RAFIQ HAMAD AMIN MAULUD
DEVELOPMENT OF A WIRELESS SMART HOME SYSTEM BASED ON INTERNET OF THINGS
NEU 2015

DEVELOPMENT OF A WIRELESS SMART HOME SYSTEM BASED ON INTERNET OF THINGS

A THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
RAFIQ HAMAD AMIN MAULUD

In Partial Fulfillment of the Requirements for the Degree of Master of Science in Software Engineering

NICOSIA, 2015

DEVELOPMENT OF A WIRELESS SMART HOME SYSTEM BASED ON INTERNET OF THINGS

A THESIS SUBMITTED TO THE GRADUATE SCHOOL OF APPLIED SCIENCES OF NEAR EAST UNIVERSITY

By
RAFIQ HAMAD AMIN MAULUD

In Partial Fulfillment of the Requirements for the Degree of Master of Science in Software Engineering

NICOSIA, 2015

Rafiq Hamad Amin MAULUD: DEVELOPMENT OF A WIRELESS SMART HOME SYSTEM BASED ON INTERNET OF THINGS

Approval of Director of Graduate School of Applied Sciences

Prof.Dr.İlkay SALİHOĞLU

We certify this thesis is satisfactory for the award of the degree of Masters of Science in Software Engineering

Examining Committee in Charge:

Prof.Dr. Rahib Abiyev	Committee Chairman, Computer Engineering Department, NEU
Assist.Prof.Dr. Boran Şekeroğlu	Committee Member, Software Engineering Department, NEU
Assist.Prof.Dr. Umit Ilhan	Committee Member, Computer Engineering Department, NEU
Assist.Prof.Dr. Elbrus Imanov	Committee Member, Computer Engineering Department, NEU
Prof.Dr. Doğan İbrahim	Supervisor, Committee Member, Computer Information Systems Department, NEU

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Rafiq Hamad Amin Maulud
Signature:
Date:

ACKNOWLEDGEMENTS

This thesis would not have been possible without the help, support and patience of my principal supervisor, my deepest gratitude goes to Prof.Dr.Doğan İbrahim, for his constant encouragement and guidance. He has walked me through all the stages of the writing of my thesis. Without his consistent and illuminating instruction, this thesis could not have reached its present from.

I would like to thank Assist.Prof.Dr. Boran Şekeroğlu who has been very helpful through the duration of my thesis.

Above all, my unlimited thanks and heartfelt love would be dedicated to my dearest family for their loyalty and their great confidence in me. I'm greatly indebted to my parents for giving me a support; encouragement and constant love have sustained me throughout my life.

Eventually, there is a long list of my friends that I would like to thank. I can't mention them all but I would like to thank them from all of my heart for their valuable help and support since I was in my early study until now.

To my parents.

ABSTRACT

This thesis presents the development of a low cost embedded wireless smart home prototype system for controlling home appliances in a building. The system provides a friendly user interface android application based smartphone, for controlling appliances and devices from anywhere in the world with an IP connectivity for accessing the embedded micro web-server. This system uses an Arduino Ethernet interface which runs a simple data web server to avoid using a personal computer as a server to keep the overall system cost to minimum. For the communication between the micro web-server node and the end nodes each has an XBee module based on the IEEE 802.15.4/ZigBee standards which offers a novel communication protocol to control the end devices remotely.

The study showed that the ZigBee devices can be considered as the most suitable wireless network technology to meet the requirements of global marketplace in developing remote control systems.

What makes ZigBee stand apart from the rest is the easy to use, extremely low power consumption, cost effective, reliability, low latency, supporting a huge number of connected nodes into a single control network.

Keywords: Android smartphone; Arduino; home automation system; internet of things; remote control; smart home system; XBee; ZigBee

ÖZET

Bu tez, bir binadaki ev aletlerini kontrol eden düşük maliyetli entegre kablosuz akıllı ev prototip sisteminin geliştirilmesini sunar. Sistem sunduğu kullanıcı dostu android bazlı arayüzü içeren bir akıllı telefon sayesinde IP bağlantısı üzerinden dünyanın neresinde olursanız olun entegre mikro web sunucusuna erişerek cihazları kontrol edebilmenizi sağlar. Bir PC'nin sunucu olarak kullanımından kaçınarak tüm maliyetleri en düşük düzeye indirmek için sistemimiz, basit bir veri ağ sunucusu çalıştıran Arduino Ethernet arabirimini kullanır. Mikro web sunucu nodu ve uç nodları arasındaki iletişimi kurmak için, her uç nodunda IEEE 802.15.4/ZigBee bazlı birer XBee modülü bulunmaktadır. Bu sayede uç noktalardaki cihazları uzaktan kontrol etmeyi sağlayan yeni bir iletişim protokolü standardından faydalanılmaktadır.

Bu çalışma ZigBee cihazlarının, uzaktan kontrol sistemlerinin geliştirilmesinde küresel pazarın ihtiyaçlarını karşılamak için en uygun kablosuz ağ teknolojisi olarak kabul edilebileceğini göstermiştir.

ZigBee standardını diğerlerinden ayıran özellikleri kolay kullanımı, son derece düşük güç tüketimi, düşük maliyeti, güvenilirliği, düşük latency süresi, ve tek bir kontrol ağı içine bağlı çok sayıda nod desteğinin olmasıdır.

Anahtar Kelimeler: Android akıllı telefon; arduino; ev otomasyon sistemi; birşeylerin İnterneti; uzaktan kumanda; akıllı ev sistemi; XBee; ZigBee

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACK:	Acknowledgement
ADC:	Analog to Digital Converter
API:	Application Programming Interface
ARP:	Address Resolution Protocol
ASIC:	Application Specific Integrated Circuit
BJT:	Bipolar Junction Transistor
CH:	Channel
CTS:	Clear to Send
DC:	Direct Current
DH & DL:	Destination High & Destination Low
EEPROM:	lectrically Erasable Programmable Read-Only Memory
EMF:	ElectroMotive Force
GND:	Ground
GSM:	Global System for Mobile (communications)
GUI:	Graphical User Interface
HTTP:	HyperText Transfer Protocol
I2C:	Inter-Integrated Circuit
IC:	Integrated Circuit
ICMP:	Internet Control Message Protocol
ICSP:	In-Circuit Serial Programming
IEEE:	Institute of Electrical and Electronics Engineers
IGMP:	Internet Group Management Protocol
iOS:	iPhone OS
IoT:	Internet of Things
LAN:	Local Area Network
LED:	Light-Emitting Diode
M2M:	Machine-to-Machine
MCU:	Micro Controller Unit
MOSFET:	Metal Oxide Semiconductor Field Effect Transistor
MQTT:	MQ Telemetry Transport

OS:	Operating System
PAN ID:	Personal Area Network Identifier
PC:	Personal Computer
PCB:	Printed Circuit Board
PNP:	Positive-Negative-Positive
PPPoE:	Point-to-Point Protocol over Ethernet
PWM:	Pulse Width Modulation
REST:	Representational State Transfer
RFID:	Radio Frequency IDentification
RTS:	Request to send
SPDT:	Single Pole Double Throw
SPI:	Serial Programming Interface
SRAM:	Static Random Access Memory
TCP/ IP:	Transmission Control Protocol/Internet Protocol
TWI:	Two Wire Interface
UART:	Universal Asynchronous Receiver/Transmitter
UDP:	User Datagram Protocol
USB:	Universal Serial Bus
UWB:	Ultra-Wide Bandwidth
ZB:	ZigBee

CHAPTER 1
INTRODUCTION

The Internet of things (IoTs) is a term used to describe different things (appliances, devices, smartphones, even humans and animals) connected with each other and with the internet. This enables communication between all the devices and provides the ability to monitor and control the devices remotely. Typically, all the devices are assigned with unique identifiers to provide them unique identity in the network. IoTs are considered as a new dimension to the world of IT and communication, because it has a huge impact on industry in the last few years. The dynamic network of IoTs is really advanced and it will continue to expand until it reaches the goal of AAA connectivity (Anyone, Anytime, Anywhere). Cisco predicts that 50 billion connected devices will be in use by 2020. (Cisco, 2013).The Internet of Things has the potential to revolutionize the sector of smart home systems, to provide intelligence, and a comfortable environment to improve the quality of live. In this thesis, a small network based on IoT theme is designed, which allows the user to control different devices by using a smartphone. The novelty of the system developed in this thesis is that the communication is based on using the ZigBee protocol, which it can be stacked up against other common wireless standards, because of having some commendable features including low power consumption, reliability, low latency, easy to implement, low cost, supporting a huge number of connected nodes into a single control network, and so on (ZigBee, 2015).

1.1 Thesis Problem

One of the major world problems today is the energy crisis. There is a continuous research going on to find new ways to harness energy and to reduce the energy consumption. Another major global issue is our changing environment, our environment's condition is deteriorating consistently and we need steps to monitor and improve its condition. And in nowadays era in many countries the number of elderly people increases, so such systems help them and provides them the ability in daily living basic activities performing and to be independent in their life.

Another issue that was tried to solve in this thesis is to decrease the cost of an IoT based system. Generally, such connected systems have high costs associated with them because of the specific and high-end hardware they use. These high costs restrain ordinary users to use such products. Therefore, we have worked on decreasing the cost of this system as much as possible to keep it in the range of public.

The internet of things is a possible solution to the above mentioned problems and many others. It allows us to monitor and control our devices wirelessly and remotely. When devices are connected to each other then they can communicate with each other.

1.2 The Aim of the Thesis

The aim of this thesis is to develop a cost effective IoT based wireless smart home system which allows the users to control their devices with the help of their smartphones. Other goals of this project include the development of a secured network of communication so that only authorized user can login the application and connect with the micro-web server to use the system.

Here is the list of functions the developed system will have:

- It will connect a light, a fan and a stepper motor with a central controller which will have the ability to communicate and control all of these nodes.
- The central controller will be connected to the smartphone application via internet. This application will allow control of the central controller and in-turn the control of all devices.
- The application can switch on/off the light.
- The direction of the fan can be controlled with the help of the application.
- The application will allow controlling the number of steps, number of revolutions, speed and direction of the stepper motor.
- The application will be password protected and can only be accessed by authorized users.

1.3 The Importance of the Thesis

The importance of this thesis finds itself in providing assistance and help to the disabled and ageing people at home and reducing the cost of establishing such systems, because this

system doesn't need to be installed during building the house. Rather than smart home systems the concepts we used in this thesis are applicable to most of the portable systems such as autonomous cars, robots, autonomous guns, quad copters within a specific range, and so on.

1.4 Limitations of the Study

The mobile application is developed for Android based smartphones and cannot be operated on iOS or other mobile operating systems.

The distance between end nodes and the main node cannot be further than 40 meters indoor and 120 meters outdoor.

1.5 Overview of the Thesis

This thesis consists of six chapters which are Introduction, Literature Review, Hardware Specification and Design, Software development, Results and Discussion, and Conclusions and Future work.

The first chapter is an introduction in which the chapter gives a brief description of the project such as the problem statement, the aim of the thesis, the importance of the thesis, and limitations of the study.

Chapter 2 covers the Literature Review, where a brief history of IoT and the applications in which the IoT plays a key role with shedding light on smart home systems is discussed, and also some previous related works are reviewed.

In Chapter 3 the hardware components used in the developed system are specified in details.

Chapters 4 discusses the Android application and hardware nodes' software developing, and the designing of the Xbee networking is shown as well.

Chapter 5 displays the result of this thesis for both Android app and the actual hardware components. Furthermore, this chapter also discusses the difficulties faced during developing this system and the results obtained.

Chapter 6 concludes the overall thesis and future work is recommended for further improvements.

CHAPTER 2
LITERATURE REVIEW

The Internet of Things is a concept of different devices connected to the internet. The basic idea of this concept is the pervasive presence of a variety of things or objects around us, such as Radio-Frequency Identification (RFID) tags, sensors, actuators, mobile phones and etc. which, through unique addressing schemes, are able to interact with each other and cooperate with their neighbors to reach common goals (Giusto et al., 2010).

IoT is not only transforming the industry but also the lives of consumers. It is used for providing better health services, energy services and enhancing the living standards. It plays a significant role in the automotive industry, logistics and etc.

This chapter shows a brief history of IoT, the fields which IoT plays a key role in, the role of IoT in Smart home and its benefits, and several projects and works in the area of IoT for home automation or smart home systems using different technologies.

2.1 History of IoT

The term of IoT or Internet of Things is widely used today to describe a lot of topics. However, this term was first coined by Keven Ashton in 1999. He used this term in a presentation to describe a future world where all devices will be connected to the internet. In 2009 RFID Journal, Ashton elaborated in his article:

> "If we had computers that knew everything there was to know about things — using data they gathered without any help from us — we would be able to track and count everything, and greatly reduce waste, loss, and cost. We would know when things needed replacing, repairing, or recalling, and whether they were fresh or past their best. The Internet of Things has the potential to change the world, just as the Internet did. Maybe even more so," (Ashton, 2009).

It seemed like a dream at that time. But Ashton has just given a different name to a technique already existed. The concept of an intelligent city, smarter devices and connected world previously existed since long ago.

In the same year, 1999, Andy Stanford-Clark and Arlen Nipper introduced MQ Telemetry Transport (MQTT). It was the first machine-to-machine protocol for connected devices. The basic idea behind it was to make a standard for inter-device communication so that devices can communicate with each other via using lightweight messages as shown in Figure 2.1, thus consuming low bandwidth (Gunner, 2013).

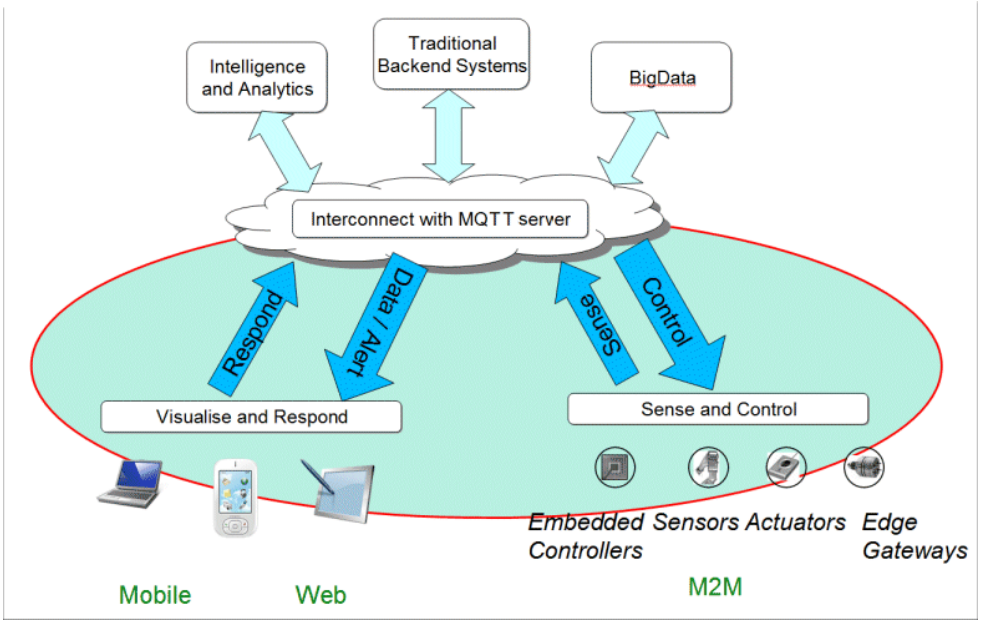


Figure 2.1: MQTT topology (Gunner, 2013)

Schoenberger (2002), published an article in Forbes entitled “The Internet of Things” he stated in his article "Stores have eyes. Now they’re getting brains. Soon tiny wireless chips stuck on shampoo bottles and jeans will track all that you wear and buy".

After that, the internet of things became a hot term and many people started development on it. This particular term was mentioned in different articles and publications in Scientific American, Boston Globe, and the Guardian. This indicates the significance of this concept.

In 2005, a team of students at the Interaction Design Institute Ivrea in Ivrea, Italy developed the Arduino (Gibb, 2010). It was a single board microcontroller based on Atmega8 (8 bit microcontroller by Atmel). It included all the basic circuitry on-board to use the microcontroller as a plug and play device. With Arduino in market, things were

changed a lot, and this small platform became the most frequently used device in the development of connected devices.

In 2011, IoT appeared in the Gartner Hype Cycle for the first time. This Hype Cycle graphic created by Gartner, shows their perspective on specific technologies and their progress from technology trigger to plateau of productivity (Gartner, 2011).

As shown in Figure 2.2 Internet of Things appeared in the phase of Technology trigger in which the IoT had gotten a breakthrough.

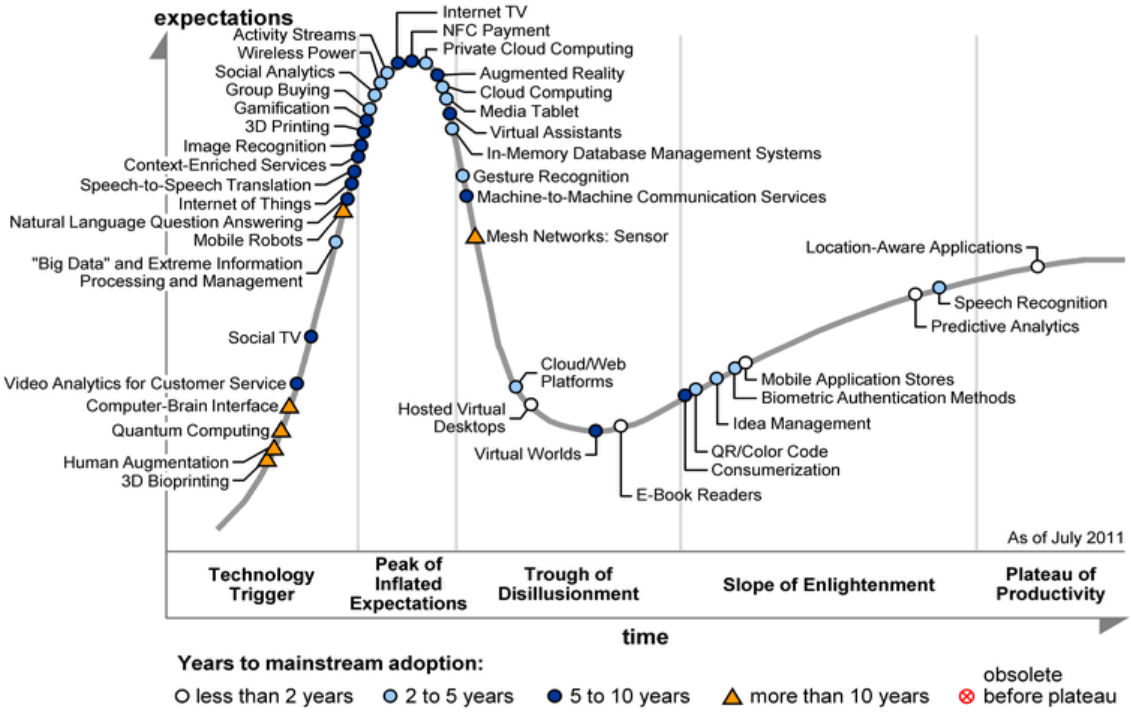


Figure 2.2: Gartner Hype Cycle (Gartner, 2011)

VB (Venture Beat) named 2014 as the year of Internet of Things (VB, 2013), because people started using connected devices and gadgets more commonly than ever before. Fitness gadgets, home security devices, digital thermostats, connected cars, self-driving vehicles and even connected robots became a quite much common stuff by this year. All of these devices are the outcome of the concept of IoT. Hilton (2012), expected that the

number of devices connected to the Internet will accumulate from 100.4 million in 2011 to 2.1 billion by the year 2021, growing at a rate of 36% per year.

2.2 Applications of IoT

IoT has a huge potential of development in almost every field. There are a large number of applications being developed on daily basis based on the IoT theme. The major application areas in which IoT plays its key role are as follows:

- Transportation and logistics
- Healthcare
- Smart environment
- Personal and social domain

Apart from the applications which have been developed and are currently under use, there are lots of applications which are mere ideas now and will be soon a reality. Figure 2.3 shows some of the applications that are developed under these major categories.

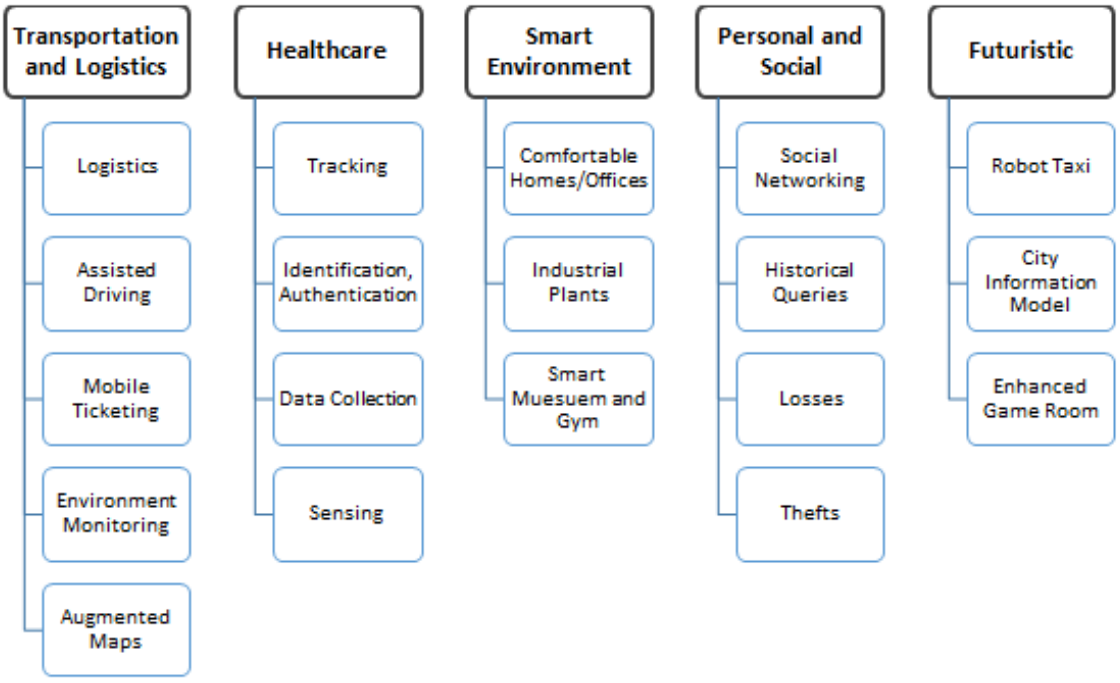


Figure 2.3: Application Domain and relevant major scenarios (Atzori et al., 2010)

We are actually using different devices based on the IoT concept daily. Devices like smartphones, smart watches, internet televisions, smart bands, heart rate and activity monitors. These are all internet-connected devices.

2.3 IoT in Home Automation Systems

The automated home, or more commonly called as smart home, is one of the many applications of the IoT Concept. Figure 2.4 gives a glimpse of a connected smart home.

Figure 2.4: Smart home, controlled from mobile application (auto webbed, 2015)

Our homes are automated since a long time, we turn on a switch at one place and the garage door opens automatically at the other place. This is automation, but it is wired, it requires us to be present at the place of action. However, IoT has revolutionized this concept, as it has introduced the concept of sensing and controlling the behavior of devices remotely. Not only this, the devices have become intelligent, these can tell us their state and can automatically take actions according to it. For example, a sensor present at your rooftop can sense whether it is day or night and can automatically switch on your garage light. It is completely independent, you don't have to supervise it or be present there for it to operate. This concept is even more enhanced by using a mobile application to control

and monitor all of your house appliances. You can have the temperate and humidity of your rooms at your mobile screen and you can have the status of your lights on your mobile. You can turn on/off your devices remotely from anywhere in the world because all of your devices are connected to the internet and you have the access to them all the time.

2.4 Benefits of Smart Homes

We have discussed the involvement of connected devices in our homes. But what are their benefits? Why do we need them and what we can achieve from them? These points are summarized in the following headings:

- **Control:** The most prominent advantage is control, remote control to be precise. You have the access to all of the devices of your house readily available. You can switch them on or off from anywhere and you can check their status from wherever you are. Let us suppose you are going to the airport and you just got a click in your mind if you have locked the main gate or not? All you need is to turn on your mobile application and it can tell you if your gate is locked or not, if it's not, then you can lock it with the touch of your finger.
- **Convenience:** With control comes convenience, a smart home will allow you to have your coffee ready when you arrive at your home from your office. It allows you to remotely grant access to certain people to your home at certain times. It gives you a peace of mind that your house is secured and under your control all the time
- **Saving**: Smart houses doesn't just allow the control of your devices but these are also capable of intelligently monitoring the power consumption of devices and switching them off when required to save the power
- **Security:** Smart houses have Wi-Fi enabled cameras and IP cameras installed which allows 24 hours monitoring of the houses. Along with these, smoke sensors, human presence sensors, laser sensors and automatic intruder alarms are installed which increase the security of the houses significantly.
- **Senior Independence:** This is another major advantage of smart houses that they allow the elderly or disabled people to control the devices easily. Apart from controlling the devices, there are also voice activated sensors and voice alarms to assist them in their daily routines.

2.5 Related Works

In the world of science the term of 'Smart Home' is not new, but it is still far more away from the vision of today's society. With the development of electronic devices, the field of home automation or smart home systems has been developing fast. Lately, a hype has been observed in its development and implementation in recent past years. The reason behind it, is the invention of Bluetooth, GSM modules, RFID, Smart phones and various other embedded wireless solutions.

One of the initial home automation solutions were Bluetooth based (Piyare and Tazil, 2011; Chiu-Chiao et al., 2011; Potts and Sukittanon, 2012; Ramlee et al., 2013; Yan and Shi, 2013). A central Bluetooth communication device (acting as a Bluetooth server) was provided and all of the devices had a Bluetooth Client in them. That server communicates and controls the devices with the help of Bluetooth, so the range was limited. Hence, this system was not truly "remote". Later on, Smart homes were introduced with Ethernet and WiFi. In those systems, the Bluetooth was replaced with one of the technologies mentioned earlier. These systems were truly based on the concept of IoT, as these were connected with the internet and had the ability to control the devices from everywhere.

Home automation based systems were also developed using GSM modules (Shahriyar et al., 2008). Such systems were controlled by sending AT commands to the GSM module which then controls the devices connected to it. Another aspect of Home Automation systems is the mode of communication between the devices and the central controller. Initially, Bluetooth was used for this purpose but with the advent of Zigbee and other similar wireless M2M solutions, it was replaced.

ElShafee and Hamed (2012) developed a smart home system using WiFi technology. The interfaced home appliances can be managed using a computer (with in-built WiFi card) based web server. The users have local access to the system through LAN or remote access via internet. The system provides the functionality for many home devices such as security and power management components.

The systems presented by (Alkar and Buhur,2005; Liang et al., 2002; Rajabzadeh et al., 2010; Sharma and Reddy, 2012) can be controlled via internet. In which a database, a webpage and a webserver dedicated to manage the home interconnected devices. Because

of using a computer in these systems, the power consumption and the cost increase significantly. In addition developing a database, designing a web page and hosting it will also result in higher costs.

Kamarudin et al., (2013) proposed a smart home system which can be activated by voice. An application is developed by using Microsoft Visual Basic which provides a GUI and hosted by a computer, and Microsoft Speech Recognition engine is used. The microcontroller receives the signal via radio frequency and sends it to all home appliances that are interfaced. It needs a computer again which causes more power consumption and increases the cost.

On the other hand, Shiu Kumar (2014) developed a system that can manage and monitor the smart home environment remotely, but this system is based on wires communication between the Arduino Ethernet server and the end devices. But with employing wiring communication, the system faces problems when it's not planned and installed during the building physical construction. This makes the system to require much effort and increases the cost as well.

All the systems described above have had a tremendous contribution to smart home systems. Nevertheless, using a PC as a server raises the prices and the power consumption, while other systems need web page hosting services and wire communication between Arduino mirco web-server and the end devices which increases extra cost. And for speech recognition the voice activation systems need either computer software or a separate module for voice recognition.

However, in the system designed, it was tried to overcome all the above described issues, by excluding PC as a server, web page hosting, communications based wiring and so on. The technique which we have used in this thesis is based on REST web service (Kamilaris, 2011) running on an Arduino server and controlled via a smart phone application designing for Android. The reason for using this combination is because of its simplicity, having low cost and high performance design. The REST API is a very light weight web service which can be used even on an Arduino server and thus decreasing the cost significantly.

CHAPTER 3
HARDWARE SPECIFICATION AND DESIGN

A wireless smart home project has been developed which is cost-effective, secure and simple to use. As an overview, this system allows the users to control their home appliances remotely from anywhere via internet. It consists of three main parts:

- Nodes with attached devices
- Central controller and server
- Android Application

These three parts are interlinked with each other. For demonstration, we have controlled three different types of devices in this thesis, which they are:

- An LED
- A Fan (attached with brushed DC motor)
- A small size stepper motor

These devices communicate wirelessly with the central controller. Xbee modules are used for wireless communication which implements the Zigbee protocol (Digi, 2015) more details on this module discussed in section 3.2.1. Each node has its own Xbee module present with it and there is one Xbee module attached to the central controller that acts as a coordinator to coordinate communication among all nodes.

Central controller is designed using Arduino, which is an open source AVR based development platform (Arduino, 2015). All of the devices can be controlled wirelessly using this central controller. For internet connectivity, an Ethernet shield is mounted on the Arduino that connects it to a router. The router acts as a gateway for communication within the local network as well as from the internet. An Ethernet server is established in the central Arduino microcontroller board using REST API.

An Android application is developed which enables the users to control their devices (home appliances) remotely from anywhere over the globe. The Android based app connects with the mini server running on the Arduino via REST API commands. The

server is made password protected so that only authorized persons can connect with it. This increases the security of the system.

Figure 3.1 shows the block diagram of the developed system. The block diagram is divided into two layers: Remote Environment and Home Environment.

Remote Environment represented by the right layer which authorized users can connect to the developed system on their Android Smart phone application using the internet, any 3G/4G network or WiFi connection can be utilized on the user devise.

The left layer shows the Home Environment which consists of a router to provide the internet or local area network connectivity, a tiny web-server based on Arduino Ethernet which it is the main component of this layer, the main task of this tiny server is to execute necessary actions that must be performed to control the end devices. The other components of this layer are the end nodes; each has an Xbee module with its connected device. Those modules communicate with the micro web-server over the air and provide wireless connectivity to electronic end devices.

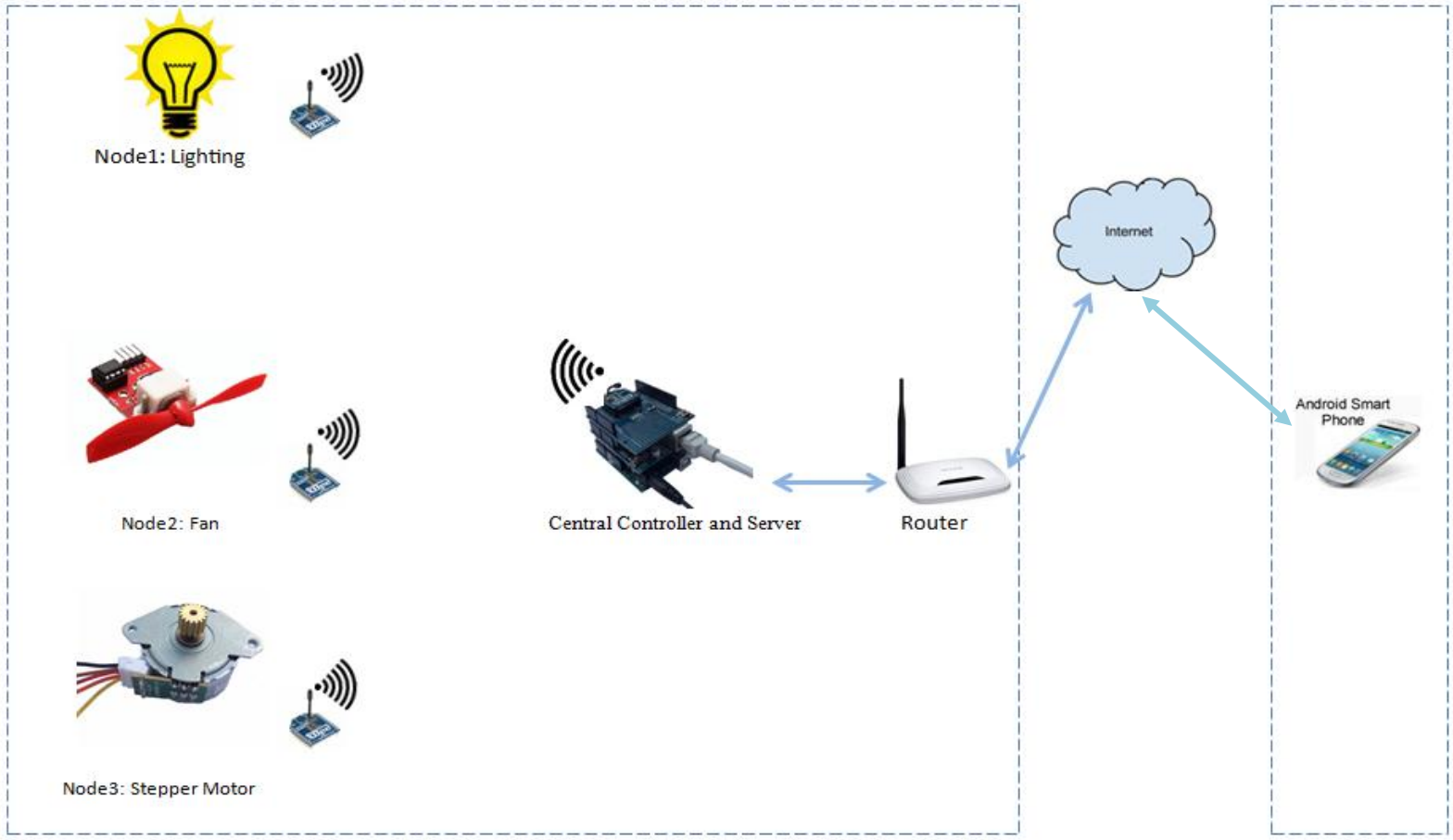


Figure 3.1: The block diagram of the developed system

3.1 Node 1: Light

As shown in Figure 3.2 Node1 composed of two LEDs, the left red LED is used to know if the node is powered, and the right green one is the LED is wanted to be controlled remotely. Because of sensitiveness of LEDs to current, two resistors are used to limit the current flow in them in a safe value. Two AA batteries are used to provide the power for this node. For making the prototyping circuit; a solderless breadboard is used to connect the components of this node, the top two row pins and the bottom two row pins are horizontally connected underneath while the others vertically connected to each other.

This node allows the following functions:

- Turn LED on
- Turn LED off

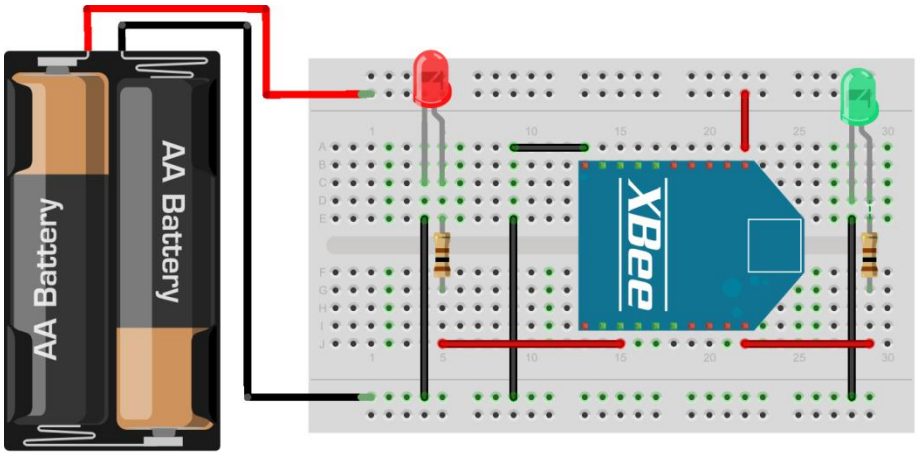


Figure 3.2: Node1 Circuit layout

Figure 3.3 shows the schematic diagram which illustrates the connections between the components of this node. The positive leads of the green LED and red LED are connected to pin20 (DIO0) and pin 13 (DIO9) of the Xbee respectively with a resistor in between for each, and the negative leads of the LEDs are connected to the ground line. The pin10 (GND) of the Xbee is connected to the ground line and pin1 (VDD) is connected to the positive end of the battery.

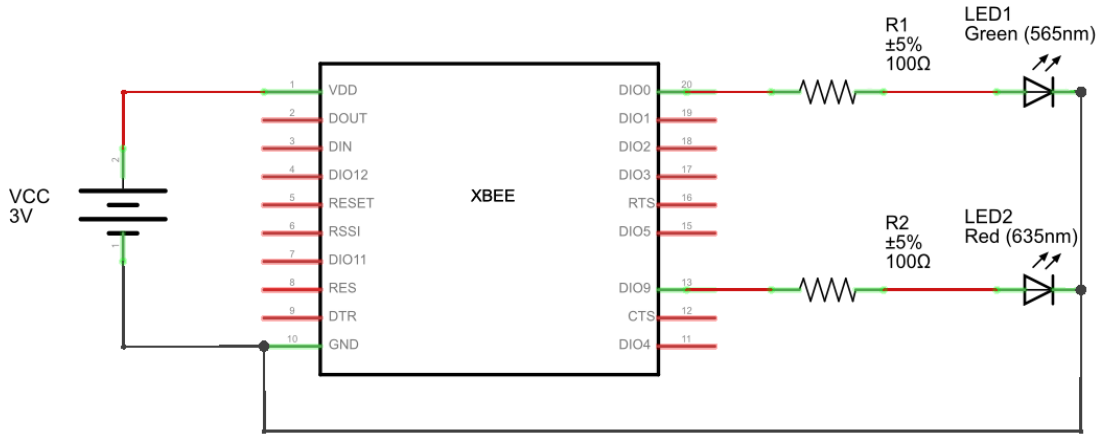


Figure 3.3: Node1 schematic diagram

The node communicates with the central controller wirelessly. After studying various available technologies for wireless communication, Zigbee is selected for this system. The other available technologies were:

- Bluetooth
- Wi-Fi
- Z-Wave
- Ultra-Wideband (UWB)
- INSTEON
- Many others.

Each of these protocols have their own pros and cons. We choose Zigbee because it is more reliable, has greater range and provides easy configuration. We choose Xbee modules by Digi International for implementing Zigbee in our project. There are following advantages of using these modules:

- These have built in MCU which saves the cost of using an extra MCU with every node.
- These are easily configurable for use via their XCTU software.
- These can easily be configured for developing different wireless networking topologies like Start Network and Mesh Network etc.
- These modules have very low power consumption which makes them a perfect choice for battery power of wireless nodes.

3.1.1 Zigbee

ZigBee is a communication protocol which is used for high-level communication (ZigBee, 2015). It is implemented using low cost and low power small digital radios. It is based on IEEE 802.15.4 standard. As the radios which are used are low power radios so its range is limited, typical 40m indoors (although there are some Zigbee modules available now with ranges as far as 10Km), but the range can be extended by making mesh networks. Mesh networks are basically personal area wireless networks in which the signal is sent from one end to other ends after passing through multiple sensor nodes, which acts as single boosters/routers. Zigbee modules has a defined data transmission rate of 250 Kbit/sec, this is best suitable for sending sensor data or small messages. Following are some of the many applications of Zigbee modules:

- Home Automation Systems
- Traffic Management Systems
- Wireless Light Switches
- Industrial Equipment
- Electrical Meters

Zigbee supports both Star and Tree network topologies and also the Mesh topology in general. However, in every network there must be one central module which acts as a coordinator. In Star networks, all of the modules communicates to a central coordinator while in Tree networks, Zigbee modules acting as “routers” are used to extend communication in the network.

Security is also a prominent feature of the Zigbee networks. 128 bit encryption keys are used in Zigbee networks for the implementation of network security protocols.

3.1.2 Xbee

Xbee is a brand name used by Digi International for a family of wireless networking modules that provides end-point connectivity to devices (Digi, 2015). Majority of these modules implements the Zigbee protocol for wireless communication. Another prominent feature of the Xbee modules is that these have an on board user-programmable

microcontroller with a sufficient amount of memory to store code. This makes them a perfect choice for sensing and wireless controlling applications.

Xbee modules are available in different series; following is a brief explanation of all available series:

- **Series 1:** These are the simplest of all the available Xbee modules. These modules do not need configuration to make them work. These are most suitable with point-to-point communication. These modules are not compatible with other Series of the Xbee modules so we cannot use them interchangeably with other series
- **Znet 2.5:** These modules are also known as the Series 2 Xbee modules, but these are obsolete now and are replaced with ZB series. Series 2 modules required configuration (XCTU software is used for this purpose) before they can be used. There are two available modes of their operation
  - AT Mode (Transparent Mode)
  - API Mode

  Both modes have their own features and applications. Each mode requires changes of the firmware using XCTU software provided by Digi International.
- **ZB:** These are the current Series 2 modules. These have almost the same hardware as the Znet 2.5 series, the difference is just in their software. These modules are specifically recommended for use in Mesh networks.
- **XSC:** These modules are quite different from other series. These are specifically recommended for applications where a long range is required. They can give a range of approximately 15 miles if used with a high gain antenna. Although their data rate is much less (10Kbps) with comparing to other Xbee modules. These modules has a different command set than the other Xbee modules, so these are not compatible with other series modules.

Another difference, which is in different available Xbee modules, is in their type of antennas.

Xbee comes with following types of antennas:

- Chip Antenna
- Wire Antenna
- U.FL Connector

- RPSMA
- Trace Antenna
- PCB Antenna

Every antenna has its own range and own performance and is suitable for separate applications. Mostly the difference is in the range and the transmitting power of the antennas.As shown in Figure 3.4 Xbee ZB (Series 2) modules are used in the developed system. The key features/specs of these modules are as follows (Digi, 2015):

- Indoor/Urban: up to 133 ft. (40m)
- Outdoor line-of-sight: up to 400 ft. (120m)
- Transmit Power: 2 mW (3dbm)
- Receiver Sensitivity: -98dbm (1% PER)
- RF Data Rate: 250 Kbps
- TX Peak Current: 40 mA (@ 3.3 V)
- RX Current: 40 mA (@ 3.3 V)
- Power-down Current: < 1 µA
- Operating Frequency: ISM 2.4GHz
- Supported Network Topologies: Point-to-point, Point-to-multipoint & Peer-to-peer

Figure 3.4: Xbee ZB (Series 2) Radio Module with Wire Antenna

The pin configuration of these modules is shown in Table 3.1.

Table 3.1: Xbee ZB (Series 2) pin confiquration (Digi, 2015)

Pin No.	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART data out
3	DIN/CONFIG	Input	UART data in
4	DO8	Output	Digital output 8
5	RESET	Input	Module reset
6	PWM0 / RSSI	Output	PWM output 0 / RX signal strength indicator
7	PWM1	Output	PWM output 1
8	[RESERVED]	-	Do not connect
9	DTR / SLEEP / DI8	Input	Pin sleep control or digital input 8
10	GND	-	Ground
11	AD4 / DIO4	Either	Analog Input 4 or Digital I/O 4
12	CTS / DIO7	Either	Clear to send flow control signal or Digital I/O 4
13	ON / SLEEP	Output	Module status indicator
14	VREF	Input	Voltage reference for A/D inputs
15	ASSOCIATE / AD5 / DIO5	Either	Associated Indicator / Analog Input 5 / Digital I/O 5
16	RTS / AD6 / DIO6	Either	Request to Send flow control signal / Analog Input 6 / Digital I/O 6
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 0
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 0
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 0
20	AD0 / DIO0	Either	Analog Input 0 or Digital I/O 0

These Xbee modules can communicate with a host controller using asynchronous serial communication. The RX and TX pins on the Xbee can be connected to the TX and RX pins respectively of any logic level / voltage level compatible host controller. Serial flow control can be achieved by software or via hardware. CTS (Clear to Send) and RTS (Request to send) pins are available on the Xbee modules for hardware flow control. The block diagram of the serial communication between two Xbee connected microcontrollers is illustrated in Figure 3.5.

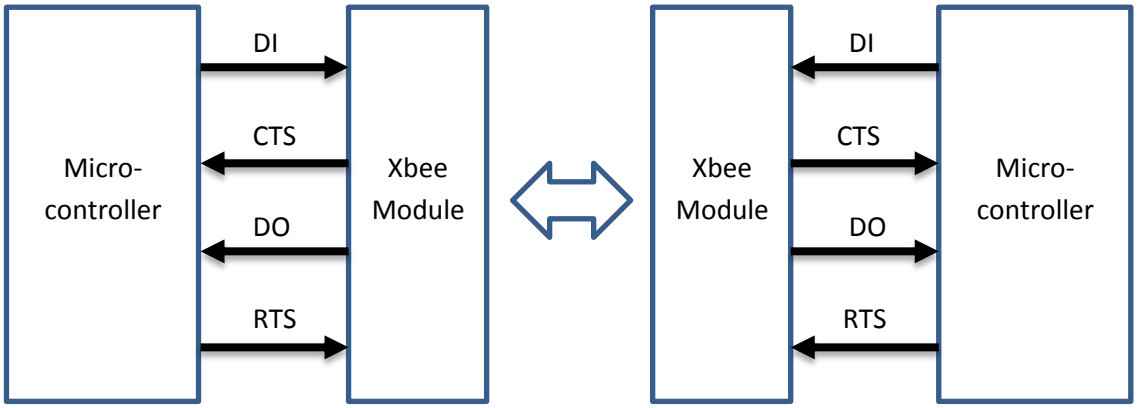


Figure 3.5: System Data Flow Diagram with UART Interface

Xbee Modules can operate in two different modes, i.e. AT mode/ Transparent mode and API mode. By default, these modules operate in the Transparent Mode. In this mode, these modules simply act as a wireless UART. They receive the data from the host, store it in their buffer and then transmit it wirelessly. Advance features, like changing the parameters of other XBee modules in the network or changing the address of the receiving Xbees can be performed via AT commands. The module has first to be transferred to the command mode for this purpose.

The API mode unleashes all of the capabilities of the XBee modules. In API mode, data is transmitted using frames. Each frame comprises of specific packets with specific values. API mode allows the user to change the destination address of each transmitted packet without entering in the command mode. So the API mode is generally more faster way of communication.

To use the API mode, the respective Xbee's firmware has to be updated first using the XCTU software , Figure 3.6 shows the main window of this software.

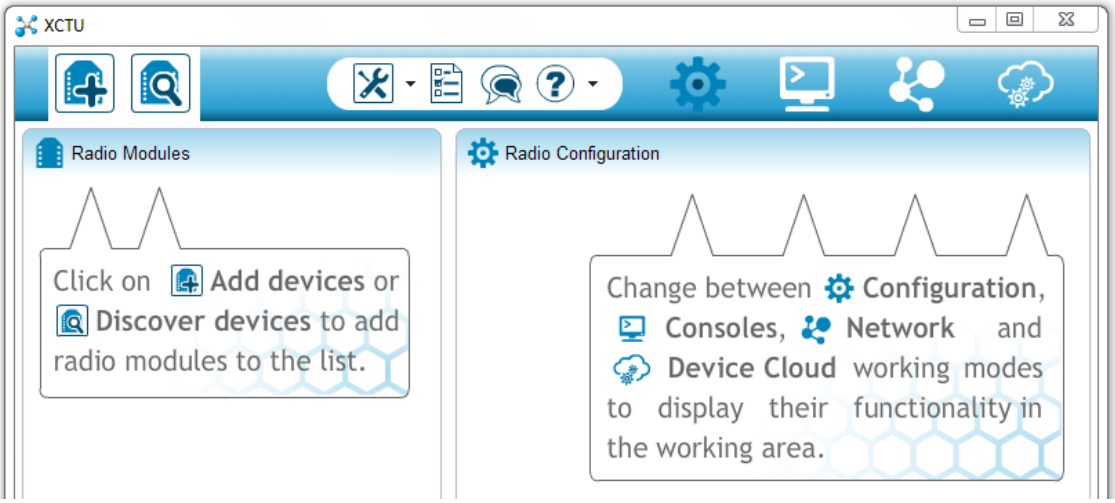


Figure 3.6: XCTU Software by Digi International

3.1.3 Xbee USB explorer dongle

For programming the Xbees or changing their firmware/configuration, they need to be connected to the laptop. This connection interface is different from a regular UART connection interface because Xbee is connected in programming mode (by pulling down its CONFIG pin). A USB dongle is used for this purpose which is also called the Xbee Explorer board. This board has the following features:

- Can use the Xbee as simple UART device
- Can connect the Xbee in its programming mode
- Has indication lights for transmission, receiving and signal strength
- Can be used to change XBee's configuration and firmaware
- Has Internal Logic Level converters for changing the Xbee transmission signal levels from 3.3V to 5V and vice versa
- Has built-in voltage regulator to give supply to the Xbee module via USB power line
- Can be used to easily reset the Xbee module while programming/firmware upgrading

There are different versions of Xbee Explorers available, each having almost the same functionality, just slight changes in the design. Figure 3.7 shows the Xbee Explorer dongle which has been used in the developed system:

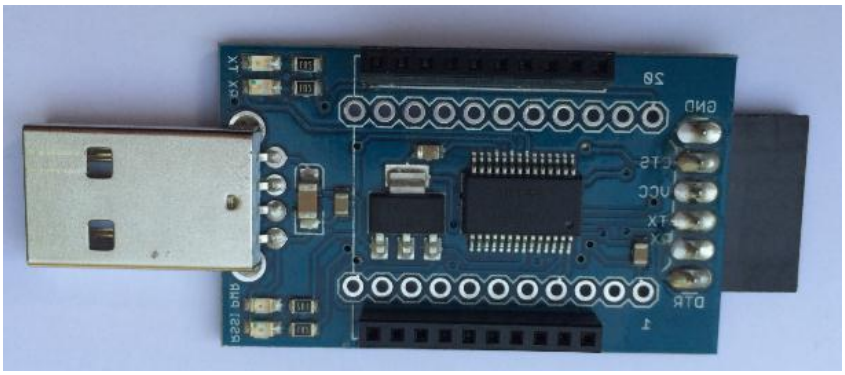


Figure 3.7: USB dongle explorer

3.2 NODE 2: Fan

Node 2 has a Fan connected to it. This fan is being operated with a DC motor. Node 2 offers the following functions:

- Turn Fan ON
- Turn Fan OFF
- Change Fan's Direction

A brushed DC motor is used to rotate the fan. This motor is controlled with a solid-state Motor Driver IC. Following are the main components used in the Node 2 and they illustrated in Figure 3.8:

- Xbee ZB (series 2)
- Brushed Motor Driver
- L9110 Motor Driver IC
- SwitchingVoltage Regulator

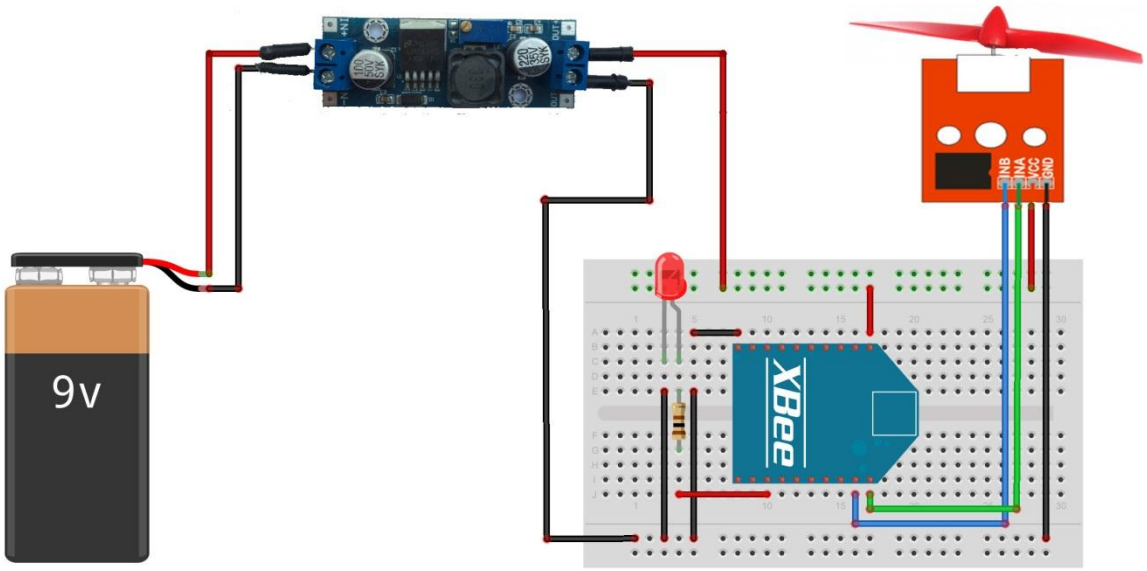


Figure 3.8: Node2 Circuit layout

The connections between the parts of this node are illustrated in Figure 3.9

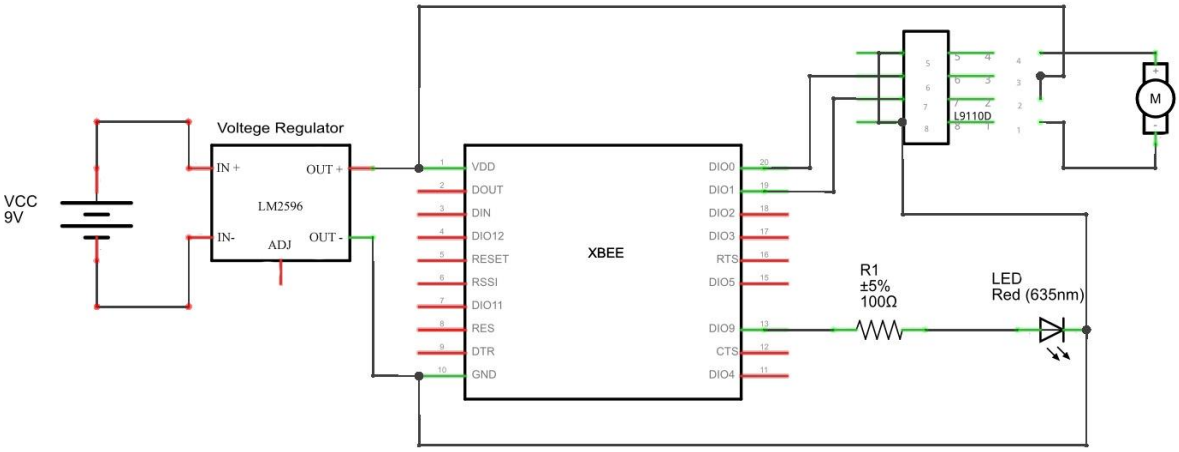


Figure 3.9: Node 2 schematic diagram

3. 2.1 DC motor

As per definition, DC motor is an electrical machine which takes direct current electrical power and converts it into mechanical power. It relies on the basic principal of electromagnetism, according to which, two same magnetic poles repel each other. Every

DC motor has some kind of mechanisms in it to change the direction of the current in its coil. It has two parts:

- Rotor
- Stator

Stator is stationary in space and its current is also stationary in space. Rotor is the rotating part but the current in it is also stationery in space and switched by commutator. In this way relative angle is maintained between stator and rotor. Maximum torque is achieved when magnetic field is at 90 degrees. There are two types of windings, rotating armature winding and static field winding, magnetic field is non-rotating for armature. We can get different speeds and torques regulation characteristics by applying different connection schemes of field and armature windings. Speed of motor is controllable generally; it can be controlled by changing voltage applied to armature or by changing current of field, if we introduce resistance in armature or field circuit allowed speed control.

They have wide applications they have eliminated local steam engines or internal combustion engines. These motors can be used directly from rechargeable batteries. They have wide range of application from small tot to steel rolling mills and paper machine. A typical brushed DC motor presented in Figure 3.10 in cross-section (Mulder, 2001).

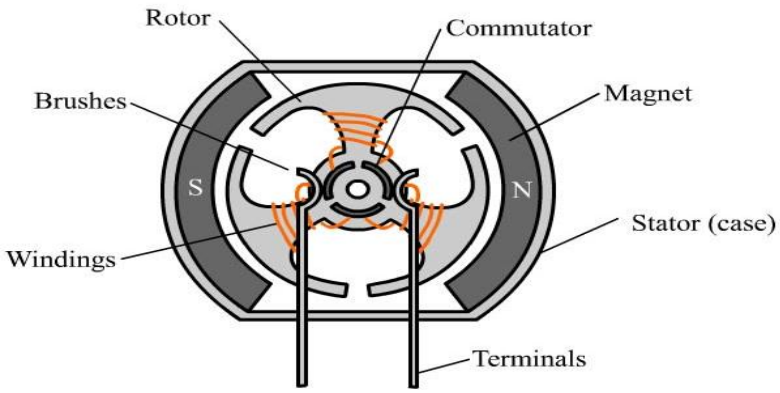


Figure 3.10: Typical Brushed DC Motor in cross-section (Mulder, 2005)

3. 2.2 Brushed DC motors

Brushed DC electric motor produces torque which is coming directly from DC power supplied to motor by internal commutation stationery magnets and rotating electrical magnets. This motor also observes torque due to Lorentz force phenomena. Brushed DC motors can be operated in variable speeds by varying operating voltage or by varying strength of magnetic field. Brushed electric motors are reliable, have low initial cost and have simple control of motor speed. But it also have some de-merits in it, one is their high maintenance if they are used for high intensity then they have low life span. Their maintenance involves regular replacing of springs and brushes, cleaning and replacing of commutator. All these things are used for transfer of electric power from outside to inner spinning wire windings of rotor inside motor. These brushes are made up of any conductor. Brushed motors are used in electric propulsion, cranes, paper machines and steel rolling mills (Mulder, 2005).

3.2.3 Motor control

It is not efficient to run DC motors directly. It has different disadvantages associated with it i.e. we cannot control the speed of the motor, we cannot control the direction of the motor. So different control strategies are employed for motor control:

3. 2.3.1 ON/OFF control

DC motor can be controlled by a number of ways, On/Off is the simplest method among all these methods. It consists of a single switch which allows the motor to move only in one direction. Either it will move with maximum speed of rotation in one direction (means ON) or it stops. This control can be made by a switch and a free wheel diode, we use this diode in order to avoid any back EMF which can damage the motor (Arthur and Cote 2003).

3.2.3.2 PWM control:

Pulse width modulation also known as pulse duration modulation is a technique that co-ordinate with the width of the pulse duration based on information on modulator signal (see Figure 3.11). This modulation technique is usually used for encoding information for transmission and its main usage is to allow the control of power supply to electrical

devices like motors. Pulse width modulation is also used in photovoltaic solar battery chargers and maximum power point tracking. Power is supplied to load via PWM by switching the switch between load and supply, on and off. The longer the switch is ON, the more power is delivered to the load (Prakash, 2000).

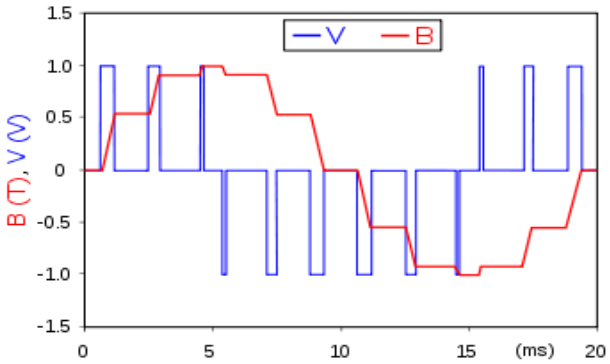


Figure 3.11: Pulse Width Modulation (Zureks, 2007**)**

Duty cycle is the term that describes the portion of time in which the switch is ON, a low duty cycle responds low power and high responds to high power, and it is expressed in percentage (see Figure 3.12). Power losses in pulse width modulation are very low. PWM is used in communication where duty cycle is used to convey information over communication channel.

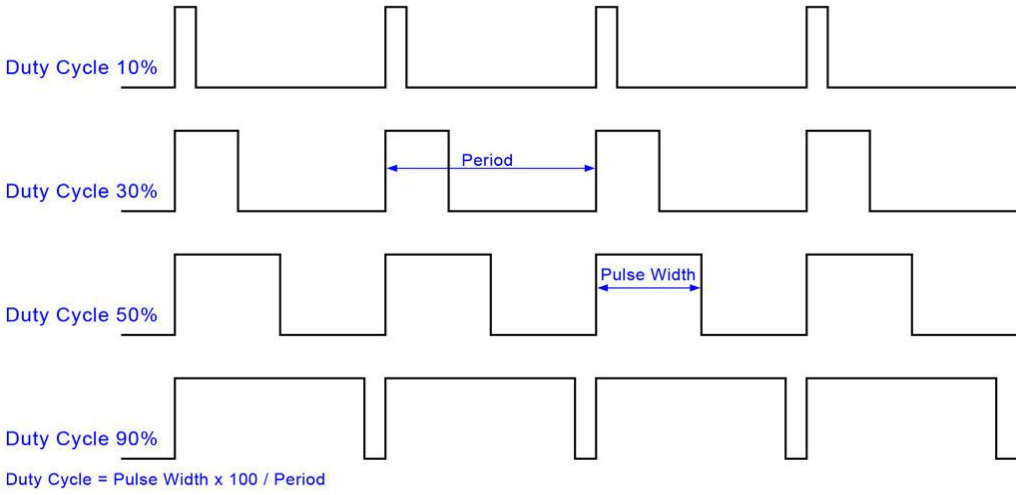


Figure 3.12: Duty Cycle and Time Period (Proto Stack, 2011)

To control speed of motor, we require variable power source. When power is supplied to motor it will start taking speed and in small time it will reach full speed. If we cut the source, the speed of motor will get down, and if we switch ON and OFF the supply quickly the motor will run at some speed between zero and full speed this is the way the pulse width modulation works, it switches motor in series of pulses. Speed is controlled by the width of pulses, which is called as pulse width modulation (Huang, 2005).

3.2.3.3 Relay control

A relay is defined as a switch which is operated on electricity, in many relays opening mechanism is controlled by electromagnet. Full voltage supply can be provided to motor as there is no voltage drop across relays.

There are some terms associated with relay and motor control (Herman, 2014):

- Normally Open contacts
- Normally Closed contacts
- Common Contacts
- Coil

Normally open contacts are used to connect circuit if relay is activated and the same circuit is off if relay is not active. Normally closed contacts are those, which show opposite behavior to normally open contacts. The common contacts are known as changeover. The coil is the electromagnet coil inside the relay. When the coil gets fully activated then the voltage at that point is called coil ratings.

We used a simple contact switch which activates the motor to drive any load with in the limit. Particularly two SPDT momentary contacts limit switches; two SPDT relays and one toggle switch are used for this purpose. Normally relays are configured in reverse conditions, in this way voltage is present on both sides of motor. One relay is normally opened while the other is normally closed. In this condition, no power is flowing and motor is stationery. Then we open that relay, power will flow and the motor starts to rotate. A proper circuitry is used for this purpose (Meyer, 2008).

3.2.4 H-Bridge

H Bridge is an electronic circuit whose function is to enable the voltage to be applied across load in either direction i.e. in positive direction and in negative direction. It is also called as full bridge. A functional block diagram of H-Bridge is shown in Figure 3.13.

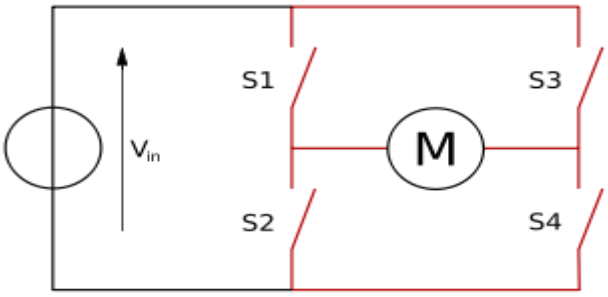


Figure 3.13: Functional block diagram of H-Bridge (Buttay, 2006a)

H-Bridge can be made from discrete components and also available as integrated circuits. It is called H Bridge because of its circuit shape which appears to be an H. This circuit consists of four switches; they may be solid-state switches or mechanical switches. These switches are shown in figure above. These four switches are often called as high side left, high side right, low side right and low side left. S1 and S4 are closed then S2 and S3 are open and positive voltage will be applied across motor. And when S2 and S3 are closed by making S1 and S4 open voltage is reversed allowing reversing operation of motor (see Figure 3.14).

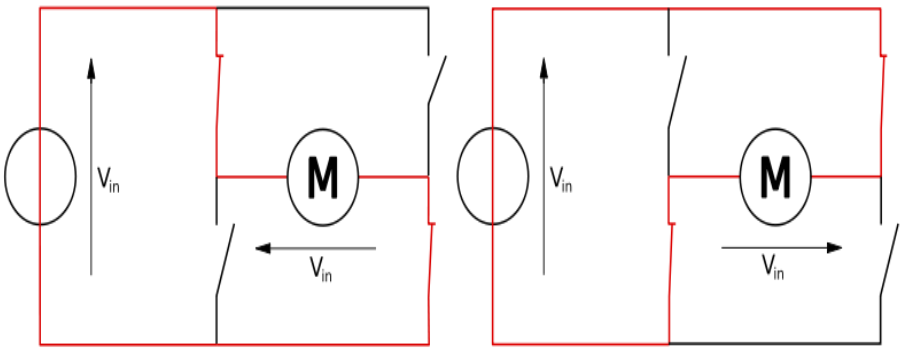


Figure 3.14: Forward operation and backward operation of H-bridge (Buttay, 2006b)

If S1 and S2 or S3 and S4 are closed at the same time it would cause short circuit, this condition is known as shoot-through. With H-bridge, a motor can also supplied with

reverse polarity current, this changes the direction of rotation of the motor. Furthermore, brake can also be applied and the motor can be allowed to work in free-run mode as well. Table 3.2 shows the operation of the h-bridge:

Table 3.2: States of H-bridge

S1	S2	S3	S4	Result
1	0	0	1	Motor moves right
0	1	1	0	Motor moves left
0	0	0	0	Motor free run
0	1	0	1	Motor brakes
1	0	1	0	Motor brakes
1	1	0	0	Shoot-through
0	0	1	1	Shoot-through
1	1	1	1	Shoot-through

H Bridge is constructed by using opposite polarity devices like BJTs or MOSFET. PNP BJT and P-channel MOSFETS are used for high voltage bus and NPN BJT and N- channel MOSFETS are used for low voltage bus. There are many ways to use these solid state switches, the most efficient way is to use N-channel MOSFETs on both sides, at high side and low side, because these have one-third of the ON resistance of P-channel MOSFETs. High side MOSFETs are driven positive with respect to DC supply. There are many other methods related to these switches (Williams, 2002).

3.2.5 L9110 motor driver

The L9110 is an Application Specific Integrated Circuit (ASIC) which is designed to control one or two DC motors. It can drive a single DC motor in bi-directional mode while two DC motors in uni-directional mode. According to its datasheet (L9110, n.d.), it consists of two push-pull amplifiers which drives the motor. The two push-pull amplifiers basically realize an H-Bridge circuit as explained above. It also has built-in clamp diodes for the protection of the driving circuitry from inductive kick of DC motor. It can supply up to 800mA continuous current and 1.5A initial current, which is well suited for our application as we are using a small DC Motor for demonstration (see Figure 3.15).

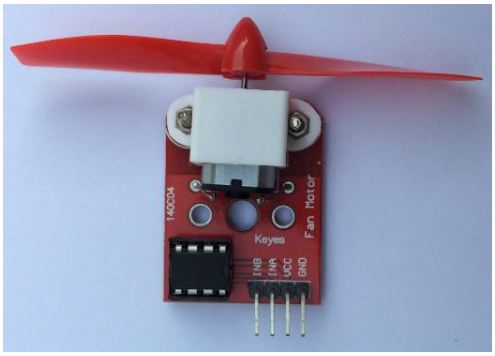


Figure 3.15: L9110 Motor Driver

Table 3.3 and Figure 3.16 show the pin connections of the IC and their description:

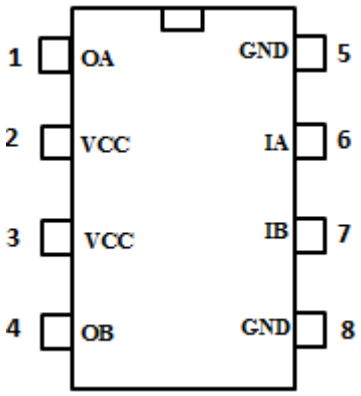


Figure 3.16: L9110 Sketch (L9110, n.d.)

Table3.3: L9110 pin discription (L9110, n.d.)

No.	Symbo	Function
1	OA	A road output pin
2	VCC	Supply Voltage
3	VCC	Supply Voltage
4	OB	B output pin
5	GND	Ground
6	IA	A road input pin
7	IB	B input pin
8	GND	Ground

3.2.6 Switching voltage regulator

Another important component which is a part of this node is the Voltage Regulator. It is only used in this node because all the other nodes have already a voltage supply of 3V (2xAA batteries). However, this node has been powered via a 9V battery because it is using a DC motor, which requires more power. The voltage regulator is used to reduce the voltages from 9V to 3.3V. Primarily, there are two types of voltage regulators:

- Linear Voltage Regulators
- Switching Voltage Regulators

Linear voltage regulators are generally cheap as compared to switching regulators, and are more easy to use. What's the catch then? These are far much less efficient as compared to the switching regulators. The reason for this is the difference in the way they step down the voltages. The linear regulators steps down the voltages but dissipating the excess power in the form of heat. This generates another issue i.e. heating. The switching regulators, on the other hand, do not reduce the power by dissipating the extra power somewhere else. They rather "switch" the power via PWM.

Figure 3.17 shows the switching regulator used in the developed system.

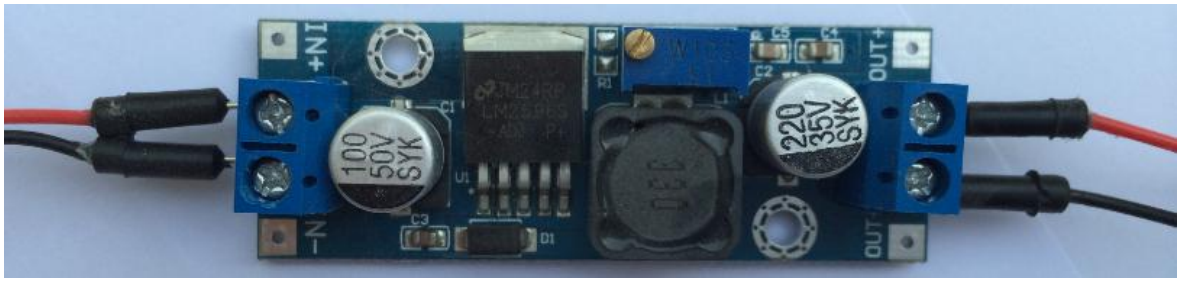


Figure 3.17: Voltege Regulator

There are three major parts of it, the switching regulator itself (with the switching transistor circuitry built in it), an inductor and capacitors. The inductors and capacitors are used to store the energy and then deliver it back to the load during the "OFF" period of the PWM waveform. In this way, the switching regulator delivers a smooth voltage level (either step-down or step-up).

3.3 Node 3: Stepper Motor

Node 3 is different from other two nodes as it has an Arduino board present on it. This node controls a stepper motor and offers the following functions:

- Move stepper motor to a particular number of steps
- Move stepper motor to a particular number of revolutions
- Change the speed of stepper motor
- Change the direction of stepper motor

Arduino is used in this node because the on-board microcontroller of the Xbee is not sufficient to drive the stepper motor. So we've used an extra controller which receives the data from the Xbee connected to its UART and then control the stepper motor accordingly. We are using a unipolar stepper motor and it requires four different signals to its four windings in specific order with specific interval to move the motor systematically in either direction. We tested it with an Arduino and it worked perfectly with it. However, when we connected it to a remote Xbee and then try to control it wirelessly with another Xbee then it didn't work. The reason for this is the communication delay between Xbees. The stepper motor requires quick switching of current from its one coil to another coil, but with Xbees it's not possible because there is inherited delay in the Xbee communication (even if we disable the ACK from the remote Xbee). Therefore, we have to use a microcontroller with the remote Xbee, which receives the command wirelessly via the remote Xbee and then moves the stepper motor according to it. Following are the key components used in this node:

- Xbee ZB (Series 2)
- Arduino Xbee shield
- Arduino UNO (revision 3)
- M35SP Stepper Motor
- ULN2003 based Stepper Motor Driver

Figure 3.18 shows the circuit layout of this node. This node is composed of an Arduino microcontroller to control the stepper motor which receives the data from the Xbee mounted on it then processes the received data to execute the command, the Xbee module in this node is mounted on an Xbee shield so as to be prevented from high voltage and then stacked on the Arduino microcontroller, it performs the wireless communication between this node and the micro web-server. A 9V battery is used to provide the power to the node. A small unipolar stepper motor is used to be controlled remotely. Because the Arduino microcontroller cannot supply sufficient current directly to the stepper motor, an ULN2003 stepper driver is used to drive the motor which amplifies the current and voltage signal from Arduino and then provides it to the stepper motor.

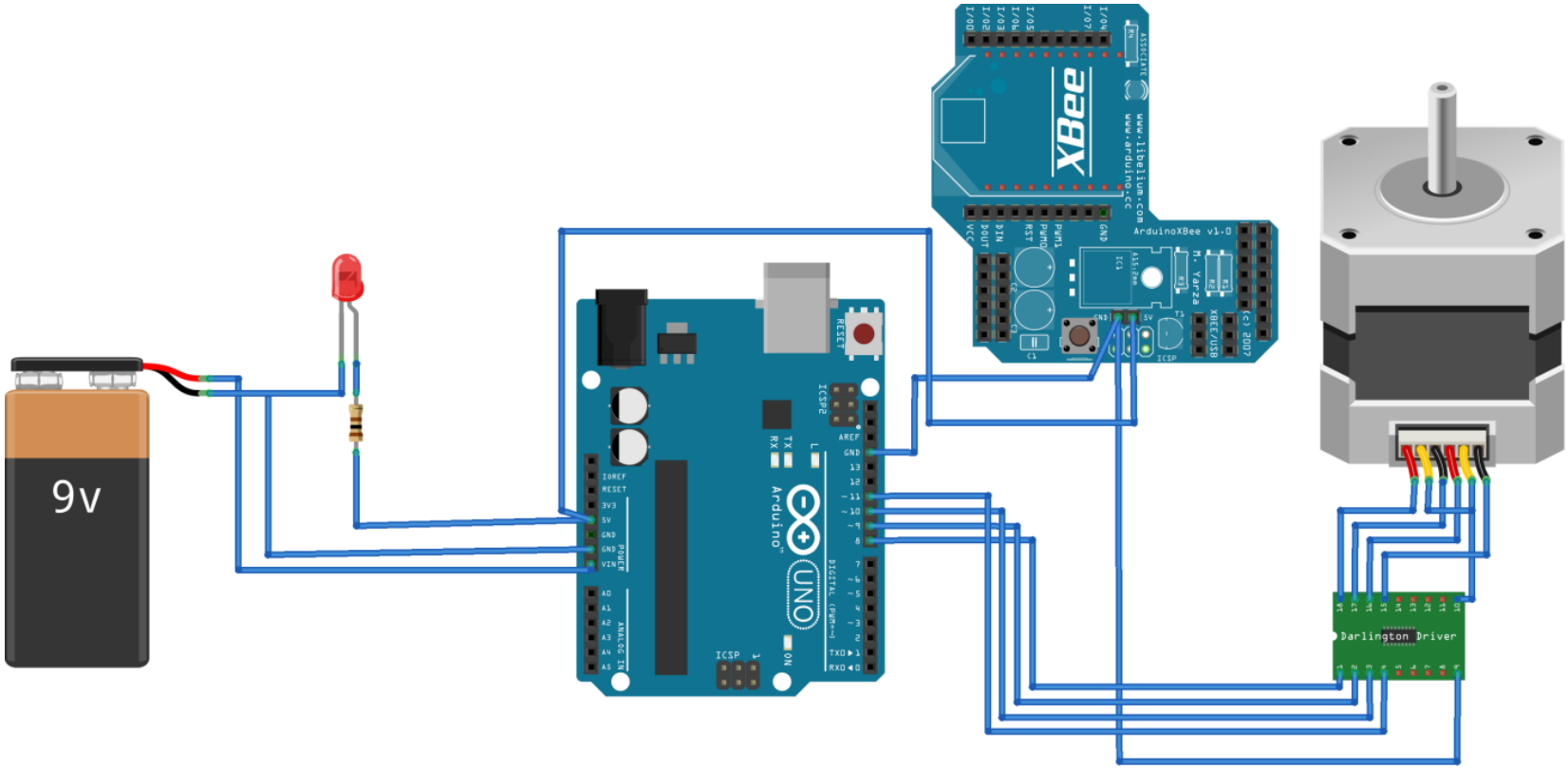


Figure 3.18: Node3 circuit layout

Figure 3.19 represents the above circuit layout in schematic diagram

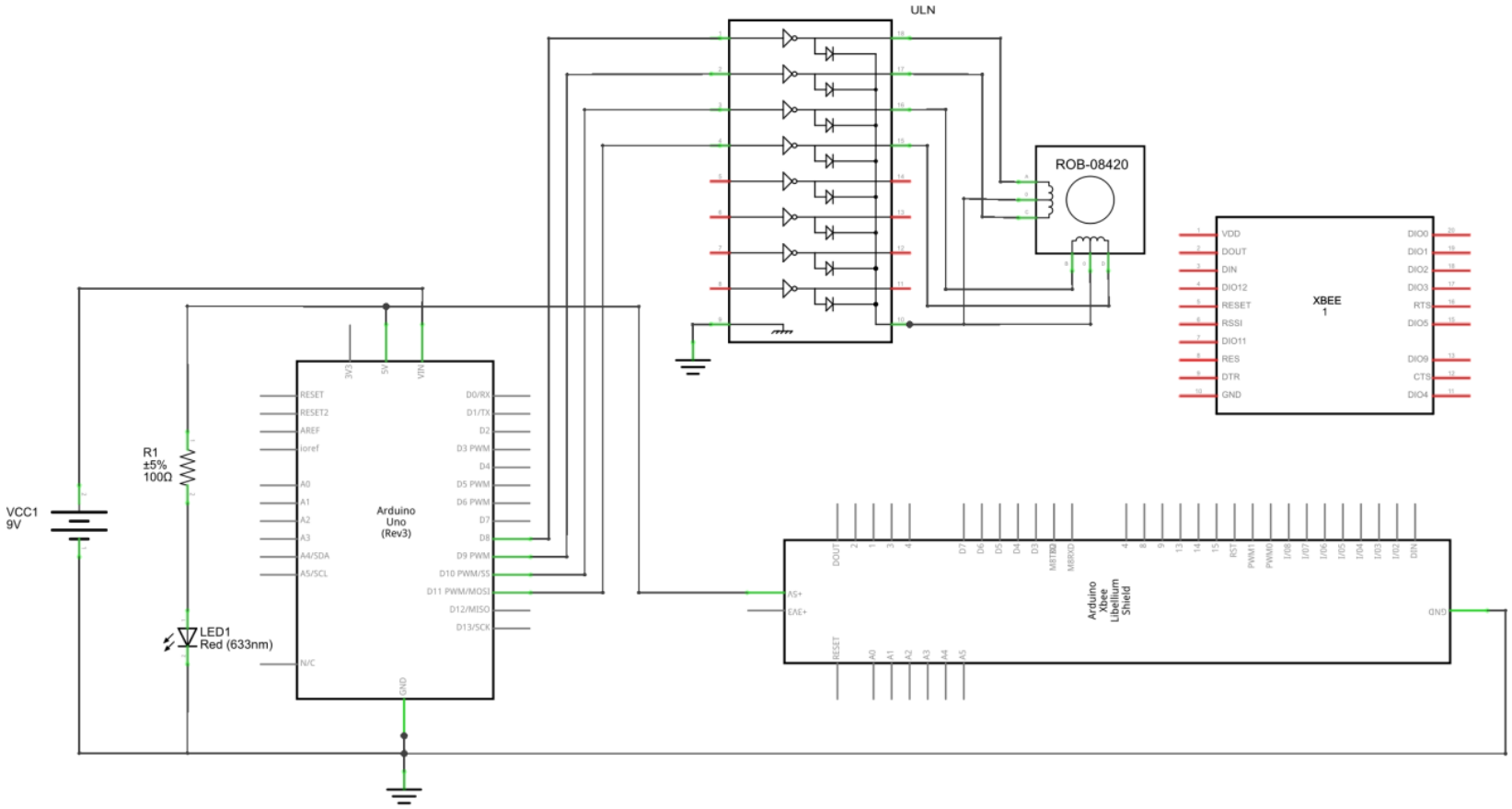


Figure 3.19: Node3 schematic diagram

3. 3.1 Arduino

Arduino is an open-source electronics platform based on AVR microcontrollers (Arduino, 2015). Its purpose is to provide easy-to-use hardware and software to electronics enthusiasts so that not only professionals, but also the beginners can make creative electronic projects out of their imaginations. Arduino community has developed several boards known as Arduino Boards. Each board has its own set of features, processing power and Input/output capabilities. These boards are further supported with several other application-oriented boards known as the Arduino Shields. There are motor shields, audio shields, Xbee shields, servo shields and even Ethernet shields to connect the Arduino to the internet. Following is a list of popular Arduino boards:

- Arduino Uno
- Arduino Yun
- Arduino Due
- Arduino Leonardo
- Arduino Mega
- Arduino Mini
- Arduino Nano
- LilyPad Arduino

Arduino has its own IDE (Integrated Development Environment) known as Arduino. This allows the developers to easily write codes, upload them directly to their Arduino boards, and test them. It is also rich with different examples to control the Arduino boards and the devices/sensors that can be interfaced with those boards. Figure 3.20 shows a screenshot of Arduino IDE.

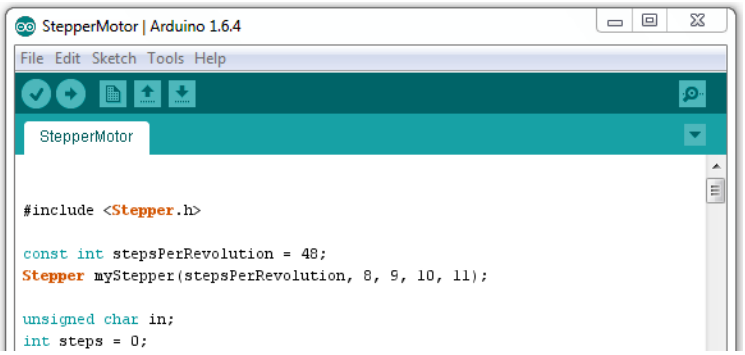


Figure 3.20: Arduino IDE screenshot

3. 3.1.1 Arduino UNO

Arduino UNO as shown in Figure 3.21 is one of the most widely used Arduino boards. It is basic and has almost all of the basic things present in it. All of the available Arduino shields can be used with it. It is based on AVR microcontroller, the Atmega328 from Atmel. It can be connected with the laptop via a USB cable and can be powered up and programmed from the USB port. Following are its key features (Arduino, 2015):

- 14 digital input/output pins
- 6 PWM channels
- 6 analog inputs
- Flash memory: 32KB
- SRAM: 2KB
- EEPROM: 1KB
- Clock Speed: 16MHz
- Communication: SPI, TWI,
  I2C, UART (hardware and software)

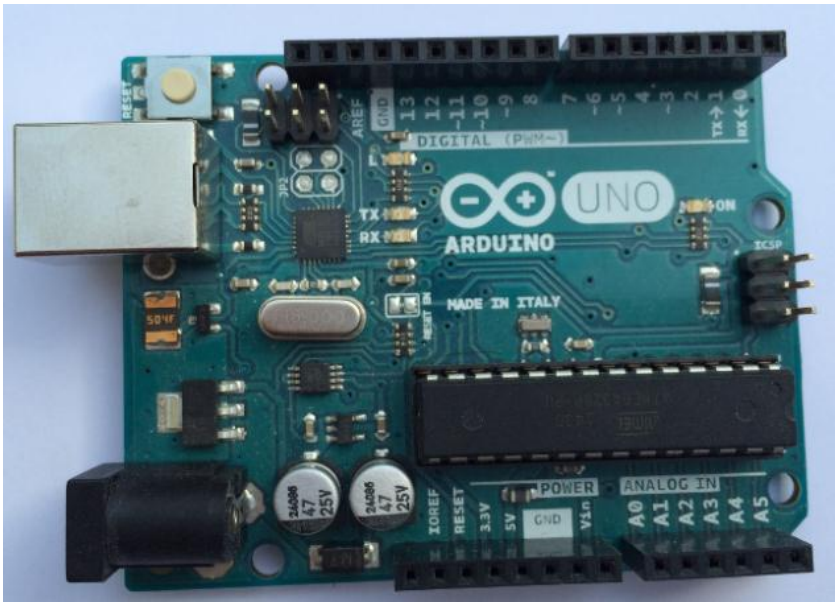


Figure 3.21: Arduino UNO Board (Revision 3)

3.3.2 Stepper motor

Stepper motor is also known as Step Motor. It is also a kind of DC motor but unlike other DC motors, it does not completely rotate on the application of current. It rather moves in discrete steps, one step at a time, that is why it is called Stepper Motor. It is a brushless motor, i.e. it does not have any brushes in it for commutation, rather the commutation and control is performed by external controller.

Stepper motor consists of a central rotor with gear teeth and multiple coils as stator. These coils generate magnetic field in a sequence and the rotor's teeth get repelled from those magnetic fields so the motor rotates. For example, as shown in Figure 3.22, one coil of the stepper motor is energized, this will pull/repel the closest teeth, and the motor's shaft will move one step, after that the next coil is energized which then pull/repel that teeth and the stepper motor will cover another step. In this way stepper motor completes its rotation. Stepper motor can be controlled by a pulse train, which will move it to a certain number of steps covering a certain angle. A prominent advantage of stepper motors is that these can rotate to a certain degrees with a great precision without using any feedback mechanism.

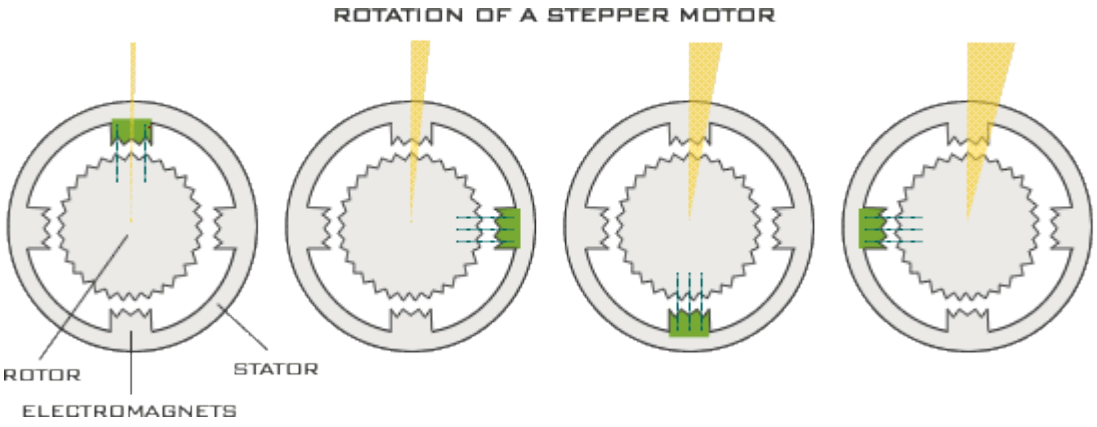


Figure 3.22: Stepper Motor internal diagram (Loop Techbology, n.d.)

Stepper motors has four main types (Bhatia et al., 2015):

- Permanent magnet stepper motor
- Hybrid synchronous stepper motor
- Variable reluctance stepper motor
- Lavet type stepping motor

As the name reflects, permanent magnet stepper motor uses permanent magnets in their rotors while the variable reluctance stepper motors has rotors made up of plain iron. On the other hand, the hybrid synchronous stepper motors uses a combination of both of these techniques to achieve even more power. Irrespective of their type, the stepper motors has following key characteristics:

- These can rotate in either directions
- These have defined steps of movements; each step corresponds to a precise change in the angular displacement
- These have a defined "holding torque" at zero speed
- These are capable of digital control

Generally, the stepper motors has "steps per revolution" of 12, 24, 72, 144, 180, and 200, corresponding to respective shaft increments of 30, 15, 5, 2.5, 2, and 1.8 degrees per step (Jones, 2014).

From the prospective of winding arrangements, stepper motor has the following two types:

- Unipolar Stepper Motor
- Bipolar Stepper Motor

3. 3.2.1 Unipolar stepper motor

As the name suggests, unipolar stepper motor requires only a single power source or a uni-polarity power source for their operation. On the contrary, the bipolar stepper motors require two power sources or a source with changing polarity to drive the motor. Let's take an example of a two winding unipolar stepper motor which is shown in Figure 3.23. This motor has total 6 poles (3 north and 3 south) and has a step angle of 30 degrees. This is one of the simplest design of a unipolar stepper motor. Generally unipolar stepper motors comes with 5 or 6 wires with one wire taken as the common wire. Each winding has a center tap and the center taps of both windings are connected together to make the common point. The common point is normally provided with positive supply and the other two terminals of each winding are alternatively grounded to rotate the motor in either directions. To rotate the motor continuously, each pole is energized (at a time) in a sequence. For example, the following control sequence will move this motor 6 steps or 1 complete revolution:

Winding 1a: 1000100010

Winding 1b: 0010001000

Winding 2a: 0100010001

Winding 2b: 0001000100

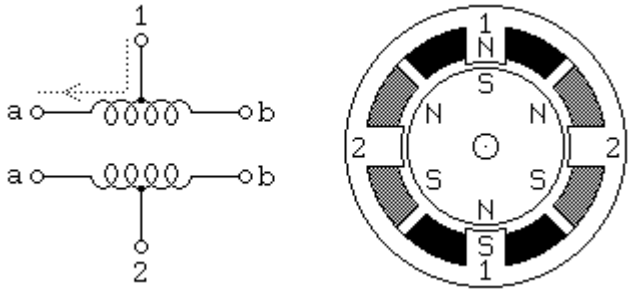


Figure 3.23: Unipolar Stepper Motor internal diagram (Jones, 2014)

3. 3.2.2 Bipolar stepper motors

The internal construction of bipolar stepper motor is same as a unipolar stepper motor. The difference lies in the connections of the windings. The two windings do not have any center taps in them. So the construction is simpler than a unipolar stepper motor but the driving circuitry is more complex. If we refer to Figure 3.24 which shows the diagram of a bipolar stepper motor, we can see that the internal construction is same, just the connections of the windings are different. An H-bridge is required to drive bipolar stepper motors, because such motors require a change in the current polarity to drive them. As explained in the earlier sections, an H-bridge allows the change of current polarity to a given load.

Bipolar stepper motors are generally more powerful as compared to unipolar stepper motors of the same size. The reason for this is the better utilization of the windings space. The windings of a bipolar stepper motor requires less space as compared to that of a unipolar stepper motor because these do not have any center taps connections.

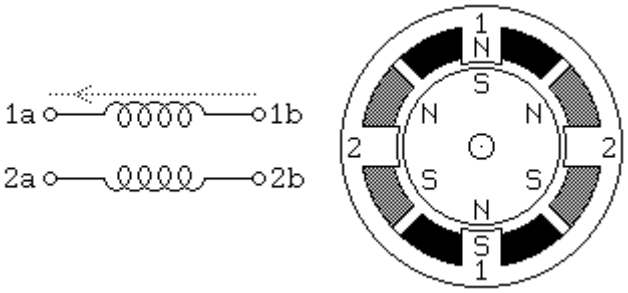


Figure 3.24: Internal diagram of a bipolar stepper motor (Jones, 2014)

3. 3.2.3 M35SP-7NP stepper motor

We have used M35SP-7NP stepper motor. It is a small unipolar stepper motor with a step angle of 7.5 degrees. Following are its key specs (M35SP-7NP datasheet):

- Outer diameter: 35mm
- Holding torque: 29.4mN·m
- Operating voltage: 6V
- No. of Phase: 4 phase
- Excitation method: 2-2 phase excitation (unipolar excitation)

Figure 3.25 shows the stepper motor we used in this thesis.

Figure 3.25: M35SP-7NP unipolar stepper motor

3. 3.3 ULN2003 stepper motor driver

The ULN2003 from Texas Instruments is basically a High Voltage, High Current Darlington Transistor Array. As it can be seen in Figure 3.26,this driver consists of seven NPN Darlington transistor pairs that feature high-voltage outputs with common-cathode clamp diodes for switching inductive loads. Each single Darlington pair is rated at 500 mA. Following is a circuit of a single Darlington pair and the logic diagram of the driver.

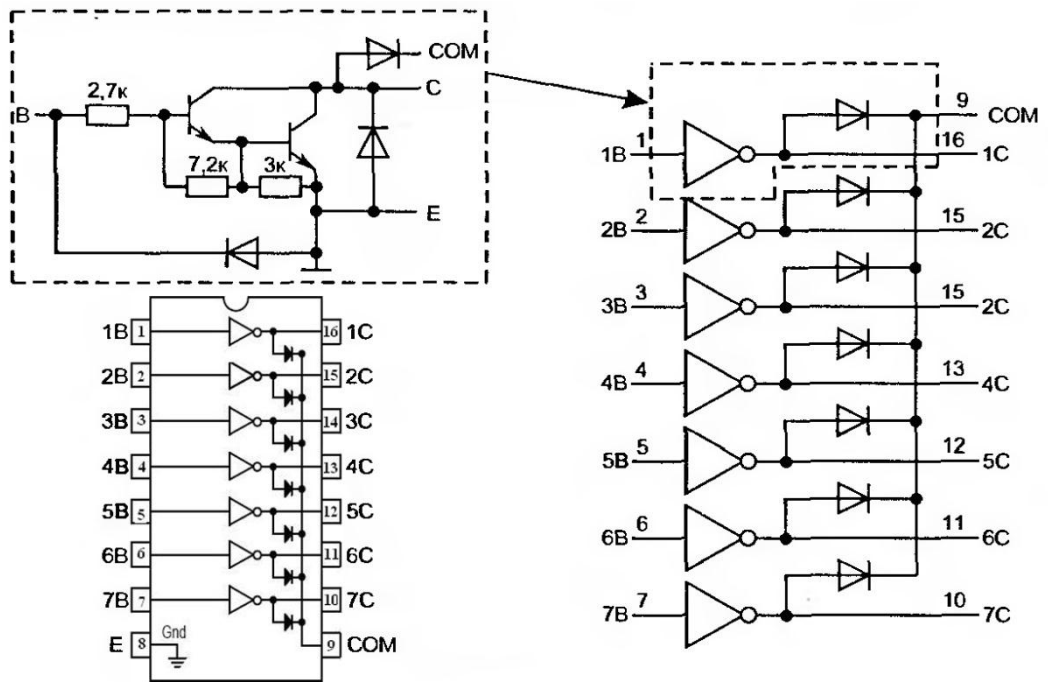


Figure 3.26: Logic diagram and internal circuit of ULN2003 (ULN2003a, 2013).

We have used this IC as a stepper motor driver because stepper motor is an inductive load and required higher currents to operate (Blum, 2013). An Arduino or any other microcontroller cannot directly supply this much current so a driver circuit is required to drive such inductive loads. It amplifies the current and voltage signal from Arduino and then provides it to the stepper motor. It also provides protection from the inductive kick of the stepper motor.

3.4 Central Node : Home Micro Web-Server

The central node is based entirely on Arduino. It is the heart of this system. It coordinates with all the nodes and serves as a medium of communication between the nodes and the android application. It consists of the following parts:

- Xbee ZB (Series 2)
- Arduino UNO
- Arduino Xbee Shield
- Arduino Ethernet Shield

The circuit layout of this node is shown in Figure 3.27.

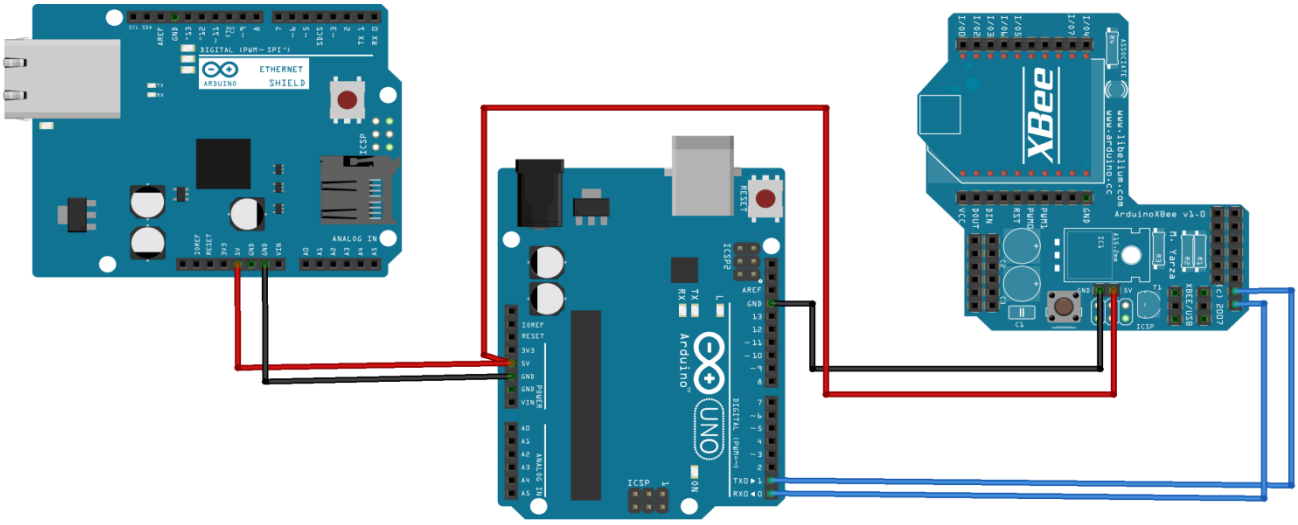


Figure 3.27: Central Node: Arduino Ethernet Server

And the connections of this node is illustrated in Figure 3.28

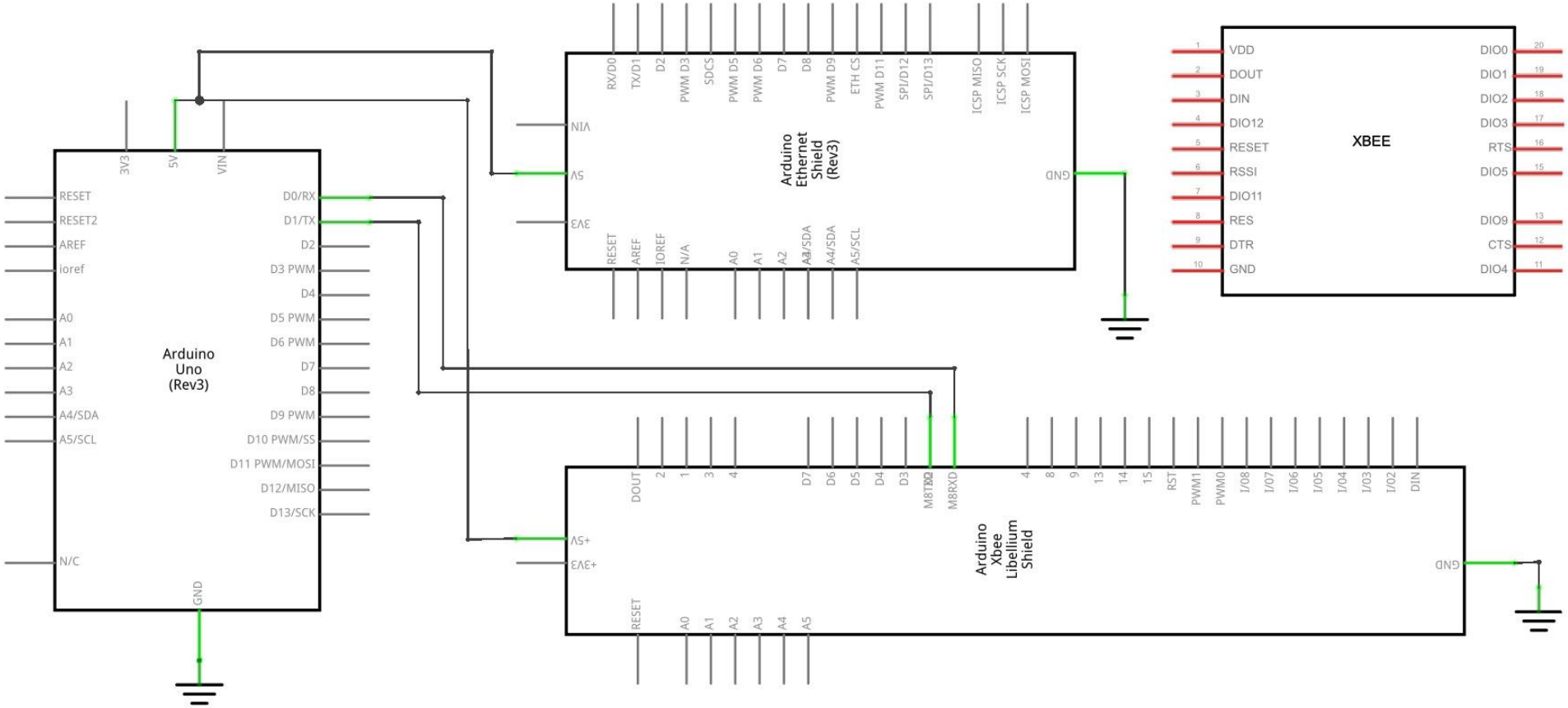


Figure 3.28: Central node schematic diagram

The Xbee mounted on this central Arduino has a different configuration as compared to the other Xbee modules present on the nodes. This Xbee is configured as a coordinator; it can communicate with every node and can change their peripherals via API frames. All the other nodes can only communicate with the coordinator; those cannot communicate with each other. The Arduino connects with the internet via the Ethernet shield mounted on it. There is a lightweight internet server running on the Arduino, which takes the requests from the internet and then process those and act according to the given instructions. The details about it are given in the next chapter.

3.4.1 Arduino Ethernet shield

Arduino Ethernet shield (see Figure 3.29) is used to connect the Arduino with any network in general, via a LAN connection (RJ45 cable), and internet in particular. This Arduino shield is an open source board like the Arduino and it is supplied with tens of examples to interface the shield with different devices. At the core of this shield is the Wiznet W5100 Ethernet chip.

The W5100 is a full-featured, single-chip Internet-enabled 10/100 Ethernet controller which is designed for embedded applications. It facilitates easy implementation of Internet connectivity without OS. The W5100 includes fully hardwired TCP/IP stack and integrated Ethernet MAC & PHY. Hardwired TCP/IP stack supports TCP, UDP, IPv4, ICMP, ARP, IGMP and PPPoE. 16Kbytes internal buffer is included for data transmission. With W5100, there is no need of consideration for handling Ethernet Controller, but simple socket programming is required. Three different interfaces like memory access way, called direct, indirect bus and SPI, are supported on the MCU side (Arduino, 2015).

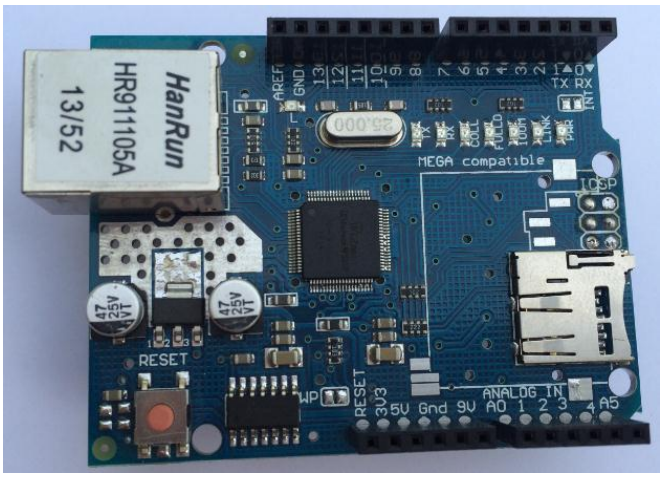


Figure 3.29: Arduino Ethernet Shield

CHAPTER 4
SOFTWARE DEVELOPMENT

4.1 Overview

This chapter is divided into the following parts:

- Xbee networking
- Nodes' software
- Main controller software
- Android app

4.2 Xbee Networking

From networking point of view, there are two sections of this development; one is the networking that is done between all the devices and the central controller and the other is the networking which connects the controller with the network and the internet. For the prior type, Xbee networking is used. Xbees can be configured in multiple ways and with different networking topologies i.e. point-to-point communication, point-to-multipoint communication, for star networks or for tree networks etc. All the configurations can be done by using the XCTU software. Xbees can be divided into three major types with respect to their behavior/function in a network:

- End Device
- Router
- Coordinator

The End Devices are mostly used with sensors/actuators in the network. These are best for least power consumption as they sleep when they are free hence saving power. The end devices can only communicate with the coordinator or the router, these cannot communicate with other nodes and also these are not able to relay data between different nodes.

The routers route the data in the network. These can communicate with coordinator, end devices and also with other routers. They act as intermediate nodes in the network. These

can also be used as the end devices in the network. Nonetheless, routers cannot sleep like end devices, because they must store packets for the end devices.

The coordinators can be termed as the most powerful nodes in a Zigbee network. These are the most capable of all the three types. In contrast with the end devices and routers, there can only be one coordinator in any Zigbee star network. This is the point from where the network originates, it stores all of the information about the network, the security keys etc. so it must not sleep, for that reason, it is mostly not battery powered. The diagram shown in Figure 4.1 illustrates an example of a Zigbee network where the coordinator can communicate directly with the router nodes as well as with the end nodes. The routers can play the role of the end nodes as can be seen on the left of the diagram or act as intermediate nodes to relay data between the nodes as appeared on the right of the diagram. The end nodes are the nodes where the sensors or the devices are connected with, they can communicate with routers as appeared at the top of the diagram or directly with the coordinator as it can be seen at the bottom of the diagram.

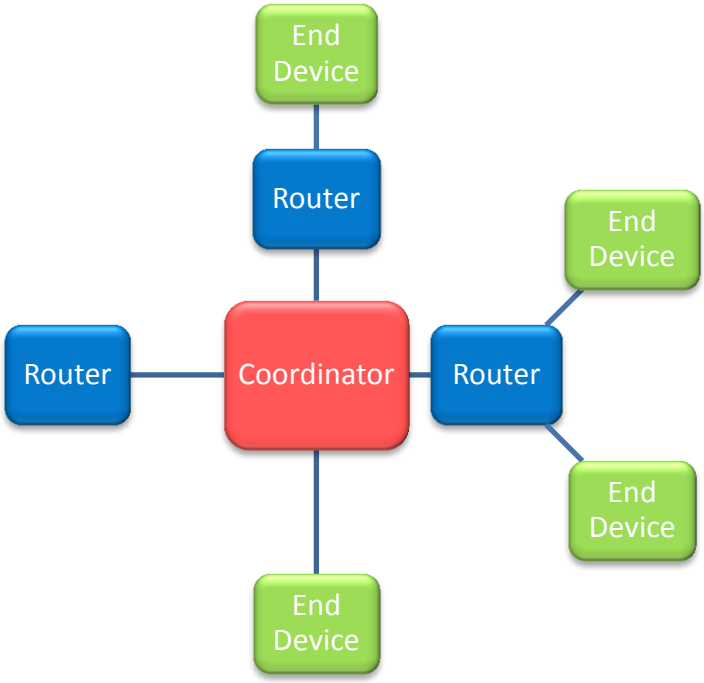


Figure 4.1: An Xbee network with three routers and four end devices

Each node in a Zigbee network has the following important configuration parameters:

- PAN ID (ID)
- Channel (CH)
- 16-bit Source Address (MY)
- 64-bit Destination Address (DH & DL)

The PAN ID or Personal Area Network ID is a unique 2 bytes ID which is assigned to a particular network to distinguish it from other networks in the same area. It can be any number from 0x0000 to 0xFFFF in hex or 0 to 65536 in decimal. Each Xbee in the same network must have the same PAN ID else it will not be able to communicate with other devices present in the network.

The next level in an Xbee network is the communication Channel. Almost all Xbees operate on the 2.4GHz 802.15.4 band; it's the channel that further divides the communication band. Like the PAN ID, all Xbees in a network must have same operating channel.

Next come is the source address and the destination address. Each Xbee must have proper source address and valid destination address; otherwise, it will not be able to communicate with other Xbees in the network. The source address represents the address of the communicating Xbee itself while the 64-bit destination address is the address of the Xbee which is being communicated. For example, if Xbee 1 has a MY address of 0x1111, and Xbee 2 has the same destination address of 0x1111, then Xbee 2 can send data to Xbee 1. But if Xbee 2 has a MY address of 0x1234, and Xbee 1 has a destination address of 0x4312, then Xbee 1 cannot send data to Xbee 2. In this case, only one-way communication is enabled between the two Xbees (only Xbee 2 can send data to Xbee 1).

There are total three nodes and one coordinator used in the developed system. The three nodes are configured as routers instead of end devices because we don't want the nodes to sleep.

Following is the configuration of our Xbee network and the backend of the physical Xbee modules shown on the right:

Product Family: XB24-ZB
PAN ID: 3324
Channel: 19

Central Node: Micro Web-Server
SH: 13A200
SL: 40DC9179
Firmware Version: 21A7

Figure 4.2: Coordinator Xbee backend

Node1: The Light
SH: 13A200
SL: 40E43C0C
Firmware Version: 23A7

Figure 4.3: Light Node Xbee backend

Node2: The Fan
SH: 13A200
SL: 40E43C04
Firmware Version: 23A7

Figure 4.4: Fan Node Xbee backend

Node3: Stepper Motor
SH: 13A200
SL: 40E43BD0
Firmware Version: 23A7

Figure 4.5: Stepper Motor Node Xbee backend

Figure 4.6 represents Xbee network design in the developed system.

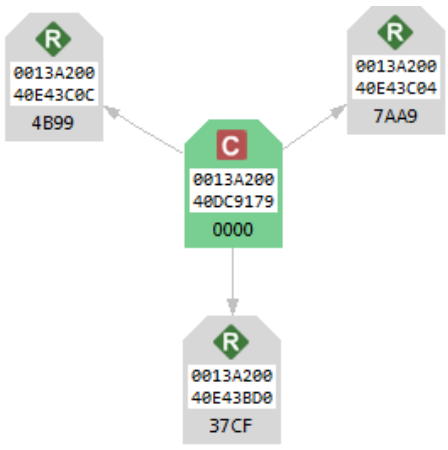


Figure 4.6: Xbees network architecture

4.3 Nodes' Software

The three nodes are programmed to communicate with the coordinator. Two of the three nodes are entirely Xbee based while the third node also has an Arduino board to control the stepper motor.

The LED and the Fan nodes are controlled by one Xbee each; configured as a router. The coordinator sends an API frame with the destination address of the respective node and a Remote AT Command to change the status of any pin on the Xbee. As a result, the LED on the LED node and the Fan on the Fan node turns on and off accordingly. After processing the given command, the respective node sends back an acknowledgement API Frame (the AT Command Response Frame) to the coordinator.

Table 4.1 shows the structure of a typical frame for Remote AT Command:

Table 4.1: Remote AT Command Frame format (Digi, 2015)

Frame Fields		**Offset**	**Example**	**Description**
Start Delimiter		0	0x7E	Fixed
Length		MSB 1	0x00	Number of bytes between the length and the checksum
		LSB 2	0x10	
Frame Specific Data	Frame Type	3	0x17	
	Frame ID	4	0x01	If set to 0, no ACK is sent
	64-bit Destination Address	MSB 5	0x00	Set to the 64-bit address of the destination device. The following addresses are also supported: 0x0000000000000000 - Reserved 64-bit address for the coordinator 0x000000000000FFFF – Broadcast address
		6	0x13	
		7	0xA3	
		8	0x21	
		9	0xD4	
		10	0x59	
		11	0x03	
		LSB 12	0x11	

Table 4.1: continued…

	16-bit Destination Network Address	MSB 13	0xFF	Set to the 16-bit address of the destination device, if known. Set to 0xFFFE if the address is unknown, or if sending a broadcast.
		LSB 14	0xFE	
	Remote Command Options	15	0x02 (apply changes)	Bitfield of supported transmission options. Supported values include the following: 0x01 - Disable retries and route repair 0x02 - Apply changes. 0x20 - Enable APS encryption (if EE=1) 0x40 - Use the extended transmission timeout
	AT Command	16	0x42 (B)	Name of the command
		17	0x48 (H)	
	Command Parameter	18	0x01	If present, indicates the requested parameter value to set the given register. If no characters present, the register is queried.
Checksum		19	0xF5	0xFF - the 8-bit sum of bytes from offset 3 to this byte.

Following prominent functions can be achieved using remote AT commands:

- Set a pin as output or input
- Make a pin HIGH or LOW
- Set ADC resolution of any pin
- Change the PWM frequency and duty cycle at any PWM enabled pin

We have used all the functions to turn On/Off the devices and changing the speed of the attached motor.

The stepper motor node has a bit complex software as compared to the other nodes, because an Arduino board is used with it. The reason of using the Arduino is explained in the earlier section. Figure 4.7 is the flow diagram which shows how the central Arduino is communicating with this node:

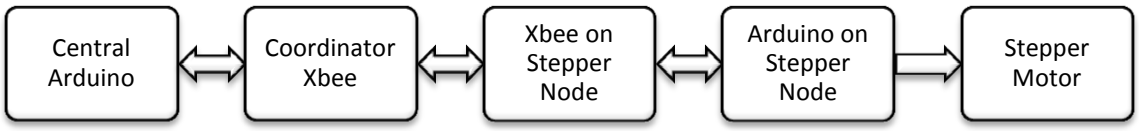


Figure 4.7: Communication between central Arduino and the stepper node

The central Arduino do not send remote AT command frames to this node, it rather sends the Remote Receive Request Frames. Because we are not controlling the Xbee directly now, rather we pass the information to the Arduino on that node and then that Arduino will control the stepper motor itself. So, Xbee is just acting as a wireless UART in this case.

4.4 Main Controller Software

The main controller or the central controller is responsible for communication between the nodes and the android application. It receives information from the Android application, validates it, and then passes it to the respective nodes. It is Arduino based with an Ethernet and an Xbee shield mounted on it. The internet communication is handled by an HTTP server running on it. There is a RESTful API interface which coordinates with the HTTP server to receive the requests and respond to those accordingly. The API allows functions and variables to be controlled with the help of simple HTTP requests. The diagram in Figure 4.8 explains how the software of the main controller is organized and how it receives commands from the internet and delivers those to the nodes.

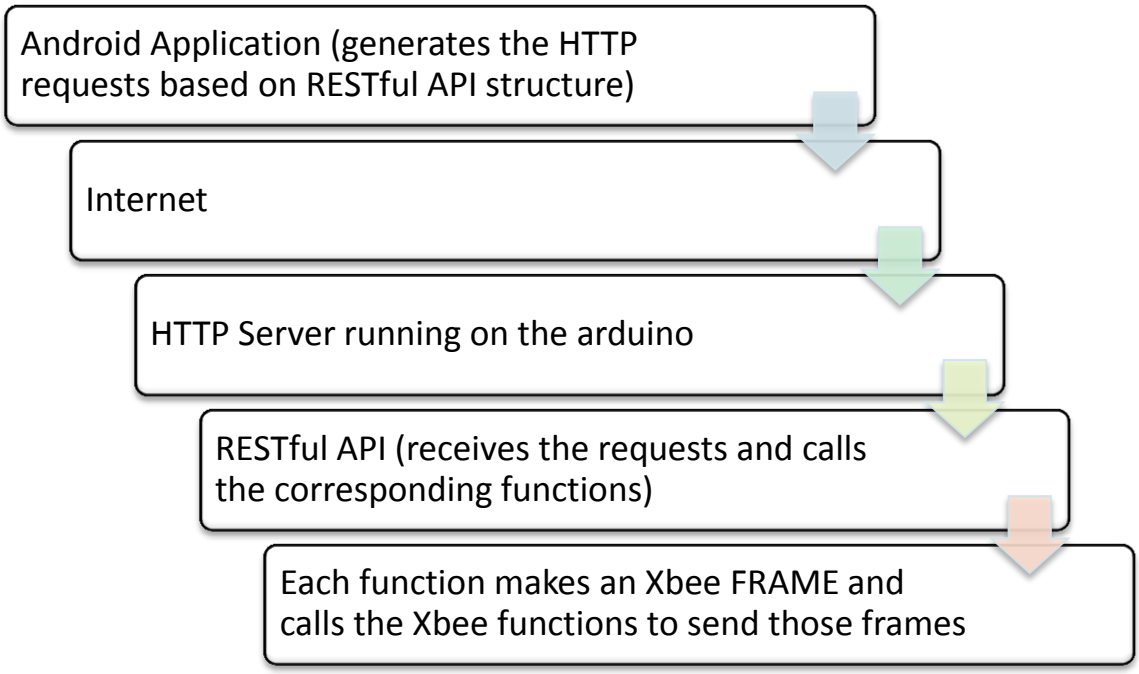


Figure 4.8: Information flow diagram of the central controller

The program continuously checks for any incoming clients which connects with the HTTP server, once a client is connected, the command is passed on to the RESTful API. The API then extracts the received command and passes it to another function called IOT. The purpose of this function is to interpret the received command, it checks if the command is valid or not, if it is valid then it passes it to the corresponding function to process that command. For example, if the received command is "21", then the IOT function will interpret it as a command to switch on the Fan in the Fan node. How it has interpreted this information? It checks the first most digit of the received command, that digit tells it which device to control, 2 means Fan, 1 means light, 3 means stepper motor and so on. The second digit tells it what to do with that device, 1 means switch it on, 2 means switch it off etc. After the information is properly interpreted and passed to the respective function, that function will first check if the user is logged in or not, if not then it will simply return a 0 value to the API which sends it back to the Android application. If the user is authenticated and logged in, then it will make a respective Xbee API frame and send it via its serial port and the Coordinator Xbee. After that, it waits for a response from that node, if there is no response received within a defined time interval then it will again send a 0 to the Android

application indicating that there is an error in the command execution. if a response is successfully received from the node then that function will return a 1 to the Android app, indicating a successful execution of the given command.

To make this system secure, the application is developed in a way only a person who has the password can access the functions of the main controller. Once a login request is generated, the Arduino validates the password and if the password is correct then it sets a flag in the interface to indicate that the user has now logged in and can access the functions. In the same way, logout and password change functions are also provided.

4.5 Android Application

Android application is developed to control the devices from anywhere over the internet. Android Studio is used as an IDE for the development of the Android app. Java is used as the programming language for this app. Any Android project has the following basic files in it:

- **AndroidManifest.xml:** This file contains basic information about the application, like the permissions to use different protocols of the device, the minimum version of android OS, the compiled version, activities names etc.
- **MainLayout.xml:** This file contains the layout information of the application, for every activity in the application there is a corresponding layout.xml file. The details related to the placement of different components (like text boxes, buttons, labels, sliders etc.) are all included in the layout file.
- **Activity class:** This is the part where java is used. Every application has at least one activity class that includes the methods and further classes in it. One important method is the OnCreate method, which defines the behavior of the application when it starts.

Figure 4.9 shows the structure of the developed Android application, the top level shows the package of the app which contains all components of the app, the lower level (second level) contains Android Manifest,Strings, Java&layouts. The third level represents the xml files which they contain the details related to used elements in graphical user interface of

the app. The fourth and the last level of the diagram shows the corresponding activity classes for each of the xml files where the java classes and methods are used to make the components functional.

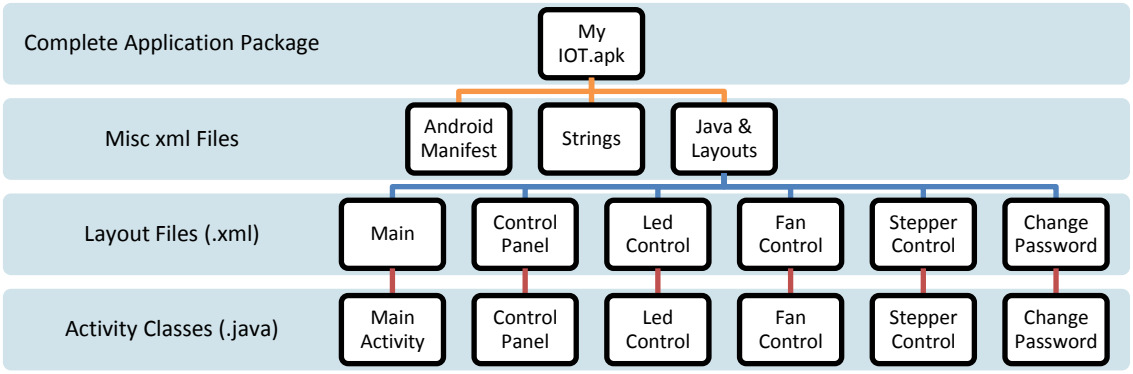


Figure 4.9: The structure of the Android application

There are a total of six screens in our application. The application starts with the main activity which presents a login screen, the user enters the IP address and the password of the server, and the application then checks for the internet connectivity first, if there is no internet connectivity, it displays an error message. If the user is connected to the internet then the application validates the password with the server. On successful validation, the control panel screen will be presented. This screen gives five options to the user:

- Control the Light
- Control the Fan
- Control the Stepper Motor
- Change the Password
- Logout

Each option shows its corresponding activity to the user.

Figure 4.10 expands each window of the app to its components; it shows the options available which the user can use to control the system.

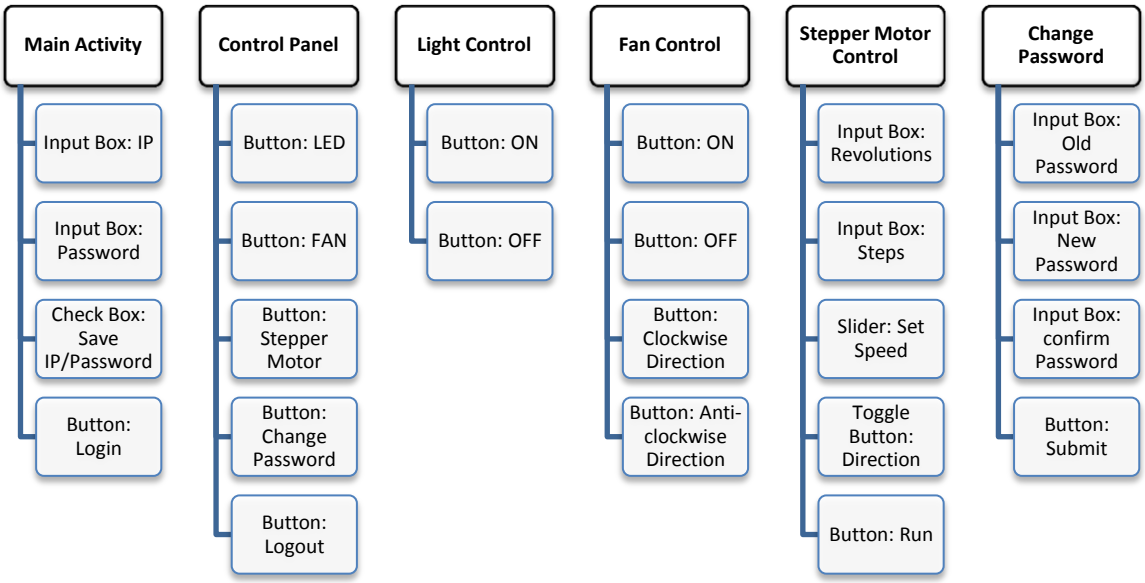


Figure 4.10: Activities' options

CHAPTER 5
RESULTS AND DISCUSSION

The wireless smart home system based on internet of things proposed in this thesis is fully developed and tested successfully. Any smart phone running Android OS and having a WiFi or 3g/4g internet connection can use this application to control the system. The screenshots of the Android app and the pictures of the actual node circuits have been presented in the next sections.

5.1 Android Wireless Smart Home Application

This application is developed in Java programming language using Android Studio, some parts of the code is shown in Appendix 1. As illustrated in Figure 5.1 the user has to enter the server real IP and the password in Login window , first it checks for the connection to the micro web-server then validates the password, on successful connection the control panel will be displayed where the user can control the devises , in the control panel if the user wanted to change the password s/he has to enter the old password and the new password, if they didn't match the user has to try again, but if they match a data packet which contains the new password is sent to the micro web-server, if the password successfully changed a response message will be received by the app. Any other actions taken by the user will be send to the micro web-server to perform the action, then it responses the user respecting to the performed action, and when the user takes the action to logout from the app, it goes to the login window. More details is given in the next sections

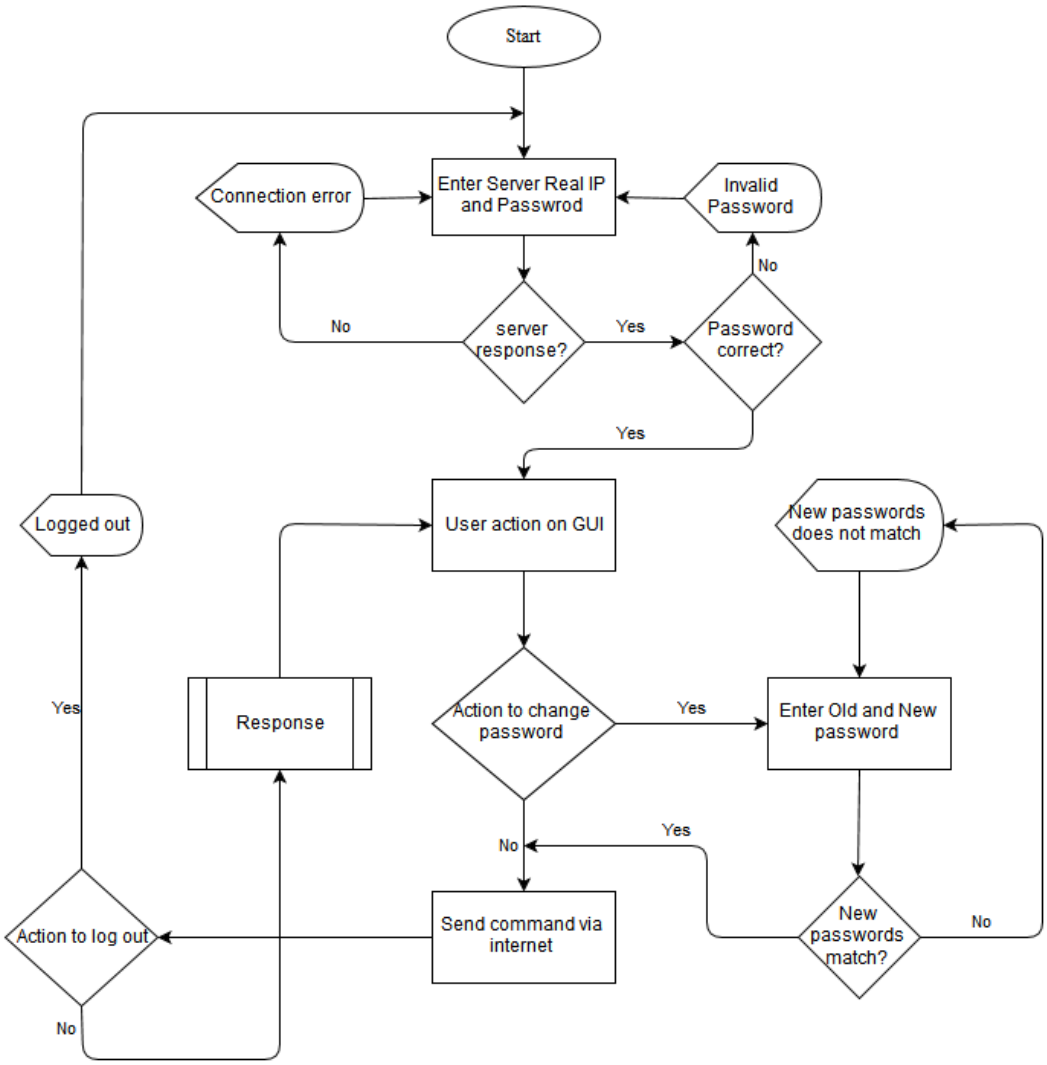
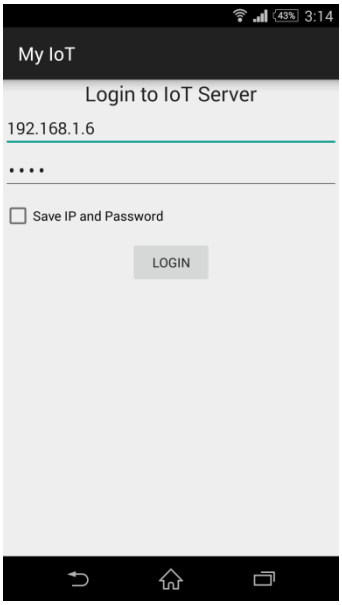


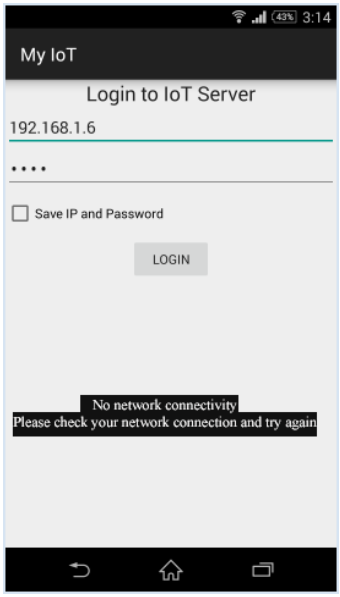
Figure 5.1: The flowchart of Android application

The application starts with the main activity which displays a login screen as shown in Figure 5.2.a, the user required to enter the IP address and the password of the server and then the application checks for the network connection or internet connectivity first, if there is no connection, a message will be displayed which indicates that there is no network connectivity as illustrated in Figure 5.2.b. If the user is connected to the network or the internet then the application validates the entered IP and the password with the server. In case an invalid IP or password were entered the app displays a message to indicate the error as shown in Figure 5.2.c,d.

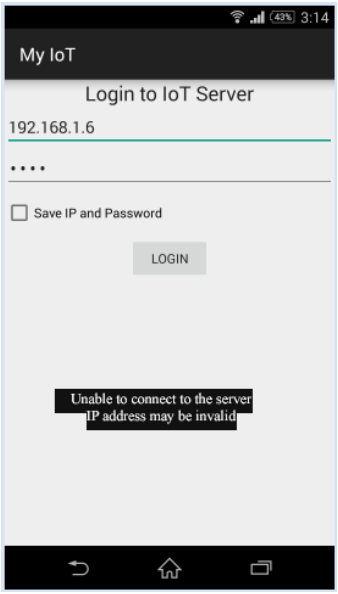
On successful validation, the control panel screen will be presented as shown in Figure 5.2.e. There is a checkbox in the main window if the user checked it , the IP and password will be saved for later use.



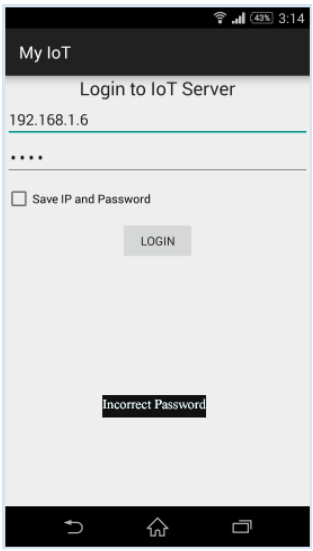
(a) Main screen



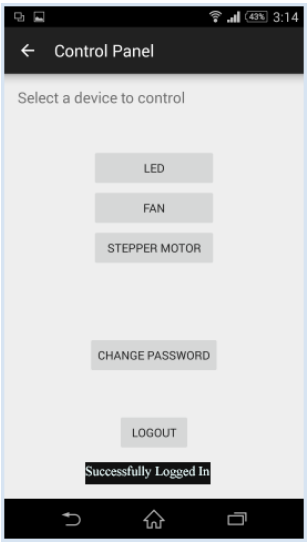
(b) When there is no network activity



(c) When invalid IP is entered



(d) When invalid password is entered network activity



(e) When successfully loged in

Figure 5.2: Screenshots of Login window with command responses

After the user inserted the correct IP and password and logged in the application successfully the control panel window will be displayed on the screen, it consists of five

command buttons as shown in Figure 5.3, the first three command buttons navigates the user to the actual end device windows, the fourth one navigates the user to change the password, the last one is used to log out from the application. And the back button located on the top left of the screen allows the user to navigate to a previously viewed window.

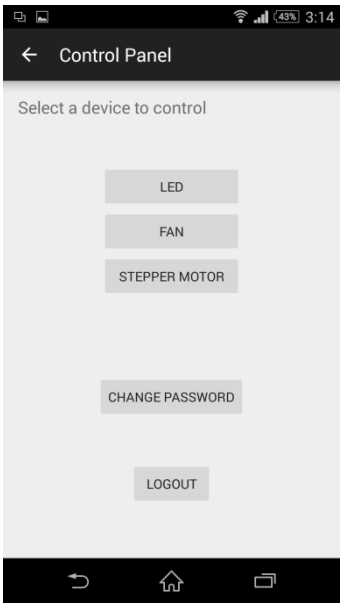
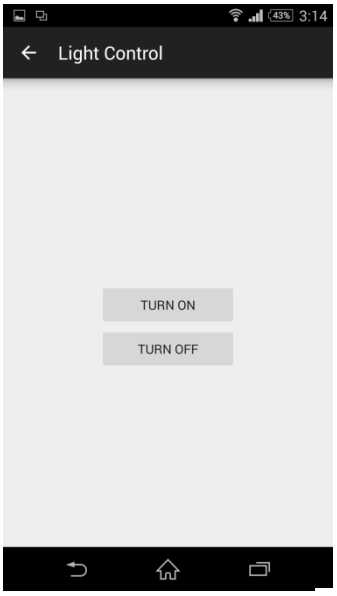


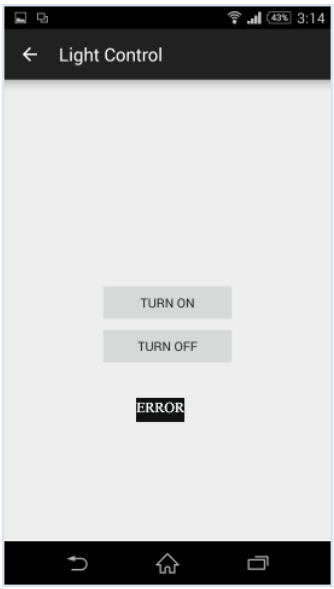
Figure 5.3 Control panel window

By clicking on the first button the LED window will be displayed as shown in Figure 5.4.a. From this window the user can turn ON/OFF the Light which represented by a LED in this prototyping project. If this node is not powered it displays a message which indicates there is an error with the command execution as shown in Figure 5.4.b, but if there is some unexpected errors (e.g. the LED node or the server side Arduino is reset), it displays a message "ERROR" which indicates that some unexpected error has occurred (and the user has to restart the application and re-login to the application to clear that error) as shown in Figure 5.4.c. Based on the performed action by the user it displays the status of the LED whether it is turned on or off as shown in Figure 5.4.d,e.



(a) LED control window

(b) When there is error with the command execution



(c) When an unexpected error accured

(d) When LED turned On

(e) When LED turned Off

Figure 5.4: LED window screenshots with command responses

The second command button leads the user to Fan control window as shown in Figure 5.5.a, from this window the user can turn On/Off the Fan and change the direction of the Fan, and a message will be displayed corresponding to the actions performed by the user to indicate the status of On and Off of the Fan as shown in Figure 5.5.b,c. And Figure 5.5.d shows the command response for changing direction.

(a) Fan control window

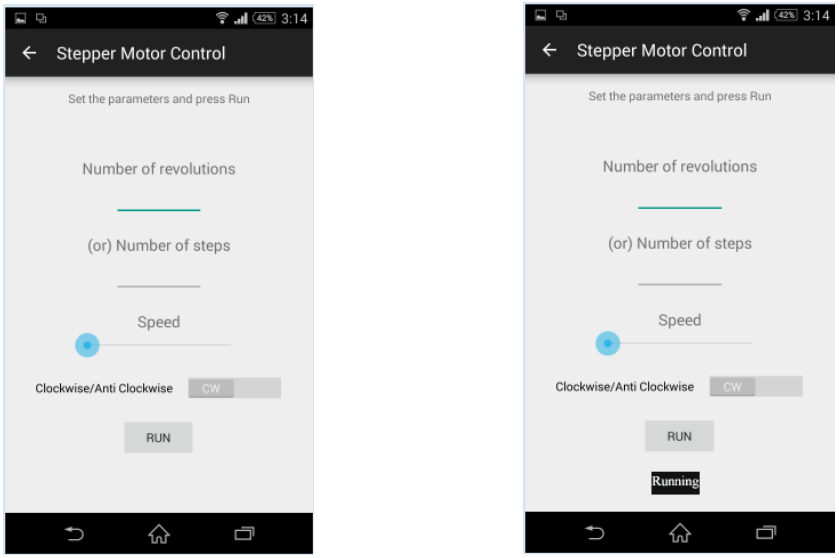
(b) When the Fan turned On

(c) When the Fan turned Off

(d) When direction changed

Figure 5.5: Screenshots of Fan control window with command responses

Figure 5.6.a. illustrates the interface of the Stepper motor control window for controlling the stepper motor, in which the user can control the number of revolutions and the number of steps. And it allows the user to change the speed of the stepper motor by dragging the slide bar to any preferred value of his/her liking. Changing the direction of the stepper motor granted as well. By pressing RUN button the stepper motor starts spinning and a message will be displayed to notify that the stepper motor is running as shown in Figure 5.6.b, and then it stops when the number of revolutions or the number of steps completed. But when the node isn't powered the user will be notified with a message that there is an error in command execution as discussed in previously.



(a) Stepper motor control window (b) When the stepper motor is running

Figure 5.6: Stepper motor window screenshots

When the user presses Change Password button in the control panel menu, the following screen will appear as shown in Figure 5.7.

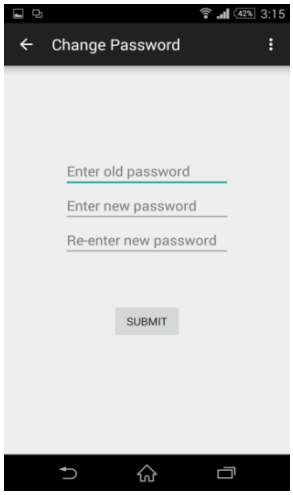
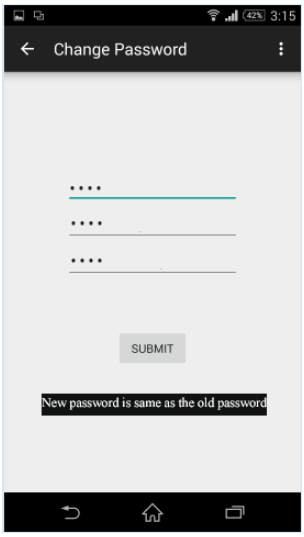


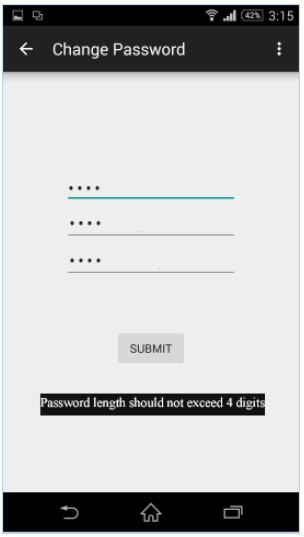
Figure 5.7: Screenshot of Change Password window

Here the user can change the password by entering the old password and then entering the new desired password and confirming it. The user has to enter the correct current password in case an invalid password was entered, the user will be notified by a message that the entered old password does not match as shown in Figure 5.8.a, and the new password cannot be same as the previous password (see figure 5.8.b) and more than four digits length as illustrated in Figure 5.8.c, and in the last text field the user should confirm the new password when a different password entered than the previous entered new password a message will be displayed which indicated that the new password doesn't match as shown in Figure 5.8.d , with passing the above mentioned restrictions the password can be changed successfully as shown in Figure 5.8.e.

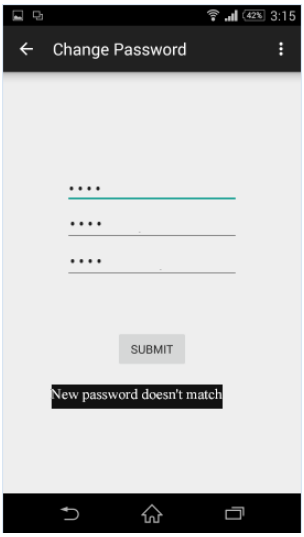
(a) When old password doesn't match



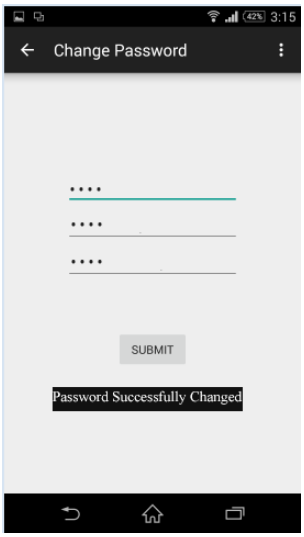
(b) When new password is same as the old



(c) When new password exceeds 4 digitd



(d) When new password doesn't match



(e) When password is successfully changed

Figure 5.8: Screenshots of command responses for changing password

5.2 Home Micro Web-Server/Central Node Design

The home micro-web server node is established as the base node with 1D (13A20040DC9179). The home server gets the data packets from the Android Wireless Smart Home application and send it to the target home appliances to control them. This design implemented with the combination of software and hardware components and the details are explained in the following topics.

5.2.1 Micro web-server hardware design results

A home server/base node designed with Arduino Uno, Ethernet shield, an Xbee module and an Xbee shield, as shown in Figure 5.9. An adapter is used to power this node which connects the Arduino microcontroller with the power source. For internet connectivity a LAN cable is used to connect the Ethernet shield to the home router to provide the internet connection.

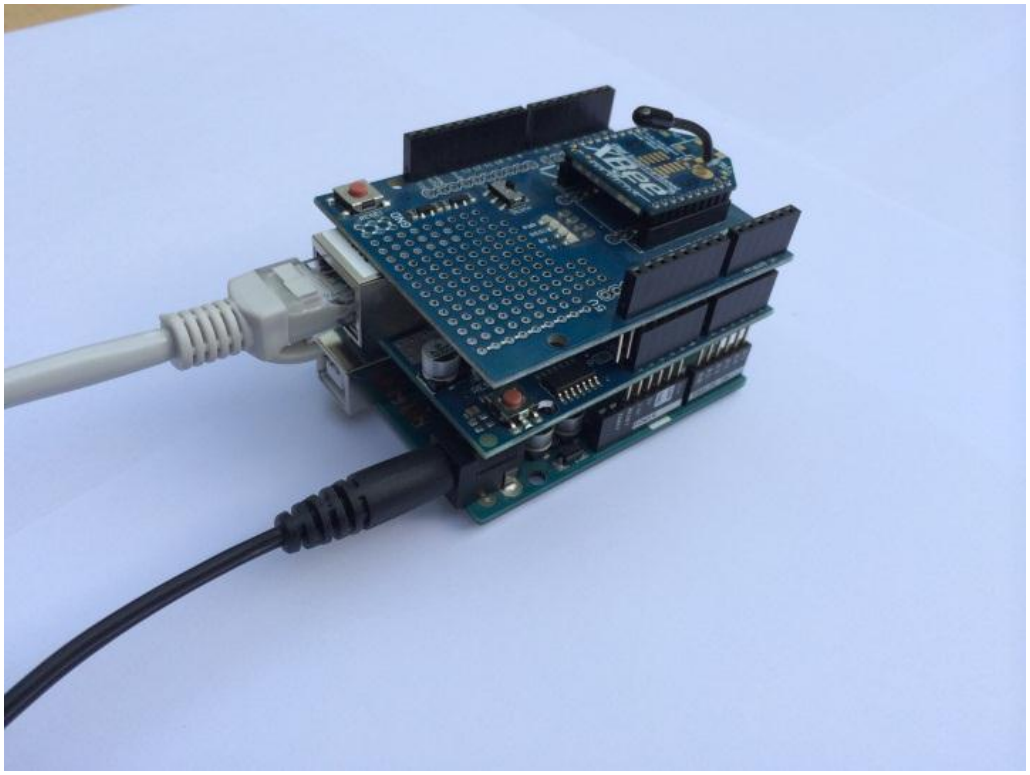


Figure 5.9: Central Node circuit

5.2.2 Micro web-server software design results

A small web server is established which will able to conduct a communication platform with Wireless Smart Home application and send the data packets to the target end device nodes via the Xbee module mounted on it. This server listens to incoming requests from the client and processes the request to perform the action. When the user clicks on the command buttons on the application, the data packets are sent to the web. Next, the data are processed by the Micro Web-Server and passes the data through its XBee.

The Arduino code sketch for the micro-web server as shown in Appendix 2 is successfully uploaded to the Arduino Uno board and working perfect.

5.3 Home Appliances/End Nodes Design

There are three remote end nodes of home appliances in this project which are LED, Fan, and a stepper motor. Each of them has a unique ID which can be seen on the back of the Xbee modules. The Xbee modules are configured in X-CTU software. The same PAN ID is set for each of the nodes, so they can only communicate with their base node/home micro-web server. The ID of each of the end nodes shown in Table 5.1.

Table 5.1: IDs of end nodes

Node Name	ID
End Node1/Lighting	0013A20040E43C0C
End Node2/ Fan	0013A200 40E43C04
End Node3/ Stepper motor	0013A20040E43BD0

5.3.1 End nodes hardware design results

The components of the nodes vary from a node to another node. The Light Node as shown in Figure 5.10 is implemented with an Xbee to receive data packets from the base node, a power source to supply power to the node, a small breadboard and an Xbee breakout board to mount the Xbee on, two LEDs are used in this node the green one to be controlled remotely and the red one just to be ensure whether the node is powered or not, and some resistors and jumper wires are used to make the connections.

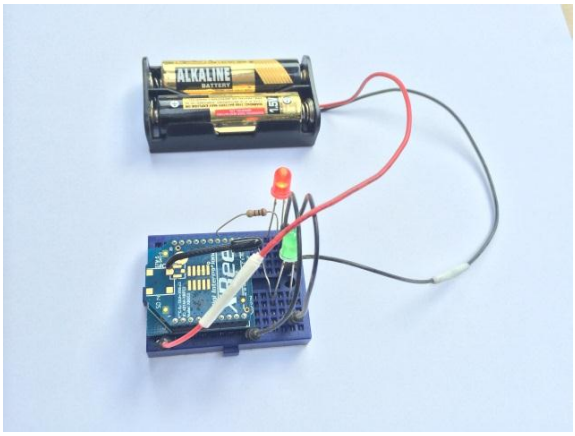


Figure 5.10: LED Node circuit

The second node which is the Fan Node as shown in Figure 5.11 is implemented with an Xbee to receive data packets from the base node, a power source to give the power to the node, a small breadboard and an Xbee breakout board to mount the Xbee on, a Fan which is being operated with a DC motor and this motor is controlled with solid-state Motor Driver IC. Because this node requires more power than the first node, so we used a 9V battery as a power source and we used a voltage regulator to prevent the Xbee from being damaged because the Xbee module can be operated by 3.3V. One red LED just to know whether the node is powered or not, and a resistor and some jumper wires.

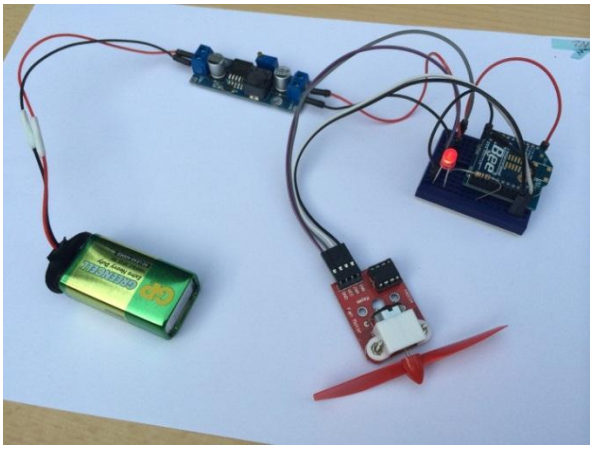


Figure 5.11: Fan Node circuit

The third end node is the stepper Motor node as presented in Figure 5.12. and it is different from the above two end nodes in components, an Arduino microcontroller is used in this node because the on-board microcontroller of the Xbee is not sufficient to drive the stepper motor. An Xbee module is used to receive data packets from the base node and it's mounted on an Xbee shield and then this shield is stacked on the Arduino microcontroller, this shield is used between them because the microcontroller operates with 9V but the Xbee module requires 3.3V. A unipolar stepper motor is used to be controlled. To operate the stepper motor a higher current is needed than the current the Arduino microcontroller provides so, a ULN2003 Stepper Motor Driver is used. Similar to the other end nodes a breadboard and some jumper wires are used to make the connections, and a red LED is used to know that the node is connected with the source of the power.

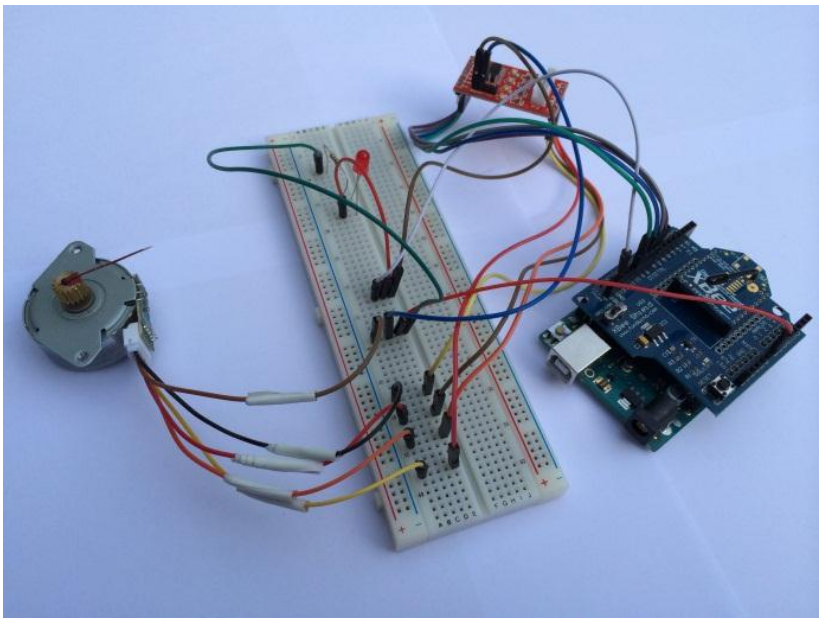


Figure 5.12: Stepper motor Node circuit

5.3.2 End nodes software design results

The Xbee module for each of the end devise nodes are configured the same. The code of Stepper motor node shown in Appendix 3 is successfully uploaded into the Arduino microcontroller. In addition, end devise nodes response perfectly to the command instruction from the user.

5.4 Difficulties Faced

The author faced numerous problems in the development of the system. Major problems were faced in the procurement of required components, as many of them weren't locally available. Furthermore, problems were faced in the implementation of the Xbee communication and configuration. The available Xbee libraries weren't completely functional so we have to test those and modify those according to our application. This process took a long time. But at the end, the system was built and tested successfully.

5.5 Comparison with Existing Wireless Technologies.

Table 5.2: WiFi, Bluetoth, and ZigBee comparison (Abinayaa and Jayan, 2014)

	WiFi	**Bluetooth**	**ZigBee**
Battery life	hours	Days	Years
Cost	Higher	Medium	Very low
Network nodes	50	8	65000+
Range	10-100 meters	10 meters	10 meters to 14+ Kilometers
Networking Topology	Star	Star	Star, Tree, Mesh
Operating Frequency	2.4 and 5 GHz	2.4 GHz	868 MHz, 915 MHz , 2.4 GHz
Ease of use	Hard	Normal	Easy
Typical network join time	Device connection requires 3-5 seconds	Device connection requires up to 10 seconds	Devices can join an existing network in under 30ms
Size	Larger	Smaller	Smallest
Typical Application	Wireless LAN connectivity, broadband Internet access	Wireless connectivity between devices such as phones, PDA, laptops, headsets.	Industrial control and monitoring, sensor networks, building automation, home control and automation

5.6 Performance of ZigBee

Timing, it takes about 30ms to transfer data. This is ok since such latency is not important in home automation.

CHAPTER 6
CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

The Internet of Things has revolutionized our way of living, under the shade of this revolution a novel architecture for an authentication wireless smart home system using Android based smartphone is designed and implemented. The smart home android application uses the internet connection and the RESTfull based web service to connect with the micro web-server. The wireless smart home app can be installed on any Android supported device to control the home appliances at anytime and from anywhere. As the microcontroller handles all the processing thus eliminating the need of using a Personal Computer as a server, thus a cost reduction was achieved in developing this system. Another important point in reducing the cost of the system is eliminating wired connections between the end devices and the central controller, so pre-existing houses do not require wiring changes. This system is highly potential for the purpose of developing remote control and portable systems

This study also has evaluated and highlighted the performance and efficiency of a ZigBee device, which help in automation and monitoring related services. In this thesis only one central node and three end nodes were used; however, there could be a huge number of nodes to be incorporated into the system. Further, systems can also be developed to monitor and measuring temperature, force, acceleration, humidity and other physical quantities.

In point of author's view; ZigBee devices are excellent in terms of remote control and portable systems within a specific range of distance. It is highlighted that ZigBee provides several additional features as compared to other standards, such as Infrared and Bluetooth.

ZigBee has the following three main advantages:

- Cost effective
- A ZigBee device can cover a range of 100m which it is long range comparing with the others.

- Minimum consumption of power, a small battery can operate a ZigBee device for months or years

Interoperability, reliability, low latency and many other features makes ZigBee devices better than the rest

The author has based his conclusion on ZigBee's use in remote controlling systems. The findings of the author in that regard are as follows:

- When compared with its own kind devices, ZigBee is not only cost-effective, but it also offers long-range connectivity and low power consumption features as well.
- Different frequency bands are provided to different areas.
- Star, Mesh, Cluster tree topologies can be supported by ZigBee which offers flexibility to the user.

6.2 Future Work

The application of IoT in the development of smart homes and home security applications is a hot topic these days. We have made our project with an aim to develop such systems with least cost. However, as it is just a prototype so there is room for further improvements in it which we couldn't do now because to time and resources limitation. Following things can be improved in future:

- The Arduino boards can be replaced with discrete controllers to reduce the cost further.
- The Xbee modules can also be replaced with their equivalent circuits.
- Currently the system connects with the Android application via a dynamic IP address, but for commercial use, a separate webserver can be used which keeps the Android application and the Arduino central controller coordinated with each other, this will increase the performance of the system.
- Instead of Arduino, other microcontroller development boards can be used.
- More devices and sensors can be connected to the system.
- Also iOS can be used, or some other operating system.

REFERENCES

Abinayaa, V., & Jayan, A. (2014). Case study on comparison of wireless technologies in industrial applications. *International Journal of Scientific and Research Publication*, *4*(2), 619-622.

Alkar, A. Z. & Buhur, U. (2005). An internet based wireless home automation system for multifunctional devices. *IEEE Transactions on Consumer Electronics*, *51*(5), 1169-1174.

Alicia, G. (2010). *New media art, design, and the Arduino microcontroller: A malleable pool* (Master's Thesis). Pratt Institute, New York, USA.

Arduino. (2015a). Arduino Ethernet Shield. Retrieved June 15, 2015 from http://www.arduino.cc/en/Main/ArduinoEthernetShield

Arduino. (2015b). Arduino Uno. Retrieved June 5, 2015 from http://www.arduino.cc/en/Main/ArduinoBoardUno

Arduino. (2015c). Why Arduino?. Retrieved June 4, 2015 from https://www.arduino.cc/en/Guide/Introduction

Arthur E., & Cote, P.E. (2003). Operation of fire detection, National Fire Protection, Inc. National Fire Protection Association, Quincy, MA.

Ashton, K. (2009). That "internet of things" thing. *RFiD Journal*. Retrieved April 22, 2015 from http://www.rfidjournal.com/articles/view?4986

Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. *Computer Networks*, *54*(15), 2787-2805.

Auto Webbed. (2015). Home automation turns your house into a smart home. Retrieved April 30, 2015 from http://www.autowebbed.com/services/home-automation/

Blum, j. (2013). Exploring Arduino: Tools and Techniques for Engineering Wizardry. Indianapolis, IN: John Wiley & Sons.

Buttay, C. (2006a). H-Bridge. Retrieved May 27, 2015 from http://fabacademy.org/archives/2013/students/saguar.ruben/schedules/output%20devices%20(Apr%2017).html

Buttay, C. (2006b). H-bridge theory of operation. Retrieved May 29, 2015 from http://fabacademy.org/archives/2013/students/saguar.ruben/schedules/output%20devices%20(Apr%2017).html

Chiu-Chiao, C., Ching Yuan, H., Shiau-Chin, W., & Cheng-Min, L. (2011). Bluetooth-based Android interactive applications for smart living. *In Proceedings of the 2nd International Conference on Innovations in Bio-inspired Computing and Applications* (pp. 309-312). Washington, DC: IEEE Computer Society.

Cisco. (2015). Internet of Things (IoT) Retrieved October 20, 2015 from http://www.cisco.com/web/solutions/trends/iot/overview.html

Digi International. (2015). XBee / XBee-PRO ZigBee RF Modules. Retrieved May 20, 2015 from http://ftp1.digi.com/support/documentation/90000976_W.pdf

ElShafee, A., & Hamed, K. (2012). Design and implementation of a WiFi based home automation system. *World Academy of Science, Engineering and Technology, 68,* 2177-2180.

Gartner, Inc. (2011). Gartner's 2011 Hype Cycle Special Report Evaluates the Maturity of 1,900 Technologies Retrieved April 27, 2015 from http://www.gartner.com/ newsroom/id/1763814

Giusto, D., Iera, A., Morabito, G., & Atzori, L. (Eds.). (2010). The Internet of Things. New York, NY: Springer-Verlag.

Gunner, B. (2013). MQTT: Enabling the Internet of Things. Retrieved April 22, 2015 from https://www.ibm.com/developerworks/community/blogs/c565c720-fe84-4f63-873f-607d87787327/entry/tc_overview?lang=ja

Herman, S. (2014). Electric Motor Control. Boston,MA: Cengage Learning.

Hilton, S. (2012, January 14). Progression from M2M to the Internet of Things: an introductory blog. Retrieved April 29, 2015 from http://blog.bosch-si.com/progression-from-m2m-to-internet-of-things-an-introductory-blog/

Huang, H. (2005). PIC Microcontroller: An Introduction to Software and Hardware Interfacing. Monkato, MN: Delmar Cengage Learning.

Jones, D. W. (2014). What Is A Stepper Motor?. Retrieved June 12, 2015 from http://www.amci.com/tutorials/tutorials-what-is-stepper-motor-5.asp

Kamarudin, M. R., M. A. F., & Yusof, M. (2012). Low cost smart home automation via Microsoft speech recognition. *International Journal of Engineering & Computer Science, 13,* 6-11.

Kamilaris, A., Trifa, V., & Pitsillides, A., (2011). Home Web: An application framework for Web-based smart homes. *In Proceedings* of the *18th International Conference* on *Telecommunications* (pp. 134-139). Ayia Napa: University of Cyprus.

L9110 Datasheet. (n.d.). Retrieved June 2, 2015 from http://www.electrodragon.com/w/File:Datasheet-l9110.pdf

Liang, N.-S., Fu, L.-C. & Wu, C.-L. (2002). An integrated, flexible, and Internet-based control architecture for home automation system in the Internet era. *In Proceedings of IEEE International Conference on Robotics and Automation* (pp. 1101-1106). Washington, DC: IEEE Robotics and Automation Society.

Loop technology dorset. (n.d.). Motion Control Stepper Motors. Retrieved June 8, 2015 from http://www.looptechnology.com/motion-control-stepper-motors.asp #.VXNFmt Kqqko

Meyer, R. (2008). Basic Motor Control and Relay Logic. Retrieved May 26, 2015 from http://www.robmeyerproductions.com/bows.html

Mulder, B. (2001). How does a brushless electric motor work? Retrieved May 22, 2015 from http://electronics.howstuffworks.com/brushless-motor.htm

Mulder, B. (2005). Electric Motors. Retrieved May 23, 2015 from http://www.southernsoaringclub.org.za/a-BM-motors-1.html

Piyare, R., & Tazil, M. (2011). Bluetooth based home automation system using cell phone. *In Proceedings of IEEE 15th International Symposium on Consumer Electronics* (pp. 192-195). Singapore: IEEE Consumer Electronics Society.

Potts, J., & Sukittanon, S., (2012). Exploiting Bluetooth on Android mobile devices for home security application. *In Proceedings of IEEE Southeastcon* Conference (pp. 1-4). Orlando, FL: IEEE Region 3.

Prakash, M. (2000). Pulse Width Modulation. Retrieved May 23, 2015 from http://fab.cba.mit.edu/classes/MIT/961.04/topics/pwm.pdf

Proto Stack. (2011, Jun 25). ATmega168A Pulse Width Modulation – PWM. Retrieved May 25, 2015 from http://www.protostack.com/blog/2011/06/atmega168a-pulse-width-modulation-pwm/

Rajabzadeh, A., Manashty, A. R., & Jahromi, Z. F. (2010). A Mobile application for smart house remote control system, *World Academy of Science, Engineering and Technology, 62*.

Ramlee, R. A., Leong, M. H., Singh, R.S.S., Ismail, M.M., Othman, M. A., & Sulaiman, H. A. (2013). Bluetooth remote home automation system using Android application. *The International Journal of Engineering & Science*. *2*, 149-153.

Schoenberger, C. R. (2002). The internet of things. Retrieved April 25, 2015 from http://www.forbes.com/global/2002/0318/092.html

Shahriyar, R., Hoque, E., Sohan, S., Naim, I., Akbar, M. M., & Khan, M. K. (2008). Remote controlling of home appliances using mobile telephony. *International Journal of Smart Home, 2,* 37-54.

Sharma, U. & Reddy, S. R. N. (2012). Design of home/office automation using wireless sensor network. *International Journal of Computer Applications, 43*, 53-60.

Shui Kumar. (2014). Ubiquitous smart home system using Android application. *International Journal of Computer Networks & Communication*, *6*(1), 534-729.

ULN2003a. (2013, February 1). Retrieved June 13, 2015 from http://www.ti.com/lit/ds/symlink/uln2003a.pdf

Williams, A. (2002). Microcontroller Projects Using the Basic Stamp. Lawrence, Kan: CMP Books.

Yan, M., & Shi, H. (2013). Smart living using Bluetooth-based Android smartphone. *International Journal of Wireless & Mobile Networks, 5*, 65-72.

ZigBee Alliance. (2015). ZigBee innovation connects simple and high-tech devices for consumers and businesses. Retrieved October 20, 2015 from http://www.zigbee.org/zigbee-for-developers/network-specifications/zigbeeip/

Zureks. (2007). PWM, 3-level. Retrieved May 25, 2015 from https://commons.wikimedia.org/wiki/File:PWM,_3-level.svg

APENDICES

Appendix 1

Android Java Code

A snippet code of the main window when user clicks on Login button

```
/** Called when the user clicks the Login button */
public void sendLogin(View view)
{
    EditText ip = (EditText) findViewById(R.id.input_ip);
    String ip_address = ip.getText().toString();
    EditText pass = (EditText) findViewById(R.id.input_password);
    String password = pass.getText().toString();
    String login_address = "http://" + ip_address + "/funLogin?params=" +
password;

    intent = new Intent(this, ControlPanel.class);
    intent.putExtra(EXTRA_IP, ip_address);

    //Check for internet connectivity
    ConnectivityManager connMgr = (ConnectivityManager)
            getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
    if (networkInfo != null && networkInfo.isConnected())
    {
        // fetch data
        new DownloadWebpageTask().execute(login_address);
    } else
    {
        Context context = getApplicationContext();
        CharSequence text = "No network connectivity. Please check your network
connection and try again.";
        int duration = Toast.LENGTH_LONG;
        Toast toast = Toast.makeText(context, text, duration);
        toast.show();
    }
}
```

A snippet code for the control panel window

```
public class ControlPanel extends ActionBarActivity
{
    public final static String EXTRA_IP = "neu.myiot.IP";
    private String ip;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        // Get the message from the intent
        Intent intent = getIntent();
        ip = intent.getStringExtra(MainActivity.EXTRA_IP);
        setContentView(R.layout.activity_control_panel);
    }

    public void controlLed(View view)
    {
        Intent intent = new Intent(this, LedControl.class);
        intent.putExtra(EXTRA_IP, ip);
```

```
        startActivity(intent);
    }

    public void controlFan(View view)
    {
        Intent intent = new Intent(this, FanControl.class);
        intent.putExtra(EXTRA_IP, ip);
        startActivity(intent);
    }

    public void controlStepper(View view)
    {
        Intent intent = new Intent(this, StepperControl.class);
        intent.putExtra(EXTRA_IP, ip);
        startActivity(intent);
    }

    public void changePassword(View view)
    {
        Intent intent = new Intent(this, ChangePassword.class);
        intent.putExtra(EXTRA_IP, ip);
        startActivity(intent);
    }
```

A snippet code for Turning the LED On or Off:

```
/** Called when the user clicks the LED On button */
public void turnLedOn(View view)
{
    cmd_off=false;
    cmd_on=true;
    String command_address = "http://" + ip + "/funIOT?params=11";

    //Check for internet connectivity
    ConnectivityManager connMgr = (ConnectivityManager)
            getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
    if (networkInfo != null && networkInfo.isConnected())
    {
        // fetch data
        new DownloadWebpageTask().execute(command_address);
    } else
    {
        Context context = getApplicationContext();
        CharSequence text = "No network connectivity. Please check your network
connection and try again.";
        int duration = Toast.LENGTH_LONG;
        Toast toast = Toast.makeText(context, text, duration);
        toast.show();
    }
}

/** Called when the user clicks the LED Off button */
public void turnLedOff(View view)
{
    cmd_off=true;
    cmd_on=false;
    String command_address = "http://" + ip + "/funIOT?params=10";

    //Check for internet connectivity
    ConnectivityManager connMgr = (ConnectivityManager)
            getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
    if (networkInfo != null && networkInfo.isConnected())
```

```
    {
        // fetch data
        new DownloadWebpageTask().execute(command_address);
    } else
    {
        Context context = getApplicationContext();
        CharSequence text = "No network connectivity. Please check your network
connection and try again.";
        int duration = Toast.LENGTH_LONG;
        Toast toast = Toast.makeText(context, text, duration);
        toast.show();
    }
}
```

A part of code for password changing option

```
** Called when the user wants to change the password */
public void submitPassword(View view)
{
    SharedPreferences sharedPref=
getSharedPreferences(getString(R.string.preference_file_key),Context.MODE_PRIVATE
);
    String saved_old_pwd = sharedPref.getString("saved_pwd", "");

    EditText temp1 = (EditText) findViewById(R.id.inputOldPassword);
    String entered_old_pwd = temp1.getText().toString();
    EditText temp2 = (EditText) findViewById(R.id.inputNewPassword);
    String entered_new_pwd = temp2.getText().toString();
    EditText temp3 = (EditText) findViewById(R.id.inputRePassword);
    String entered_re_pwd = temp3.getText().toString();

    if (!(entered_old_pwd.equals(saved_old_pwd)))
    {
        Context context = getApplicationContext();
        CharSequence text = "Old password doesn't match";
        int duration = Toast.LENGTH_LONG;
        Toast toast = Toast.makeText(context, text, duration);
        toast.show();
    }
    else if ((entered_old_pwd.equals(entered_new_pwd)))
    {
        Context context = getApplicationContext();
        CharSequence text = "New password is same as the old password";
        int duration = Toast.LENGTH_LONG;
        Toast toast = Toast.makeText(context, text, duration);
        toast.show();
    }
    else if(entered_new_pwd.length()>4)
    {
        Context context = getApplicationContext();
        CharSequence text = "Password length should not exceed 4 digits";
        int duration = Toast.LENGTH_LONG;
        Toast toast = Toast.makeText(context, text, duration);
        toast.show();
    }
    else if (!(entered_re_pwd.equals(entered_new_pwd)))
    {
        Context context = getApplicationContext();
        CharSequence text = "New password doesn't match";
        int duration = Toast.LENGTH_LONG;
        Toast toast = Toast.makeText(context, text, duration);
        toast.show();
    }
    else
```

```
    {
        String command_address = "http://" + ip + "/funChangePass?params=" +
entered_new_pwd;

        //Check for internet connectivity
        ConnectivityManager connMgr = (ConnectivityManager)
                getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
        if (networkInfo != null && networkInfo.isConnected()) {
            // fetch data
            new DownloadWebpageTask().execute(command_address);
        } else {
            Context context = getApplicationContext();
            CharSequence text = "No network connectivity. Please check your
network connection and try again.";
            int duration = Toast.LENGTH_LONG;
            Toast toast = Toast.makeText(context, text, duration);
            toast.show();
        }
    }
}
```

Appendix 2

Arduino Ethernet Micro Web-Server Code

The following are some parts of central node/micro web-server code

```
#include <SPI.h>
#include <Ethernet.h>
#include <aREST.h>
#include <XBee.h>


//variables for Server
EthernetServer server(80);    //server at port 80
aREST rest = aREST();      //rest api instance
```

```
//variables for Xbee
XBee xbee = XBee();
XBeeAddress64  node1 = XBeeAddress64(0x0013A200, 0x40E43BD0);  //the led node
XBeeAddress64  node2 = XBeeAddress64(0x0013A200, 0x40E43C04);  //the fan node
XBeeAddress64  node3 = XBeeAddress64(0x0013A200, 0x40E43C0C);    //the stepper
motor node
```

```
void setup()
{
  Serial.begin(9600);

  // Function to be exposed
  rest.function("funLogin",login);
  rest.function("funLogout",logout);
  rest.function("funChangePass",change_password);
  rest.function("funIOT",IOT);
```

```
bool send_cmd()
{
  xbee.send(remoteAtRequest);
  if(xbee.readPacket(1000))
  {
    if (xbee.getResponse().getApiId() == REMOTE_AT_COMMAND_RESPONSE)
    {
      xbee.getResponse().getRemoteAtCommandResponse(remoteAtResponse);
      if (remoteAtResponse.isOk())
      {
        ans=1;
      }
```

```
      else
      {
        ans=0;
      }
    }
  }
  else
  {
    ans=0;
  }
  return (ans);
}

int led(int CMD)
{
  if (authenticated == false)
  {
    return 0;
  }
  if (CMD == 1)    //turn on the led
  {
    cmd[0] = 'D';
    cmd[1] = '0';
    cmdValue[0] = 0x5;
    remoteAtRequest.setRemoteAddress64(node1);
    remoteAtRequest.setCommand(cmd);
    remoteAtRequest.setCommandValue(cmdValue);
    remoteAtRequest.setCommandValueLength(sizeof(cmdValue));
    return (send_cmd());
  }
  if (CMD == 0)
  {
    cmd[0] = 'D';
    cmd[1] = '0';
    cmdValue[0] = 0x4;
    remoteAtRequest.setRemoteAddress64(node1);
    remoteAtRequest.setCommand(cmd);
    remoteAtRequest.setCommandValue(cmdValue);
    remoteAtRequest.setCommandValueLength(sizeof(cmdValue));
    return (send_cmd());
  }
}
```

```
int fan(int CMD)
{
  if (authenticated == false)
  {
```

```
  return 0;
 }
 if (CMD == 1)    //turn on the fan
 {
  cmd[0] = 'D';
  if(_fan_dir)
   cmd[1] = '1';
  else
   cmd[1] = '0';
  cmdValue[0] = 0x5;
  remoteAtRequest.setRemoteAddress64(node2);
  remoteAtRequest.setCommand(cmd);
  remoteAtRequest.setCommandValue(cmdValue);
  remoteAtRequest.setCommandValueLength(sizeof(cmdValue));
  return(send_cmd());
 }
 if (CMD == 0)
 {
  cmd[0] = 'D';
  cmd[1] = '0';
  cmdValue[0] = 0x4;
  remoteAtRequest.setRemoteAddress64(node2);
  remoteAtRequest.setCommand(cmd);
  remoteAtRequest.setCommandValue(cmdValue);
  remoteAtRequest.setCommandValueLength(sizeof(cmdValue));
  if(send_cmd())
  {
   cmd[0] = 'D';
   cmd[1] = '1';
   cmdValue[0] = 0x4;
   remoteAtRequest.setRemoteAddress64(node2);
   remoteAtRequest.setCommand(cmd);
   remoteAtRequest.setCommandValue(cmdValue);
   remoteAtRequest.setCommandValueLength(sizeof(cmdValue));
   return(send_cmd());
  }
  else
   return 0;
 }
}
```

```
int fan_dir(int CMD)
{
 int tt=0;
 if (authenticated == false)
 {
  return 0;
```

```
}
if (CMD == 1)    //change the direction
{
 cmd[0] = 'D';
 cmd[1] = '0';
 cmdValue[0] = 0x4;
 remoteAtRequest.setRemoteAddress64(node2);
 remoteAtRequest.setCommand(cmd);
 remoteAtRequest.setCommandValue(cmdValue);
 remoteAtRequest.setCommandValueLength(sizeof(cmdValue));
 if(send_cmd())
 {
  cmd[0] = 'D';
  cmd[1] = '1';
  cmdValue[0] = 0x5;
  remoteAtRequest.setRemoteAddress64(node2);
  remoteAtRequest.setCommand(cmd);
  remoteAtRequest.setCommandValue(cmdValue);
  remoteAtRequest.setCommandValueLength(sizeof(cmdValue));
  return (send_cmd());
 }
 else
  return 0;
}
if (CMD == 0)    //change to anti-clockwise
{
 cmd[0] = 'D';
 cmd[1] = '0';
 cmdValue[0] = 0x4;
 remoteAtRequest.setRemoteAddress64(node2);
 remoteAtRequest.setCommand(cmd);
 remoteAtRequest.setCommandValue(cmdValue);
 remoteAtRequest.setCommandValueLength(sizeof(cmdValue));
 if(send_cmd())
 {
  cmd[0] = 'D';
  cmd[1] = '1';
  cmdValue[0] = 0x5;
  remoteAtRequest.setRemoteAddress64(node2);
  remoteAtRequest.setCommand(cmd);
  remoteAtRequest.setCommandValue(cmdValue);
  remoteAtRequest.setCommandValueLength(sizeof(cmdValue));
  return (send_cmd());
 }
 else
  return 0;
}
```

```
}
```

```
 xbee.send(zbTx);
 if(xbee.readPacket(1000))
 {
  if (xbee.getResponse().getApiId() == ZB_TX_STATUS_RESPONSE)
  {
   xbee.getResponse().getZBTxStatusResponse(txStatus);
   if (txStatus.getDeliveryStatus() == SUCCESS)
   {
    ans=1;
   }
   else
   {
    ans=0;
   }
  }
 }
 else
 {
  ans=0;
 }
 return (ans);
}
```

```
int login(String command)
{
  int pass = command.toInt();

 if (pass == PASSWORD)
 {
  authenticated = true;
  return 1;
 }
 else
 {
  authenticated = false;
  return 0;
 }}
```

```
int logout(String command)
{
 int pass = command.toInt();

 authenticated = false;
```

```
  return 1;
}
```

```
int change_password(String command)
{
  int pass = command.toInt();
  if (authenticated == false)
  {
    return 0;
  }
  else
  {
    PASSWORD = pass;
    return (1);
  }}
```

```
int IOT(String command)
{
  int parameter = 0;
  int ret = 0;
  int d,x=0;
  String t1,t2,t3;
  String temp2 = command.substring(0,1);
  String temp = command.substring(1);
  int CMD = temp2.toInt();
  if (CMD != 5)
  {
    parameter = temp.toInt();
  }
  switch(CMD)
  {
    case (1):    //LED ON/OFF control
      ret = led(parameter);
      break;
    case (2):    //FAN ON/OFF control
      ret = fan(parameter);
      break;
    case (3):    //FAN direction control
//      ret = fan_dir(parameter);
    {
      ret = fan(0);
      _fan_dir = parameter;
      delay(1000);
//      ret = fan(1);
      break;
    }
```

Appendix 3

Ardunio Stepper Motor Node code

The following is snippet code uploaded to Arduino of Node3 to control the stepper motor

```
void loop()
{
  if(Serial.available()>0)
  {
    while(Serial.available()>0)
    {
      steps = Serial.parseInt();
      in = Serial.read();
    }
    if(in=='p')
    {
      digitalWrite(13, HIGH);
      Direction=0;    //+ve direction
    }
    if(in=='n')
    {
      digitalWrite(13, LOW);
      Direction=1;    //-ve direction
    }
    if(in=='s')
      Speed=steps;
    if(in=='r')
    {
      myStepper.setSpeed(Speed);
      if(Direction == 0)
        myStepper.step(steps);
      if(Direction == 1)
        myStepper.step((-1*steps));
    }
  }
  delay(1);
}
```