**1. INTRODUCTION**

Home automation is also named as domestics or Smart home. It involves the control and automation of lighting, heating, ventilation, air conditioning and security, as well as home appliances. Wi-Fi is often used for remote monitoring and control. Home devices, when remotely monitored and controlled via Internet is a part of Internet of things. Modern systems generally consist of switches and sensors connected to a central hub called a gateway from which the system is controlled with a user interface that is interacted either with a mobile phone software, tablet, computers or a web interface, often but not always via internet cloud services. World’s demand for electricity had grown 85% between 2010 and 2018. We can decrease the electricity gross electricity wasted each year by turning off our home appliances when not in use. Unlike most of available home automation system in the market the proposed system is scalable that one server can manage many hardware interface modules as long as it exists on Wi-Fi network coverage.

**Objective**:

Home, it is the place where one fancies or desires to be after a long tiring day. People come home exhausted after a long hard-working day. Some are way too tired that they find it hard to move once they land on their couch, sofa or bed. So, any small device/technology that would help them switch their lights on or off, or play their favorite music etc. on a go with their voice with the aid of their smart phones would make their home more comfortable. Moreover, it would be better if everything such as warming bath water and adjusting the room temperature were already done before they reach their home just by giving a voice command. So, when people would arrive home, they would find the room temperature, the bath water adjusted to their suitable preferences, and they could relax right away and feel cozier and rather, feel more homely.

Human assistants like housekeepers were a way for millionaires to keep up their homes in the past. Even now when technology is handy enough only the well to do people of the society are blessed with these new smart home devices, as these devices costs are a bit high.

In recent years, the vocabulary closely related to smart homes has been understood by most families,

such as smart home local area networks, home gateways and smart devices. Some domestic

universities have also begun research and development of smart home appliances systems.S3C2410

series high-performance single-chip microcomputers are used as central control cores to extend the

Bluetooth module interface and home gateway solutions for GSM module interfaces, but they do not

relate to home node networking solutions.In addition, in a smart home network where a large number

of nodes are deployed, the use of the Bluetooth networking technology is complicated and the cost is

high. Most domestic smart homes use ZigBee as the inter-node communication module. ZigBee nodes

have the advantages of flexible networking and reliable operation. However, the ZigBee protocol stack

with high network efficiency needs to be charged and the cost of a single node is high. The gateway is

required to convert the network during operation. You can upload data to the Internet. In view of the

above situation, we use the Nodemcu chip as the main control and use GSM to upload data to form a

low-cost smart home solution.

In recent years, the vocabulary closely related to smart homes has been understood by most families,

such as smart home local area networks, home gateways and smart devices. Some domestic

universities have also begun research and development of smart home appliances systems.S3C2410

series high-performance single-chip microcomputers are used as central control cores to extend the

Bluetooth module interface and home gateway solutions for GSM module interfaces, but they do not

relate to home node networking solutions.In addition, in a smart home network where a large number

of nodes are deployed, the use of the Bluetooth networking technology is complicated and the cost is

high. Most domestic smart homes use ZigBee as the inter-node communication module. ZigBee nodes

have the advantages of flexible networking and reliable operation. However, the ZigBee protocol stack

with high network efficiency needs to be charged and the cost of a single node is high. The gateway is

required to convert the network during operation. You can upload data to the Internet. In view of the

above situation, we use the Nodemcu chip as the main control and use GSM to upload data to form a

low-cost smart home solution.

**2. LITERATURE REVIEW**

A smart home covers a variety of theoretical and practical approaches that deals with living today and in the future. There are several ways or methodologies to automate home through wireless communication technology. Some of them are as follows:

1. **Vamsi Krishna Patchava, Hari Babu Kandala, P Ravi Babu**

They used the Raspberry Pi and connected camera and motion sensors and created a web UI based home surveillance and automation system.

1. **Sarthak Jain, Anant Vaibhav, Lovely Goyal**

Explained the system that can be used to control home appliances by reading the commands the subject of an email received to the specifically programmed email address of the device.

1. **Rajeev Piyare and Seong Ro Lee**

Presented a flexible and low-cost home control and monitoring system using embedded micro server with IP connectivity for controlling devices remotely using an Android application.

**3. SYSTEM DEVLOPEMENT AND METHODOLOGY**

The complete project consists of two sections:

1) Hardware Section.

2) Software Section.

***3.1. Hardware Section***

It mainly contains following devices and modules:

1. Power Supply
2. ESP8266 Node MCU (Wi-Fi module)
3. Relays
4. Push Buttons

***System Hardware Block Diagram:***

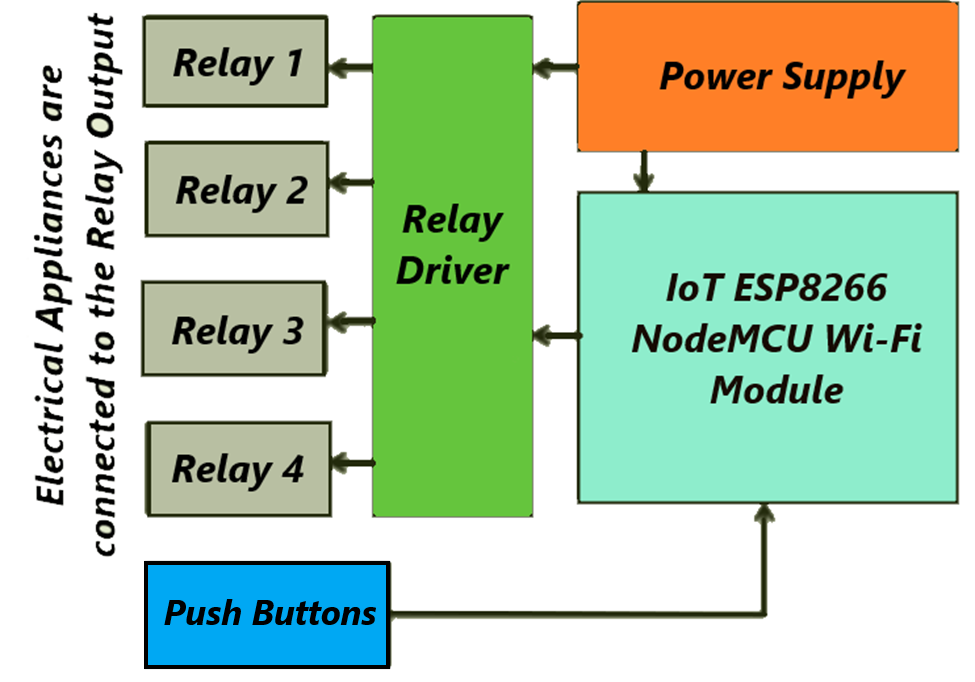


Fig 1: Block diagram of Hardware section

***3.1.1 Power Supply:***

For efficient working of overall system 5 Volt power supply is required. The required power supply is made by converting 230 V 50 Hz AC mains to 5 V DC by using transformer, rectifier, filter and regulator. The 5 V supply is given to ESP8266 Node MCU and Relay. The maximum output current provided by power supply is 0.5A. The block diagram of regulated power supply as follows:

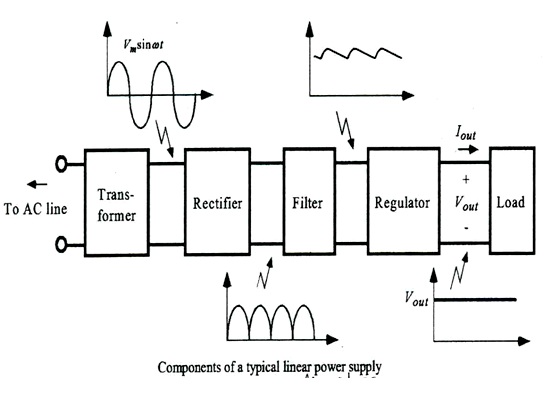


Fig 2: Block diagram of Power Supply

***3.1.2 ESP8266 Node MCU (Wi-Fi module)***

Node MCU is an open source IOT platform. It includes firmware which runs on the ESP8266 Wi-Fi SOC (System on Chip) from Expressive Systems, and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the Lua scripting language. It is based on the e-lua project, and built on the Expressive Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and spiffs.

As NodeMCU is an open source platform which can be used by anybody freely. As it consists of on-board Microcontroller as well as Wi-Fi Module it is really becomes useful board for connecting with internet tools as well as controlling (microcontroller’s work). Due to this for decreasing or to make a compact system this board make possible.

In this project NodeMCU plays significant role. This board is use for connecting with server. SSID and password of internet source (Hot-spot, router) and also Server’s link is provided to this board by programming using Arduino IDE compiler. As it connects with the internet it automatically established its connection with server and starts uploading and receiving data. It basically consists of 8 GPIO (General Purpose Input Output) which are useful of interfacing input/output devices.

**Specification:**

* Voltage:3.3V.
  + Wi-Fi Direct (P2P), soft-AP.
  + Current consumption: 10uA~170mA.
  + Flash memory attachable: 16MB max (512K normal).
  + Integrated TCP/IP protocol stack.
  + Processor: Tensilica L106 32-bit.
  + Processor speed: 80~160MHz.
  + RAM: 32K + 80K.
  + GPIOs: 17 (multiplexed with other functions).

Analog to Digital: 1 input with 1024 step resolution.

* + +19.5dBm output power in 802.11b mode
  + 802.11 support: b/g/n.
  + Maximum concurrent TCP connections: 5.

**Node MCU GPIO Interface**

The GPIO(General Purpose Input/Output) allows us to access to pins of ESP8266 , all the pins of ESP8266 accessed using the command GPIO, all the access is based on the I/O index number on the NodeMCU development kits, not the internal GPIO pin, for example, the pin ‘D7’ on the NodeMCU dev kit is mapped to the internal GPIO pin 13, if you want to turn ‘High’ or ‘Low’ that particular pin you need to called the pin number ‘7’, not the internal GPIO of the pin. When you are programming with generic ESP8266 this confusion will arise, which pin needs to be called during programming, if you are using NodeMCU devkit, it has come prepared for working with Lua interpreter which can easily program by looking the pin names associated on the Lua board. If you are using generic ESP8266 device or any other vendor boards please refer to the table below to know which IO index is associated to the internal GPIO of ESP8266.

|  |  |  |  |
| --- | --- | --- | --- |
| **Node MCU dev. kit** | **ESP8266 Pin** | **Node MCU dev. kit** | **ESP8266 Pin** |
| D0 | GPIO16 | D7 | GPIO13 |
| D1 | GPIO5 | D8 | GPIO15 |
| D2 | GPIO4 | D9 | GPIO3 |
| D3 | GPIO0 | D10 | GPIO1 |
| D4 | GPIO2 | D11 | GPIO9 |
| D5 | GPIO14 | D12 | GPIO10 |
| D6 | GPIO12 |  |  |

Table 1: GPIO Pins of ESP8266 Node MCU

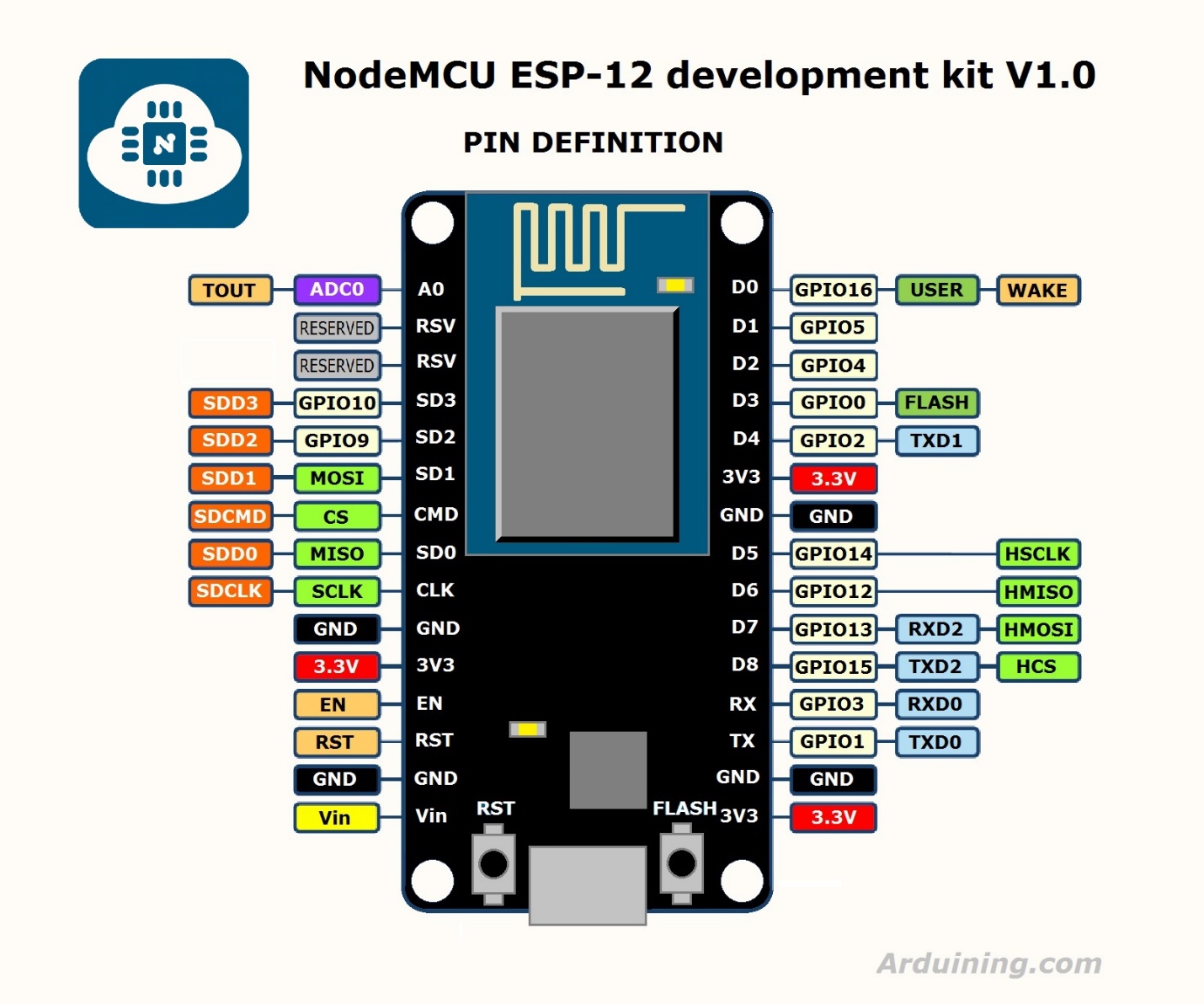


Fig 3: Pin configuration of Node MCU ESP8266

***3.1.3 Relay Module:***

**4 Channel 5V Optical Isolated Relay Module**

This is a LOW Level 5V 4-channel relay interface board, and each channel needs a 15-20mA driver current. It can be used to control various appliances and equipment with large current. It is equipped with high-current relays that work under AC250V 10A or DC30V 10A. It has a standard interface that can be controlled directly by microcontroller. This module is optically isolated from high voltage side for safety requirement and also prevent ground loop when interface to microcontroller

**Brief Data:**

• Relay Maximum output: DC 30V/10A, AC 250V/10A.

• 4 Channel Relay Module with Opto-coupler. LOW Level Trigger expansion board, which is compatible with Arduino control board.

• Standard interface that can be controlled directly by microcontroller (8051, AVR, \*PIC, DSP, ARM, ARM, MSP430, TTL logic).

• Relay of high-quality low noise relays SPDT. A common terminal, a normally open, one normally closed terminal.

• Opto-Coupler isolation, for high voltage safety and prevent ground loop with microcontroller.

**Pin Configuration:**

**Input:**

**VCC**: Connected to positive supply voltage (supply power according to relay voltage)

**GND:** Connected to supply ground.

**IN1**: Signal triggering terminal 1 of relay module

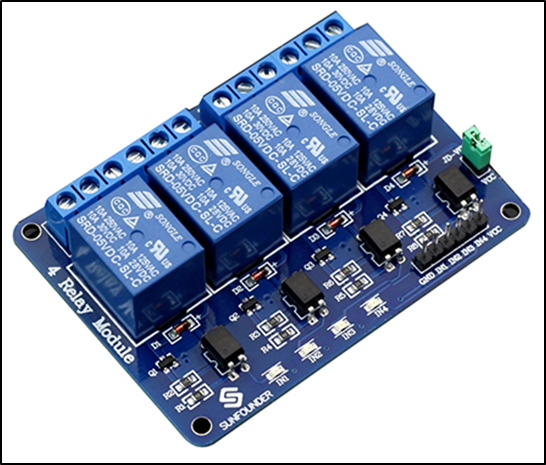
**IN2**: Signal triggering terminal 2 of relay module

**IN3**: Signal triggering terminal 3 of relay module

**IN4**: Signal triggering terminal 4 of relay module

**Output:**

Each module of the relay has one **NC** (**normally close**), one **NO** (**normally** **open**) and one **COM** (**Common**) terminal. So, there are 4 NC, 4 NO and 4 COM of the channel relay in total. NC stands for the normal close port contact and the state without power. NO stands for the normal open port contact and the state with power. COM means the common port. You can choose NC port or NO port according to whether power or not.

**Operating Principle:**

A is an electromagnet, B armature, C spring, D moving contact, and E fixed contacts. There are two fixed contacts, a normally closed one and a normally open one. When the coil is not energized, the normally open contact is the one that is off, while the normally closed one is the other that is ON.

Fig 4: Relay Module

Supply voltage to the coil and some currents will pass through the coil thus generating the electromagnetic effect. So, the armature overcomes the tension of the spring and is attracted to the core, thus closing the moving contact of the armature and the normally open (NO) contact or you may say releasing the former and the normally closed (NC) contact. After the coil is de-energized, the electromagnetic force disappears and the armature moves back to the original position, releasing the moving contact and normally closed contact. The closing and releasing of the contacts result in power on and off of the circuit.

Fig 5: Internal Structure of relay

**Schematic:**

VCC and RY-VCC are also the power supply of the relay module. When you need to drive a large power load, you can take the jumper cap off and connect an extra power to RY-VCC to supply the relay; connect VCC to 5V of the MCU board to supply input signals. NOTES: If you want complete optical isolation, connect "Vcc" to Arduino +5 volts but do NOT connect Arduino Ground. Remove the Vcc to JD-Vcc jumper. Connect a separate +5 supply to "JD-Vcc" and board Gnd. This will supply power to the transistor drivers and relay coils. If relay isolation is enough for your application, connect Arduino +5 and Gnd, and leave Vcc to JD-Vcc jumper in place.

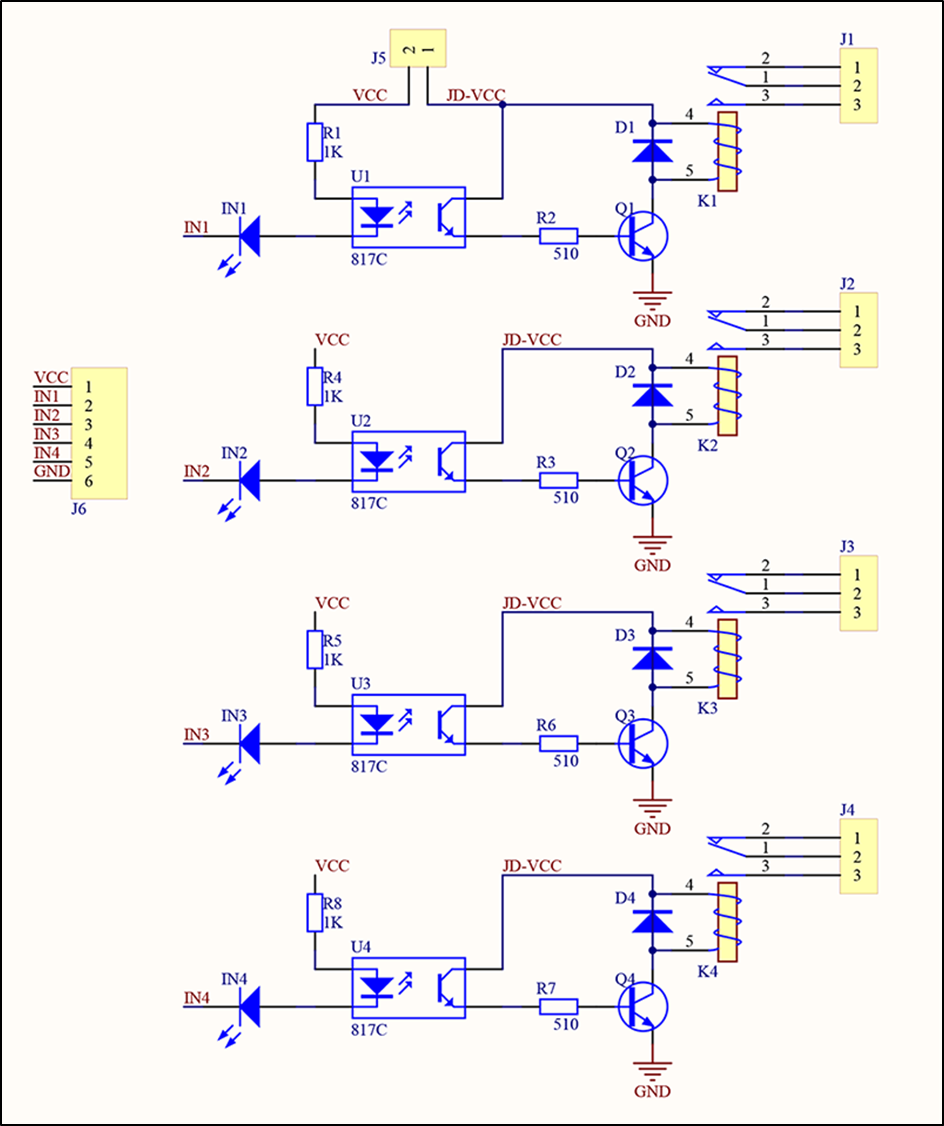


Fig 6: Schematic of relay module

***3.1.4 Push Buttons:***

As the above all hardware parts/modules are used in the project the four SPDT push buttons are also interfaced to the Node MCU ESP8266 Wi-Fi module.

IoT using i.e. by using Google Assistant and Blynk Cloud the all appliances connected at relay can be controlled only by virtually But, can’t be controlled physically either one has to turn off whole circuitry. So, the four Push Buttons are used here which and control each and every appliance physically.

By using this each and every person can perfectly use the system by pressing buttons, giving voice commands to Google Assistant and using blynk app. So, the knowledge of smartphone is not necessary for the person operating system.

Anyone from children’s, Adults up to old age persons can easily handle the appliances of their homes and offices.



Fig 7: Push Button

***3.2 Software Section***

It mainly contains following software and servers:

1. Internet of Things (IOT)
2. Blynk App and Blynk Server
3. Arduino IDE
4. IFTTT
5. Google Assistant

***Block diagram:***

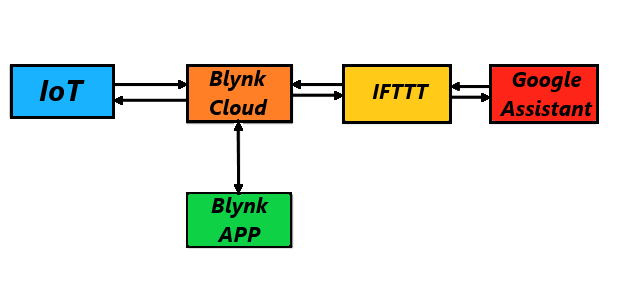


Fig 8: Block diagram of software section

***3.2.1 Internet of Things (IOT):***

The IOT device (having esp8266 PCB designed with numerous sensing element connected) is connected to Wi-fi or router in order that device is access with the assistance of web. The IOT device is Wi-Fi enabled and remotely governable employing a mobile app or by browsing to your account on the remainder API web site.

By victimization the mechanical man software package the module is programmed so it's joined with bylnk application by making a bylnk cloud network with the assistance of address code. The cloud may be a key enabler for the IoT.

***3.2.2 Blynk App and Blynk Server:***

Blynk is application software used to interact with the IOT based modules and devices such as ESP8266 Node MCU, Arduino, Raspberry Pi and similar development boards.

**How Blynk Works**

Blynk was designed for the Internet of Things. It can control hardware remotely, it can display sensor data, it can store data, visualize it and do many other cool things.

There are three major components in the platform:

* **Blynk App**

Allows to you create amazing interfaces for your projects using various widgets we provide.

* **Blynk Server**

Responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your [private Blynk server](https://docs.blynk.cc/#blynk-server) locally. It’s open-source, could easily handle thousands of devices and can even be launched on a Raspberry Pi.

* **Blynk Libraries**

For all the popular hardware platforms - enable communication with the server and process all the incoming and outcoming commands.

In short, every time you press a Button in the Blynk app, the message travels to the Blynk Cloud, where it magically finds its way to your hardware. It works the same in the opposite direction and everything happens in a blynk of an eye.

**Features**

* Similar API & UI for all supported hardware & devices
* Connection to the cloud using:
  + Wi-Fi
  + Bluetooth and BLE
  + Ethernet
  + USB (Serial)
  + GSM
* Set of easy-to-use Widgets
* Direct pin manipulation with no code writing
* Easy to integrate and add new functionality using virtual pins
* History data monitoring via SuperChart widget
* Device-to-Device communication using Bridge Widget
* Sending emails, tweets, push notifications, etc.

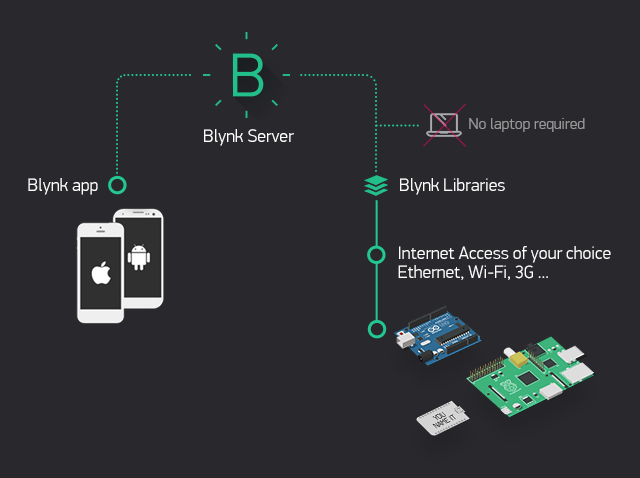
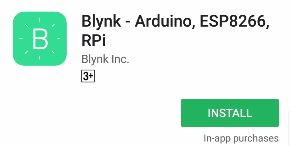


Fig 9: Virtual Blynk interface

**Steps to Configure Blynk App:**

1. Download the Blynk App from play store.
2. Create a Blynk Account by using E-mail.
3. Create a New Project.
   1. Choose Your Hardware

Here select Node MCU in Hardware

and Wi-Fi as connection type.

* 1. Auth Token

**Auth Token** is a unique identifier which is needed to connect your hardware to your smartphone. Every new project you create will have its own Auth Token. You’ll get Auth Token automatically on your email after project creation. You can also copy it manually. Click on devices section and selected required device. Don’t share your Auth Token with anyone.

Now press the **“Create”** button.

1. Add Four buttons widget for each relay.
   1. Click on Button 1 (For Light)

**PIN** – **D1** (**D** - stands for **D**igital).

As to drive Relay 1 which is connected at pin D1 of NodeMCU.

* 1. Click on Button 2 (For Fan)

**PIN** – **D2** (**D** - stands for **D**igital).

As to drive Relay 2 which is connected at pin D2 of NodeMCU.

* 1. Click on Button 3 (For socket 1)

**PIN** – **D3** (**D** - stands for **D**igital).

As to drive Relay 3 which is connected at pin D3 of NodeMCU.

1. Click on Button 4 (For socket 2)

**PIN** – **D4** (**D** - stands for **D**igital).

As to drive Relay 4 which is connected at pin D4 of NodeMCU.

1. Add notification to get notification when condition satisfied.
2. Click on Run.



***3.2.3 Arduino IDE***

The Arduino IDE software is used to write the required code for programming of Node MCU ESP8266 compile it and to upload it on the Node MCU board.

**Steps for Arduino IDE:**

1. Open Arduino IDE.
2. Click on File → New.
3. Click on Tools → Boards → NodeMCU 1.0 (ESP 12E module) → Upload speed- 115200 → Select Port where NodeMCU is connected.
4. Write following program/code.
5. Save the code.
6. Compile the code.
7. Upload into NodeMCU. After uploading code open Serial Monitor, The IP address and connection status of current network is displayed.

***3.2.4 Programming:***

#define BLYNK\_PRINT Serial // Comment this out to disable prints and save space

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>

// You should get Auth Token in the Blynk App.

// Go to the Project Settings (nut icon).

char auth[] = “Enter Auth Code HERE”; //Enter the Auth code which was send by Blink

// Your WiFi credentials.

// Set password to "" for open networks.

char ssid[] = “SSID”; //Enter your WIFI Name

char pass[] = “PASSWORD”; //Enter your WIFI Password

// Set your LED and physical button pins here

const int ledPin1 = 5;

const int ledPin2 = 4;

const int ledPin3 = 0;

const int ledPin4 = 2;

const int btnPin1 = 14;

const int btnPin2 = 12;

const int btnPin3 = 13;

const int btnPin4 = 3;

BlynkTimer timer;

void checkPhysicalButton();

int led1State = LOW;

int btn1State = HIGH;

int led2State = LOW;

int btn2State = HIGH;

int led3State = LOW;

int btn3State = HIGH;

int led4State = LOW;

int btn4State = HIGH;

// Every time we connect to the cloud...

BLYNK\_CONNECTED()

{

// Request the latest state from the server

Blynk.syncVirtual(V5);

Blynk.syncVirtual(V4);

Blynk.syncVirtual(V0);

Blynk.syncVirtual(V2);

// Alternatively, you could override server state using:

//Blynk.virtualWrite(V5, led1State);

//Blynk.virtualWrite(V4, led2State);

//Blynk.virtualWrite(V0, led3State);

//Blynk.virtualWrite(V2, led4State);

}

// When App button is pushed - switch the state

BLYNK\_WRITE(V5)

{

led1State = param.asInt();

digitalWrite(ledPin1, led1State);

}

BLYNK\_WRITE(V4)

{

led2State = param.asInt();

digitalWrite(ledPin2, led2State);

}

BLYNK\_WRITE(V0) {

led3State = param.asInt();

digitalWrite(ledPin3, led3State);

}

BLYNK\_WRITE(V2) {

led4State = param.asInt();

digitalWrite(ledPin4, led4State);

}

void checkPhysicalButton()

{

if (digitalRead(btnPin1) == LOW)

{

// btn1State is used to avoid sequential toggles

if (btn1State != LOW)

{

// Toggle LED state

led1State = !led1State;

digitalWrite(ledPin1, led1State);

// Update Button Widget

Blynk.virtualWrite(V5, led1State);

}

btn1State = LOW;

}

else

{

btn1State = HIGH;

}

if (digitalRead(btnPin2) == LOW)

{

// btnState is used to avoid sequential toggles

if (btn2State != LOW)

{

// Toggle LED state

led2State = !led2State;

digitalWrite(ledPin2, led2State);

// Update Button Widget

Blynk.virtualWrite(V4, led2State);

}

btn2State = LOW;

}

else

{

btn2State = HIGH;

}

if (digitalRead(btnPin3) == LOW)

{

// btnState is used to avoid sequential toggles

if (btn3State != LOW)

{

// Toggle LED state

led3State = !led3State;

digitalWrite(ledPin3, led3State);

// Update Button Widget

Blynk.virtualWrite(V0, led3State);

}

btn3State = LOW;

}

else

{

btn3State = HIGH;

}

if (digitalRead(btnPin4) == LOW)

{

// btnState is used to avoid sequential toggles

if (btn4State != LOW)

{

// Toggle LED state

led4State = !led4State;

digitalWrite(ledPin4, led4State);

// Update Button Widget

Blynk.virtualWrite(V2, led4State);

}

btn4State = LOW;

}

else

{

btn4State = HIGH;

}

}

void setup()

{

// Debug console

Serial.begin(9600);

Blynk.begin(auth, ssid, pass);

// You can also specify server:

//Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 8442);

//Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8442);

pinMode(ledPin1, OUTPUT);

pinMode(btnPin1, INPUT\_PULLUP);

digitalWrite(ledPin1, led1State);

pinMode(ledPin2, OUTPUT);

pinMode(btnPin2, INPUT\_PULLUP);

digitalWrite(ledPin2, led2State);

pinMode(ledPin3, OUTPUT);

pinMode(btnPin3, INPUT\_PULLUP);

digitalWrite(ledPin3, led3State);

pinMode(ledPin4, OUTPUT);

pinMode(btnPin4, INPUT\_PULLUP);

digitalWrite(ledPin4, led4State);

// Setup a function to be called every 100 ms

timer.setInterval(500L, checkPhysicalButton);

}

void loop()

{

Blynk.run(); // Initiates Blynk

timer.run(); // Initiates SimpleTimer

}

***3.2.5 IFTTT (“if this, then that”):***

IFTTT is a web service which allows other services to be programmed by means of simple conditional Statements called recipes.

IFTTT derives its name from the programming conditional statement “if this, then that.” IFTTT is both a website and a mobile app that launched in 2010 and has the slogan "Put the Internet to work for you". The idea is that you use IFTTT to automate everything from your favourite apps and websites to app-enabled accessories and smart devices. What the company provides is a software platform that connects apps, devices and services from different developers in order to trigger one or more automations involving those apps, devices and services. Here, IFTTT application is used to bridge the gap between the Google Assistant commands and the Blynk app.

Let’s look at what IFTTT comprises of as per their website information: -

**Channels**

Channels are the basic building blocks of IFTTT. Each Channel has its own Triggers and Actions

**Triggers**

The ‘this’ part of a Recipe is a Trigger. Some example Triggers are “I’m tagged in a photo on Facebook” or “I check in on Foursquare.”

**Actions**

The ‘that’ part of a Recipe is an Action. Some example Actions are “send me a text message” or “create a status message on Facebook.”

**Ingredients**

Pieces of data from a Trigger are called Ingredients. For example, the Ingredients of an Email Trigger could be: subject, body, attachment, received date, and the sender’s address.

**Recipes**

Personal Recipes are a combination of a Trigger and an Action from your active Channels

**On / Off**

Personal Recipes can be turned on and off. When turned back on, they pick up as if you had just created them.

**Polling Period**

Most Personal Recipes check for new Trigger data every 15 minutes, some are even faster

Let’s look at a process of creating an IFTTT service.

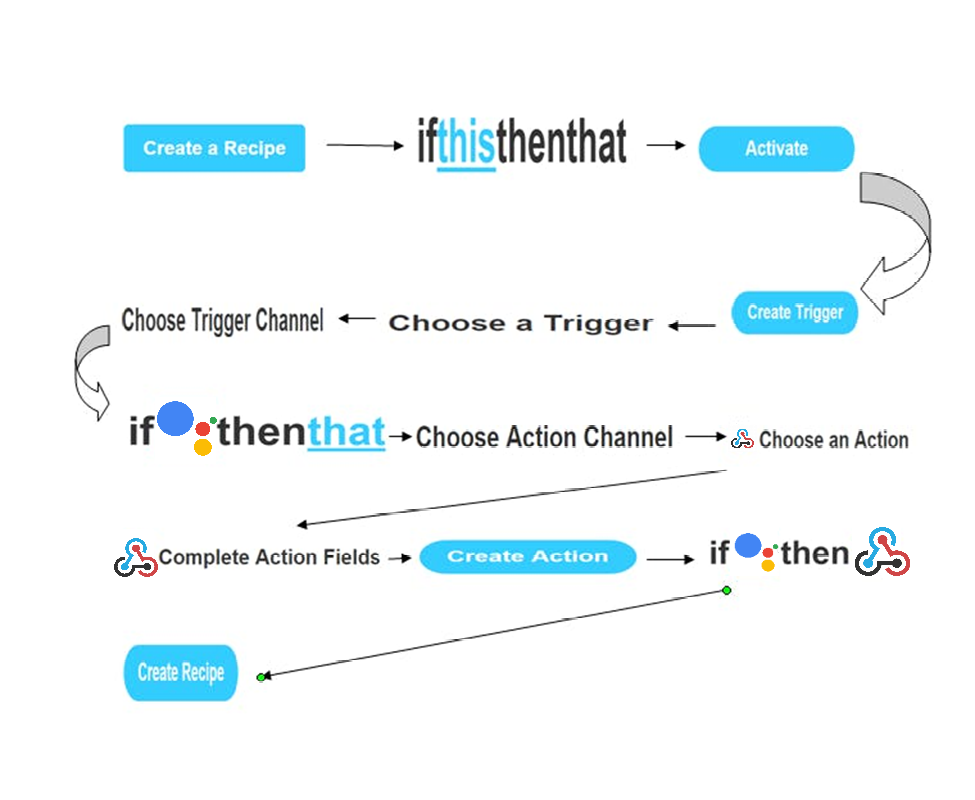


Fig 10: Virtual IFTTT interface

**Creating an IFTTT service:**

1. Go to IFTTT webpage or Download App from play store.
2. Register to IFTTT service with E-mail and if already register then login.
3. Click on My Applet → New Applet.
4. Click on ‘This’ which is highlighted then select ‘Google Assistant’.
5. Type the command it should control the appliance/relay.
6. Type the response command from Google Assistant.
7. Click on ‘That’ which is highlighted then select ‘Webhooks’.

**Webhooks** will allow us to send commands to the Blynk Server. Now, in the URL we type the IP address of the Blynk server followed by the Authentation token sent by the Blynk and then the pin number of the microcontroller to which the device to be controlled is connected. The URL should be in the following format:

***http://188.166.206.43/AuthToken/pin/CorrespondingDigitalPinNo***

1. Select ‘PUT’ in methods.
2. Select ‘Application/JSON’ as content type.
3. Write [“1”] to turn ON and [“0”] to turn OFF in body.
4. Similarly create the Applet for each of the condition of all relay/appliance.

***3.2.6 GOOGLE ASSISTANT:***

Google Assistant is an [artificial intelligence](https://en.wikipedia.org/wiki/Artificial_intelligence)-powered [virtual assistant](https://en.wikipedia.org/wiki/Virtual_assistant) developed by [Google](https://en.wikipedia.org/wiki/Google) that is primarily available on [mobile](https://en.wikipedia.org/wiki/Mobile_device) and [smart home](https://en.wikipedia.org/wiki/Home_automation) devices. Unlike the company's previous virtual assistant, [Google Now](https://en.wikipedia.org/wiki/Google_Now), Google Assistant can engage in two-way conversations.

The Google Assistant is a software which allows its users to control all the apps in their device to be controlled directly through it. It allows the users to control and command most of the apps in their devices using voice commands. This provides more convenience to the people as they only have to command the google assistant thorough voice command.



**4. SYSTEM OPERATION**

***4.1 Overview of System Operation:***

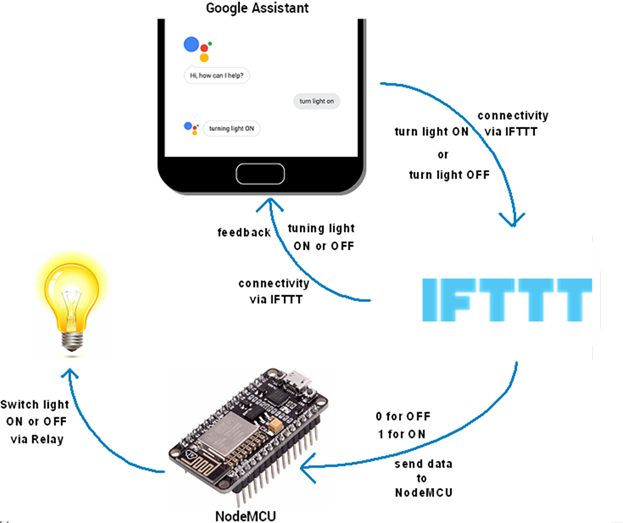


Fig 11: System Operating Cycle

***4.2 Working:***

The IOT device (having esp8266 PCB with numerous sensing element connected) is connected to Wi-fi or router in order that device is access with the assistance of web. By victimization the mechanical man software package the module is programmed so it's joined with bynk application by making a bylnk cloud network with the assistance of address code. The cloud may be a key enabler for the IoT. The IOT device is Wi-Fi enabled and remotely governable employing a mobile app or by browsing to your account on the remainder API web site. Further, this IOT devices and bylnk application is joined by Google. By loading the information as per our demand in google and chatbot is formed. Chatbots are designed on internet platforms that support multiple completely different chatbot applications. In figure 3, node MCU ESP8266 in built Wi-Fi modules connected with relay, Temperature sensor and various loads, and with power supply. The 5 V power supply is provided to node MCU with the help of voltage regulator.

The ESP8266 is a Wi-Fi module. The block diagram above shows the electrical systems that are controlled with IOT system by using wireless network. The devices are connected to the router in the house or using its own data connection (3G or 4G) network. Next, human will send information to switch ON the appliances in their home or building or industry through router and router will send the information to the Arduino. After that, Arduino will analyze the data with helping of Wi-Fi serial transceiver module (ESP8266) to communicate with the relay circuits. Then, after the relay circuit received the data on which switch going to be ON, the data will send to the appliances. The important thing to be highlight is the IoT device was connected wired with the relay circuit. Hence, the relay itself will communicate with the Arduino and the ESP 8266 module.

In this project node MCU is used as the main component. As a C program has been coded into Arduino software, it will determine what types of electrical appliances that can be controlled. Web-server is running on using the Wi-Fi module ESP8266. This Wi-Fi module has capability to be used both either as a client or a server. Next, the communication between electrical appliances systems for example light or fan to the Espresso Lite v2.0 is started. Since Arduino serial module already supports a TCP/IP stack, so it focuses on implementing software to connect it to the remote users. During the configuration stage, the Wi-Fi module establishes connection with Local Area Network (LAN) using a static IP address. To optimize the process of connection, the static IP address that is used rather than acquiring an IP via Dynamic Host Configuration Protocol (DHCP). By doing so, the control systems that connected with Wi-Fi serial module can be control either using web based or application in smart phone.

In this project four modules are present; the working of each module is different. Means each module perform different task as compare to other.

When the Voice or typed command is given to Google Assistant, For Ex. “Turn ON first lamp” or “First lamp ON” or “Lamp ON” then the given Voice command is compared with the command specified in IFTTT for specific operation.

Then IFTTT sends back feedback response associated with given command here i.e. “Turning ON first lamp”. And it triggers the event mentioned in Webhooks applet and send the link which has Blynk server IP address, Auth code and pin configuration i.e. making pin of Node MCU ‘HIGH’ or ‘LOW’.

The four push buttons switches are provided to the Node MCU ESP8266 to change the state of the pins which are connected to the relays. Whenever the button is pressed it triggers the pin and connect it to ground and the status of the pin changes to opposite of previous status. If the pin is high and the button pressed then pin becomes low. This is done physically therefore there were no need of smartphone while using push buttons.

Following are step user should do to ON and OFF appliances:

1. To Turn ON lamp:
   1. Voice command- i. Turn ON first lamp.

ii. First lamp ON.

iii. Lamp ON.

* 1. For manual operation use buttons provided in Blynk App.
  2. By pressing the push button 1.

(Blynk App shows the status of appliance)

1. To Turn OFF lamp:
   1. Voice command- i. Turn OFF first lamp.

ii. First lamp OFF.

iii. Lamp OFF.

* 1. Also, by using Blynk App.
  2. By pressing the push button 1.

1. To Turn ON fan:
   1. Voice command- i. Turn ON fan.

ii. Fan ON.

* 1. Also, by using Blynk App.
  2. By pressing the push button 2.

1. To Turn OFF fan:
   1. Voice command- i. Turn OFF fan.

ii. Fan OFF.

* 1. Also, by using Blynk App.
  2. By pressing the push button 2.

1. To Turn ON appliance at Socket 1:
   1. Voice command- i. Turn ON socket 1.

ii. Socket 1 ON.

* 1. Also, by using Blynk App.
  2. By pressing the push button 3.

1. To Turn OFF appliance at Socket 1:
   1. Voice command- i. Turn OFF socket 1.

ii. Socket 1 OFF.

* 1. Also, by using Blynk App.
  2. By pressing the push button 3.

1. To Turn ON appliance at Socket 2:
   1. Voice command- i. Turn ON socket 2.

ii. Socket 2 ON.

* 1. Also, by using Blynk App.
  2. By pressing the push button 4.

1. To Turn OFF appliance at Socket 2:
   1. Voice command- i. Turn OFF socket 2.

ii. Socket 1 OFF.

* 1. Also, by using Blynk App.
  2. By pressing the push button 4.

***4.3 Circuit Connections:***

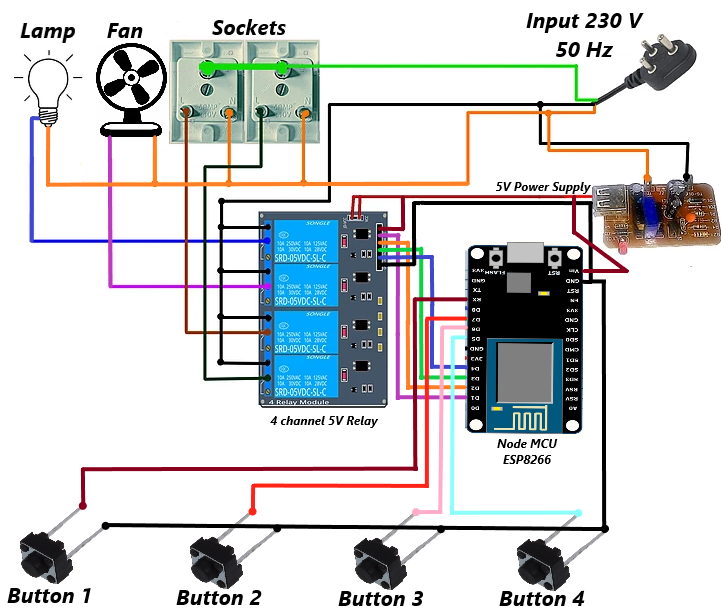
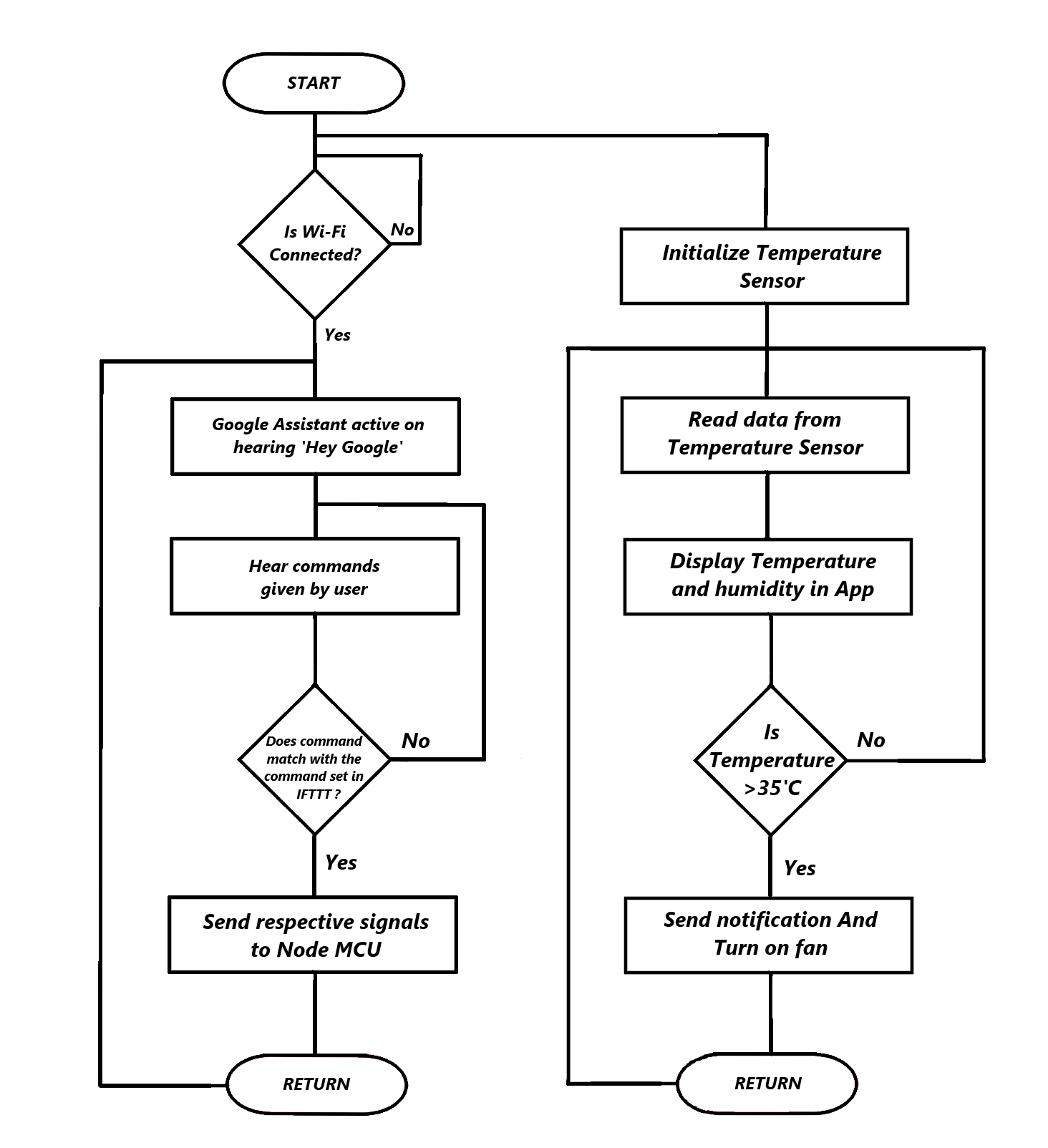
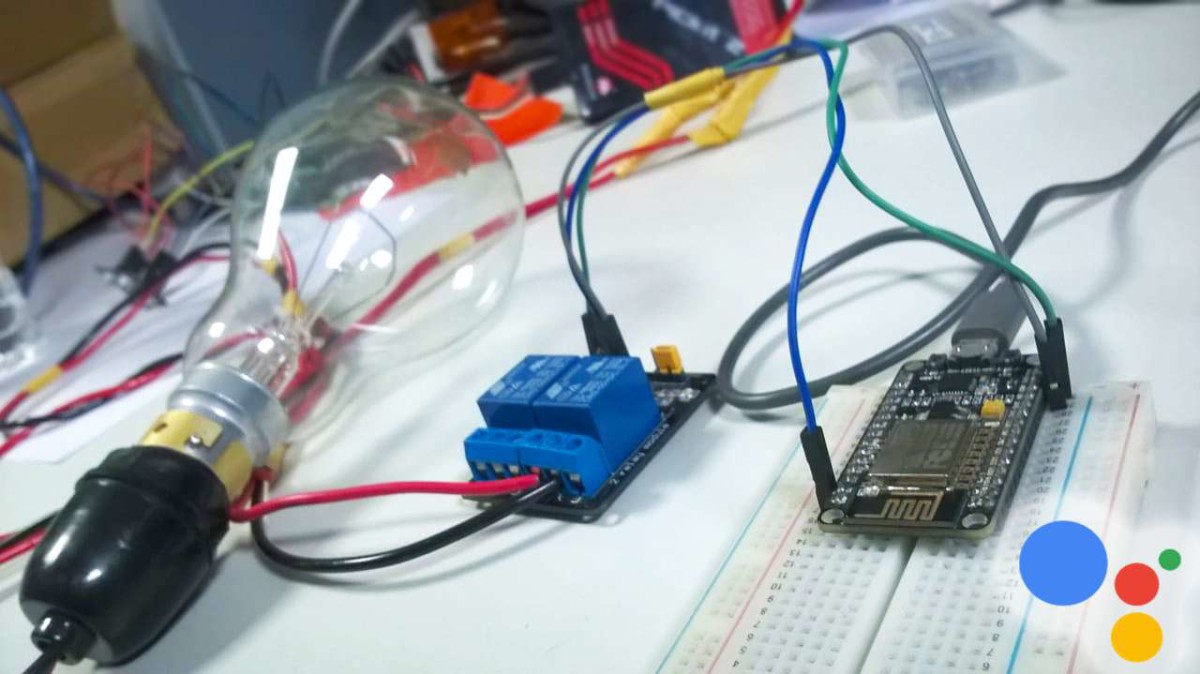
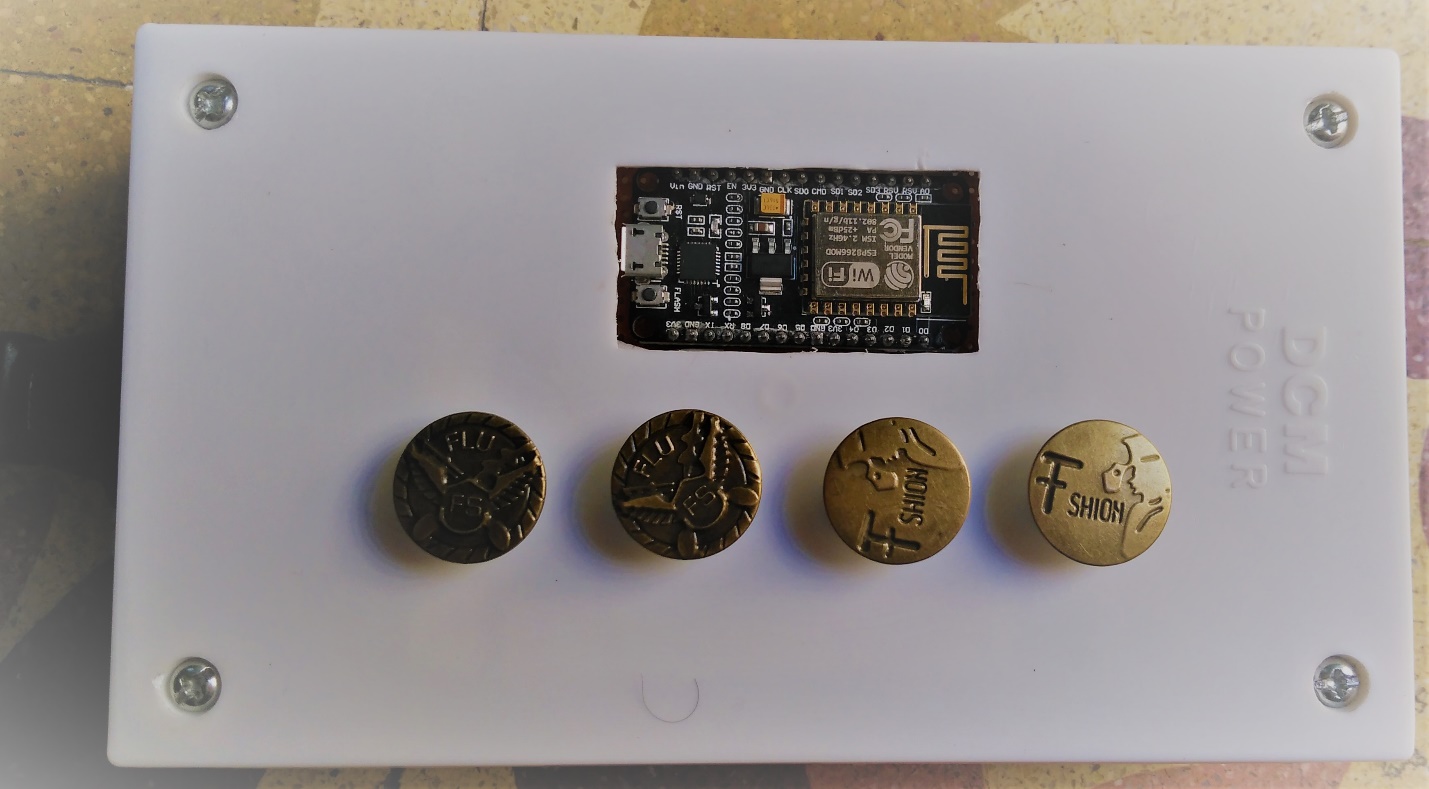
****

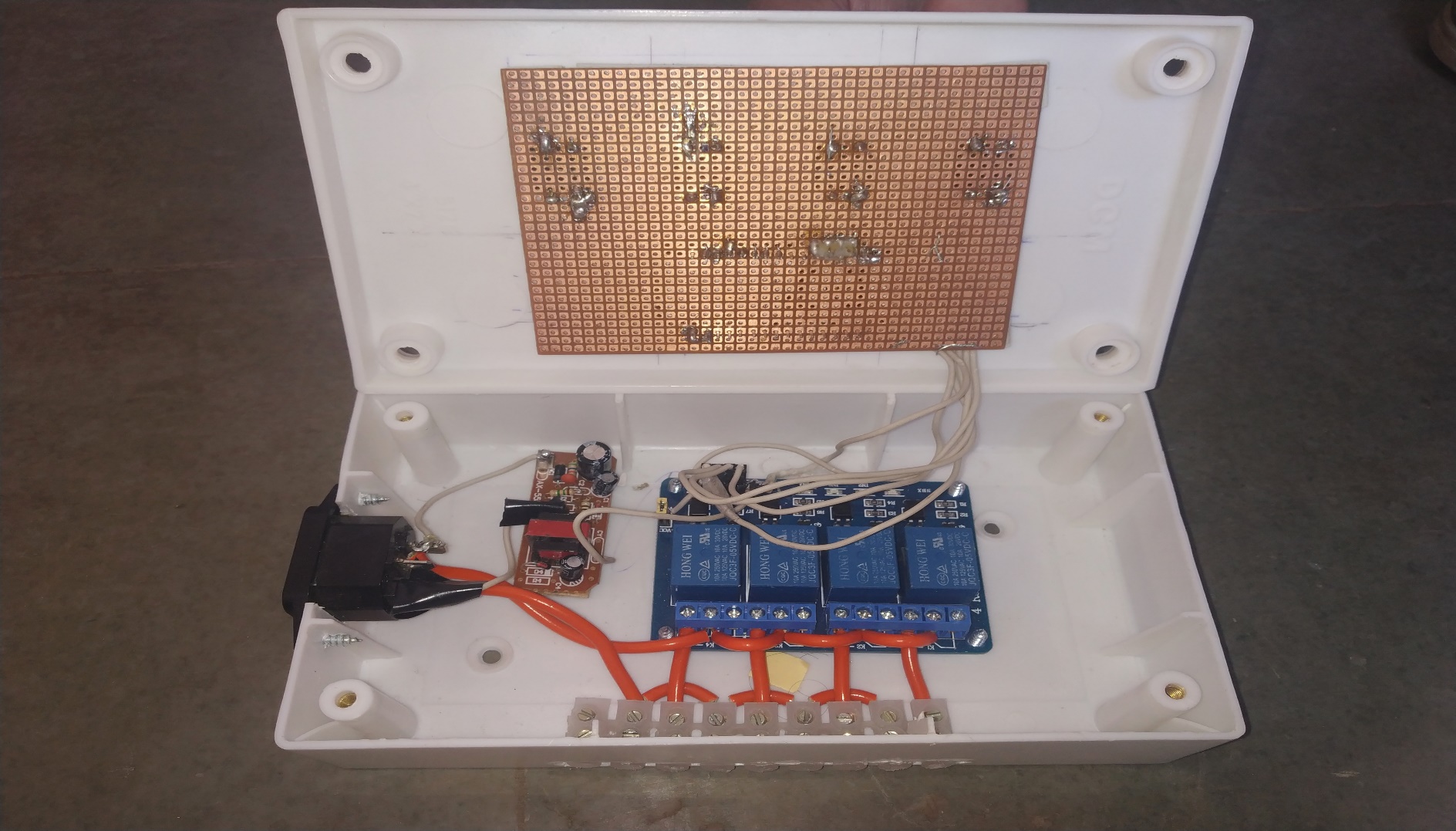
Fig 12: Circuit Connections

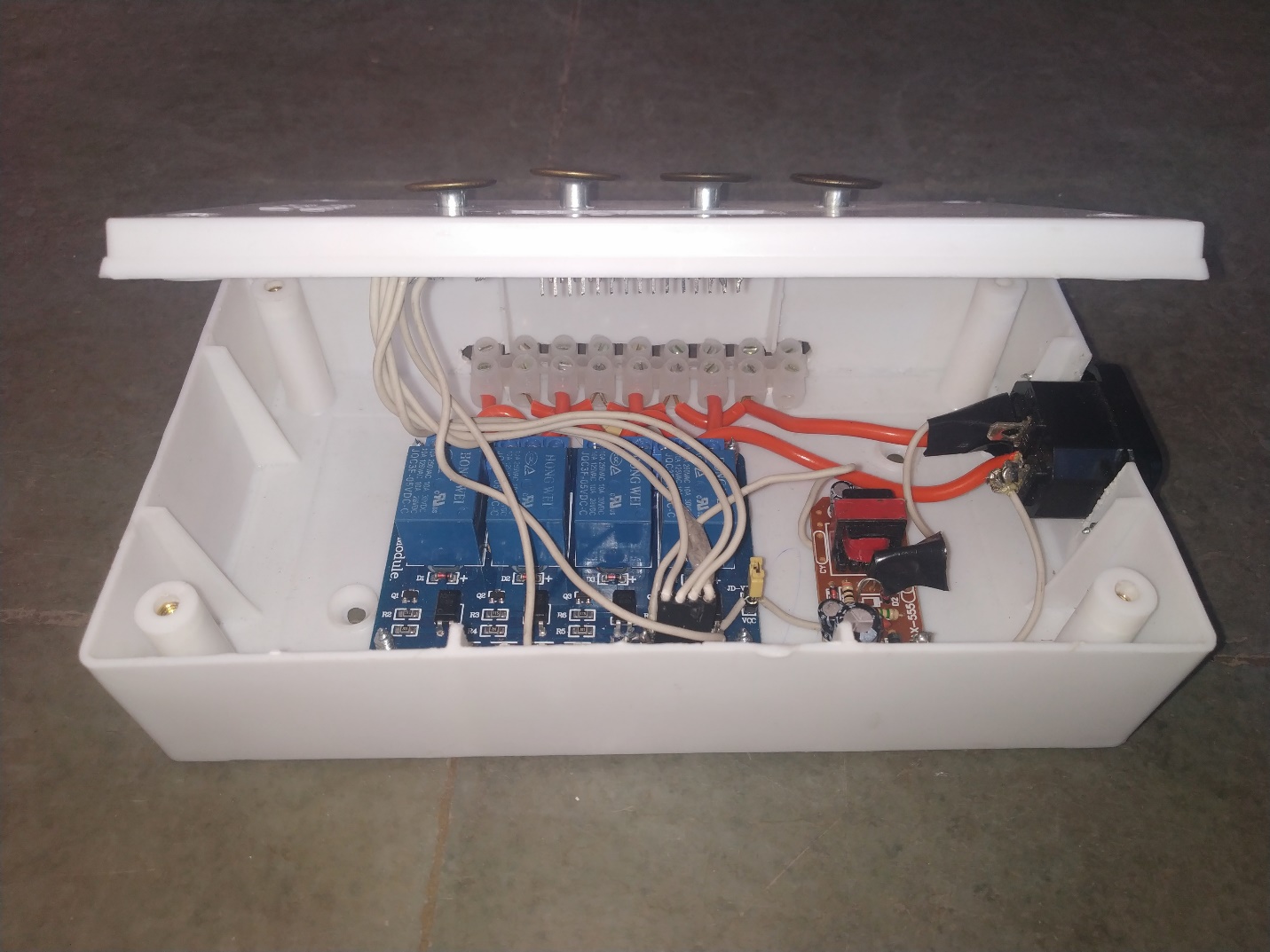
***4.4 Flow Chart:***











**5. ADVANTAGES**

Home automaton by using IoT has been around for several years. Earlier years of automation, it considered a luxury but as smart phones and tablets become increasingly common to every household, home automation is becoming easier to use and dramatically more affordable. Here are the top 5 advantages why you should install this IoT based system in your home.

**Safety**:

If the switch board in home is damaged because of some reasons, then the chances of getting shock is maximum and can be harmful to us. And if the IR remote based system is used then chances of getting shock are reduced to zero.

**Convenience**:

The ability to control everything with your fingertips is very convenient. You can control the loads by sitting on sofa by watching TV. And you can control everything with just touch of a finger. Also, you can control your home appliances from anywhere in the whole world.

**Saves Time**:

Since we are living in a very fast-paced environment, we don’t even have time to worry about our home. With this system, we can save time going back to the switch board and to switch ON/OFF or control it.

**Save Money:**

With the ability to control the light, or turning on/off on specific time will saves homeowner a great ton of money.

[**6. DISADVANTAGES**](http://powerfullsystems.com/who-we-help)

This IoT system is Reliable but There are some disadvantages also As, the IoT is used, the internet connection is must in this project whenever the connection is lost, we cannot access our system. As the system is a part of Internet it can be hacked.

**7. CONCLUSION**

Home automation system using IoT has been tested and successfully implemented. This system is highly reliable and efficient for the aged person and paralyzed person on the wheel chair who cannot reach the switch for the switching of on /off and controlling the device and are dependent on other this system has a wide scope development and modification.

Turning ON and OFF of home appliances are done by giving Voice Commands to Google Assistant simply using smartphone. Also, the temperature and humidity are displayed.

**8. FUTURE SCOPE**

As many GPIO pins of ESP8266 Node MCU is still unused, many sensors can be interface to make this system more advance and reliable or these pins can also be used to provide multi services to user. A little beyond this can also be thought of, where the device is instructed through the instructions sent in an SMS from a cellular phone

More devices can be simulated and timer could be set for automatic operation

Development of any technology never stops at a point. Today’s, we believe the future of home automation will very much ride the digital age and develop along with the computer and networking systems in the years to come.

Technologies like Google Glass, “IVEE” Android and Wi-Fi, Robotics will play a major role in the development of this field of automation. Artificial Intelligence, high speed internet smart phones and remote access to the housing system will lead to convenience and safety of house by security concerns will also arise. Home Automation is the gift of technology to the mankind. proper use and development in this field will fulfill our requirement and make our lives easy …!!!

**9. REFERENCE**

[1] Meng Kun. Smart home system design based on MSP430 [D]. Master's degree thesis, Suzhou: Suzhou University, 2014.05, 9-12.

[2] Liu Yan, Zhao Xumin, Jiang Jiawei. Development Status and Development Trend of Smart Home[J]. Household Appliances, 2016.07, (07): 17-19.

[3] Liu Miao. Research and Implementation of Intelligent Home Control System Based on Internet of Things Technology [D]. Master thesis, Lanzhou: Lanzhou Jiaotong University, 2015.10, 18-25.

[4] Wei Qingming. Design and implementation of smart home system based on wireless sensor network [D]. Master's thesis, Xi'an Xidian University, 2015.09, 5-6. [5] Chen Goujon, Chen Lei, Chen Xixime. Java 2 Programming Design Foundation [M]. Beijing: Tsinghua University Press, 2006.01, 206-210. [6] Xu Longie, Ying Wendi. Analysis and Application Practice of Smart Home Application Development[J]. Telecommunication Science, 2015.11, 31(11): 116-122.

* [www.circuitdigest.com](http://www.circuitdigest.com/)
* [www.microcontrollerslab.com](http://www.microcontrollerslab.com)
* [www.electronicsshub.org](http://www.electronicsshub.org/)
* [www.electronicsforu.com](http://www.electronicsforu.com/)
* [www.pantechsolutions.com](http://www.pantechsolutions.com)

