

SRM Institute of Science and Technology

College of Engineering and Technology

Department of Electronics and Communication Engineering

**18ECO109J-Embedded System Design Using Raspberry Pi
2023-2024 (Odd Semester)**

Mini Project Report

Name : Arnav Aggarwal
Register No. : RA2111032010002
Day / Session : 3/FN
Venue : TP1117-VLSI Simulation Lab
Project Title : SMART IRRIGATION SYSTEM
Lab Supervisor : Dr.Kanaparthi V Phani Kumar
Team Members : Ronit Kumar (RA2111032010009)

Particulars	Max. Marks	Marks Obtained
Objective & Description	05	
Algorithm,Flowchart,Program	20	
Demo verification	10	
Viva	10	
Report	05	
Total	50	

REPORT VERIFICATION

Date : 25.10.2023

Staff Name : Dr. Kanaparthi V Phani Kumar , Dr. A.Ruhan Bevi

SMART IRRIGATION SYSTEM

OBJECTIVE:

To create a Raspberry Pi-based automated plant watering system that monitors soil moisture and waters the plants when the soil becomes too dry. The system aims to provide basic and affordable solutions for plant enthusiasts, allowing them to customize and expand the project according to their needs.

ABSTRACT:

In today's fast-paced world, tending to plants often becomes a challenging task. The Smart Plant Watering System using Raspberry Pi addresses this issue by offering an innovative solution for plant care and irrigation. This project integrates the power of Raspberry Pi with sensors, actuators, and data analysis to create a comprehensive automated plant management system.

The system utilizes soil moisture sensors to measure the moisture level in the plant's environment. These sensors send data to the Raspberry Pi, which processes the information and determines whether the soil requires watering. When the soil moisture falls below a predefined threshold, the Raspberry Pi triggers a water pump to dispense the appropriate amount of water, ensuring the plants are adequately hydrated.

To enhance the system's efficiency and provide flexibility, it can be remotely controlled and monitored through a user-friendly interface. Users can set specific moisture thresholds, watering schedules, and receive notifications on their mobile devices. This level of customization allows for a tailored approach to plant care.

The Smart Plant Watering System offers a cost-effective and eco-friendly solution for plant enthusiasts, ensuring that plants receive the right amount of water at the right time, leading to healthier and more vibrant vegetation. With the power of Raspberry Pi and IoT technology, this project showcases how innovative automation can improve everyday tasks and contribute to sustainable living.

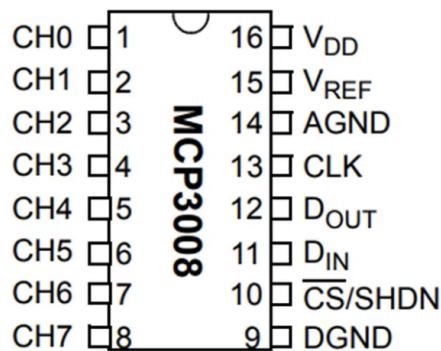
HARDWARE / SOFTWARE REQUIRED:

- Raspberry Pi
- M/M and M/F jumper wires

- Breadboard
- MCP3008 ADC(Analog-To-Digital Converter) Chip
- Capacitive Soil Moisture Sensor
- IRF520 MOS Module
- External Power Source
- Water Pump

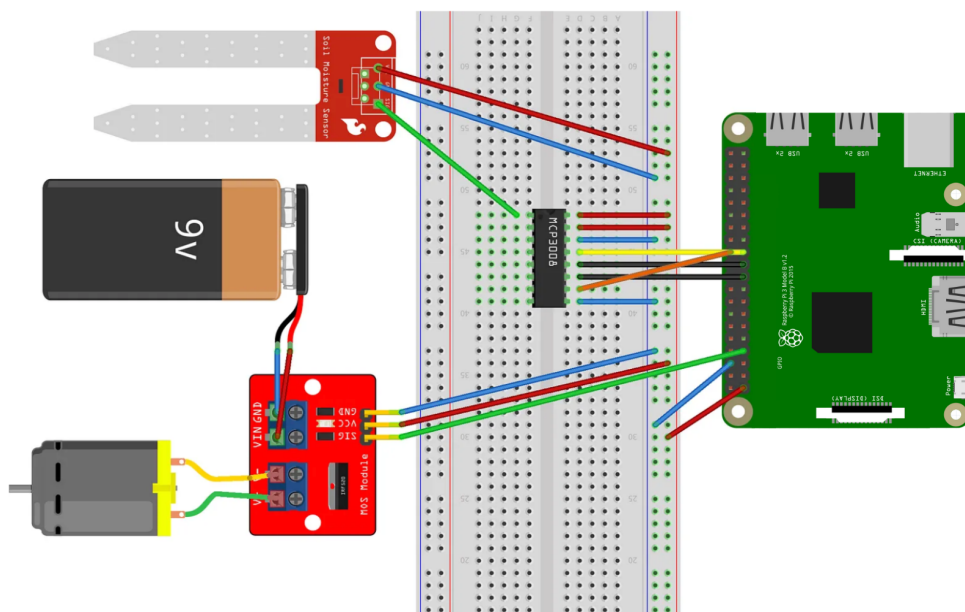
BLOCK DIAGRAM/CONNECTION DIAGRAM:

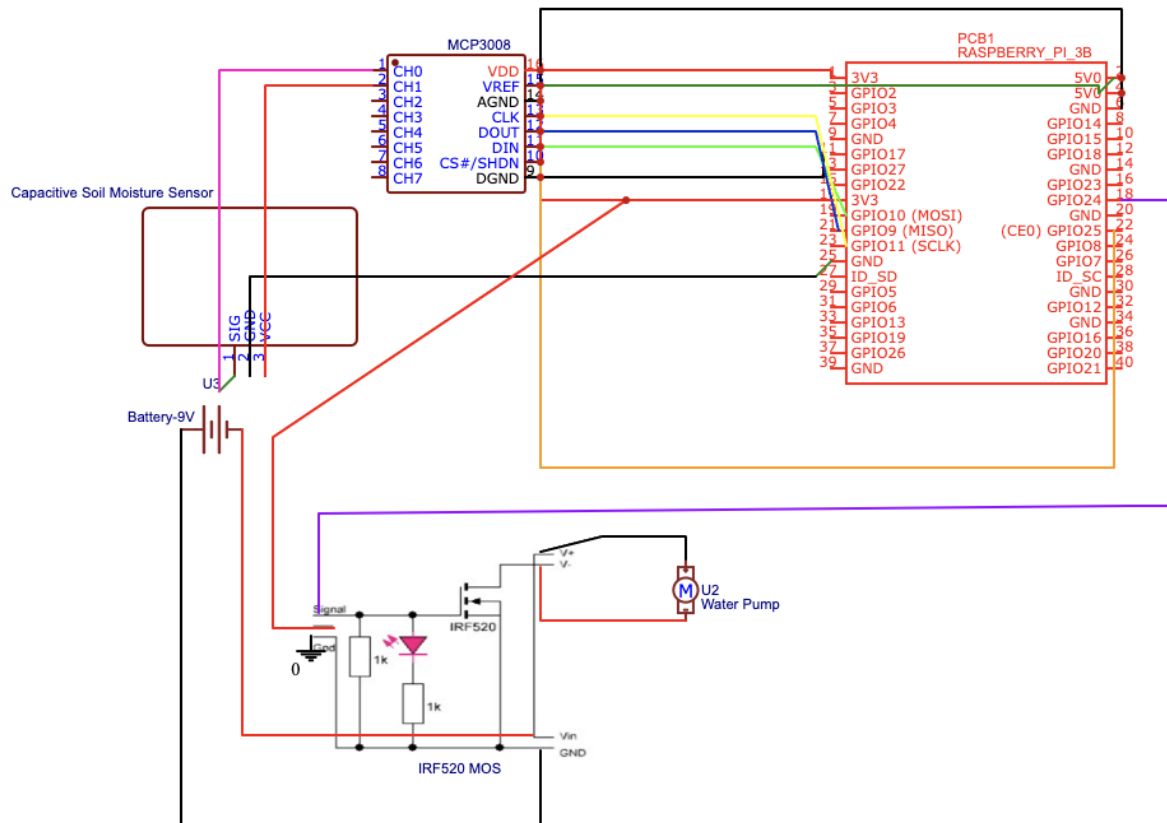
- Wire the MCP3008 to the Raspberry Pi. The following table shows how to connect the pins:



MCP3008	Raspberry Pi
VDD	3.3V
Vref	3.3V
AGND	GND
CLK	SCLK (GPIO 11)
Dout	MISO (GPIO 9)
Din	MOSI (GPIO 10)
CS/SHDN	CEO (GPIO 8)
DGND	GND

- Connect the sensor to your ADC by wiring the sensor's signal pin to one of the 8 channels. Also, connect GND and V_{CC} to the Pi's GND and 3.3V pins.
- Lastly, configure the mosfet. Connect the V_{in} with your external power source and V_{out} with your pump. Then, wire the mosfet to the Raspberry Pi: connect its signal pin to any of the Pi's GPIOs, GND to GND, and V_{DD} to 3.3V.



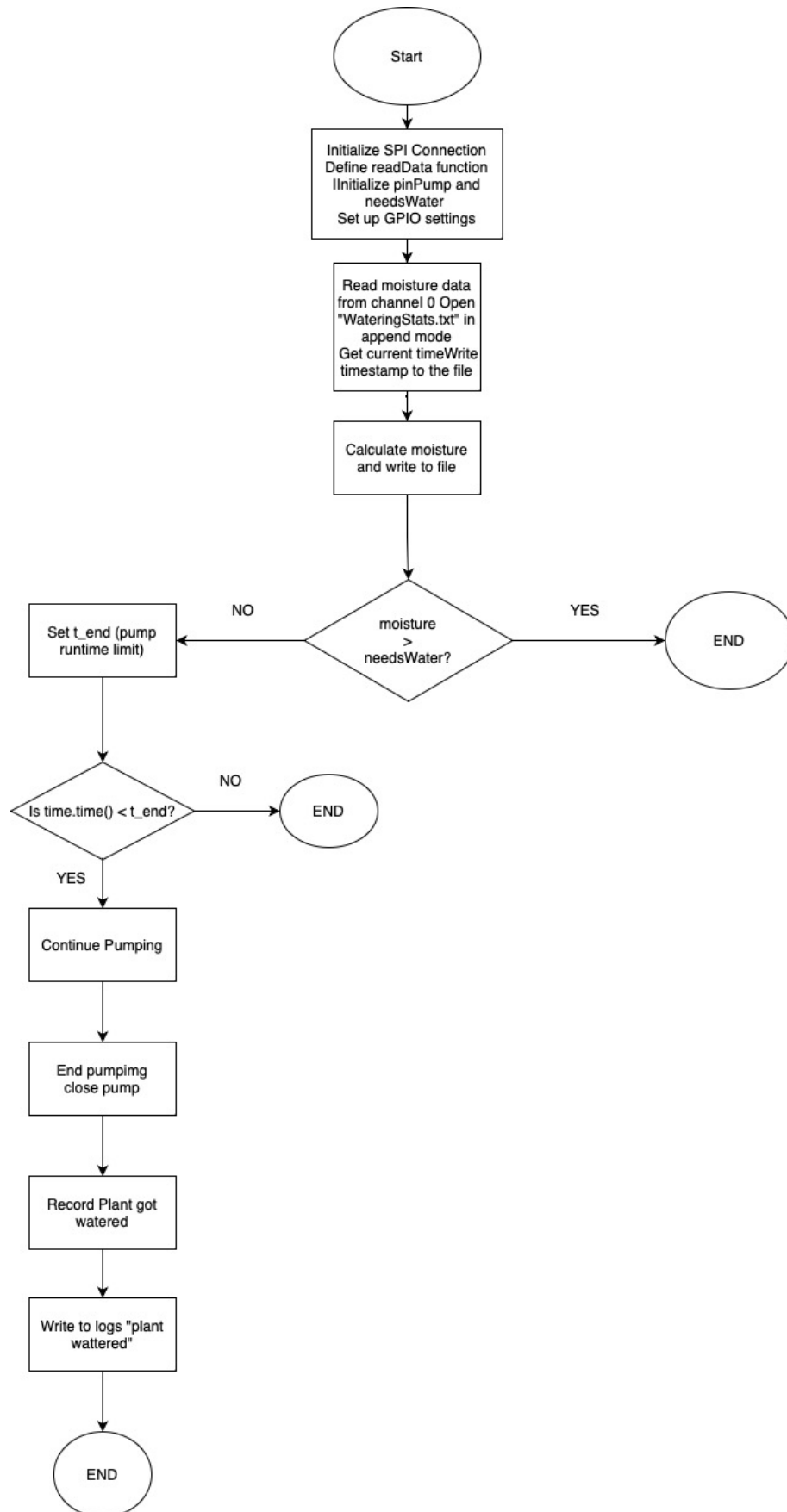


ALGORITHM:

1. Begin by importing essential libraries, such as RPi.GPIO for GPIO control, datetime for timestamps, spidev for SPI communication, and time for time-related functions.
2. Initialize an SPI connection using spidev on SPI channel 0 (CE0) with a maximum speed of 1 MHz.
3. Define a function **readData** to read data from the MCP3008 analog-to-digital converter through SPI.
4. Define variables for the GPIO pin connected to the water pump (**pinPump**) and the threshold sensor value for dry soil (**needsWater**).
5. Read the moisture data from channel 0 using the **readData** function. Open a file named **"WateringStats.txt"** in append mode and record the current timestamp.
6. Calculate the percentage moisture level by scaling the raw value from the sensor (moisture-330) to a range of 0-100.
7. Check if the moisture level is greater than the specified threshold (**needsWater**). If it is, proceed to water the plants:
 - Set a time limit (**t_end**) for how long the pump will run (4 seconds in this case).

- Activate the water pump (GPIO.HIGH) in a loop until the time limit is reached.
- Turn off the water pump (GPIO.LOW).
- Record in the statistics file that the plants have been watered.

FLOW CHART:



PROGRAM:

```
import RPi.GPIO as GPIO
import datetime
import spidev
import time

# create SPI connection
spi = spidev.SpiDev()
spi.open(0,0)
spi.max_speed_hz = 1000000 # 1 MHz

# function to read out data from MCP3008
def readData(channel):
    adc = spi.xfer2([1,(8+channel)<<4,0])
    data = ((adc[1]&3) << 8) + adc[2]
    return data

pinPump = 24 # GPIO pin of pump
needsWater = 630 # sensor value for dry air

# general GPIO settings
GPIO.setwarnings(False) # ignore warnings (irrelevant here)
GPIO.setmode(GPIO.BCM) # refer to GPIO pin numbers
GPIO.setup(pinPump, GPIO.OUT) # Pi can send voltage to pump
GPIO.output(pinPump, GPIO.LOW) # turn pump off

# read moisture data from channel 0
moisture = readData(0)

# write time and current moisture in statistic file
f = open("/home/pi/WateringStats.txt", "a")
currentTime = datetime.datetime.now()
f.write(str(currentTime) + ":\n")

# 450 = 780 - 330, moisture in %
f.write("Current moisture: " + str(round((moisture-330) / 450 * 100, 2)) +
"% (" + str(moisture) + ")\n")

# if plants are to dry, start pumping and record the moisture in file
if moisture > needsWater:
    t_end = time.time() + 4 # pump runs 4 seconds

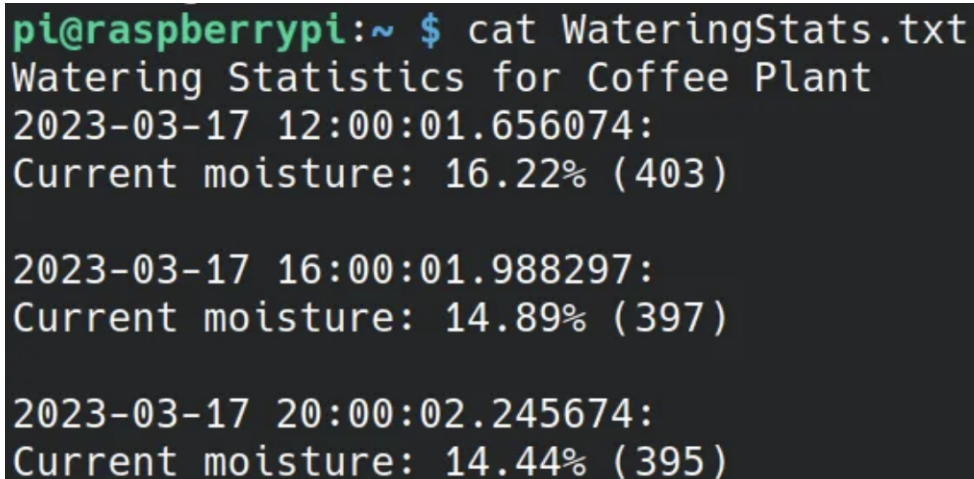
# actual pumping
while (time.time() < t_end):
```

```
GPIO.output(pinPump, GPIO.HIGH)
```

```
GPIO.output(pinPump, GPIO.LOW)    # turn pump off  
f.write("Plants get watered!\n")
```

```
f.write("\n")                    # line break for next log entry  
f.close()                        # close file  
GPIO.cleanup()                   # proper clean up of used pins
```

OUTPUT:



```
pi@raspberrypi:~ $ cat WateringStats.txt  
Watering Statistics for Coffee Plant  
2023-03-17 12:00:01.656074:  
Current moisture: 16.22% (403)  
  
2023-03-17 16:00:01.988297:  
Current moisture: 14.89% (397)  
  
2023-03-17 20:00:02.245674:  
Current moisture: 14.44% (395)
```

REAL TIME CONSTRAINTS:

The system checks soil moisture every 4 hours and waters the plants for 4 seconds when the soil is too dry. The specific time intervals can be adjusted to suit the user's preferences.

CONCLUSION:

The Raspberry Pi-based plant watering system provides an accessible and customizable solution for automating plant care. By monitoring soil moisture and acting accordingly, it ensures that plants receive water when needed, helping to maintain their health. The System can be expanded or enhanced according to the user's creativity and requirements, making it a versatile project for plant enthusiasts. Careful consideration of hardware compatibility and moisture thresholds is essential to ensure the system functions as intended.

References:

- <https://smart-watering.com/2022/03/20/irrigation-sensors-in-irrigation-systems/>
- <https://randomnerdtutorials.com/raspberry-pi-analog-inputs-python-mcp3008/>
- <https://www.ionos.com/digitalguide/hosting/technical-matters/cronjob/>