Smart Home Automation using Esp-32

A Major Project Report

Submitted in partial fulfillment of the requirement

for the degree of

Bachelor of Technology

In

Electronics & Communication Engineering

January – July 2023

Guided By
Prof. Mohit Pant

Submitted by
Chinmay Dubey
[0704ME191026]
Harshit Solanki
[0704EC191011]
Avani Parmar
[0704EC191003]

Department of Electronics and communication

Mahakal Institute of Technology, Ujjain

Affiliated to Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal

PROJECT APPROVAL SHEET

The project entitled “Smart Agriculture System” submitted by Chinmay Dubey, Harshit Solanki and Avani Parmar as partial fulfillment for the award of **Bachelor of Technology in Electronics &Communication Engineering** by Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal.

Internal Examiner	External Examiner
Date:	Date:

RECOMMENDATION

The project entitled “Smart Agriculture System” submitted by Chinmay Dubey,Harshit Parmar and Avani Parmar as partial is a satisfactory account of the bonafIde work done under our guidance is recommended towards partial fulfillment for the award of the **Bachelor of Technology in Electronics & Communication Engineering** from Mahakal Institute of Technology, Ujjain byRajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal.

Project Guide	Project Coordinator
Prof……………	Prof. Mohit Pant
Date:	Date:

Endorsed By

Head of Department of Electronics & Communication Engineering

Mahakal Institute of Technology, Ujjain

ACKNOWLEDGEMENT

The successful completion of this project is the results of dedicated efforts of many people and this report would be incomplete without giving due credit to them. This acknowledgement is small token of gratitude in recognition of their help in there endeavor.

I express my profound sense of gratitude to Mukesh Shukla, Director, MIT, Ujjain for providing their valuable guidance, spontaneous motivation and moral support in carrying out this work.

I owe special thanks to my esteemed guide Prof. Mohit Pant Dep't. Of Electronics & communication and Project Coordinator Prof.Mohit Pant involved right from the inception of ideas to the finalization of the work. They been a continuous source of information, without their guidance, suggestions and an urge to bring out the best would not have been possible.

I am thankful to Prof. Saurabh Gaur Head, Electronics and Communication Department, for providing necessary facilities and suggestions to carry out my project work. I am also thankful to all the renowned faculty members and friends for their valuable support.

Harshit Solanki

[0704EC191011]

Chinmay Dubey

[0704ME19102]

Avani Parmar

[0821EC191003]

TABLE OF CONTENT

ABSTRACT

In India now a days automation industary is growing rapidly , in many cities most of the houses are now powered with automated utilities such as automatic light bulbs,speaker,door lock system and etc. but still we don't have a universal product in market for automation realted problem so here with this project we are trying to build a universal circuit board that can be installed in any area ,room,hall or simply anywhere and with the help of our smartphone we can control all the stuff just by using a application or also with our voice using alexa and google assistant.

This project presents an idea for home automation using ESP32 with Blynk, IR remote and manual switch to control 8 relays with and without internet and monitor the real-time feedback in the Blynk app. The ESP32 is a development board that combines Wi-Fi and Bluetooth wireless capabilities, and it's dual-core. It supports a wide variety of peripherals such as capacitive touch, ADC, DAC, I2C, SPI, UART, I2S, PWM, and much more. With this home automation project, you can control & monitor the real-time feedback of the relays in the Google Home and Alexa App from anywhere in the world.

FIGURE INDEX

LIST OF ABBREVIATION

REST = REPRESENTATIONAL STATE TRANSFER

MQTT = MESSAGE QUEING TELEMETRY TRANSPORT

HTTP = HYPER TEXT PROTOCOL

DC = DIRECT CURRETN

LCD = LIQUID CRYSTAL DISPLAY

IOT = INTERNET OF THINGS

API = APPLICATION PROGRAMMING INTERFACE

GUI = GRAPHICAL USER INTERFACE

CHAPTER – 1

INTRODUCTION

1.1 INTRODUCTION –

Home automation has been a rapidly growing field in recent years, with the advent of the Internet of Things (IoT) enabling the integration of smart devices and appliances in our homes. This has paved the way for greater convenience, efficiency, and comfort in our daily lives. In this project, we will explore the implementation of an IoT-based home automation system that will allow users to control various aspects of their home remotely. This system will leverage the power of modern technology, including sensors, actuators, and wireless communication protocols, to automate tasks such as lighting, temperature regulation, and security. By the end of this project, we hope to have developed a functional and user-friendly system that can be easily deployed in any modern home.

There are various IoT devices and sensors that can be used for home automation and security. The basic ones are:

1. ***Wall switches.*** IoT-based wall switches can be mounted on the wall of a house like regular wall switches, but can be controlled remotely. Such IoT switches can also be wireless and can be connected to an IoT hub or router wirelessly.
2. ***Voltage sensors.*** IoT voltage sensors can monitor the supply voltage of the house and one can even monitor the same remotely using the Internet.
3. ***Energy monitors.*** IoT energy monitors are digital energy meters that can monitor the power consumption of the house, also helping one monitor the overall power consumption remotely.
4. ***Thermostats.*** IoT thermostats can be used to monitor the temperature of the house in real-time. Using such thermostat sensors can help monitor the temperature inside the house from anywhere using the Internet.
5. ***Smart door locks.*** Smart door locks are IoT door locks that can be controlled using a home security system and IoT OSS/app over the Internet. One can lock and unlock them remotely via the app.
6. ***Air-conditioners.*** Air-conditioning is a very important part of a house and controlling the air-conditioner remotely is a part of home automation.

7.Surveillance cameras. IoT cameras can be used for the surveillance of the house and they can give live video footage of the house remotely. Both indoor and outdoor cameras can be used to monitor the indoor and outdoor environments. Such a surveillance system can have features like motion detection and can alert whenever there is a security breach. Apart from the above IoT devices, many other devices can be installed in a house to improve the convenience and automation. Many IoT-based home appliances like washing machines, water heaters, refrigerators, dish washers, robot floor cleaners, etc can also be installed additionally to make your house fully automated.

1.2 PROBLEM SATEMETNT –

Despite the widespread availability of smart devices and appliances, the process of controlling and automating a home's various systems and devices can still be cumbersome and time-consuming. In addition, many home automation systems are often proprietary, with limited compatibility with other devices and systems, resulting in a lack of flexibility and interoperability. This can be a significant obstacle for homeowners seeking to achieve a fully integrated and seamless home automation experience. Furthermore, the current approaches to home automation often lack a unified,
centralized control system, with users often having to juggle multiple apps and interfaces to manage different devices and systems. This not only adds complexity to the user experience but can also lead to security vulnerabilities as these systems may not have robust security measures
in place. Therefore, there is a need for a comprehensive and flexible home automation system that allows users to control their homes' various systems and devices efficiently, securely, and seamlessly. This project aims to address these challenges by developing an IoT-based home automation system that integrates multiple systems and devices into a unified and user-friendly platform.

1.3 PROBLEM DEFINTION –

The ESP-32 based home automation project aims to solve the problem of inefficient and inconvenient home automation systems by utilizing the capabilities of the ESP-32 microcontroller. Despite the advancements in home automation technology, many systems still require manual input or use multiple proprietary interfaces, leading to a fragmented and cumbersome user experience. Additionally, many systems may lack flexibility, compatibility, or robust security measures, which can compromise user safety and privacy. This project seeks to provide a comprehensive and flexible solution to these issues by leveraging the powerful capabilities of the ESP-32 microcontroller. The project will aim to integrate multiple home automation systems, such as lighting, temperature control, and security, into a single platform that can be easily controlled and monitored. The ESP-32's built-in Wi-Fi and Bluetooth capabilities will allow for seamless communication between devices, enabling a more integrated and efficient system. Moreover, the project will incorporate advanced security measures to protect the system from unauthorized access and ensure user safety and privacy. Overall, the ESP-32 based home automation project aims to create a streamlined and secure home automation system that enhances the user experience and simplifies managem

1.4 EXISTING SYSTEM –

Home automation systems are software and hardware devices that allow users to control and monitor various aspects of their homes, such as lighting, climate, entertainment, security, and appliances. Home automation systems can be classified into three main types: power line based, wired or bus cable based, and wireless based. Power line based systems use the existing electrical wiring of the home to transmit signals between devices. Wired or bus cable based systems use dedicated cables to connect devices to a central controller. Wireless based systems use radio frequency, infrared, Wi-Fi, Bluetooth, ZigBee, Z-Wave, or other wireless protocols to communicate between devices. Some of the popular home automation platforms are Amazon Echo, Google Nest Hub, Wink Hub 2, Samsung SmartThings, and Apple HomeKit. These are just a few examples of the many different types of home automation systems available. The right one for you will depend on your specific needs and preferences.There are many different existing systems for home automation. Some popular options include:

• Smart home platforms like Amazon Alexa, Google Home, and Apple HomeKit, which allow you to control various devices and appliances in your home using voice commands or a mobile app.

• Smart thermostats like Nest and Ecobee, which allow you to control the temperature in your home remotely and create customized schedules.

• Smart lighting systems like Philips Hue and LIFX, which allow you to control the brightness and color of your lights using a mobile app or voice commands. • Smart security systems like Ring and ADT, which allow you to monitor your home and receive notifications when motion is detected.

CHAPTER – 2

LITERATURE REVIEW

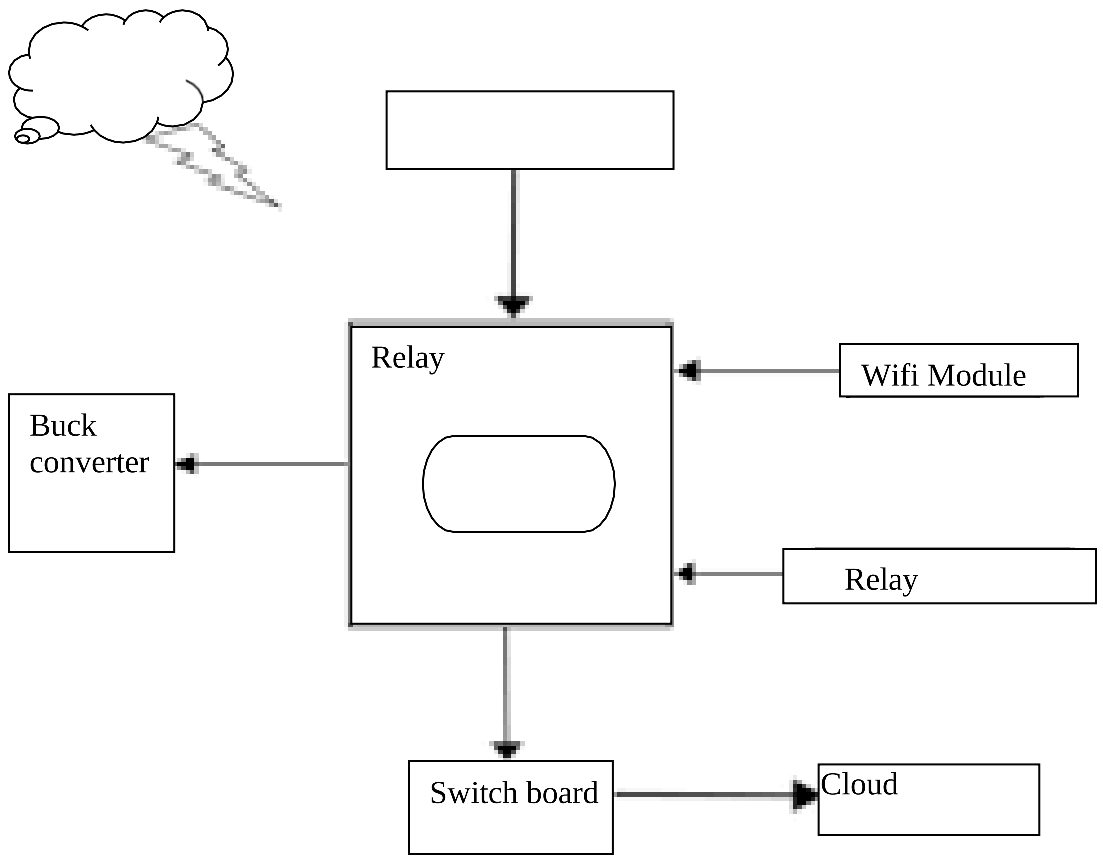
In literature survey we have come across various types of concepts related to our idea. In an IoT-based Home Automation System Using Wi-Fi Wireless Sensor Networks they focus on new human computer interaction has evolved from the Graphical User Interface (GUI) to the Conversational User Interface (CUI) and is able to be built on a mobile device and integrated with the social apps. It allows users to chat with Chatbot through the chat interface provided by the Line App, regardless of where users are. They can control the smart home system, security system, and alarm system at home to achieve integration of different systems as well as ameliorate the shortcomings of mobile apps of departed smart home. In another paper we have seen constant monitoring of people's behavior, activities are required for the purpose of protection and management of confidential data. In surveillance, installing and setup of CCTV camera systems becomes costly for normal residents and also system cannot inform the owner's automatically when the robbery happens.

The expeditiously growing internet has opened new horizons for development in various fields. The home automation industry has seen a brisk growth in the last few years. It has become a topic of interest of many people around the globe. Vishwateja Mudiam Reddy & Naresh Vinay in their paper "Internet of Things Enabled Smart Switch"designed a system which integrates the cloud and web app. With the help of flip-flops, logic gates and a processor, the switches could be controlled. The proposed model was intended for reducing the cost of these systems which was the main barrier in the wide adaptation of this technology. Khusvinder Gill & Shuang-Hua Yang created a common home gateway for ZigBee and Wi-Fi. This enabled remote control using a simple user interface. The system was cost effective and had good security inside the house. Salma and Dr. Radcliffe with an aim of increasing the popularity and reach of home automation designed a system that used the Novel Network Protocol. It gave the option of controlling the commercial devices through a mobile or laptop. An additional network device was used for remote access instead of a microcontroller. A flexible and simple system with an ability to integrate with very fewer efforts for off the shelf products was created by Carelin and I. Jacob Raglend. The system used ZigBee for home controlling and GSM for remote access. It did not provide any GUI and also it was prone to security threats as anyone could access the system. Rozita Teymourzadeh, Salah Addin Ahmed designed a GSM based system for home automation. Using the GSM protocol, it became possible to access the system by using the Short Message System (SMS). The system also gave feedback to the user about the current status of any desired device we want creating a levarage for user to interact with it easily.

CHAPTER – 3

COMPONENETS OF PROJECT

3.1 BLOCK DIAGRAM –



3.2 LIST OF HARDWARE REQUIREMENTS –

S. NO.	COMPONENT NAME	DESCRIPTION	QUANTITY
1.	Nodemcu ESP-32	All in one microcontroller with Wifi platform.	1
2.	Buck converter	For getting desired voltage.	1
3.	Relay Module	As a switch .	6
4. .	Charger module	For converting AC to stable 5v .	1
5.	Breadboard	Used for building temporary Circuit.	1
6.	Connecting wire	On board soldered wire or rainbow wire for switchboard.	--

Table 3.1

3.3 NODEMCU ESP8266 –

ESP32 is the SoC (System on Chip) microcontroller which has gained massive popularity recently. Whether the popularity of ESP32 grew because of the growth of IoT or whether IoT grew because of the introduction of ESP32 is debatable. If you know 10 people who have been part of the firmware development for any IoT device, chances are that 7−8 of them would have worked on ESP32 at some point.

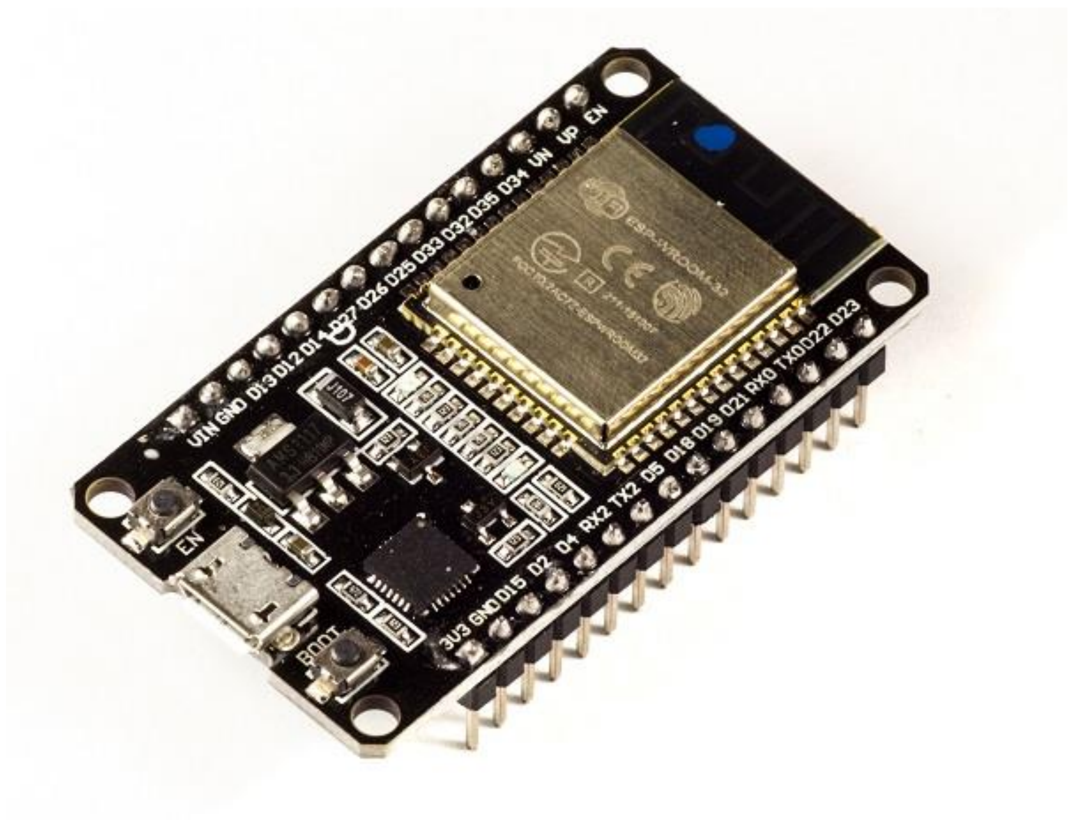

ESP-WROOM-32
EN
BOOT
VIN GND D13 D12 D14 D27 D26 D25 D33 D32 D35 D34 VN VP EN
3V3 GND D15 D2 D4 RX2 TX2 D5 D18 D19 D21 RX0 TX0 D22 D23

Fig 1.1

The specs listed below belong to the ESP32 WROOM 32 variant.−

- Integrated Crystal− 40 MHz
- Module Interfaces− UART, SPI, I2C, PWM, ADC, DAC, GPIO, pulse counter, capacitive touch sensor

- Integrated SPI flash− 4 MB
- ROM− 448 KB (for booting and core functions)
- SRAM− 520 KB
- Integrated Connectivity Protocols− WiFi, Bluetooth, BLE

- On−chip sensor− Hall sensor
- Operating temperature range− −40 − 85 degrees Celsius
- Operating Voltage− 3.3V
- Operating Current− 80 mA (average)

With the above specifications in front of you, it is very easy to decipher the reasons for ESP32's popularity. Consider the requirements an IoT device would have from its microcontroller (µC). the major operational blocks of any IoT device are sensing, processing, storage, and transmitting. Therefore, to begin with, the µC should be able to interface with a variety of sensors. It should support all the common communication protocols required for sensor interface: UART, I2C, SPI. It should have ADC and pulse counting capabilities. ESP32 fulfills all of these requirements. On top of that, it also can interface with capacitive touch sensors. Therefore, most common sensors can interface seamlessly with ESP32.

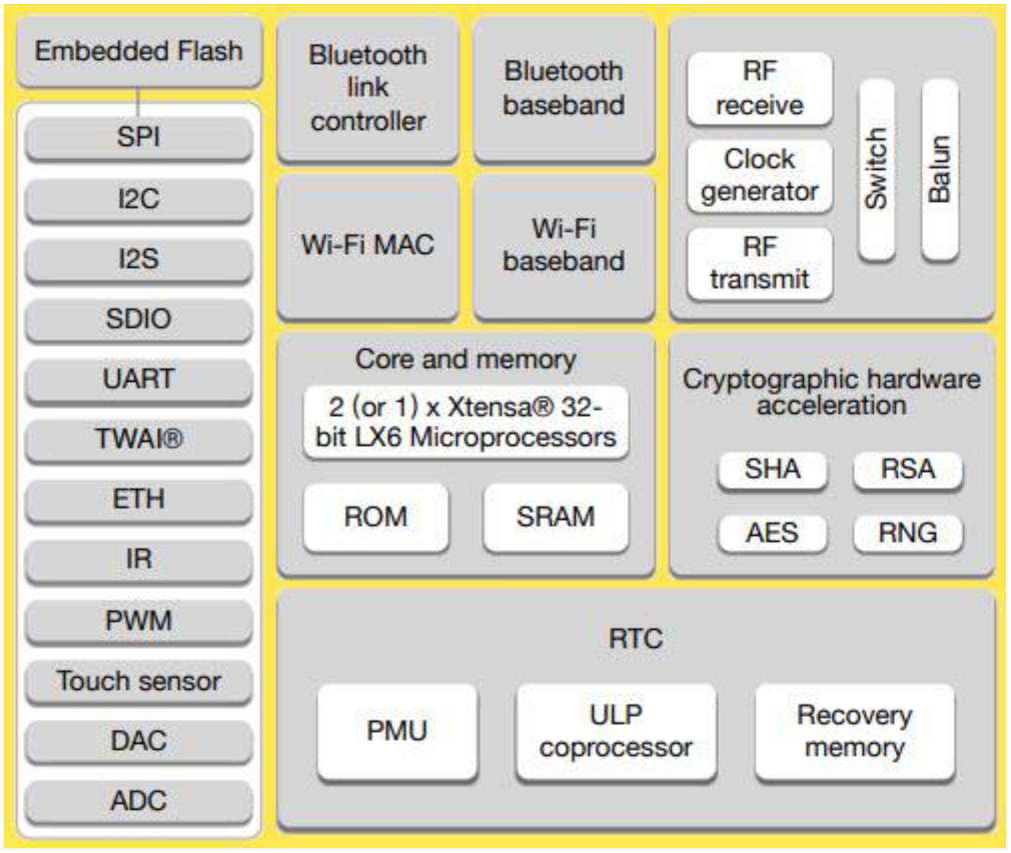
Secondly, the µC should be able to perform basic processing of the incoming sensor data, sometimes at high speeds, and have sufficient memory to store the data. ESP32 has a max operating frequency of 40 MHz, which is sufficiently high. It has two cores, allowing parallel processing, which is a further add-on. Finally, its 520 KB SRAM is sufficiently large for processing a large array of data onboard. Many popular processes and transforms, like FFT, peak detection, RMS calculation, etc. can be performed onboard ESP32. On the storage front, ESP32 goes a step ahead of the conventional microcontrollers and provides a file system within the flash. Out of the 4 MB of onboard flash, by default, 1.5 MB is reserved as SPIFFS (SPI Flash File System). Think of it as a mini−SD Card that lies within the chip itself. You can not only store data, but also text files, images, HTML and CSS files, and a lot more within SPIFFS. People have displayed beautiful Webpages on WiFi servers created using ESP32, by storing HTML files within SPIFFS.

Finally, for transmitting data, ESP32 has integrated WiFi and Bluetooth stacks, which have proven to be a game-changer. No need to connect a separate module (like a GSM module or an LTE module) for testing cloud communication. Just have the ESP32 board and a running WiFi, and you can get started. ESP32 allows you to use WiFi in Access Point as well as Station Mode. While it supports TCP/IP, HTTP, MQTT, and other traditional communication protocols, it also supports HTTPS. Yep, you heard that right. It has a crypto−core or a crypto-accelerator, a dedicated piece of hardware whose job is to accelerate the encryption process. So you cannot only communicate with your web server, you can do so securely. BLE support is also critical for several applications. Of course, you can interface LTE or GSM or LoRa modules with ESP32. Therefore, on the 'transmitting data' front as well, ESP32 exceeds expectations.

With so many features, ESP32 would be costing a fortune, right? That's the best part. ESP32 dev modules cost in the ballpark of ₹ 500. Not only that, the chip dimensions are quite small (25 mm x 18 mm, including the antenna area), allowing its use in devices requiring a very small form factor.

Finally, ESP32 can be programmed using the Arduino IDE, making the learning curve much less steep.

The block diagram of components inside Esp-32 devkit module -



The pinout diagram of Esp-32 devKit module-

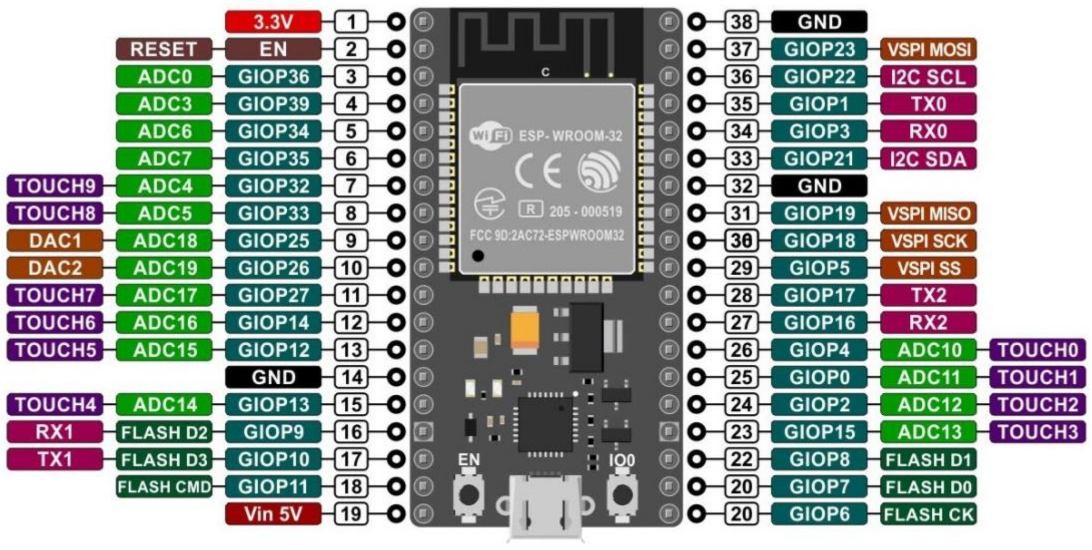

3.3V 1
RESET EN 2
ADC0 GIOP36 3
ADC3 GIOP39 4
ADC6 GIOP34 5
ADC7 GIOP35 6
TOUCH9 ADC4 GIOP32 7
TOUCH8 ADC5 GIOP33 8
DAC1 ADC18 GIOP25 9
DAC2 ADC19 GIOP26 10
TOUCH7 ADC17 GIOP27 11
TOUCH6 ADC16 GIOP14 12
TOUCH5 ADC15 GIOP12 13
GND 14
TOUCH4 ADC14 GIOP13 15
RX1 FLASH D2 GIOP9 16
TX1 FLASH D3 GIOP10 17
FLASH CMD GIOP11 18
Vin 5V 19
38 GND
37 GIOP23 VSPI MOSI
36 GIOP22 I2C SCL
35 GIOP1 TX0
34 GIOP3 RX0
33 GIOP21 I2C SDA
32 GND
31 GIOP19 VSPI MISO
30 GIOP18 VSPI SCK
29 GIOP5 VSPI SS
28 GIOP17 TX2
27 GIOP16 RX2
26 GIOP4 ADC10 TOUCH0
25 GIOP0 ADC11 TOUCH1
24 GIOP2 ADC12 TOUCH2
23 GIOP15 ADC13 TOUCH3
22 GIOP8 FLASH D1
20 GIOP7 FLASH D0
20 GIOP6 FLASH CK
ESP-WROOM-32
205 - 000519
FCC 9D:2AC72-ESPWROOM32
EN
IO0

Fig 1.1.3

Specification –

- Xtensa dual-core (or single-core) 32-bit LX6 microprocessor, running at 160 or 240 MHz
- Memory: 520 KB SRAM
- Wi-Fi: 802.11 b/g/n
- Bluetooth: v4.2 BR/EDR and BLE
- 12-bit × 18 ADC channels
- 2 × 8-bit DACs
- 10 × touch sensors (capacitive sensing GPIOs)
- 4 × SPI
- 2 × I^2S interfaces
- 2 × I^2C interfaces
- 3 × UART
- SD/SDIO/CE-ATA/MMC/eMMC host controller
- SDIO/SPI slave controller
- CAN bus 2.0
- Infrared remote controller (TX/RX, up to 8 channels)
- Motor PWM
- LED PWM (up to 16 channels)
- Hall effect sensor
- Ultra-low-power analog pre-amplifier
- All security feature of IEEE 802.11 standard, like WFA, WPA/WPA2, and WAPI, secure boot, Flash encryption
- Cryptographic hardware acceleration method like AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG)

3.2 Multiple relay module

This is a 6 channel 5V relay module with light coupling. This board is manufactured from a new and advanced design with high-end SMT processors. The terminals (C, NC, NO) are accessible through screw terminals which makes wiring up the board very easy. The relay is safely driven by transistor BC547 hence your input device, such as Arduino, is secure from the relay circuit. The inputs of this module are isolated to protect any delicate control circuitry. A wide range of microcontrollers such as Arduino, AVR, PIC, ARM, and so on can control it. The use of such a high-voltage relay eliminates the risk of heating up the relay as an electro-mechanical relay limits the current consumption in accordance with a voltage rating.

Features:-

1. Equipped with high-current relay
2. It can control both AC and DC appliances such as Motors, lights, fans, etc.
3. High-quality screw terminals
4. Freewheeling diode to protect your microcontroller

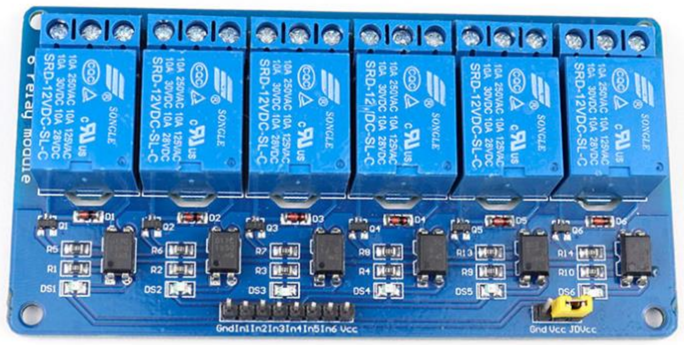
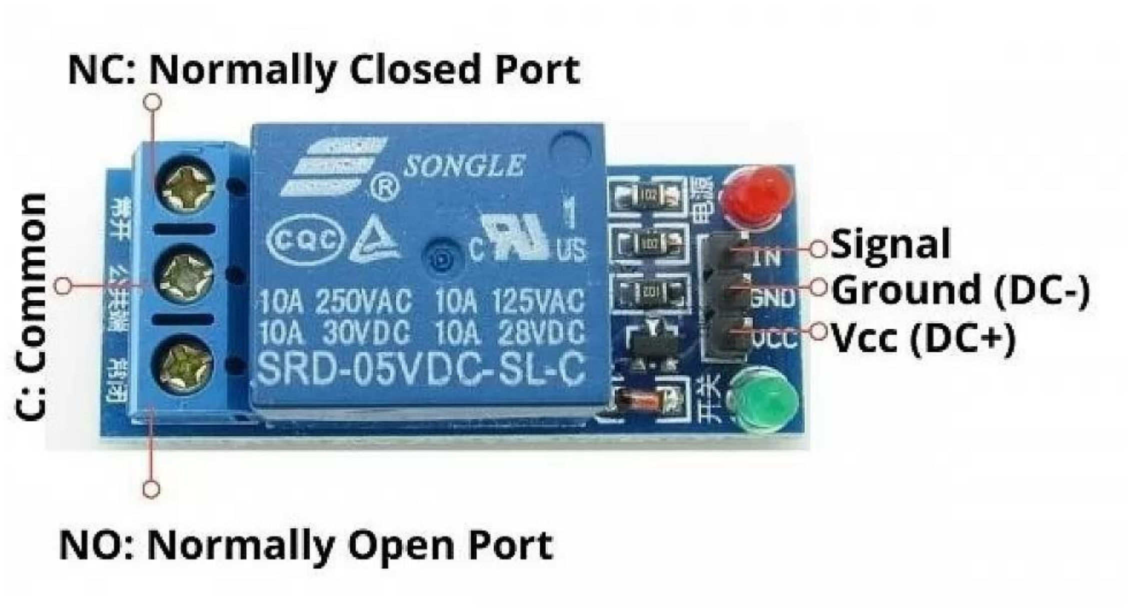


Fig 1.2

Features -

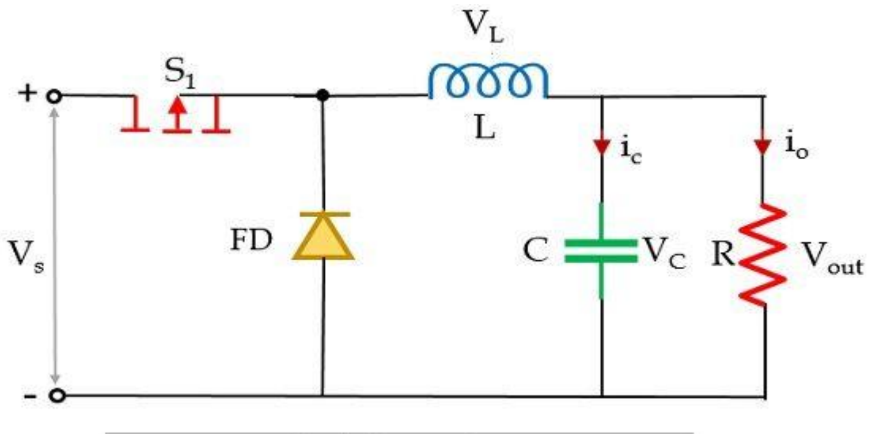
Operating Voltage (VDC)	3.3 ~ 5
Supply Current (A)	20mA
Trigger Voltage (VDC)	5
Switching Voltage (VAC)	250@10A
Switching Voltage (VDC)	30@10A
Operating Temperature (°C)	-40 to 85
Storage condition (℃)	– 65 to 125
Length (mm)	104
Width (mm)	53
Height (mm)	17
Weight (gm)	78
Shipment Weight	0.078 kg
Shipment Dimensions	9 × 15 × 2 cm

Table 2.0



3.4 Buck Converter-

Buck Converter is a type of chopper circuit that is designed to perform step-down conversion of the applied dc input signal. In the case of buck converters, the fixed dc input signal is changed into another dc signal at the output which is of lower value. This means it is designed to produce a dc signal as its output that possesses a lower magnitude than the applied input.
It is sometimes called Step-down DC to DC Converter or Step-down Chopper or Buck Regulator**.**

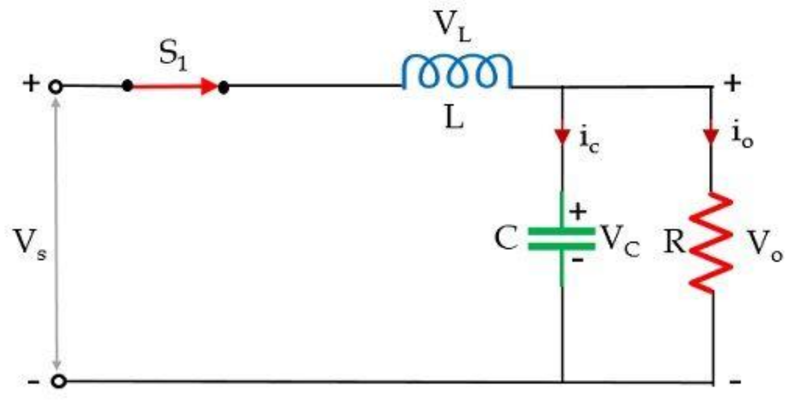


Buck Converter

In the above figure, it is clearly shown that along with the power electronics solid-state device which acts as a switch for the circuit, there is another switch in the circuit which is a freewheeling diode. The combination of these two switches forms a connection with a low-pass LC filter in order to reduce current or voltage ripples. This helps in generating regulated dc output. A pure resistor is connected across this whole arrangement that acts as a load of the circuit.

The whole operation of the circuit takes place in two modes. The first mode is the one when the power MOSFET i.e., switch S_1 is closed

In this mode of operation, switch S_1 is in closed condition thus allows the flow of current to take place through it.



When S_1 is closed

Initially when a fixed dc voltage is applied across the input terminal of the circuit then in the closed condition of switch S_1 current flows in the circuit in the manner shown above. Due to this flowing current, the inductor in the path stores energy in the form of a magnetic field. Also, there is a capacitor in the circuit and current flows through it also, therefore, it will store the charge and the voltage across it will appear across the load.

However, due to Lenz's law, the energy stored within the inductor will oppose the cause which has produced it and so an induced current will get generated and the polarity across the inductor will get reversed.

Here the total time period is a combination of T_{on} and T_{off} time.

$$\mathbf{T = T_{on} + T_{off}}$$

The duty cycle is written as:

$$D = \frac{T_{on}}{T}$$

On applying KVL, in the above-given circuit,

$$V_s = V_L + V_{out}$$
$$V_L = V_s - V_{out}$$

Also,

$$V_L = L\frac{di_L}{dt} = V_s - V_{out}$$

$$\frac{di_L}{dt} = \frac{V_s - V_{out}}{L}$$

When S1 is in closed condition then T_{on} = DT thus Δt = DT. Therefore, we can write,

$$\frac{\Delta i_L}{\Delta t} = \frac{V_s - V_{out}}{L}$$

$$\frac{\Delta i_L}{DT} = \frac{V_s - V_{out}}{L}$$

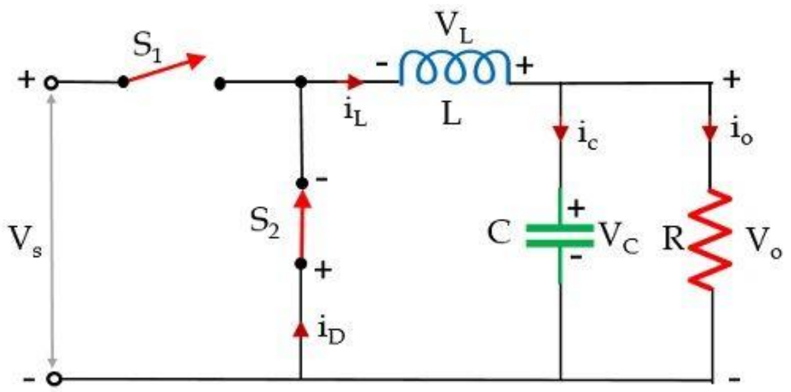
Hence,

$$\Delta i_t = (\frac{V_s - V_{out}}{L})DT$$

The above equation represents the change in current through the circuit when switch S1 is closed.

Now, the second mode of operation takes place when switch S_2 is closed and S_1 gets open. However, you must be thinking about how automatically, the switch S_2 will be closed. So, as we have discussed that the inductor in the circuit will store the energy so, once S_1 will get open the inductor in the circuit will start acting as the source.
In this mode, the inductor releases the energy which is stored in the previous mode of operation. As we have discussed that the polarity of the inductor will get reversed therefore this causes the freewheeling diode to come in a forward-biased state which was earlier present in a reverse-biased state due to the applied dc input.

Due to this, the flow of current takes place in a way as shown below:



When S₂ is closed

This flow of current will take place till the time the stored energy within the inductor gets completely collapsed. As once the inductor gets completely discharged, the diode comes in reverse biased condition leading to cause opening of switch S_2, and instantly switch S_1 will get closed and the cycle continues.

Now, let us apply KVL, in the above circuit,

$$0 = V_L + V_{out}$$

$$V_L = L\frac{di_L}{dt} = -V_{out}$$

Since, we know,

$$T = T_{on} + T_{off}$$

$$T = DT + T_{off}$$

$$T_{off} = T - DT$$

$$T_{off} = (1 - D)T$$

$$V_L = L\frac{\Delta i_L}{\Delta t} = -V_{out}$$

$$T_{off} = \Delta t = (1 - D)T$$

$$L\frac{\Delta i_L}{(1 - D)T} = -V_{out}$$

So, $$\Delta i_L = -\frac{V_{out}}{L}(1-D)T$$

This equation represents the rate of change in current through the inductor when the switch S1 is open.

As we know that the net change of current through the inductor in one complete cycle is zero. Thus,

$$\Delta i_{L(S1-closed)} + \Delta i_{L(S1-open)} = 0$$

$$\frac{V_s - V_{out}}{L} DT + \left\{-\frac{V_{out}}{L}(1-D)T\right\} = 0$$

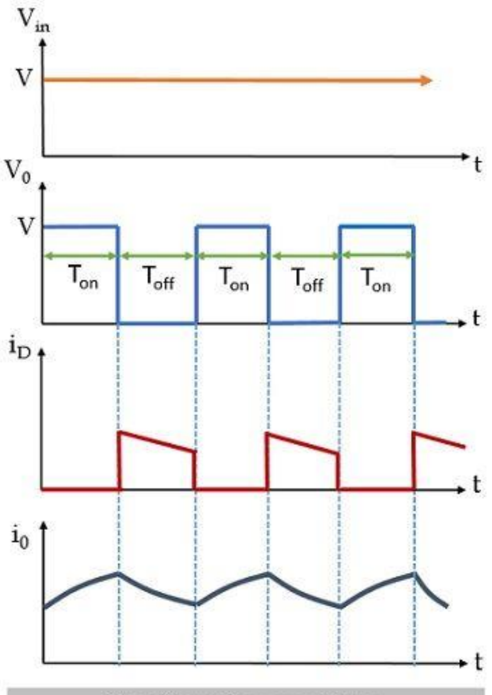
On simplifying,

$$\frac{V_s DT}{L} - \frac{V_{out} DT}{L} - \frac{V_{out} T}{L} + \frac{V_{out} DT}{L} = 0$$

$$\left(\frac{V_s DT}{L}\right) = \frac{V_{out} T}{L}$$

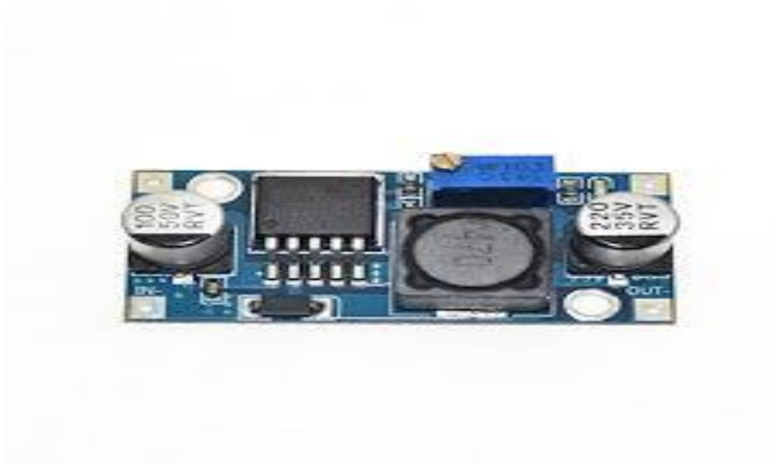
$$V_{out} = DV_s$$

The figure given below represents the waveform representation of Buck Converter:



Waveform Representation

Hence, we can say, buck converters are used to provide a lower value of dc signal from a fixed dc input



.

Fig 1.3

3.7 5v Charger Module –

This is a normal charger circuit that converts 220V AC to 5V DC. Let’s see what’s inside. Now we can see all the electronic components used in it. There are diodes, capacitors, transistors, resistors, transformers and an optocoupler. Also, there are resistors below the PCB. Once the power is on, it turns on

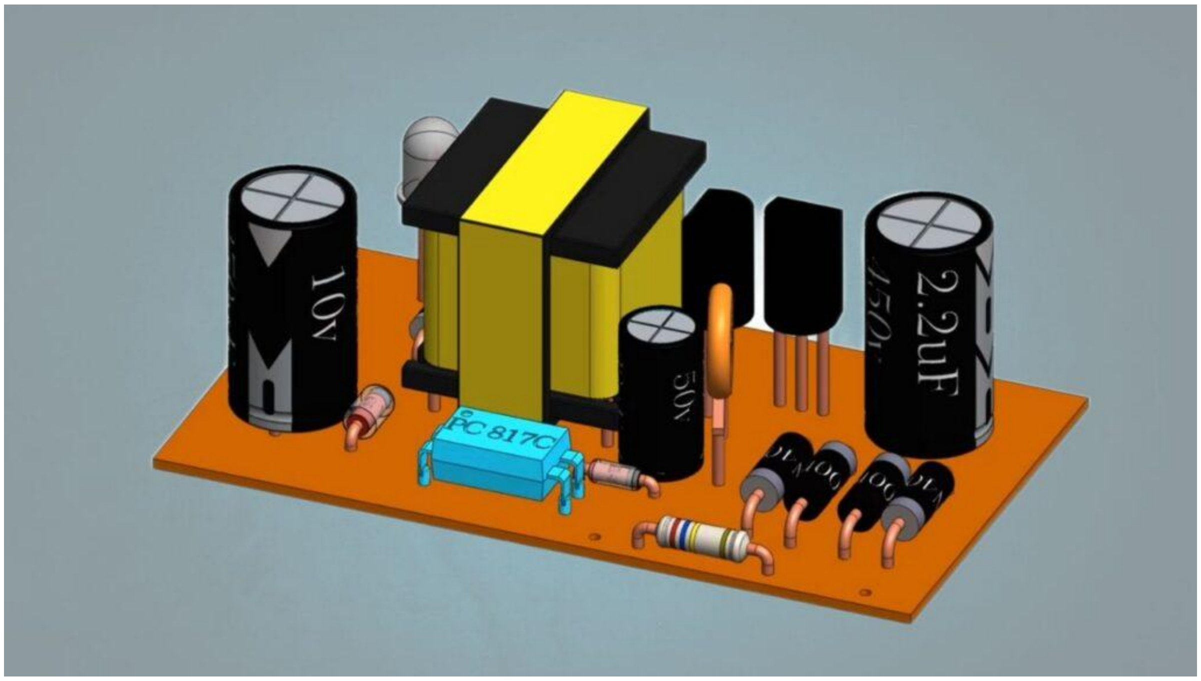
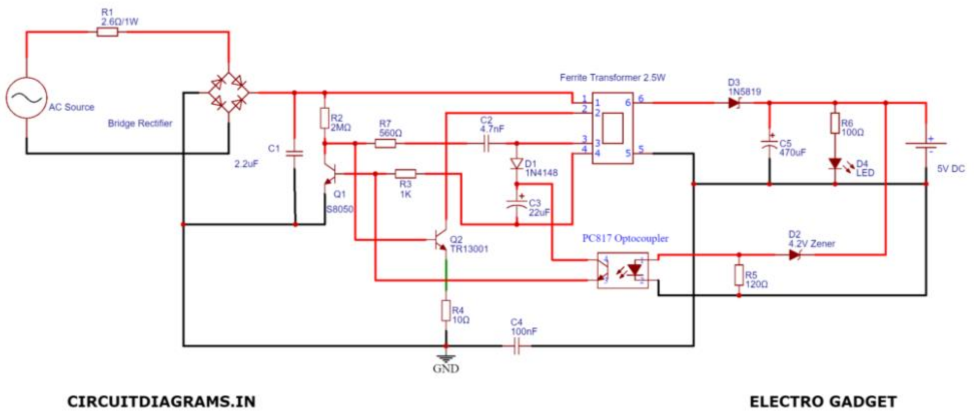


Fig 2.1

Circuit Diagram of Mobile Charger Circuit



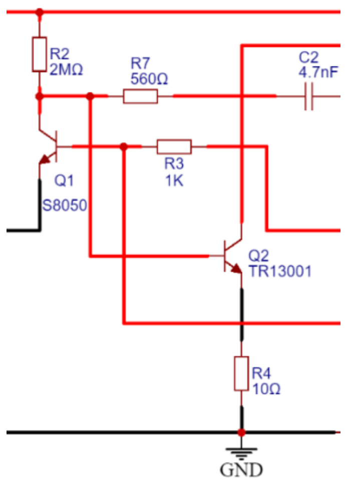
Component requirement –

- Ferrite Transformer
- PC817C Optocoupler
- 1N4007 PN Bridge Rectifier Diode (x4)

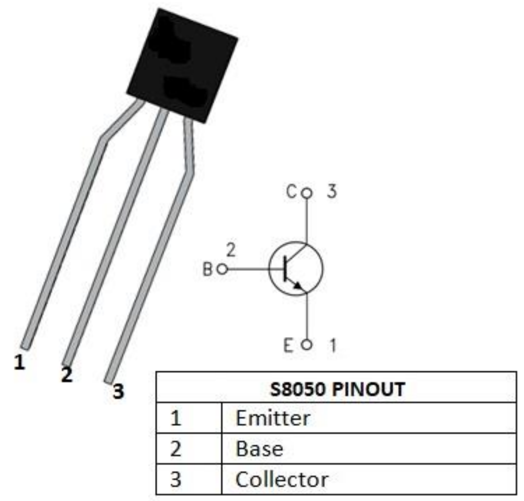
- 1N5819 Schottky Diode
- Resistor (2MΩ, 560Ω, 1KΩ, 10Ω, 120Ω, 100Ω)
- 2.6Ω/1W Fuse Resistor
- S8050 NPN Transistor
- 13001 NPN Transistor
- 2.2uF/450V Polyester Film Capacitor
- 4.7nF/100V Polyester Film Capacitor
- 470uF/25V Electrolyte Capacitor
- 22uF/25V Electrolyte Capacitor
- 100nF Ceramic Capacitor
- 4.2V Zener Diode
- Red LED
- Veroboard

Circuit Connection of Mobile Charger Circuit –

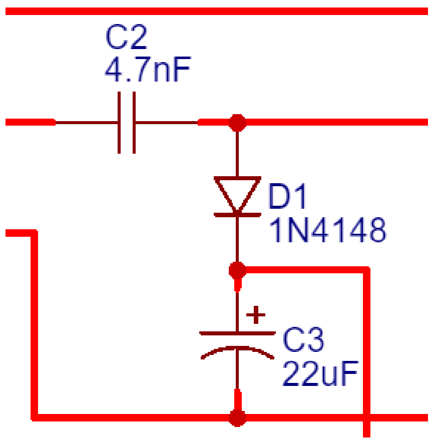
Now we can see all the components and connections. The red wire is phase wire and the black is neutral. First, we have a resistor. By observing the colour bands and reference table, we can see it is 2.6Ω. This is a fusible resistor that prevents damage from overloading. Then there is a bridge rectifier made of four 1N4007 PN junction diodes and a filter capacitor of 2.2uF/450V.



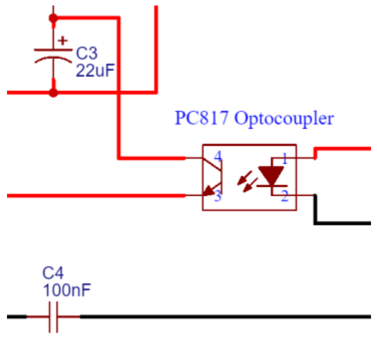
This is an oscillator circuit. This converts DC back to high-frequency AC of 15 to 50 KHz. We can see the values of the components. These are two NPN transistors S8050, and 13001, and these are their pin configurations below.



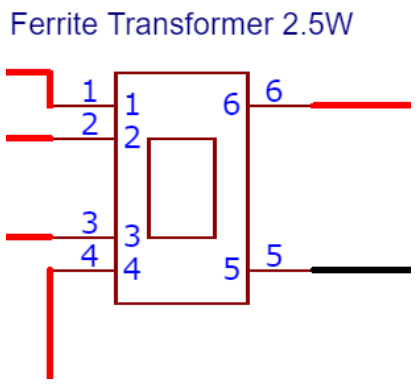
S8050 PINOUT	
1	Emitter
2	Base
3	Collector



After that portion, there is a small diode. It looks like a Zener diode, but it’s a fast switching diode i.e. 1N4148 and has a capacitor value of 22uF/63V.



This is an AC to DC converter for the phototransistor in the optocoupler, It forms a circuit like this. For the transmission of signals without contact, we use it. On the right side, we have an infrared led and on the left is a phototransistor. When the LED turns on its light turns on the base of the phototransistor turning it on. This capacitor is of 100nF used for safety purposes. It is connected between primary and secondary grounds to stop electromagnetic interference.



This is the transformer, it has three windings; primary, secondary, and auxiliary winding wrapped around the core. It is used here to step down the voltage. The auxiliary winding is used to run the oscillator circuit.

Then we have a Schottky diode 1N5819 with a capacitor of 470uF/16V to convert AC to DC and a LED for indication. Also, there is a feedback circuit that consists of an optocoupler PC817C and a 4.2V Zener diode.

3.6 BREADBOARD –

400 Tie Point Solderless Breadboard is a cute half size breadboard, good for small projects. It has 2 power buses, 30 columns, and 10 rows - a total of 400 tie in points. All pins are spaced by a standard 0.1". The two sets of five rows are separated by about 0.3", perfect for straddling a DIP package over. The board accepts wire sizes in the range of 29-20AWG. This board also has a self-adhesive on the back. The boards also have interlocking parts.

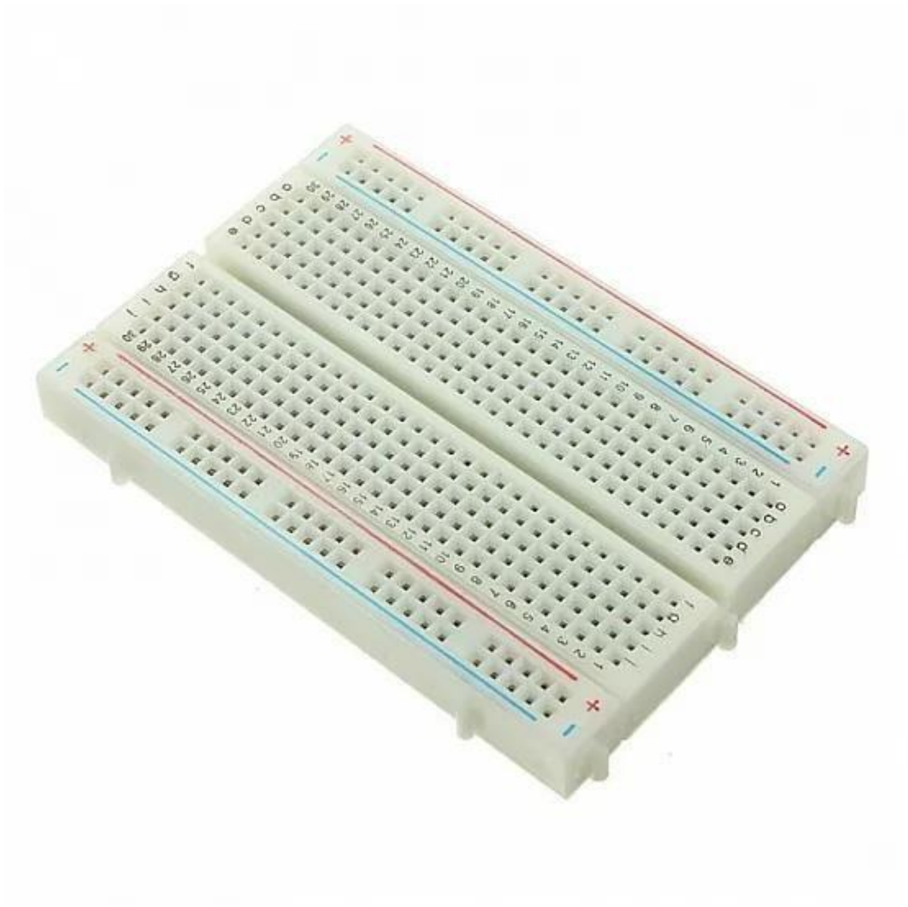


Fig 3.2

The main purpose of using breadboard is for the initial iteration or imitation of the real circuit here it provides us a good base for developing the new designs of our circuits .and it is simultaneuously very easy to roll over or to analyze certain possible permutations in our connections.

Product specifications –

Type: Plug-in breadboard

Size: 3.3 x 0.3 inches

Total 400 tie points

2 power buses

400 tie in points

0.1" space

10 rows 3.29 x 2.15 x 0.33" (83.5 x 54.5 x 8.5mm

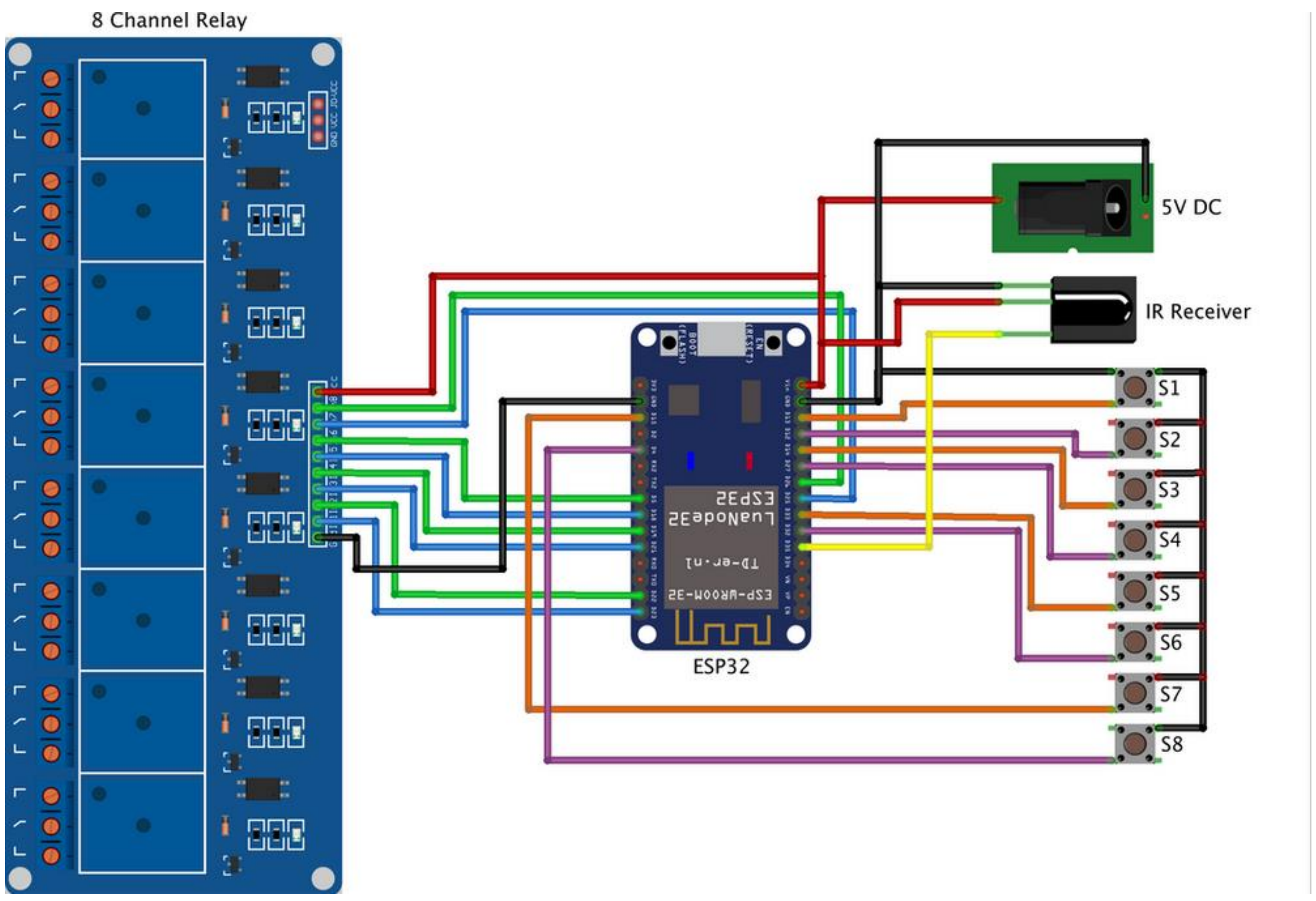
3.7 CONNECTING WIRE –

Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed

Fig 3.3

CHAPTER – 4
SYSTEM DESIGN

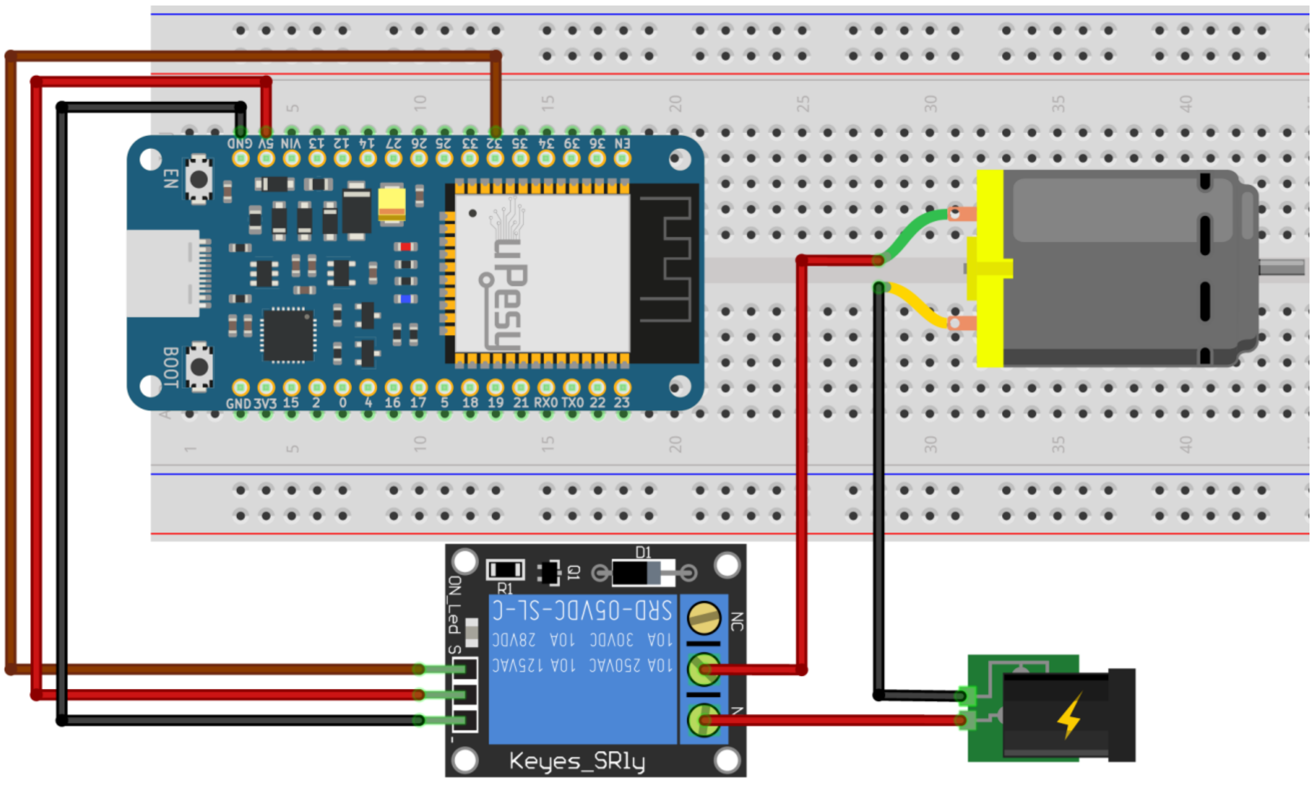
4.1 CIRCUIT DIAGRAM –



4.2 PROCESS OF MAKING –

STEP 1 :

In first step we have taken the breadboard and we have placed the Nodemcu Esp32 and relay module with motor just to check the working of one relay on it .As shown in figure .


EN
BOOT
GND 5V VIN 13 12 14 27 26 25 33 32 35 34 39 36 EN
GND 3V3 15 2 0 4 16 17 5 18 19 21 RX0 TX0 22 23
uPesy
D1
Q1
R1
SRD-05VDC-SL-C
10A 30VDC 10A 28VDC
10A 250VAC 10A 125VAC
ON_Led
NC
Keyes_SR1y

Basic block diagram of one relay interfacing

STEP 2 : In this step we have made the whole circuit on zero PCB just to plan our circuit on local basis as you can see in the above images on the next page we did all the connection via ranibow wires and press soldering.

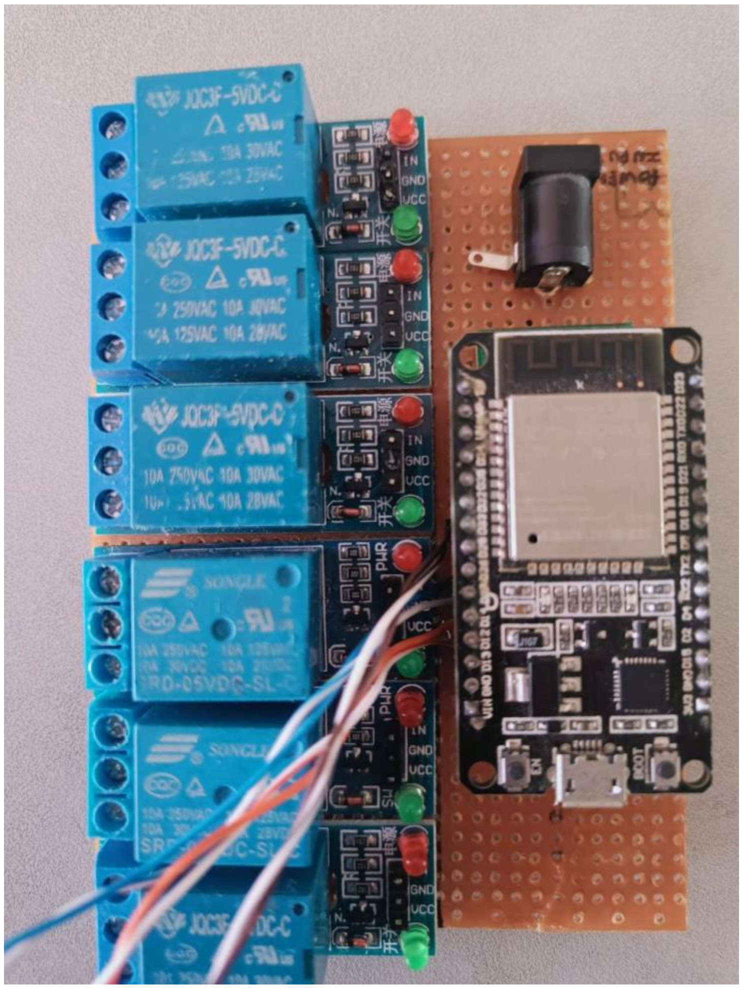
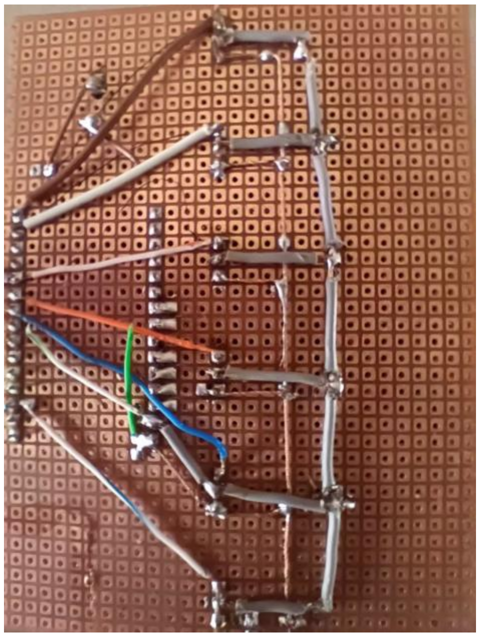


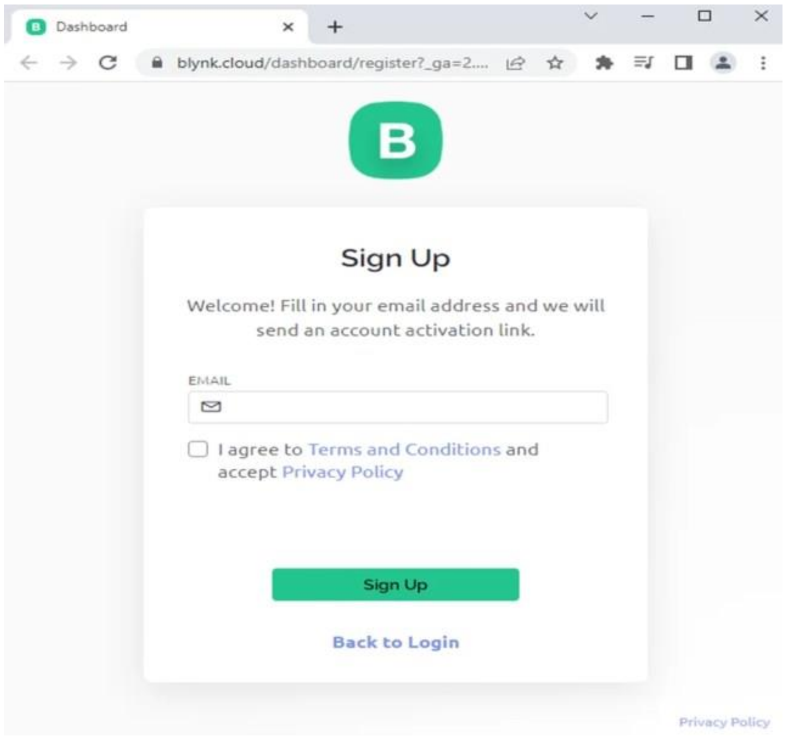
Fig 3.4

STEP 3 : In this step we are connecting all the circuitary iternally –



STEP 4 : Now for iot connectivity we are using blynk mobile application ,which we are installing from playstore . as show in figure –

1. **Frist you have to search “Blynk.io” on google.**



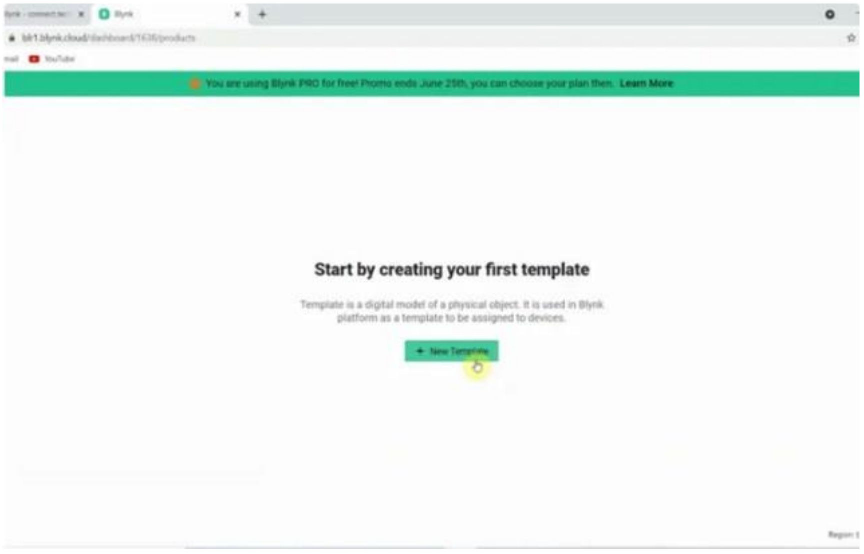
Sign Up

Welcome! Fill in your email address and we will send an account activation link.

EMAIL

I agree to Terms and Conditions and accept Privacy Policy

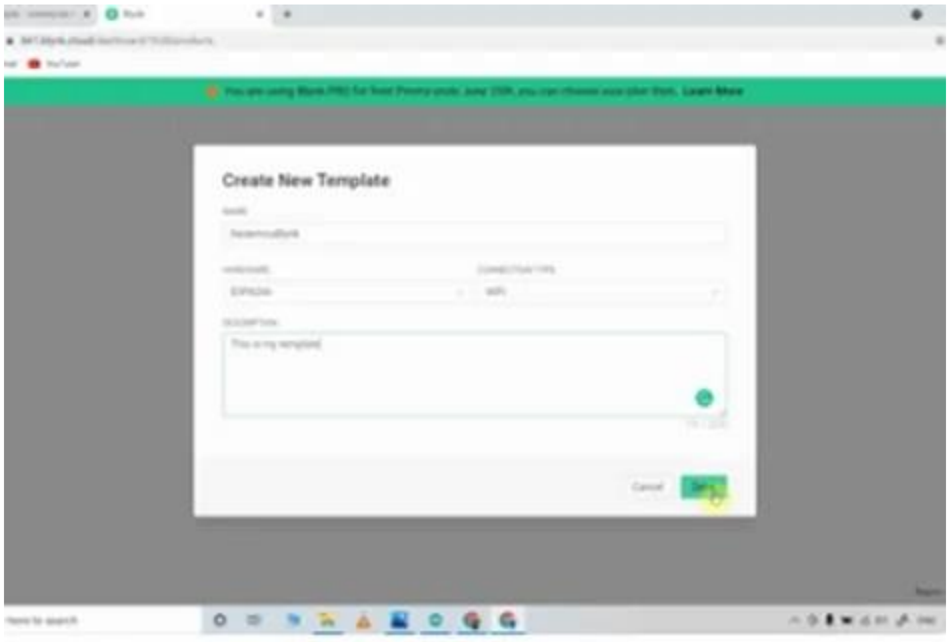
2. Create a new Template on Blynk -

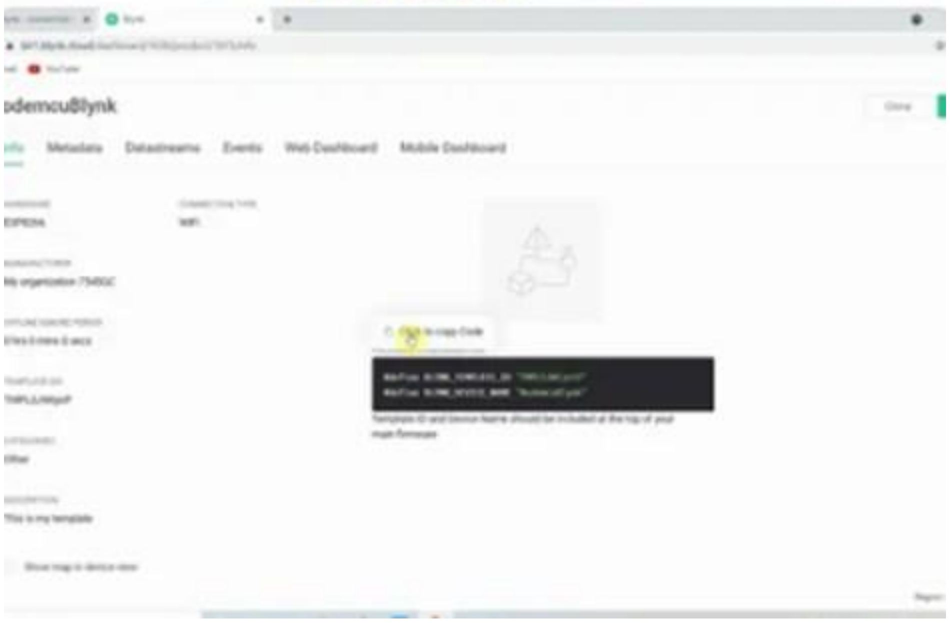


You are using Blynk PRO for free! Promo ends June 25th, you can choose your plan then. Learn More

Start by creating your first template

Template is a digital model of a physical object. It is used in Blynk platform as a template to be assigned to devices.

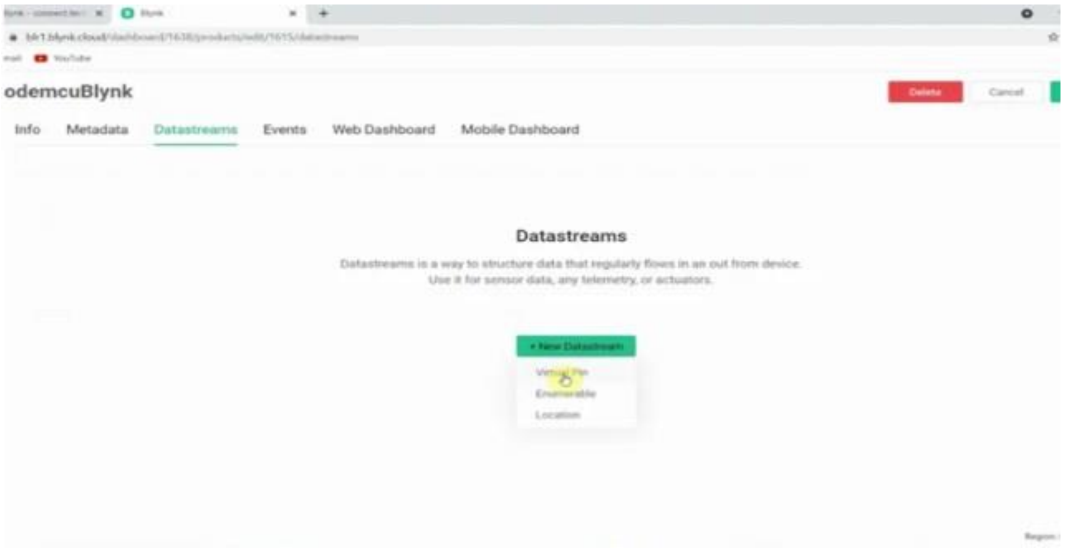




- Click on New Template.
- Select a Template name, Select hardware as esp32, Select connection type as WiFi.
- Click on done.

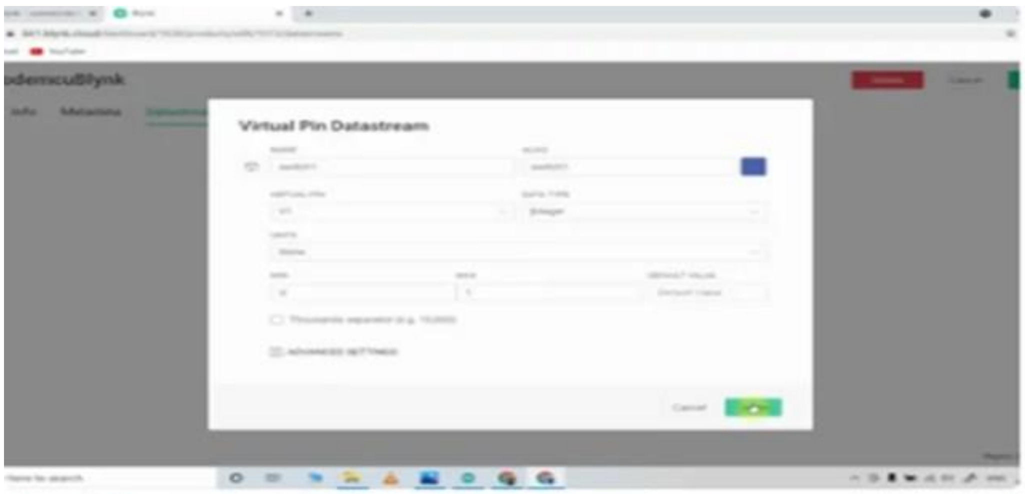
After that the BLYNK_TEMPLATE_ID, BLYNK_DEVICE_NAME and AUTH_TOKKEN will appear on screen. These will required while do programming in esp32.

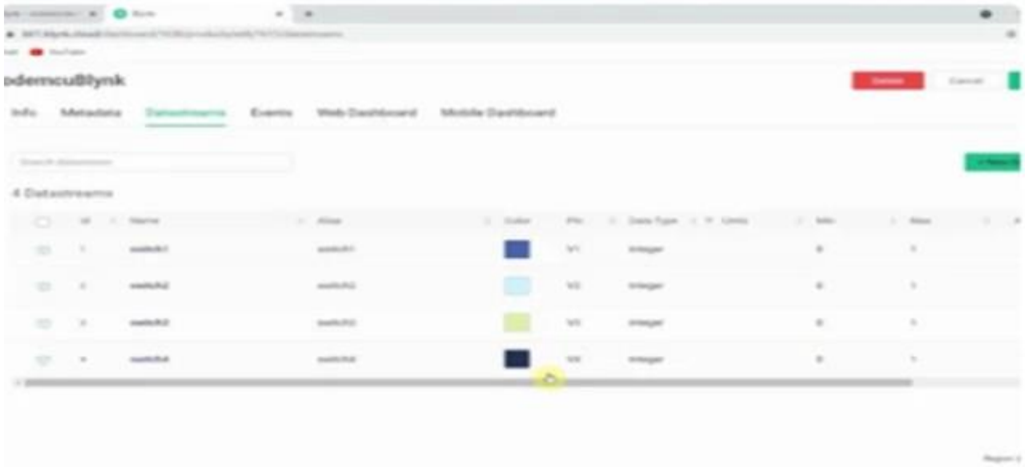
3. **Create a DataStream**



Datastreams

Datastreams is a way to structure data that regularly flows in an out from device.
Use it for sensor data, any telemetry, or actuators.



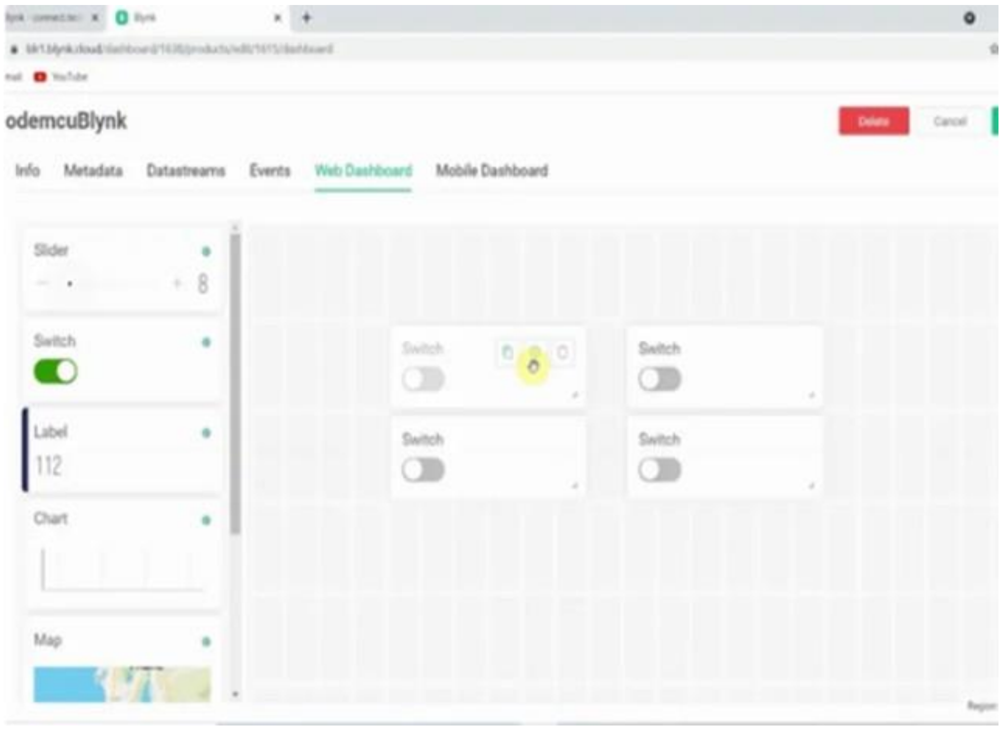


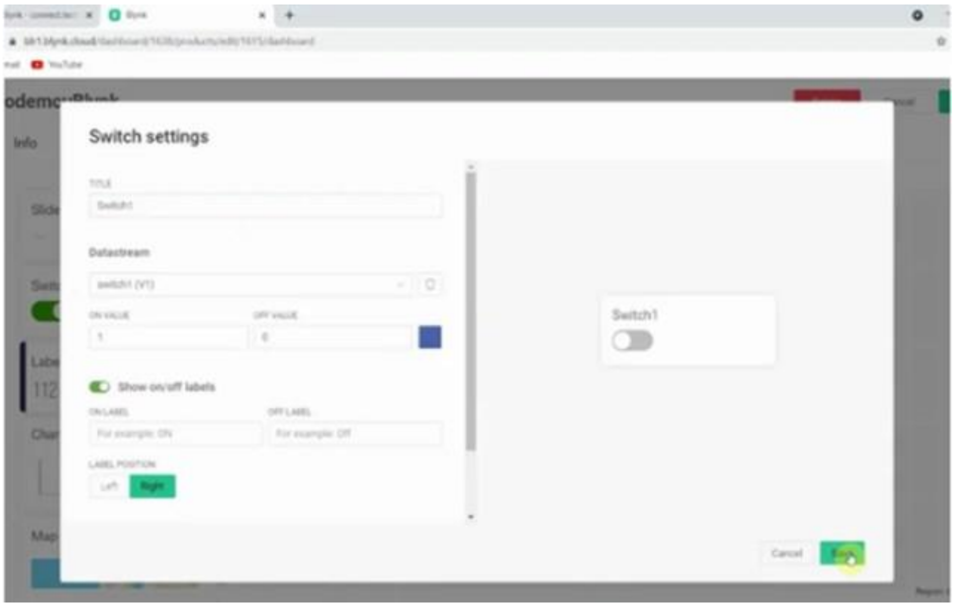
Id	Name	Alias	Color	Pin	Data Type	Units	Min	Max
1	switch1	switch1	🟦	V1	Integer		0	1
2	switch2	switch2	🟦	V2	Integer		0	1
3	switch3	switch3	🟨	V3	Integer		0	1
4	switch4	switch4	⬛	V4	Integer		0	1

we'll control 4 relay channels. So, I have to create 4 DataStream's.

- Select DataStream tab.
- Click on New DataStream and select Virtual pin.
- Enter a Name.
- Select the virtual pin V1, & select datatype be Integer.
- Click on Create.

After that create 4 DataStream similarly with virtual pin V2, V3 & V4.




Switch settings

TITLE
Switch1

Datastream
switch1 (V1)

ON VALUE
1

OFF VALUE
0

Show on/off labels

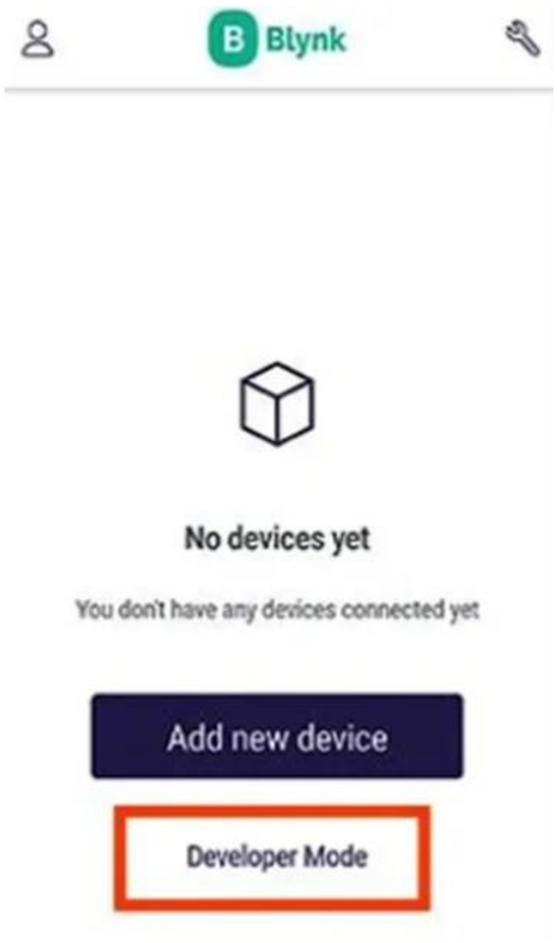
ON LABEL
For example: ON

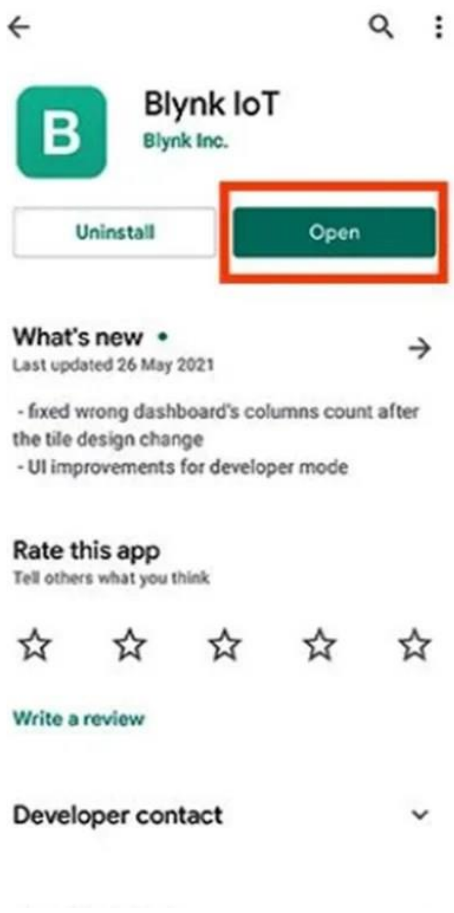
OFF LABEL
For example: Off

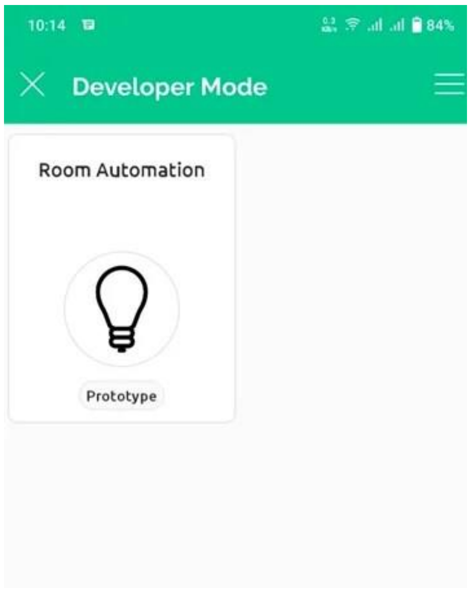
LABEL POSITION
Left Right

- Select web dashboard tab.
- Drag & drop 4 switches.
- Go to setting of all widget, and select DataStream.

5. **Install Blynk app in mobile**

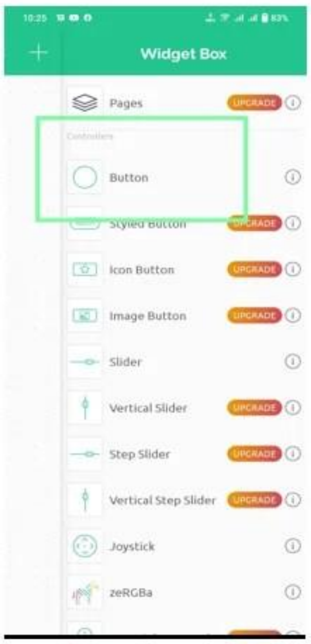


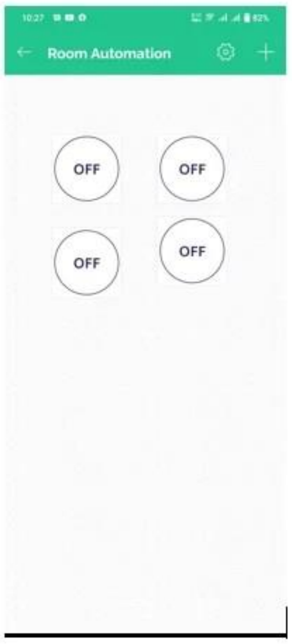

Blynk IoT
Blynk Inc.
Uninstall
Open
What's new
Last updated 26 May 2021
- fixed wrong dashboard's columns count after the tile design change
- UI improvements for developer mode
Rate this app
Tell others what you think
Write a review
Developer contact

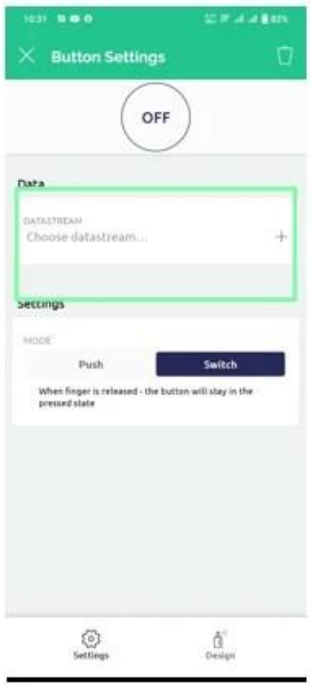


- Download the blynk Iot app from google play store.
- Log in with the same account that you have created before.
- Select developer option.

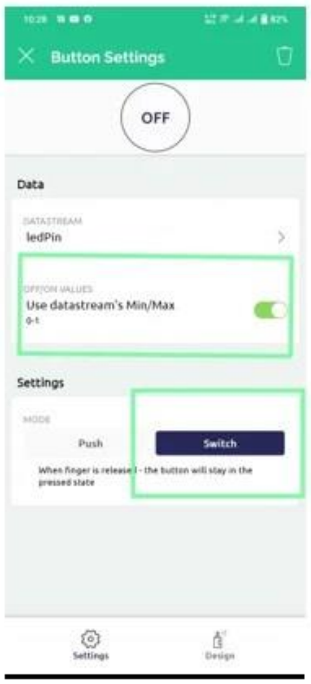
- Select the template that you have created before.
- Select the Hamburger button on the left side of your screen.
- The widget box will open.





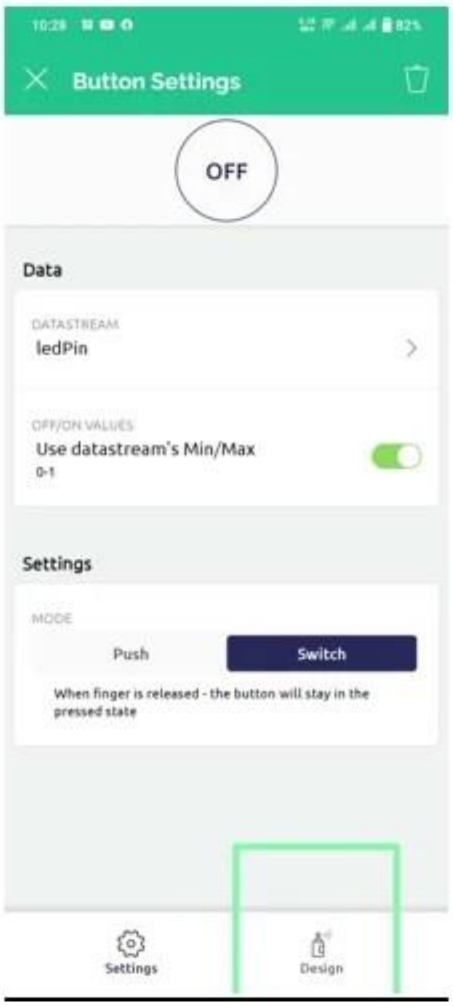


When finger is released - the button will stay in the pressed state



When finger is released - the button will stay in the pressed state

- Select 4 Buttons.
- Than tap on button.
- Tap on choose DataStream. And select a dataStream.
- Than tap on switch.



Button Settings

OFF

Data

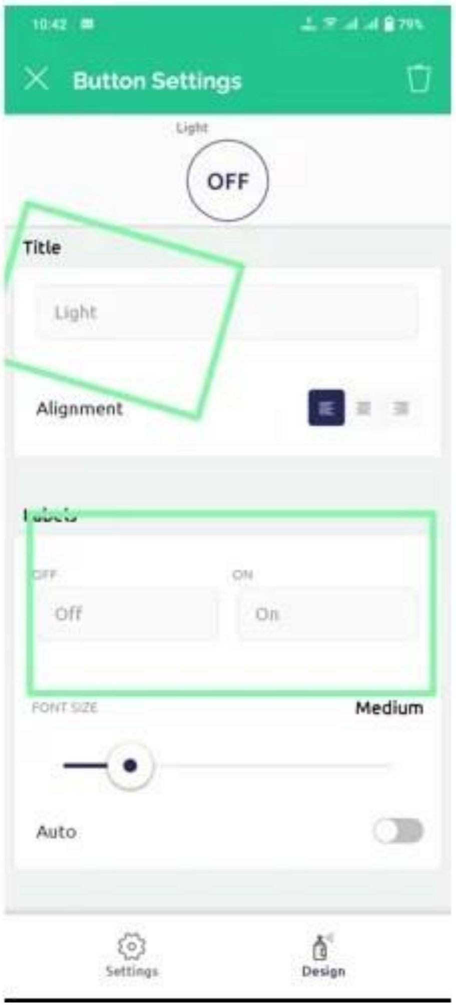
DATASTREAM
ledPin

OFF/ON VALUES
Use datastream's Min/Max
0-1

Settings

MODE
Push | Switch

When finger is released - the button will stay in the pressed state



Button Settings

Light

OFF

Title
Light

Alignment

Labels

OFF
Off

ON
On

FONT SIZE
Medium

Auto

4.3 CODING PART –

This code is written on Arduino ide after attaching all necessaries librabries needed to run this program and finally uploading it on the microcontroller .

```
/**********************************************************************************
*
 *  TITLE: Blynk 2.0 + Manual Switch (Latched) control 8 Relays using ESP32 (Real time
feedback)(No Wi-Fi control)

 *  Preferences--> Aditional boards Manager URLs :
 *                                          https://dl.espressif.com/dl/package_esp32_index.json,
http://arduino.esp8266.com/stable/package_esp8266com_index.json
 *
 *  Download Board ESP32 (1.0.6) : https://github.com/espressif/arduino-esp32
 *
 *  Download the Libraries:
 *  Blynk 1.0.1 Library:  https://github.com/blynkkk/blynk-library

**********************************************************************************
*/

// Fill-in information from your Blynk Template here
#define BLYNK_TEMPLATE_ID ""
#define BLYNK_DEVICE_NAME ""

#define BLYNK_FIRMWARE_VERSION        "0.1.0"

#define BLYNK_PRINT Serial
//#define BLYNK_DEBUG
//#define APP_DEBUG

// Uncomment your board, or configure a custom board in Settings.h
//#define USE_WROVER_BOARD


// define the GPIO connected with Relays and switches
#define RelayPin1 23  //D23
#define RelayPin2 22  //D22
#define RelayPin3 21  //D21
#define RelayPin4 19  //D19
#define RelayPin5 18  //D18
#define RelayPin6 5   //D5
#define RelayPin7 25  //D25
#define RelayPin8 26  //D26
```

```
#define SwitchPin1 13  //D13
#define SwitchPin2 12  //D12
#define SwitchPin3 14  //D14
#define SwitchPin4 27  //D27
#define SwitchPin5 33  //D33
#define SwitchPin6 32  //D32
#define SwitchPin7 15  //D15
#define SwitchPin8 4   //D4

#define wifiLed    2   //D2

#define VPIN_BUTTON_1    V1
#define VPIN_BUTTON_2    V2
#define VPIN_BUTTON_3    V3
#define VPIN_BUTTON_4    V4
#define VPIN_BUTTON_5    V5
#define VPIN_BUTTON_6    V6
#define VPIN_BUTTON_7    V7
#define VPIN_BUTTON_8    V8

#define VPIN_BUTTON_C    V9

// Relay State
bool toggleState_1 = LOW; //Define integer to remember the toggle state for relay 1
bool toggleState_2 = LOW; //Define integer to remember the toggle state for relay 2
bool toggleState_3 = LOW; //Define integer to remember the toggle state for relay 3
bool toggleState_4 = LOW; //Define integer to remember the toggle state for relay 4
bool toggleState_5 = LOW; //Define integer to remember the toggle state for relay 5
bool toggleState_6 = LOW; //Define integer to remember the toggle state for relay 6
bool toggleState_7 = LOW; //Define integer to remember the toggle state for relay 7
bool toggleState_8 = LOW; //Define integer to remember the toggle state for relay 8

// Switch State
bool SwitchState_1 = LOW;
bool SwitchState_2 = LOW;
bool SwitchState_3 = LOW;
bool SwitchState_4 = LOW;
bool SwitchState_5 = LOW;
bool SwitchState_6 = LOW;
bool SwitchState_7 = LOW;
bool SwitchState_8 = LOW;


#include "BlynkEdgent.h"

BLYNK_CONNECTED() {
  // Request the latest state from the server
  Blynk.syncVirtual(VPIN_BUTTON_1);
  Blynk.syncVirtual(VPIN_BUTTON_2);
  Blynk.syncVirtual(VPIN_BUTTON_3);
  Blynk.syncVirtual(VPIN_BUTTON_4);
  Blynk.syncVirtual(VPIN_BUTTON_5);
  Blynk.syncVirtual(VPIN_BUTTON_6);
  Blynk.syncVirtual(VPIN_BUTTON_7);
  Blynk.syncVirtual(VPIN_BUTTON_8);
}

// When App button is pushed - switch the state

BLYNK_WRITE(VPIN_BUTTON_1) {
  toggleState_1 = param.asInt();
  if(toggleState_1 == 1){
    digitalWrite(RelayPin1, LOW);
  }
  else {
    digitalWrite(RelayPin1, HIGH);
  }
}

BLYNK_WRITE(VPIN_BUTTON_2) {
  toggleState_2 = param.asInt();
  if(toggleState_2 == 1){
    digitalWrite(RelayPin2, LOW);
  }
  else {
    digitalWrite(RelayPin2, HIGH);
  }
}

BLYNK_WRITE(VPIN_BUTTON_3) {
  toggleState_3 = param.asInt();
  if(toggleState_3 == 1){
    digitalWrite(RelayPin3, LOW);
  }
  else {
    digitalWrite(RelayPin3, HIGH);
  }
}

BLYNK_WRITE(VPIN_BUTTON_4) {
  toggleState_4 = param.asInt();
  if(toggleState_4 == 1){
    digitalWrite(RelayPin4, LOW);
  }
  else {
    digitalWrite(RelayPin4, HIGH);
  }
}

BLYNK_WRITE(VPIN_BUTTON_5) {
  toggleState_5 = param.asInt();
  if(toggleState_5 == 1){
    digitalWrite(RelayPin5, LOW);
  }
  else {
    digitalWrite(RelayPin5, HIGH);
  }
}

BLYNK_WRITE(VPIN_BUTTON_6) {
  toggleState_6 = param.asInt();
  if(toggleState_6 == 1){
    digitalWrite(RelayPin6, LOW);
  }
  else {
    digitalWrite(RelayPin6, HIGH);
  }
}

BLYNK_WRITE(VPIN_BUTTON_7) {
  toggleState_7 = param.asInt();
  if(toggleState_7 == 1){
    digitalWrite(RelayPin7, LOW);
  }
  else {
    digitalWrite(RelayPin7, HIGH);
  }
}

BLYNK_WRITE(VPIN_BUTTON_8) {
  toggleState_8 = param.asInt();
  if(toggleState_8 == 1){
    digitalWrite(RelayPin8, LOW);
  }
  else {
    digitalWrite(RelayPin8, HIGH);
  }
}

BLYNK_WRITE(VPIN_BUTTON_C) {
  all_SwitchOff();
```

```
}


void all_SwitchOff(){
  toggleState_1 = 0; digitalWrite(RelayPin1, HIGH); Blynk.virtualWrite(VPIN_BUTTON_1, toggleState_1); delay(100);
  toggleState_2 = 0; digitalWrite(RelayPin2, HIGH); Blynk.virtualWrite(VPIN_BUTTON_2, toggleState_2); delay(100);
  toggleState_3 = 0; digitalWrite(RelayPin3, HIGH); Blynk.virtualWrite(VPIN_BUTTON_3, toggleState_3); delay(100);
  toggleState_4 = 0; digitalWrite(RelayPin4, HIGH); Blynk.virtualWrite(VPIN_BUTTON_4, toggleState_4); delay(100);
  toggleState_5 = 0; digitalWrite(RelayPin5, HIGH); Blynk.virtualWrite(VPIN_BUTTON_5, toggleState_5); delay(100);
  toggleState_6 = 0; digitalWrite(RelayPin6, HIGH); Blynk.virtualWrite(VPIN_BUTTON_6, toggleState_6); delay(100);
  toggleState_7 = 0; digitalWrite(RelayPin7, HIGH); Blynk.virtualWrite(VPIN_BUTTON_7, toggleState_7); delay(100);
  toggleState_8 = 0; digitalWrite(RelayPin8, HIGH); Blynk.virtualWrite(VPIN_BUTTON_8, toggleState_8); delay(100);
}

void setup()
{
  Serial.begin(115200);

  pinMode(RelayPin1, OUTPUT);
  pinMode(RelayPin2, OUTPUT);
  pinMode(RelayPin3, OUTPUT);
  pinMode(RelayPin4, OUTPUT);
  pinMode(RelayPin5, OUTPUT);
  pinMode(RelayPin6, OUTPUT);
  pinMode(RelayPin7, OUTPUT);
  pinMode(RelayPin8, OUTPUT);

  pinMode(wifiLed, OUTPUT);

  pinMode(SwitchPin1, INPUT_PULLUP);
  pinMode(SwitchPin2, INPUT_PULLUP);
  pinMode(SwitchPin3, INPUT_PULLUP);
  pinMode(SwitchPin4, INPUT_PULLUP);
  pinMode(SwitchPin5, INPUT_PULLUP);
  pinMode(SwitchPin6, INPUT_PULLUP);
  pinMode(SwitchPin7, INPUT_PULLUP);
  pinMode(SwitchPin8, INPUT_PULLUP);

  //During Starting all Relays should TURN OFF
  digitalWrite(RelayPin1, HIGH);
  digitalWrite(RelayPin2, HIGH);
  digitalWrite(RelayPin3, HIGH);
  digitalWrite(RelayPin4, HIGH);
  digitalWrite(RelayPin5, HIGH);
  digitalWrite(RelayPin6, HIGH);
  digitalWrite(RelayPin7, HIGH);
  digitalWrite(RelayPin8, HIGH);

  BlynkEdgent.begin();
}

void loop() {

   BlynkEdgent.run();

   manual_control(); //Manual Switch Control
}
```

CHAPTER – 5

CONCLUSION AND FUTURE SCOPE

5.1 CONCLUSION –

In conclusion, the ESP-32 based home automation project using the Blynk app is a powerful and efficient way to control various devices in your home remotely.
With the ability to connect multiple devices to the ESP-32 and control them through a single app, this project can streamline your home automation experience.
Through the Blynk app, you can easily control the connected devices, monitor their status, and receive alerts if anything goes wrong. This level of control and monitoring provides greater convenience and peace of mind to homeowners. Furthermore, the ESP-32's built-in Wi-Fi capabilities make it easy to connect to your home network and access the Blynk app from anywhere in the world.
This means you can remotely control and monitor your home automation devices even when you're away from home. Overall, the ESP-32 based home automation project using the Blynk app is an excellent way to bring smart home automation to your home.
With its advanced features and ease of use, this project is sure to make your life easier and more convenient.

ESP-32 based home automation project using Alexa is a great example of the power of integrating different technologies to create a smart home system that is both convenient and efficient.

With the ESP-32's capabilities and Alexa's voice control, you have developed a system that allows you to easily control your home appliances and devices with just your voice.

This project has the potential to enhance the quality of life by automating routine tasks, saving energy, and making your home more comfortable and secure. Additionally, this project can be further expanded and customized to fit your specific needs and preferences. Overall, your project highlights the endless possibilities of using technology to create smart home systems, and it is a great accomplishment that you should be proud of.

5.2 FUTURE SCOPE –

There are several exciting directions in which you can take your ESP-32 based home automation project. Here are a few potential future scopes:

Voice control: Integrate your home automation system with a voice assistant like Amazon Alexa or Google Assistant. This will allow you to control your smart home devices using voice commands, adding a new level of convenience to your system.

Energy optimization: Implement machine learning algorithms that analyze your energy usage patterns and adjust the settings of your devices to optimize energy efficiency. This can save you money on your energy bills and reduce your environmental impact.

Security features: Add security features to your home automation system, such as motion sensors and cameras, that can be accessed remotely through a mobile app. This will give you peace of mind when you're away from home and enhance the security of your property. Integration with other smart home systems: If you have other smart home devices, consider integrating them with your ESP-32 based home automation system. This will allow you to control all your devices from a single platform and create a seamless smart home experience.

Customizable user interface: Create a user-friendly interface that allows you to easily control and monitor your smart home devices. This can include a mobile app, a web-based interface, or even a physical control panel. Overall, the future scope for your ESP-32 based home automation project is vast and exciting. With a little creativity and innovation, you can continue to improve and expand your system to meet your evolving needs and prefrances and also last but not the least point that the innovation that is possible with it is way economical

REFERENCES –

- https://www.google.com
- www.youtube.com
- https://www.researchgate.net/publication/347062859_Smart_Home_Monitoring_System_Using_ESP32_Microcontrollers
- https://ijireeice.com/wp-content/uploads/2021/06/IJIREEICE.2021.9565.pdf
- Esp- 32 datasheet and pinout diagram
- Blynk app documentation
- Arduino cloud documentations