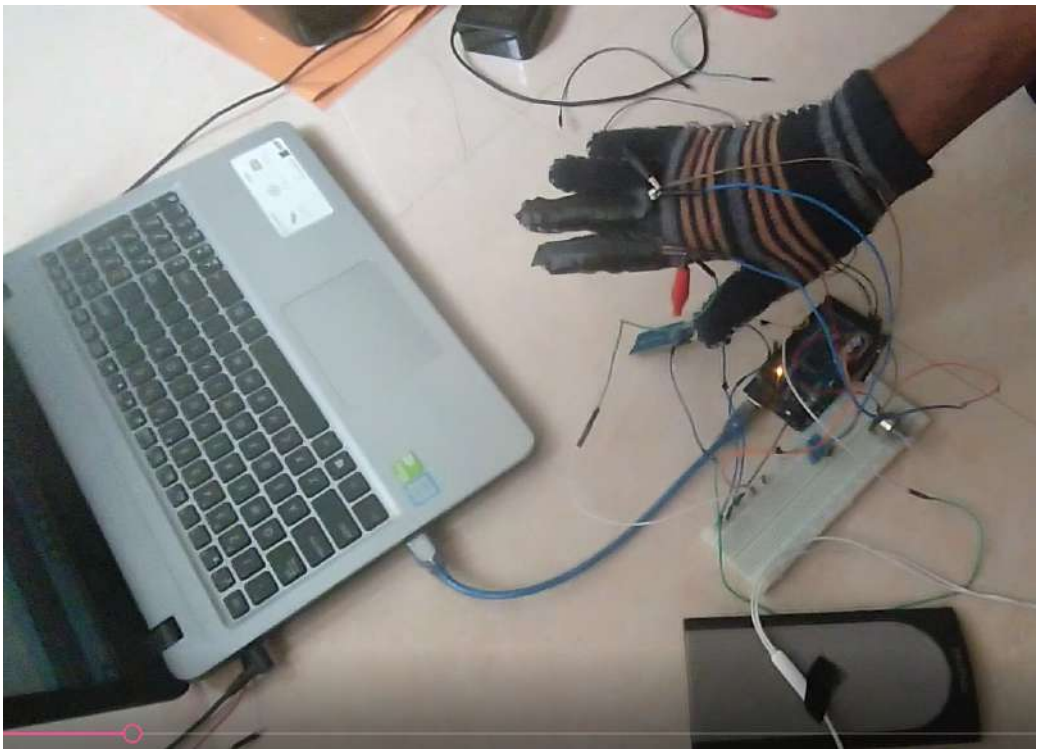
The data for each axis is stored in two bytes or registers. In order to read them all, we start with the first register, and then using the requestionFrom() function we ask to read the 6 registers. Then using the read() function, we read the data from each register. Later in the section, we have encoded a unique audio output for 4 different positions of the hand.

Chapter-5:

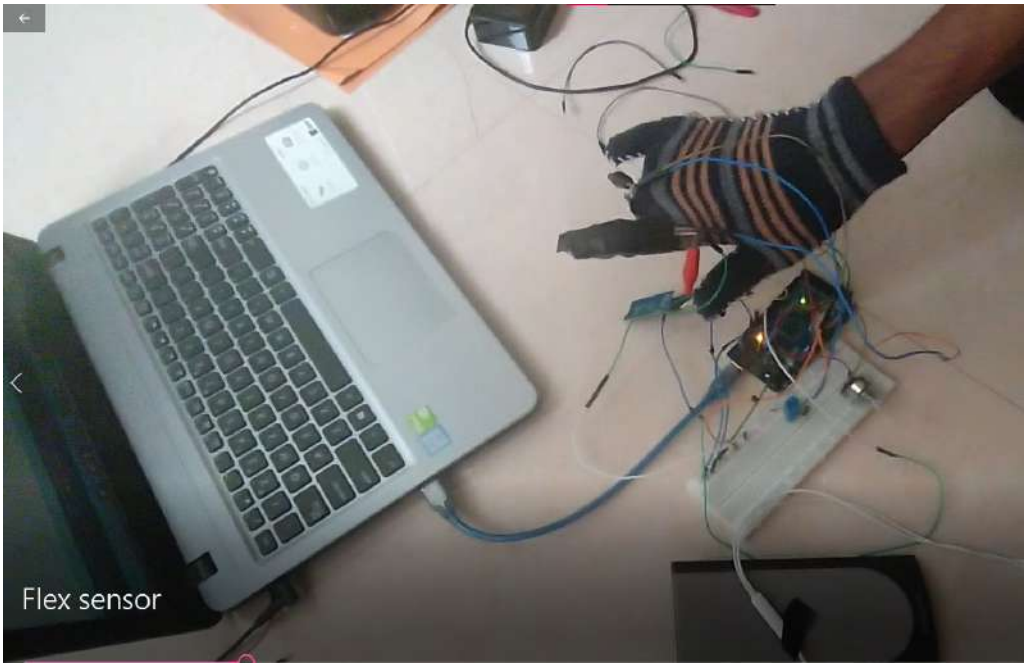
RESULTS

Section 5.1: Using flex sensor

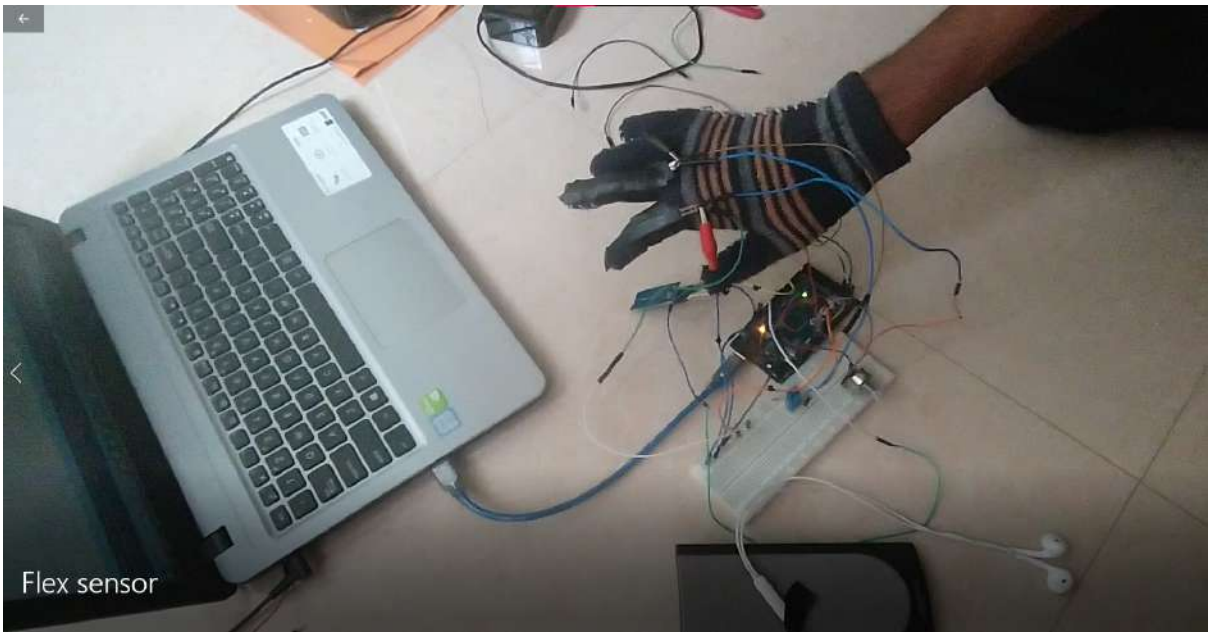
1. The following combination of flex sensors produces a “Hello, good morning” message.



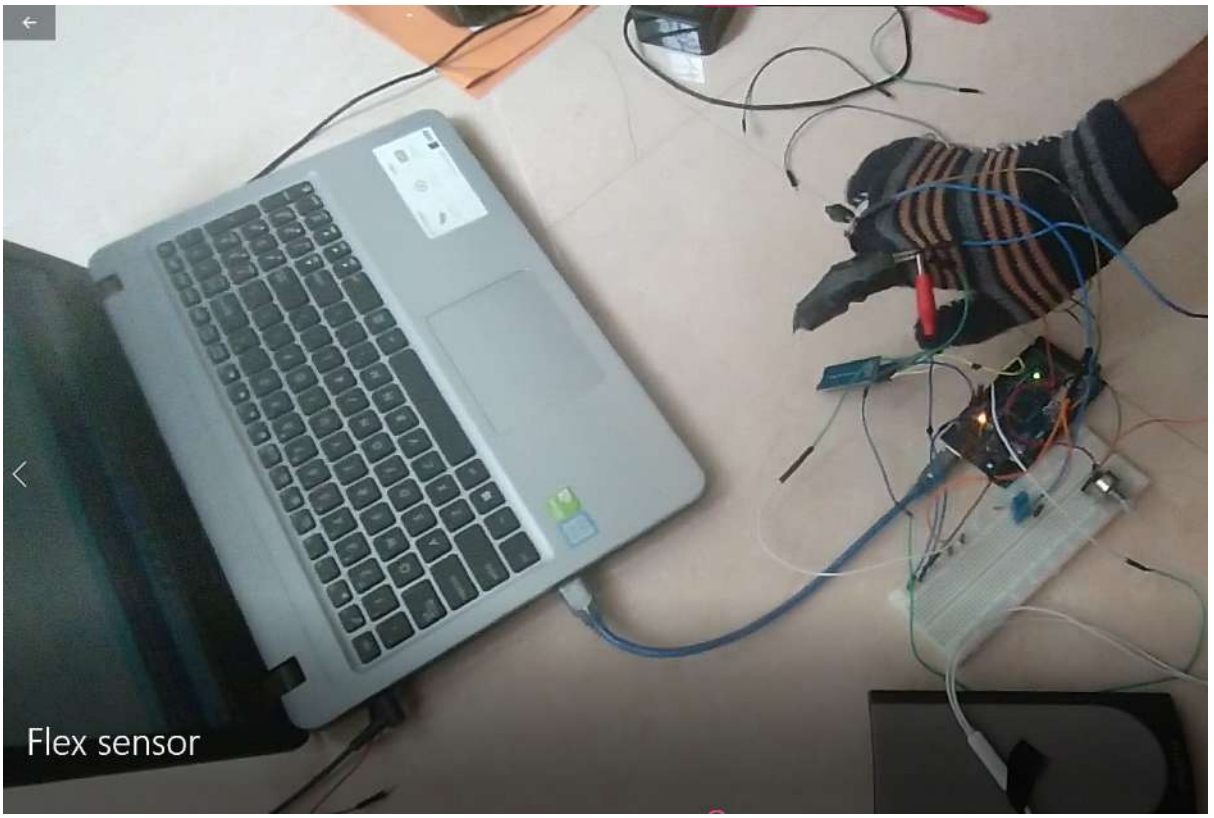
2. The following combination of flex sensors produces a “Can you please do me a favour” message.



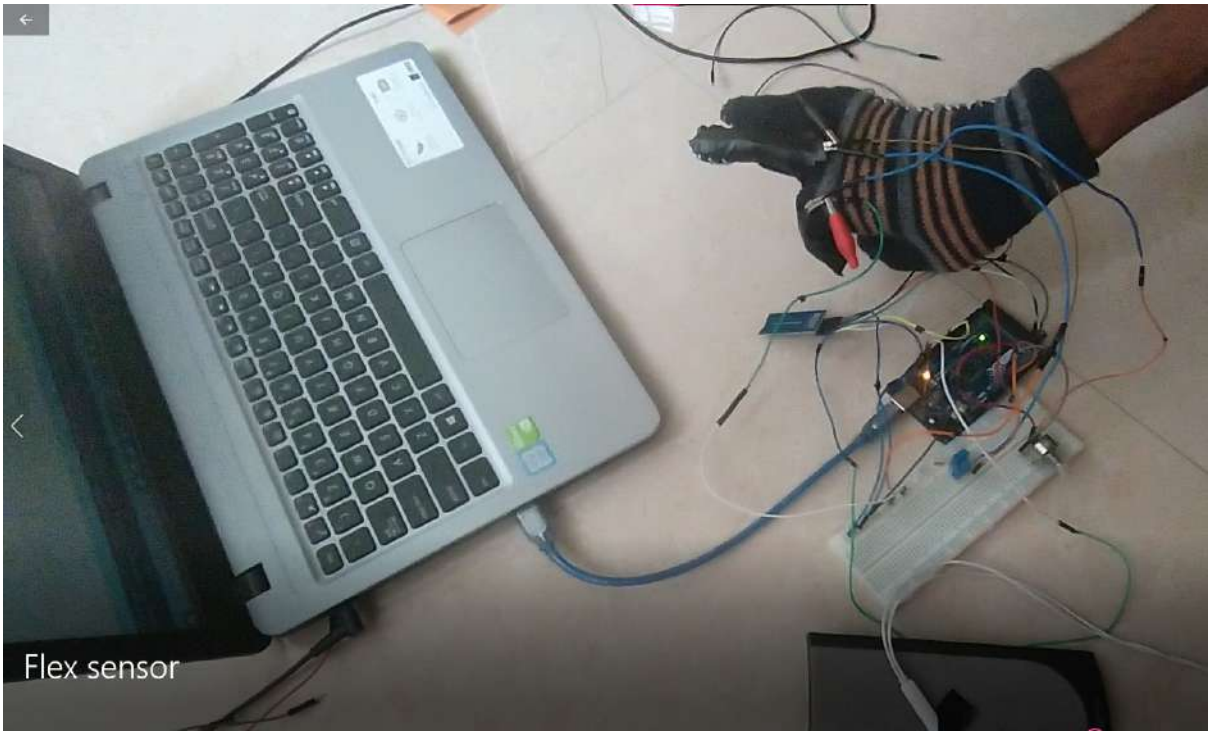
3. The following combination of flex sensors produces a “Please help me” message.



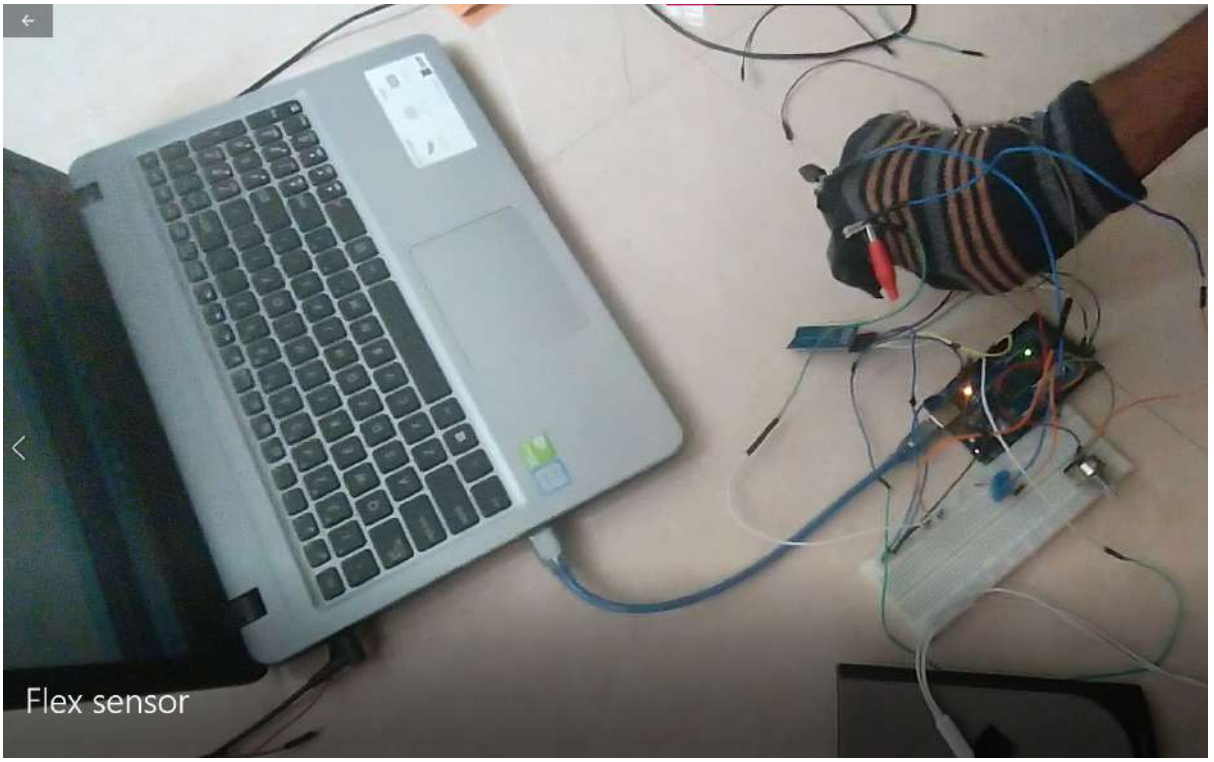
4. The following combination of flex sensors produces a “Can you help me cross the road” message.



5. The following combination of flex sensors produces a “Can you please convey my message to him” message.



6. The following combination of flex sensors produces a “Thanks a lot, I am so grateful to you” message.



Section 5.2: Using accelerometer

1. Accelerometer bent towards positive x-axis - In this position, the speaker plays “Hello, good morning” message.

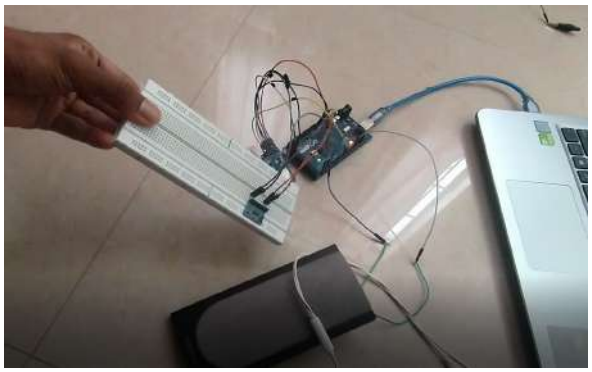


Fig 5.7 Accelerometer bent forward

2. Accelerometer bent towards negative x-axis - In this position, the speaker plays a “Please do me a favour” message.

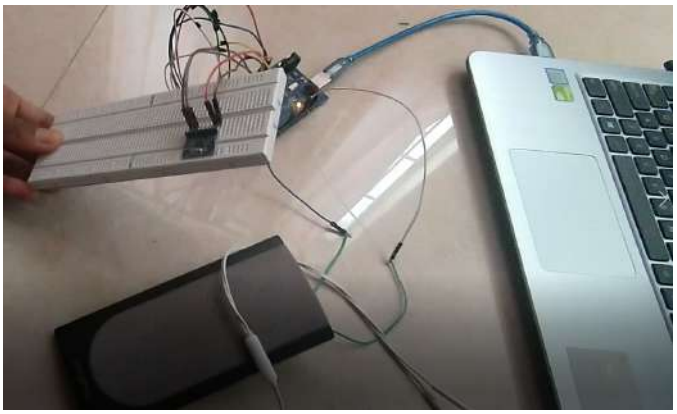


Fig 5.8 Accelerometer bent backward

3. Accelerometer bent towards positive y-axis - In this position, the speaker plays the “Please help me” message.

Fig 5.9 Accelerometer bent in left direction

4. Accelerometer bent towards negative y-axis - In this position, the speaker plays “Can you please help me cross the road” message.

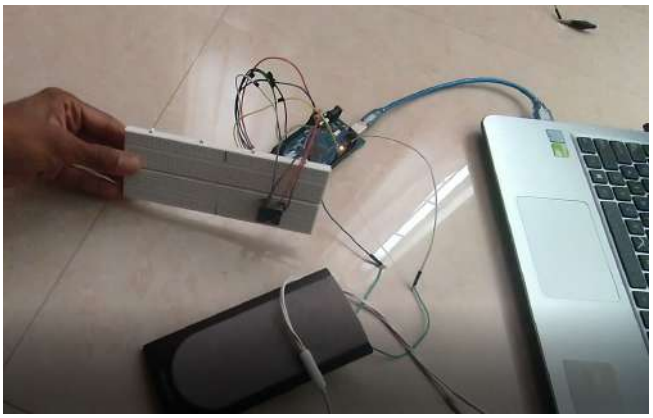


Fig 5.10 Accelerometer bent in right direction

Chapter-6:

ADVANTAGES AND APPLICATIONS

Section 6.1: Advantages

- Low cost
- Compact systems
- Flexible to users
- It takes less power to operate system

Section 6.2: Applications

- Gesture recognition and conversion.
- As a translating device for Mute people.
- It can be used for Mobiles for SMS sending.
- Translation of sign language in many regional languages.

Chapter-7:

CONCLUSION

The proposed method is tested on different gestures. It produces fairly stable and good results. Every person cannot come and share their thoughts with these physically impaired people. So we have come up with a system which would enable the dumb to communicate with each and every one by using the image processing based language converter and sign language recognition system proposed for human computer interaction using Image Processing Technique

Section 7.1: Problems faced:

- Faced difficulty in producing sound with arduino as it does 6not have built in DAC.
- Faced storage problem in arduino as it has just 64 kb of flash memory where program is also stored.
- Faced difficulty in procuring quality components during lockdown.

Section 7.2: Future scope:

The system can be further improved by interfacing digital to analog converter to improve the quality of sound and audio amplifier to produce higher volume of audio. The entire system can be made smaller so as to fit entirely on gloves. This can be done by choosing only the microcontroller without a development board. We can also have smaller speaker and the entire wire mess can be avoided by designing a dedicated single PCB of hand size which can be mounted on gloves and can fit all the necessary components.

Section 7.3: References:

7.3.1 Links:

https://components101.com/misc/microsd-card-pinout-datasheet

http://www.dejazzer.com/ee379/lecture_notes/lec12_sd_card.pdf

https://en.wikipedia.org/wiki/Diaphragm_(acoustics)

https://www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.pdf

https://github.com/TMRh20/TMRpcm/wiki

https://howtomechatronics.com/tutorials/arduino/how-i2c-communication-works-and-how-to-use-it-with-arduino/

7.3.2 Research papers:

“Deaf-Mute Communication Interpreter” by Anbarasi Rajamohan, Hemavathy R., Dhanalakshmi M. International Journal of Scientific Engineering and Technology (ISSN : 2277-1581) Volume 2 Issue 5, pp : 336-341, 1 May 2013

“Microcontroller and Sensors Based Gesture Vocalizer” by Salman Afghani, Muhammad Akmal, Raheel Yousaf . Proceedings of the 7th WSEAS International Conference on signal processing, robotics and automation

Yongjing Liu, Yixing Yang, Lu Wang, Jiapeng Xu, “Image Processing and Recognition of Multiple Static Hand Gestures for Human-Computer Interaction”, IEEE 2013 Seventh International Conference on Image and Graphics.