# 

# Smart Home Automation

## Major Project Report

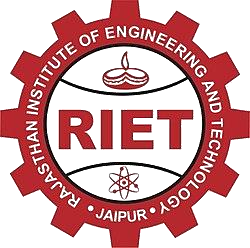
Submitted

in partial fulfilment

for the award of the Degree of

**Bachelor of Technology**

**in Department of Computer Science & Engineering**



## 

**Supervisor** : **Submitted By :**

Dr. Saroj Hiranwal Ankit Malpani

(Professor of Dept. of CSE) Roll No.: 18ERECS009

**Department of Computer Science & Engineering**

Rajasthan Institute of Engineering & Technology, Jaipur

Rajasthan Technical University, Kota

June, 2022

# Candidate’s Declaration

We hereby declare that the work, which is being presented in the Major Project, entitled **“Smart Home Automation”** in partial fulfilment for the award of Degree of “*Bachelor* of Technology” in Dept. Of Computer Science & Engineering submitted to the Department of **Computer Science & Engineering,** Rajasthan Institute of Engineering and Technology, Jaipur Rajasthan Technical University is a record of our own investigations carried under the Guidance of **Dr. Saroj Hiranwal**, Department of Computer Science & Engineering, Rajasthan Institute of Engineering And Technology**.**

We have not submitted the matter presented in this report anywhere for the award of any other Degree.

Vicky Kumar 18ERECS080

Ankit Malpani 18ERECS009

Anil Puri 18ERECS008

Vritika Jangir 18ERECS089

Rajasthan Institute of Engineering and Technology **,** Jaipur

**Counter Signed by**

.....................................

.....................................

# Acknowledgement

We take this opportunity to express our gratitude to all those people who have been directly and indirectly with us during the competition of this project.

We pay thank to **Dr. Saroj Hiranwal** who has given guidance and a light to us during this major project. His versatile knowledge about “**HOME AUTOMATION** and **EMBEDDED SYSTEMS**” has eased us in the critical times during the span of this project. We acknowledge here out debt to those who contributed significantly to one or more steps. We take full responsibility for any remaining sins of omission and commission.

Vicky Kumar

Ankit Malpani

Anil Puri Goswami  
Vritika Jangir

B.Tech IVth Year

(Computer Science & Engineering)

# Abstract

Home automation system achieved great popularity in the last decades and it increases the comfort and quality of life. This report is an overview of current and emerging home automation systems. Nowadays most home automation systems consist of a smartphone and microcontroller. A smart phone application is used to control and monitor the home appliances using different type of communication techniques. In this report the working principle of different type of wireless communication techniques such as Wi-Fi and 6LowWPAN are studied and their features. In this project work the survey of home automation systems is discussed and their advantages and drawbacks are also highlighted.

Due to the rapid development in the field of the Automation industry, human life is becoming more advanced and better in all aspects. In the present scenario, automated systems are being preferred over the non-automated system. With the rapid growth in the number of consumers using the internet over the past years, the Internet has become an important part of life, and IoT is the newest and emerging internet technology. Internet of things plays an important role in human life as well as in the educational field because they are able to provide information and complete the given tasks while we are busy doing some other work. In this report, a prototype and implementation of Smart Home Automation and smart gardening system with Wi-Fi technology are demonstrated. ESP8266 is used as a Wi-Fi technology. The proposed system consist of a hardware interface and software interface. In the hardware interface, the integration of ESP8266 Wi-Fi technology for controlling home appliances and sensors is manifested, and an application is provided for controlling to multiple users of home, with smart phones, tablets, and laptops. This system is one of the best methods for controlling home devices with ease with multiple users and one of the best method for an energy management system. The access to the whole system is given by its admin only to different users. This system is also expandable for controlling various appliances used at home through sensors as long as it exists on Wi-Fi network coverage.

Internet of things (IoT) and Embedded Systems is an emerging technology today that envisions all objects around us as a part of Internet. Automation of the devices, appliances at home and office is having extensive possibility of research with the innovation of technology in communication. Misuse of power energy can be shortened by automating the devices and appliances.

Keywords : Relay, Bluetooth Module HC-05, Arduino ATMEGA8, AC Load

**Contents**

|  |  |  |
| --- | --- | --- |
| **S.no** | **Topics** | **Page No.** |
|  | Acknowledgement | 3 |
|  | Abstract | 4 |
|  | Contents | 5-8 |
|  | List of Figures | 9-10 |
| **Chapter-1** | Introduction |  |
| 1.1 | Internet of thing | 12 |
| 1.2 | History of Internet of Things | 13 |
| 1.3 | Importance of Internet of Things | 13 |
| 1.4 | Embedded System & History | 14 |
| 1.5 | Team Work | 15 |
| 1.5.1 | Vicky Kumar(18ERECS080) | 15 |
| 1.5.2 | Ankit Malpani (18ERECS009) | 18 |
| 1.5.3 | Anil Puri Goswami (18ERECS008) | 20 |
| 1.5.4 | Vritika Jangir (18ERECS089) | 21 |
| **Chapter 2** | Stepping into the World of IOT |  |
| 2..1 | Definition | 24 |
| 2.2 | How Big is the Internet of Things | 24 |
| 2.3 | Building Blocks of IOT | 25 |
| 2.3.1 | Sensors | 26 |
| 2.3.2 | Processors | 26 |
| 2.3.3 | Gateways | 26 |
| 2.3.4 | Applications | 27 |
| 2.4 | Architecture of IOT | 27 |
| 2.5 | Working of IOT | 29 |
| 2.6 | Characteristics of IOT | 30 |
| 2.7 | Pros and Cons of IOT | 31 |
| **Chapter 3** | Arduino |  |
| 3.1 | Introduction | 33 |
| 3.2 | Official Boards | 33 |
| 3.3 | Digital Pins | 34 |
| 3.4 | Analog Pins | 35 |
| 3.5 | Power Pins | 35 |
| 3.6 | Other Pins | 36 |
| 3.7 | Shields | 37 |
| 3.8 | Software | 37 |
| 3.9 | IDE | 38 |
| **Chapter 4** | Components of IOT |  |
| 4.1 | Architecture of Internet of Things | 39 |
| 4.2 | Architecture Reference Model | 39 |
| 4.3 | 5 Layer Architecture of IOT | 40 |
| 4.4 | Architecture Reference model | 40 |
| 4.5 | Information model | 41 |
| 4.6 | Functional model | 41 |
| **Chapter 5** | The Controller |  |
| 5.1 | Controller | 42 |
| 5.2 | The “Brain” of IOT | 43 |
| 5.3 | Adafruit.IO | 44 |
| 5.4 | Dashboard | 45 |
| 5.5 | Steps to design controller | 46 |
| 5.6 | Libraries of Adafruit | 48 |
| 5.7 | Adafruit Air Lift | 49 |
| 5.8 | Adafruit ESP32 | 49 |
| 5.9 | Adafruit Ethernet | 49 |
| 5.10 | Adafruit FONA (Cellular) | 50 |
| **Chapter 6** | Circuit |  |
| 6.1 | Basics of Circuit Designing in IOT | 51 |
| 6.2 | Circuit Designing Rules | 52 |
| 6.3 | Factors affecting IOT Circuit Design | 53 |
| 6.4 | Challenges in IOT Circuit Designing | 54 |
| 6.5 | Space Constraint | 54 |
| 6.6 | Low Power Consumption, Low Range, High Bandwidth | 54 |
| 6.7 | Low Power Consumption, High Range, Low Bandwidth | 54 |
| **Chapter 7** | Home Automation |  |
| 7.1 | Introduction | 55 |
| 7.3 | Implementations | 56 |
| 7..2 | Applications and Technologies | 57 |
| 7.4 | Impact | 58 |
| 7.5 | Future Scope of Home Automation | 59 |
| 7.6 | Libraries of Adafruit included in Project | 59 |
| 7.7 | Setting up Smart Home | 60 |
| 7.8 | Components | 60 |
| 7.8.1 | ESP8266 | 60 |
| 7.8.2 | Moisture Sensor | 61 |
| 7.8.3 | Soil Moisture Comparator | 62 |
| 7.8.4 | Buck Converter | 62 |
| 7.8.5 | Relay Circuit | 62 |
| 7.8.6 | Pump & Bulb | 63 |
| 7.8.7 | ESP8266 Pin Diagram | 63 |
| 7.8.8 | Applications of Home Automation | 64 |
| 7.8.9 | Moisture Sensor Pin Diagram | 65 |
| 7.8.10 | Controller | 65 |
| 7.9 | Adafruit | 66 |
| 7.10 | Steps to create dashboard on Adafruit | 67 |
| 7.11 | Libraries of Adafruit included in Project | 70 |
| 7.12 | Software: Arduino | 72 |
|  | Literature References | 81 |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Figure Name** | **Page No.** |
| 2.1 | IoT endpoint market basement  2018-2020 | 25 |
| 2.2 | Building blocks of IoT | 25 |
| 2.3 | IoT architecture | 28 |
| 2.4 | Working of IoT | 29 |
| 2.5 | Characteristics of IoT | 30 |
| 3.1 | Arduino board diagram | 33 |
| 4.1 | Sensors / devices | 38 |
| 4.2 | Data processing | 39 |
| 5.1 | Simple microcontroller diagram | 43 |
| 5.2 | Diagram showing dashboards | 44 |
| 5.3 | Diagram showing control design | 47 |
| 6.1 | Home automation circuit design | 50 |
| 6.2 | Ohm’s law | 51 |
| 6.3 | Parallel resistance | 52 |
| 6.4 | IoT connectivity | 52 |
| 7.1 | Smart home automation system | 55 |
| 7.2 | Cat feeder | 58 |
| 7.3 | Diagram showing ESP 8266 | 60 |
| 7.4 | Diagram showing moisture sensor | 60 |
| 7.5 | Diagram showing moisture comparator | 61 |
| 7.6 | Buck convertor | 61 |
| 7.7 | Relay circuit | 62 |
| 7.8 | ESP 8266 pin diagram | 62 |
| 7.9 | Moisture pin diagram | 64 |
| 7.10 | Controller dashboard | 65 |
| 7.11 | Adafruit logo | 65 |
| 7.12 | Adafruit interface | 66 |
| 7.13 | Adafruit blocks | 67 |
| 7.14 | Adafruit dashboards | 68 |
| 7.15 | Adafruit final dashboards (controller) | 70 |
| 7.16 | Arduino IDE example | 72 |
| 7.17 | Project circuit diagram | 73 |
| 7.18 | Diagram showing internet connectivity | 74 |
| 7.19 | Project snapshot 1 | 78 |
| 7.20 | Project snapshot 2 | 80 |

**Chapter 1**

**INTRODUCTION**

#### 1.1 Internet of Things (IOT)

The internet of things, or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers ([UIDs)](https://internetofthingsagenda.techtarget.com/definition/unique-identifier-UID) and the ability to transfer data over a network without requiring human-to-human or human-to computer interaction.

A [thing](https://internetofthingsagenda.techtarget.com/definition/thing-in-the-Internet-of-Things) in the internet of things can be a person with a heart monitor implant, a farm animal with a [biochip transponder,](https://internetofthingsagenda.techtarget.com/definition/injectable-ID-chip-biochip-transponder) an automobile that has built-in [sensors](https://whatis.techtarget.com/definition/sensor) to alert the driver when tire pressure is low or any other natural or man-made object that can be assigned an Internet Protocol (IP) address and is able to transfer data over a network.

Increasingly, organizations in [a variety of industries are using IoT to operate more efficiently,](https://searchcustomerexperience.techtarget.com/news/450402550/IoT-technologies-bring-efficiency-and-customization-to-manufacturing) better understand customers to deliver enhanced customer service, improve decision-making and increase the value of the business.

The Internet of Things (IoT) describes the network of physical objects— “things”—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet. These devices range from ordinary household objects to sophisticated industrial tools. With more than 7 billion connected

To be smart, a thing doesn’t need to have super storage or a supercomputer inside of it. All a thing has to do is connect to super storage or to a supercomputer.

The internet of things (IoT) is a catch-all term for the growing number of electronics that aren't traditional computing devices, but are connected to the internet to send data, receive instructions or both.

There's an incredibly broad range of things that fall under that umbrella: Internet-connected "smart" versions of traditional appliances like refrigerators and light bulbs; gadgets that could only exist in an internet-enabled world like Alexa-style digital assistants; internet-enabled

The IoT brings the power of the internet, data processing and analytics to the real world of physical objects. For consumers, this means interacting with the global information network without the intermediary of a keyboard and screen; many of their everyday objects and appliances can take instructions from that network with minimal human intervention.

In the most general terms, the Internet of Things includes any object – or “thing” – that can be connected to an Internet network, from factory equipment and cars to mobile devices and smart watches. But today, the IoT has more specifically come to mean connected things that are equipped with sensors, software, and other technologies that allow them to transmit and receive data – to and from other things. Traditionally, connectivity was achieved mainly via Wi-Fi, whereas today 5G and other types of network platforms are increasingly able to handle large datasets with speed and reliability.

Of course, the whole purpose of gathering data is not merely to have it but to use it. Once IoT devices collect and transmit data, the ultimate point is to analyse it and create an informed action.

**1.2 History of Internet of Things team work**

The term Internet of Things is 16 years old. But the actual idea of connected devices had been around longer, at least since the 70s. Back then, the idea was often called “embedded internet”.

#### 1.3 Importance of Internet of Things

The importance of IoT is its influence. The Industrial IoT goes beyond the simple exchange of data by connecting devices, systems and people in intelligent, real-world applications to enable environments such as automated factories, smart cities and connected healthcare.

The Industrial IoT is defined as the automated interaction between smart devices and systems, exchanging data to an offsite or cloud-based solution for meaningful, time-sensitive analytics using components with very low energy consumption, that are simple to install and based on industry standards. Traditional embedded systems are at the heart of this phenomenon, having evolved from standalone systems to become a network of connected devices and systems.

IoT is regarded as the significant frontier that can improve almost all activities in our lives. Most of the devices, which have not previously been connected to the internet, can be networked and respond the same way as smart devices. By 2020, the world is set to be completely IoT oriented.

Here are the benefits, which come with this technology.

* IoT promotes efficient resource utilization.
* It minimizes human efforts in many life aspects.
* Enabling IoT will reduce the cost of production and maximizing the returns
* It makes analytics decisions faster and accurately
* It boosts the real-time marketing of products
* Provide a better client experience
* It guarantees high-quality data and secured processing

Considering the complex ecosystem of IoT, there is a need of underlining the competitive advantage of IoT and the stakeholders enabling the users to continue being in complete control of safely sharing their data and depend as much as they can their content.

#### 1.4 Embedded System and History

An embedded system combines mechanical, electrical, and chemical components along with a computer, hidden inside, to perform a single dedicated purpose. There are more computers on this planet than there are people, and most of these computers are single-chip microcontrollers that are the brains of an embedded system. Embedded systems are a ubiquitous component of our everyday lives. We interact with hundreds of tiny computers every day that are embedded into our houses, our cars, our bridges, our toys, and our work. As our world has become more complex, so have the capabilities of the microcontrollers embedded into our devices. Therefore, the world needs a trained workforce to develop and manage products based on embedded microcontrollers.

As its name suggests, Embedded means something that is attached to another thing. An embedded system can be thought of as a computer hardware system having software embedded in it. An embedded system can be an independent system or it can be a part of a large system. An embedded system is a microcontroller or microprocessor-based system which is designed to perform a specific task. For example, a fire alarm is an embedded system; it will sense only smoke.

An embedded system has three components −

* It has hardware.
* It has application software.
* It has Real Time Operating system (RTOS) that supervises the application software and provide mechanism to let the processor run a process as per scheduling by following a plan to control the latencies. RTOS defines the way the system works. It sets the rules during the execution of application program. A small-scale embedded system may not have RTOS.

An **embedded system** is a computer system—a combination of a [computer processor,](https://en.wikipedia.org/wiki/Computer_processor) [computer memory,](https://en.wikipedia.org/wiki/Computer_memory) and [input/output](https://en.wikipedia.org/wiki/Input/output) peripheral devices—that has a dedicated function within a larger mechanical or [electronic](https://en.wikipedia.org/wiki/Electronics) system. It is embedded as part of a complete device often including electrical or electronic hardware and mechanical parts. Because an embedded system typically controls physical operations of the machine that it is embedded within, it often has [real-time computing](https://en.wikipedia.org/wiki/Real-time_computing) constraints. Embedded systems control many devices in common use today.

In 2009 it was estimated that ninety-eight percent of all microprocessors manufactured were used in embedded systems.

#### 1.5 Team Work

##### 1.5.1 Vicky Kumar (18ERECS080)

• **Coding of MCU (Sensor):**

First of all, we have defined two variables; one for the soil moisture sensor pin and the other for storing the output of the sensor.

In the setup function, the “Serial. Begin (9600)” command will help in communication between the MCU and serial monitor. Then, we will print the “Reading from the Sensor …” on the serial monitor.

In the loop function, we will read from the sensor analog pin and will store the values in the

“Output\_ value” variable. Then, we will map the output values to 0-100, because the moisture is measured in percentage. When we took the readings from the dry soil, then the sensor value was 550 and in the wet soil, the sensor value was 10. So, we mapped these values to get the moisture. After that, we printed these values on the serial monitor.

##### • Controlling of Real-Time Graph

Adafruit IO is used in consideration to cut down cost of additional LCD display, as well as it is convenient to control and monitor process anywhere anytime.

It works as follows:

Connect to your Wi-Fi network.

Connect to Adafruit via Wi-Fi.

Read data from Sensor.

Upload Sensor Data to Adafruit.

And then it is represented via a real-time graph.

• **Debugging of Soil Moisture Sensor Code:**

Measurements are made in air

Factory calibration and your soil type

VWC, RWC, PAW, RAW and ε

Temperature can affect your measurements

Salinity and electrical conductivity

##### • Circuit Designing

In this project, a simple garden monitoring system based on [Internet of Things](https://www.engineersgarage.com/Articles/Internet-of-Things-IoT-basics) is designed. An IoT device built on ESP8266 and equipped with [sensors](https://www.engineersgarage.com/articles/sensors) like DHT-11 Temperature and Humidity sensor, [Moisture sensor](https://www.engineersgarage.com/electronic-circuits/soil-moisture-detector) The [**ESP8266**](https://www.engineersgarage.com/tutorials/esp8266-wifi-hotspot) Wi-Fi modem is interfaced The device is also interfaced with a water pump controlled by [L293D](https://www.engineersgarage.com/electronic-components/l293d-motor-driver-ic) motor driver IC. The water pump is automatically controlled based on the values of the various environmental factors. The various environmental factors like temperature, humidity, soil moisture and light intensity can be monitored on the Adafruitdashboard.io dashboard. to connect with a Wi-Fi hotspot.

##### • Considering Protocols

Home automation technology witnessed a huge leap forward.

Products like Amazon Echo and Google Home were finally able to break into the mainstream consumer space and start an industry trend.

Consequently, it is expected to continue this trend with technologies like facial recognition, voice commands and biometrics finding wider application in smart home solutions.

Leading tech companies are investing huge amounts in R&D to create their own connected home and office ecosystems.

The ultimate vision is to create completely automated systems which can recognize and respond to select individuals or user groups, within a home or office environment.A prototype has been implemented to show the real illustration of the proposed approach. An android mobile application has been developed to display the real-time profiles of environmental factors like soil, moisture, and humidity.

With the help of this system, users will be able to treat their gardens in a better way in terms of plant health and growth.

This report work replaces the need for gardeners and issues faced during the maintenance of gardens in big cities.

The purpose of this research is to introduce and prosper the IoT innovation towards smart cities in our society.

MQTT stands for Message Queuing Telemetry Transport. MQTT is a machine-to-machine internet of things connectivity protocol. It is an extremely lightweight and publish-subscribe messaging transport protocol. This protocol is useful for the connection with the remote location where the bandwidth is a premium. These characteristics make it useful in various situations, including constant environment such as for communication machine to machine and internet of things contexts. It is a publish and subscribe system where we can publish and receive the messages as a client. It makes it easy for communication between multiple devices. It is a simple messaging protocol designed for the constrained devices and with low bandwidth, so it's a perfect solution for the internet of things applications.

##### • Testing the MCU (Sensor) Code

Code is tested with random values and figures in order to check whether any error occurs or not.

**1.5.2 Ankit Malpani (18ERECS009)**

##### • Circuit & Designing

##### Upload the Arduino code using the cable and connect the USB to a power bank or a port for power. Connected the USB in the diagram also in a port or power bank for power. No need to supply separate power to the bolt module.

##### After uploading the code check if all devices working properly including LCD, Keypad, LDR, temperature sensor, and LED array. Don't connect heavy load in place of the LED array as it may damage the transistor. Use a MOSFET or Motor driver circuit instead. Check if serial Begin command is working properly by clicking on serial monitor tab in Arduino interface.

Next check the bolt connections, relay, micro-USB, motor and SPST switch. Use SPST switch instead of a sensor.

##### • Internet Connectivity

##### The [concept of Home Automation](https://smartify.in/knowledgebase/what-is-home-automation/) aims to bring the control of operating your everyday home electrical appliances to the tip of your finger, thus giving user affordable lighting solutions, better energy conservation with optimum use of energy. Apart from just lighting solutions, the concept also further extends to have an overall control over your home security as well as build a centralised home entertainment system and much more. The Internet of Things (or commonly referred to as IoT) based Home Automation system, as the name suggests aims to control all the devices of your smart home through internet protocols or cloud based computing.

The IoT based Home Automation system offer a lot of flexibility over the wired systems s it comes with various advantages like ease-of-use, ease-of-installation, avoid complexity of running through wires or loose electrical connections, easy fault detection and triggering and above and all it even offers easy mobility.

##### ESP Connection to Sensor

Connect the two pins of the moisture sensor to the two pins on the Amplifier circuit using jumper wires. Connect the Vcc from the Amplifier to the 3.3V pin on the NodeMCU. Connect the GND pin to the ground (GND) pin on the NodeMCU. Connect the Analog pin to the A0 pin on the NodeMCU.

##### • Stepping Down Voltage by Buck Converter

Buck converter is a step-down power supply module. It is used to bring down the power supply to the desired voltage in order to safe guard the circuit.

Voltage is stepped down to 5v.

##### • Relay Connection

Connect **NodeMCU** Ground (**GND**) pin to **-ve** pin of **Relay**.

Connect **NodeMCU** Supply (**3v3**) pin to **+ve** pin of **Relay**.

Connect **NodeMCU** Digital pin (**D4**) to **Input** pin of **Relay**.

##### • Hardware Debugging

Connecting the ESP8266 Board to the computer via USB connector.

With the use of the tester, we check the regulated power supply.

Confirming that the supply is less than or equal to the permissible limit.

* **Extra:**

We can also run a test code to check the proper functionality of the board.

* **Note:**

Do check that the LEDs on the board are functioning properly.

##### • AC Supply

For Home appliances 220v AC Supply is connected. It is directly connected to the relay circuit.

**1.5.3 Anil Puri Goswami (18ERECS008)**

##### • Circuit Designing

The most basic and crucial requirement in a home automation system, the interface is the basic communication protocol and hardware combination used for sending and receiving messages between devices and the user. Designers have many options for executing communication between devices, the user, and the overall system, depending upon the system, range, size of house, ease of use, etc. If a user wants to control the home appliances through the Internet, the designer needs to add an Ethernet/Wi-Fi interface to connect the system to the home network. If the user wants to control the system using Bluetooth from a cell phone, the designer needs to add a Bluetooth interface to communicate with the device.

The designer needs to determine the sensing requirements of the user and decide upon the required sensor to perform the task. He or she also needs to assess the sensor specifications required for different needs and usability in different environments.

##### • Coding ESP

The ESP8266 is a low-cost Wi-Fi module built by Espressif Systems. Its popularity has been growing among the hardware community thanks to it’s nice features and stability, to the point that it can be easily programmed using your Arduino IDE.

**Requirements:**

An ESP8266 module.

Arduino UNO, UARTbee or any UART to the USB device.

This module has powerful onboard processing and storage capability that allows it to be integrated with the sensors and other application through its GPIOs.

**1.5.4 Vritika Jangir (18ERECS089)**

**Controlling relay:**

Relays come in several form factors. Basically, a relay will be a rectangular block with at least 4 electrical connectors exposed.

When you activate the relay by applying an appropriate voltage to the control circuit, the electromagnetic coil in the relay activates and pulls the switch open, stopping any current flowing through the switched circuit.

##### • Debugging of ESP Code

The error messages are displayed in predefined codes. Unless you know the code, you can never understand what it says. ESP8266 error messages can be categorized into 3 types – 1) Reset causes, 2) Boot modes, and 3) Fatal Exceptions.

Though some can argue that boot mode is not exactly an “error message”, but sometimes it really helps to diagnose a runtime error.

**Note 1:** Just enabling Debug port and setting Debug level won’t work, You need to upload you code after enabling these settings.

**Note 2:** Debug port works at 115200 BAUD if you have not set the serial.

##### • Architecture

Home automation refers to the ability of your home to make its own decisions depending on environment conditions and give you the option to control it from a remote location. The heart project is the Wi-Fi enabled board that needs no introduction; the ESP8266 based [**NodeMCU** d](https://www.electronics-lab.com/project/getting-started-with-the-nodemcu-esp8266-based-development-board/)evelopment board. It is an open-source platform for developing Wi-Fi based embedded systems and it is based on the popular ESP8266 Wi-Fi Module.

**NodeMCU** was born out of the desire to overcome the limitations associated with the first versions of the ESP8266 module which was not compatible with breadboards, it was difficult to power and even more difficult to program. The NodeMCU board is easy to use, low cost and that quickly endeared it to the heart of makers and it is one of the most popular boards today.

We will add a 2-channel relay module to the ESP8266 board. The project flow involves the control of NodeMCU’s GPIOs from a webpage on any device connected on the same network as the board. The status of the GPIOs control the coils of the relays and that causes the relay to alternate between normally open (NO) and normally closed (NC) condition depending on the state of the GPIO, thus, effectively turning the connected appliance “ON” or “OFF”.

##### • Including Libraries for MCU

**ESP8266WiFi library:**

The Wi-Fi library for ESP8266 has been developed based on ESP8266 SDK, using the naming conventions and overall functionality philosophy of the Arduino WiFi library. Over time, the wealth of Wi-Fi features ported from ESP8266 SDK to esp8266 / Arduino outgrew Arduino WiFi library and it became apparent that we would need to provide separate documentation on what is new and extra.

This documentation will walk you through several classes, methods and properties of the ESP8266WiFi library. If you are new to C++ and Arduino, don’t worry. We will start from general concepts and then move to detailed description of members of each particular class including usage examples.

##### • Testing the MCU Code

Connecting the ESP8266 Board to the computer via USB connector.

With the use of the tester, we check the regulated power supply.

Confirming that the supply is less than or equal to the permissible limit.

* **Extra:**

We can also run a test code to check the proper functionality of the board.

* **Note :**

Do check that the LEDs on the board are functioning properly.

* **AC Supply**

For Home appliances 220v AC Supply is connected. It is directly connected to the relay circuit.

##### • Controlling of Appliance

The controller is the device that receives the sensors' signals, processes them and makes computations on them, and then sends instruction signals to the actuators

**Dashboard:** The user can display data from a temperature-humidity sensor alongside data from an air quality sensor and add a button to turn on the appliances in the house.

**Triggers:** Use triggers in Adafruit IO to control and react to your data. Configure triggers to email you when your system goes offline, react to a temperature sensor getting too hot, and publish a message to a new feed.

Connect projects to web services like Twitter, RSS feeds, weather services, etc.

Connect your project to other internet-enabled devices

The best part? All of the above is do-able for freewith Adafruit IO

**Chapter 2**

**Stepping into the world of IoT**

#### 2.1 Definition

The Internet of Things, or IoT, refers to the billions of physical devices around the world that are now connected to the internet, all collecting and sharing data. Thanks to the arrival of super-cheap computer chips and the ubiquity of wireless networks, it's possible to turn anything, from something as small as [a pill t](https://www.zdnet.com/article/how-sensors-enabled-eli-lilly-to-improve-the-patient-experience/)o something as big as [an aeroplane,](https://www.zdnet.com/article/ten-examples-of-iot-and-big-data-working-well-together/) into a part of the IoT. Connecting up all these different objects and adding sensors to them adds a level of digital intelligence to devices that would be otherwise dumb, enabling them to communicate real-time data without involving a human being. The Internet of Things is making the fabric of the world around us smarter and more responsive, merging the digital and physical universes.

**2.2 How big is the Internet of Things?**

Big and getting bigger -- there are already more connected things than people in the world.

Tech analyst company IDC predicts that in total there will be 41.6 billion connected IoT devices by 2025, or "things." It also suggests industrial and automotive equipment representthe largest opportunity of connected "things,", but it also sees strong adoption of smart home and wearable devices in the near term.

Another tech analyst, Gartner, predicts that the enterprise and automotive sectors will account for 5.8 billion devices this year, up almost a quarter on 2019. Utilities will be the

analysis and integration of analytics is expected to propel the utilization of the Internet of Things market over the forecast period.

Industry 4.0 and IoT are at the center of new technological approaches for development, production, and management of the entire logistics chain, otherwise known as smart factory automation. Massive shifts in manufacturing due to industry 4.0 and acceptance of IoT require enterprises to adopt agile, smarter, and innovative ways to advance production with technologies that complement and augment human labor with robotics and reduce industrial accidents caused by a process failure.

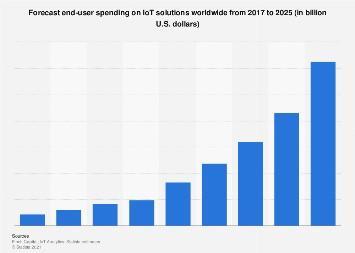


Fig.2.1 IOT endpoint Market by segment 2018-2020

#### 2.3 Building blocks of IOT

Four things form basic building blocks of the IoT system –sensors, processors, gateways, applications. Each of these nodes has to have its own characteristics in order to form an useful IoT system.

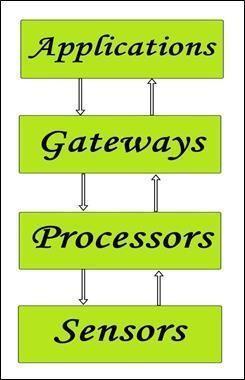


Fig.2.2: Simplified block diagram of the basic building blocks of the IoT

**2.3.1 Sensors:**

* These form the front end of the IoT devices. These are the so-called “Things” of the system. Their main purpose is to collect data from its surroundings (sensors) or give out data to its surrounding (actuators).

* These have to be uniquely identifiable devices with a unique IP address so that they can be easily identifiable over a large network.

* These have to be active in nature which means that they should be able to collect real- time data. These can either work on their own (autonomous in nature) or can be made to work by the user depending on their needs (user-controlled).

* Examples of sensors are gas sensor, water quality sensor, moisture sensor, etc.

**2.3.2 Processors:**

* Processors are the brain of the IoT system. Their main function is to process the data captured by the sensors and process them so as to extract the valuable data from the enormous amount of raw data collected. In a word, we can say that it gives intelligence to the data.
* Embedded hardware devices, microcontroller, etc are the ones that process the data because they have processors attached to it.

**2.3.3 Gateways:**

* Gateways are responsible for routing the processed data and send it to proper locations for its (data) proper utilization.

* In other words, we can say that gateway helps in to and fro communication of the data. It provides network connectivity to the data. Network connectivity is essential for any IoT system to communicate.

* LAN, WAN, PAN, etc are examples of network gateways.

**2.3.4 Applications:**

* Applications form another end of an IoT system. Applications are essential for proper utilization of all the data collected.

* These cloud-based applications which are responsible for rendering the effective meaning to the data collected. Applications are controlled by users and are a delivery point of particular services.

* Examples of applications are home automation apps, security systems, industrial control hub, etc.

#### 2.4 Architecture of IOT

The Internet of things (IoT) describes the network of physical objects—a.k.a. "things"— that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the Internet.

controlled via devices associated with that ecosystem, such as smartphones and smart speakers. The IoT can also be used in healthcare systems.

There are a number of serious concerns about dangers in the growth of the IoT, especially in the areas of privacy and security, and consequently industry and governmental moves to address these concerns have begun including the development of international standards.

So, from the above image it is clear that there is 4 layers are present that can be divided as follows: Sensing Layer, Network Layer, Data processing Layer, and Application Layer. These are explained as following below.

* **Sensing layer:** Sensors, actuators, devices are present in this Sensing layer. These Sensors or Actuators accepts data(physical/environmental parameters), processes data and emits data over network.

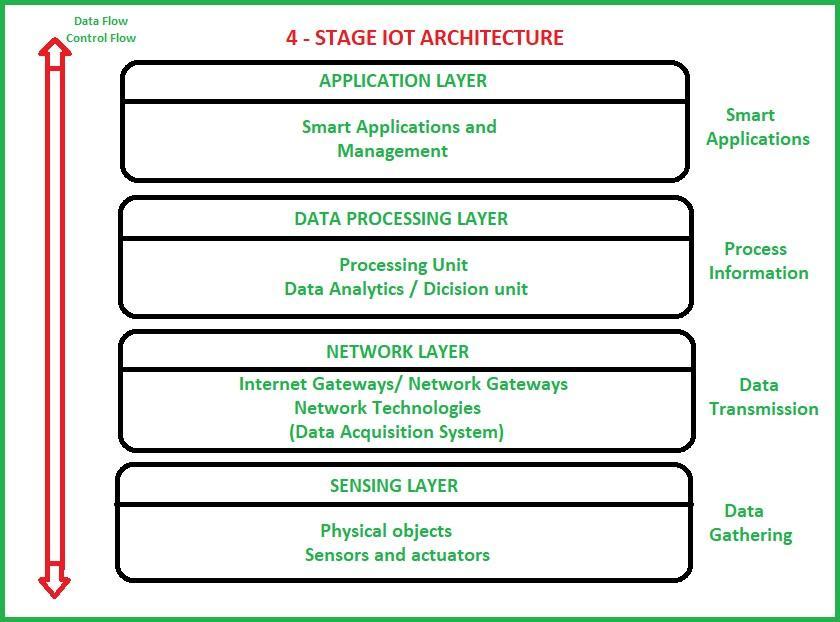


Fig.2.3: IOT Architecture

* **Network Layer:** Internet/Network gateways, Data Acquisition System (DAS) are present in this layer. DAS performs data aggregation and conversion function (Collecting data and aggregating data then converting analog data of sensors to digital data etc). Advanced gateways which mainly opens up connection between Sensor networks and Internet also performs many basic gateway functionalities like malware protection, and filtering also some times decision making based on inputted data and data management services, etc.

**Data processing Layer:** This is processing unit of IoT ecosystem. Here data is analyzed and pre-processed before sending it to data center from where data is accessed by software applications often termed as business applications where data is monitored and managed and further actions are also prepared. So here Edge IT or edge analytics comes into picture.

* **Application Layer:** This is last layer of 4 stages of IoT architecture. Data centers or cloud is management stage of data where data is managed and is used by end-user applications like agriculture, health care, aerospace, farming, defense, etc.

#### 2.5 Working of IOT

When talking about how IoT works, the process begins with devices that have built-in sensors. These devices are connected to IoT platforms which stores data from all the connected devices. The important data is then used to perform tasks that fulfil the needs of people.

In this way, IoT application works with smart systems that automate tasks to address specific needs.

However, if you still have any doubts regarding how does it work, check out this video:-

With this attended to, let’s dive deeper into IoT market and see what are the prime components of Internet of Things technology.

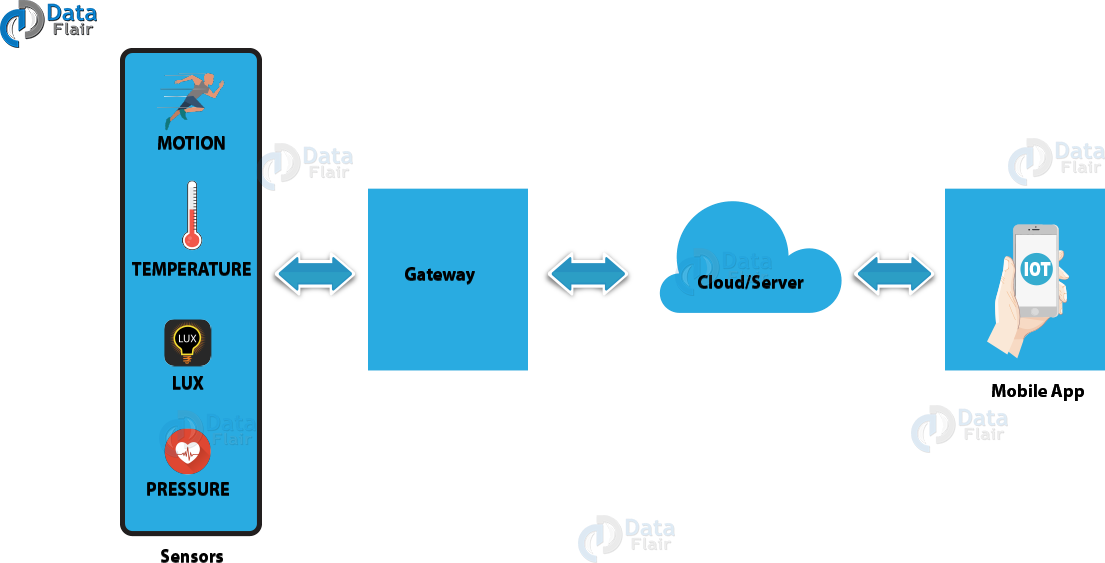


Fig.2.4: Working of IOT

#### 2.6 Characteristics of IOT

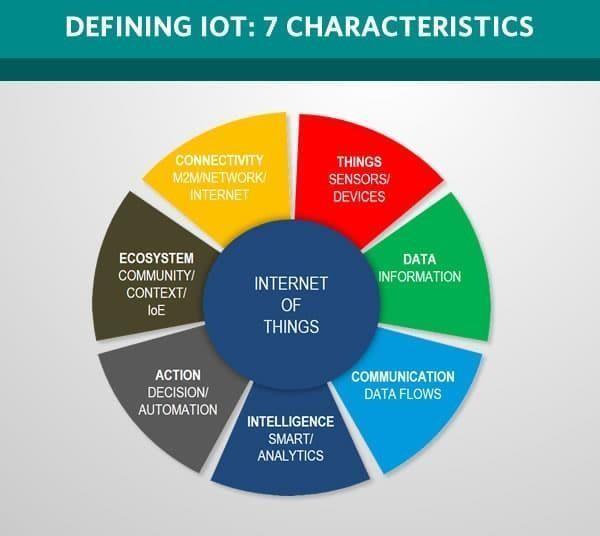


Fig.2.5: Characteristics of IOT

* **Connectivity.** This doesn’t need too much further explanation. With everything going on in IoT devices and hardware, with sensors and other electronics and connected hardware and control systems there needs to be a connection between various levels.

* **Things**. Anything that can be tagged or connected as such as it’s designed to be connected. From sensors and household appliances to tagged livestock. Devices can contain sensors or sensing materials can be attached to devices and items.

* **Data**. Data is the glue of the Internet of Things, the first step towards action and intelligence.

* **Communication**. Devices get connected so they can communicate data and this data can be analyzed. Communication can occur over short distances or over a long range to very long range. Examples: Wi-Fi, [LPWA n](https://www.i-scoop.eu/internet-of-things-guide/lpwan/)etwork technologies such as [LoRa o](https://www.i-scoop.eu/internet-of-things-guide/iot-network-lora-lorawan/)r [NB-IoT.](https://www.i-scoop.eu/internet-of-things-guide/lpwan/nb-iot-narrowband-iot/)

**Intelligence**. The aspect of intelligence as in the sensing capabilities in IoT devices and the intelligence gathered from big data analytics (also artificial intelligence).

* **Action**. The consequence of intelligence. This can be manual action, action based upon debates regarding phenomena (for instance in [smart factory d](https://www.i-scoop.eu/industry-4-0/smart-factory-scaling/)ecisions) and automation, often the most important piece.

* **Ecosystem**. The place of the Internet of Things from a perspective of other technologies, communities, goals and the picture in which the Internet of Things fits. The Internet of Everything dimension, the platform dimension and the need for solid partnerships.

#### .2.7 Security concerns of IOT

The internet of things connects billions of devices to the internet and involves the use of billions of data points, all of which need to be secured. Due to its expanded attack surface, [IoT security a](https://internetofthingsagenda.techtarget.com/definition/IoT-security-Internet-of-Things-security)nd [IoT privacy a](https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-privacy-IoT-privacy)re cited as major concerns.

In 2016, one of the most notorious recent IoT attacks was Mirai, a [botnet t](https://searchsecurity.techtarget.com/definition/botnet)hat infiltrated domain name server provider Dyn and took down many websites for an extended period of time in one of the biggest distributed denial-of-service ([DDoS)](https://searchsecurity.techtarget.com/definition/distributed-denial-of-service-attack) attacks ever seen. Attackers gained access to the network by exploiting poorly secured IoT devices.

Beyond leaking personal data, [IoT poses a risk t](https://internetofthingsagenda.techtarget.com/tip/Internet-of-Things-IOT-Seven-enterprise-risks-to-consider)o critical infrastructure, including electricity, transportation and financial services.

#### 2.8 Government regulations on IOT

One of the key drivers of the IoT is data. The success of the idea of connecting devices to make them more efficient is dependent upon access to and storage & processing of data. For Current regulatory environment.

A report published by the [Federal Trade Commission (](https://en.wikipedia.org/wiki/Federal_Trade_Commission)FTC) in January 2015 made the following three recommendations:

[**Data security** –](https://en.wikipedia.org/wiki/Data_security) At the time of designing IoT companies should ensure that data collection, storage and processing would be secure at all times. Companies should adopt a "defense in depth" approach and encrypt data at each stage.

* **Data consent** – users should have a choice as to what data they share with IoT companies and the users must be informed if their data gets exposed.

* **Data minimisation** – IoT companies should collect only the data they need and retain the collected information only for a limited time.

### Chapter 3

**Arduino**

#### 3.1 Introduction

Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices. Its hardware products are licensed under a CC-BY-SA license, while software is licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially from the official website or through authorized distributors.

By 2017 Arduino AG owned many Arduino trademarks. In July 2017 BCMI, founded by Massimo Banzi, David Cuartielles, David Mellis and Tom Igoe, acquired Arduino AG and all the Arduino trademarks. Fabio Violante is the new CEO replacing Federico Musto, who no longer works for Arduino AG.

#### 3.2 Hardware

Arduino is open-source hardware. The hardware reference designs are distributed under a Creative Commons Attribution Share-Alike 2.5 license and are available on the Arduino website. Layout and production files for some versions of the hardware are also available.

Although the hardware and software designs are freely available under copyleft licenses, the developers have requested the name Arduino to be exclusive to the official product and not be used for derived works without permission. The official policy document on use of the Arduino name emphasizes that the project is open to incorporating work by others into the official product. Several Arduino-compatible products commercially released have avoided the project name by using various names ending in -duino.

Most Arduino boards consist of an Atmel 8-bit AVR microcontroller (ATmega8, ATmega168, ATmega328, ATmega1280, or ATmega2560) with varying amounts of flash memory, pins, and features. The 32-bit Arduino Due, based on the Atmel SAM3X8E was introduced in 2012. The boards use single or double-row pins or female headers that facilitate connections for programming and incorporation into other circuits. These may connect with add-on modules termed shields. Multiple and possibly stacked shields may be individually addressable via an I²C serial bus. Most boards include a 5 V linear regulator and a 16 MHz crystal oscillator or ceramic resonator. Some designs, such as the LilyPad, run at 8 MHz and dispense with the onboard voltage regulator due to specific form-factor restrictions.

Arduino microcontrollers are pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory. The default bootloader of the Arduino Uno is the

The Arduino board exposes most of the microcontroller's I/O pins for use by other circuits. The Diecimila, Duemilanove, and current Uno provide 14 digital I/O pins, six of which can produce pulse-width modulated signals, and six analog inputs, which can also be used as six digital I/O pins. These pins are on the top of the board, via female 0.1-inch (2.54 mm) headers. Several plug-in application shields are also commercially available. The Arduino Nano, and Arduino-compatible Bare Bones Board and Boarduino boards may provide male header pins on the underside of the board that can plug into solderless breadboards.

##### 3.3 Official Boards

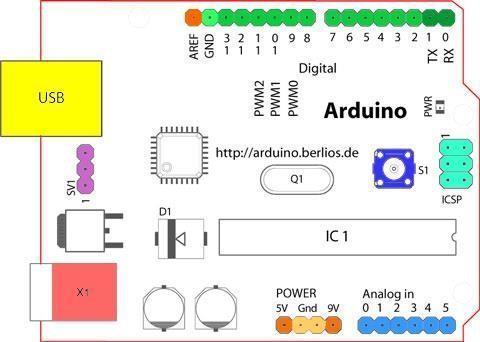


Fig.3.1: Arduino Board Diagram

###### 3.4 Digital Pins

In addition to the specific functions listed below, the digital pins on an Arduino board can be used for general purpose input and output via the [pinMode(),](https://www.arduino.cc/en/Reference/PinMode) [digitalRead(),](https://www.arduino.cc/en/Reference/DigitalRead) and [digitalWrite() c](https://www.arduino.cc/en/Reference/DigitalWrite)ommands. Each pin has an internal pull-up resistor which can be turned on and off using digitalWrite() (w/ a value of HIGH or LOW, respectively) when the pin is configured as an input. The maximum current per pin is 40 mA.

* **Serial:** 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. On the

Arduino Diecimila, these pins are connected to the corresponding pins of the FTDI USB-to- TTL Serial chip. On the Arduino BT, they are connected to the corresponding pins of the WT11 Bluetooth module. On the Arduino Mini and LilyPad Arduino, they are intended for use with an external TTL serial module (e.g. the Mini-USB Adapter).

* **External Interrupts:** 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the [attachInterrupt() f](https://www.arduino.cc/en/Reference/AttachInterrupt)unction for details.

* **PWM:** 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the [analogWrite() f](https://www.arduino.cc/en/Reference/AnalogWrite)unction. On boards with an ATmega8, PWM output is available only on pins 9, 10, and 11.

* **BT Reset:** 7. (Arduino BT-only) Connected to the reset line of the bluetooth module.

* **SPI:** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.

* **LED:** 13. On the Diecimila and LilyPad, there is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

###### 3.5 Analog Pins

In addition to the specific functions listed below, the analog input pins support 10-bit analog- to-digital conversion (ADC) using the [analogRead() f](https://www.arduino.cc/en/Reference/AnalogRead)unction. Most of the analog inputs can also be used as digital pins: analog input 0 as digital pin 14 through analog input 5 as digital pin 19. Analog inputs 6 and 7 (present on the Mini and BT) cannot be used as digital pins.

###### 3.6 Power Pins

* VIN (sometimes labelled "9V"). The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin. Note that different boards accept different input

* 5V. The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.

* 3V3. (Diecimila-only) A 3.3 volt supply generated by the on-board FTDI chip.

* GND. Ground pins.

with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one- click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus. The source code for the IDE is released under the GNU General Public License, version 2.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

##### 3.7 Pro IDE

On October 18, 2019, Arduino Pro IDE (alpha preview) was released. The system still uses Arduino CLI (Command Line Interface), but improvements include a more professional development environment, autocompletion support, and Git integration. The application frontend is based on the Eclipse Theia Open Source IDE. The main features available in the alpha release are:

##### 3.8 Sketch

A sketch is a program written with the Arduino IDE. Sketches are saved on the development computer as text files with the file extension .ino. Arduino Software (IDE) pre-1.0 saved sketches with the extension .pde.

A minimal Arduino C/C++ program consists of only two functions:

**Setup() :**This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch. It is analogous to the function main() .

**Loop():** After setup() function is executed repeatedly in loop(). function exits (

Power LED (red) and User LED (green) attached to pin 13 on an Arduino compatible board.

Most Arduino boards contain a light emitting diode (LED) and a current limiting resistorconnected between pin 13 and ground, which is a convenient feature for many tests and program functions. A typical program used by beginners, akin to Hello, World!, is

"blink", which repeatedly blinks the on-board LED integrated into the Arduino board. This program uses the functions , which are provided by the internal libraries included in the IDE environment.

#### 3.9 Operating System / Threading

There is a Xinu OS port for the atmega328p (Arduino Uno and others with the same chip), which includes most of the basic features. The source code of this version is freely available.

There is also a threading tool, named Protothreads. Protothreads are described as "extremely lightweight stackless threads designed for severely memory constrained systems, such as small embedded systems or wireless sensor network nodes. Protothreads provide linear code

execution for event-driven systems implemented in C. Protothreads can be used with or without an underlying operating system."

##### 3.10 Application

* Learning & experimenting

* Prototyping & validation

* Sacrificial concepts (user research)

* temporary installations & demonstrations

### Chapter 4

**Components of IOT**

#### 4.1 Introduction

An embedded system is a computer system—a combination of a computer processor, computer memory, and input/output peripheral devices—that has a dedicated function within a larger mechanical or electronic system. It is embedded as part of a complete device often including electrical or electronic hardware and mechanical parts. Because an embedded system typically controls physical operations of the machine that it is embedded within, it often has real-time computing constraints. Embedded systems control many devices in common use today. In 2009 it was estimated that ninety-eight percent of all microprocessors manufactured were used in embedded systems.

##### 4.2 Sensors/Devices

First, sensors or devices help in collecting very minute data from the surrounding environment. All of this collected data can have various degrees of complexities ranging from a simple temperature monitoring sensor or a complex full video feed.

A device can have multiple sensors that can bundle together to do more than just sense things. For example, our phone is a device that has multiple sensors such as GPS, accelerometer, camera but our phone does not simply sense things.

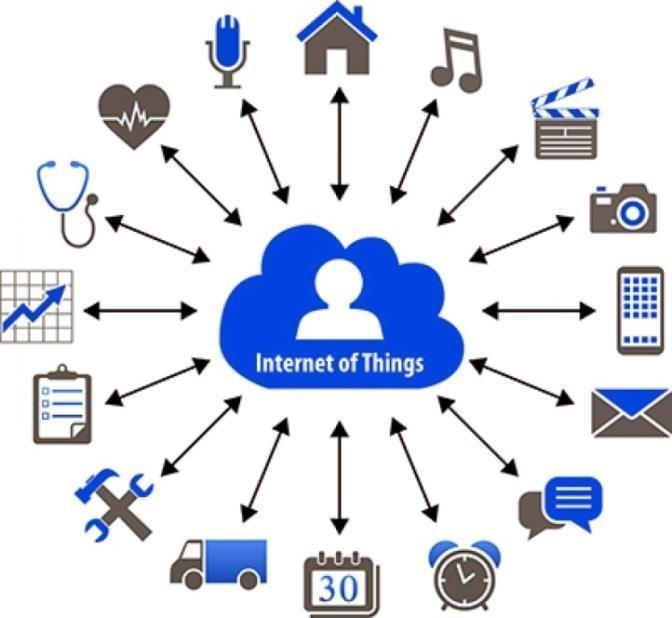


Fig.4.1: Sensors/Devices

##### 4.3 Connectivity

Next, that collected data is sent to a cloud infrastructure but it needs a medium for transport.

The sensors can be connected to the cloud through various mediums of communication and transports such as cellular networks, satellite networks, Wi-Fi, Bluetooth, wide-area networks (WAN), low power wide area network and many more.

Every option we choose has some specifications and trade-offs between power consumption, range, and bandwidth. So, choosing the best connectivity option in the IOT system is important.



Fig.4.2: Data Processing

##### 4.4 User Interface

Next, the information made available to the end-user in some way. This can achieve by triggering alarms on their phones or notifying through texts or emails.

Also, a user sometimes might also have an interface through which they can actively check in on their IOT system. For example, a user has a camera installed in his house, he might want to check the video recordings and all the feeds through a web server.

* Networks: Ethernet, LonWorks, etc.

* Fieldbuses: CAN-Bus, LIN-Bus, PROFIBUS, etc.

* Timers: PLL(s), Capture/Compare and Time Processing Units

* Discrete IO: aka General-Purpose Input/Output (GPIO)

* Analog to Digital/Digital to Analog (ADC/DAC)

* Debugging: JTAG, ISP, BDM Port, BITP, and DB9 ports.

#### 4.5 Tools

As with other software, embedded system designers use compilers, assemblers, and debuggers to develop embedded system software. However, they may also use some more specific tools:

* In circuit debuggers or emulators (see next section).

* Utilities to add a checksum or CRC to a program, so the embedded system can check if the program is valid.

* For systems using digital signal processing, developers may use a math workbench to simulate the mathematics.
* Sometimes, development tools for a personal computer can be used if the embedded processor is a close relative to a common PC processor.

#### 4.6 Process and People

Let’s start defining what a process is: A series of actions or steps taken in order to achieve a particular end. A process can be associated to thing or to a business. In the case of a thing can be the process of a person running. The person will have as a goal to run specific distance in a specific time. Using an IoT device (e.g. fitbit tracker) the person can track in a daily basis distance, time, calories, water consumption, etc. until the goal is accomplished. The person can use all the information gathered by the device to monitor his or her progress during the process and make the necessary adjustment in order to reach the goal. In the past people were relying on manual tracking which was open to human errors.

In the case of a business process, it can be a manufacturing process for example service

preventive maintenance before the original set date and this way save time and money because major maintenance will not be necessary.

### Chapter 5

### THE CONTROLLER

#### 5.1 Controller

* Your IoT device may be smaller than a coin or larger than a refrigerator.
* It may perform a simple sensing function and send raw data back to a control center
* It may combine data from many sensors, perform local data analysis, and then take action. Additionally, your device could be remote and standalone or be co-located within a larger system.
* Regardless of the function, environment, or location, your IoT device requires two components, a brain and connectivity.
* The “brain” provides local control (or decision-making). Your device’s function will determine the size and capabilities of the brain component.
* “Things,” few have evolved their thinking to a point where they are willing to publicly discuss key constructs of IoT architecture.
* The unique challenge of IoT is the high volume of sensory data and the complex challenges this creates in managing and leveraging this data.
* This challenge extends to enterprises of all sizes that are interested in connecting more effectively with customers and driving higher levels of efficiency across the value chain.
* The location of controllers and gateways will also vary based on the intelligence and constraints of the network connection to the front-end devices, environmental factors, and security concerns. Common networking between edge devices and controllers/gateways includes BT, CAN, LAN, WAN, WIFI, USB, and Internet.

##### 5.2 The “Brain” of IOT

* Your IoT device will most likely use a microcontroller as its brain.
* Think of a microcontroller as a small computer with a microprocessor core, memory, and input/output (I/O) ports.
* The microprocessor core of your microcontroller is a central processing unit.
* It handles all the number crunching and local data manipulation and decision-making.
* The memory includes Read Only Memory (ROM) and Random Access Memory (RAM).
* ROM stores the microcontroller’s software program.
* RAM stores and receives data while also supporting number crunching.
* The final microcontroller component, the I/O ports, may be either digital or analog.
* The input ports collect data from sensors. While the outputs support any necessary actuation or local control in the IoT device.

* Usually, microcontrollers control various devices or subsystems within embedded applications.
* By integrating the microprocessor, memory, and input/outputs, microcontrollers reduce cost and make development easier.
* This makes it more affordable and less complicated to control many IoT devices.

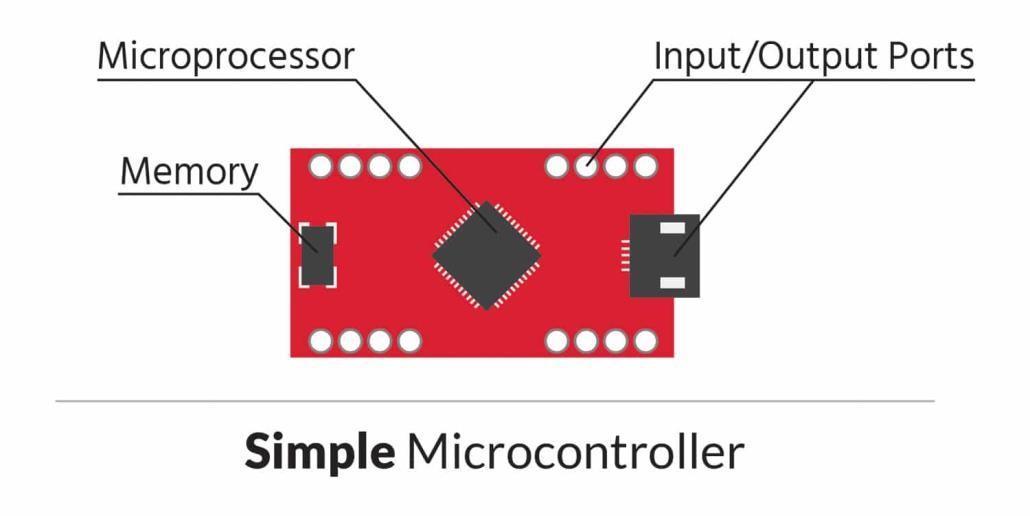


Fig.5.1 Diagram showing Simple Microcontroller

#### 5.3 Adafruit.IO

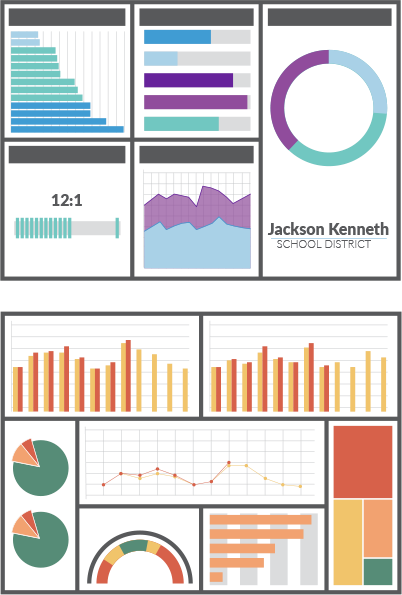
* Adafruit.io is a cloud service- that just means we run it for you and you don't have to manage it.
* We can connect to it over the Internet.
* It's meant primarily for storing and then retrieving data but it can do a lot more than just that.

* Here at Adafruit, we manufacture, support and sell all of these amazing sensors, LEDs, and robotics.
* So, naturally, we wanted a good way to interact with them over the Internet.
* We’ve covered a bunch of great services for data logging and communicating with your controller over the web, but we wanted a service that fit the needs of the Prototyping & Maker communities AND was designed for engineers of all skill levels.
* So we decided to build our own system - that is how Adafruit IO got started.
* Like many of the services we’ve introduced, we maintain both MQTT and REST APIs, which is how you will be communicating with Adafruit IO over the Internet.
* But you don’t have to be an expert programme.
* We’ve built robust client libraries with lots of examples, so you can probably start with some ready-to-go code.
* Once you’ve got a device connected, you can control and monitor using configurable dashboards.
* The web dashboards come with a dozen widgets that allow for easy 2-way interaction with your devices. You’ll get buttons, gauges, maps, sliders.
* Outside of the dashboard, you can create triggers to, say, email you when the water sensor value goes above 9000.
* It is available and free for anyone with an Adafruit account.
* Adafruit IO plus members get some nifty extras and more bandwidth.

#### 5.4 Dashboard

* A dashboard is a tool used for information management and business.

Fig.5.2 Diagram showing Dashboards



Much like the dashboard of a car, data dashboards organize, store, and display important information from multiple data sources into one, easy-to-access place.

* Using data visualization, dashboards uniquely communicate metrics visually to help users understand complex relationships in their data.
* In a data dashboard, it’s easier to draw parallels between different but related metrics, identify trends, and head off potential challenges hidden in an organization’s data.
* With the help of smartphones, tablets, and other mobile technology, dashboards are also used to convey relevant information to audiences at any time and in any place.
* The best dashboards are customized, secured, and shared with their intended end-users.
* A successful KPI dashboard should set tangible goals and targets for each department, facilitate accountability within each department and provide real-time updates on goals and progress.
* Much like KPI dashboards, project dashboards track tangible goals; however, the “goal” of a project dashboard isn’t about hitting a sales quota or increasing marketing revenue by a certain margin.
* Instead, project dashboards track specific metrics related to the progress and complete of a project.

#### 5.5 Steps to design controller

As you know that when we make an IOT project or any embedded project then we have to display the output on a screen. For this, we design an application and it takes a long time so you can prefer to use node-red.

Before creating your own dashboard, do the following:

* **Step 1: Structure your Solution** – to build an IoT application you need to create a structure encompassing: devices, variables, dashboards, and alerts.
* **Step 2: Select a Device** – devices are the individual hardware selected to sense data in a particular environment. It is very important that the device is selected based on the

environment and the requirements you’re looking for it to complete. Also please consider your connection protocol of choice, be it WiFi, Bluetooth, Sigfox, LoRa, Ethernet, etc.

* **Step 3: Sign Up or Log In at the site to build controller-** The first thing you need to do is, in case you have not registered yet, create a new account and Sign Up to our platform. Otherwise, if you already have an account, you just need to Log In to move to your IoT

Dashboard. Primarily you’ll notice that when you first access to your IoT Dashboard, a new window appears to introduce yourself to the platform and specifically, to the IoT Dashboard.

* **Step 4:Starting with IoT Dashboard-** The IoT Dashboard is the place at the site where all the data generated by your things will be displayed. Depending on which kind of information you want to see you may prefer one widget or another. Before start building your IoT Dashboard you should have at least one thing connected using our platform. If you want to start connecting things fast, you should not miss this quick guide of how to connect your first thing in 1 minute. Why? Because you need data to display! Once you have done that, we can move to the option “Edit Dashboard”, on the lower right section of your screen. The icon is yellow and this tool is the one we are going to use most during this post, so keep an eye on it!
* **Step 5: Create a Widget-** “Edit Dashboard” there is a a blue “Add Widget” icon. So now we are going to click on it and a pop-up you will have to work with will appear. At this point, you can choose the kind of widget you want your data to be displayed on your IoT Dashboard. You have different options, such as: Map, Doughnut, Lines, Bars, Logs, Value, Pie and Polar

Area. Depending on the option you choose in “Values Range” you’ll find one or the other. Finally, you can add your new widget to your IoT Dashboard.

* **Step 6: Customize your IoT Dashboard-** One of the coolest things you can do with your IoT Dashboard is to add, remove and replace your widgets easily. To do it you just need to click on “Edit Dashboard” again and just touch one of the widgets. Now you can change its place

just moving it, change its size, and also the kind of widget you want it to be! Once you finish editing your IoT Dashboard do not forget to save it!

* **Step 7: Create an Image-**You may think that’s all we can offer, but right now we do offer more! Now you can add images to your IoT Dashboard. Go to Edit Dashboard button and Add Widget. Once you are in the pop-up window, in the “Data Source” field you have to choose the option “External Image”. Here you will be able to add images from your computer or directly from an URL.



Fig.5.3 Diagram showing Controller Design

#### 5.6 Libraries of Adafruit

Before we get started using Adafruit IO with your Arduino, we'll need to select a library. We provide and support both the Adafruit IO and Adafruit MQTT libraries listed below, but we try starting with the Adafruit IO Arduino Library below: The library supports the following

##### 5.7 Adafruit [ESP8266](https://github.com/adafruit/Adafruit_IO_Arduino#adafruit-feather-huzzah-esp8266--adafruit-huzzah-esp8266-breakout)

Feather is the new development board from Adafruit, and like it's namesake it is thin, light, and lets you fly! We designed Feather to be a new standard for portable microcontroller cores.

This is the Adafruit Feather HUZZAH ESP8266 - our take on an 'all-in-one' ESP8226 WiFi development board with built in USB and battery charging. Its an ESP8266 WiFi module with all the extras you need, ready to rock. At the Feather HUZZAH's heart is an ESP8266 WiFi microcontroller clocked at 80 MHz and at 3.3V logic. This microcontroller contains a Tensilica chip core as well as a full WiFi stack. You can progam the microcontroller using the Arduino IDE for an easy-to-run Internet of Things core. We wired up a USB-Serial chip that can upload code at a blistering 921600 baud for fast development time. It also has auto-reset so no noodling with pins and reset button pressings.

##### 5.8 [Adafruit](https://learn.adafruit.com/adafruit-io-basics-airlift) [ESP32](https://github.com/adafruit/Adafruit_IO_Arduino#adafruit-feather-huzzah32-esp32)

The ESP8266 started a small revolution by bringing WiFi to a small and cheap package that also had enough processing power and enough pins to get small things done. Now get ready to take your bite-sized WiFi capabilities to the next level with the ESP32 Development Board!

The development board breaks out all the module’s pins to 0.1″ headers and provides a CP2102 USB-TTL serial adapter, programming and reset buttons, and a power regulator to supply the ESP32 with a stable 3.3 V. Espress if doubled-down on the CPU resources for the ESP32 with a dual core, running at 160MHz and tons more pins and peripherals.

The ESP32 is still targeted to developers. Not all of the peripherals are fully documented with example code, and there are some bugs still being found and fixed. We got many sensors and displays working under Arduino IDE, so you can expect things like I2C and SPI and analog reads to work. But other elements are still under development. This board is designed for use with the Tensilica tool chain (ESP IDF) so we recommend that for better hardware-support coverage. Color of PCB may vary.

##### 5.9 [Adafruit](https://learn.adafruit.com/adafruit-io-basics-airlift) [Ethernet](https://github.com/adafruit/Adafruit_IO_Arduino#adafruit-ethernet-featherwing)

Wireless is wonderful, but sometimes you want the strong reliability of a wire. If your

Feather board is going to be part of a permanent installation, this Ethernet

##### 5.10 [Adafruit](https://learn.adafruit.com/adafruit-io-basics-airlift) FONA (Cellular)

Introducing Adafruit FONA MiniGSM, an adorable all-in-one cellular phone module that lets you add voice, text, SMS and data to your project in an adorable little package.

This module measures only 1.75"x1.25" but packs a surprising amount of technology into it's little frame. At the heart is a GSM cellular module (we use the latest SIM800) the size of a postage stamp. This module can do just about everything

* Quad-band 850/900/1800/1900MHz - connect onto any global GSM network with any 2G SIM (in the USA, T-Mobile is suggested)
* Make and receive voice calls using a headset OR an external 8Ω speaker + electret microphone
* Send and receive SMS messages
* Send and receive GPRS data (TCP/IP, HTTP, etc.)
* Scan and receive FM radio broadcasts (yeah, we don't exactly know why this was included but it works really well)
* PWM/Buzzer vibrational motor control
* AT command interface with "auto baud" detection

### Chapter 6

### Circuit

#### 6.1 Basics of circuit designing in IOT

IoT networks have become exceedingly popular these days. [IoT (Internet Of Things) i](https://iot4beginners.com/internet-of-things-basic/)n the simplest words means a network of devices connected to each other to send and receive data. Of course some of these networks may be possible with pre-manufactured devices that do not require any manual circuit building. However, for some applications, you might want to fine tune the devices to fit your requirement or build a small device yourself with components and a breadboard. In such cases read this article to understand some rules and factors that you need to consider before you design a circuit for IoT application.

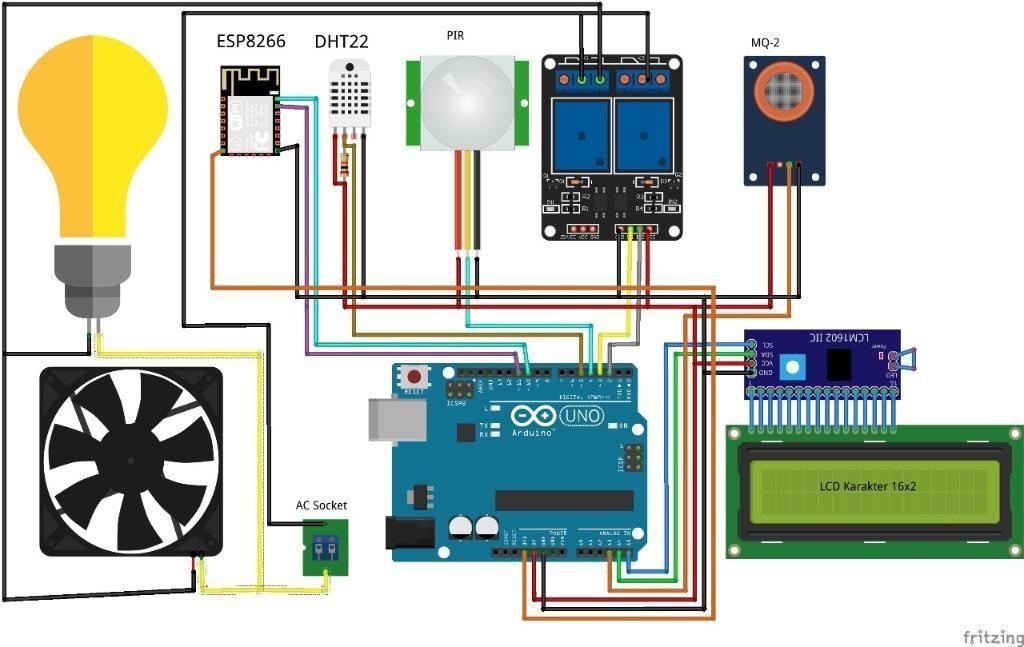


Fig.6.1: Example of Home Automation Circuit Design

#### 6.2 Circuit designing rules

In any circuit design, the basic fundamentals are very important. This knowledge will ensure that you understand the relation between potential and current and are able to pick the correct value of resistance required.

• **Ohms Law:**

As a mathematical expression Ohm’s law states that V = I . R In circuit schematic form

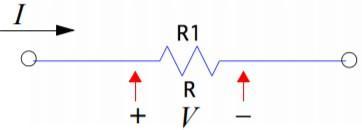


Fig.6.2: Ohms Law

Ohm’s law can be written in three equivalent forms,

V = I . R : The voltage across a resistor is the product of the current passing through the resistor times the resistance

a junction.

V = V1 +V2+V3 = IR1+IR2+IR3

So, to calculate the total resistance in series,



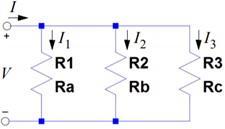


Fig.6.3 : Parallel Resistance

I = I1 + I2 + I3

So, to calculate the total resistance in parallel,



#### 6.3 Introduction to IOT connectivity

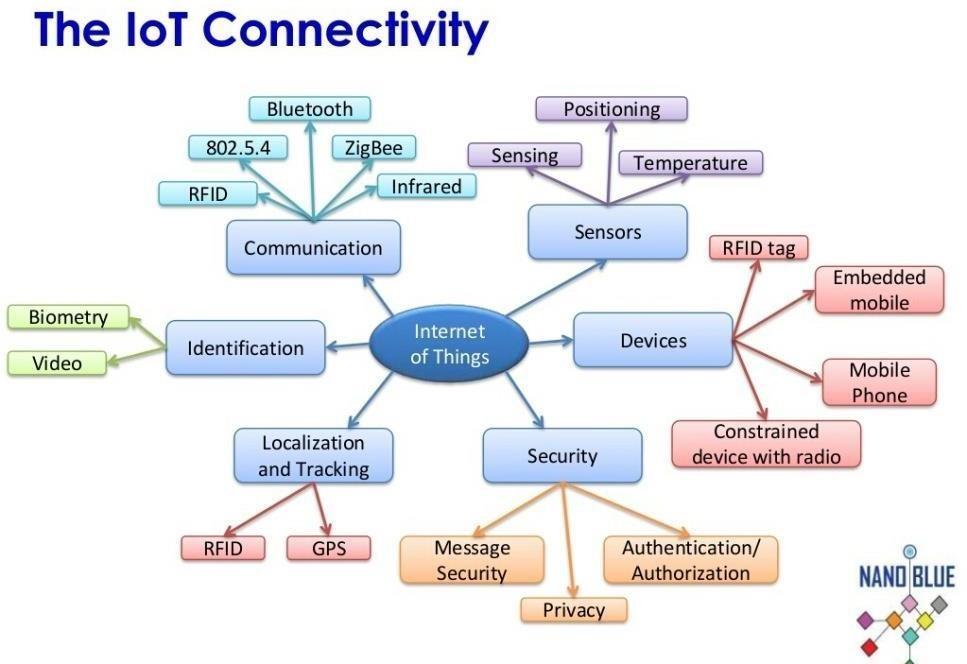


Fig.6.4: IOT Connectivity

When it comes to connecting the Internet of Things, there are a seemingly overwhelming number of options. Cellular, satellite, WiFi, Bluetooth, RFID, NFC, LPWAN, and Ethernet are just some of the possible ways to connect a sensor/device. And within each of these options there can be different providers (e.g. for cellular there’s T-Mobile, Verizon, AT&T, Sprint, etc.).

#### 6.4 Trade Off Between Power Consumption, Range, and Bandwidth

The perfect connectivity option would consume extremely little power, have huge range, and would be able to transmit large amounts of data (high bandwidth). Unfortunately, this perfect connectivity doesn’t exist.

##### 6.5 High Power Consumption, High Range, High Bandwidth

To wirelessly send a lot of data over a great distance, it takes a lot of power. A great example of this is your Smartphone. Your phone can receive and transmit large amounts of data (e.g. video) over great distances, but you need to charge it every 1–2 days. Connectivity options in this group include cellular and satellite.

##### 6.6 Low Power Consumption, Low Range, High Bandwidth

To decrease power consumption and still send a lot of data, you have to decrease the range. Connectivity options in this group include Wi-Fi, Bluetooth, and Ethernet.

Ethernet is a hard-wired connection, so the range is short because it’s only as far as the wire length. WiFi and Bluetooth are both wireless connections with high bandwidth and lower power consumption than cellular and satellite

##### 6.7 Low Power Consumption, High Range, Low Bandwidth

To increase range while maintaining low power consumption, you have to decrease the amount of data that you’re sending. Connectivity options in this group are called Low-Power Wide-Area Networks (LPWANs).

### Chapter 7

### Home Automation

#### 7.1 Introduction

A home automation system is a technological solution that enables automating the bulk of electronic, electrical and technology-based tasks within a home.

It uses a combination of hardware and software technologies that enable control and management over appliances and devices within a home.

Home automation is also known as domotics, and a home with an automation system is also known as a smart home.

Home automation or domotics is building automation for a home, called a smart home or smart house. A home automation system will monitor and/or control home attributes such as lighting, climate, entertainment systems, and appliances. It may also include home security such as access control and alarm systems. When connected with the Internet, home devices are an important constituent of the Internet of Things ("IoT").

A home automation system typically connects controlled devices to a central hub or "gateway". The user interface for control of the system uses either wall-mounted terminals, tablet or desktop computers, a mobile phone application, or a Web interface that may also be accessible off-site through the Internet.

While there are many competing vendors, there are increasing efforts towards open source systems. However, there are issues with the current state of home automation including a lack of standardized security measures and deprecation of older devices without backwards compatibility.

Home automation has high potential for sharing data between family members or trusted individuals for personal security and could lead to energy saving measures with a positive environmental impact in the future.

The home automation market was worth US$5.77 billion in 2013, predicted to reach a market value of US$12.81 billion by the year 2020.



Fig.7.1: Smart Home Automation System

#### 7.2 History

Early home automation began with labour-saving machines. Self-contained electric or gas powered home appliances became viable in the 1900s with the introduction of electric power distribution and led to the introduction of washing machines (1904), water heaters (1889), refrigerators (1913), sewing machines, dishwashers, and clothes dryers.

In 1975, the first general purpose home automation network technology, X10, was developed. It is a communication protocol for electronic devices. It primarily uses electric power transmission wiring for signalling and control, where the signals involve brief radio frequency bursts of digital data, and remains the most widely available. By 1978, X10 products included a 16 channel command console, a lamp module, and an appliance module. Soon after came the wall switch module and the first X10 timer.

By 2012, in the United States, according to ABI Research, 1.5 million home automation systems were installed. Per research firm Statists more than 45 million smart home devices will be installed in U.S. homes by the end of the year 2018.

The word "domotics" (and "domotica" when used as a verb) is a contraction of the Latin word for a home (domus) and the word robotics. The word "smart" in "smart home" refers to the system being aware of the state of its devices, which is done through the information and communication technologies (ICT) protocol and the Internet of Things (IoT).

Throughout the 2000s, smart devices and systems have been evolving at a rapid pace. It’s estimated that by 2012, there were already 1.5 million automated home systems in place. In 2014, Amazon introduced the Amazon Echo (for Prime members), and while it was originally marketed as a voice-controlled music solution, the inclusion of Alexa quickly demonstrated the use of the device as a smart home hub.

#### 7.3 Applications and Technologies

Home automation is prevalent in a variety of different realms, including:

* Heating, ventilation and air conditioning (HVAC): it is possible to have remote control of all home energy monitors over the internet incorporating a simple and friendly user interface.

* Lighting control system: a "smart" network that incorporates communication between various lighting system inputs and outputs, using one or more central computing devices.

* Occupancy-aware control system: it is possible to sense the occupancy of the home using smart meters and environmental sensors like CO2 sensors, which can be integrated into the building automation system to trigger automatic responses for energy efficiency and building comfort applications.

* Appliance control and integration with the smart grid and a smart meter, taking advantage, for instance, of high solar panel output in the middle of the day to run washing machines.

* Home robots and security: a household security system integrated with a home automation system can provide additional services such as remote surveillance of security cameras over the Internet, or access control and central locking of all perimeter doors and windows.

* Leak detection, smoke and CO detectors

* Indoor positioning systems (IPS).

* Home automation for the elderly and disabled.

* Pet and baby care, for example tracking the pets and babies' movements and controlling pet access rights.

* Air quality control (inside and outside). For example, Air Quality Egg is used by people at home to monitor the air quality and pollution level in the city and create a map of the pollution.

* Smart kitchen, with refrigerator inventory, premade cooking programs, cooking surveillance, etc.

* Voice control devices like Amazon Alexa or Google Home used to control home appliances or systems.

#### 7.4 Implementations

In a review of home automation devices, Consumer Reports found two main concerns for consumers:

* A Wi-Fi network connected to the internet can be vulnerable to hacking.

* Technology is still in its infancy, and consumers could invest in a system that becomes abandonware. In 2014, Google bought the company selling the Revolv Hub home automation system, integrated it with Nest and in 2016 shut down the servers Revolv Hub depended on, rendering the hardware useless.

IoT Software Implementation Challenges:

* Compatibility of Various IoT Systems.
* Authentication and Identification Issues in IoT.

* Integration of IoT Points with IoT Software.
* IoT Data Storage Challenge.
* Connectivity and Power Management IoT Challenges.
* Unstructured Data Processing.
* Incorrect Data Capture Difficulties.

#### 7.5 Criticism and Controversy

Home automation suffers from platform fragmentation and lack of technical standards a situation where the variety of home automation devices, in terms of both hardware variations and differences in the software running on them, makes the task of developing applications that work consistently between different inconsistent technology ecosystems hard. Customers may hesitate to bet their IoT future on proprietary software or hardware devices that use proprietary protocols that may fade or become difficult to customize and interconnect.

#### 7.6 FUTURE SCOPE OF HOME AUTOMATION

Future scope for the home automation systems involves making homes even smarter. Homes can be interfaced with sensors including motion sensors, light sensors and temperature sensors and provide automated toggling of devices based on conditions. More energy can be conserved by ensuring occupation of the house before turning on devices and checking brightness and turning off lights if not necessary. The system can be integrated closely with

home security solutions to allow greater control and safety for homeowners. The next step would be to extend this system to automate a large-scale environment, such as offices and factories. Home Automation offers a global standard for interoperable products. Standardization enables smart homes that can control appliances, lighting, environment, energy management and security as well as the expandability to connect with other networks.

Fig.7.2: Internet Enabled Cat Feeder



**7.7 Setting up Smart Home**

Home automation refers to the ability of your home to make its own decisions depending on environment conditions and give you the option to control it from a remote location. The heart project is the WiFi enabled board that needs no introduction; the ESP8266 based [NodeMCU d](https://www.electronics-lab.com/project/getting-started-with-the-nodemcu-esp8266-based-development-board/)evelopment board. It is an open source platform for developing WiFi based embedded systems and it is based on the popular ESP8266 WiFi Module.

NodeMCUwas born out of the desire to overcome the limitations associated with the first versions of the ESP8266 module which was not compatible with breadboards, it was difficult to power and even more difficult to program. The NodeMCU board is easy to use, low cost and that quickly endeared it to the heart of makers and it is one of the most popular boards today.

we will add a 2-channel relay module to the ESP8266 board. The project flow involves the control of NodeMCU’s GPIOs from a webpage on any device connected on the same network as the board. The status of the GPIOs control the coils of the relays and that causes the relay to alternate between normally open (NO) and normally closed (NC) condition depending on the state of the GPIO, thus, effectively turning the connected appliance “ON” or “OFF”.

#### 7.8 Components

##### 7.8.1 ESP8266

* ESP8266 is a low-cost WiFi module that belongs to ESP's family which you can use it to control your electronics projects anywhere in the world. It has an in-built microcontroller and a 1MB flash allowing it to connect to a WiFi.

* The maximum working voltage of the module is 3.3v.
* Power to the ESP8266 NodeMCU is supplied via the on-board MicroB USB connector. Alternatively, if you have a regulated 5V voltage source, the VIN pin can be used to directly supply the ESP8266 and its peripherals.

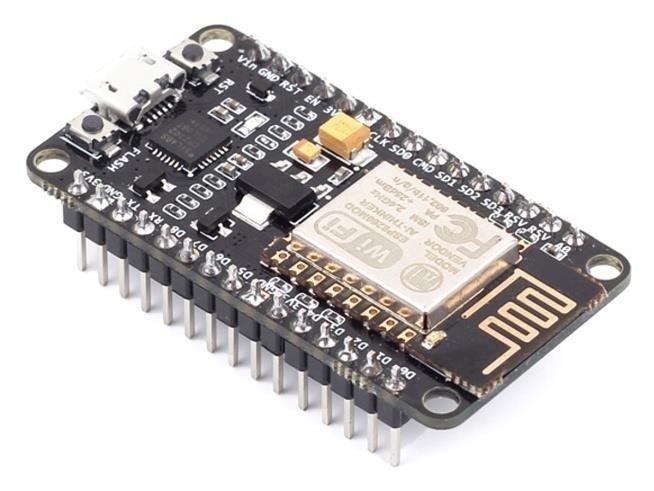


Fig.7.3 Diagram showing ESP8266

##### 7.8.2 Moisture Sensor

* The Soil Moisture Sensor measures soil moisture grace to the changes in electrical conductivity of the earth ( soil resistance increases with drought).
* The electrical resistance is measured between the two electrodes of the sensor.
* The soil moisture sensor consists of two probes that are used to measure the volumetric content of water. The two probes allow the current to pass through the soil, which gives the resistance value to measure the moisture value.

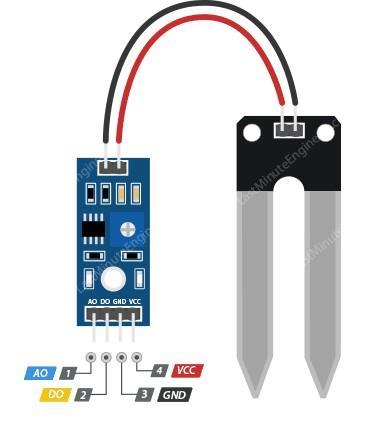


Fig 7.4 Diagram showing Moisture Sensor

##### 7.8.3 Soil Moisture Comparator

It measures the volumetric content of water inside the soil and gives us the moisture level as output. The comparator has both digital and analog outputs and a potentiometer to adjust the threshold level.

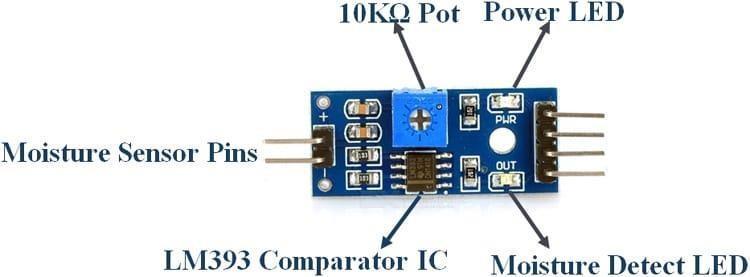


Fig 7.5 Diagram showing Soil Moisture Comparator

##### 7.8.4 Buck Converter

Buck converter is a step-down power supply module. It is used to bring down the power supply to the desired voltage in order to safeguard the circuit.



Fig 7.6 Diagram showing Buck Converter

##### 7.8.5 Relay Circuit

Relays are switches that open and close circuits electromechanically or electronically. Relays control one electrical circuit by opening and closing contacts in another circuit. When a relay contact is Normally Closed (NC), there is a closed contact when the relay is not energized.



Fig 7.7 Diagram showing Relay Circuit

##### 7.8.6 Pump & Bulb

**Pump:** it is used for water supply to the smart garden.

**Bulb:** it is home appliance that is been controlled through the relay switches.

**7.8.7 ESP8266 Pin Diagram**

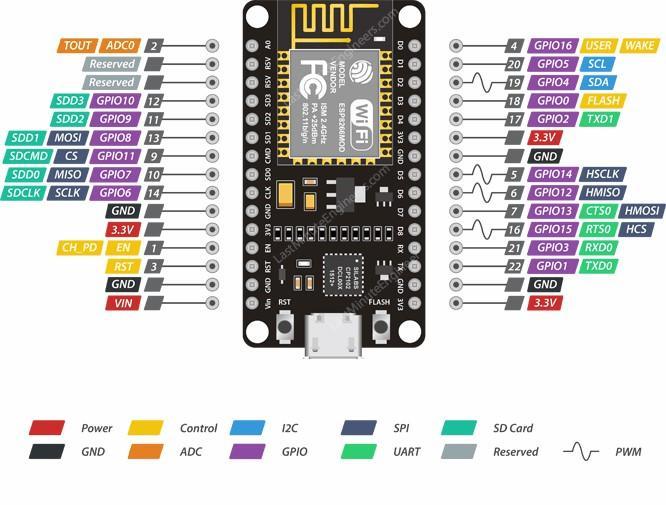


Fig 7.8 Diagram showing ESP8266 Pin Diagram

**7.8.8 Applications of Home Automation**

* **Lighting Control:** Leaving the Dark Ages and Stepping Into the Light. It turns out, quite; its capabilities are extensive. You’re able to schedule the lights should turn on and off, decide, select the level of light which should be emitted, and choose how particular lights react through controller.
* **HVAC Regulation:** No Longer Burned by Your Heating Bill. As fuel costs rise and the availability and sustainability of our resources becomes a greater concern, heating/cooling our homes efficiently is less a budgetary bonus and more of a necessity.
* **Classic smart home overview:** Smart home is the residential extension of building automation and involves the control and automation of all its embedded technology. It defines a residence that has appliances, lighting, heating, air conditioning, TVs, computers, entertainment systems, big home appliances such as washers/dryers and refrigerators/freezers, security and camera systems capable of communicating with each other and being controlled remotely by a time schedule, phone, mobile or internet.
* **Measuring home conditions:** A typical smart home is equipped with a set of sensors for measuring home conditions, such as: humidity, light and proximity. Each sensor is dedicated to capture one or more measurement. Temperature and humidity may be measured by one sensor, other sensors calculate the light ratio for a given area and the distance from it to each object exposed to it.
* **Managing home appliances:** Creates the cloud service for managing home appliances which will be hosted on a cloud infrastructure. The managing service allows the user, controlling the outputs of smart actuators associated with home appliances, such as such as lamps and fans.
* **Controlling home access:** Home access technologies are commonly used for public access doors. A common system uses a database with the identification attributes of authorized people.

**7.8.9 Moisture Sensor Pin Diagram:**



#### 7.8.10 Controller

* The controller is the device that receives the sensors' signals, processes them and makes computations on them, and then sends instruction signals to the actuators
* **Dashboard:** The user can display data from a temperature-humidity sensor alongside data from an air quality sensor and add a button to turn on the appliances in the house.
* **Triggers:** Use triggers in Adafruit IO to control and react to your data. Configure triggers to email you when your system goes offline, react to a temperature sensor getting too hot, and publish a message to a new feed.
* It can :
* Display your data in real-time, online
* Make your project internet-connected: Control motors, read sensor data, and more!
* Connect projects to web services like Twitter, RSS feeds, weather services, etc.
* Connect your project to other internet-enabled devices
* The best part? All of the above is do-able for freewith Adafruit IO

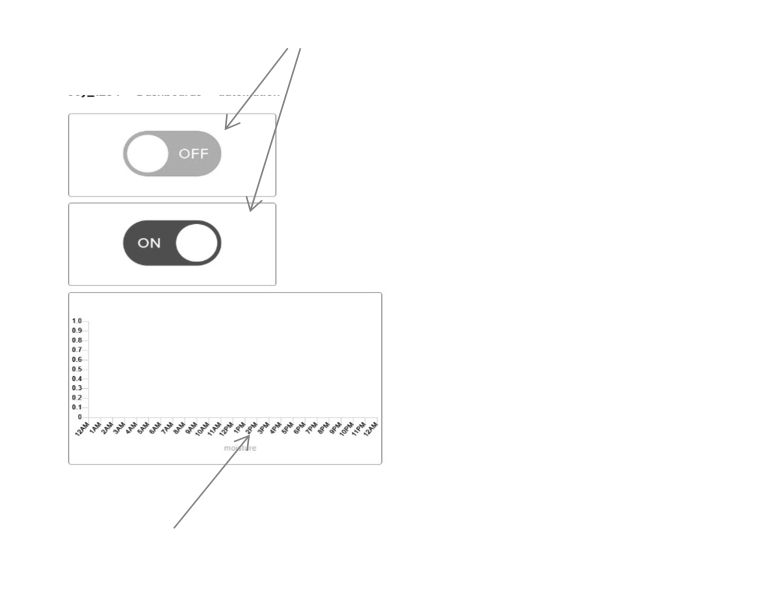


Fig 7.9 Diagram showing Controller Dashboard

#### 7.9 Adafruit

Adafruit.io is a cloud service- that just means it runs for you and you don't have to manage it. You can connect to it over the Internet. It's meant primarily for storing and then retrieving data but it can do a lot more than just that!

It can :

* Display your data in real-time, online
* Make your project internet-connected: Control motors, read sensor data, and more!
* Connect projects to web services like Twitter, RSS feeds, weather services, etc.
* Connect your project to other internet-enabled devices
* The best part? All of the above is do-able for freewith Adafruit IO.



Fig.7.10 Diagram showing Adafruit Logo

Implementation of Adafruit.io: In this project, we are going to use the ESP8266 to build the components which are very useful in home automation and smart gardening system: a sensor module, a controller, etc. We can link our project to Adafruit.io in order to make it ubiquitous, which makes it easy for the user to control it from any part of the world at any time.

##### 7.10 Steps to create dashboard on Adafruit

* When you login to your io.adafruit.com account, you will be redirected to your list of dashboards. It will look like the page seen below.

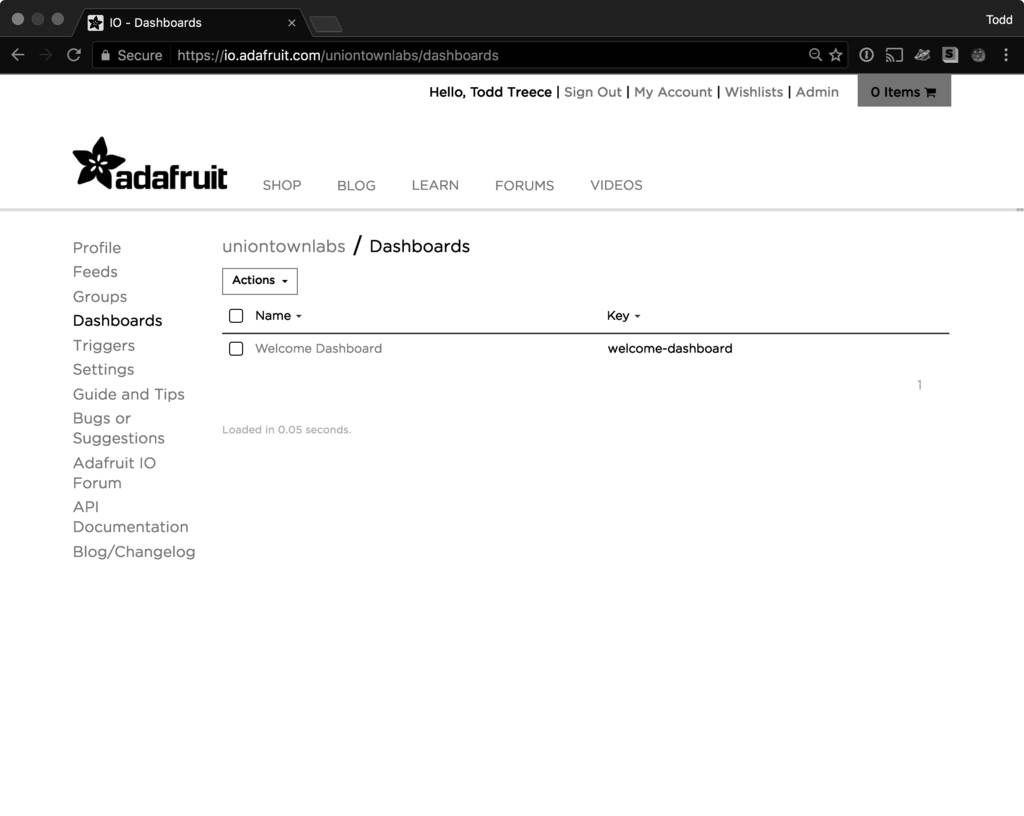


Fig.7.11 Diagram showing Adafruit Interface

* Your list of dashboards will only have the Welcome Dashboardwhen it is first loaded. You can start the dashboard creation process by clicking the **Actions** menu on the upper left hand side of the screen
* Next, select **Create a New Dashboard** from the dropdown menu.
* You can then enter the name and description of your new dashboard, and click the **Create** button once you are finished.
* Once your dashboard has been created, click on the name of your new dashboard to load it.
* You should now see your new blank dashboard.

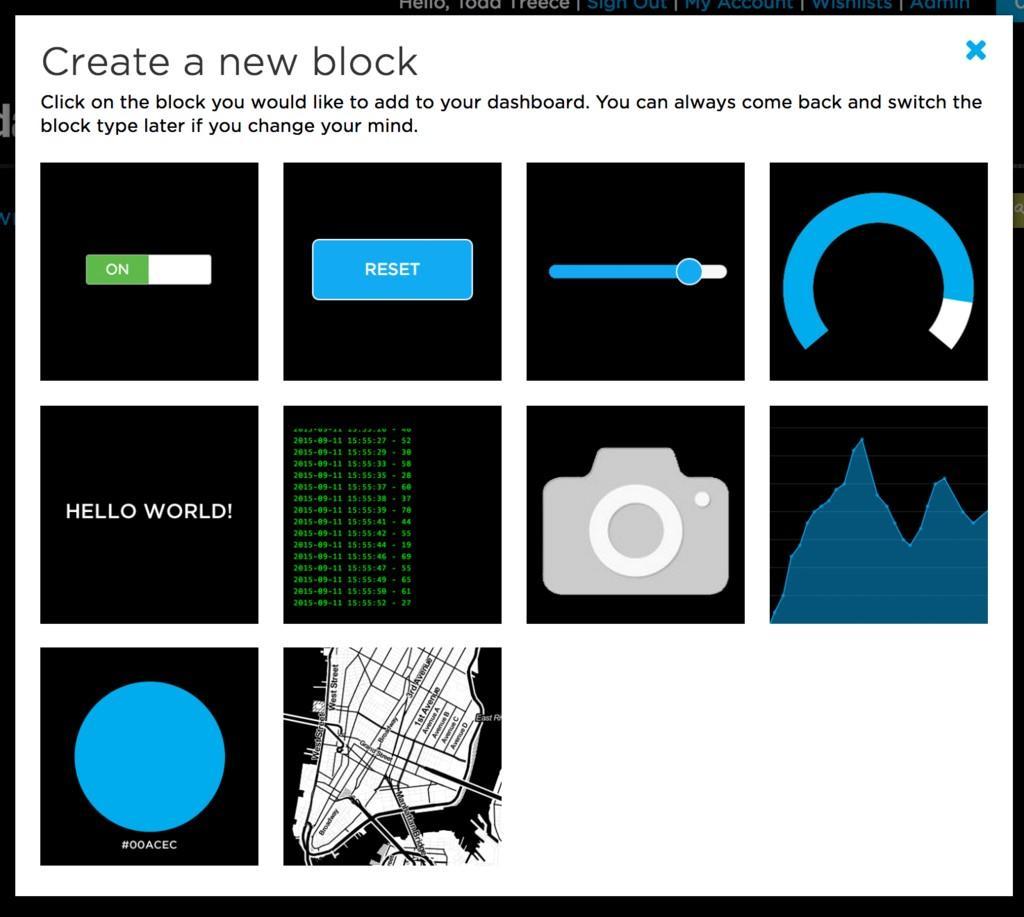


Fig.7.12 Diagram showing Adafruit Blocks

* Blocks are widgets that you can add to your dashboard. There are some blocks that can be used as outputs, and some that can be used as inputs.
* To add a new block, you can click the **+** (plus) button on the upper right hand side of the dashboard. You will then be presented with a list of block types to choose from, like the one seen below.
* **Toggle Button:** The toggle button will allow you to switch between any two text or numeric values. Unlike the momentary button, the values will stay the same until you click the button again to toggle to the second value.

* **Line Graph:** The line graph allows you to visualize numeric data over time. You can set the time range in hours that the chart will load, and set the labels for each axis of the chart. The chart will update dynamically whenever new values are pushed to the feed.

Navigating the Create New Block Form

There are three steps in the create block form:

1. Select block type
2. Select feed(s)
3. Block settings

Once you have selected a new block type by clicking on the icon, you will be presented with a list of feeds. This will allow you to select a feed to attach to your new block. Some blocks, such as the stream and line graph blocks, allow you to attach multiple feeds. Click on the checkbox next to the feed to select it, and click the **Next step** button to continue.

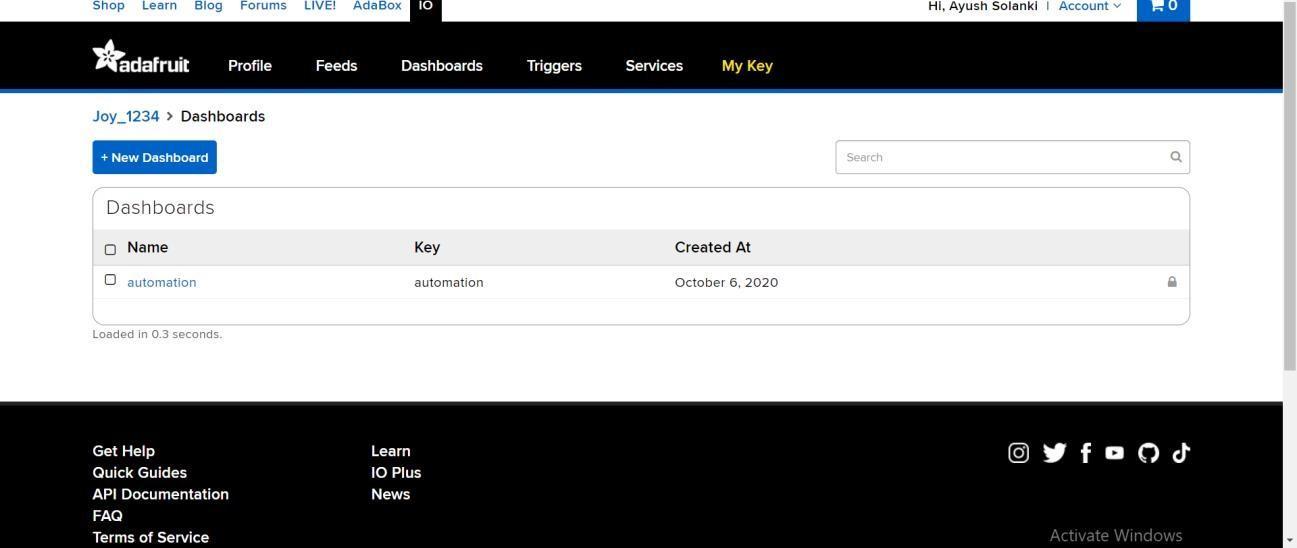


Fig.7.13 Diagram showing Adafruit Dashboards

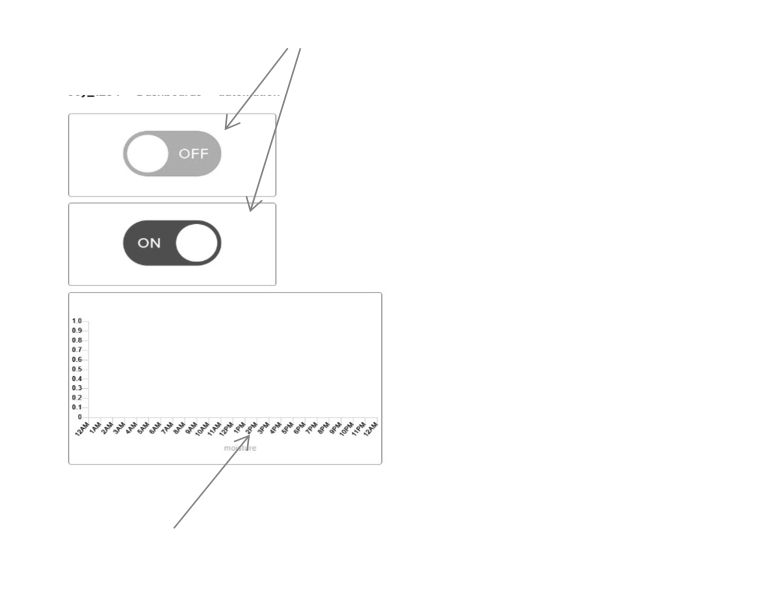


Fig 7.14 Diagram showing Adafruit Final Dashboards (Controller)

##### 7.11 Libraries of Adafruit included in Project

|  |  |  |
| --- | --- | --- |
|  | **ESP8266WiFi library:** |  |

The [Wi-Fi library for ESP8266](https://github.com/esp8266/Arduino/tree/master/libraries/ESP8266WiFi) has been developed based on [ESP8266 SDK,](https://github.com/espressif/ESP8266_NONOS_SDK) using the naming conventions and overall functionality philosophy of the [Arduino WiFi library.](https://www.arduino.cc/en/Reference/WiFi) Over time, the wealth of Wi-Fi features ported from ESP8266 SDK to [esp8266 / Arduino](https://github.com/esp8266/Arduino) outgrew [Arduino WiFi library](https://www.arduino.cc/en/Reference/WiFi) and it became apparent that we would need to provide separate documentation on what is new and extra.

This documentation will walk you through several classes, methods and properties of the [ESP8266WiFi](https://github.com/esp8266/Arduino/tree/master/libraries/ESP8266WiFi) library. If you are new to C++ and Arduino, don’t worry. We will start from general concepts and then move to detailed description of members of each particular class including usage examples.

The scope of functionality offered by the [ESP8266WiFi](https://github.com/esp8266/Arduino/tree/master/libraries/ESP8266WiFi) library is quite extensive and therefore this description has been broken up into separate documents marked with : arrow\_right :.

|  |  |
| --- | --- |
| • | **MQTT Adafruit library:** |

MQTT stands for Message Queuing Telemetry Transport. MQTT is a machine to machine internet of things connectivity protocol. It is an extremely lightweight and publish-subscribe messaging transport protocol. This protocol is useful for the connection with the remote location where the bandwidth is a premium. These characteristics make it useful in various situations, including constant environment such as for communication machine to machine and internet of things contexts. It is a publish and subscribe system where we can publish and receive the messages as a client. It makes it easy for communication between multiple devices. It is a simple messaging protocol designed for the constrained devices and with low bandwidth, so it's a perfect solution for the internet of things applications.

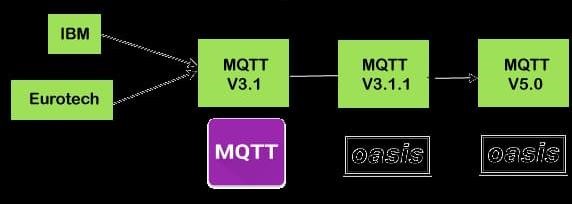
**Publish-Subscribe Model :**

This model involves multiple clients interacting with each other, without having any direct connection established between them. All clients communicate with other clients only via third party known as Broker.

**MQTT Client and Broker :**

Clients publish messages on different topics to broker. The broker is central server that receives these messages and filters them based on their topics. It then sends these messages to respective clients that have subscribed to those different topics.

Fig.7.15 Diagram showing Adafruit Final Dashboards (Controller)



#### 7.12 Software: Arduino

The **Arduino Integrated Development Environment (**[**IDE)**](https://en.wikipedia.org/wiki/Integrated_development_environment) is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) application (for [Windows,](https://en.wikipedia.org/wiki/Windows) [macOS,](https://en.wikipedia.org/wiki/MacOS) [Linux)](https://en.wikipedia.org/wiki/Linux) that is written in functions from [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++.](https://en.wikipedia.org/wiki/C%2B%2B_(programming_language)) It is used to write and upload programs to [Arduino](https://en.wikipedia.org/wiki/Arduino) compatible boards, but also, with the help of third-party cores, other vendor development boards.

The source code for the IDE is released under the [GNU General Public License,](https://en.wikipedia.org/wiki/GNU_General_Public_License) version 2. The Arduino IDE supports the languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B) using special rules of code structuring. The Arduino IDE supplies a [software library](https://en.wikipedia.org/wiki/Software_library) from the [Wiring](https://en.wikipedia.org/wiki/Wiring_(development_platform)) project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable [cyclic executive](https://en.wikipedia.org/wiki/Cyclic_executive) program with the [GNU toolchain,](https://en.wikipedia.org/wiki/GNU_toolchain) also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.By default, avrdude is used as the uploading tool to flash the user code onto official Arduino boards.



Fig.7.16 Diagram showing Arduino IDE Example

Arduino IDE is a derivative of the [Processing IDE,](https://en.wikipedia.org/wiki/Processing_(programming_language)) however as of version 2.0, the Processing IDE will be replaced with the [Visual Studio Code-](https://en.wikipedia.org/wiki/Visual_Studio_Code)based [Eclipse](https://en.wikipedia.org/wiki/Eclipse_Theia) IDE framework.

With the rising popularity of Arduino as a software platform, other vendors started to implement custom open source compilers and tools (cores) that can build and upload sketches to other microcontrollers that are not supported by Arduino's official line of microcontrollers.

#### 7.13 Circuit

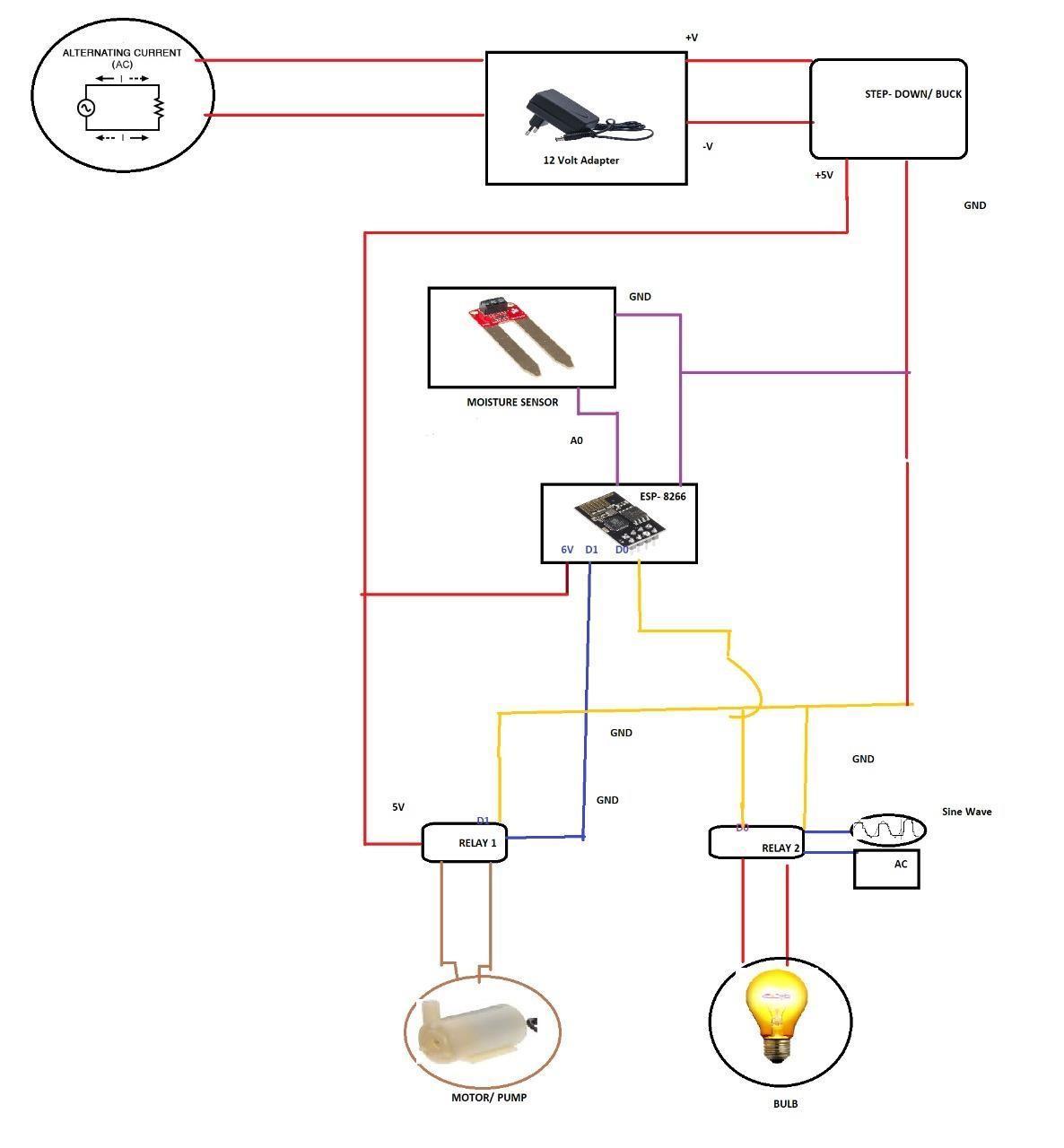


Fig.7.17 Diagram showing Project Circuit

**7.14 Internet Connectivity**

Fig 7.18 Diagram showing Internet Connectivity



The [**concept of Home Automation**](https://smartify.in/knowledgebase/what-is-home-automation/) aims to bring the control of operating your every day home electrical appliances to the tip of your finger, thus giving user affordable lighting solutions, better energy conservation with optimum use of energy.

Apart from just lighting solutions, the concept also further extends to have a overall control over your home security as well as build a centralised home entertainment system and much more. The **Internet of Things** (or commonly ).

#### 7.15 Features

* Highly secure system.
* Can be controlled worldwide.
* Can be used by multiple users.
* Can use multiple devices.
* Wireless connectivity.
* Real time sensing.

#### 7.16 Applications of project

* Lighting control.
* Smart Home Appliances.
* Improved Home safety and security.
* Home air quality and water quality monitoring.
* Smart Switches.
* Smart Energy Meters.

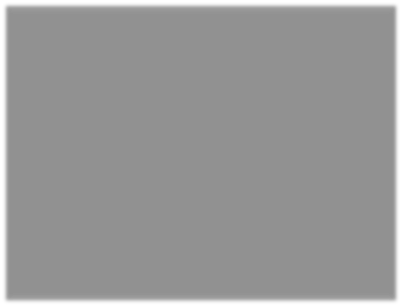
**7.17 Why this project is unique?**

* Increased efficiency, control, and customization
* Integration of Smart home devices
* smart irrigation system extends watering time for plants, and provides ideal growth condition.
* It saves time and timer delay as per the environmental condition can be added for automatic watering.
* This smart irrigation system can be adjusted and modified according to the changing environment.
* It works on a highly secured connection.
* It requires an user ID and a passkey to log in to the system.
* The user ID on which a controller is designed is a pseudo ID, i.e. it does not exist in real world and hence have no data attached to it.
* The code over which our system functions also requires the same ID and passkey for running the system.

#### 7.18 Snapshots of our Project

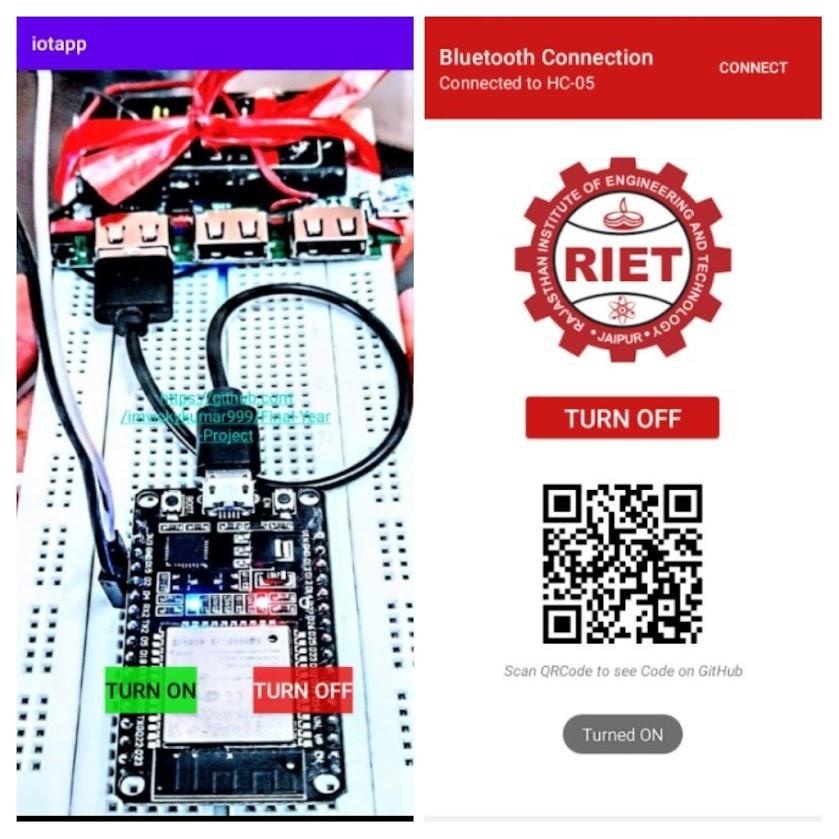
Fig.7.19 project snapshot







**\**



### LITERATURE REFERENCES

* 1. Dimitrios Serpanos Marilyn Wolf, “Internet- of- Things (IOT) systems, Architectures, Algorithms, Methodologies”, ISBN 978-3-319-69715-4, Springer International Publication, INC. 2018.
  2. [“The Internet of Things” by Samuel Greengard](https://www.amazon.com/Internet-Things-Press-Essential-Knowledge-ebook/dp/B00VB7I9VS/)
  3. “The Fourth Industrial Revolution” by Klaus Schwab
  4. [Arduino – Home [www.arduino.cc]](https://www.arduino.cc/)
  5. https://io.adafruit.com
  6. Armando Roy Delgado, Rich Picking and Vic Grout,“Remote-Controlled Home Automation Systems with Different Network Technologies”, Centre for Applied Internet Research (CAIR), University of Wales, NEWI, Wrexham, UK.
  7. Margaret Mouse. “Frequency-hopping spread spectrum”.
  8. Kim Baraka, Marc Ghobril, Sami Malek, RouwaidaKanj, AymanKayssi “Low cost Arduino/Android-based Energy-Efficient Home Automation System with Smart Task
  9. Scheduling”, 2013.
  10. www.wikkipedia.com
  11. www.tutorialspoint.com
  12. Edward Ashford Lee & Sanjit Arunkumar Seshia, “Introduction to Embedded Systems..
  13. A Cyber- physical System Approach“, UC Berkely, 2010.
  14. The official Bluetooth website from Bluetooth SIG www.Bluetooth. com, Date viewed: March 21, 2001, Bluetooth Specification Version 1.1.
  15. Bluetooth Committee, Specifications of the Bluetooth System (Core), December 1999, V1.0B
  16. Home System Specification, EHS, 1997.
  17. Ericsson Mobile Communications AB, User Manual—Bluetooth PC Reference Stack, 1543 VNX 2/901 184 Uen Version R1a, April 2000.
  18. I.S. McKenzie, The 8051 Microcontroller, Prentice Hall, Upper Saddler River, NJ, 1999.
  19. Phillips Semiconductors, 87C51 8-bit Microcontroller Data Sheet, 1999.
  20. Bluetooth Committee, Profiles of the Bluetooth System (Profiles), December 1999, V1.0B.
  21. J. Kelsey, Programming Plug and Play, Sam’s Publication, 1995.
  22. USB Design By Example, A Practical Guide of Building I/O Devices, John Hyde, Wiley, New York, 1999.
  23. Philips Semiconductors, The I2 C Bus Specification, Version 2.1, January 2000. [11] Philips Semiconductors, The I2 C Bus and How to Use It (Including Specification, Version 1.0), Technical Manual, 1997.
  24. I2 C Serial Bus Analyzer, N. Sriskanthan, Tan Sue Lim, IWNA’01, Singapore, 2001, p. 106–114.
  25. Maxim Integrated Products, Max233 12V Line Driver Chip Datasheet, 2000.
  26. Maxim Integrated Products, MAX1669 Fan controller and Remote Temperature Sensor Datasheet, 2000.
  27. W. Stalling, Data and Computer Communications, Prentice Hall, Englewood Cliffs, NJ, 1997.