INTERNET-AIDED SMART IRRIGATION AND FERTILIZATION SYSTEM

MUHAMMED AKİF YENİKAYA

DECEMBER 2017

INTERNET-AIDED SMART IRRIGATION AND FERTILIZATION SYSTEM

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF

NATURAL AND APPLIED SCIENCES OF

IZMIR UNIVERSITY OF ECONOMICS

BY

MUHAMMED AKİF YENİKAYA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

IN THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DECEMBER 2017

M.S. THESIS EXAMINATION RESULT FORM

Approval of the Graduate School of Natural and Applied Sciences

[illegible]

Prof. Dr. Abbas Kenan Çiftçi
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

[illegible]

Assoc. Prof. Dr. Cem Evrendilek
Head of Department

We have read the thesis entitled **"Internet-Aided Smart Irrigation and Fertilization System"** completed by **MUHAMMED AKİF YENİKAYA** under supervision of **Assoc. Prof. Dr. Süleyman Kondakcı** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assoc. Prof. Dr. Süleyman Kondakcı
Supervisor

Examining Committee Members	**Date:** 16.02.2018
Assoc. Prof. Dr. Süleyman Kondakcı Dept. of Engineering, İUE	[illegible]
Asst. Prof. Dr. Kaya Oğuz Dept. of Engineering, İUE	[illegible]
Asst. Prof. Dr. Tuğkan Tuğlular Dept. of Engineering, İZTECH	[illegible]

ABSTRACT

INTERNET-AIDED SMART IRRIGATION AND FERTILIZATION SYSTEM

MUHAMMED AKİF YENİKAYA
M.S. in Computer Engineering
Graduate School of Natural and Applied Sciences
Supervisor: Assoc. Prof. Dr. Süleyman Kondakcı
December 2017

In this study, it has been aimed to develop and apply an Internet-based smart soil maintenance system in order to increase productivity in automated irrigation and fertilization systems used in agriculture. The system we aim to develop performs irrigation and fertilization processes automatically with the lowest expenditure. Fertilization and irrigation can be done by controlling the humidity levels and pH values of the soil in fields and greenhouses, and communicating with the control centre via the Internet. The control centre nourishes the soil automatically on an optimal level with the stuff it needs by observing the soil values constantly with sensors. A system containing software and hardware used for this purpose has been designed to be manageable via the Internet.

Communication between the systems used in this study has been maintained through Internet. After determining the necessary water and fertilizer amounts, the fertilization values have been adjusted in the liquid tank. After that, the mixture has been distributed to the soil automatically by valves used in each section of the field. Besides, by means of the web interface of the system, it has been possible to access the measured data easily in real-time.

Keywords: Field, Greenhause, Automation, Control, Sensor, Internet.

ÖZ

İNTERNET DESTEKLİ AKILLI SULAMA VE GÜBRELEME SİSTEMİ

MUHAMMED AKİF YENİKAYA
Bilgisayar Mühendisliği, Yüksek Lisans
Fen Bilimleri Enstitüsü
Tez Danışmanı: Doc. Dr. Süleyman Kondakcı
Aralık 2017

Bu çalışmada; tarlalarda ve seralarda kullanılmakta olan sulama ve gübreleme otomasyon sistemlerinde verimliliği artırmaya yönelik akıllı bir otomatik toprak bakım sistemin geliştirilmesi, uygulanması amaçlanmıştır. Hedeflenen sistem gereken en az maliyet ile sulama ve gübreleme işlemlerini otomatik olarak gerçekleştirmektedir. Tarla ve seralardaki toprağın; nem ve pH değerlerini denetleyerek, aynı zamanda internet üzerinden merkezi bir sisteme komutlar göndererek gübreleme ve sulanması yapılabilmektedir. Merkezi sistem sürekli olarak toprak değerlerini sensörlerle gözetleyerek toprağın ihtiyaç duyduğu metaryaller ile otomatik ve optimal bir ölçüde toprağı beslemeyi amaçlamaktadır. Bu amaçla kullanılabilecek yazılım ve donanımdan oluşan bir sistem internet üzerinden kontrol edilebilecek şekilde tasarlanmıştır.

Üzerinde çalışılan sistemler arasındaki iletişim, internet teknolojisi kullanılarak sağlanmıştır. Gereken su ve gübre tespitinin ardından depo içerisinde gübreleme değerleri istediğimiz şekilde ayarlanarak, her bir bölge için kullanılacak valfler sayesinde toprağa otomatik olarak dağıtılması sağlanmıştır. Ayrıca, otomasyon sisteminin web ara yüzü sayesinde, ölçme bilgilerine kolaylıkla gerçek zamanlı erişilebilmesi mümkün olmuştur.

Anahtar Kelimeler: Tarla, Sera, Otomasyon, Kontrol, Sensör, İnternet.

ACKNOWLEDGEMENT

I present my special thanks to Assoc. Prof. Dr. Süleyman Kondakcı for his contribution and support during my research. I am also thankful to my family for their support during my master's study.

TABLE OF CONTENTS

Chapter 1

Introduction

The aim of this study is to improve productivity of fields and greenhouses by automating fertilizing and watering by use of automated systems and to eliminate disadvantages of wired measurements and cable traffic through the use of wireless technology. As automation systems are built upon a long research process and experiences, their installation and integration costs are high. Since the installation costs of these systems in fields and greenhouses are too high, they are not widely used in this sector. In the past, qualified operators to use those systems were needed. Today, however, these systems are spreading due to wide and easier use of computers. For that reason, the cost of automation installation in the fields can be amortized in a short time.

Digital electronics and sensor technologies have efficiently been used in modern greenhouses and fields. By using those technologies, it is aimed to save on the consumption of energy, water and fertilizers by creating more efficient operation conditions. Wired measurement systems and cable traffic may cause important problems. Effective solutions and control strategies are needed to have an efficient data management in greenhouses and fields. [1].

The measurement process deals with the collection of quantitive information to compare reference parameters related to physical values [2].

Chapter 2

Related Work

The importance of water for living creatures has increased the studies and researches conducted on this topic. The study entitled "Automated Irrigation System with Humidity Sensors" made by Dukes has shown 50% of water-saving [3].

Kırnak presents a computer controlled automatic dripping and irrigation system to measure soil humidity which revealed that insufficient irrigation problem can be solved by controlled watering [4].

Millaa and Kishb designed an efficient irrigation and erosion prevention system with a microchip-based infrared sensor. Irrigation quantity and timing were recorded by the system and these data were transferred to a computer for a detailed analysis when needed [5].

In a postgraduate thesis in University of Süleyman Demirel, Graduate School of Natural and Applied Sciences, an automation of rooting greenhouses were designed for nursery tree reproduction. In this study, an automation system was built in order to be used in rooting greenhouses for the purpose of nursery stocks production. In the automation, the Internet and the Internet communication techniques were used. The automation system was aimed to be feasible and low-cost [6].

In University of Çukurova, Graduate School of Natural and Applied Sciences, a computer-controlled, Internet-aided greenhouse automation was designed as a postgraduate thesis. In that study, the emphasis was laid on the computer-controlled, Internet-aided automation of systems such as irrigation and acclimatization. The system was aimed to be feasible and low-cost [1].

In their study called "Design and Implementation of Seeding and Fertilizing Agriculture Robot", a system was designed to control feeding, soil pH values, temperature, and humidity values. For this purpose, they made use of computers and the Internet [7].

Chapter 3

Background Work

3.1 Automation versus Non-Automated Processing

An automation is a control system that processes a production without human intervention, controls it and uses a feedback system inventively [8].

Automation deals with sharing the task between the human and the machine. The percentage of the total work determines the level of automation. Automation systems with higher human labour is called semi-automation, whereas the systems with more intense machine work is called full-automation. Prior to the Industrial Revolution, “production only” was far from being sufficient. Making the production fast, standardized, safe and fruitful has become a necessity in a world that has turned into an open market. The solution to this requirement in the industry is, without doubt, automation. Automation does also provide a production standard and a decrease in expenditures.

In today's economic conditions, producing goods fast, safely, competitively, and with quality has become a vital issue. The ones who want to set up a system need to know the type of automation system they require by assessing the pros and cons of the systems. Quality increase, lower costs and time saving can be named as plus aspects that these systems could bring. The greatest disadvantage of these systems is the high cost of the first time installation, but in most automation systems, these costs pay themselves off in the long run.

3.2 Automation Systems in Agriculture

These are systems that are capable of adapting the changes in the soil with peripheral units by taking into consideration the power economy, deciding on the needs of plants in the roots of plants by saving water, and managing the plant growth and nutrition strategically in terms of the total management. These automation systems consist of software and hardware parts. Hardware consists of computers, programmable logic controller (PLC) systems, microcontroller systems, or a controller system which is the mixture of these systems. Software controls the hardware and makes the decision procedures.

3.3 Non-Automated Systems in Agriculture

Non-automated field and greenhouse systems are systems that the product quality and success rate are left in the hands of the owner's experience and do not use electronic devices apart from those that are used for humidity measurement.

As no electronic systems are used in these types of fields, the field must be controlled and certain operations must be carried out in specific time intervals manually.

3.4 Emerging Microcontroller Systems

NodeMCU LoLin ESP8266 microprocessor is used to manage the humidity and pH data from the soil, to control on and off of the valves, to measure the liquid level in the tank and to transfer these data via Internet. Arduino Uno is used as the control centre, which optimize the fertilization values, controls motors and valves, and transfers the fertilization measurement data for monitoring.

3.4.1 Wireless Data Collection

NodeMCU LoLin ESP8266 has been used for wireless data collection. LoLin is a development card which has NodeMCU installed on and accommodates an ESP8266 Wi-Fi module. As it is developed by using the SDK of ESP8366, it supports GPIO, PWM, IIC, 1-Wire and ADC connections without any need for an extra microcontroller. CP2102 USB serial adapter is integrated to it [10].

NodeMCU is an electronic node which is a little bigger from a coin. It is open-source and cheap. It runs on low voltage energy. It has multiple ports. By using these ports, you may manage other electronic components that have been connected.

The communication with LoLin over Internet is done by using http libraries in the developed code. Thus, devices can be turned on and off remotely. Lua script is used as the programming language. Despite this, however, it can be programmed with the language that Arduino IDE and Arduino use. It can be connected to a computer easily, can be programmed and data communications can be set up. It is simple and low in cost. Moreover, it is a smart device and it is ready to be used and make Wi-Fi connections. Fig. 3.1 shows the pins diagram of NodeMCU LoLin ESP8266. (see Appendix A.1 for more information about the NodeMCU LoLin ESP8266).

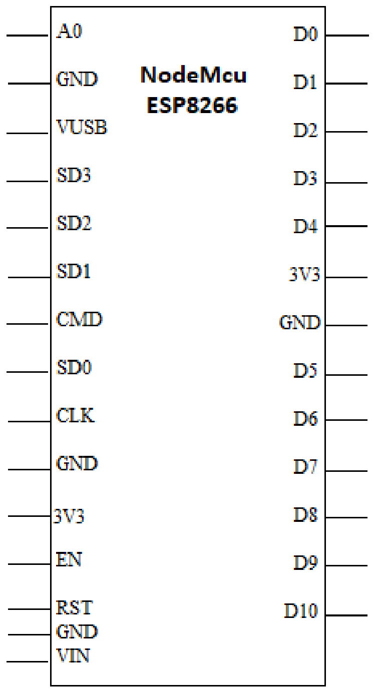


Fig. 3.1. NodeMCU LoLin ESP8266 Pins Diagram

C/C++ and their derivatives and similar language structures are often used with lolin. Script language is also a widely used language but C/C++ derivatives are more dominant. For this reason, instead of learning Lua and its language structure all over again, it would be wiser to develop NodeMCU by using C/C++ and Arduino IDE.

3.4.2 Central Computing Unit

Arduino was used for central computing unit. Arduino processors are today the most popular open platforms used to obtain a large number of applications [9]. Fig. 3.2 shows the pins diagram of Arduino Uno. (see Appendix A.2 for more information about the Arduino Uno).

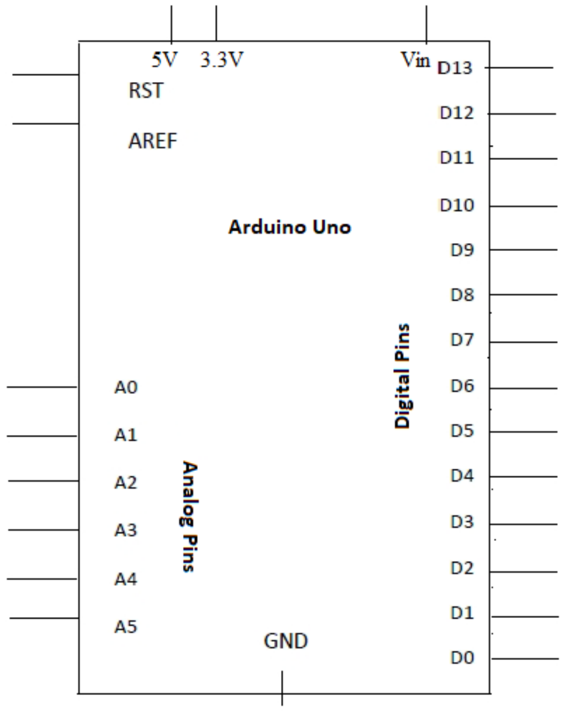


Fig. 3.2. Arduino Uno Pins Diagram

3.4.2.1 Arduino Components

Ardiuno's basic components consist of Ardiuno integrated development environment (IDE), Arduino bootloader (Optiboot), Arduino libraries, AVRDude the software that programmes microcontrollers on Arduino and a compiler (AVR-GCC).

Arduino software is formed with an integrated development environment (IDE) and libraries. The IDE is written in Java and based on the medium of a language called processing. The libraries, on the other hand, are written in C and C++ and complied with AVR-GCC and AVR Lib.

Optiboot component is Ardiuno's bootloader component. This component is the one that enables the microcontroller on Ardiuno cards to be programmed. The most important component that makes Ardiuno so high-in-demand is Arduino libraries which enables anyone to programme without requiring to have any detailed knowledge on microcontrollers. Arduino libraries comes with the development medium and are indexed under 'libraries' folder. By taking a look at the codes, it is possible to see the structure of the libraries and how the microcontrollers are programmed.

Finally, AVRDude component is used to program the compiled codes. Programming can easily be done with Ardiuno libraries, analogue and digital data can be processed. By using the signals from the sensors, robots and systems that interact with their environment can be designed [10]. Ardiuno has various cards and modules designed to satisfy different needs, and various projects can be developed. As our Ardiuno microcontroller card does not have an integrated Wi-Fi module, we also need a Wi-Fi shield to establish an Internet connection.

3.4.2.2 Why Arduino?

- Arduino can be connected with many different types of sensors.
- As it has analogue inputs and digital outputs, we do not need an extra module for the analogue inputs that we acquire from the sensor.
- There are many additional hardware products that can work with Arduino. Ardiuno Wi-Fi Shield is an example of that.
- The developer platform and the drivers of Arduino are easy to install.
- It has a wide-ranged library, therefore, complex processes can be handled.
- Programs can work quite fast as they do not run on any other platform and it does not have a language that needs interpretation [11].

3.5 Network Architecture

With the advancements in technology, there are increasingly more applications that provide an opportunity to supervise from remote points. In the infrastructure of these applications, different communication protocols and the Internet is used depending on the needs such as the distance and the data signalling rate [12].

There are two different network architecture that can be used for the implementation of entire system, dedicated local area and cloud based system, respectively.

3.5.1 Local Area Network

As known, a local area network connects many different devices and makes them communicate with one another. Via this structure, different devices can use the same sources (i.e. multiple computers can access to a common printer), and they may send and receive data through different protocols.

3.5.1.1 Advantages of Local Area Network

Its advantages are:

- Resource Sharing
- Software Applications Sharing
- Cheap Communication
- Centralized Data
- Internet Sharing

3.5.1.2 Disadvantages of Local Area Network

Its disadvantages are:

- High Setup Cost
- LAN Maintenance Job
- Covers Limited Area

3.5.2 Cloud Technology

In its most simple definition, cloud technology is an online storage service that does not need any installation, and provides operational convenience. Along with all our applications, programs and data on the Internet are stored on a virtual machine or with its most common name ‘cloud’. The entire service, by which we can access to this information, programs and data in any location having an Internet connection is called cloud technology.

3.5.2.1 Advantages of Cloud Technology

Cloud technology has a fast ease of handling, allows the cloud to interact with the computer, and enables the user to establish communication with the computer as well as connection to the interface.

Due to cloud informatics, it is possible to reach every information everywhere and by using any kind of communication device (PC, Mac, iPhone, Android or Blackberry). Without hardware-related problems, it offers a high accessibility due to the virtual computer which works faster than a physical server. It uses a flexible structure that does not need a replacement of memory and/or hard drive and it is environment friendly (saves electricity and space)and can be seen amongst many advantages of cloud informatics that draw attention in the first place [13].

To summarize, the maintenance of cloud informatics applications is very easy, because there is no need for it to be installed on every user's computer and microprocessors, and it enables them to access the system from different locations.

3.5.2.2 Disadvantages of Cloud Technology

A reliable Internet connection is needed to access the data stored on the Internet, so problems may arise when Internet connection is not available. Besides, when you are connecting from an unfamiliar location and the Internet connectivity speed is low, the data exchange will be slower as a result.

Chapter 4

System Specifications and Requirements

4.1 Features of The Proposed System

The system that is worked on is aimed to;

- be installed easily, structurally flexible as well as feasible and at low-cost.
- be able to enable the system to make adaptations not just only for a field or a type of crop in the field, but also for the new crops by re-adjusting the system even when all the crops in the field are replaced.
- be compatible with the further improvements, not just with the original form of the field or the greenhouse.
- connect to the system from anywhere to reach real-time data.

4.2 Operation of The System

After establishing Internet connections for every microcontrollers, the system shows the status of sensors and circuit components on the system interface in real-time.

After the identification of the components, the next stage is the process of entering the field and greenhouse controlling parameters to the system. This section is designed to enter commands. It allows to create different patterns according to the plant species due to its flexible structure.

After specifying the controlling parameters, the program starts running and then, it reads the data from the sensors, Thereafter, the program converts the data gathered to digital unit. It evaluates the data in order to check whether the data match the predefined parameters (i.e. pH level). If no match, the code makes devices to mix new fertilizations elements, and starts a new fertilization process. At the same time, the system displays the results. Thus, the status of the sensors, circuit components and the microcontrollers can be viewed in real-time through the Internet.

The program also enables the control to be made manually on the interface. Sending a direct information to the desired output device, on and off processes can be handled. Thus, if there are tasks to do, they can be fulfilled via Internet without the need to go to the field or greenhouse.

4.3 Application of The System

Our system has a flexible structure, which can work with or without Internet. In case of Internet cut, the system carries on automatically. In the formation of the automation system, two types of work have been carried on. One is the system with Internet of Things (IoT) builder, and the other is the system without Internet of Things (IoT) builder.

4.3.1 Application of The System with IoT Builder

It is easier to be adjusted according to the specifications of the plant types due to the triggers that is on the system. Fig. 4.1 shows below the trigger screen of the system.

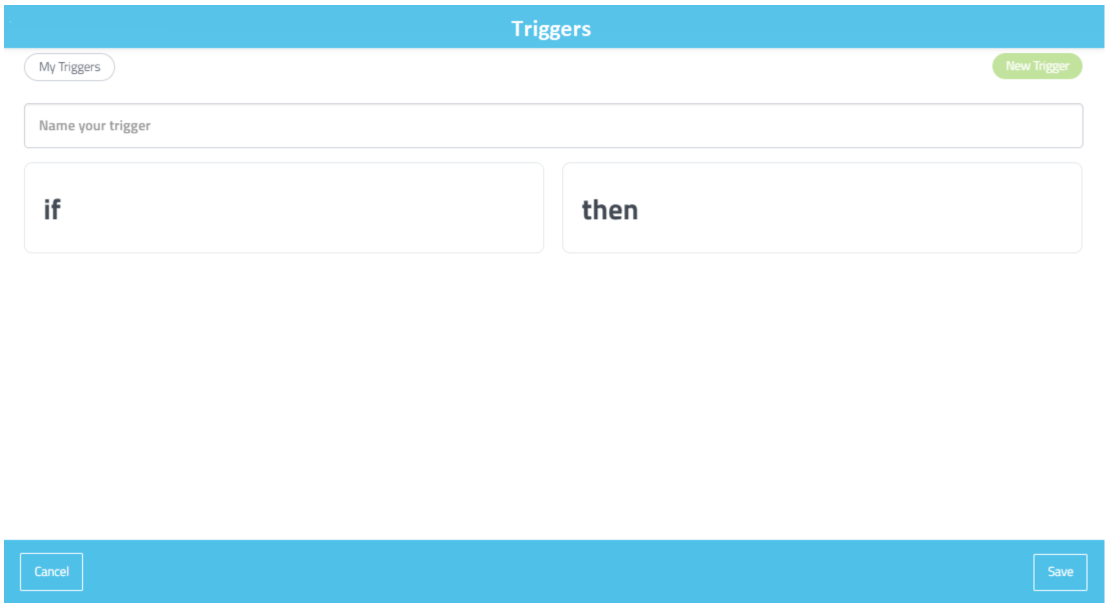
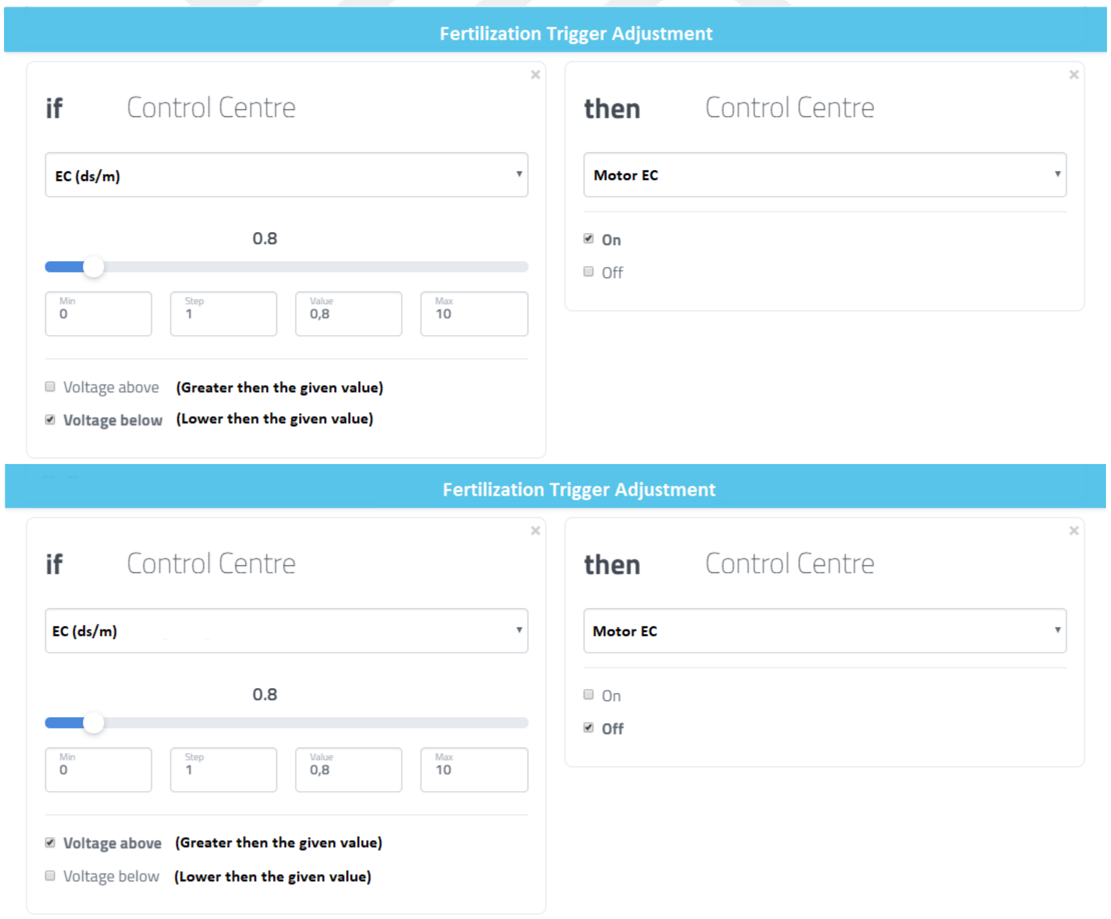


Fig. 4.1. Trigger Screen

4.3.1.1 Configuration of The Fertilization Parameters

The data gathered from the EC sensor that is in the liquid tank is processed by our device in real-time. The values acquired make it possible to stay at the desired point due to the triggers on the system.

If we take 'lettuce' as an example, we can see that the average EC demand is 0.8. To make adjustments on these parameters, it is possible to make the adjustments by clicking the trigger option. Fig. 4.2 shows how to adjust the EC values.



Fertilization Trigger Adjustment

if Control Centre

EC (ds/m)

0.8

Min	Step	Value	Max
0	1	0,8	10

Voltage above (Greater then the given value)
[✓ Voltage below] (Lower then the given value)

then Control Centre

Motor EC

[✓ On]
Off

Fertilization Trigger Adjustment

if Control Centre

EC (ds/m)

0.8

Min	Step	Value	Max
0	1	0,8	10

[✓ Voltage above] (Greater then the given value)
Voltage below (Lower then the given value)

then Control Centre

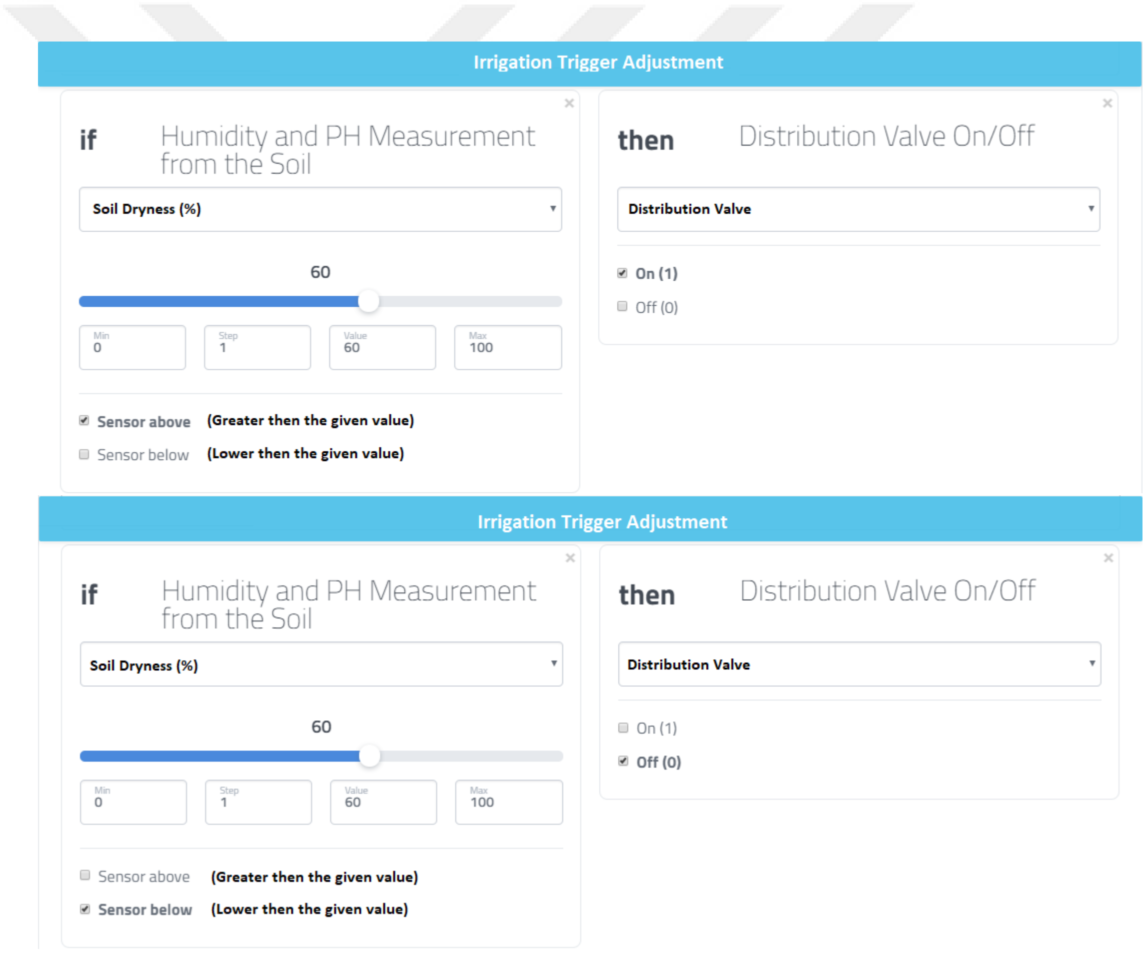
Motor EC

On
[✓ Off]

Fig. 4.2. Fertilization Trigger Adjustment

4.3.1.2 Configuration of the Irrigation Parameters

In the adjustment of the irrigation values part, the humidity value of the soil is transferred to the system. Due to the triggers on the system, it is possible to adjust the humidity of the soil to the desired value. Fig. 4.3 shows how to adjust the soil humidity values.



Irrigation Trigger Adjustment

if Humidity and PH Measurement from the Soil

Soil Dryness (%)

60

Min 0 | Step 1 | Value 60 | Max 100

✓ Sensor above (Greater then the given value)

☐ Sensor below (Lower then the given value)

then Distribution Valve On/Off

Distribution Valve

✓ On (1)

☐ Off (0)

Irrigation Trigger Adjustment

if Humidity and PH Measurement from the Soil

Soil Dryness (%)

60

Min 0 | Step 1 | Value 60 | Max 100

☐ Sensor above (Greater then the given value)

✓ Sensor below (Lower then the given value)

then Distribution Valve On/Off

Distribution Valve

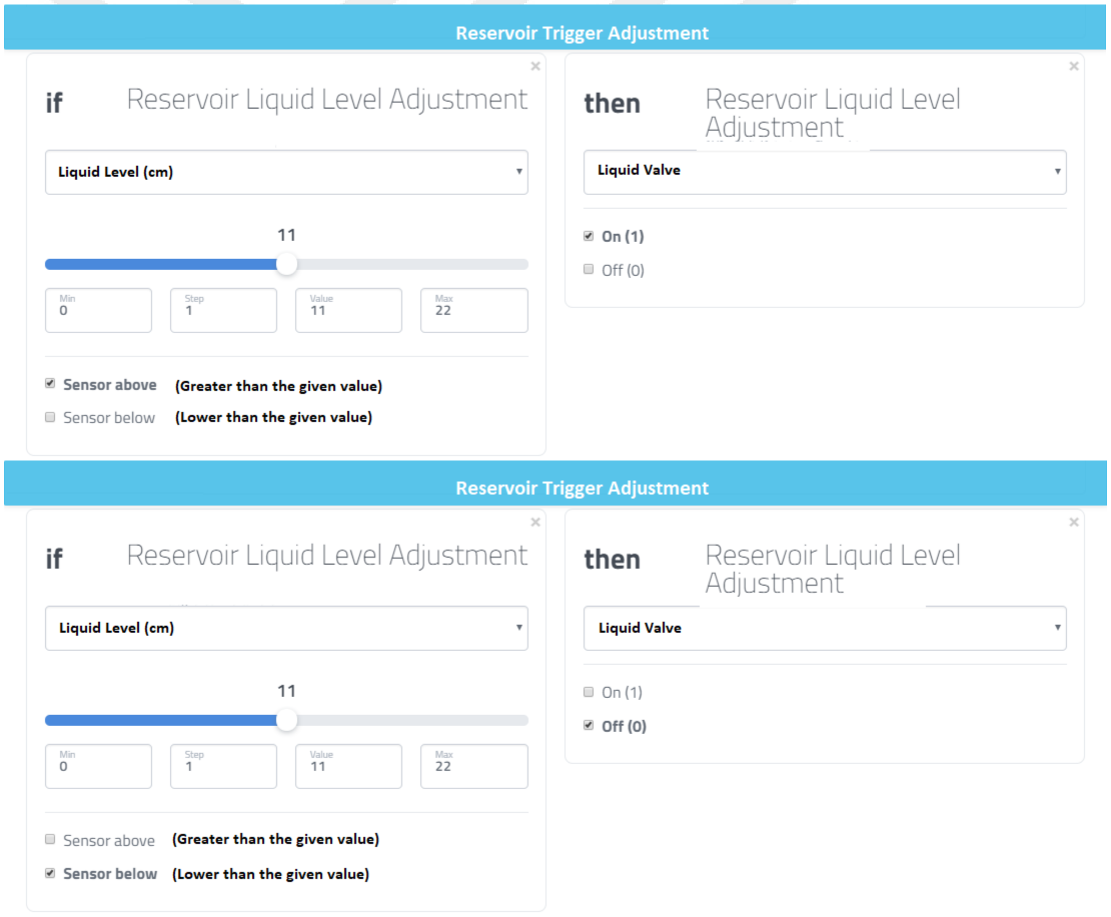
☐ On (1)

✓ Off (0)

Fig. 4.3. Irrigation Trigger Adjustment

4.3.1.3 Configuration of the Reservoir Liquid Level

The ultrasonic sensor planted in the liquid tank measures the liquid level in the reservoir. The measured sensor data can be processed and transferred to the system in real-time. Keeping the water level in the reservoir at the desired point is possible in this way, due to the triggers in the system. Fig. 4.4 shows how to adjust the reservoir water level using trigger.


Reservoir Trigger Adjustment

if Reservoir Liquid Level Adjustment

Liquid Level (cm)

11

Min 0 | Step 1 | Value 11 | Max 22

[✓ Sensor above] (Greater than the given value)

Sensor below (Lower than the given value)

then Reservoir Liquid Level Adjustment

Liquid Valve

[✓ On (1)]

Off (0)

Reservoir Trigger Adjustment

if Reservoir Liquid Level Adjustment

Liquid Level (cm)

11

Min 0 | Step 1 | Value 11 | Max 22

Sensor above (Greater than the given value)

[✓ Sensor below] (Lower than the given value)

then Reservoir Liquid Level Adjustment

Liquid Valve

On (1)

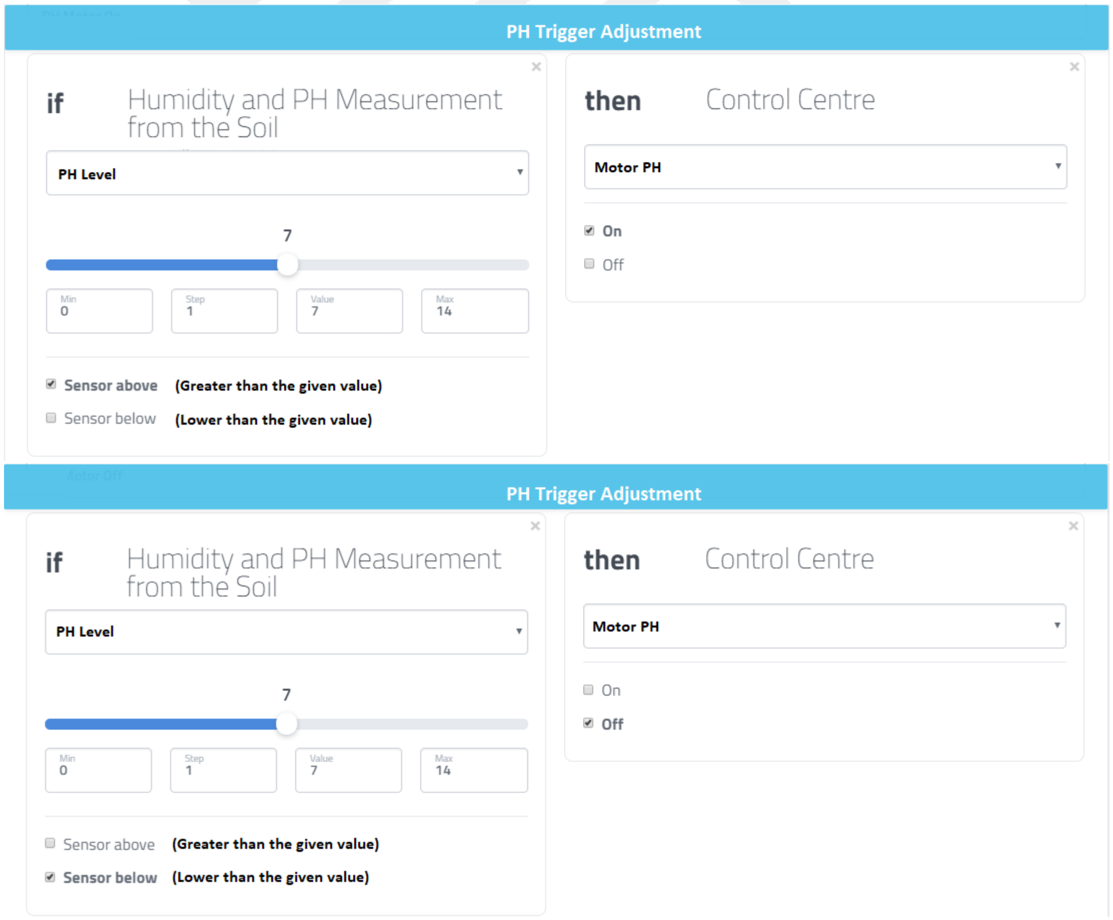
[✓ Off (0)]

Fig. 4.4. Reservoir Trigger Adjustment

4.3.1.4 Configuration of the Soil Acidity Level

The data gathered from the pH sensor planted in the soil is processed by our device and transferred to the control centre in real-time. Due to the triggers, the values gathered make it possible to remain stable at the desired values.

If we take 'lettuce' as an example, we see that the average pH value demand is 7. The identifications here are as the following: "Run the motor that is connected to the acid tank if the pH value is above 7". Fig. 4.5 shows how to adjust the soil acidity level using trigger option.



PH Trigger Adjustment

if Humidity and PH Measurement from the Soil

PH Level

7

Min 0 | Step 1 | Value 7 | Max 14

[✓ Sensor above] (Greater than the given value)
Sensor below (Lower than the given value)

then Control Centre

Motor PH

[✓ On]
Off

PH Trigger Adjustment

if Humidity and PH Measurement from the Soil

PH Level

7

Min 0 | Step 1 | Value 7 | Max 14

Sensor above (Greater than the given value)
[✓ Sensor below] (Lower than the given value)

then Control Centre

Motor PH

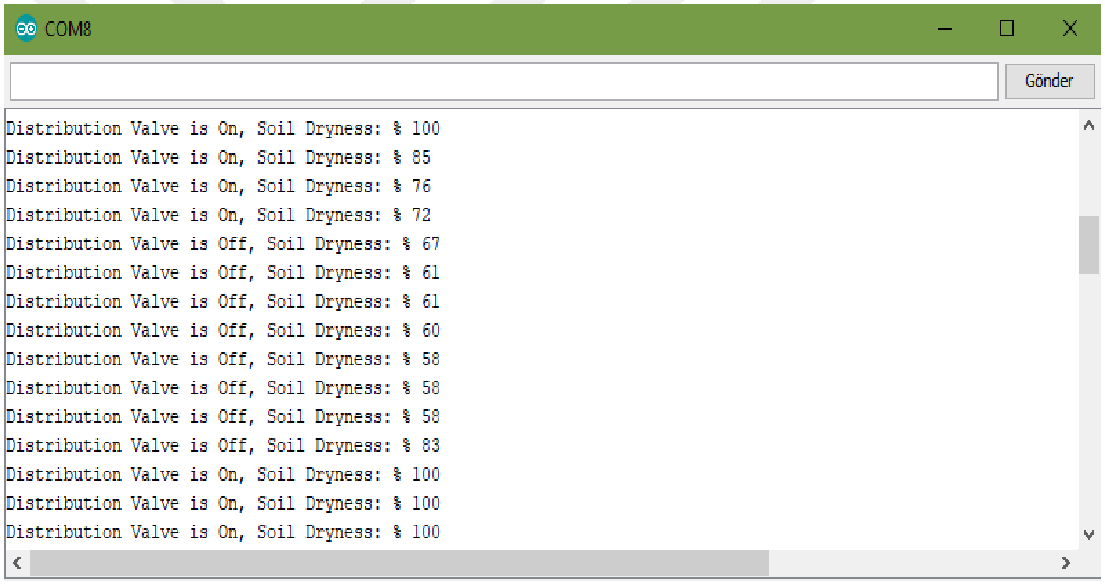
On
[✓ Off]

Fig 4.5. PH Trigger Adjustment

4.4.2 Application of The System without IoT Builder

In the system working without IoT builder, fertilization and irrigation have been supplied automatically by means of the software installed in micro-controller.

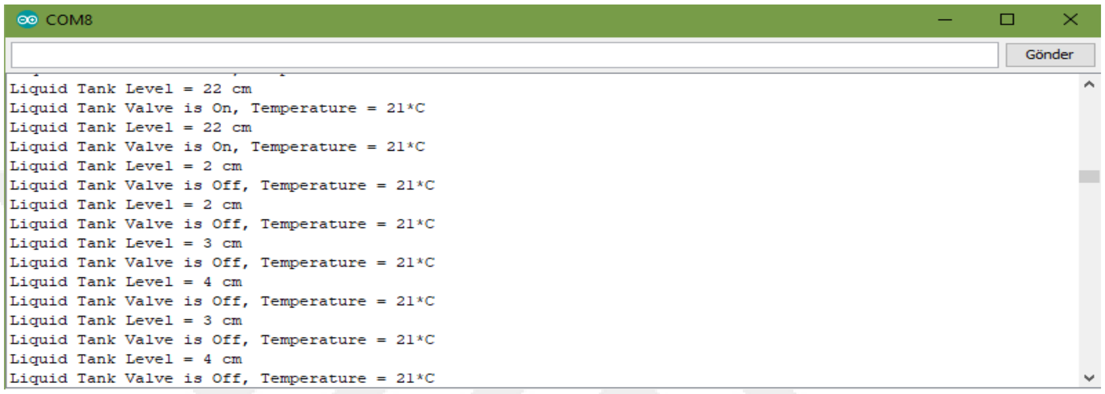
Serial Monitor of the humidity and distribution valve circuit is as the image shown in Fig. 4.6.



```
Distribution Valve is On, Soil Dryness: % 100
Distribution Valve is On, Soil Dryness: % 85
Distribution Valve is On, Soil Dryness: % 76
Distribution Valve is On, Soil Dryness: % 72
Distribution Valve is Off, Soil Dryness: % 67
Distribution Valve is Off, Soil Dryness: % 61
Distribution Valve is Off, Soil Dryness: % 61
Distribution Valve is Off, Soil Dryness: % 60
Distribution Valve is Off, Soil Dryness: % 58
Distribution Valve is Off, Soil Dryness: % 58
Distribution Valve is Off, Soil Dryness: % 58
Distribution Valve is Off, Soil Dryness: % 83
Distribution Valve is On, Soil Dryness: % 100
Distribution Valve is On, Soil Dryness: % 100
Distribution Valve is On, Soil Dryness: % 100
```

Fig 4.6. Serial Monitor of The Humidity and Distribution Valve Circuit

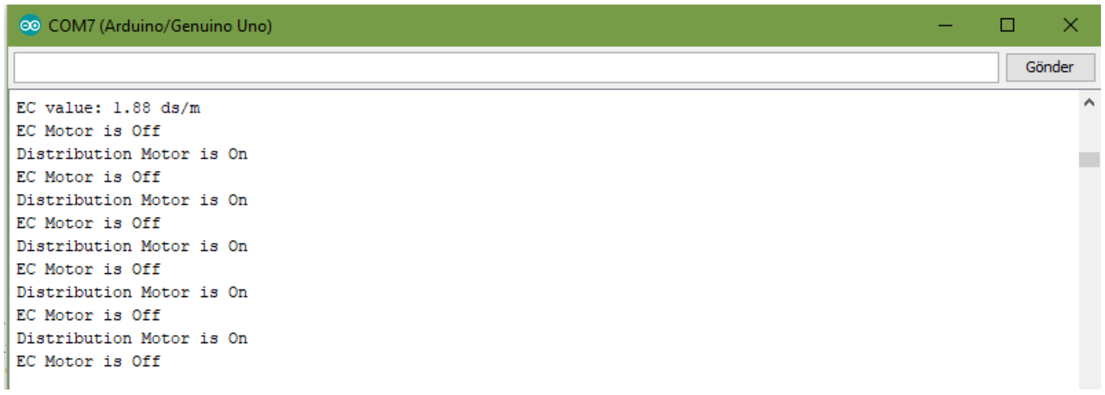
Serial Monitor of the the liquid level adjustment circuit is as the image shown in Fig. 4.7.



```
Liquid Tank Level = 22 cm
Liquid Tank Valve is On, Temperature = 21*C
Liquid Tank Level = 22 cm
Liquid Tank Valve is On, Temperature = 21*C
Liquid Tank Level = 2 cm
Liquid Tank Valve is Off, Temperature = 21*C
Liquid Tank Level = 2 cm
Liquid Tank Valve is Off, Temperature = 21*C
Liquid Tank Level = 3 cm
Liquid Tank Valve is Off, Temperature = 21*C
Liquid Tank Level = 4 cm
Liquid Tank Valve is Off, Temperature = 21*C
Liquid Tank Level = 3 cm
Liquid Tank Valve is Off, Temperature = 21*C
Liquid Tank Level = 4 cm
Liquid Tank Valve is Off, Temperature = 21*C
```

Fig 4.7. Serial Monitor of The Liquid Level Adjustment Circuit

Serial Monitor of the the control centre is as the image shown in Fig. 4.8.



```
EC value: 1.88 ds/m
EC Motor is Off
Distribution Motor is On
EC Motor is Off
Distribution Motor is On
EC Motor is Off
Distribution Motor is On
EC Motor is Off
Distribution Motor is On
EC Motor is Off
Distribution Motor is On
EC Motor is Off
```

Fig 4.8. Serial Monitor of The Control Centre

Chapter 5

System Components, Materials and Methods

The purpose of the software registered on the system is to create necessary control signals by comparing the data in the frame of pre-determined parameters.
Fig. 5.1 shows the overall structure of the system.

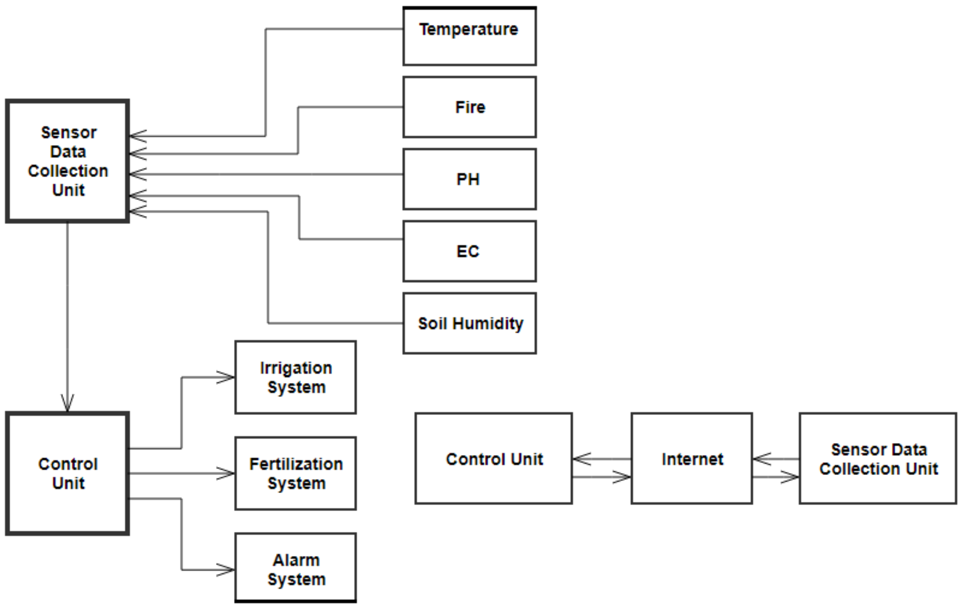


Fig. 5.1. Overall System Structure

The control centre system that provided motor control, on the other hand, turns the motors on or off, depending on the data it receives from the system. It allows manual motor control and the system to turn the EC variants in the liquid tank into numeric data, and at the same time it sends the calibrated data to the Internet. It can also detect any fire. Fig. 5.2 shows the block diagram of the control centre.

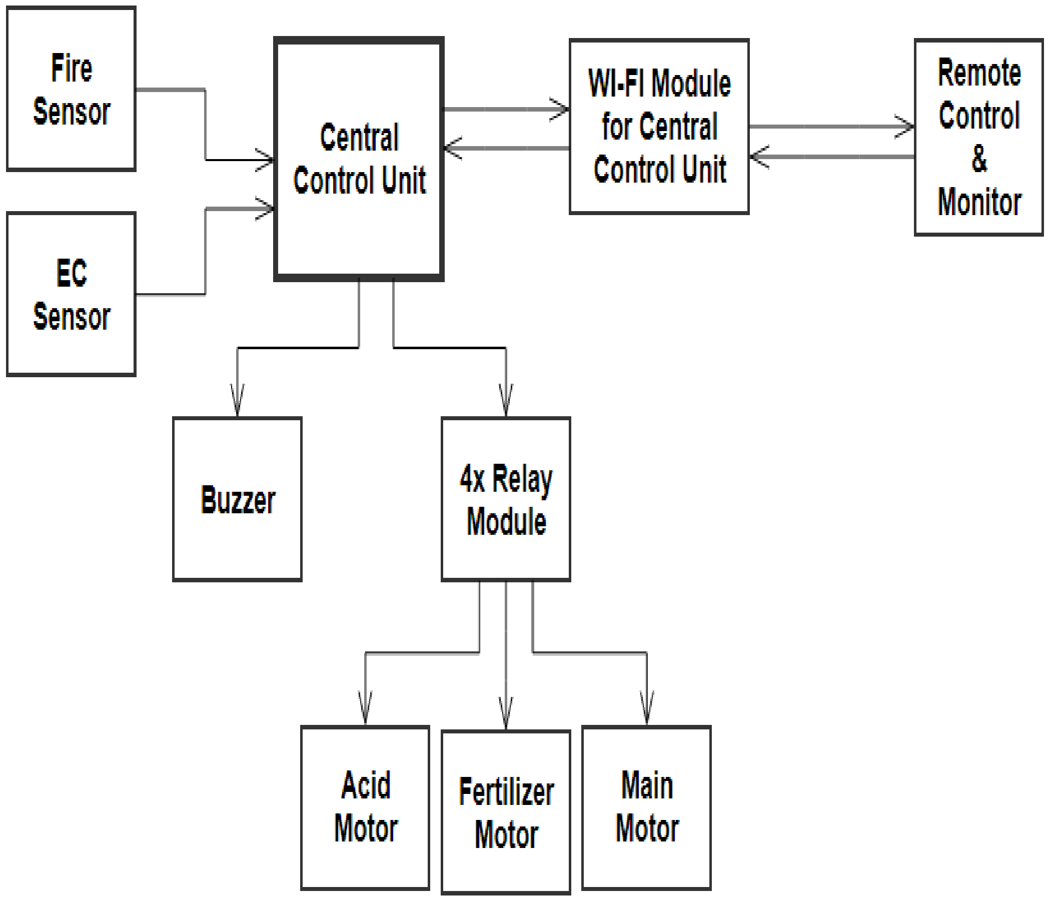


Fig. 5.2. Control Centre's Block Diagram

The system turns the humidity value and pH level variants in the soil into numeric data, and at the same time, it also sends the calibrated data to the monitor. Fig. 5.3 shows the block diagram of the pH and Soil Humidity detect system.

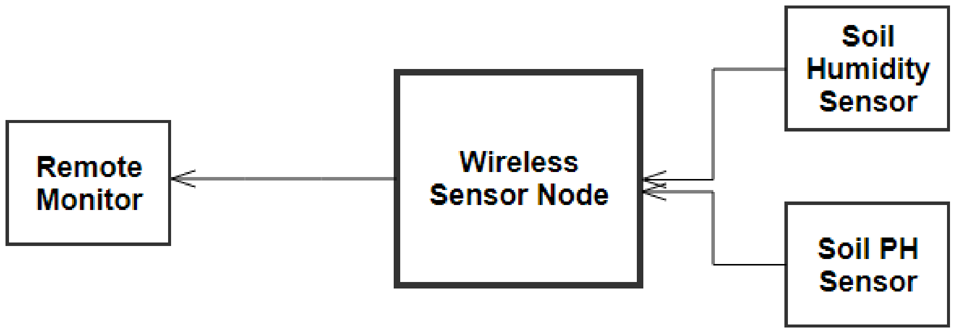


Fig. 5.3. PH Level and Soil Humidity Value Detect System's Block Diagram

The system with valve control, on the other hand, turns the valve on or off depending on the data it receives from the sensors. It reports the changes to the monitor at the same time. Fig. 5.4 shows the block diagram of the valve control system.

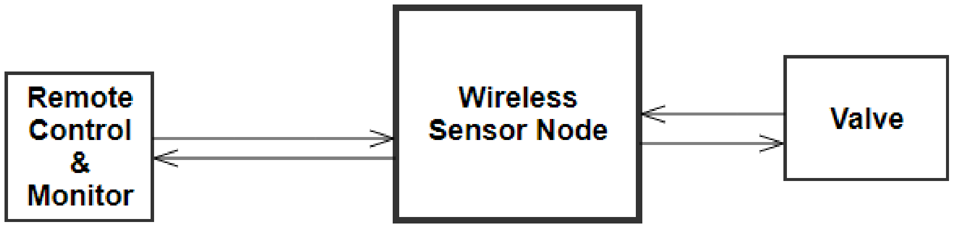


Fig. 5.4. Valve Control System's Block Diagram

The system is composed of valve, liquid level detection sensor and temperature sensor. The microcontroller turns the valve on or off depending on the data from the sensor. It reports the changes to the monitor and at the same time, it also allows manual valve control. Fig. 5.5 shows the block diagram of the reservoir control system.

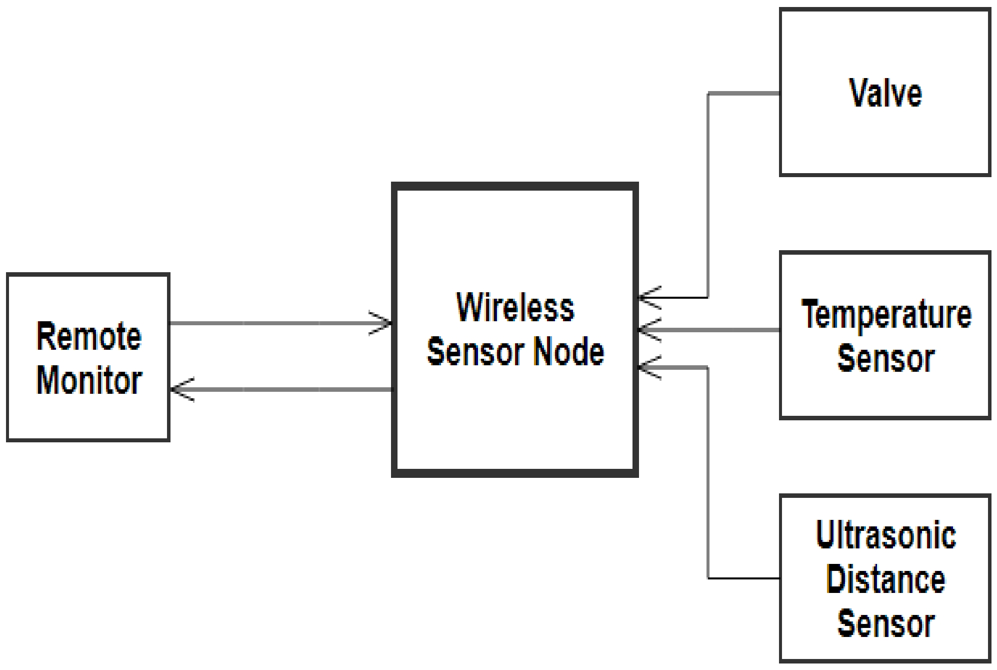


Fig. 5.5. Reservoir Control System's Block Diagram

5.1 Sensors

Sensors are devices that show electrical behaviour changes which provide the measurement of values such as temperature, distance, speed, velocity, liquid flow, light density, voltage, current, resistance, power and torque.

Sensors pick up physical changes in the greenhouses and fields, and provide necessary information. The analogue or digital data gathered from these units are transferred to the system that controls the automation process. Soil humidity and electrical conductivity in the field can be measured with the sensors. As most of the data gathered from the sensors are in analogue form, converters make them processable for digital systems. The first phase of running automatic control systems is the gathering of the data that provides a base for controlling.

5.1.1 Temperature Sensor

Since almost every chemical or biological process and reaction depends on the temperature, significant damages occur resulting from ill-conditioned temperature when the temperature levels are at a critical stage. For this reason, a unit to detect the temperature values is needed to measure the temperature ratio. Fig. 5.6 shows below the connection of temperature sensor to the wireless data collection unit. (see Appendix A.4 for more information about the Temperature Sensor).

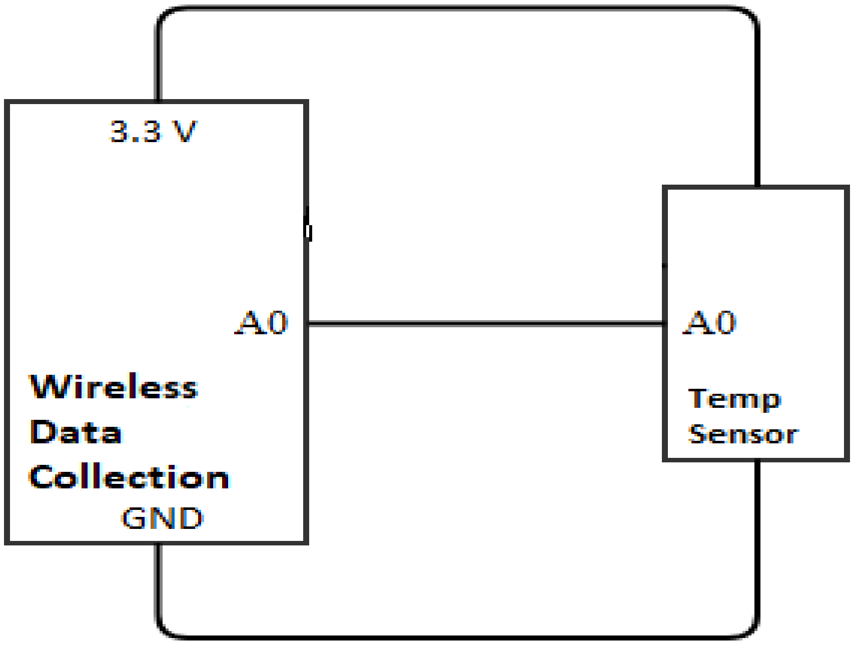


Fig. 5.6. Temperature Sensor connected to MC

5.1.2 Fire Sensor

A fire-detecting sensor card is used to detect possible fire incidents caused by chemical reactions that might take place in the medium. Fig. 5.7 shows below the connection of fire sensor to the central computing unit. (see Appendix A.5 for more information about the Fire Sensor).

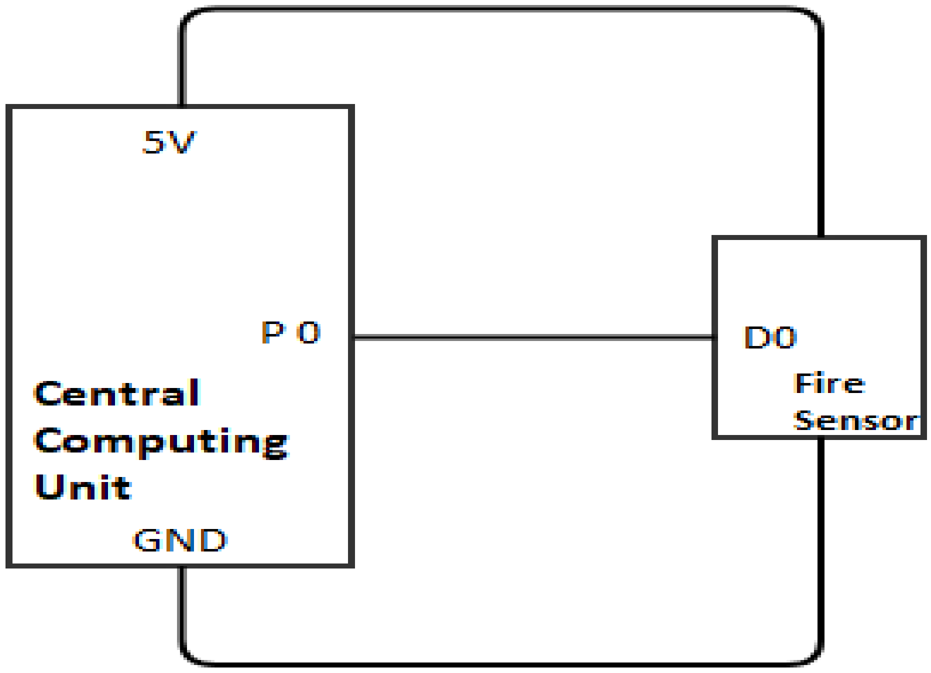


Fig. 5.7. Fire Sensor connected to MC

5.1.3 PH Sensor

PH controller provides all the necessary equipment for the precise pH measurement demands by using a microcontroller. After the necessary steps for the calibration of the measurement probe, the circuit connects to the microcontroller. Then, pH measurement circuit communicates with the microcontroller directly via serial UART connection. Fig. 5.8 shows below the connection of pH sensor to the wireless data collection unit. (see Appendix A.8 for more information about the PH Sensor).

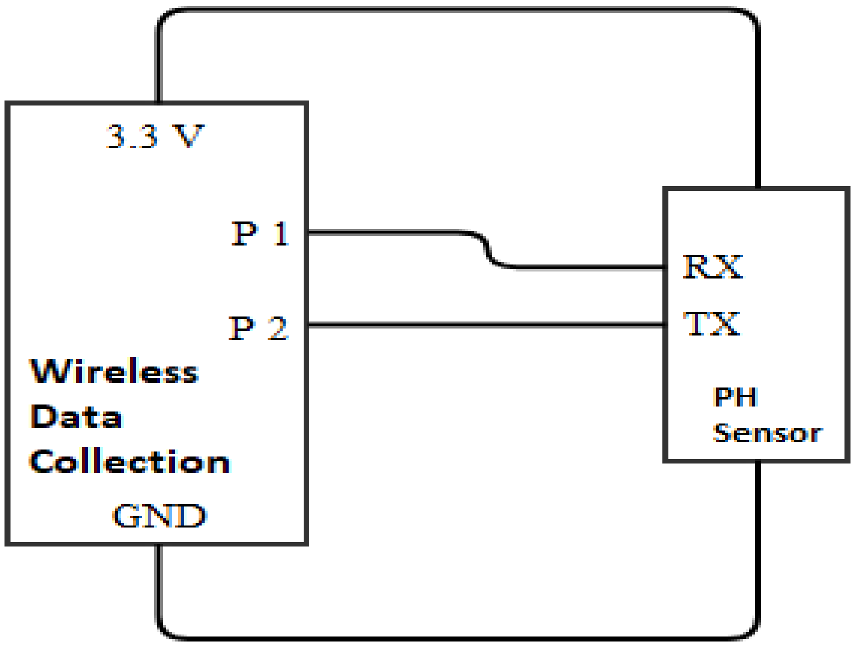


Fig. 5.8. PH sensor connected to MC

5.1.4 Electrical Conductivity Sensor

EC control provides all the necessary equipment for our precise electrical conductivity measurement demands by using a microcontroller. After taking the necessary steps to calibrate the measurement probe, the circuit is connected to the microcontroller. After that, EC measurement circuit communicates with the microcontroller directly via serial UART connection. Fig. 5.9 shows below the connection of EC sensor to the central computing unit. (see Appendix A.9 for more information about the Electrical Conductivity Sensor).

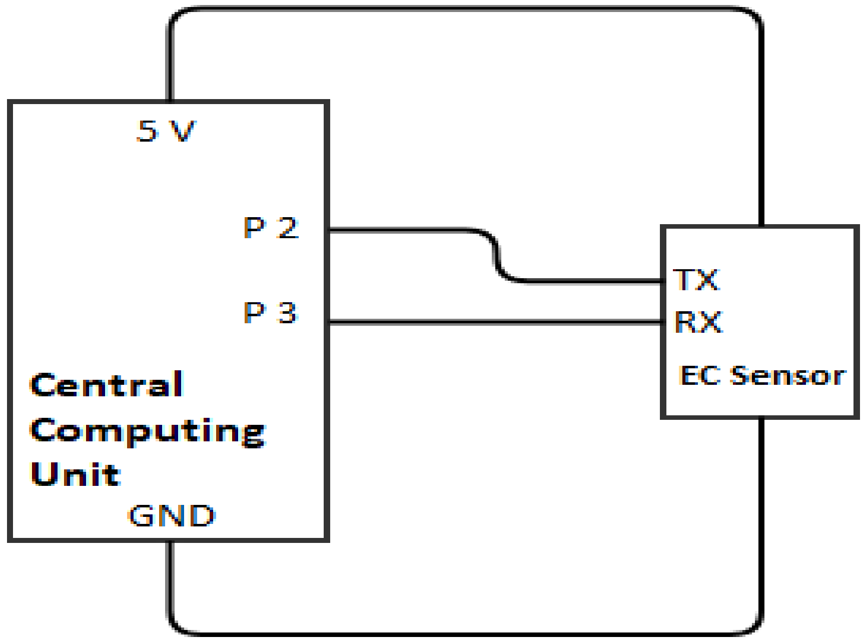


Fig. 5.9. EC sensor connected to MC

5.1.5 Soil Humidity Sensor

Soil humidity sensor measures the humidity amount in the soil or the level of a small-scale liquid. Fig. 5.10 shows below the connection of soil humidity detection sensor to the wireless data collection unit. (see Appendix A.7 for more information about the Soil Humidity Sensor).

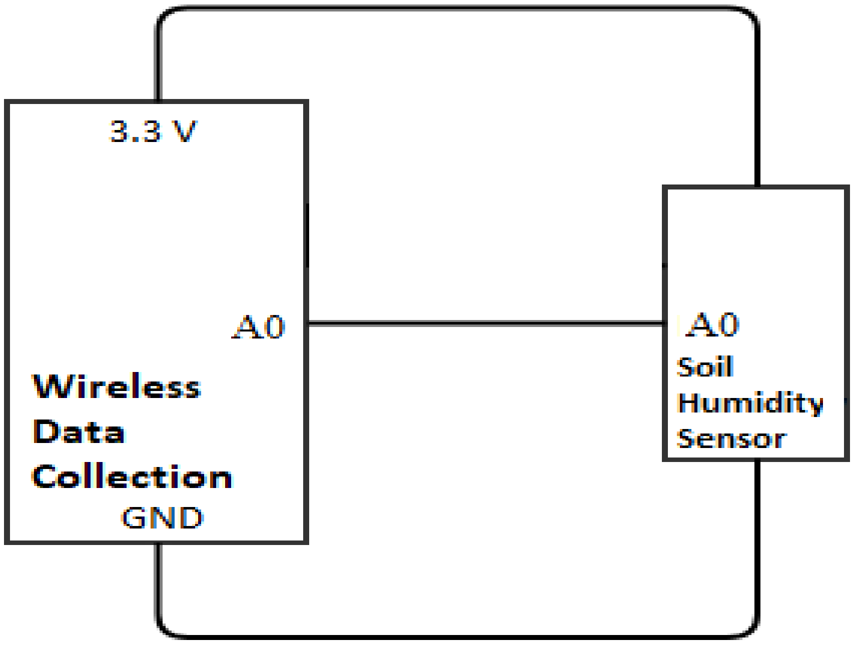


Fig. 5.10. Soil humidity detection sensor connected to MC

5.1.6 Ultrasonic Distance Sensor

Ultrasonic distance sensor is used to adjust the liquid levels in the reservoir at a desired point. Fig. 5.11 shows below the connection of ultrasonic distance sensor to the wireless data collection unit. (see Appendix A.14 for more information about the Ultrasonic Distance Sensor).

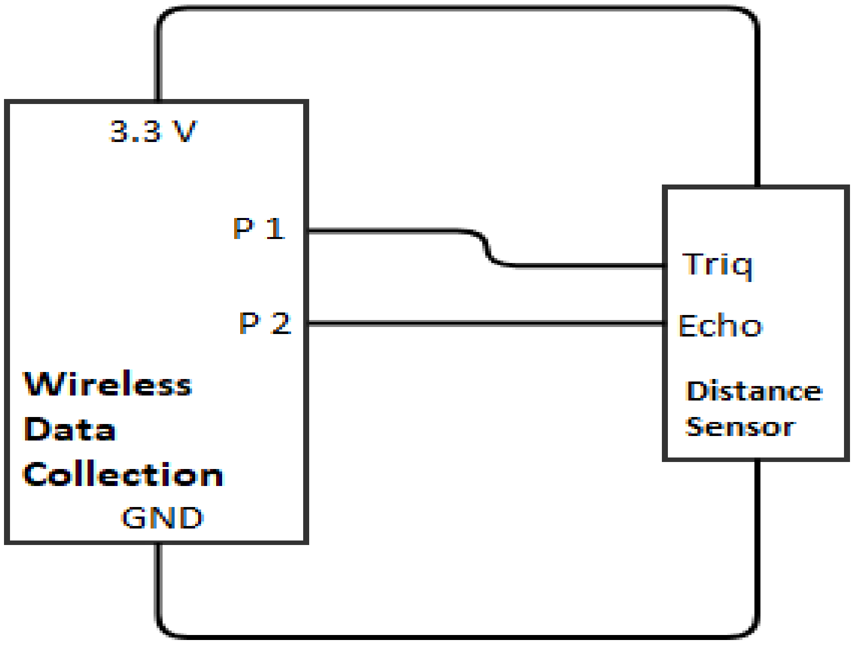


Fig. 5.11. Ultrasonic Distance Sensor connected to MC

5.2 Actuator Components

The electronic system that the microcontrollers are connected to runs on low-voltage but the field and greenhouse systems need high-voltage. For that reason control components that can run on high-voltage or high-current are needed. These control components are seen as crossover components between the field or greenhouse system and the microcontroller. They are called as driver circuit components.

Because of the fact that the control system forms the important part of the field and greenhouse automation system, a circuit group composed of relays was designed to have control over all of the driver components.

5.2.1 Relay Units

Relay is an electro-mechanical item that consists of a coil and a soft iron core. The electronic systems run on 5 to 24 volts with direct current as a standard. The microprocessors run on 7 to 12 volts with direct current. However, as most of the devices run on 12 volts or above with alternative current, relays or similar components are needed within the electronic systems. (see Appendix A.10 for more information about the Relay Units).

A simple schematic for interfacing a Single Relay using Arduino Uno is shown in Fig. 5.12.

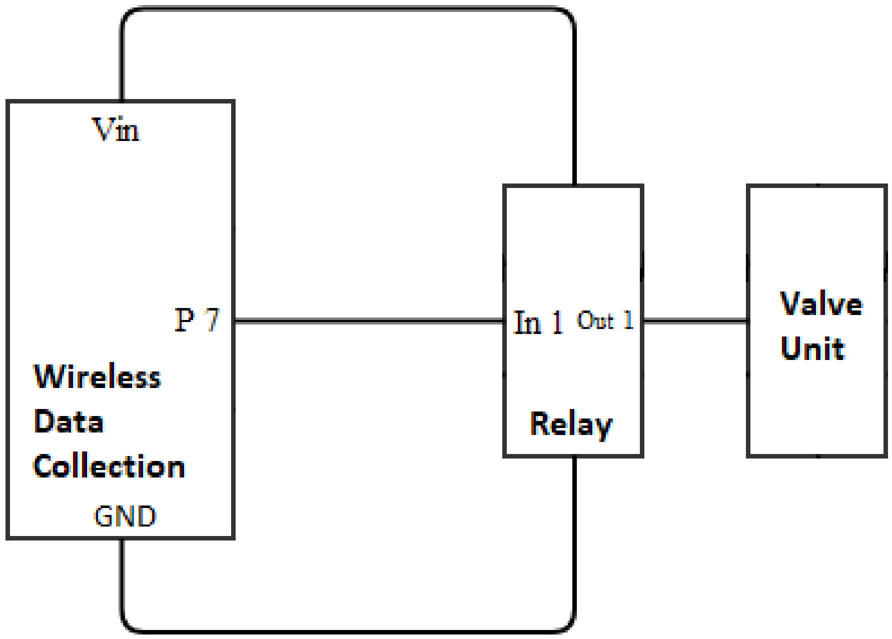


Fig. 5.12. Single Relay interfacing to MC

A simple schematic for interfacing a Quadplex Relay using Arduino Uno is shown in Fig. 5.13.

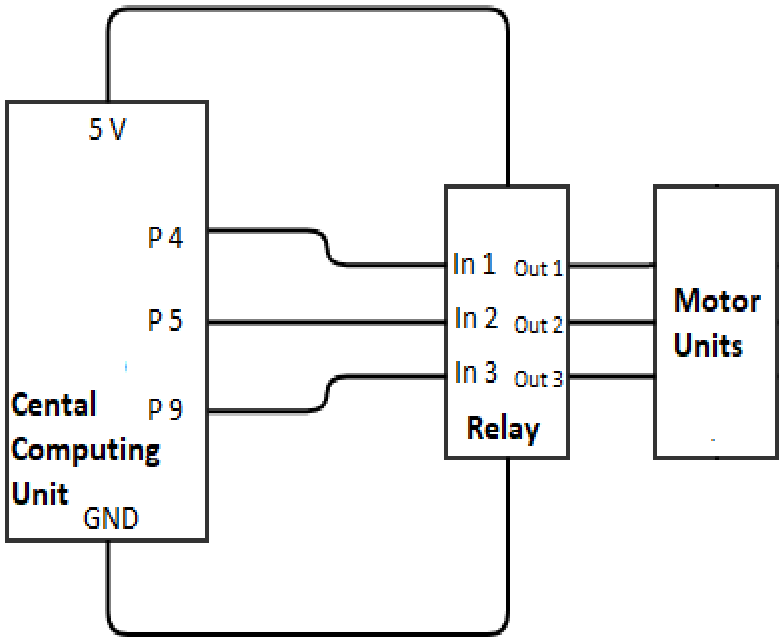


Fig. 5.13. Quadplex Relay interfacing to MC

5.2.2 Valve Units

Valves for controlling the liquid flow in field and greenhouse irrigation systems are electro-mechanical items which have models that run on 220 volts alternative current and 12 volts direct and alternative currents.

The valve for our system runs on 12 volts. Under normal circumstances, the valve is in on or off position. When 12V current is sent to its terminals it is turned on and makes it possible for the liquid to flow. (see Appendix A.11 for more information about the Valve Units).

5.2.3 Motor Units

In our system, 2 peristaltic liquid motors adjust the pH and fertilization values and 1 liquid pump transfers the adjusted liquid.

5.3 Maintenance Systems in Agriculture

5.3.1 Irrigation

The system starts or stops the irrigation system automatically by assessing whether the plant needs water through the data from the sensors to meet water requirements in the greenhouses and the fields.

In automatically controlled irrigation systems, the data gathered from the soil, the plant or the environment through the data recovery unit is processed and interpreted on the microprocessors. The irrigation operation is done according to the irrigation decision from the microprocessor.

The irrigation system is formed by circulating the irrigation water through plant stems or over the plants. The irrigation system can consist of either drip irrigation or sprinkling, or both depending on the plant species. To make the water circulation on and off automatically, an electric valve is used. Turning on the irrigation valve carries out the irrigation process until the soil and water saturation is fixed on the system.

5.3.2 Fertilization

The purpose of fertilization is to provide crops with plant nutrients which the soil lacks, and to get high quality products. Plant nutrient components must be determined exactly for high quality and productivity. It is also possible to carry out the fertilization process in the field or the greenhouse automatically. One of the methods to apply for this process is to deliver the fertilizer or the medicine to the plant using drip irrigation system. Another method is to provide homogenised distribution using sprinkler irrigation system.

Under both circumstances, the system must be taken into consideration with the irrigation system and preparation tanks must be added to the system to provide homogeneity. If the components used cause the irrigation system to clog, unclogging components must be added in those tanks [1].

EC sensor is used in fertilization. This sensor that is situated in the tank uses a peristaltic motor to provide adjustment with the data in the control centre. By means of the peristaltic motor, the liquid fertilizer that is acquired from outside is transferred to the tank.

5.3.2.1 Electrical Conductivity

EC is an abbreviation of ‘Electrical Conductivity’. Its unit is mS/cm. Chemical fertilizers consist of salts. By breaking into ions, salts conduct electricity by means of gaining positive or negative charges. Distilled water does not conduct electricity, and its conductivity increases as salt is added in it. By doing electrical conductivity measurement, the density of the fertilizers that are added to the irrigation water can be adjusted.

Every crop that is grown in the same way has specific EC demands that are idiosyncratic and dependent to the growing periods. All these factors must be adjusted and measured together, and applied to the plants. In this way, we produce the highest productivity and the quality from fruit or vegetables. To measure the EC value of the mixture situated in our reservoir, Atlas Scientific EC Sensor is used in the system. By this way, our measurements are at high-accuracy levels.

The commonly used units for measuring electrical conductivity of liquid are:

S/cm (microSiemens/cm) or dS/m (deciSiemens/m)

Where: 1000 s/cm = 1 dS/m

5.3.2.2 Potential Hydrogen

PH is an abbreviation of 'Potential Hydrogen'. When the pH value of the irrigation water is 7, then it is neutral. If it is below 7, then it is acidic. If it is above 7, it is alkaline.

In soils with low pH (acidic), absorption of heavy metals like copper, manganese, iron, zinc and aluminium, and boron increases, while absorption of magnesium, calcium, phosphorus and potassium decreases alongside with nitrification (turning of the urea into ammonium and nitrate). In soils with low pH (acidic), the productivity decreases as a result.

On the other hand, in soils with high pH (alkaline), the absorption of micro elements (iron, manganese, boron, zinc) and phosphorus by the plant is more difficult. In alkali soils, urea nitrate cycle is faster. With acidification, urease enzyme activity decreases and urea hydrolysis is slower.

These values may differ according to the plant species, stem medium (soil, turf, rockwool etc.) and climate values. If the fertilization process is carried out in ideal values, plant growth is kept at optimal levels.

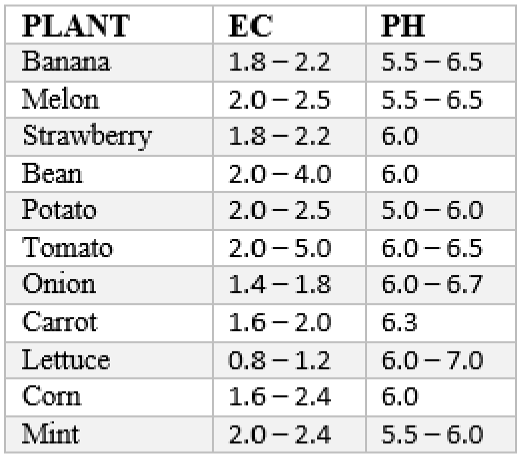
As a measure of soil acidity or alkalinity, soil pH constitutes one of the most important chemical soil parameters [14].

For example, most foresters believe that pines grow best on acidic soils while hardwoods prefer slightly acidic soils to neutral soils. While there is some truth in this, most tree species will grow well over a broad range of pH values [15].

Soil pH influences nutrient uptake and tree growth. The availability of many plant nutrients in the soil changes as a result of reactions in the soil, which are largely controlled by soil pH. Trees may or may not be able to use nutrients because of these reactions [16].

The rate of pH in the soil is on an increase depending on the irrigation and fertilization. This case means that sometimes, it may not be possible to provide the proper medium for the crop that is produced.

EC and PH value samples according to the plant species are shown in the Fig. 5.14.



PLANT	EC	PH
Banana	1.8 – 2.2	5.5 – 6.5
Melon	2.0 – 2.5	5.5 – 6.5
Strawberry	1.8 – 2.2	6.0
Bean	2.0 – 4.0	6.0
Potato	2.0 – 2.5	5.0 – 6.0
Tomato	2.0 – 5.0	6.0 – 6.5
Onion	1.4 – 1.8	6.0 – 6.7
Carrot	1.6 – 2.0	6.3
Lettuce	0.8 – 1.2	6.0 – 7.0
Corn	1.6 – 2.4	6.0
Mint	2.0 – 2.4	5.5 – 6.0

Fig. 5.14. EC and PH Value Samples

Depending on the data sent from the EC sensor to the control centre, fertilization in ideal values is provided by mixing fertilizers into the irrigation water situated in the reservoir with the help of the peristaltic motor. As the fertilizer mixture increases, so does the EC value, and as the acid mixture increases, pH value goes down.

5.4 The Alarm System

In an environment that has a control centre with a fire sensor, the alarm system activates in case of any fire detection. The alarm system is designed to be able to give an audible warning and send information via e-mail.

Chapter 6

Prototype Implementation

6.1 Design of the Control Centre

Control centre consists of three parts basically. The main part is based on a microcontroller together with its wireless shield, and the necessary software is installed here. The second part is where the necessary sensors are situated. These sensors are the fire detecting sensor, and the EC detecting sensor. The third part controls the output components that consist of relays.

The image shown in Fig 6.1 illustrates the implementation of a single unit. This unit contains (a) Buzzer, (b) Fire sensor, (c) EC sensor, (d) Arduino Uno, (e) Arduino Wi-Fi shield, (f) Relay.

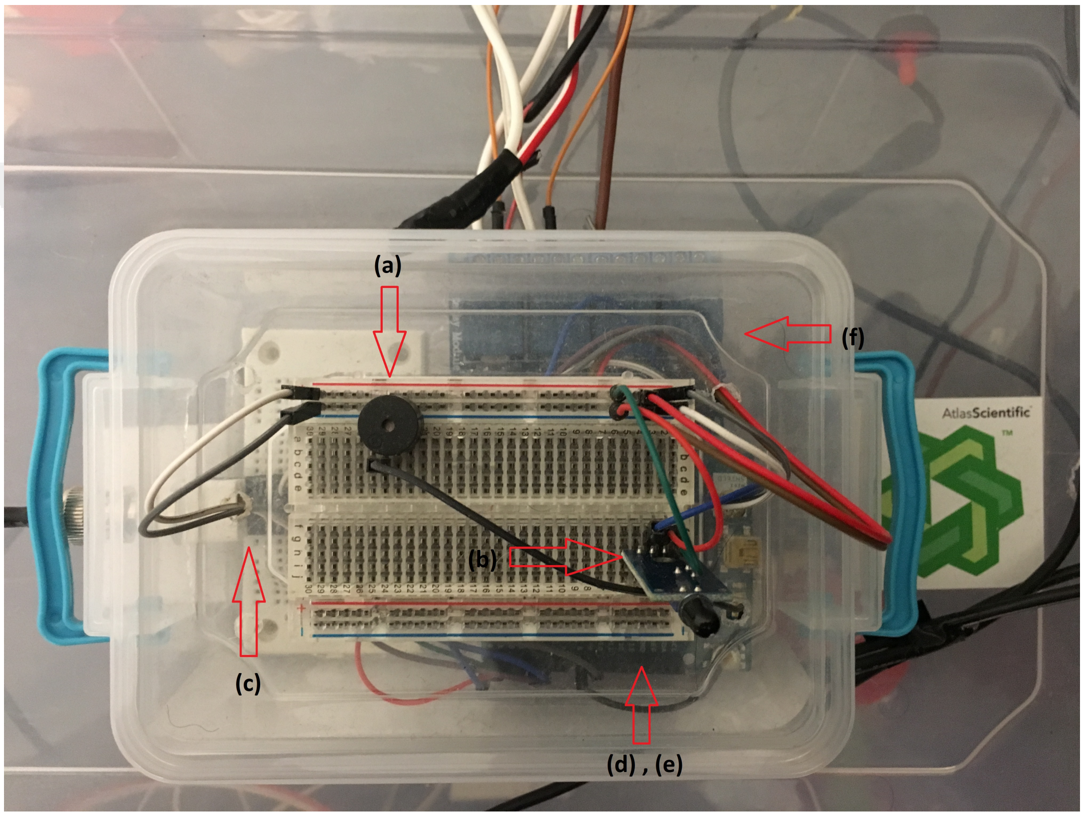


Fig. 6.1. Implementation of The Control Centre

6.2 Design of the Humidity and PH Measurement from The Soil Circuit

This is the circuit prepared to assess the humidity and pH values taken from the soil and to transfer the acquired data to the system. It consists of two parts. The main part is a microcontroller to which the sensors transfer the data to the control centre. The second part consists of the soil humidity sensor, by which the humidity value from the soil is measured, and the pH sensor, by which the pH value from the soil is measured.

The image shown in Fig 6.2 illustrates the implementation of a single unit. This unit contains (a) NodeMCU ESP8266, (b) PH sensor, (c) Soil Humidity Sensor.

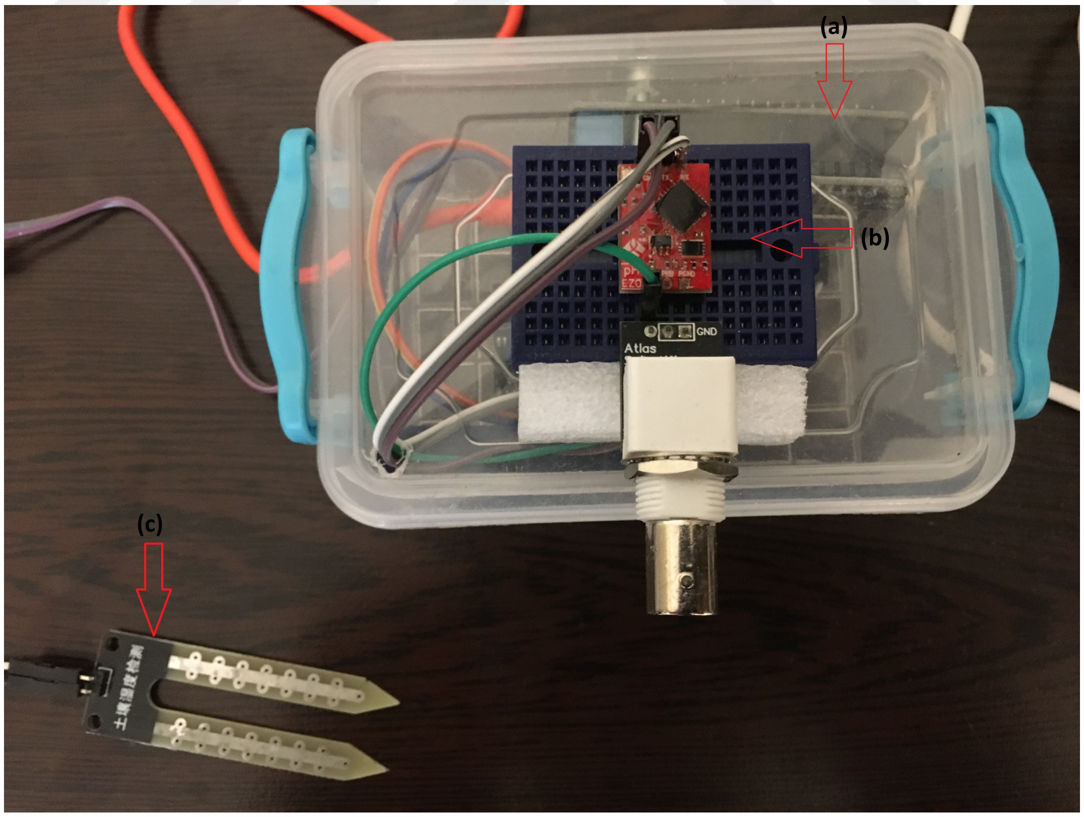


Fig. 6.2. Implementation of Humidity and PH Measurement from The Soil Circuit

6.3 Design of the Distribution Valve Control Circuit

This circuit turns the valve on and off, depending on the prompts received from the system. It utilizes a microprocessor, to which the valve is connected to execute the demands received from the system.

The image shown in Fig 6.3 illustrates the implementation of a single unit. This unit contains (a) NodeMCU ESP8266, (b) Relay, (c) Valve.

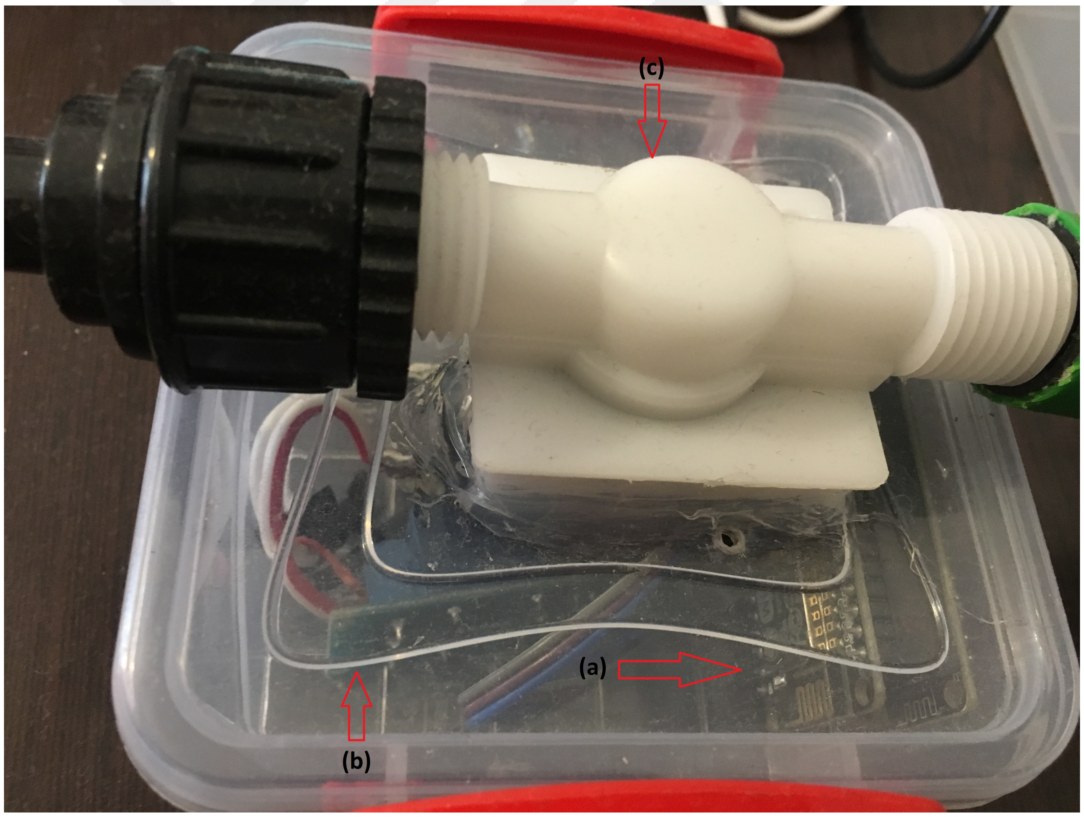


Fig. 6.3. Implementation of Valve On/Off Circuit

6.4 Design of the Reservoir Liquid Level Adjustment Circuit

This circuit provides the adjustment of the liquid level in the reservoir at a desired point. A valve, a relay and a liquid level sensor are used alongside with a microprocessor, to which these three are connected.

The image shown in Fig 6.4 illustrates the implementation of a single unit. This unit contains (a) Valve, (b) HC-SR04, (c) Relay, (d) NodeMCU ESP8266.

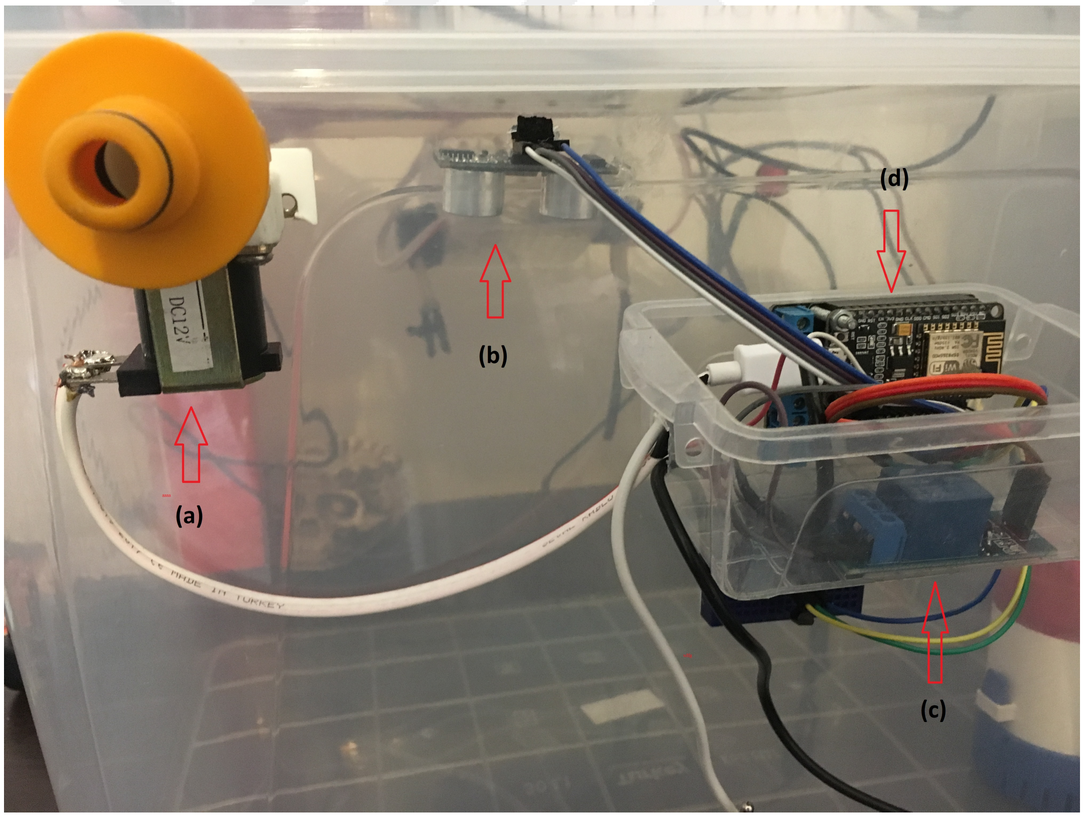


Fig. 6.4. Implementation of Reservoir Liquid Level Adjustment Circuit

6.5 Interface of The System with IoT Builder

Accessing the system, on which the controls and adjustments of the automation system are made, is achieved via World Wide Web. The device designed as the control centre consists of categories which enable the control of the motors on the system and show sensor status info.

After the device connects to the Internet, its interface is as the image shown in Fig. 6.5.

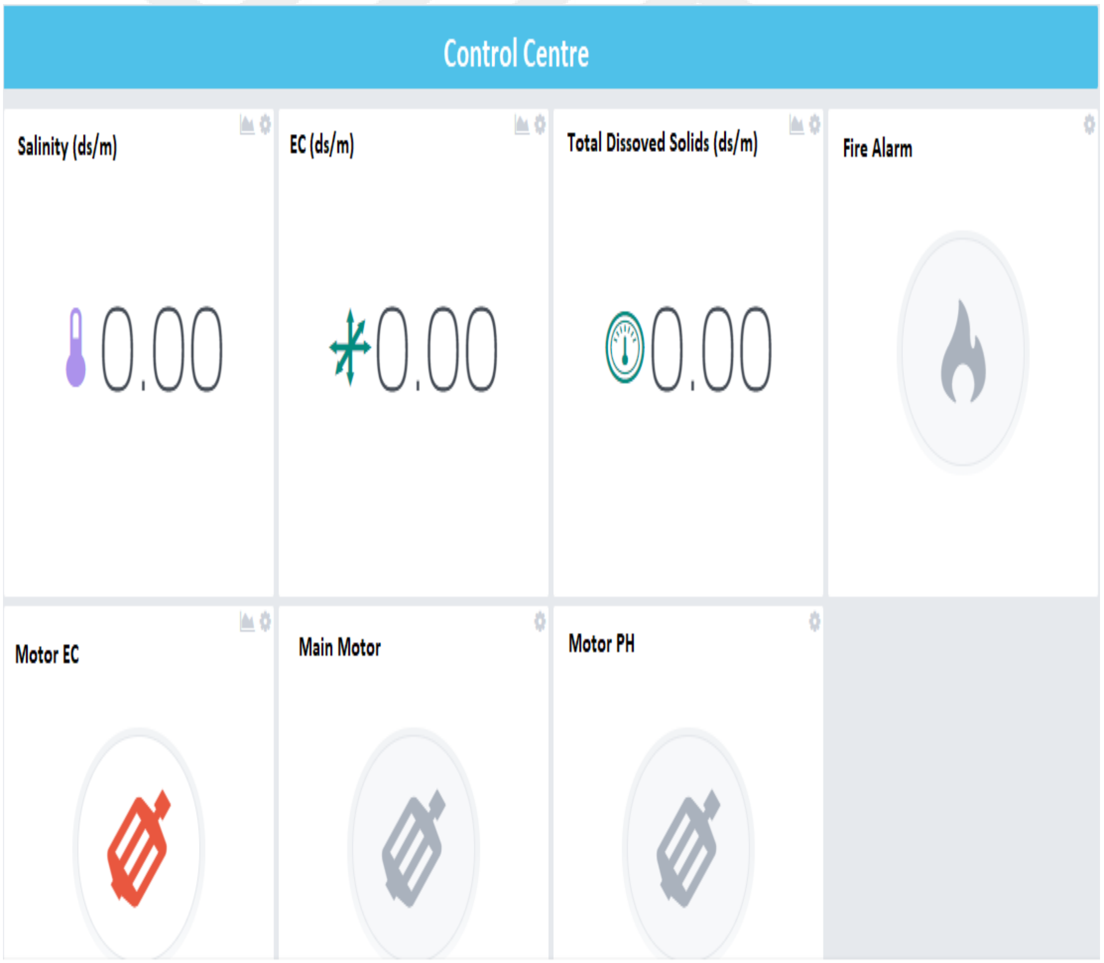


Fig. 6.5. Control Centre Interface

The device designed as the humidity and pH measurement from the soil consists of categories which show soil dryness and pH values on the system. After the device connects to the Internet, its interface is as the image shown in Fig. 6.6.

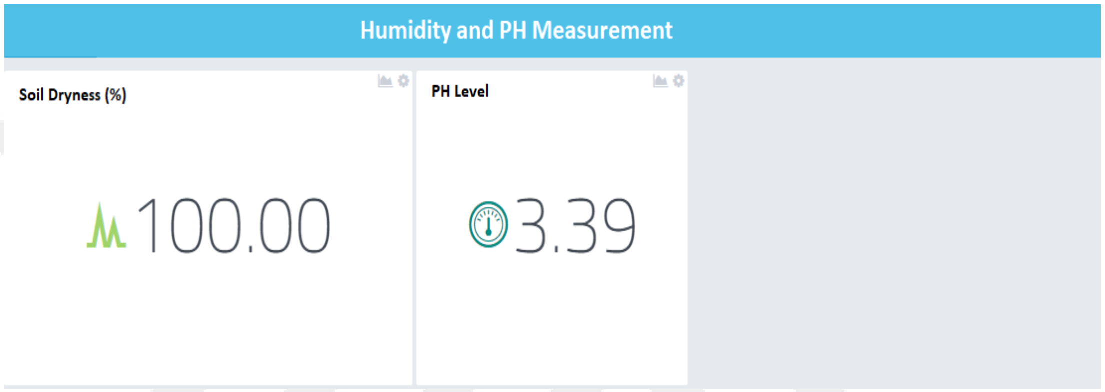


Fig. 6.6. Humidity and PH Measurement Interface

The device designed as the liquid level adjustment consists of categories which enable us to control the valve on the system, and shows sensor status info. After the device connects to the Internet, its interface is as the image shown in Fig. 6.7.

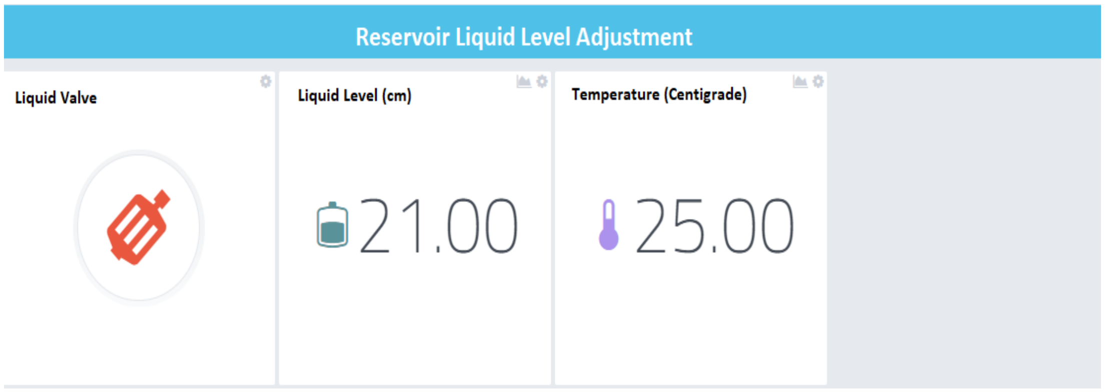


Fig. 6.7. Liquid Level Adjustment Interface

The device designed as the valve control enables us to control the valves on the system. After the device connects to the Internet, its interface is as the image shown in Fig. 6.8.

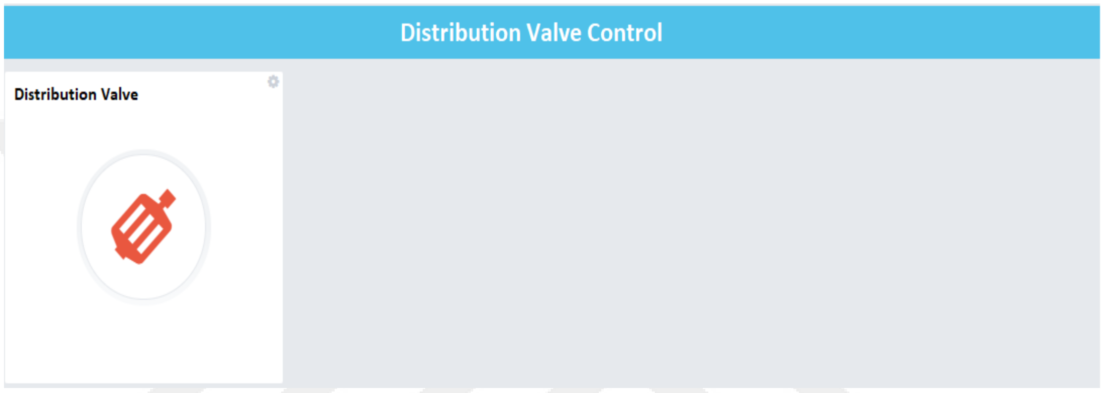


Fig. 6.8. Distribution Valve Control Interface

The system runs the command systems according to the data received from the sensors. All the operations on the microprocessors and the data received are shown on the website instantly. Every Internet user can access this information system. Besides, last measured data can be viewed on the web site in a chart (i.e. Soil Dryness Chart Fig. 6.9).

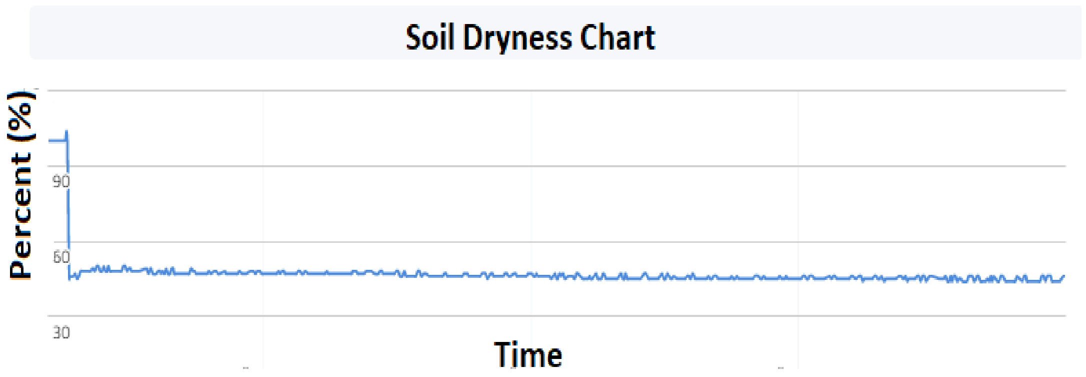
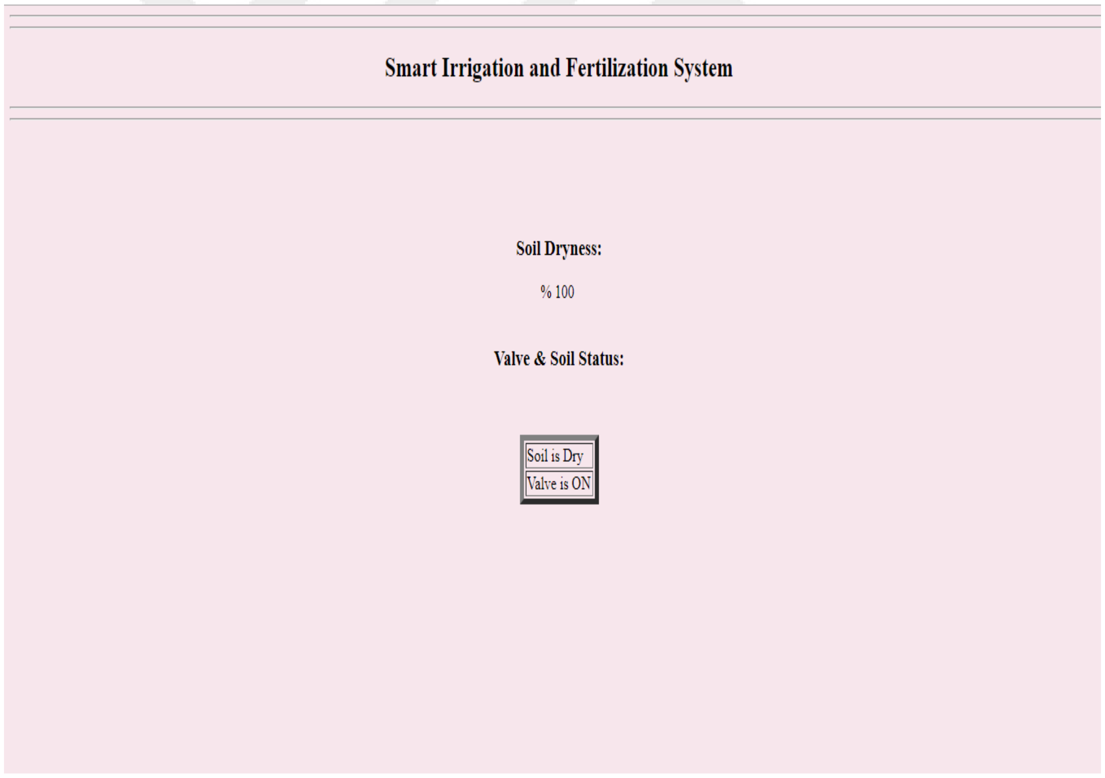


Fig. 6.9. Soil Dryness Chart

6.6 Interface of The System without IoT Builder

In the system working without IoT builder, the fertilization parameters were adjusted in the liquid tank and distributed to the soil by the system automatically. After that an interface was designed to show the status of the data obtained form the soil and the distribution valve. Its web interface is as the image shown in Fig. 6.10.


Smart Irrigation and Fertilization System

Soil Dryness:

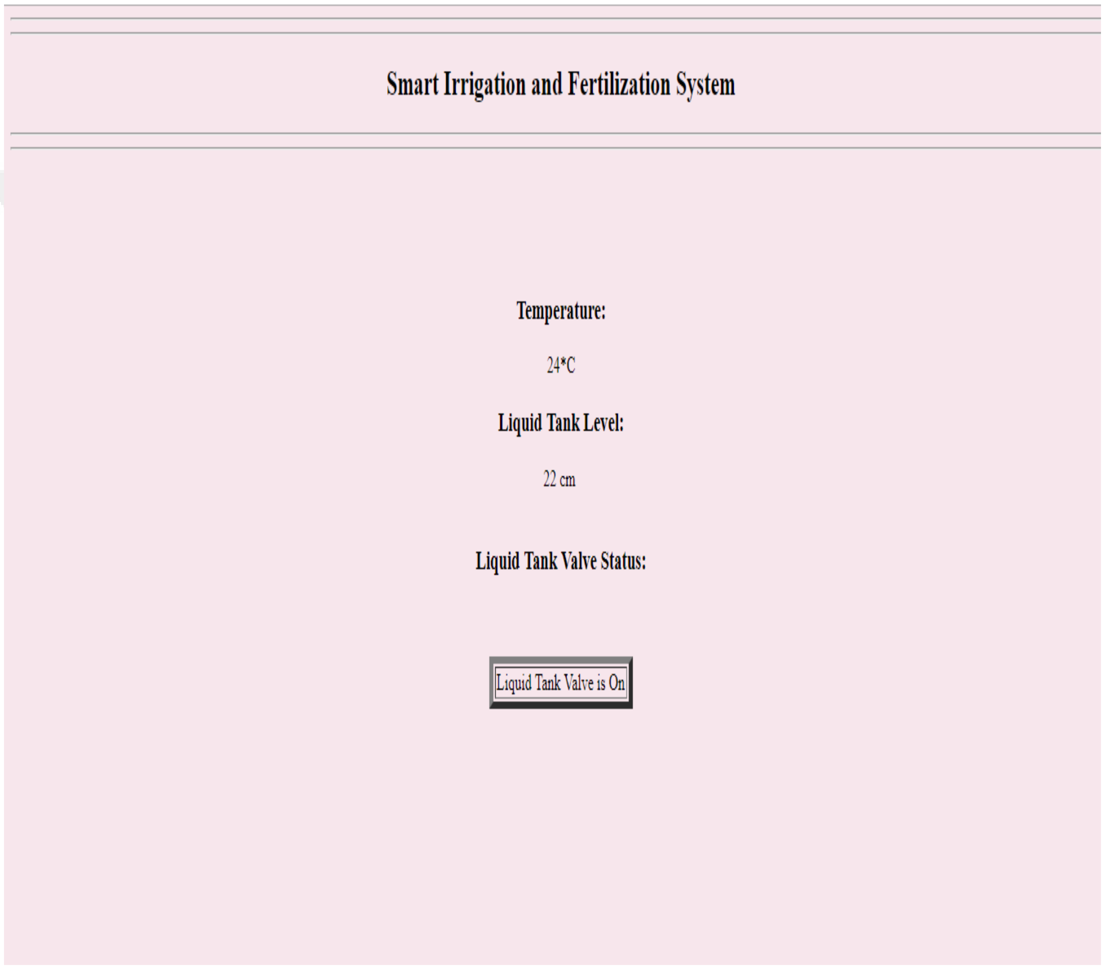
% 100

Valve & Soil Status:

Soil is Dry
Valve is ON

Fig. 6.10. Web Interface of Irrigation Centre

Web interface of the liquid level adjustment circuit is as the image shown in Fig. 6.11.



Smart Irrigation and Fertilization System

Temperature:

24*C

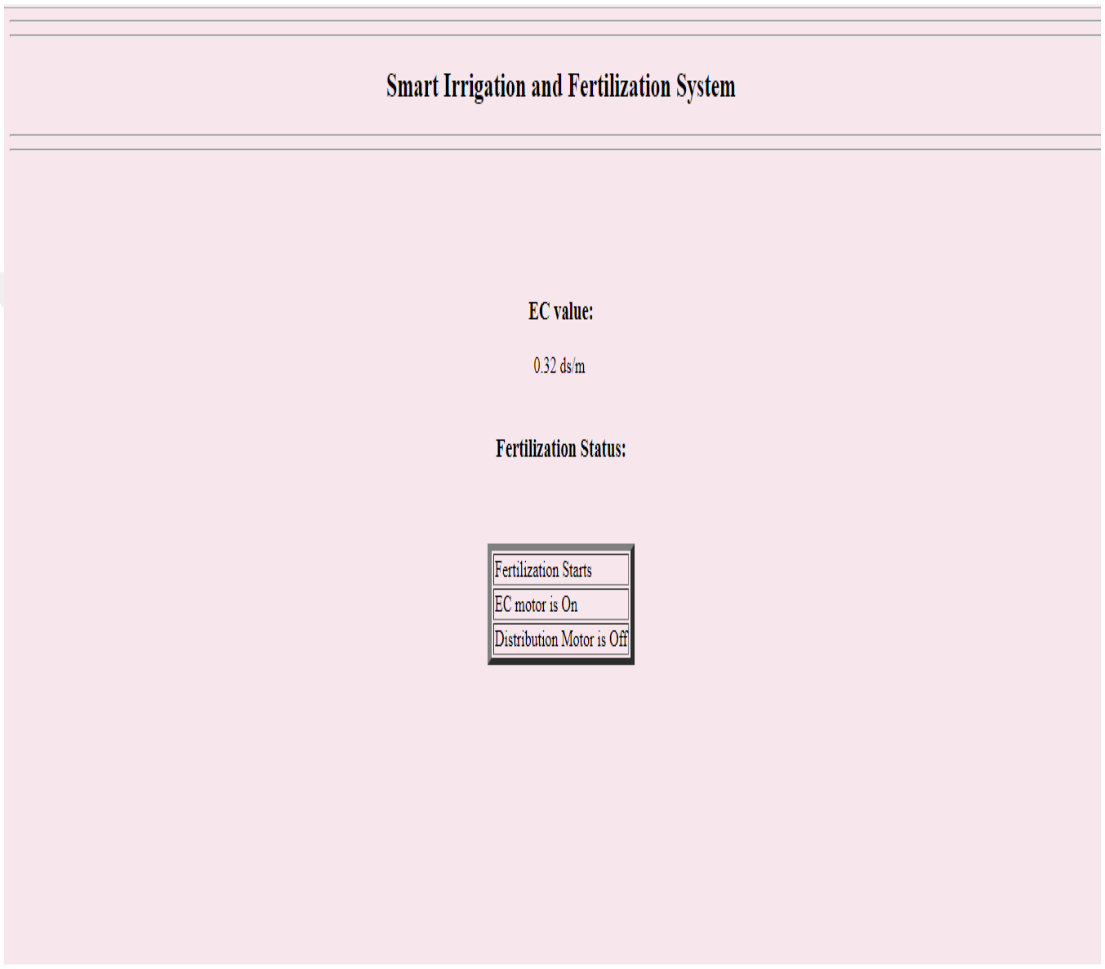
Liquid Tank Level:

22 cm

Liquid Tank Valve Status:

Fig. 6.11. Web Interface of The Liquid Level Adjustment Centre

Web interface of the fertilization centre is as the image shown in Fig. 6.12.



Smart Irrigation and Fertilization System

EC value:

0.32 ds/m

Fertilization Status:

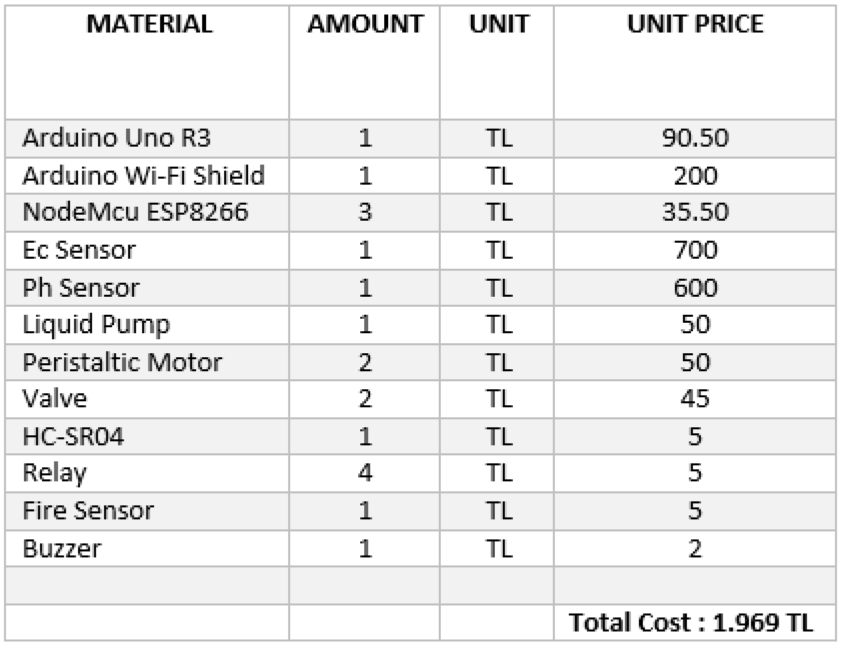
Fertilization Starts
EC motor is On
Distribution Motor is Off

Fig. 6.12. Web Interface of The Fertilization Centre

Chapter 7

Unit Cost Analysis

Fig. 7.1 shows the equipment list and unit prices for the components needed to build the system.



MATERIAL	AMOUNT	UNIT	UNIT PRICE
Arduino Uno R3	1	TL	90.50
Arduino Wi-Fi Shield	1	TL	200
NodeMcu ESP8266	3	TL	35.50
Ec Sensor	1	TL	700
Ph Sensor	1	TL	600
Liquid Pump	1	TL	50
Peristaltic Motor	2	TL	50
Valve	2	TL	45
HC-SR04	1	TL	5
Relay	4	TL	5
Fire Sensor	1	TL	5
Buzzer	1	TL	2
			Total Cost : 1.969 TL

Fig. 7.1. Equipment List and Unit Prices

Chapter 8

Conclusion and Recommendations

The leading purpose of our study is to use the water and fertilizer demands of the agricultural products that are grown on the necessary amounts to prevent overconsumption. Because our project is established on a large scale, it saves time and workforce to a great extent.

The fact that automation systems are high in costs and that there is a need for craftsmen pose the greatest obstacles for the use of automation systems in agricultural areas. By installing them in agricultural areas economically, as it has both a low-cost and a user-friendly software, the automation system developed will be compatible to the renovations in the long term. By means of the user-friendly system, there will be no need for extra workforce. When the necessary data for the crops produced are identified, the system will carry out the proper automation processes for the product.

Studies have shown that to design a system for the fields and the greenhouses, an adequate level of information about this area is needed in the first place. At this stage, analysing the current system is rather important as it may provide ideas about what the system contents will be and what other features can be added to the system in the future.

The equipment we use in the designing stage is high-quality and high-precision. Therefore, exposure to environmental conditions is in a minimum level.

Processes become much faster, more accurate and more productive in the designing stage of the system by making use of modern technologies. For this reason, wireless access is used for the Internet connection in order that the geographical conditions of the field and the greenhouse do not constitute a problem. Also, it is possible to handle the controls remotely without visiting the field or the greenhouse. In this way, the installation of the system is quite practical.

The difference of our system from other similar systems is that it contributes to accessing the data received from the sensors online or offline. Unlike other systems, the adjustment of the necessary fertilization and irrigation values for the plant in question are provided automatically. Also, those values can be viewed in real-time. As our system has a flexible structure, its development depending on the plant species can be adjusted easily.

Appendix A

Components

A.1 NodeMCU LoLin ESP8266

Fig. A.1. NodeMCU LoLin ESP8266

Technical Features:

- ESP8266 SDK based
- eLua core (Lua 5.1.4)
- Lua-cjson supported

- Spiffs file system
- It supports json, file, timer, pwm, i2c, spi, 1-wire, net, mqtt, coap, gpio, Wi-Fi, ADC, uart and system API

The definitions of NodeMCU LoLin ESP8266 pins and its systematic scheme is shown in Fig. A.2.

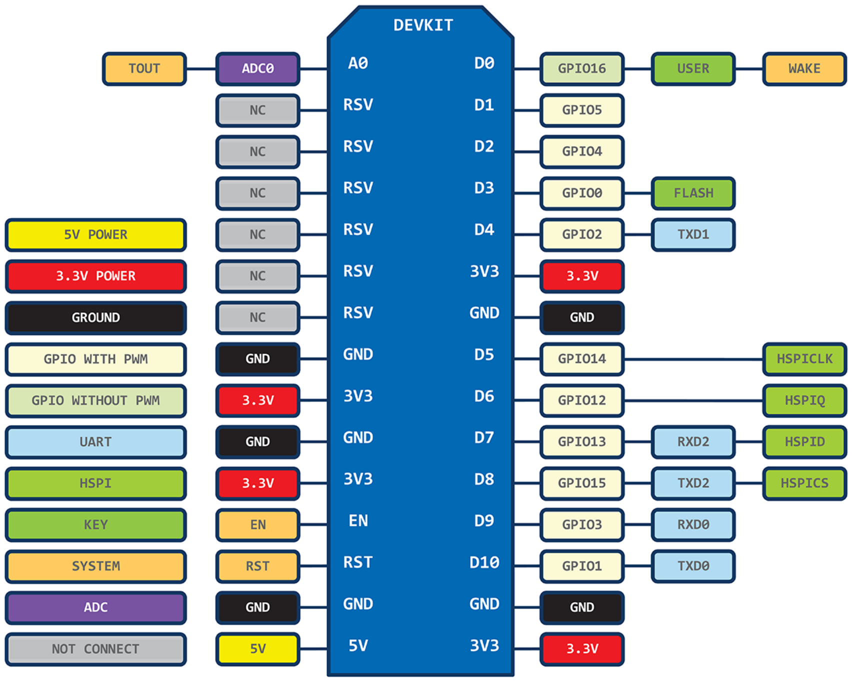


Fig. A.2. NodeMCU LoLin ESP8266 Pins

A.2 Arduino Uno

Ardiuno is a microcontroller card which is open-source and developed with an input/output card [10]. Ardiuno Uno is one of the most commonly-used Ardiuno cards in this field; which is offered to public use in 2010 and uses ATmega328 microcontroller. This platform, which runs on 7-12V, has 14 digital in and out pins. You may use 6 of those for the PWM.

Fig. A.3. Arduino Uno

Technical Features:

- Microcontroller: ATmega328
- Operating voltage: 5V
- Input Voltage (recommended): 7-12V
- Input Voltage (limit): 6-20V
- Digital I/O Pins: 14 (6 of them PWM output)

- Analog Input Pins: 6
- Current per I/O: 40 mA
- Current for 3.3V output: 50 mA
- Flash Memory: 32 KB (ATmega328) Up to 0.5 KB for bootloader
- SRAM: 2 KB (ATmega328)
- EEPROM: 1 KB (ATmega328)
- Clock Speed: 16 MHz
- Length: 68.6 mm
- Width: 53.4 mm
- Weight: 25 g

The definitions of Ardiuno UNO pins and its systematic scheme is shown in Fig. A.4.

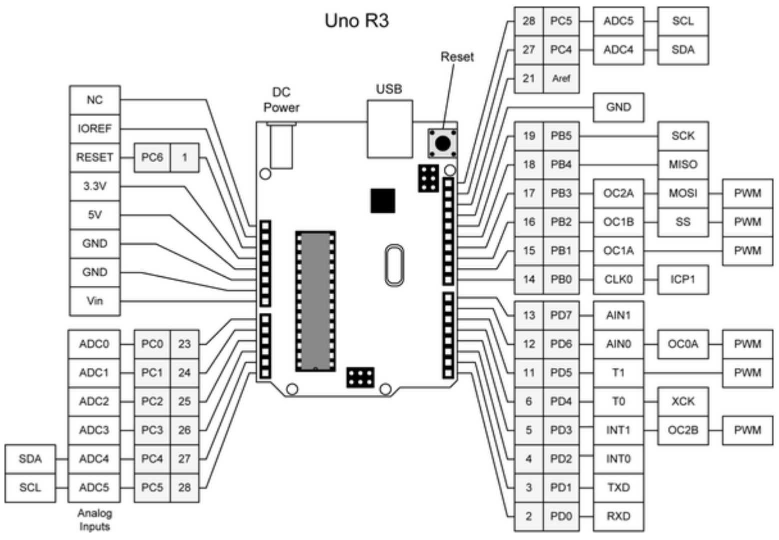


Fig. A.4. Arduino Uno Pins

A.3 WI-FI Shield for Arduino Uno

IEEE, the Wi-Fi module on it, supports 802.11 b/g/n modes up to a speed of 65 Mbit/sec maximum. In addition that it has PSB antenna on, due to the uFL connector it possesses, it enables you to connect a more powerful antenna. By means of the switch it has, it can be turned into access point mode when desired. Through USB-UART convertor, the module can be updated by connecting it to your computer. It includes a microSD card slot on board.

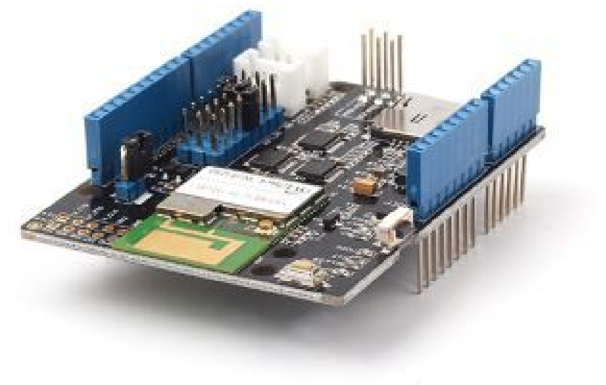


Fig. A.5. Arduino Uno WI-FI Shield

Technical Features:

- Module: WIZnet FI250
- Operating voltage: 3.3V or 5V
- Operating current: 300 mA maximum
- Supports 2.4 GHz IEEE 802.11 b/g/n protocols
- Internal PCB antenna and uFL connector

- 1 MB flash memory, 128 KB SRAM, 1 MB serial flash memory
- UART/SPI interface

Arduino Wireless Shield systematic scheme is shown in Fig. A.6.

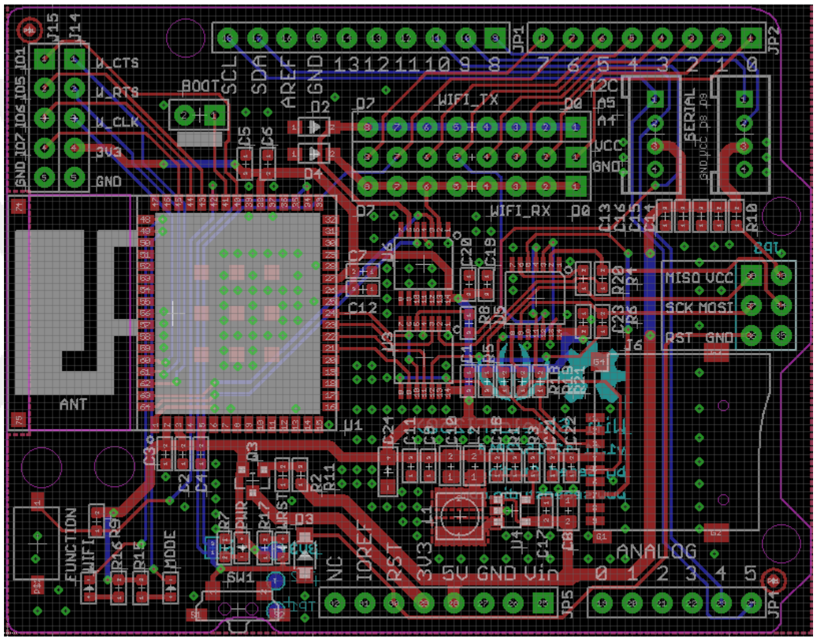


Fig. A.6. Arduino Wireless Shield Systematic Scheme

A.4 Temperature Sensor

LM35 is a cost effective and high-quality temperature sensor. It has an analogue output. It can measure temperatures between -55 °C and 150 °C. It has a precision of 10 mV/degree.

Fig. A.7. LM35

Technical Features:

- Calibrated directly in Celsius (Centigrade)
- Linear + 10.0 mV/°C scale factor
- 0.5C accuracy guaranteeable (at +25 °C)
- Rated for full -55 °C to +150 °C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Low self-heating, 0.08 °C in still air
- Low impedance output, 0.1 Ohm for 1 mA load

The definitions of LM35 pins are shown in Fig. A.8.

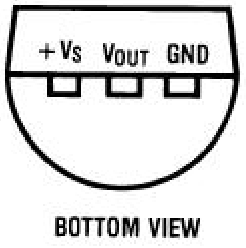


Fig. A.8. LM35 Pins

A.5 Fire Detecting Sensor

The fire detecting sensor card is a sensor card that has a wavelength between 760 nm and 1100nm and it is used for the purpose of detecting a fire incident. It accommodates an IR sensor on itself. Precision adjustments can be made by means of the trimpot on it and outputs can be acquired both in analogue and in digital form from it.

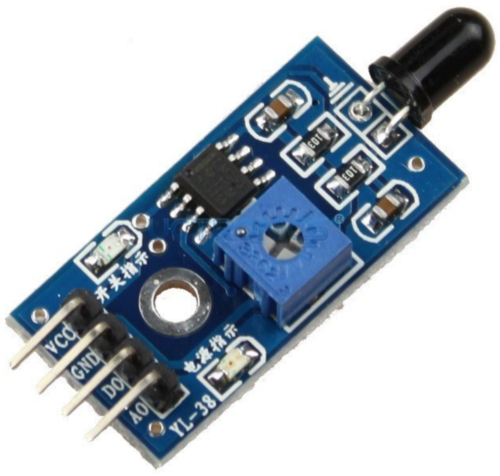


Fig. A.9. Fire Detecting Sensor

Technical Features:

- Operating Voltage: 5V
- Signal Output: 20 cm(1V) - 100 cm (4.8V)
- Can be used in fire extinguishing robots and fire detection systems
- Can be used as the IR receiver card
- Dimensions: 18 mm x 12 mm

The definitions of Fire Detecting Sensor Card pins are shown in Fig. A.10.

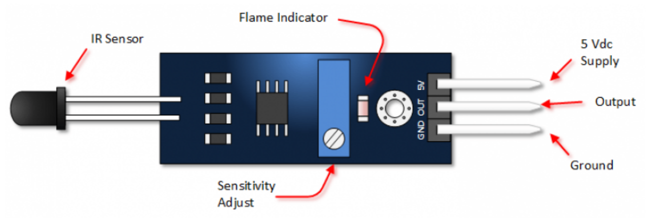


Fig. A.10. Fire Detecting Sensor Pins

A.6 Alarm Unit

Buzzer is a device that can detect different sound signals depending on the voltage given. As they are low in costs, simple in production and light in weight, their area of utilization is pretty wide. (i.e. burglar alarms, many systems that gives warnings in vehicles, some ringtones) In short, it can be used everywhere in order to receive warnings.

Fig. A.11. Buzzer

Technical Features:

- Voltage Interval: 3.0 - 20 VDC
- Noise Level: min 95db
- Output Signal Frequency: 3.500 +/- 200 Hz
- Operating Temperature Interval: -20 °C to +60 °C

The definitions of buzzer pins are shown in Fig. A.12.

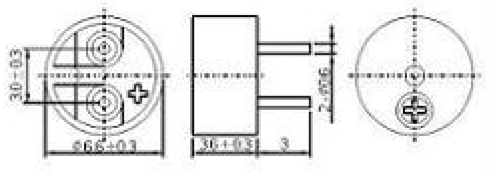


Fig. A.12. Buzzer Pins

A.7 Soil Humidity Sensor

It is used by sticking the humidity measuring probes into the medium. By analysing potential difference in the soil, humidity amount can be measured. As the humidity level in the soil increases, so does the conductivity. By means of the trimpot that is situated on the card, precision adjustments can be made.

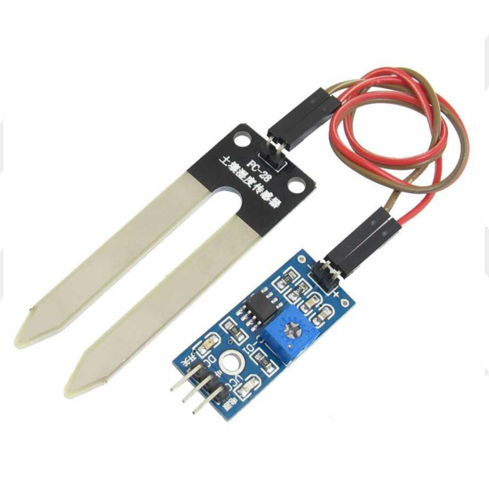


Fig. A.13. Soil Humidity Detecting Sensor

Technical Features:

- Operating voltage: 3.3V - 5V
- Output Voltage: 0 - 4.2V
- Current: 35 mA
- Output Type: Digital and Analogue

The definitions of Soil Humidity Detecting Sensor pins are shown in Fig. A.14.

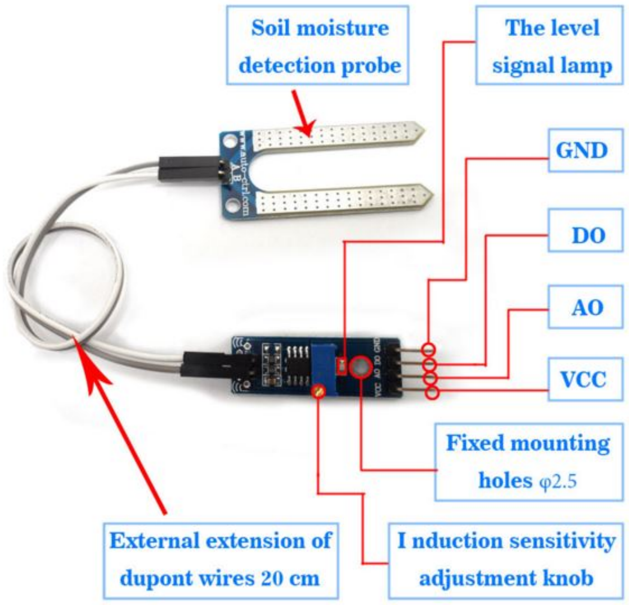

Soil moisture detection probe
The level signal lamp
GND
DO
AO
VCC
Fixed mounting holes φ2.5
External extension of dupont wires 20 cm
I nduction sensitivity adjustment knob

Fig. A.14. Soil Humidity Detecting Sensor Pins

A.8 PH Detecting Sensor

A.8.1 PH Circuit

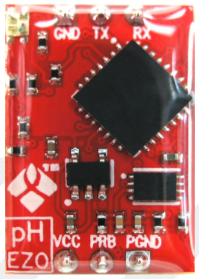


Fig. A.15. PH Circuit

Technical Features:

- Reads: pH
- Range: .001 - 14.000
- Accuracy: +/- 0.002
- Max rate: 1 reading per sec
- Supported probes: Any type and brand
- Temp compensation: Yes
- Data protocol: UART - I2C
- Operating voltage: 3.3V - 5V
- Data format: ASCII

A.8.2 PH Probe

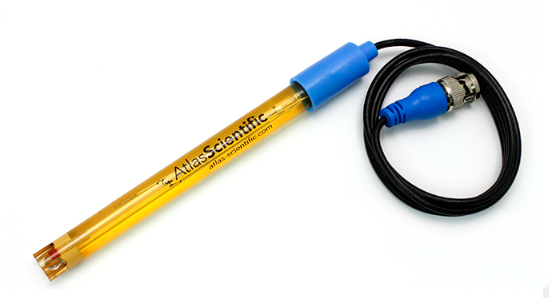


Fig. A.16. PH Probe

Technical Features:

- Reads: pH
- Range: 0 - 14
- Response time: 95% in 1s
- Max pressure: 100psi
- Temperature range: 1 °C - 99 °C
- Internal temperature sensor: No
- Cable length: 1 meter
- Weight: 49 grams
- Dimensions: 12mm x 150mm
- BNC connector: Yes

A.9 EC Detecting Sensor

A.9.1 EC Circuit

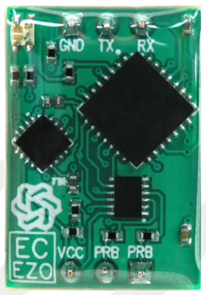


Fig. A.17. EC Circuit

Technical Features:

- Reads: Conductivity, Total dissolved solids (ppm), Salinity Specific gravity (sea water only)
- Range: 0.07 500,000+ MiliSec/cm
- Accuracy: +/ 2%
- Max rate: 1 reading per sec
- Supported probes: K 0.1 K 10 any brand
- Temp compensation: Yes
- Data protocol: UART- I2C
- Operating voltage: 3.3V 5V
- Data format: ASCII

A.9.2 EC Probe K 1.0

Fig. A.18. EC Probe K 1.0

Technical Features:

- Reads: Conductivity
- Range: 5 200,000 S/cm
- Response time: 90% in 1s
- Max pressure: 200psi
- Temperature range: 1 °C - 110 °C
- Internal temperature sensor: No
- Cable length: 1 meter
- Weight: 51 grams
- Measuring Surface: Graphite
- Dimensions: 12mm x 150mm
- BNC connector: Yes

A.10 Relay Unit

Relay control card is a relay card that enables the user to control the contact with 5V and one that can be used with Ardiuno or some other microcontrollers.

- It draws a current of 20mA from the microcontroller during the trigger signal.
- It can relay the current up to 10A in a voltage of 30VDC or 220VAC. There are control LEDs for every single relay.
- Relays are triggered by logical 0 (0V).
- For every single relay, NC, NO and COM legs are ousted. Thus, they can be utilized when it is desired to be a short fuse in the event of a triggering or open-circuit in case of triggering.

Fig. A.19. 5V Relay

A.11 Selenoid Valve

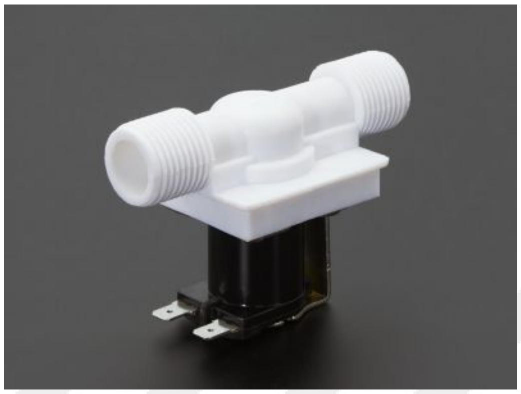


Fig. A.20. Selenoid Valve

Technical Features:

- Operating Pressure: 0.02 Mpa - 0.8 Mpa
- Operating Temperature: 1 °C - 75 °C
- Valve opening duration: less than 0.15 sec
- Valve closing duration: less than 0.3 sec
- Operating voltage: 12VDC
- Operating Lifespan: greater than 50 million times on and off
- Weight: 122gr
- Dimensions: 76.2 x 57.15 x 50.8 (mm)

A.12 Peristaltic Liquid Motor

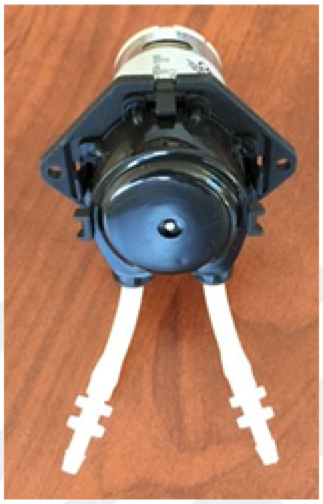


Fig. A.21. Peristaltic Liquid Motor

Technical Features:

- Flow: 3 let/hours
- Voltage: 24VDC
- Power: 5W
- Waterspout: Pharmed BPT, external radius: 4.5mm, internal radius: 2.5mm
- Measurements: 56x40x70mm

A.13 Liquid Pump

This pump used in the system, which has a liquid bearing capacity of 750 gallons (2840 litres) per hour, draw a current of 2.5A under 12V of voltage. It has a completely submerging structure and it is liquid-cooled.

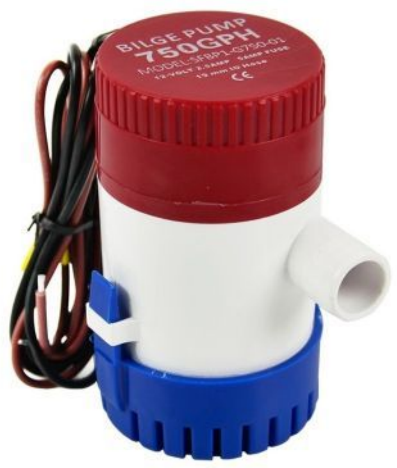


Fig. A.22. Liquid Pump

Technical Features:

- Operating voltage: 12V
- Operating current: 2,5A
- Liquid Bearing Capacity: 750 gallons/2840 litres per hour
- 3/4 inc (20mm) nozzle
- Power cable length: 1m

A.14 Ultrasonic Distance Sensor

This ultrasonic sensor, which can make measurements with 3mm of precision from 2cm to 400cm, is utilized in various distance reading areas.

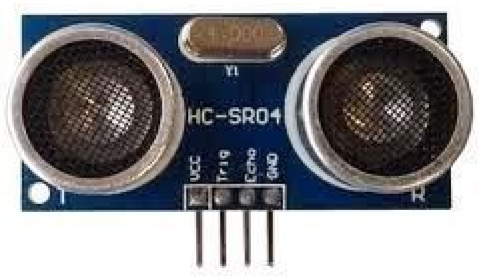


Fig. A.23. HC-SR04 Ultrasonic Distance Sensor

Technical Features:

- Operating voltage: DC 5V
- Current Drawn: 15 mA
- Operating Frequency: 40 Hz
- Maximum Visual Range: 4m
- Minimum Visual Range: 2cm
- Dimensions: 45mm x 20mm x 15mm

Appendix B

Source Codes

This sketch connects to the IoT builder (Cayenne) server using and runs the main communication loop.

The Library of IoT builder is required to run this sketch. If you have not already done so you can install it from the Arduino IDE Library Manager.

Steps:

- Set the token variable to match the Arduino token from the Dashboard.
- Set the network name and password.
- Compile and upload this sketch.

```
// Control Centre


#define CAYENNE_PRINT Serial
#include <SoftwareSerial.h>
#include <CayenneMQTTWiFi.h>
int val;
int tempPin = 0;
int EcPin = 4;

#define VIRTUAL_CHANNEL_BUZZER 1
#define VIRTUAL_CHANNEL_ECMOTOR 2
#define VIRTUAL_CHANNEL_PHMOTOR 3
#define VIRTUAL_CHANNEL_MAINMOTOR 4
#define VIRTUAL_CHANNEL_SAL 10
#define VIRTUAL_CHANNEL_EC 9
#define VIRTUAL_CHANNEL_TDS 11

char username[] = "";
char password[] = "";
char clientID[] = "";

// Your network name and password.
char ssid[] = "";
char wifiPassword[] = "";


#define rx2 3      //define what pin rx is going to be
#define tx2 2      //define what pin tx is going to be

//Define the pins.
#define BuzzerPin 8
#define EcMotorPin 4
#define PhMotorPin 5
#define MainMotorPin 9
```

```
//define how the soft serial port is going to work
SoftwareSerial myserial2(rx2, tx2);

//a string to hold incoming data from the PC
String inputstring2 = "";
//a string to hold the data from the Atlas Scientific product
String sensorstring2 = "";
//have we received all the data from the PC
boolean input_string_complete = false;
//have we received all the data from the Atlas Scientific product
boolean sensor_string_complete = false;



float f_ec;        //used to hold a floating point number that is the EC
float f_ec2;
float f_sal;       //used to hold a floating point number that is the SALINITY
float f_sal2;
float f_tds;       //used to hold a floating point number that is the TDS

 char sensorstring_array2[30];                    //we make a char array
 char *EC;
 char *SAL;
 char *TDS;
void setup()
{

  Serial.begin(9600);
  pinMode(BuzzerPin, OUTPUT);
  pinMode(EcMotorPin, OUTPUT);
  pinMode(PhMotorPin, OUTPUT);
  pinMode(MainMotorPin, OUTPUT);
  Cayenne.begin(username, password, clientID, ssid, wifiPassword);

  //set baud rate for the software serial port to 9600
  myserial2.begin(9600);
  //set aside some bytes for receiving data from the PC
  inputstring2.reserve(10);
  //set aside some bytes for receiving data from Atlas Scientific product
  sensorstring2.reserve(30);
 pinMode(EcPin, OUTPUT);

}

void serialEvent() {

 char inchar = (char)Serial.read();
 inputstring2 += inchar;
 if (inchar == '\r') {
 input_string_complete = true;
 }
}

void loop()
{
  Cayenne.loop();
  checkSensorEc();

}


void checkSensorEc(){


  myserial2. listen () ;
  delay(10);
    if (input_string_complete) {
 myserial2. print (inputstring2);
 inputstring2 = "";
 input_string_complete = false;
 }
 if (myserial2. available () > 0) {
 char inchar2 = (char)myserial2.read();
 sensorstring2 += inchar2;
 if (inchar2 == '\r') {
 sensor_string_complete = true;
 }
 }
 if (sensor_string_complete) {

//convert the string to a char array
  sensorstring2.toCharArray(sensorstring_array2, 30);
//let's pars the array at each comma
  EC = strtok(sensorstring_array2, ",");
  SAL = strtok(NULL, ",");
  TDS = strtok(NULL, ",");

  f_ec= (float)atof(EC);
  f_sal =(float)atof(SAL);
  f_tds =(float)atof(TDS);

  f_ec2= f_ec/1000;



 sensorstring2 = "";
 sensor_string_complete = false;
 }

}



CAYENNE_IN(VIRTUAL_CHANNEL_BUZZER)
{
  int value = getValue.asInt();
  CAYENNE_LOG("Channel %d, pin %d, value %d", VIRTUAL_CHANNEL_BUZZER,
      BuzzerPin, value);
  // Write the value received to the digital pin.
  digitalWrite(BuzzerPin, value);
}

CAYENNE_IN(VIRTUAL_CHANNEL_ECMOTOR)
{
  int value = getValue.asInt();
  CAYENNE_LOG("Channel %d, pin %d, value %d", VIRTUAL_CHANNEL_ECMOTOR,
      EcMotorPin, value);
  // Write the value received to the digital pin.
  digitalWrite(EcMotorPin, value);
}
```

```
CAYENNE_IN(VIRTUAL_CHANNEL_PHMOTOR)
{
  int value = getValue.asInt();
  CAYENNE_LOG("Channel %d, pin %d, value %d", VIRTUAL_CHANNEL_PHMOTOR,
       PhMotorPin, value);
  // Write the value received to the digital pin.
  digitalWrite(PhMotorPin, value);
}

CAYENNE_IN(VIRTUAL_CHANNEL_MAINMOTOR)
{
  int value = getValue.asInt();
  CAYENNE_LOG("Channel %d, pin %d, value %d",
       VIRTUAL_CHANNEL_MAINMOTOR, MainMotorPin, value);
  // Write the value received to the digital pin.
  digitalWrite(MainMotorPin, value);
}

CAYENNE_OUT(VIRTUAL_CHANNEL_SAL)
{
  // Read data from the sensor and send it to the virtual channel here.
   Cayenne.virtualWrite(VIRTUAL_CHANNEL_SAL, f_sal);
}

CAYENNE_OUT(VIRTUAL_CHANNEL_EC)
{
  // Read data from the sensor and send it to the virtual channel here.
   Cayenne.virtualWrite(VIRTUAL_CHANNEL_EC, f_ec2);
}

CAYENNE_OUT(VIRTUAL_CHANNEL_TDS)
{
  // Read data from the sensor and send it to the virtual channel here.
   Cayenne.virtualWrite(VIRTUAL_CHANNEL_TDS, f_tds);
}
```

```
// Humidity Measurement from The Soil

// Make sure you install the ESP8266 Board Package

#define CAYENNE_DEBUG
#define CAYENNE_PRINT Serial
#include <CayenneMQTTESP8266.h>
#include <ESP8266WiFi.h>
#include "CayenneArduinoDefines.h"

// WiFi network info.
char ssid[] = "";
char wifiPassword[] = "";

// Cayenne authentication info.
char username[] = "";
char password[] = "";
char clientID[] = "";

int value;
int humidity=0;

void setup() {
  Serial.begin(115200);
  Cayenne.begin(username, password, clientID, ssid, wifiPassword);
}

void loop() {
  Cayenne.loop();
   value = analogRead(humidity);
   value = map(value,0,1023,0,100);
   Serial.print("Value :");
   Serial.println(value);
   delay(500);

  Cayenne.virtualWrite(0, value);
  Cayenne.virtualWrite(2, value);
}
```

```
// Reservoir Liquid Level Adjustment and Temperature Detecting



#define CAYENNE_DEBUG
#define CAYENNE_PRINT Serial
#include <CayenneMQTTESP8266.h>
#include <ESP8266WiFi.h>
#include "CayenneArduinoDefines.h"
#define TRIGGER 5
#define ECHO  4
int val;
int tempPin = A0;

// WiFi network info.
char ssid[] = "";
char wifiPassword[] = "";

// Cayenne authentication info.
char username[] = "";
char password[] = "";
char clientID[] = "";



void setup() {

  Serial.begin(9600);
  Cayenne.begin(username, password, clientID, ssid, wifiPassword);
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
   pinMode(13, OUTPUT);
}

void loop() {
  Cayenne.loop();
  long duration, distance;
  digitalWrite(TRIGGER, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIGGER, HIGH);
  delayMicroseconds(10);

  digitalWrite(TRIGGER, LOW);
  duration = pulseIn(ECHO, HIGH);
  distance = (duration/2) / 29.1;

  Cayenne.virtualWrite(6, distance);
  delay(500);

 val = analogRead(tempPin);
float mv = ( val/1024.0)*5000;
int cel =(int) mv/10;
float farh = (cel*9)/5 + 32;
int cel1 = cel-10;

  Cayenne.virtualWrite(7, cel1);
  delay(10);
}
```

```
// Distribution Valve Control

#define CAYENNE_DEBUG
#define CAYENNE_PRINT Serial
#include <CayenneMQTTESP8266.h>
#include <ESP8266WiFi.h>
#include "CayenneArduinoDefines.h"

// WiFi network info.
char ssid[] = "";
char wifiPassword[] = "";

// Cayenne authentication info.
char username[] = "";
char password[] = "";
char clientID[] = "";


void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(115200);
  Cayenne.begin(username, password, clientID, ssid, wifiPassword);
}

void loop() {
  Cayenne.loop();
}
```

This is the source code of the system without using the IoT builder.

```
// Control Centre

#include <SPI.h>
#include <WiFi.h>
#include <SoftwareSerial.h>

// your network SSID (name)
char ssid[] = "";
 // your network password
char pass[] = "";
 // your network key Index number (needed only for WEP)
int keyIndex = 0;
int EcMotor = 4;
int DistributionMotor = 9;
#define rx2 3          //define what pin rx is going to be
#define tx2 2       //define what pin tx is going to be

 //a string to hold incoming data from the PC
String inputstring2 = "";
 //a string to hold the data from the Atlas Scientific product
String sensorstring2 = "";
 //have we received all the data from the PC
boolean input_string_complete = false;
 //have we received all the data from the Atlas Scientific product
boolean sensor_string_complete = false;



float f_ec;
float f_ec2;
float f_sal;
float f_sal2;
float f_tds;
  //we make a char array
 char sensorstring_array2[30];
 char *EC;
 char *SAL;
 char *TDS;
```

```
int status = WL_IDLE_STATUS;

WiFiServer server(80);
SoftwareSerial myserial2(rx2, tx2);

void setup() {


  pinMode(EcMotor, OUTPUT);
  pinMode(DistributionMotor, OUTPUT);


   // Initialize serial and wait for port to open:
  Serial.begin(9600);
   //set baud rate for the software serial port to 9600
  myserial2.begin(9600);
   //set aside some bytes for receiving data from the PC
  inputstring2.reserve(10);
  sensorstring2.reserve(30);



  while (!Serial) {
    ; // wait for serial port to connect.
  }

  // check for the presence of the shield:
  if (WiFi.status() == WL_NO_SHIELD) {
    Serial.println("WiFi shield not present");
    // don't continue:
    while (true);
  }

  String fv = WiFi.firmwareVersion();
  if (fv != "1.1.0") {
    Serial.println("Please upgrade the firmware");
  }

  // attempt to connect to Wifi network:
```

```
  while (status != WL_CONNECTED) {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(ssid);
    // Connect to WPA/WPA2 network. Change this line if using open or WEP network:
    status = WiFi.begin(ssid, pass);
    // wait 10 seconds for connection:
    delay(10000);
  }
  server.begin();
  // you're connected now, so print out the status:
  printWifiStatus();
}

void serialEvent() {

 char inchar = (char)Serial.read();
 inputstring2 += inchar;
 if (inchar == '\r') {
 input_string_complete = true;
 }
}

void loop() {

 myserial2.listen();
  delay(10);
   if (input_string_complete) {
 myserial2.print(inputstring2);
 inputstring2 = "";
 input_string_complete = false;
 }
 if (myserial2.available() > 0) {
 char inchar2 = (char)myserial2.read();
 sensorstring2 += inchar2;
 if (inchar2 == '\r') {
 sensor_string_complete = true;
 }
 }
 if (sensor_string_complete) {
  //convert the string to a char array
 sensorstring2.toCharArray(sensorstring_array2, 30);
  //let's pars the array at each comma
 EC = strtok(sensorstring_array2, ",");
 SAL = strtok(NULL, ",");
 TDS = strtok(NULL, ",");

  f_ec= (float)atof(EC);
  f_sal =(float)atof(SAL);
  f_tds =(float)atof(TDS);

  f_ec2= f_ec/1000;

  Serial.print("EC value: ");
  Serial.print(f_ec2);
  Serial.println(" ds/m");

sensorstring2 = "";
sensor_string_complete = false;
}

if(f_ec2 < 0.8){

  Serial.println("EC Motor is On");
  Serial.println("Distribution Motor is Off");
  digitalWrite(EcMotor, HIGH);
  digitalWrite(DistributionMotor, LOW);

  }
  else{
     Serial.println("EC Motor is Off");
     Serial.println("Distribution Motor is On");
     digitalWrite(EcMotor, LOW);
     digitalWrite(DistributionMotor, HIGH);
    }



  // listen for incoming clients
  WiFiClient client = server.available();
```

```
if ( client ) {
  Serial . println (”new client”);
  // an http request ends with a blank line
  boolean currentLineIsBlank = true;
  while (client .connected()) {
    if ( client . available ()) {
      char c = client.read();
      Serial . write(c);
      // if you’ve gotten to the end of the line (received a newline
      // character) and the line is blank, the http request has ended,
      // so you can send a reply
      if (c == ’\n’ && currentLineIsBlank) {
        // send a standard http response header
        client . println (”HTTP/1.1 200 OK”);
        client . println (”Content−Type: text/html”);
        // the connection will be closed after completion of the response
        client . println (”Connection: close”);
        // refresh the page automatically every 5 sec
        client . println (”Refresh: 5”);
        client . println ();
        client . println (”<!DOCTYPE HTML>”);
        client . println (”<html>”);
        client . println (”<head>”);
client . println (”<meta name=’apple−mobile−web−app−capable’ content=’yes’ />”);
client . println (”<meta name=’apple−mobile−web−app−status−bar−style’
    content=’black−translucent’ />”);
client . println (”</head>”);
client . println (”<body bgcolor = \”#f7e6ec\”>”);
client . println (”<hr/><hr>”);
client . println (”<h2><center> Smart Irrigation and Fertilization System
    </center></h2>”);
client . println (”<hr/><hr>”);
client . println (”<br><br>”);
client . println (”<br><br>”);
client . println (”<center>”);
client . println (”<h3>EC value: </h3>”);
client . println ( f_ec2 );
client . println (”  ds/m”);
client . println (”<br><br>”);
client . println (”<h3>Fertilization Status: </h3>”);
client . println ("</center>");
client . println ("<br>");
client . println ("<center>");

client . println ("<table border=\"5\">");
client . println ("<tr>");
if (f_ec2 < 0.8)
      {
        client . print ("<td>Fertilization Starts</td>");
      }
       else
       {
          client . print ("<td>Fertilization Stops</td>");
       }

      client . println ("<br />");

      client . println ("<tr>");
       if (digitalRead(EcMotor))
       {
         client . print ("<td>EC motor is On</td>");
      }
       else
       {
          client . print ("<td>EC motor is Off</td>");
       }
        client . println ("</tr>");
        client . println ("<tr>");
       if (digitalRead(DistributionMotor))
       {

         client . print ("<td>Distribution Motor is On</td>");

       }
       else
       {

          client . print ("<td>Distribution Motor is Off</td>");

       }
```

```
          client.println("</tr>");

          client.println("</tr>");


          client.println("</table>");

          client.println("</center>");
          client.println("</html>");
          break;
        }
        if (c == '\n') {
          // you're starting a new line
          currentLineIsBlank = true;
        } else if (c != '\r') {
          // you've gotten a character on the current line
          currentLineIsBlank = false;
        }
      }
    }
    // give the web browser time to receive the data
    delay(1);

    // close the connection:
    client.stop();
    Serial.println("client disonnected");
  }
}


void printWifiStatus() {
  // print the SSID of the network you're attached to:
  Serial.print("SSID: ");
  Serial.println(WiFi.SSID());

  // print your WiFi shield's IP address:
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);
  // print the received signal strength:
  long rssi = WiFi.RSSI();
  Serial.print("signal strength (RSSI):");
  Serial.print(rssi);
  Serial.println(" dBm");
}
```

```
//Liquid Level Adjustment

#include <ESP8266WiFi.h>

// Your network name and password.
const char* ssid = "";
const char* password = "";

#define TRIGGER 5
#define ECHO  4
int val;
int tempPin = A0;
int distance;
int  cell ;

;
WiFiServer server(80);

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(13, OUTPUT);
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  Serial.begin(115200);
  delay(10);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid,  password);

  while (WiFi.status() != WL_CONNECTED) {
    digitalWrite(LED_BUILTIN, HIGH);
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");


  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");
  digitalWrite(LED_BUILTIN, LOW);
}

void loop() {

  long duration, distance;
  digitalWrite(TRIGGER, LOW);
  delayMicroseconds(2);

  digitalWrite(TRIGGER, HIGH);
  delayMicroseconds(10);

  digitalWrite(TRIGGER, LOW);
  duration = pulseIn(ECHO, HIGH);
  distance = (duration/2) / 29.1;

  Serial.print("Liquid Tank Level = ");
  Serial.print(distance);
  Serial.println(" cm");


  delay(500);
```

```
 if (distance<=23 && distance >=11)
        {

        Serial.print("Liquid Tank Valve is On, ");
           digitalWrite(13, HIGH);
        }
         else  if (distance <11 && distance >=0)
         {

        Serial.print("Liquid Tank Valve is Off, ");
      digitalWrite(13, LOW);
       }

val = analogRead(tempPin);
float mv = ( val/1024.0)*5000;
int cel =(int) mv/10;
float farh = (cel*9)/5 + 32;
int cel1 = cel-10;
Serial.print("Temperature = ");
Serial.print(cel1);
Serial.print("*C");
Serial.println();

  delay(10);

  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {

    return;
  }

  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){

    delay(1);
  }
```

```
// Read the first line of the request
String request = client.readStringUntil('\r');
Serial.println(request);
client.flush();

// Return the response
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println("Refresh: 5");
client.println(""); // do not forget this one

client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.println("<head>");
client.println("<meta name='apple-mobile-web-app-capable' content='yes' />");
client.println("<meta name='apple-mobile-web-app-status-bar-style'
    content='black-translucent' />");
client.println("</head>");
client.println("<body bgcolor = \"#f7e6ec\">");
client.println("<hr/><hr>");
client.println("<h2><center> Smart Irrigation and Fertilization System
    </center></h2>");
client.println("<hr/><hr>");
client.println("<br><br>");
client.println("<br><br>");
client.println("<center>");
client.println("<h3>Temperature: </h3>");
client.print(cel1);
client.println("*C");
client.println("<h3>Liquid Tank Level: </h3>");
client.print(distance);
client.println(" cm");
client.println("<br><br>");
client.println("<h3>Liquid Tank Valve Status: </h3>");
client.println("</center>");
client.println("<br>");
client.println("<center>");
```

```
  client.println("<table border=\"5\">");
  client.println("<tr>");
  if (distance<=23 && distance >=11)
        {

           client.print("<td>Liquid Tank Valve is On</td>");

        }
         else  if (distance <11 && distance >=0)
         {

            client.print("<td>Liquid Tank Valve is Off</td>");

       }

          client.println("<br />");
          client.println("</tr>");
          client.println("</table>");

          client.println("</center>");
  client.println("</html>");
  delay(1);
  Serial.println("Client disonnected");
  Serial.println("");

}
```

```
// Soil Humidity and Distribution Valve

#include <ESP8266WiFi.h>

// Your network name and password.
const char* ssid = "";
const char* password = "";

int humidity=0;
int value;

; //
WiFiServer server(80);

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
  pinMode(13, OUTPUT);
  Serial.begin(115200);
  delay(10);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    digitalWrite(LED_BUILTIN, HIGH);
    delay(500);
    Serial.print(".");


  }

  Serial.println("");
  Serial.println("WiFi connected");
  // Start the server
  server.begin();
  Serial.println("Server started");


  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");
  digitalWrite(LED_BUILTIN, LOW);
}

void loop() {

   value = analogRead(humidity);
   value = map(value,0,1023,0,100);
   Serial.print("Soil Dryness: % ");
   Serial.println(value);
   delay(500);

    if (value<=100 && value >=75)
        {

        Serial.print("Distribution Valve is On, ");
           digitalWrite(13, HIGH);
        }
         else if (value<75 && value >=50)
         {

        Serial.print("Distribution Valve is Off, ");
          digitalWrite(13, LOW);
       }
```

```
// Check if a client has connected
WiFiClient client = server.available();
if (!client) {

  return;
}

// Wait until the client sends some data
Serial.println("new client");
while(!client.available()){

  delay(1);
}

// Read the first line of the request
String request = client.readStringUntil('\r');
Serial.println(request);
client.flush();

// Match the request


if (request.indexOf("/valveOn") > 0) {
  digitalWrite(13, HIGH);

}
if (request.indexOf("/valveOff") >0) {
  digitalWrite(13, LOW);

}

// Return the response
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println("Refresh: 5");
client.println(""); // do not forget this one

client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.println("<head>");
```

```
client . println ("<meta name='apple−mobile−web−app−capable' content='yes' />");
client . println ("<meta name='apple−mobile−web−app−status−bar−style'
    content='black−translucent' />");
client . println ("</head>");
client . println ("<body bgcolor = \"#f7e6ec\">");
client . println ("<hr/><hr>");
client . println ("<h2><center> Smart Irrigation and Fertilization System
    </center></h2>");
client . println ("<hr/><hr>");
client . println ("<br><br>");
client . println ("<br><br>");
client . println ("<center>");
client . println ("<h3>Soil Dryness: </h3>");
client . println ("% ");
client . println (value);
client . println ("<br><br>");
client . println ("<h3>Valve & Soil Status: </h3>");
client . println ("</center>");
client . println ("<br>");
client . println ("<center>");
client . println ("<table border=\"5\">");
client . println ("<tr>");
if (value<=100 && value>=75)
       {
         client . print ("<td>Soil is Dry</td>");
       }
        else  if (value<75 && value >=50)
        {
          client . print ("<td>Soil is Normal</td>");
      }
    else if (value<50 )
        {
          client . print ("<td>Soil is Wet</td>");
      }
      client . println ("<br />");
      client . println ("<tr>");
```

```
          if  (digitalRead(13))
           {

             client . print(”<td>Valve is ON</td>”);

          }
           else
           {

              client . print(”<td>Valve is OFF</td>”);

           }
           client . println(”</tr>”);

           client . println(”</tr>”);


           client . println(”</table>”);

           client . println(”</center>”);
  client . println(”</html>”);
  delay(1);
  Serial . println(”Client  disonnected”);
  Serial . println(””);

}
```

BIBLIOGRAPHY

[1] Ciğer, M., (2010), *Computer Controlled, Internet-Aided Greenhouse Automation*. MSc Thesis, Çukurova University, The Gradute School of Natural and Applied Sciences, Adana.

[2] Çoruh, B., (2008), *A system calibrates to the parameters of pressure, temperature and humidity*. MSc Thesis, Bakent University, The Gradute School of Natural and Applied Sciences, Ankara.

[3] Dukes, D.M., Simonne, H.E., Davis E.W., Studstill, W., Hochmuth, R., (2003). *Effect of Sensor-Based High Frequency Irrigation on Bell Pepper Yield and Water Use*. Proceedings 2nd International Conference on Irrigation and Drainage, Phoenix, AZ, May 12–15, 665–674.

[4] Kırnak, H., (2006). *Automatc Irrigaton Based on Soil Moisture for Nursery Crops* . GAP V. Engineering Congress, Harran University. Engineering Department, 26–28 April, Şanlurfa, 1540–1547.

[5] Millaa, K., Kishb, S., (2006). *A Low-cost Microprocessor and Infrared Sensor System for Automating Water Infiltration Measurements*. Computers and Electronics in Agriculture, 53, 122–129.

[6] İnan, S.A., (2002), *Meyve Fidanı Çoğaltılmasında Kullanılan Köklendirme Seralarının Otomasyonu*. MSc Thesis, SDU, The Gradute School of Natural and Applied Sciences, Isparta.

[7] Shivaprasad B., Ravishankara M., B. N. Shoba, (June 2014), *Desing and Implementation of Seeding and Fertilizing Agriculture Robot.* International Journal of Application or Innovation in Engineering and Management,Volume 3, Issue 6.

[8] Yule,D., (1989), Fundamental Science Encyclopedia, 313–314, Istanbul.

[9] Mesas-Carrascosa F., Verdu D., Merono J., Sanchez de la Orden M., (2015), *A 2015 Biosystems Engineering.* `http://www.sciencedirect.com/science/article/pii/S1537511015001208`

[10] Dale Wheat (2011), Arduino internals Paul Manning Publishers.

[11] Taşdemir, C., (2011), *Arduino*, no.19708, 1.press, Istanbul.

[12] Guofang, L., Lidong, C., Yubin, Q., Shengtao, L., Junyu X.,(2010), *Remote Monitoring System of Greenhouse Environment Based on LabVIEW.* International Conference on Computer Design and Applications (ICCDA), 89–92, Qinhuangdao.

[13] Henkoğlu, T., Külcü, O., (2013), *Cloud Computing as an Information Access Platform: A Study on Threats and Legal Requirements.* Bilgi Dünyası,14 (1), 62–86.

[14] Viscarra, R.A., McBratney, A.B.,(1999), *Calibration of a lime requirement buffer for site-specific lime applications in south-eastern Australia*, In: Stafford J.V., editor. Precision Agriculture: Proceedings of the 2nd European Conference on Precision Agriculture Part I; Sheffield, UK: Sheffield Academic Press, pp. 429-440.

[15] Williston, H.L., LaFayette, R., (1978), *Species suitability and pH of soils in southern forests.* USDA Forest Service. Southeastern Area, state and Private Forestry. Forest Management Bulletin. 4p.

[16] Londo, J., Kushla, D., Carter, C., (2010), *Soil pH and Tree Species Suitability in the South Andrew*, Jacksonville State University,

[17] Melgar, E.R., Diez, C.C., (2012), *Arduino and kinect projects*, Distributed to the book trade world wide by Springer Science Business Media, p.p 450–500, New York.

[18] Kürklü, A., Çaglayan, N., (2015), *A Study on the Development of Greenhouse Automation Systems*, 18(1), 25–34, Mediterranean University Faculty of Agriculture Journal, Antalya.

[19] Michael Margolis (2011). Arduino cookbook O Reilly Media,Inc.

[20] Texas Instrument Datasheet.
`http://www.datasheetcatalog.com/texasinstruments/1/`

[21] Atlas Scientific Datasheet.

VITA

Muhammed Akif Yenikaya was born in Erzurum, Turkey, on April 16, 1992, the son of Selahattin Yenikaya and Gülten Yenikaya. He completed his high school education in Kars. He completed his B.S degree in 2015. He received his B.S. degree in 2015 from the Department of Computer Engineering, After receiving his B.S degree with first rank, he continued his studies.