ISSN XXXX XXXX © 2019 IJESC

Research Article Volume 9 Issue No.3

Implementation of Home Automation System using IoT through ESP8266, Blynk and Google Assistant

V. S. Mahadevan[1], A. Deviparkkavi[2], I. Meenabharathi[3], B. Mokanapriya[4]
Department of Electronics and Communication Engineering
Achariya College of Engineering Technology, Puducherry, India

Abstract:
Internet of things is a friendly handling system which is connected to physical objects that are accessible through the internet. The "THING" in IOT that have been assigned an IP addresses and has the ability to collect and transfer data over a network without manual assistance or intervention. Home automation is one of the most exciting progresses in technology for the home that has come along in decades. There are hundreds of products available today that allow us control over the devices automatically, either by remote control or even by voice command. IOT intends to design an common connectivity, reasonably priced and way easy to use home automation system, which will be done by interfacing the open source Node MCU (ESP8266) microcontroller with server and by using the software connection was made to node through google assist, allowing for control of the appliances in the home model.

Keywords: IOT, ESP8266, Home Automation, Blynk Application, Google Assistant, IFTTT.

I. INTRODUCTION TO INTERNET OF THINGS

The term IoT can most likely be attributed to Kevin Ashton in 1997 with his work at Proctor and Gamble using RFID tags to manage supply chains. The work brought him to MIT in 1999 where he and a group of like-minded individuals started the Auto-ID center research consortium. Since then, IoT has taken off from simple RFID tags to an ecosystem and industry that by 2020 will cannibalize, create, or displace five trillion out of one hundred trillion global GDP dollars, or 6% of the world GDP. The concept of things being connected to the Internet up through 2012 was primarily connected smartphones, tablets, PCs, and laptops. Essentially, things that first functioned in all respects as a computer. Since the humble beginnings of the Internet starting with ARPANET in 1969, most of the technologies surrounding the IoT didn't exist. Up to the year 2000, most devices that were associated with the Internet were, as stated, computers of various sizes.

The emergence of the Internet of Things (IoT) destroys every precedent and preconceived notion of network architecture. To date, networks have been invented by engineers skilled in protocols and routing theory. But the architecture of the Internet of Things will rely much more upon lessons derived from nature than traditional (and ossified, in my opinion) networking schemes. The architecture for the Internet of Things must incorporate a fundamentally different architecture from the traditional Internet, explore the technical and economic foundations of this new architecture, and finally begin to outline a solution to the problem. The architecture of the original Internet was created long before communicating with billions of very simple devices such as sensors and appliances was ever envisioned. The coming explosion of these much simpler devices creates tremendous challenges for the current networking paradigm in terms of the number of devices, unprecedented demands for low-cost connectivity, and impossibility of managing far-flung and diverse equipment. Although these challenges are becoming evident now, they will pose a greater, more severe problem as this revolution accelerates.

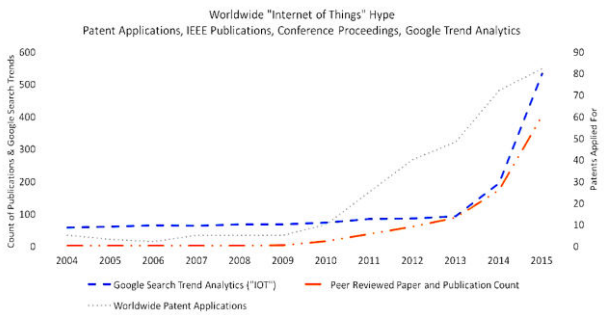

Figure.1. Analysis of keyword searches for IoT, patents, and technical publications

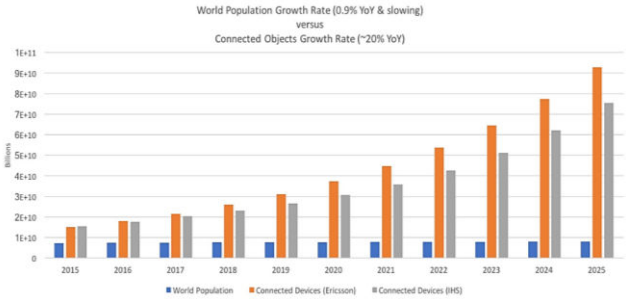

Figure.2. The disparity between human population growth versus connected thing growth.

The IoT architecture requires a much more organic approach compared with traditional networking because it represents an extreme frontier in communications. The scope and breadth of the devices to be connected are huge, and the connections to the edges of the network where these devices will be arrayed will be "low fidelity": low-speed, lossy (where attenuation and interference may cause lost but generally insignificant data, as depicted in Fig.3), and intermittent. At the same time, much of the communication will be machine-to-machine and in tiny snatches of data, which is completely the opposite of networks such as the traditional Internet. The estimated 700 billion IoT devices (see Fig.4) cannot be individually managed; they can only be accommodated. It will simply not be possible to administer the addressing of this huge population of

communicating machines through traditional means such as IPv6 nor will it be necessary to do so. Because of some quirks in the way that only part of the IPv6 address space has been released, the current theoretical number of hosts (communicating devices) on an IPv6 Internet is 3.4 × 10* 38*.

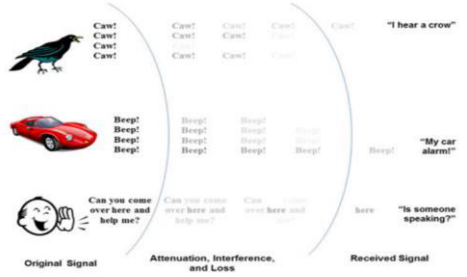

Figure.3. The results of a lossy connection at an end point

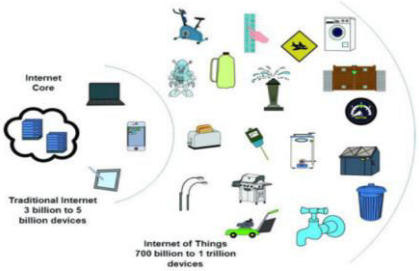

Figure.4. The quantity of devices in the Internet of Things will dwarf the traditional Internet and thus cannot be networked with current protocols, tools, and techniques

A. Zones and Neighborhoods of Interest

Another aspect of natural systems that allow them to scale is the evolution of "zones" or "neighborhoods" of interest formed by "affinities," which allow individuals to act upon a specific signal among countless other signals. A bird song is an interesting example of this phenomenon. Walking through a field, one may be struck by the songs being sung by several different bird species simultaneously. These songs can have a variety of purposes, such as advertising mating availability and suitability or defining territories. But each individual takes note only of songs from members of its own species (see Fig.5). The zones of interest, or neighborhoods of interest, of various bird species can overlap, and one communications medium (in this case audible frequencies transmitted through the air) is being used for all messages. But each individual bird acts only upon messages within its own group. Similarly, a viable architecture for the IoT must allow interested observers to define a neighborhood of interest (within the much larger Internet) and analyze or send data only from or to that neighborhood.

Figure.5. Although many different species of birds may be singing in a field, only members of the same species listen

B. In the Eyes of the Beholder

Another important aspect of scaling in the natural world is that many communications are receiver-oriented. This is in direct contrast with the sender-oriented nature of many traditional communications protocols, as described previously. Plant pollen represents an interesting example of this highly scalable characteristic of natural systems. Many of us view pollen as a (literal) irritant during hay fever season. But pollen's actual role in nature is in plant reproduction. Pollen released by the male plant is carried indiscriminately by the wind. Because pollen is a lightweight (again, literally) signal, it can be distributed hundreds or even thousands of miles by air currents. At some point, pollen falls randomly out of the air, landing on any surface. The vast majority of released pollen falls on bodies of water, bare ground, streets, or plants of another species, where it deteriorates with no effect. But some tiny portion of the total pollen released falls upon the appropriate flowering parts of a female plant of the same species. At this point, pollination takes place and seeds are generated for the next generation (see Fig.6).

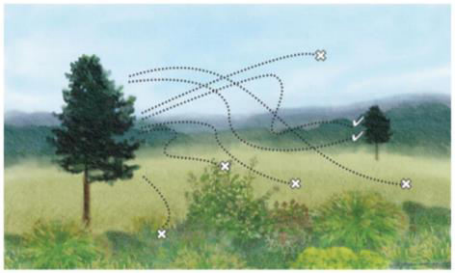

Figure.6. in nature, only the "correct" receivers act on "messages" received, such as pollen. All others discard or ignore the message

The communication of pollen is thus receiver–oriented. The zone or neighborhood of interest is defined by the receiving plant, which ignores all other signals (pollen from other species). The overall network (winds and so on) does not discriminate or actively manage the transmission of pollen in any way; it's merely a transport mechanism. The "intelligence" of nature is applied only at the receiver. In the same way, a scalable architecture for the Internet of Things out of necessity includes many elements that are receiver-oriented, with zones or neighborhoods of interest being applied at the point of data integration and collection. These integrator functions will build interesting streams of data from "neighborhoods" that are geographical, temporal, or functional. Another way of expressing these natural-world communications interactions is in term of publishers and subscribers. Many individuals may "publish" information in the form of calls, visual displays, pollen, etc. But these are moot unless other individuals "subscribe" to these messages. There is no set relationship between publisher and subscriber, as there would be in the peer-to-peer world of traditional networking–the natural world is simply too large and (obviously) unmanaged. In the IoT, the principle is the same: the only way to fully extract information from the myriad possible sources is through publish/ subscribe relationships, which can scale.

II. INTRODUCTION TO ESP8266

'Espressif Systems'Smart Connectivity Platform (ESCP) is a set of high performance, high integration wireless SOCs, designed for space and power constrained mobile platform designers. It provides unsurpassed ability to embed WiFi capabilities within other systems, or to function as a standalone application, with the lowest cost, and minimal space requirement. ESP8266EX offers a complete and self-contained WiFi networking solution; it can be used to host the application or to offload WiFi networking functions from another application processor. When ESP8266EX hosts the application, it boots up directly from an external flash. It has integrated cache to improve the performance of the system in such

applications. Alternately, serving as a WiFi adapter, wireless internet access can be added to any micro controller based design with simple connectivity (SPI/ SDIO or I2C/ UART interface).

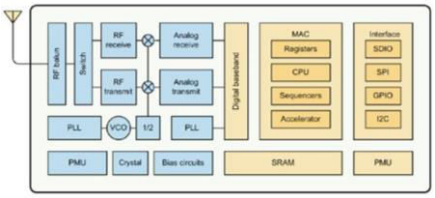

Figure.7. ESP8266EX Block Diagram

ESP8266EX is among the most integrated WiFi chip in the industry; it integrates the antenna switches, RF balun, power amplifier, low noise receive amplifier, filters, power management modules, it requires minimal external circuitry, and the entire solution, including front-end module, is designed to occupy minimal PCB area. ESP8266EX also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor, with on-chip SRAM, besides the WiFi functionalities. ESP8266EX is often integrated with external sensors and other application specific devices through its GPIOs; sample codes for such applications are provided in the software development kit (SDK).

A. Features

- Integrated low power 32-bit MCU
- Integrated TCP/ IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- 802.11 b/ g/ n WiFi 2.4 GHz, support WPA/ WPA2
- Support STA/ AP/ STA + AP operation modes
- 10-bit ADC, SDIO 2.0, (H) SPI, UART, I2C, I2S, IR Remote Control, PWM, GPIO
- Deep sleep power < 10uA, Power down leakage current < 5uA
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)
- + 20 dBm output power in 802.11b mode
- Operating temperature range -40C ~ 125C
- FCC, CE, TELEC, WiFi Alliance, and SRRC certified

B. Hardware Overview

Commonly used ESP8266 modules are ESP-12 and ESP-01.

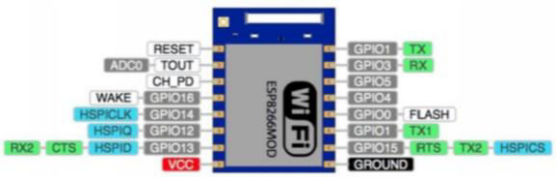

Figure. 8. ESP-12 Pin diagram

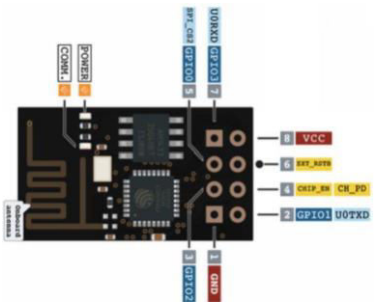

Figure. 9. ESP-01 Pin diagram

There is no difference in programming ESP-12 or ESP-01, The difference is of only the number of IOs available on the board.

C. Pin Definition

Name	Type	Function
VCC	P	Power 3.0 ~ 3.6V
GND	P	Ground
RESET	I	External reset signal (Low voltage level: Active)
ADC(TOUT)	I	ADC Pin Analog Input 0 ~ 1V
CH_PD	I	Chip Enable. High: On, chip works properly; Low: Off, small current
GPIOO (FLASH)	I/O	General purpose IO, If low while reset/power on takes chip into serial programming mode
GPIOl(TX)	I/O	General purpose IO and Serial TXd
GPIO3(RX)	I/O	General purpose IO and Serial RXd
GPIO4	I/O	General purpose IO
GPIO5	I/O	General purpose IO
GPIO12	I/O	General purpose IO
GPIO13	I/O	General purpose IO
GPIO14	I/O	General purpose IO
GPIO15 (HSPI_CS)	I/O	General purpose IO, Connect this pin to ground through 1K Ohm resistor to boot from internal flash.

TX1 and TX GPIO1 are internally connected, On board blue LED is also connected to this pin

D. Electrical Characteristics

- Working Voltage: 3.3V
- Maximum IO Driving Power IMAX: 12 mA
- Maximum IO Voltage Level VMAX: 3.6V
- Current Consumption: 100mAmp

III. HARDWARE SETUP ESP8266

A. Introduction and selection of ESP8266 module

ESP8266 requires few external components to get started. These are reset circuit, Power supply and few configurations. In this we will see each hardware aspects of ESP8266.

1) ESP-12 Module or ESP-01 Module

Choosing WiFi IOT module is very easy, I recommend you to go with ESP-12 having connections on both sides to get more IOs and Interfaces. ESP-12 that I use in most applications (Recommended)

Figure.10. ESP-12 Module

Figure.11. ESP-01 Module

This module has in-build 10-bit ADC which can take 0 to 1V as input. All IOs of ESP8266 are 3.3V logic compatible. Do not give 5V to any IO pin or supply of ESP8266. ESP8266 becomes little hot it is normal. Do not use bottom pins of this module as these pins are connected to internal flash memory. These can be used when you want to run a program from SD card. Total 8 IO, 1 serial and single ADC lines are available

on this module. Remember that GPIO0 is combined with on board LED and Serial Tx. You can use only one function of this pin. GPIO15 requires 1K resistor towards ground to boot this device from internal flash. ESP-01 Useful where size requirement is small and you want to control one or two Relays (Device on/ off). This module doesn't have ADC, so not suitable for sensor interface. There is no difference in programming these modules.

2) *Design of Power supply for ESP8266 ESP8266*

Operates at 3.3V and consumes around 100mAmp current, for designing of its power supply you can use LM1117-3.3V Low drop out (LDO) linear regulator which is having current capacity of 800mAmp. Remember that, this regulator is low drop out regulator. It is better to give 5V as input to the regulator to avoid over heating of regulator. You can create small heat sink on PCB using copper area. At lower voltages below 4.7V as Vin device will get programmed but it will not run your code.

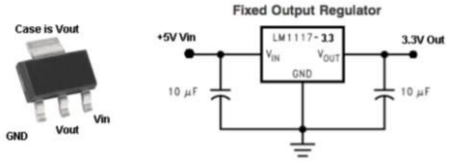

Figure.12. Power Supply 3.3V

3) *Reset Circuit*

Reset circuit is required on only ESP12 module. It's not must but it is better to have it. Putting only 10K resistor as pull-up on reset pin is ok. When RST pin is made low device will reset. This circuit is similar to the reset circuit of AVR controller. Capacitor (0.1uF) and Resistor (10K).

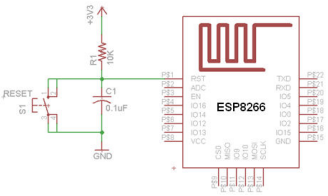

Figure.13. Reset Circuit

4) *Chip Enable and Boot selection*

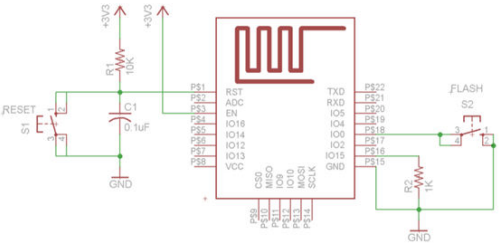

Figure.14. Boot and Chip Enable Circuit

GPIO15 (HSPI_CS), Connect this pin to ground through 1KOhm resistor to boot from internal flash. GPIO0 is used to put device in serial programming mode. When GPIO0 is low during power on or reset, this will make device go into serial programming mode.

To put device in serial programming mode

- Press and hold FLASH (S2) Button.
- Press and release RESET (S1) button while S2 is in pressed condition.
- Release FLASH(S2) button after device reset.
- These steps put ESP8266 in serial programming mode.

Once the device is programmed the internal default flash will be erased, default program can control ESP8266 using AT commands. Once you flash it using above steps you will not able to use AT commands on ESP8266. You have to re-flash with its original AT command boot code to use AT commands again.

5) *IO Level Conversion 3.3V to 5V interfacing*

Most cases we need to interface 5V (TTL) logic level devices with ESP8266. For this you can simple use 1N4148 diode and one pull-up resistor.

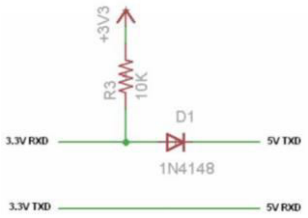

Figure.15. Logic Level Conversion for Serial

The above circuit working can be understood using Fig.16.

a) When we give logic 1 i.e. 5V to TXD diode becomes reverse bias and ESP 3.3V RXD pin gets supply of 3.3V (Logic 1) from pull-up.
b) When we give logic low 0V to TXD pin the diode becomes forward bias and the voltage at its anode terminal drops below 0.7V i.e. logic zero.
c) For TXD pin of ESP we don't need any circuit as we can read 3.3V as logic 1 and 0V as logic 0 using any 5V controller.

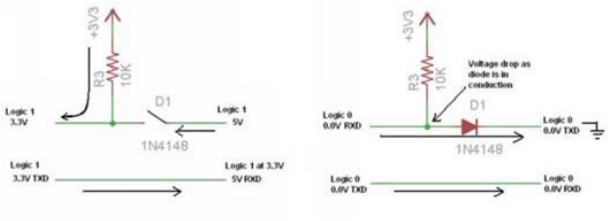

Figure.16. Logic level conversion concept

6) *Serial Interface*

For serial interface you have two options go with 3.3V USB to Serial converter or use above circuit (Fig.15) to do level conversion and use 5V serial converter. Serial interface is required to program the ESP flash. We take all our examples using only direct flashing method. We are not covering any AT command in this book. We can do a lot more using direct flashing technique. There are many USB to Serial converters are available in the market. I recommend you to go with FT232RL based converter as Chinese converter are low cost but they are unreliable and you may face driver issues. FT232RL based USB to serial converter is best suitable with voltage levels selection jumpers is recommended. Keep USB to serial converter on 3.3V level if you are directly connecting

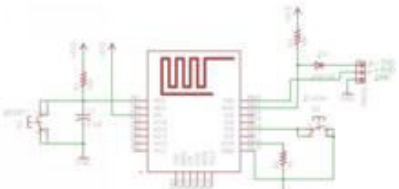

Figure.17. Serial Interface

Figure.18. USB to Serial Converter

7) *Complete hardware setup*

This is the complete hardware setup required to start with ESP8266. You can use 5V from USB also. For 3.3V Serial voltage levels no need of D1 and R3 resistors. To drive RGB LEDs, Relays use ULN2003 as driver IC. ESP8266 have PWM

feature you can use it for fading the LEDs. Let's move towards software requirements of ESP8266 and its setup.

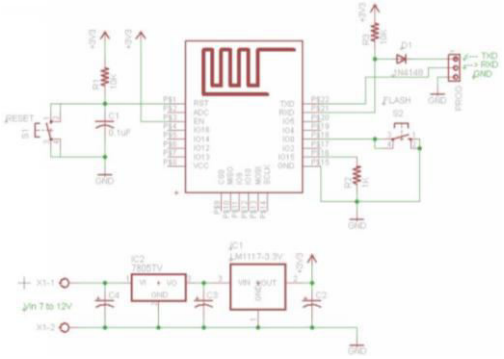

Figure.19. Complete Hardware setup for ESP8266

IV. SOFTWARE SETUP ESP8266

A. Introduction

ESP can be programmed using Arduino Software or using AT commands. In this book we are using Arduino software. Use of AT commands requires additional controller to generate AT commands. Use of AT commands give you advantage of extra IOs of controller. Direct flashing of ESP8266 using arduino gives many benefits such as no need of external controller, Program memory of 1 M bytes and RAM of 80K Bytes, Makes simple hardware and software. Let me clear you first Programming of IoT esp8266 is done in Arduino software and uses same commands that we use for Arduino Uno.

B. Steps for software setup

Step1: Download latest Arduino Software from www. Arduino .cc

Starting with 1.6.4, Arduino allows installation of third-party platform packages using Boards Manager. Use 1.6.4 or higher version of Arduino software.

Step 2: Download Arduino Core for ESP8266 WiFi Chip from GitHub Download Link (https:// github.com/ esp8266/ Arduino/)

Arduino Core for ESP8266 project brings support for ESP8266 chip to the Arduino environment. It lets you write sketches using familiar Arduino functions and libraries, and run them directly on ESP8266, no external microcontroller required. ESP8266 Arduino core comes with libraries to communicate over WiFi using TCP and UDP, set up HTTP, mDNS, SSDP, and DNS servers, do OTA updates, use a file system in flash memory, work with SD cards, servos, SPI and I2C peripherals.

Step 3: Installing with Boards Manager

Starting with 1.6.4, Arduino allows installation of third-party platform packages using Boards Manager. ESP8266 packages are available for Windows, Mac OS, and Linux (32 and 64 bit). Install Arduino 1.6.8 from the Arduino website. Start Arduino and open Preferences window.

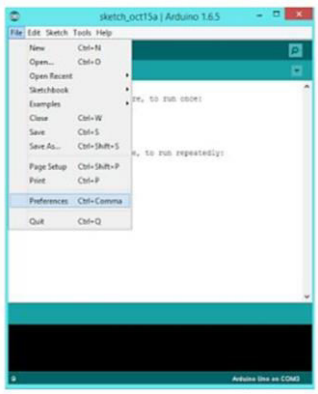

Figure.20. Preferences option

Enter http:// arduino.esp8266. com/ stable/ package_ esp 8266 com_index.json into Additional Board Manager URLs field. You can add multiple URLs, separating them with commas.

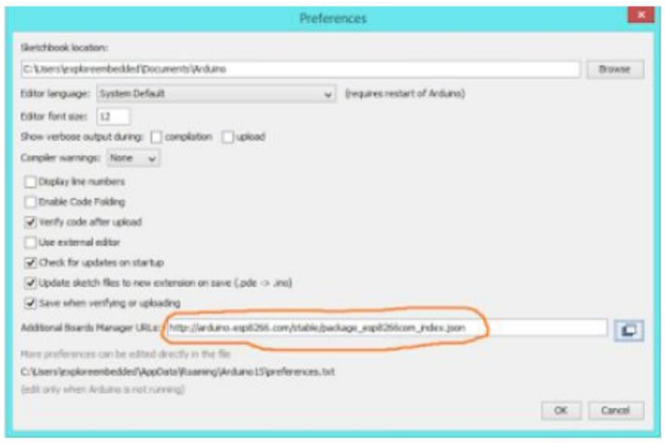

Figure.21. Preferences window

Open Boards Manager from Tools > Board menu and install esp8266 platform (and don't forget to select your ESP8266 board from Tools > Board menu after installation).

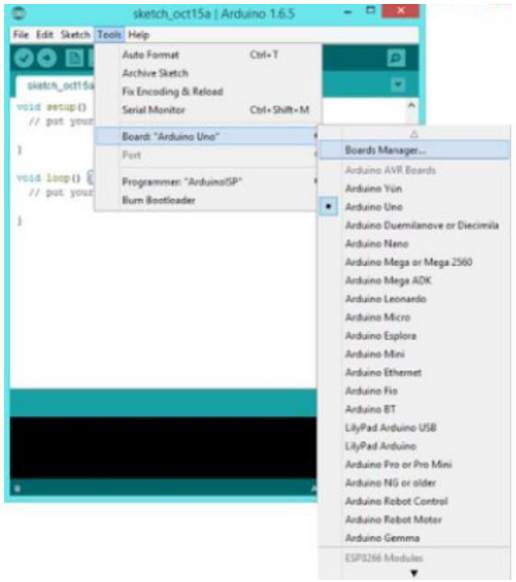

Figure.22. Board Manager Option Select ESP8266 then press install

Figure.23. Boards Manager Window

This package installation is from internet, your computer must be connected to internet it is around 250 MBytes. Read details on GitHub. After installation of ESP8266 packages for Arduino, Select board Generic ESP8266 or ESPino ESP12. Open examples > > ESP8266 > > LED blink and try compiling it.

Figure.24. LED Blink example of ESP8266

If it complies successfully skip step 4. To flash the program in ESP8266, press and hold Flash (S2) button and momentarily press Reset button and then release Flash button. Press upload button from your Arduino window. Program will start uploading. After uploading onboard blue LED will start to blink. If you passed this blink test you are ready to go further.

Step 4 (If required): After installing everything, you may find trouble in compiling missing .h files Go to your arduino installation folder (where your arduino.exe is) in this folder create folder "portable" Copy and paste copy from "C:\ Documents and Settings\ Administrator\ Application Data\ Arduino" (check that this address is shown while compilation error missing .h file) Copy all contains (folder naming "packages" and "staging" and surrounding files) Paste the copied contains in "portable" folder that you created in Arduino folder. Check that compilation works now. If all ok go ahead to test it on hardware.

V. IOT HOME AUTOMATION

With advancement of Automation technology, life is getting simpler and easier in all aspects. In today's world Automatic systems are being preferred over manual system. Wireless Home Automation system(WHAS) using IoT is a system that uses computers or mobile devices to control basic home functions and features automatically through internet from anywhere around the world, an automated home is sometimes called a smart home. It is meant to save the electric power and human energy. The home automation system differs from other system by allowing the user to operate the system from anywhere around the world through internet connection.

A. Components Required

- ESP-12 WiFi Module
- LM1117-3.3V
- Sugar Cube Relay – Qty. 4
- Resistors 10K, 1K, 4.7K
- Capacitor 1000uF, 10uF, 104 (0.1uF)
- PBT-2 Connectors Qty. 5
- ULN2003
- 12V Power Supply

B. Circuit Diagram of Home Automation

From the circuit diagram it is very clear that we have used few components, LM1117-3.3V is used for providing power supply to the ESP-12 WiFi Module. ULN2003 provides relay driving.

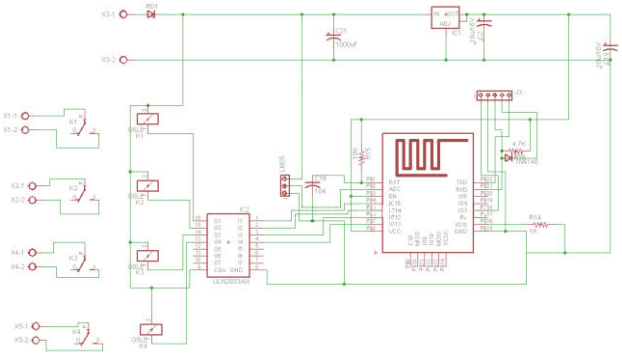

Figure.25. Home Automation Circuit

VI. RESULTS

A. Program and outcomes of Home Automation System

```
This example runs directly on ESP8266 chip.
Note: This requires ESP8266 support package:
https://github.com/esp8266/Arduino
Please be sure to select the right ESP8266 module
in the Tools -> Board menu!
Change WiFi ssid, pass, and Blynk auth token to run :)
Feel free to apply it to any other example. It's simple!
**********************************************/*
Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "8bd30c98a6494dbf8fcd3f6703c3cef2";
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "vivo 11716";
char pass[] = "sithu0007";
void setup()
{
// Debug console
Serial.begin(9600);
Blynk.begin(auth, ssid, pass);
}
void loop()
{
Blynk.run();
}
```

B. 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, vizualize 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.
- **BlynkServer** - responsible for all the communications between the smartphone and hardware. You can use our Blynk Cloud or run your private 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.

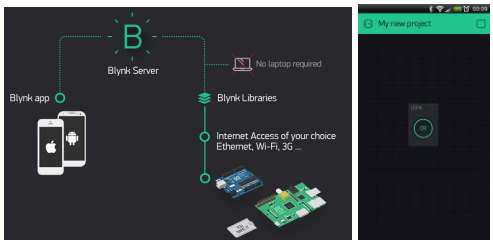

Figure.25. Blynk app

C. Google Assistant

Google Assistant is an artificial intelligence-powered virtual assistant developed by Google that is primarily available on mobile and smart home devices. Unlike the company's previous virtual assistant, Google Now, Google Assistant can engage in two-way conversations. Users primarily interact with Google Assistant through natural voice, though keyboard input is also supported. In the same nature and manner as Google Now, the Assistant is able to search the Internet, schedule events and alarms, adjust hardware settings on the user's device, and show information from the user's Google account. Google has also announced that the Assistant will be able to identify objects and gather visual information through the device's camera, and support purchasing products and sending money, as well as identifying songs. At CES 2018, the first Assistant-powered smart displays (smart speakers with video screens) were announced, with the first one being released in July 2018.

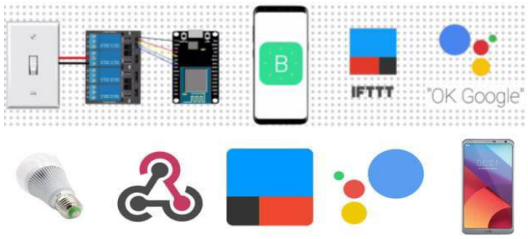

Figure.26. Standard Way to Interact

D. Way to Interact with Home Automation System

The standard way to interact with your IoT devices using Google Assistant implies that:

1) *You speak your command*
2) *Google voice services transcript it and search the different providers for one that can handle it, including IFTTT*
3) *IFTTT grabs the command and tells GA "yeah, I can do that"*
4) *IFTTT translates it again based on an applet you have defined*
5) *The action in your applet instructs IFTTT to tell WebHooks to do an HTTP request*
6) *WebHooks performs the request to an address that points to your device*
7) *Your device executes the action*

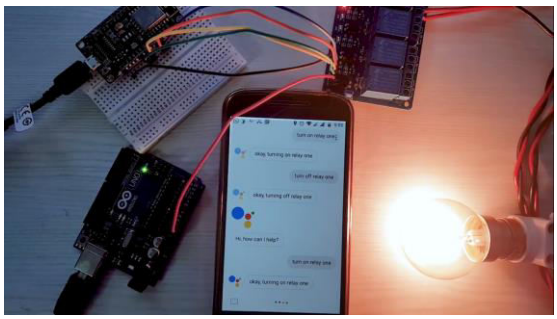

Figure.27. Working Model

VII. CONCLUSION

A tremendous potential exists for expansion of home networking in the form of chirp-enabled networking. One focus may be the TV set-top box or a smart TV that already increasingly includes Internet access. One can imagine future devices that connect not only to existing home equipment via infrared interfaces and the Internet via cable or Wi-Fi but also link the rest of the devices in the home via Power Line, Wi-Fi, or other technologies. Alternately, combination home propagator node/ APs with the appropriate chirp transceivers built in will support both Wi-Fi IP and chirp traffic. A local integrator function within the propagator node could provide the "brains" for home entertainment, climate control, security, energy management, and so on. Because this device will have access to a much broader set of devices as well as other data sources such as weather reports and utility updates, it will optimize the operation of the home as not previously possible. Unlike expensive proprietary solutions offered to date, proliferation of compatible chirp-enabled products will reduce costs, allow expansion over time, and eliminate reliance on single-vendor offerings.

VIII. REFERENCES

[1]. Perry Lea, "Internet of Things for Architects", ISBN 978-1-78847-059-9, Packt Publication, January 2018.

[2]. Francis da Costa and Byron Henderson, "Rethinking the Internet of Things, A Scalable Approach to Connecting Everything",ISBN 978-1-4302-5740-0, Apress Open, Springer 2013.

[3]. Sadi Mahmud, Safayet Ahmed and Kawshik Shikder "A Smart Home Automation and Metering System using Internet of Things (IoT)", IEEE 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), February 2019.

[4]. Md. Sadad Mahamud, Md. Saniat Rahman Zishan, Syed Ishmam Ahmad and Ahmed Rezaur Rahman "An IoT Based Smart Home Automation System", IEEE 2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST), February 2019.

[5]. Imad Jawhar, Nader Mohamed and Jameela Al-Jaroodi, "Networking architectures and protocols for smart city systems", Journal of Internet Services and Applications, Springer, ISSN1867-4828, December 2018.

[6]. O. Koksal and B. Tekinerdogan "Architecture design approach for IoT-based farm management information systems", Precision Agriculture, Springer US, ISSN1385-2256, December 2018.

[7]. Michelle D souza, Nelsha Wilfred, Rochel Pereira, Thanya Rayen and Aparna Telgote "Home automation using Internet of Things", 2017 IEEE International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS), INSPEC Accession Number: 17859074, June 2018.

[8]. Badar H. Al Lawati, Xiaowen Fang "Home Automation Internet of Things: Adopted or Diffused?", International Conference on Distributed, Ambient, Pervasive Interactions, Springer, Cham, ISBN 978-3-319-91124-3, May 2018, pp 181-190.

[9]. Tanmay Chakraborty and Soumya Kanti Datta "Home automation using edge computing and Internet of Things", 2017 IEEE International Symposium on Consumer Electronics (ISCE) INSPEC, Accession Number: 17751546 EISSN: 2159-1423 May 2018.

[10]. Shaik Naseera, Anurag Sachan and G. K. Rajini "Artificial Intelligence and Evolutionary Computations in Engineering Systems", Design of Smart Home Using Internet of Things, Springer, Singapore, ISBN 978-981-10-7867-5, March 2018, pp 349-357

[11]. Sunil Kumar Khatri, Govind Sharma, Prashant Johri and Sachit Mohan "Smart Gesture Control for Home Automation Using Internet of Things", Intelligent Computing and Information and Communication, Springer, Singapore ISBN 978-981-10-7244-4, January 2018, pp 633-641.

[12]. Rui Liu and Yongqi Ge "Smart home system design based on Internet of Things", 2017 12th International Conference on Computer Science and Education (ICCSE) E-ISSN: 2473-9464, October 2017.

[13]. Sri Harsha, S Chakrapani Reddy and S Prince Mary "Enhanced home automation system using Internet of Things", IEEE 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC) INSPEC Accession Number: 17224938, October 2017.

[14]. Marcus Oppitz and Peter Tomsu "Internet of Things", Inventing the Cloud Century, Springer, Cham, ISBN 978-3-319-61160-0, August 2017.

[15]. Suneha Ashok Patil and Vishwakarma Pinki "Home Automation Using Single Board Computing as an Internet of Things Application", Proceedings of International Conference on Communication and Networks, Springer, Singapore, April 2017, pp 245-253.

[16]. S Soumya, Malini Chavali, Shuchi Gupta and Niharika Rao "Internet of Things based home automation system", 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), INSPEC Accession Number: 16583019, January 2017.

[17]. Dietmar P. F. Moller "Introduction to the Internet of Things", Guide to Computing Fundamentals in Cyber Physical Systems, Springer, Cham, ISBN 978-3-319-25176-9, April 2016, pp 141-184.

[18]. Vibhutesh Kumar Singh, Sanjeev Baghoriya and Vivek Ashok Bohara "HELPER: A Home assisted and cost Effective Living system for People with disabilities and homebound Elderly", IEEE 26th International Symposium on Personal, Indoor and Mobile Radio Communications, 2015, pp. 2115-2119.

[19]. YAN Wenbo, WANG Quanyu, GAO Zhenwei "Smart Home Implementation Based on Internet and WiFi Technology", Proceedings of the 34th Chinese Control Conference Hangzhou, China, July 28-30, 2015, pp. 9072-9077.

[20]. Sehoon Kim, Jin-Young Hong, Seil Kim, Sung-Hoon Kim, Jun-Hyung Kim, Jake Chun "RESTful Design and Implementation of Smart Appliances for Smart Home", IEEE International Conference on Ubiquitous Intelligence and Computing, 2014, pp. 717-722.