Smart Drip Irrigation System – Construction, Working Principle, and Applications

Munim Shafi, Mehdi Ali Kirmani, Hamna Malik

Mechatronics System Design, Department of Mechatronics Engineering, NUST

Abstract:

This research paper presents the design and development of a prototype drip irrigation system, including a moisture sensor, a 12V water pump, and an Arduino UNO microcontroller. The system is designed to provide an automated and efficient watering solution, ensuring plants are given precise amounts of water based on their moisture needs. The article describes the hardware components, the system's operation principle, and the stages of its construction. With this comprehensive analysis, the research paper provides a valuable resource for those wishing to gain a deeper understanding of entry-level drip irrigation systems and provides a simple technique for those with minimal technical knowledge to understand and experiment with microcontrollers and sensors to make their plants healthier. You can use this simple system in your kitchen gardens or lawns to take care of your plant's water needs. Also, this research paper will provide both the Arduino IDE code and the C code. We have used multiple timers and the OCR function for programming our ATMEGA 16. Integration of hardware and software is done in a very simple way. In this research paper, we have explained the electronics controlling the speed of a DC motor water pump with the help of a transistor (BJT/MOSFET).

Article Info:

Keywords:

Atmega 16, Arduino UNO, Smart drip irrigation, 12V water pump, BJT, moisture sensor, plants, software, hardware, integration, and embedded system, prototype.

I. Introduction

Water scarcity and inefficient irrigation methods pose a serious challenge to modern agriculture. Drip irrigation systems are gaining popularity for their ability to conserve water and deliver it directly to plant roots. This research focuses on the design and development of a drip irrigation system.

In this project mainly we will make a prototype of controlled flow smart drip irrigation, using Atmega 328p and a soil moisture sensor. The controller will take input from the soil moisture sensor and based on that reading and the crop that is being irrigated, the pump will change the flow rate of water.

The controller will get input from the user for the crop being irrigated(can be replaced by a camera for commercial purposes) and then the Arduino will irrigate the crop according to the algorithm we have fed into it (for commercial purposes we can train models of different crops and deploy algorithm in the controller, but we are making a prototype only so we will not go in so much complexity).

II. AIMS AND OBJECTIVES

This project aims to make a beginner-level prototype for a smart drip irrigation system. As we know, as a third-world country water is a major problem so to save water usage in plantations this system is designed. As we are an agricultural country so technological improvements are required in this sector to make our economy stronger.

The listed objectives are:

- To benefit from the right expertise in embedded systems and their implementation.
- Design a prototype that is time-efficient, clean to understand, and greater economical.
- Hardware and software integration.
- Sensor calibration and speed control of water pump.
- Addressing the issue of excessive watering of plants.
- Step towards making our agricultural industry use autonomous products for better yield.

III. LITERATURE OVERVIEW

Sharmin Akter and PinkiRani Mahanta present a system that acts in a manner like an intelligent switching system that monitors the amount of soil moisture and irrigates the plant as necessary. The automatic ON/OFF motor will depend on the soil's level of dryness. Also, they claim that it saves electricity and time.[1]

In this paper the authors have designed a system to effectively use water for irrigation, an automated irrigation system has been created using sensors and Arduino. The system contains a water level sensor in a water container from which water will be piped to the plants for irrigation and a soil moisture sensor implanted into the soil of the plants. To control the amount of water in the soil and to detect the water level in tanks, an algorithm has been developed with threshold values for soil moisture sensors.[2]

Ipin Prasojo and Andino Maseleno basically highlight the problems of Indonesian farmers, who have both rainy and dry seasons. The farmer's reliance on the weather affects production, which hinders the program's goal of achieving food self-sufficiency. The issue needs to be solved using agricultural equipment based on information and autonomous technologies. The goal of the research was to create a programmed microcontroller chip that would autonomously manage irrigation based on soil moisture sensed by a household soil moisture sensor.[3]

For wireless, sensor-based irrigation control, they have designed and built a prototype cloud-connected system and tested it on basil growing in a soilless substrate under greenhouse conditions. they have used dielectric moisture/salinity sensors. Their study follow-up two experiments evaluated the effects of a gradual decline in substrate water availability, corresponding to moisture levels in the water, on gas exchange parameters and leaf water status of basil plants; ii) the short-term recovery response of plants when re-watered after substrate water content has decreased to different levels; and iii) the effects of different irrigation set-points and leaching rates on the basil crop performance over time.[4]

IV. METHODOLOGY

In our prototype, there are three important parts of the system. In the first part, we get an input signal from the moisture sensor that helps the system to apply conditions and turn on/off the water pump accordingly.

- **COMPONENTS**
- *Moisture sensor:*

  The soil moisture sensor is an electronic device for measuring and monitoring soil moisture. It is an essential part of irrigation systems and agricultural practices as it provides valuable data for efficient water management and plant health. Soil moisture sensors are typically connected to a controller or microcontroller, such as an Arduino, which processes the readings from the sensor. The control unit can be performed and programmed to turn sprinkler systems on or off based on sensors reading that water is added when needed.

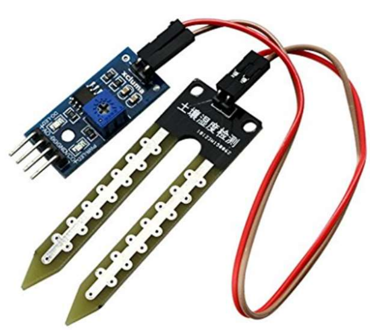

Figure 1. Soil Moisture Sensor

- *MOSFET (IRF540):*

  The IRF540 is a popular power MOSFET (Semiconductor Metal Oxide Field Effect Transistor) designed for high-power switching applications, offering high voltage capability, low on-resistance, and fast switching speed.

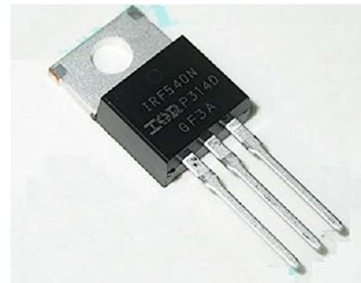

Figure 2. MOSFET IRF50

- *Arduino UNO:*

  The Arduino UNO is based on the ATmega328P microcontroller which operates at a clock frequency of 16 MHz. The microcontroller is responsible for running custom programs and controlling various board functions. It provides a range of digital and analog input/output pins. It can be programmed using the Arduino programming language, which is a simplified version of C++ and supports a variety of communication interfaces including USB for programming and serial communication, Universal Asynchronous Receiver -Transmitter (UART) for serial communication, and I2C and SPI -Protocols to communicate with other devices.

Figure 3. Arduino UNO

- *12V pump:*

  A 12V water pump is a device designed to move water from one place to another using a 12V DC power supply. It is widely used in a variety of applications such as irrigation systems, aquariums, water coolers, and small water circulation projects.

Figure 4. water pump

- *Voltage Source:*

  The voltage source is required to operate a 12V water pump. We cannot give power to the pump from Arduino UNO because of the maximum 5V it can give. You can use any battery for powering your pump.

Figure 5. Power supply

- **CONSTRUCTION**

In our project, the Hardware components required for the prototype include an Arduino UNO microcontroller, moisture sensor module, 12V water pump, IRF50, power supply, and connection cables. The moisture sensor module is connected to an Arduino board that processes the sensor readings and controls the water pump accordingly. The water pump is connected to a water source and a network of drip irrigation pipes that provide water to the plants.

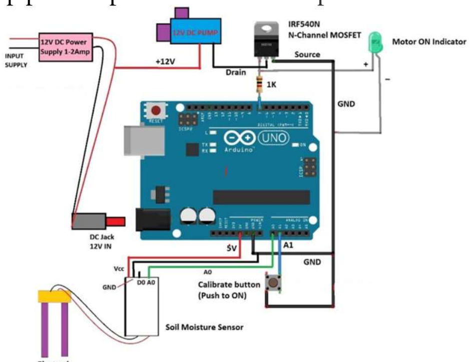

Figure 6. System Diagram

To ensure accurate watering, the moisture sensor must be calibrated to the specific needs of the plants being watered. This involves determining the ideal moisture threshold and adjusting the Arduino code accordingly. Calibration ensures the system is only when it is needed, avoiding over- or under-watering.

- **FLOWCHART**

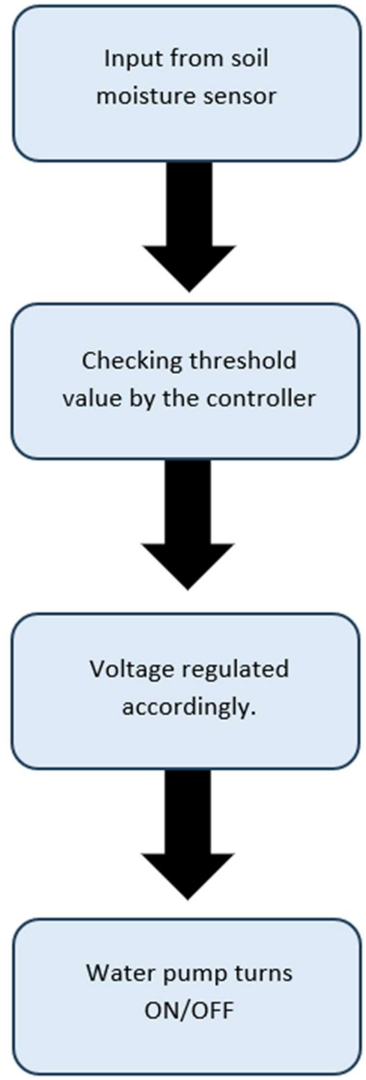

Figure 7. Flowchart

- **CODE**

  Arduino Code:

```
#define moistureSensorPin A0  // Analog pin for the moisture sensor
#define pwmPin 9              // Digital pin for PWM output

void setup() {
  pinMode(pwmPin, OUTPUT);   // Set the PWM pin as an output
  Serial.begin(9600);        // Initialize serial communication
}

void loop() {
  int sensorValue = analogRead(moistureSensorPin);  // Read the sensor value
  int moisturePercentage = map(sensorValue, 0, 1023, 100, 0);  // Convert to percentage

  Serial.print("Moisture Percentage: ");
  Serial.print(moisturePercentage);
  Serial.println("%");

  if (moisturePercentage < 20) {
    analogWrite(pwmPin, 255);   // Set PWM signal to 255
  } else if (moisturePercentage >= 20 && moisturePercentage <= 70) {
    analogWrite(pwmPin, 150);   // Set PWM signal to 150
  } else {
    analogWrite(pwmPin, 10);    // Set PWM signal to 10
  }

  delay(1000);  // Wait for 1 second before taking the next reading

}
```

Figure 8. Arduino IDE code

C code:

```c
#include <avr/io.h>
#define moistureSensorPin A0  // Analog pin for the moisture sensor
#define pwmPin 9              // Digital pin for PWM output

void setup() {
  OutputPin(pwmPin);   // Set the PWM pin as an output
  Serial.begin(9600);        // Initialize serial communication
}

void loop() {
  int sensorValue = analogVal(moistureSensorPin);  // Read the sensor value
  int moisturePercentage = mapValue(sensorValue, 0, 1023, 100, 0);  // Convert to percentage

  Serial.print("Moisture Percentage: ");
  Serial.print(moisturePercentage);
  Serial.println("%");

  if (moisturePercentage < 20) {
    PWM(pwmPin, 255);   // Set PWM signal to 255
  } else if (moisturePercentage >= 20 && moisturePercentage <= 70) {
    PWM(pwmPin, 150);   // Set PWM signal to 150
  } else {
    PWM(pwmPin, 10);    // Set PWM signal to 10
  }

  delay(1000);  // Wait for 1 second before taking the next reading
}
void PWM(uint8_t pin, int value) {
  // Check if the pin supports PWM
  if (pin == 3 || pin == 5 || pin == 6 || pin == 9 || pin == 10 || pin == 11) {
    // Set the corresponding Timer/Counter for the pin
    switch (pin) {
      case 3:
        TCCR2A |= _BV(COM2B1) | _BV(WGM20) | _BV(WGM21);
        TCCR2B |= _BV(CS21);
        OCR2B = value;
        break;
      case 5:
        TCCR0A |= _BV(COM0B1) | _BV(WGM00) | _BV(WGM01);
        TCCR0B |= _BV(CS01);
        OCR0B = value;
        break;
      case 6:
        TCCR0A |= _BV(COM0A1) | _BV(WGM00) | _BV(WGM01);
        TCCR0B |= _BV(CS01);
        OCR0A = value;
        break;
      case 9:
        TCCR1A |= _BV(COM1A1) | _BV(WGM10);
        TCCR1B |= _BV(CS11) | _BV(WGM12);
        OCR1A = value;
        break;
      case 10:
        TCCR1A |= _BV(COM1B1) | _BV(WGM10);
        TCCR1B |= _BV(CS11) | _BV(WGM12);
        OCR1B = value;
        break;
      case 11:
        TCCR2A |= _BV(COM2A1) | _BV(WGM20) | _BV(WGM21);
        TCCR2B |= _BV(CS21);
        OCR2A = value;
    break;
    }
  } else if (pin == 13) {
    // Onboard LED (special case)
    if (value > 127) {
      PORTB |= _BV(PORTB5);
    } else {
      PORTB &= ~_BV(PORTB5);
    }
  }
}
int analogVal(uint8_t pin) {
  // Set the reference voltage to AVCC
  ADMUX = (1 << REFS0);

  // Set the ADC channel to read from the specified pin
  ADMUX |= pin & 0x07;

  // Enable ADC and start the conversion
  ADCSRA = (1 << ADEN) | (1 << ADSC) | (1 << ADPS2) | (1 << ADPS1) | (1 << ADPS0);

  // Wait for the conversion to complete
  while (ADCSRA & (1 << ADSC));

  // Read and return the converted value
  return ADC;
}
```

Figure 9. C code

- **WORKING PRINCIPLE**

The detailed working principle of a smart drip irrigation system involves the integration of sensors, control systems, and actuators to monitor soil moisture levels and efficiently supply water to plants. uses soil moisture sensors placed in the root area of the plants. These sensors measure soil moisture and provide real-time data on soil moisture content.

The collected soil moisture data is processed and analyzed by a control system such as a microcontroller or a computer program. The analysis compares current soil moisture levels to set thresholds or moisture targets for optimal plant growth.

Based on the data analysis, the control system determines whether irrigation is required or whether the existing moisture level is sufficient. If the soil moisture drops below the desired threshold, the system starts the irrigation process.

The control system activates solenoid valves to open or close water flow to specific drip lines or areas. By controlling valves, the system can direct water to areas of low moisture preventing overflow in areas with adequate moisture up to.

Intelligent drip irrigation systems optimize water consumption by integrating sensors, control systems, and actuators.

V. APPLICATIONS

- In regions with limited water resources or water scarcity, intelligent drip irrigation systems play a key role in saving water. By supplying water directly to plants and monitoring soil moisture, these systems minimize water loss and improve overall water use efficiency.
- Space is often at a premium in urban environments and vertical farm configurations. Smart drip irrigation systems offer a space-saving and water-efficient solution for growing plants in limited spaces such as rooftops or vertical gardens.
- Intelligent drip irrigation systems are widely used in agriculture to ensure efficient and precise watering of crops. By monitoring soil moisture levels and delivering water directly to plant roots, these systems help optimize water use, improve crop yields, and reduce water and fertilizer waste.

VI. RESULTS AND CONCLUSION

The project is based on ana prototype of a smart drip irrigation system. The project is concluded as follows:

1. The power supply provides the input voltage to the water pump.
2. Input from the moisture sensor will generate a pulse that will check the threshold conditions.
3. ATmega328P helps us to integrate hardware and software.
4. PWM of a water pump is controlled by MOSFET.
5. Water pump turns on according to plant's water needs,

ACKNOWLEDGMENT

We gratefully thank Sir Anjum Naeem (Lecturer – National University Of Sciences And Technology) who has supplied a tremendous deal of ethical help and assistance without which the assignment and the convention paper might now no longer be possible. We increase our gratitude to them for their treasured help, cooperation, and time.

We take this opportunity to thank our friends for their treasured tips and ideas. Lastly, we would love to increase our innermost gratitude to our circle of relatives and contributors for adjusting to our specific and unconventional timetable and supplying the important ethical help that contributed to the improvement of the assignment

REFERENCES

[1] Developing a Smart Irrigation System using Arduino, Sharmin Akter

[2] Automatic Irrigation System using Arduino, Kriti Taneeja

[3] Design of Automatic Watering System Based on Arduino, Ipin Prosojo

[4] Sensor-based irrigation management of soilless basil using a new smart irrigation system: Effects of a set point on plant physiological responses and crop performance, Franco Fabiono Montens.