**Project Report**

**on**

**Automatic Irrigation System**

**to be developed to fulfill the requirements for**

**Major Project (CA133)**

**Submitted to**

**Department of Computer Applications**

**Chitkara University, Punjab**



**under the supervision of**

**Dr. Preetinder Brar (Associate Professor)**

**Submitted by**

**Eashan Verma**

**1910992560**

**Bachelor of Computer Applications**

**(Batch 2019-22)**

**CERTIFICATE**

This is to certify that the report on IoT based project titled “Automatic Irrigation System” submitted by Eashan Verma (1910992560) to the Department of Computer Applications, Chitkara University, Punjab in partial fulfillment for the completion of the course Major Project (CA133) in the fifth semester of Bachelor of Computer Applications is a Bona fide record of work carried out under my supervision.

**Date:13th Jan 2022 Dr. PREETINDER BRAR**

Associate Professor

Department of Computer Application

Chitkara University, Rajpura, Punjab

**ABSTRACT OF THE PROJECT**

Too much water actually is bad for most of our trees and plants. If soil becomes waterlogged and stays that way, the water can cut off the air supply to the roots and to the microorganisms that live in healthy soil. It can lead to root rot or other diseases and same goes for the situation when water is less than normal in the soil. A low-cost Arduino based Automatic Irrigation system using Soil moisture sensor is presented in this paper in which the soil moisture sensor gives its output depending on the conditions of the soil and later with the help of Arduino it gets worked. As agriculture is given higher priority in the life of an economy so for the better agricultural growth, water is essential factor. Lack of enough water and excess of water leads to damage of plants. So, we need an effective and efficient technology for better farming. Thus, the usage of a low cost arduino based automatic irrigation system using soil moisture sensor is expected to be useful to for the irrigation process in agriculture. This system requires an Arduino-UNO which contains ADC converter in it. A soil moisture sensor is the main component in this system which is used to measure the conditions of the soil like whether the soil is dry or wet. For displaying the conditions of the soil, 16×2 LCD display is the better choice to display and at last a motor is used to pump the water to the plant or crops.

Soil needs to have the medium amount of water/moisture according to conditions, so that the roots don’t die. My project ‘AIS’ i.e., Automatic Irrigation System works on this problem and is beneficial for any person, the one working in their fields, the one having plants in their house gardens, in nurseries or anywhere.

**INTRODUCTION**

Now-a-days it's been a big challenge for conservation of the water as nearly 80% of the water present on the earth is used for agricultural purpose. As population is increasing day by day, the demand for food never dies and so some smart technologies have been implemented for the better agricultural growth. Considering the electronics region, many smart projects have been implemented for agricultural purpose and one of them is A Low-Cost Arduino based Automatic Irrigation System using soil moisture sensor. It automatically turns ON/OFF depending on the conditions of the soil. If the soil is in dry condition, then the motor turns ON automatically and in case if it is in Wet condition, then it automatically gets turned OFF. An Automatic Irrigation system not only turns ON/OFF depending on the soil conditions but also reduces the human interference and human labour towards the work. It also saves the time and also the water as excess water leads to damage of the crops. During summers, a great many people are as well languid to water the pruned plants on their roof gardens consistently. In the realm of advance gadgets life of individuals ought to be less complex thus to make life less complex and advantageous, so I have made "Automatic Irrigation System".

**\*System Requirements: -**

1. **Product Definition: -**

\***Problem Statement:**

Many farmers nowadays, unknowingly, pour more water in the soil than required which makes it watery and soil loses its nutrients which makes crops difficult to grow on such soil. Our system, that is a low-cost irrigation system, will help them to get to know about the right amount of water to be put in the soil to make it worth planting the crops. It is a low-cost and easy to use system and will be feasible with farmers and anyone fascinated with the plants at their homes can to use so that flowers don’t shed their grace.

**\*Processing Elements: -**

* **Desktop Client: -**

|  |  |
| --- | --- |
| RAM | 12 Gigabytes |
| Processor | Intel core i-5 |
| OS version | Window 10 Home |
| Space | 1. gigabytes |

**\*Hardware: -**

* 16×2 LCD.
* 5v adapter.
* Relay circuit 5V.
* Potentiometer 10 ohm.
* 5v wires acting as float switch connected to potentiometer.
* 1k ohm resistor.
* Pump.
* Float pipe.
* Soil Moisture

**\* Software: -**

* Operating System: Window 10.
* Arduino IDE.

**b) Programming Languages and Development Tools:**

**\*Programming Language: -**

**Arduino Embedded C**: - It is a hardware dependent form of C language that depends on the semantics of the hardware for which you are writing your code. It is hardware dependent in nature. The basics concepts of C languages is the same. What changes is how you will access ports, pins and various peripheral of your board. Technology is constantly changing. New microcontrollers become available every year. The one thing that has stayed the same is the C programming language used to program these microcontrollers. Arduino is the hardware platform used to teach the C programming language as Arduino boards are available worldwide and contain the popular AVR microcontrollers. The reason why one should use embedded C for coding arduino is because it gives you a lot of control over how you will program your arduino board for creating some Embedded systems Application.

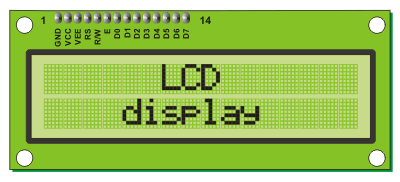
**\*Development Tools: -**

**1) Arduino Uno: -** Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analogue inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong, worst-case scenario you can replace the chip for a few dollars and start over again.

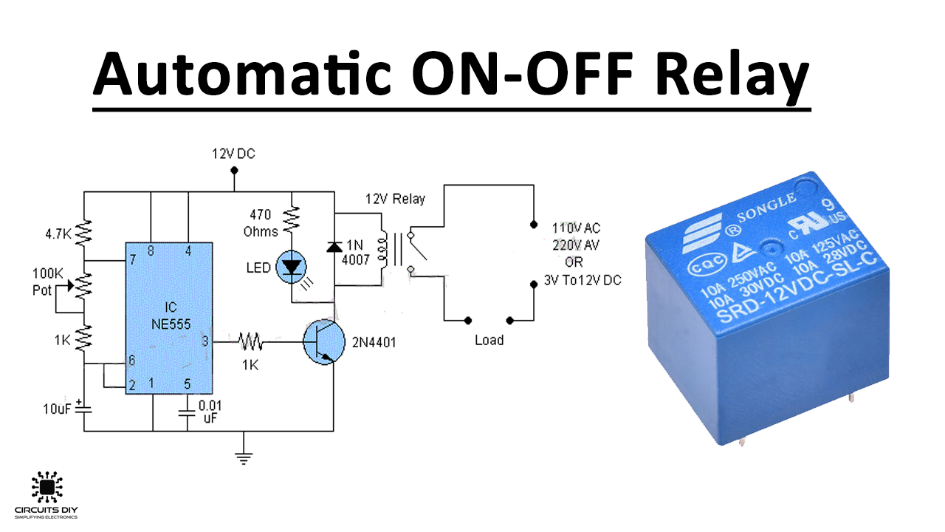
"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.



**2) LCD Display: -** These components are “specialized” for being used with the microcontrollers, which means that they cannot be activated by standard IC circuits. They are used for writing different messages on a miniature LCD.



**3) Relay Circuit**: - The relay circuits used for controlling relays often use semiconductor devices. Although the simplest relay circuits would simply involve a switch closing a circuit, the applications of relays often require a small signal, possibly from some form of a microcontroller circuit or other device to actuate the relay. When driven in this way, it is necessary to some form of semiconductor driver. The simplest is a bipolar transistor, although FETs work equally well. The relay is actuated by a coil. This creates the magnetic field that is used to actuate the relay, whether it is a reed relay or an electromechanical relay. This means that when the semiconductor switch is in its ON state, current will start to flow. It will rise gradually as a result of the inductance and this will mean that there will be a certain time before the relay actuates. However, when the switch is suddenly opened a large back EMF will be generated. This could be sufficiently large to damage the driver device.

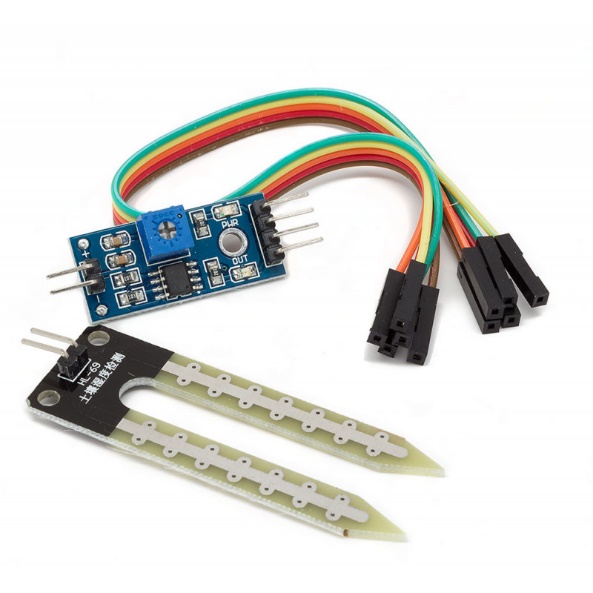


**4) Soil moisture sensor: -**

Soil moisture sensors measure the volumetric [water content](https://en.wikipedia.org/wiki/Water_content) in [soil](https://en.wikipedia.org/wiki/Soil). Since the direct [gravimetric measurement](https://en.wikipedia.org/wiki/Gravimetric_analysis) of free soil moisture requires removing, drying, and weighing of a sample, soil moisture sensors measure the volumetric water content indirectly by using some other property of the soil, such as electrical resistance, dielectric constant, or interaction with [neutrons](https://en.wikipedia.org/wiki/Neutron), as a proxy for the moisture content.

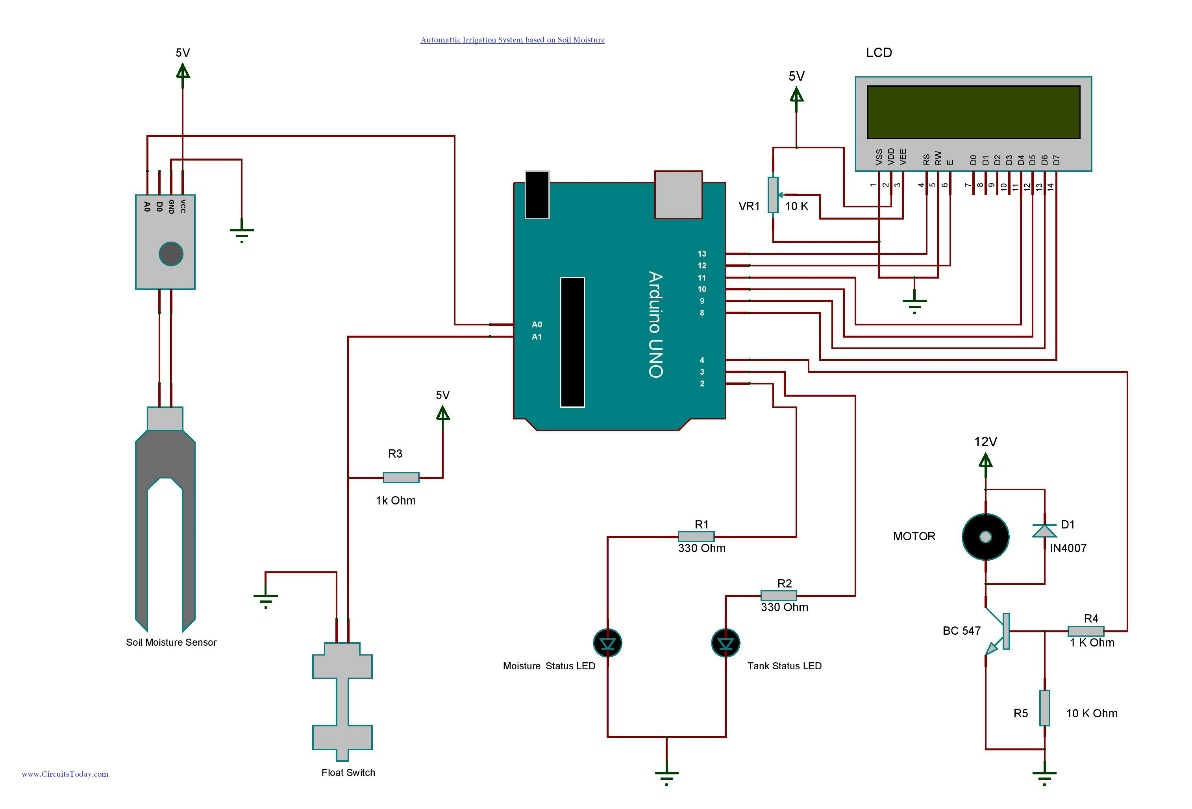
The relation between the measured property and soil moisture must be calibrated and may vary depending on environmental factors such as [soil type](https://en.wikipedia.org/wiki/Soil_type), [temperature](https://en.wikipedia.org/wiki/Temperature), or [electric conductivity](https://en.wikipedia.org/wiki/Electric_conductivity). Reflected [microwave](https://en.wikipedia.org/wiki/Microwave) radiation is affected by the soil moisture and is used for [remote sensing](https://en.wikipedia.org/wiki/Remote_sensing) in [hydrology](https://en.wikipedia.org/wiki/Hydrology) and agriculture. Portable probe instruments can be used by farmers or gardeners.

Soil moisture sensors typically refer to sensors that estimate volumetric water content. Another class of sensors measure another property of moisture in soils called [water potential](https://en.wikipedia.org/wiki/Water_potential); these sensors are usually referred to as soil water potential sensors and include [tensiometers](https://en.wikipedia.org/wiki/Tensiometer_(soil_science)) and gypsum blocks.

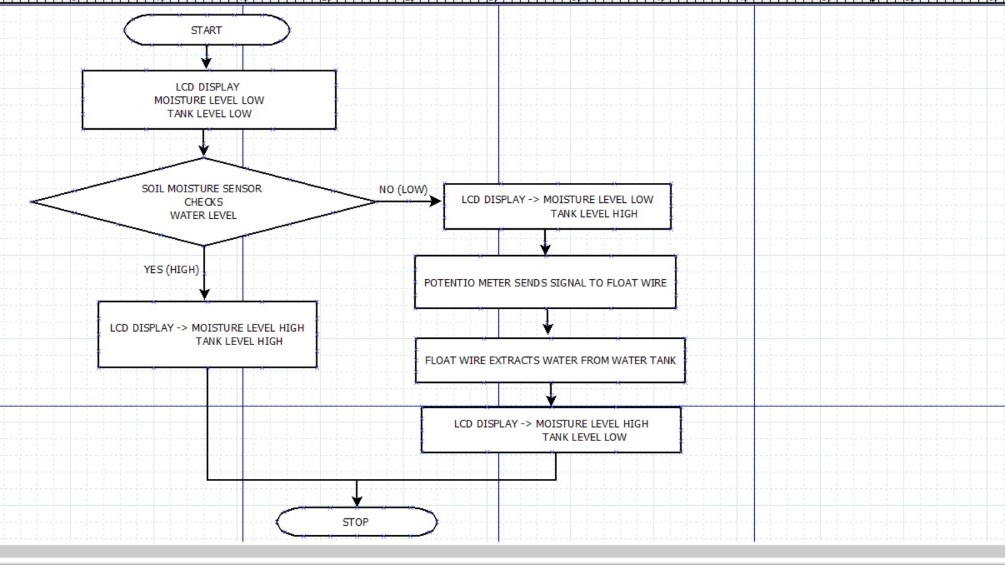


**\*Design: -**

**1. Architectural Design (DFD): -**



**2. Working Model (DFD): -**



**\*Test Plan**

**Test Case Table:**

|  |  |  |  |
| --- | --- | --- | --- |
| Description | Expected result | Actual outcome | Remarks |
| Soil with more moisture. | Water will not float from tank to another container containing soil. | Same. | Pass. |
| Soil with less moisture. | Water will float from tank to another container containing soil. | Same. | Pass. |

1. **Performance Tests: -**

In our project, we have also performed the performance testing which is used to check the speed, response time, stability, reliability, scalability and resource usage under a particular workload. The performance of ‘Automatic Irrigation System’ is measured in the terms of the output provided by the system. These are broadly classified as:

**-Portability –** The system is easy to use and easily understandable.

**-Availability –** The system is easy to assemble.

**-Reliability –** The systemis responsible for displaying the accurate results to the user.

**-Maintainability –** Any type of modification can be easily done with less effort.

**\*Project Legacy**

**\*Current status of project: -** Automatic Irrigation System is working perfectly and giving accurate results as expected by the user. All the modules in this project are working fine. The user can access the system anytime.

**\*Remaining areas of concerns: -** Coding basic errors of storage issues.

**\*Technical and managerial lesson learnt: -**

-We have learnt about the Arduino Technology and its properties.

-We have got hardware and materialistic knowledge.

**\*Future Recommendations: -**

-Data can be transferred into excel sheet using database.

-This project can be uploaded online to make it accessible worldwide.

**BIBLIOGRAPHY**

1. <https://www.arduino.cc/en/software> (Arduino IDE)

2. <https://en.wikipedia.org/wiki/Soil_moisture_sensor> (Soil Moisture Sensor)

3. <https://www.arduino.cc/>

4. <https://www.elektor.com/c-programming-with-arduino>

5. <https://www.circuitstoday.com/automatic-irrigation-system-arduino>

**Source Code: -**

#include<LiquidCrystal.h>

#define moisture\_sensorPin A0

#define float\_switchPin A1

#define motorPin 4

#define soil\_statusPin 2

#define tank\_statusPin 3

LiquidCrystal lcd(13,12,11,10,9,8);

const int avg\_moisture = 800;

void setup()

{

Serial.begin(9600);

lcd.begin(16,2);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Soil Moisture ");

lcd.setCursor(0,1);

lcd.print("Sensor ");

delay(2000);

pinMode(moisture\_sensorPin,INPUT);

pinMode(float\_switchPin,INPUT);

pinMode(motorPin,OUTPUT);

pinMode(soil\_statusPin,OUTPUT);

pinMode(tank\_statusPin,OUTPUT);

digitalWrite(motorPin,LOW);

digitalWrite(soil\_statusPin,LOW);

digitalWrite(tank\_statusPin,LOW);

}

void loop()

{

lcd.begin(16,2);

lcd.setCursor(0,0);

lcd.print(" MOISTURE - ");

if(analogRead(moisture\_sensorPin) > avg\_moisture){

lcd.print("HIGH");

digitalWrite(soil\_statusPin,HIGH);}

if(analogRead(moisture\_sensorPin) < avg\_moisture){

lcd.print(" LOW");

digitalWrite(soil\_statusPin,LOW);}

lcd.setCursor(0,1);

lcd.print("TANK LEVEL- ");

if( digitalRead(float\_switchPin) == HIGH){

lcd.print("HIGH");

digitalWrite(tank\_statusPin,LOW);}

if( digitalRead(float\_switchPin) == LOW){

lcd.print(" LOW");

digitalWrite(tank\_statusPin,HIGH);}

digitalWrite(motorPin,LOW);

if(analogRead(moisture\_sensorPin) < avg\_moisture && digitalRead(float\_switchPin) == HIGH)

{

while(analogRead(moisture\_sensorPin) < avg\_moisture && digitalRead(float\_switchPin) == HIGH)

{

lcd.setCursor(0,0);

lcd.print(" MOISTURE - LOW ");

lcd.setCursor(0,1);

lcd.print(" MOTOR IS ON ");

digitalWrite(soil\_statusPin,LOW);

digitalWrite(tank\_statusPin,LOW);

digitalWrite(motorPin,HIGH);

}

if(analogRead(moisture\_sensorPin) > avg\_moisture){

lcd.setCursor(0,0);

lcd.print(" MOISTURE - HIGH");

lcd.setCursor(0,1);

lcd.print(" MOTOR - OFF ");

digitalWrite(soil\_statusPin,HIGH);

digitalWrite(motorPin,LOW);

delay(3000);}

if(digitalRead(float\_switchPin) == LOW){

lcd.setCursor(0,0);

lcd.print(" TANK LEVEL- LOW");

lcd.setCursor(0,1);

lcd.print(" MOTOR - OFF ");

digitalWrite(tank\_statusPin,HIGH);

digitalWrite(motorPin,LOW);

delay(3000);}

}

delay(500);

}