LOVELY PROFESSIONAL UNIVERSITY

ECE363

SMART AGRICULTURE SYSTEM

Project Report

Submitted By:

Abhinav wadia - 11612507
Chinmay dhok – 11610932
Gaurav thakur – 11610713
Krishna kanhaiya-11611104

ACKNOWLEDGEMENT

We, four students of Computer Science Engineering of 4th year in Lovely Professional University, Phagwara are preparing a final year project name “Smart Agriculture System”. We whole heartedly express our sincere gratitude to Mrs. Aarti who guided us for the completion of the project. We are also thankful to Mr. Swapnil Bagwari for explaining on critical aspect of topics related to the project. We are also grateful to the assistances of Workshop for permitting us to have some help from them.

INDEX

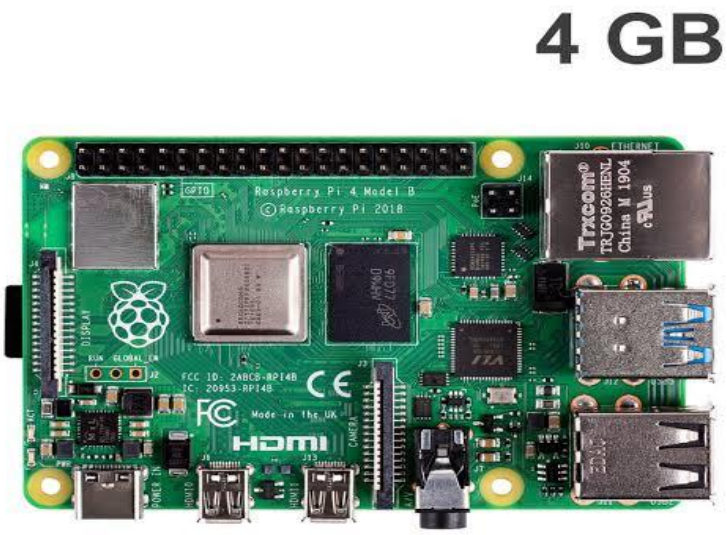
COMPONENTS USED

- Raspberry pi
- DHT 11
- Ultrasonic sensor
- MQ2 Smoke sensor
- Analog to Digital converter

RASBERRY PI

The **Raspberry Pi** is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. It is a capable little device that enables people of all ages to explore computing, and to learn how to program in languages like Scratch and Python.

HOW IT WORKS: An SD card inserted into the slot on the board acts as the hard drive for the Raspberry Pi. It is powered by USB and the video output can be hooked up to a traditional RCA TV set, a more modern monitor, or even a TV using the HDMI port.

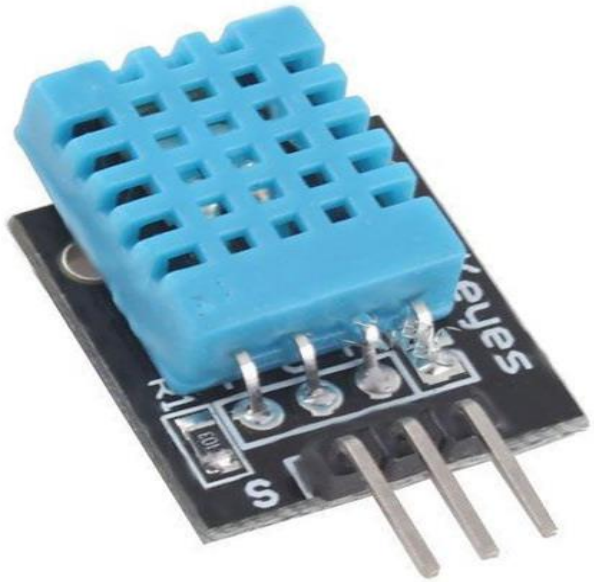


DHT 11

The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermostat to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). It is fairly simple to use, but requires careful timing to grab data.

Working:

The DHT11 calculates relative humidity by measuring the electrical resistance between two electrodes. The humidity sensing component of the DHT11 is a moisture holding substrate with the electrodes applied to the surface.



DHT11 Specifications:

- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%
- Resolution: Temperature and Humidity both are 16-bit

- Accuracy: ±1°C and ±1%

ULTRASONIC SENSOR

Ultrasonic sensors measure distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception.

Working: Ultrasonic sensors work by emitting sound waves at a frequency too high for humans to hear. They then wait for the sound to be reflected back, calculating distance based on the time required. This is similar to how radar measures the time it takes a radio wave to return after hitting an object.

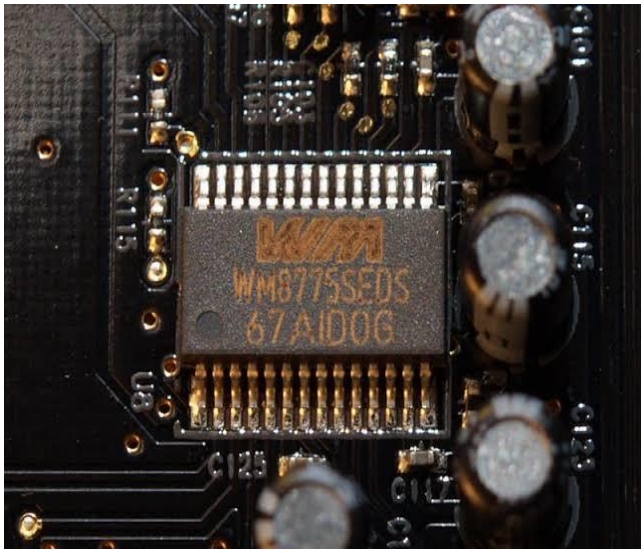
SMOKE SENSOR

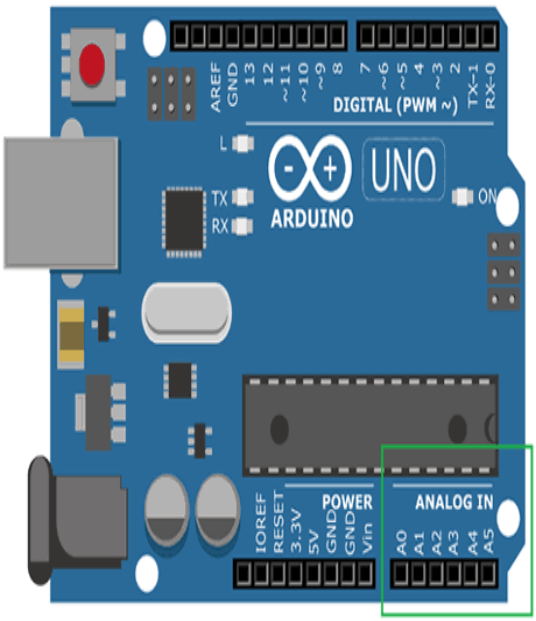
A smoke detector is a device that senses smoke, typically as an indicator of fire. Commercial security devices issue a signal to a fire alarm control panel as part of a fire alarm system, while household smoke detectors, also known as smoke alarms, generally issue a local audible or visual alarm from the detector itself or from a number of detectors if there are multiple smoke detectors interlinked.

Working: Photoelectric-type alarms aim a light source into a sensing chamber at an angle away from the sensor. Smoke enters the chamber, reflecting light onto the light sensor; triggering the alarm. For each type of smoke alarm, the advantage it provides may be critical to life safety in some fire situations.

ANALOG TO DIGITAL CONVERTER

It is a system that converts an analog signal, such as a sound picked up by a microphone or light entering a digital camera, into a digital signal. An ADC may also provide an isolated measurement such as an electronic device that converts an input analog voltage or current to a digital number representing the magnitude of the voltage or current. Typically the digital output is a two's complement binary number that is proportional to the input, but there are other possibilities.





CODE for MAIN

```
import RPi.GPIO as GPIO
import dht11
import sys
import time
import urllib2
from time import sleep
import datetime
import csv
import smtplib
MAX_TEMP = 45
MIN_TEMP = 15
MAX_HUMIDITY = 75
MIN_HUMIDITY = 35
SENDER = "gouravthakur2191999@gmail.com"
RECEIVER = "chinmaydhok2013@gmail.com"
def send_warning(val):
    try:
        sender = SENDER
        receiver = RECEIVER
        server = smtplib.SMTP('smtp.gmail.com', 587)
        server.ehlo()
        server.starttls()
        server.login(sender, "thakurgourav219")
        subject = "Warning"
        text = "Please check the room humidity and temperature!"
        if val == 0:
            subject = "Temperature risen above %d C!" % MAX_TEMP
            text = "Warning the temperature has increased above %d" % MAX_TEMP
```

```
        elif val == 1:
            subject = "Humdity risen above %d percent!" % MAX_HUMIDITY
            text = "Warning the humidity has increased above %d" % MAX_HUMIDITY
        from email.Message import Message
        m = Message()
        m['X-Priority'] = '2'
        m['Subject'] = subject
        m.set_payload(text)
        server.sendmail(sender,receiver,m.as_string())
        print("Warning sent")
        sleep(20)
    except Exception, ex:
        print(ex)
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
instance = dht11.DHT11(pin=24)
while True:
    try:
        result = instance.read()
        #sleep(30)
        #print'f'
        if result.is_valid():
            temp = result.temperature
            humi = result.humidity
        #print'a'
            if int(temp) > MAX_TEMP:
                send_warning(0)
            if int(humi) > MAX_HUMIDITY:
                send_warning(1)
```

```
        currentDT = datetime.datetime.now()
        date = currentDT.strftime("%Y/%m/%d")
        time = currentDT.strftime("%H:%M:%S")
        myCsvRow = [temp, humi]
        print(myCsvRow,date,time)

        with open(r'TempHumidity.csv', 'a') as fd:
            writer = csv.writer(fd);
            writer.writerow(myCsvRow)

        myAPI = 'P58SIT323JAFS83X'
        baseURL = 'https://api.thingspeak.com/update?api_key=%s' % myAPI
        url = urllib2.urlopen(baseURL + '&field1=%s&field2=%s' % (temp, humi))
        sleep(5)
    else:
        # print"error while acccessing GPIO/ connecting in 2 seconds"
        print("Connecting to https://api.thingspeak.com/.....")
        sleep(5)
except Exception as e:

    print (e)
    break
```

CODE FOR MQ2

```
import RPi.GPIO as GPIO
import time
import urllib2
SPICLK = 11
SPIMISO = 9
SPIMOSI = 10
SPICS = 8
mq2_dpin = 26
mq2_apin = 0
def init():
        GPIO.setwarnings(False)
        GPIO.cleanup()
        GPIO.setmode(GPIO.BCM)
        GPIO.setup(SPIMOSI, GPIO.OUT)
        GPIO.setup(SPIMISO, GPIO.IN)
        GPIO.setup(SPICLK, GPIO.OUT)
        GPIO.setup(SPICS, GPIO.OUT)
        GPIO.setup(mq2_dpin,GPIO.IN,pull_up_down=GPIO.PUD_DOWN)
def readadc(adcnum, clockpin, mosipin, misopin, cspin):
       if ((adcnum > 7) or (adcnum < 0)):
              return -1
       GPIO.output(cspin, True)

       GPIO.output(clockpin, False)
       GPIO.output(cspin, False)

       commandout = adcnum
       commandout |= 0x18
```

```
        commandout <<= 3
        for i in range(5):
                if (commandout & 0x80):
                        GPIO.output(mosipin, True)
                else:
                        GPIO.output(mosipin, False)
                commandout <<= 1
                GPIO.output(clockpin, True)
                GPIO.output(clockpin, False)

        adcout = 0


        for i in range(12):
                GPIO.output(clockpin, True)
                GPIO.output(clockpin, False)
                adcout <<= 1
                if (GPIO.input(misopin)):
                        adcout |= 0x1

        GPIO.output(cspin, True)

        adcout >>= 1
        return adcout
#main ioop
def main():
         init()
         print"please wait..."
         time.sleep(5)
```

```
        while True:
            COlevel=readadc(mq2_apin, SPICLK, SPIMOSI, SPIMISO, SPICS)
            myAPI = 'P58SIT323JAFS83X'
            baseURL = 'https://api.thingspeak.com/update?api_key=%s' % myAPI
            url = urllib2.urlopen(baseURL + '&field3=%s' % (COlevel))

            if mq2_dpin > 700:
               # print("Gas leakage")
                print(COlevel)

                #print"Current Gas AD vaule = " +str("%.2f"%((COlevel/1024.)*3.3))+" V"
                time.sleep(3)
            else:
               # print("Gas not leaked.")
                print(COlevel)
                time.sleep(3)

if __name__ =='__main__':
        try:
            main()
            pass
        except KeyboardInterrupt:
            pass

GPIO.cleanup()
```

CODE for ULTRASONIC SENSOR

```
import RPi.GPIO as GPIO
import time
import urllib2
GPIO.setmode(GPIO.BCM)
GPIO_TRIGGER = 21
GPIO_ECHO = 20
GPIO.setwarnings(False)
GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)

def distance():
    GPIO.output(GPIO_TRIGGER, True)
    time.sleep(0.0001)
    GPIO.output(GPIO_TRIGGER, False)
    StartTime = time.time()
    StopTime = time.time()
    while GPIO.input(GPIO_ECHO) == 0:
        StartTime = time.time()
    while GPIO.input(GPIO_ECHO) == 1:
        StopTime = time.time()
    TimeElapsed =StopTime - StartTime
    distance = (TimeElapsed * 34300) / 2
    return distance
if __name__ == '__main__':
    try:
        while True:
            dist = distance()
            if dist < 100:
```

```
            print ("Someone has been showed on the radar.Measured Distance = %.1f cm" %
dist)
        else:
            print ("FARM  IS SECURED.")
        #myAPI = 'OAQRNCAU6GWBL3C0'
        #baseURL = 'https://api.thingspeak.com/update?api_key=%s' % myAPI
        #url = urllib2.urlopen(baseURL + '&field4=%s' % (dist))
        time.sleep(10)
    except KeyboardInterrupt:
        print("Measurement stopped by User")
        GPIO.cleanup()
```