Connecting Rooms
(Arduino + Android at your service)

Group members:
Rochak Saini
Puneet Chawla
Ritik Agarwal
Saksham Gupta
Suyash Agarwal

INTRODUCTION

It is always difficult for a paralysed or a disabled person to stand up, get on their crutches just to change the settings of the room. Also, families with a disabled person always find it a bit problematic to be always present at the service. Although, other people in the family can't really back out from this because backing out of a family member is not even a considerable option.
On a recent survey conducted, disabled people find it hard to change room settings even after the switches are made available to them.

MOTIVATION OF TAKING THIS PROJECT

As of the latest survey conducted by the government of India in 2011, about 21 million of the total population suffer from one or other kind of disability. India alone has 340 million people using smartphones. But the population living in smart-homes is relatively low. The main reason is that the smart-homes are considered as a luxury in India. Although, people living in metro cities can afford this luxury, population residing in remote areas can still not manage to have the minimal smart devices.

THE BASIC IDEA

Considering that even the basic smart devices i.e. smart lights and smart locks connected via wifi or ethernet cost a luxury, we decided to take a step back. A disabled person only needs to control his/her room. Therefore, instead of using wifi or internet, bluetooth will come in handy. Using Bluetooth opens a plethora of options. We can make application not only for the latest smartphones, but also for the classic feature phones. Also, using Bluetooth consumes less battery as compared to wifi.
There is an Bluetooth equipped Arduino which is connected to the lights, locks, fans etc by the means of relay. The app is installed in any Android device. The user can speak the command i.e. "Lights On" etc. The app records the audio, synthesizes it over the server and converts it into the string. The string is then parsed and is searched for

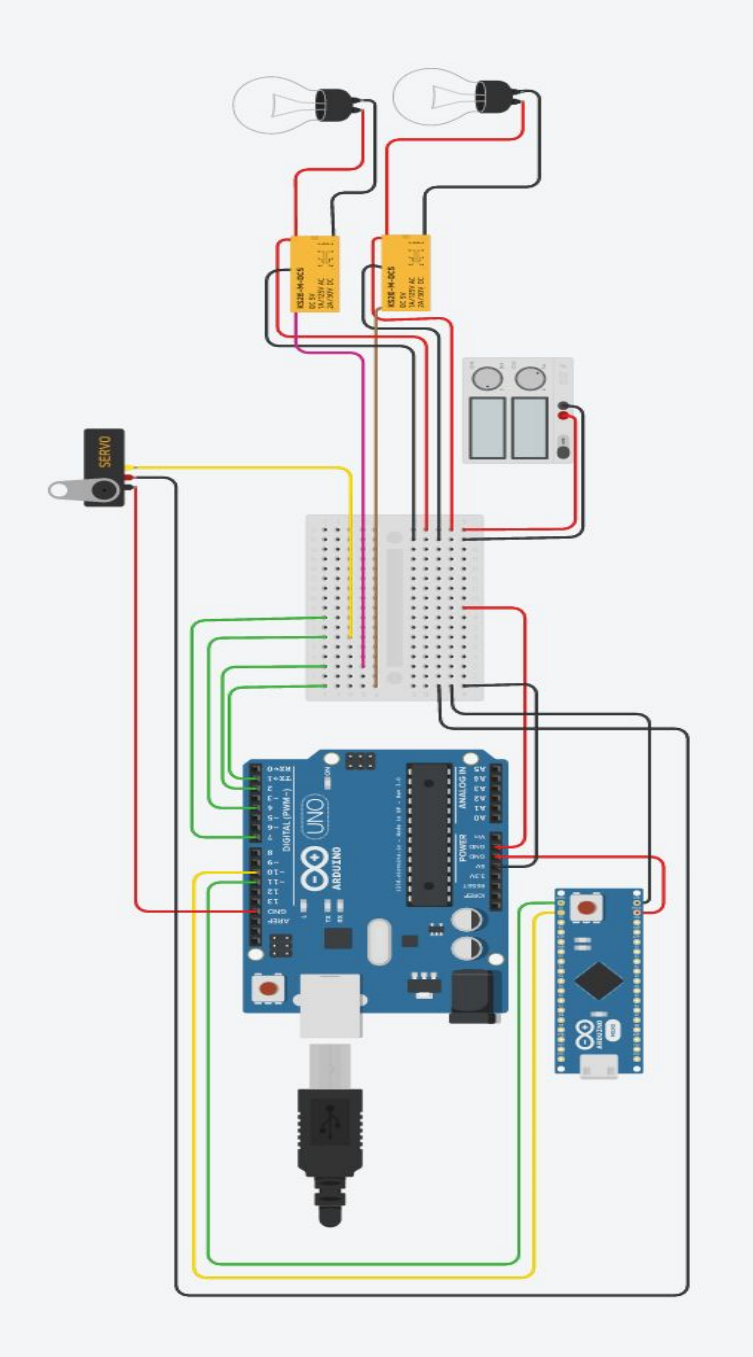
various keywords that specify the corresponding action done by the Arduino.
The most popular voice synthesizing library is “BitVoicer” which also costs more than Rs. 1000. But our design uses the combination of development tools made available by google and are free. Our application also provides an interactive UI listing all the devices which can be controlled via a single touch.
The main USP of the app is not the hardware, but the application that lets you manage the hardware.

HARDWARE & IMPLEMENTATION

The hardware components that are needed for the accomplishment of this project are

1. Arduino UNO
2. Bluetooth Module
3. 4-channel relay board
4. Servo motor
5. An Android device
6. A bulb
7. A lamp
8. Wires
9. Resistors

Circuit Diagram



←←

Step by Step working of the project

1. User runs the Android application.
2. App shows a list of all available devices and Bluetooth status of mobile.
3. App automatically connects with the Arduino.
4. App shows the status and the device with which the smartphone is connected, in this case “HC-05”
5. The user can enter the “Speech Mode” or the “Hands Mode” based on his preferences.
6. Upon entering the Speech Mode, a dialog box appears which prompts the user to speak a command.
7. The user speaks the command, e.g. “Lock the door”, “Turn the lamp on” etc.
8. App sends the data to the server and the audio is parsed and converted to a string.
9. The command corresponding to that audio is sent to the Arduino.
10. Consider the command “Unlock the door”, the Arduino runs the specific code related to this case, which is rotating a servo motor to a certain degree so that the door opens.

Comparison with other projects

There are many projects where user can control home with his smartphone but they all are way out of common man’s league. Besides, our project doesn’t even need a smartphone. A feature mobile with Bluetooth can be used to control Ardiuno i.e. your room.

Code of the project

Arduino Code

```
#include <SoftwareSerial.h>
#include <stdio.h>
#include <Servo.h>


int lamp = 9;
int light = 2;
int fan = 3;
Servo lock;



int bluetoothTX = 11 ;
int bluetoothRX = 10 ;
char receivedValue ;
```

```
SoftwareSerial bluetooth ( bluetoothTX, bluetoothRX );

void setup()
{
  Serial.begin(9600);
  Serial.println("console> ");

  pinMode(lamp, OUTPUT);
  pinMode(light, OUTPUT);
  pinMode(fan, OUTPUT);
  lock.attach(8);

  bluetooth.begin(115200);
  bluetooth.print("$$$");
  delay(100);
  bluetooth.println("U,9600,N");
  bluetooth.begin(9600);

}


void loop()
{
int brightness = 0;
int led = 0;
signed int data = 0;

 if( bluetooth.available() )
  {
        data = (int) bluetooth.read();
        Serial.println( data );                  // for debugging, show received data

        if(data == 11)
        {
        digitalWrite(light, HIGH);
        }
        else if(data == 10)
        {
        digitalWrite(light, LOW) ;
        }
        else if(data == 21)
        {
        digitalWrite(lamp, HIGH);
        }
        else if(data == 20)
```

```
{
digitalWrite(lamp, LOW);
}
else if(data == 31)
{
digitalWrite(fan, HIGH);
}
else if(data == 30)
{
digitalWrite(fan, LOW);
}
else if(data == 41)
{
lock.write(0);
}
else if(data == 40)
{
lock.write(70);
}
else if(data >= 57 && data <=144)
{
int val = 111+data;
Serial.println("lamp");
Serial.println(val);
analogWrite(lamp, data);
Serial.println("Leaving lamp");
}
else if(data>=157 && data <=233)
{
Serial.println("Entering fan");
Serial.println("With an intensity of");
Serial.println(data);
analogWrite(fan, data);
Serial.println("Leaving fan");
}
else if(data == 50)
{
digitalWrite(light, LOW);
digitalWrite(lamp, LOW);
digitalWrite(fan, LOW);
lock.write(70);
}
else if(data == 51)
{
digitalWrite(light, HIGH);
```

```
        digitalWrite(fan, HIGH);
        lock.write(0);

        }


        bluetooth.flush();          // IMPORTANT clean bluetooth stream, flush stuck data

  }

}
```

Android code

The main part of the android code is when we are making a bluetooth connection and sending data to the module.

```
void connect2Board(BluetoothDevice device)
   {
       UUID uuid = UUID.fromString("00001101-0000-1000-8000-00805f9b34fb") ;
      try {
         // ATTENTION: This was auto-generated to implement the App Indexing API.
         blsocket = device.createInsecureRfcommSocketToServiceRecord(uuid);
         blsocket.connect();
         pairedBluetoothDevice = device;
         bluetoothPaired.setText("PAIRED: "+device.getName());
         bluetoothPaired.setTextColor(getResources().getColor(R.color.green));

         Toast.makeText(getApplicationContext(), "Device paired
successfully!",Toast.LENGTH_LONG).show();
      }catch(IOException ioe)
      {
         Log.e("taha>", "cannot connect to device :( " +ioe);
         Toast.makeText(getApplicationContext(), "Could not
connect",Toast.LENGTH_LONG).show();
         pairedBluetoothDevice = null;
      }
   }
```

Where “device” is a reference of our bluetooth module i.e. HC-05.

How data is being sent to the Arduino

```
void send2Bluetooth(int device, int stat)
   {
      //make sure there is a paired device
      if ( pairedBluetoothDevice != null && blsocket != null )
      {
          try
          {
             taOut = blsocket.getOutputStream();
             taOut.write(device + stat);
             Toast.makeText(this, (device+stat) + "", Toast.LENGTH_LONG).show();

             taOut.flush();
          }
          catch(IOException ioe)
          {
             Log.e( "app>" , "Could not open a output stream "+ ioe );
          }
      }
   }
```

Screenshots from the application