SMART GARDEN

INTERNSHIP PROJECT REPORT

by

S RESHMA

Department of Electronics and Communication Engineering

SRI VENKATESWARA COLLEGE OF ENGINEERING

Post Bag No.1 Pennalur Village Chennai - Bengaluru Highways

Sriperumbudur (off Chennai) Tk. - 602 117 Tamil Nadu

2019-2023

1. ABSTRACT

"Water, water everywhere, not a drop of water to drink" will become true in the near future if we don't develop sustained development goals with respect to water. Water, is used for direct purposes like drinking, bathing, cooking, washing and many other indirect purposes. The bulk of world's water use is for AGRICULTURE. This project, works perfectly for watering home plants and gardens. Watering plants may be a hobby for many people and they love doing it. Then, there are times when we are on a vacation or busy and we totally forget to water them, leaving the plants to dry and later die. I'm sure no one wants their favourite plants to suffer, this project gives a perfect solution to automate the process. This project deals with an automatic plant irrigation system which senses the moisture content of the soil and water the plants accordingly. Moisture sensor is used to sense the moisture value. AtMega328 microcontroller is programmed such that when the moisture level decreases below a minimum threshold level the relay is turned on to drive the motor to water the plants till the moisture value equals or goes up the threshold. When the moisture level is above the threshold the relay is turned off. This project is tested and found to be working fine to minimize manpower and conserve time thereby making ours, a SMART GARDEN.

2. TABLE OF CONTENTS

- Abstract

CHAPTER 1: INTRODUCTION

- Introduction
- Existing Method
- Algorithm
- Hardware Architecture

CHAPTER 2: PROPOSED SYSTEM AND HARDWARE ARCHITECTURE

- Methodology
- Components List

CHAPTER 3: SYSTEM IMPLEMENTATION

- Implementation
- Algorithm
- Hardware Architecture

CHAPTER 4: CONCLUSION AND FUTURE SCOPE

- Conclusion
- Future Scope

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

"Water, water everywhere, not a drop of water to drink" will become true in the near future if we don't develop sustained development goals with respect to water. Water, is used for direct purposes like drinking, bathing, cooking, washing and indirect purposes. The bulk of world's water use is for AGRICULTURE. Therefore, resources should be managed as an integral part of nation's social and economic development. The habit should start from our home.

This project, which is works perfectly for watering home plants and gardens. Watering plants may be a hobby for many people and they love doing it. Then there are times when we are on a vacation or busy and we totally forget to water them, leaving the plants to dry and later die. I'm sure no one wants their favourite plants to suffer, this project gives a perfect solution to automate the process.

This project deals with an automatic plant irrigation system which senses the moisture content of the soil and water the plants accordingly. Moisture sensor is used to sense the moisture value. AtMega328 microcontroller is programmed such that when the moisture level decreases above a minimum threshold level the relay is turned on to drive the motor to water the plants till the moisture value equals or goes up the threshold. When the moisture level increases above the threshold the relay is turned off.

In our homes water (for plants) is separately stored in a drum. What if the drum gets empty and the plant's moisture level is below the threshold? In situations like these the relay is turned on but the motor doesn't drive water, since the drum is empty. The motor keeps running and never gets turned off, this may damage the motor.

This project gives solution to situations like these too. After the relay gets turned on, the motor starts driving(free-running). The sensor keeps note of the moisture value and checks if the value is same even after some amount of time. The same value indicates that the motor is not driving water and that the drum is empty. In this case, the relay is turned off automatically and buzzer is kept to indicate us. To resolve this problem, the Arduino can be programmed by reading the moisture level from the sensor by giving delay. But the delay must be set accurately and is different for plants that vary in size. Another way which works efficiently is by measuring the voltage of the motor when it drives water and when it doesn't. The power of the motor during the free running will be less than the power when it is actually driving water. Since the voltage which is directly proportional to the power (P=V.I), the calculated voltage will make the difference. This project is tested and found to be working fine to minimize manpower, conserve time and at the same time avoiding over and under watering.

1.2 EXPLANATION

The existing system uses an Arduino UNO, soil moisture sensor, relay and a motor. The function of this system is very simple. It checks the moisture level of the soil to ensure if the plant needs water or not. The soil moisture sensor works just like a variable resistor whose resistance varies according to the water content in the soil. This resistance is inversely proportional to the soil moisture:

- The more water in the soil means better conductivity and will result in a lower resistance.
- The less water in the soil means poor conductivity and will result in a higher resistance.

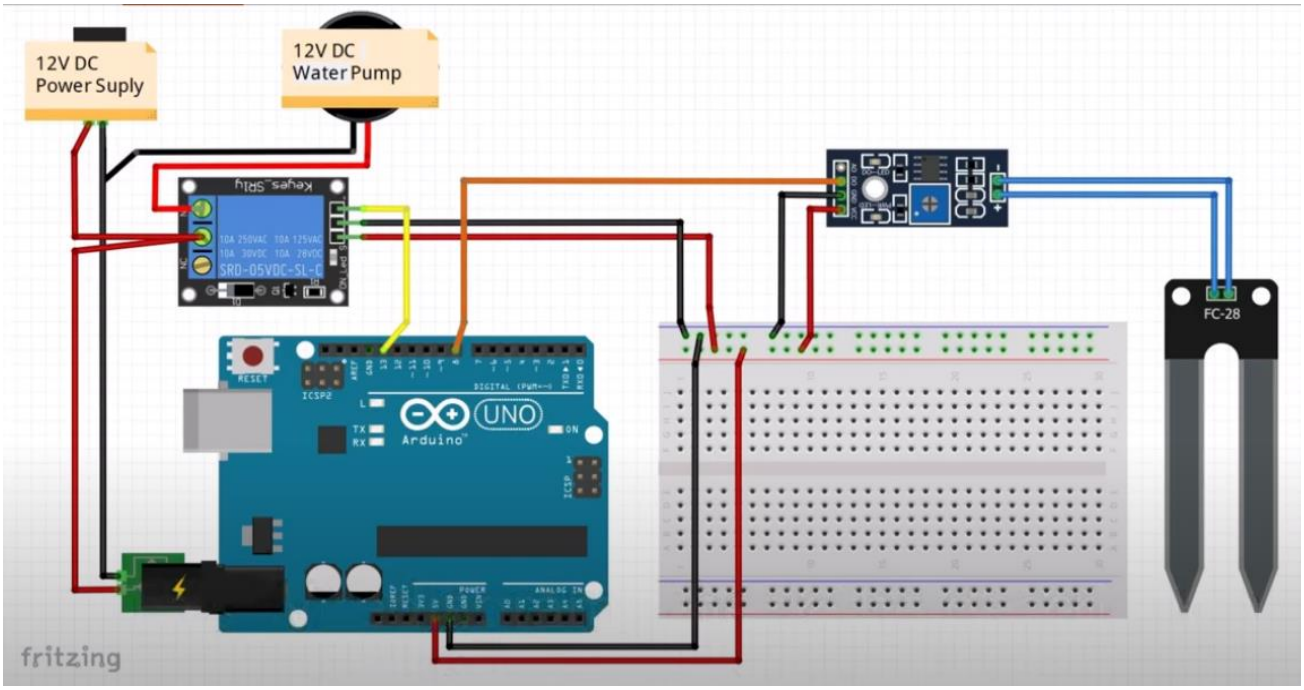
The sensor produces an output voltage according to the resistance, which by measuring we can determine the moisture level. Initially the moisture of the soil is alone measured for different conditions like dry, wet and optimum. Then the threshold is set. A relay module is used as a switch here to turn on the motor in the cases when the moisture level is too high. The soil sensor measures the resistance, if the resistance is measured above the threshold, it means that the plant needs water. Since the reading of the moisture is in terms of the resistance the water present will be inversely proportional.

1.3 ALGORITHM:

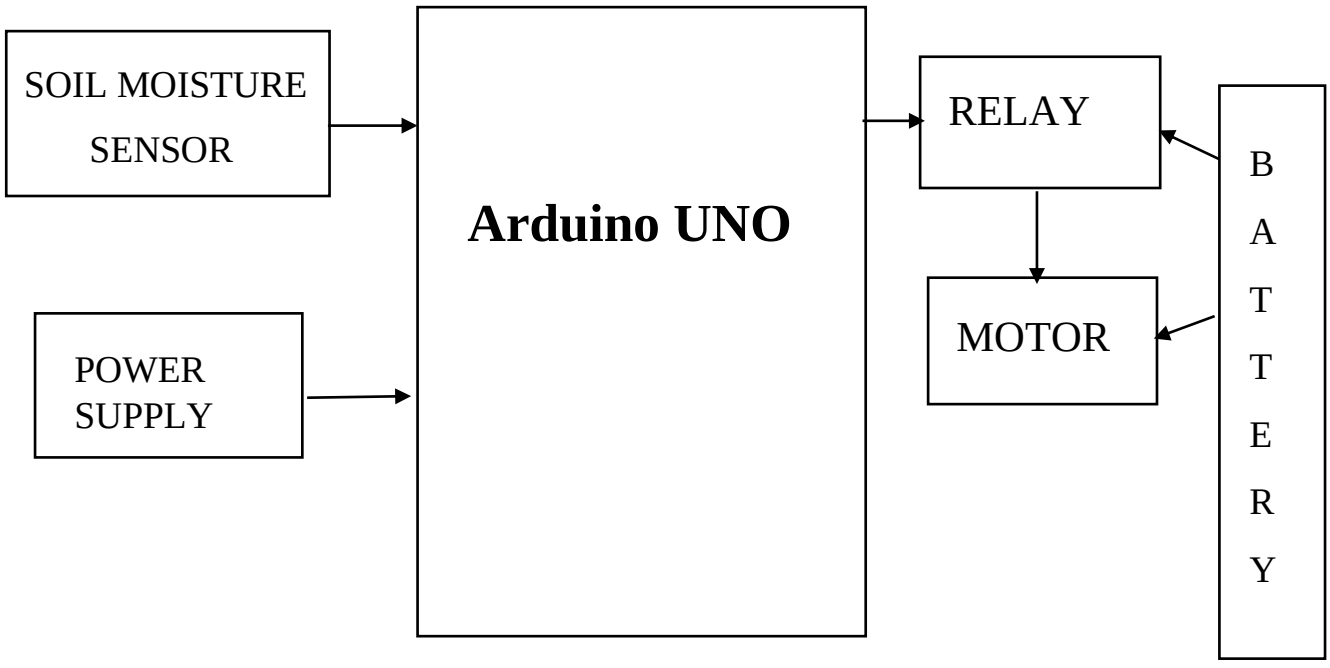
1. START
2. SENSOR MEASURES THE MOISURE LEVEL
3. IF VALUE > THRESHOLD
   3.1 RELAY- ON
   3.2 WATER TILL VALUE REACHES THE THRESHOLD
4. IF VALUE<300
   4.1 RELAY- OFF
5. STOP

1.4 ARCHITECTURE:

Pic credits: YouTube



1.5 BLOCK DIAGRAM:



CHAPTER 2: PROPOSED SYSTEM AND HARDWARE ARCHITECTURE

2.1 METHODOLOGY:

In our homes, water (for plants) is separately stored in a drum. We will have to refill every time the drum gets empty. If we use the existing system here, a question will arise. What if the drum gets empty and the plant's moisture level is below the threshold? In situations like these the relay is turned on but the motor doesn't drive water, since the drum is empty. The motor keeps running and never gets turned off, this may damage the motor. This project gives solution to situations like these too.

SOLUTION 1: In situation like these after the relay gets turned on, the motor starts driving(free-running). The sensor keeps note of the moisture value and checks if the value is same even after some amount of time. Arduino can be programmed by reading the moisture level from the sensor by giving delay. The same value indicates that the motor is not driving water and that the drum is empty. Here relay is turned off automatically and buzzer is kept to indicate us. The problem here is that, delay must be set accurately and is different for plants that vary in size.

Thinking more about how to make it efficient we came up with another solution.

SOLUTION 2: Another way which works efficiently is by measuring the voltage of the motor when it drives water and when it doesn't. The power of the motor during the free running will be less than the power when it is actually driving water. Since the voltage which is directly proportional to the power (P=V.I), the calculated voltage will make the difference. A resistor can be connected across the motor and the battery. An analog pin must be dedicated for resistor to read the voltage of the motor when it is actually driving water and when it doesn't.

2.2 COMPONENTS LIST:

Components used in this proposed system are:

i) Hardware:

- Arduino UNO/ AtMega 328
- Soil Moisture Sensor
- 9V DC Motor
- 5V Relay (1 channel)
- Buzzer
- Resistor (0.22ohms)
- 9V Adapter
- Vinyl Tube

ii) Software

- Arduino IDLE

CHAPTER 3: SYSTEM IMPLEMENTATION

3.1 IMPLEMENTATION:

Now let us see the implementation of the two solutions and which of could be practically implemented in real time solution.

3.1.1 IMPLEMENTATION OF SOLUTION 1

Implementation of solution one is straight forward, only the code needs to be modified by giving a delay. The situation can be identified to have happened it the value to moisture level is same even after the set delay. But this delay should set properly after experimenting. **The code for this implementation is as follows:**

```
//initializing the variable
int moisture_sensor=A1;
int relay=12;
int moisture_value=0;
int buzzer=8;

//code written under these are executed once.
void setup() {
  Serial.begin(9600);
  pinMode(moisture_sensor,INPUT);
  pinMode(relay ,OUTPUT);
  pinMode(buzzer ,OUTPUT);
}
```

```
//code written under these are executed in a loop
void loop() {
  moisture_value=analogRead(moisture_sensor);
  Serial.print("MOISTURE VALUE= ");
  Serial.print(moisture_value);
  Serial.println();
  delay(1000);
//first check
  if(moisture_value>=450)
  {
    digitalWrite(relay,LOW);
    delay(1000);
    //second check
    if(moisture_value>=450)
    {
      delay(1000);
      //final check
      if(moisture_value>=500)
      {
        digitalWrite(relay,HIGH);
        delay(1000);
        digitalWrite(buzzer,HIGH);
        delay(2000);
        digitalWrite(buzzer,LOW);
      }
      else
        digitalWrite(relay,LOW);
    }
```

```
  }
  if(moisture_value<=300)
  {
    digitalWrite(relay, HIGH);
    delay(100);
  }
}
```

After implementing the solution 1, many practical difficulties were faced. The delay time seemed very small for big plants. The order should be given as mentioned in the code. But when it was implemented, the value measured by the moisture sensor seemed to stay less than 400, giving us false results. So, we moved on to implementing solution 2.

3.1.2 IMPLEMENTATION OF SOLUTION 2:

The implementation of solution 2 uses a resistor across the motor and the 9V battery. In simple words the work done by the motor when it drives water is definitely more than work done during free running. The power also changes accordingly. The voltage drop across the resistor when the motor is driving water will usually be larger than the voltage drop across the resistor when the motor isn't driving water. Hence the voltage across the resistor can be calculated to find the voltage differences. First using multi-meter, it is found that:

- Voltage when it drives water: 200mV
- Free running voltage: 0.3mv

Then removing the soil moisture sensor, the Arduino was programmed to read the voltage value across. The scaled voltages were found to be:

- Voltage when it drives water: >0.08

- Free running voltage: <0.05

From these values the voltage threshold is decided. Finally, the Arduino is programmed using this algorithm. If the voltage level goes below the threshold, relay is switched off and a buzzer is set to on for 10 secs for indication.

Using these solutions, we can reduce the damage caused to the motor during free running. From these calculations resistor value is chosen as **0.22ohms.**

Resistor Value Calculation:

1) When motor drives water:
   Current found using multimeter is 0.12 Amp
   Voltage found using multimeter is 0.200 V

   R = V/I = 0.200/(1.2)
   = 0.166 ohms

2) When motor drives water:
   Current found using multimeter is 0.15 Amp
   Voltage found using multimeter is 0.033 V

   R = V/I = 0.33/(0.15)
   = 0.22 ohms

Hence 0.22ohms is selected as the resistance value.

Motor Threshold Calculation:

Using analog read function in Arduino idle. The voltage across the resistor is read. Ultimately that is the voltage of the motor.

1) When motor drives water/ when water in our drum is filled
   The voltage value is >0.08

2) When motor does not drive water/ when water is in our drum is empty
   The voltage value is <0.05

Hence the voltage threshold of motor is set as **0.08.** That means the motor is driving water only when the motor voltage is above 0.08V. Below this value the motor is not driving water and that the relay should be turned OFF.

The code implementation is as follows:

```
//initializing the variables
int moisture_sensor = A0;
int motor = A1;
int relay = 13;
int buzzer = 8;
int moisture_value = 0;
int motor_value;
float motor_voltage = 0;
int d1 = 100;


//code written here are executed once
void setup() {
```

```
  Serial.begin(9600);
  pinMode(moisture_sensor, INPUT);
  pinMode(motor, INPUT);
  pinMode(relay , OUTPUT);
  pinMode(buzzer , OUTPUT);
}

//code executed here are executed in loops
void loop() {
  moisture_value = analogRead(moisture_sensor);
  Serial.print("MOISTURE VALUE= ");
  Serial.print(moisture_value);
  Serial.println();
  delay(d1);
//moisture level check
  if (moisture_value >= 500)
  {
    digitalWrite(relay, LOW);
    delay(d1);

    //motor value calculate
    motor_value = analogRead(motor);
    motor_voltage = (5. / 1023.) * motor_value;
    Serial.print("   MOTOR VOLTAGE= ");
    Serial.print(motor_voltage);
    Serial.println();

    //motor value check
    if (motor_voltage <= 0.08)
```

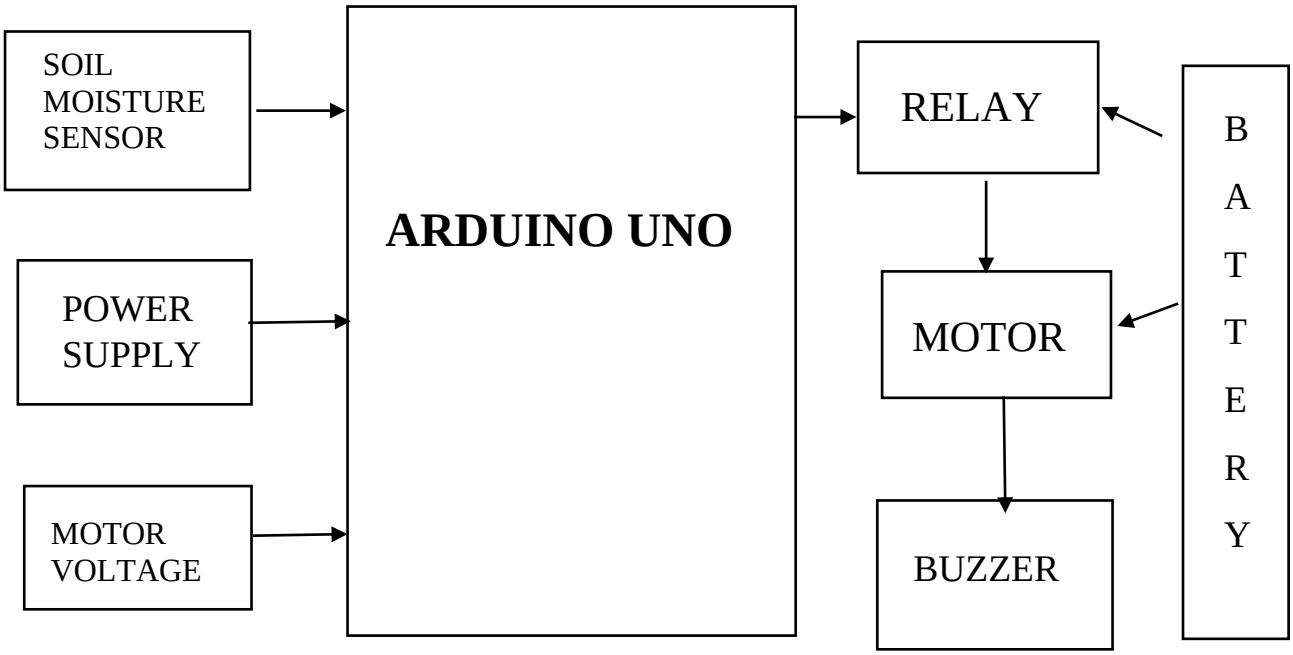
```
    {
      digitalWrite(relay, HIGH);
      delay(d1);
      digitalWrite(buzzer, HIGH);
      delay(1000);
      digitalWrite(buzzer, LOW);
    }
  }
  if (moisture_value <= 300)
  {
    digitalWrite(relay, HIGH);
    delay(d1);
  }
  delay(5000);
}
```

After executing both these methods, we come to a conclusion that solution 2 seemed to be more practical and promising in real time situations. Hence solution is implemented.

3.2 ALGORITHM:

1. START
2. SENSOR MEASURES THE MOISTURE LEVEL
3. IF VALUE > MOISTURE THRESHOLD
   - 3.1. RELAY- ON
   - 3.2. CHECK VOLTAGE ACROSS RESISTOR
   - 3.3. IF VOLTAGE> VOLTAGE THRESHOLD
     - 3.3.1 RELAY- ON
     - 3.3.2 WATER TILL VALUE REACHES THRESHOLD
   - 3.4. IF VOLTAGE < VOLTAGE THRESHOLD
     - 3.4.1. RELAY- OFF
     - 3.4.2. BUZZER ON FOR 10 SECONDS
     - 3.4.3. BUZZER OFF
4. IF VALUE < MOISTURE THRESHOLD
   - 4.1 RELAY- OFF
5. STOP

3.3 BLOCK DIAGRAM:



CHAPTER 4: CONCLUSION AND FUTURE SCOPE

4.1 CONCLUSION:

The automatic irrigation system using Arduino UNO has been experimented and the results were promising for home plants. Hence this system can be used when we are on a vacation. Also, there were many challenges faced while performing this experiment. The controllers to the Arduino were quite challenging, because of a single mistake can damage any electrical part. Furthermore, the batteries of this project drain quickly in use. All these challenges were resolved with suitable solutions. Finally, this project is tested and found to be working fine to minimize manpower and conserve time thereby making ours, a SMART GARDEN.

4.2 FUTURE SCOPE:

This system can be expanded to include various other options in small scale as well as large scale. According to the researchers,

“Early morning is the optimal time to water plants, avoiding the hottest part of the day when this will quickly evaporate and allowing a chance for plant roots to take in what they need. Late afternoon or early evening is ok for watering, so that the plants have time to drink up the water before the heat of the following day.”

Hence, we can make our system work only during those times using RTC module. This can also be varied seasonally by using temperature and humidity sensor. This project has even wider scopes to be dealt with.