**19CSE303- EMBEDDED SYSTEMS**

**SMART IRRIGATION SYSTEM USING RASPBERRY-PI**

* **Materials Required:**

Raspberry Pi (any model with GPIO pins)

Rain sensor

Moisture sensor

DHT11 sensor (temperature and humidity)

LCD display (16x2 or 20x4)

Relay module

DC motor pump

Jumper wires

Breadboard

Power supply for Raspberry Pi and motor pump

* **Pin Connections:**

***Rain Sensor:***

VCC: 5V (Raspberry Pi Pin 2)

GND: GND (Raspberry Pi Pin 6)

DO: GPIO pin (e.g., GPIO17 - Raspberry Pi Pin 11)

***Moisture Sensor:***

VCC: 5V (Raspberry Pi Pin 2)

GND: GND (Raspberry Pi Pin 6)

A0: Analog GPIO pin (e.g., GPIO18 - Raspberry Pi Pin 12)

***DHT11 Sensor:***

VCC: 5V (Raspberry Pi Pin 2)

GND: GND (Raspberry Pi Pin 6)

Data: GPIO pin (e.g., GPIO27 - Raspberry Pi Pin 13)

***LCD Display:***

VCC: 5V (Raspberry Pi Pin 2)

GND: GND (Raspberry Pi Pin 6)

SDA: GPIO pin (e.g., GPIO2 - Raspberry Pi Pin 3)

SCL: GPIO pin (e.g., GPIO3 - Raspberry Pi Pin 5)

***Relay Module:***

VCC: 5V (Raspberry Pi Pin 2)

GND: GND (Raspberry Pi Pin 6)

IN: GPIO pin (e.g., GPIO17 - Raspberry Pi Pin 11)

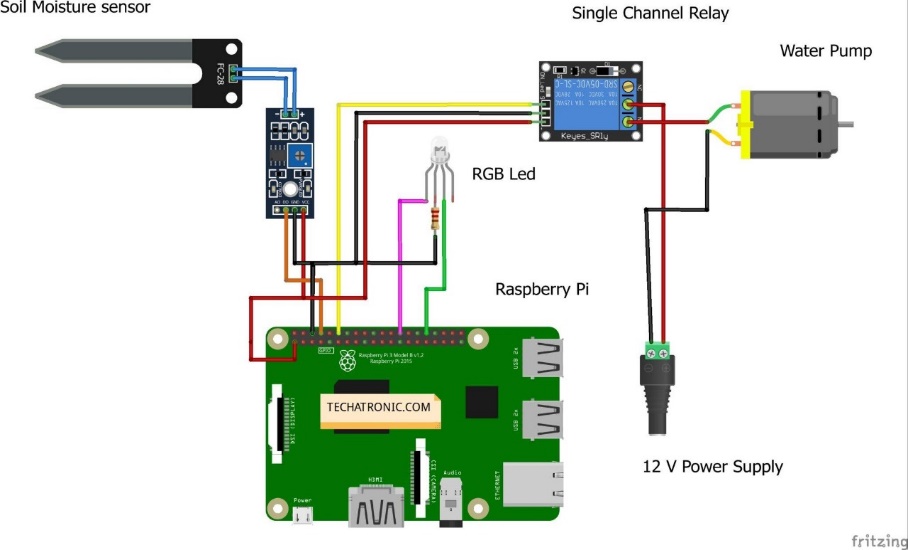
***Motor Pump:***

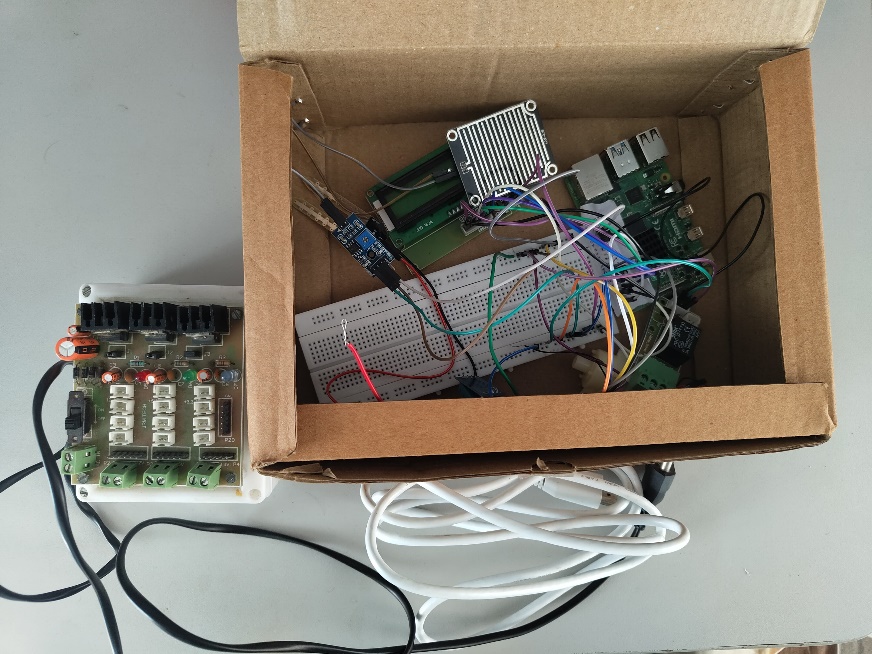
Connect the motor to the relay as per the relay module's specifications.

* **Working of Each Sensor**

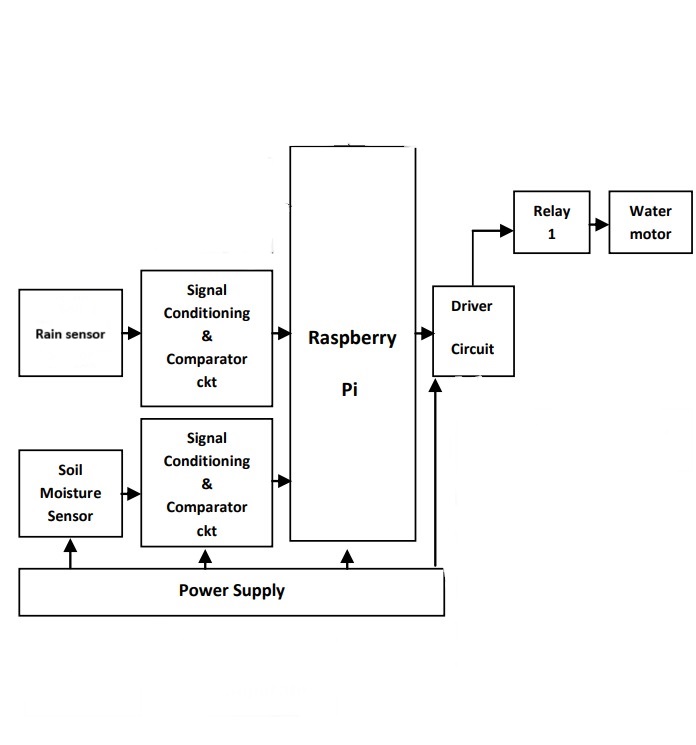
1. **Rain Sensor**: A rain sensor is a device that detects the presence of rain. It typically consists of two electrodes that are separated by a small gap. When raindrops fall on the electrodes, they complete the circuit and create a signal that can be detected by a microcontroller.
2. **Moisture Sensor**: A moisture sensor is a device that measures the amount of water in soil. It typically consists of two electrodes that are inserted into the soil. The electrical resistance between the electrodes changes as the moisture content of the soil changes. This change in resistance can be measured by a microcontroller and used to determine the moisture content of the soil.
3. **DHT11**: The DHT11 is a temperature and humidity sensor. It measures the temperature and humidity of the air and outputs the values as digital signals. The DHT11 is a popular choice for hobbyists and makers because it is inexpensive and easy to use.

* **Circuit Diagram**





* **Block Diagram**

****

* **Explanation**

The smart irrigation system is an automated system that uses sensors to monitor the moisture content of the soil, the presence of rain, and the temperature and humidity of the air. Based on the sensor data, the system will activate a motor pump to water the plants only when necessary. This will help to conserve water and prevent the plants from being overwatered or underwatered.

* **Code**

#!/usr/bin/python

import time

import os

import RPi.GPIO as GPIO

import Adafruit\_DHT

from urllib.parse import urlparse

import paho.mqtt.client as paho

import os,sys

GPIO.setmode(GPIO.BOARD)

GPIO.setwarnings(False)

# Set sensor type : Options are DHT11,DHT22 or AM2302

sensor=Adafruit\_DHT.DHT11

# Define GPIO to LCD mapping

LCD\_RS = 7

LCD\_E  = 11

LCD\_D4 = 12

LCD\_D5 = 13

LCD\_D6 = 15

LCD\_D7 = 16

DHT11\_Sensor\_Pin = 24 #gpio18 in board

RELAY\_PIN = 29

SOIL\_SENSOR\_PIN = 31

Rain\_SENSOR\_PIN = 33

'''

define pin for lcd

'''

# Timing constants

E\_PULSE = 0.0005

E\_DELAY = 0.0005

delay = 1

GPIO.setup(LCD\_E, GPIO.OUT)  # E

GPIO.setup(LCD\_RS, GPIO.OUT) # RS

GPIO.setup(LCD\_D4, GPIO.OUT) # DB4

GPIO.setup(LCD\_D5, GPIO.OUT) # DB5

GPIO.setup(LCD\_D6, GPIO.OUT) # DB6

GPIO.setup(LCD\_D7, GPIO.OUT) # DB7

GPIO.setup(RELAY\_PIN,GPIO.OUT)   # Set pin function as output

GPIO.setup(SOIL\_SENSOR\_PIN,GPIO.IN)   # Set pin function as input

GPIO.setup(Rain\_SENSOR\_PIN,GPIO.IN)   # Set pin function as input

def on\_connect(self, mosq, obj, rc):

    print("Connected with MQTT Server")

def on\_publish(mosq, obj, mid):

    print("mid: " + str(mid))

def on\_subscribe(mosq, obj, mid, granted\_qos):

    print("Subscribed: " + str(mid) + " " + str(granted\_qos))

mqttc = paho.Client()                        # object declaration

# Assign event callbacks

mqttc.on\_connect = on\_connect

mqttc.on\_publish = on\_publish

mqttc.on\_subscribe = on\_subscribe

url\_str = os.environ.get('CLOUDMQTT\_URL', 'tcp://broker.emqx.io:1883')

url = urlparse(url\_str)

mqttc.connect(url.hostname, url.port)

# Define some device constants

LCD\_WIDTH = 16    # Maximum characters per line

LCD\_CHR = True

LCD\_CMD = False

LCD\_LINE\_1 = 0x80 # LCD RAM address for the 1st line

LCD\_LINE\_2 = 0xC0 # LCD RAM address for the 2nd line

'''

Function Name :lcd\_init()

Function Description : this function is used to initialized lcd by sending the different commands

'''

def lcd\_init():

  # Initialise display

  lcd\_byte(0x33,LCD\_CMD) # 110011 Initialise

  lcd\_byte(0x32,LCD\_CMD) # 110010 Initialise

  lcd\_byte(0x06,LCD\_CMD) # 000110 Cursor move direction

  lcd\_byte(0x0C,LCD\_CMD) # 001100 Display On,Cursor Off, Blink Off

  lcd\_byte(0x28,LCD\_CMD) # 101000 Data length, number of lines, font size

  lcd\_byte(0x01,LCD\_CMD) # 000001 Clear display

  time.sleep(E\_DELAY)

'''

Function Name :lcd\_byte(bits ,mode)

Fuction Name :the main purpose of this function to convert the byte data into bit and send to lcd port

'''

def lcd\_byte(bits, mode):

  # Send byte to data pins

  # bits = data

  # mode = True  for character

  #        False for command

  GPIO.output(LCD\_RS, mode) # RS

  # High bits

  GPIO.output(LCD\_D4, False)

  GPIO.output(LCD\_D5, False)

  GPIO.output(LCD\_D6, False)

  GPIO.output(LCD\_D7, False)

  if bits&0x10==0x10:

    GPIO.output(LCD\_D4, True)

  if bits&0x20==0x20:

    GPIO.output(LCD\_D5, True)

  if bits&0x40==0x40:

    GPIO.output(LCD\_D6, True)

  if bits&0x80==0x80:

    GPIO.output(LCD\_D7, True)

  # Toggle 'Enable' pin

  lcd\_toggle\_enable()

  # Low bits

  GPIO.output(LCD\_D4, False)

  GPIO.output(LCD\_D5, False)

  GPIO.output(LCD\_D6, False)

  GPIO.output(LCD\_D7, False)

  if bits&0x01==0x01:

    GPIO.output(LCD\_D4, True)

  if bits&0x02==0x02:

    GPIO.output(LCD\_D5, True)

  if bits&0x04==0x04:

    GPIO.output(LCD\_D6, True)

  if bits&0x08==0x08:

    GPIO.output(LCD\_D7, True)

  # Toggle 'Enable' pin

  lcd\_toggle\_enable()

'''

Function Name : lcd\_toggle\_enable()

Function Description:basically this is used to toggle Enable pin

'''

def lcd\_toggle\_enable():

  # Toggle enable

  time.sleep(E\_DELAY)

  GPIO.output(LCD\_E, True)

  time.sleep(E\_PULSE)

  GPIO.output(LCD\_E, False)

  time.sleep(E\_DELAY)

'''

Function Name :lcd\_string(message,line)

Function  Description :print the data on lcd

'''

def lcd\_string(message,line):

  # Send string to display

  message = message.ljust(LCD\_WIDTH," ")

  lcd\_byte(line, LCD\_CMD)

  for i in range(LCD\_WIDTH):

    lcd\_byte(ord(message[i]),LCD\_CHR)

# Define delay between readings

delay = 5

lcd\_init()

lcd\_string("welcome ",LCD\_LINE\_1)

time.sleep(2)

lcd\_string("Smart Irrigation ",LCD\_LINE\_1)

lcd\_string("Project",LCD\_LINE\_2)

time.sleep(2)

GPIO.output(RELAY\_PIN,GPIO.HIGH)  #Relay OFf

motor\_staus = 1

while 1:

    rc = mqttc.loop()

    Moisture\_output = GPIO.input(SOIL\_SENSOR\_PIN)

    Rain\_output = GPIO.input(Rain\_SENSOR\_PIN)

    print("Rain\_output" + str(Rain\_output))

    # Reading the DHT11 is very sensitive to timings and occasionally

    # the Pi might fail to get a valid reading. So check if readings are valid.

    # Use read\_retry method. This will retry up to 15 times to

    # get a sensor reading (waiting 2 seconds between each retry).

    humidity, temperature = Adafruit\_DHT.read\_retry(sensor, DHT11\_Sensor\_Pin)

    if humidity is not None and temperature is not None:

        lcd\_byte(0x01,LCD\_CMD) # 000001 Clear display

        lcd\_string("Temperature="+ str(temperature),LCD\_LINE\_1)

        lcd\_string("Humidity ="+ str(humidity),LCD\_LINE\_2)

        mqttc.publish("hum",str(humidity))

        mqttc.publish("Tem",str(temperature))

        time.sleep(1)

        #print('Temp={0:0.1f}\*C  Humidity={1:0.1f}% '.format(temperature, humidity))

    else:

        lcd\_byte(0x01,LCD\_CMD) # 000001 Clear display

        lcd\_string("Failed to connect",LCD\_LINE\_1)

        lcd\_string("DHT11 sensor",LCD\_LINE\_2)

        time.sleep(1)

        break

    if(Rain\_output == 1):

        if(((temperature > 30) and (humidity < 27)) or (Moisture\_output == 1)):

            lcd\_byte(0x01,LCD\_CMD) # 000001 Clear display

            lcd\_string("Motor On",LCD\_LINE\_2)

            time.sleep(1)

            mqttc.publish("mot","1")

            GPIO.output(RELAY\_PIN,GPIO.LOW)  #Relay ON

        else:

            lcd\_byte(0x01,LCD\_CMD) # 000001 Clear display

            lcd\_string("Motor OFF",LCD\_LINE\_2)

            time.sleep(1)

            mqttc.publish("mot","0")

            GPIO.output(RELAY\_PIN,GPIO.HIGH)  #Relay ON

    else:

        GPIO.output(RELAY\_PIN,GPIO.HIGH)  #Relay OFf

        mqttc.publish("mot","0")

        #Rain is detected do not start the motor

        lcd\_byte(0x01,LCD\_CMD) # 000001 Clear display

        lcd\_string("Rain Detected",LCD\_LINE\_1)

        lcd\_string("Motor Stop",LCD\_LINE\_2)

        time.sleep(1)