HAPTIC COMMUNICATION SYSTEM FOR DEAFBLIND

MINIPROJECT REPORT

As partial fulfillment of the curriculum

By

ABDUL SALAM C (NSAOEEC001)

AISWARYA LAKSHMI TAKKA RAVUNNIY (NSAOEEC007)

ALNITTO P THOMAS (NSAOEEC012)

APOORVA ELIZA JOHN (NSAOEEC020)

Under the guidance of

Prof. ANOOP S PILLAI

Department of Electronics & Communication Engineering

NSS College Of Engineering, Palakkad

June – 2017

N.S.S.COLLEGE OF ENGINEERING

PALAKKAD, KERALA 678008

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that this is a bonafied report of the project titled **"HAPTIC COMMUNICATION SYSTEM FOR DEAFBLIND"** done by **ABDUL SALAM C (NSAOEEC001), AISWARYA LAKSHMI TAKKA RAVUNNIY (NSAOEEC007), ALNITTO P THOMAS (NSAOEEC012)** and **APOORVA ELIZA JOHN (NSAOEEC020)** during the academic year 2017 under the guidance of **PROF.ANOOP S PILLAI** as a part of the partial fulfillment for the award of Bachelor of Technology in **ELECTRONICS AND COMMUNICATION ENGINEERING** from University of Calicut and no part of this work has been submitted earlier for the award of any degree.

PROJECT GUIDE	**STAFF IN CHARGE**	**HEAD OF THE DEPARTMENT**
Anoop S Pillai	Anoop S Pillai	Kala.L
Associate Professor	Associate Professor	Associate Professor

ACKNOWLEGDEMENT

First of all, we thank the Almighty God, for granting us strength, courage and knowledge to complete this project successfully.

We acknowledge our sincere thanks to the management of NSS College of Engineering for their help in the successful progression of our project. We would also like to express our gratitude to principal **Dr.T.Sudha** for providing us with adequate infrastructure and congenial environment.

We take this opportunity to express our profound gratitude to our project guide and staff incharge **Prof. Anoop.S.Pillai** and Head of Department **Prof. Kala.L**, for all their guidance and valuable support towards making our project a grand success.

We would also like to thank all the non-teaching staff of our department for their constant encouragement throughout our project. Last, but not the least we take pleasant privilege in expressing our heartfelt thanks to our family and friends who offered precious help in completing our project.

ABDUL SALAM C

AISWARYA LAKSHMI TAKKA RAVUNNIY

ALNITTO P THOMAS

APOORVA ELIZA JOHN

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

NSS COLLEGE OF ENGINEERING, PALAKKAD

ABSTRACT

According to the current scenario, the communication technologies for deafblind are outdated and inadaptable to the current technologies. Also, till now, the technologies or methods of communication between a normal person and deafblind person or between two deafblind persons required certain level of proximity. That is, long distance communication has been very difficult to achieve for the deafblind community. Our prime aim is to reduce or do away with the dependence on proximity.

This project aims to eradicate the difficulty of long distance communication for deafblind people using a tactile communication system connected to a network. By implementing, tactile input system as well as derma sensible output system there is no requirement of learning the codes or keys of the system as in cases of existing systems. They can simply use the sign language they use to communicate face to face.

In this project, as a prototype, we have built a half duplex system, that will intake a touch input and transmit the same over a network and corresponding haptic output will be delivered at the receiver's side.

CONTENTS

Page No.

LIST OF FIGURES

Page No.

LIST ACRONYMS AND ABBREVIATIONS

AP – Access Point

DHCP – Dynamic Host Configuration Protocol

GPIO – General Purpose Input/output

HCSD – Haptic Communication System for Deafblind

IP – Internet Protocol

MCU – Micro Controller Unit

RX – Receive

SSID – Service Set Identifier

TCP – Transmission Control Protocol

TX – Transmit

WEP – Wireless Encryption Protocol

WPA – Wi-Fi Protected Access

CHAPTER 1

INTRODUCTION

Deaf blindness is a unique disability which combines varying degrees of both visual and hearing impairment. All individuals who are deafblind experience extreme challenges with communication and mobility. In India, an estimated 5,00,000 people; both children and adult; are deaf blind.(survey conducted by sense international India). There is little awareness about the deaf blind in India.

We live in a world where technology is constantly changing and improving. It is important to remain up-to-date on changing opportunities presented by technology for people who are deaf-blind. The existing communication systems for deafblind people warrant a certain level of proximity. Also their adaptability to the modern technology is very less. Hence long distance communication with similar degree of closeness as in face to face communication, like video calling for normal people is not possible for deafblind community.

1.1 HAPTICS

Haptic communication is a branch of nonverbal communication that refers to the ways in which people and animals communicate, and interact via the sense of touch. The sense of touch is the fundamental component of haptic communication for interpersonal relationships. Touch signals is a generic term referring to the practice of using various methods of touch on the body. They are used to convey visual, social, and environmental information discreetly and in real-time to a person who is deaf-blind. The use of touch signals emerged over time out of a natural instinct to share information with individuals who are deaf-blind about their surroundings in a quick and unobtrusive manner. There is an emerging trend and growing body of research about the use of touch signals with other populations including individuals who are hearing-blind and individuals who have minimal language skills. Additionally, there is research in the use of touch signals in mobility instruction and competitive sporting activities with people who are deaf-blind. The possibilities of the uses for touch signals are endless. Haptic or kinesthetic communication recreates the sense of touch by applying forces, vibrations, or motions to the user. This mechanical stimulation can be used to assist in the creation of virtual objects in a computer simulation, to control such virtual objects, and to

enhance the remote control of machines and devices (tele-robotics). Haptic devices may incorporate tactile sensors that measure forces exerted by the user on the interface.

We are aiming for a system which can enable the deafblind people, with any degree of deafness and blindness, to use the system alike, by the sign languages they use for day to day communication, by using haptic electronics.

The device works in the following way: sender can draw some pattern on a platform which is fixed on their hand, the motion of finger over platform is taken as input and these values are sent via a network to the receiver and the pattern is recreated on the receiver's hand (on skin where tactile output can be felt), all these are carried out simultaneously as a full duplex system. Thus we can remove proximity out of the tactile communication in deafblind. That's how deafblind can communicate over internet, thus opening new a door for them.

In this project we have tried to implement a prototype

For the implementation we use a esp8266-01 (esp-01) module and arduino Uno board at the receiver part and esp8266-12E (NodeMCU) module at the sender, the input (motion of finger) is recorded through pushbuttons and output part where tactile output is needed we use localized vibrational motors .

The chapters are organized as follows. Chapter 2 presents the project description. It contains the objectives, methodology and economic importance. Chapter 3 presents the implementation of the project. It has the block diagram of project, general working and algorithm and flow charts. Chapter 4 lists the future works. Chapter 5 contains the results. Finally Chapter 6 concludes the project.

CHAPTER 2

PROJECT DESCRIPTION

2.1 OBJECTIVES

The existing technologies for deafblind communication either require some degree of proximity between the users or the user must be partially able to hear or see. That is, a person with higher degrees of blindness cum deafness cannot expect to mingle with the society as he/she wishes and their interaction with outside world becomes limited. Also these technologies are outdated such that they cannot be used with today's fast developing technological platform. Our objectives are:

1. To develop a system which will remove the problem of proximity
2. Achieve communication between differently abled persons without difference.
3. Using a medium of transmission that shall allow long distance communication.
4. Ensuring adaptability to newer developments.
5. To make sure the message transmission is cent percent successful for tactile perception.

2.2 METHODOLOGY

The process can be divided into the following stages:

1. Selection

   In this process, we select different boards and circuits for our project. Estimating the cost, comparing various parameters, durability, quality and making a decision for the most compact and affordable boards and equipments is done in this process.

2. Access point creation

   Choosing a suitable router/AP to act as the server. Here we have used our Android phone's hotspot to act as the server to which the clients NodeMCU and Esp-01 will connect. This server will assign the clients their IP addresses which are required for further programming.

3. Interfacing and Programming

   In this stage all the programming of NodeMCU, Esp-01 and arduino Uno is done. Interfacing of arduino Uno with the esp-01 and the vibrational motor array with the arduino Uno and also interfacing of pushbutton array with NodeMCU is done.

4. Hardware setup

   This stage includes coordination and connection of all the hardware components and testing and verification of expected results.

5. Final Checking of hardware and software

   Final checking includes comparison of our actual hardware model and its functioning with expected results in terms of accuracy of message reception, time delay and response time.

2.3 APPLICATION/SOCIO-ECONOMIC IMPORTANCE

Deaf blindness is a unique disability which combines varying degrees of both visual and hearing impairment. All individuals who are deafblind experience extreme challenges with communication and mobility. In India, an estimated 5,00,000 people; both children and adult; are deaf blind.(survey conducted by sense international India)There is little awareness about the deaf blind in India. They use different methods of communication. The communication method varies with each person depending on the causes and degree of visual and hearing impairment. Any of the methods used by the deafblind has big limitation: Proximity. The people involved in the conversation have to be very near to each other. Either form of communication requires a tactile input.

That is to say, deafblind people are limited in communicating with people who are at greater distance away from them.

They may send a message using text or keyboard specially designed for them, but there has not been much innovations on them receiving a message.

So we thought of a solution

- To reduce the dependence on proximity
- To enable people with any degree of visual or auditory ability to interact with each other
- To enable these people who have been long ignored by society and mistaken to be mentally retarded to become active members of society
- To enable long distance communication for the deafblind
- To develop a system which can be used by anyone.

CHAPTER 3
IMPLEMENTATION

The proposed HCSD includes separate input and output sections. The input section includes the switch array connected to the NodeMCU and output contains an esp-01, arduino Uno and a vibrational motor array. These two units are interlinked via a Wi-Fi AP/Router, or here hotspot of android phone.

3.1 BLOCK DIAGRAM

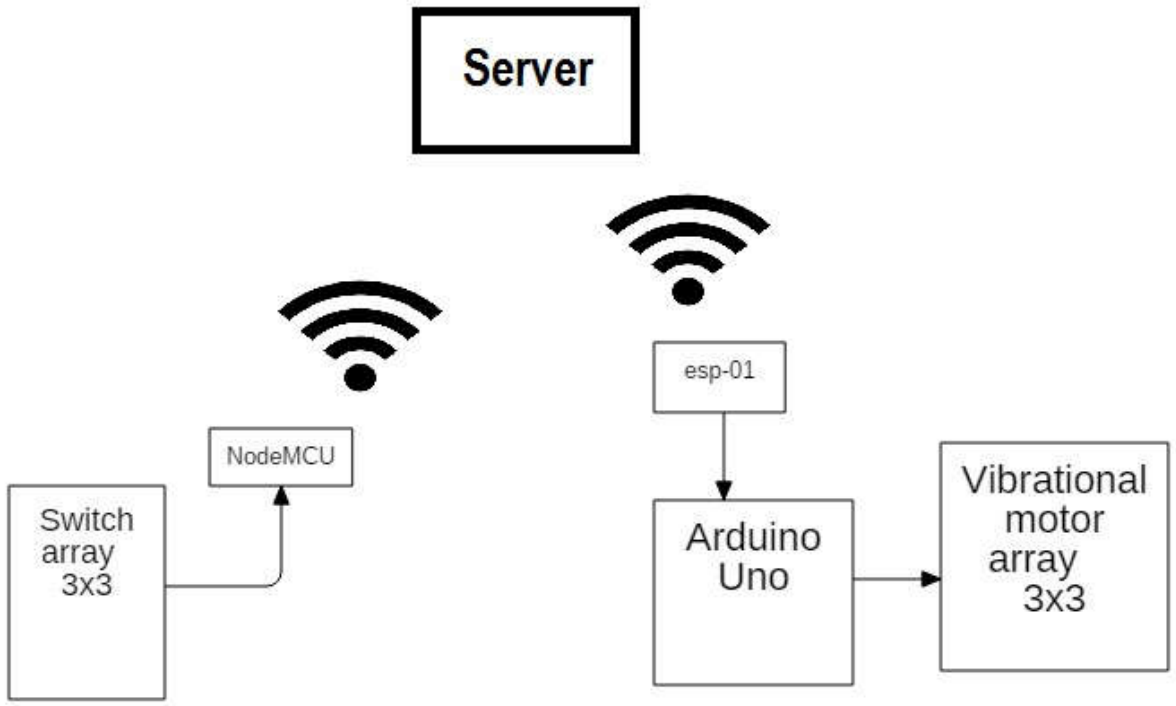

Fig 3.1 Block Diagram of HCSD

3.1.1 Switch Array

A push-button (also spelled pushbutton) or simply button is a simple switch mechanism for controlling some aspect of a machine or a process. Buttons are typically made out of hard material, usually plastic or metal. The surface is usually flat or shaped to accommodate the human finger or hand, so as to be easily depressed or pushed. This switch typically has two terminals. It is commonly referred to as a simple on-off switch and can be used to switch the power supply to a circuit. SPST switches can also work as "push-to-make" on, where when the button is released it returns to its normally open (off) position or vice-versa.

Here, an array of push button switches are used as input. The user can press the required pattern which will get transmitted.

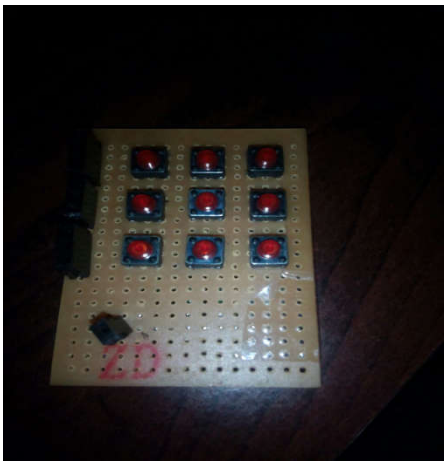

Fig 3.2 Switch Array

3.1.2 NodeMCU module

Node MCU module has an in build ESP8266 based wifi transceiver, which is capable of creating a wifi hotspot region with WPA2PSK authentication. It has 16 GPIO pins which can be configured both as input or output and an analog input pin for interacting with analog input signals. Node MCU is capable of establishing IP based connectivity in one network. This is capable of handling an http server, DHCP server etc, it has USART interface for communicating

with microcontrollers. This is utilized for interfacing with Arduino board. Node MCU run on a clock frequency of 80MHz and wifi module has a band of 2.4 GHz. Fig 3.3 shows the node MCU module.

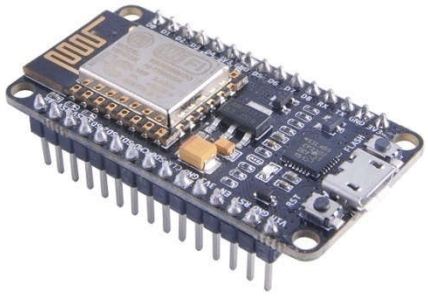

Fig 3.3 NodeMCU module

The NodeMCU module has 16 GPIO pins. These are initialized as input pins and state changes of these pins is monitored. The NodeMCU module also has an ESP8266 based WiFi module which is used to connect wirelessly to a server. A half duplex connection is established between the server and the NodeMCU. The NodeMCU module sends http requests to server according to state of its GPIO pins.

3.1.3 ESP-01

ESP-01 is an ESP8266 board with 8 pins that allows microcontrollers to access a WiFi network and make simple TCP/IP connections using Hayes-style commands. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another

application processor. The ESP-01 has a dedicated UART interface with pins labeled TX and RX. The TX pin is the ESP8266 transmission (outbound from ESP-01) and the RX pin is used to receive data (inbound into the ESP-01). These pins can be connected to a UART partner. We can also use its pin 0 and 2 as software serial pins. An ESP8266 device can play the role of an Access Point, a Station or both at the same time. Fig 3.4 shows an Esp-01 module.

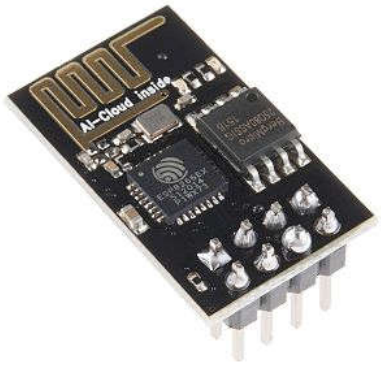

Fig 3.4 ESP-01 Module

Here esp-01 is used as a station to connect to a certain server. It receives http requests and sends back requested data to server. Also via Software Serial connection it is connected to the arduino Uno board.

3.1.4 Arduino Uno

This Arduino Uno board is the basic platform board of Arduino board. This is based on an ATmega 328 microcontroller, which has digital and analog pins, USART, TIMERS, COUNTERS, etc. This can be programmed using Arduino IDE. Here it is used to control the vibrational motors and also to interpret the control words given by the Esp-01. Arduino's digital

pins 4 to 11 are connected to vibrational motors. Pins 2 and 3 are used for serial communication with Esp-01.

Fig 3.5 Arduino Uno

3.1.5 Vibrational Motor Array

Vibration motor is a compact size coreless DC motor. Vibration motors are widely used in a variety of applications including cell phones, handsets, pagers, and so on. The main features of vibration motor is the magnet coreless DC motor are permanent, which means it will always have its magnetic properties (unlike an electromagnet, which only behaves like a magnet when an electric current runs through it); another main feature is the size of the motor itself is small, and thus light weight. Moreover, the noise and the power consumption that the motor produce while using are low. Based on those features, the performance of the motor is highly reliable. Here the vibration motor is used as a means to provide a feeling of touch on the receiver's hands.

The pattern transmitted by pushing the buttons from the input side is replicated by making the corresponding motor vibrate thus creating the same pattern on the receiver's side.

Here we have isolated vibration of each motor from others by using rubber structure as shown.

Fig. 3.6 Vibrational Array structure

3.1.6 Access Point/Server

We are using the mobile hotspot present in our android phone to function as the server. It has WEP and WPA encryption, so only stations with correct SSID and Password will be able to connect this server. A single server can have multiple clients and hence multiple people can talk at the same time. Also if the server is connected to the internet, global coverage is also possible.

3.1.7 Voltage Regulator – 3.3V

This is the basic LD1117V33 voltage regulator, a low drop positive regulator with a 3.3V fixed output voltage. This fixed regulator provides a great amount of stability and protection to the esp-01 as its maximum tolerable voltage is 3.3 V.

3.2 GENERAL WORKING

All the blocks are connected as shown in the connection below. The coordination and functioning of each unit is mentioned below.

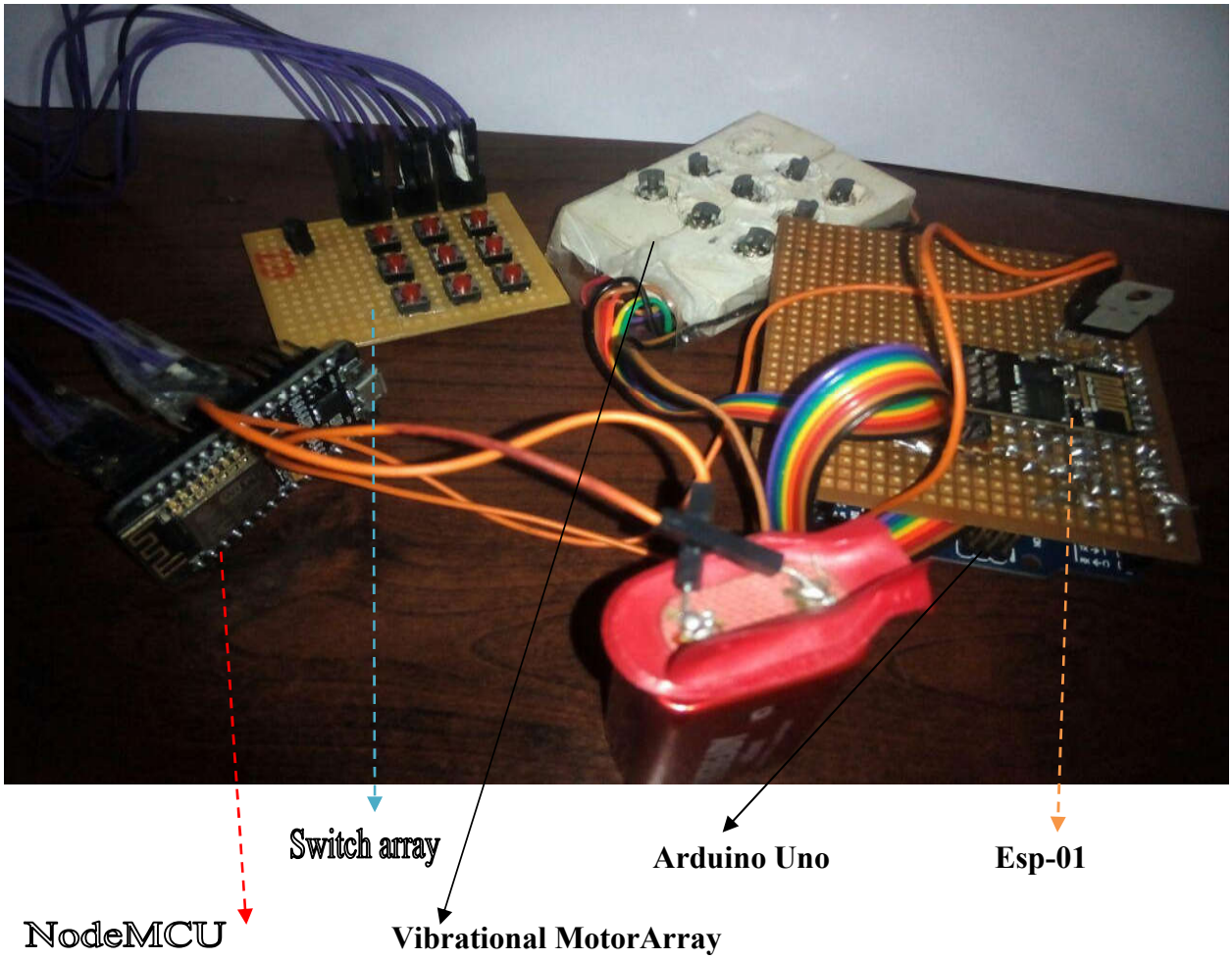

Fig. 3.7 Hardware Setup of HCSD

3.2.1 SENDING A MESSAGE

Upon switching ON, the NodeMCU at the input tries to connect to the server with SSID as in its program. When connection has been authorized by the server, then it starts to monitor its GPIO pins. The user can create send their message by creating the pattern/message on the pushbutton array. Each push button is connected to one of the digital pins of the NodeMCU. As soon as a state change is detected, depending on whether it was ON signal or OFF signal, NodeMCU sends an http request to the server. The request will contain the ip address of the receiver followed by an argument which gives the information about which pushbutton was detected and what is its

current state. The pushbuttons are initialized with pullup that is we have active low logic. So when the buttons are grounded we detect a HIGH.

Ex. When pushbutton 3 is pressed, it is grounded and a change in state is detected and so an http request will be sent for the following URL.

http://192.168.43.198/3/y

Here 192.168.43.198 is the ip address of the receiver, /3 gives which pushbutton changed its state and /y says it had been pressed.

When the button is released again we have state change and this is noted as a LOW since ground has been removed. So again an http request is sent as: http://192.168.43.198/3/n

Here /n will inform that the button has been released.

Since NodeMCU will only send requests when there is a change of state, we can ensure status of button is only sent once and hence bit stream traffic is not overloaded.

3.2.2 RECEPTION OF MESSAGE

The Esp-01 at the receiver side will first try to connect to the server with the SSID and password as given in its program, upon starting up the receiver module. When the server gives the authentication it will wait for any http requests from server. When the server receives http requests corresponding to ip address of esp-01, the esp-01 will give a response to the request according to the request. If the request has only simply the ip address of the esp-01, it will send a " hello from esp8266!" to the server.

If there is an argument present after the ip address, then the esp-01 will check the argument and send a response accordingly.

Ex. If the request has argument /3/y it will send a value 200 followed by "okay – light is ON". This indicates that a button was pressed. At the same time it sends a message to arduino Uno to ON the vibrational motor corresponding to push button 3. This motor will remain ON until the OFF signal is received, that is, until /3/n argument is being requested. Hence the motor will

vibrate for the entire duration the button was pressed, without sending recurring signal to keep the motor ON because we are monitoring only change in state at the input side.

The motor will be turned OFF by the arduino when /3/n request arrives.

The arduino Uno when switched ON will first initialize its pins 4 to 11 as output pins. These are connected to the vibrational motor. When it receives a serial input from the esp-01, it compares input and changes the state of appropriate vibrational motor.

3.2.3 MESSAGE INTERPRETATION

Deafblind people with a high degree of deafness and blindness use print on palm technique to communicate. Similarly here, symbols usually printed on palm are recreated on the switch array. The message will hence be transmitted and recreated on the skin of the person who is wearing the vibrational array unit on their hand or palm. Since motors will only stay ON for the duration of the button being pressed down, the pattern will recreated faithfully.

3.3 ALGORITHM AND FLOW CHART FOR NODEMCU PROGRAMME

3.3.1 ALGORITHM

Step 1: Establish connection with the access point via WiFi

Step 2: Initialize the general purpose pins as inputs.

Step 3: Monitor the digital I/O pins

Step 4: Send an http get request to the server , according to the pin status.

Step 5: Print the request

Step 6: Print the request response payload.

Step 7: Repeat steps 3 through 6

Step 8: Stop

3.3.2 FLOWCHART

Fig 3.7 shows flow chart corresponding to NodeMCU.

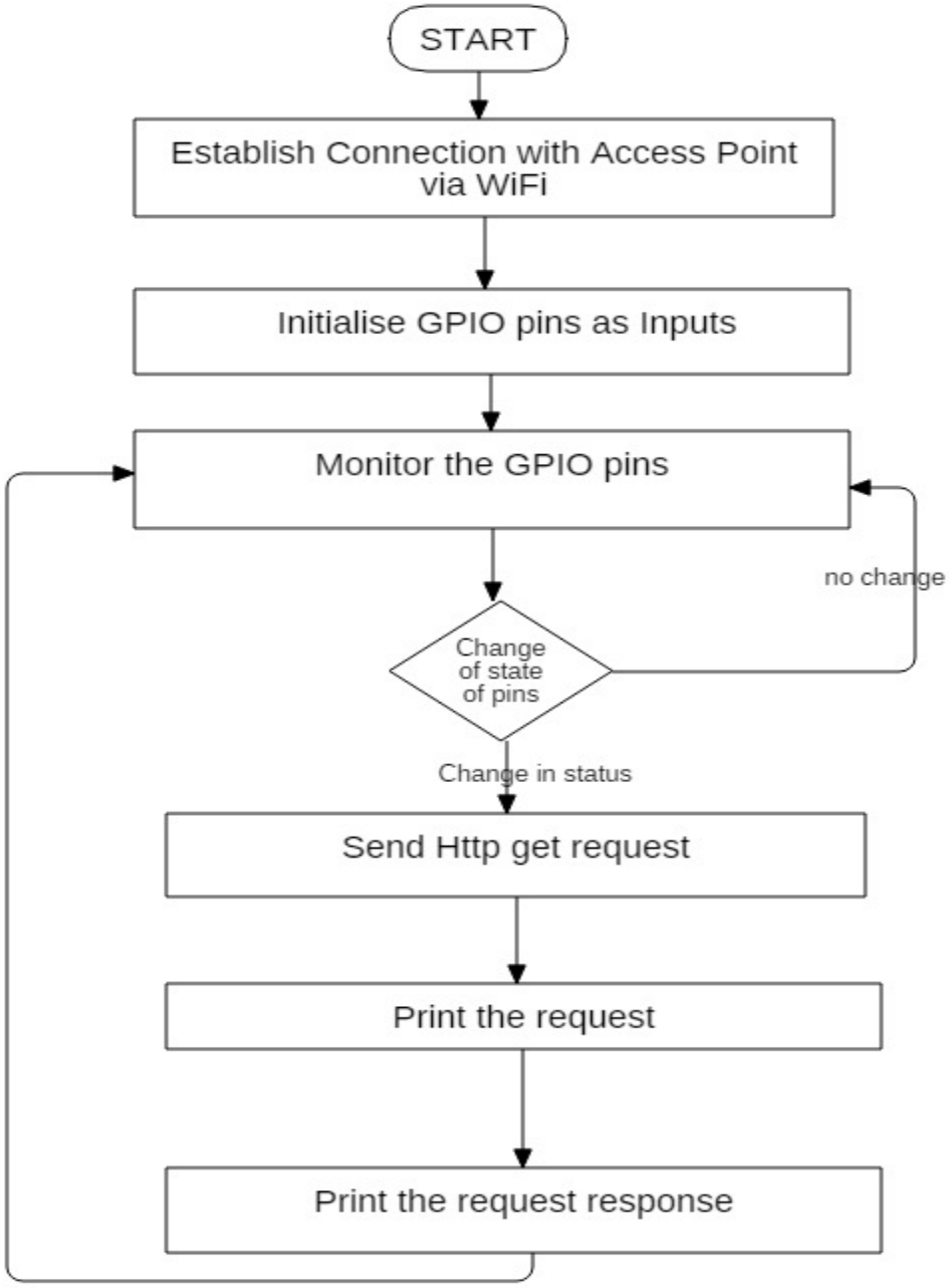

Fig. 3.8 Flow Chart of NodeMCU

3.4 ALGORITHM AND FLOW CHART FOR ESP-01

3.4.1 ALGORITHM

Step 1: Establish connection with access point via WIFi

Step 2: Initialize GPIO 0 and GPIO 2 as software serial pins.

Step 3: Establish a TCP connection with the server.

Step 4: Monitor the server for Http requests.

Step 5: If argument of URL is not present, send " hello from esp".

Step 6: If argument present, send appropriate response.

Step 7: Serial print according to received argument, to the arduino.

Step 8: Repeat steps 4 to 7.

Step 9: Stop

3.4.2 FLOW CHART

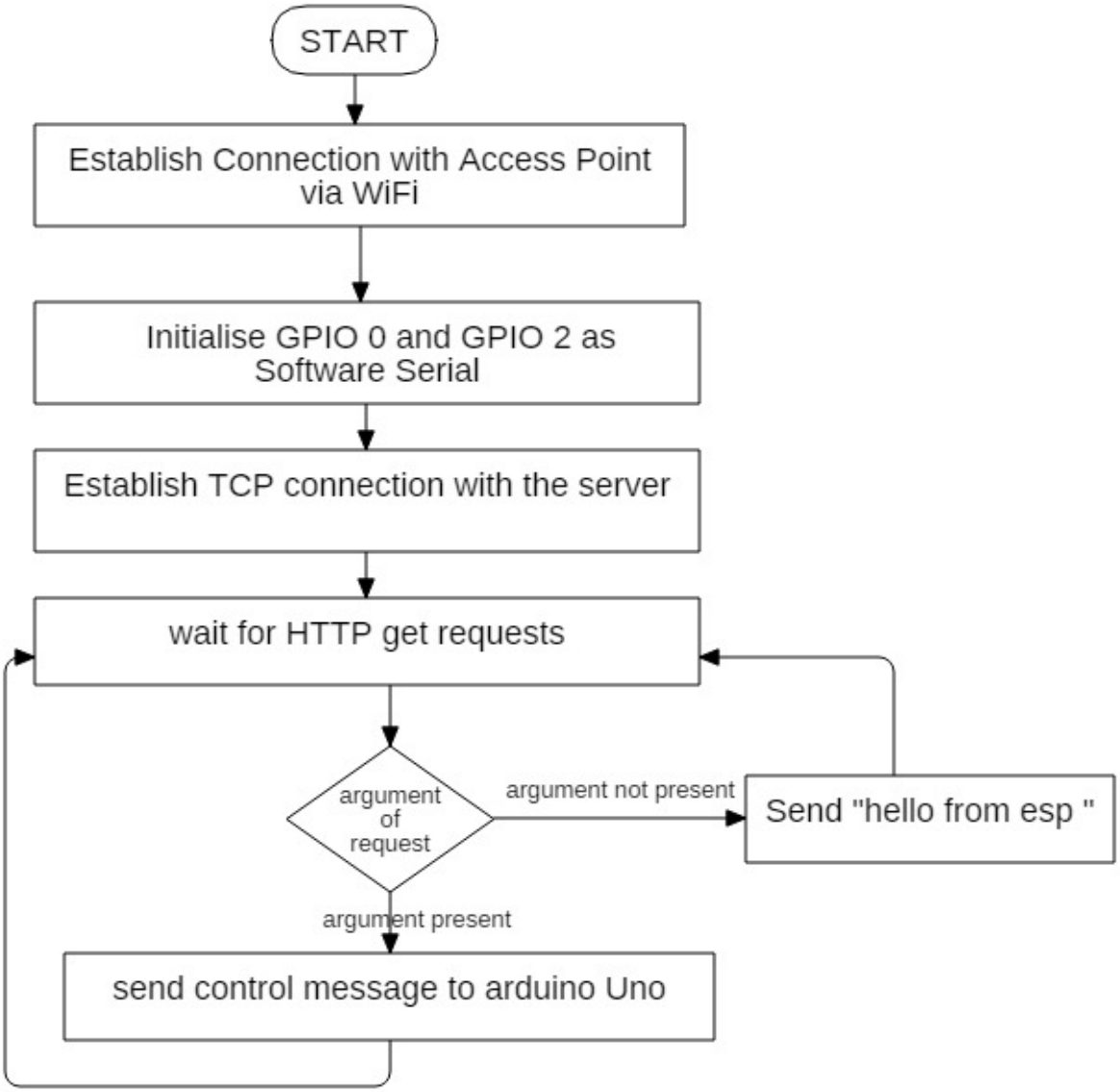

Fig 3.8 Flow Chart of esp-01

3.5 ALGORITHM AND FLOW CHART FOR ARDUINO UNO

3.5.1 ALGORITHM

Step1: Intialise the digital pins as outputs.

Step 2: Intialise pins 2 and 3 as Software serial pins.

Step 3: Monitor the serial input from esp8266.

Step 4: Change the state of the appropriate digital pin, according to received serial input.

Step 5: Repeat steps 3 and 4.

Step 6: Stop.

3.5.2 FLOW CHART

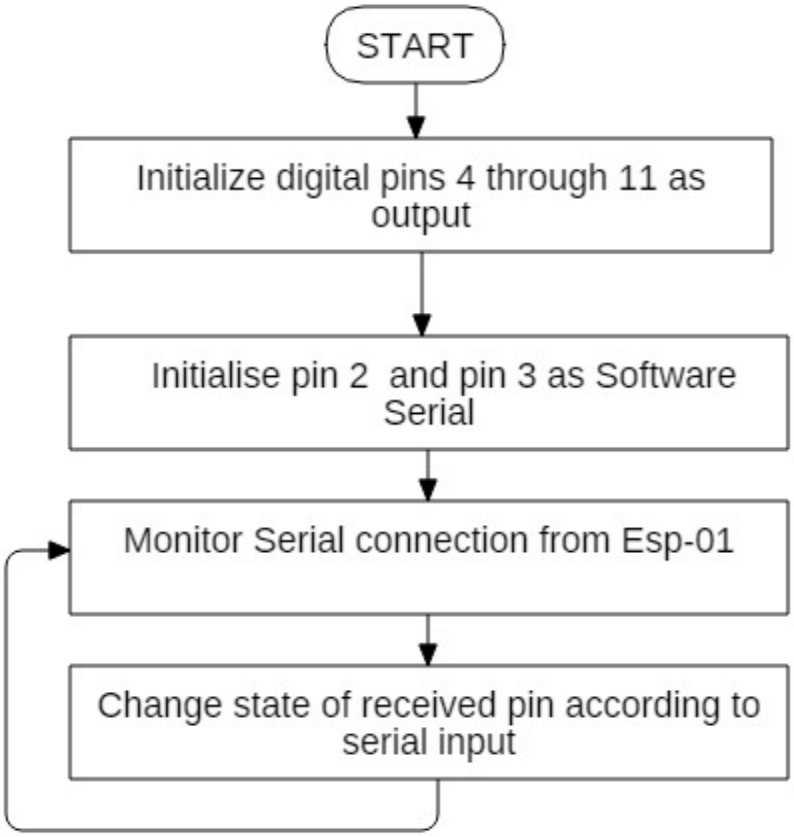

Fig.3.10 Flow Chart of Arduino Uno

CHAPTER 4

RESULTS

The proper working of the device was verified. A pattern was given as input at the switch array and similar pattern was obtained at the vibrational motor array with negligible delay. The vibrations of each motor could be perceived and differnetiated on the skin of the user as individual points or locations. The transmission between server and NodeMCU and between server and esp-01 has been found to be continuos even with a distance of more than 100m between the same. It has also been found that a message can be sent by simply inputing the appropriate url at server itself.

Overall the device has exhibited expected outputs.

CHAPTER 5

CONCLUSION

The proposed HCSD can be used to replace current deafblind communicators. It has been found that real time transmission of message is possible over Wi-Fi. And multiple numbers of users can access this system through the same server enabling communication with multiple users simultaneously. There is no or negligible delay in transmission over Wi-Fi. The proposed system can also send data to user directly from server enabling access of internet also via this system. The proposed system has successfully enabled interpreting a touch input and recreating the same touch pattern at the output over a distance via a network. Hence this system can be the considered as part of the developing haptic technologies where the aim is to recreate the sense of touch.

CHAPTER 6

FUTURE WORK

The future scope of this project is vast. A few upgrades we would like to make in the future are listed below:

1. The input can be changed to a Touchpad. So that the pattern or symbol will have continuity and any kind of pattern can be drawn or recognized.
2. There is much research going on in the field of ultrasound haptics. In future, output can be changed to similar Haptic solution which can recreate the feeling of touch with pressure variations and also continuous patterns can be formed.
3. Connecting the server to the internet will provide global coverage for transmission.
4. An android and iOS app can be developed for normal phone users who can simply draw their messages on their phones and these can be transmitted via the server.

CHAPTER 7

APPENDICES

A.1 NodeMCU Program

```
#include <ESP8266WiFi.h>

#include <ESP8266HTTPClient.h>


const char* ssid = "Danjo";

const char* password = "Openadaa";

const int PIN1 = 16;

const int PIN2= 5;

const int PIN3 = 4;

const int PIN4 = 0;

const int PIN5 = 2;

const int PIN6 = 14;

const int PIN7 = 12;

const int PIN8 = 13;

const int PIN9 = 15;

 int p1= LOW;

 int p9= LOW;

 int p3=LOW;

 int p2= LOW;
```

```
 int p4= LOW;

 int p5=LOW;

 int p6= LOW;

 int p7= LOW;

 int p8=LOW;

    HTTPClient http;


void setup () {


  Serial.begin(9600);

  WiFi.begin(ssid, password);


    pinMode(PIN1, INPUT_PULLUP);

    pinMode(PIN2, INPUT_PULLUP);

    pinMode(PIN3, INPUT_PULLUP);

    pinMode(PIN4, INPUT_PULLUP);

    pinMode(PIN5, INPUT_PULLUP);

    pinMode(PIN6, INPUT_PULLUP);

    pinMode(PIN7, INPUT_PULLUP);

    pinMode(PIN8, INPUT_PULLUP);

    pinMode(PIN9, INPUT_PULLUP);
```

```
  while (WiFi.status() != WL_CONNECTED) {


    delay(1000);
    Serial.print("Connecting..");


  }Serial.print("Connected");
}


void loop() {
  if (WiFi.status() == WL_CONNECTED) { //Check WiFi connection status


 //Declare an object of class HTTPClient
 if ((digitalRead(PIN1)==LOW)&&(p1== LOW)){
  p1=HIGH;
    Serial.println("1 is high");
    http.begin("http://192.168.43.198/1/y");  //Specify request destination
    int httpCode = http.GET();
    Serial.println(httpCode);//Send the request


    if (httpCode > 0) { //Check the returning code
```

```
    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload
  }

   http.end(); }  //Close connection
   if ((digitalRead(PIN1)==HIGH)&&(p1==HIGH)){
     p1=LOW;
     Serial.println("1 is low");
   http.begin("http://192.168.43.198/1/n");  //Specify request destination
  int httpCode = http.GET();                                          //Send the request
    Serial.println(httpCode);//Send the request

   if (httpCode > 0) { //Check the returning code

     String payload = http.getString();   //Get the request response payload
     Serial.println(payload);                    //Print the response payload
    }
   http.end();  } //Close connection
if ((digitalRead(PIN2)==LOW)&&(p2== LOW)){
  p2=HIGH;
```

```
  Serial.println("2 is high");
    http.begin("http://192.168.43.198/2/y");  //Specify request destination
    int httpCode = http.GET();                                            //Send the request
     Serial.println(httpCode);//Send the request

    if (httpCode > 0) { //Check the returning code

      String payload = http.getString();   //Get the request response payload
      Serial.println(payload);                    //Print the response payload

    }

    http.end();   }//Close connection
    if ((digitalRead(PIN2)==HIGH)&&(p2== HIGH)){
      p2=LOW;
      Serial.println("2 is low");
 http.begin("http://192.168.43.198/2/n");  //Specify request destination
   int httpCode = http.GET();                                            //Send the request
     Serial.println(httpCode);//Send the request

    if (httpCode > 0) { //Check the returning code
```

```
    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload

  }

  http.end();  } //Close connection

  if ((digitalRead(PIN3)==LOW)&&(p3== LOW)){
p3=HIGH;
  Serial.println("3 is high");
  http.begin("http://192.168.43.198/3/y");  //Specify request destination
  int httpCode = http.GET();
  Serial.println(httpCode);//Send the request

  if (httpCode > 0) { //Check the returning code

    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload

  }
```

```
  http.end(); }  //Close connection
  if ((digitalRead(PIN3)==HIGH)&&(p3==HIGH)){
    p3=LOW;
    Serial.println("2 is low");
  http.begin("http://192.168.43.198/3/n");  //Specify request destination
 int httpCode = http.GET();                                            //Send the request
   Serial.println(httpCode);//Send the request

  if (httpCode > 0) { //Check the returning code

    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload

  }

  http.end();  } //Close connection

  if ((digitalRead(PIN4)==LOW)&&(p4== LOW)){
p4=HIGH;
  Serial.println("4 is high");
```

```
http.begin("http://192.168.43.198/4/y");  //Specify request destination
int httpCode = http.GET();
Serial.println(httpCode);//Send the request

if (httpCode > 0) { //Check the returning code

  String payload = http.getString();   //Get the request response payload
  Serial.println(payload);                    //Print the response payload

}

http.end(); }  //Close connection
if ((digitalRead(PIN4)==HIGH)&&(p4==HIGH)){
  p4=LOW;
  Serial.println("4 is low");
http.begin("http://192.168.43.198/4/n");  //Specify request destination
int httpCode = http.GET();                                          //Send the request
 Serial.println(httpCode);//Send the request

if (httpCode > 0) { //Check the returning code
```

```
    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload

  }

  http.end();  } //Close connection
  if ((digitalRead(PIN5)==LOW)&&(p5== LOW)){
p5=HIGH;
  Serial.println("5 is high");
  http.begin("http://192.168.43.198/5/y");  //Specify request destination
  int httpCode = http.GET();
  Serial.println(httpCode);//Send the request

  if (httpCode > 0) { //Check the returning code

    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload

  }

  http.end(); }  //Close connection
```

```
  if ((digitalRead(PIN5)==HIGH)&&(p5==HIGH)){
    p5=LOW;
    Serial.println("5 is low");
  http.begin("http://192.168.43.198/5/n");  //Specify request destination
 int httpCode = http.GET();                                          //Send the request
   Serial.println(httpCode);//Send the request

  if (httpCode > 0) { //Check the returning code

    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload

  }

  http.end();  } //Close connection
  if ((digitalRead(PIN6)==LOW)&&(p6== LOW)){
p6=HIGH;
  Serial.println("6 is high");
  http.begin("http://192.168.43.198/6/y");  //Specify request destination
  int httpCode = http.GET();
  Serial.println(httpCode);//Send the request
```

```
  if (httpCode > 0) { //Check the returning code

    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload

  }

  http.end(); }  //Close connection
  if ((digitalRead(PIN6)==HIGH)&&(p6==HIGH)){
    p6=LOW;
    Serial.println("6 is low");
  http.begin("http://192.168.43.198/6/n");  //Specify request destination
 int httpCode = http.GET();                                          //Send the request
   Serial.println(httpCode);//Send the request

  if (httpCode > 0) { //Check the returning code

    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload
```

```
  }


  http.end();  } //Close connection
  if ((digitalRead(PIN7)==LOW)&&(p7== LOW)){
p7=HIGH;
  Serial.println("7 is high");
  http.begin("http://192.168.43.198/7/y");  //Specify request destination
  int httpCode = http.GET();
  Serial.println(httpCode);//Send the request


  if (httpCode > 0) { //Check the returning code


    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload


  }


  http.end(); }  //Close connection
  if ((digitalRead(PIN7)==HIGH)&&(p7==HIGH)){
    p7=LOW;
    Serial.println("7 is low");
```

```
  http.begin("http://192.168.43.198/7/n");  //Specify request destination
 int httpCode = http.GET();                                          //Send the request
   Serial.println(httpCode);//Send the request


  if (httpCode > 0) { //Check the returning code


    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload


  }


  http.end();  } //Close connection
  if ((digitalRead(PIN8)==LOW)&&(p8== LOW)){
p8=HIGH;
  Serial.println("8 is high");
  http.begin("http://192.168.43.198/8/y");  //Specify request destination
  int httpCode = http.GET();
  Serial.println(httpCode);//Send the request


  if (httpCode > 0) { //Check the returning code
```

```
    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload


  }


  http.end(); }  //Close connection
  if ((digitalRead(PIN8)==HIGH)&&(p8==HIGH)){
    p8=LOW;
    Serial.println("8 is low");
  http.begin("http://192.168.43.198/8/n");  //Specify request destination
 int httpCode = http.GET();                                          //Send the request
   Serial.println(httpCode);//Send the request


  if (httpCode > 0) { //Check the returning code


    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload


  }


  http.end();  } //Close connection
```

```
  if ((digitalRead(PIN9)==LOW)&&(p9== LOW)){
 p9=HIGH;
  Serial.println("9 is high");
  http.begin("http://192.168.43.198/9/y");  //Specify request destination
  int httpCode = http.GET();
  Serial.println(httpCode);//Send the request

  if (httpCode > 0) { //Check the returning code

    String payload = http.getString();   //Get the request response payload
    Serial.println(payload);                    //Print the response payload

  }

  http.end(); }  //Close connection
  if ((digitalRead(PIN9)==HIGH)&&(p9==HIGH)){
    p9=LOW;
    Serial.println("9 is low");
  http.begin("http://192.168.43.198/9/n");  //Specify request destination
 int httpCode = http.GET();                                          //Send the request
```

```
    Serial.println(httpCode);//Send the request


   if (httpCode > 0) { //Check the returning code


     String payload = http.getString();   //Get the request response payload

     Serial.println(payload);                    //Print the response payload


   }


   http.end();  } //Close connection



 }


}
```

A.2 ESP-01 Program

```
#include <SoftwareSerial.h>


#include <ESP8266WiFi.h>

#include <WiFiClient.h>
```

```
#include <ESP8266WebServer.h>

#include <ESP8266mDNS.h>


SoftwareSerial toduino(0,2); // rx,tx!!

const char* ssid = "Danjo";

const char* password = "Openadaa";

MDNSResponder mdns;

ESP8266WebServer server(80);


void handleRoot() {

  server.send(200, "text/plain", "hello from esp8266!");

}




void setup(void){

  toduino.begin(9600);

  Serial.begin(115200);

  WiFi.begin(ssid, password);

  Serial.println("");
```

```
  // Wait for connection
  while (WiFi.status()!= WL_CONNECTED) {
    delay(500);
    Serial.print(".");

  }
  Serial.println("");
  Serial.print("Connected to ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());



  if (mdns.begin("esp8266", WiFi.localIP())) {
    Serial.println("MDNS responder started");
  }

  server.on("/", handleRoot);

  server.on("/1/y", []{
```

```
server.send(200, "text/plain", "Okay -- Light is ON!");
toduino.println("ON1");
});
 server.on("/9/y", [](){
server.send(200, "text/plain", "Okay -- Light is ON!");
toduino.println("ON9");
});
server.on("/9/n", [](){
server.send(200, "text/plain", "Okay -- Light is OFF!");
toduino.println("OFF9");
});
server.on("/1/n", [](){
server.send(200, "text/plain", "Okay -- Light is OFF!");
toduino.println("OFF1");
});
 server.on("/2/y", [](){
server.send(200, "text/plain", "Okay -- Light is ON!");
toduino.println("ON2");
});
server.on("/2/n", [](){
server.send(200, "text/plain", "Okay -- Light is OFF!");
```

```
toduino.println("OFF2");
});

server.on("/3/y", [](){
server.send(200, "text/plain", "Okay -- Light is ON!");
toduino.println("ON3");
});
server.on("/3/n", [](){
server.send(200, "text/plain", "Okay -- Light is OFF!");
toduino.println("OFF3");
});

server.on("/4/y", [](){
server.send(200, "text/plain", "Okay -- Light is ON!");
toduino.println("ON4");
});
server.on("/4/n", [](){
server.send(200, "text/plain", "Okay -- Light is OFF!");
toduino.println("OFF4");
});
server.on("/5/y", [](){
```

```
server.send(200, "text/plain", "Okay -- Light is ON!");
toduino.println("ON5");
});
server.on("/5/n", []{
server.send(200, "text/plain", "Okay -- Light is OFF!");
toduino.println("OFF5");
});
server.on("/6/y", []{
server.send(200, "text/plain", "Okay -- Light is ON!");
toduino.println("ON6");
});
server.on("/6/n", []{
server.send(200, "text/plain", "Okay -- Light is OFF!");
toduino.println("OFF6");
});
server.on("/7/y", []{
server.send(200, "text/plain", "Okay -- Light is ON!");
toduino.println("ON7");
});
server.on("/7/n", []{
server.send(200, "text/plain", "Okay -- Light is OFF!");
```

```
  toduino.println("OFF7");
  });
  server.on("/8/y", []{
  server.send(200, "text/plain", "Okay -- Light is ON!");
  toduino.println("ON8");
  });
  server.on("/8/n", []{
  server.send(200, "text/plain", "Okay -- Light is OFF!");
  toduino.println("OFF8");
  });




  server.begin();
  Serial.println("HTTP server started");
}


void loop(void){
  server.handleClient();
}
```

A.3 Arduino Uno Program

```
#include <SoftwareSerial.h>

const int pin1 = 4;

const int pin2 = 5;

const int pin3 = 6;

const int pin4 = 7;

const int pin5 = 8;

const int pin6 = 9;

const int pin7 = 10;

const int pin8 = 11;

const int pin9 = 12;


SoftwareSerial esp(2,3);


void setup() {

Serial.begin(115200);

esp.begin(9600);

pinMode(pin1,OUTPUT);

pinMode(pin2,OUTPUT);

pinMode(pin3,OUTPUT);

pinMode(pin4,OUTPUT);
```

```
pinMode(pin5,OUTPUT);

pinMode(pin6,OUTPUT);

pinMode(pin7,OUTPUT);

pinMode(pin8,OUTPUT);

pinMode(pin9,OUTPUT);

}


void loop() {


String line = esp.readStringUntil('\r');

Serial.print(line);


if(line=="ON1"){

  digitalWrite(pin1, HIGH);

}

if(line=="OFF1"){

  digitalWrite(pin1, LOW);

}

if(line=="ON9"){

  digitalWrite(pin9, HIGH);

}
```

```
if(line=="OFF9"){
  digitalWrite(pin9, LOW);
}
if(line=="ON2"){
  digitalWrite(pin2, HIGH);
}
if(line=="OFF2"){
  digitalWrite(pin2, LOW);
}

if(line=="ON3"){
  digitalWrite(pin3, HIGH);
}
if(line=="OFF3"){
  digitalWrite(pin3,LOW);
}
if(line=="ON4"){
  digitalWrite(pin4, HIGH);
}
if(line=="OFF4"){
  digitalWrite(pin4, LOW);
```

```
}if(line=="ON5"){
  digitalWrite(pin5, HIGH);
}
if(line=="OFF5"){
  digitalWrite(pin5, LOW);
}if(line=="ON6"){
  digitalWrite(pin6, HIGH);
}
if(line=="OFF6"){
  digitalWrite(pin6, LOW);
}if(line=="ON7"){
  digitalWrite(pin7, HIGH);
}
if(line=="OFF7"){
  digitalWrite(pin7, LOW);
}if(line=="ON8"){
  digitalWrite(pin8, HIGH);
}
if(line=="OFF8"){
  digitalWrite(pin8, LOW);
} }
```

CHAPTER 8

RFERENCES

1. *KOLBAN"S BOOK ON ESP8266. Neil Kohan,November 2016.*
2. *GETTING STARTED WITH ARDUINO. Massimo Benzi. Make(2011).*
3. *www.senseintindia.org*
4. *http://www.perkins.org/stories/category/assistive-technology*