

HARDWARE ROOT OF TRUST FOR IoT SECURITY IN SMART HOME SYSTEMS

**Submitted in partial fulfillment of the requirements for the award of Bachelor
of Engineering Degree in
Electronics and Communication Engineering**

by

S.SRIHARI (39130452)



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
SCHOOL OF ELECTRICAL AND ELECTRONICS**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHISALAI, CHENNAI – 600119

APRIL - 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY) Accredited

with "A" grade by NAAC

Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of S.SRIHARI(39130452) who carried out the project entitled "HARDWARE ROOT OFTRUST FOR IoT SECURITY IN SMART HOME SYSTEMS" under supervision from November 2022 to April 2023.

INTERNAL GUIDE

M.R. Ebenezer Jebarani

Dr.M.R. EBENEZAR JEBARANI M.E.,Ph.D.,

HEAD OF THE DEPARTMENT

T. Ravi

Dr.T.RAVI M.E.,Ph.D.,

Submitted for Viva voce Examination held on 23.04.2023

[Signature]

Internal Examiner

[Signature]

External Examiner

DECLARATION

I S.SRIHARI(39130452) hereby declare that the Project Report entitled “**HARDWARE ROOT OF TRUST FOR IoT SECURITY IN SMART HOME SYSTEMS**”. done by us under the guidance of Dr. M.R.EBENEZER JEBARANI M.E., Ph.D.,at **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Electronics and communication Engineering.

DATE: 23.04.2023

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

1. 

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

I convey our thanks to **Dr.N.M.NANDHITHA, M.E.,Ph.D.**, Prof & Dean of school of Electrical and Electronics Engineering and **Dr.T.RAVI, M.E.,Ph.D.**, Heads of the Department, Department of Electronics and communication Engineering for providing us necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to our Project Guide **Dr.M.R.EBENEZER JEBARANI, M.E.,Ph.D.** for their valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

I express my thanks to our project panel members **Dr.E.Annadevi., M.E.,Ph.D** And **Dr.Thaj MaryDelsy.,ME.,Ph.D** for their constant support and encouragement throughout our project review.

We wish to express my thanks to all Teaching and Non-teaching staff members of the Department of Electronics and communication engineering who were helpful in many ways for the completion of the project.

Abstract:

This project presents the overall design of Home Automation System (HAS) with low cost and wireless system. It specifically focuses on the development of an IOT based home automation system that is able to control various components via internet or be automatically programmed to operate from ambient conditions. In this project, we design the development of a firmware for smart control which can successfully be automated minimizing human interaction to preserve the integrity within whole electrical devices in the home. We used Node MCU, a popular open source IOT platform, to execute the process of automation. Different components of the system will use different transmission mode that will be implemented to communicate the control of the devices by the user through Node MCU to the actual appliance. The main control system implements wireless technology to provide remote access from smart phone. We are using a cloud server-based communication that would add to the practicality of the project by enabling unrestricted access of the appliances to the user irrespective of the distance factor. We provided a data transmission network to create a stronger automation. The system intended to control electrical appliances and devices in house with relatively low cost design, user-friendly interface and ease of installation. The status of the appliance would be available, along with the control on an android platform. This system is designed to assist and provide support in order to fulfil the needs of elderly and disabled in home. Also, the smart home concept in the system improves the standard living at home.

TABLE OF CONTENTS

NO	CHAPTER	TITLE	PAGE NO
		ABSTRACT	v
		LIST OF FIGURES	viii
1		INTRODUCTION	9
		1.1 objective	12
2		LITERATURE SURVEY	13
3		INVESTIGATION	
		3.1 Existing system	15
		3.2 Proposed system	16
		3.3 Block diagram	17
		3.4 Hardware used	17
4		HARDWARE DESCRIPTION	18
		4.1 Internet of Things	18
		4.2 How Does Internet of Thing(IoT)Work?	19
		4.3 Features of IoT	20
		4.4 Advantages and Disadvantages of(IoT)	21
		4.5 Embeded Devices (System) in (IoT)	22
		4.6 Embeded System Hardware	24
		4.7 IoT Ecosystem	24
		4.8 IoT Decision Framework	26
		4.9 Decisison Area	27

5	RESULTS and DISCUSSIONS	65
6	CONCLUSION and FUTURE SCOPE	65
7	REFERENCES	66

LIST OF FIGURES

FIGURE NO	TITLE	PAGE NO
3.1	Block diagram	14
4.1	IoT Figure	15
4.2	IoT Cloud Diagram	16
4.3	IoT Architecture	17
4.4	Internet of Things	20
4.5	Basic Embedded System	21
4.6	Typical IoT Ecosystem	22
4.7	IoT Technology Stack	24
4.8	Decision Area of the IoT Decision Framework	25
4.9	Stages of IoT Solutions Architecture	26
4.10	The 4 stages IoT Solutions Architecture	28
4.11	smart objects	29
4.12	IoT Devices	30
4.13	IoT Devices and Technology	32
4.14	NodeMCU	34
4.15	Relay	38
4.16	General Block Of Power Supply Unit	40
5.1	Hardware Photos	53
5.2	Output Photos	54

CHAPTER-1

INTRODUCTION:-

This decade has been characterized by a tremendous interest in sensors and sensor networks. Where it is predicted that billions of devices would be connected to the internet to communicate in some fashions. Considering the recent progress of Internet of Things (IoTs), 'Insider Intelligence' expects that 64 billion devices will be connected on the planet by 2025. The major devices include, but are not limited to, smart appliances (air conditioners, refrigerators, washers, ovens etc.), safety and security systems (sensors, cameras, monitors, etc.), and other smart homes energy equipment, like actuators, thermostats. and smart lighting. These physical devices that are being connected to the internet are helping to accelerate the development of the IoTs and further due to their importance, these are becoming a mandatory part of our lives as well. A vibrant society deals with diversified information where safe, economic, comfortable and convenient life has become ideal for every member and family. The wide range of IoT-based applications are improving our quality of lives in many ways where Home Automation (HA) is one of them..

1.1 Objective

Smart homes allow you to have **greater control of your energy use**, all while automating things like adjusting temperature, turning on and off lights, opening and closing window treatments. The project Voice controlled home automation project **helps to control the electrical loads based on ThingSpeak**

CHAPTER-2

LITERATURE SURVEY

Title:1Smart Home-Control and Monitoring System Using Smart Phone

Author:Rajeev Piyare1 and Seong Ro Lee2013

This paper presents a low cost and flexible home control and monitoring system using an embedded micro-web server, with connectivity for accessing and controlling devices and appliances remotely using Android based Smart phone app. The proposed system does not require a dedicated server PC with respect to similar systems and offers a novel communication protocol to monitor and control the home environment with more than just the switching functionality.

Title:2Internet of Things: Ubiquitous Home Control andMonitoring System using Android based Smart Phone

Author:Rajeev Piyare2013

This paper presents a low cost and flexible home control and monitoring system using an embedded micro-webserver, with IP connectivity for accessing and controlling devices and appliances remotely using Android based Smart phoneapp. The proposed system does not require a dedicated server PC with respect to similar systems and offers a novel communication protocol to monitor and control the home environment with more than just the switching functionality.

To demonstrate the feasibility and effectiveness of this system, devices such as light switches, power plug, temperature sensor and current sensor have been integrated with the proposed home control system.

Title:3Internet-of-Things based Smart Home Automation System using Android Phone

Author:Salman Iqbal¹ ,Zubair Sharif^{1,2} , Malik Ali Shahid¹ , and Muhammad Zahid Abbas¹²⁰²¹

A smart home automation system can be greatly advantageous for energy saving and management where it not only helps to save money but can be highly beneficial for the environment. It offers optimum consumption of the home resources but also leads to higher comfort and conveniences for the users and residents. Further integration of Internet of Things (IoTs) services by embedding intelligence into sensors and actuators and connecting them smartly are gaining great popularity to enhance the comfort and quality of life.

CHAPTER 3

INVESTIGATION:

3.1 EXISTING SYSTEM:

Design and Realization of Home Appliances Control System Based on The Android Smartphone present the information about the remote appliances control system based on the Android smart phone is designed and realized. A user logs into the smart phone interface, and clicks the buttons gently to send message commands which will be transmitted to home information Centre through the GSM network. Then the PIC processor recognizes the specified command, and controls the home appliance switches in the wireless radio frequency manner to achieve remote control of appliances ultimately. Exploiting Bluetooth on android mobile devices for home security application present the information about mobile device has been integrated into our everyday life. Home automation and security are becoming increasingly prominent features on mobile devices the mobile device and security system communicates via Bluetooth because a short-range-only communication system was desired. With the help of android mobile we can control task such as locking the doors, turning on/off lights remotely.

Disadvantages

- It's difficult to link systems from different vendors. Many manufacturers develop distinct systems, making it difficult to integrate new devices with your existing ones made by a different brand. ...
- The systems can be costly. ...
- These devices have security flaws and can be hacked.

3.2 Proposed system:

Concluding, in this system, an IOT based hardware system is proposed which is used for smart home automation and with the same single MCU hardware. Unlike the other trends in traditional IOT based automation, this system mainly focuses on hardware optimization and energy efficiency. For other devices, separate micro-controllers are used for home automation or thereby saving energy using just single hardware. With easy user access and security, it would give a significant contribution to the advancement of IOTbased automation systems.

3.2.1 Advantages

- Maximizing home security.
- Remote control of home functions.
- Increased energy efficiency.
- Improved appliance functionality.
- Home management insights

3.3Block diagram:

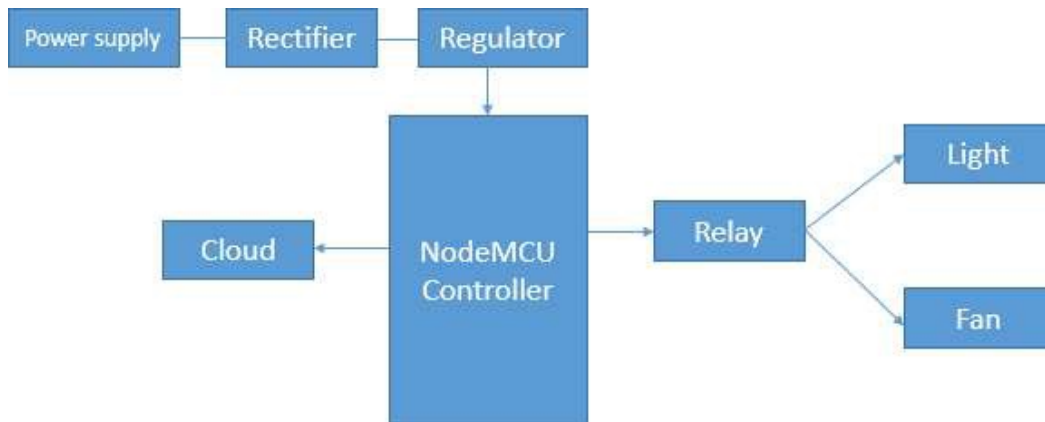


fig 3.1 block diagram

3.4 Hardware used:

- NODEMCU Controller
- RELAYS
- Fan
- LCD Display
- Light
- Power Supply
- Thingspeak

CHAPTER 4

HARDWARE DESCRIPTION

4.1 IOT (Internet of things)

IoT stands for Internet of Things, which means accessing and controlling daily usable equipments and devices using Internet.

Our IoT tutorial includes all topics of IoT such as introduction, features, advantage and disadvantage, ecosystem, decision framework, architecture and domains, biometric, security camera and door unlock system, devices, etc

4.1.1 What is an Internet of Things (IoT)

Let's us look closely at our mobile device which contains GPS Tracking, Mobile Gyroscope, Adaptive brightness, Voice detection, Face detection etc. These components have their own individual features, but what about if these all communicate with each other to provide a better environment? For example, the phone brightness is adjusted based on my GPS location or my direction.

Connecting everyday things embedded with electronics, software, and sensors to internet enabling to collect and exchange data without human interaction called as the Internet of Things (IoT).

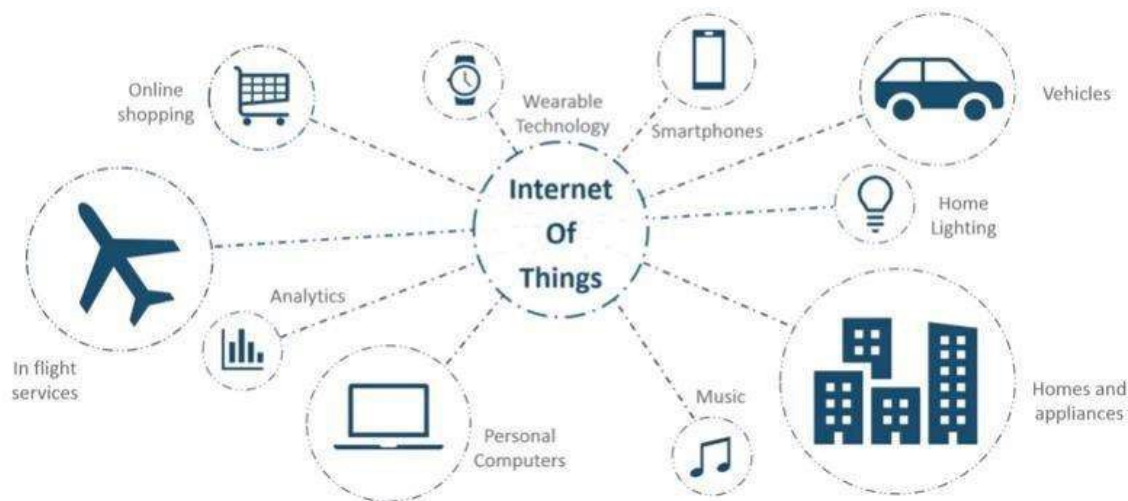


Fig 4.1 IoT figure

IoT is an advanced automation and analytics system which deals with artificial intelligence, sensor, networking, electronic, cloud messaging etc. to deliver complete systems for the product or services. The system created by IoT has greater transparency, control, and performance.

As we have a platform such as a cloud that contains all the data through which we connect all the things around us. For example, a house, where we can connect our home appliances such as air conditioner, light, etc. through each other and all these things are managed at the same platform. Since we have a platform, we can connect our car, track its fuel meter, speed level, and also track the location of the car.



Fig 4.2 IoT Cloud figure

If there is a common platform where all these things can connect to each other would be great because based on my preference, I can set the room temperature. For example, if I love the room temperature to be set at 25 or 26-degree Celsius when I reach back home from my office, then according to my car location, my AC would start before 10 minutes I arrive at home. This can be done through the Internet of Things (IoT).

4.1.2 How does Internet of Thing (IoT) Work?

The working of IoT is different for different IoT echo system (architecture). However, the key concept of there working are similar. The entire working process of IoT starts with the device themselves,

such as smartphones, digital watches, electronic appliances, which securely communicate with the IoT platform. The platforms collect and analyze the data from all multiple devices and platforms and transfer the most valuable data with applications to devices.

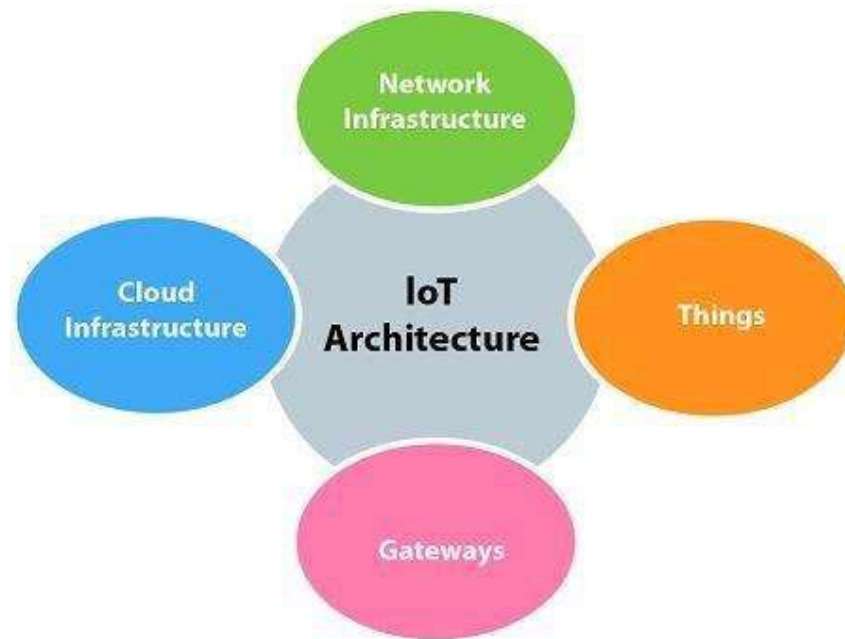


Fig4.3 IoT Architecture

4.1.3 Features of IOT

The most important features of IoT on which it works are connectivity, analyzing, integrating, active engagement, and many more. Some of them are listed below:

Connectivity: Connectivity refers to establish a proper connection between all the things of IoT to IoT platform it may be server or cloud. After connecting the IoT devices, it needs a high speed messaging between the devices and cloud to enable reliable, secure and bi-directional communication.

Analyzing: After connecting all the relevant things, it comes to real-time analyzing the data collected and use them to build effective

business intelligence. If we have a good insight into data gathered from all these things, then we call our system has a smart system.

Integrating: IoT integrating the various models to improve the user experience as well.

Artificial Intelligence: IoT makes things smart and enhances life through the use of data. For example, if we have a coffee machine whose beans have going to end, then the coffee machine itself order the coffee beans of your choice from the retailer.

Sensing: The sensor devices used in IoT technologies detect and measure any change in the environment and report on their status. IoT technology brings passive networks to active networks. Without sensors, there could not hold an effective or true IoT environment.

Active Engagement: IoT makes the connected technology, product, or services to active engagement between each other.

Endpoint Management: It is important to be the endpoint management of all the IoT system otherwise, it makes the complete failure of the system. For example, if a coffee machine itself order the coffee beans when it goes to end but what happens when it orders the beans from a retailer and we are not present at home for a few days, it leads to the failure of the IoT system. So, there must be a need for endpoint management.

4.1.4 Advantages and Disadvantages of (IoT)

Any technology available today has not reached to its 100 % capability. It always has a gap to go. So, we can say that **Internet of Things** has a significant technology in a world that can help other technologies to reach its accurate and complete 100 % capability as well.

Let's take a look over the major, advantages, and disadvantages of the Internet of Things.

Advantages of IoT

Internet of things facilitates the several advantages in day-to-day life in the business sector. Some of its benefits are given below:

- **Efficient resource utilization:** If we know the functionality and the way that how each device work we definitely increase the efficient resource utilization as well as monitor natural resources.
- **Minimize human effort:** As the devices of IoT interact and communicate with each other and do lot of task for us, then they minimize the human effort.
- **Save time:** As it reduces the human effort then it definitely saves out time. Time is the primary factor which can save through IoT platform.
- **Enhance Data Collection:**
- **Improve security:** Now, if we have a system that all these things are interconnected then we can make the system more secure and efficient.

Disadvantages of IoT

As the Internet of things facilitates a set of benefits, it also creates a significant set of challenges. Some of the IoT challenges are given below:

4.1.5 Embedded Devices (System) in (IoT)

It is essential to know about the embedded devices while learning the IoT or building the projects on IoT. The embedded devices are the objects that build the unique computing system. These systems may or may not connect to the Internet.

An embedded device system generally runs as a single application. However, these devices can connect through the internet connection, and able communicate through other network devices.

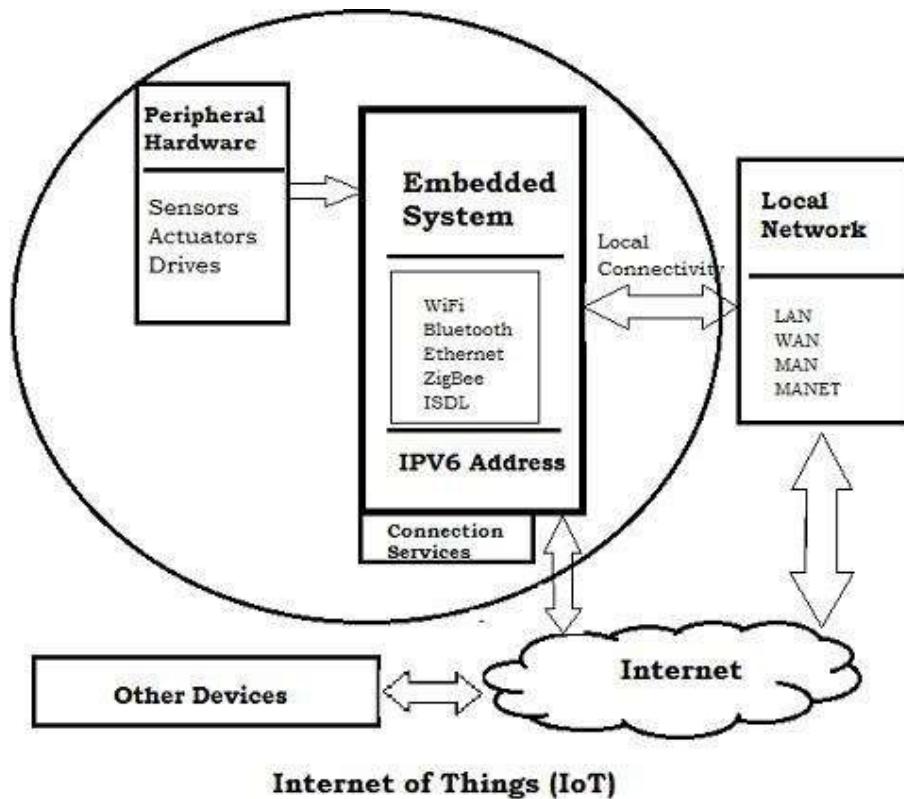


Fig 4.4 Internet of things

4.1.6 Embedded System Hardware

The embedded system can be of type microcontroller or type microprocessor. Both of these types contain an integrated circuit (IC).

The essential component of the embedded system is a RISC family microcontroller like Motorola 68HC11, PIC 16F84, Atmel 8051 and many more. The most important factor that differentiates these microcontrollers with the microprocessor like 8085 is their internal read and writable memory. The essential embedded device components and system architecture are specified below.

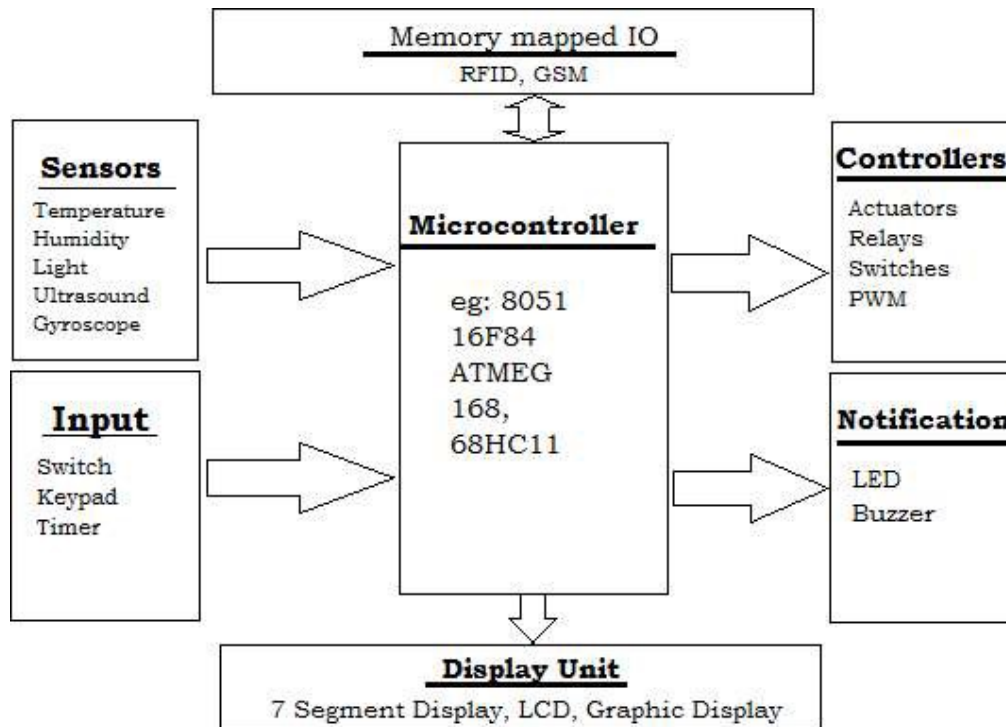


Fig4.5 Basic Embedded System

4.1.6 Embedded System Software

The embedded system that uses the devices for the operating system is based on the language platform, mainly where the real-time operation would be performed. Manufacturers build embedded software in electronics, e.g., cars, telephones, modems, appliances, etc. The embedded system software can be as simple as lighting controls running using an 8-bit microcontroller. It can also be complicated software for missiles, process control systems, airplanes etc.

4.1.8 IoT Ecosystem

The IoT ecosystem is not easy to define. It is also difficult to capture its proper image due to the vastness and emerging possibility and the rapidity with which it is expanding in the entire sector. However, the IoT ecosystem is a connection of various kind of devices that sense

and analyze the data and communicates with each other over the networks.

In the IoT ecosystem, the user uses smart devices such as smartphones, tablet, sensors, etc. to send the command or request to devices for information over the networks. The device response and performs the command to send information back to the user through networks after analyzed.

The typical IoT ecosystem is shown in below image, where the smarter devices send and receive data from the devices themselves in the environment that are integrate over network and Cloud Computing.

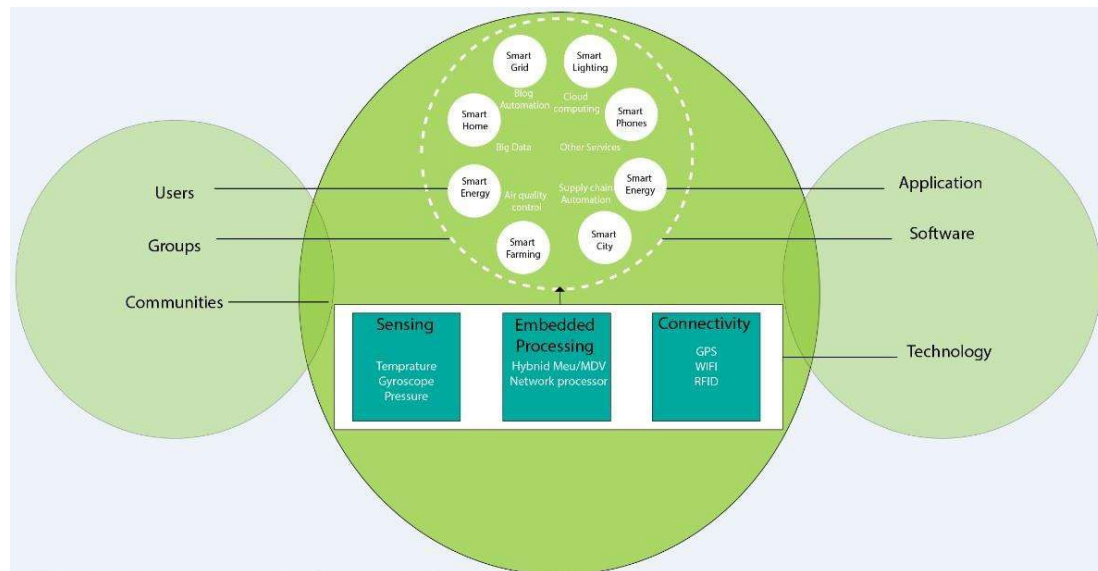


Fig 4.6 Typical IoT ecosystem

The IoT is itself an ecosystem of network devices that transfer the data. It is also well interconnected with Big Data and Cloud Computing.

- **Sensing, Embedded processing, Connectivity:** The IoT ecosystem senses its surrounding like temperature, gyroscope, pressure, etc. and make the embedded processing using devices. These devices are connected through any type of devices such as GPS, WiFi, RFID, etc. over the networks.
- **Smart devices and environment, Cloud Computing, Big Data:** The data transfer or receive through smart devices and environments are communicated through Cloud Computing or others Servers and stored as Big Data.
- **Technology, Software, Application:** The IoT ecosystem uses any of different technologies, software and application to communicate and connect with smart devices and environment.
- **Users or groups of community:** The product or services generated by the IoT ecosystem are consumed by the users or the group of communities to serve the smart life.

4.1.9 IoT Decision Framework

The IoT decision framework provides a structured approach to create a powerful IoT product strategy. The IoT decision framework is all about the strategic decision making. The IoT Decision Framework helps us to understand the areas where we need to make decisions and ensures consistency across all of our strategic business decision, technical and more.

The IoT decision framework is much more important as the product or services communicates over networks goes through five different layers of complexity of technology.

1. Device Hardware
2. Device Software
3. Communications

4. Cloud Platform
5. Cloud Application

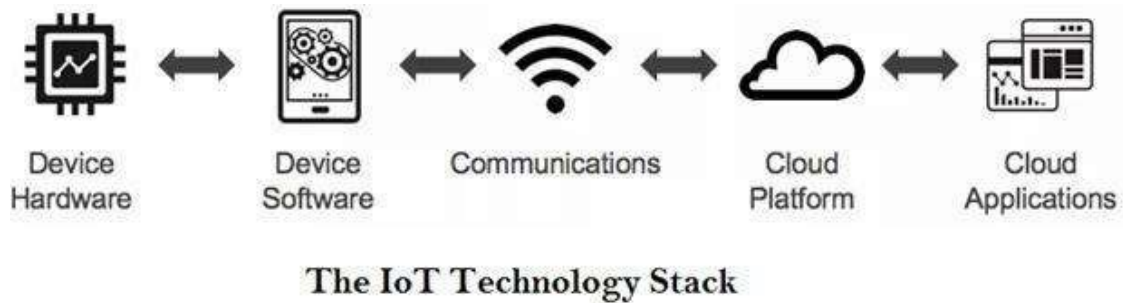


Fig 4.7 IoT Technology Stack

4.1.10 Decision Area

The IoT decision framework pays attention to six key decision areas in any IoT product. These decision areas are:

1. User Experience (UX)
2. Data
3. Business
4. Technology
5. Security
6. Standards & Regulations

Each of these decision areas is evaluated at each of the IoT Technology Stack. The User Experience will be evaluated at Device Hardware, Device Software and so to provide the better user experience. Then at the next step Data Decision Area, we have to explore data considerations for all the stages of IoT Technology Stack.

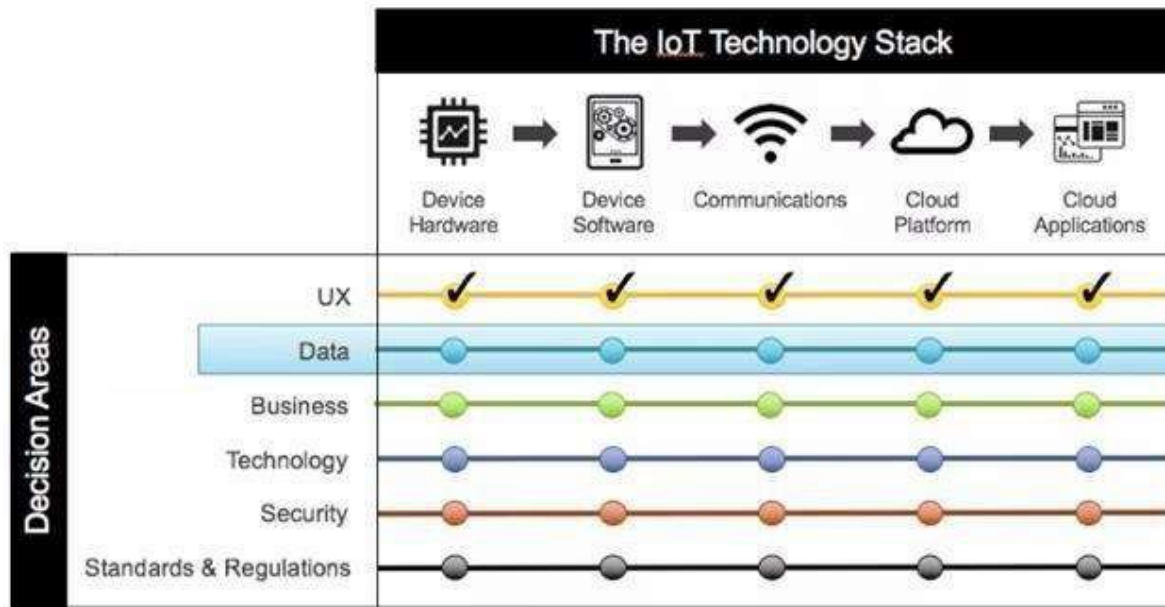


Fig 4.8 Decision Area of the IoT Decision Framework

Let's see each of the Decision Area of IoT Decision Framework in detail:

1. **User Experience Decision Area:** This is the area where we concentrate about who are the users, what are their requirements and how to provide a great experience at each step of IoT stack without worrying about the technical details.
2. **Data Decision Area:** In this area, we make the overall data strategy such as the data flow over the entire IoT stack to fulfill the user's requirements.
3. **Business Decision Area:** Based on the previous decisions area, we make the decision how product or services will become financial potential. At each of the IoT Stack level are monetized about the costs of providing services.
4. **Technology Decision Area:** In this area, we work with the technology for each layer to facilitate the final solution.

5. **Security Decision Area:** After going through the implementation of technology it is important to decide and provide the security at each stage of the IoT Stack.
6. **Standards & Regulations Decision Area:** At the last stage of IoT Decision Area, we identify the standards and regulations of product or services that will affect your product at each layer of the IoT Stack.

4.1.11 IoT Architecture

There is not such a unique or standard consensus on the Internet of Things (IoT) architecture which is universally defined. The IoT architecture differs from their functional area and their solutions. However, the IoT architecture technology mainly consists of four major components:

Components of IoT Architecture

- Sensors/Devices
- Gateways and Networks
- Cloud/Management Service Layer
- Application Layer

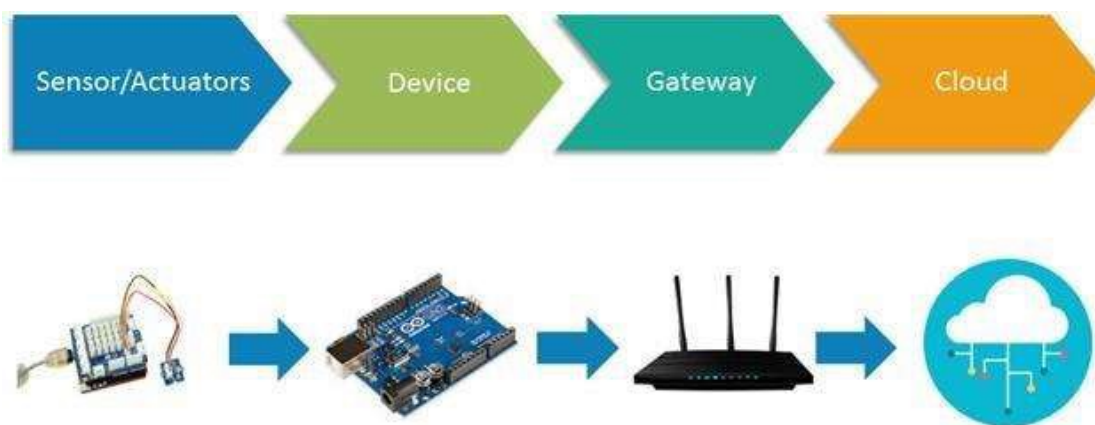


Fig 4.9 Stages of IoT Solutions Architecture

There are several layers of IoT built upon the capability and performance of IoT elements that provides the optimal solution to the business enterprises and end-users. The IoT architecture is a fundamental way to design the various elements of IoT, so that it can deliver services over the networks and serve the needs for the future.

Following are the primary stages (layers) of IoT that provides the solution for IoT architecture.

1. **Sensors/Actuators:** Sensors or Actuators are the devices that are able to emit, accept and process data over the network. These sensors or actuators may be connected either through wired or wireless. This contains GPS, Electrochemical, Gyroscope, RFID, etc. Most of the sensors need connectivity through sensors gateways. The connection of sensors or actuators can be through a Local Area Network (LAN) or Personal Area Network.
2. **Gateways and Data Acquisition:** As the large numbers of data are produced by this sensors and actuators need the high-speed Gateways and Networks to transfer the data. This network can be of type Local Area Network (LAN such as WiFi, Ethernet, etc.), Wide Area Network (WAN such as GSM, 5G, etc.).
3. **Edge IT:** Edge in the IoT Architecture is the hardware and software gateways that analyze and pre-process the data before transferring it to the cloud. If the data read from the sensors and gateways are not changed from its previous reading value then it does not transfer over the cloud, this saves the data used.
4. **Data center/ Cloud:** The Data Center or Cloud comes under the Management Services which process the information through analytics, management of device and security controls. Beside this security controls and device management the cloud transfer the data to the end users application such as Retail, Healthcare, Emergency, Environment, and Energy, etc

The 4 Stage IoT Solutions Architecture

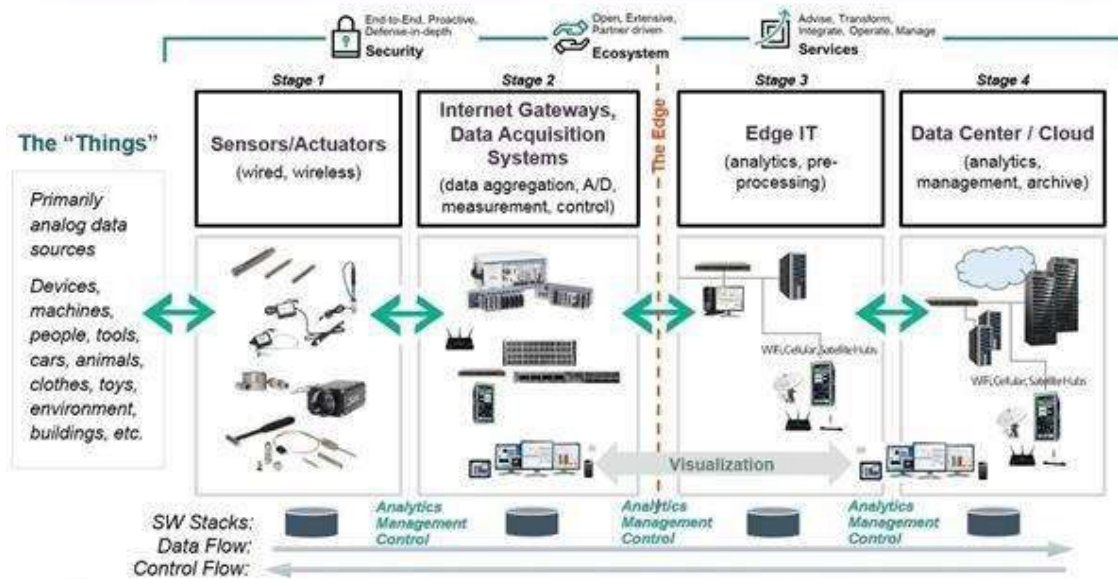


Fig 4.10 The 4 Stages IoT Solutions Architecture

What are Smarts Objects in IoT

The concept of smart in IoT is used for physical objects that are active, digital, networked, can operate to some extent autonomously, reconfigurable and has local control of the resources. The smart objects need energy, data storage, etc.

A **smart object** is an object that enhances the interaction with other smart objects as well as with people also. The world of IoT is the network of interconnected heterogeneous objects (such as smart devices, smart objects, sensors, actuators, RFID, embedded computers, etc.) uniquely addressable and based on standard communication protocols.

In a day to day life, people have a lot of object with internet or wireless or wired connection. Such as:

- Smartphone
- Tablets
- TV computer

These objects can be interconnected among them and facilitate our daily life (smart home, smart cities) no matter the situation, localization, accessibility to a sensor, size, scenario or the risk of danger.



Fig 4.11 Smart objects

Smart objects are utilized widely to transform the physical environment around us to a digital world using the Internet of things (IoT) technologies.

A smart object carries blocks of application logic that make sense for their local situation and interact with human users. A smart object sense, log, and interpret the occurrence within themselves and the environment, and intercommunicate with each other and exchange information with people.

The work of smart object has focused on technical aspects (such as software infrastructure, hardware platforms, etc.) and application scenarios. Application areas range from supply-chain management and enterprise applications (home and hospital) to healthcare and

industrial workplace support. As for human interface aspects of smart-object technologies are just beginning to receive attention from the environment.

4.1.12 IoT Devices

Internet of Things Devices is non-standard devices that connect wirelessly to a network with each other and able to transfer the data. IoT devices are enlarging the internet connectivity beyond standard devices such as smartphones, laptops, tablets, and desktops. Embedding these devices with technology enable us to communicate.



Fig 4.12 IoT Devices

There are large varieties of IoT devices available based on IEEE 802.15.4 standard. These devices range from wireless motes, attachable sensor-boards to interface-board which are useful for researchers and developers.

IoT devices include computer devices, software, wireless sensors, and actuators. These IoT devices are connected over the internet and enabling the data transfer among objects or people automatically without human intervention.

Some of the common and popular IoT devices are given below:

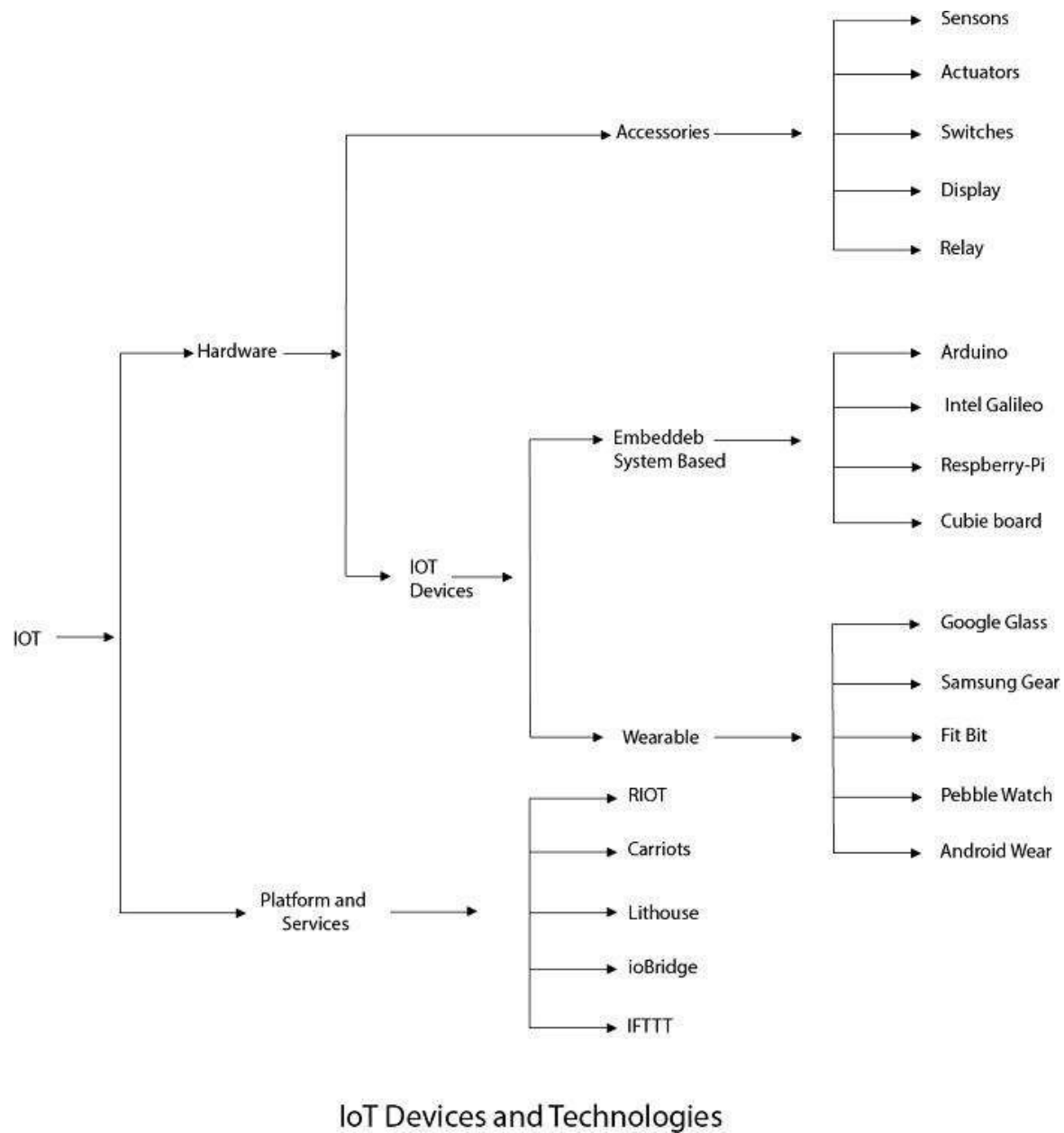


Fig 4.13 IoT Devices and Technologies

4.2 NODE MCU

4.2.1 NodeMCU is an open source IoT platform.^{[4][5]} It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.^{[6][7]} 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 eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson^[8] and SPIFFS

NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems^[6] began production of the ESP8266.^[10] The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core,^[citation needed] widely used in IoT applications (see related projects). NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub.^[11] Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9.^[12] Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform,^[13] and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib^[14] to NodeMCU project,^[15] enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the NodeMCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs.

ESP8266 ArduinoCore^[edit]

As Arduino.cc began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the Arduino IDE so that it would be relatively easy to change the IDE to support alternate toolchains to allow Arduino C/C++ to be compiled for these new processors. They

did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an ArduinoC/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 WiFiSoC, popularly called the "ESP8266 Core for the Arduino IDE".^[16] This has become a leading software development platform for the various ESP8266-based modules and development boards, including NodeMCUs.

introduction

NodeMCU is an open source LUA based firmware developed for ESP8266 wifi chip. By exploring functionality with ESP8266 chip, NodeMCU firmware comes with ESP8266 Development board/kit i.e. NodeMCU Development board.

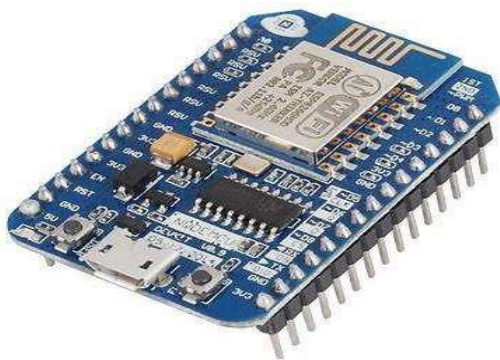


Fig 4.14 NodeMCU

NodeMCU Development Board/kit v0.9 (Version1)

Since NodeMCU is open source platform, their hardware design is open for edit/modify/build.

NodeMCU Dev Kit/board consist of ESP8266 wifi enabled chip. The **ESP8266** is a low-cost Wi-Fi chip developed by Espressif

Systems with TCP/IP protocol. For more information about ESP8266, you can refer ESP8266 WiFi Module.

There is Version2 (V2) available for NodeMCU Dev Kit i.e. **NodeMCU Development Board v1.0 (Version2)**, which usually comes in black colored PCB.



NodeMCU Development Board/kit v1.0 (Version2)

For more information about NodeMCU Boards available in market refer NodeMCU Development Boards

NodeMCU Dev Kit has **Arduino like** Analog (i.e. A0) and Digital (D0-D8) pins on its board.

It supports serial communication protocols i.e. UART, SPI, I2C etc.

Using such serial protocols we can connect it with serial devices like I2C enabled LCD display, Magnetometer HMC5883, MPU-6050 Gyro meter + Accelerometer, RTC chips, GPS modules, touch screen displays, SD cards etc.

How to start with NodeMCU?

NodeMCU Development board is featured with wifi capability, analog pin, digital pins and serial communication protocols.

To get start with using NodeMCU for IoT applications first we need to know about how to write/download NodeMCU firmware in NodeMCU Development Boards. And before that where this NodeMCU firmware will get as per our requirement.

There is online NodeMCU custom builds available using which we can easily get our custom NodeMCU firmware as per our requirement.

To know more about how to build custom NodeMCU firmware online and download it refer [Getting started with NodeMCU](#)

How to write codes for NodeMCU?

After setting up ESP8266 with Node-MCU firmware, let's see the IDE (Integrated Development Environment) required for development of NodeMCU.

NodeMCU with ESPlorer IDE

Lua scripts are generally used to code the NodeMCU. Lua is an open source, lightweight, embeddable scripting language built on top of C programming language.

For more information about how to write Lua script for NodeMCU refer [Getting started with NodeMCU using ESPlorerIDE](#)

NodeMCU with Arduino IDE

Here is another way of developing NodeMCU with a well-known IDE i.e. Arduino IDE. We can also develop applications on NodeMCU using Arduino development environment. This makes easy for Arduino developers than learning new language and IDE for NodeMCU.

For more information about how to write Arduino sketch for NodeMCU refer [Getting started with NodeMCU using ArduinoIDE](#)

Difference in using ESPlorer and Arduino IDE

Well, there is a programming language difference we can say while developing application for NodeMCU using ESPlorer IDE and Arduino IDE.

We need to code in C\C++ programming language if we are using Arduino IDE for developing NodeMCU applications and Lua language if we are using ESPlorer IDE.

Basically, NodeMCU is Lua Interpreter, so it can understand Lua script easily. When we write Lua scripts for NodeMCU and send/upload it to NodeMCU, then they will get executes sequentially. It will not build binary firmware file of code for NodeMCU to write. It will send Lua script as it is to NodeMCU to get execute.

In Arduino IDE when we write and compile code, ESP8266 toolchain in background creates binary firmware file of code we wrote. And when we upload it to NodeMCU then it will flash all NodeMCU firmware with newly generated binary firmware code. In fact, it writes the complete firmware.

That's the reason why NodeMCU not accept further Lua scripts/code after it is getting flashed by Arduino IDE. After getting flashed by Arduino sketch/code it will be no more Lua interpreter and we got error if we try to upload Lua scripts. To again start with Lua script, we need to flash it with NodeMCU firmware.

Since Arduino IDE compile and upload/writes complete firmware, it takes more time than ESPlorer IDE.

Relay:

A relay is an electrically operated switch. Current flowing through the coil of the relay creates a magnetic field which attracts a

lever and changes the switch contacts. The coil current can be on or off so relays have two switch positions and they are doublethrow (changeover) switches. Relays allow one circuit to switch a second circuit which can be completely separate from the first. For example a low voltage battery circuit can use a relay to switch a 230V AC mains circuit. There is no electrical connection inside the relay between the two circuits; the link is magnetic and mechanical.

The coil of a relay passes a relatively large current, typically 30mA for a 12V relay, but it can be as much as 100mA for relays designed to operate from lower voltages. Most ICs (chips) cannot provide this current and a transistor is usually used to amplify the small IC current to the larger value required for the relay coil. The maximum output current for the popular 555 timer IC is 200mA so these devices can supply relay coils directly without amplification.



Fig 4.15 Relay

Relays are usually SPDT or DPDT but they can have many more sets of switch contacts, for example relays with 4 sets of changeover contacts are readily available. Most relays are designed for PCB mounting but you can solder wires directly to the pins providing you take care to avoid melting the plastic case of the relay. The animated picture shows a working relay with its coil and switch contacts. You can see a lever on the left being attracted by magnetism when the coil is switched on. This lever moves the switch contacts.

There is one set of contacts (SPDT) in the foreground and another behind them, making the relay DPDT.



The relay's switch connections are usually labeled COM, NC and NO:

- **COM** = Common, always connect to this, it is the moving part of the switch.
- **NC** = Normally Closed, COM is connected to this when the relay coil is **off**.
- **NO** = Normally Open, COM is connected to this when the relay coil is **on**.

POWER SUPPLY

All electronic circuits work only in low DC voltage, so we need a power supply unit to provide the appropriate voltage supply for their proper functioning. This unit consists of transformer, rectifier, filter & regulator. AC voltage of typically 230volts rms is connected to a transformer voltage down to the level to the desired ac voltage. A diode rectifier that provides the full wave rectified voltage that is initially filtered by a simple capacitor filter to produce a dc voltage. This resulting dc voltage usually has some ripple or ac voltage variation. A regulator circuit can use this dc input to provide dc voltage that not only has much less ripple voltage but also remains the same dc value even the dc voltage varies some what, or the load connected to the output dc voltages changes.

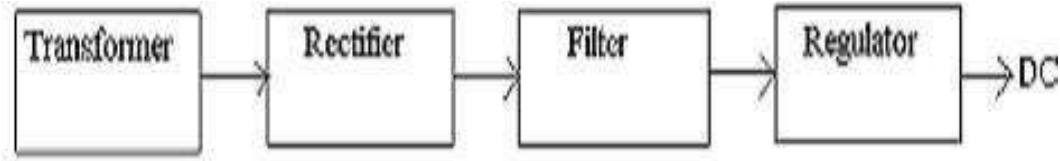


Fig 4.16 General Block Of Power Supply Unit

TRANSFORMER:

A transformer is a static piece of which electric power in one circuit is transformed into electric power of same frequency in another circuit. It can raise or lower the voltage in the circuit, but with a corresponding decrease or increase in current. It works with the principle of mutual induction. In our project we are using a step down transformer to providing a necessary supply for the electronic circuits. Here we step down a 230volts ac into 12volts ac.

RECTIFIER:

A dc level obtained from a sinusoidal input can be improved 100% using a process called full wave rectification. Here in our project for full wave rectification we use bridge rectifier. From the basic bridge configuration we see that two diodes(say D2 & D3) are conducting while the other two diodes (D1 & D4) are in off state during the period $t = 0$ to $T/2$. Accordingly for the negative cycle of the input the conducting diodes are D1 & D4 .Thus the polarity across the load is the same.

In the bridge rectifier the diodes may be of variable types like 1N4001, 1N4003, 1N4004, 1N4005, 1N4007 etc.... can be used. But here we use 1N4007, because it can withstand up to 1000v.

FILTERS:

In order to obtain a dc voltage of 0 Hz, we have to use a low pass filter. So that a capacitive filter circuit is used where a capacitor is connected at the rectifier output & a dc is obtained across it. The filtered waveform is essentially a dc voltage with negligible ripples & it is ultimately fed to the load.

REGISTERS:

The controller IC has two 8 bit registers, an instruction register (IR) and a data register (DR). The IR stores the instruction codes and address information for display data RAM (DD RAM) and character generator RAM (CG RAM). The IR can be written, but not read by the MPU.

The DR temporally stores data to be written to /read from the DD RAM or CG RAM. The data written to DR by the MPU, is automatically written to the DD RAM or CG RAM as an internal operation.

When an address code is written to IR, the data is automatically transferred from the DD RAM or CG RAM to the DR. data transfer between the MPU is then completed when the MPU reads the DR. likewise, for the next MPU read of the DR, data in DD RAM or CG RAM at the address is sent to the DR automatically. Similarly, for the MPU write of the DR, the next DD RAM or CG RAM address is selected for the write operation.

The dot-matrix liquid crystal display controller and driver LSI displays alphanumeric, Japanese kana characters, and symbols. It can be configured to drive a dot-matrix liquid crystal display under the control of a 4- or 8-bit microprocessor. Since all the functions such as display RAM, character generator, and liquid crystal driver, required for driving a dot-matrix liquid crystal display are internally provided on one chip, a minimal system can be interfaced with this controller/driver

REGULATORS:

The output voltage from the capacitor is more filtered & finally regulated. The voltage regulator is a device, which maintains the output voltage constant irrespective of the change in supply variations, load variations & temperature changes. Here we use fixed voltage regulator namely LM7805. The IC LM7805 is a +5v regulator which is used for microcontroller.

FEATURES AND DESCRIPTION OF REGULATORS

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

Software requirements

ARDUINO UNO AND ITS PROGRAMMING

Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer. It's an open-

source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board.

Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs. Arduino projects can be stand-alone, or they can be communicate with software running on your computer. The boards can be assembled by hand or purchased preassembled; the open-source IDE can be downloaded for free.

The Arduino programming language is an implementation of Wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment.

2.1 Overview

The Arduino microcontroller is an easy to use yet powerful single board computer that has gained considerable traction in the hobby and professional market. The Arduino is open-source, which means hardware is reasonably priced and development software is free. This guide is for students in ME 2011, or students anywhere who are confronting the Arduino for the first time. For advanced Arduino users, prowl the web; there are lots of resources.

This guide covers the Arduino Uno board (Spark fun DEV-09950, \$29.95), a good choice for students and educators. With the Arduino board, you can write programs and create interface circuits to read switches and other sensors, and to control motors and lights with very little effort. Many of the pictures and drawings in this guide were taken from the documentation on the

This is what the Arduino ide looks like.



The Duemilanove board features an Atmel ATmega328 microcontroller operating at 5 V with 2 Kb of RAM, 32 Kb of flash memory for storing programs and 1 Kb of EEPROM for storing parameters. The clock speed is 16 MHz, which translates to about executing about 300,000 lines of C source code per second. The board has 14 digital I/O pins and 6 analog input pins. There is a USB connector for talking to the host computer and a DC power jack for connecting an external 6-20 V power source, for example a 9 V battery, when running a program while not connected to the host computer. Headers are provided for interfacing to the I/O pins using 22 g solid wire or header connectors.

The Arduino programming language is a simplified version of C/C++. If you know C, programming the Arduino will be familiar. If you do not know C, no need to worry as only a few commands are needed to perform useful functions.

An important feature of the Arduino is that you can create a control program on the host PC, download it to the Arduino and it will run automatically. Remove the USB cable connection to the PC, and the program will still run from the top each time you push the reset button. Remove the battery and put the Arduino board in a closet for six months. When you reconnect the battery, the last program you stored will run. This means that you connect the board to the host PC to develop and debug your program, but once that is done, you no longer need the PC to run the program.

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

Arduino Software (IDE)

- Writing Sketches
 - File
 - Edit
 - Sketch
 - Tools
 - Help
- Sketchbook

- Tabs, Multiple Files, and Compilation
- Uploading
- Libraries
- Third-Party Hardware
- Serial Monitor
- Preferences
- Language Support
- Boards

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.

Writing Sketches

Programs written using Arduino Software (IDE) are called **sketches**. These sketches are written in the text editor and are saved with the file extension `.ino`. The editor has features for cutting/pasting and for searching/replacing text. The message area gives feedback while saving and exporting and also displays errors. The console displays text output by the Arduino Software (IDE), including complete error messages and other information. The bottom righthand corner of the window displays the configured board and serial port. The toolbar buttons allow you to verify and upload programs, create, open, and save sketches, and open the serial monitor.

NB: Versions of the Arduino Software (IDE) prior to 1.0 saved sketches with the extension `.pde`. It is possible to open these files with version 1.0, you will be prompted to save the sketch with the `.ino` extension on save.



Verify

Checks your code for errors compiling it.



Upload

Compiles your code and uploads it to the configured board.
See uploading below for details.

Note: If you are using an external programmer with yourboard, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"



New

Creates a new sketch.



Open

Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the **File | Sketchbook** menu instead.



Save

Saves your sketch.



Serial

Monitor

Opens the serial monitor.

Additional commands are found within the five menus: **File, Edit, Sketch, Tools, Help**. The menus are context

sensitive, which means only those items relevant to the work currently being carried out are available.

File

- **New**
Creates a new instance of the editor, with the bare minimum structure of a sketch already in place.
- **Open**
Allows to load a sketch file browsing through the computer drives and folders.
- **Open** Recent
Provides a short list of the most recent sketches, ready to be opened.
- **Sketchbook**
Shows the current sketches within the sketchbook folder structure; clicking on any name opens the corresponding sketch in a new editor instance.
- **Examples**
Any example provided by the Arduino Software (IDE) or library shows up in this menu item. All the examples are structured in a tree that allows easy access by topic or library.
- **Close**
Closes the instance of the Arduino Software from which it is clicked.
- **Save**
Saves the sketch with the current name. If the file hasn't been named before, a name will be provided in a "Save as.." window.
- **Save** as...
Allows to save the current sketch with a different name.
- **Page** Setup
It shows the Page Setup window for printing.
- **Print**
Sends the current sketch to the printer according to the settings defined in Page Setup.

- Preferences
Opens the Preferences window where some settings of the IDE may be customized, as the language of the IDE interface.
- Quit
Closes all IDE windows. The same sketches open when Quit was chosen will be automatically reopened the next time you start the IDE.

Edit

- Undo/Redo
Goes back of one or more steps you did while editing; when you go back, you may go forward with Redo.
- Cut
Removes the selected text from the editor and places it into the clipboard.
- Copy
Duplicates the selected text in the editor and places it into the clipboard.
- Copy for Forum
Copies the code of your sketch to the clipboard in a form suitable for posting to the forum, complete with syntax coloring.
- Copy as HTML
Copies the code of your sketch to the clipboard as HTML, suitable for embedding in web pages.
- Paste
Puts the contents of the clipboard at the cursor position, in the editor.
- Select All
Selects and highlights the whole content of the editor.
- Comment/Uncomment
Puts or removes the // comment marker at the beginning of each selected line.

- Increase/Decrease Indent
Adds or subtracts a space at the beginning of each selected line, moving the text one space on the right or eliminating a space at the beginning.
- Find
Opens the Find and Replace window where you can specify text to search inside the current sketch according to several options.
- Find Next
Highlights the next occurrence - if any - of the string specified as the search item in the Find window, relative to the cursor position.
- Find Previous
Highlights the previous occurrence - if any - of the string specified as the search item in the Find window relative to the cursor position.

Sketch

- Verify/Compile
Checks your sketch for errors compiling it; it will report memory usage for code and variables in the console area.
- Upload
Compiles and loads the binary file onto the configured board through the configured Port.
- Upload Using Programmer
This will overwrite the bootloader on the board; you will need to use Tools > Burn Bootloader to restore it and be able to Upload to USB serial port again. However, it allows you to use the full capacity of the Flash memory for your sketch. Please note that this command will NOT burn the fuses. To do so a Tools -> Burn Bootloader command must be executed.
- Export Compiled Binary
Saves a .hex file that may be kept as archive or sent to the board using other tools.

- Show Sketch Folder
Opens the current sketch folder.
- Include Library
Adds a library to your sketch by inserting `#include` statements at the start of your code. For more details, see libraries below. Additionally, from this menu item you can access the Library Manager and import new libraries from .zip files.
- Add File...
Adds a source file to the sketch (it will be copied from its current location). The new file appears in a new tab in the sketch window. Files can be removed from the sketch using the tab menu accessible clicking on the small triangle icon below the serial monitor one on the right side of the toolbar.

Tools

- Auto Format
This formats your code nicely: i.e. indents it so that opening and closing curly braces line up, and that the statements inside curly braces are indented more.
- Archive Sketch
Archives a copy of the current sketch in .zip format. The archive is placed in the same directory as the sketch.
- Fix Encoding & Reload
Fixes possible discrepancies between the editor char map encoding and other operating systems char maps.
- Serial Monitor
Opens the serial monitor window and initiates the exchange of data with any connected board on the currently selected Port. This usually resets the board, if the board supports Reset over serial port opening.
- Board
Select the board that you're using. See below for descriptions of the various boards.

- **Port**
This menu contains all the serial devices (real or virtual) on your machine. It should automatically refresh every time you open the top-level tools menu.
- **Programmer**
For selecting a hardware programmer when programming a board or chip and not using the onboard USB-serial connection. Normally you won't need this, but if you're burning a bootloader to a new microcontroller, you will use this.
- **Burn** **Bootloader**
The items in this menu allow you to burn a bootloader onto the microcontroller on an Arduino board. This is not required for normal use of an Arduino or Genuino board but is useful if you purchase a new ATmega microcontroller (which normally come without a bootloader). Ensure that you've selected the correct board from the **Boards** menu before burning the bootloader on the target board. This command also set the right fuses.

Help

Here you find easy access to a number of documents that come with the Arduino Software (IDE). You have access to Getting Started, Reference, this guide to the IDE and other documents locally, without an internet connection. The documents are a local copy of the online ones and may link back to our online website.

- **Find** **in** **Reference**
This is the only interactive function of the Help menu: it directly selects the relevant page in the local copy of the Reference for the function or command under the cursor.

Sketchbook

The Arduino Software (IDE) uses the concept of a sketchbook: a standard place to store your programs (or sketches). The sketches in

your sketchbook can be opened from the **File > Sketchbook** menu or from the **Open** button on the toolbar. The first time you run the Arduino software, it will automatically create a directory for your sketchbook. You can view or change the location of the sketchbook location from with the **Preferences** dialog.

Beginning with version 1.0, files are saved with a .ino file extension. Previous versions use the .pde extension. You may still open .pde named files in version 1.0 and later, the software will automatically rename the extension to .ino.

Tabs, Multiple Files, and Compilation

Allows you to manage sketches with more than one file (each of which appears in its own tab). These can be normal Arduino codefiles (no visible extension), C files (.c extension), C++ files (.cpp), or header files (.h).

Uploading

Before uploading your sketch, you need to select the correct items from the **Tools > Board** and **Tools > Port** menus. The boards are described below. On the Mac, the serial port is probably something like **/dev/tty.usbmodem241** (for an Uno or Mega2560 or Leonardo) or **/dev/tty.usbserial-1B1** (for a Duemilanove or earlier USB board), or **/dev/tty.USA19QW1b1P1.1** (for a serial board connected with a Keyspan USB-to-Serial adapter). On Windows, it's probably COM1 or COM2 (for a serial board) or COM4, COM5, COM7, or higher (for a USB board) - to find out, you look for USB serial device in the ports section of the Windows Device Manager. On Linux, it should be **/dev/ttyACMx** , **/dev/ttyUSBx** or similar. Once you've selected the correct serial port and board, press the upload button in the toolbar or select the **Upload** item from the **Sketch** menu. Current Arduino boards will reset automatically and begin the upload. With older

boards (pre-Diecimila) that lack auto-reset, you'll need to press the reset button on the board just before starting the upload. On most boards, you'll see the RX and TX LEDs blink as the sketch is uploaded. The Arduino Software (IDE) will display a message when the upload is complete, or show an error.

When you upload a sketch, you're using the Arduino **bootloader**, a small program that has been loaded on to the microcontroller on your board. It allows you to upload code without using any additional hardware. The bootloader is active for a few seconds when the board resets; then it starts whichever sketch was most recently uploaded to the microcontroller. The bootloader will blink the on-board (pin 13) LED when it starts (i.e. when the board resets).

Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from the **Sketch > Import Library** menu. This will insert one or more **#include** statements at the top of the sketch and compile the library with your sketch. Because libraries are uploaded to the board with your sketch, they increase the amount of space it takes up. If a sketch no longer needs a library, simply delete its **#include** statements from the top of your code.

There is a list of libraries in the reference. Some libraries are included with the Arduino software. Others can be downloaded from a variety of sources or through the Library Manager. Starting with version 1.0.5 of the IDE, you can import a library from a zip file and use it in an open sketch. See these instructions for installing a third-party library.

To write your own library, see this tutorial.

Third-Party Hardware

Support for third-party hardware can be added to the **hardware** directory of your sketchbook directory. Platforms installed there may include board definitions (which appear in the board menu), core libraries, bootloaders, and programmer definitions. To install, create the **hardware** directory, then unzip the third-party platform into its own sub-directory. (Don't use "arduino" as the sub-directory name or you'll override the built-in Arduino platform.) To uninstall, simply delete its directory.

For details on creating packages for third-party hardware, see the Arduino IDE 1.5 3rd party Hardware specification.

Serial Monitor

This displays serial sent from the Arduino or Genuino board over USB or serial connector. To send data to the board, enter text and click on the "send" button or press enter. Choose the baud rate from the drop-down menu that matches the rate passed to **Serial.begin** in your sketch. Note that on Windows, Mac or Linux the board will reset(it will rerun your sketch) when you connect with the serial monitor. Please note that the Serial Monitor does not process control characters; if your sketch needs a complete management of the serial communication with control characters, you can use an external terminal program and connect it to the COM port assigned to your Arduino board.

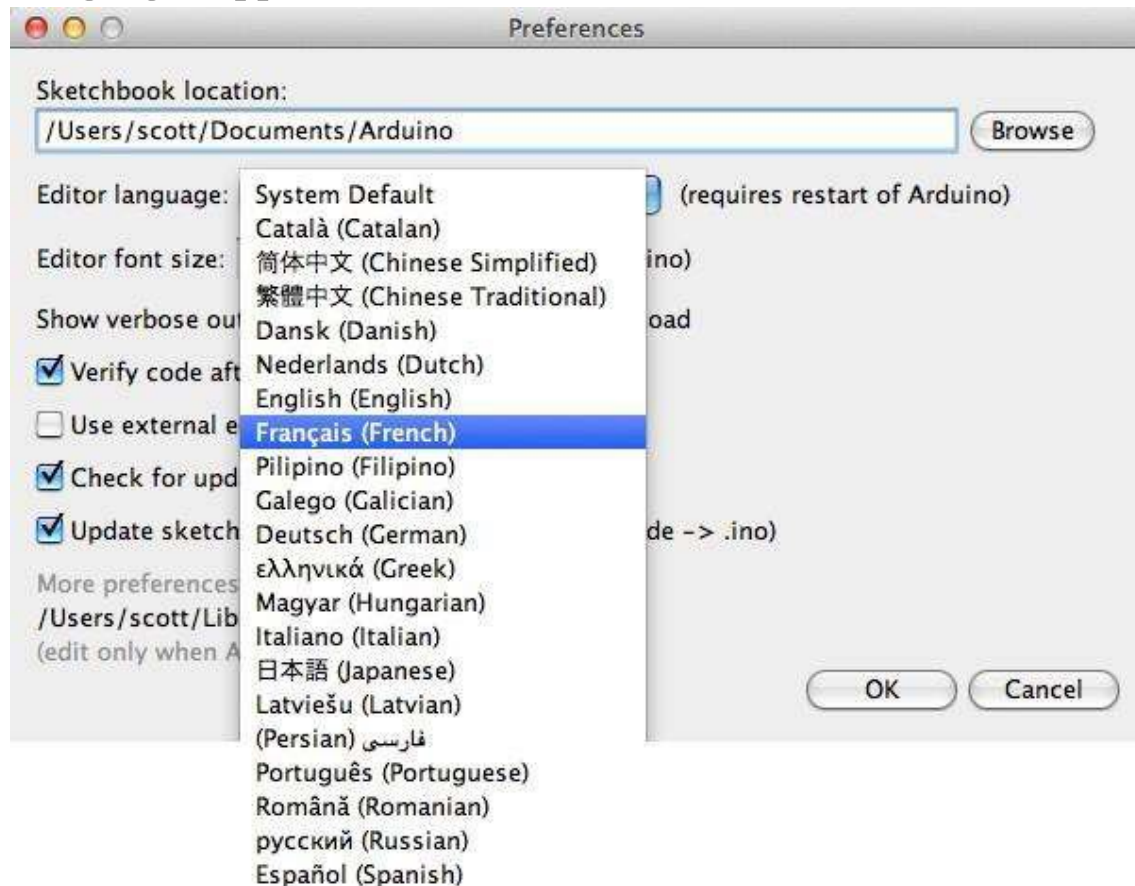
You can also talk to the board from Processing, Flash, MaxMSP, etc (see the interfacing page for details).

Preferences

Some preferences can be set in the preferences dialog (found under the **Arduino** menu on the Mac, or **File** on Windows and Linux). The

rest can be found in the preferences file, whose location is shown in the preference dialog.

Language Support



Since version 1.0.1 , the Arduino Software (IDE) has been translated into 30+ different languages. By default, the IDE loads in the language selected by your operating system. (Note: on Windows and possibly Linux, this is determined by the locale setting which controls currency and date formats, not by the language the operating system is displayed in.)

If you would like to change the language manually, start the Arduino Software (IDE) and open the **Preferences** window. Next to the **Editor Language** there is a dropdown menu of currently supported languages. Select your preferred language from the menu, and restart the software to use the selected language. If your operating

system language is not supported, the Arduino Software (IDE) will default to English.

You can return the software to its default setting of selecting its language based on your operating system by selecting **System Default** from the **Editor Language** drop-down. This setting will take effect when you restart the Arduino Software (IDE). Similarly, after changing your operating system's settings, you must restart the Arduino Software (IDE) to update it to the new default language.

Boards

The board selection has two effects: it sets the parameters (e.g. CPU speed and baud rate) used when compiling and uploading sketches; and sets the file and fuse settings used by the burn bootloader command. Some of the board definitions differ only in the latter, so even if you've been uploading successfully with a particular selection you'll want to check it before burning the bootloader. You can find a comparison table between the various boards [here](#).

Arduino Software (IDE) includes the built in support for the boards in the following list, all based on the AVR Core. The Boards Manager included in the standard installation allows to add support for the growing number of new boards based on different cores like Arduino Due, Arduino Zero, Edison, Galileo and so on.

- ArduinoYùn
An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
- Arduino/Genuino Uno
An ATmega328P running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
- ArduinoDiecimila or Duemilanove w/ ATmega168
An ATmega168 running at 16 MHz with auto-reset.

- Arduino Nano w/ ATmega328P
An ATmega328P running at 16 MHz with auto-reset. Has eight analog inputs.
- Arduino/Genuino Mega 2560
An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
- Arduino Mega
An ATmega1280 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
- Arduino Mega ADK
An ATmega2560 running at 16 MHz with auto-reset, 16 Analog In, 54 Digital I/O and 15 PWM.
- Arduino Leonardo
An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
- Arduino/Genuino Micro
An ATmega32u4 running at 16 MHz with auto-reset, 12 Analog In, 20 Digital I/O and 7 PWM.
- ArduinoEsplora
An ATmega32u4 running at 16 MHz with auto-reset.
- Arduino Mini w/ ATmega328P
An ATmega328P running at 16 MHz with auto-reset, 8 Analog In, 14 Digital I/O and 6 PWM.
- Arduino Ethernet
Equivalent to Arduino UNO with an Ethernet shield: An ATmega328P running at 16 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
- ArduinoFio
An ATmega328P running at 8 MHz with auto-reset. Equivalent to Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328P, 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino BT w/ ATmega328P
ATmega328P running at 16 MHz. The bootloader burned (4 KB)

includes codes to initialize the on-board bluetooth module, 6 Analog In, 14 Digital I/O and 6 PWM..

- LilyPad Arduino USB
An ATmega32u4 running at 8 MHz with auto-reset, 4 Analog In, 9 Digital I/O and 4 PWM.
- LilyPad Arduino
An ATmega168 or ATmega132 running at 8 MHz with auto-reset, 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328P
An ATmega328P running at 16 MHz with auto-reset. Equivalent to ArduinoDuemilanove or Nano w/ ATmega328P; 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino NG or older w/ ATmega168
An ATmega168 running at 16 MHz without auto-reset. Compilation and upload is equivalent to ArduinoDiecimila or Duemilanove w/ ATmega168, but the bootloader burned has a slower timeout (and blinks the pin 13 LED three times on reset); 6 Analog In, 14 Digital I/O and 6 PWM.
- Arduino Robot Control
An ATmega328P running at 16 MHz with auto-reset.
- Arduino Robot Motor
An ATmega328P running at 16 MHz with auto-reset.
- Arduino Gemma
An ATtiny85 running at 8 MHz with auto-reset, 1 Analog In, 3 Digital I/O and 2 PWM.

For instructions on installing support for other boards, see third-party hardware above.

The text of the Arduino getting started guide is licensed under a Creative Commons Attribution-ShareAlike 3.0 License. Code samples in the guide are released into the public domain.

Chapter 5

5.1 Results and Discussion

The experimental model was made according to the circuit diagram and the results were as expected. The home appliances could be remotely switched over Wi-Fi network. Both the switch mode and the voice mode control methodologies were successfully achieved. The Blynk application was also successful in displaying the status of every application.

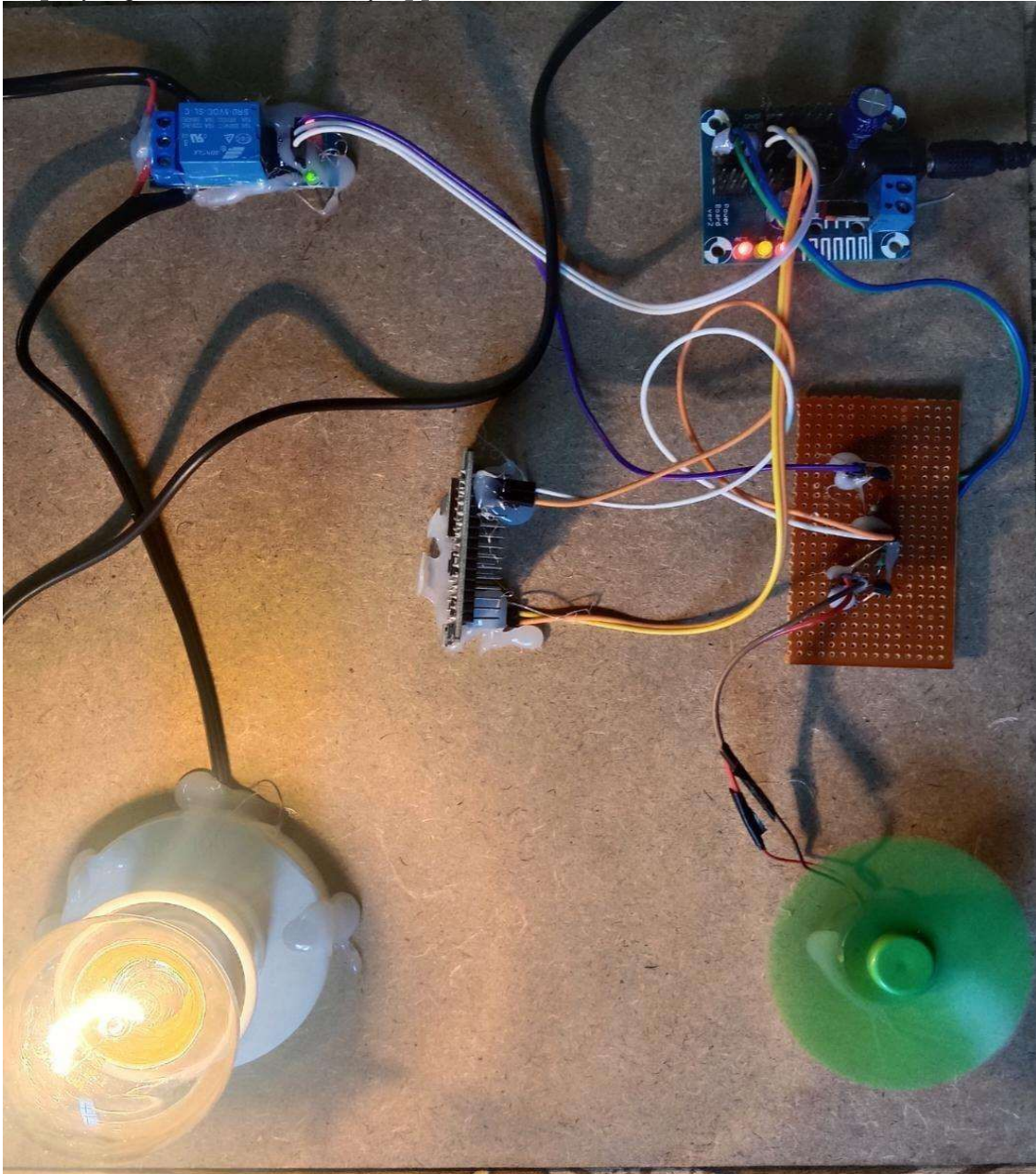


Fig 5.1 Hardware Photo

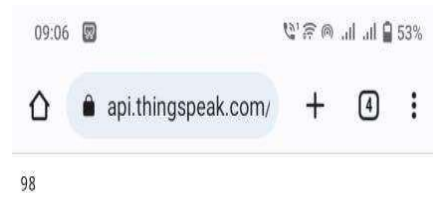
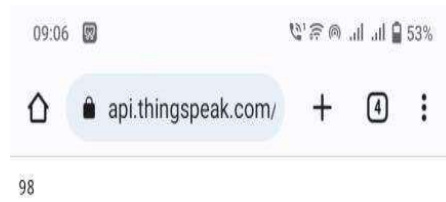
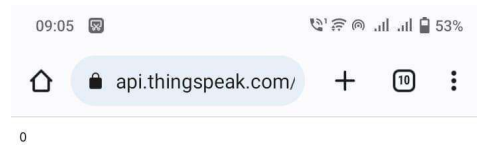
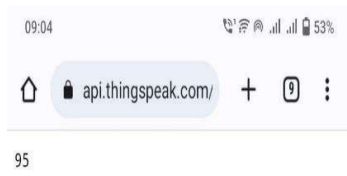


Fig 5.2 Output Photos

Chapter 6

Conclusion And Future scope

A Smart Home technology integrates the electrical devices in a home with one another. In this paper, we've given a low price and versatile home management and monitoring system using android platformed smart phone. The projected design utilizes REST full based net services as a practical application layer for act between the remote user and the home devices. In our project we have a tendency to eliminate most of the human interaction by providing intelligent system. Development of such sensible Home came through with the help of internet of Things technologies. By using this system we can truly manage to make low price, flexible sensible homes to regulate its environmental factors and to resolve the issues with energy saving technology.

References:

[1] OvidiuVermesan, Peter Friess, Internet Of Things Applications: From Research and Innovation To Market Deployment, River Publishers Series In Communications,2014.

- [2] PrahladBhadani, Dr.VVashisht, “Soil Moisture, Temperature, and Humidity Measurement Using Arduino”, International Conference on Cloud Computing, Data Science & Engineering, Jan 2019,pp.567- 571.
- [3] L.Calcara, G.Felice, F.Ferrucci, F.Marchetti, “Soil thermal resistivity: innovative measurement system”, IEEE International Conference on Environment and Electrical Engineering, June 2019.
- [4] N.Agrawal, S.Singhal, "Smart Drip Irrigation System using Raspberry Pi and Arduino", International Conference on Computing, Communication and Automation, May 2015,pp.928-932.
- [5] H.Durani, M.Sheth, M.Vaghasia, S.Kotech, "Smart Automated Home Application using IoT with Blynk App", Proceedings of the 2nd International Conference on Inventive Communication and Computational Technologies, April 2018,pp.393-397.
- [6] M.Gamba, A.Gonella, C.Palazzi, "Design Issues and Solutions in a Modern Home Automation System", 2015 International Conference on Computing, Networking and Communications, Feb. 2015, pp.1111-1115.
- [7] K.Ghanshala, R.Chauhan, R. C Joshi, "A Novel Framework for Smart Crop Monitoring Using the Internet of Things (IoT)", International Conference on Secure Cyber Computing and Communication, Dec. 2018 pp. 62-67.
- [8] S.Ivanović, S.Milivojša, T. Erić, M.Vidaković, "Collection and Analysis of System Usage Data in Smart Home AutomationSystems", International Conference on Consumer Electronics, December 2017, pp. 65-66.
- [9] R.K.Jha, S.Kumar, K.Joshi, R.Pandey, "Field Monitoring Using IoT in Agriculture", International Conference on Intelligent Computing, Instrumentation and Control Technologies, July 2017,pp.1417-1420.

[10] K. Krishna Kishore, M. H. Sai Kumar, M. B. S. Murthy, "Automatic Plant Monitoring System", International Conference on Trends in Electronics and Informatics, 2017, pp. 744-748