VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

INTERNET OF THINGS LAB

Submitted by

PRIYADARSHINI K M(1BM22CS413)

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
NOV-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 **PRITADARSHINI K M(1BM22CS413),** 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. The Lab report has been approved as it satisfies the academic requirements in respect of a **Internet of things lab - (22CS5PCIOT)** work prescribed for the said degree.

Karanam sunil kumar
Assistant professor
Department of CSE
BMSCE, Bengaluru

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

`

Index

1. LED Blinking

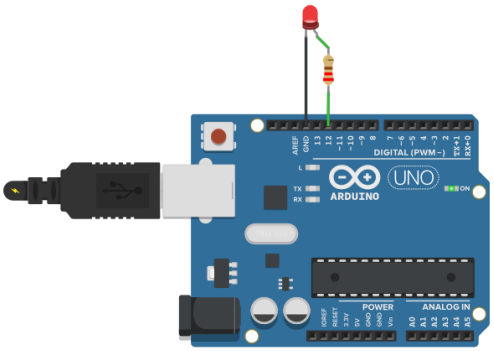
Aim: Turns on an LED on for one second, then off for one second, repeatedly

Hardware Required:

• Arduino Board

• LEDs

Connection:



Code:

```
// Pin 13 has an LED connected on most Arduino boards
int led = 13;
void setup()
{
pinMode(led, OUTPUT);
}
void loop() {
forever
digitalWrite(led, HIGH);
delay(1000);
digitalWrite(led, LOW);
delay(1000); }
```

Program - 01

Blinking of LED

AIM:- To do Blinking of a LED in Arduino for one sec & then off for one second. Repeadedly

components :- Arduino Board, LED, USB cable.

Procedure :- connect longer pin of led (+ve) to Pin 13 and shorter one to ground connect USB cable to arduin and another end to PowerSupply. Now seled board arduino UNO. Now compile, verify and upload the code in arduino IDE. Now we can see blinking of LED.

Code:-

```
void setup()
{
  pinMode(13, OUTPUT);
}
void loop()
{
  digitalWrite(13, HIGH)
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

2. LED ON/OFF Using Pushbutton

Aim: Turn an LED ON /OFF using a Pushbutton.

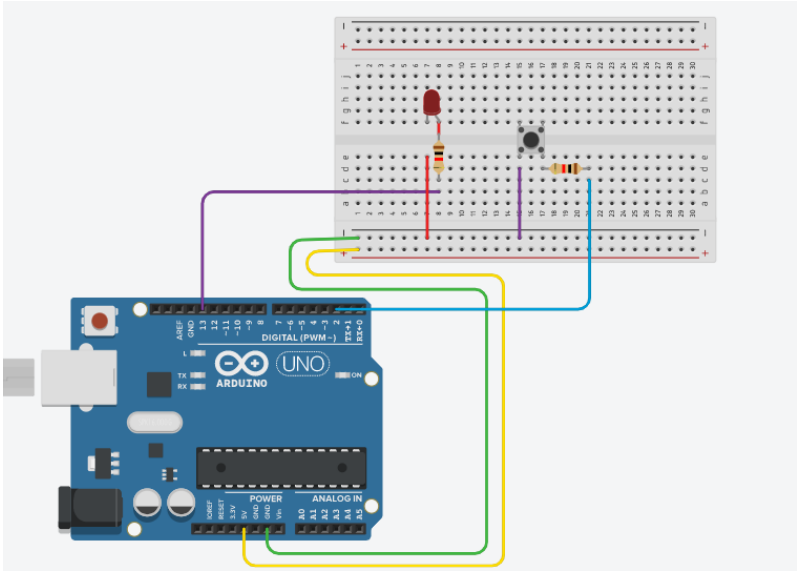
Hardware Required:

• Arduino Board

• LED

• Push button

Connection:



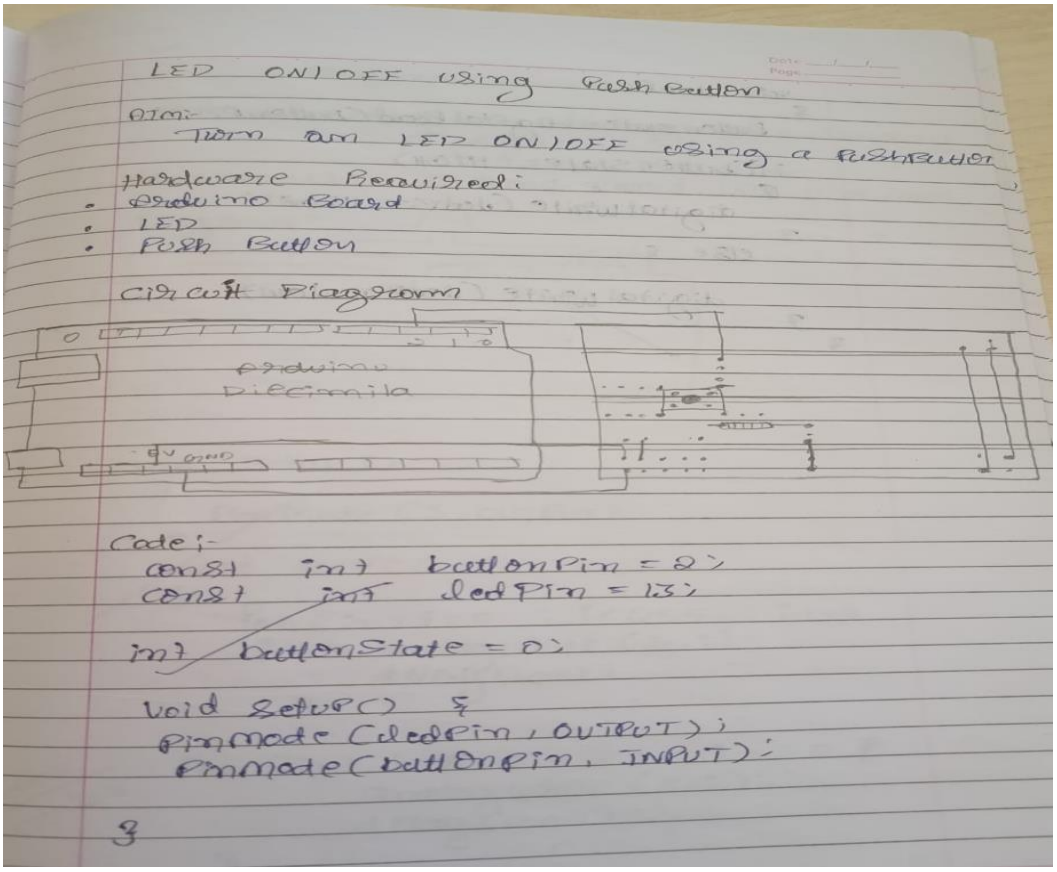
Code:

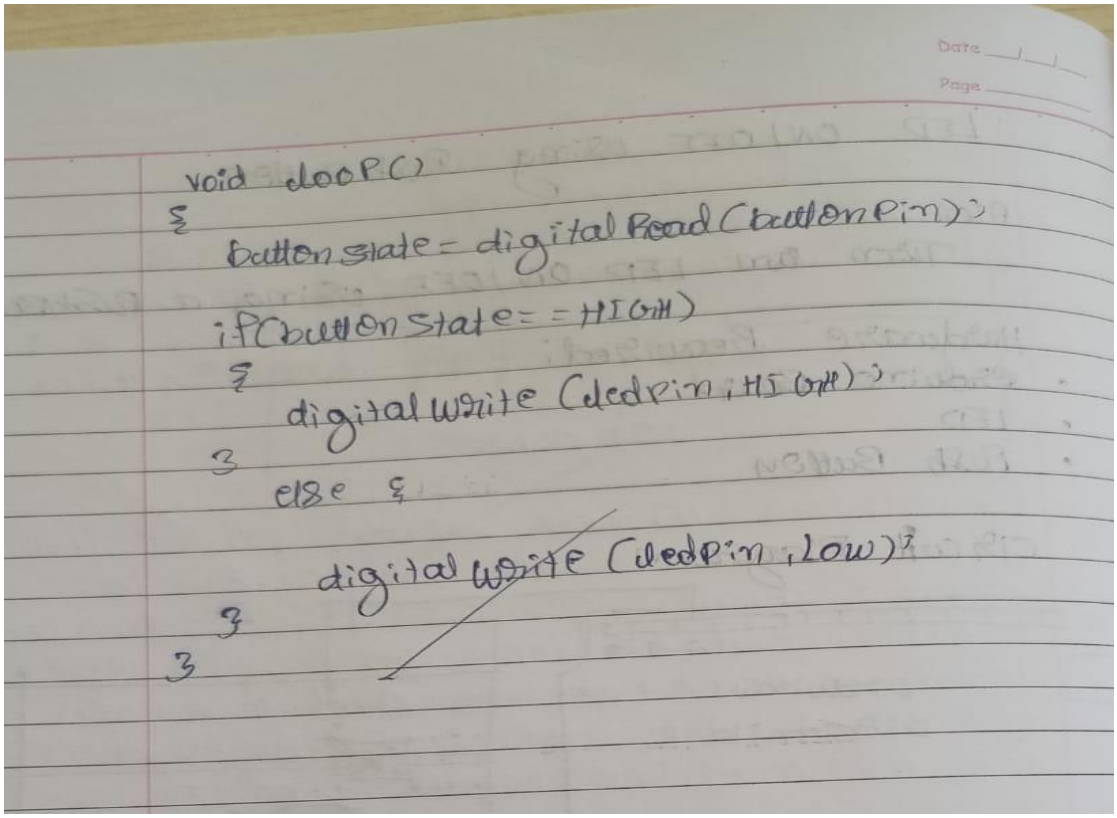
```
const int buttonPin = 2;
const int ledPin = 13;
int buttonState = 0;
void setup() {
pinMode(ledPin, OUTPUT);
pinMode(buttonPin, INPUT);
}
void loop() {
buttonState = digitalRead(buttonPin);
if (buttonState == HIGH) {
digitalWrite(ledPin, HIGH);
} else {
```

digitalWrite(ledPin, LOW); // Turn off the LED

}

}





3. LED Fading using Potentiometer

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

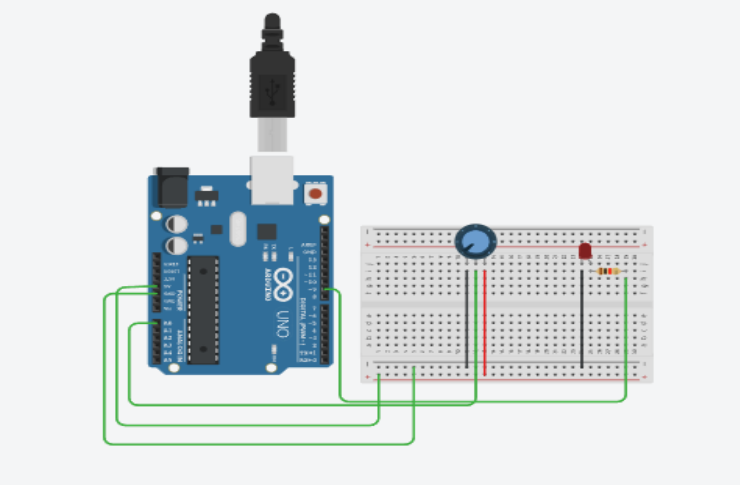
Hardware Required:

• Arduino Board

• LED

• Potentiometer

Connection:



Code:

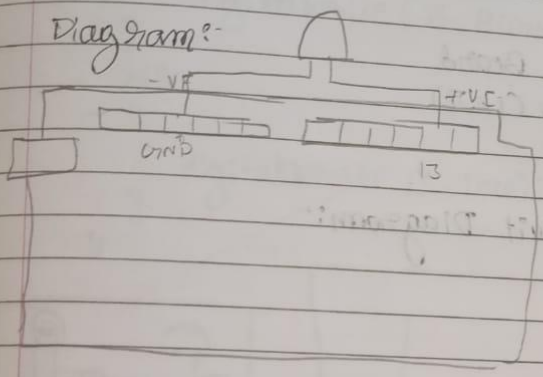
```
const int potPin = A0; // Pin connected to the potentiometer
const int ledPin = 9; // Pin connected to the LED
void setup() {
pinMode(ledPin, OUTPUT); // Initialize the LED pin as an output
}
void loop() {
int potValue = analogRead(potPin); // Read the value from the potentiometer (0-1023)
int brightness = map(potValue, 0, 1023, 0, 255); // Map the potentiometer value to brightness (0-255)
analogWrite(ledPin, brightness); // Set the brightness of the LED
}
```

Program 2:

AIM:- To observe the fading of intensity of light.

Components:- Arduino Board, LED, USB cable.

Diagram:-




```
void setup()
{
  PinMode(3, OUTPUT)
}
void loop()
{
  for(int i=0 ; i<255 ; i++)
  { analogWrite(3, i)
    delay(100);
  }

  for(int i=255 ; i>0 ; i-=5){
    analogWrite(3, i);
    delay(100);
  }
}
```

5. Nightlight Simulation

Aim: Simulating a night light using LDR and PIR

Hardware Required:

• 1 LED

• 1 LDR

• 110K register

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

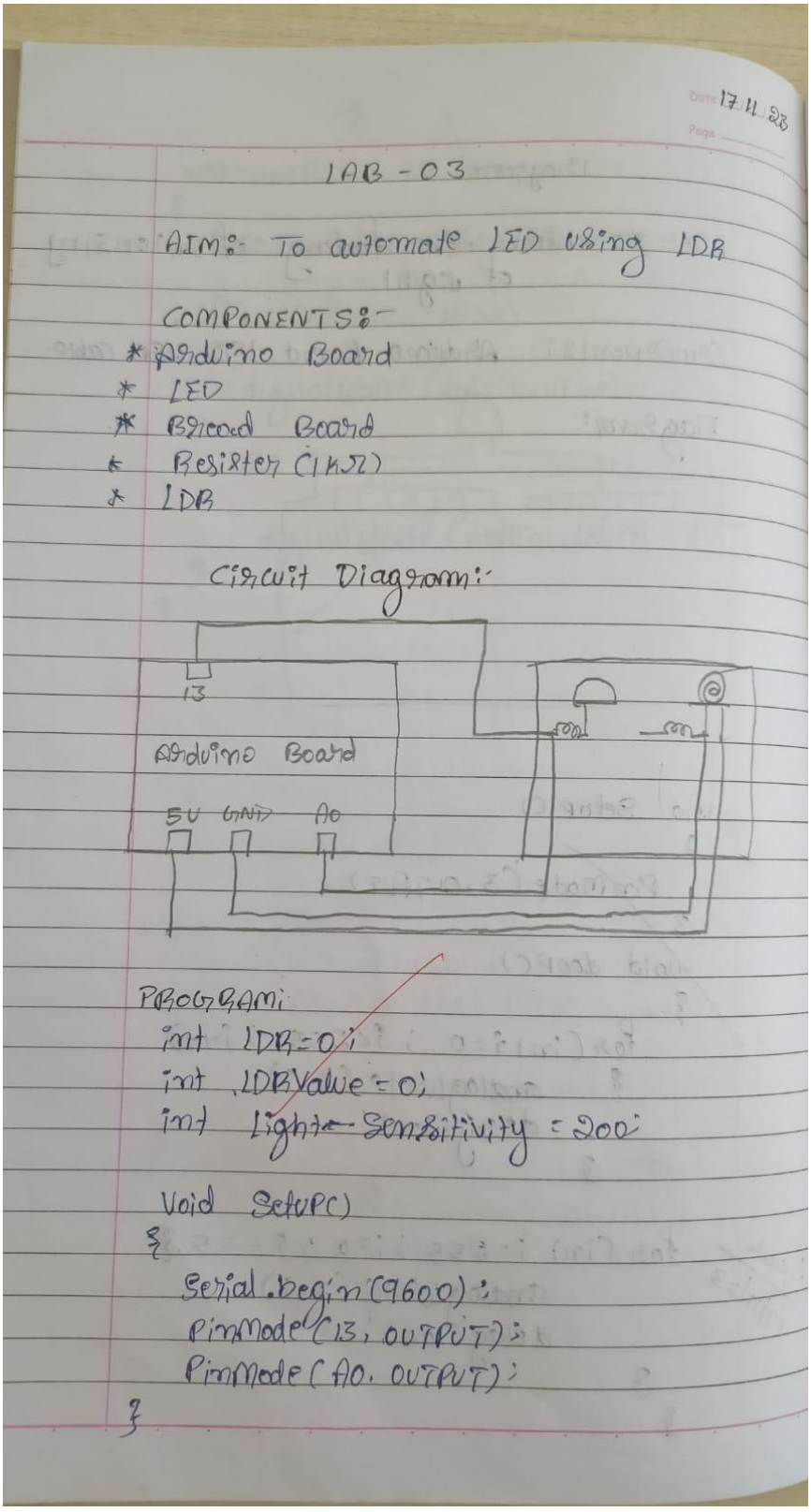
3. Attach another leg of register to the ground

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 buad
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);
}
```



Date 17.11.23
Page

LAB - 03

AIM: To automate LED using LDR

COMPONENTS:
* Arduino Board
* LED
* Bread Board
* Resister (1KΩ)
* LDR

Circuit Diagram:

PROGRAM:

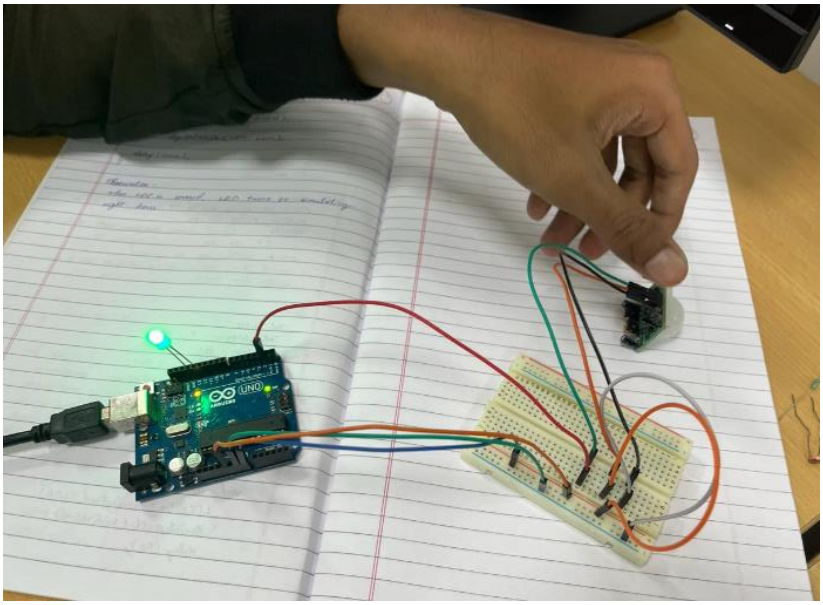
```
int LDR=0;
int LDRValue=0;
int LightSensitivity=200;

Void SetUP()
{
Serial.begin(9600);
PinMode(13, OUTPUT);
PinMode(A0, OUTPUT);
}
```

```
void ldrac()
{
  LDR value = AnalogRead (LDR);
  Serial. Println (LDRValue);
  if(LDR Value < digit. Sensitivity)
  {
     digitalWrite (13, HIGH);
  }
  else
  {
     digitalWrite( 13, low);
  }
}
```

6. PIR with Arduino UNO

Aim: To Simulate infrared detection using PIR sensor and arduino UNO.

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);

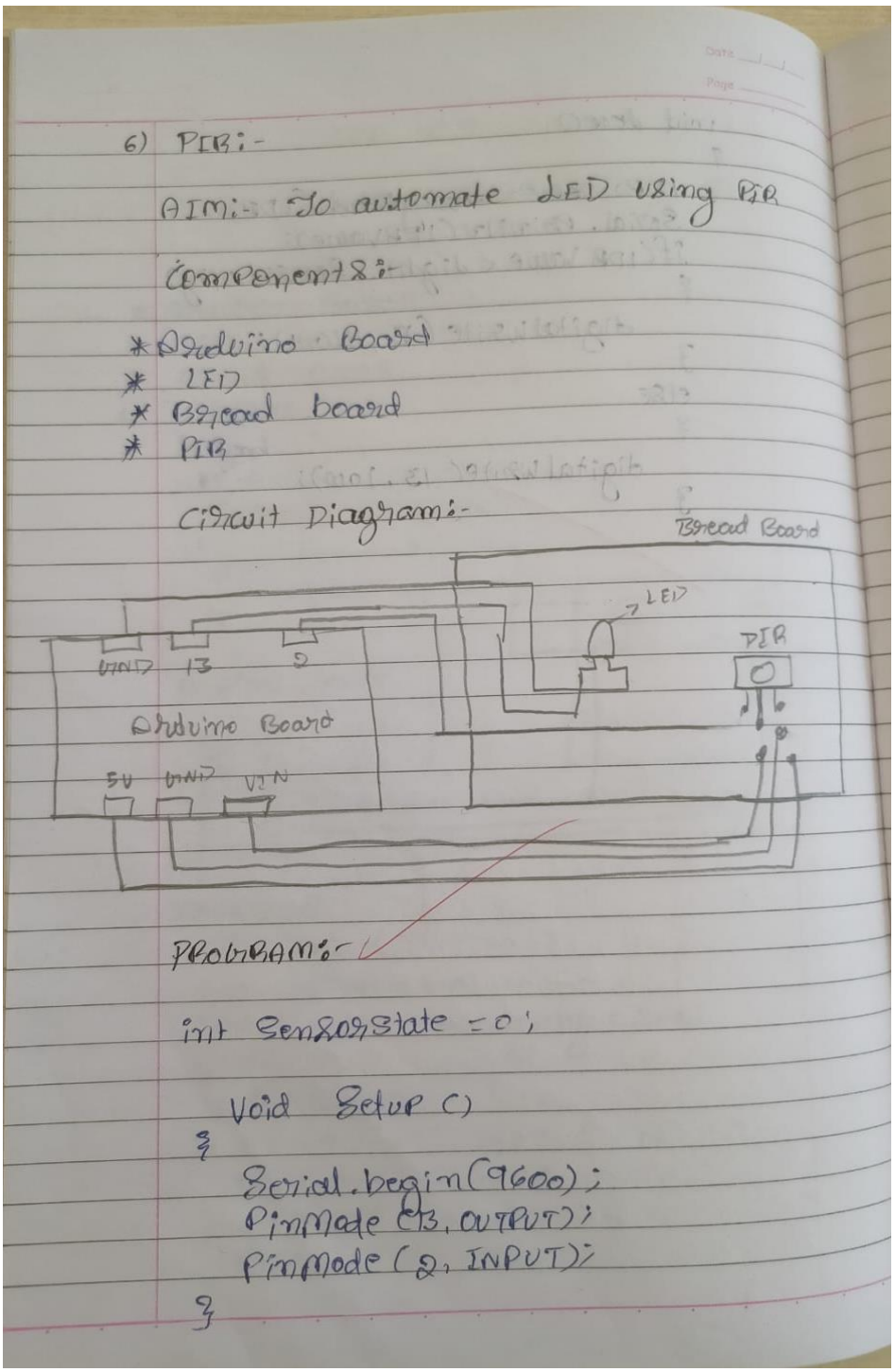
Serial.println("Sensor activated!");

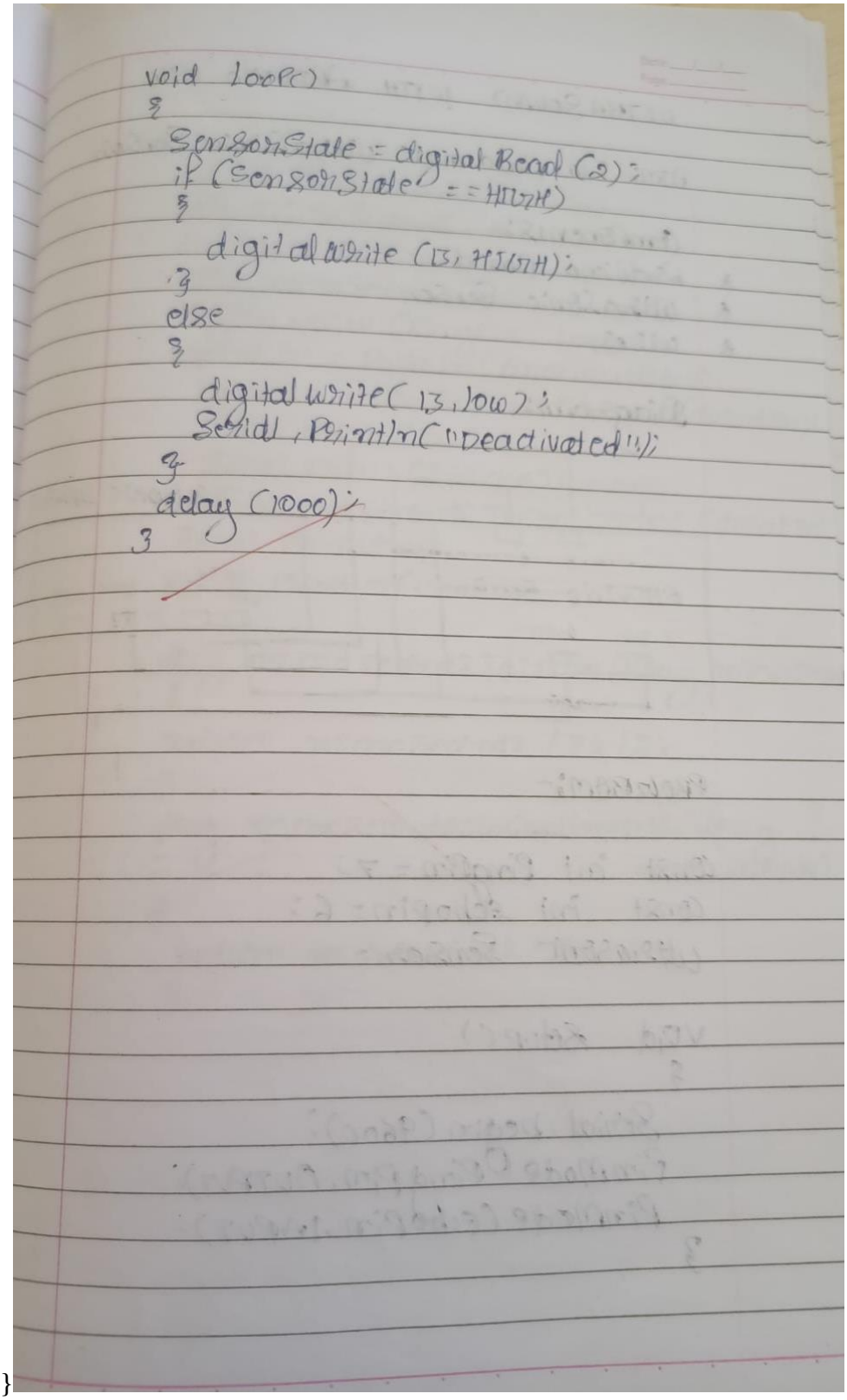
} else {

digitalWrite(13, LOW);

}

delay(10);





```
void loop()
{
  SensorState = digitalRead(2);
  if (SensorState == HIGH)
  {
    digitalWrite(13, HIGH);
  }
  else
  {
    digitalWrite(13, low);
    Serial.println("Deactivated");
  }
  delay(1000);
}
```

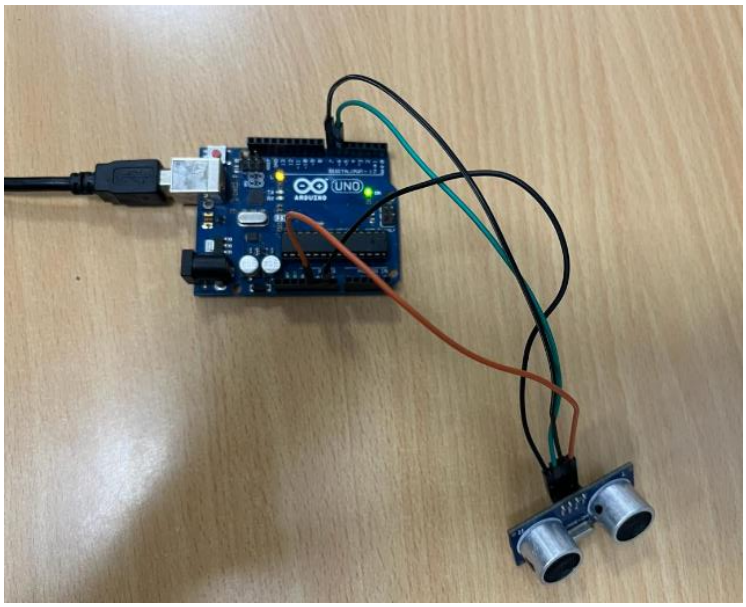
7. Ultrasound with Arduino UNO

Aim: To measure distance using ultrasound sensor and Arduino UNO.

Hardware Required:

- Ultrasound sensor
- Arduino UNO
- Jump/Connection wires

Connection:



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);
}
```

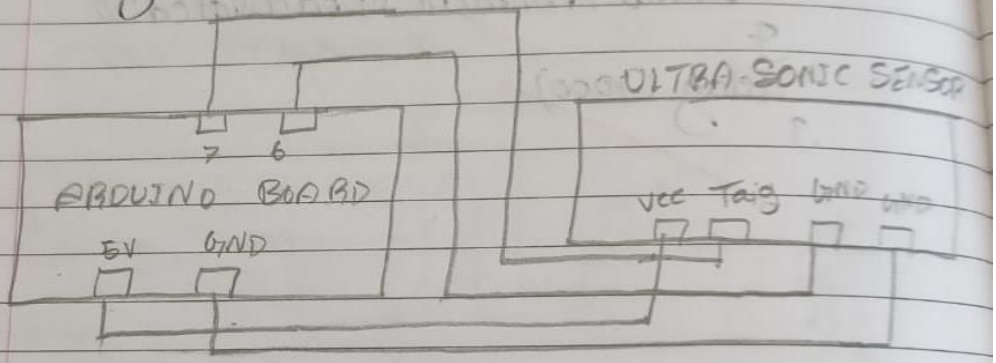
```
void loop() {
long duration, inches, cm;
digitalWrite(pingPin, LOW);
delayMicroseconds(2);
digitalWrite(pingPin, HIGH);
delayMicroseconds(10);
digitalWrite(pingPin, LOW);
duration = pulseIn(echoPin, HIGH);
inches = microsecondsToInches(duration);
Serial.print(inches);
Serial.print("inches");
cm = microsecondsToCentimeters(duration);
Serial.print(cm);
Serial.println("cm");
}
long microsecondsToInches(long microseconds) {
return microseconds / 74 / 2;
}
long microsecondsToCentimeters(long microseconds) {
return microseconds / 29 / 2;
}
```

ULTRA SOUND WITH ARDUINO UNO

AIM: To Automate ultrasonic Sensor

Components:
* Arduino Board
* ultrasonic Sensor
* wires,

Diagram:-



PROGRAM:-

```
const int PingPin = 7;
const int EchoPin = 6;
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;
}
```

8. Fire Alert

Aim: Fire alarm simulation

Hardware Required:

- Flame sensor (Analogue Output)
- Arduino
- Bread board
- LED
- Buzzer
- Connecting wires

Connections:

- **Flame sensor interfacing to Arduino**

  Flame sensor to Arduino

  vcc to vcc

  gnd to gnd

  A0 to A0

- **Led interfacing to Arduino**

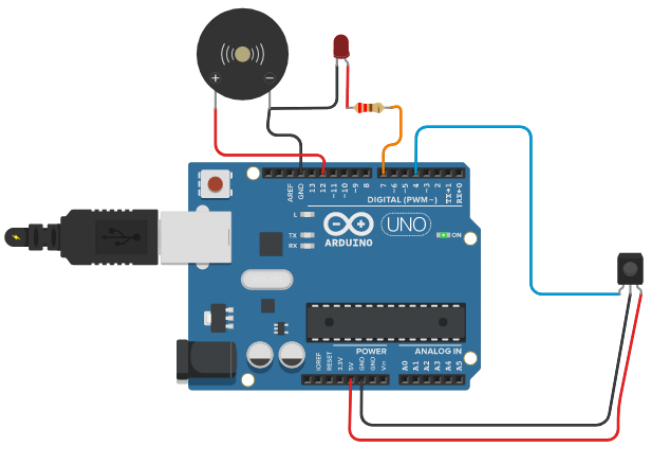
  LED +ve is connected to 9th pin of Arduino

  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:

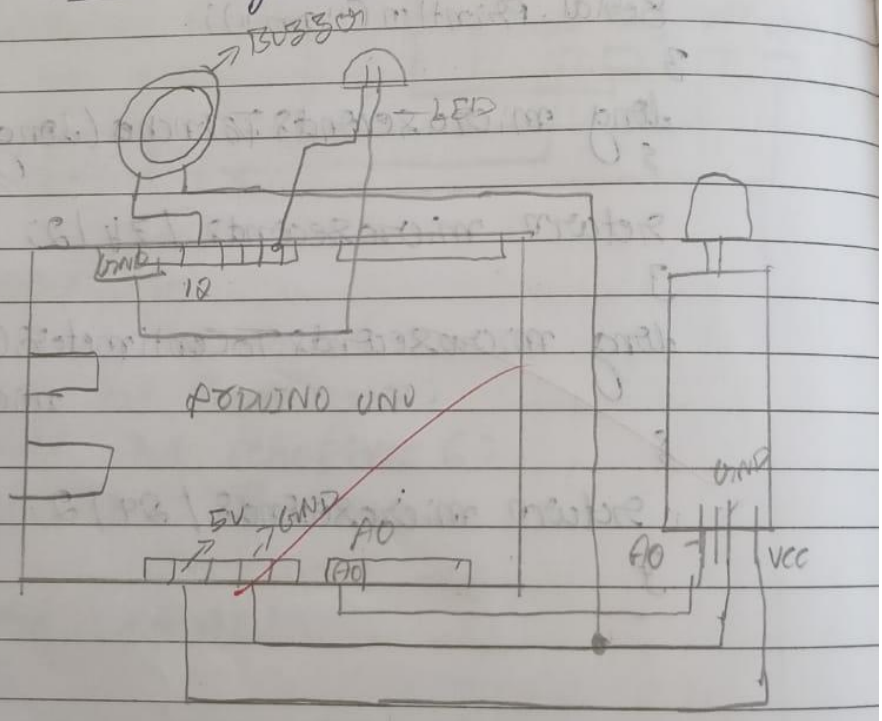
```
int sensorPin = A0; // select the input pin for the LDR
int sensorValue = 0; // variable to store the value coming from the sensor
int led = 9; // Output pin for LED
int buzzer = 12; // Output pin for Buzzer
void setup() {
// declare the ledPin and buzzer as an OUTPUT:
pinMode(led, OUTPUT);
pinMode(buzzer,OUTPUT);
Serial.begin(9600);
}
void loop()
{
sensorValue = analogRead(sensorPin);
Serial.println(sensorValue);
if (sensorValue &lt; 100)
{
Serial.println(&quot;Fire Detected&quot;);
Serial.println(&quot;LED on&quot;);
digitalWrite(led,HIGH);
digitalWrite(buzzer,HIGH);
delay(1000);
}
digitalWrite(led,LOW);
digitalWrite(buzzer,LOW);
delay(sensorValue);
}
```

FIRE ALERT

AIM:-
Fire alarm simulation

Hardware Required:-
Flame sensor (analog output)
Arduino
Bread board
LED
Buzzer
connecting wires

Circuit Diagram:-



Flame Detection using Arduino

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);
}
```

9. Automatic irrigation controller simulation

Aim:

Sensing the soil moisture and sprinkling the Water simulation

Hardware Required:

- Arduino
- Moisture Sensor
- Breadboard
- Min servo motor

Connections:

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)
```

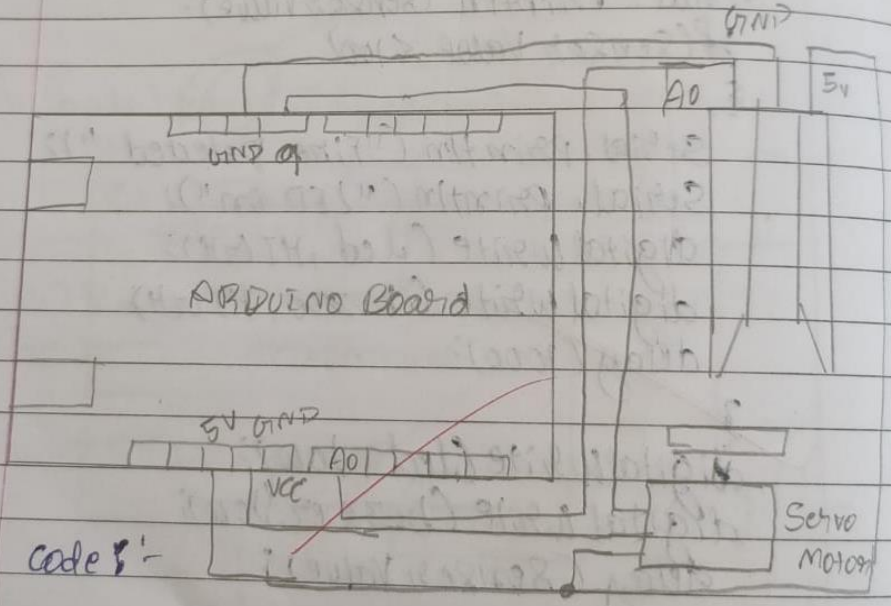
```
{
for (pos = 0; pos < 180; pos += 1) { // goes from 0 degrees to 180 degrees
// in steps of 1 degree
myservo.write(pos);
delay(15); // waits 15ms for the servo to reach the position
}
for (pos = 180; pos < 0; pos -= 1) { // goes from 180 degrees to 0 degrees
myservo.write(pos); // tell servo to go to position in variable &#39;pos&#39;
delay(15); // waits 15ms for the servo to reach the position
}
}
delay (1000);
}
```

AUTOMATIC IRRIGATION CONTROLLER SIMULATION

AIM:-
Sensing the soil moisture and Pumping the water Simulation

Hardware Required

Arduino
moisture Sensor
Breadboard
min Servo motor



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(){
sensor value = analogRead (SensorPin);
Serial.println (SensorValue);
if(Sensor value > 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);
}
```

Lavanya
22/12/23

10. Reading the code present on RFID tag

Aim:

The following code will read the code present on RFID tag and print it in serial monitor.

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);
}
}
```

RFID TAG READING

Aim:- The following code will read the code present on RFID tag and print it in serial monitor.

Connection:

5V - Arduino 5V
GND - Arduino GND
Tx - Pin 9

Code:

```
#include <SoftwareSerial.h>;
SoftwareSerial mySerial(9,10);
int count = 0;
char input[12];
boolcom 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 output in the Serial monitor is the RFID tag number, and it allows for real-time monitoring and verification of the data read from the RFID tag. The code, when executed continuously checks for available data on the software Serial Port, captures the RFID tag code and promptly displays it in the Serial monitor.

11. Access control through RFID

Aim:

The following code will 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.

Connection:

5V-Arduino 5V

GND-Arduino GND

Tx-pin 9

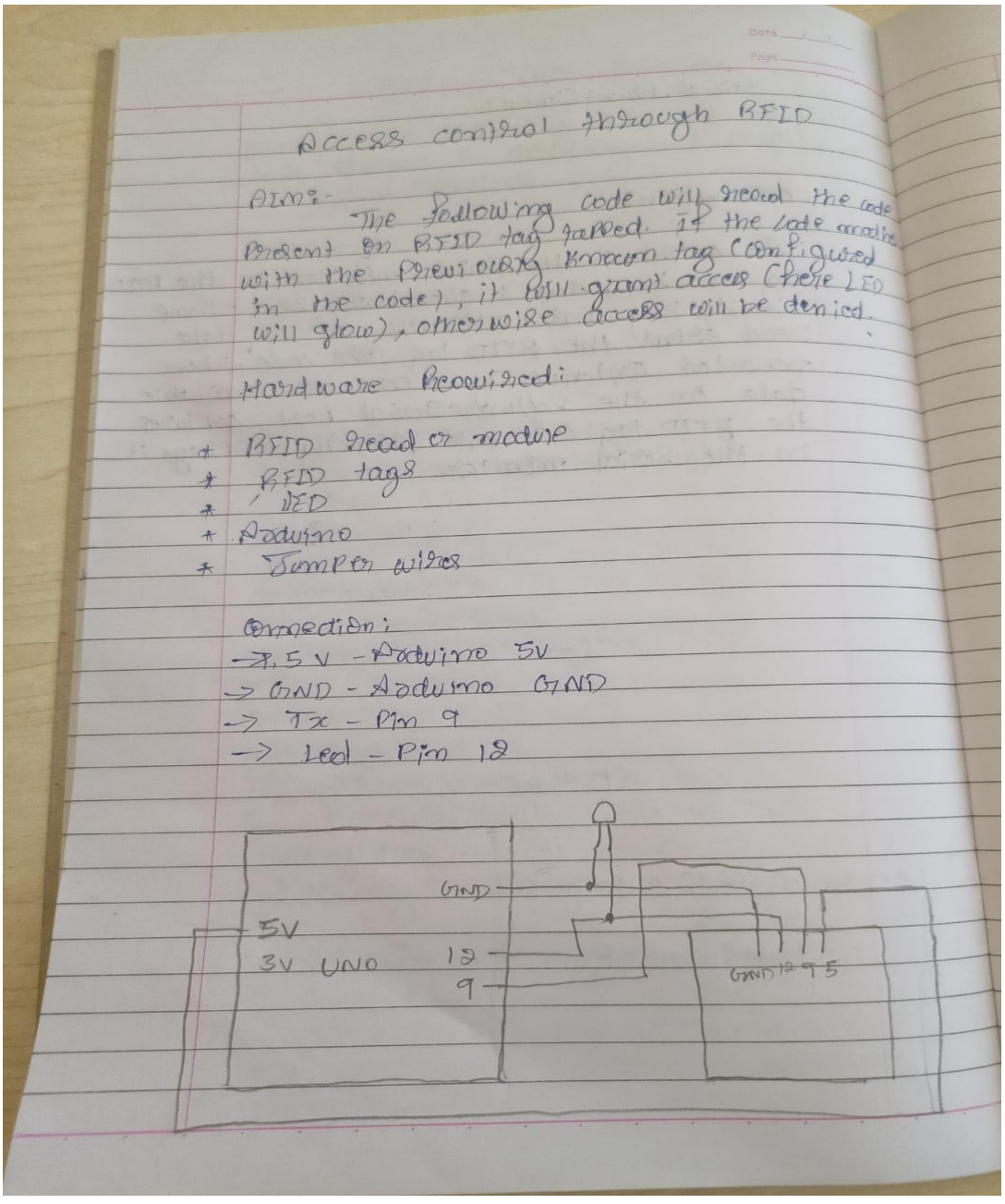
Led-pin 12

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);
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;
while(mySerial.available() &amp;amp;&amp;amp; count &amp;lt; 12)
{
input[count] = mySerial.read();
count++; // increment counter
delay(5);
}
if(count == 12)
{
count =0; // reset counter varibale to 0
flag = 1;
while(count<12 && flag !=0)
{
if(input[count]==tag[count])
flag = 1;
else
flag=0;
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);
}
for(count=0; count&lt;12; count++)
{
input[count]= &#39;F&#39;;
}
count = 0; // Reset counter variable
}
}
```



code:-

```
#include <SoftwareSerial.h>
SoftwareSerial myserial(9,10);
#define LEDPIN 13
char tag[] = "5300298DD087";
char word[12];
int count = 0;
character array
boolean flag = 0;

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

void loop()
{
 if(myserial.available())
   Buffer
 {
   count = 0;

while(myserial.available() && count < 12)
{
  input[count] = myserial.read();
  count++;
delay(5);
}
  if(count == 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<8;i++;i2;count++)
{
  input[count]=0x39,0x39;
}
count=0;
```

33

HC-05 Bluetooth Module

HC-05 PinOut (Right) :

• KEY: If brought HIGH before power is applied, forces AT Command Setup Mode.

LED blinks slowly (2 seconds)

• VCC: +5 Power

• GND: System / Arduino Ground

• TXD: Transmit Serial Data from HC-05 to Arduino Serial Receive. NOTE: 3.3V

HIGH level: OK for Arduino

• RXD: Receive Serial Data from Arduino Serial Transmit

• STATE: Tells if connected or not

12. HC-05 at Command prompt:

Code:

(For this program to work, HC-05 must be in command mode)

```
#include <SoftwareSerial.h>;
SoftwareSerial BTSerial(10, 11); // RX | TX
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());
}
```

HC-05 Bluetooth Module

HC-05 PinOut(Right):

- KEY : If brought HIGH before Power is applied, forced AT Command Setup mode, LED blinks slowly (2 seconds)
- VCC : +5 Power
- GND : System/Arduino Ground
- TXD : Transmit Serial Data from HC-05 to Arduino Serial Receive, NOTE: 3.3V HIGH level : OK for Arduino
- RXD : Receive Serial Data from Arduino Serial Transmit
- STATE: Tells if connected or not.

HC-05 at command Prompt

Aim:-
The following code will help establish communication between arduino board and HC-05 Bluetooth module

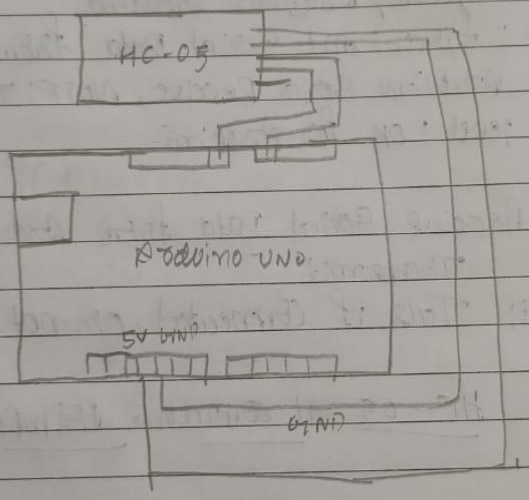
Hardware Required:

- HC-05 Bluecthooth module
- Arduino UNO
- Jumper wires.

Connections.

1. VCC of Bluetooth to 5v of arduino.
   GND of Bluetooth
2. GND of Bluetooth to Ground of arduino
3. TXD of bluetooth of Rx of arduino
4. RXD of bluetooth of Tx of arduino

Circuit Diagram:-



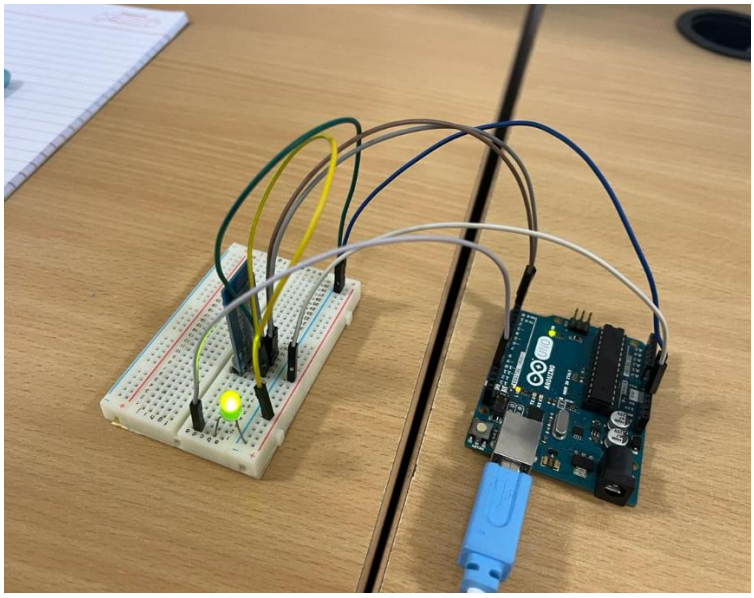
Code:-

```
#include <SoftwareSerial.h>
SoftwareSerial BTSerial(10,11);

void setup()
{
  Serial.begin(9600);
  Serial.println("Enter AT commands:");
  BTSerial.begin(38400);
}
```

```
void loop()
{
  if(BTSerial.available())
  Serial.write(BTSerial.read());

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

13. HC-05 Controlled by mobile

Connection:



Code:

(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){
// Checks whether data is comming from the serial port
state = Serial.read(); // Reads the data from the serial port
}
if (state == “0”) {
digitalWrite(ledPin, LOW); // Turn LED OFF
Serial.println(“LED: OFF”);
```

```
state = 0;
}
else if (state == “1”) {
digitalWrite(ledPin, HIGH);
Serial.println(“LED: ON”);;
state = 0;
}
}
```

HC-05 Controller by Mobile

Aim: To control an LED using a Bluetooth module in data mode, with commands sent from an Arduino Bluetooth app

Code:

```
#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;    } }
```

14. BT-Master Slave

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() {
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){
message = BTSerial.readString();
if(message.indexOf(“SWITCH ON”)<=0)
{
digitalWrite(ledPin, HIGH); // LED ON
}
else if(message.indexOf(“SWITCH OFF”)<=0)
{
digitalWrite(ledPin, LOW); // LED OFF
}
delay(100);
}
delay(10);
}
```

BT-MASTER SLAVE

Aim: To establish communication between a Bluetooth master device and a Bluetooth slave device to control an LED wirelessly.

Hardware Required:

BT-Slave

→ Arduino UNO
→ HC-05 Bluetooth Module
→ Jumper wires.

For BT-master

→ Arduino UNO
→ HC-05 Bluetooth Module
→ LED
→ Resistor
→ Jumper wires

Connections:

1) Bluetooth Slave (BT-Slave) Connections:

HC-05 Bluetooth Module:

* connect the Tx Pin to Arduino digital Pin 10
* connect the Rx pin to Arduino digital Pin 11
* connect the VCC Pin to Arduino 5V.
* connect the GND Pin to Arduino GND

2. Bluetooth Master (BT-Master) Connections:
* Connect the TX Pin to Arduino digital Pin 10.
* Connect the RX Pin to Arduino digital Pin 11.
* Connect the VCC Pin to Arduino 5V.
* Connect the GND Pin to Arduino GND.

3. LED and Resistor:
* Connect the anode to Arduino digital Pin 9.
* Connect the cathode of the LED to one end of a current-limiting resistor.
* Connect the other end of the resistor to Arduino GND.

Code:

```
#include<SoftwareSerial.h>
SoftwareSerial BTSerial(10,11);

void setup(){
Serial.begin(9600);
BTSerial.begin(38400);
}

void loop(){
if(Serial.available())
{
String message = Serial.readString();
BTSerial.write(message.c_str());
}
}
```

BT-Master Program:

```
#include <SoftwareSerial.h>
SoftwareSerial BTSerial(10, 11);

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

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

void loop() {
if (BTSerial.available() <0) {
message = BTSerial.readStringUntil();
if (BTSerial.message.indexOf("SWITCH ON")
                <=0)
{
digitalWrite(ledPin, HIGH);
}
else if (message.indexOf("SWITCH OFF")<=0)
{
digitalWrite(ledPin, low)
}
delay(100);
}
delay(100);
}
```

15. GSM Module

1. GSM Module: Call to a particular number

Aim:

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

Program:

```
#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() {
}
```

GSM module

GSM module : Call do a particular number

1) Aim :
Call using Arduino and GSM Module

Code :

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

void Setup () {
 cell.begin (9600);
 delay (500);
 Serial.begin(9600);
 Serial.Println("CALLING ....");
 cell.println ("ATD+9743699768;");
 delay (2000);
}

 void loop() {
}
```

2. Call to a particular number on an alert

Aim:

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

Connections for flame sensor:

Arduino Flame Sensor

5V VCC

GND

A0

Program:

```
#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&lt;50)
{
Serial.println(“CALLING.....”);
cell.println(“ATD+919742980606;”);
delay(10000);
cell.println(“ATH”); // Attention Hook Control
}
}
```

2) Aim:
Call a specified mobile number mentioned in the Program and GSM module when a flame sensor detects "fire".

Code:

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

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

void loop() {
 ival = analogRead(A0);
 Serial.println(val);
 delay(1000);
 if(val < 1;50)
 {
  Serial.println("CALLING....");
  cell.println("AT+97436997688");
  delay(10000);
  cell.println("ATH");
 }
}
```

3. Sending and Receiving Message

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.

Program:

Note: According to the code, message will be sent and received when ‘s’ and ‘r’ are pressed through serial monitor respectively.

```
#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());
}
voidSendMessage()
{
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=\"+919742980606\"\r"); // AT+CMGS, Send Message
// Replace withyour 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);
delay(1000);
}
voidRecieveMessage()
{
mySerial.println("AT+CNMI=2,2,0,0,0");
delay(1000);

}
```

5. AIM: Sending and Receiving Message:

```
#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.available())
                  read
   {
       case "s";
       send message();
       break;
      case "r";
      Receive Message();
      break;
   }
   if(mySerial.available()>0)
     Serial.write(mySerial.read());
   }

   Void Send Message()
   {
     mySerial.Println("AT+CMGF=1");
```

```
    delay(2000);
}
void Receive_Message()
{
    mySerial.println("AT+CNMI=2,2,0,0,0");
    delay(1000);
}
```

4. Controlling LED through received messages:

Aim:

Use received message through Arduino and GSM Module to control Switching ON / OFF the LED.

Connection: Attach LED to pin 13 and GND.

Program:

```
#include <SoftwareSerial.h>
SoftwareSerial cell(2,3);
Void readfn()
{
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();
//New SMS alert
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...”);
}
}
}
```

4. Aim: Controlling LED through received message:

code:

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

Void readfnc()
{
  if(Cell.available()){
  while(Cell.available()>0)
  Serial.write(Cell.read());
  }
}

void loop(){
if(Cell.available())
{
  String message = Cell.readString();
  if(message.indexOf("SWITCHON")>0)
  {
    digitalWrite(13,HIGH);
  }
else if(message.indexOf("SWITCHOFF")>0)
  {
    digitalWrite(13,LOW);
  }
}
}
```