CHAPTER 1

INTRODUCTION

1.1 Internet of Things :

The Internet of things (IoT) is about extending the power of internet beyond computers and smartphones to a whole range of other things, processes and environments. Those connected things are used to gather information, send information back, or both. IoT allows businesses and people to be more connected to the world around them and to do more meaningful, high-level work. It is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.

An internet connection is a wonderful thing. It gives us all sorts of benefits that just weren't possible before. If you are old enough, think of your cell phone before it was a smart phone. The point is that connecting things to the internet yields many amazing benefits. It means taking all the things in the world and connecting them to the internet.

The IoT is significant because an object that can represent itself digitally becomes something greater than the object by itself. No longer does the object relate just to its user, but it is now connected to surrounding objects and database data. When many objects act in unison, they are known as having "ambient intelligence."

The Internet of things is a difficult concept to define precisely. In fact, there are many different groups that have defined the term, although its initial use has been attributed to Kevin Ashton, an expert on digital innovation. Each definition shares the idea that the first version of the internet was about data created by people, while the next version is about data created by things. In 1999, Ashton said it best in this quote from an article in the RFID journal:

If we had computers that knew everything there was to know about things- using data they gathered without any help from us- we would be able to track and count everything, and greatly reduce waste, loss, and cost. We would know when things needed replacing, repairing or recalling, and whether they were fresh or past their best.

Most people think about being connected in terms of computers, tablets and smartphones. IoT describes a world where just about anything can be connected and communicate in an intelligent fashion. In other words, with the internet of things, the physical world is becoming one big information system.

1.2 Why do connected devices need to share data? :

An argument has been raised that only because something can be connected to the internet doesn’t mean it should be, but each device collects data for a specific purpose that may be useful to a buyer and impact the wider economy.

Within industrial applications, sensors on product lines can increase efficiently and cut down on waste. One study estimates 35 percent of US manufactures are using data from smart sensors within their set-ups already. US firm concrete sensors has created a device that can be inserted into concrete to provide data on the material’s condition, for instance.

IoT offers us opportunity to be more efficient in how do we do things, saving us time, money and often emissions in the process. It allows companies, governments and public authorities to rethink how they deliver services and produce goods.

The quality and scope of the data across the Internet of things generates an opportunity for much more contextualised and responsive interactions with devices to create a potential for change.

1.3 Where does the IoT go next? :

Even those who have purchased one of the myriad smart home products – from light bulbs, switches, to motion sensors – will attest to the fact IoT is in its infancy. Products don't always easily connect to each other and there are significant security issues that need to be addressed.

A report from Samsung says the need to secure every connected device by 2020 is "critical". The firm's Open Economy document says "there is a very clear danger that technology is running ahead of the game". The firm said more than 7.3 billion devices will need to be made secure by their manufacturers before 2020.

“We are looking at a future in which companies will indulge in digital Darwinism, using IoT, AI and machine learning to rapidly evolve in a way we’ve never seen before," Brian Solis, from Altimeter Group, who helped on the research said.

IoTbotnets, created using a network of out-of-date devices took large websites and services offline in 2016. A Chinese firm later recalled 4.3 million unsecured connected cameras. The ease of bringing down the internet using IoT devices was revealed when instead of malicious purposes, the botnet was revealed to have been created to game Minecraft.

At the centre of creating a vast, reliable IoT network lies one significant issue: compatible standards. Connected objects need to be able to speak to each other to transfer data and share what they are recording. If they all run on different standards, they struggle to communicate and share. The Institute of Electrical and Electronics Standards Association lists a huge number of standards being developed and worked on for different applications."Additional needs are emerging for standardisation," the Internet Society says. If standardisation happens it will let more devices and applications be connected.

CHAPTER 2

LITERATURE SURVEY

2.1 Smart Irrigation overview:

Generally, Irrigation is an artificial application of water to land for the purpose of agricultural production. Effective irrigation will influence the entire growth process from seedbed preparation, germination, root growth, nutrient utilisation, plant growth and regrowth, yield and quality.

The key to maximising irrigation efforts is uniformity. The producer has a lot of control over how much water to supply and when to apply it but the irrigation system determines uniformity. Deciding which irrigation systems is best for your operation requires a knowledge of equipment, system design, plant species, growth stage, root structure, soil composition, and land formation. Irrigation systems should encourage plant growth while minimising salt imbalances, leaf burns, soil erosion, and water loss. Losses of water will occur due to evaporation, wind drift, run-off and water (and nutrients) sinking deep below the root zone. Proper irrigation management takes careful consideration and vigilant observation.

2.2 The value of irrigation:

Irrigation allows primary producers,

- to grow more pastures and crops
- to have more flexibility in their systems/operations as the ability to access water at times when it would otherwise be hard to achieve good plant growth (due to a deficit in soil moisture) is imperative. Producers can then achieve higher yields and meet market/seasonal demands especially if rainfall events do no occur.
- to produce higher quality crops/pastures as water stress can dramatically impact on the quality of farm produce
- to lengthen the growing season (or in starting the season at an earlier time)

- To have 'insurance' against seasonal variability and drought.
- to stock more animals per hectare and practice tighter grazing management due to the reliability of pasture supply throughout the season
- To maximise benefits of fertiliser applications. Fertilisers need to be 'watered into' the ground in order to best facilitate plant growth.
- To use areas that would otherwise be 'less productive'. Irrigation can allow farmers to open up areas of their farms where it would otherwise be 'too dry' to grow pasture/crops. This also gives them the capability to carry more stock or to conserve more feed.
- to take advantage of market incentives for unseasonal production
- to have less reliance on supplementary feeding (grain, hay) in grazing operations due to the more consistent supply & quality of pastures grown under irrigation
- To improve the capital value of their property. Since irrigated land can potentially support higher crops, pasture and animal production, it is considered more valuable. The value of the property is also related to the water licensing agreements or 'water right'.
- To cost save/obtain greater returns. The cost benefits from the more effective use of fertilisers and greater financial benefits as a result of more effective agricultural productivity (both quality and quantity) and for 'out of season' production are likely.

2.2.1Choice of an irrigation system:

There is a huge diversity in the types of irrigation technologies/systems used, which is attributable to,

- Variations in soil types
- Varying topography of the land
- Availability of power sources
- Availability of water
- Sources of water
- The period of time when the system was installed
- The size of the area being irrigated
- On farm water storage capacity
- Availability of labour/financial resources

2.3 IoT in Irrigation systems:

Technology is continuing to make our life better and often in unprecedented ways. This time, it is about finding a solution to the drought-ridden agriculture. Yes, you have guessed it right. Advanced technologies have arrived to offer an effective solution to various irrigation problems. What kind of technology it is and how it is going to contribute to irrigation, we are discussing all this in detail right here.

The interconnected objects referred as Internet of Things (IoT) is continuing to evolve offering more control over our living environment and allowing more ease in doing things. Many consider this as the next big horizon in the evolution of the Internet. Thanks to the robust capability of collecting, storing, analysing and distributing data among diverse interfaces, apps and devices, the freedom for real-time application of data and data-driven insights has become easier than ever before.

But how this pool of data-driven insights can help irrigation? Well, it is more about obtaining the real-time information about various aspects pertaining to irrigation. For instance, there are sensors to remotely offer bore meter reading, deliver real-time data about water level in a river or even the moisture level in a farm land. All these data input from high-tech sensors can help us managing irrigation and irrigation resources better.

In drought-ridden places, such maneuvers can really bring green revolution by fulfilling the demands of irrigation through resource management. We cannot forget to mention that aerial drones surveying the farmlands can also furnish us with valuable insights in this respect.

2.4 Smart water meters and sensors:

How water usage in homes, agricultural lands or in business places is measured? Well, it is measured through analog meters in the majority of cases. These meters are incapable of offering real-time water usage data to the authorities because they remain unconnected. On the other hand, the real-time report about the usage of water usage can not only control but can also keep track of the usage. Smart connected meters with this capability to notify about the water level and usage can help prevent misuse and underutilisation of water resources.

The smart sensors in the farmland are additionally capable of notifying in a real time about the moisture level in lands and can prevent spoiling of water. This capability can be further aggravated if the real-time sensor data can trigger action in the meter by switching it off or on. The meter can be automatically switched on or off depending on the need of irrigation and level of the water resource.

Moreover, smart sensors can detect the faults in the irrigation system in a real time and can prevent draining of water in the absence of proper supervision. For leaks in pipelines, such smart sensors are doubly effective as they are often hidden from view and make detection of leaks really troublesome. The prompt detection of leaks and faults in will enable users addressing the issue quickly and will help saving water.

Farming needs sustained irrigation that itself consumes the highest percentage of water resource in any area. For instance, in California alone, irrigation of various ornamental plants and crops consumes over 80 percent of water usage. As experts always perceived, a high percentage of this irrigation water is actually wasted due to lack of supervision and real-time monitoring. In many places like California drought is a recurring problem due to lack of control in the irrigation water.

To solve this problem smart irrigation systems powered by latest IoT technology can help conservation of water resources better by monitoring irrigation through remote sensing technologies. Smart connected sensors work with smart sprinklers leveraging is a great example of the application of IoT in agriculture. To equip the smart irrigation further smart sensors can be embedded in the farmland soil that can easily measure the level of moisture. The moisture measuring sensors of the farmland can further relay the information about moisture level to the smart sprinkler and it can start sprinkling the right amount of water upon the soil.

2.5 Latest IoT devices helping irrigation and agriculture doing smart:

The recent development in remote sensor-enabled smart irrigation systems gave birth to several IoT devices and help make irrigation and agriculture smarter and better. Here are two of the latest IoT devices developed and successfully deployed to make agriculture smarter a reality.

2.5.1 CropX:

CropX is the latest IoT-enabled smart irrigation device that is built combining the soil sensor and cloud-based data analytics. CropX is affordable and offers exceptional ease in day-to-day usage by farmers with least technical know-how. The device will not only help improving yields of crops but will also contribute to reducing the cost of water resources.

Adequate water supply is crucial for proper yields while conserving water resource is important to prevent a shortage. CropX can address both of these aspects of agriculture. The device remains connected with a website that remotely offers data about the moisture level of farmland and the user in the real-time can schedule irrigation through this web interface.

2.5.2 eBee Ag:

The use of drone or UAV has been very popular to monitor the farmland remotely. Through the drone captured images of the acres of farmland, the farmers can obtain accurate view and data about their yields, soil condition and water level. eBee Ag is an advanced drone developed specifically for the agricultural purpose. The drone can provide accurate images of every meter of a large land even on a clouded day. Moreover, the images provided by this drone are resolution-wise better than those provided by the satellites.

The images corresponding to various agricultural aspects like the plant growth, yield of crops, harvesting and fertilization needs can further be processed into a graphic output offering crystal clear spatial information about every meter area of the farm field. This would obviously help cutting agriculture cost, managing harvesting and yield and making irrigation better.

2.6 ESP8266 Wi-Fi module:

ESP8266 is Wi-Fi enabled system on chip (SoC) module developed by Espressif system. It is mostly used for development of IoT (Internet of Things) embedded applications.

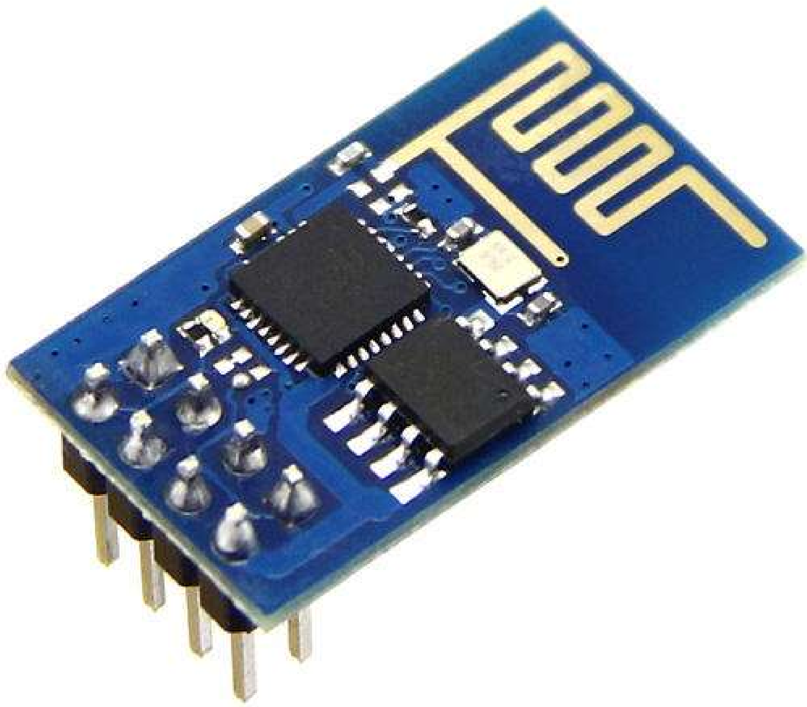


Fig 2.6.1 ESP8266 Wi-Fi module.

ESP8266 comes with capabilities of

- 2.4 GHz Wi-Fi (802.11 b/g/n, supporting WPA/WPA2),
- general-purpose input/output (16 GPIO),
- Inter-Integrated Circuit (I^2C) serial communication protocol,
- analog-to-digital conversion (10-bit ADC)
- Serial Peripheral Interface (SPI) serial communication protocol,
- I^2S (Inter-IC Sound) interfaces with DMA(Direct Memory Access) (sharing pins with GPIO),
- UART (on dedicated pins, plus a transmit-only UART can be enabled on GPIO2), and

- Pulse-width modulation (PWM).

It employs a 32-bit RISC CPU based on the TensilicaXtensa L106 running at 80 MHz (or overclocked to 160 MHz). It has a 64 KB boot ROM, 64 KB instruction RAM and 96 KB data RAM. External flash memory can be accessed through SPI.

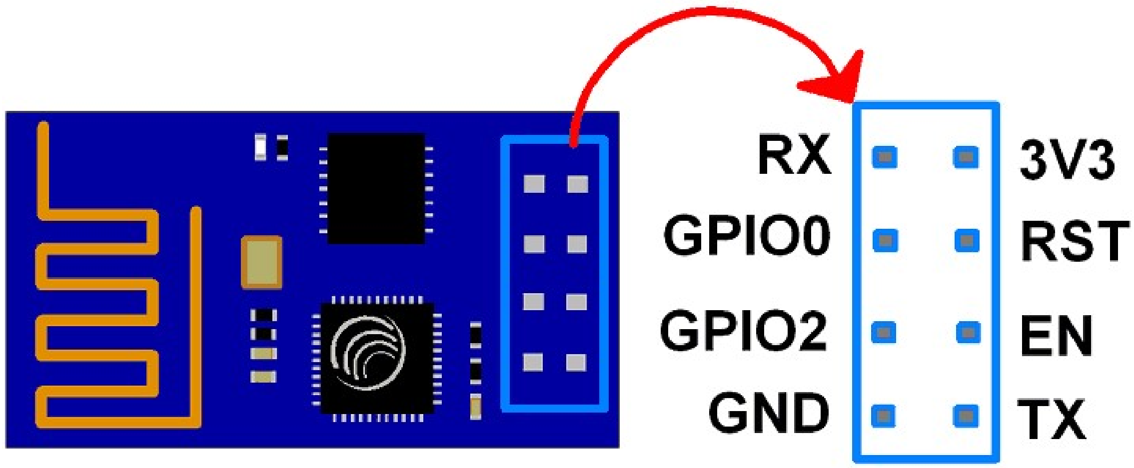


Fig. 2.6.2 ESP8266-01 Module pins

2.7 Arduino – an introduction:

2.7.1 Arduino Uno:

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board.

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package.

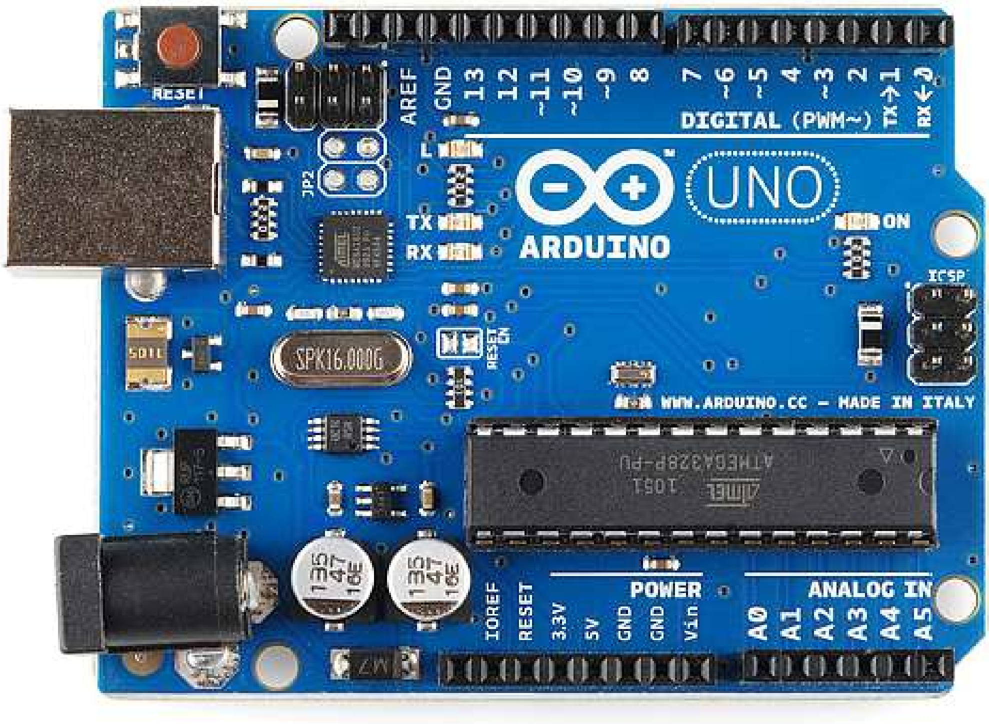


Fig 2.7.1An Arduino Uno

The Uno is one of the more popular boards in the Arduino family and a great choice for beginners. We'll talk about what's on it and what it can do later in the tutorial.

The Arduino hardware and software was designed for artists, designers, hobbyists, hackers, newbies, and anyone interested in creating interactive objects or environments. Arduino can interact with buttons, LEDs, motors, speakers, GPS units, cameras, the internet, and even your smart-phone or your TV! This flexibility combined with the fact that the Arduino software is free, the hardware boards are pretty cheap, and both the software and hardware are easy to learn has led to a large community of users who have contributed code and released instructions for a **huge** variety of Arduino-based projects.

For everything from robots and a heating pad hand warming blanket to honest fortune-telling machines, and even a Dungeons and Dragons dice-throwing gauntlet, the Arduino can be used as the brains behind almost any electronics project.

Power (USB / Barrel Jack)

Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply (like this) that is terminated in a barrel jack. In the picture above the USB connection is labelled **(1)** and the barrel jack is labelled **(2)**.

The USB connection is also how you will load code onto your Arduino board. More on how to program with Arduino can be found in our Installing and Programming Arduino tutorial.

NOTE: Do NOT use a power supply greater than 20 Volts as you will overpower (and thereby destroy) your Arduino. The recommended voltage for most Arduino models is between 6 and 12 Volts.

Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF)

The pins on your Arduino are the places where you connect wires to construct a circuit (probably in conjunction with a breadboard and some wire. They usually have black plastic 'headers' that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labelled on the board and used for different functions.

- **GND (3)**: Short for 'Ground'. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- **5V (4) & 3.3V (5)**: As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
- **Analog (6)**: The area of pins under the 'Analog In' label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.
- **Digital (7)**: Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
- **PWM (8)**: You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). We have a tutorial on PWM, but for now, think of these pins as being able to simulate analog output (like fading an LED in and out).

- **AREF (9)**: Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Reset Button

Just like the original Nintendo, the Arduino has a reset button **(10)**. Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn't repeat, but you want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn't usually fix any problems.

Power LED Indicator

Just beneath and to the right of the word "UNO" on your circuit board, there's a tiny LED next to the word 'ON' **(11)**. This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check your circuit!

TX RX LEDs

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. In our case, there are two places on the Arduino UNO where TX and RX appear -- once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs **(12)**. These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).

Main IC

The black thing with all the metal legs is an IC, or Integrated Circuit **(13)**. Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATmega line of IC's from the ATMEL Company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software. This information can usually be found in writing on the top side of the IC. If you want to know more about the difference between various IC's, reading the datasheets is often a good idea.

Voltage Regulator

The voltage regulator **(14)** is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it's for. The voltage regulator does exactly what it says -- it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up your Arduino to anything greater than 20 volts.

The Arduino Family

Arduino makes several different boards, each with different capabilities. In addition, part of being open source hardware means that others can modify and produce derivatives of Arduino boards that provide even more form factors and functionality.

Arduino Uno (R3)

The Uno is a great choice for your first Arduino. It's got everything you need to get started, and nothing you don't. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a USB connection, a power jack, a reset button and more. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

2.7.2 Arduino -pin diagram:

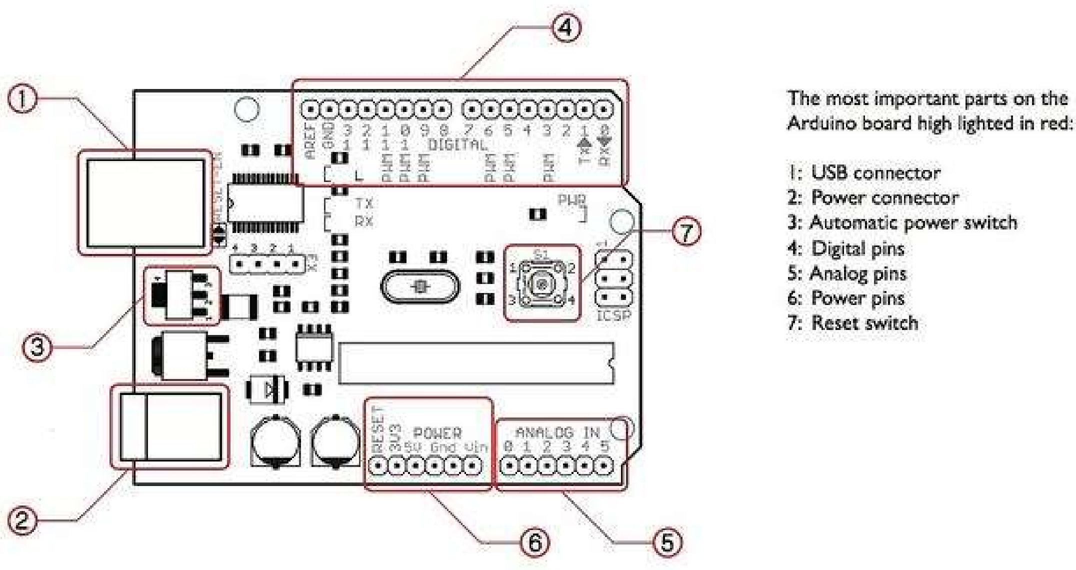


Fig 2.7.2 Arduino pin diagram

Basically, the processor of the Arduino board uses the Harvard architecture where the program code and program data have separate memory. It consists of two memories such as program memory and data memory. Wherein the data is stored in data memory and the code is stored in the flash program memory. The Atmega328 microcontroller has 32kb of flash memory, 2kb of SRAM 1kb of EPROM and operates with a 16MHz clock speed.

The main advantage of the Arduino technology is, you can directly load the programs into the device without the need of a hardware programmer to burn the program. This is done because of the presence of the 0.5KB of boot loader that allows the program to be dumped into the circuit. The Arduino tool window contains a toolbar with a various buttons like new, open, verify, upload and serial monitor. And additionally it comprises of a text editor (employed to write the code), a message space (displays the feedback) like showing the errors, the text console, that displays the o/p & a series of menus just like the file, tool menu & edit.

- Programming into the Arduino board is called as sketches. Each sketch contains of three parts such as Variables Declaration, Initialization and Control code. Where, Initialization is written in the setup function and Control code is written in the loop function.
- The sketch is saved with .ino and any operation like opening a sketch, verifying and saving can be done using the tool menu.
- The sketch must be stored in the sketchbook directory.
- Select the suitable board from the serial port numbers and tools menu.
- Select the tools menu and click on the upload button, then the boot loader uploads the code on the microcontroller.

2.7.2.1 Basic Functions of Arduino Technology:

- Digital read pin reads the digital value of the given pin.
- Digital write pin is used to write the digital value of the given pin.
- Pin mode pin is used to set the pin to I/O mode.
- Analog read pin reads and returns the value.
- Analog write pin writes the value of the pin.
- Serial. Begins pin sets the beginning of serial communication by setting the rate of bit.

2.8 Soil moisture sensor:

The soil moisture sensor consists of two probes which are used to measure the volumetric content of water. The two probes allow the current to pass through the soil and then it gets the resistance value to measure the moisture value.

When there is more water, the soil will conduct more electricity which means that there will be less resistance. Therefore, the moisture level will be higher. Dry soil conducts electricity poorly, so when there will be less water, then the soil will conduct less electricity which means that there will be more resistance. Therefore, the moisture level will be lower. This sensor can be connected in two modes; Analog mode and digital mode. First, we will connect it in Analog mode and then we will use it in Digital mode.

To connect the soil moisture sensor FC-28 in the digital mode, we will connect the digital output of the sensor to the digital pin of the Arduino. The Sensor module contains a potentiometer with it, which is used to set the threshold value. This threshold value is then compared with the sensor output value using the LM393 comparator which is placed on the sensor module.

The LM393 comparator will compare the sensor output value and the threshold value and then gives us the output through the digital pin. When the sensor value will be greater than the threshold value, then the digital pin will give us 5V and the LED on the sensor will light up and when the sensor value will be less than this threshold value, then the digital pin will give us 0V and the light will go down.

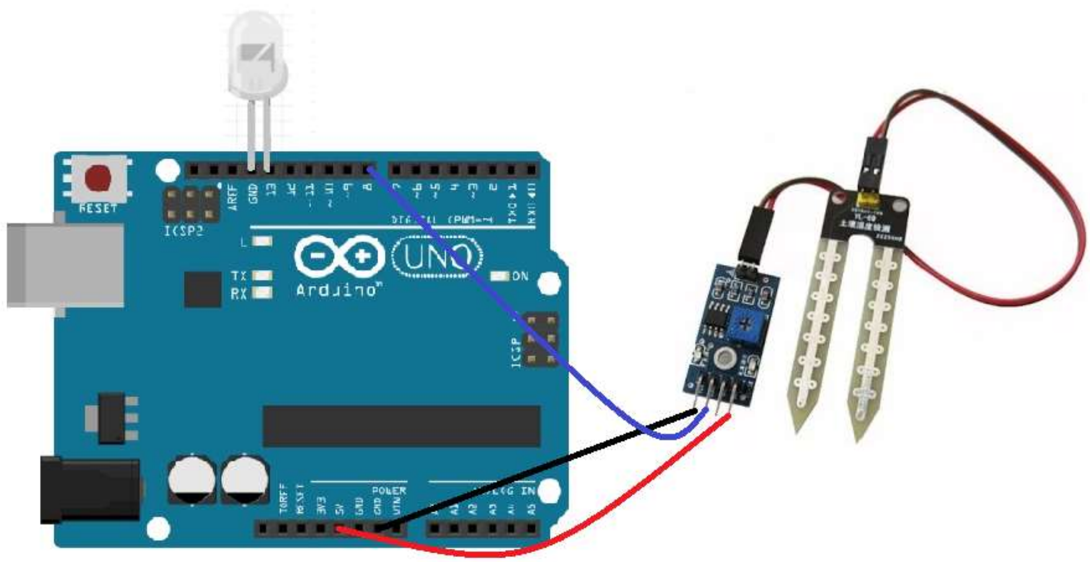


Fig 2.8 Arduino- soil moisture connection

2.9 Embedded C:

Embedded C Programming is the soul of the processor functioning inside each and every embedded system we come across in our daily life, such as mobile phone, washing machine, and digital camera.

Each processor is associated with an embedded software. The first and foremost thing is the embedded software that decides functioning of the embedded system. Embedded C language is most frequently used to program the microcontroller.

Earlier, many embedded applications were developed using assembly level programming. However, they did not provide portability. This disadvantage was overcome by the advent of various high level languages like C, Pascal, and COBOL. However, it was the C language that got extensive acceptance for embedded systems, and it continues to do so. The C code written is more reliable, scalable, and portable; and in fact, much easier to understand.

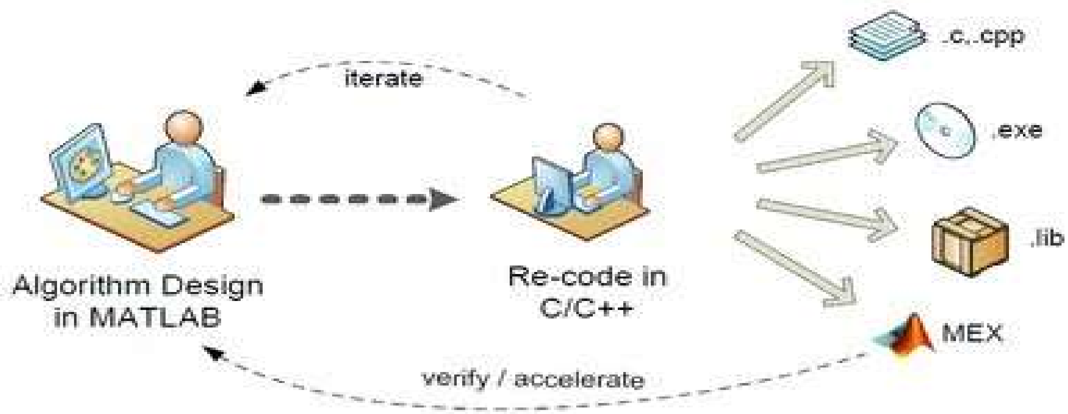


Fig2.9. embedded C programming

Looking around, we find ourselves to be surrounded by various types of embedded systems. Be it a digital camera or a mobile phone or a washing machine, all of them has some kind of processor functioning inside it. Associated with each processor is the embedded software. If hardware forms the body of an embedded system, embedded processor acts as the brain, and embedded software forms its soul. It is the embedded software which primarily governs the functioning of embedded systems.

During infancy years of microprocessor based systems, programs were developed using assemblers and fused into the EPROMs. There used to be no mechanism to find what the program was doing. LEDs, switches, etc. were used to check correct execution of the program. Some 'very fortunate' developers had In-circuit Simulators (ICEs), but they were too costly and were not quite reliable as well.

As time progressed, use of microprocessor-specific assembly-only as the programming language reduced and embedded systems moved onto C as the **embedded programming language** of choice. C is the most widely used programming language for embedded processors/controllers. Assembly is also used but mainly to implement those portions of the code where very high timing accuracy, code size efficiency, etc. are prime requirements.

Initially C was developed by Kernighan and Ritchie to fit into the space of 8K and to write (portable) operating systems. Originally it was implemented on UNIX operating systems. As it was intended for operating systems development, it can manipulate memory addresses.

Also, it allowed programmers to write very compact codes. This has given it the reputation as the language of choice for hackers too.

As assembly language programs are specific to a processor, assembly language didn't offer portability across systems. To overcome this disadvantage, several high level languages, including C, came up. Some other languages like PLM, Modula-2, Pascal, etc. also came but couldn't find wide acceptance. Amongst those, C got wide acceptance for not only embedded systems, but also for desktop applications. Even though C might have lost its sheen as mainstream language for general purpose applications, it still is having a strong-hold in embedded programming. Due to the wide acceptance of **C in the embedded systems**, various kinds of support tools like compilers & cross-compilers, ICE, etc. came up and all this facilitated development of **embedded systems using C**.

Subsequent sections will discuss **what Embedded C is**, **features of C language**, similarities and **difference between C and embedded C**, and **features of embedded C programming**.

Embedded systems programming is different from developing applications on a desktop computers. Key characteristics of an embedded system, when compared to PCs, are as follows:

· Embedded devices have resource constraints (limited ROM, limited RAM, limited stack space, less processing power)

· Components used in embedded system and PCs are different; embedded systems typically uses smaller, less power consuming components. · Embedded systems are more tied to the hardware.

Two salient **features of Embedded Programming** are code speed and code size. Code speed is governed by the processing power, timing constraints, whereas code size is governed by available program memory and use of programming language. Goal of embedded system programming is to get maximum features in minimum space and minimum time.

Embedded systems are programmed using different type of languages:

- Machine Code
- Low level language, i.e., assembly
- High level language like C, C++, Java, Ada, etc.
- Application level language like Visual Basic, scripts, Access, etc.

Assembly language maps mnemonic words with the binary machine codes that the processor uses to code the instructions. Assembly language seems to be an obvious choice for programming embedded devices. However, use of assembly language is restricted to developing efficient codes in terms of size and speed. Also, assembly codes lead to higher software development costs and code portability is not there. Developing small codes are not much of a problem, but large programs/projects become increasingly difficult to manage in assembly language. Finding good assembly programmers has also become difficult nowadays. Hence high level languages are preferred for embedded systems programming.

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATIONS

3.1 Hardware specifications:

- Processor : Pentium 3, Pentium 4 and higher
- RAM : 256MB/4GB RAM and higher
- Hard disk : 40GB and higher
- Arduino Uno Board
- 1 Pin jumper wires –female to female
- 1 Pin jumper wires – male to male
- 1 pin jumper wire – male to female
- 9V battery snap connector
- DC barrel connector jackpin lead type male
- Double side tape 10cm
- ESP8266 wifi module
- L293d motor driver
- Screw driver
- Water pump DC 3-6V micro submersible mini
- Wire strippers
- Wooden chassis – 12cm*12cm
- YL-69 Soil moisture probe
- YL-38 Soil moisture sensor module

3.2 Software specifications:

- Operating System : Windows 7, Windows 8 (or higher versions)
- Language : Embedded C
    - Mozilla Firefox (or any browser)
    - Microsoft Excel or Open office.
- Arduino IDE installation
- Virtuno application
- ThingspeakIoT platform

CHAPTER 4

SYSTEM ANALYSIS

4.1 ESP8266 Wi-fi module:

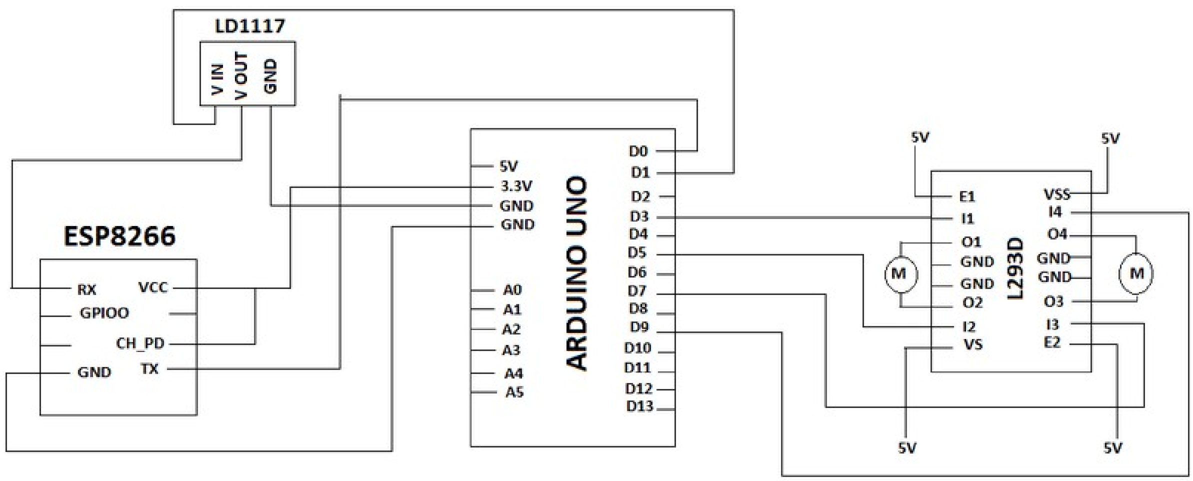


Fig 4.1 Circuit diagram of ESP8266 Wi-fi module

ESP8266-01 Features

- Low cost, compact and powerful Wi-Fi Module
- Power Supply: +3.3V only
- Current Consumption: 100mA
- I/O Voltage: 3.6V (max)
- I/O source current: 12mA (max)
- Built-in low power 32-bit MCU @ 80MHz
- 512kB Flash Memory
- Can be used as Station or Access Point or both combined
- Supports Deep sleep (<10uA)
- Supports serial communication hence compatible with many development platform like Arduino
- Can be programmed using Arduino IDE or AT-commands or Lua Script

4.2 Feasibility study:

A credibility contemplate expects to fair-mindedly and soundly uncover the qualities and inadequacies of a present business or proposed meander, openings and threats present in nature, the benefits required to bring through, and in the long run the prospects for advance. In its most clear terms, the two criteria to judge believability are incurred significant injury required and motivator to the fulfilled.

An inside and out arranged feasibility ponder should give a recorded establishment of the business or wander, a delineation of the thing or organization, accounting explanations, purposes of enthusiasm of the operations and organization, publicizing examination and game plans, budgetary data, authentic necessities and cost duties. All things considered, plausibility looks at go before specific change and wander utilization. There are three sorts of attainability

- Economical Feasibility
- Technical Feasibility
- Operational Feasibility

Economical feasibility

The electronic structure manages the present existing system's data stream and technique absolutely and should make each one of the reports of the manual structure other than a substantial gathering of the other organization reports. It should be filled in as an electronic application with specific web server and database server. Advance a segment of the associated trades happen in different ranges. Open source programming like TOMCAT, JAVA, MySQL and Linux is used to restrict the cost for the Customer. No extraordinary wander need to manage the instrument.

Technical feasibility

Surveying the particular probability is the trickiest bit of a believability consider. This is in light of the fact that, starting at the present moment, not a lot of point by point layout of the system, making it difficult to get to issues like execution, costs on (by excellence of the kind of development to be passed on) et cetera.

Different issues must be considered while doing a particular examination. Grasp the differing progressions required in the proposed system. Before starting the wander, we should be clear about what are the advances that are to be required for the change of the new system. Check whether the affiliation by and by has the required advancements. Is the required development open with the affiliation?

In case so is the utmost sufficient?

For instance – "Will the present printer have the ability to manage the new reports and structures required for the new system?"

Operational feasibility

Proposed wanders are profitable just if they can be changed into information systems that will meet the affiliations working necessities. Simply communicated, this trial of probability asks with reference to whether the structure will work when it is made and presented. Are there genuine obstacles to Implementation? Here are questions that will help test the operational achievability of a wander.

- Is there sufficient help for the wander from organization from customers? In case the present structure is particularly cherished and used to the extent that individuals won't have the ability to see purposes behind change, there may be resistance.
- Are the present business methodologies qualified to the customer? If they are not, Users may welcome a change that will accomplish a more operational and supportive systems.

Have the customer been locked in with the orchestrating and change of the wander? Early commitment decreases the chances of impenetrability to the structure.

4.3 Arduino app installation:

Get the latest version from the download page. You can choose between the Installer (.exe) and the Zip packages. We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers. With the Zip package you need to install the drivers manually. The Zip file is also useful if you want to create aportable installation.

When the download finishes, proceed with the installation and please allow the driver installation process when you get a warning from the operating system.

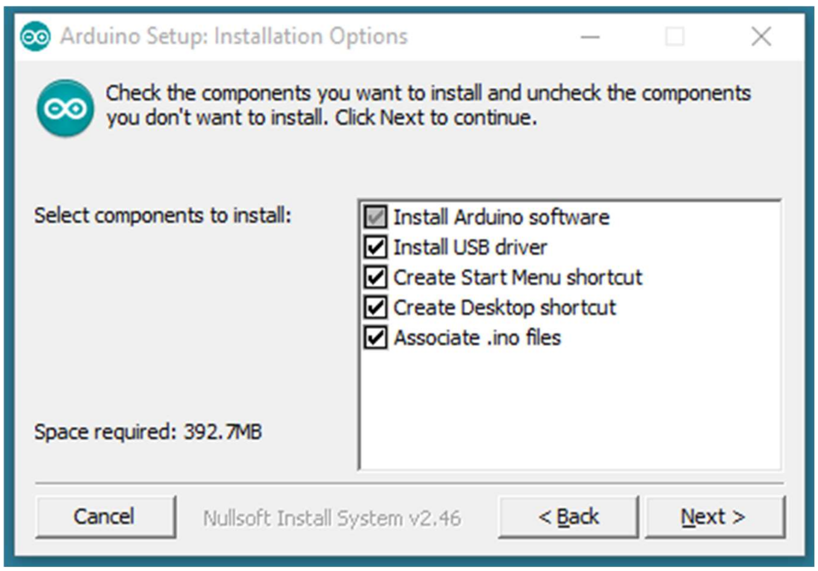


Fig 4.3.1 choose the components to install

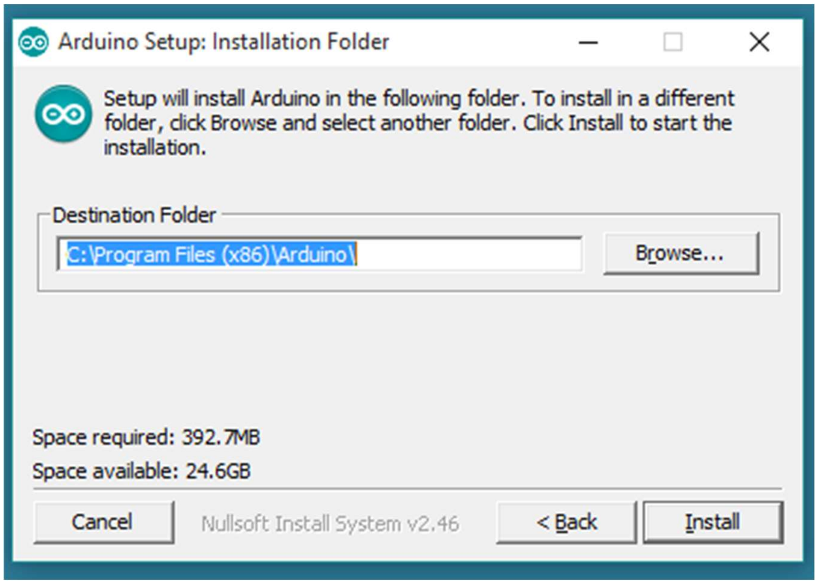


Fig 4.3.2 choose the installation directory

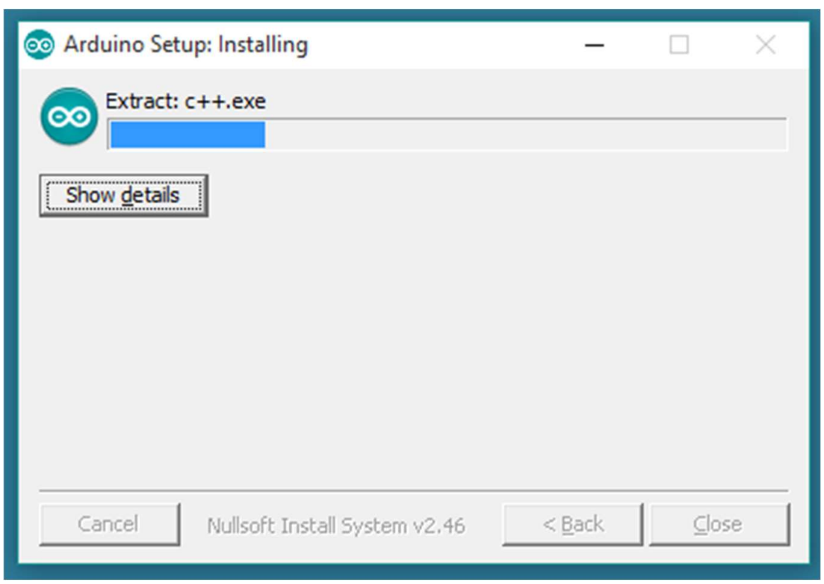


Fig 4.3.3 The process will extract and install all the required files

CHAPTER 5

SYSTEM DESIGN

5.1 System architecture:

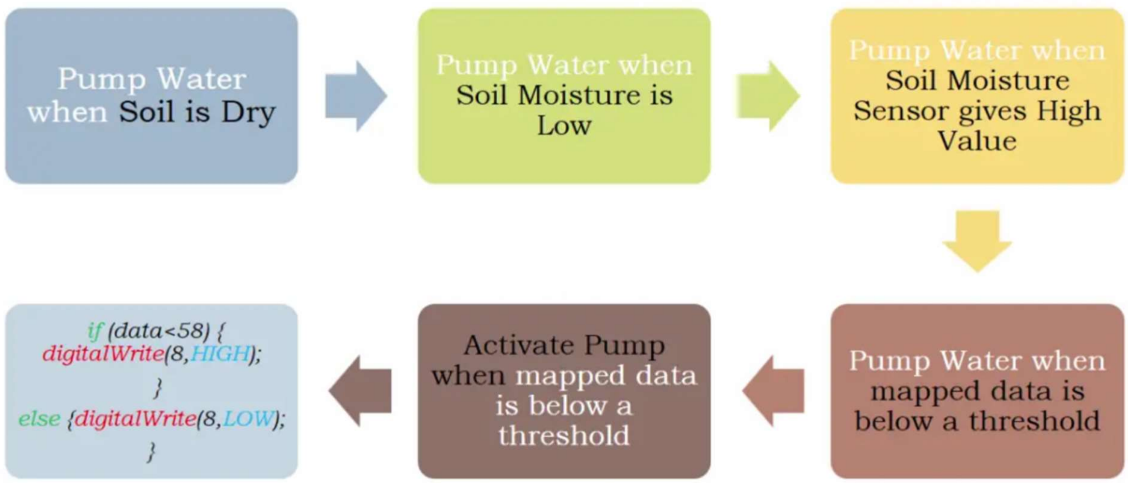


Fig 5.1. System architecture of smart irrigation. Basically this helps us to get an overview of the motor-arduino interaction

5.2 System specifications:

Function	**AT Commands**	**Response**
Working	AT	OK
Restart	AT+RST	OK Ready
Firmware Version	AT+GMR	<AT version info> information about AT version <SDK version info> information about SDK version <compile time> time of the bin was compiled OK

Function	**AT Commands**	**Response**
List Access Point	AT+CWLAP	+CWLAP:<ecn>,<ssid>,<rssi>,<mac>,<ch>,<freq offset> OK
Query Joined Access Point	AT+CWJAP?	+CWJAP:<ssid>,<bssid>,<channel>,<rssi> OK
Join Access Point	AT+CWJAP=”SSID”,”Password”	WIFI CONNECTED WIFI GOT IP OK
Quit Access Point	AT+CWQAP	OK WIFI DISCONNECTED
Get IP Address	AT+CIFSR (Assuming AT+CWMODE=3)	+CIFSR:APIP,<IP address> +CIFSR:APMAC,<mac address> +CIFSR:STAIP,<IP address> +CIFSR:STAMAC,<mac address> OK
Query WiFi Mode	AT+CWMODE?	+CWMODE:<mode>
Set WiFi Mode	AT+CWMODE=<mode> Mode: - 1 = STA (station) 2 = AP (Access Point) 3 = BOTH i.e. STA & AP	OK
Query TCP/UDP Connection	AT+CIPMUX?	+CIPMUX:<mode>
Set TCP/UDP Connection	AT+CIPMUX=<mode> Mode: - 0 = Single Connection 1 = Multiple Connection	OK

Function	AT Commands	Response
TCP/IP Connection status	AT+CIPSTATUS	STATUS:<status> Possible statuses are 2: Got IP 3: Connected 4: Disconnected
Query TCP transmission mode	AT+CIPMODE?	+CIPMODE:<mode>
Set TCP transmission mode	AT+CIPMODE=<mode> Mode: - 0 = Normal mode 1 = Transparent mode	OK
Set up TCP/UDP connection	(CIPMUX=0) AT+CIPSTART = <type>,<addr>,<port> (CIPMUX=1) AT+CIPSTART= <id>,<type>,<addr>, <port> Example (CIPMUX=0): AT+CIPSTART="TCP","192.168.101.110",80	CONNECT OK
Send Data	(CIPMUX=0) AT+CIPSEND=<data length> (CIPMUX=1) AT+CIPSEND=<id>,<data length>	OK > (Note: write your data after > and enter it to send it will return status like.) Recv<data length> bytes SEND OK (after we receive response from server if any for default auto receive mode) (CIPMUX=0): + IPD, <length>: <data> (CIPMUX=1): + IPD, <id>, <length>: <data>

Function	AT Commands	Response
Close TCP/UDP Connection	AT+CIPCLOSE	CLOSED OK

Table 5.2. AT command table of Arduino Uno

5.3 Arduino IDE:

The Arduino is a fantastic single-board microcontroller solution for many DIY projects, and, in this blog, we will look at the Integrated Development Environment, or IDE, that is used to program it.

First, you must download the IDE and install it. Start by visiting Arduino’s software page. The IDE is available for most common operating systems, including Windows, Mac OS X, and Linux, so be sure to download the correct version for your OS. If you are using Windows 7 or older, do not download the Windows app version, as this requires Windows 8.1 or Windows 10.

Once the installer has downloaded, go ahead and install the IDE. Chances are you will want to enable all options on the installer, including any USB drivers and libraries, but do make sure to read the EULA!

The Arduino IDE is incredibly minimalistic, yet it provides a near-complete environment for most Arduino-based projects. The top menu bar has the standard options, including “File” (new, load save, etc.), “Edit” (font, copy, paste, etc.), “Sketch” (for compiling and programming), “Tools” (useful options for testing projects), and “Help”. The middle section of the IDE is a simple text editor that where you can enter the program code. The bottom section of the IDE is dedicated to an output window that is used to see the status of the compilation, how much memory has been used, any errors that were found in the program, and various other useful messages.

Projects made using the Arduino are called sketches, and such sketches are usually written in a cut-down version of C++ (a number of C++ features are not included). Because programming a microcontroller is somewhat different from programming a computer, there are a number of device-specific libraries (e.g., changing pin modes, output data on pins, reading analog values, and timers). This sometimes confuses users who think Arduino is programmed in an “Arduino language.” However, the Arduino is, in fact, programmed in C++. It just uses unique libraries for the device.

There are plenty of other features available to consider on the IDE. But, having used many different types of microcontrollers and having been involved in multiple programming environments, it is shocking how simple the Arduino and its IDE is! In less than two minutes, you can get a simple C++ program uploaded onto the Arduino and have it running.

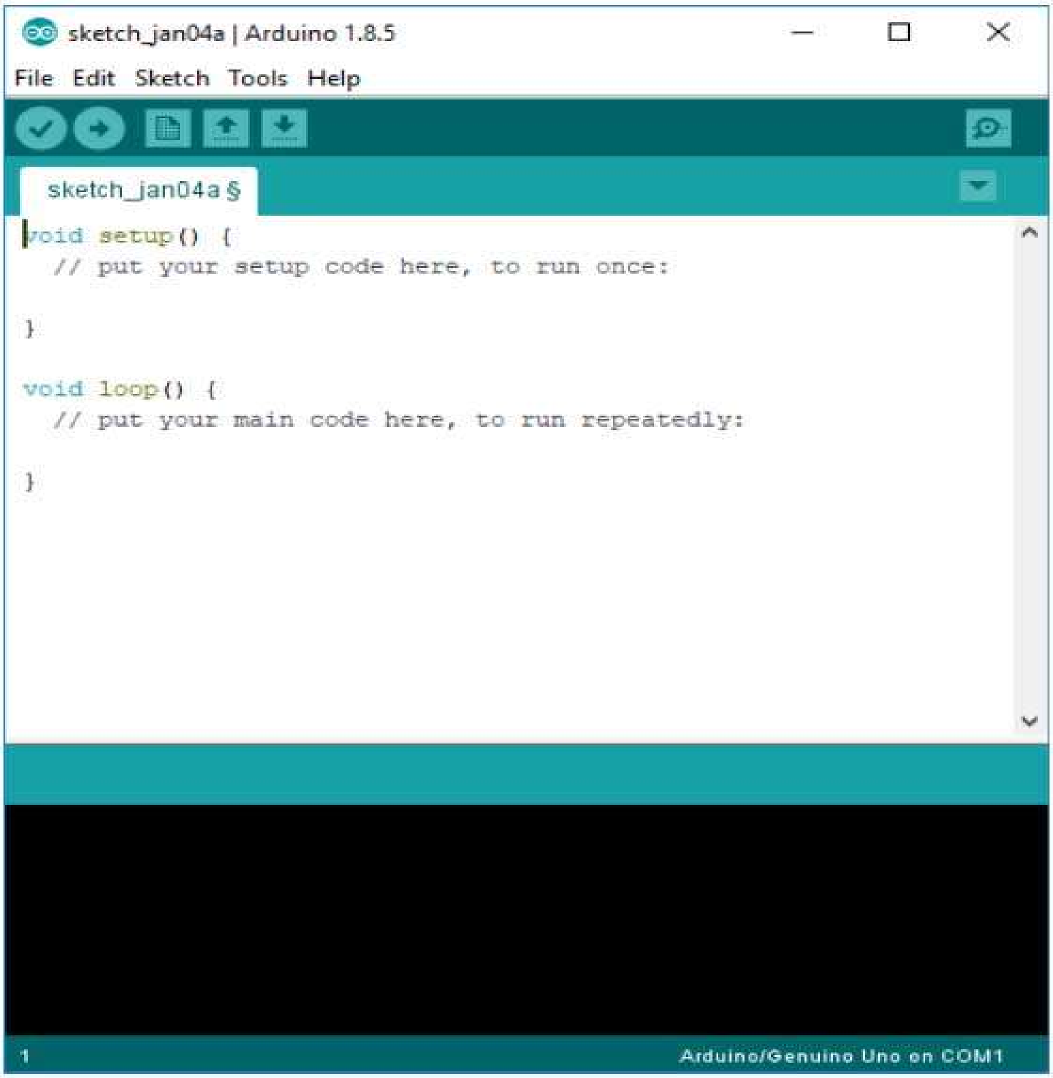


Fig 5.3. Arduino IDE at its default state.

5.4 UML diagrams:

UML (Unified Modelling Language) is a standard vernacular for choosing, envisioning, making, and specifying the collectibles of programming structures. UML is a pictorial vernacular used to make programming blue prints. It is in like way used to exhibit non programming structures similarly like process stream in a gathering unit and so forth.

UML is not a programming vernacular yet rather instruments can be utilized to make code in different tongues utilizing UML graphs. UML has an incite relationship with question composed examination and outline. UML expect a fundamental part in portraying trade viewpoints of a structure.

The structural diagrams represent the static aspect of the system. These static aspects represent those parts of a diagram, which forms the main structure and are therefore stable.

These static parts are represented by classes, interfaces, objects, components, and nodes. The four structural diagrams are −

- Class diagram
- Object diagram
- Component diagram
- Deployment diagram

Any complex system is best understood by making some kind of diagrams or pictures. These diagrams have a better impact on our understanding. If we look around, we will realize that the diagrams are not a new concept but it is used widely in different forms in different industries.

We prepare UML diagrams to understand the system in a better and simple way. A single diagram is not enough to cover all the aspects of the system. UML defines various kinds of diagrams to cover most of the aspects of a system.

5.3.1 Class diagram:

The class graph is the most normally pulled in layout UML. It addresses the static course of action perspective of the structure. It solidifies the strategy of classes, interfaces, joint attempts and their affiliations.

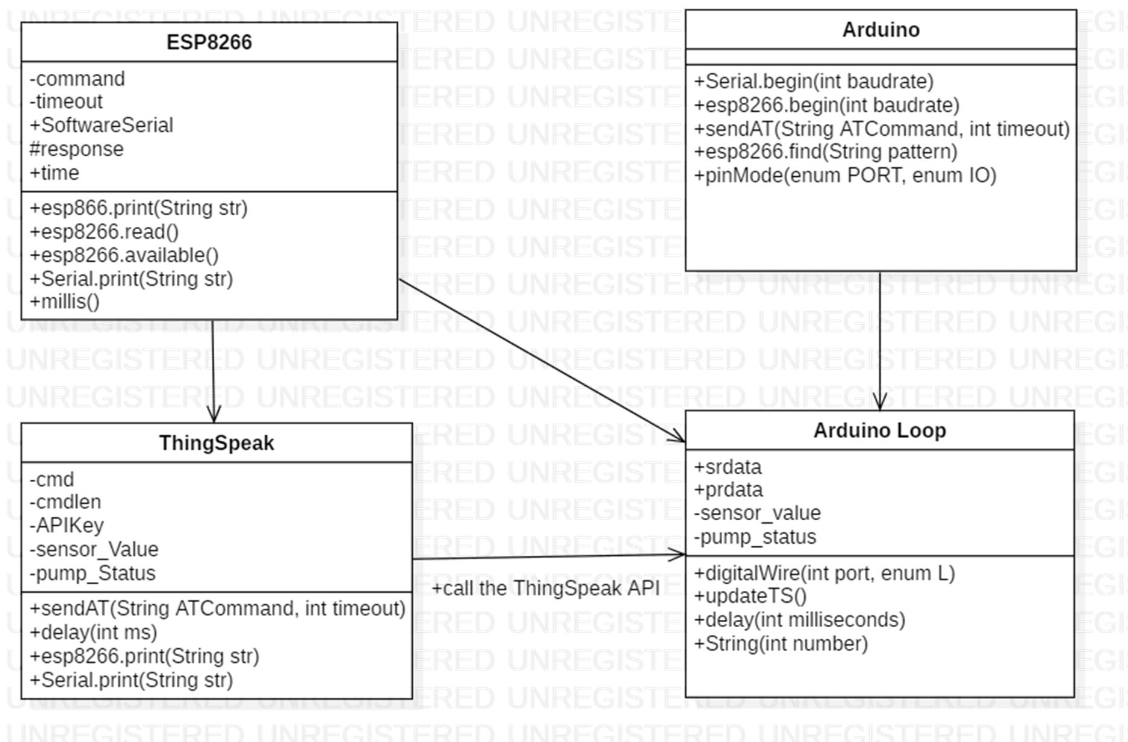


Fig 5.3.1 Class diagram of smart irrigation module interaction.

The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.

UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application, however class diagram is a bit different. It is the most popular UML diagram in the coder community.

5.3.2 Component diagram:

The imperative portion of part format is segment. This diagram demonstrates within parts, connectors and ports that understand the piece. Precisely when section is instantiated, duplicates of inside parts are besides instantiated.

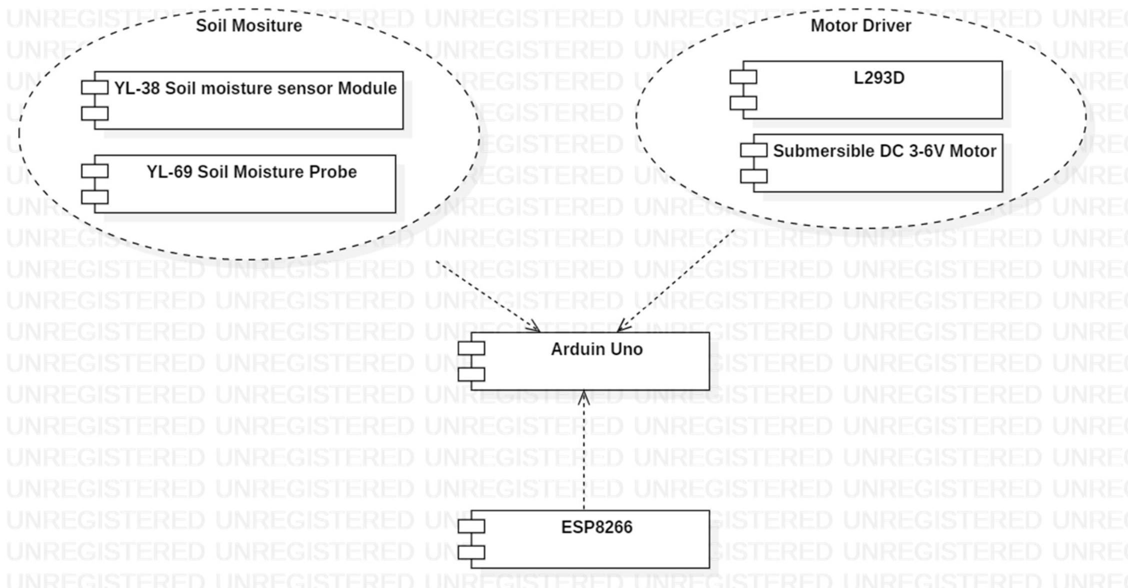


Fig 5.3.2 Component diagram of Hardware interaction with soil.

Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system but it describes the components used to make those functionalities. Thus from that point of view, component diagrams are used to visualize the physical components in a system. These components are libraries, packages, files, etc.

Component diagrams can also be described as a static implementation view of a system. Static implementation represents the organization of the components at a particular moment.

5.3.3 Use case diagram:

To model a system, the most important aspect is to capture the dynamic behaviour. Dynamic behaviour means the behaviour of the system when it is running/operating. Only static behaviour is not sufficient to model a system rather dynamic behaviour is more important than static behaviour. In UML, there are five diagrams available to model the dynamic nature and use case diagram is one of them. Now as we have to discuss that the use case diagram is dynamic in nature, there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors. Use case diagrams consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system.

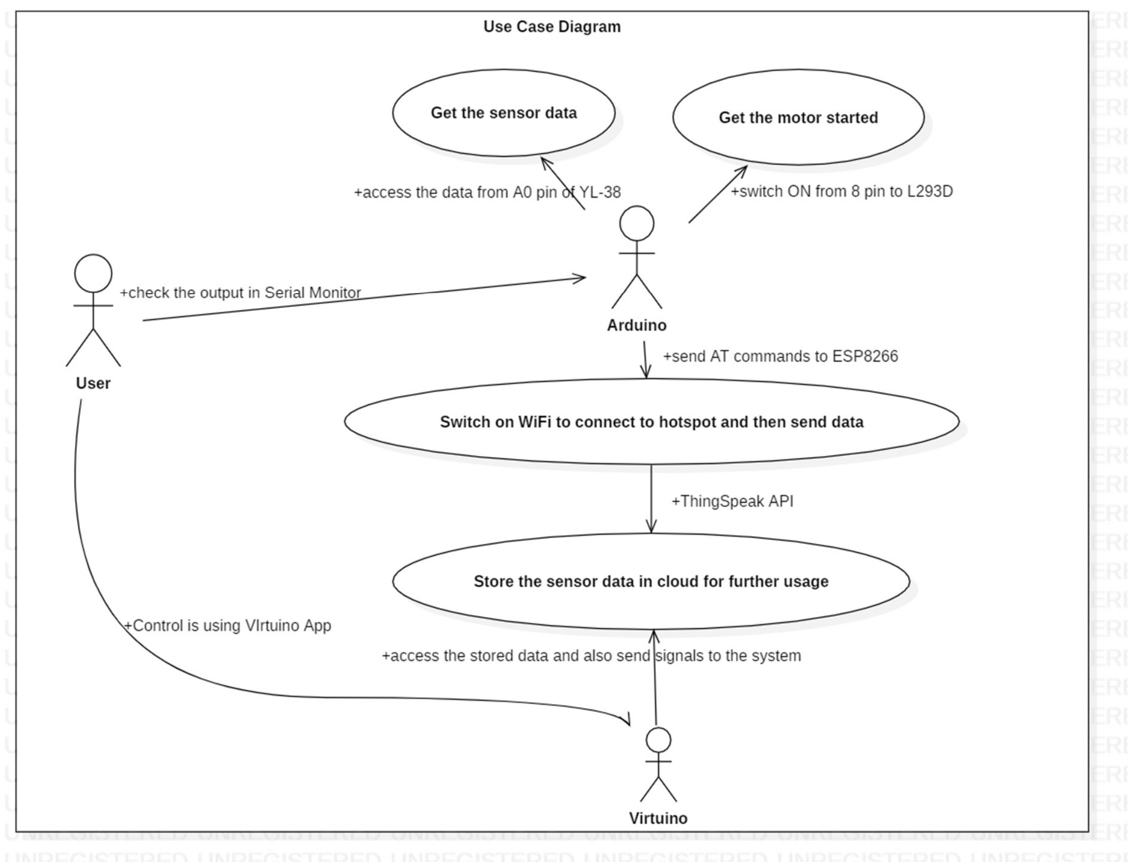


Fig 5.3.3 Use case diagram of hardware-monitor interaction

5.3.4 Activity diagram:

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

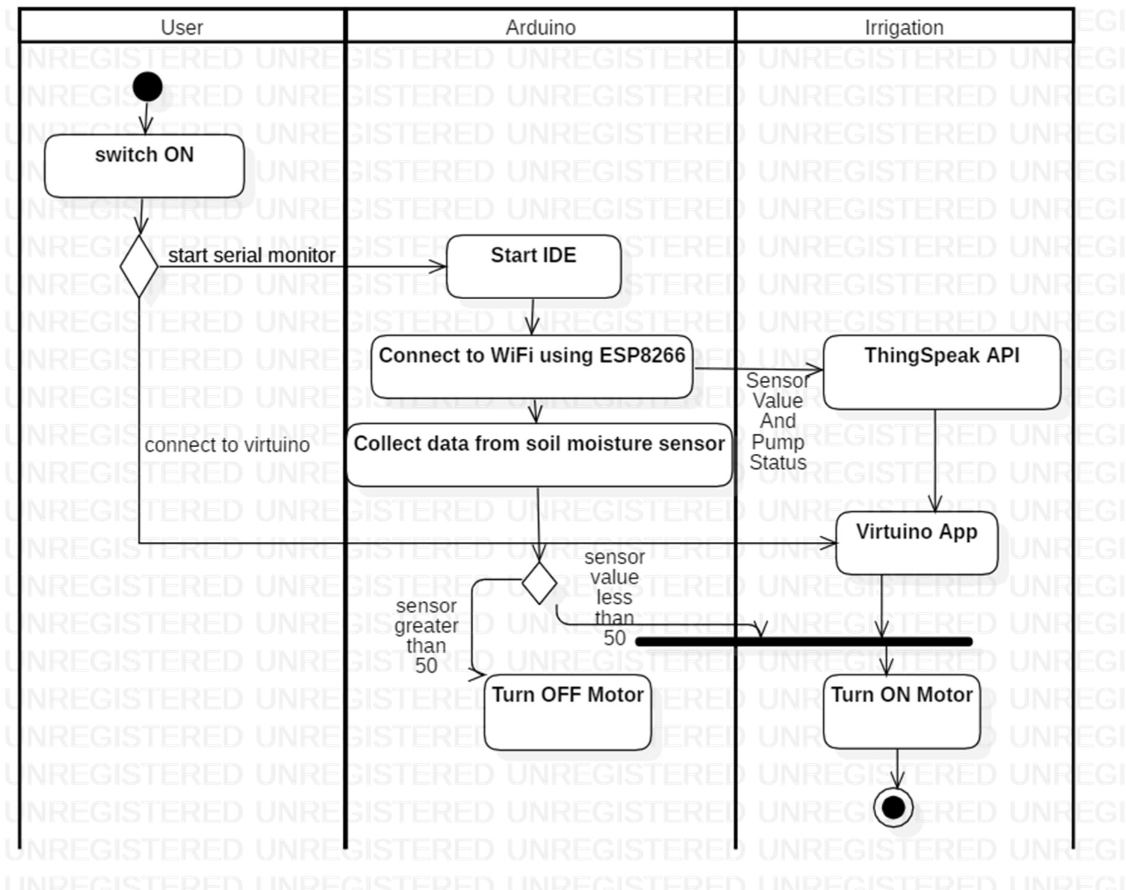


Fig 5.3.4 Activity diagram for monitor-hardware-software connection

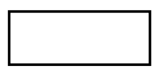
5.3.5 Data flow diagrams:

An information stream design (DFD) is a graphical portrayal of the "stream" of information through a data framework, demonstrating its strategy edges. A DFD is a significant part of the time utilized as a preparatory stroll to make an overview of the framework, which can later be cleared up. DFDs can in like way be utilized for the depiction of information prepare. A DFD indicates what sort of data will be sense of duty regarding and yield from the structure, where the information will begin from and go to, and where the information will be secured. It doesn't demonstrate data about the organizing of process or data about whether strategy will work in game-plan or in parallel.

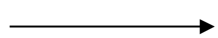
DFD Symbols:

In the DFD, there are four symbols

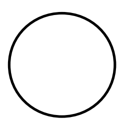
- A square defines a source or destination of system data.



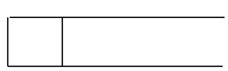
- An arrow identifies data flow. It is the pipeline through which the information flows.



- A circle represents a process that transforms incoming data flow into outgoing data flow.



- An open rectangle is a data store, data at rest or a temporary repository of data.



CHAPTER 6

CODING & IMPLEMENTATION

6.1 Sensor readings.ino:

```
#include<SoftwareSerial.h>

intsrdata;

intprdata;

intpump_status;

SoftwareSerialesp8266(3,4);

#define SSID "adi"

#define PASS "123456678789"


String sendAT(String command,constint timeout){

 String response = "";

esp8266.print(command);

longint time=millis();

while((time+timeout) >millis()){

while(esp8266.available())

    {

char c=esp8266.read();

response+=c;
```

```
    }
  }
Serial.print(response);
return response;
}
void setup() {
  // put your setup codae here, to run once:
Serial.begin(9600);
esp8266.begin(9600);
sendAT("AT+RST\r\n",2000);
sendAT("AT\r\n",1000);
sendAT("AT+CWMODE=1\r\n",1000);
sendAT("AT+CWJAP=\"adi\",\"123456678789\"\r\n",2000);
while(!esp8266.find("OK"))
  {
        //Check until you find OK
  }
sendAT("AT+CIFSR\r\n",1000);
sendAT("AT+CIPMUX=0\r\n",1000);
pinMode(A0,INPUT);
pinMode(8,OUTPUT);
}

void loop() {
```

```
  // put your main code here, to run repeatedly:
srdata=analogRead(A0);
prdata=map(srdata,0,1023,100,0);
Serial.print("Sensor Data");
Serial.print(prdata);
  String sensor_value=String(prdata);
if(prdata<50)
  {
digitalWrite(8,LOW);
pump_status = 100;
  }
else
  {
digitalWrite(8,HIGH);
pump_status = 0;
  }
  String pump = String(pump_status);
updateTS(sensor_value,pump);
delay(2000);
}
voidupdateTS(String T,String P)
{
Serial.print("");
sendAT("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\n",1000);
```

```
delay(2000);
  String cmdlen;
  String cmd = "GET /update?key=1SXLYYHBH920LFW9&field1="+T+"&field2="+P+"\r\n";
cmdlen=cmd.length();
sendAT("AT+CIPSEND="+cmdlen+"\r\n",2000);
esp8266.print(cmd);
Serial.println("");
sendAT("AT+CIPCLOSE\r\n",2000);
Serial.println("");
delay(15000);
}
```

6.2 sensor introduction sketch:

```
intsrdata;

intprdata;

void setup() {

  // put your setup codae here, to run once:

Serial.begin(9600);

pinMode(A0,INPUT);

}


void loop() {

  // put your main code here, to run repeatedly:

srdata=analogRead(A0);

prdata=map(srdata,0,1023,100,0);

Serial.print("Sensor Data");

Serial.print(prdata);

delay(2000);

}
```

CHAPTER 7

TESTING

7.1 INTRODUCTION TO TESTING

Testing is a procedure, which uncovers blunders in the program. Programming testing is a basic component of programming quality affirmation and speaks to a definitive audit of determination, outline and coding. The expanding perceivability of programming as a framework component and chaperon costs related with a product disappointment are propelling variables for we arranged, through testing. Testing is the way toward executing a program with the plan of finding a mistake. The plan of tests for programming and other built items can be as trying as the underlying outline of the item itself It is the significant quality measure utilized amid programming improvement. Amid testing, the program is executed with an arrangement of experiments and the yield of the program for the experiments is assessed to decide whether the program is executing as it is relied upon to perform.

7.2 TESTING STRATEGIES

A technique for programming testing coordinates the outline of programming experiments into an all around arranged arrangement of steps that outcome in fruitful improvement of the product. The procedure gives a guide that portrays the means to be taken, when, and how much exertion, time, and assets will be required. The procedure joins test arranging, experiment configuration, test execution, and test outcome gathering and assessment. The procedure gives direction to the specialist and an arrangement of points of reference for the chief. Due to time weights, advance must be quantifiable and issues must surface as ahead of schedule as would be prudent.

Unit Testing

Unit Testing is done on singular modules as they are finished and turned out to be executable. It is restricted just to the planner's prerequisites. It centers testing around the capacity or programming module. It Concentrates on the interior preparing rationale and information structures. It is rearranged when a module is composed with high union

- Reduces the quantity of experiments
- Allows mistakes to be all the more effectively anticipated and revealed

Black Box Testing

It is otherwise called Functional testing. A product testing strategy whereby the inward workings of the thing being tried are not known by the analyzer. For instance, in a discovery test on a product outline the analyzer just knows the information sources and what the normal results ought to be and not how the program touches base at those yields. The analyzer does not ever inspect the programming code and does not require any further learning of the program other than its determinations. In this system some experiments are produced as information conditions that completely execute every single practical prerequisite for the program. This testing has been utilizations to discover mistakes in the accompanying classifications:

- Incorrect or missing capacities
- Interface blunders
- Errors in information structure or outside database get to
- Performance blunders
- Initialization and end blunders.

In this testing just the yield is checked for rightness.

White Box testing

It is otherwise called Glass box, Structural, Clear box and Open box testing . A product testing procedure whereby express learning of the inner workings of the thing being tried are utilized to choose the test information. Not at all like discovery testing, white box testing utilizes particular learning of programming code to inspect yields. The test is precise just if the analyzer comprehends what the program should do. He or she would then be able to check whether the program veers from its expected objective. White box testing does not represent blunders caused by oversight, and all obvious code should likewise be discernable. For an entire programming examination, both white box and discovery tests are required.

In this the experiments are produced on the rationale of every module by drawing stream diagrams of that module and sensible choices are tried on every one of the cases. It has been utilizations to produce the experiments in the accompanying cases:

- Guarantee that every single free way have been Executed.
- Execute every single intelligent choice on their actual and false Sides.

Integration Testing

Coordination testing guarantees that product and subsystems cooperate an entirety. It tests the interface of the considerable number of modules to ensure that the modules carry on legitimately when coordinated together. It is characterized as a deliberate procedure for developing the product engineering. In the meantime reconciliation is happening, lead tests to reveal blunders related with interfaces. Its Objective is to take unit tried modules and assemble a program structure in view of the recommended outline

Two Approaches of Integration Testing

- Non-incremental Integration Testing
- Incremental Integration Testing

System Testing

Framework testing includes in-house testing of the whole framework before conveyance to the client. Its point is to fulfill the client the framework meets all necessities of the customer's determinations. This testing assesses working of framework from client perspective, with the assistance of particular report. It doesn't require any inward learning of framework like plan or structure of code.

It contains utilitarian and non-useful zones of utilization/item. Framework Testing is known as a super arrangement of a wide range of testing as all the significant sorts of testing are shrouded in it. In spite of the fact that attention on sorts of testing may differ on the premise of item, association procedures, course of events and necessities. Framework Testing is the start of genuine testing where you test an item all in all and not a module/highlight.

Acceptance Testing

Acknowledgment testing, a testing method performed to decide if the product framework has met the prerequisite particulars. The principle motivation behind this test is to assess the framework's consistence with the business necessities and check in the event that it is has met the required criteria for conveyance to end clients. It is a pre-conveyance testing in which whole framework is tried at customer's site on genuine information to discover blunders. The acknowledgment test bodies of evidence are executed against the test information or utilizing an acknowledgment test content and afterward the outcomes are contrasted and the normal ones.

The acknowledgment test exercises are completed in stages. Right off the bat, the essential tests are executed, and if the test outcomes are palatable then the execution of more intricate situations are done.

7.3 TEST APPROACH

A Test approach is the test system usage of a venture, characterizes how testing would be done. The decision of test methodologies or test technique is a standout amongst the most intense factor in the achievement of the test exertion and the precision of the test designs and gauges.

Testing should be possible in two ways

- Bottom up approach
- Top down approach

Bottom up Approach

Testing can be performed beginning from littlest and most reduced level modules and continuing each one in turn. In this approach testing is directed from sub module to primary module, if the fundamental module is not built up a transitory program called DRIVERS is utilized to recreate the principle module. At the point when base level modules are tried consideration swings to those on the following level that utilization the lower level ones they are tried exclusively and afterward connected with the already inspected bring down level modules

Top down Approach

In this approach testing is directed from fundamental module to sub module. in the event that the sub module is not built up an impermanent program called STUB is utilized for mimic the sub module. This sort of testing begins from upper level modules. Since the nitty gritty exercises more often than not performed in the lower level schedules are not given stubs are composed. A stub is a module shell called by upper level module and that when achieved legitimately will restore a message to the calling module demonstrating that appropriate association happened.

7.4 VALIDATION

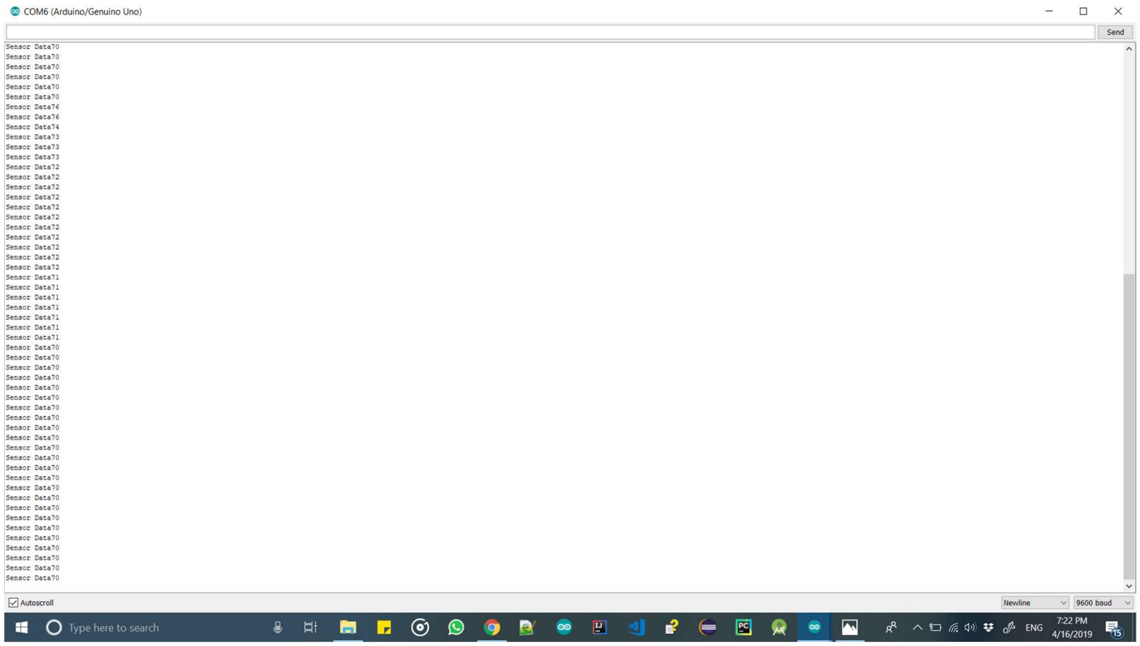
The way toward assessing programming amid the improvement procedure or toward the finish of the advancement procedure to decide if it fulfills determined business prerequisites. Approval Testing guarantees that the item really addresses the customer's issues. It can likewise be characterized as to exhibit that the item satisfies its proposed utilize when sent on proper condition.

The framework has been tried and actualized effectively and along these lines guaranteed that every one of the prerequisites as recorded in the product necessities determination are totally satisfied.

7.5 TEST CASES

Experiments include an arrangement of steps, conditions and sources of info that can be utilized while performing testing undertakings. The principle expectation of this action is to guarantee whether a product passes or bombs as far as usefulness and different perspectives. The way toward creating experiments can likewise help discover issues in the prerequisites or plan of an application. Experiment goes about as the beginning stage for the test execution, and in the wake of applying an arrangement of information esteems, the application has a conclusive result and leaves the framework at some end point.

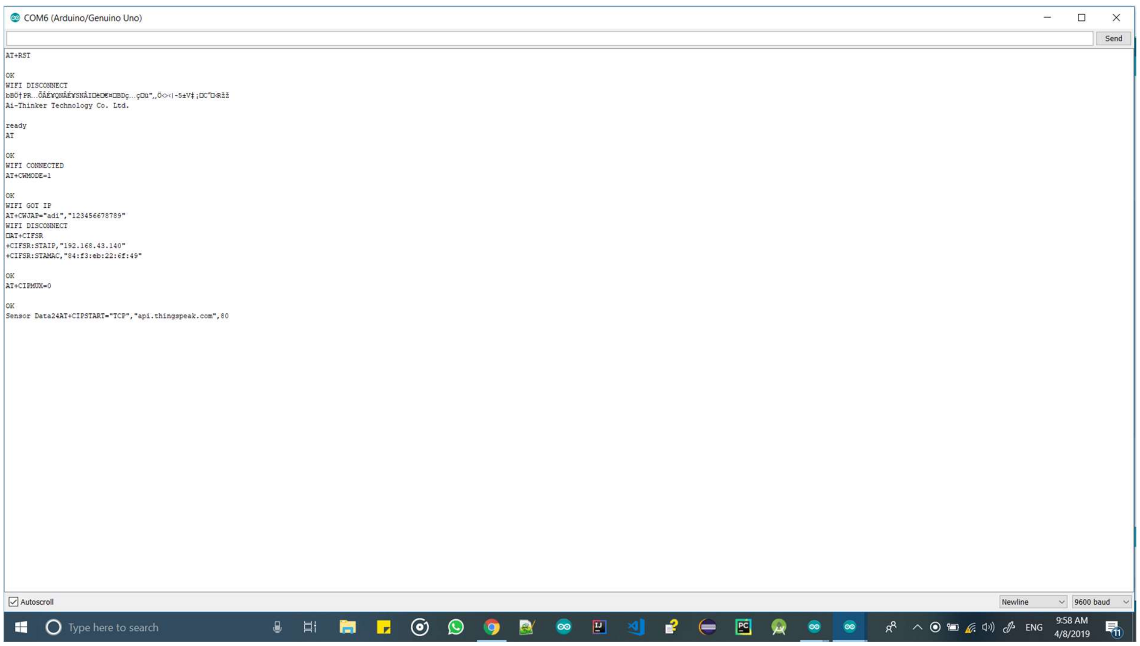
Basic sensor data tested when there is a Wi-fi detection ambiguity in Arduino Uno:



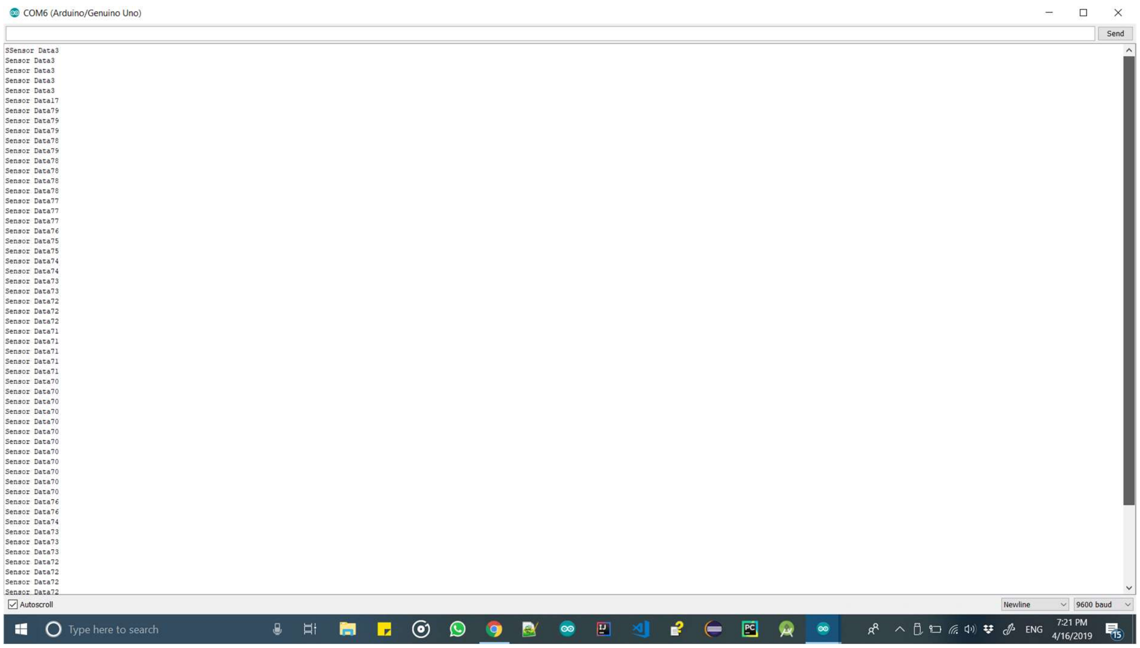
CHAPTER 8

SCREENSHOTS AND OUTPUTS

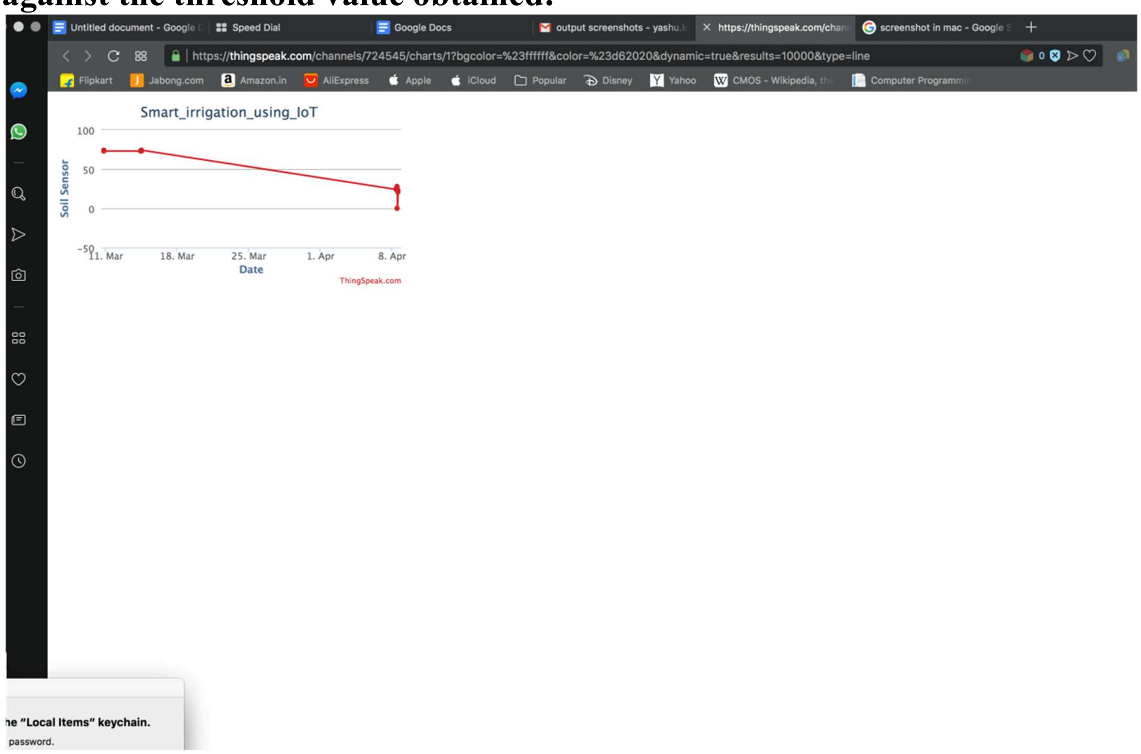
8.1 Wi-Fi interaction explaining the AT-command specification with the smart irrigation arduino server:



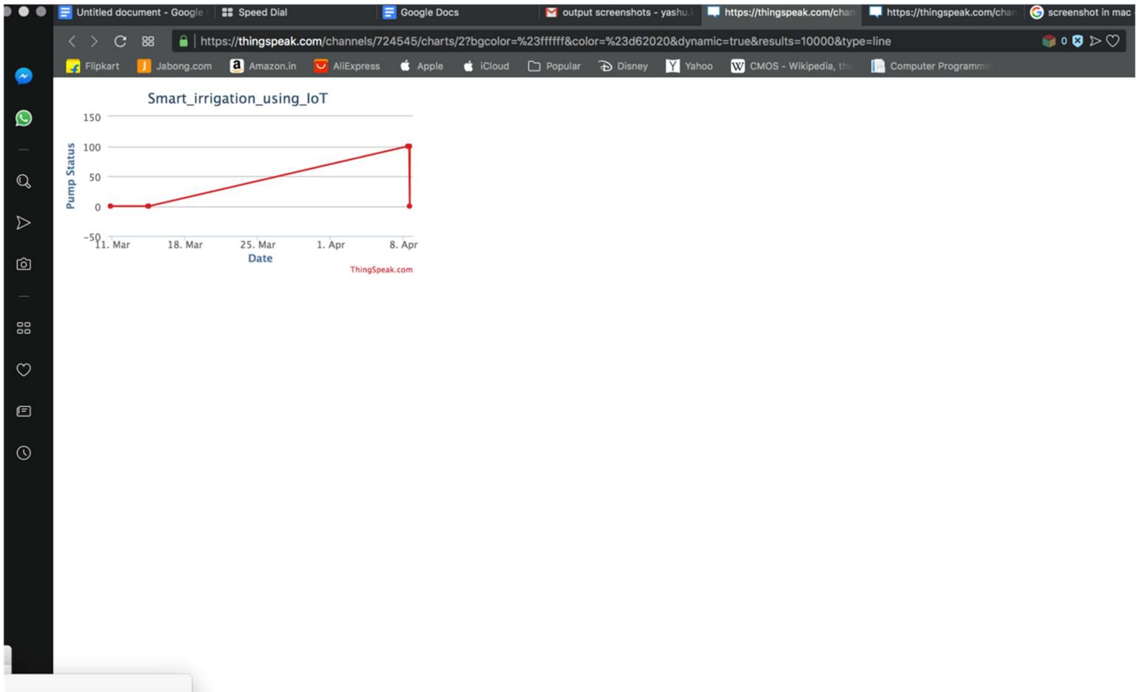
8.2 Sensor data generated from the soil moisture sensor which is used to pump water by holding a threshold point:



8.3 Thingspeak IOT analytics platform used for soil moisture detection against the threshold value obtained:



8.4 Thingspeak IOT analytics platform used to explain the pumping of water from the motor by observing the threshold of moisture:



CHAPTER 9

CONCLUSION

With the help of Internet Of Technology, each and every component of our surrounding is getting connected to Internet. And with the excessive usage of sensor, we're able scrape out the in-depth data collected from theses sensors and hence giving us a complete overview of our surrounding. Smart Irrigation System can be one such model which can be very helpful for the nature as it can be used efficiently to irrigate our plants or tree or crops when the soil moisture is low and once it gets watered, the pump gets stopped. Hence, using this system, we can drastically reduce the water wastage by a lot of percentage and also the energy usage is very less. This project can be further implemented into a large scale, mass system to irrigate an entire crop based upon the sensor values of that particular region. Also, as the world is moving forward towards technology, smartphones are becoming more common making it to be accessed by any level of the society. Therefore, the smart irrigation can also be operated using app to operate whether to switch on the pump or switch it off the pump based upon the user's necessity.

CHAPTER 10

REFERENCES

1. Internet of things getting started with objective reference.
2. System interaction of the Arduino-uno design board reactions.
3. https://www.analyticsvidhya.com/blog/2016/06/11-read-books-summer-internet-iot/
4. https://thingspeak.com/pages/how_to#analyze
5. A book on theoretical analysis of soil moisture detection with the help of using various versions.
6. https://www.techopedia.com/definition/28247/internet-of-things-iot
7. Building in internet of things.