PROJECT REPORT

ON

USE OF DIGITAL TECHNOLOGY FOR MICRO IRRIGATION SYSTEM TO IMPROVE WATER EFFICIENCY OF IRRIGATION SECTOR

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Bachelor of Engineering

In

Electronics and Telecommunication

By

Shivam Govindrao Mahalankar (Exam No - B190353063)
Shivam Kishor Patange (Exam No - B190353076)
Rushikesh Madhav Sagare (Exam No - B190353089)

Under the Guidance of

Mr. S. R. Trankatwar

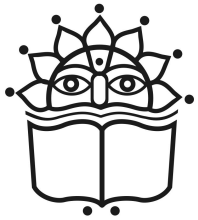
Department of Electronics and Telecommunication Engineering,

VIDYA PRATISHTHAN'S KAMALNAYAN BAJAJ INSTITUTE OF ENGINEERING & TECHNOLOGY, BARAMATI

2022-23

Affiliated to

Savitribai Phule Pune University



Department of Electronics and Telecommunication Engineering,

Vidya Pratishthan's Kamalnayan Bajaj Institute of Engineering & Technology, Baramati

CERTIFICATE

This is to Certify that the Project Report Entitled

"USE OF DIGITAL TECHNOLOGY FOR MICRO IRRIGATION SYSTEM TO IMPROVE WATER EFFICIENCY OF IRRIGATION SECTOR"

Submitted By

Shivam Govindrao Mahalankar (Exam No - B190353063)

Shivam Kishor Patange (Exam No - B190353076)

Rushikesh Madhav Sagare (Exam No - B190353089)

is a bonafied work carried out satisfactorily by them under supervision and guidance and it is submitted towards the partial fulfilment of the requirements of Savitribai Phule Pune University, Pune for the award of degree, Bachelor of Engineering (Electronics and Telecommunication) during the academic year 2022-23.

Place: Baramati
Date:

Mr. S. R. Trankatwar
Guide

Dr. B. H. Patil
Head of the Department

Dr. R. S. Bichkar
Principal
VPKBIET, Baramati.

ACKNOWLEDGMENT

We would like to thank **Mr. S. R. Trankatwar**, our guide who has helped us through this whole journey, our project coordinator **Mr. S. D. Biradar** for guiding and being with us throughout the project development, **Dr. B. H. Patil**, Head of Department, Electronics and Telecommunication Engineering and our honorable Principal **Dr. R. S. Bichkar** for motivation and their cooperation in completing our project on the topic "Use Of Digital Technology For Micro Irrigation System To Improve Water Efficiency of Irrigation Sector". We also want to express our gratitude towards our friends for being with us and assisting us in very minor to major help in the development phase of the project. We have been very thankful to our family and friends for keeping our moral high through their verbal support both family and friends have played an important role in this process. We would like to express our deepest gratitude to the entire engineering community for their continuous efforts in pushing the boundaries of knowledge and innovation. The collective knowledge and inspiration provided by researchers, scientists and engineers around the world have paved the for projects like ours and we are honored to be the part of this community. This project has been an incredible learning experience and we are grateful to everyone who as directly and indirectly contributed to the success of this project. We would also like to thank the AWS for providing such a great cloud service for such cheap rates, it has opened ways for students like us to learn and step into the world of the cloud computing.

Lastly we would like to express our gratitude to all of our teachers for their help and guidance. We would not have been able to complete this project without their help and cooperation. And we are very thankful for this opportunity that we got.

ABSTRACT

India is an agricultural country and the majority of its population depends on it. Owing to global warming and the gradual depletion of natural resources, water resources should be used efficiently and precisely for farming. Unsupervised and traditional methods of watering crops not only result in excessive watering of the crop but also degradation of the soil, resulting in loss of nutrients; however, a large amount of water will be wasted. On the other hand, insufficient watering of crops results in malnutrition. For watering a large farming area, we also need a huge amount of electric power, which will be used to run watering motors because of the unsupervised and continuous use of a large amount of electric power; the watering motor is also at a chance of being burned out.

This project proposes to automate the tedious process by proposing a Internet Of Things based system for automatic smart drip irrigation and to decide whether to turn on or off the watering of the crop. Taking into consideration the weather, soil, and crop parameters, the quantity of water that should flow accordingly through drip irrigation can be predicted with the help of sensors. This can control the moisture content of the soil in the cultivation field. Not only will it help the farmer to use water wisely in the future, but also the water supply to crops will be automated based on the conditions that are a win-win situation for both the farmer and the environment, leading to a good crop yield in a consistent manner. The Idea creates a solution that will be able to collect, store, and process environmental values to make decisions depending upon them. Initialize by sensing the moisture values using sensors. This will send those values to the cloud platform via IoT technology, which will be processed there and finally decides whether to turn on or off the motor. This provides an efficient and long-term solution to this problem.

Contents

List of Figures

List of Tables

List of Abbreviations

	Abbreviation	Meaning
1.	IoT	Internet Of Things
2.	ML	Machine Learning
3.	CC	Cloud Computing
4.	MQTT	MQ Telemetry Transport
5.	NO	Normally Open
6.	NC	Normally Close
7.	AWS	Amazon Web Services
8.	SSH	Secure Shell
9.	SCP	Secure Copy Protocol
10.	HTTP	Hypertext Transfer Protocol
11.	PLC	Programmable Logic Controllers
12.	DIY	Do It Yourself
13.	IC	Integrated Circuit
14.	LED	Light Emitting Diode
15.	IO	Input Output
16.	ADC	Analog to Digital Converter
17.	amps	Amperes
18.	IPV4	Internet Protocol Version 4

1 INTRODUCTION

Traditional farming and irrigation practices in India have deep roots and vary across different regions due to diverse climatic conditions and cultural practices. Traditional farmers in India have long relied on canal systems and tanks to distribute and store water for irrigation. Canals, often constructed manually, divert water from rivers or reservoirs to agricultural fields, ensuring a steady supply of water. Tanks, which are small to large artificial reservoirs, collect rainwater during the monsoon and provide irrigation water during dry periods. Digging wells is a common traditional method of accessing groundwater for irrigation. Traditional wells, usually hand-dug, tap into the water table and provide water for nearby fields. In recent times, tube wells have become popular, utilizing mechanized drilling to access deeper groundwater sources for irrigation. While traditional irrigation methods still persist in many parts of India, modern irrigation techniques such as sprinkler and drip irrigation have gained popularity. These modern methods offer more precise and efficient water usage, reducing water wastage and increasing crop productivity. However, traditional farming and irrigation practices continue to play a significant role, especially in rural areas, where farmers rely on traditional knowledge and available resources to sustain their agricultural practices.

Traditionally, drip irrigation farming has been performed manually, which requires the farmer to be physically present in the field. In agriculture, the major problem faced by Indian farmers is water scarcity, which is becoming a critical issue. In addition, there are quite a few areas in our country that also face drought. To improve the usage of water, an accurate amount of water required by a particular crop must be provided only at the time it needs.

This project proposes a “Automated Drip Irrigation System using ML, IoT and Cloud Computing”. Our Sensors will detect the moisture values from the soil and send them to the cloud platform using a NodeMCU ESP8266 microcontroller (which will be responsible for all the communication and controlling actions) unit and from the cloud where we will run our Machine Learning algorithms to train our model and send the instructions back to the microcontroller and switching unit, which will be responsible for the switching of the watering motor as per the soil condition.

1.1 Motivation

India is an agricultural country, and the majority of its population depends on it. Due to global warming and gradual depletion of natural resources, use of water resources should be done in an efficient and precise manner for farming. Unsupervised and tradition way of watering crop not only results in excessive watering of the crop but also degradation of the soil resulting in loosing the nutrients in it nonetheless a huge of amount of water will get wasted. On the other hand due insufficient watering crop will result in malnutrition of the crop. For watering a huge farming area we also need a huge amount of electric power which will used to run watering motors and due to unsupervised and continuous use a large amount of electric power is also consumed nonetheless the watering motor is also at a chance of getting burned out.

Micro-irrigation systems are designed to deliver water directly to the roots of plants using low-pressure, low-volume irrigation methods. These systems are highly efficient, and have been shown to significantly reduce water use in agricultural and landscaping applications. The motivation for implementing a micro-irrigation system is to improve water use efficiency and reduce water waste. Traditional irrigation systems, such as flood irrigation and sprinkler systems, can result in significant water loss owing to evaporation, runoff, and wind drift. Micro-irrigation systems can reduce water loss by up to 70%, resulting in significant water savings and increased crop yields. Additionally, micro-irrigation systems can help reduce soil erosion and nutrient leaching as water is delivered directly to the root zone of plants. This can improve soil quality and reduce the need for fertilizers, pesticides, and herbicides. Finally, micro-irrigation systems are typically easy to install and maintain, making them cost-effective solutions for both small- and large-scale irrigation applications. This makes them particularly attractive to farmers and landscapers who are looking to improve their water use efficiency while minimizing their environmental impact and reducing their operational costs.

Despite the many advantages of micro-irrigation systems, there is still room for improvement in terms of automation and smartness. The micro-irrigation system is efficient but lacks in terms of deciding the right amount of water or when to start and stop watering, making it completely manual, labor-based, and unsupervised. Due to the unsupervised use of the water in the irrigation, not only do the crops suffer from excess watering, but also the water source undergoes depletion; however, water is wasted, which may lead to the scarcity of water after a period of time. We need a solution that can not only monitor and supervise

the soil contents, but also suggest when to stop and start watering.

We are proposing a system which will be motoring the soil contents such as moisture levels but also will be able to control the motor switching upon such instructions. In this solution, we employed technologies such as IoT, Cloud Computing and Machine Learning to achieve the optimum performance of our project.

1.2 Objective

The primary objective of an automatic smart drip irrigation system is to conserve water. By delivering water directly to the roots of plants in a slow and controlled manner, the system minimizes water loss due to evaporation, runoff, and deep percolation. This efficient water delivery helps to reduce overall water consumption in agriculture. The system aims to provide precise and targeted irrigation to individual plants or specific areas of the field. By delivering water directly to the root zone of plants, it ensures that water is utilized effectively, minimizing water wastage. This precision irrigation approach also helps in avoiding water stress or over-irrigation, which can be detrimental to plant health. The automatic smart drip irrigation system aims to optimize the use of water and energy resources. By minimizing water wastage and reducing the energy required for pumping and distribution, it promotes efficient resource utilization. This leads to cost savings for farmers and contributes to sustainable agriculture practices. Another objective of the system is to enable remote monitoring and control of irrigation operations. By integrating sensors, controllers, and communication technologies, farmers can monitor soil moisture levels, weather conditions, and system performance remotely. This allows for timely adjustments in irrigation schedules, ensuring optimal water and resource management. The system collects data on soil moisture, weather conditions, and crop water requirements. This data can be analyzed to gain insights into crop water needs, irrigation scheduling, and overall system performance. By utilizing data-driven decision-making, farmers can make informed choices about irrigation practices, leading to more efficient water management and improved crop outcomes.

To develop a hardware system which will be able to read moisture values from the soil in real time, send them to the cloud platform using IoT which will run Decision Tree ML model. The ML model was trained using the moisture values provided and achieved at least 95 percent of accuracy in predicting the soil condition from the provided values. The main objective of this project is to turn the watering motor on and off in accordance with the input provided in real-time. The objective of this project is to provide a web-based control panel.

1.3 Block Schematic of Proposed System

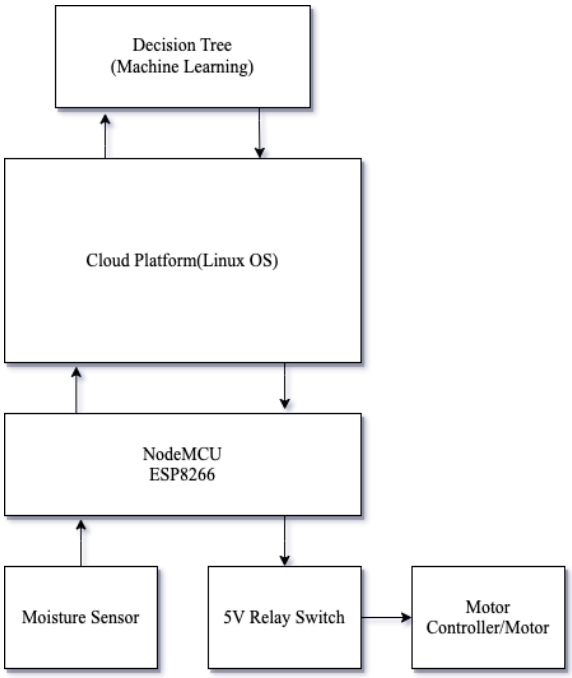


Figure 1: Functioning and Data Flow of the Model

Starting with a moisture sensor at the bottom of the block diagram, which will be placed in the soil for collecting moisture-related data, which will be in the form of numbers ranging from 1024 to 0. 1024 represents the soil being totally dry, while 0 represents complete current flow and negligibly low resistance between the terminals of the moisture sensor, representing a soil being very wet. Then the NodeMCU ESP8266 will collected those sensed moisture values and convert range them from 0 to 1(0 being completely dry(1024) and 1 being completely wet(0)) using a map function as,

```
float scaled_value = map(sensed_data, min_value, max_value, min_scale, max_scale);
```

These values were uploaded to the cloud platform using the MQQT service running on the same Ubuntu Linux cloud platform. These values are then fed to the Decision Tree ML algorithm to determine whether to turn the watering motor on or off. Upon successfully obtaining the decision, it is sent back to the NodeMCU ESP8266 microcontroller, which will use a 5V relay switch to turn on or off the motor depending upon the input.

2 LITERATURE REVIEW

When it comes to agriculture, the majority of farming uses traditional methodologies; however, in modern times, the world is moving toward newer technologies, including India. This revolution in agriculture was accompanied by the introduction of micro-irrigation and drip systems. This revolution was later contributed by many individuals and organizations. In 1977-78 J. Doorenbos and W. Pruitt published guidelines for predicting crop water requirements in the Food and Agriculture Organization (UN), Rome. Later in 1986-87 Brouwer, C., Heibloem published Irrigation Water Management, Training manual which was focused on effective ways of water management in the Irrigation field. These Two studies have helped many people and researchers to understand effective water management in the irrigation sector.

Later in 2016-17 G. Kavianand and S. Lalitha, published a research on Smart Irrigation system for sustainable agriculture in IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR), Chennai. This is useful for understanding the automation in agriculture. But it was not enough as the world was moving towards the Internet of Things era, this technology had a huge scope in the agriculture field as well So in 2018-19 D. Mishra, A. Khan, R. Tiwari and S. Upadhyay published research on the Automated Irrigation System-IoT-based Approach at the 3rd International Conference on Internet of Things Smart Innovation and Usages (IoT-SIU), Bhimtal, which effectively gathered agriculture and the IOT under the same umbrella and demonstrated how it can be done, which was a major leap in both the agriculture and IOT domains.

But then in 2021 Shilpa Chandra, Samiksha Bhilare, Mugdha Asgekar and Ramya R. B. Published a research on Crop water requirement prediction in automated drip irrigation system using ML and IOT in IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR), Chennai which used a ML based approach which was not just a emerging technology but also way more effective, adaptable and reliable alternative over hard coded if-else conditions, this research was one of the major motivation for this project.

3 SYSTEM DEVELOPMENT

3.1 Hardware Components

Beginning with hardware, this project is operational in the soil. Hardware has been an integral part of the project, from collecting data to performing received operations in switching circuits. This project consists of hardware components such as a NodeMCU ESP8266 microcontroller, FC-28 Soil Hygrometer (Moisture) Sensor, Jumper wires, Breadboard, 5V relay switch, and 5V-2A power supply.

3.1.1 NodeMCU ESP8266 Microcontroller

A NodeMCU is a family of low-cost microcontrollers with built-in WiFi connectivity used for communication in both local and public networks. The WiFi on the NodeMCU is used to connect the microcontroller with the Internet and implement an IoT application. Con-

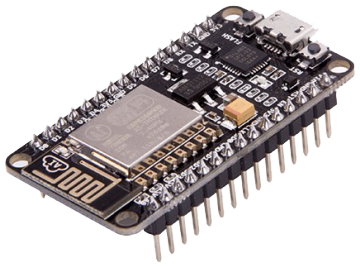


Figure 2: NodeMCU ESP8266 Microcontroller

sidering its affordability, NodeMCU has been used in many IOT applications, such as home automation, and NodeMCU ESP8266 can be used to control and switch home appliances, such as fans, lights, thermostats, and security systems.. With its Wi-Fi connectivity, it can connect with other devices as well, making it a prime choice for smart home setups. Weather station: By interfacing environmental sensors such as temperature, humidity, pressure, and wind speed, we can collect information about the weather of a certain place, and with its WiFi connectivity, we can public that information on the remote server or webpage as well. Internet of Things (IoT) Projects, Nodemcu is widely used in IoT projects due to its WiFi capabilities, Low power consumption, smaller size, etc. It can also be used to monitor and control devices to collect data remotely and communicate with cloud services. Environmental monitoring and NodeMCU can be integrated with sensors to monitor environmental

parameters, such as temperature, wind speed, noise levels, and pollution. The collected data can be used for analysis and visualization. Smart Agriculture, NodeMCU can be used in smart irrigation for monitoring wetness of the soil, implementing automation, optimizing crop growth, reducing water usage, etc., Security Systems, NodeMCU can also be used in security systems for protecting homes by interfacing various sensors like motion sensors, window opening and closing mechanisms, triggering alarms on detected events, etc., Energy Monitoring, By connecting NodeMCU to energy meters or current sensors, it's possible to monitor power consumption in real-time. This information can be used for energy optimization and to identify power consuming devices. Smart Lighting, NodeMCU can control LED strips or smart bulbs, enabling you to create dynamic lighting effects, adjust color and brightness, or integrate with other smart home systems. Industrial Automation, NodeMCU can be utilized in industrial automation systems to monitor and control machinery, collect sensor data, and enable remote management of industrial processes. Prototyping and Education, Due to its ease of use, affordability, and extensive community support, NodeMCU is often used for prototyping new ideas and for educational purposes, helping beginners learn about IoT, electronics, and programming. These are just a few examples, but the versatility and flexibility of NodeMCU make it suitable for a wide range of projects in the field of IoT and automation.

3.1.2 FC-28 Soil Hygrometer (Moisture) Sensor

The FC-28 Soil Hygrometer, also known as a moisture sensor, is a commonly used sensor to measure the moisture content in soil. It is widely used in various applications related to plant growth, irrigation, and agriculture. The FC-28 Soil Hygrometer sensor operates based on the principle of electrical conductivity. It consists of two probes that are inserted into the soil. When the soil is moist, it conducts electricity, allowing current to flow between the probes. The resistance between the probes is measured, and it is inversely proportional to the moisture content in the soil. The FC-28 Hygrometer posseses features like Analog Output, The sensor provides an analog output voltage that corresponds to the moisture level in the soil. The voltage can be read by an analog-to-digital converter (ADC) of a microcontroller or an Arduino board for further processing. Adjustable Sensitivity, The sensor typically includes a potentiometer that allows you to adjust the sensitivity of the sensor according to the soil type and moisture levels required for your specific application. Easy to Use, The FC-28 Soil Hygrometer sensor is relatively easy to use and can be directly connected to microcontrollers or development boards with analog input capabilities. The

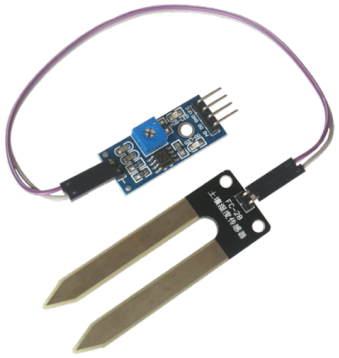


Figure 3: FC-28 Soil Hygrometer (Moisture) Sensor

FC-28 moisture sensor has some key applications as Agriculture and Plant Monitoring, The sensor is widely used in agriculture for monitoring soil moisture levels in fields, greenhouses, or indoor plantations. It helps optimize irrigation systems by providing real-time data on soil moisture, enabling efficient water management for plants. Automated Irrigation Systems, The sensor can be integrated into automated irrigation systems to measure soil moisture levels. When the moisture level drops below a threshold, the sensor triggers the irrigation system to water the plants, ensuring they receive adequate moisture for healthy growth. Gardening and Plant Care, Whether you have a small home garden or indoor potted plants, the FC-28 Soil Hygrometer sensor can be used to monitor the moisture levels in the soil. It helps avoid overwatering or underwatering, ensuring optimal growing conditions for plants. Research and Experimentation, The sensor is commonly used in scientific research, experiments, and educational projects related to plant physiology, hydroponics, and soil science. It provides valuable data for studying the effects of different moisture levels on plant growth. Environmental Monitoring, The FC-28 Soil Hygrometer sensor can be employed in environmental monitoring applications to measure soil moisture in natural habitats, such as forests or agricultural landscapes. This data can help in assessing soil health, drought conditions, and water resource management. It's important to note that while the FC-28 Soil Hygrometer sensor is a convenient tool for measuring soil moisture, its accuracy and reliability can vary depending on factors such as sensor calibration, soil composition, and environmental conditions. Regular calibration and proper installation are essential for obtaining accurate and consistent measurements.

3.1.3 5V Relay Switch

A relay switch is an electromechanical device used to control the flow of electricity in a circuit. It consists of a coil and a set of contacts. When a current is passed through the coil, it generates a magnetic field that activates the contacts, either opening or closing the circuit. The term "5V" refers to a voltage level of 5 volts, which is a common power supply voltage used in many electronic systems and devices. In the context of a relay switch, "5V" typically refers to the voltage required to activate or energize the relay coil. To use a relay switch with a 5V coil, you would typically connect the relay coil terminals to a 5V power supply. When the 5V power is applied to the coil, the magnetic field generated causes the contacts of the relay to change position, either opening or closing the circuit connected to the relay's contact terminals. Relay switches are commonly used in various applications, including

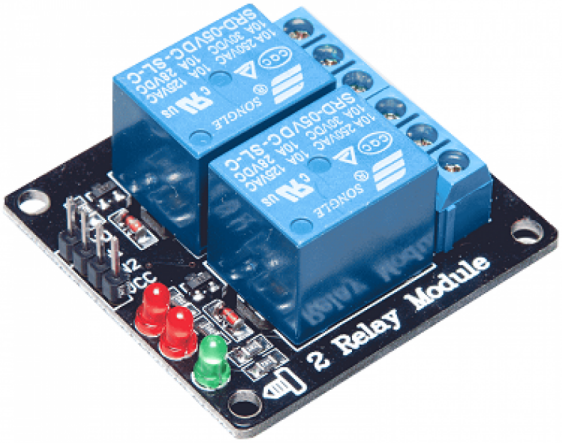


Figure 4: 5V Relay Switch

automation, control systems, and electronics projects. They can be used to switch high voltage or high current circuits using a lower voltage or signal-level control. For example, a 5V relay switch can be controlled by a microcontroller or a digital output pin of an Arduino board to control devices or systems operating at higher voltages or currents. When working with relay switches, it's important to consider factors such as the current and voltage ratings of the relay contacts, as well as the coil voltage and current requirements. It's also essential to ensure proper circuit protection and follow the relay's datasheet or specifications for correct wiring and usage .5V relay switches find applications in various electronic systems and projects where low-voltage control is required to switch higher voltage or current circuits. some of its other applications are Home Automation, 5V relay switches are commonly used in home automation projects to control lights, fans, and other electrical appliances. They

allow low-voltage control signals from microcontrollers or sensors to switch the higher voltage mains circuits. Internet of Things (IoT) Devices, In IoT applications, 5V relay switches can be used to remotely control devices and appliances over the internet. They enable the integration of low-voltage control signals from IoT platforms with higher voltage or current devices. Robotics, 5V relay switches are utilized in robotics projects to control motors, actuators, and other high-power components. They allow microcontrollers or embedded systems to safely switch and control these high-power devices. Automotive Electronics, In automotive applications, 5V relay switches can be employed to control various functions, such as headlights, windshield wipers, power windows, and door locks. They provide a safe and reliable method to switch high-current circuits in vehicles. Industrial Control Systems, 5V relay switches are widely used in industrial control systems for switching and controlling motors, pumps, valves, and other machinery. They provide a means to interface low-voltage control signals from programmable logic controllers (PLCs) or other industrial control devices with higher voltage industrial loads. Power Supply Control, Relay switches can be used in power supply control applications to switch power sources, select different voltage levels, or implement battery charging and discharging circuits. Security Systems, 5V relay switches are employed in security systems to control components such as door locks, alarm systems, and access control devices. They enable low-voltage control signals from sensors or keypads to switch the higher voltage circuits associated with security systems. Prototyping and DIY Projects, 5V relay switches are often used in prototyping and DIY electronics projects. They allow hobbyists and electronics enthusiasts to control various devices and circuits using low-voltage control signals, expanding the capabilities of their projects. These are just a few examples of the wide range of applications for 5V relay switches. The versatility, ease of use, and compatibility with microcontrollers and other control systems make them a popular choice in many electronic projects and systems.

3.1.4 5V-2A Power Supply

A 5V-2A power supply refers to a power source that provides a constant voltage of 5 volts and a maximum current output of 2 amperes. This type of power supply is commonly used in various electronic devices and applications that require a 5-volt power source capable of delivering up to 2 amps of current. Devices requiring 5V 2A power supply are Mobile Phones and Tablets, Many mobile phones and tablets require a 5V power supply to charge their batteries. A 5V-2A power supply can provide the necessary power to charge these devices efficiently. Single-Board Computers, Some single-board computers, such as the

Figure 5: 5V-2A Power Supply

Raspberry Pi, require a 5V power supply to operate. A 5V-2A power supply can provide the required power to run the board and any connected peripherals. USB-Powered Devices, Numerous USB-powered devices, including portable speakers, LED lights, and small fans, can be powered by a 5V-2A power supply. The USB connection allows these devices to draw power directly from the power supply. Arduino Projects, Arduino microcontroller boards often require a 5V power supply to operate. A 5V-2A power supply can provide sufficient power to drive the Arduino board and various components connected to it. DIY Electronics Projects, In general, many DIY electronics projects and prototypes may require a 5V power supply. The 5V-2A power supply can supply power to various components and circuits in these projects.

3.2 Hardware Design

This project senses the environmental values from the ground level and begins by sensing the moisture values using a moisture sensor, which will be placed in the soil for collecting the moisture-related data, which will be in the form of numbers ranging from 1024 to 0. 1024 represents the soil being totally dry, while 0 represents complete current flow and negligibly low resistance between the terminals of the moisture sensor, representing a soil being very wet. Then, the NodeMCU ESP8266 collected the sensed moisture values and converted them from 0 to 1(0 being completely dry(1024) and 1 being completely wet(0)) using a map. After collecting moisture values ranging from 0 to 1, The NodeMCU ESP8266 microcontroller sends these values to the cloud platform using the WiFi network. WiFi

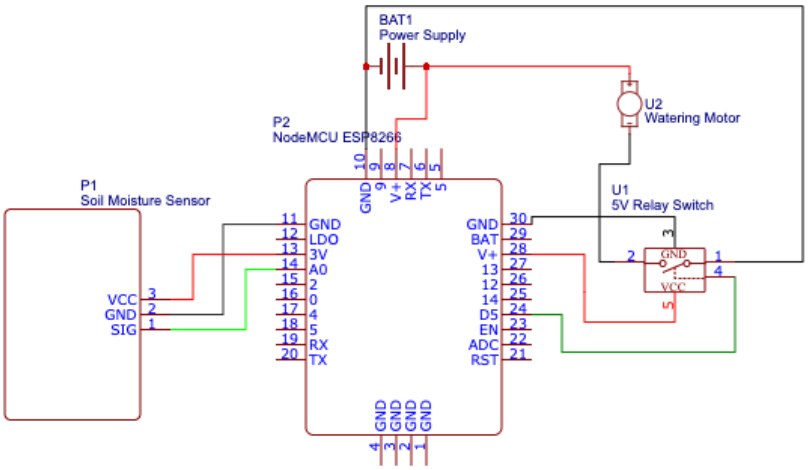


Figure 6: Hardware Design

needs to have an active Internet connection to receive those values on the cloud platform. Once the decision has been made and sent back to the NodeMCU ESP8266, it will turn off the 5V relay switch. The 5V relay switch is a relay-switching device that normally operates in the N0 and NC modes. A watering motor pump is used to pump water from the water source and provide it to the crops. Upon making a decision, the cloud server sends that decision back to the NodeMCU board, which is used to either turn on or off the motor. This ensures that the soil remains moist and intelligently provides appropriate water to the crops.

3.3 Software Components

For the software and cloud related components we have almost everything from environmental data collection to decision making and controlling the switching operation of the motor dependent on software components. In this project for uploading C code to the NodeMCU ESP8266 we are employing Arduino IDE software along with NodeMCU ESP8266 libraries such as ESP8266WiFi.h, PubSubClient.h, Wire.h, etc. for using specific library related service such as Wi-Fi connectivity, publishing and subscribing to mqtt service, collection of moisture data from the soil, etc. Then after for cloud platform we are using AWS's EC2 type cloud instance. Linux Ubuntu as a operating system for running our machine learning code. And for communicating with cloud platform we are employing services like SSH, SCP, HTTP, MQTT, etc. For SSH we will be using OpenSSH, for HTTP Apache2, for MQQT Mosquitto MQTT, etc. We will be using Python3.7 programming language for setting up

machine learning algorithm. For Machine Learning we will be using libraries such as pandas, for Decision Tree Classification tree from sklearn library, joblib library for compiling and saving the ML model into a file, random library for creating random client ids for accessing MQTT service. As an additional feature a web based "Control Panel" application which should be able to switch the motor manually, for this we created an python application using Flask framework.

3.3.1 Arduino IDE

The Arduino IDE (Integrated Development Environment) is a software tool used for programming Arduino boards. It provides a user-friendly interface for writing, compiling, and uploading code to Arduino microcontrollers. The Arduino IDE is open-source and freely available for Windows, Mac OS X, and Linux operating systems. Key features and compo-

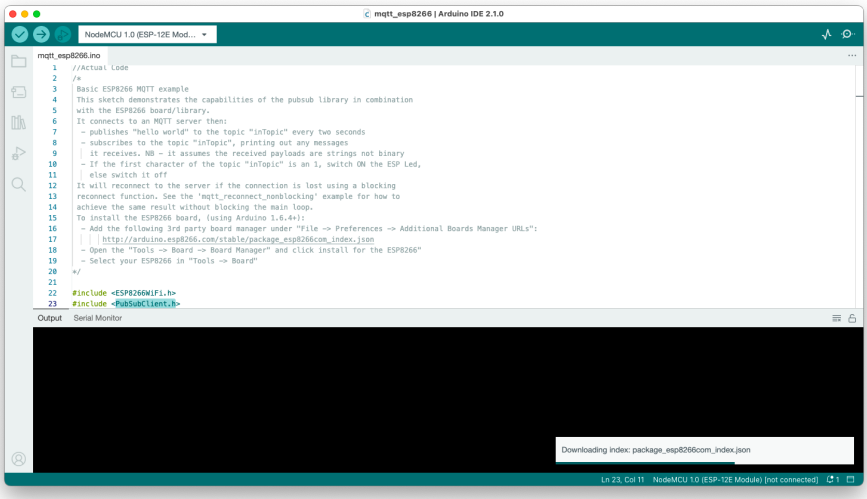


Figure 7: Arduino IDE

nents of the Arduino IDE include Code Editor, The IDE includes a text editor where you write your Arduino code. It supports syntax highlighting, auto-completion, and other features to enhance the coding experience. Library Manager, The Arduino IDE has a built-in library manager that allows you to easily search, install, and manage libraries. Libraries are collections of pre-written code that provide additional functionality and simplify the development process. Serial Monitor, The IDE provides a serial monitor tool that allows you to send and receive data between your Arduino board and your computer via the serial port. It is useful for debugging and monitoring the behavior of your Arduino program. Board Manager, The IDE includes a board manager that enables you to install and manage

different Arduino board definitions. This allows you to select the appropriate board variant and configure the compiler and upload settings accordingly. Sketches and Projects, In the Arduino IDE, programs are referred to as "sketches." The IDE organizes your sketches and associated files within a project folder, making it easy to manage multiple projects. Compilation and Upload, The Arduino IDE provides a streamlined process for compiling your code into machine language and uploading it to the Arduino board. It automatically handles the necessary steps, such as converting the code to a binary file and communicating with the board over a serial connection. Examples, The IDE includes a collection of example sketches that demonstrate various Arduino functionalities and concepts. These examples can serve as starting points for your own projects or as a way to learn how to use specific features of the Arduino platform. Overall, the Arduino IDE simplifies the process of programming Arduino boards, making it accessible to beginners while still offering flexibility for advanced users. It provides a comprehensive set of tools and features to develop, test, and deploy code on Arduino microcontrollers. The Arduino IDE can be used with the NodeMCU ESP8266 development board, allowing you to program the ESP8266 microcontroller using the Arduino programming language and libraries. Here's how we can set up the Arduino IDE for programming NodeMCU ESP8266, Download and Install Arduino IDE, Visit the Arduino website (https://www.arduino.cc/en/software) and download the latest version of the Arduino IDE for your operating system. Follow the installation instructions to install the IDE on your computer. Install ESP8266 Board Package, Open the Arduino IDE and go to "File" - "Preferences." In the "Additional Boards Manager URLs" field, enter the following URL: http://arduino.esp8266.com/stable/package_esp8266com_index.json
Click "OK" to save the preferences. Install ESP8266 Board, Go to "Tools" - "Board" - "Boards Manager." In the Boards Manager window, search for "esp8266" and select the "esp8266 by ESP8266 Community" package. Click "Install" to install the ESP8266 board package. Select NodeMCU Board, After installation, go to "Tools" - "Board" and select the appropriate NodeMCU board variant. For example, "NodeMCU 1.0 (ESP-12E Module)." Make sure the other settings, such as the CPU frequency and flash size, are correctly set for your specific NodeMCU board. Connect NodeMCU to Computer: Connect your NodeMCU board to your computer using a USB cable. Ensure that the board is recognized and that the appropriate drivers are installed if necessary. Select Serial Port, Go to "Tools" - "Port" and select the serial port that corresponds to your NodeMCU board. Test the Setup, To test the setup, you can upload a simple sketch. For example, go to "File" - "Examples" -

"ESP8266" - "Blink" and select the "Blink" sketch. Click the "Upload" button (right arrow) to compile and upload the sketch to the NodeMCU board. The onboard LED should start blinking if the upload is successful. Now we're ready to write your own Arduino sketches for the NodeMCU ESP8266 board using the Arduino IDE. You can make use of the various Arduino libraries and functions to interface with sensors, actuators, and other peripherals connected to the NodeMCU board.

3.3.2 ESP8266WiFi.h

The 'ESP8266WiFi.h' library is a header file included in the Arduino ESP8266 core. It provides functions and classes to enable Wi-Fi connectivity and network communication on ESP8266-based devices, such as the NodeMCU ESP8266 board. This library allows you to connect to Wi-Fi networks, perform network-related operations, and communicate over the internet. Here are some key features and functionalities provided by the 'ESP8266WiFi.h' library. Wi-Fi Connection Management, The library offers functions to connect to Wi-Fi networks, configure network settings, and manage Wi-Fi connectivity. You can connect to both open and secured (password-protected) networks using the 'WiFi.begin()' function. Network Status and Information, The library provides functions to retrieve information about the current Wi-Fi connection, such as the local IP address, MAC address, signal strength, and network SSID (name). You can use these functions to monitor and display network information on your device. Network Scanning, With the 'WiFi.scanNetworks()' function, you can scan for available Wi-Fi networks in your vicinity. This allows you to obtain a list of networks along with their signal strengths, channel information, and security settings. DNS Resolution, The library includes functions to resolve domain names to IP addresses using the Domain Name System (DNS). You can use the 'WiFi.hostByName()' function to convert a domain name to its corresponding IP address. Web Server and Client, The 'ESP8266WiFi.h' library also supports creating web servers and web clients. You can use the 'WiFiServer' class to create a server that listens for incoming HTTP requests and responds accordingly. The 'WiFiClient' class allows you to create a client that can send HTTP requests to web servers and retrieve responses. UDP Communication, The library provides functions for sending and receiving data over UDP (User Datagram Protocol). You can use the 'WiFiUDP' class to establish UDP connections and exchange data with other devices or servers. These are just a few examples of the functionalities provided by the 'ESP8266WiFi.h' library. It simplifies the process of implementing Wi-Fi connectivity and network communication on ESP8266-based devices, making it easier to develop IoT

applications, web servers, or client applications using the Arduino framework and ESP8266 board.

3.3.3 PubSubClient.h

The 'PubSubClient.h' library is a popular Arduino library used for MQTT (Message Queuing Telemetry Transport) communication. MQTT is a lightweight messaging protocol commonly used in IoT (Internet of Things) applications for exchanging data between devices and servers. The 'PubSubClient' library simplifies the process of connecting Arduino boards to MQTT brokers and subscribing to or publishing messages. Here are some key features and functionalities provided by the 'PubSubClient.h' library MQTT Connection: The library allows you to establish a connection to an MQTT broker. You can configure the broker's address, port, and client ID using the 'PubSubClient' class. Publish Messages, You can use the library to publish messages to an MQTT topic using the 'publish()' function. Messages can be of various data types, such as strings, integers, or byte arrays. Subscribe to Topics: The library enables you to subscribe to specific MQTT topics using the 'subscribe()' function. When a message is published to a subscribed topic, the library invokes a callback function that you can define to handle the received message. Message Callback, The library provides a callback mechanism to handle incoming messages. You can define a callback function that gets triggered whenever a new message is received on a subscribed topic. The callback function can perform custom logic to process the received data. Quality of Service (QoS), The library supports different levels of QoS for message delivery. You can specify the desired QoS level (0, 1, or 2) when publishing or subscribing to a topic. The QoS level determines the reliability and guarantee of message delivery. Retained Messages, The library allows you to publish retained messages. Retained messages are special messages that are stored by the broker and delivered to any new subscribers. They can provide persistent or initial state information to subscribers. Connection Handling, The 'PubSubClient' class includes functions to handle MQTT connection-related events, such as connection establishment, disconnection, and reconnection. It provides mechanisms for automatic reconnection in case of network or broker failures. The 'PubSubClient.h' library provides a straightforward and convenient way to integrate MQTT functionality into your Arduino projects. It is commonly used for building IoT systems, home automation projects, sensor networks, and other applications where lightweight and efficient messaging is required.

3.3.4 Wire.h

The ‘Wire.h‘ library is an Arduino library used for I2C (Inter-Integrated Circuit) communication. I2C is a widely used serial communication protocol that allows multiple devices to communicate with each other using a shared bus. The ‘Wire.h‘ library simplifies the process of implementing I2C communication on Arduino boards. Here are some key features and functionalities provided by the ‘Wire.h‘ library, I2C Master and Slave Operations, The library supports both master and slave modes of I2C communication. You can configure an Arduino board as either an I2C master or slave device. Master Functions, As a master, you can use the library to initiate communication with I2C slave devices, send data (commands or requests), and receive data from the slaves. The library provides functions such as ‘Wire.begin()‘ to initialize the I2C bus and ‘Wire.requestFrom()‘ to request data from a slave device. Slave Functions, As a slave, you can use the library to respond to master requests, receive data from the master, and send data back. The library provides functions such as ‘Wire.onReceive()‘ and ‘Wire.onRequest()‘ to handle incoming data and outgoing data requests from the master. Multiple Devices, The ‘Wire.h‘ library supports multiple I2C devices connected to the same bus. Each device is identified by a unique 7-bit or 10-bit address. The library provides functions to select the target device by specifying its address. Data Transmission, The library handles the low-level details of I2C data transmission, including generating start and stop conditions, addressing devices, and transmitting and receiving data using the I2C protocol. Timeouts and Error Handling, The library includes timeout mechanisms to handle situations where devices do not respond or communication fails. It provides error codes and functions to handle errors and exceptions during I2C communication. The ‘Wire.h‘ library is commonly used in various Arduino projects that involve communication between multiple devices, such as sensors, displays, and other modules. It provides a simple and convenient way to implement I2C communication, allowing for efficient and reliable data exchange between devices on the same I2C bus.

3.3.5 AWS EC2 Instance

An AWS EC2 (Elastic Compute Cloud) instance is a virtual server in the cloud provided by Amazon Web Services (AWS). It enables you to run applications and services in a scalable and flexible manner without the need to invest in physical hardware. Here are some key points and features of AWS EC2 instances Virtual Servers, EC2 instances are virtual machines that can be launched in the AWS cloud. You have full control over the operating

Figure 8: AWS EC2 Instance

system, networking, and applications running on the instance. Scalability, EC2 instances can be easily scaled up or down to meet the demand of your applications. You can add or remove instances as needed, allowing you to handle varying workloads efficiently. Multiple Instance Types, AWS offers a wide range of instance types optimized for different use cases. Each instance type has varying combinations of CPU, memory, storage, and network capacity to cater to specific application requirements. Storage Options, EC2 instances provide various storage options. You can choose from instance store, which is temporary and local storage, or Amazon EBS (Elastic Block Store), which provides persistent block-level storage volumes. Networking Capabilities, EC2 instances are connected to virtual networks called Amazon Virtual Private Cloud (VPC). You have control over networking configuration, including subnets, IP addresses, routing tables, and security groups, allowing you to isolate and secure your instances. Availability Zones, EC2 instances can be deployed in different Availability Zones within an AWS Region. Availability Zones are separate data centers with redundant power, networking, and cooling infrastructure. Deploying instances across multiple Availability Zones increases availability and fault tolerance. Security, AWS provides several security features for EC2 instances. These include security groups, network ACLs (Access Control Lists), private subnets, and integration with AWS Identity and Access Management (IAM) for access control and authentication. Integration with Other AWS Services, EC2 instances can seamlessly integrate with other AWS services, such as Amazon S3 for object storage, Amazon RDS for managed databases, and Amazon SQS for message queueing, to create comprehensive and scalable architectures. Billing, EC2 instances are billed based on usage, typically by the hour or by the second, depending on the instance type and pricing model you choose. AWS offers various pricing options, including on-demand, reserved instances, and spot instances for cost optimization. EC2 instances are

widely used by businesses and developers to host websites, run applications, perform data processing, and handle various workloads in a flexible and scalable manner. The versatility and extensive feature set of EC2 make it a fundamental building block of many cloud-based architectures on AWS.

3.3.6 Linux Ubuntu

Linux Ubuntu is a popular operating system distribution that can be used as a server for a wide range of applications. It offers stability, security, and a vast ecosystem of software packages and tools. Here are some key points to consider when using Linux Ubuntu as a server Installation and Configuration Ubuntu Server can be easily installed on a physical server or a virtual machine. During installation, you can choose the specific packages and services you need, or you can install a minimal system and add components later. Con-

Figure 9: Linux Ubuntu

figuration options are available for network settings, storage, user management, and more. Package Management, Ubuntu uses the APT (Advanced Package Tool) package management system, which makes it easy to install, update, and manage software packages. You can use the ‘apt-get‘ or ‘apt‘ command to search for, install, and remove packages from the Ubuntu repositories. A vast collection of open-source software is available for Ubuntu, including web servers, databases, programming languages, and more. Security, Ubuntu Server emphasizes security and provides robust features to protect your server. It includes firewall configuration tools like ‘ufw‘ (Uncomplicated Firewall) for managing network traffic, and you can also take advantage of tools like fail2ban for intrusion prevention. Regular security updates are released, ensuring that your server is protected against known vulnerabilities. Server Applications, Ubuntu is suitable for hosting various server applications and services. It supports popular web servers like Apache HTTP Server and Nginx, as well as databases like MySQL and PostgreSQL. Additionally, Ubuntu has excellent support for containerization technologies such as Docker and Kubernetes, allowing you to deploy and manage

applications in a scalable and efficient manner. Command Line Interface (CLI), Ubuntu Server primarily uses the command line interface, which offers flexibility and efficiency for server management. The command line allows you to perform tasks quickly, automate processes, and access advanced configurations. You can connect to the server remotely via SSH (Secure Shell) for secure command line access. Community and Support, Ubuntu has a large and active community, providing extensive documentation, forums, and resources for assistance. The Ubuntu community is known for its helpfulness and expertise, making it easy to find answers to questions and solutions to issues you may encounter while running Ubuntu Server. Long-Term Support (LTS) Releases, Ubuntu provides Long-Term Support (LTS) releases that receive updates and security patches for an extended period, typically up to five years. LTS releases are well-suited for servers requiring long-term stability and support. Linux Ubuntu as a server offers a reliable, secure, and flexible platform for hosting various applications and services. Its robust package management system, active community support, and regular updates make it a popular choice for server deployments. Whether you're running a small web server or managing a complex infrastructure, Ubuntu Server provides the tools and capabilities necessary for efficient server administration.

3.3.7 OpenSSH SSH Service

OpenSSH is an open-source implementation of the SSH (Secure Shell) protocol suite. It provides a secure remote access and file transfer functionality for Linux and Unix-like systems. The OpenSSH SSH service allows users to securely connect to a remote server, execute commands, transfer files, and establish secure tunnels for various purposes. Here are some key points and features of the OpenSSH SSH service Secure Remote Access, The OpenSSH SSH service enables secure remote access to a server over an encrypted connection. It provides a command-line interface (CLI) that allows users to remotely log in to the server and execute commands as if they were physically present at the server. Encryption and Authentication, OpenSSH uses strong encryption algorithms, such as AES (Advanced Encryption Standard), to secure the communication between the client and the server. It also supports various authentication methods, including password-based authentication, public key authentication, and certificate-based authentication. Port Forwarding and Tunneling, OpenSSH supports port forwarding and tunneling, allowing users to create secure connections to services running on a remote server. It enables the creation of secure tunnels for accessing services like databases, web servers, and remote desktops through an encrypted connection. File Transfer, OpenSSH includes the 'sftp' (SSH File Transfer Protocol) command, which allows users

to securely transfer files between the client and the server. 'sftp' provides a similar user interface to traditional FTP, but with the added security of SSH encryption. X11 Forwarding, OpenSSH supports X11 forwarding, which allows users to run graphical applications on a remote server and display them on the client machine. This feature is useful when accessing remote desktops or running GUI-based applications on a server. Configuration and Security, OpenSSH provides a configuration file ('/etc/ssh/sshd_config') that allows administrators to customize various aspects of the SSH service, including authentication methods, allowed user accounts, and security settings. It also supports features like rate limiting and connection throttling to protect against brute-force attacks. Public Key Infrastructure, OpenSSH supports public key authentication, which allows users to authenticate with a private-public key pair instead of a password. This provides a more secure and convenient way to authenticate to the server. OpenSSH is widely used in Linux and Unix-like systems for secure remote administration, file transfer, and tunneling purposes. It is a trusted and reliable tool that provides strong encryption and authentication mechanisms, ensuring the security of remote connections and data transfers.

3.3.8 Apache2 HTTP Service

Apache HTTP Server, commonly referred to as Apache, is a widely used open-source web server software. It is a powerful and flexible server that can host websites and serve web content on various platforms, including Linux, Unix, and Windows. Apache is known for its stability, performance, and extensive feature set. Here are some key points and features of

Figure 10: Apache2 HTTP Service

the Apache2 HTTP service Web Server Functionality, Apache2 is primarily used as a web server, capable of delivering web content to clients over the HTTP or HTTPS protocols. It supports static content, dynamic content generated by scripting languages (such as PHP, Python, and Perl), and CGI (Common Gateway Interface) scripts. Virtual Hosts, Apache2 allows the configuration of multiple virtual hosts, enabling the hosting of multiple websites or domains on a single server. Each virtual host can have its own settings, including separate

document roots, log files, and configurations. Modules and Extensions, Apache2 is highly extensible through modules. It comes with a wide range of built-in modules, and additional modules can be added to enhance functionality. Modules provide features such as SSL/TLS encryption, URL rewriting, authentication and authorization, caching, proxying, and more. Configuration, Apache2 has a flexible configuration system that allows fine-grained control over server behavior. The configuration files are written in plain text, making it easy to customize and manage the server settings. The main configuration file is usually located at '/etc/apache2/apache2.conf' in Linux distributions. Performance and Scalability, Apache2 is known for its performance and scalability. It can handle a large number of concurrent connections and has various settings for tuning its performance, such as configuring the maximum number of worker processes, managing caching mechanisms, and optimizing resource usage. Logging and Monitoring, Apache2 provides detailed logging capabilities to track and monitor web server activity. Access logs record every request made to the server, while error logs capture any server errors or warnings. Log files can be analyzed to troubleshoot issues, monitor traffic, and gather statistics. Security Features, Apache2 includes security features to protect against common web attacks and vulnerabilities. It supports SSL/TLS encryption for secure communication over HTTPS. Additionally, Apache2 offers modules for access control, authentication, and authorization, enabling administrators to restrict access to resources and secure the server. Community and Support, Apache has a large and active community of users and developers who provide support, documentation, and resources. The Apache Software Foundation (ASF) maintains and supports the Apache HTTP Server, ensuring regular updates, bug fixes, and security patches. Apache2 is widely used in both small and large-scale deployments, ranging from personal websites to enterprise-level applications. Its stability, performance, and extensive feature set make it a reliable choice for hosting web content and building robust web applications.

3.3.9 Mosquitto MQTT Service

Mosquitto is an open-source MQTT (Message Queuing Telemetry Transport) broker that provides a lightweight and efficient messaging protocol for connecting IoT (Internet of Things) devices. It acts as a central hub for devices to publish and subscribe to messages, facilitating real-time communication and data exchange between devices and applications. Here are some key points and features of the Mosquitto MQTT service MQTT Broker, Mosquitto acts as an MQTT broker, which is responsible for receiving messages published by MQTT clients and delivering them to interested subscribers. It supports the

Figure 11: Mosquitto MQTT Service

publish/subscribe messaging pattern, where devices can publish messages to specific topics, and other devices or applications can subscribe to those topics to receive the messages. Lightweight and Efficient, Mosquitto is designed to be lightweight and efficient, making it suitable for constrained devices and low-bandwidth networks. It uses a simple binary protocol that requires minimal overhead, allowing for efficient communication and reduced network traffic. QoS (Quality of Service) Levels, Mosquitto supports different levels of QoS to ensure reliable message delivery. The QoS levels include QoS 0 (at most once), QoS 1 (at least once), and QoS 2 (exactly once). Clients can choose the appropriate QoS level based on the desired reliability and delivery guarantees. Security, Mosquitto provides various security features to protect MQTT communication. It supports authentication and authorization mechanisms, allowing clients to authenticate with usernames and passwords or client certificates. TLS/SSL encryption can be enabled to secure the communication channel. Persistence and Retained Messages, Mosquitto can store messages persistently, allowing clients to retrieve missed messages or historical data upon reconnection. It also supports retained messages, which are messages that are kept by the broker and delivered to subscribers when they subscribe to a specific topic. Bridging and Clustering, Mosquitto supports bridging, which enables the connection of multiple MQTT brokers to exchange messages. This feature is useful for creating a distributed system or connecting devices across different networks. Additionally, Mosquitto can be deployed in a clustered setup to improve scalability and availability. Integration with Other Systems, Mosquitto can integrate with various systems and platforms, making it versatile for IoT and messaging applications. It has client libraries available for multiple programming languages, allowing easy integration with different devices and applications. Mosquitto can also be integrated with popular IoT platforms and frameworks. Community and Support, Mosquitto is an open-source project with an active community. It is maintained and supported by the Eclipse Foundation, which provides regular updates, bug fixes, and security patches. The community offers documen-

tation, forums, and resources for assistance and collaboration. Mosquitto MQTT service is widely used in IoT deployments and messaging applications. Its lightweight nature, efficient protocol, and rich feature set make it suitable for connecting and managing a large number of devices and facilitating real-time communication between them.

3.3.10 Python3.7

Python 3 is the latest major version of the Python programming language as of my knowledge cutoff in September 2021. It is a powerful and versatile programming language that emphasizes code readability and simplicity. Python 3 introduces several improvements and

Figure 12: Python3.7

new features compared to Python 2, including Print function, In Python 3, the print statement was replaced with the print function, which requires parentheses around the arguments. This change makes the syntax more consistent and allows for better control over printing. Unicode support, Python 3 handles text and strings as Unicode by default, making it easier to work with international characters and different character sets. Division operator, The division operator (/) in Python 3 performs true division, always returning a floating-point result. To perform integer division, you can use the double-slash operator (//). Improved syntax and features, Python 3 includes various syntax enhancements and additions, such as the 'yield from' expression for generator delegation, extended iterable unpacking, f-strings for formatted string literals, and more. Performance improvements, Python 3 offers performance enhancements over Python 2, including optimized bytecode execution and improved memory management. It's important to note that there may be newer versions of Python 3 released after my knowledge cutoff. You can always refer to the official Python website for the latest information and updates.

3.3.11 pandas

Pandas is an open-source data analysis and manipulation library for the Python programming language. It provides data structures and functions that allow you to efficiently work with structured data, such as tabular data (similar to a spreadsheet) and time series data. The key data structures in Pandas are the DataFrame and the Series. A DataFrame is a two-dimensional table-like data structure with labeled columns and rows, similar to a spreadsheet or a SQL table. A Series is a one-dimensional array-like object that can hold any data type and is similar to a column in a DataFrame. Pandas provides a wide range of functions and methods to manipulate, clean, transform, analyze, and visualize data. Some of the operations you can perform with Pandas include Reading and writing data, Pandas supports reading data from various file formats such as CSV, Excel, SQL databases, and more. It can also write data back to these formats. Data cleaning and preprocessing, Pandas allows you to handle missing data, remove duplicates, handle outliers, and perform various data cleaning operations. It also provides functions for data normalization and data type conversion. Data manipulation and transformation, Pandas enables you to select, filter, and transform data based on certain conditions. You can perform operations like sorting, grouping, merging, joining, and reshaping data. Data analysis and statistics, Pandas provides functions for descriptive statistics, aggregations, pivot tables, and time series analysis. It also supports advanced operations like data slicing, indexing, and handling categorical data. Data visualization, Pandas integrates well with other libraries such as Matplotlib and Seaborn to create various types of plots, charts, and graphs for data visualization. Pandas is widely used in data analysis, scientific research, finance, economics, social sciences, and many other fields where structured data needs to be analyzed and manipulated. It is a powerful tool for handling and exploring data efficiently in Python.

3.3.12 scikit-learn

Scikit-learn is a popular open-source machine learning library for Python. It provides a wide range of algorithms and tools for tasks such as classification, regression, clustering, dimensionality reduction, and model selection. Here are some key features and components of scikit-learn Consistent API Scikit-learn provides a consistent and intuitive API across different algorithms, making it easy to use and switch between models. The library follows a fit-predict paradigm, where you train a model on labeled data and then use it to make predictions on new, unseen data. Supervised learning algorithms, Scikit-learn includes a

variety of algorithms for supervised learning, such as linear regression, logistic regression, support vector machines (SVM), decision trees, random forests, gradient boosting, and more. These algorithms can be used for classification and regression tasks. Unsupervised

Figure 13: scikit-learn

learning algorithms, Scikit-learn offers several unsupervised learning algorithms, including clustering algorithms like k-means, hierarchical clustering, and DBSCAN. It also provides dimensionality reduction techniques like principal component analysis (PCA) and feature extraction methods. Model evaluation and selection, Scikit-learn provides tools for evaluating model performance using various metrics such as accuracy, precision, recall, F1-score, and more. It also offers techniques for model selection and hyperparameter tuning, including cross-validation and grid search. Preprocessing and feature engineering, Scikit-learn includes various preprocessing modules for handling data preprocessing tasks, such as scaling, normalization, handling missing values, encoding categorical variables, and feature extraction. Integration with NumPy and pandas, Scikit-learn seamlessly integrates with other popular Python libraries like NumPy and pandas. This allows for easy data manipulation and transformation, as well as efficient integration with the broader Python data science ecosystem. Scikit-learn is widely used in industry and academia for machine learning tasks due to its simplicity, extensive documentation, and rich set of functionalities. It provides a solid foundation for building machine learning models and performing data analysis tasks efficiently.

3.3.13 Decision Tree Classification

Decision tree classification is a machine learning algorithm used for solving classification problems. It is a supervised learning algorithm that builds a decision tree model based on the training data, which can then be used to classify new, unseen instances. Here's an overview of how decision tree classification works Tree Construction, The algorithm starts

with the entire training dataset and selects the best feature to split the data based on certain criteria (e.g., Gini impurity or information gain). The feature that provides the most useful information for classification is chosen as the root node of the tree. The dataset is then split into subsets based on the values of that feature. Recursive Splitting, The process of

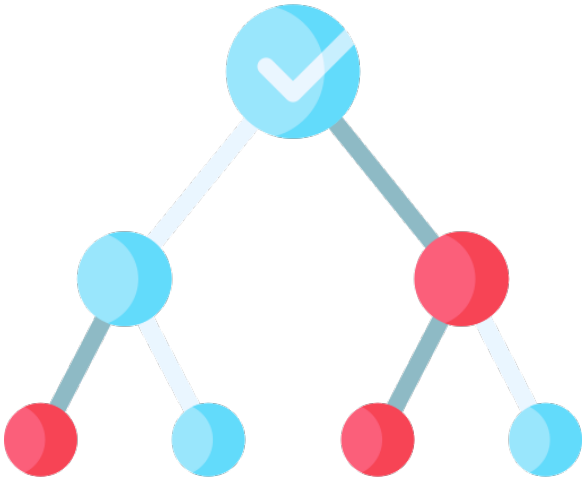


Figure 14: Decision Tree Classification

selecting the best feature and splitting the dataset is repeated recursively for each subset. This creates a tree-like structure, where each internal node represents a feature and each leaf node represents a class label or a decision. Stopping Criteria, The splitting process continues until a stopping criterion is met. This criterion could be reaching a maximum depth of the tree, reaching a minimum number of instances in a leaf node, or other pre-defined conditions. Classification, Once the tree is constructed, it can be used to classify new instances. Starting from the root node, each instance traverses down the tree by following the feature conditions until it reaches a leaf node. The class label associated with that leaf node is assigned to the instance. Decision trees have several advantages as Interpretability, Decision trees provide a clear and interpretable structure, as the decisions made at each node can be easily understood and visualized. Handling Nonlinear Relationships, Decision trees can capture nonlinear relationships between features and the target variable by employing different splitting criteria. Handling Mixed Data Types, Decision trees can handle both categorical and numerical features without requiring extensive preprocessing. Robustness to Outliers, Decision trees are less affected by outliers compared to some other algorithms since the splitting process is based on relative ordering rather than the absolute values of features. However, decision trees also have some limitations as Overfitting, Decision trees

are prone to overfitting, especially when the tree becomes too complex and captures noise or irrelevant patterns in the data. Techniques like pruning or using ensemble methods (e.g., random forests) can help mitigate this issue. Instability, Small changes in the training data can lead to significantly different decision trees, which can make the model less stable. Bias towards Unbalanced Classes, Decision trees tend to favor features with more levels or more frequent values, which can result in a bias towards classes with a larger number of instances. Scikit-learn, the popular machine learning library in Python, provides an implementation of decision tree classification as part of its 'sklearn.tree' module. You can use the 'DecisionTreeClassifier' class to train and use decision tree models for classification tasks.

3.3.14 Joblib

Joblib is a Python library used for efficiently serializing and deserializing Python objects to disk. It provides a set of tools for saving and loading objects, including models, in a binary format. Joblib is particularly useful when working with large objects, such as machine learning models, that you want to persist across sessions or share with others. Here are some key features and use cases of Joblib as Serialization, Joblib allows you to save Python objects, such as NumPy arrays, Scikit-learn models, and other custom objects, to disk. It uses a binary format that efficiently stores the data and associated metadata. Compression, Joblib provides built-in compression for the serialized objects, which can significantly reduce the disk space required to store them. This is especially beneficial when dealing with large datasets or models. Fast IO, Joblib is designed to efficiently handle IO operations, making it suitable for working with large objects that may not fit entirely in memory. It uses memory mapping to enable reading and writing data directly from disk, which can speed up the process. Parallel Processing, Joblib supports parallel processing, allowing you to leverage multiple CPU cores for faster serialization and deserialization. It offers simple APIs to parallelize computations, making it easy to distribute the workload across multiple processors. Integration with Scikit-learn, Joblib is commonly used with Scikit-learn to save and load machine learning models. It seamlessly integrates with Scikit-learn's model persistence functionalities, allowing you to store trained models for later use or deployment. To use Joblib, you need to import the library and use its 'dump()' function to save objects to disk and 'load()' function to load them back into memory. Joblib is widely used in the Python ecosystem, especially in machine learning and data science workflows. It provides a convenient and efficient way to store and retrieve Python objects, making it easier to work

with large datasets and models.

3.3.15 Flask Web Framework

Flask is a popular web framework for building web applications in Python. It is known for its simplicity, flexibility, and ease of use. Flask is considered a "micro" framework, which means it provides only the essential features needed for web development, allowing developers to add additional libraries and tools as needed. Here are some key features and

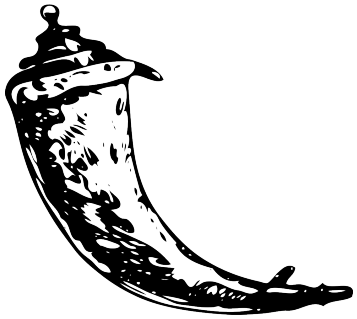


Figure 15: Flask Web Framework

concepts related to Flask Routing, Flask uses a routing mechanism to map URLs to functions or methods. You can define routes using decorators, such as '@app.route('/path')', which associate a URL with a specific function. Views and Templates, Flask follows the Model-View-Controller (MVC) pattern, where views are responsible for handling requests and returning responses. Views often render templates, which are HTML files containing dynamic data using templating engines like Jinja2. Request Handling, Flask provides convenient access to request-related information, such as request headers, form data, and query parameters, through the 'request' object. You can retrieve data from the request using methods like 'request.form.get('key')' or 'request.args.get('key')'. Responses, Flask allows you to return various types of responses. For example, you can return plain text, HTML, JSON, or even files. The response object is usually created using the 'make_response()' function. Flask Extensions, Flask has a rich ecosystem of extensions that provide additional functionalities, such as handling database interactions, user authentication, form validation, and more. Some popular extensions include Flask-SQLAlchemy, Flask-Login, Flask-WTF, and Flask-RESTful. Configuration, Flask allows you to define application configurations using Python objects or configuration files. You can specify settings like the application secret key, database connection details, and debug mode. Flask CLI, Flask provides a command-line interface (CLI) for managing and running your application. It allows you to start a development server, create database tables, and perform other common tasks. To get started with

Flask, you need to install it using pip, the Python package manager. You can then create a new Flask application by defining a Python file, importing the Flask module, and writing the necessary routes and views. Finally, you run the application using the Flask CLI or by executing the Python file.

3.4 Software Design

In the software design process, we begin by receiving moisture values from the microcontroller on the cloud platform. Our cloud platform hosts the brain of our overall project, which is a decision-tree algorithm. We will use Linux based Ubuntu operating system on the cloud for running our Decision Tree machine learning algorithm using Python3 The software

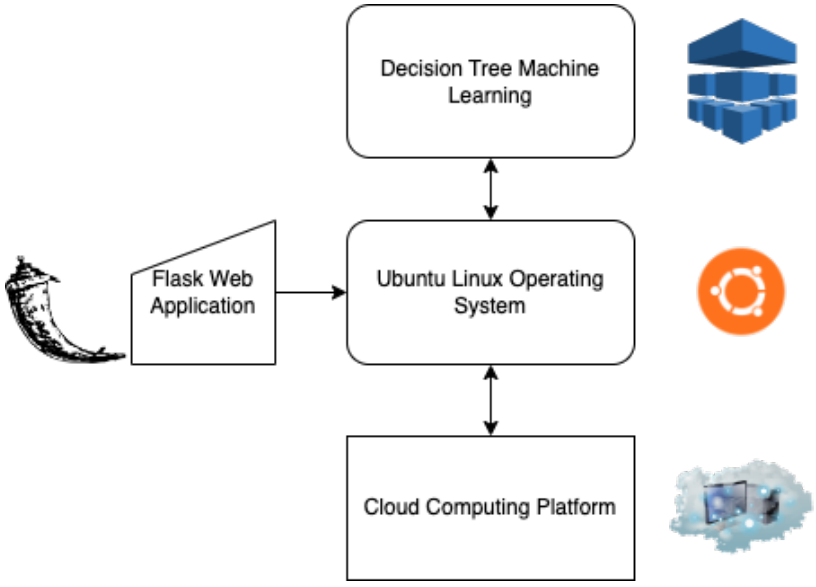


Figure 16: Software Design

part starts by receiving the moisture values from the NodeMCU ESP8266 microcontroller module to the cloud platform. The NodeMCU ESP8266 will publish the sensed values on topic "inTopic" of "Mosquitto MQQT" service running on the same cloud platform. Then the "inTopic" topic will be used to retain those values to our Decision Tree machine learning algorithm. The Decision Tree machine learning algorithm, written in Python3, decide whether to turn the water pump on or off depending on the soil condition. Once the Decision Tree ML file is decided in the form of either 1(for on) or 0(for off) on the topic "outTopic." This will be subscribed by the NodeMCU ESP8266 microcontroller, and depending on either 1 or 0, the NodeMCU ESP8266 will turn on or off the water pump, respectively. After confirming the automatic working of the proposed cloud based system, There is still a room for manual control, suppose the user wants to turn off the motor regardless of soil condition,

lets say it is dry. The moisture sensor along with whole cloud based setup including the ML will find it dry and as per the programming it will try to turn on the motor to make it wet and achieve moisturised condition. But suppose the well which is being used to pump water has ran out of water and became bone dry. In this condition the motor will definitely get burned to avoid this and bring human supervision in the project we need to have a manual control unit which will reside above the automatic system. To achieve we are using a python based Flask Web Application which will be able to manually turn on, off the motor and set it either in manual mode or back to the automatic mode as per the input from the user. As in the figure above, The Flask Web Framework Application will be hosted on the port

Figure 17: Web Based Control Panel

8080 of the cloud platform and it will be accessible from the pubic IPV4 domain address of the cloud based server. The usage is as simple as giving a text input in the input field. To turn on the motor in manual mode we have to type "ON" and submit similarly for turning off we have to type "OFF" regardless of case the control panel will switch accordingly. But once after giving a manual input either as "ON" or "OFF", the control panel will stay in the manual mode. In manual mode the system will not consider any responses from the ML model unless switched back to the auto mode again To do that we have to give input as "AUTO", once given the system will start processing received moisture values again and the ML model will produce output depending upon them and send it to the NodeMCU for switching the motor's state. In auto mode the ML model will send its output to the NodeMCU repeatedly which introduces redundancy in the switching. To avoid that will be creating and using our own configuration file residing at "/etc/project.conf" location.

3.5 Database Description

The database used for this project is purely experimental. The moisture values databases for this project has been collected using experimentation in real time using soil moisture sensor along with date-time timestamp and irrigation status from 2023-04-13 08:00:35 to 2023-04-16 10:11:08. A total of 104731 moisture values were recorded and used to train the

Figure 18: Moisture Values Collection In Real Time

decision-tree ML algorithm. The ML model uses all these moisture values and irrigation status to predict whether to turn on or off the watering pump upon feeding the moisture values. For example, when the feeding value is 0.1(which is considered dry) the model predicts it as dry, which is a correct prediction. Therefore, it sends decision 1(to turn on) and sends it back to the NodeMCU ESP8266 microcontroller to turn on the motor. Assuming a value of 0.9(which is considered wet) the model predicts it as wet, which is a correct prediction. Therefore, it sends decision 0(to turn off) and similarly sends back the decision to the NodeMCU ESP8266 microcontroller to turn off the watering motor.

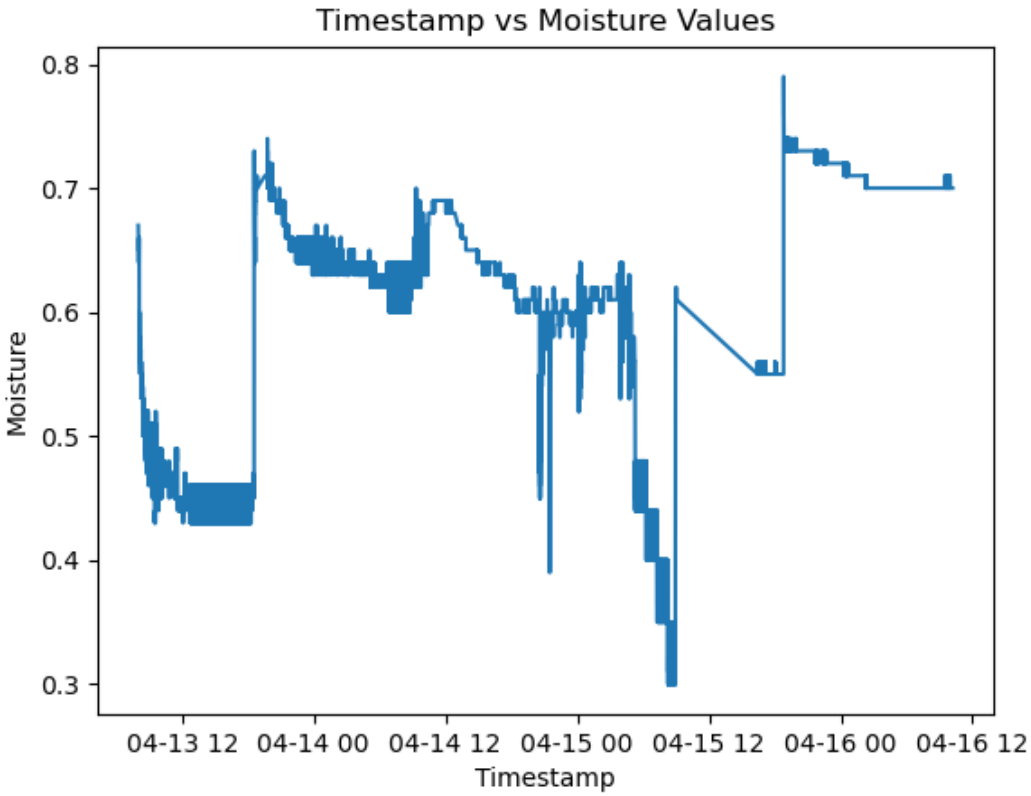


Figure 19: Moisture vs. Timestamp Graph

4 RESULTS

4.1 Simulation Results

For the simulations, the cloud server's copy is executed locally to see and verify if the cloud ports are working as expected. As shown in the following figure, the cloud server is hosted locally on the VMware Fusion application: The service is hosted at port 1234 of the cloud's

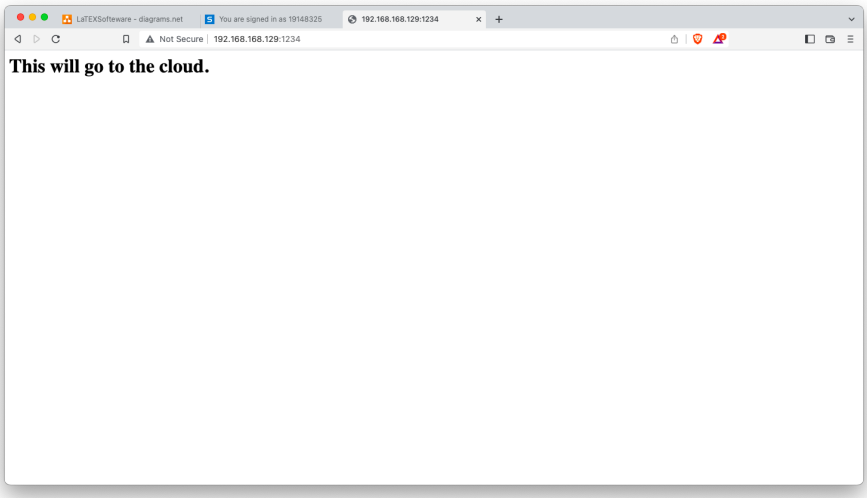


Figure 20: Software Simulation Results

local virtual machine. As of now the server is printing the message "This will go to the cloud." on the screen when they visited it. This confirms that the server is working properly and is able to communicate with the client and the microcontroller board.

The next step is to set up an AWS-based EC2 instance of a cloud platform that will be used to run our ML model and generate an output depending on the information fed. First, we signed up to the AWS cloud platform, chose the EC2 type instance and initiated it. For the operating system, we chose Linux Ubuntu as it is lightweight, easy, and fast to operate, as shown in figure 6. During implementation, we must ensure that the AWS cloud platform firewall rules are updated along with Ubuntu's "ufw" firewall for the services needed for operation of the services. For instance, we must update the wizards of the AWS's security groups to open the ports needed for data communication for services such as SSH, MQTT, and HTTP.

Once opened the required ports for services like SSH, we can start with communicating the instantiated cloud instance. to do that we have to use SSH service, for that we have the command,

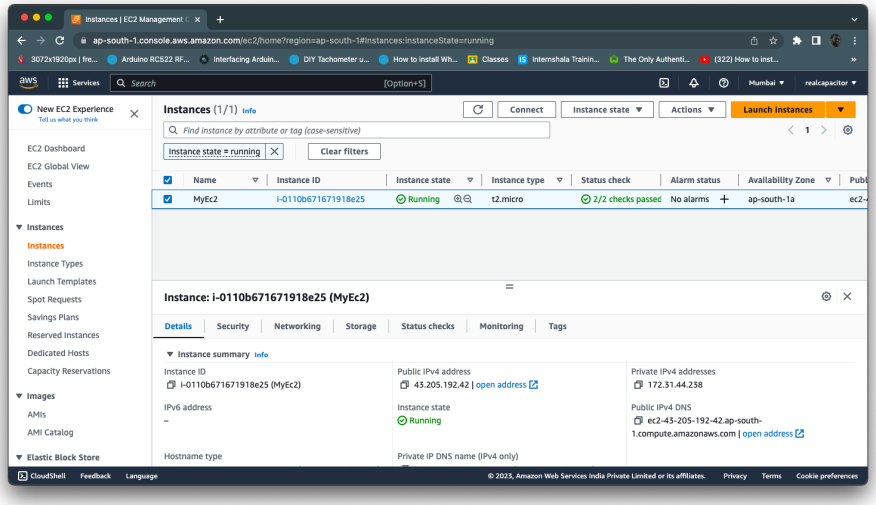


Figure 21: AWS Cloud Platform Running Linux Ubuntu

```
ssh -i AWSkey.pem ubuntu@45.205.192.142
```

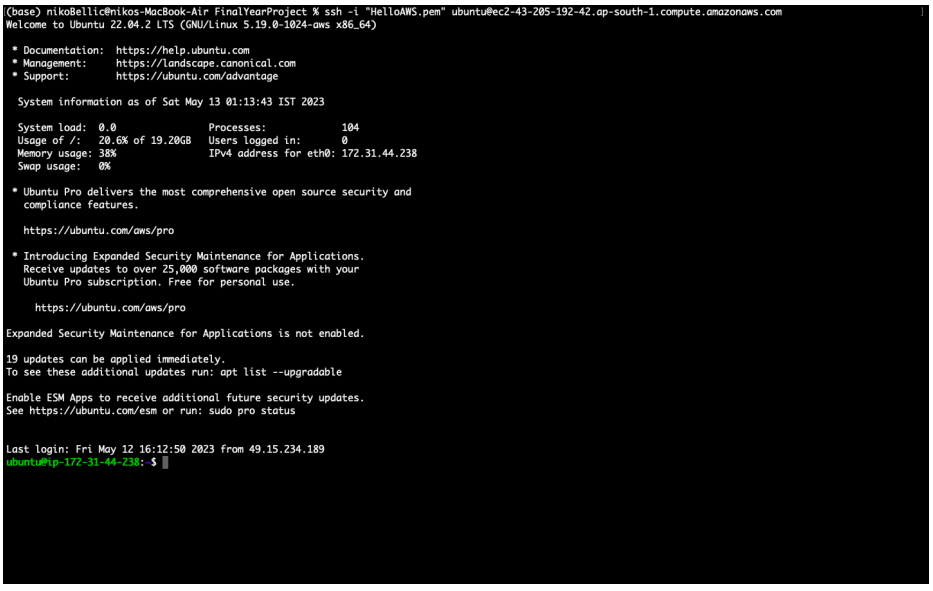


Figure 22: SSH From Local System to Cloud Platform

Our next step will be updating, upgrading and installing all the necessary services and tools required for the project using "apt" command,

```
sudo apt update
```

Our next step will be to create and test the Decision Tree ML algorithm against recorded values and measure accuracy. for that we will use "Jupyter Notebook." The model was tested using the model using the model using "sklearn" library by splitting values from 80

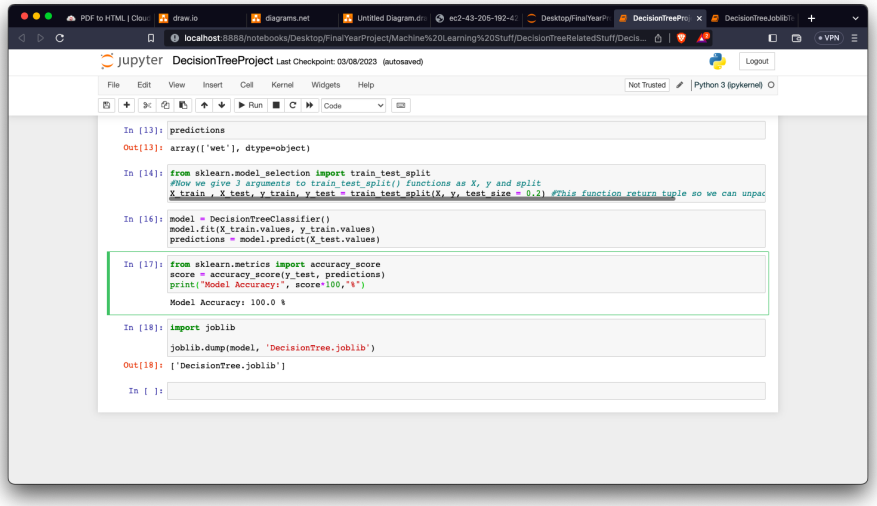


Figure 23: Accuracy of the ML Model

to 20. The 80 percent for training and 20 percent for testing accuracies were 100 percent. The next step is to send the decision made from the cloud platform's Decision Tree ML model to the NodeMCU ESP8266 microcontroller. for the Decision Tree ML model to publish the obtained decision on the "outTopic" topic, and NodeMCU ESP8266 will subscribe to it and perform action according to the instructions. As shown in the figure 9 message "1" has been

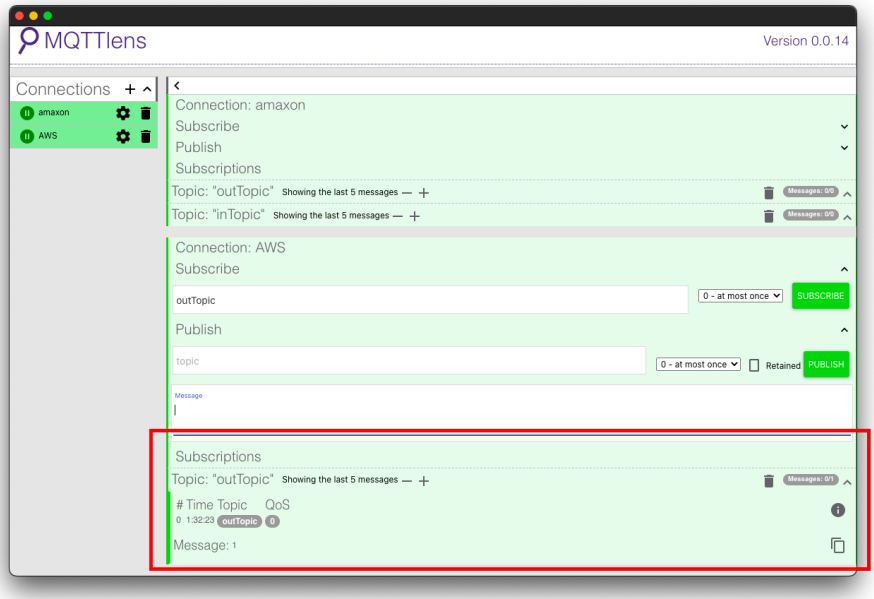


Figure 24: "outTopic" Received Message

received, which means to turn on the motor pump.

4.2 Hardware Implementation Results

So far majority of the work was located on cloud but actual application of the project lies on the ground. As planned in the hardware schematics all the components are successfully connected and all of them are up and running as the power supply is provided. As in the

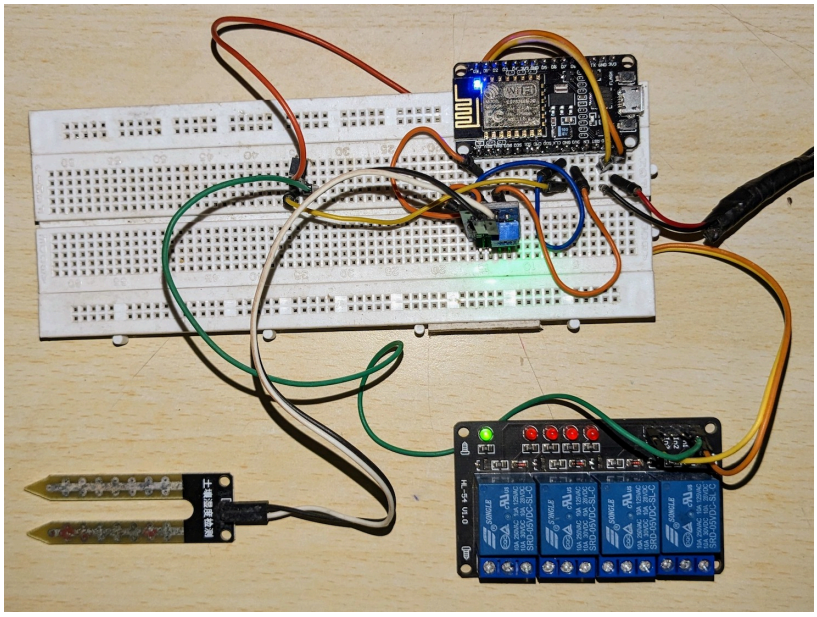


Figure 25: Hardware Model

figure above LED's of NodeMCU ESP8266, FC-28 Hygrometer (Moisture) sensor and 5V relay switch are glowing which indicates the power has reached them and they are working as expected. When the probe of FC-28 Hygrometer (Moisture) sensor are dipped in the soil it started collecting moisture values and providing them to the NodeMCU ESP8266 microcontroller. The NodeMCU ESP8266 which is connected to the local Wi-Fi network will upload and publish those values on the 'inTopic' topic of the MQTT service running on the cloud platform. upon successfully making decision it will send back its decision to the NodeMCU ESP8266 microcontroller to either turn on or off the motor switch. Suppose the soil was dry and resulting decision is turn on then the NodeMCU ESP8266 microcontroller will implement that operation on the 5V relay switch circuit. In this hardware simulatio the hardware along with the software part was working and behaving as intended and expected. The hardware setup is also working as expected when operated in a manual mode. In manual mode of operation when input fed as 'ON' or 'MANUAL' and then 'ON' the switches were closing and creating connection as expected. and When fed "OFF" or 'MANUAL' and then 'OFF' as a input the relay switch was opening and breaking the connection and it was no longer considering any input from the ML model unit "AUTO" input had been given.

4.3 Conclusion

The main purpose of this project is to provide a tool that farmers and cultivators may use to verify the compatibility of environmental conditions to boost crop growth and productivity in an efficient manner, resulting in the highest possible yield. Compared to the pricey soil testing equipment, this project provides virtually correct data. The model can be applied to any location. Once we upload the code in the microcontroller and on the cloud, it is not necessary to reprogram it later. Using this technology, many other projects can be conducted. By measuring the values of soil moisture, we can accordingly vary the amount of water needed to irrigate the soil in a particular season in a more efficient manner. Such a device is necessary for every farmer and would increase the water efficiency to its best. Thus, groundwater issues will also be resolved, and water wastage will also be reduced. The use of diesel pumps will also be reduced, saving non-renewable resources.

References

[1] J. Doorenbos, W. Pruitt, Guidelines for predicting crop water requirements. FAO Irrigation and Drainage Paper 24, Food and Agriculture Organization, UN, Rome, 1977.

[2] Brouwer, C., Heibloem, M., Irrigation Water Management. Training Manual, vol. 3. Prov. Ed., 60 pp. Rome, Italy: FAO, 1986.

[3] G. Kavianand, V.M. Nivas, R. Kiruthika and S. Lalitha, "Smart drip irrigation system for sustainable agriculture", 2016 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR), Chennai, 2016, pp. 19-22

[4] D. Mishra, A. Khan, R. Tiwari and S. Upadhyay, "Automated Irrigation System-IOT Based Approach", 2018 3rd International Conference On Internet of Things: Smart Innovation and Usage (IoT-SIU), Bhimtal, 2018, pp. 1-4.

[5] R. Varghese and S. Sharma, "Affordable Smart Farming Using IoT and Machine Learning," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2018, pp. 645-650.

[6] Parihar, Yogendra Singh, (2019) Internet of Things and Nodemcu A Review of use of Nodemcu ESP8266 in IoT products. 6. 1085.

[7] Agavanakis, Kyriakos & Karpetas, George & Taylor, Michael & Pappa, Evangelia & Michail, Christos & Filos, John & Trachana, Varvara & Kontopoulou, Lamprini, "Practical machine learning based on cloud computing resources," AIP Conference Proceedings 2123, 020050 (2019).

[8] Shilpa Chandra, Samiksha Bhilare, Mugdha Asgekar and Ramya R. B., "Crop Water Requirement Prediction in Automated Drip Irrigation System using ML and IoT ," 2021 International Conference on Nascent Technologies in Engineering (ICNTE 2021)

Appendix

Table 1: Bill of Material

Sr. No.	**Component Name**	**Unit**	**Price per Unit**	**Cost**
1	NodeMCU ESP8266	1	Rs 281	Rs 281
2	FC-28 Hygrometer (Moisture) Sensor	1	Rs 40	Rs 40
3	5V Relay Switch	1	Rs 167	Rs 167
4	Breadboard	1	Rs 132	Rs 132
5	Jumper Cables	12	Rs 3	Rs 36
6	AWS EC2	8 Months	Rs 9 per Month	Rs 72
7	Other Expenses	-	-	Rs 500
Total Cost	-	-	-	Rs 1228

ESP8266EX

Datasheet

Version 6.9

Espressif Systems

About This Guide

This document introduces the specifications of ESP8266EX.

Release Notes

Date	Version	Release Notes
2015.12	V4.6	Updated Chapter 3.
2016.02	V4.7	Updated Section 3.6 and Section 4.1.
2016.04	V4.8	Updated Chapter 1.
2016.08	V4.9	Updated Chapter 1.
2016.11	V5.0	Added Appendix II “Learning Resources”.
2016.11	V5.1	Changed the power consumption during Deep-sleep from 10 μA to 20 μA in Table 5-2.
2016.11	V5.2	Changed the crystal frequency range from “26 MHz to 52 MHz” to “24 MHz to 52 MHz” in Section 3.3.
2016.12	V5.3	Changed the minimum working voltage from 3.0 V to 2.5 V.
2017.04	V5.4	Changed chip input and output impedance from 50 Ω to 39 + j6 Ω.
2017.10	V5.5	Updated Chapter 3 regarding the range of clock amplitude to 0.8 V ~ 1.5 V.
2017.11	V5.6	Updated VDDPST from 1.8 V ~ 3.3 V to 1.8 V ~ 3.6 V.
2017.11	V5.7	• Corrected a typo in the description of SDIO_DATA_0 in Table 2-1; • Added the testing conditions for the data in Table 5-2.
2018.02	V5.8	• Updated Wi-Fi protocols in Section 1.1; • Updated description of the integrated Tensilica processor in 3.1.

Date	Version	Release Notes
2018.09	V5.9	• Update document cover; • Added a note for Table 1-1; • Updated Wi-Fi key features in Section 1.1; • Updated description of the Wi-Fi function in 3.5; • Updated pin layout diagram; • Fixed a typo in Table 2-1; • Removed Section AHB and AHB module; • Restructured Section Power Management; • Fixed a typo in Section UART; • Removed description of transmission angle in Section IR Remote Control; • Other optimization (wording).
2018.11	V6.0	• Added an SPI pin in Table 4-2; • Updated the diagram of packing information.
2019.08	V6.1	Removed description of the GPIO function in Section 4.1.
2019.08	V6.2	Updated notes on CHIP_EN in Section 5.1
2019.12	V6.3	Add feedback links.
2020.04	V6.4	• Removed the description of “Antenna diversity”; • Updated the feedback links.
2020.07	V6.5	• Updated the description of HSPI in Section 4.3; • Updated links in Appendix.
2020.10	V6.6	• Fixed a typo in Figure 2-1; • Updated the link of *ESP8266 Pin List.*
2022.07	v6.7	• Updated Figure 2-1; • Updated the link of *ESP8266 Hardware Resources.*
2022.10	v6.8	Updated typos in Chapter 6.
2023.02	v6.9	Added link to *Xtensa® Instruction Set Architecture (ISA) Summary* in Section 3.1.1.

Documentation Change Notification

Espressif provides email notifications to keep customers updated on changes to technical documentation. Please subscribe at https://www.espressif.com/en/subscribe.

Certification

Download certificates for Espressif products from https://www.espressif.com/en/certificates**.**

Table of Contents

1. Overview

Espressif's ESP8266EX delivers highly integrated Wi-Fi SoC solution to meet users' continuous demands for efficient power usage, compact design and reliable performance in the Internet of Things industry.

With the complete and self-contained Wi-Fi networking capabilities, ESP8266EX can perform either as a standalone application or as the slave to a host MCU. When ESP8266EX hosts the application, it promptly boots up from the flash. The integrated high-speed cache helps to increase the system performance and optimize the system memory. Also, ESP8266EX can be applied to any microcontroller design as a Wi-Fi adaptor through SPI/SDIO or UART interfaces.

ESP8266EX integrates antenna switches, RF balun, power amplifier, low noise receive amplifier, filters and power management modules. The compact design minimizes the PCB size and requires minimal external circuitries.

Besides the Wi-Fi functionalities, ESP8266EX also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor and on-chip SRAM. It can be interfaced with external sensors and other devices through the GPIOs. Software Development Kit (SDK) provides sample codes for various applications.

Espressif Systems' Smart Connectivity Platform (ESCP) enables sophisticated features including:

- Fast switch between sleep and wakeup mode for energy-efficient purpose;
- Adaptive radio biasing for low-power operation
- Advance signal processing
- Spur cancellation and RF co-existence mechanisms for common cellular, Bluetooth, DDR, LVDS, LCD interference mitigation

1.1. Wi-Fi Key Features

- 802.11 b/g/n support
- 802.11 n support (2.4 GHz), up to 72.2 Mbps
- Defragmentation
- 2 x virtual Wi-Fi interface
- Automatic beacon monitoring (hardware TSF)
- Support Infrastructure BSS Station mode/SoftAP mode/Promiscuous mode

1.2. Specifications

Table 1-1. Specifications

Categories	Items	Parameters
Wi-Fi	Certification	Wi-Fi Alliance
	Protocols	802.11 b/g/n (HT20)
	Frequency Range	2.4 GHz ~ 2.5 GHz (2400 MHz ~ 2483.5 MHz)
	TX Power	802.11 b: +20 dBm
		802.11 g: +17 dBm
		802.11 n: +14 dBm
	Rx Sensitivity	802.11 b: –91 dbm (11 Mbps)
		802.11 g: –75 dbm (54 Mbps)
		802.11 n: –72 dbm (MCS7)
	Antenna	PCB Trace, External, IPEX Connector, Ceramic Chip
Hardware	CPU	Tensilica L106 32-bit processor
	Peripheral Interface	UART/SDIO/SPI/I2C/I2S/IR Remote Control
		GPIO/ADC/PWM/LED Light & Button
	Operating Voltage	2.5 V ~ 3.6 V
	Operating Current	Average value: 80 mA
	Operating Temperature Range	–40 °C ~ 125 °C
	Package Size	QFN32-pin (5 mm x 5 mm)
	External Interface	-
Software	Wi-Fi Mode	Station/SoftAP/SoftAP+Station
	Security	WPA/WPA2
	Encryption	WEP/TKIP/AES
	Firmware Upgrade	UART Download / OTA (via network)
	Software Development	Supports Cloud Server Development / Firmware and SDK for fast on-chip programming
	Network Protocols	IPv4, TCP/UDP/HTTP
	User Configuration	AT Instruction Set, Cloud Server, Android/iOS App

📖 ***Note:***

The TX power can be configured based on the actual user scenarios.

1.3. Applications

- Home appliances
- Home automation
- Smart plugs and lights
- Industrial wireless control
- Baby monitors
- IP cameras
- Sensor networks
- Wearable electronics
- Wi-Fi location-aware devices
- Security ID tags
- Wi-Fi position system beacons

2. Pin Definitions

Figure 2-1 shows the pin layout for 32-pin QFN package.

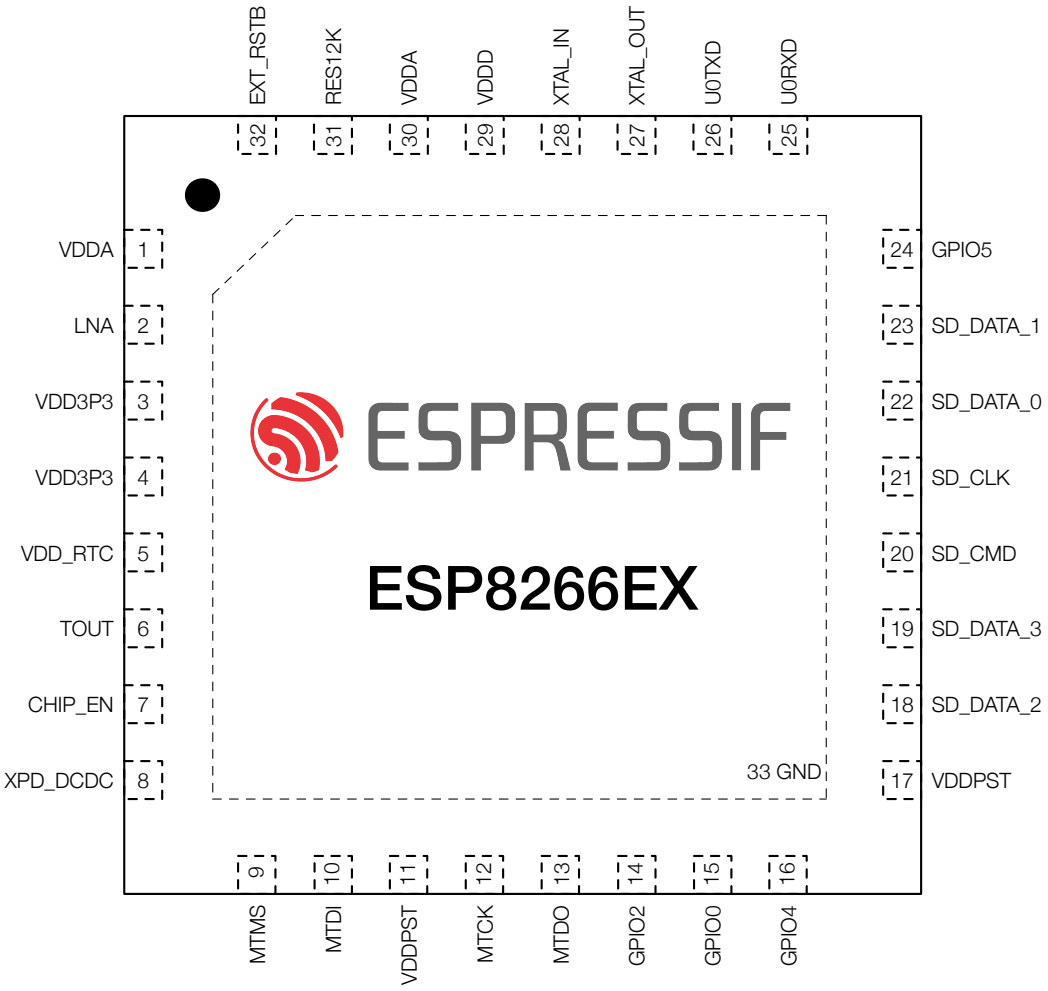


Figure 2-1. Pin Layout (Top View)

Table 2-1 lists the definitions and functions of each pin.

Table 2-1. ESP8266EX Pin Definitions

Pin	Name	Type	Function
1	VDDA	P	Analog Power 2.5 V ~ 3.6 V
2	LNA	I/O	RF antenna interface Chip output impedance = 39 + j6 Ω. It is suggested to retain the π-type matching network to match the antenna.
3	VDD3P3	P	Amplifier Power 2.5 V ~ 3.6 V

Pin	Name	Type	Function
4	VDD3P3	P	Amplifier Power 2.5 V ~ 3.6 V
5	VDD_RTC	P	NC (1.1 V)
6	TOUT	I	ADC pin. It can be used to test the power-supply voltage of VDD3P3 (Pin3 and Pin4) and the input power voltage of TOUT (Pin 6). However, these two functions cannot be used simultaneously.
7	CHIP_EN	I	Chip Enable High: On, chip works properly Low: Off, small current consumed
8	XPD_DCDC	I/O	Deep-sleep wakeup (need to be connected to EXT_RSTB); GPIO16
9	MTMS	I/O	GPIO 14; HSPI_CLK
10	MTDI	I/O	GPIO 12; HSPI_MISO
11	VDDPST	P	Digital/IO Power Supply (1.8 V ~ 3.6 V)
12	MTCK	I/O	GPIO 13; HSPI_MOSI; UART0_CTS
13	MTDO	I/O	GPIO 15; HSPI_CS; UART0_RTS
14	GPIO2	I/O	UART TX during flash programming; GPIO2
15	GPIO0	I/O	GPIO0; SPI_CS2
16	GPIO4	I/O	GPIO4
17	VDDPST	P	Digital/IO Power Supply (1.8 V ~ 3.6 V)
18	SDIO_DATA_2	I/O	Connect to SD_D2 (Series R: 20 Ω); SPIHD; HSPIHD; GPIO9
19	SDIO_DATA_3	I/O	Connect to SD_D3 (Series R: 200 Ω); SPIWP; HSPIWP; GPIO10
20	SDIO_CMD	I/O	Connect to SD_CMD (Series R: 200 Ω); SPI_CS0; GPIO11
21	SDIO_CLK	I/O	Connect to SD_CLK (Series R: 200 Ω); SPI_CLK; GPIO6
22	SDIO_DATA_0	I/O	Connect to SD_D0 (Series R: 200 Ω); SPI_MISO; GPIO7
23	SDIO_DATA_1	I/O	Connect to SD_D1 (Series R: 200 Ω); SPI_MOSI; GPIO8
24	GPIO5	I/O	GPIO5
25	U0RXD	I/O	UART Rx during flash programming; GPIO3
26	U0TXD	I/O	UART TX during flash programming; GPIO1; SPI_CS1
27	XTAL_OUT	I/O	Connect to crystal oscillator output, can be used to provide BT clock input
28	XTAL_IN	I/O	Connect to crystal oscillator input
29	VDDD	P	Analog Power 2.5 V ~ 3.6 V
30	VDDA	P	Analog Power 2.5 V ~ 3.6 V

Pin	Name	Type	Function
31	RES12K	I	Serial connection with a 12 kΩ resistor and connect to the ground
32	EXT_RSTB	I	External reset signal (Low voltage level: active)

📖 ***Note:***

1. *GPIO2, GPIO0, and MTDO are used to select booting mode and the SDIO mode;*
2. *U0TXD should not be pulled externally to a low logic level during the powering-up.*

3. Functional Description

The functional diagram of ESP8266EX is shown as in Figure 3-1.

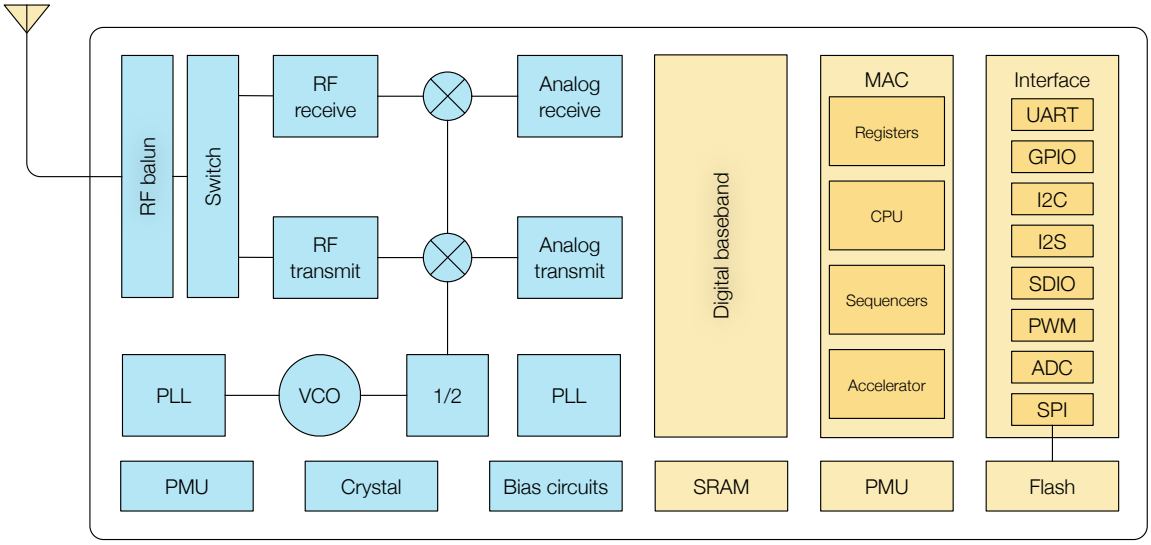


Figure 3-1. Functional Block Diagram

3.1. CPU, Memory, and Flash

3.1.1. CPU

The ESP8266EX integrates a Tensilica L106 32-bit RISC processor, which achieves extra-low power consumption and reaches a maximum clock speed of 160 MHz. The Real-Time Operating System (RTOS) and Wi-Fi stack allow 80% of the processing power to be available for user application programming and development. The CPU includes the interfaces as below:

- Programmable RAM/ROM interfaces (iBus), which can be connected with memory controller, and can also be used to visit flash.
- Data RAM interface (dBus), which can connected with memory controller.
- AHB interface which can be used to visit the register.

For information about the Xtensa® Instruction Set Architecture, please refer to *Xtensa® Instruction Set Architecture (ISA) Summary*.

3.1.2. Memory

ESP8266EX Wi-Fi SoC integrates memory controller and memory units including SRAM and ROM. MCU can access the memory units through iBus, dBus, and AHB interfaces. All memory units can be accessed upon request, while a memory arbiter will decide the

running sequence according to the time when these requests are received by the processor.

According to our current version of SDK, SRAM space available to users is assigned as below.

- RAM size < 50 kB, that is, when ESP8266EX is working under the Station mode and connects to the router, the maximum programmable space accessible in Heap + Data section is around 50 kB.
- There is no programmable ROM in the SoC. Therefore, user program must be stored in an external SPI flash.

3.1.3. External Flash

ESP8266EX uses external SPI flash to store user programs, and supports up to 16 MB memory capacity theoretically.

The minimum flash memory of ESP8266EX is shown below:

- OTA disabled: 512 kB at least
- OTA enabled: 1 MB at least

> ⚠ **Notice:**
>
> *SPI mode supported: Standard SPI, Dual SPI and Quad SPI. The correct SPI mode should be selected when flashing bin files to ESP8266. Otherwise, the downloaded firmware/program may not be working properly.*

3.2. Clock

3.2.1. High Frequency Clock

The high frequency clock on ESP8266EX is used to drive both transmit and receive mixers. This clock is generated from internal crystal oscillator and external crystal. The crystal frequency ranges from 24 MHz to 52 MHz.

The internal calibration inside the crystal oscillator ensures that a wide range of crystals can be used, nevertheless the quality of the crystal is still a factor to consider to have reasonable phase noise and good Wi-Fi sensitivity. Refer to Table 3-1 to measure the frequency offset.

Table 3-1. High Frequency Clock Specifications

Parameter	Symbol	Min	Max	Unit
Frequency	FXO	24	52	MHz
Loading capacitance	CL	-	32	pF
Motional capacitance	CM	2	5	pF

Parameter	Symbol	Min	Max	Unit
Series resistance	RS	0	65	Ω
Frequency tolerance	ΔFXO	–15	15	ppm
Frequency vs temperature (–25 °C ~ 75 °C)	ΔFXO,Temp	–15	15	ppm

3.2.2. External Clock Requirements

An externally generated clock is available with the frequency ranging from 24 MHz to 52 MHz. The following characteristics are expected to achieve good performance of radio.

Table 3-2. External Clock Reference

Parameter	Symbol	Min	Max	Unit
Clock amplitude	VXO	0.8	1.5	Vpp
External clock accuracy	ΔFXO,EXT	–15	15	ppm
Phase noise @1-kHz offset, 40-MHz clock	-	-	–120	dBc/Hz
Phase noise @10-kHz offset, 40-MHz clock	-	-	–130	dBc/Hz
Phase noise @100-kHz offset, 40-MHz clock	-	-	–138	dBc/Hz

3.3. Radio

ESP8266EX radio consists of the following blocks.

- 2.4 GHz receiver
- 2.4 GHz transmitter
- High speed clock generators and crystal oscillator
- Bias and regulators
- Power management

3.3.1. Channel Frequencies

The RF transceiver supports the following channels according to IEEE802.11 b/g/n standards.

Table 3-3. Frequency Channel

Channel No.	Frequency (MHz)	Channel No.	Frequency (MHz)
1	2412	8	2447
2	2417	9	2452
3	2422	10	2457

Channel No.	Frequency (MHz)	Channel No.	Frequency (MHz)
4	2427	11	2462
5	2432	12	2467
6	2437	13	2472
7	2442	14	2484

3.3.2. 2.4 GHz Receiver

The 2.4 GHz receiver down-converts the RF signals to quadrature baseband signals and converts them to the digital domain with 2 high resolution high speed ADCs. To adapt to varying signal channel conditions, RF filters, automatic gain control (AGC), DC offset cancelation circuits and baseband filters are integrated within ESP8266EX.

3.3.3. 2.4 GHz Transmitter

The 2.4 GHz transmitter up-converts the quadrature baseband signals to 2.4 GHz, and drives the antenna with a high-power CMOS power amplifier. The function of digital calibration further improves the linearity of the power amplifier, enabling a state of art performance of delivering +19.5 dBm average TX power for 802.11b transmission and +18 dBm for 802.11n (MSC0) transmission.

Additional calibrations are integrated to offset any imperfections of the radio, such as:

- Carrier leakage
- I/Q phase matching
- Baseband nonlinearities

These built-in calibration functions reduce the product test time and make the test equipment unnecessary.

3.3.4. Clock Generator

The clock generator generates quadrature 2.4 GHz clock signals for the receiver and transmitter. All components of the clock generator are integrated on the chip, including all inductors, varactors, loop filters, linear voltage regulators and dividers.

The clock generator has built-in calibration and self test circuits. Quadrature clock phases and phase noise are optimized on-chip with patented calibration algorithms to ensure the best performance of the receiver and transmitter.

3.4. Wi-Fi

ESP8266EX implements TCP/IP and full 802.11 b/g/n WLAN MAC protocol. It supports Basic Service Set (BSS) STA and SoftAP operations under the Distributed Control Function

(DCF). Power management is handled with minimum host interaction to minimize active-duty period.

3.4.1. Wi-Fi Radio and Baseband

The ESP8266EX Wi-Fi Radio and Baseband support the following features:

- 802.11 b and 802.11 g
- 802.11 n MCS0-7 in 20 MHz bandwidth
- 802.11 n 0.4 µs guard-interval
- up to 72.2 Mbps of data rate
- Receiving STBC 2 x 1
- Up to 20.5 dBm of transmitting power
- Adjustable transmitting power

3.4.2. Wi-Fi MAC

The ESP8266EX Wi-Fi MAC applies low-level protocol functions automatically, as follows:

- 2 × virtual Wi-Fi interfaces
- Infrastructure BSS Station mode/SoftAP mode/Promiscuous mode
- Request To Send (RTS), Clear To Send (CTS) and Immediate Block ACK
- Defragmentation
- CCMP (CBC-MAC, counter mode), TKIP (MIC, RC4), WEP (RC4) and CRC
- Automatic beacon monitoring (hardware TSF)
- Dual and single antenna Bluetooth co-existence support with optional simultaneous receive (Wi-Fi/Bluetooth) capability

3.5. Power Management

ESP8266EX is designed with advanced power management technologies and intended for mobile devices, wearable electronics and the Internet of Things applications.

The low-power architecture operates in the following modes:

- Active mode: The chip radio is powered on. The chip can receive, transmit, or listen.
- Modem-sleep mode: The CPU is operational. The Wi-Fi and radio are disabled.
- Light-sleep mode: The CPU and all peripherals are paused. Any wake-up events (MAC, host, RTC timer, or external interrupts) will wake up the chip.
- Deep-sleep mode: Only the RTC is operational and all other part of the chip are powered off.

Table 3-4. Power Consumption by Power Modes

Power Mode	Description	Power Consumption
Active (RF working)	Wi-Fi TX packet	Please refer to Table 5-2.
	Wi-Fi RX packet	
Modem-sleep①	CPU is working	15 mA
Light-sleep②	-	0.9 mA
Deep-sleep③	Only RTC is working	20 uA
Shut down	-	0.5 uA

📖 ***Notes:***

① ***Modem-sleep*** *mode is used in the applications that require the CPU to be working, as in PWM or I2S applications. According to 802.11 standards (like U-APSD), it shuts down the Wi-Fi Modem circuit while maintaining a Wi-Fi connection with no data transmission to optimize power consumption. E.g., in DTIM3, maintaining a sleep of 300 ms with a wakeup of 3 ms cycle to receive AP's Beacon packages at interval requires about 15 mA current.*

② *During* ***Light-sleep*** *mode, the CPU may be suspended in applications like Wi-Fi switch. Without data transmission, the Wi-Fi Modem circuit can be turned off and CPU suspended to save power consumption according to the 802.11 standards (U-APSD). E.g. in DTIM3, maintaining a sleep of 300 ms with a wakeup of 3ms to receive AP's Beacon packages at interval requires about 0.9 mA current.*

③ *During* ***Deep-sleep*** *mode, Wi-Fi is turned off. For applications with long time lags between data transmission, e.g. a temperature sensor that detects the temperature every 100 s, sleeps for 300 s and wakes up to connect to the AP (taking about 0.3 ~ 1 s), the overall average current is less than 1 mA. The current of 20 μA is acquired at the voltage of 2.5 V.*

4. Peripheral Interface

4.1. General Purpose Input/Output Interface (GPIO)

ESP8266EX has 17 GPIO pins which can be assigned to various functions by programming the appropriate registers.

Each GPIO PAD can be configured with internal pull-up or pull-down (XPD_DCDC can only be configured with internal pull-down, other GPIO PAD can only be configured with internal pull-up), or set to high impedance. When configured as an input, the data are stored in software registers; the input can also be set to edge-trigger or level trigger CPU interrupts. In short, the IO pads are bi-directional, non-inverting and tristate, which includes input and output buffer with tristate control inputs.

These pins, when working as GPIOs, can be multiplexed with other functions such as I2C, I2S, UART, PWM, and IR Remote Control, etc.

4.2. Secure Digital Input/Output Interface (SDIO)

ESP8266EX has one Slave SDIO, the definitions of which are described as Table 4-1, which supports 25 MHz SDIO v1.1 and 50 MHz SDIO v2.0, and 1 bit/4 bit SD mode and SPI mode.

Table 4-1. Pin Definitions of SDIOs

Pin Name	Pin Num	IO	Function Name
SDIO_CLK	21	IO6	SDIO_CLK
SDIO_DATA0	22	IO7	SDIO_DATA0
SDIO_DATA1	23	IO8	SDIO_DATA1
SDIO_DATA_2	18	IO9	SDIO_DATA_2
SDIO_DATA_3	19	IO10	SDIO_DATA_3
SDIO_CMD	20	IO11	SDIO_CMD

4.3. Serial Peripheral Interface (SPI/HSPI)

ESP8266EX has two SPIs.

- One general Slave/Master SPI
- One general Slave/Master HSPI

Functions of all these pins can be implemented via hardware.

4.3.1. General SPI (Master/Slave)

Table 4-2. Pin Definitions of SPIs

Pin Name	Pin Num	IO	Function Name
SDIO_CLK	21	IO6	SPICLK
SDIO_DATA0	22	IO7	SPIQ/MISO
SDIO_DATA1	23	IO8	SPID/MOSI
SDIO_DATA_2	18	IO9	SPIHD
SDIO_DATA_3	19	IO10	SPIWP
U0TXD	26	IO1	SPICS1
GPIO0	15	IO0	SPICS2
SDIO_CMD	20	IO11	SPICS0

📖 ***Note:***

SPI mode can be implemented via software programming. The clock frequency is 80 MHz at maximum when working as a master, 20 MHz at maximum when working as a slave.

4.3.2. HSPI (Master/Slave)

Table 4-3. Pin Definitions of HSPI

Pin Name	Pin Num	IO	Function Name
MTMS	9	IO14	HSPICLK
MTDI	10	IO12	HSPIQ/MISO
MTCK	12	IO13	HSPID/MOSI
MTDO	13	IO15	HPSICS

📖 ***Note:***

SPI mode can be implemented via software programming. The clock frequency is 20 MHz at maximum.

4.4. I2C Interface

ESP8266EX has one I2C, which is realized via software programming, used to connect with other microcontrollers and other peripheral equipments such as sensors. The pin definition of I2C is as below.

Table 4-4. Pin Definitions of I2C

Pin Name	Pin Num	IO	Function Name
MTMS	9	IO14	I2C_SCL
GPIO2	14	IO2	I2C_SDA

Both I2C Master and I2C Slave are supported. I2C interface functionality can be realized via software programming, and the clock frequency is 100 kHz at maximum.

4.5. I2S Interface

ESP8266EX has one I2S data input interface and one I2S data output interface, and supports the linked list DMA. I2S interfaces are mainly used in applications such as data collection, processing, and transmission of audio data, as well as the input and output of serial data. For example, LED lights (WS2812 series) are supported. The pin definition of I2S is shown in Table 4-5.

Table 4-5. Pin Definitions of I2S

I2S Data Input			
Pin Name	Pin Num	IO	Function Name
MTDI	10	IO12	I2SI_DATA
MTCK	12	IO13	I2SI_BCK
MTMS	9	IO14	I2SI_WS
MTDO	13	IO15	I2SO_BCK
U0RXD	25	IO3	I2SO_DATA
GPIO2	14	IO2	I2SO_WS

4.6. Universal Asynchronous Receiver Transmitter (UART)

ESP8266EX has two UART interfaces UART0 and UART1, the definitions are shown in Table 4-6.

Table 4-6. Pin Definitions of UART

Pin Type	Pin Name	Pin Num	IO	Function Name
UART0	U0RXD	25	IO3	U0RXD
	U0TXD	26	IO1	U0TXD
	MTDO	13	IO15	U0RTS
	MTCK	12	IO13	U0CTS
UART1	GPIO2	14	IO2	U1TXD
	SD_D1	23	IO8	U1RXD

Data transfers to/from UART interfaces can be implemented via hardware. The data transmission speed via UART interfaces reaches 115200 x 40 (4.5 Mbps).

UART0 can be used for communication. It supports flow control. Since UART1 features only data transmit signal (TX), it is usually used for printing log.

> 📖 ***Note:***
>
> *By default, UART0 outputs some printed information when the device is powered on and booting up. The baud rate of the printed information is relevant to the frequency of the external crystal oscillator. If the frequency of the crystal oscillator is 40 MHz, then the baud rate for printing is 115200; if the frequency of the crystal oscillator is 26 MHz, then the baud rate for printing is 74880. If the printed information exerts any influence on the functionality of the device, it is suggested to block the printing during the power-on period by changing (U0TXD, U0RXD) to (MTDO, MTCK).*

4.7. Pulse-Width Modulation (PWM)

ESP8266EX has four PWM output interfaces. They can be extended by users themselves. The pin definitions of the PWM interfaces are defined as below.

Table 4-7. Pin Definitions of PWM

Pin Name	Pin Num	IO	Function Name
MTDI	10	IO12	PWM0
MTDO	13	IO15	PWM1
MTMS	9	IO14	PWM2
GPIO4	16	IO4	PWM3

The functionality of PWM interfaces can be implemented via software programming. For example, in the LED smart light demo, the function of PWM is realized by interruption of the timer, the minimum resolution reaches as high as 44 ns. PWM frequency range is adjustable from 1000 µs to 10000 µs, i.e., between 100 Hz and 1 kHz. When the PWM

frequency is 1 kHz, the duty ratio will be 1/22727, and a resolution of over 14 bits will be achieved at 1 kHz refresh rate.

4.8. IR Remote Control

ESP8266EX currently supports one infrared remote control interface. For detailed pin definitions, please see Table 4-8 below.

Table 4-8. Pin Definitions of IR Remote Control

Pin Name	Pin Num	IO	Function Name
MTMS	9	IO14	IR TX
GPIO5	24	IO 5	IR Rx

The functionality of Infrared remote control interface can be implemented via software programming. NEC coding, modulation, and demodulation are supported by this interface. The frequency of modulated carrier signal is 38 kHz, while the duty ratio of the square wave is 1/3. The transmission range is around 1m which is determined by two factors: one is the maximum current drive output, the other is internal current-limiting resistance value in the infrared receiver. The larger the resistance value, the lower the current, so is the power, and vice versa.

4.9. ADC (Analog-to-Digital Converter)

ESP8266EX is embedded with a 10-bit precision SAR ADC. TOUT (Pin6) is defined as below:

Table 4-9. Pin Definition of ADC

Pin Name	Pin Num	Function Name
TOUT	6	ADC Interface

The following two measurements can be implemented using ADC (Pin6). However, they cannot be implemented at the same time.

- Measure the power supply voltage of VDD3P3 (Pin3 and Pin4).

Hardware Design	TOUT must be floating.
RF Initialization Parameter	The 107th byte of ***esp_init_data_default.bin*** (0 ~ 127 bytes), `vdd33_const` must be set to `0xFF`.
RF Calibration Process	Optimize the RF circuit conditions based on the testing results of VDD3P3 (Pin3 and Pin4).
User Programming	Use `system_get_vdd33` instead of `system_adc_read`.

- Measure the input voltage of TOUT (Pin6).

Hardware Design	The input voltage range is 0 to 1.0 V when TOUT is connected to external circuit.
RF Initialization Parameter	The value of the 107th byte of ***esp_init_data_default.bin*** (0 ~ 127 bytes), `vdd33_const` must be set to the real power supply voltage of Pin3 and Pin4. The unit and effective value range of `vdd33_const` is 0.1 V and 18 to 36, respectively, thus making the working power voltage range of ESP8266EX between 1.8 V and 3.6 V.
RF Calibration Process	Optimize the RF circuit conditions based on the value of `vdd33_const`. The permissible error is ± 0.2 V.
User Programming	Use `system_adc_read` instead of `system_get_vdd33`.

📖 ***Notes:***

esp_init_data_default.bin *is provided in SDK package which contains RF initialization parameters (0 ~ 127 bytes). The name of the 107th byte in* ***esp_init_data_default.bin*** *is* `vdd33_const`*, which is defined as below:*

- *When vdd33_const = 0xff, the power voltage of Pin3 and Pin4 will be tested by the internal self-calibration process of ESP8266EX itself. RF circuit conditions should be optimized according to the testing results.*
- *When 18 = < vdd33_const = < 36, ESP8266EX RF Calibration and optimization process is implemented via (vdd33_const/10).*
- *When vdd33_const < 18 or 36 < vdd33_const < 255, vdd33_const is invalid. ESP8266EX RF Calibration and optimization process is implemented via the default value 3.3 V.*

5. Electrical Specifications

5.1. Electrical Characteristics

Table 5-1. Electrical Characteristics

Parameters		Conditions	Min	Typical	Max	Unit
Operating Temperature Range		-	–40	Normal	125	°C
Maximum Soldering Temperature		IPC/JEDEC J-STD-020	-	-	260	°C
Working Voltage Value		-	2.5	3.3	3.6	V
I/O	V_{IL}	-	–0.3	-	0.25 V_{IO}	V
	V_{IH}		0.75 V_{IO}		3.6	
	V_{OL}	-	-	-	0.1 V_{IO}	
	V_{OH}		0.8 V_{IO}		-	
	I_{MAX}	-	-	-	12	mA
Electrostatic Discharge (HBM)		TAMB = 25 °C	-	-	2	KV
Electrostatic Discharge (CDM)		TAMB = 25 °C	-	-	0.5	KV

Notes on CHIP_EN:

The figure below shows ESP8266EX power-up and reset timing. Details about the parameters are listed in Table 5-2.

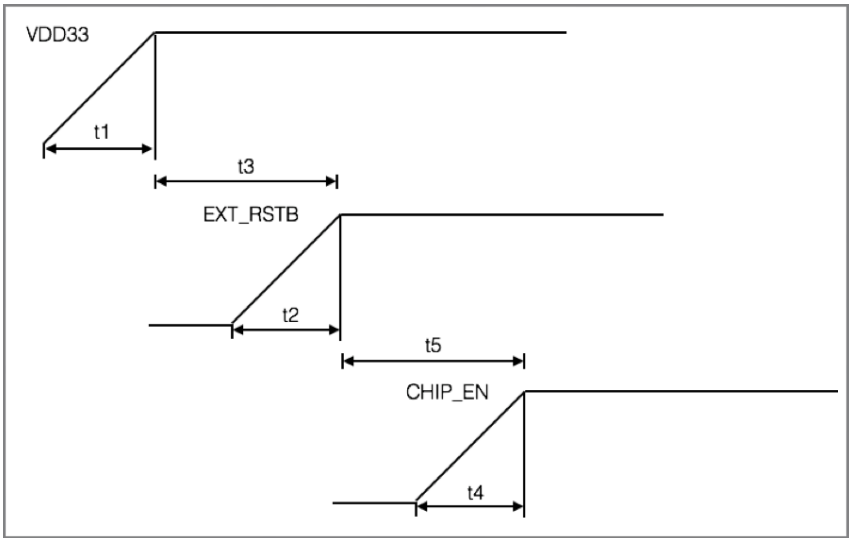


Figure 5-1. ESP8266EX Power-up and Reset Timing

Table 5-2. Description of ESP8266EX Power-up and Reset Timing Parameters

	Description	Min	Max	Unit
t1	The rise-time of VDD33	10	2000	µs
t2	The rise-time of EXT_RSTB	0	2	ms
t3	EXT_RSTB goes high after VDD33	0.1	-	ms
t4	The rise-time of CHIP_EN	0	2	ms
t5	CHIP_EN goes high after EXT_RSTB	0.1	-	ms

5.2. RF Power Consumption

Unless otherwise specified, the power consumption measurements are taken with a 3.0 V supply at 25 °C of ambient temperature. All transmitters' measurements are based on a 50% duty cycle.

Table 5-3. Power Consumption

Parameters	Min	Typical	Max	Unit
TX802.11 b, CCK 11 Mbps, P_{OUT} = +17 dBm	-	170	-	mA
TX802.11 g, OFDM 54Mbps, P_{OUT} = +15 dBm	-	140	-	mA
TX802.11 n, MCS7, P_{OUT} = +13 dBm	-	120	-	mA
Rx802.11 b, 1024 bytes packet length, –80 dBm	-	50	-	mA
Rx802.11 g, 1024 bytes packet length, –70 dBm	-	56	-	mA
Rx802.11 n, 1024 bytes packet length, –65 dBm	-	56	-	mA

5.3. Wi-Fi Radio Characteristics

The following data are from tests conducted at room temperature, with a 3.3 V power supply.

Table 5-3. Wi-Fi Radio Characteristics

Parameters	Min	Typical	Max	Unit
Input frequency	2412	-	2484	MHz
Output impedance	-	39 + j6	-	Ω
Output power of PA for 72.2 Mbps	15.5	16.5	17.5	dBm
Output power of PA for 11b mode	19.5	20.5	21.5	dBm
Sensitivity				
DSSS, 1 Mbps	-	–98	-	dBm
CCK, 11 Mbps	-	–91	-	dBm
6 Mbps (1/2 BPSK)	-	–93	-	dBm
54 Mbps (3/4 64-QAM)	-	–75	-	dBm
HT20, MCS7 (65 Mbps, 72.2 Mbps)	-	–72	-	dBm
Adjacent Channel Rejection				
OFDM, 6 Mbps	-	37	-	dB
OFDM, 54 Mbps	-	21	-	dB
HT20, MCS0	-	37	-	dB
HT20, MCS7	-	20	-	dB

6. Package Information

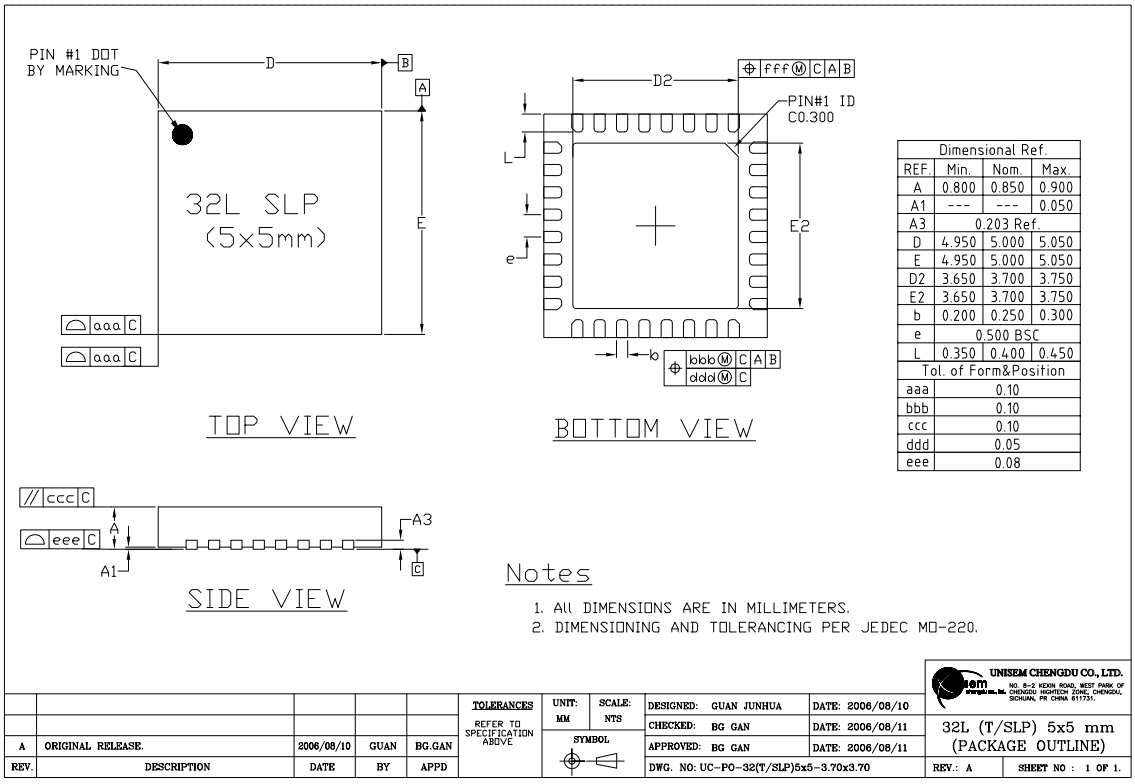


Figure 6-1. ESP8266EX Package

I. Appendix - Pin List

For detailed pin information, please see *ESP8266 Pin List*.

- Digital Die Pin List
- Buffer Sheet
- Register List
- Strapping List

> 📖 ***Notes:***
>
> - *INST_NAME refers to the IO_MUX REGISTER defined in* ***eagle_soc.h****, for example MTDI_U refers to PERIPHS_IO_MUX_MTDI_U.*
> - *Net Name refers to the pin name in schematic.*
> - *Function refers to the multifunction of each pin pad.*
> - *Function number 1 ~ 5 correspond to FUNCTION 0 ~ 4 in SDK. For example, set MTDI to GPIO12 as follows.*
>   - `#define FUNC_GPIO12  3 //defined in eagle_soc.h`
>   - `PIN_FUNC_SELECT(PERIPHS_IO_MUX_MTDI_U,FUNC_GPIO12)`

II. Appendix - Learning Resources

II.1. Must-Read Documents

- *ESP8266 Quick Start Guide*

  Description: This document is a quick user guide to getting started with ESP8266. It includes an introduction to the ESP-LAUNCHER, instructions on how to download firmware to the board and run it, how to compile the AT application, as well as the structure and debugging method of RTOS SDK. Basic documentation and other related resources for the ESP8266 are also provided.
- *ESP8266 SDK Getting Started Guide*

  Description: This document takes ESP-LAUNCHER and ESP-WROOM-02 as examples of how to use the ESP8266 SDK. The contents include preparations before compilation, SDK compilation and firmware download.
- *ESP8266 Pin List*

  Description: This link directs you to a list containing the type and function of every ESP8266 pin.
- *ESP8266 Hardware Design Guideline*

  Description: This document provides a technical description of the ESP8266 series of products, including ESP8266EX, ESP-LAUNCHER and ESP-WROOM.
- *ESP8266 Technical Reference*

  Description: This document provides an introduction to the interfaces integrated on ESP8266. Functional overview, parameter configuration, function description, application demos and other pieces of information are included.
- *ESP8266 Hardware Resources*

  Description: This zip package includes manufacturing BOMs, schematics and PCB layouts of ESP8266 boards and modules.
- *FAQ*

II.2. Must-Have Resources

- *ESP8266 SDKs*

  Description: This webpage provides links both to the latest version of the ESP8266 SDK and the older ones.
- *ESP8266 Tools*

Description: This webpage provides links to both the ESP8266 flash download tools and the ESP8266 performance evaluation tools.

- *ESP8266 Apps*
- *ESP8266 Certification and Test Guide*
- *ESP8266 BBS*
- *ESP8266 Resources*

Espressif IOT Team

www.espressif.com

International Scientific Journal of Engineering and Management ISSN: 2583-6129
Volume: 02 Issue: 05 | May – 2023 DOI: www.isjem.com
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

Use Of Digital Technology For Micro Irrigation System To Improve Water Efficiency Of Irrigation Sector(Using IoT, ML and Cloud Computing)

Shivam Kishor Patange[1], Rushikesh Madhav Sagare[2], Shivam Govindrao Mahalanakar[3], Assistant Prof. Sachin Trankatwar[4]

[1]Shivam Kishor Patange, Electronics and Telecommunication Engineering, VPKBIET, Baramati
[2]Rushikesh Madhav Sagare, Electronics and Telecommunication Engineering, VPKBIET, Baramati
[3]Shivam Govindrao Mahalankar, Electronics and Telecommunication Engineering, VPKBIET, Baramati
[4]Assistant Prof. Sachin Trankatwar, Electronics and Telecommunication Engineering, VPKBIET, Baramati

---

Abstract - The bulk of the people in India is dependent on agriculture. Due to climate change and the steady loss of natural resources, water resources should be used for farming in an effective and exact manner. By suggesting an Internet of Things (IoT) based system for automated smart drip irrigation and monitoring, this research aims to automate the laborious operation. With the aid of sensors, it will decide whether and when to turn on or off the watering pump and water flow through drip irrigation in accordance with weather, soil, and crop conditions. This has the ability to regulate the amount of moisture in the cultivating field's soil.This has the ability to regulate the amount of moisture in the cultivating field's soil. The Idea is to create such a solution which will be able to collect, store and process the environmental values to make decision depending upon them respectively. Initialising with sensing the moisture and humidity values using the sensors. Which will send those values to the cloud via IOT technology, which will be processes there. And finally decided whether to turn on or off the motor. This will provide an efficient as well as long term solution for this problem.

Key Words: Drip Irrigation, IoT, NodeMCU, ESP8266, ML, Decision, Cloud Computing

1. INTRODUCTION

Traditionally, drip irrigation farming has been performed manually, which requires the farmer to be physically present in the field. In agriculture, the major problem faced by Indian farmers is water scarcity, which is becoming a critical issue. In addition, there are quite a few areas in our country that also face drought. To improve the usage of water, an accurate amount of water required by a particular crop must be provided only at the time it needs.

This solution proposes a "Automated Drip Irrigation System using ML, IoT and Cloud Computing". Our Sensors will detect the moisture values from the soil and send them to the cloud platform using a NodeMCU ESP8266 microcontroller (which will be responsible for all the communication and controlling actions) unit and from the cloud where we will run our Machine Learning algorithms to train our model and send the instructions back to the microcontroller and switching unit, which will be responsible for the switching of the watering motor as per the soil condition.

2. Drip Irrigation

A. Working

Farming in India is done various irrigation methods which involves

•Surface Irrigation

•Localized Irrigation

•Sprinkler Irrigation

•Drip Irrigation

•Centre Pivot Irrigation

•Sub Irrigation

•Manual Irrigation

Out of these drip irrigation is found to be most suitable as well as water efficient if compared to others, in Indian weather conditions and Indian crops like fruits and vegetables. In drip irrigation water and nutrients are allowed to drip slowly in to the root zone of the soil. The pipes are laid across the field and these pipes have holes in them after a some distance which makes sure the water should drip close to the root which minimizes evaporation.

B. Problem

Even after being so advanced and water efficient it still has some problems such as, when the motor starts pumping water into drip irrigation pipes the farmers have to be there to make sure if the water has reached to last crop in the field. operator often have no idea if the provided water is sufficient for that particular crop or not which ends up being either under-watering or over-watering the crop. Under-watering will result in insufficient nutrition to the plants whereas over-watering will wash out nutrients from the soil. This clearly shows the absence of automation as well as machine intelligence in the irrigation sector and creates a room for automation, monitoring as well machine intelligence.

3. Literature Review

When it comes to agriculture, the majority of farming uses traditional methodologies; however, in modern times, the world is moving toward newer technologies, including India. This revolution in agriculture was accompanied by the introduction of micro-irrigation and drip systems. This revolution was later contributed by many individuals and

organizations. In 1977-78 J. Doorenbos and W. Pruitt published guidelines for predicting crop water requirements in the Food and Agriculture Organization (UN), Rome. Later in 1986-87 Brouwer, C., Heibloem published Irrigation Water Management, Training manual which was focused on effective ways of water management in the Irrigation field. These Two studies have helped many people and researchers to understand effective water management in the irrigation sector.

Later in 2016-17 G. Kavianand and S. Lalitha, published a research on Smart Irrigation system for sustainable agriculture in IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR), Chennai. This is useful for understanding the automation in agriculture. But it was not enough as the world was moving towards the Internet of Things era, this technology had a huge scope in the agriculture field as well So in 2018-19 D. Mishra, A. Khan, R. Tiwari and S. Upadhyay published research on the Automated Irrigation System-IoT-based Approach at the 3rd International Conference on Internet of Things Smart Innovation and Usages (IoT-SIU), Bhimtal, which effectively gathered agriculture and the IOT under the same umbrella and demonstrated how it can be done, which was a major leap in both the agriculture and IOT domains.

But then in 2021 Shilpa Chandra, Samiksha Bhilare, Mugdha Asgekar and Ramya R. B. Published a research on Crop water requirement prediction in automated drip irrigation system using ML and IOT in IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR), Chennai which used a ML based approach which was not just a emerging technology but also way more effective, adaptable and reliable alternative over hard coded if-else conditions, this research was one of the major motivation for this solution.

4. Motivation

Despite the many advantages of micro-irrigation systems, there is still room for improvement in terms of automation and smartness. The micro-irrigation system is efficient but lacks in terms of deciding the right amount of water or when to start and stop watering, making it completely manual, labor-based, and unsupervised. Due to the unsupervised use of the water in the irrigation, not only do the crops suffer from excess watering, but also the water source undergoes depletion; however, water is wasted, which may lead to the scarcity of water after a period of time. We need a solution that can not only monitor and supervise the soil contents, but also suggest when to stop and start watering.

We are proposing a system which will be motoring the soil contents such as moisture levels but also will be able to control the motor switching upon such instructions. In this solution, we employed technologies such as IoT, Cloud Computing and Machine Learning to achieve the optimum performance of our solution.

5. Proposed System

This paper proposes a system which will extract the moisture data from the soil environment using FC-28 Soil Hygrometer (Moisture) Sensor and send that data to the AWS's Ubuntu cloud based Machine Learning model using NodeMCU ESP8266 microcontroller. The Machine Learning code will classify those values and classify that soil data as either dry or wet by referring a certain threshold set as 0.5 and depending upon the that send back a signal to the NodeMCU ESP8266 microcontroller to either turn on or turn off the watering motor.

A. Hardware Components

The hardware components used in this solution can be classified in to types as Sensor, Microcontroller and Switching unit.

a. Sensors

The sensor used in this solution is FC-28 Soil Hygrometer (Moisture) Sensor. FC-28 Soil Hygrometer (Moisture) sensor is used to measure the moisture present in the soil is terms of percentage. Both terminals of the sensors are kept in the soil, then the microcontroller passes current through one terminal and measures the resistance between the two terminals and using returns the soil moisture in form of digital values ranging from 1023 to 0. 0 being completely wet as it is showing zero resistance to the current flowing through the terminal while 1023 indicates completely dry state as no current passed from one terminal to other.

b. Microcontroller

The microcontroller used in this solution is NodeMCU ESP8266 which has an inbuilt Wi-Fi technology used for communication both in local as well public networks. The NodeMCU ESP8266 will convert the sensed moisture values ranging from 0 to 1023 on the scale of 0 to 1. 0 being completely dry and 1 being completely wet and other values will lie in between using map() function in the embedded C. In this solution NodeMCU ESP8266 is employed due to its Wi-Fi capabilities. Using Wi-Fi connectivity the NodeMCU ESP8266 will public collected moisture data to the AWS based cloud platform. Also receive the decisions from the cloud platform and implement them on the switching circuits of watering motor.

c. Switching Unit

The switching unit for switching the water motor in this solution is 5V relay switch. Which is an affordable and microcontroller controlled switch used for switching high voltage applications. It is similar to other switches as it read 0 as Low which opens the circuit (Breaks the Connection) and 1 as High which closes the circuit (Connects the Circuit). It works on 5V supply which can be provided from any available power supply system or microcontroller itself. But in this solution as the maximum output current provided by

ISSN: 2583-6129
www.isjem.com

the NodeMCU ESP8266 is 3.3V, We need to provide an external 5V DC supply to the 5V relay switch. Switching digitally using binary input as '1' and '0' makes it an ideal choice for our solution. When the decision about the switching is generated and published from the cloud, it is directly reflected on the NodeMCU ESP8266 microcontroller which can feed it to the 5V relay switch to either turn on or off the switch. Also its working voltage is 5V which is easily available on our work bench.

B. Software Components

The software components used in this solution are spreads all the way from programming the microcontroller to the cloud in designing the machine learning model to the establishing an MQTT server over there for communication and exchange of messages. Some of the key software components used in this solution are AWS EC2 Cloud for cloud computing, Linux Ubuntu as operating system on the cloud platform, Decision Tree Classification as Machine Learning Algorithm for classification of moisture values as wither wet or dry, etc.

a. AWS EC2 Cloud

The Amazon's AWS provides exclusive services to the cloud computing needs. Its is subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis. In this solution we are using AWS's EC2 due to its elastic and nature and affordability. Our need for this solution lies perfectly in the spectrum of EC2's offerings. We barely need 2GBs of RAW memory, and 20GBs of on cloud storage space to put and run our machine learning model. Due its simplicity and availability AWS's EC2 became our prime choice.

b. Linux Ubuntu

Linux Ubuntu is a Debian based linux distribution considered best for the cloud based applications. Ubuntu is widely famous for its simplicity and usability in the server arena so it is widely used. Ubuntu in written in C and is capable of running many programming languages such as Python3, Ruby, etc. Which is essential for our solution to run our machine learning model. Also Ubuntu is ideal choice for IoT applications due to its open source nature and ease of implementing MQTT protocol communication. So Linux's Ubuntu became a key point in this solution.

c. Decision Tree Classification

The Decision Tree Classification algorithm is supervised machine learning algorithm used for both classification as well as regression. The Decision Tree algorithm uses a set of input parameters for its training which can CSV files, etc. and the predict output based on provided test input. Here in this research we used experimental moisture values for classifying its nature either as dry or wet. These experimental values are collected using FC-28 Hygrometer Soil (Moisture) sensor. Decision Tree has many applications in classification as well regression and also has great accuracy in predictions.

C. Working

The working of the proposed system can be broadly classified into two sections as Hardware Design and Software Design.

a. Hardware Design

This project senses the environmental values from the ground level and begins by sensing the moisture values using a moisture sensor, which will be placed in the soil for collecting the moisture-related data, which will be in the form of numbers ranging from 1024 to 0. 1024 represents the soil being totally dry, while 0 represents complete current flow and negligibly low resistance between the terminals of the moisture sensor, representing a soil being very wet. Then, the NodeMCU ESP8266 collected the sensed moisture values and converted them from 0 to 1(0 being completely dry(1024) and 1 being completely wet(0)) using a map.

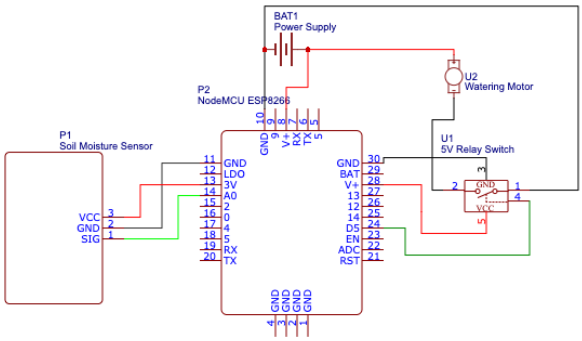


Figure 1: Hardware Design

After collecting moisture values ranging from 0 to 1, The NodeMCU ESP8266 microcontroller sends these values to the cloud platform using the WiFi network. WiFi needs to have an active Internet connection to receive those values on the cloud platform. Once the decision has been made and sent back to the NodeMCU ESP8266, it will turn off the 5V relay switch. The 5V relay switch is a relay-switching device that normally operates in the N0 and NC modes. A watering motor pump is used to pump water from the water source and provide it to the crops. Upon making a decision, the cloud server sends that decision back to the NodeMCU board, which is used to either turn on or off the motor. This ensures that the soil remains moist and intelligently provides appropriate water to the crops.

b. Software Design

In the software design process, we begin by receiving moisture values from the microcontroller on the cloud platform. Our cloud platform hosts the brain of our overall project, which is a decision-tree algorithm. We will use Linux

ISSN: 2583-6129
www.isjem.com

based Ubuntu operating system on the cloud for running our Decision Tree machine learning algorithm using Python3.

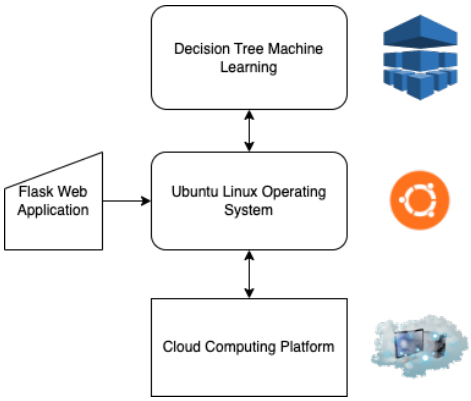


Figure 2: Software Design

The software part starts by receiving the moisture values from the NodeMCU ESP8266 microcontroller module to the cloud platform. The NodeMCU ESP8266 will publish the sensed values on topic "inTopic" of "Mosquitto MQQT" service running on the same cloud platform. Then the "inTopic" topic will be used to retain those values to our Decision Tree machine learning algorithm. The Decision Tree machine learning algorithm, written in Python3, decide whether to turn the water pump on or off depending on the soil condition. Once the Decision Tree ML file is decided in the form of either 1(for on) or 0(for off) on the topic "outTopic." This will be subscribed by the NodeMCU ESP8266 microcontroller, and depending on either 1 or 0, the NodeMCU ESP8266 will turn on or off the water pump, respectively.

After confirming the automatic working of the proposed cloud based system, There is still a room for manual control, suppose the user wants to turn off the motor regardless of soil condition, let's say it is dry. The moisture sensor along with whole cloud based setup including the ML will find it dry and as per the programming it will try to turn on the motor to make it wet and achieve moisturised condition. But suppose the well which is being used to pump water has ran out of water and became bone dry. In this condition the motor will definitely get burned to avoid this and bring human supervision in the project we need to have a manual control unit which will reside above the automatic system. To achieve we are using a python based Flask Web Application which will be able to manually turn on, off the motor and set it either in manual mode or back to the automatic mode as per the input from the user.

6. Data Collection

The database used for this project is purely experimental. The moisture values databases for this project has been collected using experimentation in real time using soil moisture sensor along with date-time timestamp and irrigation status from 2023-04-13 08:00:35 to 2023-04-16 10:11:08. A total of 104731 moisture values were recorded and used to train the decision-tree ML algorithm. The ML model uses all these moisture values and irrigation status to predict whether to turn on or off the watering pump upon feeding the moisture values.

Figure 3: Moisture Values Collection In Real Time

In the graph figure below the moisture values are mapped with respect to the time it was recored. At certain points the moisture surges up in the graph that point is the time when the irrigation is started (watering pump) till the soil gets sufficiently wet.

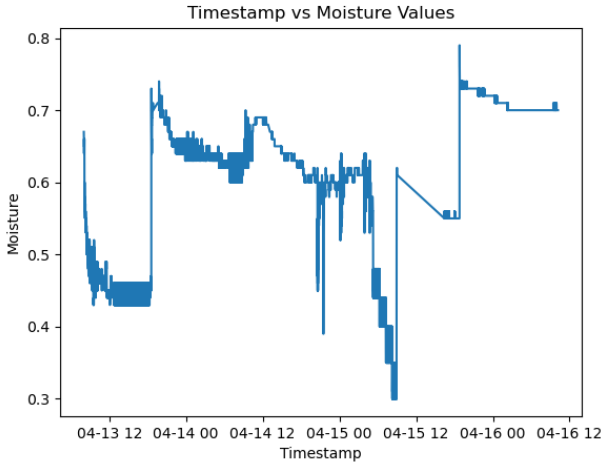


Figure 4: Moisture vs. Timestamp Graph

In the figure above the collected moisture data is represented in the graph with respect to when its collected.

7. Result

A.Software Simulation

The machine learning model is trained using the experimental data set values. For simulation purpose the Decision Tree model is implemented in the Jupyter Notebook and also for calculating accuracy of the trained model. The total of 104731 values had been used to train the machine learning model. Which include the time timestamp, moisture values (ranging from 0 to 1), irrigation status (1 for On and 0 for Off) and soil condition as either dry or wet (If moisture values is greater

International Scientific Journal of Engineering and Management
Volume: 02 Issue: 05 | May – 2023 DOI:
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
www.isjem.com

than 0.5 wet and If less that 0.5 dry). The accuracy of the implemented model has been calculated using python libraries such as sklearn and train_test_split modules. The software simulation also includes successfully receiving decision messages from the cloud to the NodeMCU ESP8266 microcontroller. It can seen using the MQTT lens tool, which will be used to subscribe to the "outTopic" topic on the cloud's MQTT service and see the received message responses.

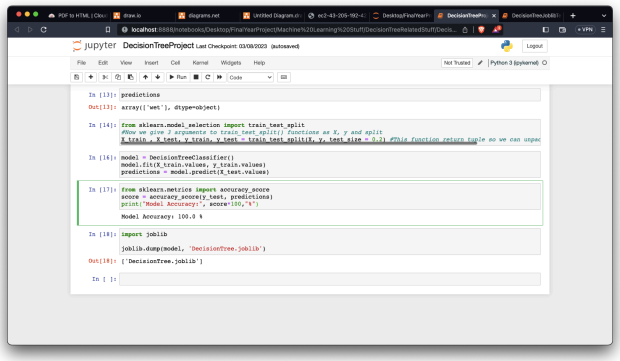


Figure 5: Accuracy of the Machine Learning Model

As in the figure above, the overall accuracy of the models predication has came to be 100%. That is due to we used only two parameters for training as moisture value and soil condition (Wet or Dry). This gives justification to the models accuracy.

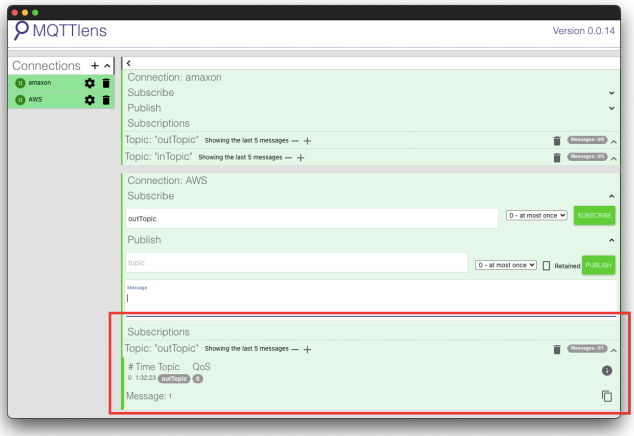


Figure 6: "outTopic" Message Received

In the figure above the message 1 has been received on the "outTopic" topic. So it is clear that the communication between the software part that is cloud and hardware part that is NodeMCU ESP8266 microcontroller is taking place as expected.

B. Hardware Result

The overall hardware including sensors, microcontroller and switching unit. By combining all of these as a single unit as per the schematics shown before completes the overall hardware part of this project.

So far majority of the work was located on cloud but actual application of the project lies on the ground. As planned in the hardware schematics all the components are successfully connected and all of them are up and running as the power supply is provided.

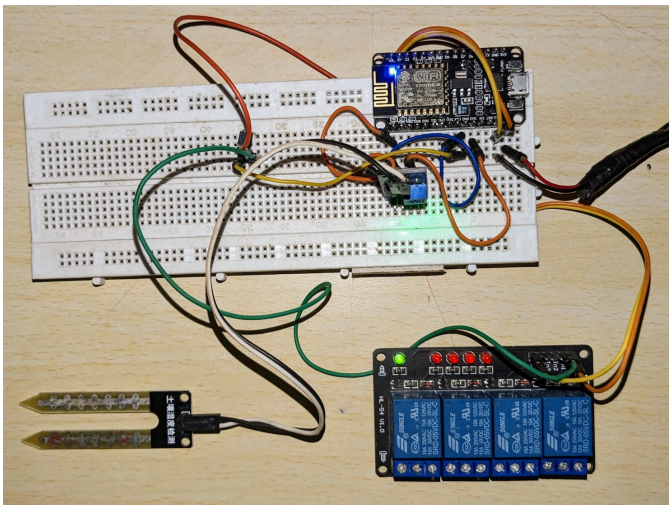


Figure 7: Hardware Implementation Result

As in the figure above LED's of NodeMCU ESP8266, FC-28 Hygrometer (Moisture) sensor and 5V relay switch are glowing which indicates the power has reached them and they are working as expected. When the probe of FC-28 Hygrometer (Moisture) sensor are dipped in the soil it started collecting moisture values and providing them to the NodeMCU ESP8266 microcontroller. The NodeMCU ESP8266 which is connected to the local Wi-Fi network will upload and publish those values on the 'inTopic' topic of the MQTT service running on the cloud platform. upon successfully making decision it will send back its decision to the NodeMCU ESP8266 microcontroller to either turn on or off the motor switch. Suppose the soil was dry and resulting decision is turn on then the NodeMCU ESP8266 microcontroller will implement that operation on the 5V relay switch circuit. In this hardware simulatio the hardware along with the software part was working and behaving as intended and expected.

CONCLUSION

The main purpose of this research is to provide a tool that farmers and cultivators may use to verify the compatibility of environmental conditions to boost crop growth and productivity in an efficient manner, resulting in the highest possible yield. Compared to the pricey soil testing equipment, this project provides virtually correct data. The model can be applied to any location. Once we upload the code in the microcontroller and on the cloud, it is not necessary to reprogram it later. Using this technology, many other projects can be conducted. By measuring the values of soil moisture, we can accordingly vary the amount of water needed to irrigate the soil in a particular season in a more efficient manner. Such a device is necessary for every farmer and would increase the water efficiency to its best. Thus, groundwater issues will also be resolved, and water wastage will also be reduced. The use of diesel pumps will also be reduced, saving non-renewable resources

ACKNOWLEDGEMENT

We would like to thank Mr. Sachin Trankatwar, our guide who has helped us through this whole journey, our project coordinator Mr. S. D. Biradar for guiding and being with us throughout the project development, Dr. B. H. Patil, Head of Department, Electronics and Telecommunication Engineering and our honorable Principal Dr. R. S. Bichkar, VPKBIET, Baramati for motivation and their cooperation in completing our research on the topic ”Use Of Digital Technology For Micro Irrigation System To Improve Water Efficiency of Irrigation Sector”.

We would like to express our gratitude to all of our teachers for their help and guidance. We would not have been able to complete this project without their help and cooperation. And we are very thankful for this opportunity that we got.

REFERENCES

1. J. Doorenbos, W. Pruitt, Guidelines for predicting crop water requirements. FAO Irrigation and Drainage Paper 24, Food and Agriculture Organization, UN, Rome, 1977.
2. Brouwer, C., Heibloem, M., Irrigation Water Management. Training Manual, vol. 3. Prov. Ed., 60 pp. Rome, Italy: FAO, 1986.
3. G. Kavianand, V.M. Nivas, R. Kiruthika and S. Lalitha, “Smart drip irrigation system for sustainable agriculture”, 2016 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR), Chennai, 2016, pp. 19-22
4. D. Mishra, A. Khan, R. Tiwari and S. Upadhyay, “Automated Irrigation System-IOT Based Approach”, 2018 3rd International Conference On Internet of Things: Smart Innovation and Usage (IoT-SIU), Bhimtal, 2018, pp. 1-4.
5. R. Varghese and S. Sharma, "Affordable Smart Farming Using IoT and Machine Learning," 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2018, pp. 645-650.
6. Parihar, Yogendra Singh, (2019) Internet of Things and Nodemcu A Review of use of Nodemcu ESP8266 in IoT products. 6. 1085.
7. Agavanakis, Kyriakos & Karpetas, George & Taylor, Michael & Pappa, Evangelia & Michail, Christos & Filos, John & Trachana, Varvara & Kontopoulou, Lamprini, “Practical machine learning based on cloud computing resources,” AIP Conference Proceedings 2123, 020050 (2019).
8. Shilpa Chandra, Samiksha Bhilare, Mugdha Asgekar and Ramya R. B., “Crop Water Requirement Prediction in Automated Drip Irrigation System using ML and IoT ,” 2021 International Conference on Nascent Technologies in Engineering (ICNTE 2021)

ISSN: 2583-6129

INTERNATIONAL SCIENTIFIC JOURNAL OF ENGINEERING & MANAGEMENT

An International Scholary || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

ACCEPTANCE CERTIFICATE

This is to certify that the submitted Paper titled

Use Of Digital Technology For Micro Irrigation System To Improve Water Efficiency Of Irrigation Sector(Using IoT, ML and Cloud Computing)

has been accepted for publication in

INTERNATIONAL SCIENTIFIC JOURNAL OF ENGINERING AND MANAGEMENT

ISSN: 2583-6129

Volume 02 Issue 05 May, 2023

Editor-in-Chief
e-Mail: editor@isjem.com
www.isjem.com

turnitin

Similarity Report ID: oid:7916:36635287

PAPER NAME
Final.pdf

AUTHOR
Shivam

WORD COUNT
14408 Words

CHARACTER COUNT
79198 Characters

PAGE COUNT
50 Pages

FILE SIZE
9.1MB

SUBMISSION DATE
Jun 1, 2023 10:15 AM GMT+5:30

REPORT DATE
Jun 1, 2023 10:17 AM GMT+5:30

18% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 7% Internet database
- Crossref database
- 17% Submitted Works database
- 4% Publications database
- Crossref Posted Content database