VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT on

Internet of Things Lab

Submitted by:

Goutham S Pujar (1BM21CS277)

Under the Guidance of

Prof. Sandhya A Kulkarni
Assistant Professor Dept. of CSE, BMSCE

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019 November 2023 - March 2024

1

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum) **Department of Computer Science and Engineering**

CERTIFICATE

This is to certify that the Lab work entitled "**Internet of Things Lab**" carried out by **Goutham S Pujar (1BM21CS277),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2023-24. The Lab report has been approved as it satisfies the academic requirements in respect of **Internet of Things Lab- (22CS5PCIOT)** work prescribed for the said degree.

Prof. Sandhya A Kulkarni
Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

Table Of Contents

Experiment Title			Page No.
Experiments			**1**
	Experiment - 1		**1**
	1	**Aim:** LED Blinking. Turn ON the LED for one second, then off for one second repeatedly.	**1**
	2	**Hardware Components**	**1**
	3	**Circuit Diagram**	**1**
	4	**Code**	**1**
	5	**Observation**	**1**
	Experiment - 2		**2 - 3**
	2.1	**Aim:** Turn an LED ON/OFF using a push button.	**2**
	2.2	**Hardware Components**	**2**
	2.3	**Circuit Diagram**	**2**
	2.4	**Code**	**2**
	2.5	**Observation**	**3**
3.	**Experiment - 3**		**4**
	3.1	**Aim:** To control the brightness of an LED using a Potentiometer.	**4**
	3.2	**Hardware Components**	**4**

	3.3	**Circuit Diagram**	**4**
	3.4	**Code**	**4**
	3.5	**Observation**	**4**
4.	**Experiment - 4**		**5**
	4.1	**Aim:** LED Fading	**5**
	4.2	**Hardware Components**	**5**
	4.3	**Circuit Diagram**	**5**
	4.4	**Code**	**5**
	5	**Observation**	**5**
5.	**Experiment - 5**		**6 - 7**
	5.1	**Aim:** Simulating a night light using LDR and PIR.	**6**
	5.2	**Hardware Components**	**6**
	5.3	**Circuit Diagram**	**6**
	5.4	**Code**	**6**
	5.5	**Observation**	**7**
6.	**Experiment - 6**		**8 - 9**
	6.1	**Aim:** PIR sensor with Arduino	**8**
	6.2	**Hardware Components**	**8**
	6.3	**Circuit Diagram**	**8**
	6.4	**Code**	**8**
	6.5	**Observation**	**9**
7.	**Experiment - 7**		**10 - 11**

	7.1	**Aim:** Ultrasound sensor with Arduino	**10**
	7.2	**Hardware Components**	**10**
	7.3	**Circuit Diagram**	**10**
	7.4	**Code**	**10**
	7.5	**Observation**	**11**
8.	**Experiment - 8**		**12 - 13**
	8.1	**Aim:** Design and implement Fire alarm system using flame sensor and buzzer.	**12**
	8.2	**Hardware Components**	**12**
	8.3	**Circuit Diagram**	**12**
	8.4	**Code**	**12**
	8.5	**Observation**	**13**
9.	**Experiment - 9**		**14 - 15**
	9.1	**Aim:**	**14**
		Design and implement smart irrigation system using Soil Moisture sensor and Servo Motor.	
	9.2	**Hardware Components**	**14**
	9.3	**Circuit Diagram**	**14**
	9.4	**Code**	**14**
	9.5	**Observation**	**15**
10.	**Experiment - 10**		**16 - 17**
	10.1	**Aim:** Read the temperature and print it in serial port.	**16**
	10.2	**Hardware Components**	**16**
	10.3	**Circuit Diagram**	**16**

	10.4	**Code**	**16**
	10.5	**Observation**	**17**
11	**Experiment - 11**		**18 - 20**
	11	**Aim:** Implement RFID. Design and implement an access control system using RFID.	**18**
	12	**Hardware Components**	**18**
	13	**Circuit Diagram**	**18**
	14	**Code**	**18**
	15	**Observation**	**20**
12	**Experiment - 12**		**21 - 22**
	11	**Aim:** Working with Arduino Bluetooth Module.	**21**
	12	**Hardware Components**	**21**
	13	**Circuit Diagram**	**21**
	14	**Code**	**21**
	15	**Observation**	**22**
13	**Experiment - 13**		**23 - 25**
	13.1	**Aim:** Design and implement a system to realize Bluetooth Master/Slave scenario.	**23**
	13.2	**Hardware Components**	**23**
	13.3	**Circuit Diagram**	**23**
	13.4	**Code**	**23**
	13.5	**Observation**	**25**
15	**Experiment - 15**		**27**

	15.1	**Aim:** Setting up a GSM module. Call using a Arduino and GSM module to a specified mobile number inside the program.	**27**
	15.2	**Hardware Components**	**27**
	15.3	**Circuit Diagram**	**27**
	15.4	**Code**	**27**
	15.5	**Observation**	**27**
16	**Experiment - 16**		**28 - 29**
	16.1	**Aim:** Call using a Arduino and GSM module to a specified mobile number inside the program when a flame sensor detects fire.	**28**
	16.2	**Hardware Components**	**28**
	16.3	**Circuit Diagram**	**28**
	16.4	**Code**	**28**
	16.5	**Observation**	**29**

	17	**Experiment - 17**		**30 - 31**
		17.1	**Aim:** **a)** Send a SMS using Arduino and GSM module to a specific mobile number inside the program. **b)** Receive SMS using Arduino and GSM module to the SIM card loaded in the GSM Module.	**30**
		17.2	**Hardware Components**	**30**
		17.3	**Circuit Diagram**	**30**
		17.4	**Code**	**30**
		17.5	**Observation**	**31**
	18	**Experiment - 18**		**32 - 33**

		18.1	**Aim:** Use received message through Arduino and GSM module to control switching ON/OFF the LED.	**32**
		18.2	**Hardware Components**	**32**
		18.3	**Circuit Diagram**	**32**
		18.4	**Code**	**32**
		18.5	**Observation**	**33**

1 Experiment - 1

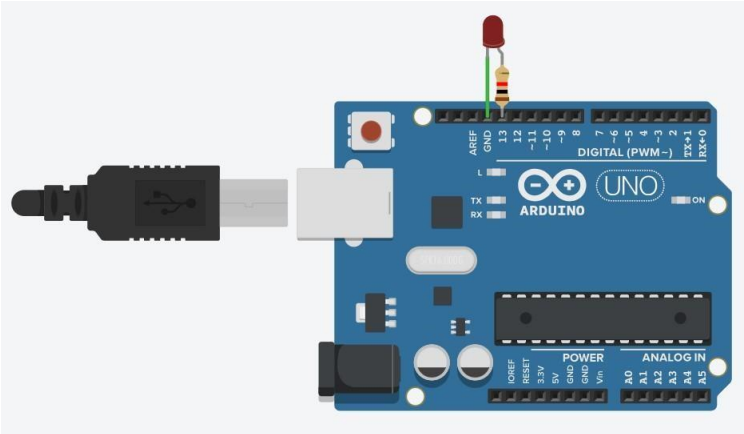
Aim:

LED Blinking. Turn ON the LED for one second, then off for one second repeatedly.

Hardware Components:

Arduino Uno, LEDs, Resistor

Circuit Diagram:



Code:

```
int led = 13;   void setup() {
pinMode(led, OUTPUT);
}  void loop() { digitalWrite(led,
HIGH);  digitalWrite(led,
LOW);
}
```

Observation:

LED switches ON/OFF periodically.

2 2

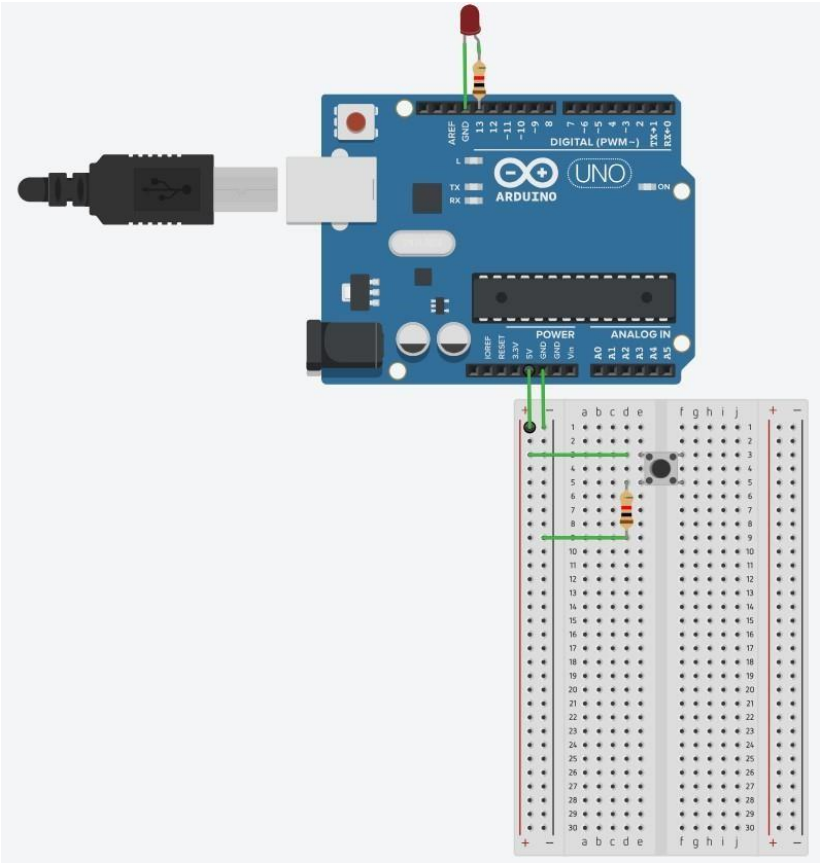
Turn an LED ON/OFF using a push button.

Hardware Components:

Experiment - Aim:

Arduino Uno, LED, push-button, breadboard

Circuit Diagram:



Code: const int buttonPin =
2; const
int ledPin 13;
int buttonState = 0;

void setup() { pinMode(ledPin,
OUTPUT); pinMode(buttonPin,
INPUT); } void loop() { buttonState =
digitalRead(ButtoPin); if (buttonState
==
HIGH) { digitalWrite(ledPin, HIGH);} else
{ digitalWrite(ledPin, LOW):}
}

Observation:

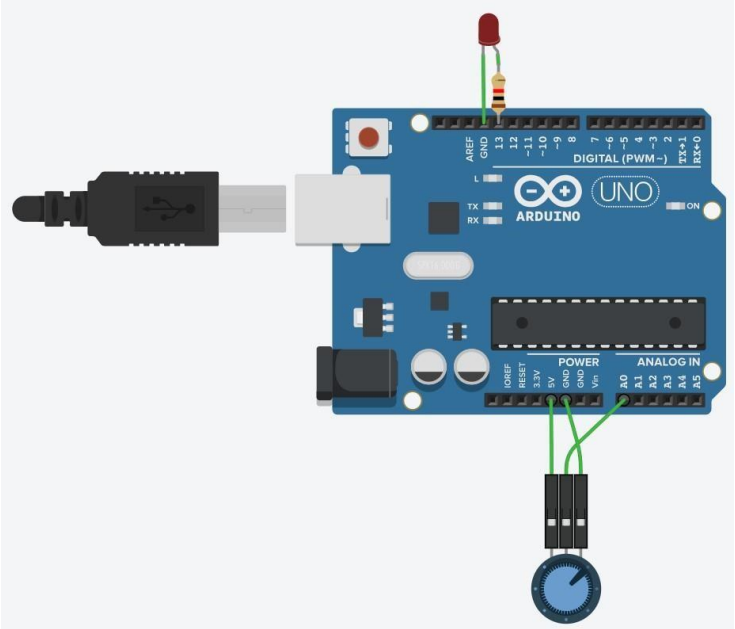
We can power on the LED only when the button is pressed, or make the LED blink when you release the button.

3 3

To control the brightness of an LED using a Potentiometer.

Hardware Components:
Arduino Uno, LED, resistor, potentiometer

Circuit Diagram:



Code:

```
const int analogInPin = A0;
const int analogOutPin = 13; int sensorValue
= 0;
int outputValue = 0;

void setup() {
        Serial.begin(9600);
}

void loop() {  sensor Value = analogRead(analogInPin);
        outputValue = map(sensorValue, 0, 1023, 0, 255);
        analogWrite(analogOutPin, outputValue);
```

Experiment - Aim:

```
Serial.print(sensorValue);
Serial.println(outputValue); delay(2);  }
```

Observation:

While varying the resistance using the shaft of the potentiometer, fading in the brightness of the LED is seen.

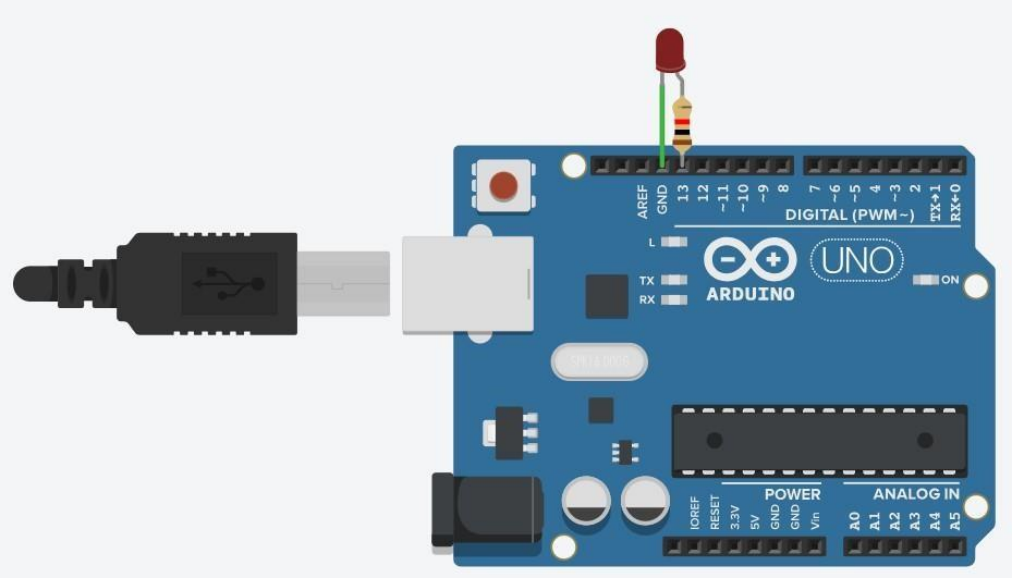
Experiment - Aim:

4 4

LED Fading.

Hardware Components:
Arduino Uno, resistor, LED.

Circuit Diagram:



Code:

```
int ledPin = 9;

void setup() {  }

void loop()
{
        for (int fadeValue = 0 ; fadeValue <= 255; fadeValue += 5)
        { analogWrite(ledPin, fadeValue);
delay(30);
        }

        for (int fadeValue = 255 ; fadeValue >= 0; fadeValue -= 5)
```

Experiment - Aim:

```
        { analogWrite(ledPin, fadeValue);
delay(30);
        }
}
```

Observation:

The LED fades from high value to low value at a delay of 30ms.

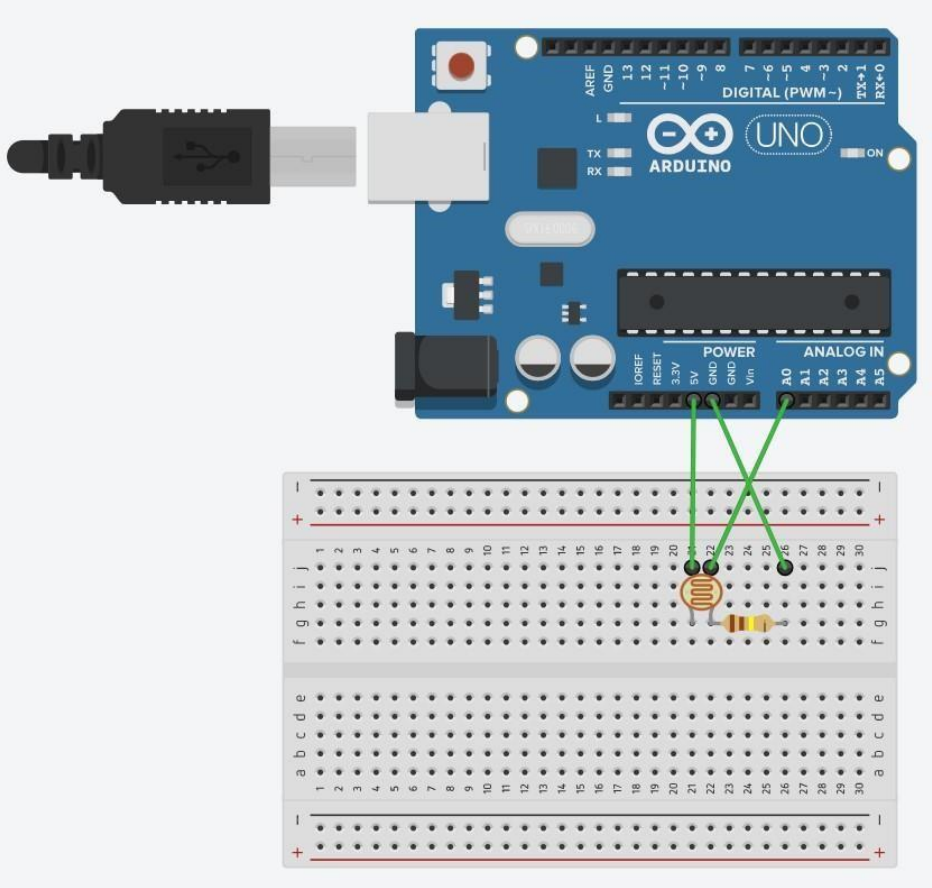
5 5

Simulating a night light using LDR and PIR.

Hardware Components:

Arduino Uno, Breadboard, LED, PIR, LDR, resistor

Circuit Diagram:



Code:

```
int LDR = 0;  int
LDRValue = 0;  int
light_sensitivity = 500;

void setup()
{
        Serial.begin(9600);
        pinMode(11, OUTPUT);   }

void loop()
{
        LDRValue = analogRead(LDR);  Serial.println(LDRValue);
        delay(50);
        if (LDRValue < light_sensitivity)
        {  digitalWrite(11, HIGH);
        }  else
        {                digitalWrite(11,
LOW);
        }
        delay(1000);
 }
```

Observation:

While lights are switched off in the room, LED should switch ON, when lights are switched on in the room, LED should switch off immediately.

Experiment - Aim:

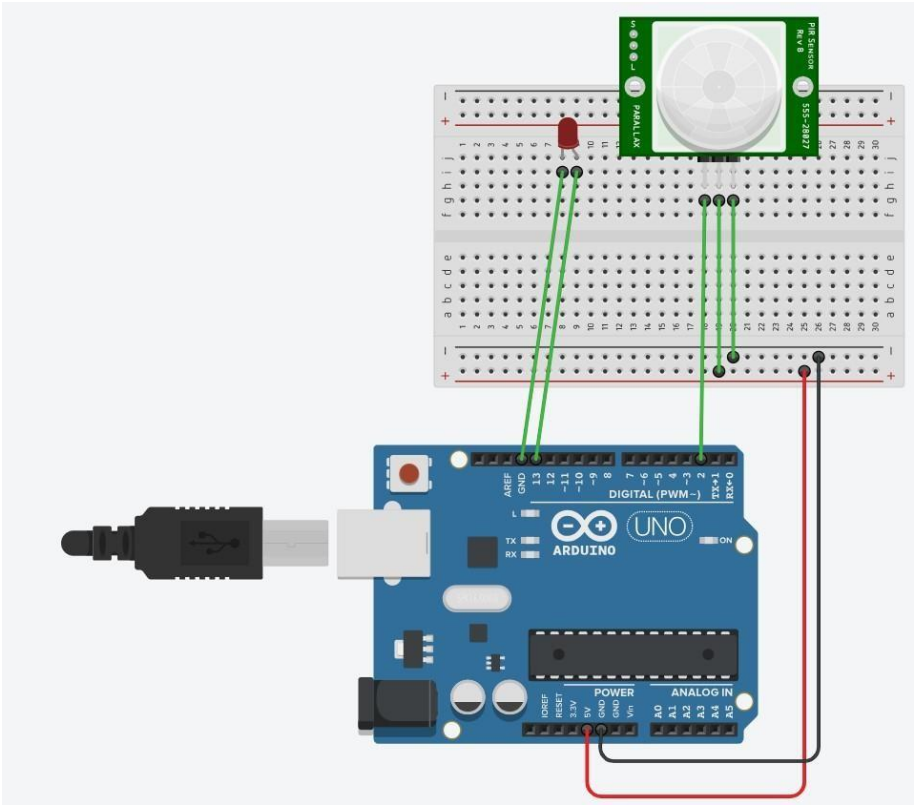
6 6

PIR with Arduino UNO

Hardware Components:

Arduino Uno, Breadboard, LED, PIR

Circuit Diagram:



Code:

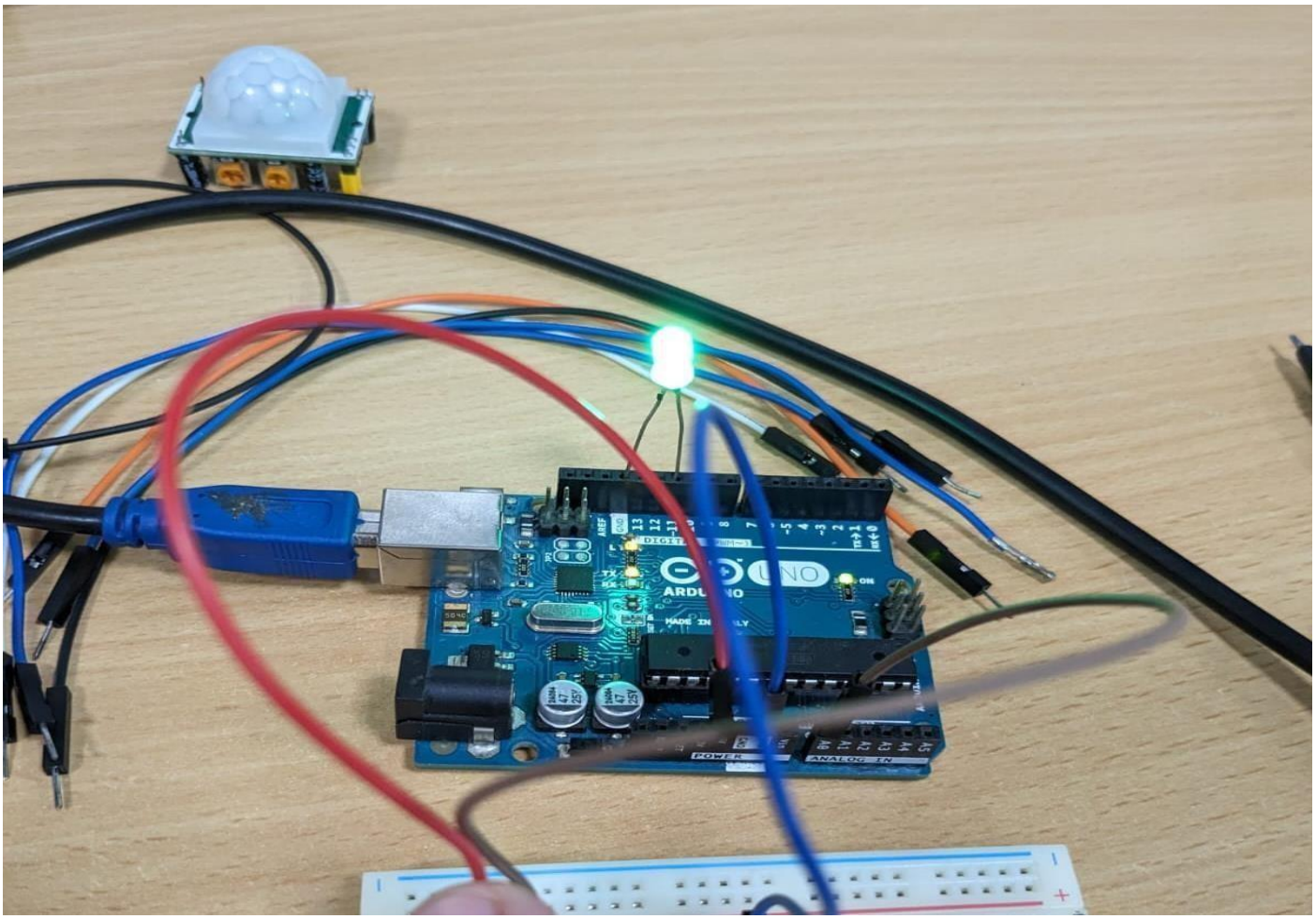
```
int sensorState = 0;

void setup()
{
        pinMode(2, INPUT);
        pinMode(13, OUTPUT);
        Serial.begin(9600);
}
```

```
void loop()
{  sensorState = digitalRead(2);   if
        (sensorState == HIGH)
        {  digitalWrite(13, HIGH);
                Serial.println("Sensor activated!");
        }  else
        {
                digitalWrite(13, LOW);
        }
        delay(10);
}
```

Observation:

On covering the PIR sensor, the LED bulb glows and when the sensor is exposed to light, the LED goes off.

Experiment - Aim:



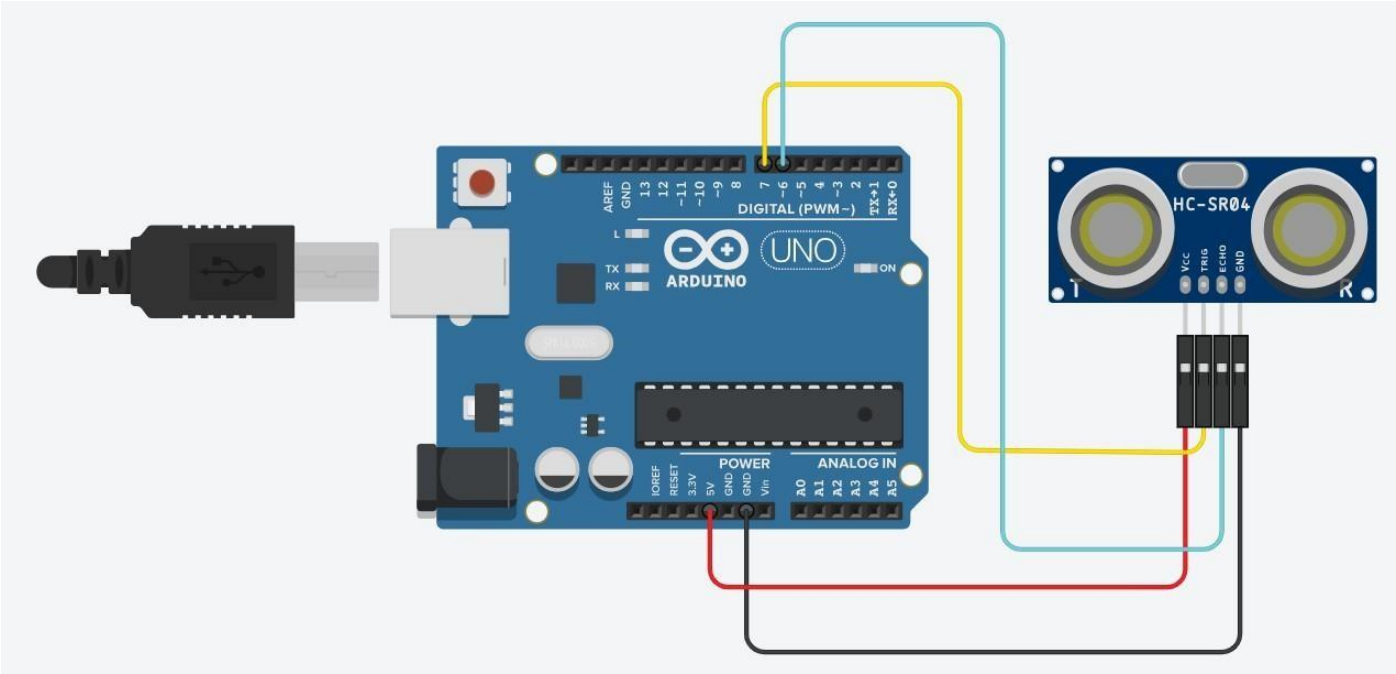
7 Experiment - 7

Aim:

Ultrasound with Arduino Uno

Hardware Components: Arduino Uno, Ultrasound sensor

Circuit Diagram:



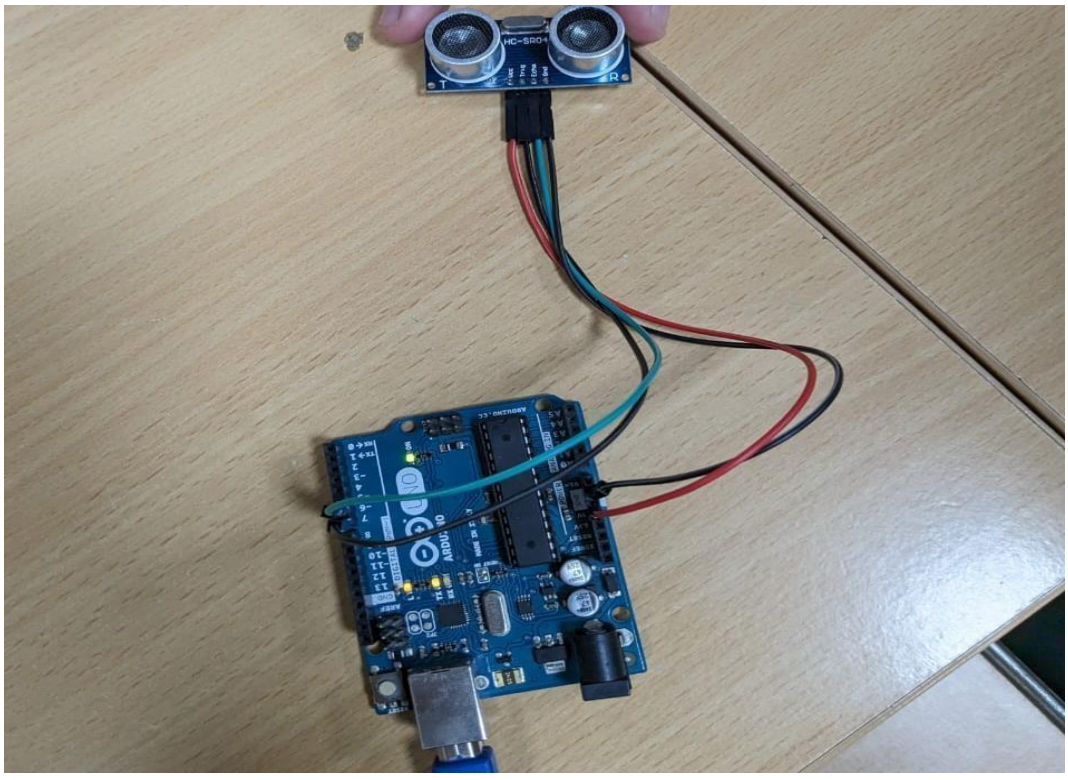
Code:

```
const int pingPin = 7;
const int echoPin=6;

void setup()
{
        Serial.begin(9600);  pinMode(pingPin,
        OUTPUT);
        pinMode(echoPin, INPUT);
}
```

```
void loop()
{  long duration, inches, cm;  digitalWrite(pingPin,
        LOW);   delayMicroseconds(2);
        digitalWrite(pingPin, HIGH);
        delayMicroseconds(10);
        digitalWrite(pingPin, LOW);  duration =
        pulseIn(echoPin, HIGH);  cm =
        microsecondsToCentimeters(
        duration);   Serial.print(cm);
        Serial.println("cm");
}
long microsecondsToCentimeters(long microseconds)
{  return microseconds / 29 / 2;    }
```

Observation:

The ultrasound sensor measures the distance of objects when we move at different positions.



8 Experiment - 8

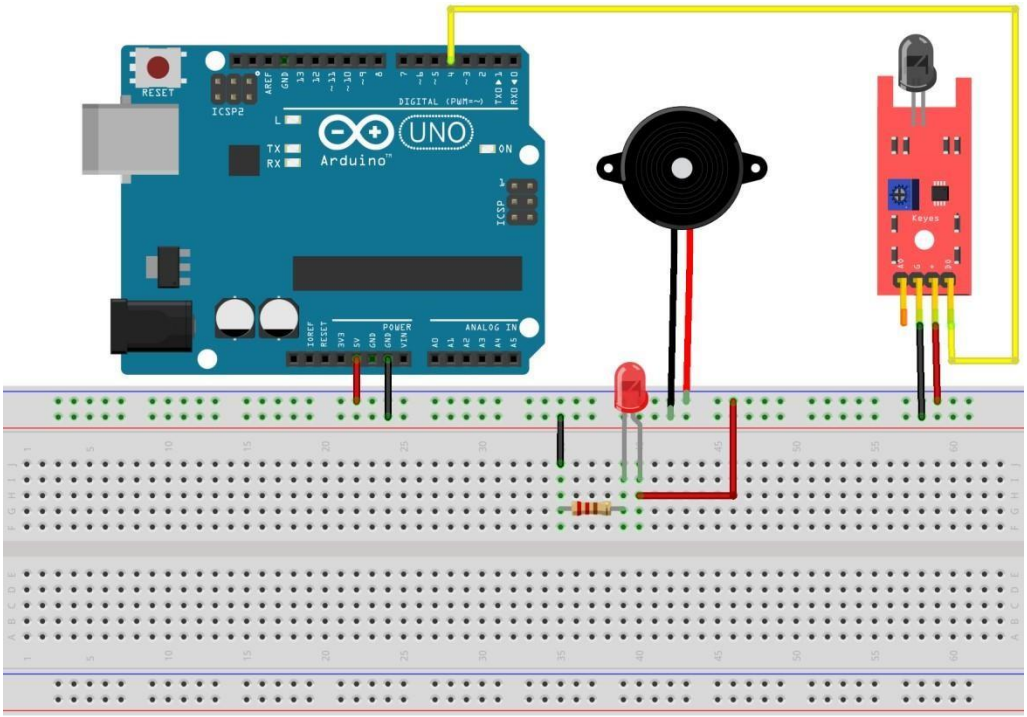
Aim:

Design and implement Fire alarm system using flame sensor and buzzer.

Hardware Components:

Arduino Uno, Flame sensor, Breadboard, LED, buzzer

Circuit Diagram:



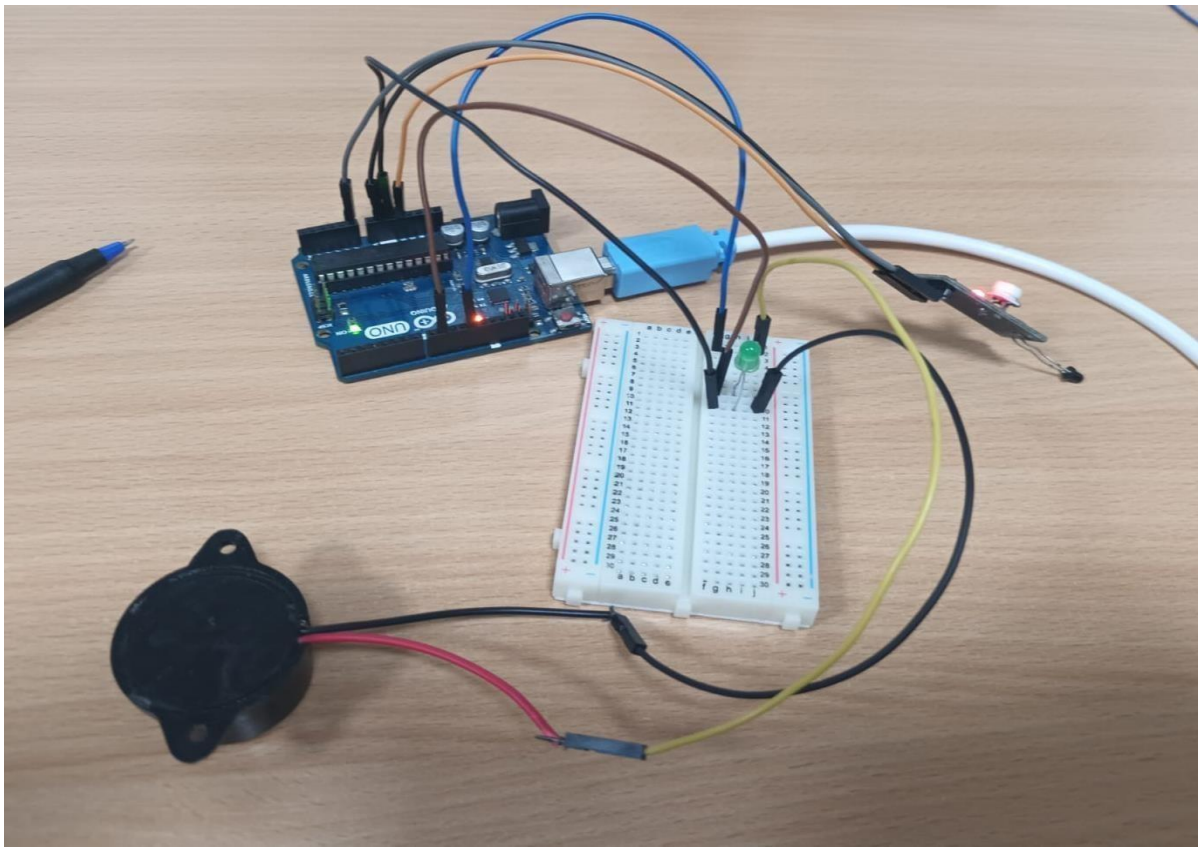
Code:

```
int sensorPin = A0; int
sensorValue = 0; int led
= 9;
int buzzer = 12;

void setup()
{
```

```
        pinMode(led, OUTPUT);
        pinMode(buzzer,OUTPUT); Serial.begin(9600);
}

void loop()
{ sensorValue = analogRead(sensorPin);  Serial.println(sensorValue);
        if
        (sensorValue < 100)
        {
                Serial.println("Fire Detected");
                Serial.println("LED on");
                digitalWrite(led,HIGH);
                digitalWrite(buzzer,HIGH); delay(1000);
        }
        digitalWrite(led,LOW);
        digitalWrite(buzzer,LOW); delay(sensorValue);
}
```

Observation:

Flame sensor detects radiation (heat) and detects the distance from the flame.



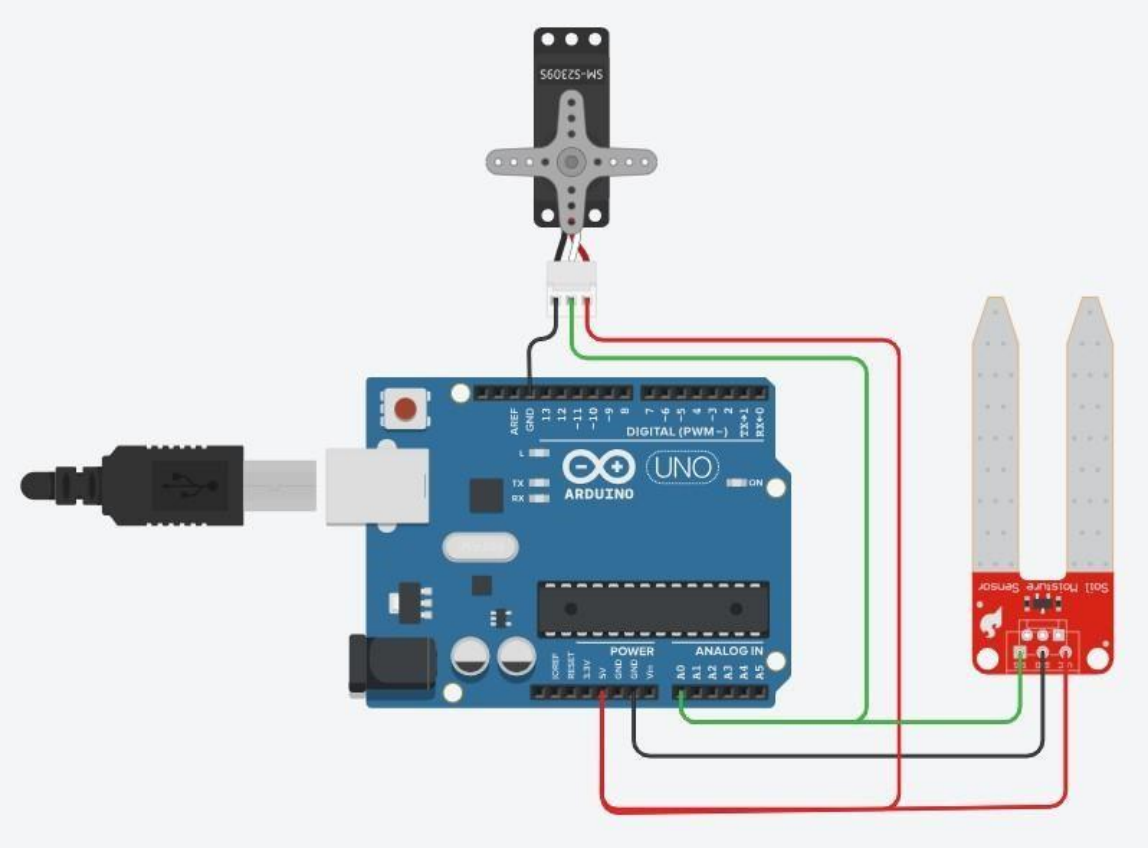
9 Experiment - 9 Aim:

Design and implement smart irrigation system using Soil Moisture sensor and Servo Motor.

Hardware Components:

Arduino Uno, Moisture sensor, Breadboard, Servo Motor

Circuit Diagram:



Code:

#include <Servo.h>

```
Servo myservo; int
pos = 0; int sensorPin
= A0; int sensorValue
= 0; void setup()
{ myservo.attach(9);
Serial.begin(9600);  }

void loop()
{ sensorValue = analogRead(sensorPin);
        Serial.println(sensorValue); if(sensorValue>500)
        {
                for (pos = 0; pos <= 180; pos += 1)
                { myservo.write(pos);
                        delay(15);
                }
                for (pos = 180; pos >= 0; pos -= 1)
                {
 myservo.write(pos);
                        delay(15);
                }
        }  delay
        (1000);
}
```

1.9.1 Observation:

When the moisture sensor is placed in contact with water, the servo motor wings rotate and when it is placed in dry environment, no movement is seen.

10 Experiment – 10

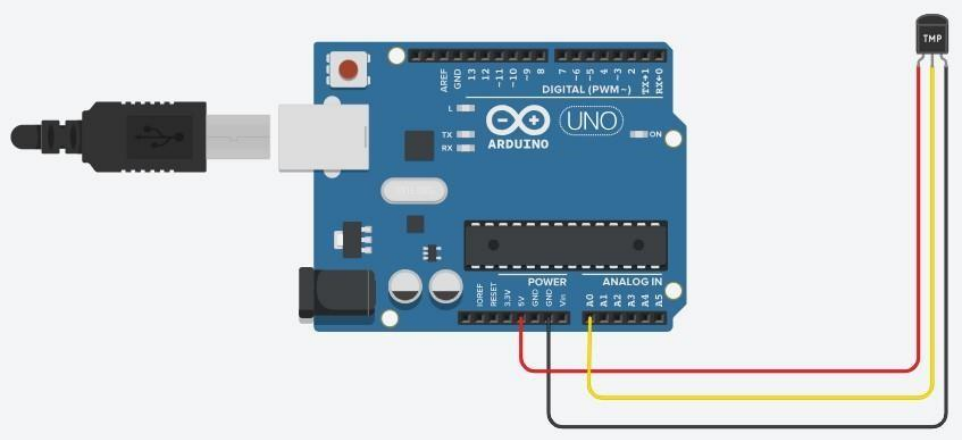
Aim:

Read the temperature and print it in serial port.

Hardware Components:

Arduino Uno, Temperature Sensor (LM35), breadboard

Circuit Diagram:

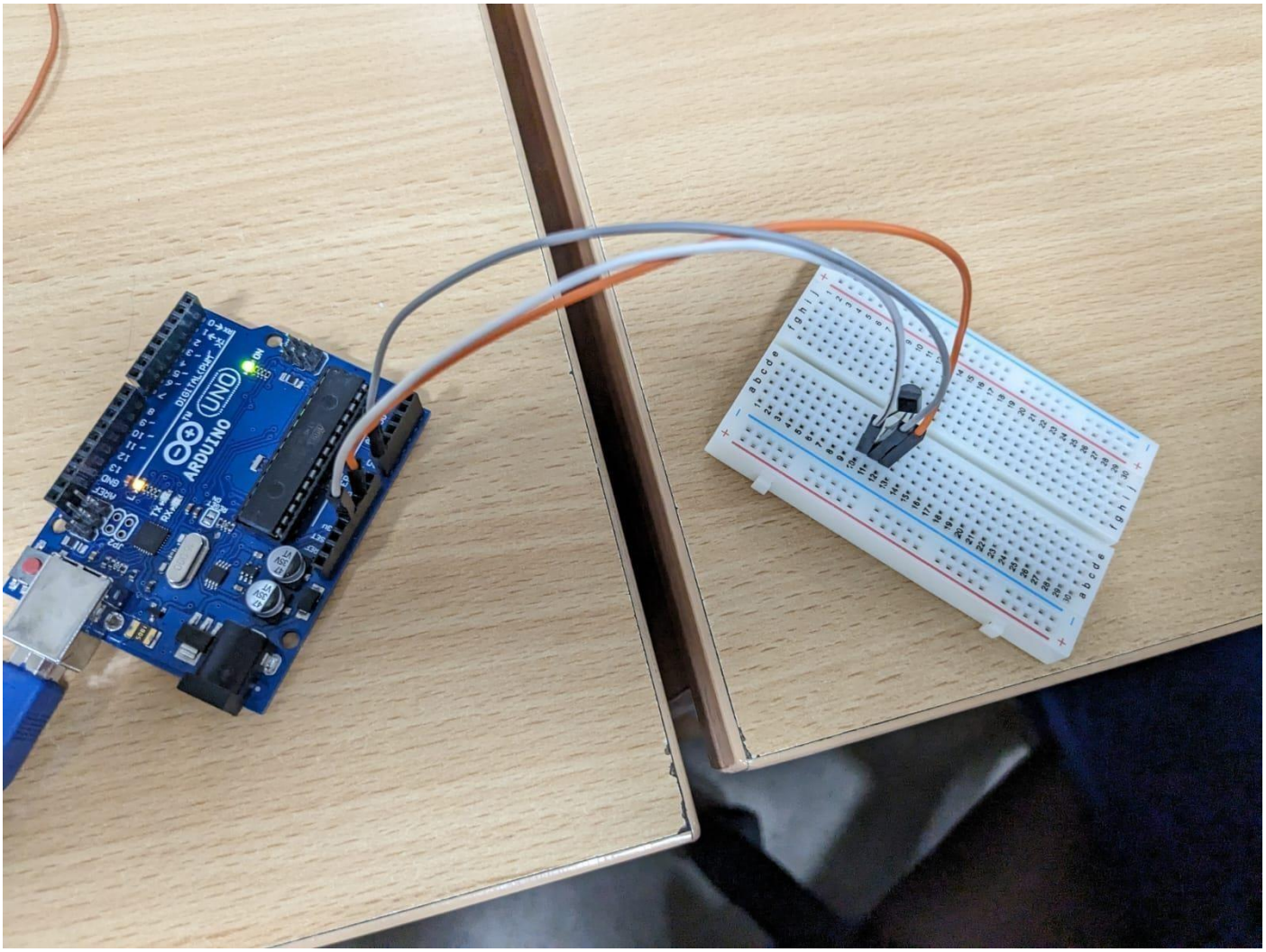


Code:

```
int sensorPin = A0;

void setup()
{
        Serial.begin(9600);
}


void loop()
{ int reading = analogRead(sensorPin);   float
        voltage = reading * 5.0 / 1024;
        Serial.print(voltage);  Serial.println(" volts");
        float temperatureC = (voltage - 0.5) * 100 ;
        Serial.print(temperatureC); Serial.println(" degress C"); float temperatureF
        = (temperatureC * 9 / 5) + 32;
        Serial.print(temperatureF); Serial.println(" degress F");
        delay(1000);
}
```

Observation:

The LM35 is a low power, low-cost, high precision temperature sensor. This IC provides a voltage output that is linearly proportional to the change in temperature.


ARDUINO
UNO
DIGITAL (PWM~)

11 Experiment - 11

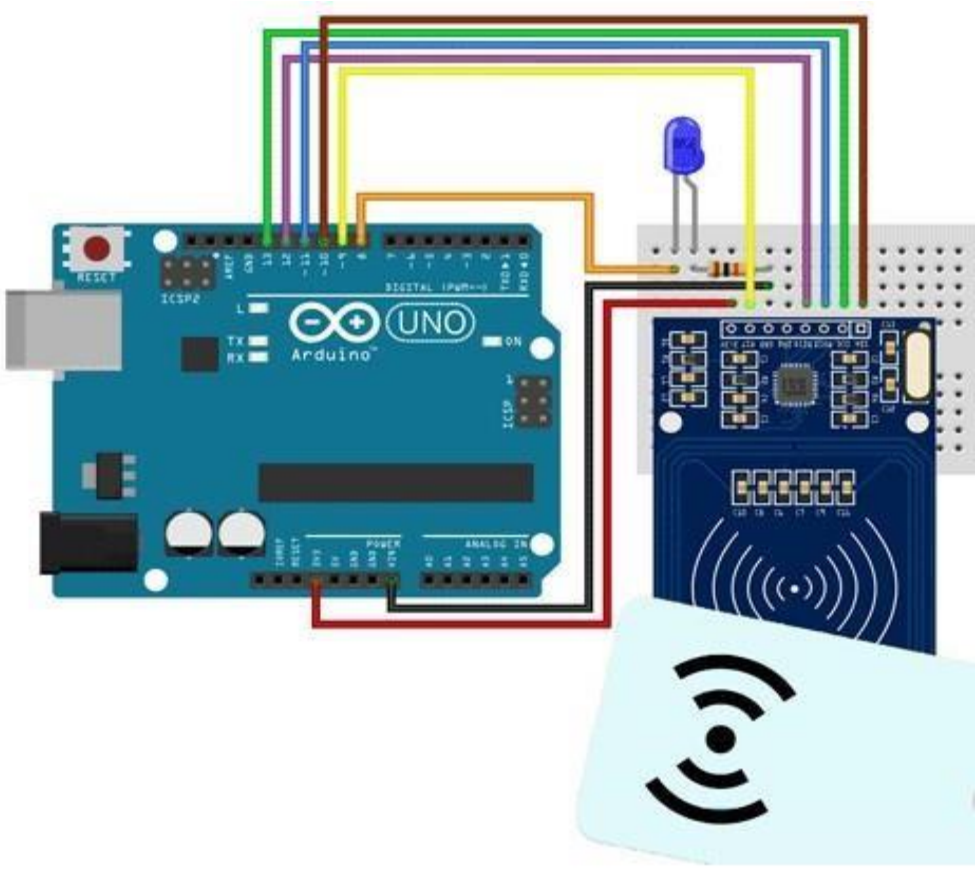
Aim:

Design and implement an access control system using RFID.

Hardware Components:

Arduino Uno, Breadboard, RFID reader, RFID Tag, LED, resistor

Circuit Diagram:



Initial Code:

```
#include<SoftwareSerial.h>

SoftwareSerial mySerial(9,
10);   int count = 0; char
input[12];
boolean flag = 0;

void setup()
```

```
{
        Serial.begin(9600); mySerial.begin(9600);   }

void loop()
{  if(mySerial.available())
        {  count = 0;   while(mySerial.available() &&
                count < 12)
                {  input[count] =mySerial.read();
                        count++;
                delay(5);
                }
        Serial.print(input); // Print RFID tag number  }
}
```

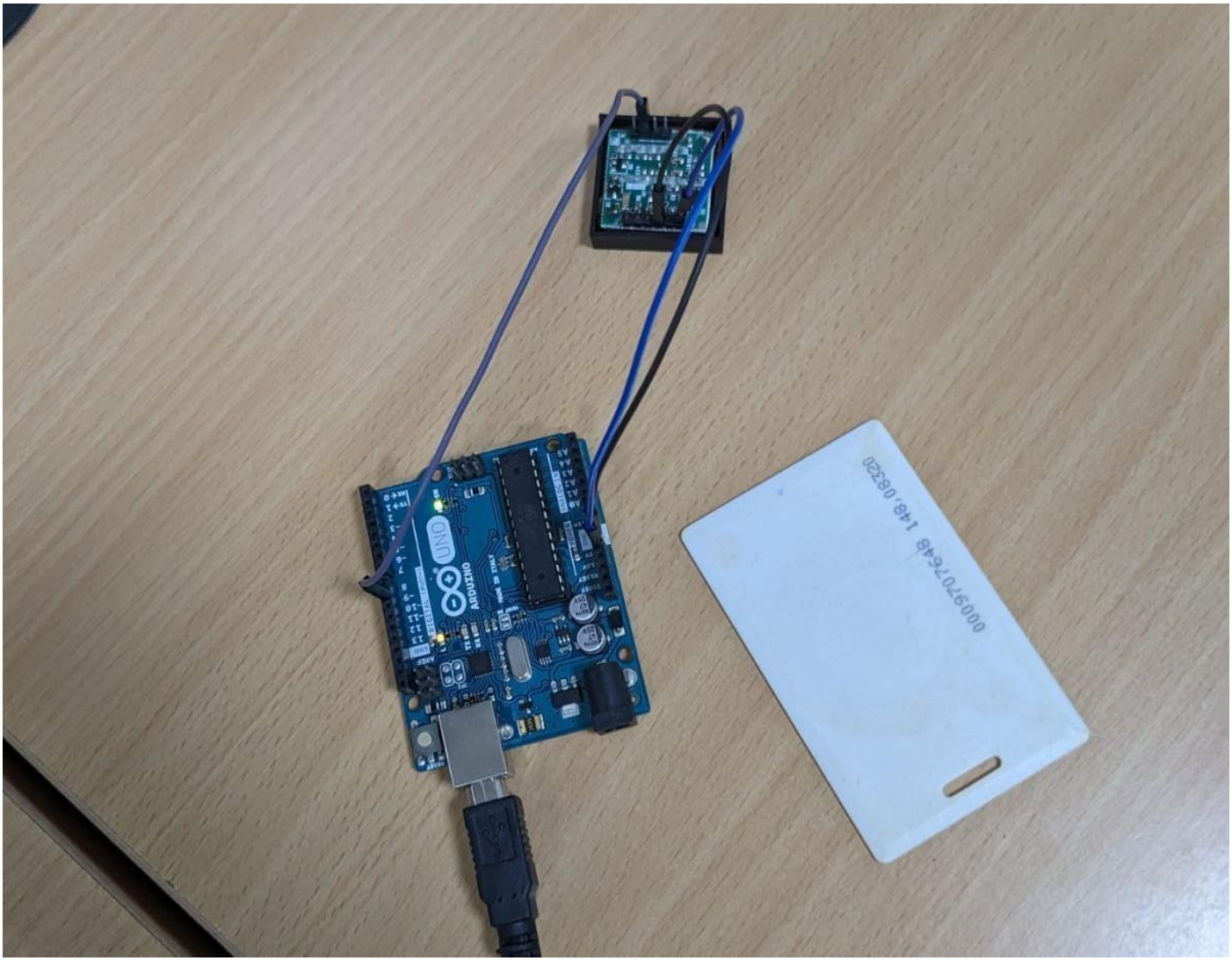
Code:

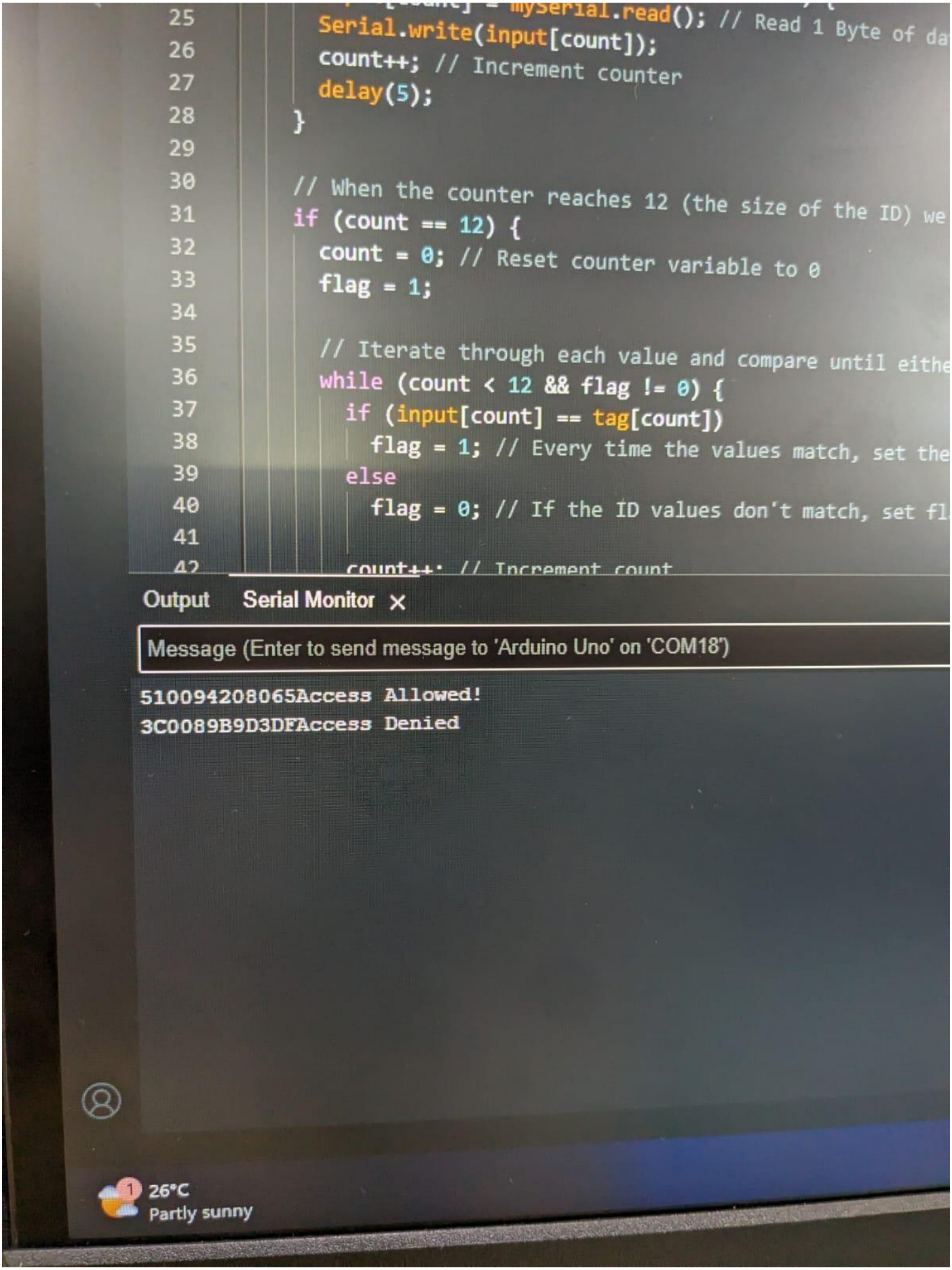
```
#include<SoftwareSerial.h>

SoftwareSerial mySerial(9, 10);
#define LEDPIN 12  char tag[]
="0900970C8517";          char
input[12];  int count = 0;
boolean flag = 0;

void setup()
{
        Serial.begin(9600);  mySerial.begin(9600);
        pinMode(LEDPIN,OUTPUT);
}  void loop()
{  if(mySerial.available())
        {  count = 0;  while(mySerial.available() &&
                count < 12)
                {  input[count] =
                        mySerial.read();
                        Serial.write(input[count]);
                        count++;  delay(5);
                }
                if(
```

```
co
un
t
==
12
)
{
        count =0;
        flag = 1;
        while(count<12 && flag !=0)
        {  if(input[count]==tag[count])
                flag = 1; else   flag= 0;
                count++;
        }
}

if(flag == 1)
{
        Serial.println("Access Allowed!");  digitalWrite(LEDPIN,HIGH);
        delay
        (2000);   digitalWrite
        (LEDPIN,LOW);
}  else
{
        Serial.println("Access Denied"); digitalWrite(LEDPIN,LOW);   delay(2000);
}
for(count=0; count<12; count++)
{  input[count]= 'F';
}   count = 0; // Reset counter
variable  }   }
```

Observation:

The above code will read the code present on the RFID tag tapped. If the code matches with the previously known tag, it will grant the access (LED Glows) otherwise access will be denied.


UNO
ARDUINO
0009707648 148,08220


Serial.write(input[count]);
count++; // Increment counter
delay(5);
}
// When the counter reaches 12 (the size of the ID) we
if (count == 12) {
count = 0; // Reset counter variable to 0
flag = 1;
// Iterate through each value and compare until eithe
while (count < 12 && flag != 0) {
if (input[count] == tag[count])
flag = 1; // Every time the values match, set the
else
flag = 0; // If the ID values don't match, set fl
Output
Serial Monitor
Message (Enter to send message to 'Arduino Uno' on 'COM18')
510094208065Access Allowed!
3C0089B9D3DFAccess Denied
26°C
Partly sunny

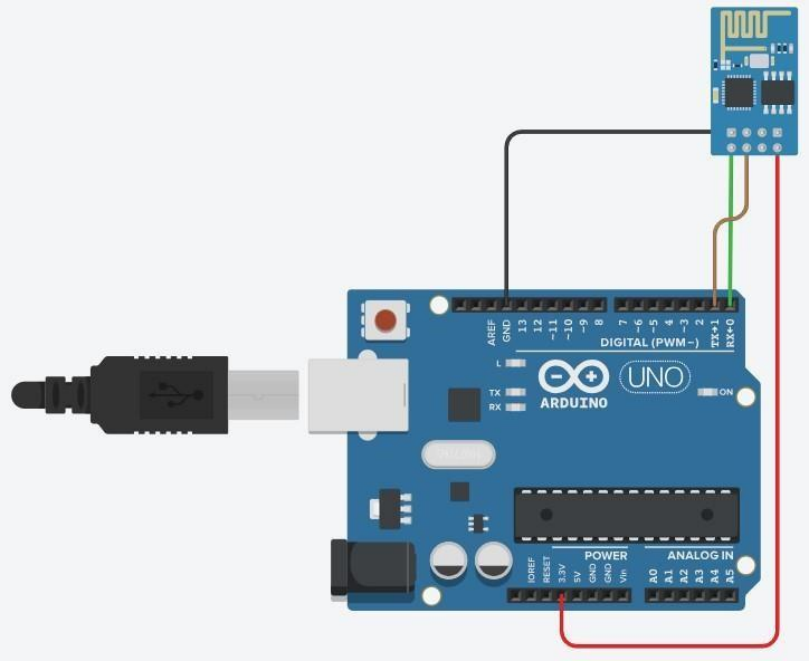
12 Experiment - 12

Aim:

Working with Arduino Bluetooth Module.

Hardware Components:
Arduino Uno, Bluetooth module (HC-05)

Circuit Diagram:



Code:
(a) HC - 05 at Command Prompt:
(For this program to work, HC-05 must be in command mode)

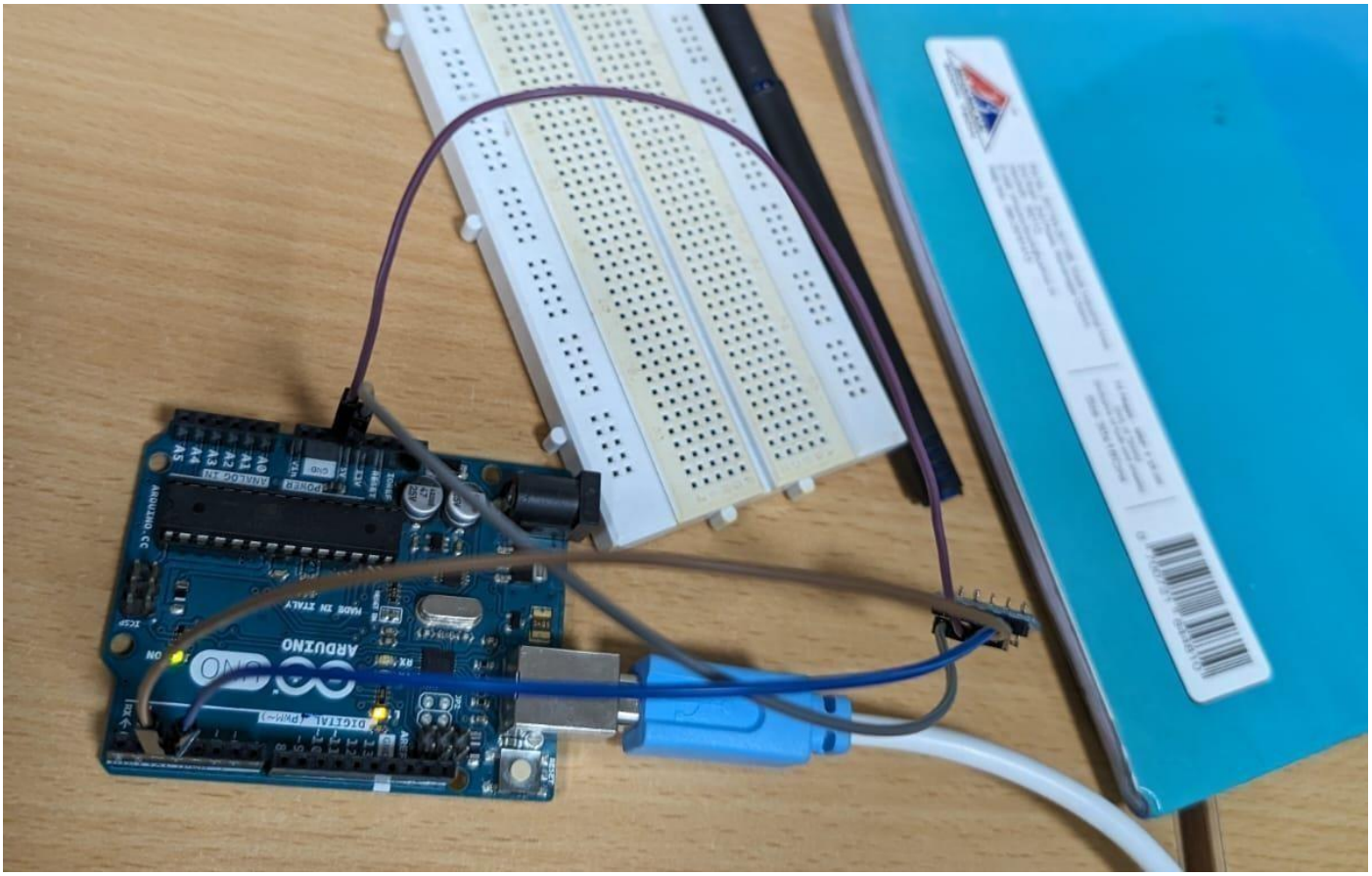
```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(10, 11);

void setup()
{
        Serial.begin(9600);
        Serial.println("Enter AT commands:");
         BTSerial.begin(38400); // HC-05 default speed in AT command more
}  void loop()
{  if (BTSerial.available())
```

```
        Serial.write(BTSerial.read());  if
    (Serial.available())
        BTSerial.write(Serial.read());  }
```
(b)

HC - 05 Controlled by Mobile:

(For this code to work, HC-05 must be in DATA mode and Arduino Bluetooth App)

```
#define ledPin 13
int state = 0;

void setup()
{
    pinMode(ledPin, OUTPUT);
    digitalWrite(ledPin, LOW);  Serial.begin(38400);
}

void loop()
{  if(Serial.available() > 0)
    {  state = Serial.read();
    }

    if (state == '0')
    {
        digitalWrite(ledPin, LOW); Serial.println("LED:
        OFF");   state = 0;
    }   else if (state ==
    '1')
    {
    digitalWrite(ledPin, HIGH);
    Serial.println("LED: ON");;   state = 0;
    }  }
```


ARDUINO
UNO

13 Experiment - 13

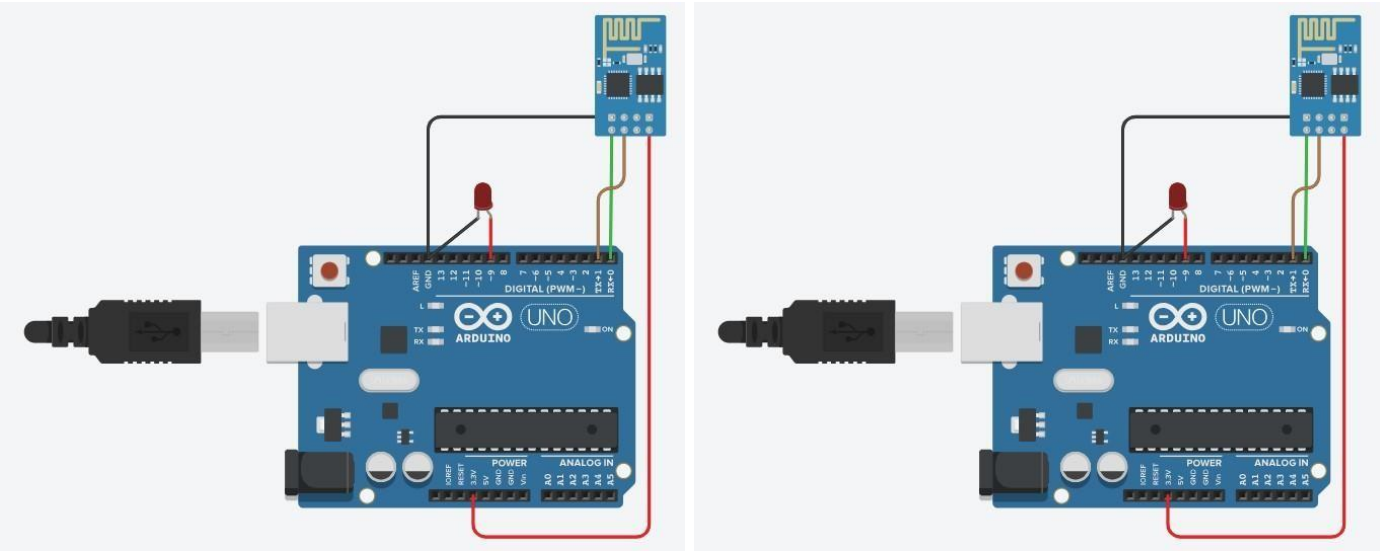
Aim:

Design and implement a system to realize Bluetooth Master/Slave scenario.

Hardware Components:

Arduino Uno - 2, Bluetooth module (HC - 05) - 2

Circuit Diagram:



Master Device **Slave Device**

Code:

<u>Slave Mode:</u>

Commands to configure bluetooth module (HC - 05) as a slave device:

AT+ORGL

AT+RMAAD

AT+ROLE=0

AT+ADDR

<u>Slave Code:</u>

```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(10, 11);
```

```
void setup()
{
        Serial.begin(9600);
        BTSerial.begin(38400);
}   void
loop()
{  if(Serial.available())
        {
                String message = Serial.readString();
                Serial.println (message);
                BTSerial.write(message.c_str());  }
}
```

Master Mode:

Commands to configure bluetooth module (HC - 05) as a slave device:

AT+RMAAD
AT+ADCN
AT+ROLE=1
AT+CMODE=0
AT+PSWD=1234
AT+BIND=<address>
AT+LINK=<address>
AT+INIT

Master Code:

```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(10, 11); // RX | TX

#define ledPin 9  String message;
int potValue = 0;

void setup()
{
        pinMode(ledPin, OUTPUT);
        digitalWrite(ledPin, LOW);  Serial.begin(9600);   BTSerial.begin(38400);
```

```
}

void loop()
{  if(BTSerial.available() > 0)
        {  message = BTSerial.readString();  if(message.indexOf("SWITCH
              ON")>=0)
              {
                    digitalWrite(ledPin, HIGH);
              }
              else if(message.indexOf("SWITCH OFF")>=0)
              {
                    digitalWrite(ledPin, LOW);
              }  else
              {
                    Serial.println("Noting to do");
              }
              delay(100);
        }
        delay(10);
}
```

1.13.1 Observation:

The provided Arduino code establishes a Bluetooth communication link between a Master and Slave device using HC-05 modules. The Master controls an LED on the Slave through specific commands, showcasing a basic wireless interaction.

14 Experiment - 14

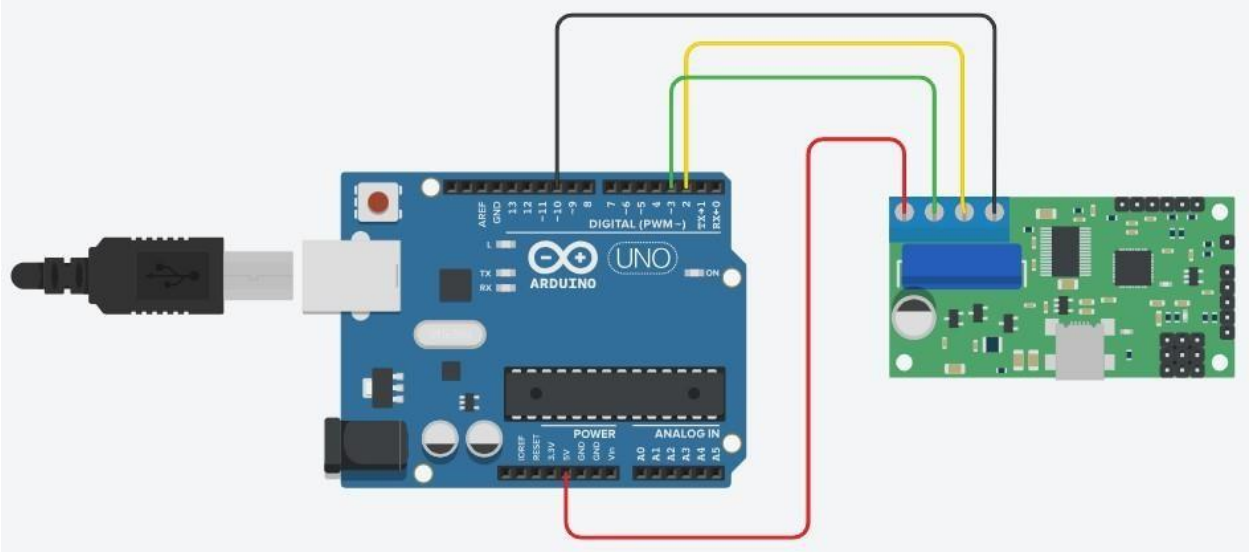
Aim:

Call using a Arduino and GSM module to a specified mobile number inside the program.

Hardware Components:

Arduino Uno, GSM module

Circuit Diagram:



Code:

```
#include <SoftwareSerial.h>

SoftwareSerial cell(2,3);

void setup()
{ cell.begin(9600);  delay(500);
        Serial.begin(9600);
        Serial.println("CALLING.....");  cell.println("ATD+919538433364;");  delay(20000);
}

void loop() { }
```

Observation:

When we connect GSM module to the Arduino Uno, the mobile number that is mentioned in the code is called using GSM module and we receive the call on that specified number.

15 Experiment - 15

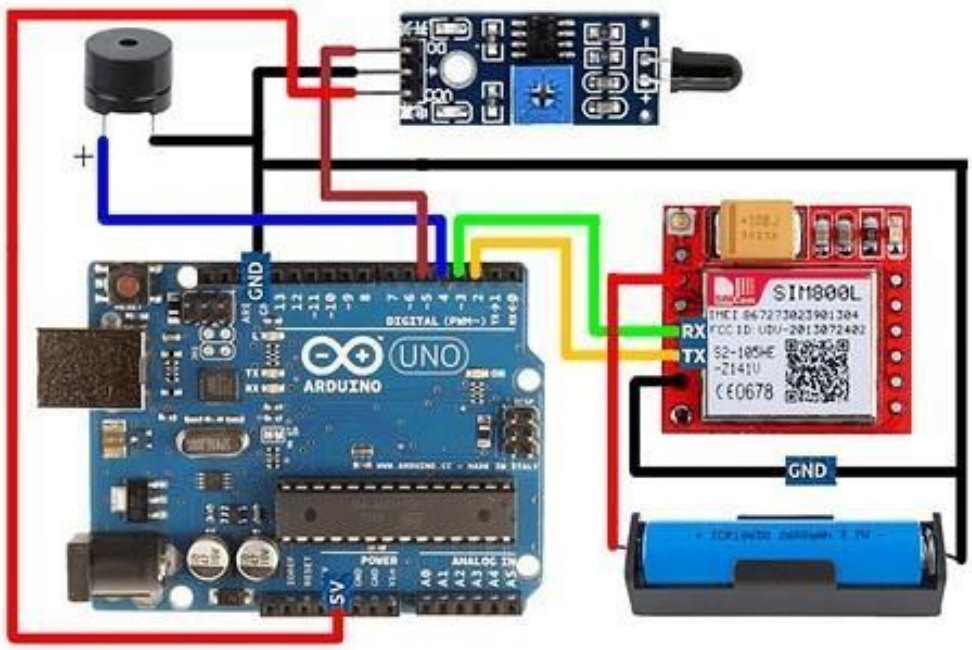
Aim:

Call using a Arduino and GSM module to a specified mobile number inside the program when a flame sensor detects fire.

Hardware Components:

Arduino Uno, GSM module, Flame sensor

Circuit Diagram:



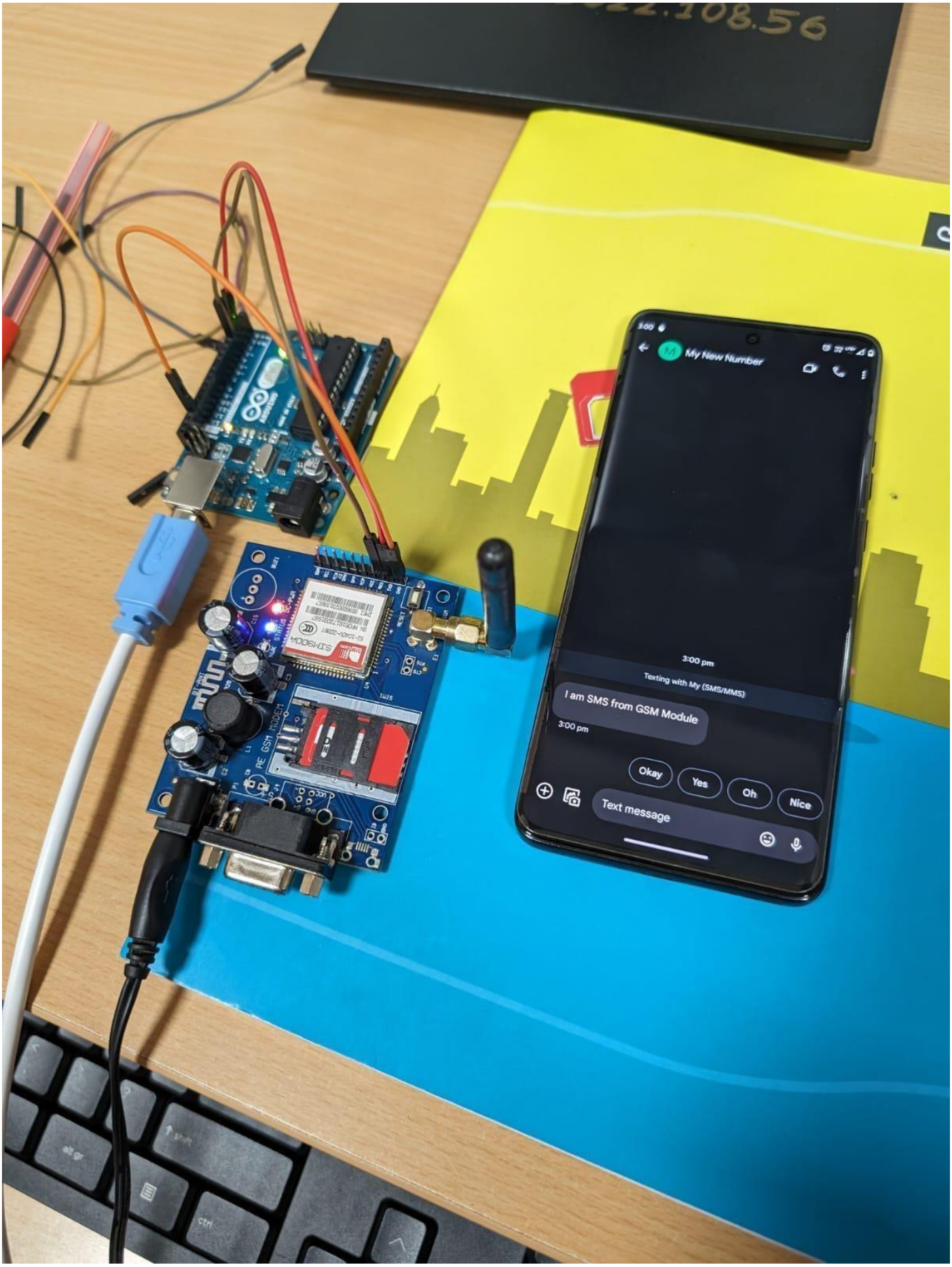
Code:

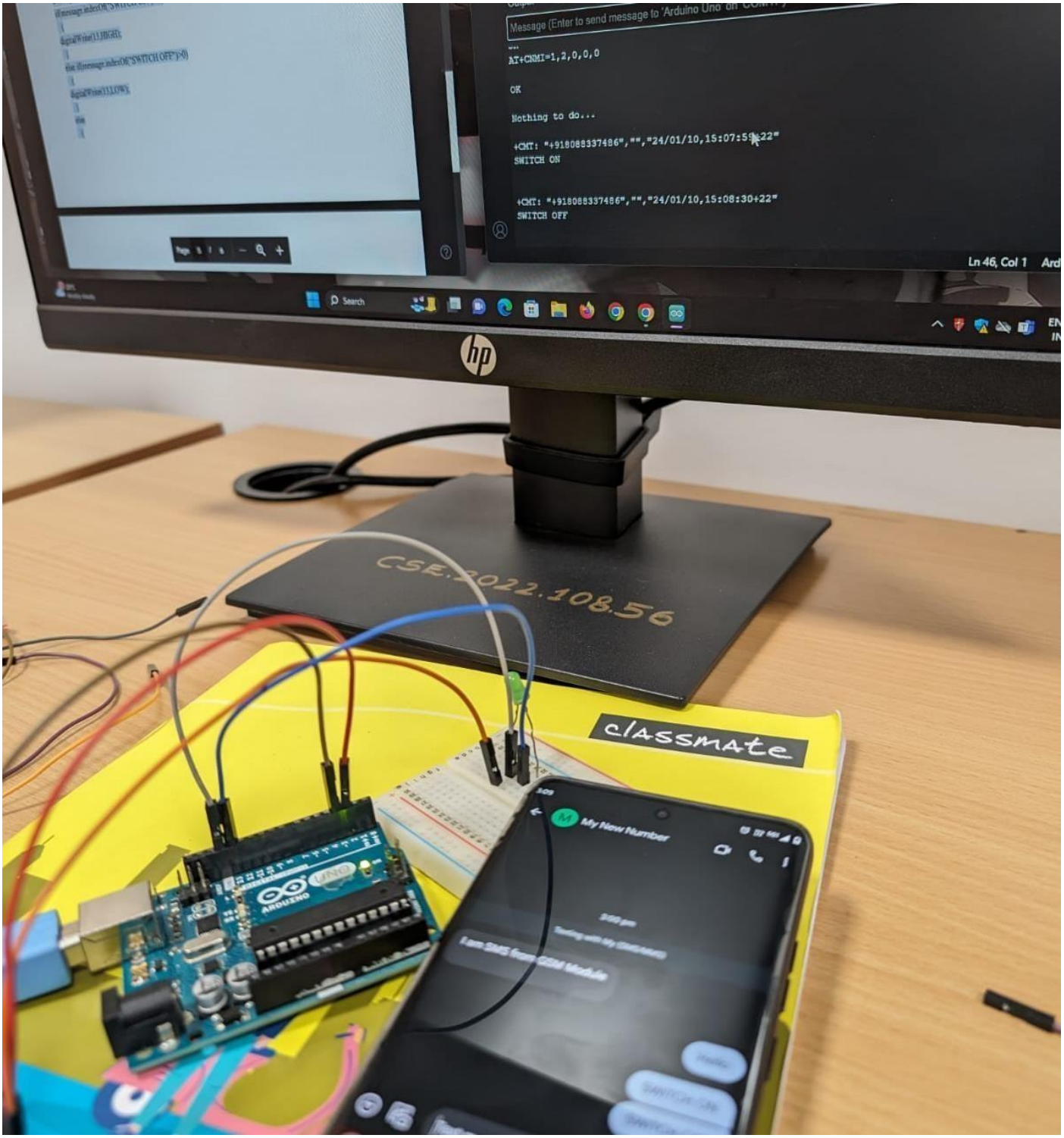
```
#include <SoftwareSerial.h>

SoftwareSerialcell(2,3);

void setup()
{  cell.begin(9600);  delay(500);
        Serial.begin(9600);
}

void loop()
{
        intval=analogRead(A0);
        Serial.println(val);  delay(1000);
        if
        (val<50)
        {
                Serial.println("CALLING.....");  cell.println("ATD+919742980606;");
                delay(10000);
                cell.println("ATH");
        }
```

}

Observation:

When the flame sensor detects fire, a call is placed to a phone number using GSM module.


My New Number
3:00 pm
Texting with My (SMS/MMS)
I am SMS from GSM Module
3:00 pm
Okay
Yes
Oh
Nice
Text message


Message (Enter to send message to 'Arduino Uno' on
AT+CNMI=1,2,0,0,0
OK
Nothing to do...
+CMT: "+918088337486","","24/01/10,15:07:59+22"
SWITCH ON
+CMT: "+918088337486","","24/01/10,15:08:30+22"
SWITCH OFF
Ln 46, Col 1
CSE-2022.108.56
classmate
My New Number

16 Experiment - 16

Aim:

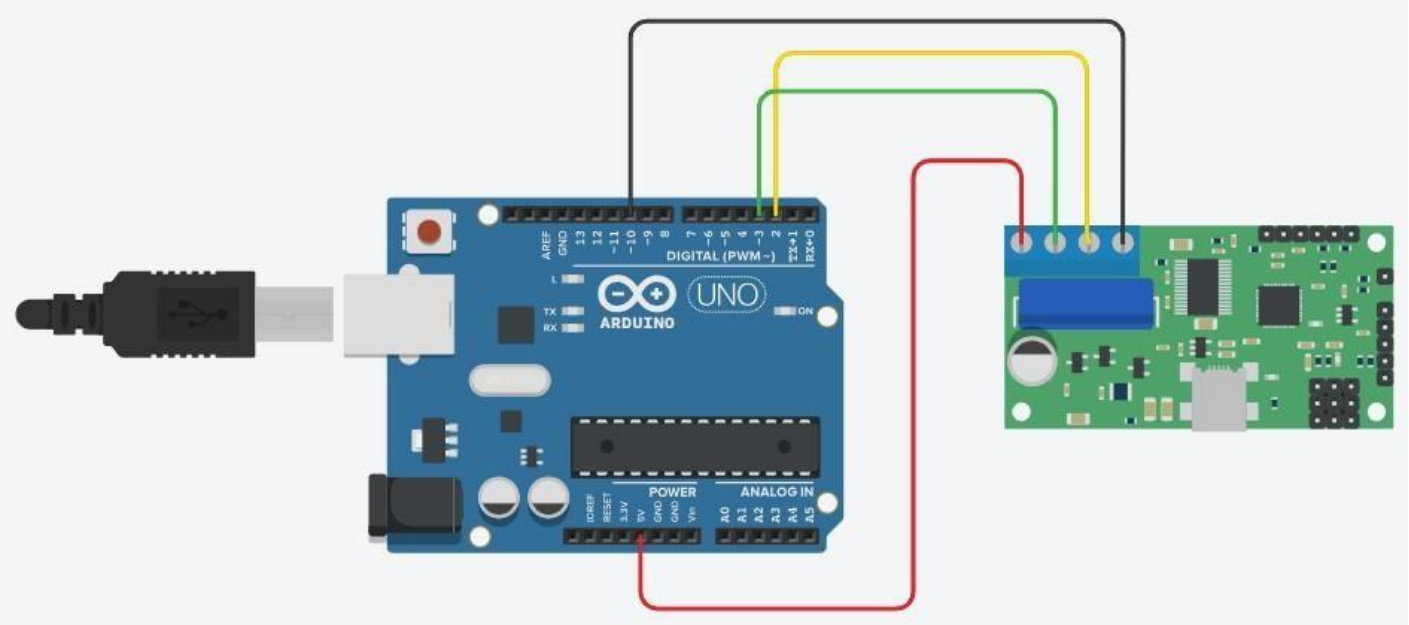
a) Send a SMS using Arduino and GSM module to a specific mobile number inside the program.

b) Receive SMS using Arduino and GSM module to the SIM card loaded in the GSM Module.

Hardware Components:

Arduino Uno, GSM Module

Circuit Diagram:



Code:

```
#include <SoftwareSerial.h>

SoftwareSerial mySerial(2, 3);

void setup()
{
        mySerial.begin(9600);
        Serial.begin(9600); delay(100);
}
```

```
void loop()  {  if
        (Serial.available()>0)
        switch(Serial.read())   {  case 's':
        SendMessage();   break;
                case 'r': RecieveMessage();  break;
        }

        if (mySerial.available()>0)  Serial.write(mySerial.read());   }

voidSendMessage()
{
        mySerial.println("AT+CMGF=1");   delay(1000);
        mySerial.println("AT+CMGS=\"+919742980606\"\r");  delay(1000);
        mySerial.println("I  am  SMS  from  GSM  Module");    delay(100);
        mySerial.println((char)26); delay(1000);
}

voidRecieveMessage()
{
        mySerial.println("AT+CNMI=2,2,0,0,0"); delay(1000);
}
```

Observation:
The Arduino program utilizes a GSM module and SoftwareSerial library to send an SMS to a specific number upon receiving the 's' command. It also enables SMS reception ('r' command) by configuring the GSM module. This basic setup facilitates bidirectional communication between the Arduino and GSM module for SMS functionality.

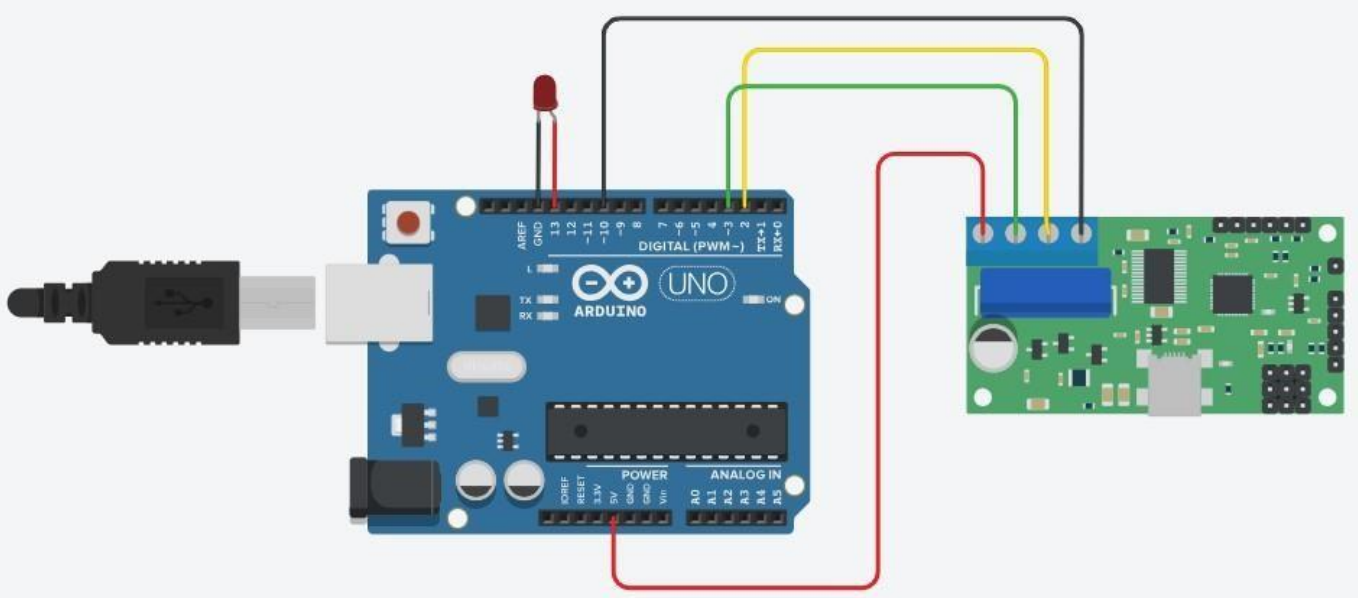
17 Experiment - 17

Aim:

Use received message through Arduino and GSM module to control switching ON/OFF the LED.

Hardware Components: Arduino
Uno, GSM Module, LED

Circuit Diagram:



Code:

```
#include <SoftwareSerial.h>

SoftwareSerial cell(2,3);

voidreadfn()
{  if (cell.available())
        {  while (cell.available())
                {
                        Serial.write(cell.read());  }   }
}

void setup()
{
        pinMode(13,OUTPUT);
        Serial.begin(9600);  cell.begin(9600);
        cell.println("AT");  delay(1000);
        readfn();
        cell.println("AT+CNMI=1,2,0,0,0");
}

void loop()
```

```
{  if(cell.available())
        {
                String message =cell.readString();
                Serial.println(message);

                if(message.indexOf("SWITCH ON")>0)
                {
                        digitalWrite(13,HIGH);
                }

                else if(message.indexOf("SWITCH OFF")>0)
                {
                        digitalWrite(13,LOW);
                }

                else
                {
                        Serial.println ("Nothing to do...");  }
        }
}
```

Observation:

When we send a message “SWITCH ON” through GSM Module, the LED Turns ON. When we send a message “SWITCH OFF” through GSM Module, the LED Turns OFF.