Online Device Control And Ambient Monitoring Using Mesh Topology

Submitted by:

Raza Nawaz	2014-EE-069
Muhammad Usman Qadeer	2014-EE-070
Mamoon Naseer	2014-EE-071
Shahroze Liaquat	2014-EE-072

Supervised by: Dr. Syed A. R. Kashif / Sir Ali Shafique

Department of Electrical Engineering

University of Engineering and Technology Lahore

Declaration

I declare that the work contained in this thesis is my own, except where explicitly stated otherwise. In addition this work has not been submitted to obtain another degree or professional qualification.

Signed: _____
Date: _____

Signed: _____
Date: _____

Signed: _____
Date: _____

Signed: _____
Date: _____

Contents

List of Figures v

List of Tables vii

Abbreviations viii

Abstract ix

1 Introduction 1
- 1.1 Literature Overview . . . 1
- 1.2 Project Overview . . . 1

2 Motivations and Problem Statement 3
- 2.1 Communication Protocols . . . 4
  - 2.1.1 Bluetooth . . . 4
  - 2.1.2 Wi-Fi . . . 4
  - 2.1.3 ZigBee . . . 5
  - 2.1.4 Z-Wave . . . 6
- 2.2 Proposed Approach – Digimesh . . . 7
  - 2.2.1 Advantages of DigiMesh: . . . 8
  - 2.2.2 Layers of networking protocol . . . 8

3 Methodology 10
- 3.1 XBee Connection . . . 10
- 3.2 API frames in DigiMesh . . . 11
- 3.3 Communication between Microcontroller and XBee . . . 12
- 3.4 Communication between XBee and Raspberry Pi . . . 12
- 3.5 Communication between Raspberry Pi and IBM Watson . . . 13
- 3.6 Communication between Android Application and IBM Watson . . . 13

4 Hardware Implementation 15
- 4.1 Digital Dimmer Circuit . . . 15
  - 4.1.1 Bridge IC . . . 15
  - 4.1.2 Optocoupler 4n35: . . . 15
  - 4.1.3 Optocoupler MOC3021 . . . 16
  - 4.1.4 Triac BT136 . . . 17
- 4.2 Relay Driver Circuit . . . 17
  - 4.2.1 Relay . . . 17

4.2.2 Diode . . . 18
4.2.3 Buffer Circuit . . . 18
4.3 Debouncer Circuit . . . 19
4.4 Atmega328p microcontroller . . . 19
4.4.1 Burn the Bootloader . . . 19
4.4.2 Uploading the sketch . . . 21
4.5 PCB design . . . 21

5 IBM watson IoT platform 23
5.1 Connecting devices to IoT platform . . . 23
5.2 MQTT protocol . . . 24
5.3 Quality of Service . . . 24
5.4 Message Format . . . 24
5.5 Device Side program . . . 24
5.5.1 Publish events . . . 25
5.5.2 Subscribe commands . . . 25
5.6 Application Side program . . . 25
5.6.1 Subscribe events . . . 26
5.6.2 Publish commands . . . 26

6 Android Control 27
6.1 Introduction to Android Studio . . . 27
6.1.1 Project Structure . . . 27
6.1.2 Layout editor . . . 27
6.2 IBM cloud subscription . . . 28
6.2.1 Connecting to the Watson IoT Platform . . . 29
6.2.2 Subscribing to device events . . . 29
6.2.3 Handling events from devices . . . 29
6.2.4 Publishing Commands to devices . . . 30
6.3 Threading in android . . . 30
6.3.1 Thread and Handler class . . . 31
6.3.2 AsyncTask . . . 31
6.4 Retaining data in application . . . 32
6.4.1 Data Storage Options in android . . . 32
6.4.2 Save data in android using SQLite: . . . 33
6.5 Home Automation Application . . . 35

7 Raspberry Pi Control 37
7.1 Setting up Raspberry pi . . . 37
7.2 Serial Communication with Xbee module . . . 39
7.2.1 Sending Data to Xbee module . . . 39
7.2.2 Receiving Data from Xbee module . . . 40
7.3 Connection with IBM Watson IoT Platform . . . 41
7.3.1 Event Publishing . . . 41
7.3.2 Command Handling . . . 41
7.4 Interface with DHT22 Sensor . . . 41
7.5 App Development . . . 42

7.5.1 Functions used 43

8 Conclusion and Recommendations **45**

References **47**

List of Figures

2.1 Automation using Bluetooth module 4
2.2 Network in Zigbee 6

3.1 Block Diagram 11
3.2 XBee pin configuration 11
3.3 Structure of API frame 12
3.4 Flow Diagram 14

4.1 Digital Dimmer Circuit 16
4.2 Relay Driver Circuit 18
4.3 Debouncer Circuit 19
4.4 Arduino ISP sketch 20
4.5 Atmega328p connection diagram 20
4.6 Burn Bootloader Circuit 21
4.7 Burn Bootloader option 21
4.8 back side of PCB design of smart switch board 22
4.9 Front side PCB design of smart switch board 22

5.1 Device credentials 25
5.2 Communication of device and application with IBM Watson platform . . 26

6.1 Android Studio Project Structure 28
6.2 Graphical User Interface 28
6.3 Blocking Queue 29
6.4 working of Thread 31
6.5 Adding Runnables 32
6.6 Dispatching Runnables 32
6.7 AsyncTask Flow 33
6.8 Structure of Content URI 34
6.9 Flow diagram for query, insert, update and delete from SQLite database . 35
6.10 Home Automation Application 35

7.1 Raspberry pi model 2 37
7.2 Raspbian operating system 38
7.3 Connection diagram 38
7.4 Pin configuration of Raspberry pi 39
7.5 Xbee connection diagram 40
7.6 Communication diagram 40
7.7 Connection with IBM IOT platform 42

7.8 dht22 sensor connections 42
7.9 Main screen of the application 43
7.10 Second screen of the application 43
7.11 Third screen of the application 44

List of Tables

2.1 Comparison of LAN, WAN and PAN 3
2.2 Comparison of ZigBee and DigiMesh 9

5.1 QoS levels 24

Abbreviations

HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
IP	Internet Protocol
JSON	JavaScript Object Notation
LAN	Local Area Network
M2M	Machine to Machine
MQTT	Message Queue Telemetry Trasnport
PAN	Personal Area Network
PCB	Printed Circuit Board
QoS	Quality of Service
SQL	Structured Query Language
UI	User Interface
URI	Uniform Resource Identifier
UTF	Unicode Transformation Format
WAN	Wide Area Network
XML	Extensible Markup Language

Abstract

This project presents a design and prototype implementation of a new smart system that is power efficient and uses Wi-Fi technology to remotely control various home appliances. The network comprises of gateway, users and devices connected through MQTT and DigiMesh protocols. Every device is the same, and capable of routing, sleeping for power optimization, and communicating via a mesh network. Xbee modules Series I have been used for forming the mesh network of devices which ensures no single point failure in a network. IBM Watson platform is used for connecting the gateway and the application user. A bidirectional communication is used to control and monitor the devices. An android application has been developed that allows user to control appliances remotely. The status of device can be altered in two ways. One is manual control and the other is by using an app. Corresponding device will respond and the LED turns ON indicating current status of device. Mechanical synchronization is also done to control devices manually even in the absence of internet. In addition to Wide Area Network (WAN), Local Area Network (LAN) has also been created between the devices connected to the same router for more efficient operation of the devices. Common features of the project are appliance control, thermostat control, energy management and backup protection. The project will be implemented in Electrical Engineering Department of University of Engineering and Technology, Lahore.

Chapter 1

Introduction

1.1 Literature Overview

The idea of home automation has been explored by science fiction for decades but first major achievement in the field of home automation was unlocked in the early twentieth century. It started with invention of vacuum cleaner that was engine powered in 1901. Then in the next two decades a series of electronic devices were invented that made human life much easier. Although these devices cannot be called as smart but they were implausible achievements of that time.

The evolution of internet of things in late twentieth century started with invention of ECHO IV which was considered smart unlike electronic home appliances that were developed previously[1]. It could control the states of home appliances as well as its humidity. It could also enumerate the shopping lists. Due to its difficult operation and economical aspects it could not gain recognition at that time. Certain measures were taken to improve the quality of life in last decade of twentieth century including gerontechnology. It was meant for ageing individuals to meet their requirements.

Smart homes came in the range of budget with development of modern technology in the early 2000s. Its applications in scheduling home appliances and video surveillance for security purpose became popular with invention of domestic technologies and protocol devices. These smart devices ensured minimum power consumption which improved its worth in terms of economic aspects for consumers. Also user could remotely monitor and control home appliances that took smart home automation to another level.

1.2 Project Overview

The term smart home intends to provide user energy at affordable price, comfort, convenience and security all the time irrespective of remoteness of user from home. In smart home all electronic appliances can communicate with each other and they are integrated by a centralized control network. These can be controlled and scheduled by the user anywhere in the world with the help of mobile or laptop and internet.

It provide comfort to the user by adjusting itself in the existing home environment i.e. there is no need of change in infrastructure of home. Elderly people and people with disabilities such as handicapped can also carry out various home tasks just with help of mobile phone due to advancement in the field of smart home technologies.
It is getting cheaper in a way that amount invested on home automation can be recovered in few years just because of reduced power consumption. It is now affordable to secure home and control our various home appliances according to customers need.
Our project presents and implements a wireless system with a low cost using Arduino and Raspberry pi microcontroller. This home automation system intend to control various home appliances in home with ease of installation and relatively cheaper due to less power consumption. All appliances are controlled using Wi-Fi technology by creating a local mesh network. This system implements wireless technology and provides remote and access to user to control and timely schedule appliances using android application. Devices can also be controlled manually by using push buttons on smart switch board.

Chapter 2

Motivations and Problem Statement

In this highly competitive world, every individual pursue high quality of life in which there are maximum number of facilities, comfort and security. In home or anywhere in the world an individual needs access to control and manage various electronic appliances, also it is very difficult for elder or physically handicapped people to perform manual operations to change the state of devices present at various niches of home.
A system is required that verifies a user and allow it to remotely control and monitor homer appliances using smart phone where an individual also can check the status of a particular device by getting feedback from it. This system must be able to integrate itself in the existing home environment and is intended to save power.
Internet of Things (IoT) has shown a rapid growth in the recent past years. IoT is the future technology in which all devices in a home are connected to internet and each other and can communicate with each other. In this technology all devices can be controlled and timely scheduled by user from anywhere in the world via voice recognition or with help of android application. There are many protocols used for (M2M) machine to machine communication and data transfer in an internet of things network. Conditions should be checked that which protocol is efficient for our use.
A comparison of few wireless networks and related technologies is given below[2]:

Table 2.1: Comparison of LAN, WAN and PAN

	PAN	**LAN**	**WAN**
Standard	Zigbee, Bluetooth, Wi-Fi etc.	802.11 HiperLAN 2	GS, GSPRS 2.5-3G, 802.16
Speed	less than 1Mbps	11 to 54 Mbps	10 to 384 kbps
Range	short	medium	long
Applications	peer to peer	enterprise networks	Mobile phones, PDAs

2.1 Communication Protocols

RF protocols are used for creating wireless networks. A comparison of different protocols according to their frequencies, bandwidth and networking features which make themselves distinctive from each other according to industrial or individual needs is given below[3]:

2.1.1 Bluetooth

One of the protocol used for home automation system is Bluetooth. Mobile phones Bluetooth wirelessly communicate with Bluetooth module which is connected with Arduino microcontroller. In this way android application control devices and appliances which are connected to controller through relays.

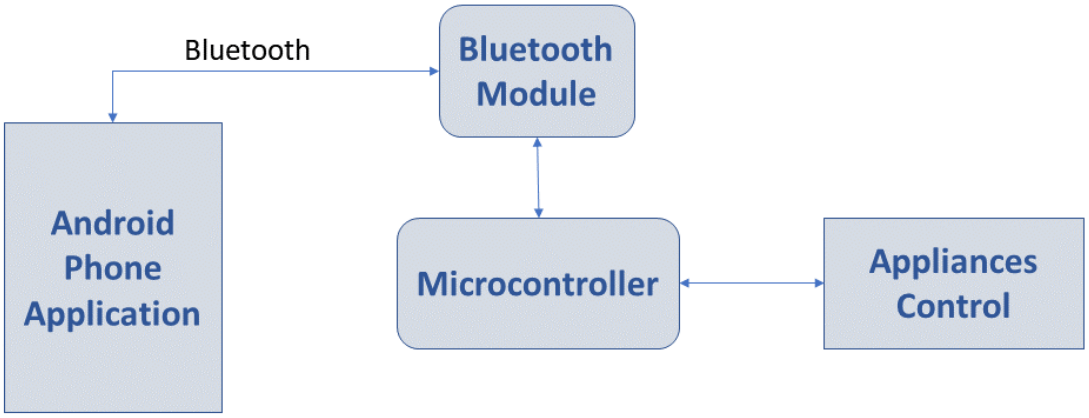

FIGURE 2.1: Automation using Bluetooth module

It has a decent bandwidth and it relatively consumes less power. Bluetooth has the same frequency range as that of Wi-Fi but it has some disadvantages as wel[4].

- It is a very short range communication protocol usually a few meters in pragmatic use.
- It cannot support mesh networking which limits it range and scale.
- It is not a secure protocol for critical infrastructure.

2.1.2 Wi-Fi

Devices in home can also be controlled using Wi-Fi which is also a wireless communication. Android application communicates with Wi-Fi module connected to controller. Electronic home appliances are also connected to microcontroller through a set of relays. Hence all devices can be controlled via application or some website using Wi-Fi. Microcontroller decodes the message sent via app and manages the state of corresponding device[5].

Wi-Fi is a good option to control devices in homes with router and fast data rates are obtained with the help of it because of its larger bandwidth. Also it has better security features as compared to Bluetooth and it meets most of the current networking demands. But it has some disadvantages as well.

- Internet connection is a mandatory need in smart home otherwise devices will not work.
- Drop in wireless signals can cause the devices to operate slowly. Also greater number of devices connected to Wi-Fi reduce bandwidth that also causes slow operation. Separate IPs need to be assigned to every device.
- It is not capable of forming a mesh network.
- It requires a power source, making it inefficient in power consumption perspective.

2.1.3 ZigBee

ZigBee is one of the most popular protocol used in home automation equipment[6]. In machine to machine communication it fulfills economic aspects as well as power consumption issues. Also it provides lower bandwidth with faster data connectivity and it is an asynchronous protocol which enhances its networking features. It supports mesh networking. In mesh networking intermediate radios help in carrying message to desire radio if two radios are not in range of each other. This multi-hopping capability also enhances its range to a great extent. If radio is not present there due to any obstacle or intervention then alternative routes are available for message to transfer. Thus robustness and reliability of network is increased due to these redundant routes. It is also less expensive than other wireless protocols including Wi-Fi and Bluetooth. The categorization of ZigBee is as follows:

- **ZigBee Coordinator:** There is only one coordinator in a network. It has all the information of a network. It provides basis to a network and act as a bridge to other networks.
- **ZigBee Router:** It helps the data to reach its desired location by providing an appropriate path.
- **ZigBee End Device:** It can only coordinate with router and coordinator so it remains in sleep mode and consumes very low power which also enhances battery life.

The communication between coordinator, router and end device is shown in figure 2.2.

ZigBee uses low power radio modules known as XBee for making personal area networks (PAN). XBee is a brand name by Digi for radio communication modules. These small radio modules are programmable. They transmit and receive data with each other with

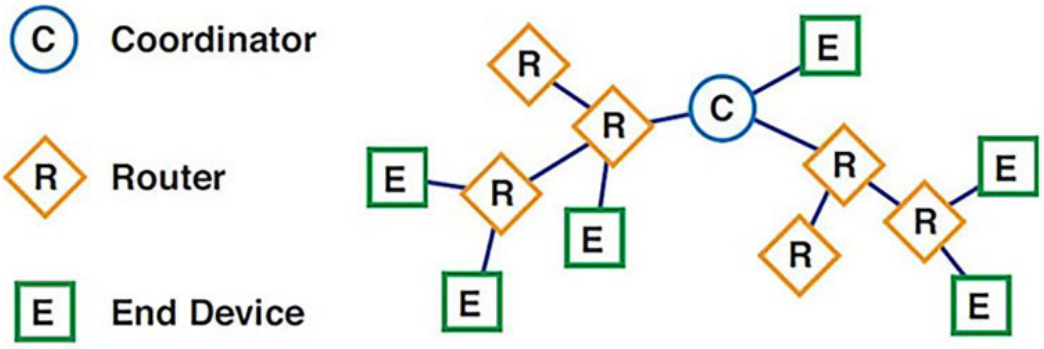

FIGURE 2.2: Network in Zigbee

help of radio frequency signals using ZigBee protocol. In such network, addition and removal of devices takes place without any interruption. When XBee is not active it goes in sleep mode and consumes very less power. Digi XBee ZigBee uses ZigBee protocol and has a range of about 1.2 km.

Despite being the most popular protocol to develop a local network, ZigBee has some disadvantages as well which are as follows.

- One of the major drawback of ZigBee is, it is prone to single point failure. If coordinator which forms basis of a network fails to operate, there is immediate failure of whole network.
- Another disadvantage is disability of nodes to go in sleep mode. They consume energy all the time and hence its efficiency reduces as power consumption increases.
- It is not a secure protocol and there is a high degree of risk for any confidential information from unauthenticated user.

2.1.4 Z-Wave

It is also a wireless communication protocol that supports mesh network similar to ZigBee. It allows the nodes to communicate with each other whether they are in range or not by linking with nodes that are in range to reach the desired location. It also uses radio frequency waves for signaling between two devices. Other than main central device which is connected to internet, all devices uses Z-Wave technology to communicate with each other and central node. It uses two controllers, one for providing source and home ID and the other controller executes the routing table.
It is a cheaper alternative to ZigBee and Wi-Fi because it power consumption is very less. Also Z-Wave devices can exchange information with other wireless devices in internet of things platform. It also has no interference with Wi-Fi because it works on separate frequency band but it has disadvantages as well.

- Its data transfer rate is 100 Kbps which much lesser as compared to other protocols.
- It has limited range. To increase range more Z-Wave devices are required which enhances overall cost.
- It supports only 232 nodes which are very few as compared to ZigBee.

There are some other protocols and technology platforms that support home automation which includes Insteon, Universal Powerline Bus, Thread, Apple Homekit etc. They have their pros and cons as well. Therefore a protocol is required that has ability to cop up with all demerits of other protocols. It should meet the following requirements:

- Cost effective
- Low power consumption
- Appropriate data transfer rate
- No interference issues
- Better device interoperability
- Support larger number of devices

2.2 Proposed Approach – Digimesh

DigiMesh[7] also supports mesh networking topology in which each node is connected to other nodes around it on a network. In this topology nodes can directly communicate with each other through radio frequency waves and provides the path to data to desired location. All devices in a network have equal importance and are of same type and can interchange place with each other. All nodes can go in sleep mode when they are idle which is major advantage in power optimization and improves its efficiency. It has several advantages over other protocols and has ability to cope up many of the problems faced by other protocols.
DigiMesh network works without any parent child relationship and devices in a network cooperate with each other to route data or message to desired location. Similar to ZigBee radio frequency modules here form a network. Unlike ZigBee there is no coordinator (central unit) or any node that forms the basis of a network which means network is not entirely dependent on central node whose failure may cause shut down of entire system. Differences between ZigBee and Digimesh are described in table 2.2 [8]. There is no complicated architecture to define nodes in a network and devices can communicate directly.

2.2.1 Advantages of DigiMesh:

- This network develop itself automatically without any kind of interference of human. No central node exists in such network. There is no effect on addition or removal of a device on network. Nodes can interchange place with each other. This called as Ad hoc network creation.
- It is really efficient in power consumption because all nodes go in sleep mode when they are idle.
- All nodes can directly communicate with each other and hierarchal relationships are needed. This is also referred to as peer to peer architecture.
- If any node goes out of operation due to any interference then network automatically provides an alternative path to data or message to reach desired location which is called self-healing of a network.
- When message reaches a desired location, acknowledgement is received which ensures data has been delivered successfully.

2.2.2 Layers of networking protocol

There are seven different layers of networking protocol namely:

- Layer 1 : Physical Layer
- Layer 2 : Data Link Layer
- Layer 3 : Network Layer
- Layer 4: Transport Layer
- Layer 5 : Session Layer
- Layer 6 : Presentation Layer
- Layer 7 : Application Layer

DigiMesh works on third layer which is known as network layer. Based on different factors such as condition of network, priority of service this network layer provides path to the data to reach its final destination. Based on information (source and destination address) network layer decides whether to move data to transport layer or shift it to previous layers. If desired destination is reached then data is converted to packets and sent to transport layer otherwise it updates the destination address and data to lower layers. In this way this layers do addressing and routing.

Table 2.2: Comparison of ZigBee and DigiMesh

	ZIGBEE	**DIGIMESH**
Node types and their benefits	Multiple: Coordinators, Routers, End Devices.	Single: One homogeneous node type, with more flexibility to expand the network.
Battery Deployed Networks	Coordinators and routers must be mains powered	All nodes are capable of battery operation and can sleep, No single point of failure
Over-the-air firmware updates	Yes	Yes
Range	2 miles for each hop	40+ miles for each hop.
Frame payload	Up to 80 bytes.	Up to 256 bytes,
Supported frequencies and RF data rates	Predominantly 2.4 GHz (250 kbps)	900 MHz (Up to 250 Kbps), 868MHz, 2.4 GHz (Up to 250 Kbps)
Security	128-bit AES encryption. Can lock down the network and prevent other nodes from joining.	Both 128 and 256-bit AES encryption. Can lock down the network and prevent other nodes from joining.
Interference tolerance	Direct-Sequence Spread Spectrum (DSSS).	900 MHz: Frequency-Hopping Spread Spectrum (FHSS). 2.4GHz: Direct-Sequence Spread Spectrum (DSSS).
Addressing	Two layers. MAC address (64 bit) and Network address (16 bit).	MAC address (64 bit) only.

Chapter 3

Methodology

Mesh topology is implemented with the help of smart switchboards which would replace the original switchboards of home or any place without any change in infrastructure. Each switchboard contains an ATMEGA328p chip as microcontroller and XBee module. The presence of XBee module makes each switchboard a single node in a mesh network. All electronic appliances (fans, lights etc.) are connected to switchboard through relays. Push buttons are present on switchboards for manual switching of electronic appliances. All switchboard are connected to each other due to radio frequency communication between XBee modules connected to each switch board.
As in Internet of Things all devices are connected to each other and exchange data with the help of internet so a gateway is required to connect the devices to the internet. Raspberry Pi microcontroller in our project acts as a gateway. XBee connected to this microcontroller communicates with XBee connected to any of the switchboard in a mesh network to check the status of device. All information is then sent to cloud by it. IBM Watson platform is used to store and manage all the data. This platform is chosen because it is secure, cheaper and provides extensive documentation (user friendly). All information between Raspberry Pi and cloud is exchanged with MQTT protocol. MQTT is publish/ subscribe based messaging protocol to connect devices with servers and clients. Android application is also connected to IBM Watson, through which status of devices can also be changed. Information is sent to cloud. XBee on Raspberry pi that is connected to cloud communicates with respective XBee in a mesh network and Arduino controller decodes the information and change the status of respective device on a switchboard.
The block diagram showing bidirectional communication between devices is given in figure 3.1:

3.1 XBee Connection

The XBee S1 module has 5 pins.The two pins 5V and GND are for powering the module.The pins TX and RX are for the serial communication with the other micro controllers.The pin configuration of the XBee module is shown as

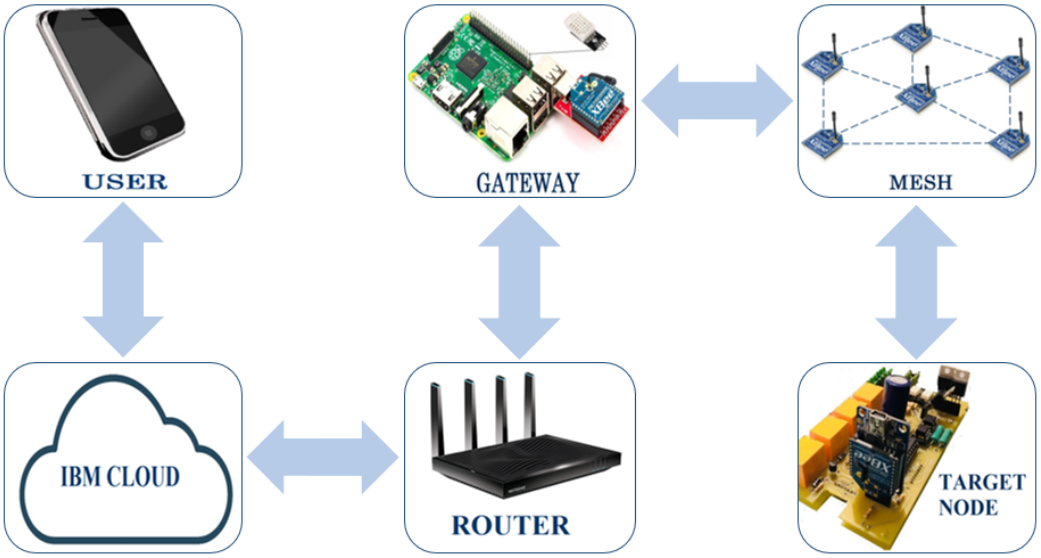

FIGURE 3.1: Block Diagram

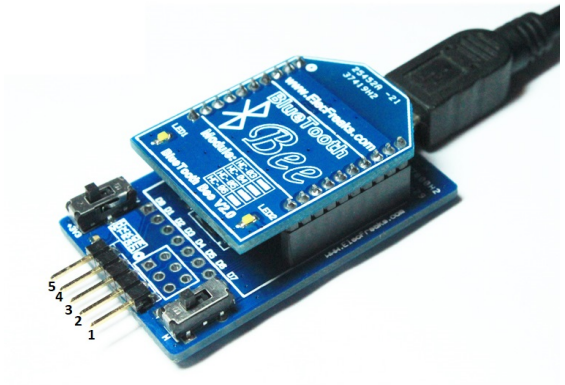

FIGURE 3.2: XBee pin configuration

where pin 1=5 V, pin 2=GND, pin 3=TX, pin 4=RX

3.2 API frames in DigiMesh

There are two modes for operation of XBee. One is transparent mode and other is API mode. API mode has been used. In API frame, start delimiter indicates the start of the frame. Every transmit request frame begins with the hex value of 0x7E which corresponds to the start delimiter. The length of the frame has two bytes the most significant byte(MSB) and the least significant byte(LSB).These two bytes indicate the total length of the API frame. Then there are data bytes. The 64 bit destination address corresponds to the mac address of the device and the RF data field is the message part of the frame. The checksum byte tells the validity of the frame. The structure of API frame[7] is shown in figure 3.3. The method for calculating the checksum is as follow.

- Leave the start delimiter and the two length bytes,sum all the remaining bytes of the frame

- Take the bitwise AND of the sum with 0xFF to keep the last 8 bits of the sum
- Take the difference with 0xFF to get the checksum value.

$$Check\ sum = \texttt{0xFF} - (\sum data\ bytes)\ \ \&\ \ \texttt{0xFF}$$

Start Delimeter	Length		Frame Data							Checksum
			Frame Type	Data						
1	2	3	4	5	6	7	8	...	n	n+1
0x7E	MSB	LSB	API FRAME TYPE	DATA						Single Byte

FIGURE 3.3: Structure of API frame

3.3 Communication between Microcontroller and XBee

Both Atmega chip and XBee module are present on a switchboard. Devices whose status is meant to be changed are also connected to switchboard through relays. XBee module is configured with help of XCTU software. Serial communication is done for Arduino to talk with XBee using software serial library to develop a mesh network. Usually XBee remains in sleep mode but when there is transition in the status of device, it becomes active and receives the frame from another XBee device in a mesh network wirelessly (via radio frequency wave) and goes back to sleep mode which makes this entire process highly energy efficient. Arduino controller collects information via UART and decodes the information to change the status of respective device.

3.4 Communication between XBee and Raspberry Pi

XBee connected to Raspberry Pi and any XBee in a local mesh network wirelessly communicate with each other by exchanging API frames. Command parameters in API frame configure XBee which is energy efficient as well as reliable over AT commands where a single command parameter is given at a time Information or data from Arduino microcontroller (any of the XBee node) is wirelessly transmitted by enclosing data with parameters such source address, destination address, checksum etc. so XBee on Raspberry Pi receives it without error.

Then UART serial communication is done between Raspberry Pi and XBee. Controller decodes the required information. Raspberry Pi acts as a gateway and sends the required information received from mesh network to cloud. Similarly Raspberry Pi can also transmit information received from cloud to mesh network to take the required action through XBee communication. In this way bidirectional communication is done.

3.5 Communication between Raspberry Pi and IBM Watson

Raspberry Pi acting as a gateway make sure that all information it receives from nodes in a local mesh network is sent to the cloud. This gateway can send or receive its own data or the devices connected to it with cloud. To ensure availability of data for client at right time and with full security, IBM Watson cloud platform is used. IBM Watson provides access to Internet of Things devices and data. It interconnects the devices via gateway over internet. Communication between Raspberry Pi and cloud is done with help of MQTT protocol. MQTT (Message Queue Telemetry Transport) is publish /subscribed based messaging protocol.
Whenever there is change in the status of device sensed by Arduino controller it sends information to Raspberry Pi which publishes event to IBM Watson cloud platform with the help of MQTT protocol. A java application sense it and make changes by subscribing to events. It takes the required action by sending information back to Arduino through cloud and gateway. Quality of service is selected as 2 for the delivery of message to be assured.

3.6 Communication between Android Application and IBM Watson

An android application written in JAVA detects any change in the status of device by subscribing an event from IBM Watson cloud platform through MQTT protocol published by Raspberry Pi. There is also a scan option in application to detect the total number of devices in a network. User can control the status of device remotely by sending commands to IoT platform through MQTT protocol. Gateway can access these changes by subscribing commands from cloud. And then information is sent from gateway to controller via XBee communication to take the desired action i.e. to change the status of device connected to controller through relays. In this way bidirectional communication is done. Further details of communication of android application with IBM Watson Iot Platform is given in section 6.2

The flow diagram describing how the the system of automation work is shown in figure 3.4. Transition in status of device can be done in two ways as follows:

- Manual control (by using push buttons on switchboard)
- By using android application

Mechanical synchronization is done to provide access to user to control devices both manually and remotely. Device status can be changed from anywhere in the world by using android application provided internet is available. If it is not available devices can be controlled by using pushbuttons available on switchboard. On pressing button, device connected to the pin of Arduino controller through relays make a transition in

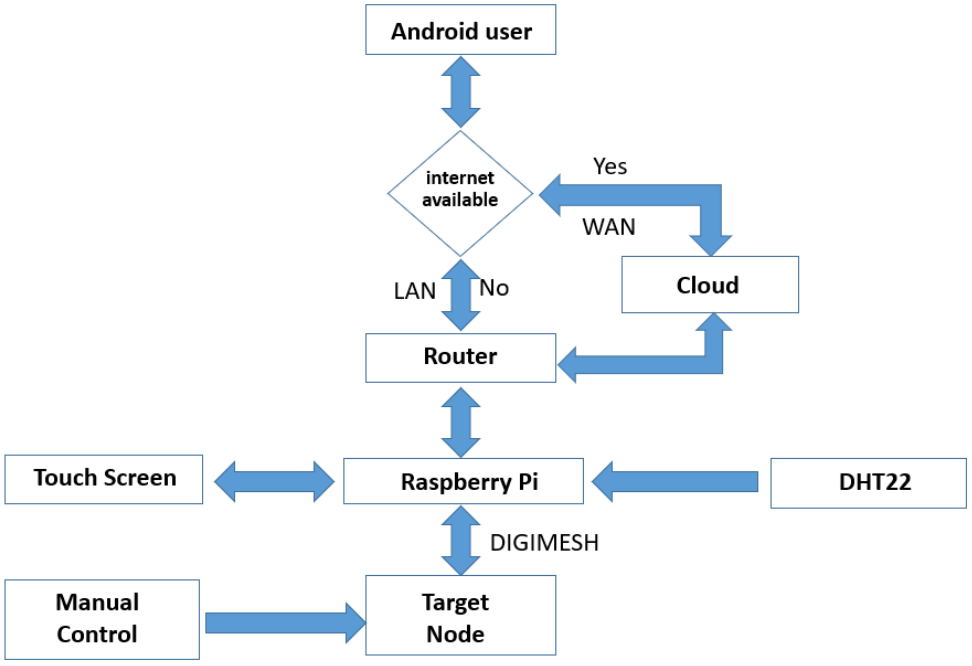

FIGURE 3.4: Flow Diagram

its status. Also local area network (LAN) can also be used if internet is not available. User can also control appliances using android application. When user change status of device using application, changes in data are stored in cloud connected to application through a protocol. Raspberry Pi connected to cloud gets information though it and send this information to desired device in a mesh network via Xbee to make a transition in the status of device. In this way both manual and remote access is provided to user.

Chapter 4

Hardware Implementation

A smart switch board has been designed. It consists of two layered double-sided PCB (front side and back side). The PCBs are designed using proteus 8 professional (version 8.6). The switch boards back side mainly comprises of

- Digital dimmer circuit
- Relay driver circuit
- Debouncer Circuit
- Atmega328p burner circuit.
- Xbee

The front side mainly consists of LEDs, push buttons with de-bouncer circuit. The front side plays the role of manual operation.
The design of different circuits are discussed below:

4.1 Digital Dimmer Circuit

Digital dimmer circuit consists of zero crossing detection circuit[9]. There are two digital dimmer circuits fabricated on the PCB and they are used to operate the fans. The components used in digital dimmer circuit are briefly discussed below:

4.1.1 Bridge IC

Single phase bridge rectifier IC has been used to convert WAPDA supplied AC voltage to DC voltage. DI106 IC has been used whose maximum input voltage is 420V i.e greater than 240 volts.

4.1.2 Optocoupler 4n35:

Microcontroller is operating at the very low voltage level and there is huge current form the AC input which if flows back toward the microcontroller may damage the

microcontroller, so there is need to protect it. Also for the dimming operation detection of zero crossings of the AC input is required. Optocoupler 4n35 servers fulfilled these two purpose at very low cost. This is the opto coupler used to isolate the ac voltage level from the micro controller in order to protect it and separating the two optically.

Operation:

When dc output from bridge IC is applied across the diode circuit of 4n35 it emits infrared light which then falls on the photo-transistor and thus conductivity occurs. Base pin of the transistor is opened so that there is maximum conductivity. Whenever there is interruption in the input, then there is no light and thus conductivity stops. Because of this characteristics, we are using it for the zero crossing detection and this output is then fed to the micro-controller which is then further processed to control the dimming operation.

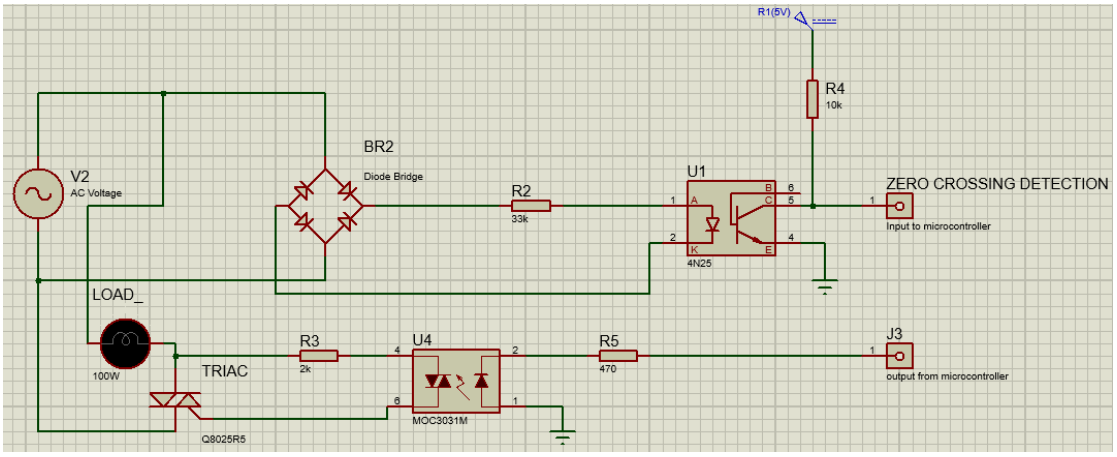

FIGURE 4.1: Digital Dimmer Circuit

4.1.3 Optocoupler MOC3021

The load for which this switchboard is being designed requires AC input. There is need to process the input in both cycles positive and negative so that an AC output is generated for the load. So MOC3021 serves this purpose because it has diac in its secondary circuit. This is the opto coupler used to provide the triggering pulse to the triac in order to operate the load. It also separates the ac voltage circuit from the microcontroller because the huge current flowing from the AC may damage the microcontroller.

Operation:

MOC3021 uses a diac in its secondary circuit instead of a photo transistor. Diac conducts in both directions either cycle is positive or negative. Input is applied to MOC3021 at its diode circuit from the micro-controller and it emits a light which falls upon the photo-diac and the output current flows through the circuit containing photo-diac, resistor and through one of the main terminals of triac and thus completing the loop through the gate of the triac. This operation is done for both cycles of ac because diac is bi-directional device.

4.1.4 Triac BT136

This circuit is used to control the input given to the load so there is a need of element which controls the amount of input at the instant we want, so Triac is the device which serves this purpose. For the triggering action triac is used in this circuit. BT136 is used for this purpose having two main terminals and the gate terminals. The pulse is applied at the gate terminal.

Operation:

The output from the MOC3021 is applied at one of the main terminals and at the gate. The other main terminal is connected with the ac supply. It conducts when it is triggered through the gate and in both directions. For firing at the we need a zero-crossing detector which is done by using 4N35 opto-coupler. This reference is then used to provide the desired delay which is generated by using microcontroller. At every zero-crossing there is an interrupt generated by the controller and it is used as a reference, then a desired delay is provided through the microcontroller. This technique is used to control the voltage provided to the load devices (in this case they are fans) and thus controlling their speed digitally.

4.2 Relay Driver Circuit

Relay driver circuit consists of relays, diodes, transistors and resistors. There are four relay driver circuits used in the PCB in order to operate four lamps.

4.2.1 Relay

The load for which this switchboard is being designed needs huge amount of current to operate. But the input has very low current so relay is the only device which require less amount of current at the input and gives large value of current at the output. The relay used in the circuit is operated on 12V. There is switching operation done by the microcontroller to switch the device state from on to off state and vice versa. So for this switching operation done we need relay.

This type of relay has five pins. Two pins have coil in between them, one of the remaining pins is common pin which is movable, and other two pins are fixed contacts named as normally closed and normally open. Relay operates on the principle of electromagnetic induction followed by the energization of coil and then lead to opening and closing of contacts.

When input is applied to coil contacts, it energizes the electromagnet (coil) producing magnetic field. This magnetic field, thus attracts the common pin which is in the normally open state to close its contacts and go towards normally closed state, thus giving the output. When the coil is de-energized then common pin goes to the normally open state.

4.2.2 Diode

The diode which is used in this relay driver circuit is a simple diode which is 1N4007. Diode is the element in electronics which conducts only when positive terminal of voltage is at its P side of PN junction and negative terminal of the voltage is at its N side of PN junction and this state is called forward bias, when the polarity of the voltage is reversed then it will not conduct and this state called reverse bias.
Because the relay coil is an inductor and it resists the sudden change in the current, so when the relay is de-energized there will be unnecessary tripping of the relay. To overcome this problem a diode is used in parallel with the relay providing a path for the current which blocks the current flowing back into the relay because diode is in its reverse bias state.

4.2.3 Buffer Circuit

The transistor used in the buffer circuit circuit is 2N3904. As we know that one of the transistors functions is switching when used in the saturated state and cut-off state. In this operation only a small amount of current at the base causes a large amount of current at the collector which is proportional to the voltage and transistor is ON (saturation). If base current is zero, then there will be no output at the collector and transistor will be OFF (cut-off).
To on or off the device an output is given by the microcontroller. As it is from controller so this 5V (1) or 0V (0) and it is in digital state. Relay is operating at 12V and we need to do the switching in analog mode so thats why we need transistor scheme to shift from 5V to 12V and also for switching.

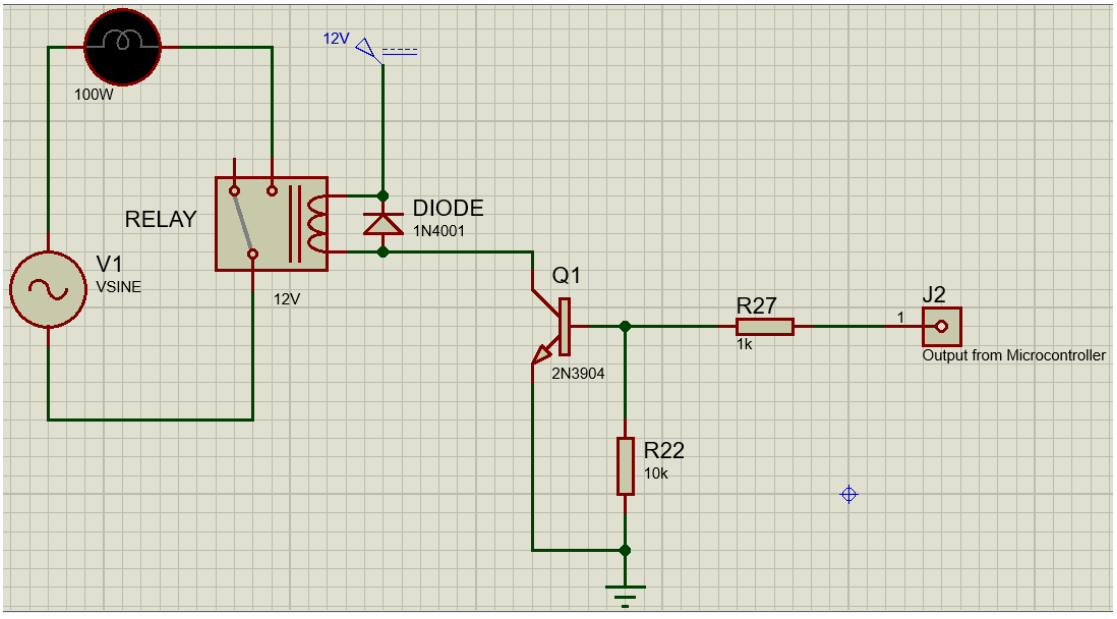

FIGURE 4.2: Relay Driver Circuit

4.3 Debouncer Circuit

When a push button is pressed, then it may be pressed many times because it is mechanical, it can generate multiple signals so we need to make sure that a button is pressed or not. Thats why we need De-bouncer circuit to be used with every push button . De-bouncer circuit contains a resistor(10k) and a capacitor(100nF). The product of R and C generates a delay of 1ms as:

$$t = RC = 10k \times 100nF = 1ms$$

Whenever a button is pushed a delay of 1ms is generated, it will wait for 1ms to make sure that whether the button is pressed or not and checking for the output at the button that it is either 1 or 0. 1 means that the button is pressed and zero means the button is not pressed.

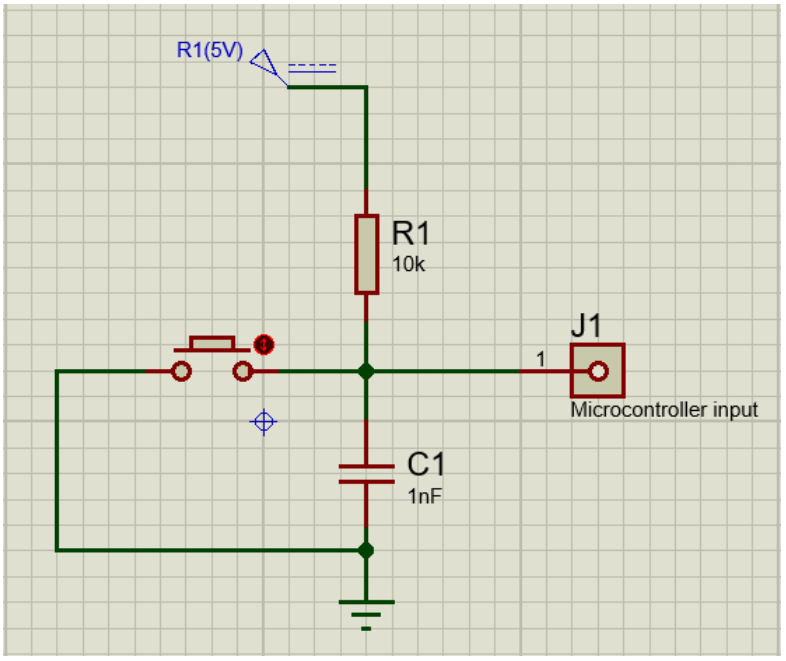

FIGURE 4.3: Debouncer Circuit

4.4 Atmega328p microcontroller

The Arduino Uno has microcontroller chip Atmega328p connected to the board along with the clock source and header for the users to access the general purpose input output pins of the controller. The chip Atmega328p can be used independently of the board and all the pins of the chip can be used on the external hardware circuit.The following are the steps to use micro controller chip externally to any circuit.

4.4.1 Burn the Bootloader

If the new Atmega328p is used then the first step is to burn the bootloader.If the chip is already bootloaded then there is no need for this step.To burn the bootloader the follow

the steps as

- Open the arduino IDE.From File select the Examples and find the sketch ArduinoISP.Upload this sketch to the Arduino board.Before uploading select the compatible COM and the board type of the micro controller.

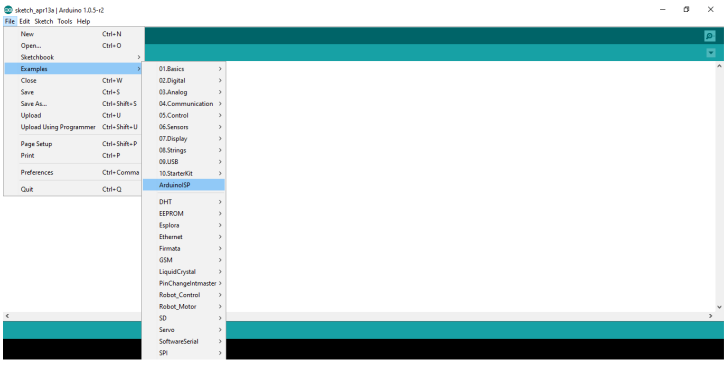

FIGURE 4.4: Arduino ISP sketch

- Connect the circuit as shown in the figure **??**

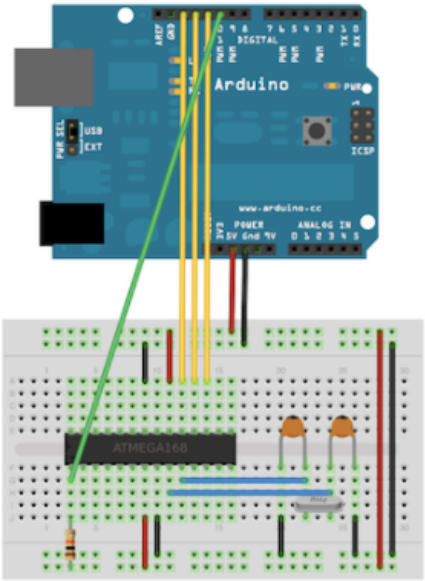

FIGURE 4.5: Atmega328p connection diagram

The capacitors used are 22 pico farad capacitors.The oscillator is of the value 16 MHz.Moreover a 10k resistor is connected with the reset pin of the chip.

- From Tools select Board and from Board select Arduino Duemilanovew/ ATmega328.

- Select Arduino as ISP from the Programmer.

- From the Tools click on Burn Bootloader.This will take some time and after completion the IDE shows a message which indicates that the bootloader is burned successfully.Once this step completes the chip is ready to be used externally.The following figure shows the different menu options in the IDE

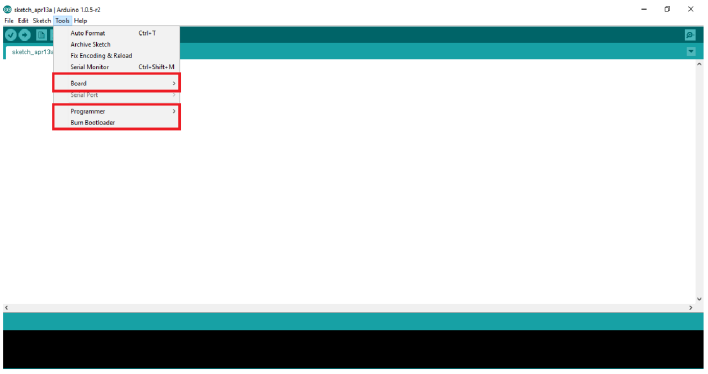

FIGURE 4.6: Burn Bootloader Circuit

4.4.2 Uploading the sketch

To upload the sketch remove the chip from the arduino board.To burn bootloader on a new chip arduino board having already bootloaded Atmega328p is required.But once the first step is complete the chip from the arduino board is removed.After removing the chip connect RX and TX of the board with the pin number 2 and 3 of the external Atmega328p respectively.
Select Arduino Duemilanovew/ ATmega328 from the Board and then upload the sketch as usual by clicking on the upload button.The following diagram shows the connection of the external Atmega with the board for uploading the sketch

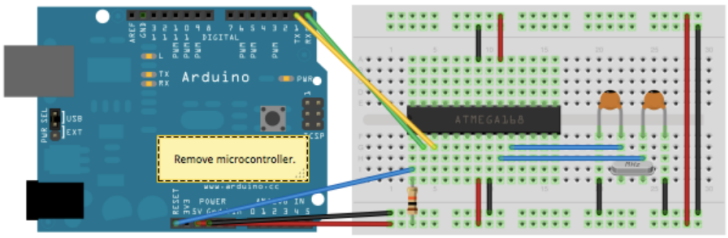

FIGURE 4.7: Burn Bootloader option

4.5 PCB design

PCB of smart switch board has been designed on Proteus. It has been designed in such a way that it can replace usual switch boards used in homes. Its dimensions are 13cm by 7cm. There is back side of PCB which contains relay driver circuit, arduino to breadboard circuit, digital dimmer circuit and XBee. The front side of PCB contains

push buttons, leds to show status of appliances and potentiometer for speed control of fans.The back side is shown in figure 4.8 and front side is shown in figure 4.9.

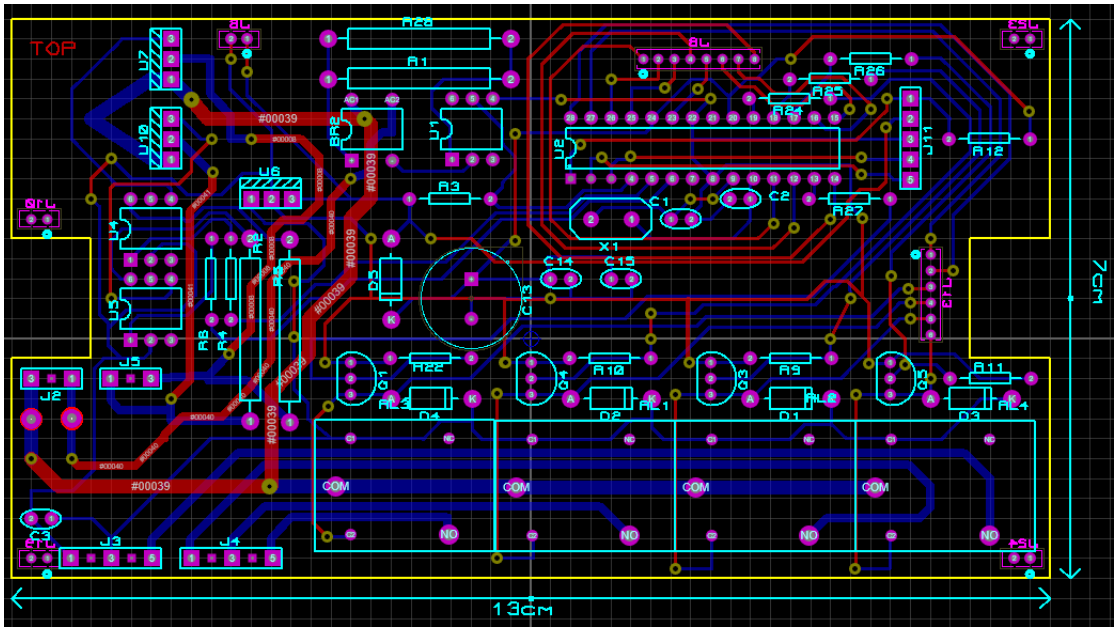

FIGURE 4.8: back side of PCB design of smart switch board

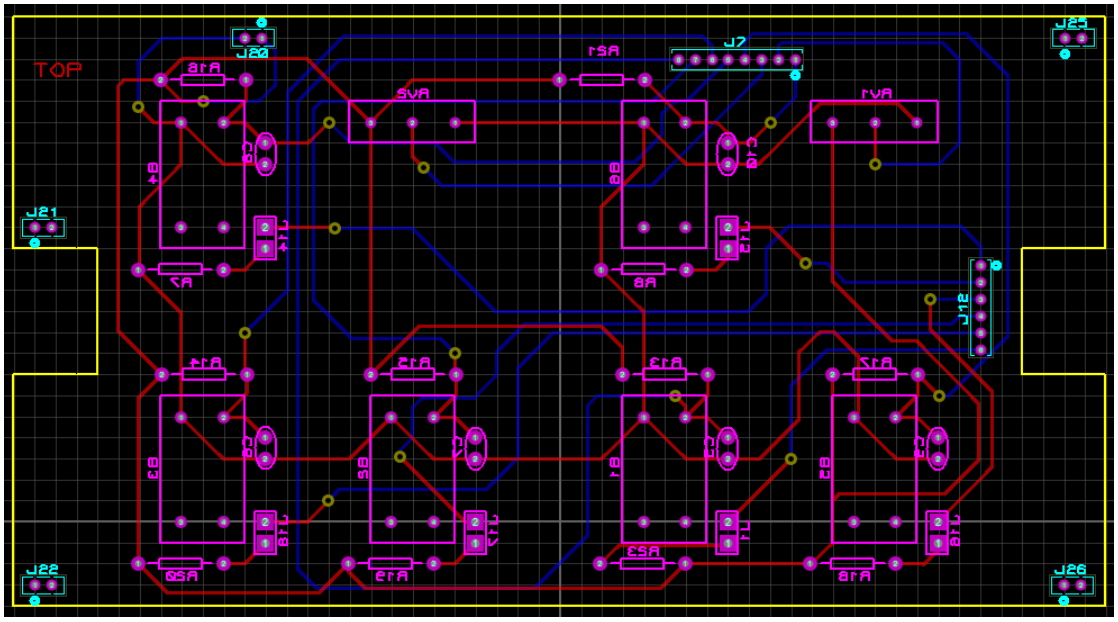

FIGURE 4.9: Front side PCB design of smart switch board

Chapter 5

IBM watson IoT platform

Internet plays a significant role in establishing interaction between user and appliances to be controlled. Cloud provides various application services and hence it is an important part of Internet of Things Based on their services including development of application, data management, monitoring and security of data, visualization of data patterns etc. there are many famous IoT cloud platforms that are trending including Arryant connect, Echelon, IBM IoT, Oracle IoT cloud, Intel IoT, Texas Instruments etc. In account of specifications and services provided by cloud platforms, IBM Watson is preferred over other platforms due to following reasons

- Extensive documentation of Python and Java is available for application development[10].
- It provides a secure interaction between user and devices.
- It is economical as compared to other platforms.

IBM Watson platform is basically a service provided by IBM Cloud which can be accessed by using dashboard in IBM cloud. It allows application for device and data management in IoT. Large number of devices (along with gateway) can be connected via Watson IoT platform and cloud enables us to save and ingress device data whenever required.

5.1 Connecting devices to IoT platform

Devices can be connected to IBM watson platform by HTTP or MQTT protocol. MQTT protocol has been used that will be explained in next section.
First create an instance of Watson platform by initiating a new service in Internet of Things platform using launch button. IoT devices are connected to IBM Watson platform to receive data and configure IoT devices[11]. Devices are created by using add device option in cloud platform and putting the required registration information such as network ID, device type etc. Device to be connected must meet all standard required by platform. Save the device information as it is required for a device to connect to platform. Instead connecting each and every device, a gateway device can also be

connected to IBM Watson platform. Gateway device directly send and receive its data on cloud and also manages data of devices connected to it. Devices such as gateway and applications connects to cloud using MQTT protocol.

5.2 MQTT protocol

MQTT (Message Queue Telemetry Transport) is a compatible messaging protocol based on publish/subscribe methodology[**?**]. It is a flexible network protocol for communication of gateway and application with cloud and is very effective for communication at long distances. Bidirectional communication is made possible with cloud by publishing and subscribing commands or events by gateway and application. IT collects the data and send data to IT infrastructure.

5.3 Quality of Service

It measures and evaluates performance of cloud computing service based on parameters like transmission rate, packet loss etc. It provides priority to different data flows according to requirement to maintain high efficiency[12]. QoS level 0 is fast but it does not assure delivery of message. Whereas as level progress it assures delivery of message with delayed speed. A description of QoS level is as follows

Table 5.1: QoS levels

QoS	**Description**	
0	At most once	Fire and Forget
1	At least once	Acknowledged Delivery
2	Exactly once	Assured Delivery

5.4 Message Format

Message used for communication is written in JSON format. JSON (Java Script Object Notation) is a syntax for saving data. Data is exchanged in JSON format between application or gateway and cloud.
JSON object is key-value data in curly braces. For example:
{
name : board1
mac: 000000000000FFFF
message: onAll
}
On left side of colon are JSON keys and on right side are the JSON values. JSON value can be a string, integer, object, array or boolean.

5.5 Device Side program

Raspberry Pi is a device used for communication with cloud via MQTT protocol by publishing events and subscribing commands. The following are the parameters of the device.

- Organization ID
- Device Type
- Device ID
- Authentication method
- Authentication token

The following figure shows the different parameters value of Raspberry Pi as a device.

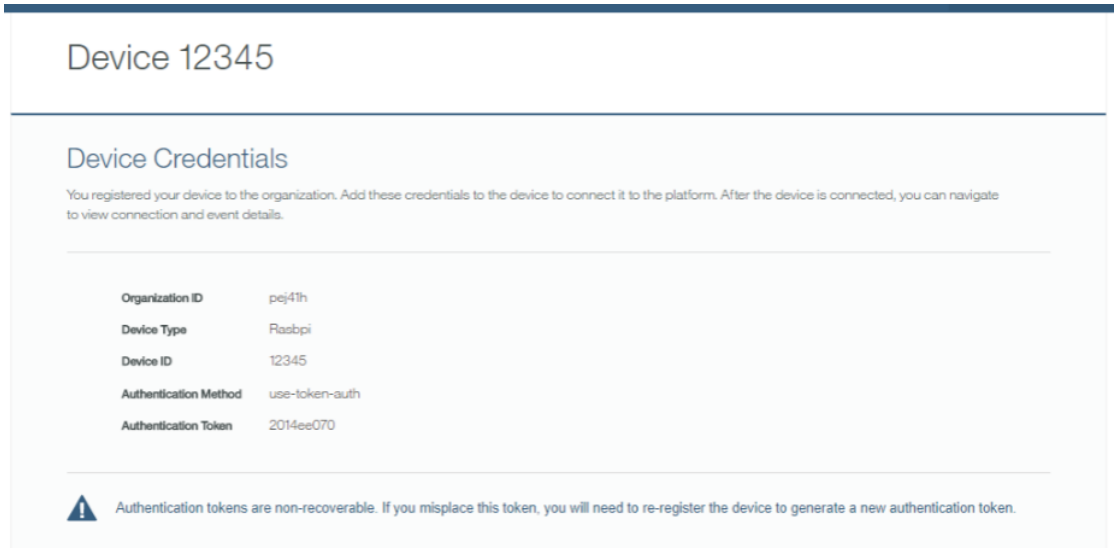

FIGURE 5.1: Device credentials

5.5.1 Publish events

Raspberry pi publishes event on IBM cloud platform with the help of MQTT protocol. Event is basically a counter published by a device to cloud updated at every five seconds. IBM cloud platform publishes this event so that application connected to cloud may subscribe to events and update any changes by accessing device data.

5.5.2 Subscribe commands

Raspberry pi connected to cloud receive commands from android application registered in same platform. Command handler thread is used to subscribe commands that are published by user on android application and that is also done via MQTT protocol. A separate queue is used for pushing and reading commands in order to receive commands in sequence.

5.6 Application Side program

An android application is developed to communicate with cloud that publishes command as well as subscribe events.

5.6.1 Subscribe events

An android application developed and connected with cloud platform receives event broadcasted by cloud and published by gateway via MQTT protocol. It subscribes events in order to update its information such as device status in case if any manual operation is done. Gateway publish event if it detects any change of device data on cloud for the application to subscribe it.

5.6.2 Publish commands

Whenever status of device is changed by user on android application, a command is published on cloud by passing device information on cloud via MQTT protocol. It is a message written on JSON format that is subscribed by gateway to update the status of respective device linked to it.

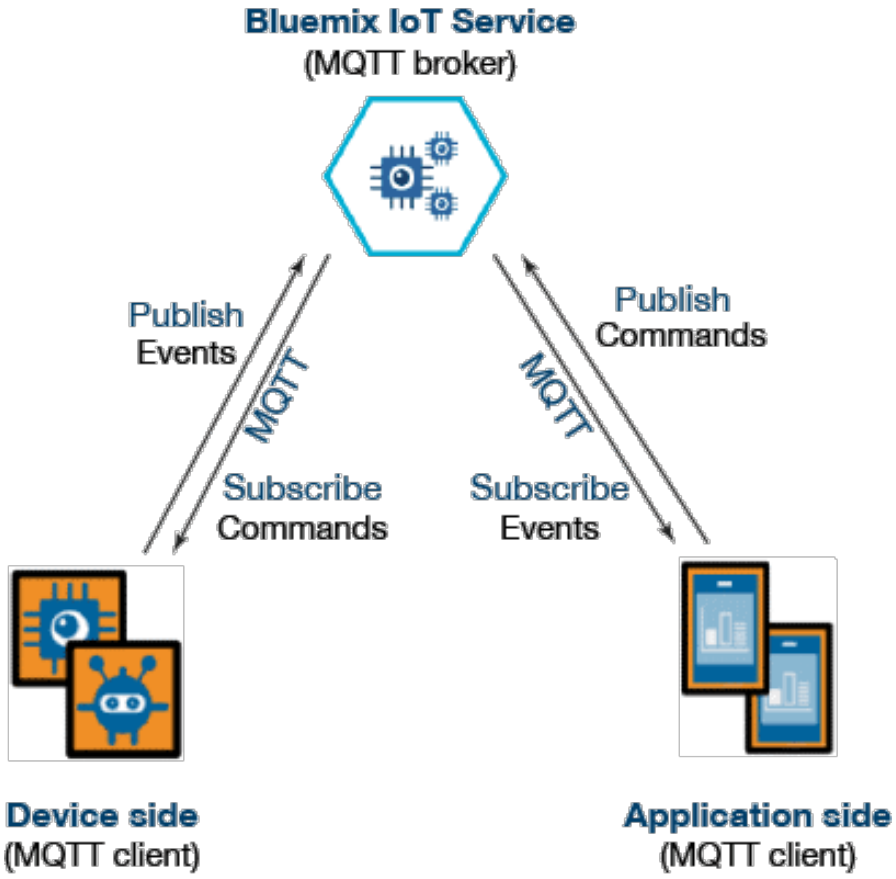

FIGURE 5.2: Communication of device and application with IBM Watson platform

Chapter 6

Android Control

Android application has been developed to control and monitor appliances remotely. The application has been developed using Android Studio.

6.1 Introduction to Android Studio

Android Studio[13] provide fastest tools for the development of apps on any type of android device.It is official Integrated Development Environment for Android application development.
Besides Android Studio, developers also use Eclipse. However, Android studio is designed particularly for android development to make it simpler and faster. On the other hand, Eclipse can be used for distinctive platforms and cannot adpat absolutely for android app development. Some features of android studio are discussed below:

6.1.1 Project Structure

A project in android studio contains source code files and resource files. A project contains the following folders:

- **manifests:** Contains the AndroidManifest.xml file.
- **java:** Contains the Java source code files and JUnit test code.
- **res:** Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

All build files are visible under Gradle Scripts. The project structure is shown in figure 6.1.

6.1.2 Layout editor

The layout and graphical user interface can be designed either by writing code in xml or by drag and drop UI components. Drag and drop feature of android studio is shown in figure 6.2. It also gives an option to preview layouts on multiple screens configurations. Other features of android studio includes:

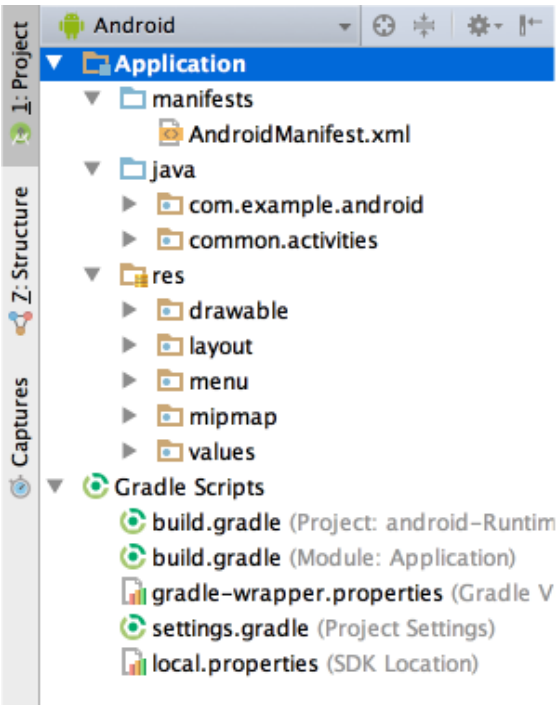

FIGURE 6.1: Android Studio Project Structure

- gradle based build system
- A fast emulator
- Instant run to make changes to a running app
- C++ and Javascript support

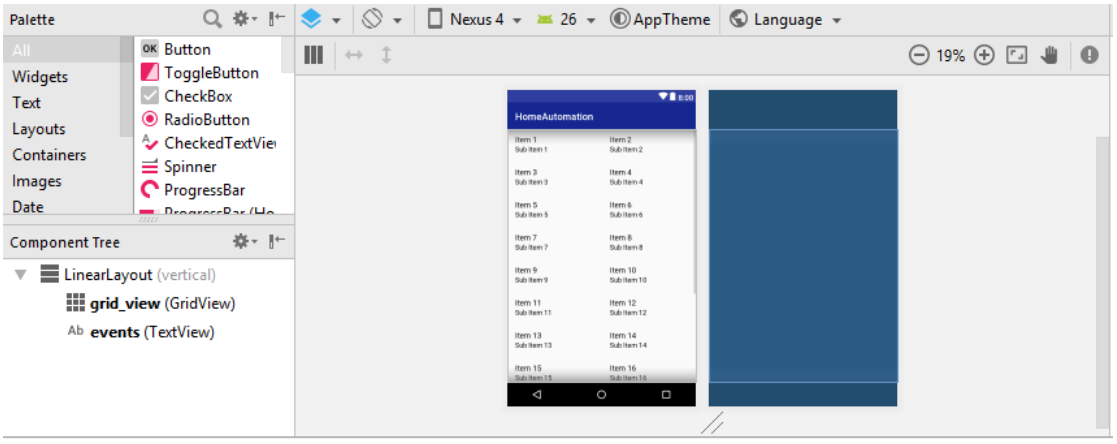

FIGURE 6.2: Graphical User Interface

6.2 IBM cloud subscription

IBM watson platform provides library for Java developers. IBM Watson has provided the documentation under the heading, Java for application developers.

6.2.1 Connecting to the Watson IoT Platform

In order to connect an application to IBM Watson Iot plarform, first of all, an instance of application client is created and device registration details are passed to it. Then **connect()** function of application client is called that connects the application client to the IBM Watson Iot plarform. If device registration details are false, an MqttSecurityException occurs.
There is a method **setKeepAliveInterval(int)** that can be used to set maximum interval of time between messages sent or received.The application client ensures that at least one message travels across the network within each 'keep alive' interval period. If number of data-related messages are zero during the timeout period, the client sends a 'ping' message. The default value of the interval is 60 seconds. In order to disable the 'keep alive' processing feature on the client, setKeepAliveInterval(int) value is set to 0. After succesful connection to Watson Iot platform, application client can subsscribe to device events and status and can also publish commands.

6.2.2 Subscribing to device events

Application can subscribe to any of the events from any of the connected device. By default, application clinets function **subscribetoDeviceEvents()** subscribe to all events from all connected devices.In order to subscribe to particular event or particular device, device ID, event or message format etc. can be specified as arguments of the function. In our case, application client subscribes to events published by Raspberry Pi, and the events are published in JSON format. These specifications can be given as argument to the function.

6.2.3 Handling events from devices

There is an event callback method that is used to process the events received by subscriptions. There is an Event class which has parameters including device type, device ID, event name and format. All the messages are returned as an instance of this class. **Blocking Queue** has been used to store events. A blocking queue is a queue that blocks a thread to dequeue from it if it is empty and block a thread to enqueue into it if it is full. One thread puts resources into it and other takes from it as shown in figure 6.3. It stores elements in First In, First Out (FIFO).

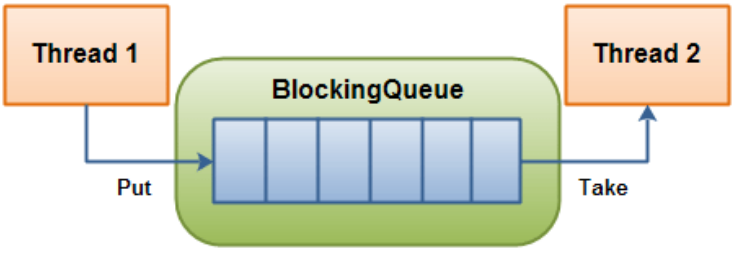

FIGURE 6.3: Blocking Queue

This whole process of receiving events is run in a separate thread. A brief review of threading in android will be given in Section 6.3.
All the above tasks including connecting to Watson Iot platform, subscribing and handling of events run on AsyncTask which will be explained alongwith threading in Section 6.3.

6.2.4 Publishing Commands to devices

Application can publish commands to all connected devices. A function **publishCommad()** is used to do so. The data is published in JSON format. The application publishes following commands to Raspberry Pi:

- **toggle:** This command is meant to toggle the status of specific appliance of specific switch board. Keys of corresponding JSON object includes
  - name of the board
  - mac_address address of Xbee connected to the smart switch board
  - the message i.e. a keyword which corresponds to specific appliance. For example b10 is to switch off light and b11 is to switch on light. For speed control devices like fan, a numerical value corresponding to the speed is concatenated with the message.
- **refresh:** This command is meant to query Raspberry Pi about the status of appliances. The Raspberry Pi responds by publishing an event to IBM watson Iot platform with data including status of all lights and fans and numerical values corresponding to the speeds of fans.

6.3 Threading in android

In android, there is main thread which is created when an application is launched. The purpose of main thread is to dispatch events to user interface widgets. That is why, sometimes it is called UI thread. But it is possible that main thread might not be the UI thread
When an app has to perform intensive work like network operations and database queries, one should avoid using main thread for such operations. These operations takes long time, so calling them in main thread would result in poor performance of the application. UI will be unresponsive until the operation is completed[14]. There are two rules for threading:

- Do not block UI thread
- Do not access the Android UI components from outside the UI thread

Following are ways to do long running tasks in another thread.

6.3.1 Thread and Handler class

One method of creating a thread is through thread class. Thread class implements Runnable interface which has method run(). Anything that needs to be done in a thread is placed inside run() method of runnable instance. Figure 6.4 shows how thread works. Some methods of thread class includes

- **start()**: starts the thread
- **sleep(long millis)**: sleeps the thread for given period of time. Time is given in milliseconds.

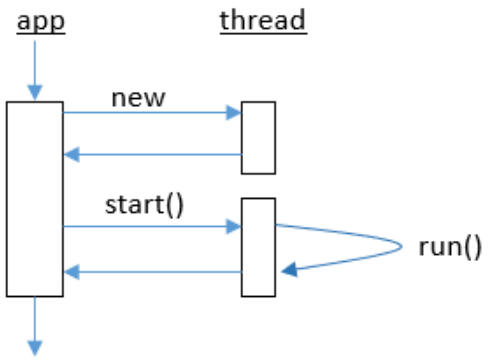

FIGURE 6.4: working of Thread

Handler belongs to a thread. One Thread can send work to another Thread by sending Messages or posting Runnables to a Handler that is associated with the other thread. An android thread is associated with message queue and a looper.

- Runnables are added to MessageQueue by calliing post() method of handler and looper calls run() method in MessageQueue's thread to dispatch them. Figure 6.5 shows how runnables are added to MessageQueue.
- Messages are added to MessageQueue by calling handler's sendMessage() method and looper calls handleMessage() method to dispatch them. Figure 6.6 shows how to dispatch runnables from MessageQueue.

In the Home Automation application, thread handlers are implemented in main thread. post() method of Handler class has been used to make changes in main thread from another thread.

6.3.2 AsyncTask

AsyncTask is used for short time operations and allows to perform operations in background and post them to UI thread without using threads and handlers. An AsyncTask is defined by three generic types including Params, Progress and Result. There are four steps in asynchronous task[15] which includes:

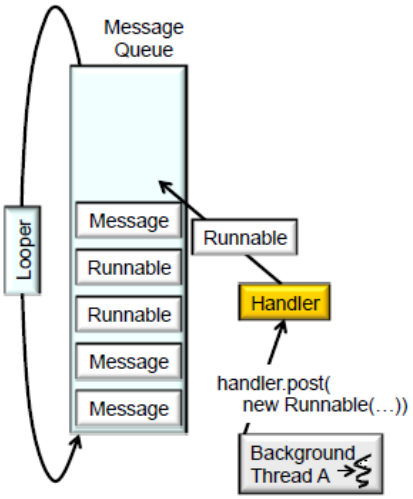

FIGURE 6.5: Adding Runnables

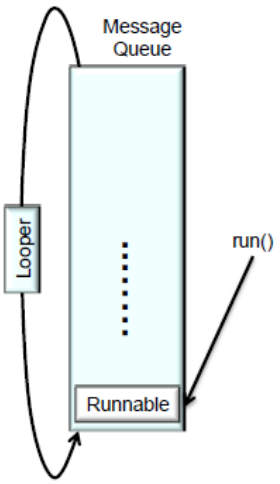

FIGURE 6.6: Dispatching Runnables

- **onPreExecute()**
  It runs in UI thread before doInBackground method.

- **doInBackGround(Params...params)**
  The return type of this method is Result. It performs operations in background thread

- **onProgressUpdate(Progress...values)** This method is invoked in UI thread in response to publishProgressProgress...() method. As depicted from name of the method, it is used to publish progress on UI thread while background operation is being executed.

- **onPostExecute(Result result)** This method also runs in UI thread after doIn-Background method.

An instance of task is created and then execute() method of AsyncTask is called. This is done on UI thread.

6.4 Retaining data in application

HomeAutomation Application needs to retain data. It has to save how many smart switch boards are there in the network and what are the status of applinaces connected to each board.

6.4.1 Data Storage Options in android

Android provides four ways to store data:

- **SharedPreferences**
  This method is used to save primitive data in key-value pairs. key is always

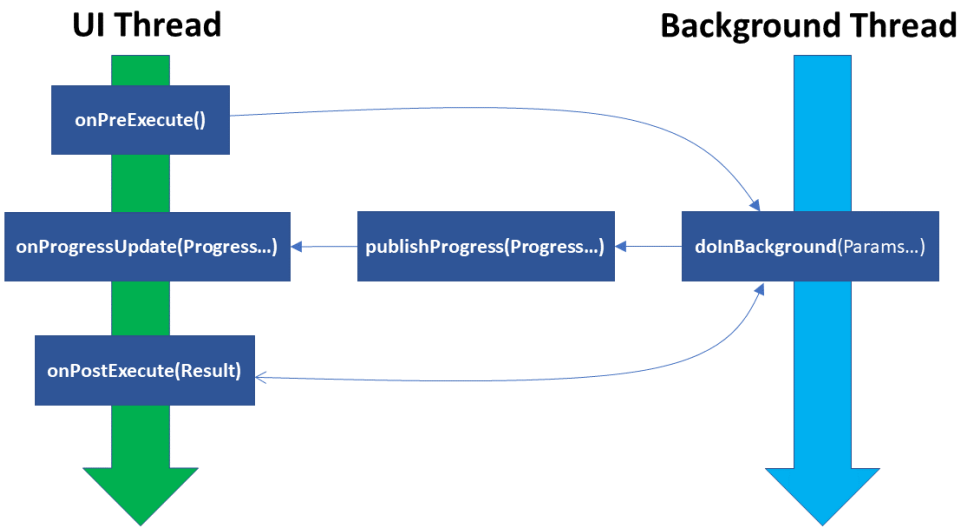

FIGURE 6.7: AsyncTask Flow

a string and corresponding values can be either integer, float, boolean or string. SharedPreferneces are stored in xml file in private directory. There can be multiple SharedPreferences files for an application.

- **Internal Storage**
  Files stored by this method are private to the app. They can't be accessed by other apps. When app is unisntalled, files get removed too. Cache files are also written on internal storage space.

- **External Storage**
  This method is used when user wants data files to be accessible by other apps. In order to write to external storage, WRITE_EXTERNAL_STORAGE, permission must be requested in manifest file.

- **SQLite database**
  Android provides support to store data using SQLite database. It is used to store structured and repeating data. In HomeAutomation application, this method has been used. As data is organized in database, it is easy to manage data.

6.4.2 Save data in android using SQLite:

The follwing steps have been follow to save data in android using SQLite[16]:

- A **Contract class** has been created which contains names of URIs, tables and columns.It allows to use same constants across all classes

- A helper class has been created that extends SQLiteOpenHelper class. SQLiteOpenHelper class provides two methods i.e. onCreate(SQLiteDatabase db) and

onUpgrade. It is responible for opening and creating databases. onCreate method is called once in application lifecycle and onUpgrade method is called when there is any update in existing version. CREATE command of SQL is passed as input to the excecSQL() method of SQLiteDatabase class to create table.

- In order to avoid calling database helper class direclty to access data and to avoid invalid data into database, content providers have been used in HomeAutomation application. Content providers[17] manages access to structured set of data. They provide a good abstraction layer between data source and UI code. It can share data with other apps and works well with other android framework classes. They provide data to application through ContentResolver interface.

- There are four actions to perform including:

  1. Query
  2. Insert
  3. Update
  4. Delete

  And the data on which action is being performed may be the whole table or single row of table. Content URIs are used to signal what data to act on. Structure of Content URI using is shown in figure 6.8. Contents URIs in HomeAutomation application are:

  – All rows in boards table
  content://com.example.android.boards/boards

  – Single row in boards table based on id
  content://com.example.android.boards/boards/#

FIGURE 6.8: Structure of Content URI

- In HomeAutomtion Application, BoardActivity (an activity class) calls on content resolver using query, insert, update or delete method. Content URI is passed to method in order to help content resolver know which provider to resolve to. When content resolver has gotten the information from the method, it used identical method of its own on content provider. Next, the method translated parameters that is passed into SQL.
  For qurey method, return type is a cursor which contains rows of interest. This cursor goes all way back to activity which called for it. In case of insert method return type is content URI telling where new item has been inserted. For update and delete method, return type is number of rows updated and deleted respectively.

- Next step is loading data using CursorLoaders. Create a class that extends CursorAdapter class. Cursor Loader is a loader that queries content resolver with specific URI and returns a cursor. It automatically loads data on background thread whenever there are any data changes. It works well with ContentProvier because a loader is tied to a URI. In order to use CursorLoader, LoaderCallbacks interface is used.

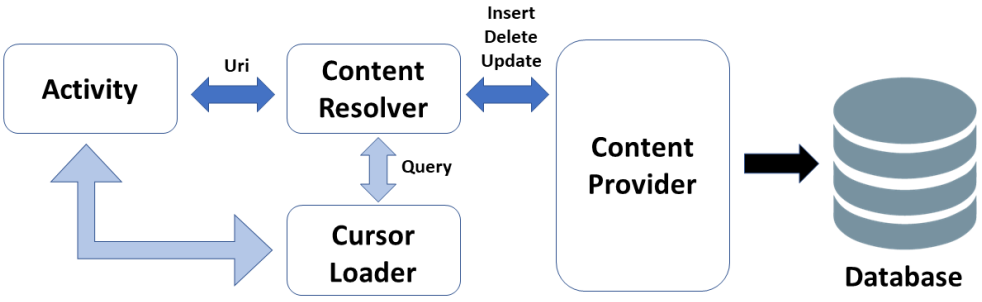

FIGURE 6.9: Flow diagram for query, insert, update and delete from SQLite database

6.5 Home Automation Application

In Home Automation application, basically there are two activities i.e. Home Automationmain activity and Smart board activity . First activity i.e main activity allows you to choose among boards. There is a menu on Main activity which has options including

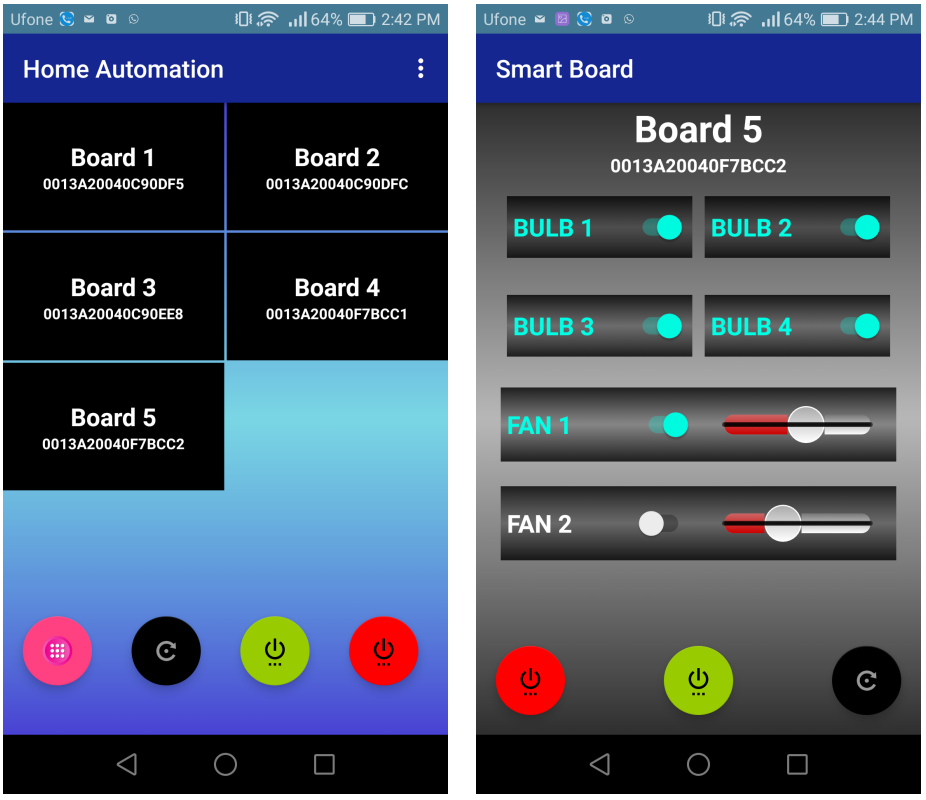

FIGURE 6.10: Home Automation Application

scan and help. Scan allows user to detect all smart switch boards in the network. When user clicks on any of the boards, Smart board activity opens which displays the current state of each appliance of the board.

Smart board activity allows user to control appliances connected to that boad. There are buttons to control the status of appliances and seekbar to control the speed of fans. In both activities, there are three buttons at the bottom. These button are displayed on clicking pink colored floating button.

- Red button is used to off all appliances
- Green button is used to on all appliances
- black button is to refresh app

Chapter 7

Raspberry Pi Control

The model used in the project for the interfacing and communication purposes is Raspberry pi 2.It is a B model which has a clock frequency of 900 Mega hertz and supports ARMv7 instruction set. The four USB ports of micro controller supports different peripheral devices like keyboard,mouse and USB device.It has an Ethernet port for developing the the internet connection either directly with the router or by developing peer to peer communication with other nodes.It has 17 general purpose input output pins which includes one hardware serial.The power rating of the pi B model is 800 mA or 4.0 W.It also includes a 15 pin connector for interfacing the camera module.

FIGURE 7.1: Raspberry pi model 2

7.1 Setting up Raspberry pi

The following steps are followed to setup the micro controller so that it can be used for the interfacing and communication purposes[18].

- The first step in setting up the micro controller is to download the operating system compatible for the pi 2 model.The operating systems compatible for raspberry pi is Raspbian[19]. Also available at the download section of the raspberrypi.org is the NOOBS which is a Raspbian installer.

NOOBS is the recommended one for the beginners.NOOBS further contains two operating system installers NOOBS and NOOBS Lite.The previous one includes the Raspbian preloaded.

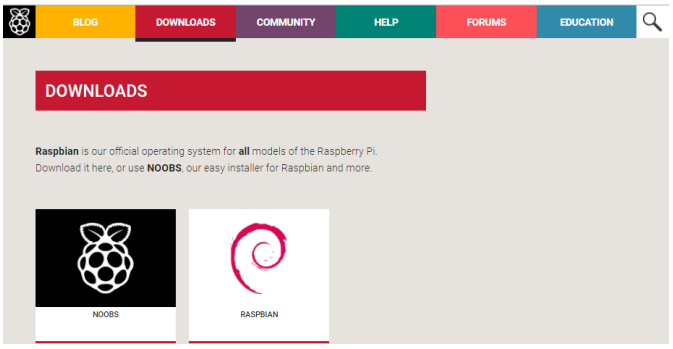

FIGURE 7.2: Raspbian operating system

- The second step is to format the SD card using the SD memory card formatter.After formatting the micro SD card extract the files in the NOOBS folder and place them in the SD card.Carefully remove the SD card from the computer device.
- The third step is to boot the raspberry pi.Connect the keyboard and mouse with the micro controller using the USB ports.Connect the adapter to the charging pin of the micro controller.Using vga to hdmi converter connect the LCD display with the controller.

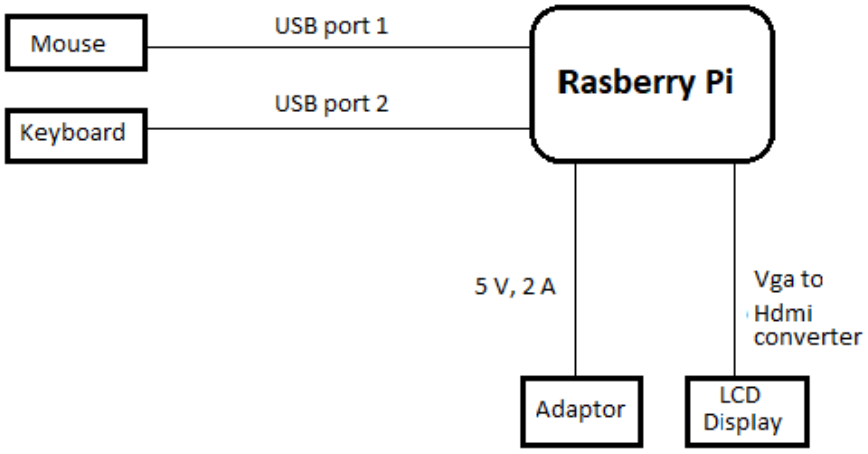

FIGURE 7.3: Connection diagram

- The final step is to select the operating system.Select the Raspbian and click on install button.It will run through the installation process.After the installation completes log in to your pi to access the graphical user interface.

7.2 Serial Communication with Xbee module

The hardware serial of raspberry pi is available on gpio 14 and 15 which corresponds to TX and RX respectively.The serial port of raspberry pi is initially used for tty console.To disable the console a change is made in the root directory of the raspberry pi. Delete the command of **console=ttyAMA0,115200** in the **cmdline.txt**.Exit the terminal and reboot the raspberry pi to save the changes.

After making the pins 14 and 15 available for serial communication by following the steps mentioned above connect the TX and RX of the Xbee module with the RX and TX of the raspberry pi for communication.Connect the 5 Volts and ground pin of the pi with the pin number 1 and 2 of Xbee respectively.The 5 volts are available on the pin number 2 and 4 whereas ground connection is available on the pins 6, 9, 14, 20 and 25

The pin configuration is shown below

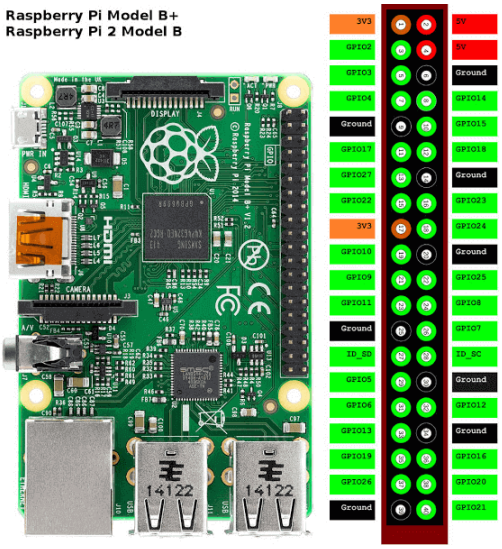

FIGURE 7.4: Pin configuration of Raspberry pi

The connection diagram of raspberry pi with Xbee module is given as

The pin number 8 and 10 corresponds to GPIO 14 and 15 respectively.

7.2.1 Sending Data to Xbee module

An object of the class serial is created to open the port having different parameters.The baudrate of 9600 is used for serial communication.Along with baudrate one stop bit,byte size of 8 bits and a timeout of approximately 2 seconds is used.Timeout defines the delay between the two successive openings of the serial port for sending and receiving the data packets.When the timeout is set to zero the port is continuously sharing data with the Xbee module without any intentional delay. The Transmit Request frame is used to send data to the Xbee module.The frame structure is defined as

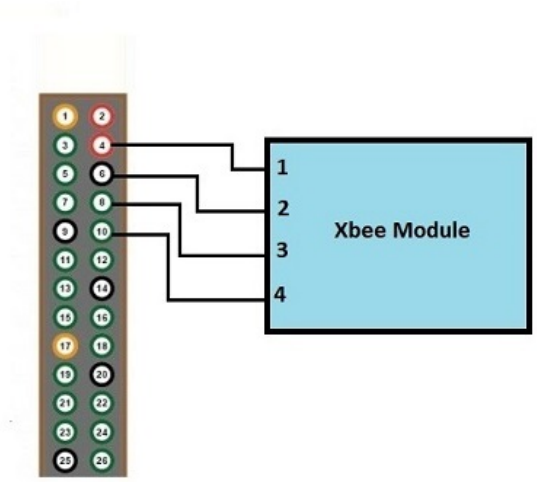

FIGURE 7.5: Xbee connection diagram

To send the data to Xbee module two input parameters are defined,the 64 bit destination address of the device and the message which corresponds to the RF data field.Whenever raspberry pi receives these two parameters from the user it generates a transmit request api frame and sends it to Xbee module connected to it through serial.This Xbee module is in a mesh network of the remaining nodes and sends the data to the destination node which is dictated by the 64 bit address as defined by the user.

7.2.2 Receiving Data from Xbee module

Whenever raspberry pi receives data from the Xbee module connected to it,it is in the form of receive packet API frame which has a frame type of 0x90.The message part of the frame is extracted which is in the form of the hexadecimal values and it is decoded to get the message in the ASCII format to send it back to the user.The communication diagram is shown below.

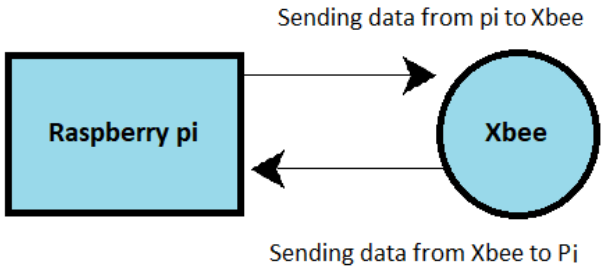

FIGURE 7.6: Communication diagram

7.3 Connection with IBM Watson IoT Platform

Client libraries have been used for developing the connection of raspberry pi with IBM.The first step in setting up the connection is to configure raspberry pi as a device having specific credentials. Credentials of device have been described in section 5.5.

An instance of Client is created using a constructor.The python dictionary named as **options** is created and the above credentials are passed to it.The client instance accepts this dictionary and makes a connection with the IBM cloud using try and except method.Error handling is done in python using this method.The client instance connects with the cloud and if any error occurs while making a connection an exception is raised which indicates the connection problem.

7.3.1 Event Publishing

Event publishing is used to send data or publish events to IBM cloud through devices.QoS 1 have been used to publish events to IBM platform. The events are published in the form of **json objects**. In order to publish the event,a json object is created which includes the data to be published and the method of **publishEvent** of the class Client is used to publish the event.

7.3.2 Command Handling

The device publish events to IBM cloud and handles the commands from the clients.When the connection is made between the client and the device the device automatically responds to any command which is meant for him. In order to handle the commands a **command callback method** is used. The data is received in the form of the instances of the Command class which has different attributes which specifies the command. The following are the properties of the instance of the command class.

- **command** which tells the type of the command to be processed.
- **format** for example json which indicates the format of the command.
- **data** which corresponds to the message part of the command.

The following figure shows the connection between a device and IBM cloud.

7.4 Interface with DHT22 Sensor

For measuring the room temperature and humidity dht22 sensor is interfaced with the raspberry pi.This sensor has four pins,two pins for powering up the sensor,one pin is the data pin while one pin has no connection.The connection diagram is shown below.

The Adafruit Sensor Library is used to interface the sensor with the pi. The temperature and humidity are measured and are placed in the form of a tuple.By using the event publishing method as described in the previous section the temperature and humidity

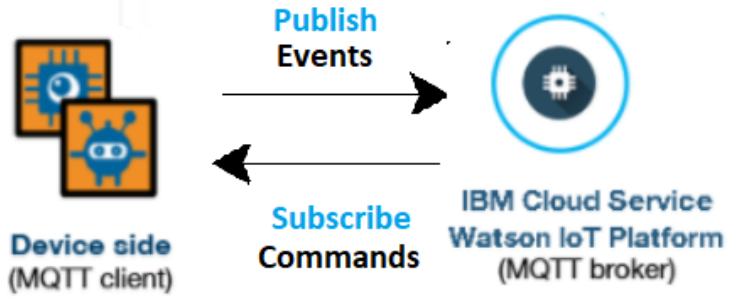

FIGURE 7.7: Connection with IBM IOT platform

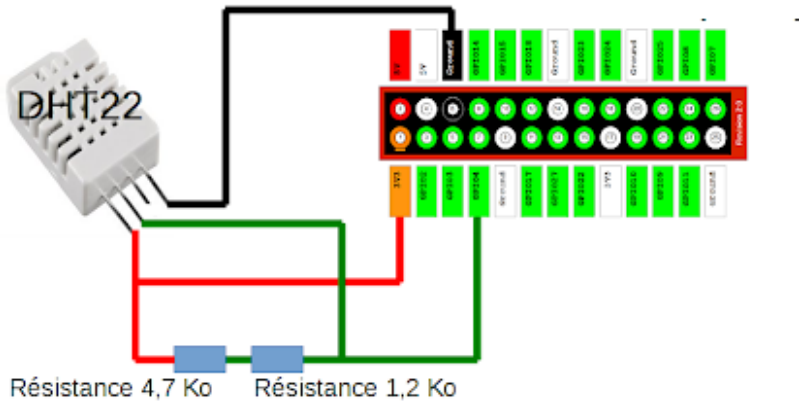

FIGURE 7.8: dht22 sensor connections

values are passed on to IBM cloud from where user can access these parameters.Based on these values of temperature and humidity a secure decision can be taken to switch on or off the fans without sending any command to the gateway.

7.5 App Development

In addition to the mobile application a desktop or touch screen application is developed using the python Kivy module. The application has three graphical user interface screens. The first screen is the main screen and it displays the temperature values and the message received from the gateway. The layout for the main screen is given as

To select the switch board a second screen is developed which shows the switch board numbers. There are total of five switch boards in a network so board number 1 to board number 2 are shown on the second screen using the default button widgets of the kivy module

The layout for the second screen is shown as

After the switch board is selected six buttons are shown to control the devices.Four buttons are to switch on and off the light bulbs whereas the remaining two are for

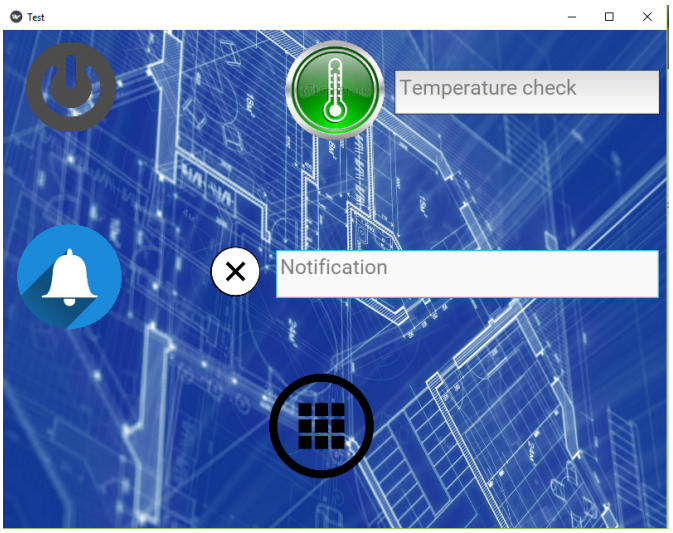

FIGURE 7.9: Main screen of the application

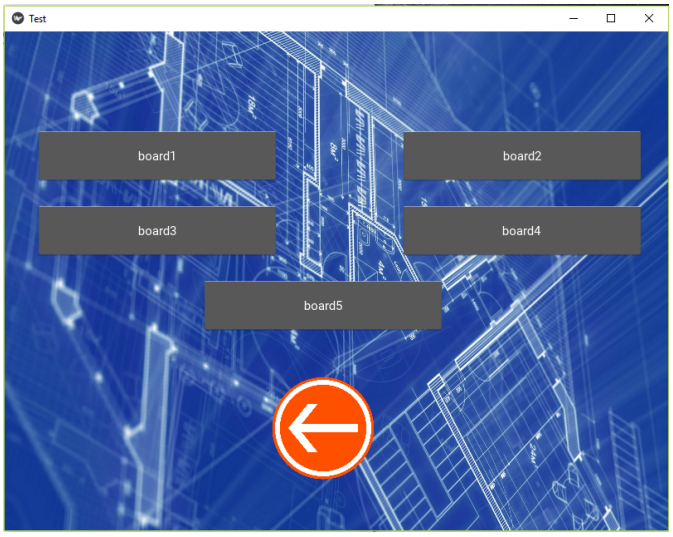

FIGURE 7.10: Second screen of the application

controlling the fans.The layout for the third screen is shown as

7.5.1 Functions used

The application is developed using the following functions of the Kivy module

- To place the widgets at the various locations the **Float Layout** is used.This layout has an advantage that a widget can be placed at any coordinate values of x and y.Two functions size hint and position hint defines the size and position of the widget with respect to the size of the root window so that if the size of the root window changes the widget adjusts itself according to this new window size.
- The **ButtonBehavior** class is used to give the properties of the default kivy module button widget to the random image files.

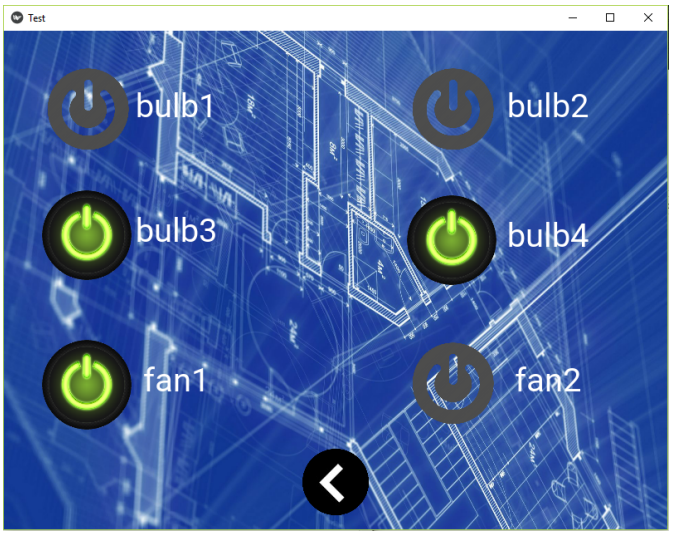

FIGURE 7.11: Third screen of the application

- **Bind** function is used to associate an event with the button widget.It further has two properties on press or on release which defines the functions to be executed when a particular button is pressed or released

- To display an image or the background wallpaper of the screen the **Image** widget is used.Its input parameter is the image file directory.To show the toggling of the button the image file directory is changed on each button press which changes the current file directory with the new one and as a result the corresponding image file at the new directory changes the previous one at the same x and y coordinates of the float layout.The image files should be placed in the same folder as the python script in order to access them

- To make a transition between the different screens the **Screen manager** widget is used.Three screens having the titles ScreenOne, ScreenTwo and ScreenThree are defined.Each screen has a layout and a button widget at the bottom to shift between the screens.Whenever this button is pressed a function is called which changes the title of the current screen and makes a transition from one screen to the other as defined by the screen title.

- The **Label** widget is used to enter the text at the specific locations on the layout.The unicode strings and the ASCII are the supported formats.Its input parameter is the text which is a string of characters to be displayed.

- The **TextInput** widget is used to display the temperature values and the notifications.A variable holding the temperature value is used and the instance of the TextInput widget class shows this variable value.

Chapter 8

Conclusion and Recommendations

In the nutshell, the developed system is secure, flexible and reliable. It can be extended to number of buildings due to its flexibility. The project is fascinating and attractive. The features of the project include appliance control, thermostat control, energy management and backup protection. The project has been implemented Control Lab of Electrical Engineering Department in University of Engineering and Technology, Lahore.The system can be used in industries, homes, hospitals and parkings since only gateways require internet access. The project focuses on middle class people as well as industrial sectors. Disabled people will also find it helpful. The project will have huge demand in market in future.
There is major development coming in this field of internet of things, so following are the recommendations and future aspects in the perspective of this device, either they are related to medical, security, transport, automation, socializing or agriculture.

- Esp8266 wifi module can be used instead of Xbee S1 which is much cheaper and will decrease the cost of smart switch board to such extent that people of lower middle class can afford it. Only disadvantage of using esp8266 is its lesser flexibilty to expand as compared to DigiMesh. Although it is cheaper and very suitable for a place where limited number of devices have to be controlled.
- Voice recognition can also be added to system.
- Smart wearable devices are the most developing thing in this field. These are installed with software and collect users data and then processing it. And these gives the results of users daily routine by monitoring it.

- Industrial automation is the most developing aspect. It enable the user to keep track of goods, communication among the suppliers and retailers, automated delivery quality control and sustainability and due to this increased efficiency and profits.
- The most curious aspect of this field is the smart city. This includes smart surveillance, transportation, smart energy management system, environmental monitoring, urban security.
- There is a great advantage for the farmers because they can take best yields based upon the data from sensors which is about soil moisture, nutrients in the soil, controlled water usage and prediction of the fertilizer to be used based upon this data.
- Smart grid concept is gaining the attraction of power generation field. The concept is to collect data and analyzing, it will be able to detect the power outages more quickly, improving efficiency and economics, and more efficient energy distributed system.

References

[1] D. Hendricks, "The history of smart homes," *IoT Evolution World*, 2014.

[2] Y. Zhang, J. Luo, and H. Hu, *Wireless mesh networking: architectures, protocols and standards*. CRC Press, 2006.

[3] J.-S. Lee, Y.-W. Su, and C.-C. Shen, "A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi," in *Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE*. Ieee, 2007, pp. 46–51.

[4] C. Gomez, J. Oller, and J. Paradells, "Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology," *Sensors*, vol. 12, pp. 11 734–11 753, 2012.

[5] L. Li, H. Xiaoguang, C. Ke, and H. Ketai, "The applications of wifi-based wireless sensor network in internet of things and smart grid," in *Industrial Electronics and Applications (ICIEA), 2011 6th IEEE Conference on*. IEEE, 2011, pp. 789–793.

[6] Z. Alliance, "Zigbee home automation public application profile," *IEEE J. Select. Areas Commun*, 2007.

[7] P. M. XBee and X.-P. DigiMesh, "2.4 oem rf modules, digi international," 2010.

[8] H. E. Güner and M. Ambarkütük, "Building wireless mesh sensor networks using xbee and arduino," 2013.

[9] J. P. Kirby, P. D. Dudley, and D. A. Schwendner, "Digital electronic dimmer," May 2 1978, uS Patent 4,087,702.

[10] IBM. Internet of things platform. [Online]. Available: https://console.bluemix.net/docs/services/IoT

[11] K. Kobylinski, J. Bennett, N. Seto, G. Lo, and F. Tucci, "Enterprise application development in the cloud with ibm bluemix," in *Proceedings of 24th Annual International Conference on Computer Science and Software Engineering*. IBM Corp., 2014, pp. 276–279.

[12] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, "Mqtt publish/subscribe protocol for wireless sensor networks," in *Communication systems software and middleware and workshops, 2008. comsware 2008. 3rd international conference on.* IEEE, 2008, pp. 791–798.

[13] A. Studio. (2016) the official ide for android. [Online]. Available: https://developer.android.com/studio/index.html

[14] V. Matos and R. Grasser, "An experience on multithreading using android's handler class," *Journal of Computing Sciences in Colleges*, vol. 30, pp. 80–86, 2014.

[15] A. Goransson, *Efficient android threading: Asynchronous processing techniques for android applications.* " O'Reilly Media, Inc.", 2014.

[16] L. Vogel, "Android sqlite database and contentprovider-tutorial," *Java, Eclipse, Android and Web programming tutorials*, vol. 8, 2010.

[17] A. Developers, "Content providers," *Dostupno*, vol. 12, pp. 10–45, 2014.

[18] M. Richardson and S. Wallace, *Getting started with raspberry Pi.* " O'Reilly Media, Inc.", 2012.

[19] W. Harrington, *Learning Raspbian.* Packt Publishing Ltd, 2015.