Smart Irrigation System using Arduino (TinkerCAD Simulation)

Project Report

Author: Atufa Vora

Organization: MacroEdTech

Guide: Sagar Sakalley

April 4, 2026

Aim

Designing and simulating a Smart Irrigation System with an Arduino UNO via TinkerCAD which will measure the moisture content of the soil and automatically control the watering process according to soil condition in real-time.

Abstract

Automated irrigation plays a key part in today's farming practices. Water oversight improves when manual effort drops, thanks to such technology. This work introduces a setup built around an Arduino chip, modeled using TinkerCAD software. Moisture levels underground guide watering actions without delay. Operation adjusts itself based on real-time ground conditions.

Embedded within modern farming tools, small computing devices help manage crops more accurately while keeping expenses down. Because these setups sense conditions, handle data, and adjust operations efficiently, they fit well into tight budgets. Electrical traits of earth shift when water levels do - this fact allows machines to track dampness without guesswork. When dirt holds less moisture, its ability to carry current drops, creating measurable differences. An open-source circuit board takes incoming readings from probes, changing raw voltages into numbers a program can use. Instead of relying on assumptions, automated choices emerge through calculated rules built into the code.

Starting with simulation in TinkerCAD, the full setup gets tested for circuit operation alongside controller behavior, while also enabling live tracking. Efficiency appears clearly in water use because irrigation happens solely once soil dampness drops under a preset threshold.

1 Introduction

1.1 Background

One major issue today involves limited access to fresh water, directly impacting how much food farms can grow. Because farming uses vast amounts of water across the planet, outdated systems frequently waste it - losing supplies to seepage, excess flow, or timing errors. Since better methods reduce such losses, smarter watering supports both crop yields and ecosystem health.

Despite appearances, older watering methods depend entirely on human checks or rigid timers - ignoring shifts in ground moisture. Because sensors track surroundings nonstop, adjustments happen instantly, guided by live readings instead of guesswork.

Starting with a microcontroller, farming tasks can be automated affordably at various scales. Instead of manual checks, devices like Arduino link sensors, computers, and mechanical parts into small self-running setups. Because they work on their own, these tools cut down hands-on labor significantly. With time, precision increases - alongside smarter use of water, energy, or fertilizer. Fewer mistakes happen when operations follow sensor-driven logic.

1.2 Scientific Motivation

Soil is a complex physical medium consisting of three primary phases:

- Solid particles (minerals and organic matter)
- Water contained within pore spaces
- Air occupying remaining voids

Water movement, electrical behavior, and nutrient transport shift based on the distribution of these stages. As roots absorb moisture, plant functions such as leaf evaporation and cellular chemistry adjust accordingly.

Beneath wet conditions, earth alters electrical behavior. Since moisture holds small charged bits, conductivity rises noticeably. As a result, resistance inside dirt drops quite clearly. The effective conductivity σ of soil can be expressed conceptually as:

$$\sigma = f(\theta, C_i, T)$$

where θ represents volumetric water content, C_i denotes ionic concentration, and T is temperature. Increased moisture leads to higher conductivity and lower resistivity.

Plants thrive only when water amounts remain tightly balanced. Without enough wetness, nutrients crawl too slowly through earth. Should ground soak up too much rain, oxygen gets pushed out by liquid. Breathing turns hard for roots in such soggy zones. Watching soil moisture prevents either end of the spectrum from causing harm. Over time, monitoring changes makes it possible to act early, stopping problems before they grow. When moisture levels stay consistent, roots and shoots progress at an even pace. Growth gains balance when water supply does not swing widely.

1.3 Embedded Systems in Agriculture

Modern farming machines are equipped with small computers that have sensors which monitor the situations constantly. Since these computers process information around the clock, machines react instantly if needed. The hardware and software do not work independently but interact and adjust constantly according to changing situations. Therefore, small changes in environments cause immediate reactions as needed. The speed is most significant when changing environments are unpredictable.

Machines can sense dampness in the air through the usage of detectors. Then, depending on certain set limits, a decision is taken by the small computer with regard to irrigation systems. Since changing situations are constantly taking place in the environment, machines react instantaneously.

$$\text{Sensor} \rightarrow \text{Controller} \rightarrow \text{Actuator} \rightarrow \text{Environment}$$

Real time sensing is useful for making adaptive decisions in irrigation systems based on changing soil conditions, not time schedules.

2 Objectives of the Project

The major goal of this project is to design and simulate an automated system of irrigation using Arduino, which incorporates physical sensors and mathematical decision-making techniques in order to optimize water usage in agricultural environments.

1. **Monitoring Soil Moisture using Sensor Physics**

   The system will be designed in a manner such that it will measure the moisture in the soil using physical sensors, which are based on electrical properties of the soil. The electrical properties of the soil change with moisture levels, which are a result of the movement of ions in the moisture contained in the soil.

2. **Conversion of Analog Signals into Digital Data**

   The soil moisture sensor will provide a voltage level that is proportional to the moisture in the soil. The Arduino UNO will incorporate an Analog to Digital Converter (ADC), which will convert analog signals to numerical values. The conversion of analog signals to digital signals can be represented by:

   $$D = \frac{V_{in}}{V_{ref}} \times (2^n - 1)$$

   where D represents the digital output value, V_{in} is sensor voltage, V_{ref} is reference voltage (5V), and $n = 10$ for the 10-bit ADC of the ATmega328P.

3. **Implementation of Mathematical Decision Making**

   A threshold-based control algorithm is employed to control the irrigation requirements. The algorithm compares the moisture level with the threshold level:

   $$\text{If } M < M_{threshold} \Rightarrow \text{Pump ON}$$
   $$\text{If } M \geq M_{threshold} \Rightarrow \text{Pump OFF}$$

   This logical comparison forms the basis of automated decision making.

4. **Automation of Irrigation Control**

   "The Arduino micrpcontroller is used as the controller to control the irrigation pump by sending digital signals. This provides feedback control because the controller is directly affected by the sensors."

5. **Simulation of System Behavior**

   The entire irrigation system is simulated using the TinkerCAD platform to understand the circuit operation, control, and reliability with different soil moisture levels.

3 Physics of Soil Moisture Measurement

3.1 Electrical Conductivity of Soil

The soil acts like a resistor with changing electrical resistance, which varies considerably with water content. The moisture in the soil contains ions, which are helpful in carrying electrical current through the soil.

The relationship between voltage, current, and resistance is expressed by Ohm's Law:

$$V = IR$$

where:

- V = applied voltage across the soil
- I = current flowing through the soil
- R = electrical resistance of soil

If there is more moisture in the soil, then more current will pass through the soil with a given voltage, which is expressed as:

$$\text{Wet soil} \Rightarrow \text{Lower resistance}$$

$$\text{Dry soil} \Rightarrow \text{Higher resistance}$$

The soil moisture sensor detects this change in resistance and gives a voltage output, which can be read by the Arduino microcontroller.

3.2 Resistivity Model

Electrical resistance of soil can be modeled using the resistivity relation:

$$R = \rho \frac{L}{A}$$

where:

- ρ = soil resistivity
- L = distance between sensor probes
- A = effective contact area between soil and probes

The resistivity of soil decreases with increased moisture, as water contains ions in solution, which help in carrying electrical current.

$$\rho \propto \frac{1}{\text{Moisture Content}}$$

This physical principle forms the basis of resistive soil moisture sensing.

3.3 Capillary Physics in Soil

The water retention in the soil is due to capillary action, which takes place in microscopic spaces between two particles in the soil.

Capillary rise is described by:

$$h = \frac{2\gamma \cos\theta}{\rho g r}$$

where:

- h = capillary rise height
- γ = surface tension of water
- θ = contact angle between water and soil surface
- ρ = density of water
- g = gravitational acceleration
- r = radius of soil pore

The size of the pores in the soil decreases with increased capillary rise, which enables the soil to hold moisture despite gravity. This explains the non-uniform moisture distribution in the soil, which is detected by moisture sensors in the soil.

4 Mathematical Modeling of Control System

4.1 Analog to Digital Conversion (ADC)

The Arduino UNO has a 10-bit Analog to Digital Converter (ADC) circuit that converts the analog signal obtained from the soil moisture sensor into digital form. Since a 10-bit ADC has $2^{10} = 1024$ quantization levels, the analog signal is converted into a digital signal ranging from 0 to 1023.

The conversion can be written as:

$$D = \frac{V_{in}}{V_{ref}} \times (2^{10} - 1)$$

where:

- D = digital output value (0–1023)
- V_{in} = input voltage from soil moisture sensor
- V_{ref} = reference voltage (5V for Arduino UNO)

Therefore, any variation in soil resistance causes a variation in voltage.

4.2 Moisture Threshold Logic

The decision regarding irrigation is based on a threshold value of M_T. The Arduino board is programmed to continuously read the moisture value and make a decision based on a threshold value of M_T as follows:

$$M = \text{Measured Moisture Value}$$

Control logic is defined as:

$$\text{If } M < M_T \Rightarrow \text{Pump ON}$$
$$\text{If } M \geq M_T \Rightarrow \text{Pump OFF}$$

This mathematical formulation transforms a sensor into a decision-making tool. The threshold value of M_T is determined experimentally based on soil types and irrigation requirements of plants.

4.3 Control System Concept

The smart irrigation system works in a closed loop feedback control system. The sensor measurements have an impact on the control of the actuators, which in turn impacts the physical environment.

The feedback loop can be represented as:

$$\text{Sensor} \rightarrow \text{Arduino Controller} \rightarrow \text{Decision Logic} \rightarrow \text{Pump} \rightarrow \text{Soil} \rightarrow \text{Sensor}$$

In control theory terms:

- Sensor acts as the measurement element
- Arduino functions as the controller
- Pump serves as the actuator
- Soil moisture represents the controlled variable

The objective of the system is to minimize the moisture error:

$$e(t) = M_T - M(t)$$

where $e(t)$ is the difference between desired moisture level and measured moisture. The controller responds by activating irrigation until the error approaches zero, achieving stable soil moisture conditions.

5 Arduino UNO R3 Architecture

5.1 ATmega328P Microcontroller

The Arduino UNO R3 is built around the ATmega328P microcontroller, which serves as the computational core of the embedded system. It executes programmed instructions, processes sensor inputs, and controls output devices such as actuators and pumps.

The microcontroller operates at a clock frequency of:

$$f = 16 \text{ MHz}$$

which means the processor executes approximately 16×10^6 clock cycles per second. Each instruction is synchronized with this clock signal, enabling deterministic real time control.

Digital Logic Processing

The ATmega328P performs digital operations using binary logic based on Boolean algebra. Input signals are interpreted as logical states:

$$0 \rightarrow 0\,\text{V (LOW)}, \quad 1 \rightarrow 5\,\text{V (HIGH)}$$

Logical decisions controlling irrigation are implemented through conditional statements derived from threshold comparisons.

Memory Structure

The microcontroller contains three primary memory units:

- Flash Memory (32 KB): Stores program code
- SRAM (2 KB): Temporary data storage during execution
- EEPROM (1 KB): Non volatile memory for persistent parameters

Program execution follows the fetch decode execute cycle, fundamental to microprocessor operation.

5.2 Analog Pins

Arduino UNO includes six analog input pins (A0–A5) used for reading continuous voltage signals from sensors such as the soil moisture sensor.

ADC Conversion Principle

Analog signals are sampled and converted into digital values using the internal 10 bit ADC:

$$\text{Resolution} = \frac{V_{ref}}{2^{10}} = \frac{5}{1024} \approx 4.88 \text{ mV per step}$$

This defines the smallest detectable voltage change.

Signal Sampling

Sampling converts continuous time signals into discrete measurements:

$$x[n] = x(nT_s)$$

where:

- T_s = sampling interval
- $x[n]$ = sampled signal value

Proper sampling ensures accurate representation of changing soil moisture conditions.

5.3 Power System

Voltage Regulation

The Arduino UNO contains an onboard voltage regulator that maintains a stable 5V supply required for microcontroller operation. Stable voltage is essential because ADC accuracy depends on a constant reference voltage.

Power relation follows:

$$P = VI$$

where P is electrical power consumed by the circuit.

USB Power Interface

The board can be powered through a USB connection, which provides:

- Electrical power (5V supply)
- Serial communication for program uploading
- Real time data monitoring via Serial Monitor

The USB interface acts as both a programming channel and communication bridge between the computer and embedded system.

6 Components Used

The Smart Irrigation System is built on the basic components of embedded systems and integrated into the simulation environment provided by TinkerCAD. The components are selected for a particular role in the overall control process.

- **Arduino UNO R3**

  Audrino Uno R3 is used as a processing unit in the Smart Irrigation System. This unit receives analog signals from the soil moisture sensor and processes the information to make a decision based on threshold values. It then controls the irrigation actuator. This device runs on the ATmega328P microcontroller clocked at 16 MHz.

- **Soil Moisture Sensor**

  The soil moisture sensor is used to sense moisture levels in soil. This sensor measures water content in soil based on conductivity between two probes. More moisture in soil results in greater conductivity, resulting in lower resistance and a high analog voltage. This analog signal is then input into the Arduino analog input pin for processing.

- **DC Water Pump (Simulated)**

  The DC water pump represents the irrigation actuator in the Smart Irrigation System. This device represents the physical process of providing water to the soil when moisture levels are below a specified threshold in the Smart Irrigation System simulation environment provided by TinkerCAD.

- **Transistor or Relay**

  Since the Arduino cannot directly control high current devices, a switching device is necessary.

  - A **transistor** acts as an electronic switch controlled by a digital output pin.
  - A **relay** provides electrical isolation and enables control of higher power loads.

  The switching device increases current according to transistor gain theory:

  $$I_C = \beta I_B$$

  where I_C is collector current, I_B is base current, and β is current gain.

- **Breadboard**

  The breadboard is a device used for prototyping purposes. It allows for circuit construction without soldering. The conductive strips on the breadboard are used for creating nodes for interconnecting components electrically.

- **Connecting Wires**

  The jumper wires are used for creating paths for electric current between components. This facilitates signal transmission, grounding, and power supply within a circuit.

- **USB Interface**

  The USB port provides power for the Arduino and facilitates serial communication for uploading code and for real-time monitoring purposes.

- **TinkerCAD Simulation Environment**

  TinkerCAD is a virtual electronics lab used for designing, simulating, and testing embedded systems before physical implementation. It allows for verification of control logic and circuit operation under simulated environmental conditions.

7 Circuit Connections

Electrical connections are made between the sensing, controlling, and actuation parts of the Smart Irrigation System. The Arduino Uno measures the soil moisture level and manages the relay that controls the DC water pump based on soil moisture.

7.1 Sensor Connections

Soil Moisture Sensor

Soil moisture sensors measure soil conductivity in an analog form of voltage. The following connections are used:

- Sensor VCC → Arduino 5V
- Sensor GND → Arduino GND
- Sensor SIG → Arduino Analog Pin A0

The analog voltage varies with respect to soil resistance:

$$V_{out} \propto \frac{1}{R_{soil}}$$

where lower soil resistance represents higher moisture levels.

7.2 Relay Connections (SPDT)

A Single Pole Double Throw (SPDT) relay separates the high current load from the Arduino to avoid voltage disturbances.

- Relay Terminal 5 → Arduino GND
- Relay Terminal 6 → DC Motor Terminal 2
- Relay Terminal 8 → Arduino Digital Pin 7
- Relay Terminal 12 → External Power Supply Positive

The relay acts as an electromechanical switch. If the Arduino supplies the HIGH voltage at the input terminal, the relay coil allows the current flow creating a magnetic field. The magnetic field closes the switching contacts.

Relation of magnetic force:

$$F \propto I^2$$

where I is coil current.

7.3 DC Motor (Water Pump)

The DC motor simulates the irrigation pump responsible for delivering water to soil.

- DC Motor Terminal 1 → Power Supply Negative
- DC Motor Terminal 2 → Relay Terminal 6

The motor transforms electrical energy into mechanical energy:

$$P = VI$$

where P is electrical power supplied to the pump.

7.4 Power Supply Connections

External power supplies adequate current needed for the motor operations.

- Power Supply Negative → DC Motor Terminal 1
- Power Supply Positive → Relay Terminal 12

Power separation between actuator and Arduino guarantees voltage stability for the microcontroller.

7.5 System Signal Flow

The pathway of the operational signal transmission is described by the diagram below:

Soil Moisture → Sensor → Arduino → Relay → Pump → Soil

This design constitutes a closed-loop automatic irrigation control system.

8 Arduino IDE / TinkerCAD Procedure

For programming and testing the Smart Irrigation System, either the Arduino IDE or TinkerCAD simulation can be used. Below is an outline of the entire process, from preparing the circuitry to real-time monitoring of the results.

8.1 Circuit Assembly

- Construct the circuit according to the schematic diagram.
- Ensure proper connections between the soil moisture sensor, Arduino UNO, relay module, and DC motor.
- Verify power and ground continuity before powering the system.

Proper wiring will ensure accuracy in receiving signals from the sensors.

8.2 Connecting the Arduino

- Connect the Arduino UNO to the computer using a USB cable.
- The USB connection provides both electrical power and serial communication.

The USB connects the computer with the microcontroller using serial communication based on the UART protocol.

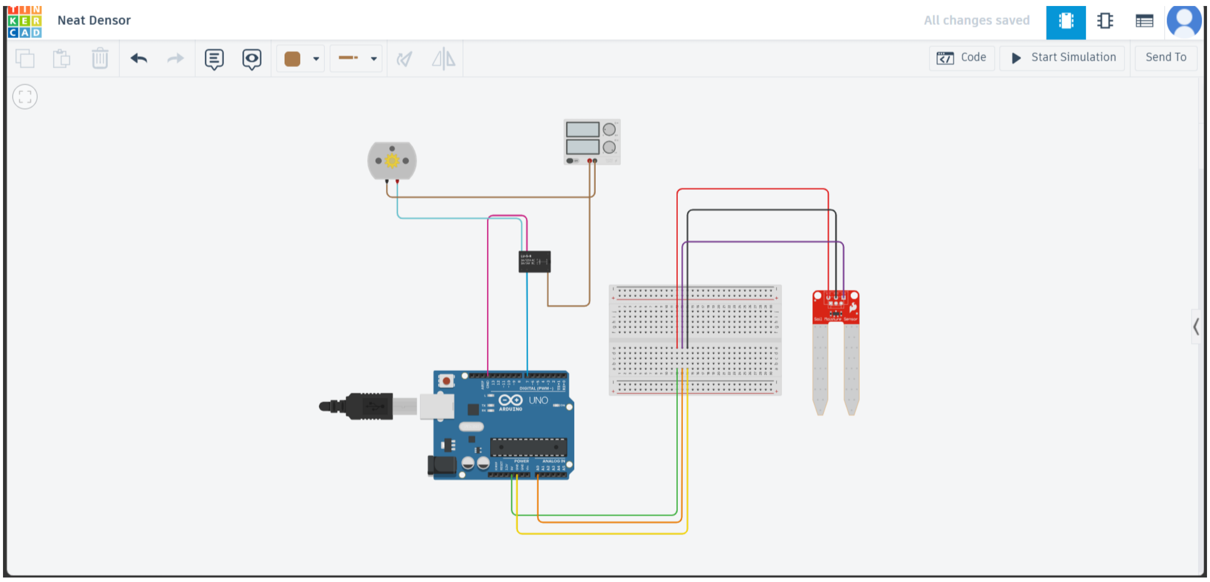

Figure 1: Complete Smart Irrigation Circuit in TinkerCAD

8.3 Opening Arduino IDE / TinkerCAD Code Editor

- Launch the Arduino IDE software or open the TinkerCAD simulation environment.
- Open the Code Editor section.
- Create a new sketch or load the irrigation system program.

In the code editor, programs can be compiled and run using embedded C/C++ code.

8.4 Board Selection

- Navigate to **Tools** $\rightarrow$ **Board**.
- Select **Arduino UNO**.

Choosing the proper board will ensure the software will work with the ATmega328P microcontroller.

8.5 COM Port Selection

- Go to **Tools** $\rightarrow$ **Port**.
- Select the available COM port assigned to the Arduino device.

The communication port is the link between the computer and the microcontroller.

8.6 Code Verification

- Click the **Verify** button.
- The compiler checks syntax errors and converts source code into machine instructions.

The compiler translates the instructions into machine language (hexadecimal).

8.7 Uploading the Program

- Click the **Upload** button.
- The compiled program is transferred to the Arduino flash memory through serial communication.

Once uploaded, the microcontroller will operate autonomously, controlling the irrigation process.

8.8 Serial Monitor Observation

- Open **Tools** $\rightarrow$ **Serial Monitor**.
- Set baud rate to match program configuration (typically 9600 baud).
- Observe real time soil moisture readings and system status messages.

The Serial Monitor will show sensor information sent using serial communication:

$$\text{Data Rate} = 9600 \text{ bits per second}$$

permitting real-time monitoring of the environment and system performance.

9 Arduino Program Code

The code for Arduino takes soil moisture measurement from the analog sensor and automatically operates the water pump based on a certain threshold set beforehand.

Listing 1: Arduino Code for Automated Irrigation System

```

// Automatic Plant Watering System
// Soil Moisture Feedback Control

const int soilPin = A0;       // Sensor input
const int relayPin = 7;       // Relay control
int moistureValue = 0;

void setup() {

  pinMode(relayPin, OUTPUT);
  Serial.begin(9600);

}

void loop() {

  // Read sensor value
  moistureValue = analogRead(soilPin);

  Serial.print("Moisture Level: ");
  Serial.println(moistureValue);

  // Control Logic
  if (moistureValue > 600) {
    digitalWrite(relayPin, HIGH);    // Pump ON
    Serial.println("Soil Dry : Pump ON");
  }
  else {
    digitalWrite(relayPin, LOW);     // Pump OFF
    Serial.println("Soil Wet : Pump OFF");
  }

  delay(1000);

}
```

10 Arduino Program Code

The output from the Serial Monitor shows live data for soil moisture levels and irrigation status from the Arduino code.

Arduino Serial Monitor Output

```
Moisture Level:  742
Soil Dry :  Pump ON

Moisture Level:  710
Soil Dry :  Pump ON

Moisture Level:  520
Soil Wet :  Pump OFF

Moisture Level:  498
Soil Wet :  Pump OFF
```

11 Experimental Conditions

Two different soil scenarios were created to test the efficiency of the Smart Irrigation System in the TinkerCAD simulation platform. The soil cases simulate extreme moisture contents to study the sensor's reaction and electrical and control system behavior.

11.1 Case 1: Dry Soil Simulation

With dry soil, the presence of water in the soil particles is very little. Electrical conductivity in soils mainly depends on the ions moving in the water. Thus, less moisture in soils causes more soil resistance.

Physical Characteristics:

- High electrical resistance
- Low ionic conductivity
- Reduced charge carrier mobility

Electrical Behavior:
According to Ohm's Law:

$$V = IR$$

where:

- V = applied voltage
- I = current through soil
- R = soil resistance

More resistance leads to lower current flow and greater voltage difference, which is measured by the analog input.

ADC Response:

The Arduino code converts the analog voltage into digital using an Analog-to-Digital Converter (ADC):

$$ADC = \frac{V_{in}}{V_{ref}} \times 1023$$

Dry soil produces:

- Higher sensor voltage
- Higher ADC value
- Moisture value exceeding threshold

System Action:

- Arduino interprets soil as dry
- Relay is activated
- Water pump turns ON

This corresponds to the irrigation switching stage in the feedback control system.

11.2 Case 2: Wet Soil Simulation

Wet soil has a higher moisture content than dry soil.

Physical Characteristics:

- Low electrical resistance
- High conductivity
- Increased ion mobility

Electrical Behavior:
Moisture reduces resistivity:

$$\rho = R\frac{A}{L}$$

where:

- ρ = soil resistivity
- A = effective contact area
- L = probe separation distance

Decreasing the value of resistivity causes the amount of current to increase and the voltage to decrease.

ADC Response:

- Lower sensor voltage
- Lower ADC value

- Moisture value below threshold

System Action:

- Arduino detects sufficient moisture
- Relay deactivated
- Water pump turns OFF

This is an example of using an automatic cutoff for irrigation to save water.

11.3 Control Interpretation

The above examples prove the theory of closed loop control:

$$\text{Soil State} \rightarrow \text{Sensor Measurement} \rightarrow \text{Decision Logic} \rightarrow \text{Actuator Response}$$

Experiment proves that electrical characteristics of the soil directly affect the decision on irrigation.

12 Observations

While performing the simulation for the Smart Irrigation System using TinkerCAD, there was a recorded variation in the analog sensor readings due to the continuous monitoring of soil moisture. The analysis of the system performance involved the study of electrical response, mathematical computation of threshold, and performance of actuator switching.

12.1 Sensor Output Behaviour

There were analog voltage readings that depended on the soil conductivity from the soil moisture sensor.

- The analog readings changed continuously depending on the soil moisture level.
- Dry soil conditions produced higher ADC values.
- Wet soil conditions produced lower ADC values.
- Sensor response showed smooth transitions rather than abrupt jumps, indicating stable signal sampling.

From a mathematical perspective,
It was noted that:

$$ADC = \frac{V_{in}}{V_{ref}} \times 1023$$

where changes in soil resistance altered the sensed voltage V_{in}.

12.2 Threshold Based Irrigation Response

A predefined moisture threshold governed irrigation control:

$$\text{If } M > M_T \Rightarrow \text{Pump ON}$$
$$\text{If } M \leq M_T \Rightarrow \text{Pump OFF}$$

where:

- M = measured moisture value
- M_T = threshold value

Observations confirmed:

- Threshold comparison reliably triggered decision making.
- Logical conditions executed without delay.
- Control action remained consistent across repeated simulations.

12.3 Automated Switching Performance

There was stability in the actuator switching process for the relay-controlled pump.

- Pump activated under dry soil simulation.
- Pump deactivated under wet soil simulation.
- No oscillatory switching observed near threshold values.
- System maintained steady feedback operation.

This shows the success of applying an elementary closed-loop embedded control system, wherein environmental input controlled the output actuator.

13 Results Analysis (Mathematical)

Evaluation of the performance of the Smart Irrigation System has been done mathematically by analyzing the relationship between the electrical characteristics of the soil, the sensor response and the efficiency of the automation process.

13.1 Sensor Response Relation

The soil moisture sensor is a variable resistor that changes resistance according to the amount of moisture in the soil. With higher moisture, there would be higher ionic conduction lowering the resistance.

From Ohm's Law:

$$V = IR$$

where:

- V = voltage measured at sensor output
- I = current through soil medium
- R = effective soil resistance

The Arduino reads voltage through an ADC:

$$ADC = \frac{V_{out}}{V_{ref}} \times 1023$$

Thus,

$$ADC \propto R_{soil}$$

which implies:

$$\text{Dry Soil} \Rightarrow R_{soil} \uparrow \Rightarrow ADC \uparrow$$
$$\text{Wet Soil} \Rightarrow R_{soil} \downarrow \Rightarrow ADC \downarrow$$

There exists a clear mathematical relationship between the soil moisture and its corresponding value in terms of ADC readings.

13.2 Threshold Decision Model

The decision for water irrigation is done using the following comparative control:

$$U(t) = \begin{cases} 1, & M(t) > M_T \\ 0, & M(t) \leq M_T \end{cases}$$

where:

- $U(t)$ = control signal (pump state)
- $M(t)$ = measured moisture value
- M_T = predefined moisture threshold

The decision making is a discrete control system that resembles a binary control feedback.

13.3 Efficiency Concept

Efficiency of water usage can be written as

$$\eta = \frac{W_{required}}{W_{supplied}} \times 100$$

where:

- η = irrigation efficiency
- $W_{required}$ = water actually needed by soil
- $W_{supplied}$ = total water delivered

In manual irrigation:

$$W_{supplied} > W_{required}$$

leading to wastage.
In automated irrigation:

$$W_{supplied} \approx W_{required}$$

thereby increasing efficiency.

13.4 Impact of Automation

Automation leads to continuous sensing and control as well:

$$\text{Sensor} \rightarrow \text{Measurement} \rightarrow \text{Decision} \rightarrow \text{Actuation}$$

Mathematically, this forms a closed loop feedback system minimizing error:

$$e(t) = M_T - M(t)$$

The controller acts to reduce $e(t)$ toward zero, maintaining optimal soil moisture conditions.

Hence, automation improves irrigation efficiency by reducing over watering, conserving energy, and maintaining stable environmental conditions for plant growth.

14 Discussion

Performance of the Smart Irrigation System highlights the application of physics principles in sensors and control algorithms. Multiple physical and computing phenomena affect accuracy of measurement and effectiveness of the irrigation process.

14.1 Sensor Sensitivity and Soil Composition

Sensor output depends heavily on the characteristics of soil. Soil is a composite material made up of solid, liquid, ionized, and gaseous phases. The soil electrical conductivity is dependent on:

- Soil texture (sand, clay, loam)
- Ionic concentration in water
- Compaction and density
- Temperature variations

Effective soil resistance can be calculated as:

$$R = \rho \frac{L}{A}$$

where:

- ρ = soil resistivity
- L = distance between probes
- A = effective conduction area

Soils with high concentration of clay have better water and ion retention capabilities than sandy soils. This leads to a situation where the same amount of moisture yields different readings for different soils.

14.2 ADC Quantization and Measurement Precision

Arduino UNO uses a 10-bit ADC, which translates voltages to discrete digital numbers ranging from 0 to 1023. The voltage quantization step is:

$$\Delta V = \frac{V_{ref}}{2^{10}} = \frac{5}{1024} \approx 4.88 \text{ mV}$$

This means any small changes in voltage levels below the quantization step are not taken into account. Thus:

- Measurement precision is limited.
- Minor moisture fluctuations may remain unresolved.
- Digital output represents approximated rather than continuous values.

14.3 Threshold Calibration Requirement

Irrigation will depend on an initial threshold value of moisture M_T. Incorrect calibration would make the system run inefficiently:

- Threshold too high → excessive irrigation.
- Threshold too low → insufficient watering.
- Environmental drift may shift optimal threshold over time.

Mathematically, optimal performance occurs when:

$$M_T \approx M_{optimal}$$

where $M_{optimal}$ represents moisture required for plant health.

For this reason, the calibration process will have to be carried out experimentally using various crops and soils.

14.4 Simulation Versus Real World Variation

The irrigation system was created and simulated in TinkerCAD, which uses an assumption of ideal behavior of electricity. When used in the real world, other complications arise:

- Sensor corrosion and aging
- Electrical noise and interference
- Temperature dependent conductivity changes
- Non uniform soil moisture distribution

Random variations can lead to variations in data collected:

$$M(t) = M_{ideal}(t) + n(t)$$

where $n(t)$ represents environmental noise.

From this point, it is clear that while simulations prove that the system logic works and the design is good, actual deployment will need to incorporate filtering methods, calibration, and hardware protection.

To conclude, it is evident that the irrigation system works well as a basic cyber physical control framework.

15 Conclusion

In the design of this system, all the concepts in physics, mathematics, and programming in embedded devices have been used for efficient control.

15.1 Physics Based Sensing

This project proves the practical use of the principles of electrical conduction in soil. Change in the soil's resistance due to change in moisture content directly affected the voltage and current behavior as per Ohm's Law:

$$V = IR$$

This physical concept made it possible for soil moisture sensors to detect the changes in the environment by translating them to an electrical form. Thus, the principle of control in such situations lies in electromagnetic and material properties.

15.2 Mathematical Control Logic Validation

Analog voltage levels have been converted to a digital format with the help of 10-bit ADCs and further analyzed using threshold based decision models:

$$U(t) = \begin{cases} 1, & M(t) > M_T \\ 0, & M(t) \leq M_T \end{cases}$$

It proves that the principle of mathematical comparison has been used successfully for controlling actuators. Thus, the relationship from resistance to voltage to digital conversion and then to the binary code shows the whole process of measurement to control.

15.3 Embedded Automation in Agriculture

A closed loop feedback model was developed as well:

$$\text{Sensing} \rightarrow \text{Processing} \rightarrow \text{Decision} \rightarrow \text{Actuation}$$

Therefore, although simulations verify the logic and functioning of the system, implementation on the ground requires the use of filters, calibration, and physical protection measures.

In general, the discussion brings to light the fact that the irrigation system works well as a basic cyber physical control system.

16 Future Scope

The proposed Smart Irrigation System is a base embedded control system that can later on be enhanced to become a very intelligent farming system. There are various aspects under which one could enhance this system to achieve improved results.

16.1 IoT Based Cloud Monitoring

Firstly, the irrigation system can be integrated into an IoT system by using wireless communication techniques, such as the WiFi or GSM module. This way, the data collected by the sensors can be transmitted to the server in the cloud via such modules.

From the point of view of mathematics, since there is constant monitoring taking place, then the environment can be modeled according to time:

$$M = M(t)$$

Here, the soil moisture level is considered a dynamic parameter that can change over time.

16.2 Machine Learning Based Irrigation Prediction

Moreover, machine learning algorithms could be used for making predictions about the environment through predictive analysis. For example, parameters such as humidity, temperature, and previous history of soil moisture could be used to make some predictions.

The equation for the simplified prediction model can be stated as follows:

$$I = f(M, T, H, t)$$

where:

- I = irrigation requirement

- M = soil moisture
- T = temperature
- H = humidity
- t = time or seasonal parameter

This would ensure efficient utilization of water resources and optimize the crop output.

16.3 Multi Sensor Distributed Systems

In terms of future innovations, the use of several sensors in the field can address the spatial variability in soil properties. Spatially distributed sensors can facilitate the mapping of moisture content in each area.

Spatial variation in moisture content can be described by:

$$M = M(x, y)$$

Here, moisture content is dependent upon geographical location on earth.

16.4 Solar Powered Autonomous Farming Units

This facilitates zoning of the irrigation process rather than uniform irrigation throughout the area.

Integration of such a system with energy generation from solar photovoltaic cells enables the system to operate independently of any external sources.

The energy balance equation can be described as:

$$E_{generated} \geq E_{consumed}$$

This ensures continuous operation of the system in an autonomous fashion.

This would transform a basic irrigation system into a smart agriculture system.

References

[1] Microchip Technology Inc., *ATmega328P Microcontroller Datasheet*, Microchip Corporation, USA. Available: `https://www.microchip.com`

[2] Arduino Official Documentation, *Arduino UNO R3 Technical Specifications and User Guide*, Arduino.cc. Available: `https://www.arduino.cc`

[3] Corwin, D. L., and Lesch, S. M., "Applications of Soil Electrical Conductivity in Precision Agriculture," *Agronomy Journal*, vol. 97, no. 2, pp. 455–471.

[4] Zhang, N., Wang, M., and Wang, N., "Precision Agriculture A Worldwide Overview," *Computers and Electronics in Agriculture*, Elsevier.

[5] Autodesk Inc., *TinkerCAD Circuits Documentation and Simulation Guide*, Available: `https://www.tinkercad.com`

[6] Ogata, K., *Modern Control Engineering*, 5th Edition, Prentice Hall.

[7] Barr, M., and Massa, A., *Programming Embedded Systems*, O'Reilly Media.