IP Based Air Conditioner Universal Control Unit

By

Syed Maaz Shahid

Wardah Sarmad

Aoun Ali Naqvi

Submitted to the Faculty of Electrical Engineering

National University of Sciences and Technology, Islamabad in partial fulfillment

for the requirements of a B.E. Degree in Telecom Engineering

JUNE 2015

ABSTRACT

The aim of the project is to design a system which enables a user to control an air conditioner remotely using an Android based device and/or a PC, over an IP network. This report documents the details of the tasks done that have led towards the completion of the design objectives. It contains details pertaining to the system architecture, and focuses on some of the components that used as part of our hardware design. Detailed designs of the hardware layout have been included. The report also documents progress made in the application development for both the Android Operating System (using Java and XML) and the Windows OS (using Visual Basic). Final layouts for the applications have been included, and the applications' functionality has been explained in detail. The report also contains testing approaches, along with details of the results that were obtained during each phase.

CERTIFICATE OF CORRECTNESS AND APPROVAL

It is certified that the work contained in this thesis title "IP Based Air Conditioner Universal Control Unit", carried out by Syed Maaz Shahid, Aoun Ali Naqvi and Wardah Sarmad under the supervision of Lt. Col. Dr. Adil Masood Saddique in partial fulfillment of Degree of Bachelor of Telecommunication Engineering, is correct and approved.

Approved by

_____

LT. COL. DR. Adil Masood Saddique

PROJECT SUPERVISOR

Military College Of Signals, NUST

DECLARATION

No portion of the work presented in this dissertation has been submitted in support of another award or qualification either at this institution or elsewhere.

ACKNOWLEDGMENTS

There is no success without the will of ALLAH Almighty. We are grateful to ALLAH, who has given us guidance, strength and enabled us to accomplish this task. Whatever we have achieved, we owe it to Him, in totality. We are also grateful to our parents and family and well-wishers for their admirable support and their critical reviews. We would thank our supervisor Lt. Col. Dr. Adil Masood for his guidance and support throughout the year. We would specially like to thank Lecturer Intisar Rizwan e Haque for his commendable guidance through each and every phase of our project. We would also like to thank Mughees Sarwar for helping us when we were critically stuck. The team would like to pay gratitude to the team leader for bearing us the whole year and for billion repeated document edits.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

SR No	Symbols	Abbreviation
1	LED	Light Emitting Diode
2	AC	Air Conditioner
3	IP	Internet Protocol
4	PC	Personal Computer
5	IR	Infrared
6	Wi-Fi	Wireless Fidelity
7	HVAC	Heating, Ventilation and Air Conditioning
8	IC	Integrated Circuits
9	TCP/IP	Transmission Control Protocol/Internet Protocol
10	DHCP	Dynamic Host Configuration Protocol
11	GUI	Graphical User Interface
12	GSM	Global System for Mobile Communications
13	SMS	Short Message Service
14	VB	Visual Basic
15	XML	Extensible Markup Language
16	IDE	Integrated Development Environment
17	ADT	Android Development Tools
18	UI	User Interface
19	API	Application Program Interface
20	SDK	Software Development Kit
21	UDP	User Datagram Protocol
22	PDIP	Plastic Dual-In-Line Package
23	RAM	Random Access Memory
24	I/O	Input/output
25	EEPROM	Electrically Erasable Programmable Read-Only Memory

26	UART	Universal Asynchronous Receiver/Transmitter
27	USB	Universal Serial Bus
28	AC to DC	Alternating Current to Direct Current
29	CPU	Central Processing Unit
30	SPI	Service Provider Interface
31	PCB	Printed Circuit Board
32	LAN	Local Area Network
33	TCP	Transmission Control Protocol

CHAPTER ONE

1 INTRODUCTION

1.1 OVERVIEW

The goal of our project is to build a system consisting of an Android device "The Internet of Things", which, when connected to the network via Wi-Fi, will be capable of controlling Air conditioners. This appliance will be assigned unique IP address and the user will be able to switch them On/Off and control its parameter through a control packet sent over the IP network. An android and PC application will be developed in order to provide the user an easy platform, where, by simply touching the icon, the user will be able to switch the appliance in accordance with the user's requirements. At the other end, controller hardware will be interfaced with the appliances, which, after receiving a control packet, will perform the desired operations. This will allow the user to roam freely and access the devices from anywhere as long as the user is connected to the network.

1.2 PROBLEM STATEMENT

Today, many electrical appliances can be controlled remotely, such as fans, LED lights and cameras. Technology has advanced to a stage where appliances can be connected to the internet and operated using a single device, which has given rise to the term 'Internet of Things'. As with other home appliances, there is a need to control air conditioners remotely as well. An AC remote can operate an AC unit within a room as it needs a line of sight. We cannot obtain any information about

the status of a device if we are outside the building. Our goal is to build a universal control unit that can control any air conditioner via an IP based network, using a smartphone and/or a PC. We aim to provide a system where all the functions of the air conditioner can be controlled via a Local Area Network. Our goal is also to design and implement a centralized monitoring system and a power management mechanism, which will further enhance the usefulness and functionality of the system that we aim to create.

1.3 APPROACH

1. Design of universal learning IR protocol algorithm, which will decode the IR protocols and learn it.
2. Hardware for learning, storing IR protocols and transmitting of IR code to control the functionality of any AC is designed
3. The hardware works in two modes: Learning and transmitting mode.
4. Microcontroller will be interfaced with the control hardware
5. To give the controlling unit internet connectivity, the hardware is to be interfaced with Wi-Fi module.
6. PC and Android application are developed.
7. The microcontroller will receive the data from applications and process the data.
8. Based on it, the controller hardware will generate IR code to operate an Air Conditioner.

1.4 SCOPE

Air conditioner is controlled through internet using smart phones which can be used in any buildings such as offices, universities etc. The controlling system will be placed near the air conditioner, which will work as a normal remote, to control the air conditioner parameters such as its state (on/off) and temperature remotely. This system will be operated with android application or with a PC application. User will be able to control the air conditioner with his smart phone or PCs anywhere. Centralized controlling and monitoring cell can be deployed in any organization. This can be widely used in industry.

1.5 OBJECTIVES

1.5.1 Learning objectives:

1. Learning of IR protocols
2. Programming of microcontrollers
3. Android programming
4. Visual Basic programming
5. Circuit designing

1.5.2 Project Objectives:

1. Design a Control Module with following specifications:
   - **1.1.** Able to read IP packet and its contents
   - **1.2.** Perform switching of AC
2. AC on/off and change its parameter

2.1. Read status information from devices and send data back to control application i.e. Centralized monitoring of air conditioners within the building.

3. Design of front end application on Android device

4. Design of front end application on PC

CHAPTER TWO

2 LITERATURE REVIEW

2.1 Previous projects related to this concept in MCS (NUST)

1) **Automated home system using DAS with GSM support**: The aim of this project was to control, manage and monitor all the electrical devices either digital or analog remotely by using the GSM modem and through PC using serial port. GSM mechanism of communication between the microcontroller and the client via Short Message Service (SMS). User will be able to control many electrical gadgets by sending SMS (standard format) to the control system based on microcontroller and validated.
2) **Android based home automation system:** The aim of this project was to develop a home automation system, which is mainly integration of hardware devices control module and Android application. The hardware control part was done using microcontrollers, where the front end application was based on Android. The Android based application was used for performing several applications from mobile set. The Android application included image/video acquisition from camera, capable to ON and OFF certain switches, control fan speeds, brightness of bulbs and even manual control over all appliances too.
3) **IP network based energy management and control system:** The user would be able to turn On/Off light bulb, turn On/Off fan with triple speed control using android application. The devices would also be able to

communicate with the system and send their status to the system for monitoring and management of energy consumption of devices.[1]

2.2 In Pakistan

Taking a look at what Pakistan has in store regarding this technology, there is a promising product launch "Future Energy". It is a complete Home Automation System in Pakistan. LightwaveRF is a radio system (like Bluetooth) that lets you operate light switches and power sockets with your phone where ever you are. The LightwaveRF Connect series is the beginning of a new revolution in home automation technology. LighwaveRF Lighting with built-in wireless communications protocol allows switches and dimmers to be installed into standard 25/35mm back boxes, in the same way as any standard switch is wired. All LightwaveRF devices can be operated by any of the range of remote controllers, the LightwaveRF Master wall switch or even a smartphone, iPad or iPod touch, from anywhere in the world.

2.3 Projects being carried out at the commercial level in rest of the world

One of the most foremost projects being carried out on the commercial level in the field of IP based modules is AT&T's Digital Life. It is an IP-Based Home Automation And Security System With 24/7 Monitoring Centers: AT&T has just announced a new security and home-automation system called Digital Life, which will be an IP-based platform that allows users to monitor and detect activity throughout their house remotely, and "take action" (as AT&T put it) on devices

like PCs, tablets, and smartphones. Functions include access to automation, energy and water controls, and security systems. AT&T will thus be introducing a new branch called the Digital Life group, which will work in "AT&T owned-and-operated 24/7 security monitoring centers."

Now, since IP network is an evergreen technology, it only makes sense in formulating a system where the back end technology remains constant for a very long time and we keep on updating the front end operating systems as required either by the innovation in technology or the requirements of the user. Therefore, our project provides the first stepping stone in achieving this goal in our country.

CHAPTER THREE

3 DESIGN AND DEVELOPMENT

3.1 SYSTEM ARCHITECTURE

The control system consists of IR receiver, IR transmitter, microcontroller, serial port and wifi router. The system will be controlled by smart phones with android application or PC application through internet. When instruction is selected from the android or PC application, this instruction will go to the controlling device through IP based network. Wi-Fi sends the receiving instruction to the microcontroller through the serial port which will decode it and then a code is send to the IR transmitter. It will operate the air conditioner just like a normal remote of an AC.

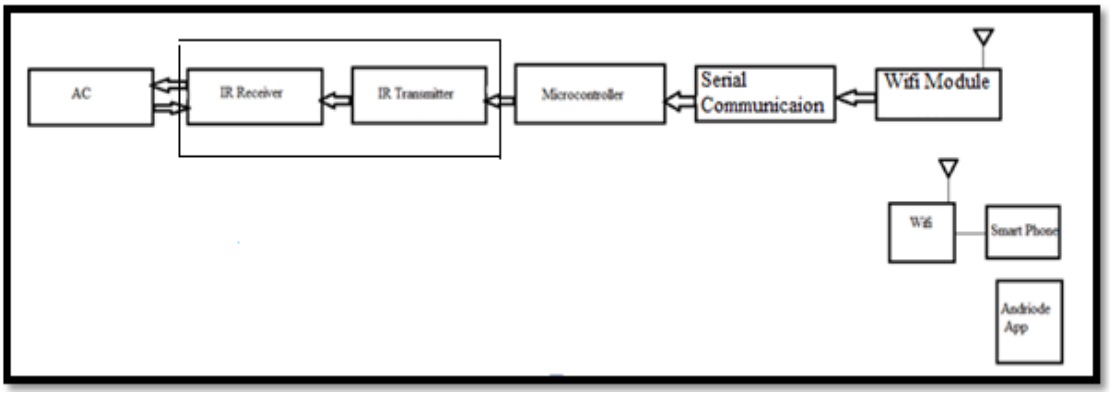

Figure 1 System Model Diagram

The above figure is the block diagram of Universal Control Unit. The universal module consists of a microcontroller, IR transmitter and IR receiver, Wi-Fi module as shown above.

3.2 DESIGN AND IMPLEMENTATION

It is an extensible and flexible system. This section describes the resulting top-level design that reflects these goals. An Automation System has three logical levels, as seen in Figure 1. The top-most layer is the input, which can be any other kind of input but in this project the input is given by the user via PC or an Android application. The control unit comprises the next layer. This layer reads the inputs and performs actions depending on the values of the inputs and the control program specified by the user. In our project, microcontroller acts as the control layer. Finally, the control unit outputs commands to the actuator layer. The actuator layer is responsible for forming and sending commands to the real-world systems such as air conditioners and other electrical appliances. For our system, we will design our own control module. The details and corresponding subcomponents of each of these layers are described in the remainder of this section.

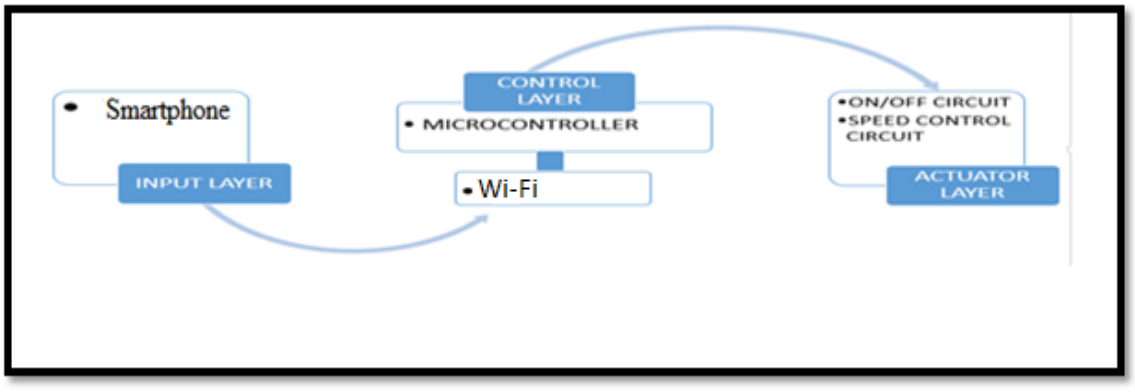

Figure 2 Flow chart of logical layers of universal module

The above figure shows that input layer comprises of smartphone or pc application which will pass packets to control layer. In control layer, we have microcontroller, Wi-Fi module and learning, storing and transmitting IR circuit. Control layer passes the command to actuator layer in order to perform these commands.

3.2.1 INPUT LAYER

Input layer comprises of two software applications which are as follows:

- Visual Basic application for PC
- Android application for Mobile Phone

3.2.1.1 VISUAL BASIC APPLICATION FOR PC

The project is “IP based AC Universal control System”. User can control their AC of any company from anywhere using their smart phones or PCs. Hence, Visual Basic is used in project to develop PC application.

Visual Basic was originally created to make it easier to write programs for the Windows computer operating system. The basis of Visual Basic is an earlier programming language called BASIC. Visual Basic is easily the most widely used computer programming system in the history of software. Visual Basic was one of the first systems that made it practical to write programs for the Windows operating system. This was possible because VB included software tools to automatically create the detailed programming required by Windows. These software tools not only create Windows programs, they also take full advantage of

the graphical way that Windows works by letting programmers "draw" their systems with a mouse on the computer. This is why it's called "Visual" Basic.

Visual Basic also provides unique and complete software architecture. "Architecture" is the way computer programs, such as Windows and VB programs, work together. A programmer can create an application using the components provided by the Visual Basic program itself.

It is the part of Visual Basic 2012. Visual Basic is comprised of different portions.

Visual Basic operates in three modes.

1) **Design** mode - used to build application
2) **Run** mode - used to run the application
3) **Break** mode - application halted and debugger is available

The different portions of VB are as follows:

1) **Form Window:**

The Form Window is central to developing Visual Basic applications. It is where application is drawn.

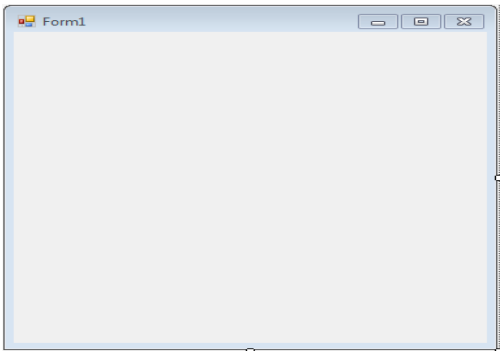

Figure 3-Form Window

The above figure is one of the portions of VB, where objects are placed.

2) **Tool Box**:

The Toolbox is the selection menu for controls used in your application.

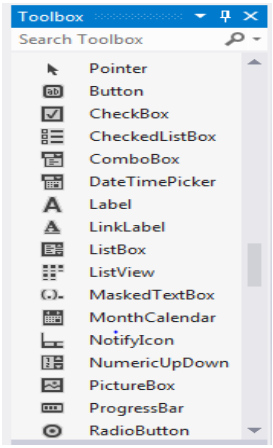

Figure 4-Tool Box

Figure 4 represents tool box which contains the objects used for making application.

All the objects which are required for designing an application are dragged from the toll box.

3) Properties Windows:

The **Properties Window** is used to establish initial property values for objects. Two views are available: Alphabetic and Categorized. Under this box are the available properties of the currently selected object.

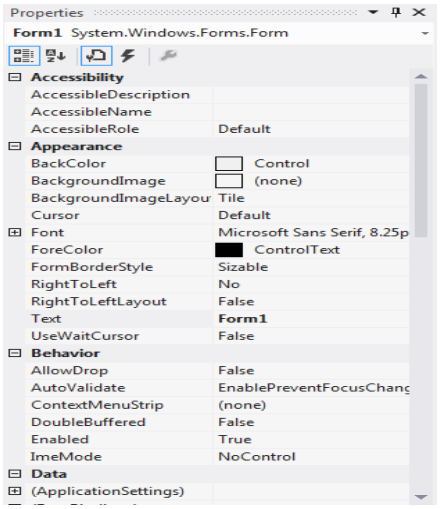

Figure 5-Properties Window

The above figure shows all the characteristics of objects which can be changed according to our requirement.

Every object has its own property window.

3.2.1.1.1 LAYOUT OF PC APPLICATION

Figure 6-Layout of PC Application

The above figure is actual display of application. It consists of 7 buttons, 7 radio buttons and two text boxes.

The name of Application is “IP Based Air Conditioner Control System Application”. The layout shows two text box which will show what button is pressed and the upper text box will show the temperature and whether light is present or not. There are seven buttons each of which will perform a specific function. They are power on, power off, AC swing, transmit and learn, temperature and light button. It consists of two group boxes, one of them is fan speed and the other is temperature. Each temperature has its own unique IR code. Due to this reason, four temperatures are selected. In a similar manner for fan speed three different radio buttons are used for different fan speed. All of these objects are dragged from the tool box in the form window and their properties are being changed from the properties window which includes Appearance, font, design, size and many more. Similarly, properties of form which is basically the application and text box are changed according to requirement.

As layout is complete now, the next step is the network programming. Each of the objects will be programmed. For programming application, Visual studio 2012 is required and VB is part of it. By clicking each object, the programming window will be opened and the object is programmed according to the requirement. Application is programmed using UDP. When any of the buttons is clicked, it will pass the command according to programming, to the microcontroller through IP network and the control unit will generate the command accordingly.[2]

3.2.1.2 Android application for Mobile Phone

For the development of our mobile application, we have chosen the Android platform. Android is an Operating System developed by Google. It currently runs on a wide variety of devices, including tablets, smartphones, smart watches and even television sets. Broadly speaking, Java serves as the backbone for Android app development, since logarithms are implemented in Java. XML is used to design layouts and physical attributes of the application. [3]

We have also chosen Eclipse as an Integrated Development Environment (IDE). The Eclipse ADT (Android Development Tools) plugin extends the capabilities of Eclipse to let developers set up new Android projects, create an application UI, add packages based on the Android Framework API, debug their applications using the Android SDK tools, and export signed (or unsigned) .apk files in order to distribute their applications.

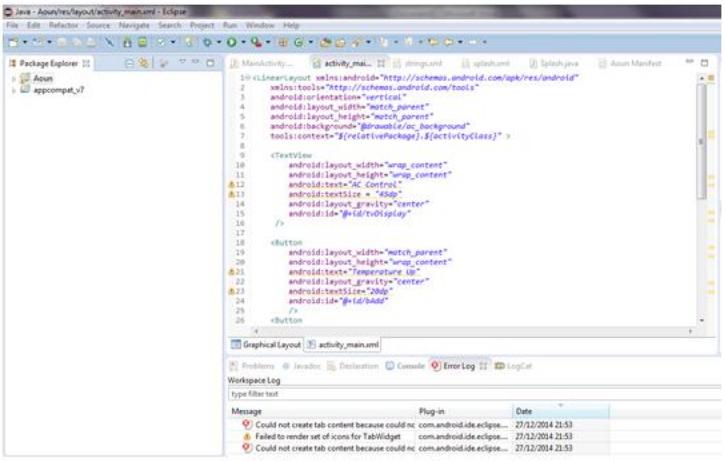

Figure 7 Eclipse as an Integrated Development Environment (IDE).

The above figure is the eclipse window as an IDE.

As of now, we have started working on the layout of the application using XML.

Our splash currently looks like this:

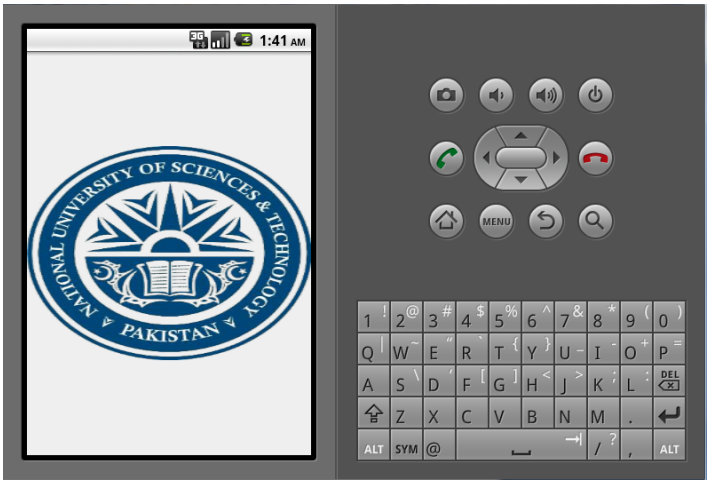

Figure 8 Layout of Android Application

Above figure shows the screen that comes on when you first select the application on your device.

The main screen appears five seconds later:

Figure 9 Main screen on Smart phone

Above figure is the emulator that is running Android 2.2 (Froyo). This is because, even though our application is compiled using Android 5.0, it is being targeted for devices running Android 2.2 and above, so as to cater to both low end and high end devices running Android.

3.2.1.2.1 LAYOUT OF ANDROID APPLICATION

We designed the Android application layout to be as user-friendly and aesthetic as possible.

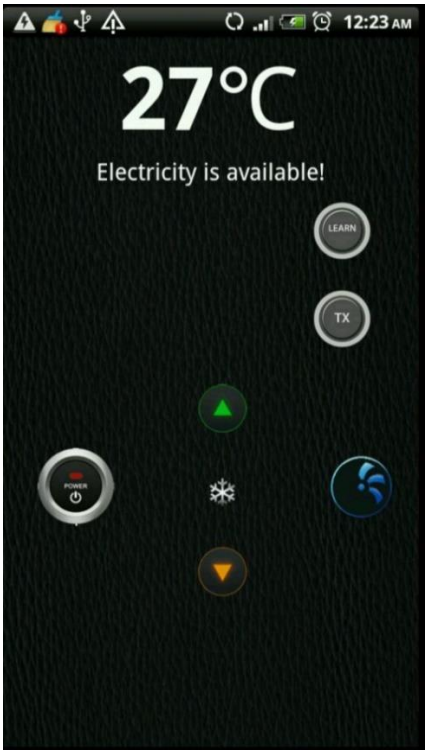

Figure 10- Android application layout

The above figure is actual display of android application. By pressing each button, a particular character will be sent to the universal control module.

3.2.2 CONTROL LAYER

3.2.2.1 MICROCONTROLLER

The purpose of the control would be to receive the input from the upper layer, process it and send the desired command to the actuator layer. The microcontroller acts as perfect control system for us.

Atmega16 can be considered as an 8-bit Computer with 32K Bytes in-System Programmable Flash available in 40 PDIP. As Micro-controller is a mini computer, it has a RAM of 2K Byte, 32 I/O general purpose registers, 1024 Bytes of EEPROM. The speed of this computer is Maximum 16 MHz. Micro-processor of PC has much more capabilities, but you can't use it for your own embedded system/circuit. They are not available as programmable like micro-controller. It is million times programmable (readable/writable/erasable, 20 years data retention).

Figure 11-Atmega16 40pin-PDIP

Figure 11 is the packaging of Atemga16. It is rectangular in shape and has two parallel rows of pins.

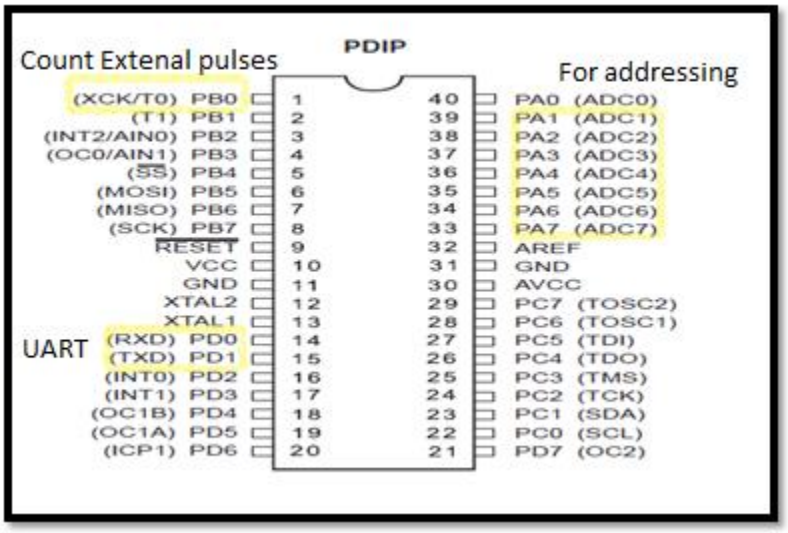

Figure 12 ATMEGA16 Pin Configuration along with the pins which have been used [4]

Figure 12 is Atmega pin configuration in which blocks is showing which pins are used.

3.2.2.2 ARDUINO MEGA

The Arduino Mega is a microcontroller board based on the ATmega1280. It has 54 digital input/output pins, 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. We used Arduino to configure the Wi-Fi module through AT commands.

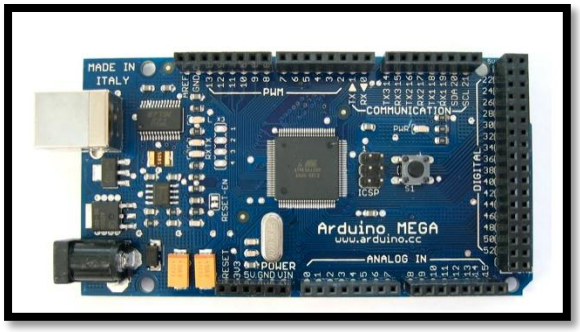

Figure 13 -Arduino Mega

The above figure is Arduino Mega (microcontroller board based on Atmega1280).

3.2.2.3 ESP8266 SERIAL WIFI MODULE

ESP8266 offers a complete and self-contained Wi-Fi networking solution, allowing it to either host the application. Serving as a Wi-Fi adapter, wireless internet access can be added to any microcontroller-based design with simple connectivity through UART. It will use to connect our module to the android or pc application.

It has many features:

1. 802.11 b/g/n protocol
2. Integrated TCP/IP protocol stack
3. Integrated temperature sensor
4. Supports antenna diversity
5. Integrated low power 32-bit CPU could be used as application processor
6. SDIO 2.0, SPI, UART

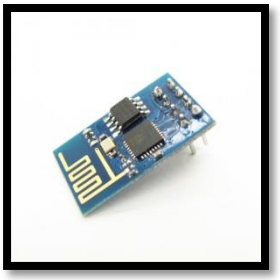

Figure 14- ESP8266 WIFI Module

Figure 14 is the display of the Wi-Fi module used in this project.

3.2.2.4 HARDWARE DESIGN OF IR LEARNING AND TRANSMITTING UNIT

The learning, storing and transmitting IR code circuit will run in two modes. The first one is **learning** mode and the second one is **transmitting** mode. In learning mode, the user will pass the command “learn” from the pc or android application and the circuit will be in learning mode and learn the IR code and store it in the EEPROM corresponding to the button which user has pressed. After it, the user will press the “transmit” button and the circuit will now run in transmit mode and transmit the codes which it had stored earlier and will always remain in transmit mode.

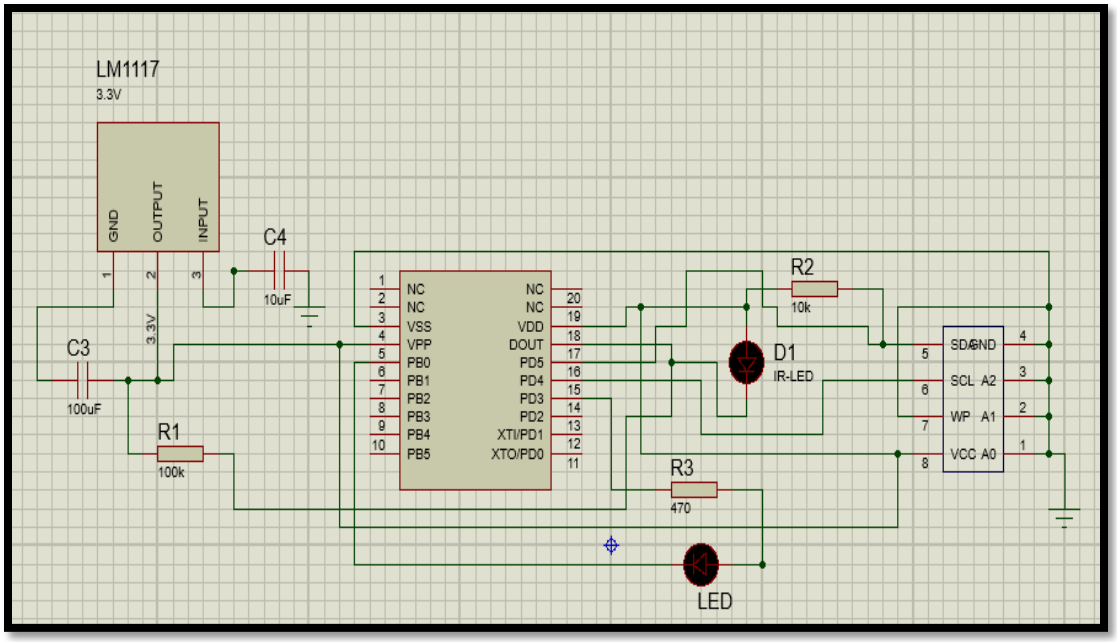

Figure 15-Learning, storing and transmitting IR protocol circuit

The figure shows the circuit design consists of IR protocol learning IC and EEPROM.

IR protocol learning and transmitting IC which 16 pins IC connected EEPROM which is 8 pins IC. Learning IC will learn the IR protocol from the air conditioner remote and store it in EEPROM. Analog switch array is connected to IR protocol learning and transmitting IC. Each switch in analog switch array) correspond to unique IR code stored in EEPROM. Analog switch array is connected with microcontroller atmega16 which is connected to Wi-Fi module through serial communication. Microcontroller passes the 7 bits address to switch array and according to the packet it switch is triggered and learning IC transmit correspond IR code in EEPROM as AC remote.

3.2.2.5 ANALOG SWITCH ARRAY

The device contains 8 x 16 array of cross point switches along with a 7 to 128 line decoder and latch circuits. Any one of the 128 switches can be addressed by selecting the appropriate seven address bits. The selected switch can be turned on or off by applying a logical one or zero to the DATA input. VSS is the ground reference of the digital inputs. The range of the analog signal is from VDD to VEE. Chip Select (CS) allows the cross point array to be cascaded for matrix expansion.

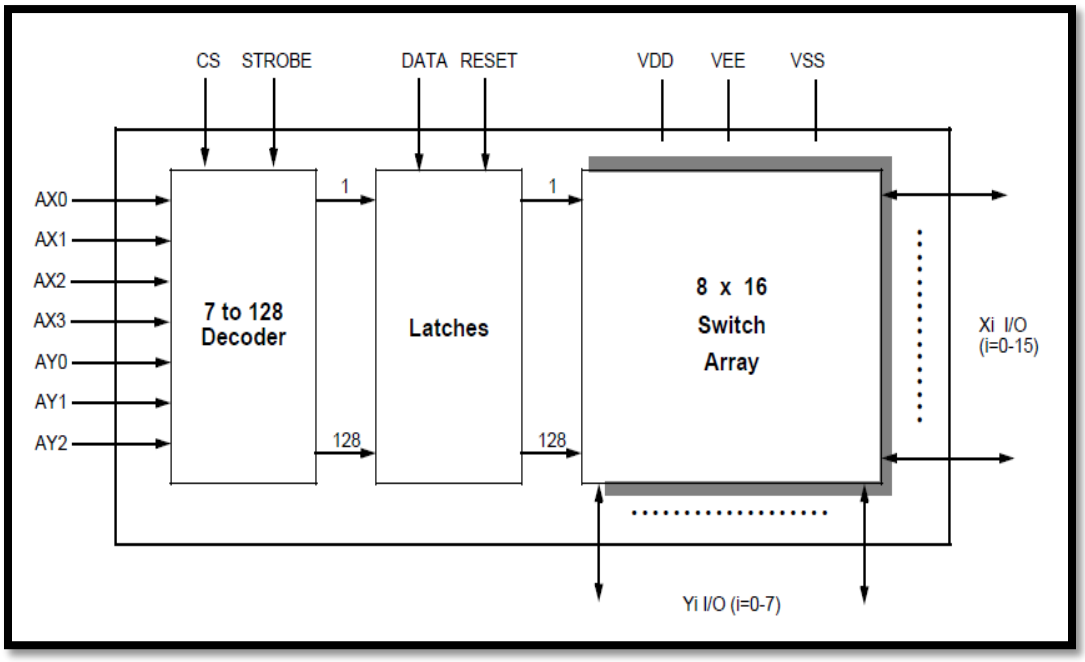

Figure 16 Functional Block diagram

Figure 16 is the internal structure of the IC which works as a keypad in this module.

3.2.2.6 COMMUNICATION PROTOCOLS

Communication between Atmega16 and Arduino mega is UART (Universal Asynchronous Receiver/Transmitter) base. UART is serial communication between two serial ports of microcontroller. In UART communication no clock is required and baud rate must be equal for both microcontrollers in order to receive correct data.

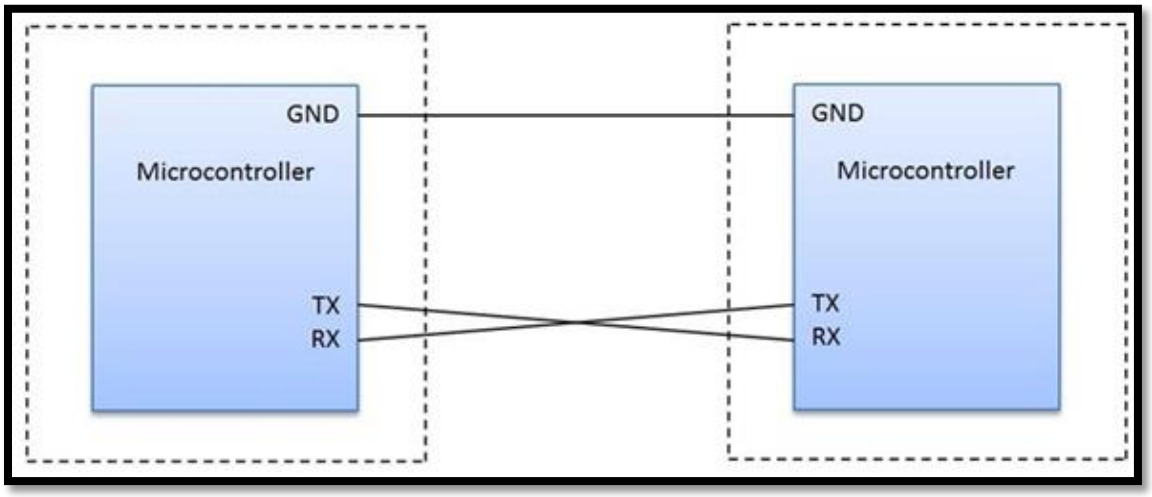

Figure 17 UART communication structure [5]

Figure 17 is showing UART communication (Universal Asynchronous Receiver Transmitter)

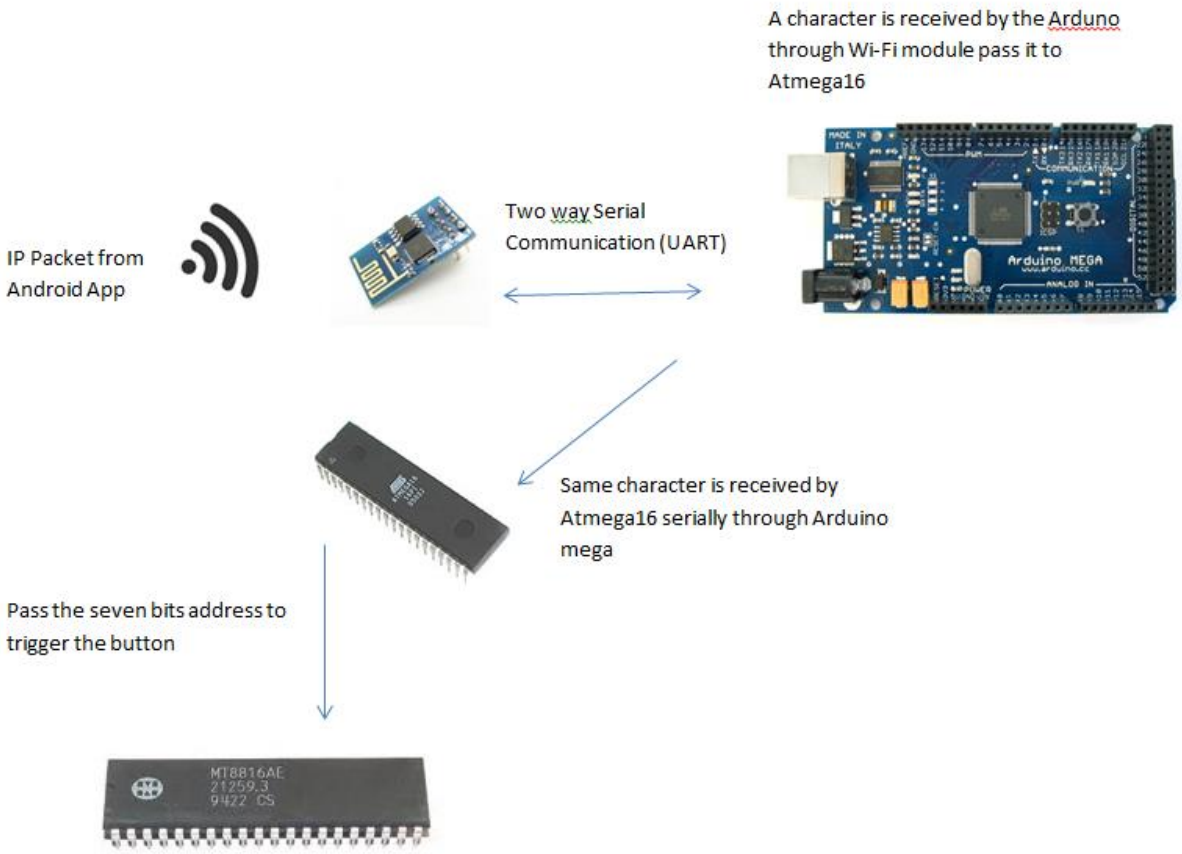

Figure 18 System diagram of module

The figure 18 is showing the components and the order of communication.

3.2.2.7 TEMPERATURE SENSOR CIRCUIT

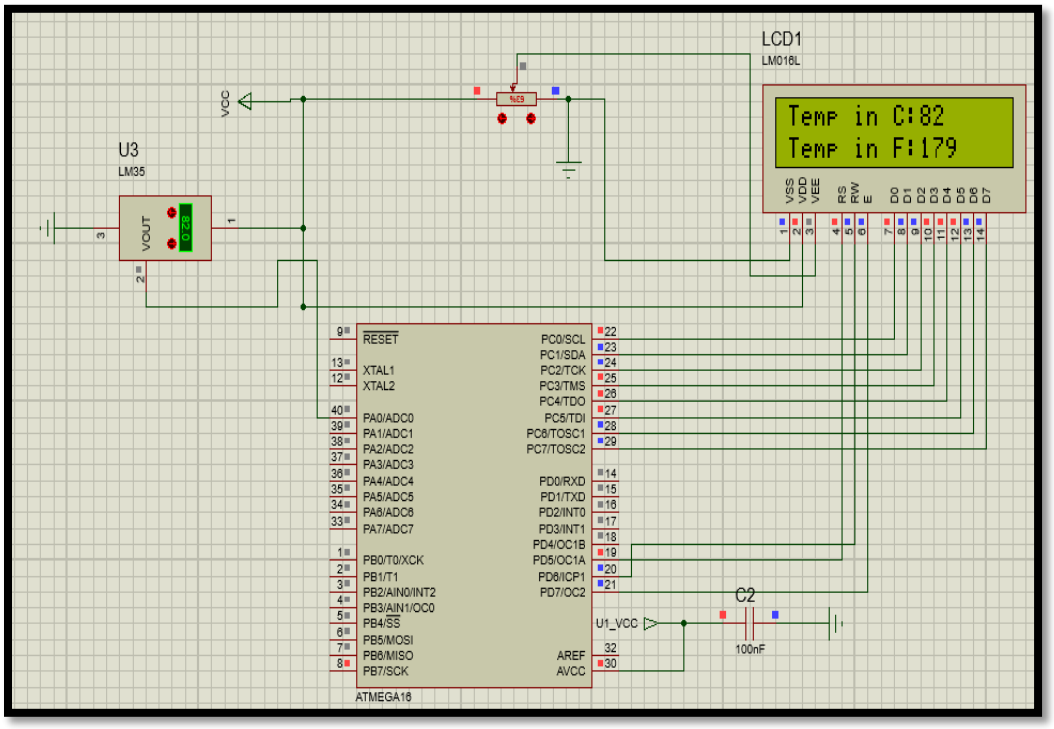

Figure 19-Temperature Sensor Circuit

Figure 19 has circuit which shows the temperature of a room.

Circuit consists of LM35 temperature sensor which converts the temperature into analog voltage and it is connected to Atmega16 which takes the analog signal and convert analog signal into a digital form. [6]

CHAPTER FOUR

4 ANALYSIS AND EVALUATION

The analysis will go through each phase of development step-wise and document what the output at each stage was. The first step is test both the application i.e. PC application and android application. After this, testing of hardware was done. Each of them is explained step wise.

4.1 INPUT LAYER

4.1.1 VISUAL BASIC APLICATION

Earlier, when the control module was not developed, UDP client server module was used to test the application. The textbox in Figure 13 shows that ON was pressed. Once the button was pressed, it transmitted "ON)" command, which was received by UDP client server that was acting as a control module here.

Figure 20- PC Application Layout

Figure 20 is the actual layout of PC application which is consisting of seven buttons, seven radio buttons and two text boxes.

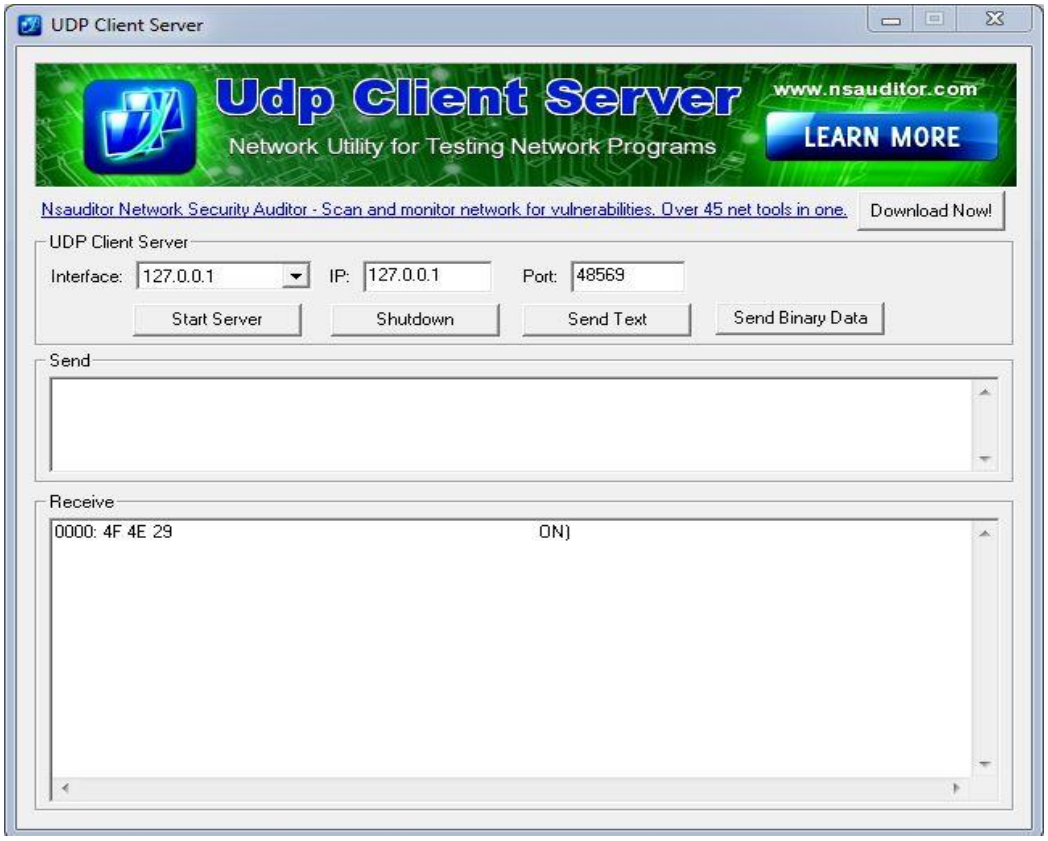

Figure 21- UDP Client Server

The above figure is UDP Client server which is showing that it has received the character and application is successfully communicating.

4.1.2 ANDROID APPLICATION

In order to test our Android application's network functionality, we created a test layout with a textbox that updates in real-time to show the status of ongoing tasks. We also set the destination IP to 'localhost'. In this way, the application sent a command to itself using UDP. If the command was received, it was displayed in

the textbox. In this way, both the sending and receiving functionality of the application was tested.

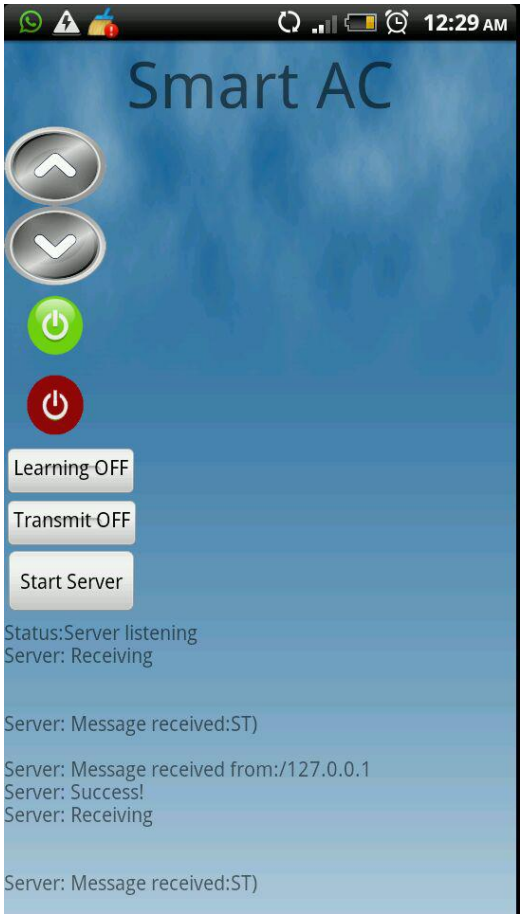

Figure 22- Android test layout

The test results shown in Figure 22 has indicated that the client/server functionality of the application was working as desired.

4.2 CONTROL LAYER

Results on Bread Board:

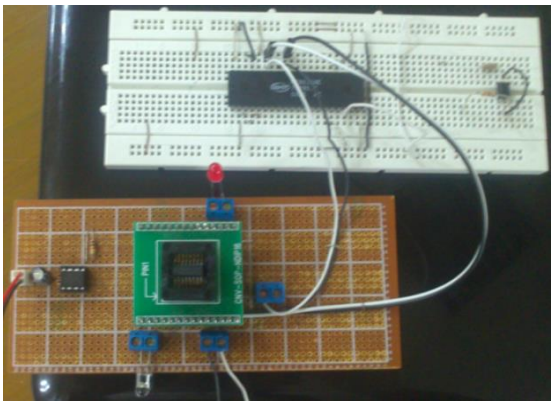

Figure 23 Learning Circuit

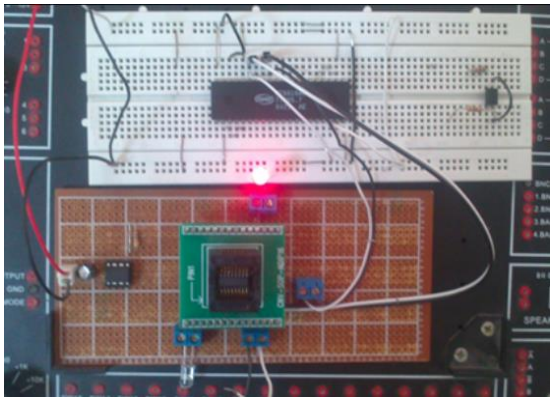

Figure 24 Transmitting Circuit

Figure 22-23 shows the assembled circuit on breadboard.

This is the initial interfacing of IR learning and transmitting circuit with analog switch array. Circuit consist of EEPROM which store IR codes which are received by IR learning and transmitting IC. When two pins of IR learning and transmitting IC are short respective IR code is transmitted.

This circuit operates in two modes:

1. **Learning Mode**

   In this mode, circuit receives the IR codes through the remote of any appliance and stores it in EEPROM.

2. **Transmitting Mode**

   In this mode module will transmit the IR code to operate devices.

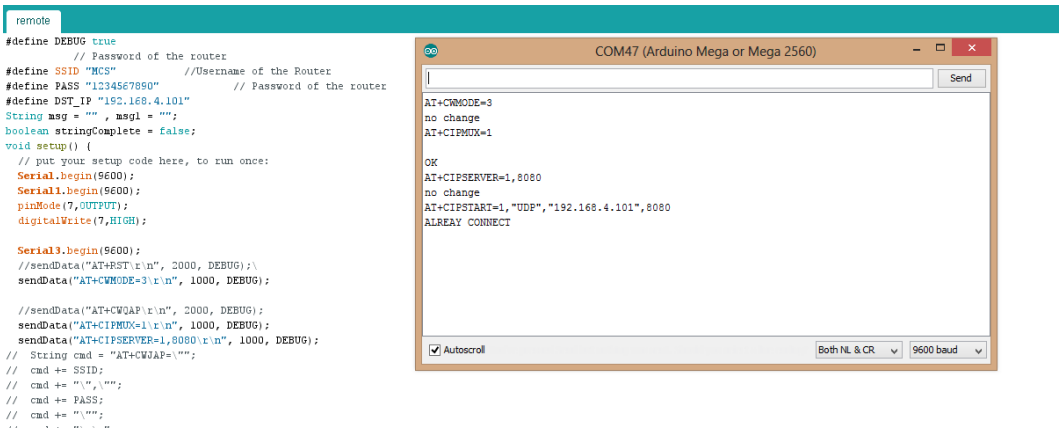

Figure 25 Serial monitor of Arduino Mega

Wi-Fi module receives the data from android application and passes it to Arduino mega through UART communication. Wi-Fi module receives IP packet and send a character against each button on application. Arduino mega passes this character to Atmega16 through serial and IR code is generated according to the character.

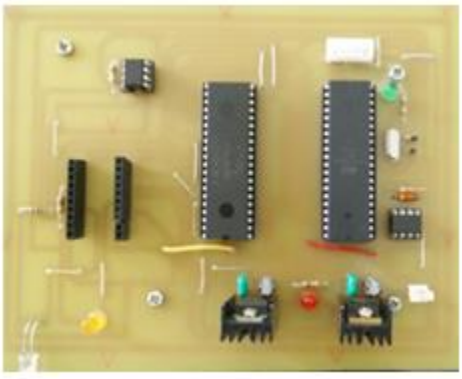

Figure 26 TX circuit with Atmga

Figure 27 Arduino mega with Wi-Fi

Figure 24-25 shows the assembled circuit on PCB.

CHAPTER FIVE

5 FUTURE WORK

5.1 INTRODUCTION

This chapter will focus on the ideas and suggestions given as input during the work done on the project. These ideas show how to expand the project into a more practical solution for real life controlling and monitoring system of IR devices within a building.

5.2 AVENUES OF FUTURE DEVELOPMENT

Any future work will be along the lines of making the product much smarter and expanding its functionality.

5.2.1 CONNECTIVITY WITH INTERNET

The Control Module developed is working on LAN. In future, it can be deployed on Internet where the users just have to buy a static IP for their own modules. And can operate the IR devices from anywhere in the world.

5.2.2 MONITORING SYSTEM

Centralized monitoring system can be established with it, where the control of all the AC and all other IR devices like multimedia or cameras is with one person and it ensures security and control over all building.

5.2.3 STATUS OF IR DEVICES

This module in now restricted to get room temperature and shows the availability of main supply, if we want get status of any IR device through this module whether it is on or off etc. we will be able to get status on android or PC application from remote places.

5.2.4 IMPLEMENTATION OF TCP

Communication between android application and Wi-Fi module through UDP protocol which is connection less and data is loss between communications. Communications on TCP will never loss data and Wi-Fi will receive the IP packet which is send by the application.

5.3 CONCLUSION

This section has offered future work areas that were perceived to be practical and were offered as critical but constructive criticism during the project work. They are also the results of the problem areas the team members faced as well as ideas got during brainstorming on how to improve this product with future effort.

CHAPTER SIX

6 CONCLUSION

6.1 OVERVIEW

To build a system consisting of an Android device "The Internet of Things", which, when connected to the network via Wi-Fi, will be capable of controlling Air conditioners. This appliance will be assigned unique IP address and the user will be able to switch them On/Off and control its parameter through a control packet sent over the IP network. An android application will be developed in order to provide the user an easy platform, where, by simply touching the icon, the user will be able to switch the appliance in accordance with the user's requirements. At the other end, controller hardware will be interfaced with the appliances, which, after receiving a control packet, will perform the desired operations. This will allow the user to roam freely and access the devices from anywhere as long as the user is connected to the network.

6.2 OBJECTIVES ACHIEVED

We designed the control module which has following specifications:

1) Able to read IP packet and its contents

2) Perform switching of AC or any IR device

3) Read status information from devices and send data back to control application i.e. Centralized monitoring of IR devices within the building.

We developed front end application on Android device and PC.

6.3 LIMITATIONS

The whole project is deployed on LAN. For bringing it on internet, a static IP is needed to be bought which would have increased the cost of the project as it is expensive. In order to avoid it, the project is restricted to LAN only.

6.4 APPLICATIONS

A person will be able to control the AC unit or any other IR device operations remotely just using his smart phone.

Centralized monitoring and controlling system of AC or any other IR device units within a building like university can be established.

A person will be able to get status of IR devices whether device is on or off or room temperature on his smart phone and PC.

7 Bibliography:

[1] Waleed Ahmed, "IP network based energy management and control system" Military College of Signals (NUST),2013.

[2] Microsoft Developer Network, "Introduction to the Visual Basic Programming Language" [Online]. Available: http://msdn.microsoft.com/library/xk24xdbe(v=vs.90).aspx

[3] Switkin,D, "Android Application Development", Senior Software Engineer, Google Inc.

[4] Ali-Mazidi, "AVR hardware connection, hex file and flash loaders" in AVR-Microcontroller-and-Embedded-Systems, ch 3, pp 30

[5] Ali-Mazidi, "Serial Communication" in AVR-Microcontroller-and-Embedded-Systems, ch 7, pp 303

[6] Prof Yusuf Mulge, "Remote Temperature Monitoring Using LM35 sensor and Intimate Android user via C2DM Service", Department of Computer Science and Engineering, PDM College of Engineering for Women, MDU, Rohtak, BAHADURGARH, India, Vol. 2, Issue. 6, June 2013, pg.32 – 36

[7] "Control Daikin air conditioner over the Internet" [Online]. Available: http://www.instructables.com/id/Control-Daikin-air-conditioner-over-the-Internet/step3/null/

[8]] James F. Krouse, Keith W. Rose, "IP layer" in Computer Networks, 6th edition, ch 3, pp 100-177

[8] Delgado, Picking, Groud" Remote-Controlled Home Automation Systems with Different Network Technologies", CAIR, University of Wales, NEWI, Wrexham.

[9] Frey, Donald. Smith, Vernon, "Advanced Automated HVAC Fault Detection And Diagnostics Commercialization Program", Architectural Energy Corporation, Boulder, 2008. Publication number: CEC-500-2013-054.

APPENDIX A

Visual Basic Code

```
Imports System.Net
Imports System.Text.Encoding
Public Class Form1
    Dim udpclient As New Sockets.UdpClient(0)
    Dim subscriber As New Sockets.UdpClient(8080)
    Dim ipaddrs As String = "192.168.4.1"
    Dim port As String = "8080"
    Dim rcvbyte1() As Byte = Nothing
    Dim endpoint As IPEndPoint = New IPEndPoint(IPAddress.Any, 0)
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        udpclient.Connect(ipaddrs, port)
        subscriber.Client.ReceiveTimeout = 100
        subscriber.Client.Blocking = False
        Timer2.Enabled = True
    End Sub

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click        ' Power Off
        ' Me.Button2.Visible = False
```

```
        Dim sendbyte() As Byte = ASCII.GetBytes("F)")
        udpclient.Send(sendbyte, sendbyte.Length)
        TextBox1.Text = "OFF"
        Me.Button1.Visible = True
    End Sub
    Private Sub Button6_Click(sender As Object, e As EventArgs) Handles
Button6.Click           ' window Swing
        Dim sendbyte() As Byte = ASCII.GetBytes("W)")
        udpclient.Send(sendbyte, sendbyte.Length)
        TextBox1.Text = "Window Swing"
    End Sub
    Private Sub Button7_Click(sender As Object, e As EventArgs) Handles
Button7.Click          ' Learning mode
        Dim sendbyte() As Byte = ASCII.GetBytes("L)")
        udpclient.Send(sendbyte, sendbyte.Length)
        TextBox1.Text = "It is in learning mode"
        ' Me.Button7.Visible = False
    End Sub

    Private Sub Button8_Click(sender As Object, e As EventArgs) Handles
Button8.Click           ' Transmitting mode
        Dim sendbyte() As Byte = ASCII.GetBytes("T)")
        udpclient.Send(sendbyte, sendbyte.Length)
```

```
        TextBox1.Text = "It is in transmitting mode"
        ' Me.Button8.Visible = False
    End Sub
    Private Sub Button11_Click(sender As Object, e As EventArgs) Handles
Button11.Click            ' Temperature status
        Dim sendbyte() As Byte = ASCII.GetBytes("ST)")
        udpclient.Send(sendbyte, sendbyte.Length)
    End Sub
    Private Sub RadioButton2_Click(sender As Object, e As EventArgs) Handles
RadioButton2.Click            ' Medium
        Dim sendbyte() As Byte = ASCII.GetBytes("MED)")
        udpclient.Send(sendbyte, sendbyte.Length)
        TextBox1.Text = "Fan speed is medium"
    End Sub
    Private Sub RadioButton5_Click(sender As Object, e As EventArgs) Handles
RadioButton5.Click            ' temp is 20
        Dim sendbyte() As Byte = ASCII.GetBytes("20)")
        udpclient.Send(sendbyte, sendbyte.Length)
        TextBox1.Text = "Temperature is 20"
    End Sub
    Private Sub RadioButton6_Click(sender As Object, e As EventArgs) Handles
RadioButton6.Click            'temp is 22
        Dim sendbyte() As Byte = ASCII.GetBytes("22)")
```

```
        udpclient.Send(sendbyte, sendbyte.Length)
        TextBox1.Text = "Temperature is 22"
    End Sub
    Private Sub RadioButton7_Click(sender As Object, e As EventArgs) Handles
RadioButton7.Click           ' temp is 24
        Dim sendbyte() As Byte = ASCII.GetBytes("24)")
        udpclient.Send(sendbyte, sendbyte.Length)
        TextBox1.Text = "Temperature is 24"
    End Sub
    Private Sub Timer2_Tick(sender As Object, e As EventArgs) Handles
Timer2.Tick
        Try
            Dim  endpoint As IPEndPoint = New IPEndPoint(IPAddress.Any, 0)
            Dim rcvbyte() As Byte = subscriber.Receive(endpoint)
            TextBox2.Text = ASCII.GetString(rcvbyte)
        Catch ex As Exception
    End Sub
    Private Sub RadioButton1_Click(sender As Object, e As EventArgs) Handles
RadioButton1.Click          ' low
        Dim sendbyte() As Byte = ASCII.GetBytes("LOW)")
        udpclient.Send(sendbyte, sendbyte.Length)
        TextBox1.Text = "Fan speed is low"
    End Sub
```

APPENDIX B

Code of Arduino Mega and ATMEGA16

```
char re;
int state = LOW;
String check = "";
boolean complete = false;
void setup()
{  Serial.begin(9600);
  pinMode(14,OUTPUT);
  digitalWrite(14,LOW);
  DDRB=DDRB | 0b00000110;
  PORTB=0x00;
  DDRA=0xFF;}
void loop()
{if (complete == true)
{ if (check[0] == 'L')
  { digitalWrite(14,HIGH);
    PORTA=0b00000000;
  PORTB= PORTB | 0b00000010;
    delay(100);
```

```
    PORTB= PORTB | 0b00000100;
    PORTB= PORTB & 0b11111101;
    delay(2500);
    PORTB= PORTB & 0b11111011;
    PORTB= PORTB | 0b00000010;
    PORTB= PORTB & 0b11111101;
    digitalWrite(14,LOW);
    re = ' ';
    delay(2500);}
else if (check[0] == '2')
  { digitalWrite(14,HIGH);
    PORTA=0b00101000;
    PORTB= PORTB | 0b00000010;
    delay(100);
    PORTB= PORTB | 0b00000100;
    PORTB= PORTB & 0b11111101;
    delay(500);
    PORTB= PORTB & 0b11111011;
    PORTB= PORTB | 0b00000010;
    delay(200);
    PORTB= PORTB & 0b11111101;
```

```
    digitalWrite(14,LOW);
complete = false;
 check = ""; }}
void serialEvent() {
  while (Serial.available()) {
    char inChar = (char)Serial.read();
    check += inChar;
    if (inChar == ')') {
      complete = true;}}}
// Arduino Code
#define DEBUG true
#define SSID "MCS"           //Username of the Router
#define PASS "1234567890"           // Password of the router
#define DST_IP "192.168.4.101"
String msg = "" , msg1 = "";
boolean stringComplete = false;
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  Serial1.begin(9600);
  pinMode(7,OUTPUT);
```

```
  digitalWrite(7,HIGH);
  Serial3.begin(9600);
  String cmd1 = "AT+CIPSTART=1,";    cmd1 += "\"UDP""\","""\"";
  cmd1 += DST_IP;
  cmd1 += "\",8080";
  cmd1 += "\r\n";
  sendData(cmd1, 1000, DEBUG);}
void loop() {
  if (stringComplete) {
    int len = msg.length();
    int i = 0 , check = 0;
    String newvalue = "";
    for (i = 0 ; i < len ; i++)
    {if (msg[i] == ')') {
       check = 2;}
      if (check == 1) {
        newvalue += msg[i];}
      if (msg[i] == ':') {
        check = 1;}}
     Serial3.print(newvalue);
     Serial3.print(")");
```

```
 Serial.println(msg);
if(newvalue == "ST")
{     int long temp = analogRead(A3);
  int long temperature = (float) (500 * temp)/1023;
    sendData("AT+CIPSEND=1,7\r\n", 500, DEBUG);
    Serial1.write("temp=");
    Serial1.print(temperature);
    sendData("\r\n", 500, DEBUG);    }
else if (newvalue == "GI")
{ int long lit = analogRead(A7);
  Serial.println(lit);
  if (lit > 300)
  {
    sendData("AT+CIPSEND=1,2\r\n", 500, DEBUG);
    Serial1.write("ON");
    sendData("\r\n", 500, DEBUG);
  }
  else if (lit  < 50)
  {sendData("AT+CIPSEND=1,3\r\n", 500, DEBUG);
    Serial1.write("OFF");
    sendData("\r\n", 500, DEBUG);}}
```

```
void serialEvent1() {
  while (Serial1.available()) {
    char inChar = (char)Serial1.read();
    msg += inChar;
    if (inChar == ')') {
  stringComplete = true;}}}
String sendData(String command, const int timeout, boolean debug)
{String response = "";
  Serial1.print(command); // send the read character to the Serial1
  long int time = millis();
  while ( (time + timeout) > millis())
  {while (Serial1.available())
    {char c = Serial1.read(); // read the next character.
      response += c;}}
  if (debug)
  { Serial.print(response); }
  return response;}
```