

BLUETOOTH BASED HOME AUTOMATION USING ANDROID SMARTPHONE

*A Graduate Project Report submitted to Manipal University in partial
fulfilment of the requirement for the award of the degree of*

BACHELOR OF TECHNOLOGY In Electronics and Communication Engineering

Submitted by:

**JAYESH MEHTA
110907009**

Under the guidance of

H.Srikanth Kamath

**Assistant Professor - Selection Grade
Dept. of Electronics & Communication**

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
MANIPAL INSTITUTE OF TECHNOLOGY**

**(A Constituent College of Manipal University)
MANIPAL – 576104, KARNATAKA, INDIA**



MAY 2016





DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

MANIPAL INSTITUTE OF TECHNOLOGY

(A Constituent College of Manipal University)
MANIPAL – 576 104 (KARNATAKA), INDIA



Manipal
12-05-2016

CERTIFICATE

This is to certify that the project titled **BLUETOOTH BASED HOME AUTOMATION USING ANDROID SMARTPHONE** is a record of the bonafide work done by **JAYESH MEHTA** (*Reg. No. 110907009*) submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Engineering (BE) in **ELECTRONICS AND COMMUNICATION ENGINEERING** of Manipal Institute of Technology Manipal, Karnataka, (A Constituent College of Manipal University), during the academic year 2015-2016.

H.Srikanth Kamath
Assistant Professor - SI Grade
Project Guide

Prof. Dr. Somashekara Bhat
HOD, E & C.
M.I.T, MANIPAL

ACKNOWLEDGMENTS

On the occasion of successful completion and submission of report I would like to express our deep gratitude to the **Director, Dr. G.K Prabhu**, Manipal Institute of Technology and **Prof.(Dr.) Somashekara Bhat , Head of Department**, Electronics and Communications Department for providing us with a platform that enabled us to complete this project.

I are deeply indebted to **Mr. H.Srikanth Kamath**, internal department guide for devoting time to guide me with various aspects of the project. His guidance contributed immensely in making this a rich learning experience for me without whom this project would not have been possible.

I would also like to thank all the members in the panel, **Dr. Kumara Shama, Mr. Vijay S R, Mr. Shailendra Kumar Tiwari, Ms. Mary Ann George** for their constant guidance and helping me in the improvement of the project.

Finally I would take this opportunity to thank all the professors of the department whose teachings I will carry with me throughout our life.

ABSTRACT

Due to the advancement of wireless technology, there are several different technologies introduced such as GSM, WIFI, ZIGBEE, and Bluetooth. Each of the connection has their own unique specifications and applications. Among the four popular wireless connections that often implemented in Home Automation project, Bluetooth is being chosen with its suitable capability. Bluetooth with globally available frequencies of 2.4GHz is able to provide connectivity up to 100 meters at speed of up to 3Mbps depending on the Bluetooth device class. This project presents the design and implementation of a low cost but yet flexible smartphone based home automation system. The design is based on a standalone Arduino Bluetooth board and the home appliances are connected to the input/output ports of this board via relays. The communication between the Smartphone and the Arduino BT board is wireless. This system is designed to be low cost and scalable allowing variety of devices to be controlled with minimum changes to its core. The main objective of home automation and security is to help handicapped and old aged people who will enable them to control home appliances.

For this Home automation project a system outline is proposed keeping the objectives in consideration. The Arduino microcontroller is studied using the Arduino Data sheet. All ports/pins on the Arduino Microcontroller are carefully studied. We use the Arduino Integrated development environment to program the microcontroller. The programming language used is C/C++. Bluetooth module HC-05 is used for communication between the Arduino and Android Smartphone. Data sheet for HC-05 is also referred. With all the knowledge from the above studies the hardware is connected according the required specifications and Arduino is programmed. For testing purposes we use a PC to send commands to Arduino over Bluetooth.

The hardware is connected, we are successfully able to establish a wireless connection between the Arduino Microcontroller and PC over Bluetooth. We use an emulator on the PC to send commands to the Serial port of the Arduino. The Arduino executes those commands to turn on/off devices.

With the above system we are able to control different home appliances with a single remote i.e. android smartphone. The control is almost instantaneous and reliable. In this projects we have used software tools which include Arduino IDE, Tera Term Emulator, Android SDK, Android Studio, and Eclipse IDE.

LIST OF TABLES

Table No	Table Title	Page No
1.1	Project Schedule	3
3.1	Technical Specification Of UNO R3	13

LIST OF FIGURES

Figure No	Figure Title	Page No
2.1	Basic System Architecture	5
2.2	System Architecture Circuit	6
3.1	Arduino Pin diagram	14
3.2	Arduino – HC05 Connection	15
3.3	HC-05 module	16
3.4	HC-05 Pin Out	17
3.5	2- Channel Relay	18
3.6	Arduino IDE	18
3.7	Android Studio : Android Remote Activity	20
3.8	Android Studio : Layout.XML	21
3.9	Android Studio : Layout HTML Code	22
3.10	Microcontroller program Flowchart	24
3.11	Android GUI	25
4.1	TeraTerm Window	26
4.2	Android App Running on Smartphone	27
4.3	Project Hardware Snapshot	28

Contents			
			Page No
Acknowledgement			i
Abstract			ii
List of Figures			iii
Chapter 1		INTRODUCTION	
	1.1	Introduction	1
	1.2	Motivation	1
	1.3	Objectives	2
	1.4	Components Used	3
	1.5	Project Schedule	4
Chapter 2		BACKGROUND THEORY	
	2.1	System Architecture	5
	2.2	Development Platform	6
	2.2	Bluetooth Connectivity	9
Chapter 3		METHODOLOGY	
	3.1	Introduction	12
	3.2	Review of Past Home Automation systems	12
	3.3	Hardware Design	13
	3.4	Software Design	17
Chapter 4		RESULT ANALYSIS	
	4.1	Introduction	25
	4.2	Results	25
Chapter 5		CONCLUSION AND FUTURE SCOPE	
	5.1	Work Conclusion	29
	5.2	Future Scope of Work	30
REFERENCES			31
ANNEXURES			32
PROJECT DETAILS			43

CHAPTER 1

INTRODUCTION

1.1: Introduction

The “Home Automation” concept has existed for many years. The terms “Smart Home”, “Intelligent Home” followed and has been used to introduce the concept of networking appliances and devices in the house. Home automation Systems (HASs) represents a great research opportunity in creating new fields in engineering, architecture and computing. HASs becoming popular nowadays and enter quickly in this emerging market.

Home automation refers to the use of computer and information technology to control home Appliances and features (such as windows or lighting). Systems can range from simple remote control of lighting through to complex computer/micro-controller based networks with varying degrees of intelligence and automation. Home automation is adopted for reasons of ease, security and energy efficiency. In modern construction in industrialized nations, most homes have been wired for electrical power, telephones, TV outlets (cable or antenna), and a doorbell. Many household tasks were automated by the development of specialized Appliances. For instance, automatic washing machines were developed to reduce the manual labor of cleaning clothes, and water heaters reduced the labor necessary for bathing. However, end users, especially the disabled and old aged due to their complexity and cost, do not always accept these systems.. Our Aim is to design a kit that can be used for controlling AC Loads from Android phone by using Arduino as microcontroller. This system is designed to be low cost and scalable allowing variety of devices to be controlled with minimum changes to its core.

1.2: Motivation

Based on the study of Home automation System projects done by researchers and developers, they have implemented Microcontroller in wireless Home Automation Systems. The system

implemented microprocessor and GSM (SMS) control method by a GSM modem. The system mentioned as low cost but the cost of GSM modem is not considered. Also, long term cost by the GSM is not fully accepted by every user. This problem can be overcome by using local networking or by remote control.

The capabilities of Bluetooth are more than enough to be implemented in the design. Also, most of the current laptop/notebook or cell phones are come with built-in Bluetooth adapter. It will indirectly reduce the cost of this system.

Since the project is Bluetooth capable we could use a PC's GUI as host to control the system, the only disadvantage being that we lose remote portability. To overcome this we design an android application. Android is an Open Source Operating System and has a wide availability in the market. The android GUI would provide the same functionality as that of a PC.

1.3: Objectives

- The system should be scalable so that new devices can easily be integrated into it.
- It should provide a user- friendly interface on the host side, so that the devices can be easily setup, monitored and controlled.
- The system should be cost effective in order to justify its application in home automation
- The overall system should be fast enough to realize the true power of wireless technology
- To provide safer physical control to the user compared to the conventional high voltage switches
- To design a portable remote control method which is able to assist the disabled people who are bedridden.

1.4: Components Used

- **Hardware**
 - Arduino UNO – ATmega328 microcontroller

- Bluetooth Module HC-05
- 4-channel Relay Module
- LED
- Buzzer
- DC fan
- LM35 Temperature Sensor
- Breadboards
- Single Strand Wires
- 12V DC power module
- Android Smartphone
- **Software:**
 - Tera term Emulator
 - Android Studio
 - Arduino Integrated Development Environment
 - Eclipse IDE

1.5: Project Schedule:

Table 1.1: Project Schedule

<i>January 2016</i>	<ul style="list-style-type: none"> ✓ General outline of project ✓ Research from IEEE papers to understand past problems with Home automation systems
<i>February 2016</i>	<ul style="list-style-type: none"> ✓ Assembly of Hardware to given Specification ✓ Learning to work with Arduino Integrated Development Environment

	<ul style="list-style-type: none"> ✓ Programming Arduino to control system over Bluetooth
<i>March 2016</i>	<ul style="list-style-type: none"> ✓ Learning to work with Android Studio and Eclipse IDE ✓ Designing layout for the Android Application
<i>April 2016</i>	<ul style="list-style-type: none"> ✓ JAVA code for the android application ✓ Configuring android app to work with Arduino ✓ Testing of project
<i>May 2016</i>	<ul style="list-style-type: none"> ✓ Documentation ✓ Final Report

CHAPTER 2

BACKGROUND THEORY

2.1: System Architecture

The Home Automation System (HAS) was developed using Arduino IDE and Android Studio. A User Interfaced (UI) Android Application program was implemented on an Android based Bluetooth enabled mobile phone, and Arduino microcontroller based relay driver circuit with Serial Bluetooth Module, which is able to communicate with the Home Appliances over Bluetooth link. The system is based on serial data transmission using Bluetooth wireless communication in order to facilitate the appliance control in Home Automation Systems. The system ensures a secure exchange of data over wireless communication. It also supports conventional ON/OFF system of appliances. A user interface (UI) on the Android enabled mobile phone offers system connection and control utilities. A 4-Channel channel relay module and serial Bluetooth module HC-05 as well as Arduino IDE is used to compile the C program. Figure 2.1 and Figure 2.2 illustrate the basic system Architecture and circuit connections.

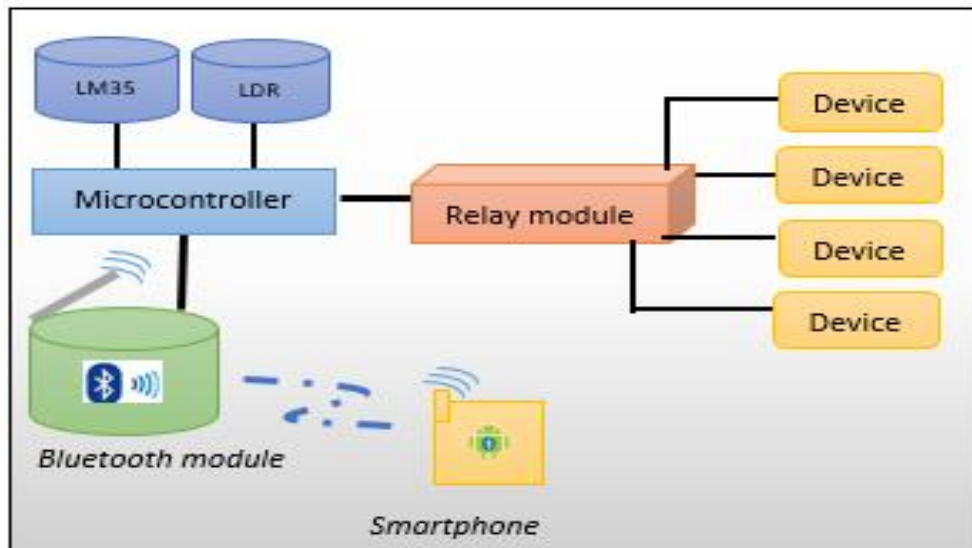


Figure 2. 1: Basic System Architecture

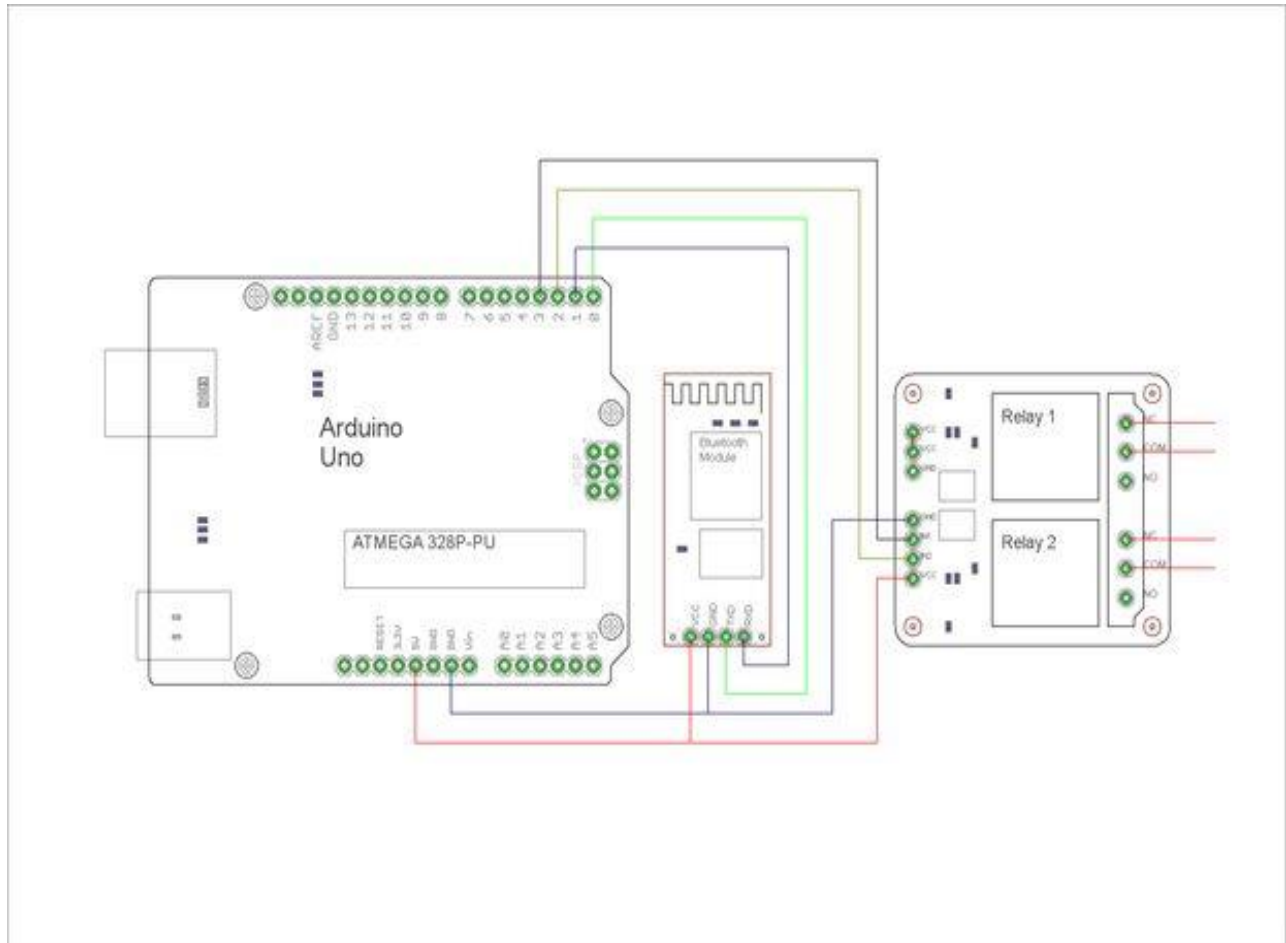


Figure 2.2: System Architecture circuit

2.2: Development Platform

This section describes the technologies used for developing the mobile phone application of the Home Automation System. The mobile phone application development presented in this paper is based on the following technologies: Android, Bluetooth. Android is a platform developing and deploying android based applications on mobile devices supporting it. Bluetooth has its own benchmark as a wireless communication technology for permanent and itinerant devices. Combining the power of Bluetooth, the best known and used wireless tech short range communication provide a facility to create Android based mobile applications using the Bluetooth Wireless Technology.

2.2.1: Android

Android is everywhere. Present days Phones, Tablets, TV's and set-top boxes powered by Google TV, Soon Android will be in cars, in in-flight entertainment systems on planes, and even in robots. Android was originally created by Andy Rubin as an operating system for mobile phones, around the dawn of this twenty-first century. In 2005, Google acquired Android Inc., and made Andy Rubin the Director of Mobile Platforms for Google. Many think the acquisition was largely in response to the emergence of the Apple iPhone around that time; however, there were enough other large players, such as RIM Blackberry, Nokia Symbian, and Microsoft Windows Mobile, that it seemed a salient business decision for Google to purchase the talent and intellectual property necessary to assert the company into this emerging space, which has become known as Internet 2.0.

Android's releases prior to 2.0 (1.0, 1.5, 1.6) were used exclusively on mobile phones. Most Android phones and some Android tablets now use a 2.x release and Android 3.0 was a tablet-oriented release but does not officially run on mobile phones. The current Android version is 6.0. Android's releases are nicknamed after sweets or dessert items like Cupcake (1.5), Frozen Yogurt ("Froyo") (2.2), Ginger Bread (2.3), Honeycomb (3.0), Ice Cream Sandwich (4.0), Jelly Bean (4.1) and Marshmallow (6.0) being the recent one. Android applications are written in the Java programming language. The Android SDK provides tools for code compilation and packaging data and resource files into an archive file with '.apk' extension called as an Android package. Android devices used the '.apk' file to install the application.

Android's application framework allows for the creation of extremely feature rich and novel applications by using a set of reusable components. The amalgamation of the Android development environment with the Bluetooth wireless technology is known by Android's support for the Bluetooth network stack, which permits a device to wirelessly exchange data with another Bluetooth device. The application framework enables access to the Bluetooth functionality using the Android Bluetooth APIs. These APIs allow wireless applications to connect to other Bluetooth devices for point-to-point and multipoint wireless features. Using the Bluetooth APIs, an Android application can carry out the following functions:

- Scrutinize for other Bluetooth devices
- Enquire about the local Bluetooth adapter for paired Bluetooth devices
- Establish the RFCOMM channels
- Connect to other devices through service discovery
- Exchange data to and from other devices
- Administer multiple connections

Android had unique support for Bluetooth in Android-powered devices including: Classic Bluetooth for more battery-intensive operations such as streaming and communicating and with low power requirements, Android 4.3 (API Level 18) introduces API support for Bluetooth Low Energy.

2.2.2: Bluetooth

Wireless networks for short range communications have a wide spread usage of Bluetooth radio transmissions between 2400–2480 MHz by Telecom vendor Ericsson since 1994. Bluetooth technology forms small ad hoc networks termed as Personal Area Networks (PANs) also provides a mechanism to emulate the RS-232 data cables, supervised by the Bluetooth Special Interest Group, since 1998. Modern mobile devices embed small, low-powered and cheap integrated chips functioning as short-range radio transceivers for Bluetooth radio communications. Device pairing, authentication, encryption and authorization techniques have given recognition to Bluetooth technology due to its vital security mechanisms. Different types of Bluetooth applications can be developed using Android platform architecture using the Bluetooth profiles. The device manufacturers provide the services using the support of these profiles in their devices to maintain compatibility for the Bluetooth technology. The Bluetooth profile used in Home Automation System (HAS) Android mobile phone application is the Bluetooth Serial Port Profile (btspp). RFCOMM is a connection-oriented protocol. It provides streaming communication between the devices. The btspp profile and RFCOMM protocol are used in the application to access the serial port and communicate using streaming data. All of the Bluetooth APIs is available in the `android.bluetooth` package.

2.3: Bluetooth Connectivity

Home Automation system application has the capability to exchange the ASCII data with HAS circuit through Bluetooth facility of mobile phone. The android Smartphone comprises of Bluetooth network stack .this allows a device to wirelessly exchange data with Bluetooth devices. This application framework then provides access to the Bluetooth functionality with the help of android Bluetooth API's. These API's make the application to connect wirelessly to other Bluetooth devices, for point-to-point and multipoint wireless features.

- A. **The Bluetooth APIs:** All of the Bluetooth APIs' are available in the Android Bluetooth package. The following is the overview of the classes needed during the application's development.
- **BluetoothAdapter:** Represents the local Bluetooth adapter (Bluetooth radio)
 - **BluetoothDevice:** Represents a remote Bluetooth device, to query information such as its name, address, class, and bonding state.
 - **BluetoothSocket:** Represents the interface for a Bluetooth socket (similar to a TCP Socket).
 - **BluetoothClass:** Describes the general characteristics and capabilities of a Bluetooth device.
- B. **Bluetooth Permissions:** In order to use Bluetooth features in an Android application, at least one of two Bluetooth permissions: `BLUETOOTH` and `BLUETOOTH_ADMIN` are needed to be declared.
- C. **Methods for Bluetooth connectivity:** Normally, before commencing communication, devices can use two methods for initiating communication with each other which can be done normally either by discovering other nearby devices to detect the address and services that are provided by other devices or by knowing the device address beforehand and directly using that address for further communication process.

1) **The Discovery method:** The devices participating in the communication process must be set to the discovery mode.

2) **The Known Address method:** The communication with a known remote device is helpful in faster communication as the discovery time is avoided. In this automation system, the appliances would be already known to the Bluetooth module as and when required. It is established in the following manner:

a) **SPP (Serial Port Profile)** in the Bluetooth profiles is implemented as the Bluetooth Serial Port Profile (btspp). Bluetooth profiles are the implementation of the Bluetooth protocols in full or partial manner as defined and adopted by the Bluetooth SIG. They reside over the Bluetooth protocol stack for their full or partial support. The implementation hence uses the support of Bluetooth Serial Port Profile (btspp) and RFCOMM protocol which is a connection-oriented protocol for Radio Frequency Communication, the replacement for the RS-232 cable to provide serial emulation.

b) **MAC Address** Bluetooth devices have a 12 hexadecimal digit MAC address which is to be known beforehand.

A complete specification for the connectivity in Home Appliance Control is done using the Known Method as follows. The entire setup described here includes the completion of these important steps using all classes and interfaces of the Android Bluetooth APIs available in the android.bluetooth package.

Step one: Bluetooth verification and enabling process

- Check for Bluetooth support.
- This can be accomplished by using the BluetoothAdapter in the application which serves as an entry point to all Bluetooth interactions. There is only one adapter for entire system and it represents the devices' Bluetooth radio (adapter). If it is null the device does not have Bluetooth support.
- Enable Bluetooth

- Check to make sure it is turned on in the application itself. Otherwise, request the user to turn on Bluetooth without leaving the application.

Step two: Set up a pointer to the remote node using its MAC address.

- The BluetoothAdapter from step one, is able to instantiate a BluetoothDevice using its pre-known MAC address. Two things are needed to make a connection:
 - **A MAC address.** We get it from the Bluetooth module's MAC address. For example, a 12 digit hexadecimal MAC address can be represented as 00:12:08:17:21:55.
 - **A Service ID or UUID.** In this case we are using the UUID for SPP. Services can be identified by a UUID. A Universally Unique Identifier (UUID) identifies each service and service attribute in Bluetooth uniquely. Each such identifier is guaranteed to be unique across all time and space. The UUID class in util package of java can be represented by short (16- or 32-bit) and long (128-bit) UUIDs. Constructors create a UUID from a String or from a 16- or 32-bit value, a method to compare two UUIDs (if both are 128-bit), and a method to convert a UUID into a String. The UUID instances are immutable, and only services identified by UUIDs are discoverable.

Step three: Establish the connection.

- After obtaining the BluetoothDevice object that represents the remote device, it is used to get the BluetoothSocket and initiate the connection by creating the insecure 'rfcomm socket' to service record by passing the SPP UUID to it that is hard coded before.

Step four: Create a data stream.

- The data stream helps to send message to the remote device, here, this helps to talk to the appliances finally in the Home Appliance Control application

CHAPTER 3

METHODOLOGY

3.1: Introduction

The methodology that is carried for this project is divided into 3 parts. These parts will be discussed in more detail in coming sections. The methodology begins with understanding research papers which are related with the project in question followed by analysis of case studies. After understanding the basic concepts, the next requirement was Hardware. The hardware is assembled according to the required specifications. Each and every component is tested. Lastly the software part of the project is undertaken which includes

- Programming the Arduino microcontroller using C/C++ on the Arduino IDE
- Development of android application with a basic GUI to control the system using Android Studio and Eclipse IDE

3.2: Review of Past Home Automation systems

In this section, an intense research was done on different types of Home Automation Systems. To fulfil our objective the best type of home automation is selected with minimum disadvantages. The selected systems is thoroughly studies and all problems associated with it are listed. Online material such as Videos, data sheets are refereed to know more about Home automation System and its working. Problems in the above mentioned system are addressed with practical solutions.

3.3: Hardware Design

After gaining the optimal knowledge about Home automation systems, an objective approach is taken to decide the hardware specifications of the project. The hardware used in this project includes:

- **Arduino UNO R3:** The Arduino UNO R3 is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs),

Technical specs	
Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Table 3.1: Technical Specifications

6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP Header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. The ATmega328P on the Arduino is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega328P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed. Technical Specifications are listed in the Table 3.1. Pin description for the UNO board are listed below, Figure 3.1 illustrates the PIN configuration of the microcontroller:

- **Vin.** The input voltage to the Uno board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.
- **IOREF.** This pin on the Uno board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.
- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial

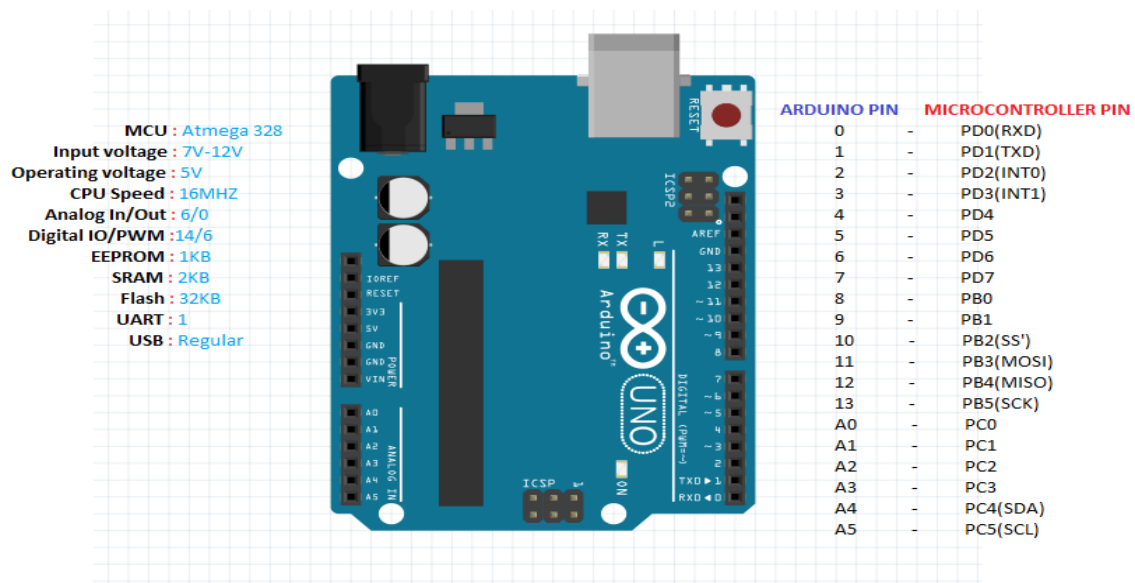


Figure 3.1: Pin Diagram

data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the `attachInterrupt()` function for details.
 - **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the `analogWrite()` function.
 - **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.
 - **LED: 13.** There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
 - **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the Wire library.
- **Bluetooth Module HC-05:** 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)

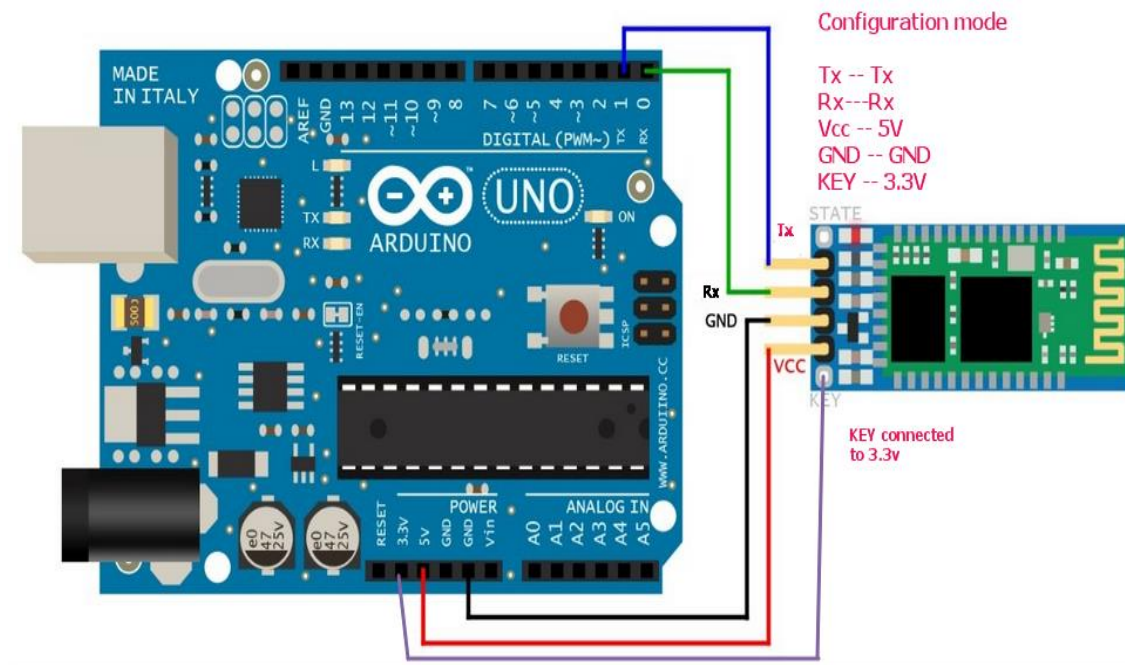


Figure 3.2: HC-05

3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR Bluecore 04-External single chip Bluetooth system with CMOS technology and with AFH(Adaptive Frequency Hopping Feature).The Figure 3.2 illustrates the connections to the Arduino Board.

○ **Specifications :**

- Typical -80dBm sensitivity
- Up to +4dBm RF transmit power
- Low Power 1.8V Operation ,1.8 to 3.6V I/O
- PIO control
- UART interface with programmable baud rate
- With integrated antenna
- With edge connector



Figure 3.3: HC-05 Module (Front)



Figure 3.4: HC-05 Module Pin Out(back)

- 4-Channel Relay Module:** The **relay module** is a separate hardware device used for remote device switching. With it you can remotely control devices over a network or the Internet. Devices can be remotely powered on or off with commands coming from a microcontroller. We are using a relay module which has 4 relays with rating of 10A @ 250 and 125 V AC and 10A @ 30 and 28 V DC. The high voltage output connector has 3 pins, the middle one is the common pin and as we can see from the markings one of the two other pins is for normally open connection and the other one for normally closed connection. On the other side of the module we have these 2 sets of pins. The first one has 6 pins, a Ground and a VCC pin for powering the module and 4 input pins In1 to In4. The second set of pins has 3 pins with a jumper between the JDVcc and the Vcc pin. With a configuration like this the electromagnet of the relay is directly powered from the Arduino Board and if something goes wrong with the relay the microcontroller could get damaged. In Figure 3.5 connections to a relay module are shown.

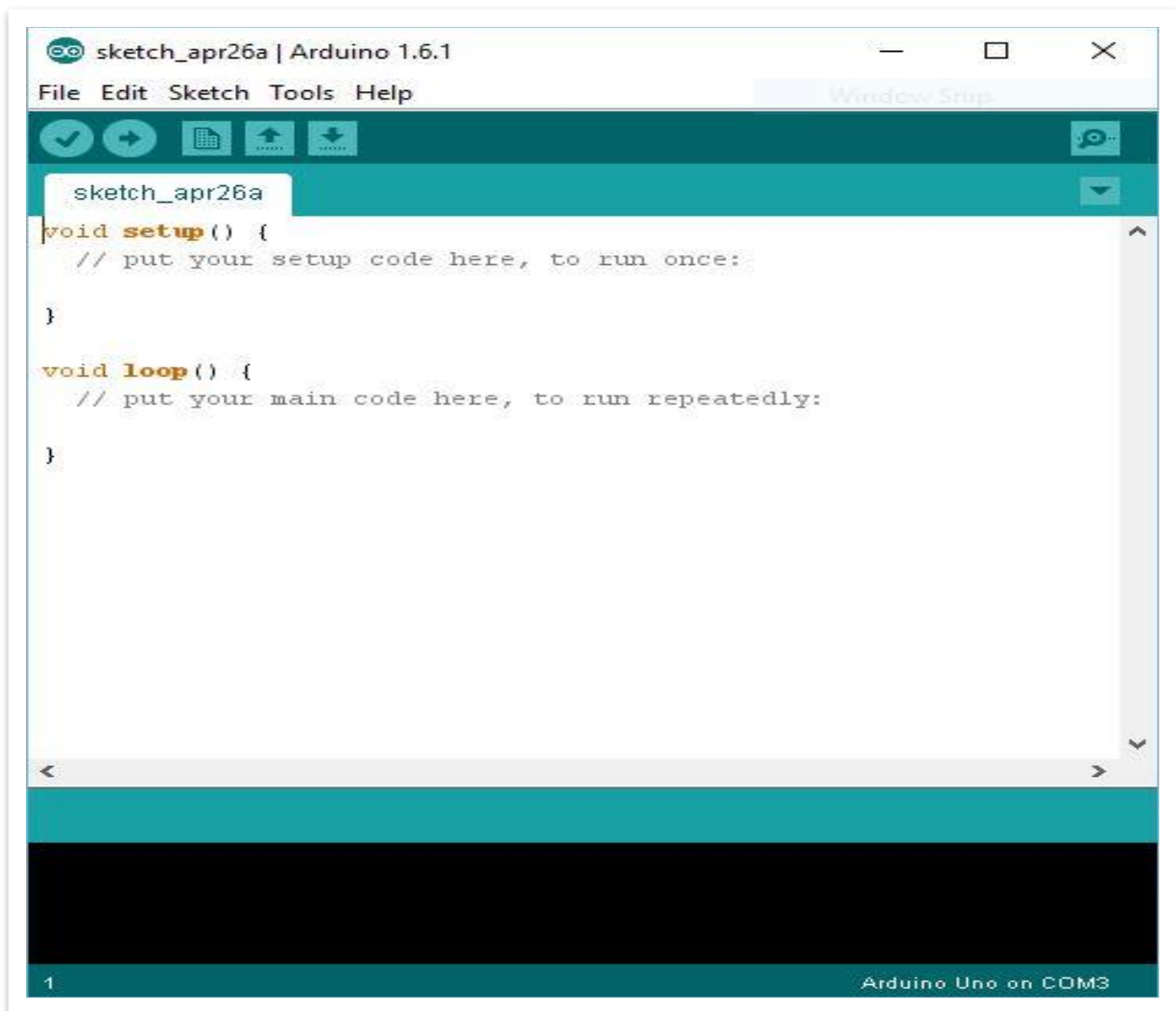


Figure 3.6: Arduino IDE

3.4: Software Design

3.4.1: Software Tools Used

- **Arduino IDE:** The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them. Programs written using Arduino Software (IDE) are called sketches. These sketches are written in the text editor and are saved with the file extension .ino. The editor has features for cutting/pasting and for searching/replacing text. The Editor window is shown in Figure 3.6 .The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom right-hand corner of the window displays the configured board and serial port.
- **Android Studio:** Android Studio is the official IDE for Android application development, based on IntelliJ IDEA. On top of the capabilities you expect from IntelliJ, Android Studio
 - Flexible Gradle-based build system
 - Build variants and multiple apk file generation
 - Code templates to help you build common app features
 - Rich layout editor with support for drag and drop theme editing lint tools to catch performance, usability, version compatibility, and other problems ProGuard and app-signing capabilities Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

Figure 3.7 shows the MainActivity.java window in the Android Studio GUI .The main code for the android app is written in this window. The programming Language used is JAVA. The layout of the application is coded in HTML. We may use the Graphic Editor (Figure 3.8) to edit the layout.xml file.



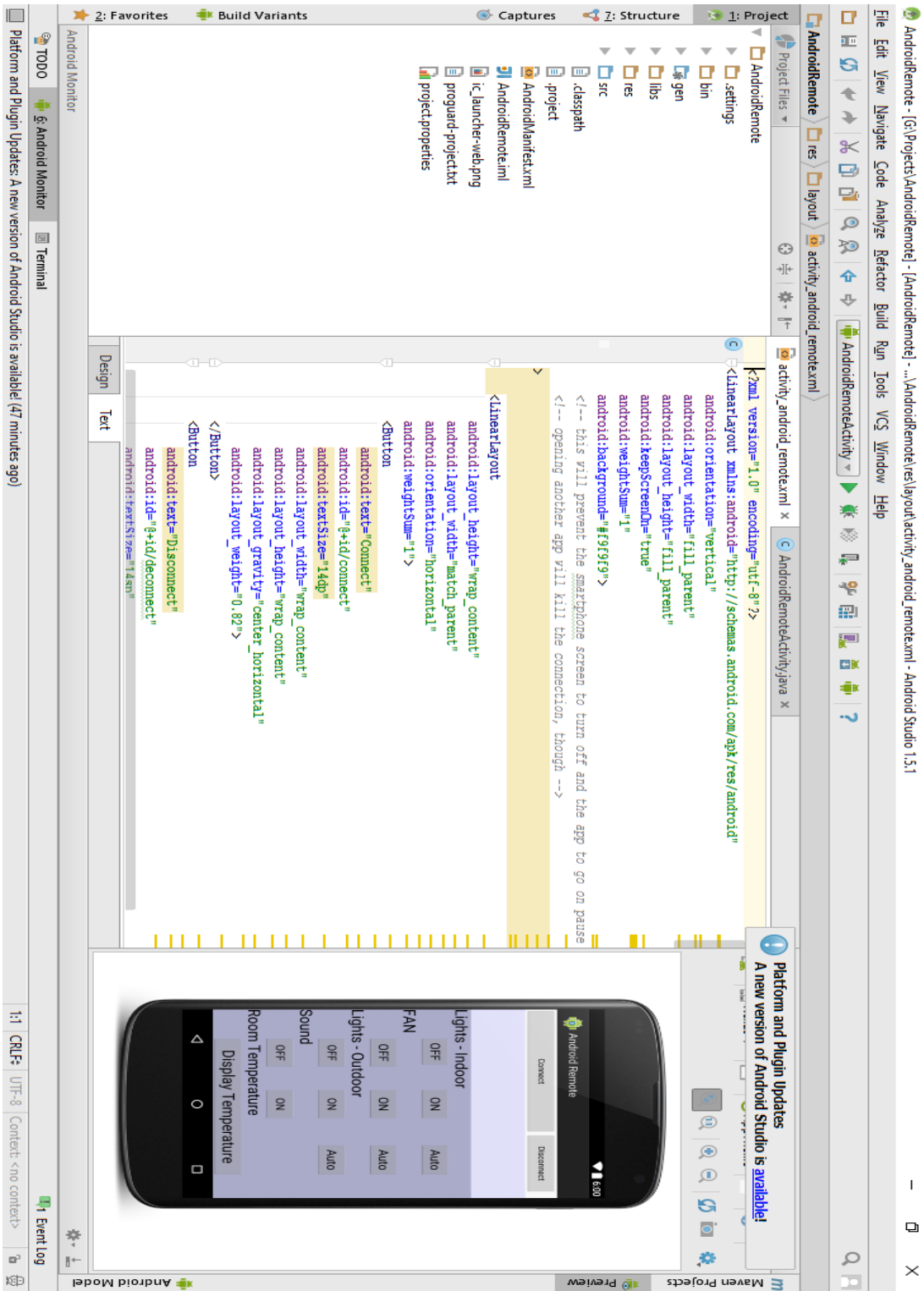


Figure 3.9: Android Studio: Layout HTML Code

- **Eclipse IDE:** Eclipse is a Java-based open source platform that allows a software developer to create a customized development environment (IDE) from plug-in components built by Eclipse members. Eclipse is managed and directed by the Eclipse.org Consortium. In the enterprise, a major advantage to an open source development platform is that it allows an IT department to mix and match development tools rather than being committed to a single vendor's suite of development products. Although the Eclipse Platform is written in Java, it supports plug-ins that allow developers to develop and test code written in other languages.
- **TeraTerm:** Tera Term is an open-source, free, software implemented, and terminal emulator (communications) program. It emulates different types of computer terminals, from DEC VT100 to DEC VT382. It supports telnet, SSH 1 & 2 and serial port connections.

3.4.2 Microcontroller Programming

The microcontroller receives commands from the user over Bluetooth from the Android GUI. The Microcontroller uses serial communication. The user has the option to manually control the home appliances or set it to automatic. When set to automatic the Arduino uses sensor data to smartly turn On/Off home appliances. The change in status of the home appliances is displayed on the Android GUI. Arduino changes the voltage of the relay switch to change its status. The Flowchart for the same program is illustrated in the Figure 3.10. The Programming language used for coding the microcontroller is C/C++. A delay on 200ms is also added to ensure smooth functioning of the microcontroller.

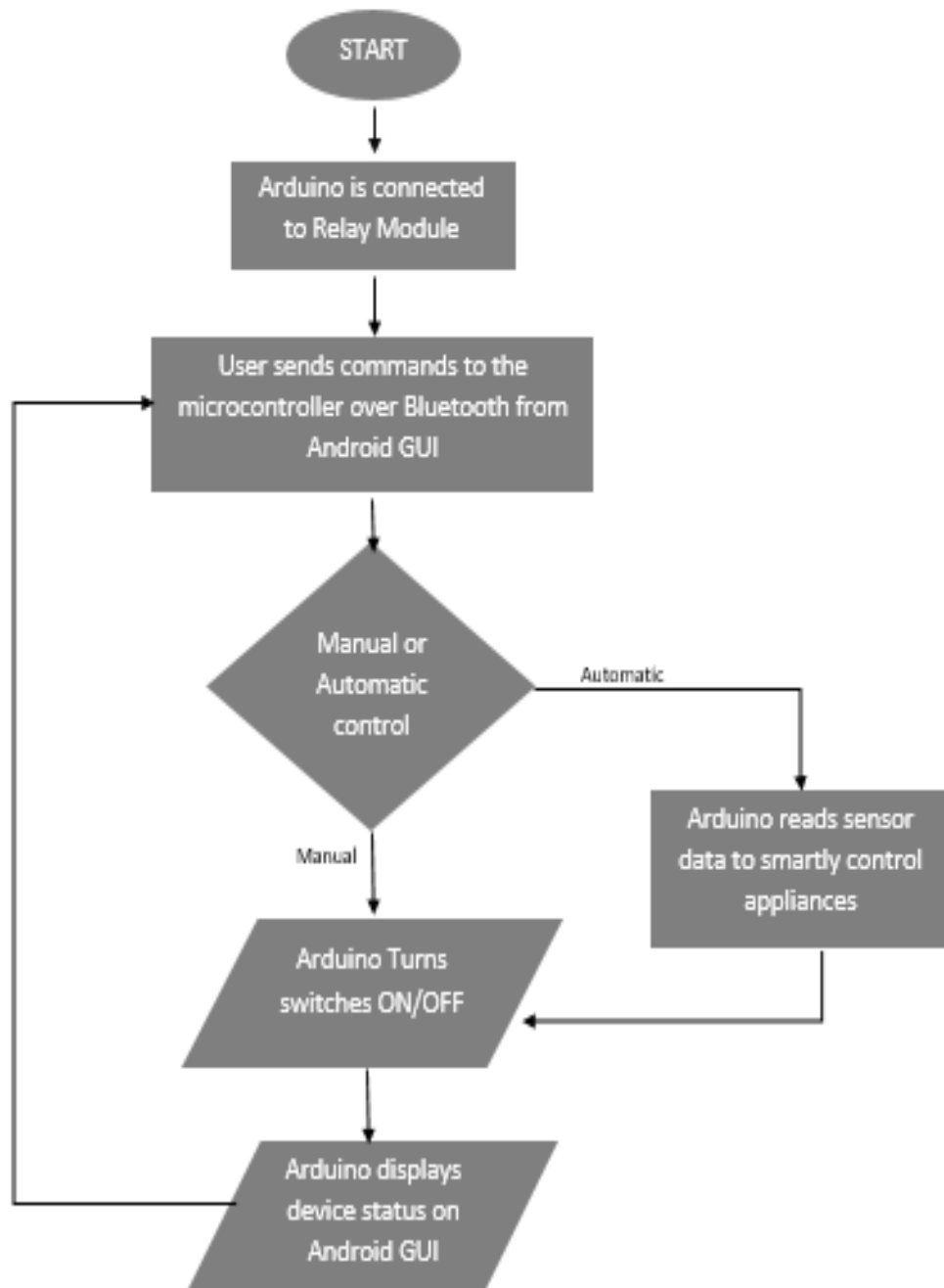


Figure 3.10: Flowchart for Microcontroller

3.4.3: Android GUI

The application is designed for android version 4.0 (Ice cream sandwich) with API level 16. This API level supports 99% of available Android devices. The Android interface is very simple. It has two buttons on the top to connect/disconnect to the system, a logview which displays the status of various home devices. The user can simply touch a button on the GUI to control the home appliances. Figure 3.11 illustrates the design of the Android GUI.

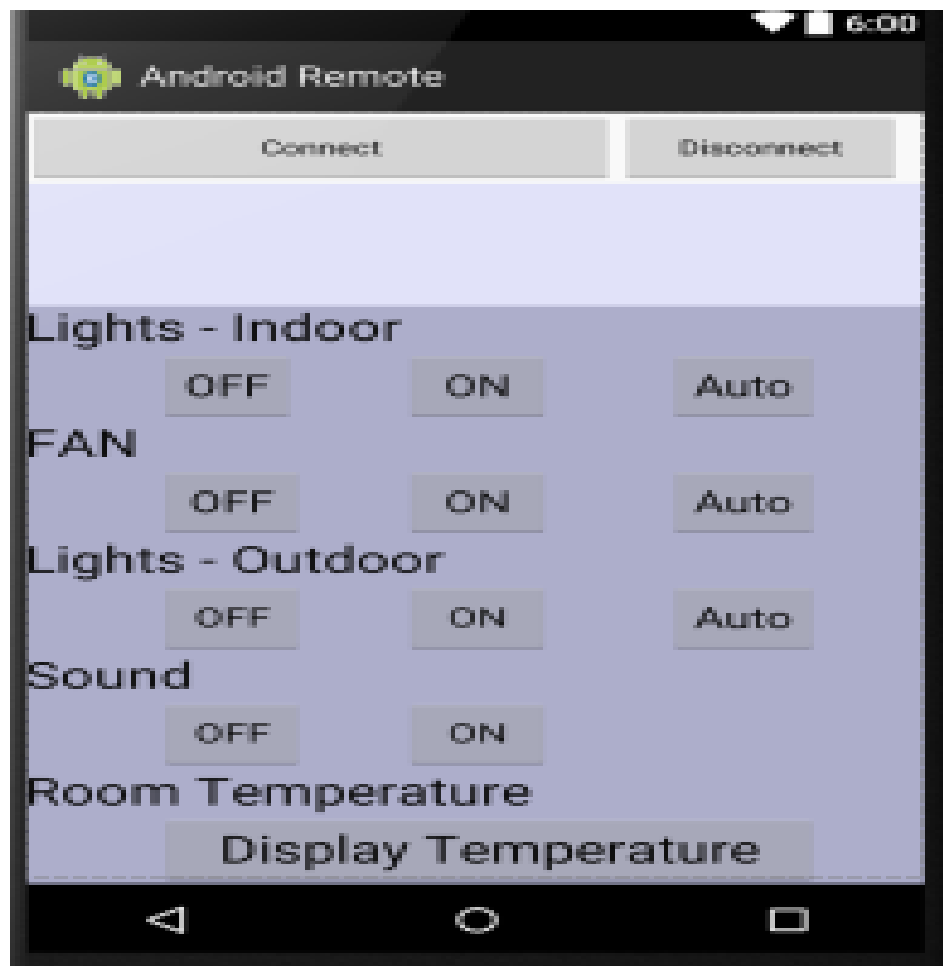


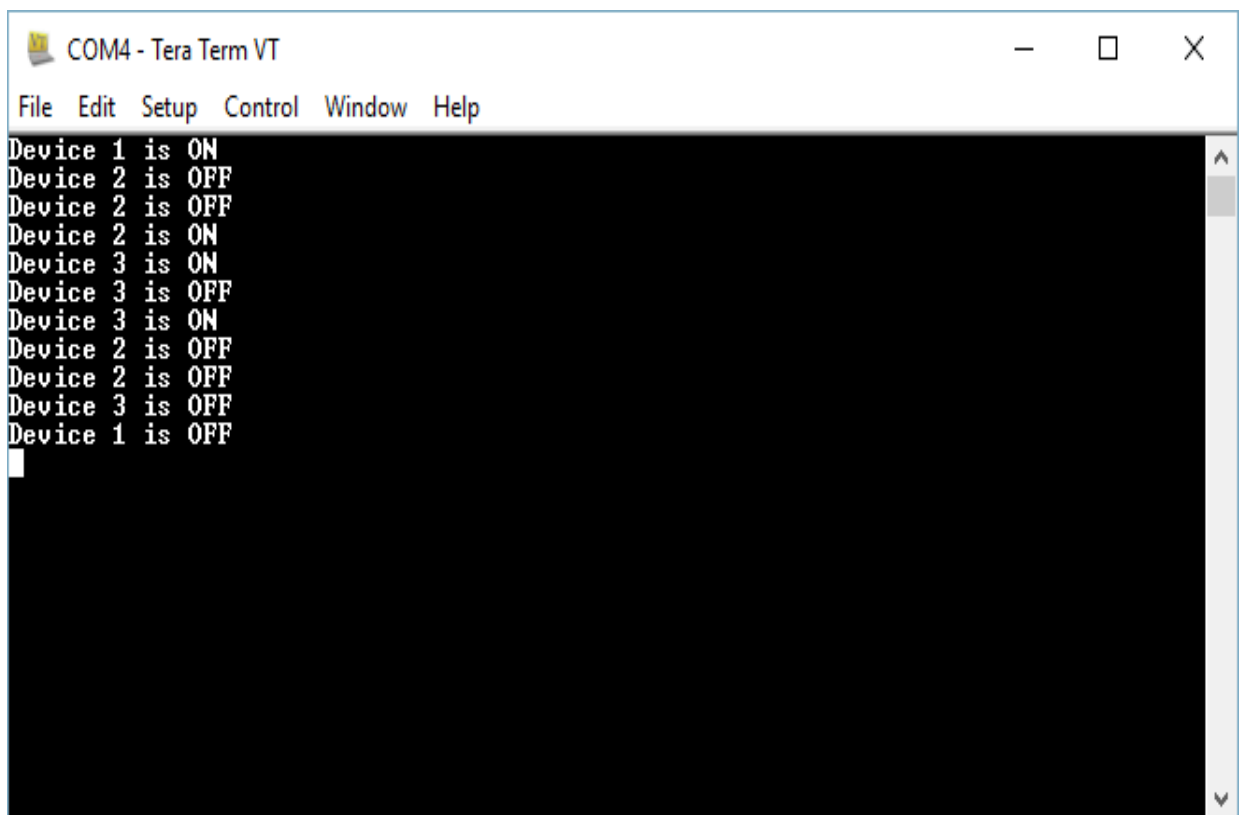
Figure 3.11: Android GUI

CHAPTER 4

RESULT ANALYSIS

4.1: Introduction

This section discusses about the outputs that were obtained after connecting the given hardware. We were successfully able to control the relay switches using the Terminal emulator (TeraTerm- Figure 4.1) and turn ON/OFF appliances connected to the system. This demonstration was carried out on a PC. The procedure is then demonstrated on the Android GUI.



The image shows a screenshot of a TeraTerm VT terminal window. The window title is "COM4 - Tera Term VT". The menu bar includes "File", "Edit", "Setup", "Control", "Window", and "Help". The terminal output displays the status of three devices, with each status being reported twice. The sequence of messages is: "Device 1 is ON", "Device 2 is OFF", "Device 2 is OFF", "Device 2 is ON", "Device 3 is ON", "Device 3 is OFF", "Device 3 is ON", "Device 2 is OFF", "Device 2 is OFF", "Device 3 is OFF", and "Device 1 is OFF". A white cursor is visible on the line following the last message.

```
COM4 - Tera Term VT
File Edit Setup Control Window Help
Device 1 is ON
Device 2 is OFF
Device 2 is OFF
Device 2 is ON
Device 3 is ON
Device 3 is OFF
Device 3 is ON
Device 2 is OFF
Device 2 is OFF
Device 3 is OFF
Device 1 is OFF

```

Figure 4.1: TeraTerm

4.2: Results

The results obtained show that we were successfully able to complete the project and meet all our objectives. The project snapshot is shown in Figure 4.3.

The relay module is successfully installed. It is an active low device. We see that when the microcontroller gives a 0V signal to the relay the appliance is turned ON. This shows the relay module is running properly. The Bluetooth Module HC-05 is functioning properly and is able to receive commands from the smartphone from a considerable distance. The Figure 4.3 shows the output GUI and successful connection to the microcontroller. On pressing the buttons on the GUI the smartphone sends a unique ASCII code to the microcontroller. The Microcontroller then reads the ASCII value and executes the corresponding part of code. When the ON/OFF button is pressed on the GUI for the particular device, it turns ON/OFF. We can now control any device which is hooked to the Automation system.

The sensors are also successfully installed. When the user presses the AUTO button on the android GUI for the lights the LDR sensor detects the ambient light and automatically control the home lights, similarly the Temperature sensor LM35 detects the ambient temperature to turn ON/OFF the air conditioning. This results in a lot of energy conservation since the system moderates the use of appliances.

The Android application is also working flawlessly. It takes a very reasonable time to connect to the system. The reaction time to turn ON/OFF appliance from the android application is less than 200ms. We are also able to see the status of the different devices connected to the system on the android application in the logview window. The working application is shown in figure 4.2. It is running on Android 4.3.

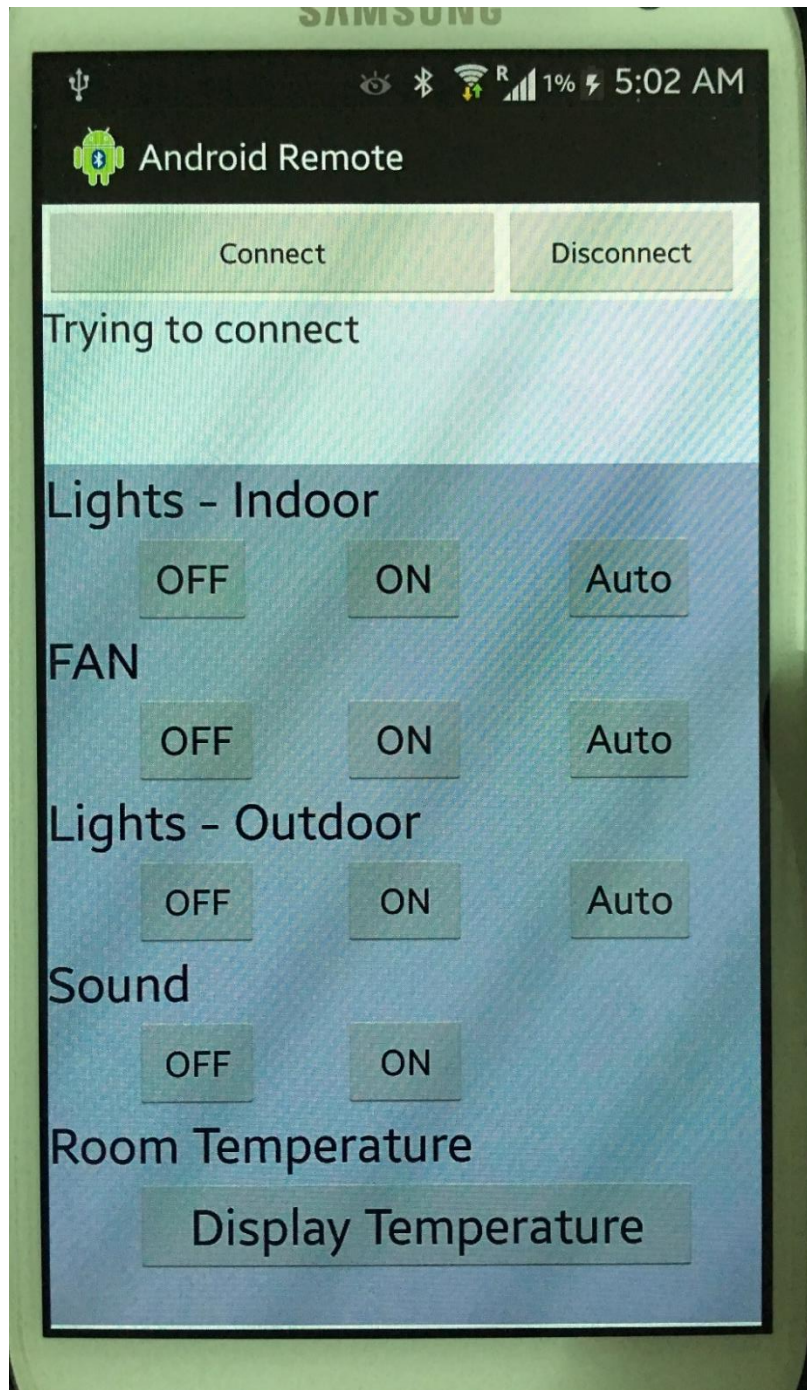


Figure 4.2: Android Application on Smartphone

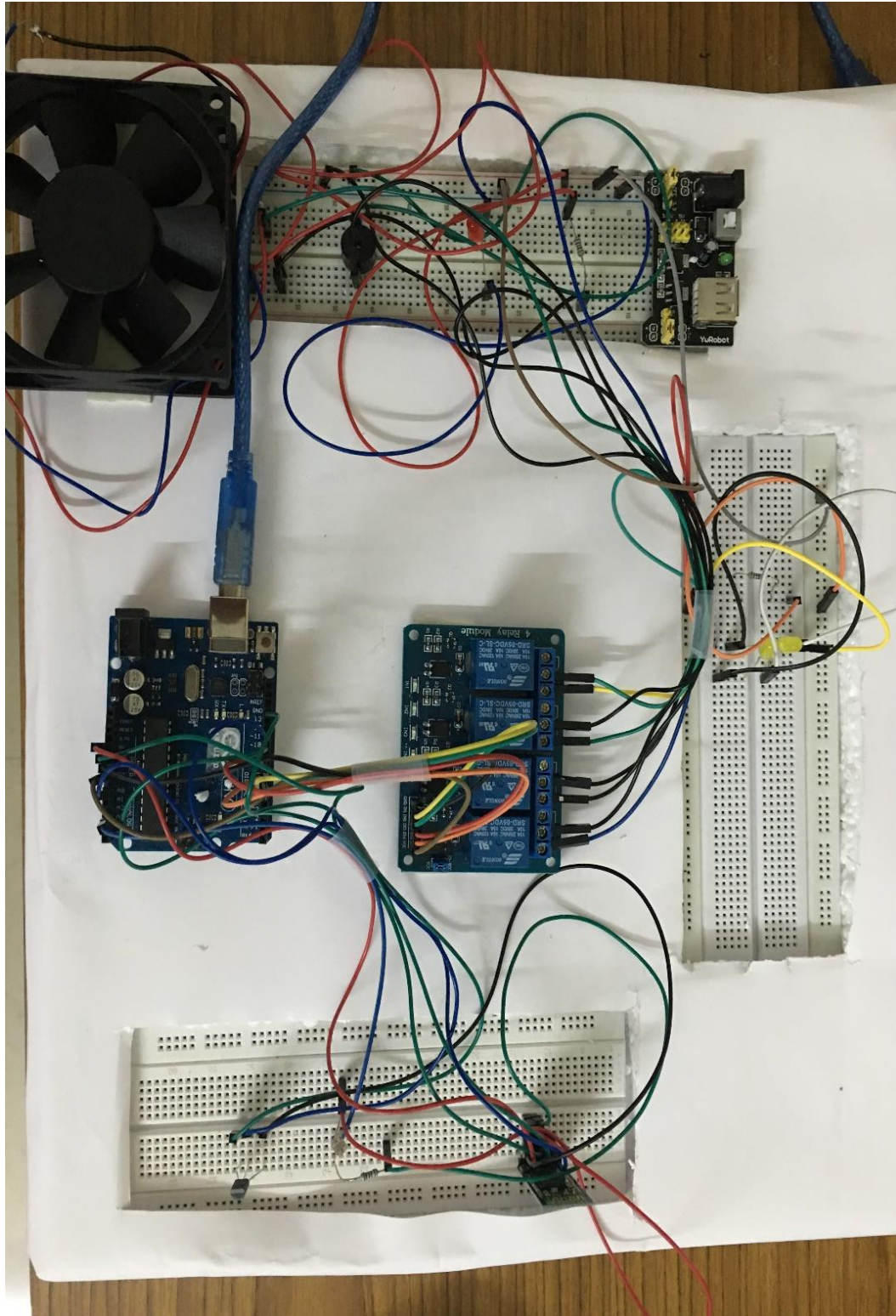


Figure 4.3: Project Snapshot

CHAPTER 5

CONCLUSION & FUTURE WORK

5.1: Conclusion

Home Automation is undeniably a resource which can make a home environment automated. People can control their electrical devices via these Home Automation Systems and set up the controlling actions using the mobile GUI. I think this product has high potential for marketing in today's scenario.

In conclusion, this low cost system is designed to improve the standard living in home. The remote control function by smart phone provides help and assistance especially to disabled and elderly. In order to provide safety protection to the user, a low voltage activating switches is replaced current electrical switches. Moreover, implementation of wireless Bluetooth connection ensures that the system is connected in a simple way. The control board is directly installed beside the electrical switches whereby the switching connection is controlled by relay. Flexible types of connections are designed as backup connections to the system. The GUI's on the mobile are designed in such a way that they indicate the real-time switches status. The system is designed in user-friendly interface. The easy to use interface on Android GUI provides simple control by the elderly and disabled people. This low cost system is bought under \$20, considering the monthly savings of \$2/month on electricity, the cost of the system can be recovered over a period of 10 months, which makes it appealing to the consumer.

Furthermore, energy conservation is the need of the hour, with the growing energy demand and limited resources, this system can help reduce energy consumption to a great extent. It is also important to note that it would also help reduce the greenhouse footprint by regulating the use of home appliances

5.2: Future Scope of Work

For future work, we could develop a Desktop Application for Windows which would have a similar interface as that of the Android app .Furthermore, we could implement Speech recognition to both Windows GUI and Android App. The mobile app could be used as a mobile microphone which would be transmitted over Bluetooth. All the voice signal inputs to the smart phone will be transmitted to the Window GUI for signal processing. Also, the push buttons implemented in low voltage activating switches will be replaced by capacitive sensing switches. We can also implement the same system over Wi-Fi thereby increasing the range. The system could also be connected to the internet to enable Remote Control from a different location. The scope of research and development for such systems is endless.

REFERENCES

Journal / Conference Papers

- [1] The International Journal of Engineering and Science (IJES) ||Volume|| 2 ||Issue|| 01 ||2013||ISSN: 2319 – 1813 ISBN: 2319 – Bluetooth Remote Home Automation System Using Android Application 1R.A.Ramlee, 2M.H.Leong, 3R.S.S.Singh, 4M.M.Ismail, 5M.A.Othman, 6H.A.Sulaiman, 7M.H.Misran, 8M.A.Meor Said
- [2] R. Piyare and M. Tazil, “Bluetooth Based Home Automation System using Cell Phone,” in Consumer Electronics, 2011, pp.192-195
- [3] M. Chan, D. Esteve, C. Escriba, E. Campo, “A review of smart homes—Present state and future challenges”, Computer methods and programs in biomedicine, Elsevier, 91, pp. 55-81, 2008.
- [4] S. Panth, M. Jivani, “Designing Home Automation system (HAS) using Java ME for Mobile Phone”, International Journal of Electronics and Computer Science Engineering, Vol. 2 No. 02, pp. 798-807, July 2013

Web

- [1] The official Bluetooth website from Bluetooth SIG:<http://www.bluetooth.com>
- [2] Official Arduino BT website: <http://www.arduino.cc/en/Guide/ArduinoBT>
- [3] BluetoothAPI Guides, <https://developer.android.com/guide/topics/connectivity/bluetooth.html>

ANNEXURES

Arduino Code

```
#include <SoftwareSerial.h>
int Dev1 = 7;
int Dev2 = 8;
int Dev3 = 6;
int Dev4 = 5;
int DevAutoFan ;
int DevAutoLightIn ;
int DevAutoLightOut ;
float TempC;
int tempin = 0;
SoftwareSerial BT(10, 11);
int val;
float TempMin = 21.0 ;
float TempMax = 28.0 ;
int Auto;
int AutoL;
int AutoLIn;
int LDR = 3 ;

void setup() {
  BT.begin(9600);
  BT.println("Bluetooth On ");
  digitalWrite(Dev1,HIGH);
  digitalWrite(Dev2,HIGH);
  digitalWrite(Dev3,HIGH);
  digitalWrite(Dev4,HIGH);
  pinMode(Dev1, OUTPUT);
  pinMode(Dev2, OUTPUT);
  pinMode(Dev3, OUTPUT);
  pinMode(Dev4,OUTPUT);

}
```



```

void loop() {

    delay(1000);

    float TempC;
    TempC = analogRead(tempin);
    TempC = ((TempC / 1024.0)) * 500 ;

    if (TempC > 26.0 && DevAutoFan != LOW)
    {
        constrain (TempC , 21.0 , 28.0);
        Auto = map (TempC , TempMin , TempMax , 1 , 0 );
        digitalWrite(Dev3 , Auto);
    }
    int LDRdata ;
    LDRdata = analogRead(LDR);
    LDRdata = constrain (LDRdata , 200 , 400);

    if ( DevAutoLightOut == HIGH )
    {
        AutoL = map (LDRdata , 200 , 400 , LOW , HIGH);
        digitalWrite(Dev4 , AutoL);
    }
    if ( DevAutoLightIn == HIGH )
    {
        AutoLin = map (LDRdata , 200 , 400 , LOW , HIGH);
        digitalWrite(Dev1 , AutoLin);
    }

    int val = BT.read()- '0';

    if (val == 1)
    {BT.println("Indoor Lights ON ");
    digitalWrite(Dev1 , LOW);
    DevAutoLightIn = LOW;

```

```

}

else if (val == 2)
{BT.println("Indoor Lights OFF ");
digitalWrite(Dev1 , HIGH);
DevAutoLightIn = LOW;
}
    else if (val == 3)
{BT.println("Sound ON ");
digitalWrite(Dev2 , LOW);
}
else if (val == 4)
{BT.println("Sound OFF ");
digitalWrite(Dev2 , HIGH);
}
else if (val == 5)
{BT.println("Fan ON ");
DevAutoFan = LOW;
digitalWrite(Dev3 , LOW);
}
    else if (val == 6)
{BT.println("Fan OFF ");
DevAutoFan = LOW;
digitalWrite(Dev3 , HIGH);
}
    else if ( val == 7 )
{
DevAutoFan = HIGH ;
BT.println("Enable AUTO Fan");
}
    else if ( val == 8 )
{BT.println("Outdoor Lights ON ");
digitalWrite(Dev4 , LOW);
DevAutoLightOut = LOW;
}
else if (val == 9)

```

```

{
BT.println("Outdoor Lights OFF");
digitalWrite(Dev4 , HIGH);
DevAutoLightOut = LOW;
}
else if (val == 0)
{
BT.println("Enable AUTO Indoor Lights");
DevAutoLightIn = HIGH ;
}
else if (val == 49)
{
BT.println("Enable AUTO Outdoor Lights");
DevAutoLightOut = HIGH ;
}
else if (val == 51)
{
BT.println("Room Temp Is ");
BT.println(TempC ) ;
delay(200);
}
}

```

Android code

```

PACKAGE COM.EXAMPLE.ANDROIDREMOTE;
IMPORT ANDROID.APP.ACTIVITY;
IMPORT ANDROID.BLUETOOTH.BLUETOOTHADAPTER;
IMPORT ANDROID.CONTENT.INTENT;
IMPORT ANDROID.OS.BUNDLE;
IMPORT ANDROID.OS.HANDLER;
IMPORT ANDROID.OS.MESSAGE;
IMPORT ANDROID.UTIL.LOG;
IMPORT ANDROID.VIEW.VIEW;

```

```

IMPORT ANDROID.VIEW.VIEW.ONCLICKLISTENER;
IMPORT ANDROID.WIDGET.BUTTON;
IMPORT ANDROID.WIDGET.IMAGEBUTTON;
IMPORT ANDROID.WIDGET.IMAGEVIEW;
IMPORT ANDROID.WIDGET.TEXTVIEW;
IMPORT ANDROID.WIDGET.TOAST;
IMPORT ANDROID.WIDGET.TOGGLEBUTTON;

PUBLIC CLASS ANDROIDREMOTEACTIVITY EXTENDS ACTIVITY IMPLEMENTS
ONCLICKLISTENER {
    PRIVATE TEXTVIEW LOGVIEW;
    PRIVATE BUTTON CONNECT;
    PRIVATE BUTTON DECONNECT;
    PRIVATE    BUTTON LIGHTINON , LIGHTINOFF , LIGHTINAUTO ;
    PRIVATE BUTTON FANON , FANOFF , FANAUTO ;
    PRIVATE BUTTON LIGHTOUTON ,LIGHTOUTOFF , LIGHTOUTAUTO ;
    PRIVATE BUTTON SOUNDON , SOUNDOFF ;
    PRIVATE BUTTON TEMP ;
    PRIVATE BLUETOOTHADAPTER MBLUETOOTHADAPTER = NULL;
    PRIVATE STRING[] LOGARRAY = NULL;
    PRIVATE BTINTERFACE BT = NULL;
    STATIC FINAL STRING TAG = "BTMODULE";
    STATIC FINAL INT REQUEST_ENABLE_BT = 3;
        //THIS HANDLER LISTENS TO MESSAGES FROM THE BLUETOOTH
INTERFACE AND ADDS THEM TO THE LOG
    FINAL HANDLER HANDLER = NEW HANDLER() {
    PUBLIC VOID HANDLEMESSAGE(MESSAGE MSG) {
        STRING DATA = MSG.GETDATA().GETSTRING("RECEIVEDDATA");
        ADDTOLOG(DATA);
    }
};
        //THIS HANDLER IS DEDICATED TO THE STATUS OF THE BLUETOOTH
CONNECTION
    FINAL HANDLER HANDLERSTATUS = NEW HANDLER() {
    PUBLIC VOID HANDLEMESSAGE(MESSAGE MSG) {
        INT STATUS = MSG.ARG1;

```

```

        IF (STATUS == BTINTERFACE.CONNECTED) {
            ADDTOLOG("CONNECTED");
        } ELSE IF (STATUS == BTINTERFACE.DISCONNECTED) {
            ADDTOLOG("DISCONNECTED");
        }
    }
};

//HANDLES THE LOG VIEW MODIFICATION
//ONLY THE MOST RECENT MESSAGES ARE SHOWN
PRIVATE VOID ADDTOLOG (STRING MESSAGE){
    FOR (INT I = 1; I < LOGARRAY.LENGTH; I++){
        LOGARRAY[I-1] = LOGARRAY[I];
    }
    LOGARRAY[LOGARRAY.LENGTH - 1] = MESSAGE;

    LOGVIEW.SETTEXT("");
    FOR (INT I = 0; I < LOGARRAY.LENGTH; I++){
        IF (LOGARRAY[I] != NULL){
            LOGVIEW.APPEND(LOGARRAY[I] + "\n");
        }
    }
}

/** CALLED WHEN THE ACTIVITY IS FIRST CREATED. */
@OVERRIDE
PUBLIC VOID ONCREATE (BUNDLE SAVEDINSTANCESTATE) {
    SUPER.ONCREATE (SAVEDINSTANCESTATE);
    SETCONTENTVIEW (R.LAYOUT.ACTIVITY_ANDROID_REMOTE);
    TOAST.MAKETEXT (ANDROIDREMOTEACTIVITY.THIS, "WELCOME TO
    ANDROID REMOTE 1.0", TOAST.LENGTH_SHORT).SHOW();
    TOAST.MAKETEXT (ANDROIDREMOTEACTIVITY.THIS, "APP
    DESIGNED BY JAYESH MEHTA", TOAST.LENGTH_LONG).SHOW();

    //FIRST, INFLATE ALL LAYOUT OBJECTS, AND SET CLICK LISTENERS
    LOGVIEW = (TEXTVIEW)FINDVIEWBYID (R.ID.LOGVIEW);
    //I CHOSE TO DISPLAY ONLY THE LAST 5 MESSAGES

```

```

LOGARRAY = NEW STRING[2];
    CONNECT = (BUTTON)FINDVIEWBYID(R.ID.CONNECT);
    CONNECT.SETONCLICKLISTENER(THIS);
        DECONNECT = (BUTTON)FINDVIEWBYID(R.ID.DECONNECT);
        DECONNECT.SETONCLICKLISTENER(THIS);

        LIGHTINOFF =(BUTTON)FINDVIEWBYID(R.ID.LIGHTINOFF);
        LIGHTINOFF.SETONCLICKLISTENER(THIS);
        LIGHTINON = (BUTTON)FINDVIEWBYID(R.ID.LIGHTINON);
        LIGHTINON.SETONCLICKLISTENER(THIS);
        LIGHTINAUTO = (BUTTON)FINDVIEWBYID(R.ID.LIGHTINAUTO);
        LIGHTINAUTO.SETONCLICKLISTENER(THIS);
        FANOFF = (BUTTON)FINDVIEWBYID(R.ID.FANOFF);
        FANOFF.SETONCLICKLISTENER(THIS);
        FANON = (BUTTON)FINDVIEWBYID(R.ID.FANON);
        FANON.SETONCLICKLISTENER(THIS);
        FANAUTO = (BUTTON)FINDVIEWBYID(R.ID.FANAUTO);
        FANAUTO.SETONCLICKLISTENER(THIS);
        LIGHTOUTOFF =(BUTTON)FINDVIEWBYID(R.ID.LIGHTOUTOFF);
        LIGHTOUTOFF.SETONCLICKLISTENER(THIS);
        LIGHTOUTON = (BUTTON)FINDVIEWBYID(R.ID.LIGHTOUTON);
        LIGHTOUTON.SETONCLICKLISTENER(THIS);
        LIGHTOUTAUTO = (BUTTON)FINDVIEWBYID(R.ID.LIGHTOUTAUTO);
        LIGHTOUTAUTO.SETONCLICKLISTENER(THIS);
        SOUNDOFF = (BUTTON)FINDVIEWBYID(R.ID.SOUNDOFF);
        SOUNDOFF.SETONCLICKLISTENER(THIS);
        TEMP = (BUTTON)FINDVIEWBYID(R.ID.TEMP);
        TEMP.SETONCLICKLISTENER(THIS);
        SOUNDON = (BUTTON )  FINDVIEWBYID(R.ID.SOUNFON);
        SOUNDON.SETONCLICKLISTENER(THIS);
    }

```

```

//IT IS BETTER TO HANDLE BLUETOOTH CONNECTION IN ONRESUME (IE ABLE
TO RESET WHEN CHANGING SCREENS)

```

```

@Override

```

```

PUBLIC VOID ONRESUME() {

```

```

SUPER.ONRESUME();
//FIRST OF ALL, WE CHECK IF THERE IS BLUETOOTH ON THE PHONE
MBLUETOOTHADAPTER = BLUETOOTHADAPTER.GETDEFAULTADAPTER();
IF (MBLUETOOTHADAPTER == NULL) {
    // DEVICE DOES NOT SUPPORT BLUETOOTH
    LOG.V(TAG, "DEVICE DOES NOT SUPPORT BLUETOOTH");
}
    ELSE{
//DEVICE SUPPORTS BT
IF (!MBLUETOOTHADAPTER.ISENABLED()){
    //IF BLUETOOTH NOT ACTIVATED, THEN REQUEST IT
    INTENT          ENABLEBTINTENT          =          NEW
INTENT(BLUETOOTHADAPTER.ACTION_REQUEST_ENABLE);
    STARTACTIVITYFORRESULT(ENABLEBTINTENT, REQUEST_ENABLE_BT);
}
    ELSE{
        //BT ACTIVATED, THEN INITIATE THE BTINTERFACE OBJECT TO
HANDLE ALL BT COMMUNICATION
        BT = NEW BTINTERFACE(HANDLERSTATUS, HANDLER);
    }
}
}

//CALLED ONLY IF THE BT IS NOT ALREADY ACTIVATED, IN ORDER TO
ACTIVATE IT
PROTECTED VOID ONACTIVITYRESULT(INT REQUESTCODE, INT
RESULTCODE, INTENT MOREDATA){
    IF (REQUESTCODE == REQUEST_ENABLE_BT){
        IF (RESULTCODE == ACTIVITY.RESULT_OK){
            //BT ACTIVATED, THEN INITIATE THE BTINTERFACE OBJECT TO
HANDLE ALL BT COMMUNICATION
            BT = NEW BTINTERFACE(HANDLERSTATUS, HANDLER);
        }
        ELSE IF (RESULTCODE == ACTIVITY.RESULT_CANCELED)
            LOG.V(TAG, "BT NOT ACTIVATED");
        ELSE

```

```

        LOG.V(TAG, "RESULT CODE NOT KNOWN");
    }
    ELSE{
        LOG.V(TAG, "REQUEST CODE NOT KNOWN");
    }
}

//HANDLES THE CLICKS ON VARIOUS PARTS OF THE SCREEN
//ALL BUTTONS LAUNCH A FUNCTION FROM THE BTINTERFACE OBJECT
@OVERRIDE
PUBLIC VOID ONCLICK(VIEW V) {
    IF(V == CONNECT) {
        ADDTOLOG("TRYING TO CONNECT");
        BT.CONNECT();
    }
    ELSE IF(V == DECONNECT) {
        ADDTOLOG("CLOSING CONNECTION");
        BT.CLOSE();
    }
    ELSE IF(V == LIGHTINON) {

        BT.SENDDATA("1");
    }
    ELSE IF(V == LIGHTINOFF) {

        BT.SENDDATA("2");
    }
    ELSE IF(V == LIGHTINAUTO) {

        BT.SENDDATA("0");
    }
    ELSE IF(V == FANON) {

        BT.SENDDATA("5");
    }
    ELSE IF(V == FANOFF) {

```



```

        BT.SENDDATA("6");
    }
    ELSE IF(V == FANAUTO) {

        BT.SENDDATA("7");
    }
    ELSE IF(V == LIGHTOUTON) {

        BT.SENDDATA("8");
    }
    ELSE IF(V == LIGHTOUTOFF) {

        BT.SENDDATA("9");
    }
    ELSE IF(V == LIGHTOUTAUTO) {

        BT.SENDDATA("A");
    }
    ELSE IF(V == SOUNDON) {

        BT.SENDDATA("3");
    }
    ELSE IF(V == SOUNDOFF) {

        BT.SENDDATA("4");
    }
    ELSE IF(V == TEMP) {

        BT.SENDDATA("C");
    }
}
}

```

PROJECT DETAILS

<i>Student Details</i>			
Student Name	JAYESH MEHTA		
Register Number	110907009	Section / Roll No	B- 64
Email Address	JAYESH_M@OUTLOOK.COM	Phone No (M)	9035130243
<i>Project Details</i>			
Project Title	BLUETOOTH BASED HOME AUTOMATION USING ANDRIOD SMARTPHONE		
Project Duration	4 MONTHS	Date of reporting	11-JAN-2016
<i>Internal Guide Details</i>			
Faculty Name	H.SRIKANTH KAMATH		
Full contact address with pin code	Dept of E & C Engg, Manipal Institute of Technology, Manipal – 576 104 (Karnataka State), INDIA		
Email address	SRIKANTH.KAMATH@MANIPAL.EDU		