VOICE COMMAND BASED HOME AUTOMATION SYSTEM

Thesis submitted in partial fulfillment of the requirements for the degree

of

Bachelor of Technology

in

Instrumentation Engineering

by

Swdwmsri Baglary (Gau-C-11/81)
Aman Chetry (Gau-C-11/69)
Mridul Deka (Gau-C-11/L-202)

Under the guidance of
Mr. Dipankar Sutradhar
(Assistant Professor I.E Department)

DEPARTMENT OF INSTRUMENTATION ENGINEERING

केन्द्रीय प्रौद्योगिकी संस्थान कोकराझार

CENTRAL INSTITUTE OF TECHNOLOGY KOKRAJHAR

(A Centrally Funded Institute under Ministry of HRD, Govt. of India)

Website: www.cit.ac.in , May 2015

DECLARATION

We hereby declare that work entitled "**VOICE COMMAND BASED HOME AUTOMATION SYSTEM**", is an authentic record of our own work carried out at Central Institute of Technology, Kokrajhar, for the award of Degree of **Bachelor of Technology** in Instrumentation Engineering. Project comprises of our original work pursued under the guidance of **Mr. Dipankar Sutradhar.** The results embodied in this report have not been submitted to any other Institute or University for any award. The information provided is correct to the best of our knowledge and belief.

Date: 18-05-2015

Aman Chetry.
(AMAN CHETRY)
University Roll No: Gau-C-11/69
University Registration No: 014747 of the
year 2011-12

Swdwmsri Baglary.
(SWDWMSRI BAGLARY)
University Roll No: Gau-C-11/81
University Registration No: 014603 of the
year 2011-12

Mridul Deka
(MRIDUL DEKA)
University Roll No: Gau-C-11/L-202
University Registration No: 081768 of the
year 2012-13

DEPARTMENT OF INSTRUMENTATION ENGINEERING

केन्द्रीय प्रौद्योगिकी संस्थान कोकराझार

CENTRAL INSTITUTE OF TECHNOLOGY, KOKRAJHAR

(An Autonomous Institute under MHRD)

Kokrajhar – 783370, BTAD, Assam, India

CERTIFICATE

This is to certify that ***Ms .Swdwmsri Baglary, Mr. Aman Chetry, Mr.Mridul Deka*** have successfully completed their project entitled "**VOICE COMMAND BASED HOME AUTOMATION SYSTEM**" which is a bonafide work carried out by themselves in partial fulfilment of **Bachelor of Technology** in Instrumentation Engineering from Central Institute of Technology, Kokrajhar. The work was carried out under my supervision during the academic session Jan-June, 2015.

(Dipankar Sutradhar)
Assistant Professor
Department of Instrumentation Engineering

CENTRAL INSTITUTE OF TECHNOLOGY, KOKRAJHAR
(An Autonomous Institute under MHRD)
Kokrajhar – 783370, BTAD, Assam, India

CERTIFICATE BY THE BOARD OF EXAMINERS

This is to certify that the project work entitled "**VOICE COMMAND BASED HOME AUTOMATION SYSTEM**" submitted by, **Ms.Swdwmsri Baglary, Mr.Aman Chetry**, **Mr.Mridul Deka** to the Department of Instrumentation Engineering in Central Institute of Technology, Kokrajhar has been examined and evaluated.

The project work has been prepared as per the regulations of Central Institute of Technology, kokrajhar and qualifies to be accepted in partial fulfillment of the requirement for the degree of Bachelor of Technology.

24/06/15
Project Co-ordinator

Board of Examiners

DEPARTMENT OF INSTRUMENTATION ENGINEERING
CENTRAL INSTITUTE OF TECHNOLOGY, KOKRAJHAR
May 2015

CERTIFICATE

This is to certify that **Ms. Swdwmsri Baglary, Mr. Mridul Deka**, and **Mr. Aman Chetry** successfully completed their project entitled "**VOICE COMMAND BASED HOME AUTOMATION SYSTEM"** which is a bonafide work carried out by themselves in partial fulfilment of Bachelor of Technology in Instrumentation Engineering from Central Institute of technology, Kokrajhar. The work was carried out under my knowledge during the academic session Jan-June, 2015.

Jeet Dutta

(Mr. Jeet Dutta)

HoD, Dept. of Instrumrntation Engineering

ABSTRACT

The main objective of this work reported in this thesis is to automate the switching of home appliances using voice command. It is designed to assist and provide support in order to fulfill the needs of elderly and physically challenged individuals. The main control system implements wireless bluetooth technology to provide access from an android phone, intended to control electrical appliances and devices in house with relatively low cost, with rich user friendly interface and ease of installation. Traditional interface with appliances is composed of keyboard or buttons which strictly demands the presence of controller in person. It is a hurdle in achieving an intelligent system for automation in home appliances. The project discussed below eliminates all the hurdles in achieving intelligent control of electrical appliances and paves a new path for future intelligent applications.

ACKNOWLEDGEMENT

We want to express our deep sense of gratitude to our guide Mr. Dipankar Sutradhar for his patience, immense knowledge, encouragement and constant support. The guidance from him has helped us throughout our project work and in writing our project report.

Besides our guide, we would also like to convey our gratitude to all the faculty of our department for their contribution in developing the knowledge of engineering in our mind through their classes in our Bachelor of Technology curriculum.

We would also like to thank all the staff of our department for their help in ways like supplying for sudden urgent demand of electronic components for the completion of this Project.

And most importantly our special thanks to our parents and friends for their never ending support throughout the project and our life.

Aman Chetry (Gau-C-11/69)

Swdwmsri Baglary(Gau-C-11/81)

Mridul Deka(Gau-C-11/L-202)

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The major section of this project work is focused on "Home automation" which is not a new concept in today's world, it is used to provide convenience for user to remotely control and monitor the appliances and it provides an efficient use of electricity. The growth in PC (personal computers), internet, mobile phone and wireless technology makes it easy for a user to remotely access and control the appliances. Services of Internet, GSM and Bluetooth technology provide method that reduces the wiring and complexity of the system. The role of home automation ranges from actuation of electrical appliances by non conventional methods to high tech security using biometrics or gesture recognition. Some residents love their home so much that they don't hesitate to pour fortune in their home automation. And in return they enjoy voice interacting smart controllers that serve them with butler like facilities. There are automation system reminding owners taking their dog to walk, record favourite TV shows, read e-mails, playing favourite songs and as far as human desire stretches out.

The second big thing in this project is Speech recognition, a technology where the system understands the words (not its meaning) given through speech. With the daily updated technology, there will be a growing need for more convenient and natural way for information exchange between human and machine. The answer for that requirement is "Speech Recognition" because speech is the most natural, effective and convenient communication method in information exchange.

The next important thing to discuss is concept of Bluetooth Technology; it is a gift by Ericsson, Nokia, Intel and Toshiba to enable cable free connection between computers, mobile phones, PDAs, printers, etc. The "Bluetooth" in Bluetooth communications protocol in these devices is named after the king, because he unified Denmark and Norway much like the technology whose goal was to unify computers and cellular phones.1

Bluetooth is a wireless protocol utilizing short-range communications technology facilitating data transmission over short distances from fixed and/or mobile devices, creating wireless personal area networks (PANs). The intent behind the development of Bluetooth was the creation of a single digital wireless protocol, capable of connecting multiple devices and overcoming issues arising from synchronization of these devices. Bluetooth provides a way to connect and exchange information between devices such as mobile phones, telephones, laptops, personal computers, printers, GPS receivers, digital cameras, and video game consoles over a secure, globally unlicensed Industrial, Scientific, and Medical (ISM) 2.4 GHz short-range radio frequency bandwidth.

The processing in this project work is the work of "Android" which is a must to know about to understand this project. The Android is basically an operating system based on the Linux kernel, and designed primarily for touch screen mobile devices such as smart phones and tablet computers. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. Android boasts a healthy array of connectivity options, including Wi- Fi, Bluetooth and wireless data over a cellular connection for example GPRS, EDGE (Enhanced Data rates for GSM Evolution) and 3G. Android provides access to a wide range of useful libraries and tools that can be used to build rich applications. In addition, Android includes a full set of tools that have been built from the ground alongside the platform providing developers with high productivity and deep insight into their applications. The use of Android is very interesting because it is based on direct/physical manipulation, like using touch inputs of swiping, tapping, pinching and reverse pinching to manipulate on-screen objects. For example adjusting the screen from portrait to landscape depending on how the device is oriented. The Android is so due to its internal hardware such as accelerometers, gyroscopes and proximity sensors to respond to additional user actions.

The last and revolutionary integral part of this project is "Arduino". It is a powerful and simplest tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.[3 http://www.arduino.cc/en/Guide/Introduction].It is inexpensive , multi-platform meaning it can run on Windows, Macintosh OSX, Linux operating systems, open source because free access, extensible hardware and software.

1.1Motivation and objectives

It is mainly focused on the elderly, physically challenged and for people who are unable to type text or face difficulties in typing. For the disable people, it is quite difficult to operate the home appliances physically or they are unable or feel uncomfortable to switch on/off the relative device. Hence a system run by least physical movement of limbs and user will best suited as a solution to above problem. Thus development of wireless system will be effective to both monitor and control such appliances.

The requirement of user for switching a device needs to be understood by such system. This can be done in two simple ways either system has to sense it or else user has to provide such information. In this work, second option is chosen to be followed and to accomplish it "voice commands" will be used as a tool to act on behalf of user; as it is evident to be using none of limb activities.

The voice commands has to reach controlling device which is physically connected to switches to turn it on/off. In order to do so we need a medium of transfer that has properties of wireless communication. The medium has to be reliable, familiar, proven ,and certified; thus "Bluetooth" will be most encouraging because we are considerably familiar with it. In addition it is long range operational and comparatively cheap and simple to WiFi communication.

As humans and electronic components do not speak the same language, hence either of language has to be converted into another language. In other words, there is requirement of operation on voice commands such that it can be understood by electronic computing devices like microcontroller and microprocessor. Also receiver of voice commands will be middleware for machine and human therefore in common terms it can be called "a voice remote controller". Perhaps more than just a remote as it has to understand what its user is commanding and make that command understandable to machine connected to other side. Thus the remote has to be intelligent, reliable and robust with its own processing power. A portable computer will be great or a mini computer that will be small and handy or how about a "Smart Mobile Phone". Besides it will have its own operating system, and it will be always around the user. But it cannot be an iPhone due to its high cost but it can be an "Android Phone" and as additional benefit Android provides Android Development Kit (ADK) available for free to develop application for running on Android platform. The Android kit is basically a Software Development Kit (SDK), collection of tools that allows the creation of application for a certain software package, software framework, hardware platform, computer system, video game console, operating system, or similar development platform [2 http://en.wikipedia.org/wiki/Software_development_kit].The Android besides being low cost give access to develop user and purpose specific application, that can be a game application for fun or a smart "voice remote control" as desired in this project. Finally the use of Android platform also widens future scope of this work because android market is exponentially blooming and pouring of technology in Android market.

As the issue of smart remote control is dealt in peace, next step taken is to dig out a smart executer for smart remote command. In addition the controller also has to fit in the low-cost bracket and also has be to reliable and Android + Bluetooth + Hardware & Software compatible. For this higher requirement for a controller, a very special, worthy and unique hybrid of controller is chosen for this project. The name goes by "Arduino" and it suits best for intelligent application like this. The induction of Arduino Board gives this project a head start in the race of better and more intelligent future deployment. This increases the overall efficiency of the project work while not costing a fortune to its user.

This project it is aimed to control electrical appliances by passing voice commands to an Arduino board using a serial Bluetooth Transreceiver. Since the vocabulary of speech in real world is ceaseless so we need a solution of plentiful resources to address this abundance of words. Hence we will be using Google Speech Recognition platform to exponentially increase the potential of application irrespective of language and pronunciation. The positive result of system can be increased multifold by choosing a phone based communication available to majority of population, so the use of World Wide Web or mobile Internet Service Provider (ISP) for android device in this domain has highest potential. As internet is a non-negotiable feature essential for smart phones hence the above project will not cost any additional service of data usage.

CHAPTER 2

BACKGROUND THEORY

The most amount of time spent on learning about development of home automation is spent on reading online journal papers, online published articles and following technical blogs. And among numerous articles the few found most significant are discussed in following section.

2.1 BLUETOOTH TECHNOLOGY

The Bluetooth specification divides the Bluetooth protocol stack into three logical groups. They are the Transport Protocol group, the Middleware Protocol group and the Application group.

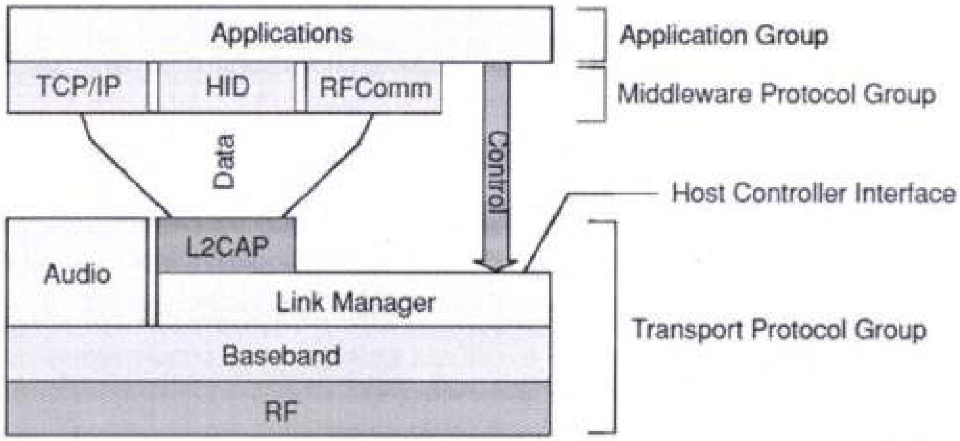


Figure.1 Bluetooth protocol stack

The Transport group protocols allow Bluetooth devices to locate each other, and to manage physical and logical links with higher layer protocols and applications. These protocols correspond to the Data-Link and Physical layers of the OSI model. The Radio, Baseband, Link Manager, Logical Link Control and Adaptation (L2CAP) layers and the Host Controller Interface (HCI) are included in the Transport Protocol group. These protocols support both asynchronous and synchronous transmission. All the protocols in this group are required to support communications between Bluetooth devices. The

Middleware Protocol group includes third-party and industry-standard protocols, as well as Bluetooth SIG developed protocols. These protocols allow existing and new applications to operate over Bluetooth links. Industry standard protocols include Point-to-Point Protocol (PPP), Internet Protocol (IP), Transmission Control Protocol (TCP), wireless application protocols (WAP), and object exchange (OBEX) protocols, adopted from Infrared Data Association (IrDA). Bluetooth SIG-developed protocols include

1) a serial port emulator (RFCOMM) that enables legacy applications to operate seamlessly over Bluetooth transport protocols,

2) a packet based telephony control signaling protocol (TCS) for managing telephony operations, and

3) a service discovery protocol (SDP) that allows devices to obtain information about each other's available services. Reuse of existing protocols and seamless interfacing to existing applications was a high priority in the development of the Bluetooth specifications, as shown in figure

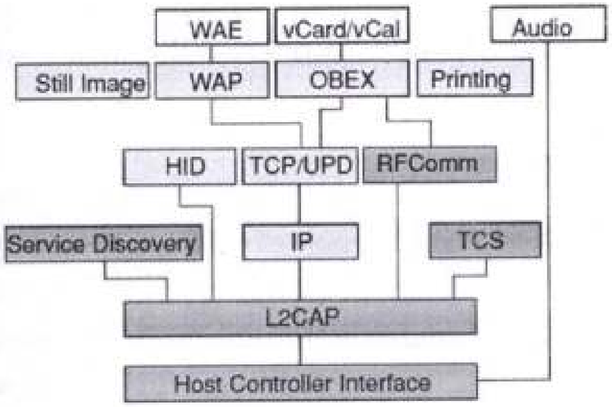


Figure.2 Interoperability

The Application group consists of actual applications that use Bluetooth links. They can include legacy applications as well as Bluetooth-aware applications. A brief discussion of the layers in the Transport group follows.

Radio layer. The specification of the Radio layer is primarily concerned with the design of the Bluetooth transceivers, discussed in detail later.

Baseband layer. This layer defines how Bluetooth devices search for and connect to other devices. The master and slave roles that a device may are defined here, as are the frequency-hopping sequences used by devices. The devices use a time division duplexing (TDD), packet-based polling scheme to share the air-interface. The master and slave each communicate only in their pre-assigned time slots. Also, defined here are the types of packets, packet processing procedures and the strategies for error detection and correction, signal scrambling (whitening), encryption, packet transmission and retransmissions.

The Baseband layer supports two types of links: Synchronous Connection-Oriented (SCO) and Asynchronous Connection-Less (ACL). SCO links are characterized by a periodic, single-slot packet assignment, and are primarily used for voice transmissions that require fast, consistent data transfer. A device that has established a SCO link has, in essence, reserved certain time slots for its use. Its data packets are treated as priority packets, and will be serviced before any ACL packets. A device with an ACL link can send variable length packets of 1, 3 or 5 time-slot lengths. But it has no time slots reserved for it.

Link Manager layer. This layer implements the Link Manager Protocol (LMP), which manages the properties of the air interface link between devices. LMP manages bandwidth allocation for general data, bandwidth reservation for audio traffic, authentication using challenge response methods, trust relationships between devices, encryption of data and control of power usage. Power usage control includes the negotiation of low power activity modes and the determination of transmission power levels.

L2CAP layer. The Logical Link Control and Adaptation Protocol (L2CAP) layer provides the interface between the higher-layer protocols and the lower-layer transport protocols. L2CAP supports multiplexing of several higher layer protocols, such as RFComm and SDP. This allows multiple protocols and applications to share the air-interface. L2CAP is also responsible for packet segmentation and reassembly, and for maintaining the negotiated service level between devices. *HCI layer*. The Host Controller Interface (HCI) layer defines a standard interface for upper level applications to access the lower layers of the stack. This layer is not a required part of the specification. Its purpose is to enable interoperability among devices and the use of existing higher level protocols and applications.

A Bluetooth transceiver is a frequency hopping spread-spectrum (FHSS) device that uses the unlicensed (worldwide) 2.4 GHz ISM (Industrial, Scientific, Medical) frequency band. The nominal bandwidth for each channel is 1MHz.

When connected to other Bluetooth devices, a Bluetooth device hops (changes frequencies) at the rate of 1600 times per second for typical use, with a residence time of 625 μ sec. When in inquiry or page mode, it hops at 3200 hops per second with a residence time of 312.5 μ sec.

A Bluetooth transceiver uses all 79 channels, and hops pseudo-randomly across all channels at a rate of 1600 hops per second for standard transmissions. It has a range of approximately 10 meters, although ranges up to 100 meters can be achieved with amplifiers. Because the transceiver has an extremely small footprint, it is easily embedded into physical devices, making it a truly ubiquitous radio link.

The Bluetooth specification uses time division duplexing (TDD) and time division multiple access (TDMA) for device communication. A single time slot is 625 microsecond in length, representing the length of a single-slot packet. At the Baseband layer, a packet consists of an access code, a header, and the payload.

All communication between devices takes place between a master and a slave, using time-division duplex (TDD), with no direct slave-to slave communication. The master will poll each active slave to determine if it has data to transmit. The slave may only transmit data when it has been polled. Also, it must send its data in the time slot immediately following the one in which it was polled.

The master transmits only in even numbered time slots, while the slaves transmit only in odd-numbered time slots. In each time slot, a different frequency channel *f* is used (a hop in the hopping sequence).

The piconet

The Bluetooth specification defines a piconet as an ad-hoc, spontaneous clustering of Bluetooth devices. In it, one device holds the role of master, while the rest of the devices are slaves. While there is no limit to the total number of slaves in a piconet, a maximum of seven slaves can be active in a piconet at any given point in time. If there are more than seven slaves, the rest of the slaves must be "parked." The maximum number of "parked" slaves is 255 per piconet with direct addressing via a *parked slave address* as defined by the SIG; however, indirect addressing of parked slaves by their specific *Bluetooth device address* is also permitted, effectively allowing any number of parked slaves. To reactivate a parked slave, the master must first place a currently active slave into a parked state.

When two Bluetooth devices enter into communication range, they will attempt to communicate with each other. If no piconet is available at that time, a negotiation process will ensue. One device will become the master (usually the device which initiated the communication) and the other will become a slave.

Any Bluetooth device can function within a piconet as a master, a slave or a bridge. These roles are temporary and exist only as long as the piconet itself exists. The master device selects the frequency, the frequency-hopping sequence, the timing (when the hops will actually occur) and the polling order of the slaves. The master is also responsible for instructing the slave devices to switch to different device states for periods of inactivity.

A master and slave must exchange address and clock information in order for the slave to join the master's piconet. Bluetooth devices each have a unique Global ID used to create a hopping pattern. The master radio shares its Global ID and clock offset with each slave in its piconet, providing the offset into the hopping pattern. A slave must be able to recreate the frequency-hopping sequence of the piconet it has joined, must know which frequency to use at which time, and must synchronize itself with the master's clock. The slave device does not actually adjust its own clock. Rather it tracks the amount of clock drift between its clock and the master's, and adjusts its transmission schedule accordingly.

A Bluetooth bridge device (or gateway) interconnects two or more piconets for multi-hop communication. The bridge communicates with all the piconets connected to it by aligning itself with the clocking of each piconet when it is ready to communicate. However, it can only communicate with one piconet at a time. Because the bridge incurs additional overhead shifting from one clocking to another to communicate with each connected piconet, it has the potential to become a bottleneck.

A bridge device may be a slave in all of the piconets to which it is connected, or it may be a master in one piconet and a slave in the others. The interconnection of two or more piconets via bridge devices results in the formation of a Bluetooth *scatternet*.

A Bluetooth device can be in one of the following states: standby, inquiry, page, connected, transmit, hold, park or sniff. A device is in *Standby* mode when it is powered on but has not yet joined a piconet. It enters the *Inquiry* state when it sends out requests to find other devices to which it might connect. A master of an existing piconet may also be in a *Page* state, sending out messages looking for devices that it can invite to join its piconet.

When successful communication is made between the master and the new device, the new device assumes the slave role, enters the *Connected* state, and receives an active address. While connected, the slave can transmit data when the master polls it. During the transmission of its data, the slave is in a *Transmit* state. At the end of its transmission, it returns to the *Connected* state.

The *Sniff* state is a low-power consumption state in which the slave "sleeps" for a pre-determined number of time slots. It wakes up at its appointed time slot for data transmission. It then returns to the inactive state until its next designated *Sniff* time slot arrives. The *Hold* state is another low-power state in which the slave is not active for a predetermined amount of time. However, there is no data transfer within the *Hold* state.

When a slave device has no data to send or receive, the master may instruct the slave to enter the *Park* state. When it enters a *Park* state, the slave relinquishes its active address in the piconet. The address will then be given to another slave that the master is reactivating from *Park* state.

2.2 SENDING AND RECEIVING DATA VIA BLUETOOTH WITH AN ANDROID DEVICE

It details on method to utilize Bluetooth in order to communicate with a microcontroller. Many smart phones today have the capability to communicate using Bluetooth. The Bluetooth is useful as a wireless communication protocol and the Bluetooth tools available to an Android developer make it useful to send and receive data to and from android device wirelessly. The issues to be addressed a solution are Android must support Bluetooth, and if it does, when Bluetooth is turned on. Then, it must pair and connect with the Bluetooth module on the Arduino. Finally, the Android must actually send and receive data.

The first programming step in Android begins with Android SDK supporting platform. In our project we are depending on Eclipse, a Java Development Kit (JDK), program codes are using java platform. The first programming step is to create a new Android Application Project in Eclipse. Doing so will generate code like this

```
package com.example.bluetoothconnection;

import droid.support.v7.app.ActionBarActivity;
import android.os.Bundle;

public class MainActivity extends ActionBarActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

The second thing to do is determine if the Android device supports Bluetooth. To do this, create a BluetoothAdapter object using the function getDefaultAdapter(). If this returns null, then the Android device does not support Bluetooth. If getDefaultAdapter does not return null, then the Android supports Bluetooth.

```
mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
if (mBluetoothAdapter == null) {
 // Device does not support Bluetooth
}
```

The next step is to determine if Bluetooth is enabled, and if it is not enabled, to enable it with following codes

```
if (!mBluetoothAdapter.isEnabled()) {
    Intent enableBtIntent= new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
    startActivityForResult(enableBtIntent, 1);
}
```

Next, the program has to retrieve the actual Bluetooth device it will communicate with, in this case the Arduino's Bluetooth module. The BluetoothAdapter's getBondedDevices() function will do this. This function puts all of the Android's currently-paired devices into a storage structure called a "Set". Since only the Arduino's Bluetooth module is paired with the Android (which was done at the beginning of these instructions), only this device will be in the Set. Now we assign this device to a BluetoothDevice variable by following codes.

```
Set<BluetoothDevice> pairedDevices = mBluetoothAdapter.getBondedDevices();
if (pairedDevices.size() > 0) {
   for (BluetoothDevice device : pairedDevices) {
       mDevice = device;
   }
 }
```

The next objective is to form the connection between the Android and the Arduino. This work should take place in separate thread. This is because forming a connection can block a thread for a significant amount of time. Up until now, all of the program's code has been written in the main thread, or "user interface thread" (UI thread). The UI thread should never be blocked. Therefore, we create a new thread class where the connection will form following are the code to accomplish this. It is added as an inner class of the main class.

```
private class ConnectThread extends Thread {
    private final BluetoothSocket mmSocket;
    private final BluetoothDevice mmDevice;
    private static final UUID MY_UUID = UUID.fromString("00001101-0000-1000-8000-
    00805f9b34fb");

    public ConnectThread(BluetoothDevice device) {
        BluetoothSocket tmp = null;
        mmDevice = device;
        try {
            tmp = device.createRfcommSocketToServiceRecord(MY_UUID);
         } catch (IOException e) { }
         mmSocket = tmp;
    }

    public void run() {
        mBluetoothAdapter.cancelDiscovery();
        try {
            mmSocket.connect();
         } catch (IOException connectException) {
            try {
                mSocket.close();
            } catch (IOException closeException) { }
            return;
         }
    }
    public void cancel() {
        try {
            mmSocket.close();
        } catch (IOException e) { }
    }
}
```

This thread requires a BluetoothDevice as a parameter and uses it to create a BluetoothSocket. This socket is what Bluetooth uses to transfer data between devices. The UUID used in above block of code tells the socket that data will be transferred serially, which means one byte at a time. In order to use this thread, add code in following block to the end of onCreate().

```
mConnectThread = new ConnectThread(mDevice);
mConnectThread.start();
```

The code will now connect the Arduino's Bluetooth module with the Android. The last objective is to send and receive data using this connection. Like connecting, transferring data is time-intensive and can block the thread, so this work should also take place in a separate thread. Another inner thread class is added which is most complex chunk of code in entire program. The code block is as following

```
private class ConnectedThread extends Thread {
    private final BluetoothSocket mmSocket;
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;

    public ConnectedThread(BluetoothSocket socket) {
        mmSocket = socket;
        InputStream tmpIn = null;
        OutputStream tmpOut = null;
        try {
            tmpIn = socket.getInputStream();
            tmpOut = socket.getOutputStream();
        } catch (IOException e) { }

        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }
    public void run() {
    byte[] buffer = new byte[1024];
    int begin = 0;
    int bytes = 0;
    while (true) {
        try {
           bytes += mmInStream.read(buffer, bytes, buffer.length - bytes);
           for(int i = begin; i < bytes; i++) {
               if(buffer[i] == "#".getBytes()[0]) {
                  mHandler.obtainMessage(1, begin, i, buffer).sendToTarget();
                  begin = i + 1;
                  if(i == bytes - 1) {
                  bytes = 0;
                  begin = 0; }
                  }
               }
            } catch (IOException e) {
              break;
            }
    }
}
public void write(byte[] bytes) {
    try {
       mmOutStream.write(bytes);
    } catch (IOException e) { }
}
   public void cancel() {
     try {
            mmSocket.close();
            } catch (IOException e) { }
        }
    }
```

The thread requires a BluetoothSocket as a parameter and uses it to create an InputStream and and OutputStream. The InputStream is used for reading data coming from the Arduino, and the OutputStream is used for sending data to the Arduino.

Writing data is easier than reading data. The only data this program has to send is the asterisk character, for when it wants to receive a random number from the Arduino. To do so, convert "*" to a byte using the String class's getBytes() function. Then, call the ConnectedThread's write(byte[]) function, using the converted "*" as the parameter. This will put the "*" into the Android's OutputStream, which will send "*" to the Arduino's InputStream. In turn, the Arduino will send a random number between 0 and 999 followed by "#", which will arrive in the Android's InputStream.

Reading the InputStream is much tougher than writing. This is because multiple samples of data can arrive in the InputStream between readings. This is why the "#" terminating character is required. When data is read in the InputStream, it is moved to the end of a buffer. After this happens, the buffer is iterated through, beginning at the location after the last "#" was found (this may have happened before this InputStream reading). If another "#" is found, the program "handles" the data between the previously-found "#" and this new "#". This process continues until the end of the buffer. If the last character in the buffer is a "#", the buffer is cleared after the data between the last 2 "#"s is handled. When the program obtains the data between 2 "#"s, it is often desired to use it in the UI thread. In order to send data from a separate thread to the UI thread, a Handler must be used.

The following code is for the handler which manages this data and adds it to main class.This Handler converts the entire buffer's contents from a byte array to a string, and then saves the correct portion of it into writemessage. This writemessage string stores the random number generated and sent from the Arduino.

```
Handler mHandler = new Handler() {
    @Override
    public void handleMessage(Message msg) {
        byte[] writeBuf = (byte[]) msg.obj;
        int begin = (int)msg.arg1;
        int end = (int)msg.arg2;

        switch(msg.what) {
            case 1:
                String writeMessage = new String(writeBuf);
                writeMessage = writeMessage.substring(begin, end);
                break;
        }
    }
};
```

The program is almost complete. The last step is to create and start the data thread. To do this add the code in following code block to the end of ConnectThread's run() function.

```
mConnectedThread = new ConnectedThread(mmSocket);
mConnectedThread.start();
```

With the completed code, the program will run correctly. When the Android program is first opened, it determines if Bluetooth is supported by the device, as well as whether or not Bluetooth is enabled. After this, it finds the Arduino's Bluetooth module paired with the Android and uses it to form a connection. This connection process creates a socket that both devices will use to receive and transmit data. When the Android sends the Arduino a "*" character, the Arduino sends back a random number between 0 and 999, followed by a "#" character. The Android parses the incoming data and stores only the number portion in a string called writemessage. Thus, data transmission using Bluetooth is achieved.

CHAPTER 3

DESIGN OF THE PROJECT

The project is designed to be affordable to everyone, cheap, easy to install, as easy and interesting as it can get. The processing is based on mobile application which efficiently converts the voice command to text and passes it to the Bluetooth medium so no wired communication between the phone remote, appliances control switch is required. Moreover being based on Bluetooth medium distance between remote and controller will be least bothering factor and thus adds up to advantage list.

The working of this project is designed based on its effectiveness and accuracy of data acquisition from human user and conversion of that instruction into machine understanding language. There is a great role of communication between devices thus there is a central hub for information handling and processing. The processing of information is so complex that it cannot be delivered by normal computational process. The project involves proper functioning of both hardware and software interfaces in complete synchronisation, it is a fusion of cutting edge technologies from wireless communication, multi-platform software development kits, advanced level of programming concepts and language.

The central controller in this project has to perform data transfer between devices, communicate as both listener and speaker, sync time perfectly between communications and finally process the information. It has to act as a central hub of information from Android phone through Bluetooth and process voice commands to produce respective output.

3.1 OVERALL DESIGN

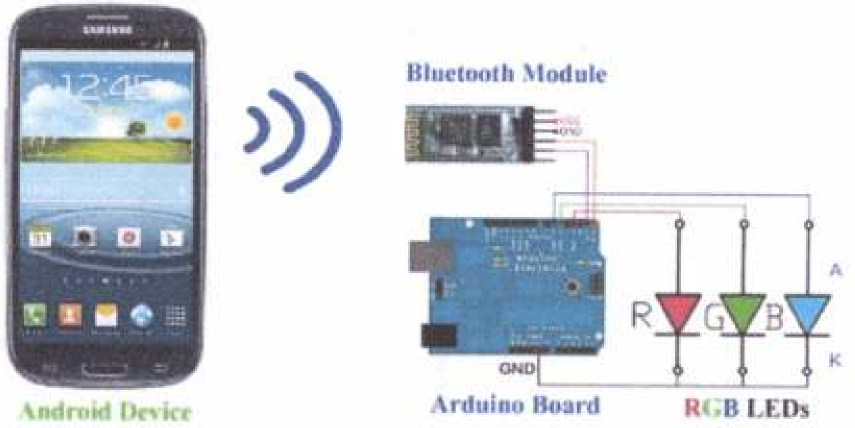
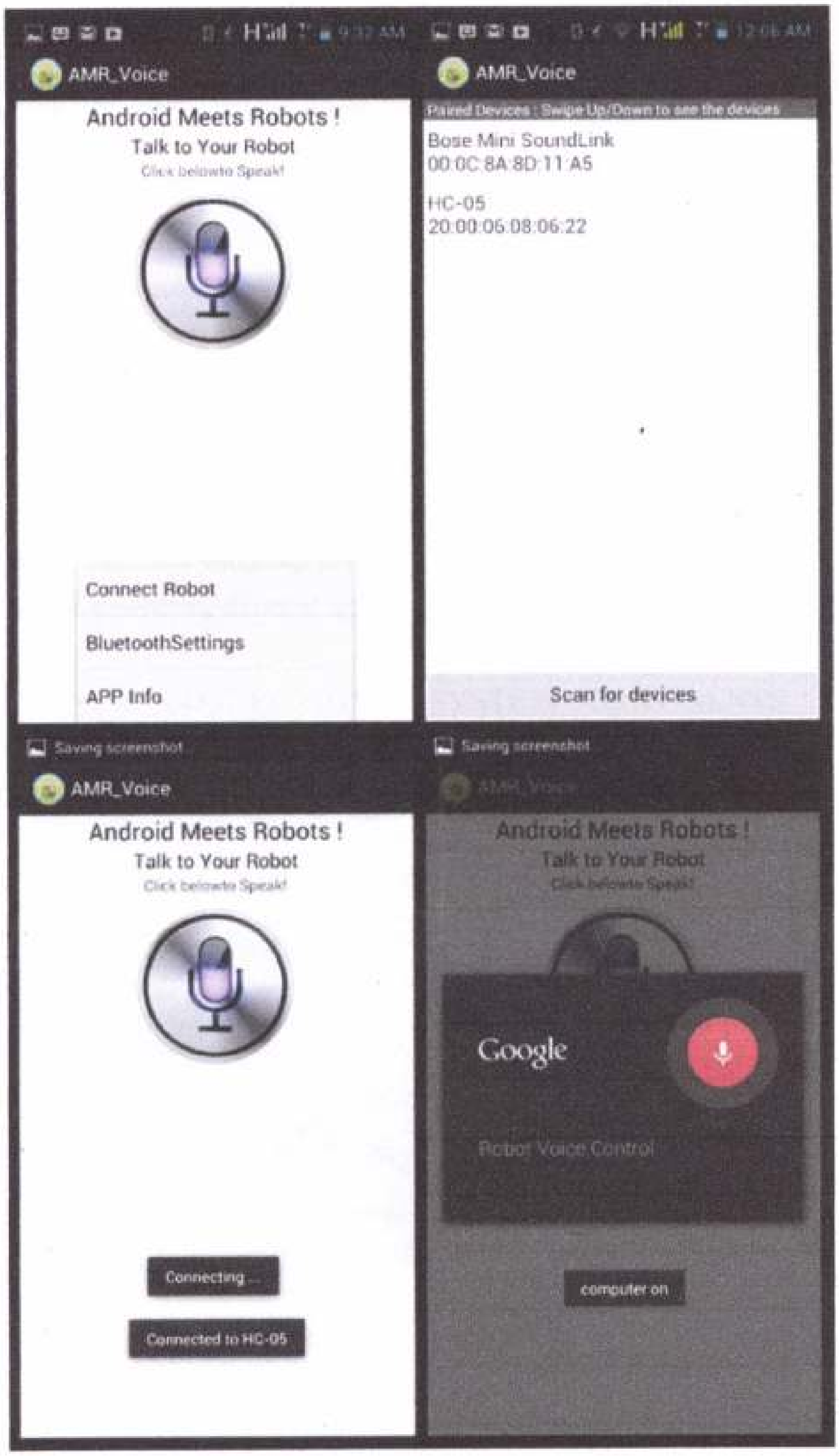


Figure.3 Overall design

The Android phone is utilized as a smart remote receiving voice commands from user. The phone uses an application "Android Meets Robot (AMR)" developed for Android platform which listen to voice commands, when a microphone icon is activated by human touch. The voice is then processed using Google Speech Recognition as the AMR access the available internet connection. The AMR then converts the spoken words into a "string" containing speech between * and # character for e.g. *please#. The special character * and # play a very crucial role in processing of spoken works, it helps to distinguish between message and garbage.

The AMR uses Bluetooth adapter of Android phone to wirelessly transmit that string to controller of electrical appliances. The first thing AMR does on its starting is enabling Bluetooth adapter of the android device it is installed. The AMR has to be paired with a Bluetooth device before AMR begins to listen to user voice commands.



Bose Mini SoundLink
00:0C:8A:8D:11:A5

HC-05
20:00:06:08:06:22

Figure.4 Screenshot of AMR

The controller is this project is an "Arduino Uno Board" which has IDE Integrated Development Environment for handling hardware using flexible programming methods. The Arduino board is mounted withAtmel328 microcontroller and it will be loaded with program codes to process strings from Bluetooth paired AMR app. The program will be comparing strings from AMR with preloaded voice commands hence when a voice command matches, program will actuate relays to switch AC devices that may be air conditioner, music system, computers etc.

The Arduino has in built serial communication transmitter (Tx) and receiver (Rx) pins and has software libraries for using serial pins. The serial pins are physically wired to Tx and Rx pin with Rx and Tx respectively of serial Bluetooth transreceiver HC05. The AMR wirelessly passes the string to HC05 which is then passed to Arduino board for switching electrical appliances.

3.2 INTER CONNECTION OF SYSTEM COMPONENTS

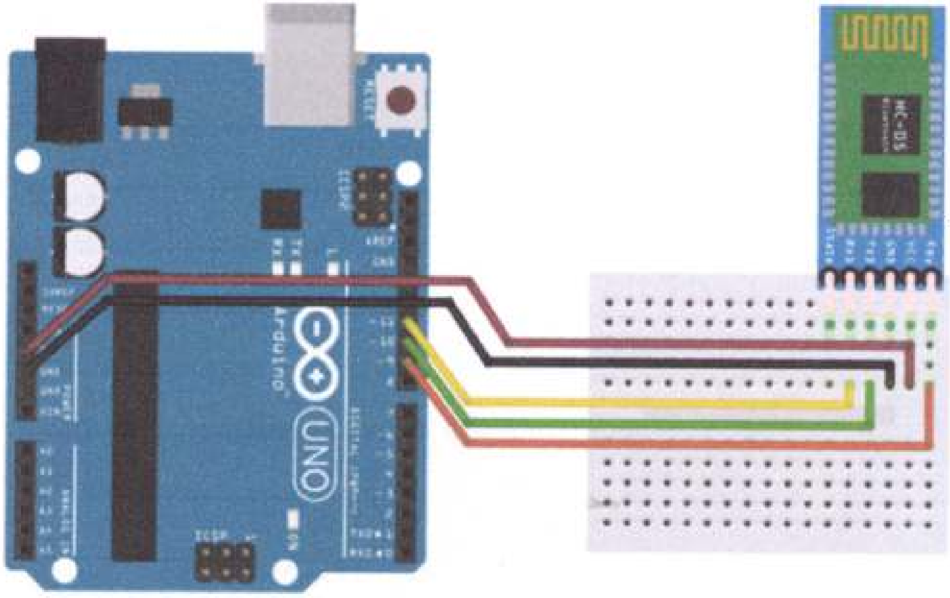


Fig.5 Interconnection of system components

3.3 BLUETOOTH MODULE HC 05 CONNECTION

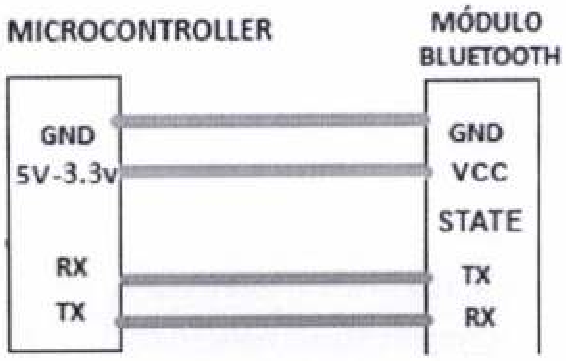


Figure.6 Bluetooth HC-05 connection

3.4 PROGRAMMING OF ARDUINO BOARD

```
#define Relay1  4        //Connect Relay no. 1 To Pin #4
#define Relay2  5        //Connect Relay no. 2 To Pin #5
#define Relay3  6        //Connect Relay no.3 To Pin #6
#define Relay4  7        //Connect Relay no.4 To Pin #7
String voice;
void setup() {
 Serial.begin(9600);
 pinMode(Relay 1, OUTPUT);
 pinMode(Relay 2, OUTPUT);
 pinMode(Relay 3, OUTPUT);
 pinMode(Relay 4, OUTPUT);
}
//------------------Call A Function----------------------//
void allstart(){
   digitalWrite(Relay 1, HIGH);
   digitalWrite(Relay 2, HIGH);
   digitalWrite(Relay 3, HIGH);
   digitalWrite(Relay 4, HIGH);
  }
//-----------------------------------------------------------//
```

```
void allstop(){
   digitalWrite(Relay 1, LOW);
   digitalWrite(Relay 2, LOW);
   digitalWrite(Relay 3, LOW);
   digitalWrite(Relay 4, LOW);
   }
//---------------------------------------------------------------//
void loop() {
 while (Serial.available()){  //Check if there is an available byte to read
 delay(10);                   //Delay added to make thing stable
 char c = Serial.read();  //Conduct a serial read
 if (c == '#') {break;}  //Exit the loop when the # is detected after the word
 voice += c;                  //Shorthand for voice = voice + c
  }
 if (voice.length() > 0) {
  Serial.println(voice);
 //----------Control Multiple Pins/ Relays----------//
 if(voice == "*all lights start") {allstart();}  //Turn on All Relay Pins (Call Function)
 else if(voice == "*all lights stop"){allstop();} //Turn off  All Relay Pins (Call Function)
 //----------Turn On One-By-One----------//
 else if(voice == "*white light start") {digitalWrite(Relay 1, HIGH);}
 else if(voice == "*yellow light start") {digitalWrite(Relay 2, HIGH);}
 else if(voice == "*blue light start") {digitalWrite(Relay 3, HIGH);}
 else if(voice == "*red light start") {digitalWrite(Relay 4, HIGH);}
 //----------Turn Off One-By-One----------//
 else if(voice == "*white light stop") {digitalWrite(Relay 1, LOW);}
 else if(voice == "*yellow light stop") {digitalWrite(Relay 2, LOW);}
 else if(voice == "*blue light stop") {digitalWrite(Relay 3, LOW);}
 else if(voice == "*red light stop") {digitalWrite(Relay 4, LOW);}
//-----------------------------------------------------//
voice="";}} //Reset the variable after initiating
```

3.5 OVERVIEW OF ARDUINO UNO R3 (REVISION 3)

The Arduino Uno is a microcontroller board based on the ATmega328. The Uno differs from all preceding boards in it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pin	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40mA
DC Current for 3.3V Pin	50mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

3.5.1 PIN DIAGRAM

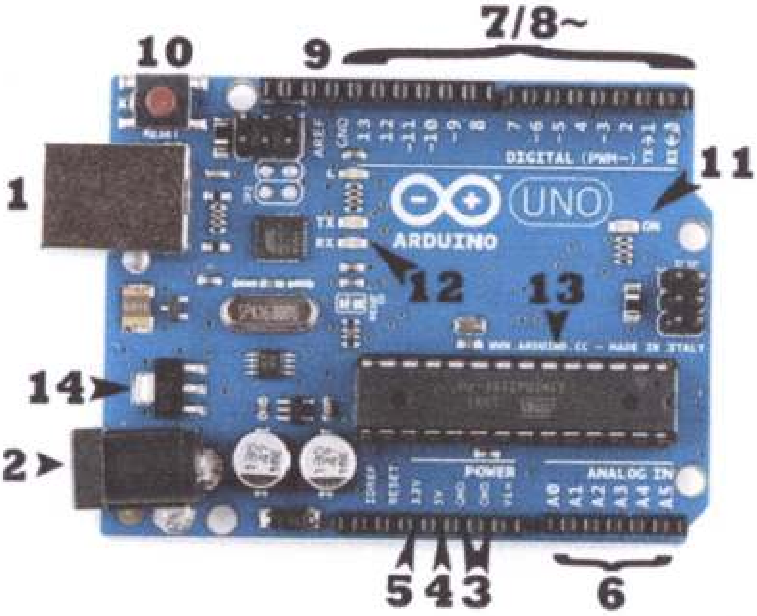


Figure.7 Arduino Pin diagram

3.6 BLUETOOTH DEVICE

Bluetooth Technology is useful when transferring information between two or more devices that are near each other in low-bandwidth situations. It exists in many products, such as telephones, printers, modems and headsets. It is commonly used to transfer sound data with telephones (i.e. with a Bluetooth headset) or byte data with hand-held computers (transferring files).Bluetooth protocols simplify the discovery and setup of services between devices.

3.6.1OVERVIEW OF BLUETOOTH TRANSRECEIVER MODULE

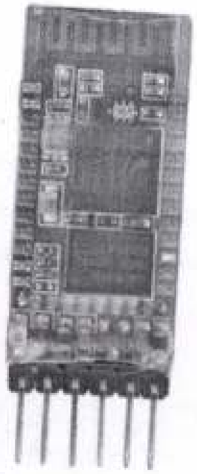


Figure.8 Serial Bluetooth transreceiver

HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. Serial port Bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Blue core 04-External single chip Bluetooth system with CMOS technology and with AFH (Adaptive Frequency Hopping Feature).

Specification

- Typical -80dBm sensitivity.
- Up to +4dBm RF transmit power.
- Low Power 1.8V Operation, 3.3 to 5 V I/O.
- PIO control.
- UART interface with programmable baud rate.
- With integrated antenna.
- With edge connector.

3.7 FOUR CHANNEL RELAY BOARD

Relay

In this project the primary interface between the systems with the electrical appliance is the relay. We are using a 4-channel relay board for our project.

- A relay is an electrically operated switch.
- Relays are used where it is necessary to control a circuit by a low-power signal.
- Relays protect electrical circuits from overload or faults.

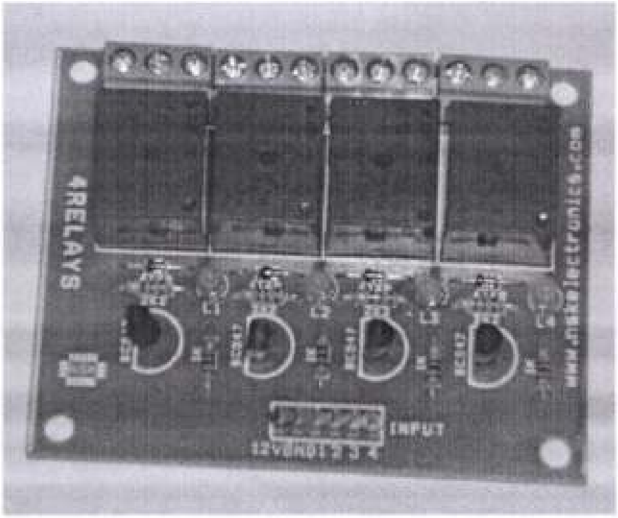
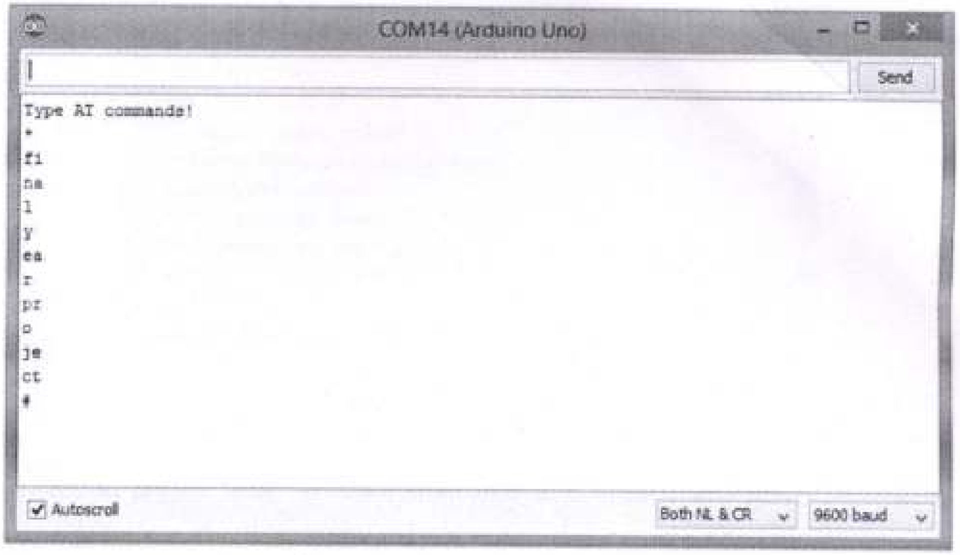


Figure.9 Four channel relay board

CHAPTER 4

RESULTS AND DISCUSSIONS

The first test on AMR is performed to see what string AMR is sending to Arduino. It was observed that whatever voice command AMR is receiving it is prefixed by * and terminated with # character as shown in figure below



```
Type AT commands!
*
fi
na
l
y
ea
r
pr
o
je
ct
#
```

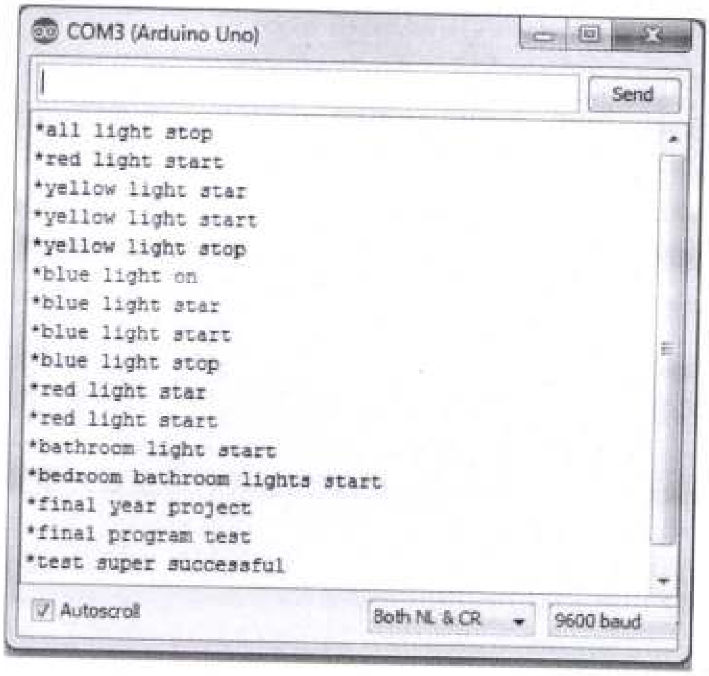
☑ Autoscroll

Both NL & CR

9600 baud

The program code was printing string two characters per line taking space key hit as one character. The print of this characters illustrate the working of program code, it indicates that program is holding only two characters at a time. This limits the accepted number of characters for voice command or in other words the AMR is successfully communicating serially with Bluetooth and passing voice commands as string to Arduino but limited of holding only two characters at a time.

This limitation is overcome by using another set of codes which are presented above and a figure supporting our claim is provided below.



```
*all light stop
*red light start
*yellow light star
*yellow light start
*yellow light stop
*blue light on
*blue light star
*blue light start
*blue light stop
*red light star
*red light start
*bathroom light start
*bedroom bathroom lights start
*final year project
*final program test
*test super successful
```

☑ Autoscroll   Both NL & CR   9600 baud

From the figure it can be seen that a lot longer string can be hold at a time. This hold of sentences can be used to trigger physical world action by using if else statement. In Arduino programming, sentence from user is compared with string declared in programming e.g. "red light start" and when result turns to be positive a corresponding pin is made high which actuates a relay and finally load connected to relay. However due care has to be taken for declared string as they are case sensitive say "RED light start" and "red light start" sounds alike but as a string they are totally different. Also it is to be noted that words separation is made by hitting space key e.g. the appropriate string to turn on a red light will be "red light start" instead of "redlighton" or "red_light_on".

Due to IDE of Arduino the voice command string is not limited to only one type of string, as we are using Google Speech Recognition, it has ceaseless number of words phrases and sentences imaginable. For example "cat" or "frog" or "help" can also be used to turn on "red light" or any electrical appliance. The use of appropriate words is completely on wisdom of program writer, instructions can be made extremely simple, precise, appropriate or super complex, irrelevant or foolish.

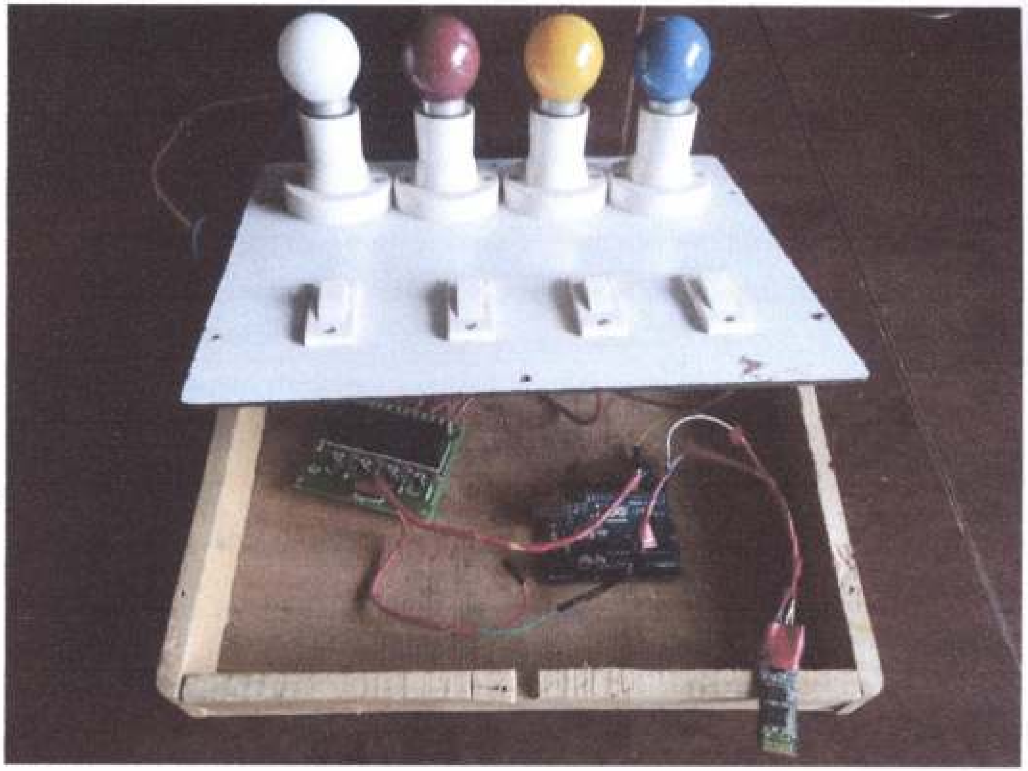


Figure 12 Indegeniously developed voice controlled home automation system

CHAPTER 5

CONCLUSION AND FUTURE WORK

The final result of this project is we are able to voice control four ac electrical appliances running on household ac supplies. To distinguish between appliances we have used four coloured bulbs, however any type of ac domestic appliance can be controlled. The system is tested in noisy environments and found to be working non significantly affected, it has been tested number of times by increasing distance between system and phone remote, it works irrespective of obstructions encountered in the line of sight between control setup and phone remote.

There have been minor mistakes in speech recognition like extra 's' detected during utterance of "lights off" e.g. "speaker says " red light stop" but AMR detects "red lights top". However our regular utterance on a particular phone has made AMR recognize our voice command with very high success. As in average out of 13 instructions given to AMR it has been able to recognize it 11 times in the first time irrespective of our speed in utterance and noisy environment.

For the future this project has numerous hidden potential, the most significant and effective version of this will have replaced Bluetooth by internet and devices can be controlled from any point in the world. Besides the fusion of communication devices and home automation will also have some pieces in security application. It is based on speech recognition but it can be modified as voice recognizer as well using modern programming approach. The application of both voice and speech recognition will open a new portal of more secure communication which be comfortable and as per convenience of each different individual.

Also same environment of smart phone and Integrated Development software can be utilized to design a gesture recognition app that can help elderly and physically challenge person to much greater extent.

BIBLIOGRAPHY/REFERENCES

1. Patricia McDermott-Wells; 0278-6648/04/$20.00 © 2004 IEEE
2. YONG LI "Centralized control of home electric appliances based on short range wireless technology" Inst. of Mech.Manuf.@Autom,South China Univ. of Technol. Guangzhou, China Date of conference:22-24 June 2009 INSPEC Accession Number:10837559
3. Ismail, N.H;Tukiran, Z;Shamsuddin, N.N;Saadon, EIS
   Android Based Home Automation System via Bluetooth
   Control System, Computing and Engineering (ICCSCE), 204
   Publication year: 2014
4. Android-based home door locks application via Bluetooth for disabled people Ismail, N.H.; Tukiran, Z.; Shamsuddin, N.N.; Saadon, E.I.S. Control System, Computing and Engineering (ICCSCE), 2014 IEEE International Conference on DOI: 10.1109/ICCSCE.2014.7072720 Publication Year: 2014, Page(s): 227- 231 IEEE CONFERENCE PUBLICATIONS
5. Designing Android Applications Using Voice Controlled Commands: For Hands Free Interaction with Common Household Devices Mulhern, N. ; McCaffrey, N. ; Beretta, N. ; Chabot, E. ; Ying Sun Bioengineering Conference (NEBEC), 2013 39th Annual Northeast DOI: 10.1109/NEBEC.2013.13 Publication Year: 2013 , Page(s): 265- 266 Cited by: Papers (1) IEEE CONFERENCE PUBLICATIONS

6. Research on Bluetooth expansion of communication based on android system
   Jianzhong Wu ; Meimei Huo ; Jianping Cai ; Minghui Wu ; YunWu Wang
   World Automation Congress (WAC), 2012
   Publication Year: 2012 , Page(s): 1- 4
   IEEE CONFERENCE PUBLICATIONS

7. Approach for performance investigation of different Bluetooth modules and communication modes Warriach, E.U.; Witte, S. Emerging Technologies, 2008. ICET 2008. 4th International Conference on DOI: 10.1109/ICET.2008.4777494
   Publication Year: 2008 , Page(s): 167- 171
   IEEE CONFERENCE PUBLICATIONS

8. Design and Research about Voice Communication System Based on Bluetooth Module Wang Zhengyan ; Yang Wei ; Shao Xingling ; Zhai Hongqiang
Measuring Technology and Mechatronics Automation (ICMTMA), 2011 Third International Conference on Volume:1
DOI: 10.1109/ICMTMA.2011.228
Publication Year: 2011 , Page(s): 911- 914
IEEE CONFERENCE PUBLICATIONS

9. Design and Implementation of a Bluetooth based General Purpose Controlling Module Wijetunge, S.P. ; Wijetunge, U.S. ; Peiris, G.R.V. ; Aluthgedara, C.S. ; Samarasinghe, A.T.L.K.
Information and Automation for Sustainability, 2008. ICIAFS 2008. 4th International Conference on
DOI: 10.1109/ICIAFS.2008.4783997
Publication Year: 2008 , Page(s): 206- 211
Cited by: Papers (1)
IEEE CONFERENCE PUBLICATIONS