

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT

on

INTERNET OF THINGS LAB

Submitted by

PRASANNA KUMAR R (1BM21CS138)

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

Oct-2023 to Feb-2024

**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 **PRASANNA KUMAR R (1BM21CS138)**, who is a 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. The Lab report has been approved as it satisfies the academic requirements in respect of a **Internet of things lab (21CS5PCIOT)**work prescribed for the said degree.

Dr.Seema patil
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

TABLE OF CONTENTS

Sl no.	Date	Name of the Experiment	page no.
1	23/11/2023	LED Blinking	3-4
2	23/11/2023	LED ON/OFF Using pushbutton	5-6
3	23/11/2023	LED fading using potentiometer	7-8
4	23/11/2023	LED fading without using potentiometer	9-10
5	07/12/2023	Simulating a night light using LDR	11-12
6	07/12/2023	Simulating a night light using PIR	13-14
7	07/12/2023	Simulating ultrasound with Arduino UNO and ultrasonic sensor	15-17
8	07/12/2023	Fire alarm Simulation	18-20
9	07/12/2023	Automatic irrigation controller simulation	21-23
10	21/12/2023	Read the code present on RFID tag	24-25
11	21/12/2023	Access control through RFID	26-29
12	21/12/2023	Temperature sensing	30-32
13	28/12/2023	Call to a specific number using GSM module	33-34
14	11/01/2024	Call a specific mobile number when flame sensor detects fire	35-36
15	11/01/2024	Send SMS using Arduino and GSM module	37-39
16	18/01/24	Control LED in bluetooth master device using bluetooth client	40-44

Program no: **01**

Program Title: **LED BLINK**

Date:23/11/2023

Aim: To control the LED using arduino (to turn ON/OFF LED)

Hardware/components Required

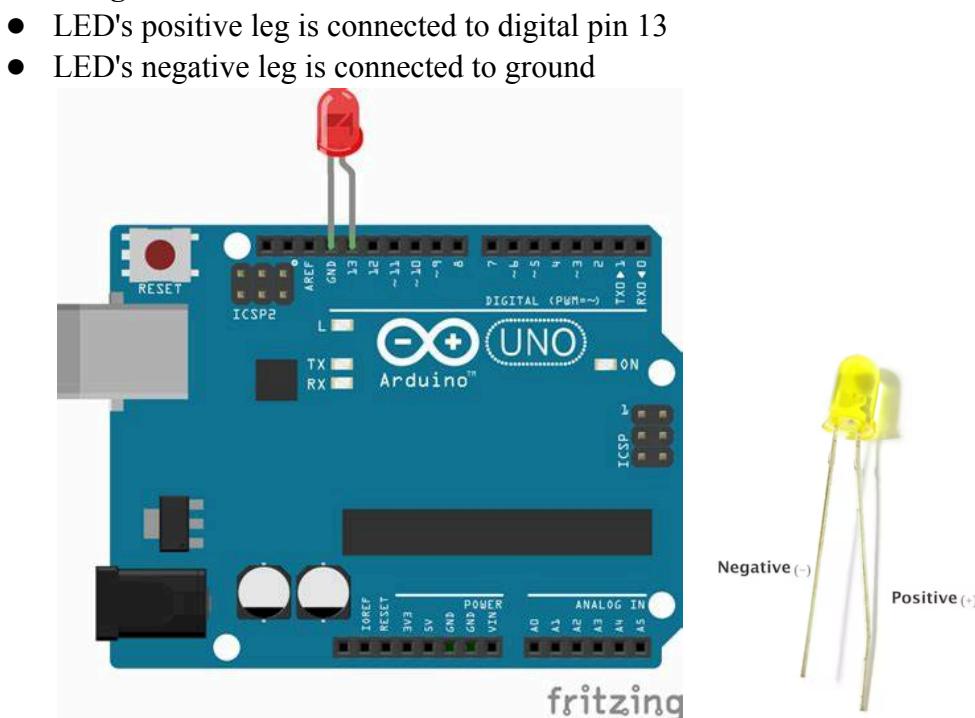
Arduino Uno board - 1

USB Cable - 1

LED - 1

Jumper wires

Circuit Diagram / Pin connection



Code:

```
void setup()
{
    // initialize digital pin 9 as an output.
    pinMode(13, OUTPUT);

    // the loop function runs over and over again forever
    void loop()
    {
        digitalWrite(13, HIGH);
        delay(1000);
        digitalWrite(13, LOW);
    }
}
```

Handwritten code pic:

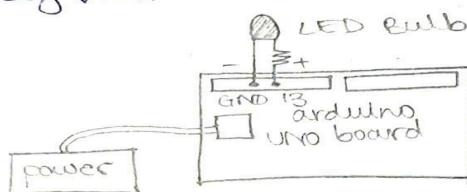
Experiment - 1

01/11/18

Aim: LED Blinking: turns on LED for 1 second, then off for 1 second, repeatedly.

Hardware components req: Arduino board, LED, Resistor.

Circuit diagram:



Observation:

LED switches ON/OFF Periodically

Digital Pin connections:

LED's +ve Leg is connected to digital Pin 13

LED's -ve Leg is connected to ground

Code:

Pin 13 has LED connected on most arduino boards

```
void setup() {  
    pinMode(led, OUTPUT);  
}  
  
void loop() {  
    digitalWrite(led, HIGH);  
    delay(1000);  
    digitalWrite(led, LOW);  
    delay(1000);  
}
```

Observation: LED switches ON/OFF periodically. Digital output visualization using Arduino Uno.

Program no: **02**

Program Title: **LED ON/OFF**

Date:23/11/2023

Aim: To turn an LED ON /OFF using a Pushbutton.

Hardware/components Required

Arduino Uno board - 1

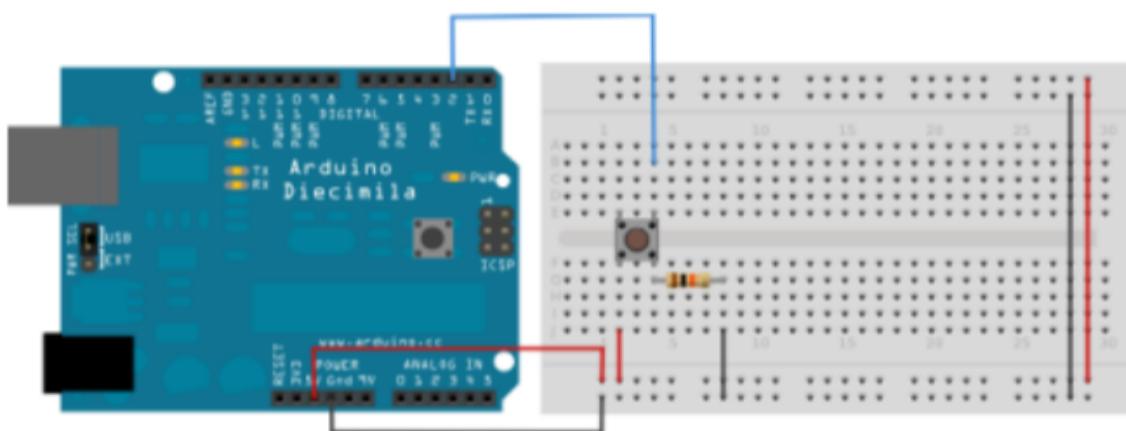
USB Cable - 1

LED - 1

Pushbutton

Jumper wires

Circuit Diagram / Pin connection



Code:

```
const int buttonPin=2;
const int ledPin=13;
int buttonState=0;
void setup()
{
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin,OUTPUT);
}

void loop()
{
  buttonState=digitalRead(buttonPin);
  if(buttonState==HIGH){
    digitalWrite(ledPin,HIGH);
  }else{
```

```

digitalWrite(ledPin,LOW);
}
}

```

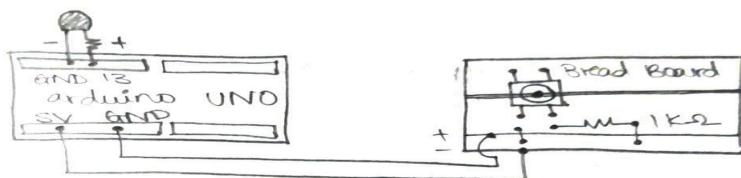
Handwritten code pic:

Experiment - 2

Aim: Turn ON LED ON IOFF using Push button

Hardware components req: Arduino board, LED, Push button.

Circuit Diagram:



```

const int buttonPin=2;
const int ledPin=13;
int buttonState=0;
void setup()
{
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
}
void loop()
{
  buttonState=digitalRead(buttonPin);
  if(buttonState==HIGH)
    digitalWrite(ledPin, HIGH);
  else
    digitalWrite(ledPin, LOW);
}

```

Observation:

To control the blinking of the LED using a push button. The LED blinks each time, when push button is pushed.

Observation: LED turns ON when push button is pressed and turns OFF when it is released.

Digital output visualization using Arduino Uno.

Program no: **03**

Program Title: **LED FADING**

Date:23/11/2023

Aim: To control the brightness of an LED using a Potentiometer.

Hardware/components Required

Arduino Uno board - 1

USB Cable - 1

LED - 1

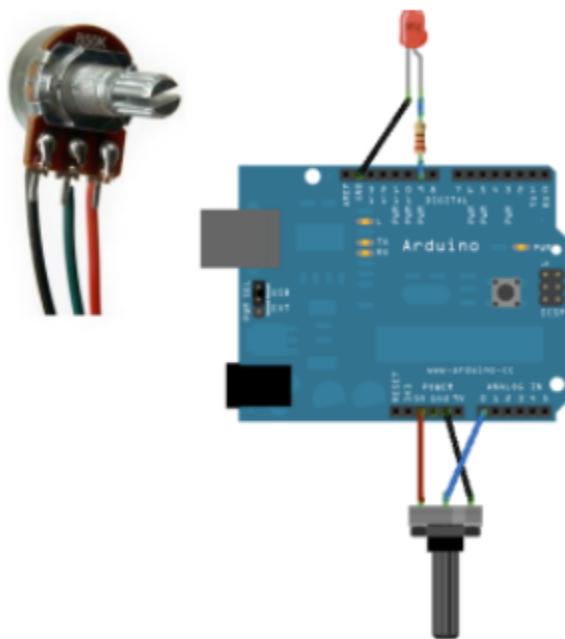
Potentiometer

Jumper wires

Circuit Diagram / Pin connection

LED positive to pin 9, LED negative to ground

Potentiometer: VCC - 5V , A0 -A0 , GND-GND



Code:

```
const int analogPin=A0;
const int analogOutPin=9;
int sensorValue=0;
int outputValue=0;
void setup(){
  Serial.begin(9600);
}
void loop(){
  sensorValue=analogRead(analogPin);
  outputValue=map(sensorValue,0,1023,0,255);
```

```

analogWrite(analogOutPin,outputValue);
Serial.print(sensorValue);
Serial.print(outputValue);
delay(2);
}

```

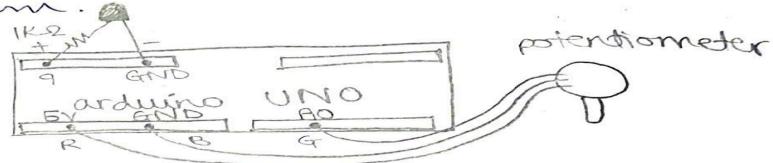
Handwritten code pic:

Experiment -3

Aim: To control brightness of LED using a Potentiometer.

Hardware components reqd: Arduino board, LED, Potentiometer.

Circuit diagram:



```

const int analogInPin = A0;
const int analogOutPin = 9;
int sensorValue = 0;
int OutValue = 0;

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

void loop() {
    sensorValue = analogRead(analogInPin);
    OutValue = map (sensorValue, 0, 1023, 0, 255);
    analogWrite(analogOutPin, OutValue);
    Serial.print (sensorValue);
    Serial.print (OutValue);
    delay(20);
}

```

~~Observation:~~

~~while varying the resistance vary the shaft of potentiometer the LED starts to fade in brightness~~

Observation: Based on the potentiometer shaft rotation output varies. LED glows if we rotate towards right and fades if we rotate towards left..

Program no: **04**

Program Title: **LED FADING**

Date:23/11/2023

Aim: To control the brightness of an LED without using a Potentiometer.

Hardware/components Required

Arduino Uno board - 1

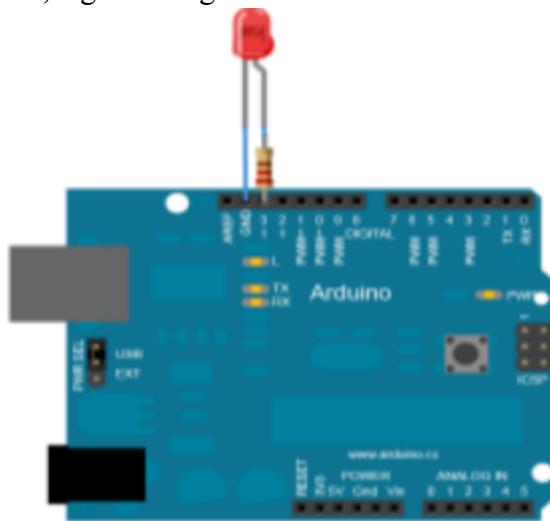
USB Cable - 1

LED - 1

Jumper wires

Circuit Diagram / Pin connection

LED positive to pin 9 ,negative to ground



Code:

```
int ledPin = 9; // LED connected to digital pin 9

void setup() {
}

void loop()
{
    // fade in from min to max in increments of 5 points:
    for (int fadeValue = 0 ; fadeValue <= 255; fadeValue += 5) {
        // sets the value (range from 0 to 255):
        analogWrite(ledPin, fadeValue);
        delay(30); // wait for 30 milliseconds to see the dimming effect
    }
}
```

```

// fade out from max to min in increments of 5 points:
for (int fadeValue = 255 ; fadeValue >= 0; fadeValue -= 5) {
    // sets the value (range from 0 to 255):
    analogWrite(ledPin, fadeValue);
    delay(30); } }

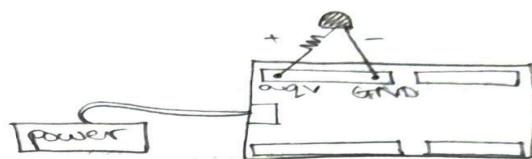
```

Handwritten code pic:

Experiment - 4

Am: LED Fading

Hardware Components req: Arduino Board, LED
Circuit Diagram:



```

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

```

Observation:

Fades at LED using the condition of 'for' loops at a delay of 30ms

S.P.
23/1/23

Observation: LED fades and glows periodically , output is visualized using arduino uno.

Program no: **05**

Program Title: **Nightlight Simulation**

Date:07/12/2023

Aim: Simulating a night light using LDR

Hardware/components Required

Arduino Uno board - 1

USB Cable - 1

LED - 1

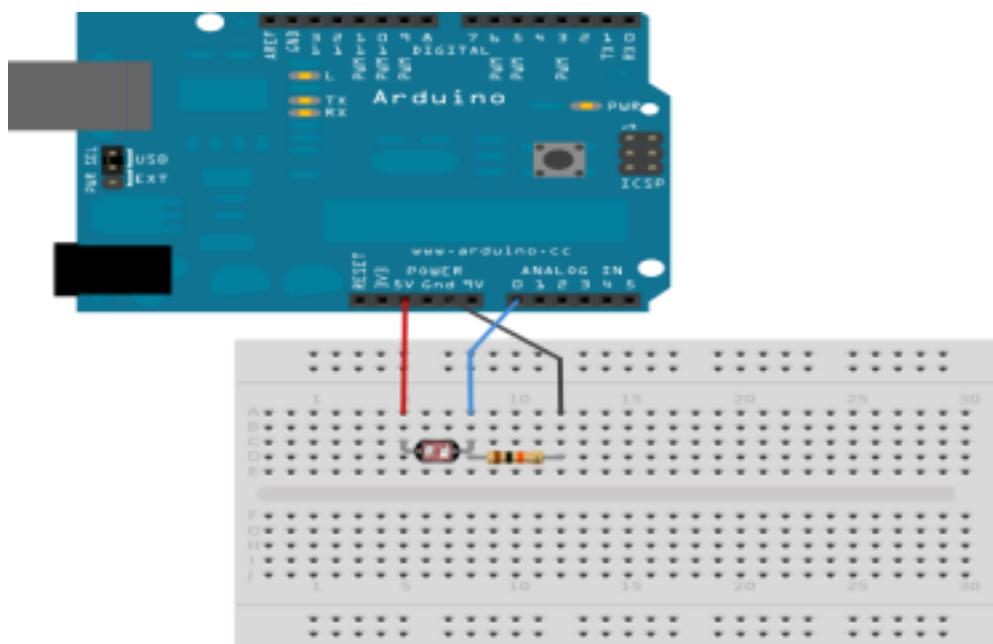
LDR-1

10K resistor-1

Jumper wires

Circuit Diagram / Pin connection

1. Attach one leg of LDR to 5V and another leg to Arduino Analog pin A0
2. Attach one leg of 110K register with that leg of LDR connected to A0
3. Attach another leg of register to the ground
4. Connect the positive leg of LED to pin 11 and negative to GND



Code:

```
int LDR = 0; //analog pin to which LDR is connected, here we set it to 0 so it means A0 int
LDRValue = 0; //that's a variable to store LDR values
int light_sensitivity = 500; //This is the approx value of light surrounding your LDR

void setup()
{
    Serial.begin(9600); //start the serial monitor with 9600 baud
    pinMode(11, OUTPUT); //attach positive leg of LED to pin 11
}

void loop()
{
    LDRValue = analogRead(LDR); //reads the ldr's value through LDR
    Serial.println(LDRValue); //prints the LDR values to serial monitor
    delay(50);
    //This is the speed by which LDR sends value to arduino

    if (LDRValue < light_sensitivity)
    {
        digitalWrite(11, HIGH);
    }
    else
    {
        digitalWrite(11, LOW);
    }
    delay(1000);
}
```

Handwritten code pic:

Lab:-03

Title:- Nightlight simulation (LDR)

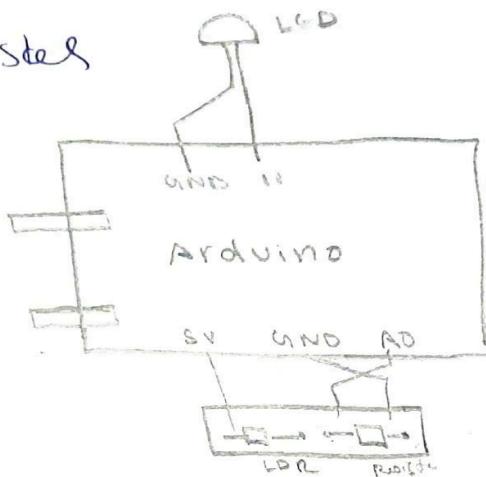
Aim:- Simulating a night light using LDR

Hardware Required:-

1) LED

2) LDR

3) 110K register



Connections:-

- 1) Attach one leg of LDR to 5V & another leg to Arduino analog pin A0
- 2) Attach one leg of 110K register with that leg of LDR connected to A0
- 3) Attach another leg of register to ground
- 4) Connect the Positive leg of LED Pin 11 an negative to GND.

Code:-

```
int LDR = 0;  
int LDR_value=0;  
int light_sensitivity=500;  
void setup()  
{  
    Serial.begin(9600);  
    PinMode(11, OUTPUT);  
}
```

```

void loop()
{
    LDRvalue = analogRead(LDR);
    Serial.println(LDRvalue);
    delay(500);
    if (LDR value < lightSensitivity)
    {
        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.

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.

Program no: **06**

Program Title: **Nightlight Simulation**

Date:07/12/2023

Aim: Simulating a night light using PIR

Hardware/components Required

Arduino Uno board - 1

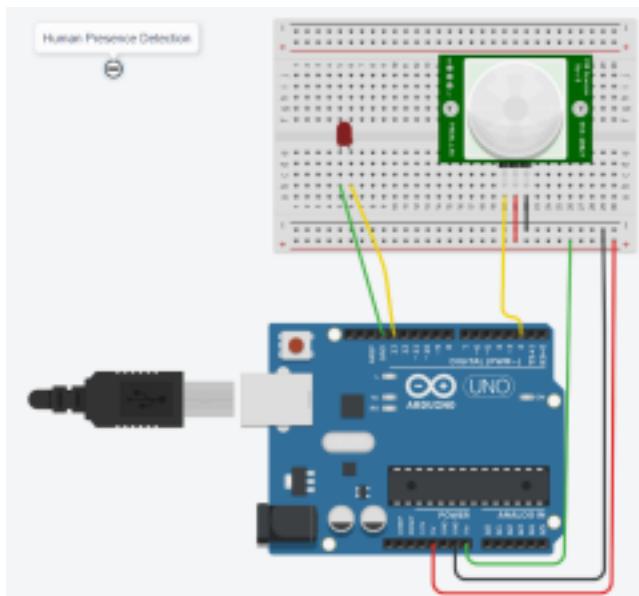
USB Cable - 1

LED - 1

PIR sensor-1

Jumper wires

Circuit Diagram / Pin connection



Code:

```
int sensorState = 0;

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

void loop()
{
  // read the state of the sensor/digital input
```

```
sensorState = digitalRead(2);
// check if sensor pin is HIGH. if it is, set the //
LED on.
if(sensorState == HIGH) {
    digitalWrite(13, HIGH);
    Serial.println("Sensor activated!");
} else {
    digitalWrite(13, LOW);
}
delay(10);
}
```

Handwritten code pic:

Program 1b

Title:- Night Light Simulation (PIR)

Aim:- Simulating a Night Light using PIR

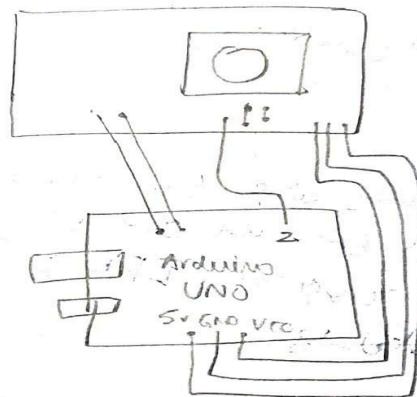
Hardware Required:-

1 LED

1 PIR Sensor

1 Arduino

1 Bread board



Coder

```

int sensorState=0;
void setup()
{
    pinMode (2, INPUT);
    pinMode (3, OUTPUT);
    Serial.begin (9600);
}

void loop()
{
    sensorState = digitalRead (2);
    if (sensorState == HIGH)
    {
        digitalWrite (3, HIGH);
        Serial.println ("Sensor detected");
    }
    else
        digitalWrite (3, LOW);
    delay (10);
}

```

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

Program no: **07**

Program Title: **Ultrasound sensing**

Date:07/12/2023

Aim: Simulating ultrasound with Arduino UNO and Ultrasonic sensor

Hardware/components Required

Arduino Uno board - 1

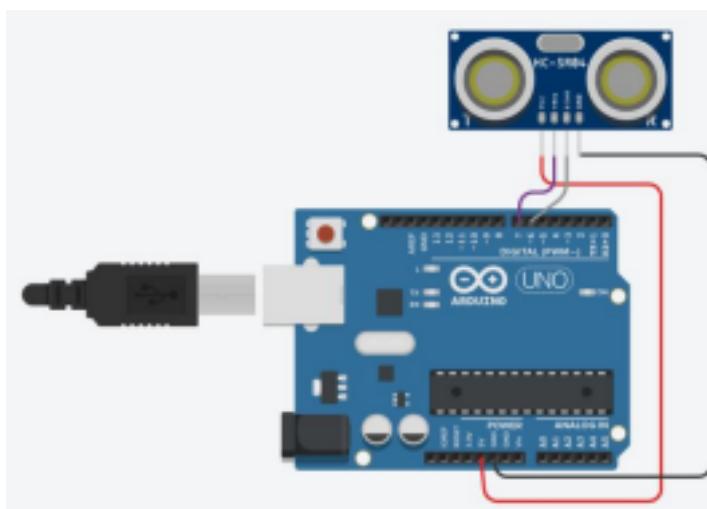
USB Cable - 1

Ultrasonic sensor-1

Jumper wires

Circuit Diagram / Pin connection

VCC-5V , GND-GND , pingpin-7 , echopin - 6



Code:

```
const int pingPin = 7;  
  
const int echoPin=6;// Trigger Pin of Ultrasonic Sensor const int echoPin = 6; // Echo Pin of  
Ultrasonic Sensor  
  
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);  
inches = microsecondsToInches(duration);  
Serial.print(inches);  
Serial.print("inches");  
cm = microsecondsToCentimeters(duration);  
Serial.print(cm);  
Serial.println("cm");  
}  
  
long microsecondsToInches(long microseconds) {  
    return microseconds / 74 / 2;  
}  
  
long microsecondsToCentimeters(long microseconds) {  
    return microseconds / 29 / 2;  
}
```

Handwritten code pic:

Observation

LED turns on in dark & switch off in light

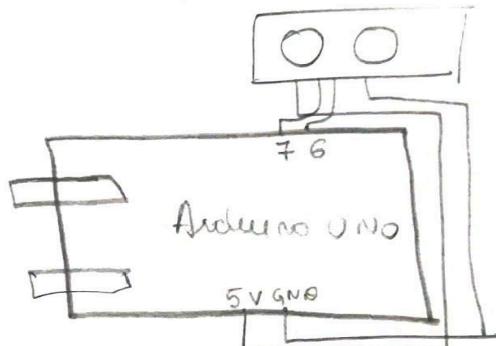
Program - 7

Title : Ultrasound with arduino uno

Hardware

Arduino

ultrasonic sensor



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);  
    duration = pulseIn(EchoPin, HIGH);  
}
```

5 waves = microseconds To Inches (duration);
Serial. Port (inches)

Serial. Port (" inches");

Convert microseconds To Centimeters (duration);

Serial. Port (cm);

long microseconds To Inches (long microseconds)
{ between microseconds 17418; }

long microseconds To Centimeters (long microseconds)

{ between microseconds, 2912; }

Observation-

Distance is measured based on vibrations.



Observation: Based on vibrations of sound, distance will be measured

Aim: Fire alarm simulation

Hardware/components Required

Flame sensor (Analogue Output)

Arduino

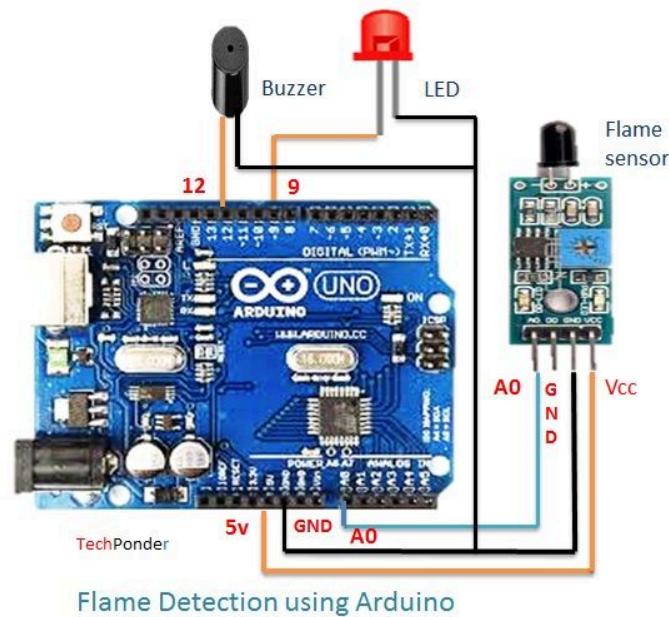
Bread board

LED

Buzzer

Connecting wires

Circuit Diagram / Pin connection



Flame sensor interfacing to Arduino

Flame sensor to Arduino

vcc -> vcc

gnd -> gnd

A0 -> A0

Led interfacing to Arduino

LED +ve is connected to **9th pin** of Arduino

LED -ve is connected to **gnd pin** of arduino

Buzzer interfacing to Arduino

Buzzer +ve is connected to **12th pin** of Arduino

Buzzer -ve is connected to **GND** pin of Arduino

Code:

```
int sensorPin = A0; // select the input pin for the LDR  
int sensorValue = 0; // variable to store the value coming from the sensor  
int led = 9; // Output pin for LED  
int buzzer = 12; // Output pin for Buzzer  
  
void setup() {  
    // declare the ledPin and buzzer as an OUTPUT:  
    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);
}

```

Handwritten code pic:

```

Code :-
int sensorPin = A0;
int sensorValue = 0;
int led = 9;
int led = 10; // buzzer = 10;
void setup() {
    PinMode(led, OUTPUT);
    PinMode(buzzer, OUTPUT);
    Serial.begin(9600);
}

void loop() {
    Sensor Value = analogRead(sensorPin);
    Serial.print (Sensor Value);
    if (Sensor value > 100) {
        Serial.print ("Fire Detected");
        Serial.print ("LED On");
        digitalWrite(led, HIGH);
        digitalWrite(buzzer, HIGH);
        delay(1000);
        digitalWrite(led, LOW);
        digitalWrite(buzzer, LOW);
        delay (Sensor Value);
    }
}

```

Observation:-

If fire is detected LED & buzzer turns on

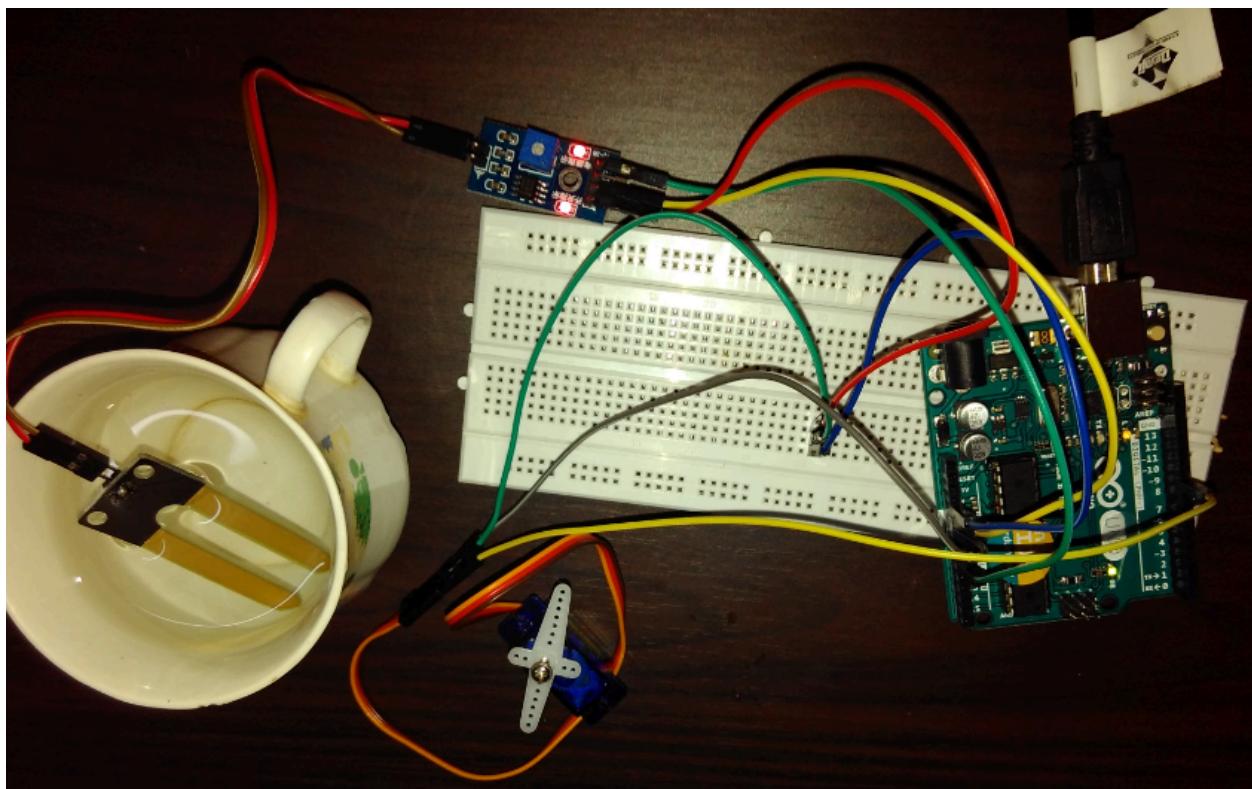
Observation: When flame is detected, LED and buzzer turns ON.

Aim: Sensing the soil moisture and sprinkling the Water simulation

Hardware Required

Arduino
Moisture Sensor
Breadboard
Min servo motor

Circuit diagram



Moisture sensor VCC to Arduino 5V
Moisture sensor GND to Arduino GND
Moisture sensor A0 to Arduino A0

Servo motor VCC to Arduino 5V
Servo motor GND to Arduino GND

Servo Motor Signal to Arduino digital pin 9

Code:

```
#include <Servo.h>
Servo myservo; // create servo object to control a servo
// twelve servo objects can be created on most boards
int pos = 0; // variable to store the servo position

int sensorPin = A0; // select the input pin for the potentiometer
int sensorValue = 0; // variable to store the value coming from the sensor
void setup() {
  myservo.attach(9); // attaches the servo on pin 9 to the servo object
  Serial.begin(9600);
}
void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  Serial.println (sensorValue);
  if(sensorValue>500)
  {
    for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
      // in steps of 1 degree
      myservo.write(pos); // tell servo to go to position in variable 'pos'
      delay(15); // waits 15ms for the servo to reach the position
    }
    for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
      myservo.write(pos); // tell servo to go to position in variable 'pos'
      delay(15); // waits 15ms for the servo to reach the position
    }
  }
  delay (1000);
}
```

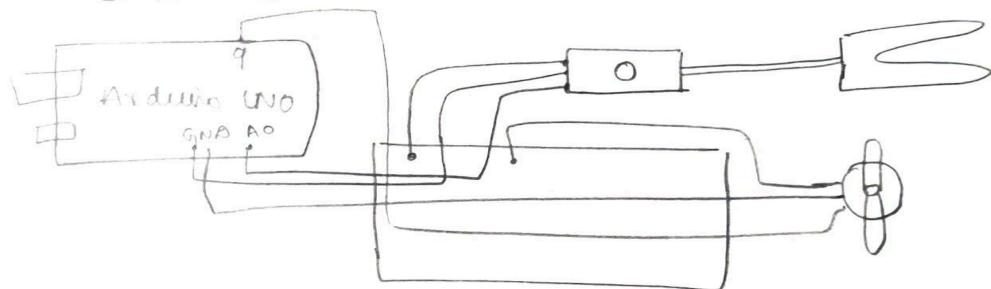
Handwritten code pic:

Program - 09

Title :- Automatic irrigation controller simulation
Aim:- Densifying the soil moisture by periodically
Irrigation simulation.

Hardware:-

- 1) Arduino Uno
- 2) Moisture sensor
- 3) Bread board
- 4) Servo motor



moisture sensor VCC to Arduino 5V
GND to Arduino GND
Sensor AO to Arduino A0

Servo motor VCC to Arduino 5V
GND to GND
Signal to digital pin

Code:-

```
#include <Servo.h>
Servo myServo;
int pos = 0;
int servoPin = A0;
int servoValue = 0;
void setup(){
    myServo.attach(9);
    Serial.begin(9600);
}
```

```

void loop()
{
    servovalue = analogRead (sensorPin);
    Serial.print (servovalue);
    if (servovalue > 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);
    }
}

```

Observation:-

water moisture is sensed and the value is visible in serial monitor

 S.P.
 21/12/23

Observation: Soil moisture sensor continuously detects the soil moisture and servo motor would turn ON when there is a low moisture level.

Aim: To read the code present on RFID tag and print it in serial monitor.

Hardware/components Required

Arduino Uno board - 1

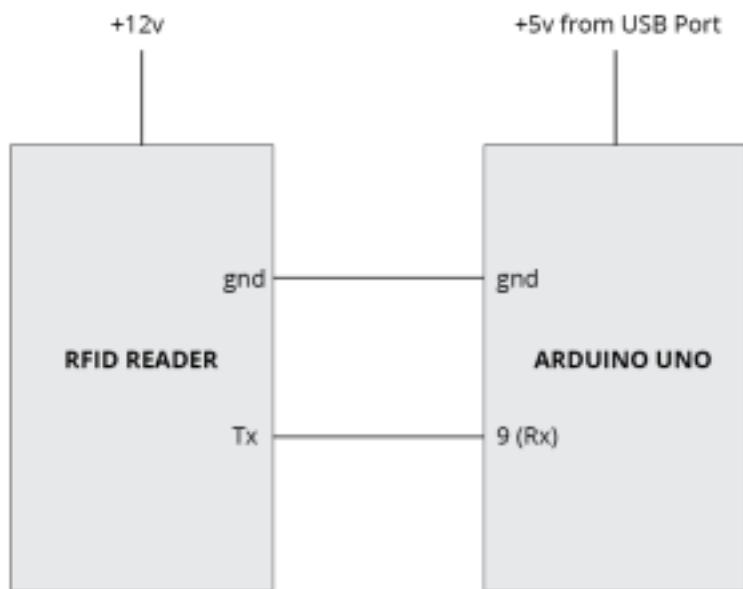
USB Cable - 1

RFID tag

Jumper wires

Circuit Diagram / Pin connection

5V-Arduino 5V GND-Arduino GND Tx-pin 9



Interfacing RFID Reader to Arduino

Code:

```
#include<SoftwareSerial.h>
```

```
SoftwareSerial mySerial(9, 10);
```

```

int count = 0; // count = 0
char input[12]; // character array of size 12
boolean flag = 0; // flag =0
void setup()
{
    Serial.begin(9600); // begin serial port with baud rate 9600bps mySerial.begin(9600);
}
void loop()
{
    if(mySerial.available())
    {
        count = 0;
        while(mySerial.available() && count < 12) // Read 12 characters and store them in input
array
        {
            input[count] =mySerial.read();
            count++;
            delay(5);
        }
        Serial.print(input); // Print RFID tag number
    }
}

```

Handwritten code pic:

sim : reading the code present on RFID tag connection :

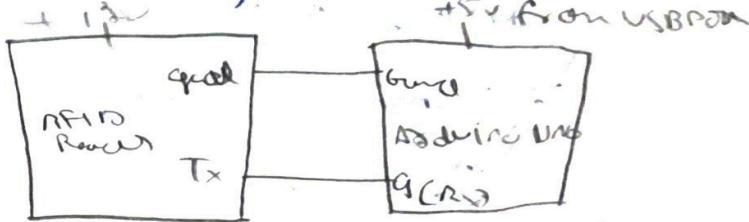
5V - Arduino 5V

GND - Arduino GND

TX - Pin 9

(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);
    }
}
```



Observation: The Program reads the code 01010096268069

Observation: The output consists of 12 character ASCII data, where first 10 bits will be the tag number and last 2 bits will be the XOR result of the tag number which can be used for error correction.

Aim: To read the code present on RFID tag tapped. If the code matches with the previously known tag(configured in the code), it will grant access(here LED will glow), otherwise access will be denied.

Hardware/components Required

Arduino Uno board - 1

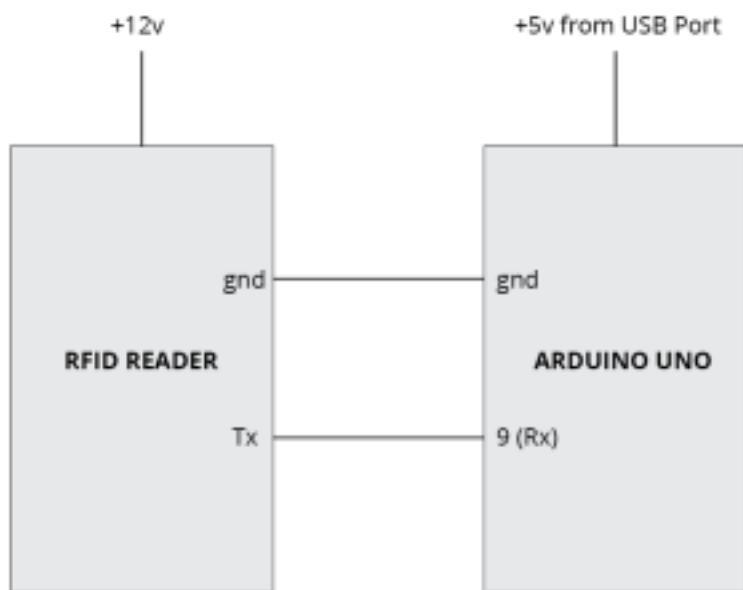
USB Cable - 1

RFID tag

Jumper wires

Circuit Diagram / Pin connection

5V-Arduino 5V GND-Arduino GND Tx-pin 9



Interfacing RFID Reader to Arduino

Code:

```
#include<SoftwareSerial.h>
```

```
SoftwareSerial mySerial(9, 10);
```

```

#define LEDPIN 12
char tag[] = "5300292DD087"; // Replace with your own Tag ID
char input[12]; // A variable to store the Tag ID being presented int count = 0; // A counter
variable to navigate through the input[] character array
boolean flag = 0; // A variable to store the Tag match status void setup()
{
    Serial.begin(9600); // Initialise Serial Communication with the Serial Monitor
    mySerial.begin(9600);
    pinMode(LEDPIN,OUTPUT); //WRONG TAG INDICATOR
}
void loop()
{
    if(mySerial.available())// Check if there is incoming data in the RFID Reader Serial
Buffer.
    {
        count = 0; // Reset the counter to zero
        /* Keep reading Byte by Byte from the Buffer till the RFID Reader Buffer is
empty
            or till 12 Bytes (the ID size of our Tag) is read */
        while(mySerial.available() && count < 12)
        {
            input[count] = mySerial.read();
            // Read 1 Byte of data and store it in the input[] variable
            Serial.write(input[count]);
            count++; // increment counter
            delay(5);
        }
        /* When the counter reaches 12 (the size of the ID) we stop and compare each
value
            of the input[] to the corresponding stored value */
        if(count == 12) //
        {
            count = 0; // reset counter varibale to 0
            flag = 1;
            /* Iterate through each value and compare till either the 12 values are
all matching or till the first mismatch occurs */
            while(count<12 && flag !=0)
            {
                if(input[count]==tag[count])
                    flag = 1; // everytime the values match, we set the flag variable
to 1
                else
                    flag= 0;
            }
            /* if the ID values don't match, set flag variable to 0 and
stop comparing by exiting the while loop */
            count++; // increment i
        }
    }
}

```

```

        }
    }
    if(flag == 1) // If flag variable is 1, then it means the tags match
    {
        Serial.println("Access Allowed!");
        digitalWrite(LEDPIN,HIGH);
        delay (2000);
        digitalWrite (LEDPIN,LOW);
    }
else
{
    Serial.println("Access Denied"); // Incorrect Tag Message
    digitalWrite(LEDPIN,LOW);
    delay(2000);
}
/* Fill the input variable array with a fixed value 'F' to overwrite
all values getting it empty for the next read cycle */
for(count=0; count<12; count++)
{
    input[count]='F';
}
count = 0; // Reset counter variable
}
}

```

Handwritten code pic:

Program - 08

sim. access control through RFID
connection: 5V - Arduino 5V
GND - Arduino GND
TX - Pin
LED - Pin 12

code:

```
#include <SoftwareSerial.h>
SoftwareSerial myserial(9,10);
#define LEDPIN 12
char tag[7] = "53002a200087";
char input[18];
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 < 10)
        {
            input[count] = myserial.read();
            count++;
        }
        Serial.println(input);
        delay(5);
    }
    if (count == 10)
    {
        count = 0;
        flag = 1;
    }
}
```

Observation: If the code matches with the previously known tag(configured in the code), it will grant access(here LED will glow), otherwise access will be denied.

Aim: To monitor the temperature using LM35.

Hardware/components Required

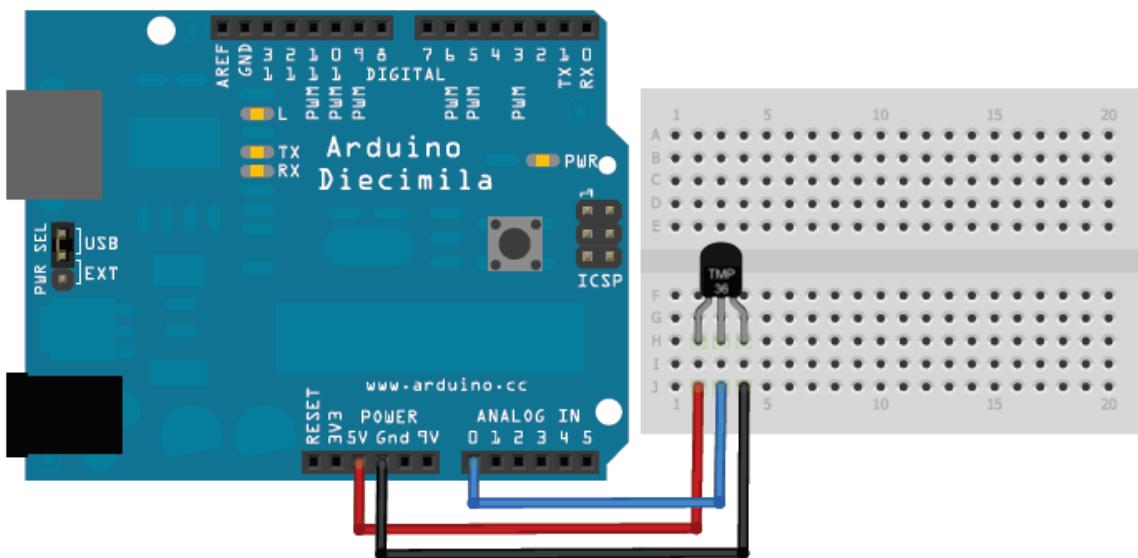
Arduino Uno board - 1

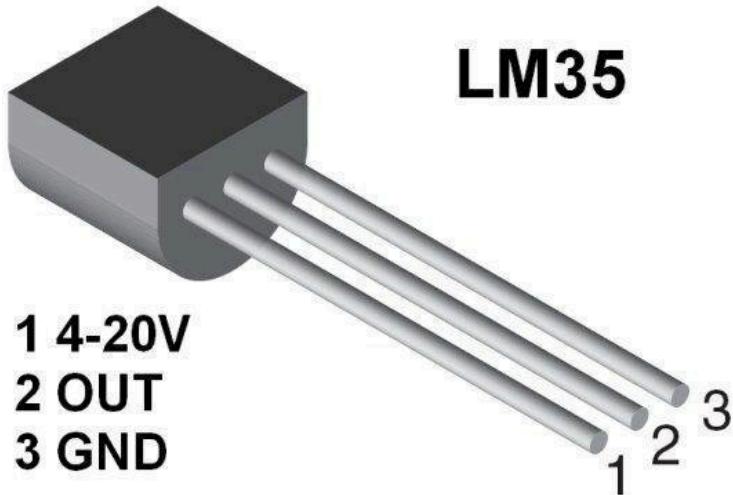
USB Cable - 1

Temperature sensor LM35

Jumper wires

Circuit Diagram / Pin connection





Code:

```

int sensorPin = 0; //the analog pin the TMP36's Vout (sense) pin is connected to
                    //the resolution is 10 mV / degree centigrade with a
                    //500 mV offset to allow for negative temperatures

/*
 * setup() - this function runs once when you turn your Arduino on
 * We initialize the serial connection with the computer
 */
void setup()
{
    Serial.begin(9600); //Start the serial connection with the computer
                        //to view the result open the serial monitor
}

void loop()          // run over and over again
{
    //getting the voltage reading from the temperature sensor
    int reading = analogRead(sensorPin);
}

```

```
// converting that reading to voltage, for 3.3v arduino use 3.3
float voltage = reading * 5.0 / 1024;

// print out the voltage
Serial.print(voltage); Serial.println(" volts");

// now print out the temperature
float temperatureC = (voltage - 0.5) * 100; //converting from 10 mv per degree wit 500 mV
offset
                                //to degrees ((volatge - 500mV) times 100)
Serial.print(temperatureC); Serial.println(" degress C");

// now convert to Fahrenheight
float temperatureF = (temperatureC * 9 / 5) + 32;
Serial.print(temperatureF); Serial.println(" degress F");

delay(1000);                  //waiting a second
}
```

Handwritten code pic:

Observation:

A user will be granted if the code makes
the task. Otherwise it is denied.

Program 3

Aim: Temperature sensing

Hardware: Arduino Uno

Breadboard

Temperature sensor

Code:

```
int sensor = Pin 0;  
void setup ()  
{  
    Serial.begin(9600);  
}  
  
void loop ()  
{  
    int reading = analogRead (sensorPin);  
    float Voltagge = reading * 5.0 / 1024;  
    Serial.print(Voltagge);  
    Serial.println ("Voltagge");  
    float Temperature = (Voltage - 0.5) * 100;  
    Serial.print (Temperature);  
    Serial.println ("Degree C");  
    float TemperatureF = (Temperature * 9/5) + 32;  
    Serial.print (TemperatureF);  
    Serial.println ("Degree F");  
    delay (1000);  
}
```

Observation: Sensor senses the temperature of the surroundings as 21C

Program no: 13

Program Title: **GSM CALLING**

Date:28/12/2023

Aim: Call using Arduino and GSM Module – to a specified mobile number inside the program.

Hardware/components Required

Arduino Uno board - 1

USB Cable - 1

GSM module

SIM slot

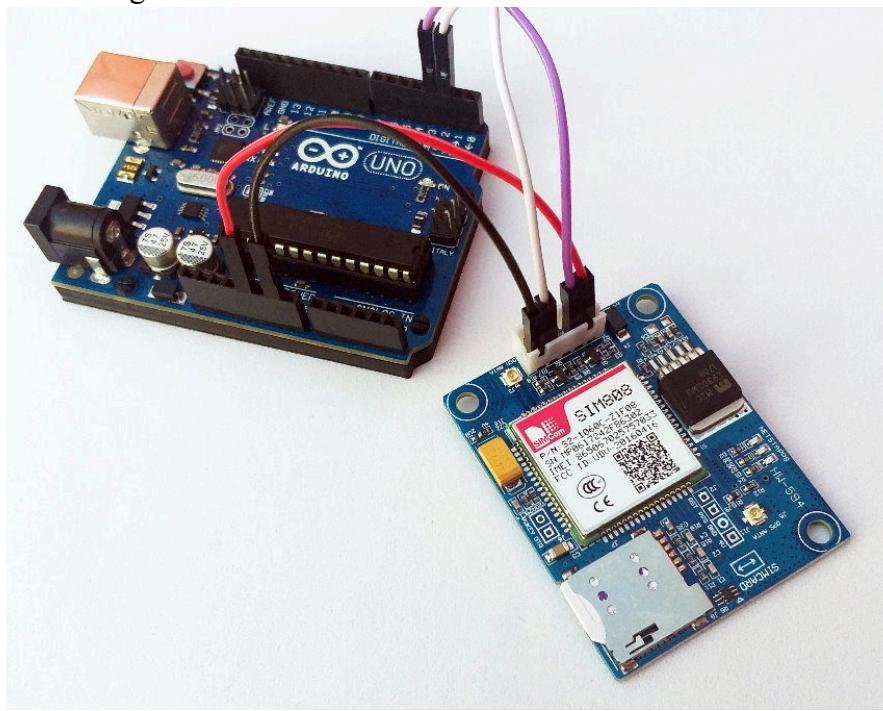
Jumper wires

Circuit Diagram / Pin connection:

GSM Tx → Arduino Rx (Here pin 2)

GSM Rx → ArduinoTx. (Here pin 3)

Make the ground common between Arduino and GSM modem.



Code:

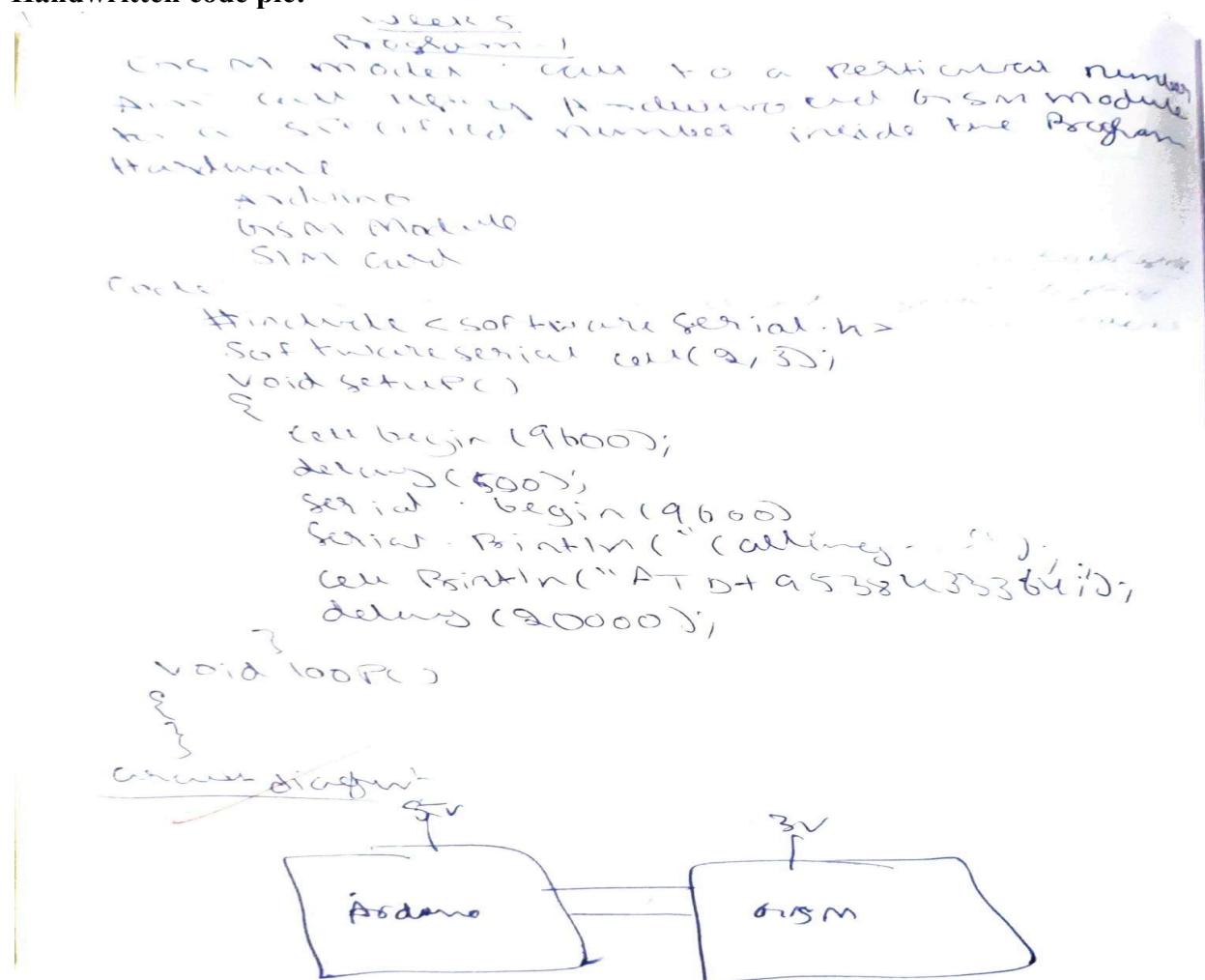
```
#include <SoftwareSerial.h>
SoftwareSerial cell(2,3); // (Rx, Tx)
```

```

void setup() {
cell.begin(9600);
delay(500);
Serial.begin(9600);
Serial.println("CALLING.....");
cell.println("ATD+9538433364;"); // ATD – Attention Dial
delay(20000);
}
void loop() {
}

```

Handwritten code pic:



Observation: Call to a entered number occurs with a sound

Observation: Calling to GSM module , you'll get beep sound

Program no: 14

Program Title: **GSM CALLING FIRE ALERT**

Date: 11/01/2024

Aim: Call a specified mobile number mentioned in the program using Arduino and GSM Module when a flame sensor detects “fire”.

Hardware/components Required

Arduino Uno board - 1

USB Cable - 1

GSM module

SIM slot

Flame sensor

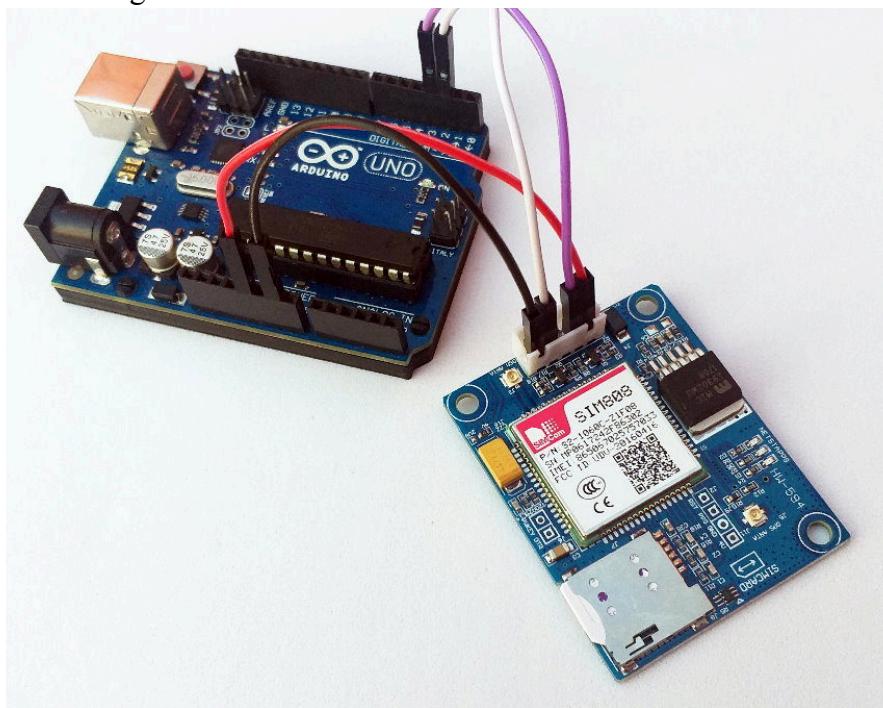
Jumper wires

Circuit Diagram / Pin connection:

GSM Tx → Arduino Rx (Here pin 2)

GSM Rx → Arduino Tx. (Here pin 3)

Make the ground common between Arduino and GSM modem.



Code:

```
#include <SoftwareSerial.h>
SoftwareSerial cell(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"); // Attention Hook Control
}
}
```

Handwritten code pic:

Program 8

Call to a particular number on alert
Ain call a Specified mobile number mentioned
in the Program using Arduino Ad GSM module
when a flame sensor detects fire.

Connections for Flame sensor :-

Arduino	Flame Sensor
5V	VCC
GND	GND
A0	A0

Code:-

```
#include <SoftwareSerial.h>
SoftwareSerial serial(2, 3);
void setup()
{
    Serial.begin(9600);
    delay(500);
    serial.begin(9600);
}

void loop()
{
    int val = analogRead(A0);
    serial.println(val);
    delay(1000);
    if (val < 50)
    {
        serial.println("CALLING... ");
        serial.println("AT&T+91 9748980606");
        delay(10000);
        serial.println("ATH");
    }
}
```

Observation: When there is a flame, a particular specified number will get a call as an alert.

Aim:

- 1) Send SMS using Arduino and GSM Module – to a specified mobile number inside the program**
- 2) Receive SMS using Arduino and GSM Module – to the SIM card loaded in the GSM Module.**

Hardware/components Required

Arduino Uno board - 1

USB Cable - 1

GSM module

SIM slot

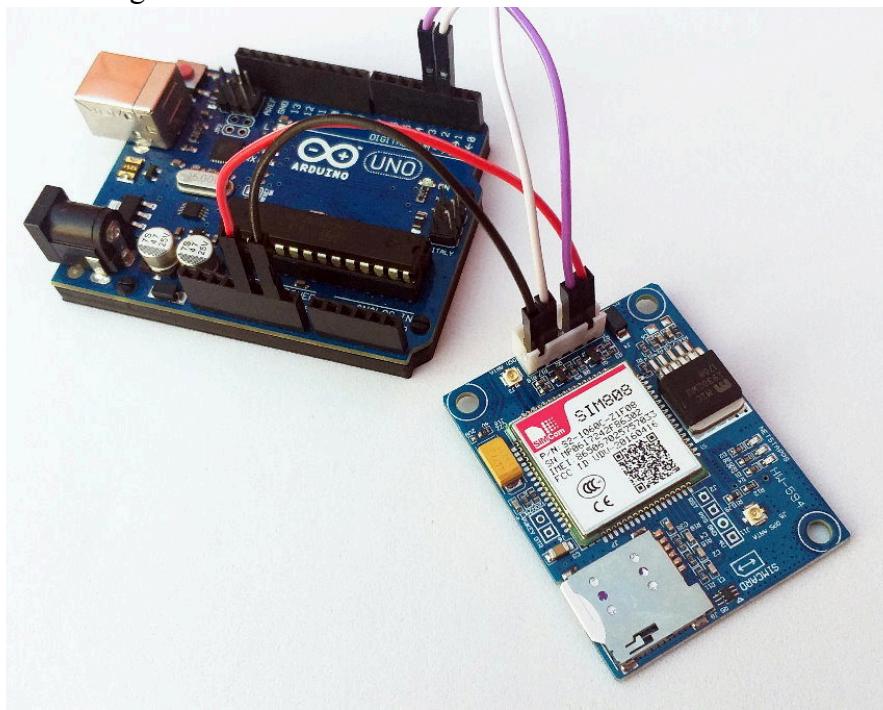
Jumper wires

Circuit Diagram / Pin connection:

GSM Tx → Arduino Rx (Here pin 2)

GSM Rx → Arduino Tx. (Here pin 3)

Make the ground common between Arduino and GSM modem.

**Code:**

```

#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3);

void setup()
{
mySerial.begin(9600); // Setting the baud rate of GSM Module
Serial.begin(9600); // Setting the baud rate of Serial Monitor (Arduino)
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());
}

void SendMessage()
{
mySerial.println("AT+CMGF=1"); //Sets the GSM Module in Text Mode //AT+CMGF, SMS Format
delay(1000); // Delay of 1000 milli seconds or 1 second
mySerial.println("AT+CMGS=\\"+91974298060\\r"); // AT+CMGS, Send Message
// Replace with your mobile number
delay(1000);
mySerial.println("I am SMS from GSM Module");
// The SMS text you want to send
delay(100);
mySerial.println((char)26); // ASCII code of CTRL+Z , to terminate the message
delay(1000);
}

```

```
}
```

```
void RecieveMessage()
{
mySerial.println("AT+CNMI=2,2,0,0,0"); // AT+CNMI, New Message Indications
// AT Command to recieve a live SMS
delay(1000);
}
```

Handwritten code pic:

Program - 03
Sending and Receiving message

Aim:

- 1) Send SMS using Arduino and GSM Module to a specified mobile number inside the program
- 2) Recieve SMS using Arduino and GSM Module - to the SIM card loaded in the GSM Module

Code:

```
#include <SoftwareSerial.h>
SoftwareSerial mySerial (2, 3);
Void Setup()
{
    mySerial.begin(9600);
    Serial.begin(9600);
    delay(1000);
}

Void loop()
{
    if (Serial.available(>0))
        switch (Serial.read())
    {
        case 's':
            SendMessage();
            break;
        case 'r':
            ReceiveMessage();
            break;
    }
    if (mySerial.available(>0))
        Serial.write(mySerial.read());
}
```

Observation: According to the code, messages will be sent and received when 's' and 'r' are pressed through serial monitor respectively.

Program no: 16 Program Title: BLUETOOTH MASTER SLAVE Date: 18/01/2024

Aim: To control the LED in the master device by client device.

Hardware/components Required

Arduino Uno board - 2

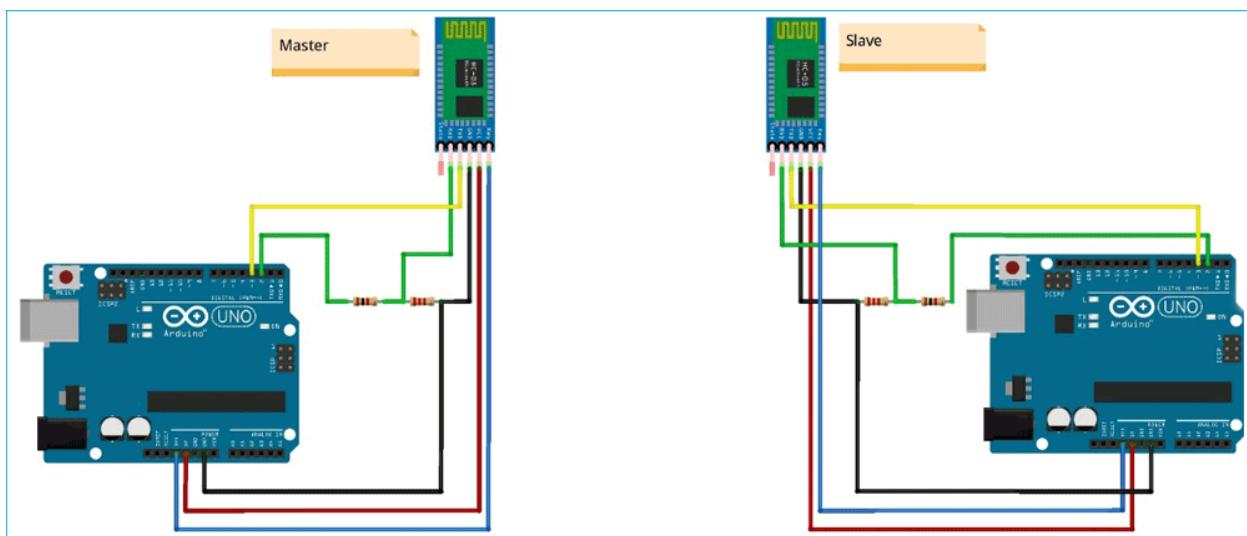
USB Cable - 1

Jumper wires

LED-1

HC-05 bluetooth module-2

Circuit Diagram / Pin connection:



Slave Mode:

The HC-05 bluetooth module can also act as a slave. There are fewer commands to set this up:

AT+ORGL Reset to defaults

AT+RMAAD Clear any paired devices

AT+ROLE=0 Set mode to SLAVE

AT+ADDR Display SLAVE address //+ADDR:98d3:33:807822

Master Mode:

To configure the module as Bluetooth Master and to pair with another bluetooth module follow these steps. First we need to put the module into command mode Enter these commands in order:

AT+RMAAD Clear any paired devices

AT+ADCN

AT+ROLE=1 Set mode to Master

AT+CMODE=0 Allow master to ONLY connect to bound address (slave). This allows the master to automatically connect to the slave when switched on AT+PSWD=1234 Set PIN. Should be same as slave device

AT+BIND=<address> Set bind address to the slave address

AT+LINK=<address> Connect to slave.

AT+INIT

Note: If it shows any Error, then check if both the bluetooth modules are blinking in sync. If so then both the bluetooth modules are synchronized.

BT-Slave Program:

```
#include <SoftwareSerial.h>

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

void setup() {
  Serial.begin(9600);
  BTSerial.begin(38400); // HC-05 default speed in AT command more }
void loop() {
  // Reading the button
  if(Serial.available())
  {
    String message = Serial.readString();
    Serial.println (message);
    BTSerial.write(message.c_str());
  }
}
```

BT-Master Program:

```
#include <SoftwareSerial.h>

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

#define ledPin 9

String message;
int potValue = 0;
void setup() {
  pinMode(ledPin, OUTPUT);
```

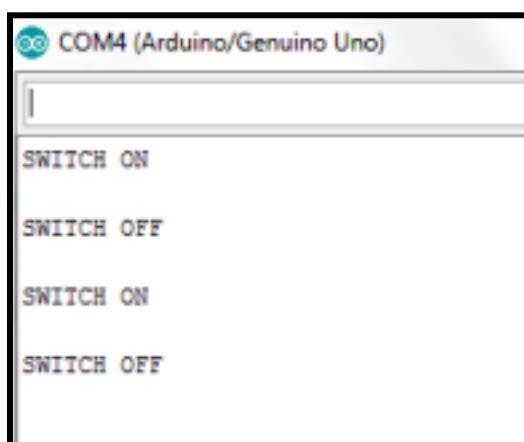
```

digitalWrite(ledPin, LOW);
Serial.begin(9600);
BTSerial.begin(38400); // HC-05 default speed in AT command more }

void loop() {
if(BTSerial.available() > 0){
// Checks whether data is comming from the serial port //
Reads the data from the serial port
message = BTSerial.readString();
// Controlling the LED
if(message.indexOf("SWITCH ON")>=0)
{
  digitalWrite(ledPin, HIGH); // LED ON
}
else if(message.indexOf("SWITCH OFF")>=0)
{
  digitalWrite(ledPin, LOW); // LED OFF
}
else
{
  Serial.println("Noting to do");
}
delay(100);

}
delay(10);
}

```



Handwritten code pic:

Program BLUETOOTH

Aim : To control the LED in the master device by client device.

Hardware Required :

Arduino Uno board - 1

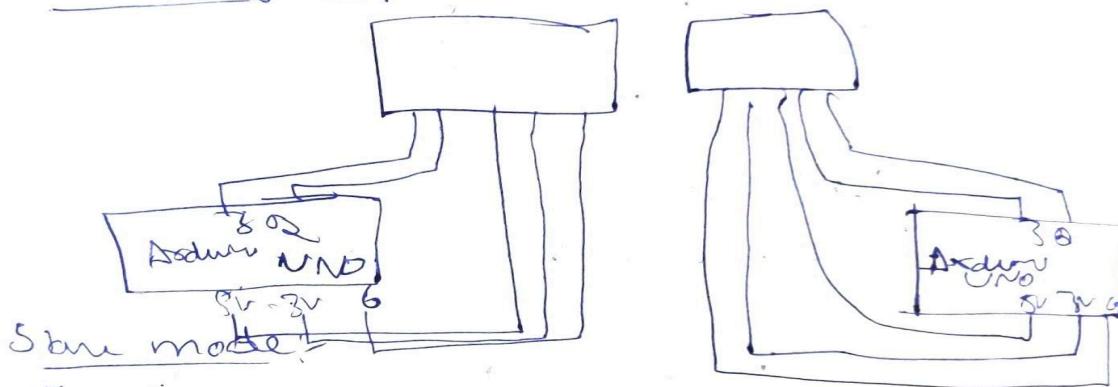
USB cable - 1

Jumper wires

LED - 1

HC-05 bluetooth module - 2

Circuit Diagram :



```
#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.print("Message: ");
    Serial.println(message);
    BTserial.write(message.c_str());
  }
}
```

Makkelij

Attire < software serv. h?

Software-Schul-Büro (10,11)

Adeline Lee Pin 4

Stay message;

(Mr. Pot-Vane : 0)

```
void getuser(){
```

```
PinMode (ledPin, OUTPUT);
```

Daguerreotype (red Pin, Louisj)

Issue · begin (9600);
RTS

BT Seqn begin (3 seqn),

۳

void loops()

$\text{f}(\text{CBT}_{\text{open}} \cdot \text{area}(w) > 0)$

5

message = B7Seine - 2nd Story(1);

If Complexity index of C_{SWITCHON}

S

digital voice (red pin, AT/PA)

3

else if (military, police, of,

5

Mayer Wahr (Lead P₂O₅, low)

Elmer

- { serv. Borsig ("Wolky todo") }

```
}
```

```
delay(10);
```

```
}
```

MPR

Switch ON

Switch OFF

Switch ON

Switch OFF

Observe what device sends the message

"SWITCH ON", LED turns ON in every
OFF if the message is "SWITCH OFF",
it shows "Nothing to do" in serial monitor

Observation: Whenever Client device sends the message “SWITCH ON”, LED turns ON and turns OFF if the message is “SWITCH OFF” otherwise it prints “Nothing to do” in the serial monitor.