Measuring Moisture of Soil

and

Self-Watering Plant System

Team Members:

Mohit Anil Manghnani (17BCE0558)

Krishnagopal Rajgopal Nair (17BCE2286)

Report submitted for the

Final Project Review of: Measuring Moisture of Soil and Self-Watering Plant System

Course Code: CSE2006 – MICROPROCESSOR AND INTERFACING

SLOT: A1

Professor: RAJARAJAN G.

School of Computer Science and Engineering

VIT University

Vellore-632014, India

TABLE OF CONTENTS

1. OBJECTIVE

Our ultimate objective was to make a self-watering plant model that can be scaled up depending on its specific application. For home use, the current model is sufficient. For large scale implementation, a larger microcontroller as well as a greater number of sensors and motors would be required. Cost can be minimized if the watering system is spread across a large field which has similar soil moisture throughout. In this case, a single sensor to measure soil moisture and a single motor to control the water supply would be sufficient.

2. ABSTRACT

In times during which water scarcity is becoming more and more apparent, it is automatically an engineer's onus to find a solution to this problem. Therefore, we tried to find something that can be implemented cheaply, but effectively so that we can show that saving water is an easy task if we put our mind to it.

3. INTRODUCTION

Many systems that we came across were large scale and required complex hardware and sensors for implementation. One of the small-model implementable ideas we came across was a self-watering plant. It was small enough and implementable using an Arduino Uno and a few peripheral components.

A self-watering plant is both convenient and environment friendly. If one wants to maintain plants at home without the added responsibility of watering them, or hiring someone else to take care of them, then a self-watering plant is very convenient. It being self-watering, we can water the plant at smaller intervals, using less water. This is what makes this project environment friendly.

4. LITERATURE SURVEY

The model that we found at https://www.instructables.com/id/Arduino-Self-Watering-Plant-Pot/ was quite inspiring for us in this project, but the cost it took for implementation discouraged us. They have used a water pump and a 3D printer for the implementation of their project. These were out of our budget. Then we decided to implement the idea in a different manner, which would considerably cut down costs.

We had to understand whether a simple digital input will be required or an analog input with readings ranging from 0 to 1023 will be required for this project. For this, we read the article at http://www.circuitstoday.com/arduino-soil-moisture-sensor. It was very helpful.

5. EXISTING AND PROPOSED SYSTEM

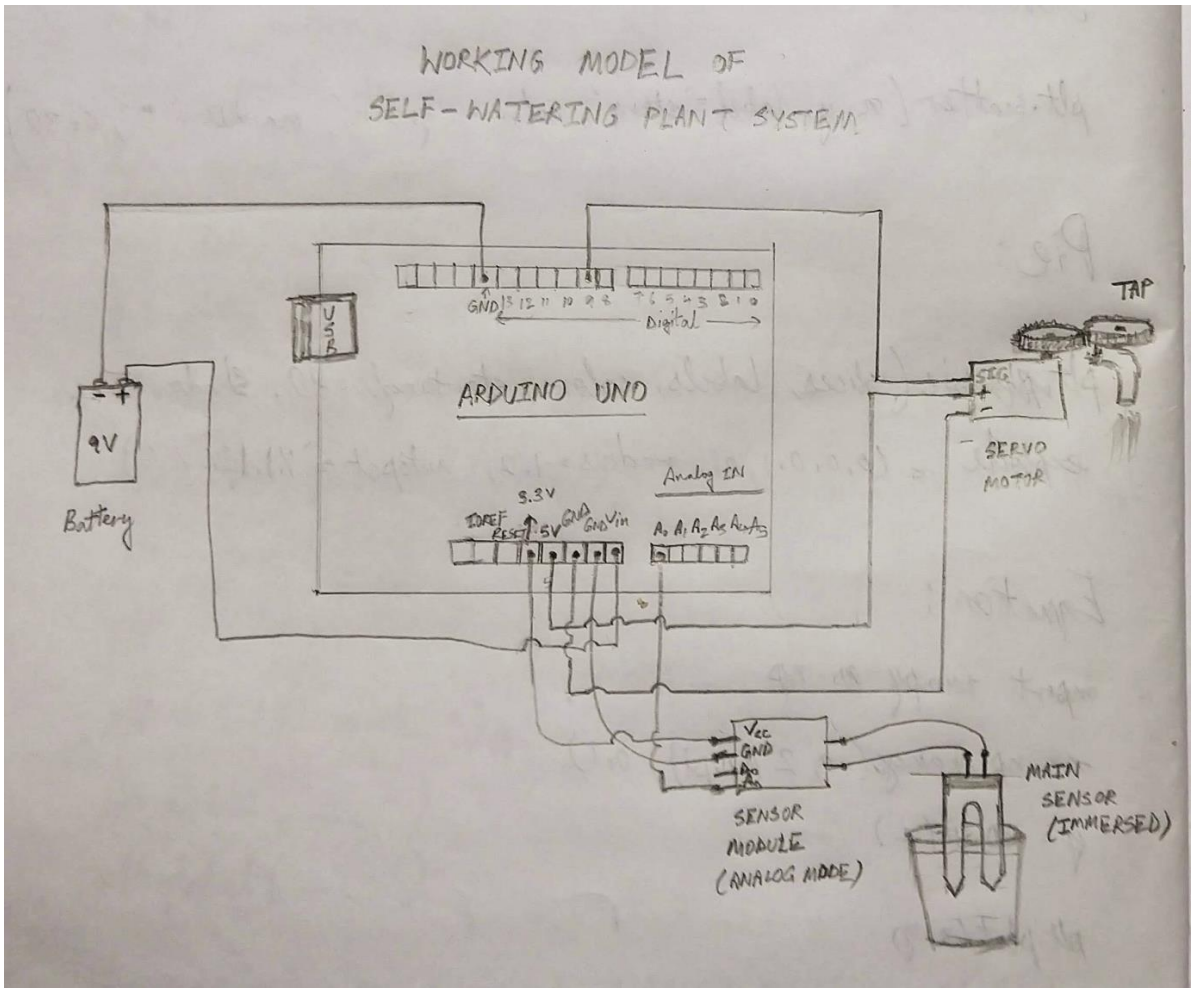
Description about existing model:

Since a soil moisture sensor already exists, there are many projects of self-watering plant. They use a motor to pump the water from a container into the pot. We used the potential energy of water flowing in taps. This is more energy efficient than all existing models that we have seen.

Proposed model:

We have used the Arduino Uno board to regulate the flow of water by using a servo motor to control a tap mechanism. We will use an Arduino soil moisture sensor to read the analog value of moisture in the soil. According to the reading, the tap is opened or closed by rotating the servo motor. Our new system consumes low amount of energy, which enables us to power the whole system using a single 9V battery.

6. SYSTEM DESIGN ARCHITECTURE



In this diagram, we can see that the Arduino Uno board is directly powered by the 9V battery via the V_{in} pin. The board in turn, powers the servo motor as well as the soil moisture sensor. The servo motor is powered via the 5V output and the soil sensor is powered via the 3.3V output.

The A_0 (analog output) pin of the sensor is connected to the A_0 pin of the Arduino board. The signal wire of the servo motor is connected to Digital In pin 9 of the Arduino board.

The servo motor is attached to a firm surface and connected to the tap via gears or directly (based on the availability of a firm surface).

7. ALGORITHM

- Initially, the tap is closed.
- The soil moisture reads the value of moisture from the soil and sends it to the Arduino board.
- If moisture in the soil is less than the predefined level, the Arduino boards sends a signal to the servo motor to rotate so that the tap is opened.
- The sensor continues to read moisture.
- Once the moisture level is greater than the predefined value, the servo motor is made to rotate in the opposite direction. This closes the tap.
- This process is repeated continuously as long as the system is active.

8. RESULT

An independent self-watering plant is ready which does not require human intervention for operation.

The tap is opened when moisture level of the soil of the plant is low. Hence, the plant is watered, causing the moisture level in the soil to rise and close the tap. When the moisture level of the plant decreases again, the process repeats.

9. REFRENCES

https://www.arduino.cc/en/Tutorial/BuiltInExamples

Examples of servo motor “Knob” operation and analog read. Complete documentation of Arduino board application is available here.

http://www.circuitstoday.com/arduino-soil-moisture-sensor

For interfacing Soil moisture and Arduino board.

https://www.instructables.com/id/Arduino-Self-Watering-Plant-Pot/

For topic idea.

10. ARDUINO CODE

```
#include <Servo.h>
Servo myservo;

const int VAL_PROBE = 0; // Analog pin 0
const int MOISTURE_LEVEL = 50; // the value beyond wh
ich tap is turned off

    void setup()   //code to initialize
    {
        Serial.begin(9600); //setting clock rate
         myservo.attach(9);  //Attaching pin 9 to
input for servo
    }

    void loop()   //code to repeat
    {
        //reading analog input from sensor
        int moisture = analogRead(VAL_PROBE);
        //converting input to percentage
        moisture= map(moisture,0,1023,100,0);
        //Printing values on screen(for analysis)
        Serial.println(moisture);

        //checking moisture level
        if(moisture > MOISTURE_LEVEL)
        {
            myservo.write(0); //closing tap
        }
        else
        {
            myservo.write(90);  //opening tap
        }
        delay(100);  //keeping buffer time for servo
input
    }
```