MAHARAJA INSTITUTE OF TECHNOLOGY MYSORE
BELAWADI, SRIRANGAPATNA TQ, MANDYA-571477

Department of Master of Computer Applications

Internet of Things Lab – A section			
Sub Code: M23MCAL308		**CIE Marks:50**	
Teaching Hours/Week (L:P: SDA): 0.2.0		**Exam Hours: 3**	
Total Hours:		**SEE Marks: 50**	
CREDITS-01			
Sl. no	**Topic Description**	**Proposed date of coverage**	**Actual date of coverage**
1	Study the fundamentals of IOT software's and components	09/01/2025	
2	Familiarization with the detailed concepts of Arduino / Raspberry Pi.	16/01/2025	
3	i. Interfacing the Light Emitting Diode (LED) ii. Interfacing the RGB LED with the Arduino iii. Traffic Signal light Simulation	23/01/2025	
4	i. Interfacing of temperature sensor LM35 with Arduino ii. Interfacing Servo Motor with the Arduino	30/01/2025	
5	i. Interfacing of the Active Buzzer with Arduino ii. Interfacing of the Relay with Arduino.	06/02/2025	
6	Calculate the distance to an object with the help of an ultrasonic sensor and display it on an LCD	13/02/2025	
7	i. Controlling the LED blink rate with the potentiometer interfacing with Arduino ii. ON/OFF control based on light Intensity – using LDR sensor	20/02/2025	
8	Interfacing the regular USB webcam with the device and capture the image.	27/02/2025	
9	Build a circuit using Arduino based weather reporting over IOT	06/03/2025	

1. Study the fundamentals of IOT software's and components

1. IoT Software (Arduino & Raspberry Pi)

a. Operating Systems & Development Environment

- **Arduino** – Uses **Arduino IDE**, which supports C/C++. No operating system, just firmware.
- **Raspberry Pi** – Runs **Raspberry Pi OS (Linux-based)** and supports programming in Python, C++, Java, etc.

b. Communication Protocols (Used for data transfer)

- **Wired** – UART (Serial), SPI, I2C (for sensor communication).
- **Wireless** – Wi-Fi, Bluetooth, Zigbee, LoRa, MQTT (for internet/cloud).

c. IoT Programming Languages

- **Arduino** – C/C++ (written in Arduino IDE).
- **Raspberry Pi** – Python (mostly), also supports C, Java, and JavaScript.

d. Cloud & Data Storage

- **Arduino** – Sends data to the cloud using Wi-Fi/GSM modules (e.g., Firebase, ThingSpeak).
- **Raspberry Pi** – Stores data locally or in the cloud (e.g., AWS IoT, Google Cloud IoT).

2. IoT Hardware Components (Arduino & Raspberry Pi)

a. Microcontrollers & Microcomputers

- **Arduino** – Small microcontroller board (e.g., Arduino Uno, Mega, Nano).
- **Raspberry Pi** – Mini-computer with full OS support (e.g., Raspberry Pi 4, 3B+, Zero).

b. Sensors & Actuators (For collecting and responding to data)

- **Common Sensors:**
  - Temperature (DHT11, LM35)
  - Motion (PIR, Ultrasonic)
  - Light (LDR)
  - Gas (MQ Series)
- **Common Actuators:**
  - Motors (Servo, DC Motor)
  - Relays (For switching high-power devices)
  - LEDs, Buzzers

c. Connectivity Modules (For IoT communication)

- **Wi-Fi Modules** – ESP8266, ESP32 (used with Arduino).
- **Bluetooth Modules** – HC-05, HC-06 (used with both).
- **LoRa/Zigbee Modules** – Used for long-range IoT networks.
- **GSM Modules** – SIM800L, SIM900 for cellular communication.

d. Power Management

- **Arduino** – Powered via USB or 9V battery.
- **Raspberry Pi** – Needs a 5V/3A power adapter.

2. Familiarization with the detailed concepts of Arduino / Raspberry Pi

Arduino Board

Arduino is a small electronic board used to build IoT and automation projects. It has a microcontroller (a tiny computer) that can read sensors, control lights, motors, and send data.

Common Arduino Boards

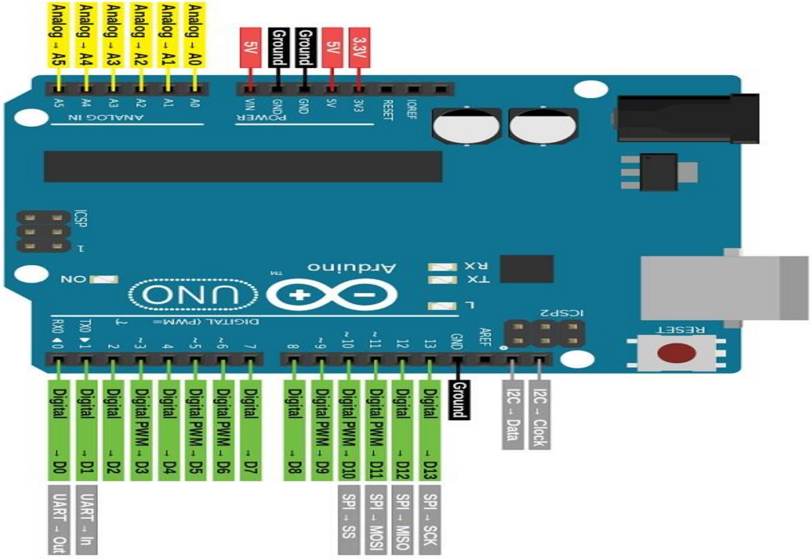
- **Arduino Uno** – Best for beginners, has 14 digital and 6 analog pins.
- **Arduino Mega** – More powerful, with 54 digital and 16 analog pins.
- **Arduino Nano** – A smaller version of Uno, useful for compact projects.
- **Arduino Leonardo** – Has built-in USB support (acts like a keyboard/mouse).
- **Arduino Due** – Faster, with a 32-bit microcontroller.

Key Parts of an Arduino Board

- **Microcontroller** – The brain (e.g., ATmega328P in Uno).
- **Digital & Analog Pins** – Connect sensors and actuators.
- **Power Port** – Runs on USB or battery (5V/9V).
- **USB Port** – Connects to a computer for programming.
- **Reset Button** – Restarts the board.

How to Use Arduino?

- Write code using the Arduino IDE (C/C++ based).
- Upload code via USB to the board.
- Connect sensors & actuators (LEDs, motors, etc.).
- Run the project!



Raspberry Pi

Raspberry Pi is a small **computer** (not just a microcontroller like Arduino) that can run an operating system, connect to the internet, and control hardware like sensors and motors. It's great for IoT, robotics, and AI projects.

Common Raspberry Pi Models

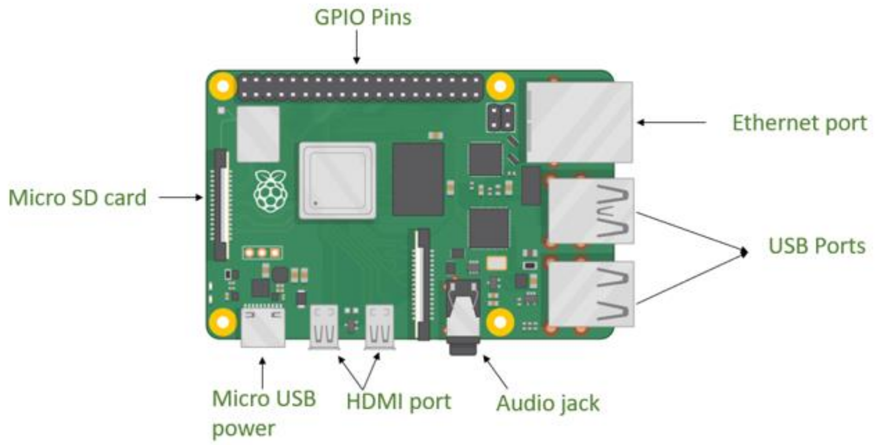
- **Raspberry Pi 4** – Most powerful, supports 4K video, Wi-Fi, Bluetooth.
- **Raspberry Pi 3B+** – Good for IoT projects, has built-in Wi-Fi and Bluetooth.
- **Raspberry Pi Zero** – Smallest, best for low-power IoT applications.
- **Raspberry Pi Pico** – A microcontroller (not a full computer) like Arduino.

Key Parts of Raspberry Pi

- CPU (Processor) – Works like a mini-computer.
- RAM – Temporary memory for running programs.
- GPIO Pins – Connect sensors, LEDs, and motors.
- USB & HDMI Ports – Connect a keyboard, mouse, and display.
- MicroSD Card Slot – Stores the operating system (e.g., Raspberry Pi OS).
- Wi-Fi & Bluetooth – Available in most models for wireless connections.

Key Parts of Raspberry Pi

- Install OS – Use Raspberry Pi OS on a microSD card.
- Connect to Display – Use HDMI to connect a monitor.
- Attach Keyboard & Mouse – Just like a regular computer.
- Write Code – Use Python, C, or Java for projects.
    - Control Electronics – Use GPIO pins to connect sensors and actuators.



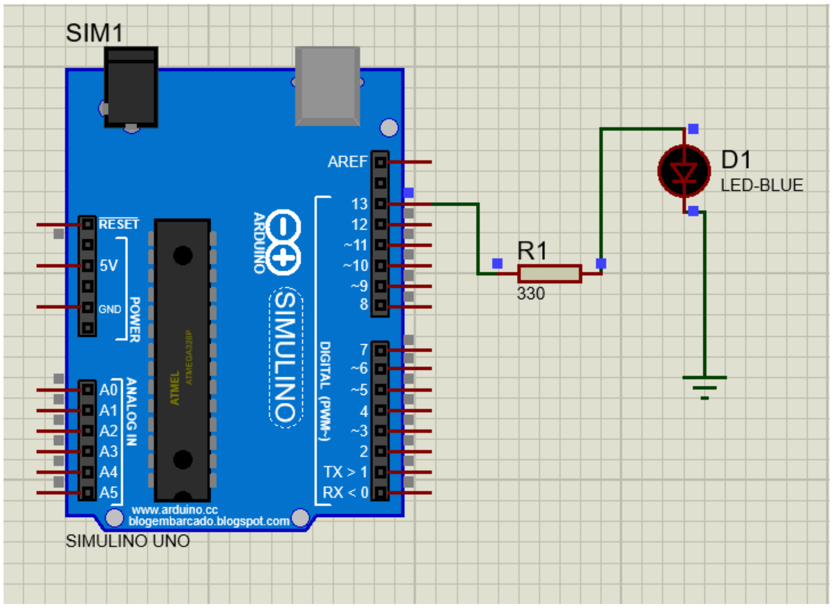
3. i. Interfacing the Light Emitting Diode (LED)
ii. Interfacing the RGB LED with the Arduino
iii. Traffic Signal light Simulation

i. Interfacing the Light Emitting Diode (LED)

Requirements

1. Arduino Uno
2. Led -1
3. Resistor – 330ohm
4. Connecting wires

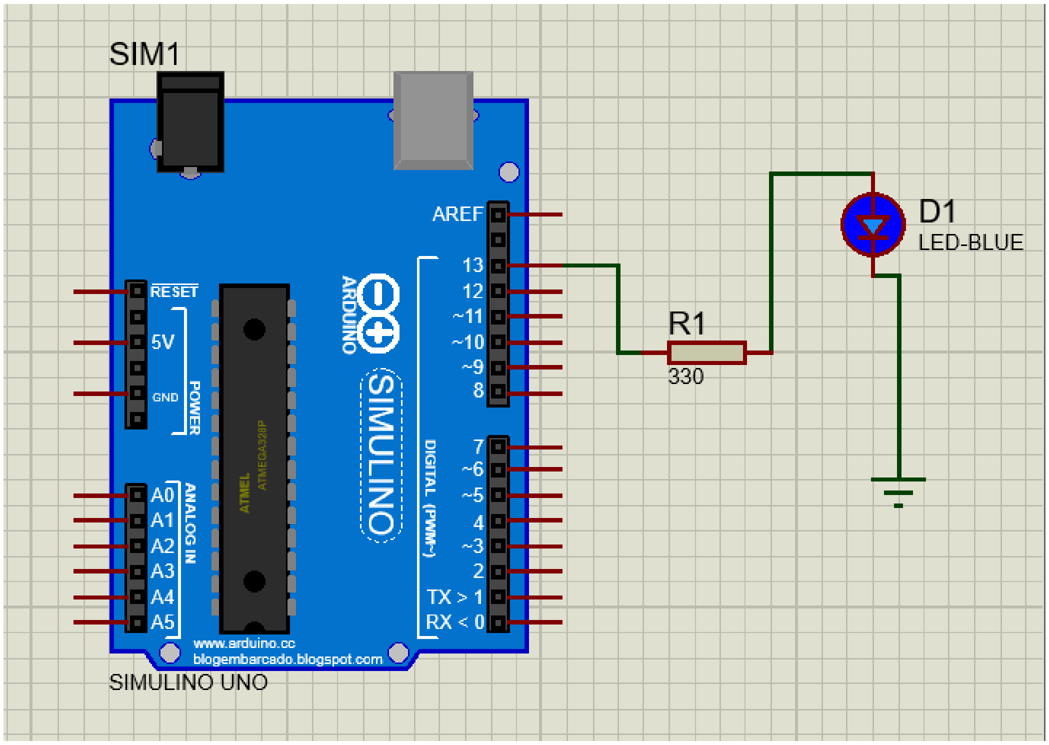
Circuit Connection



Program

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

OUTPUT

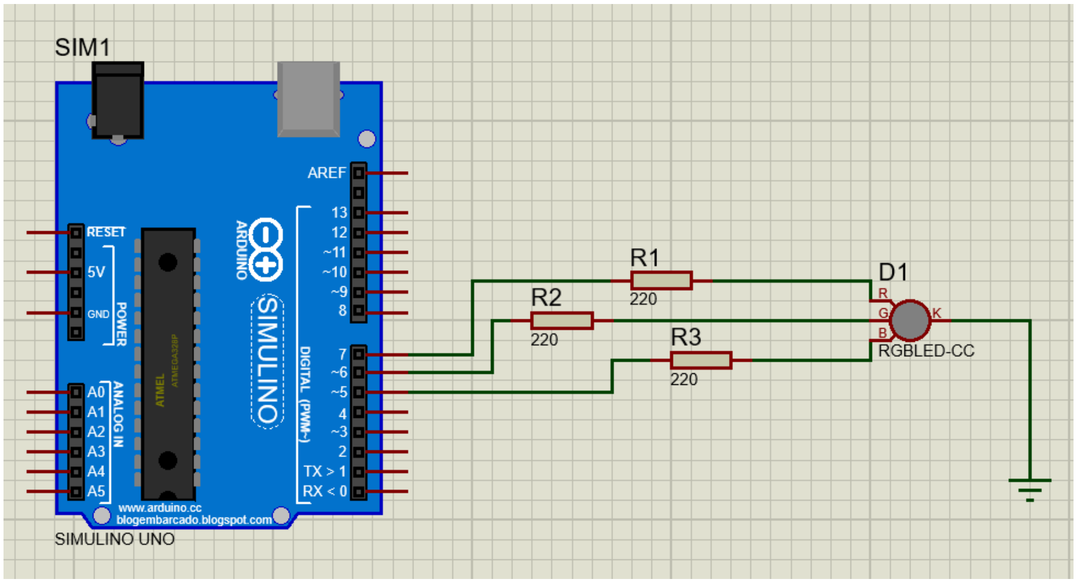


<u>ii. Interfacing the RGB LED with the Arduino</u>

Requirements

1. Arduino Uno
2. RGBLed -1
3. Resistor – 220 ohm – 3Nos
4. Connecting wires

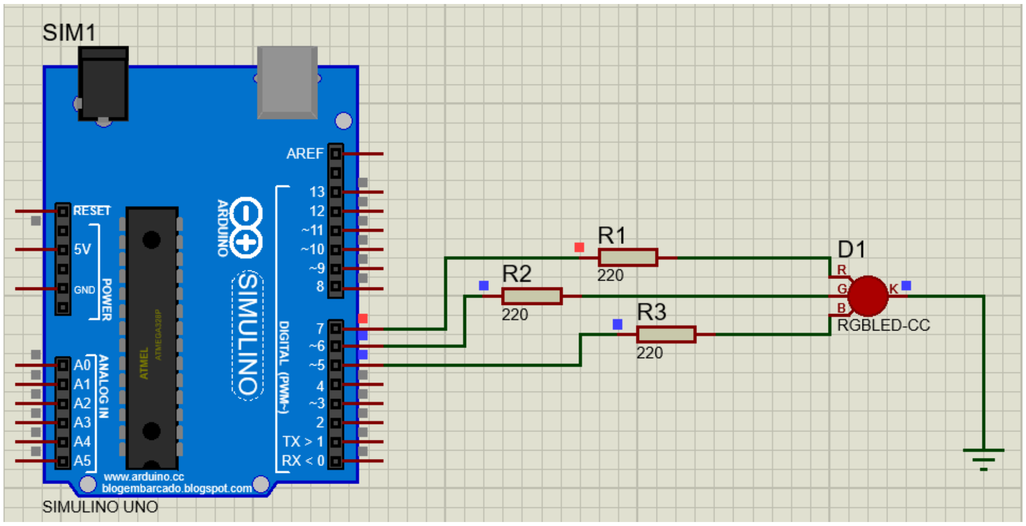
Circuit Connection

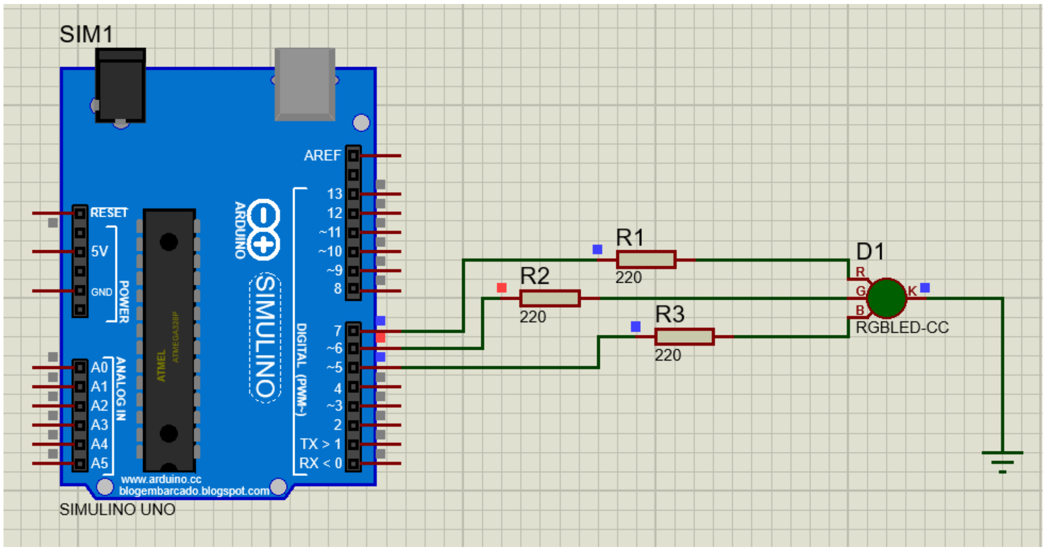


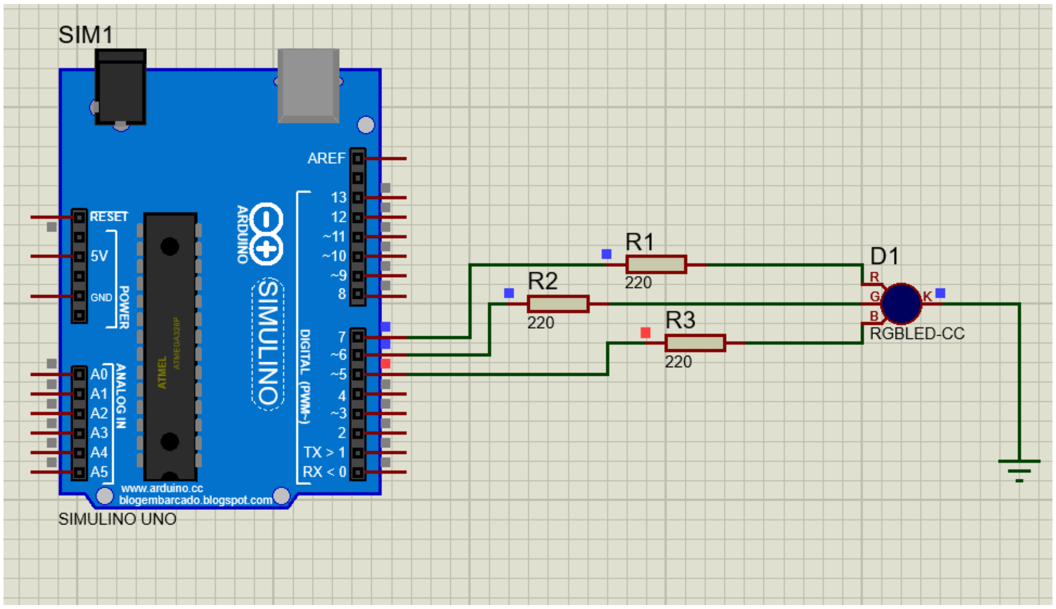
Program

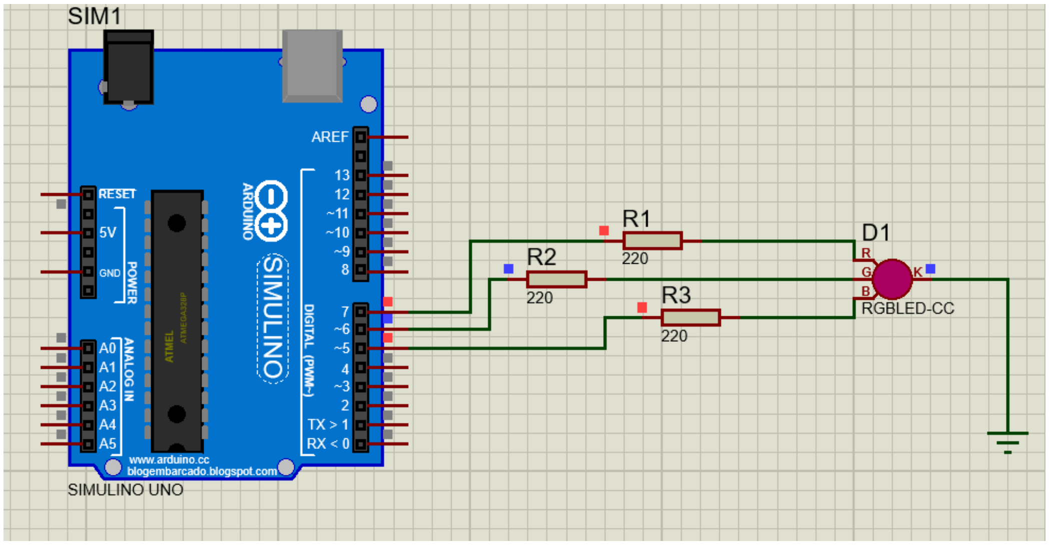
```
int redPin= 7;
int greenPin = 6;
int bluePin = 5;

void setup() {
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
}
void loop() {
  setColor(255, 0, 0); // Red Color
  delay(1000);
  setColor(0, 255, 0); // Green Color
  delay(1000);
  setColor(0, 0, 255); // Blue Color
  delay(1000);
  setColor(255, 255, 255); // White Color
  delay(1000);
  setColor(170, 0, 255); // Purple Color
  delay(1000);
}
void setColor(int redValue, int greenValue, int blueValue) {
  analogWrite(redPin, redValue);
  analogWrite(greenPin, greenValue);
  analogWrite(bluePin, blueValue);
}
```

OUTPUT


SIM1
AREF
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX > 1
RX < 0
RESET
5V
GND
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
ARDUINO
SIMULINO
DIGITAL (PWM~)
ATMEL
ATMEGA328P
www.arduino.cc
blogembarcado.blogspot.com
SIMULINO UNO
R1
220
R2
220
R3
220
D1
R
G
B
K
RGBLED-CC


SIM1
AREF
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX > 1
RX < 0
RESET
5V
GND
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
ARDUINO
SIMULINO
DIGITAL (PWM~)
ATMEL
ATMEGA328P
www.arduino.cc
blogembarcado.blogspot.com
SIMULINO UNO
R1
220
R2
220
R3
220
D1
R
G
B
K
RGBLED-CC


SIM1
AREF
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX > 1
RX < 0
RESET
5V
GND
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
ARDUINO
SIMULINO
DIGITAL (PWM~)
ATMEL
ATMEGA328P
www.arduino.cc
blogembarcado.blogspot.com
SIMULINO UNO
R1
220
R2
220
R3
220
D1
R
G
B
K
RGBLED-CC

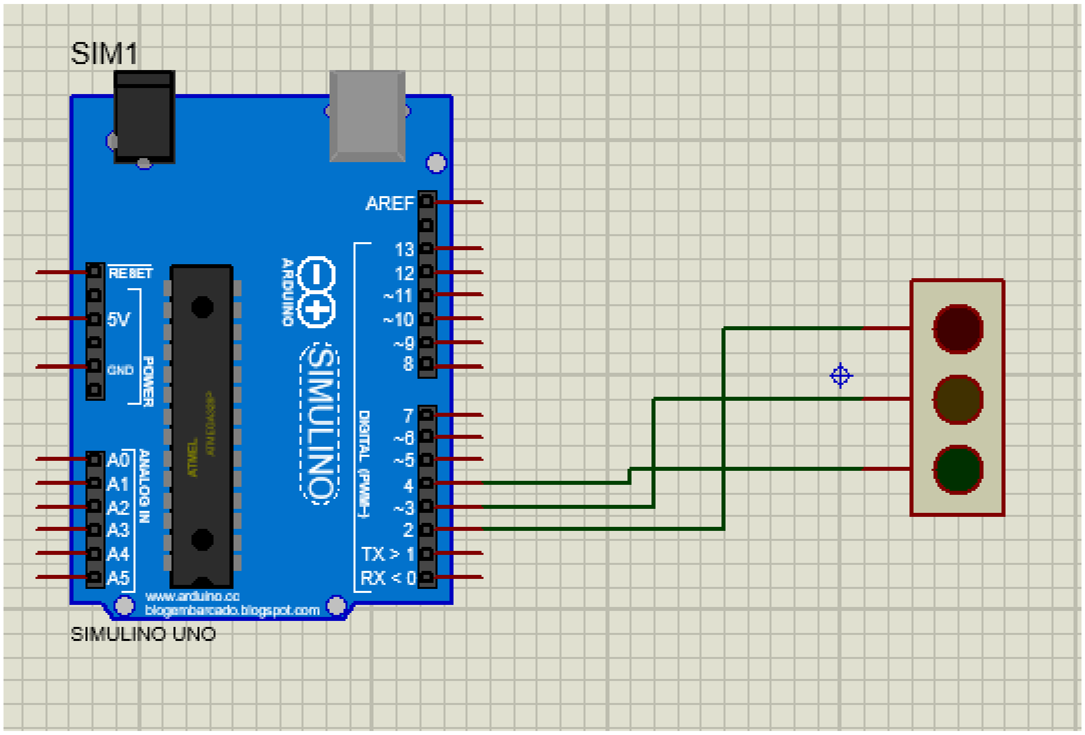


iii. Traffic Signal light Simulation

Requirements

1. Arduino Uno
2. Traffics Light LED -1
3. Connecting wires

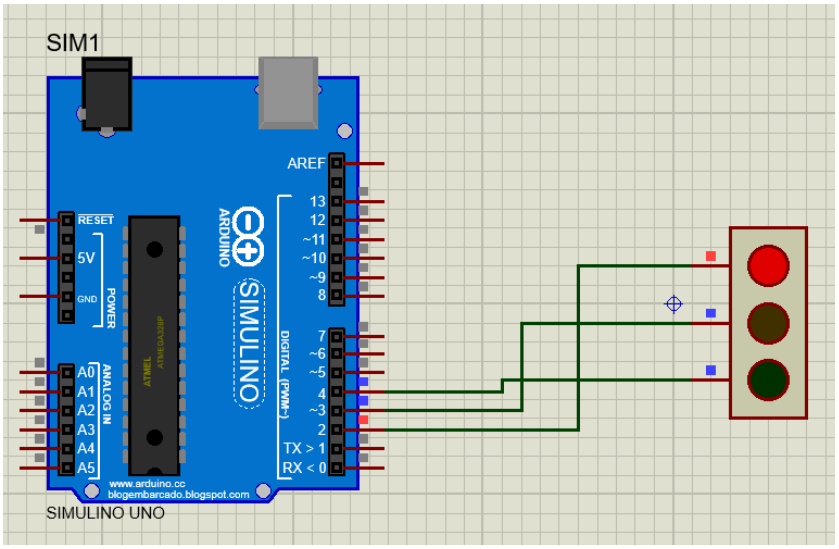
Circuit Connection

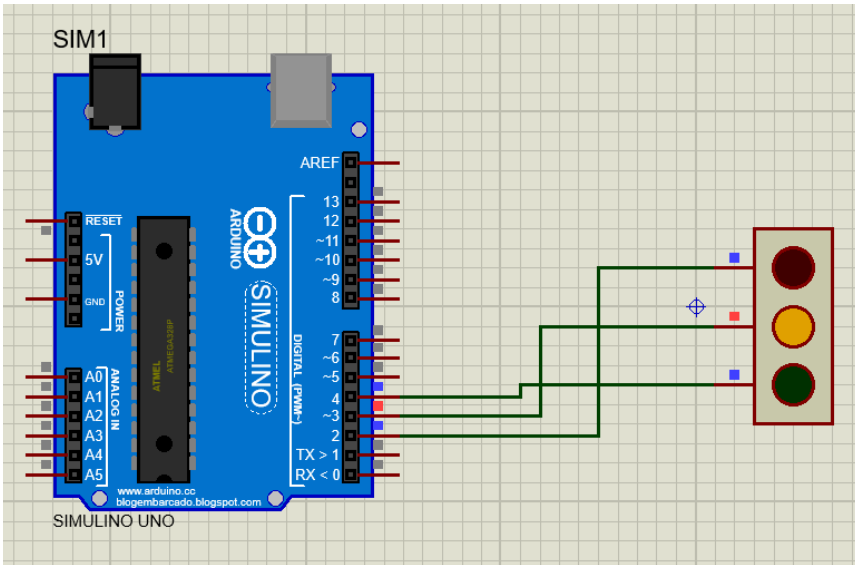


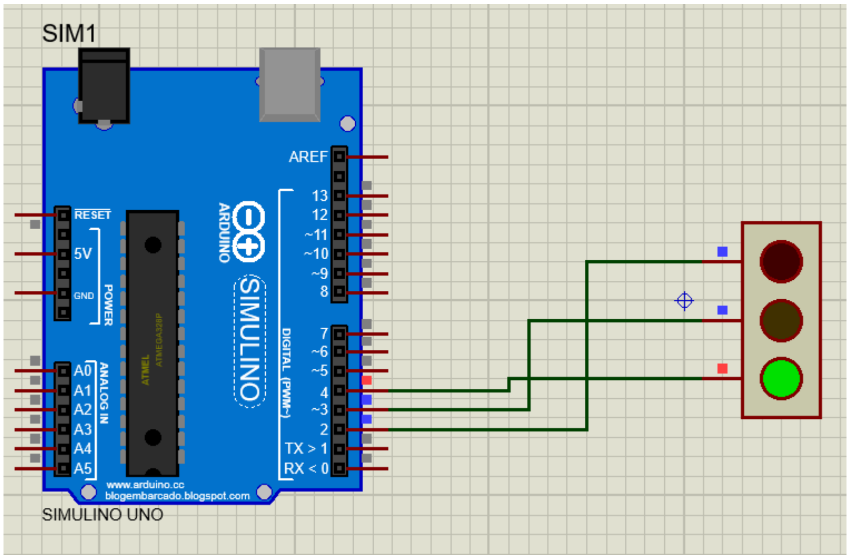
Program

```
int rled=2;
int yled=3;
int gled=4;
void setup() {
  // put your setup code here, to run once:
  pinMode(rled,OUTPUT);
  pinMode(yled,OUTPUT);
  pinMode(gled,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(rled,HIGH);
  digitalWrite(yled,LOW);
  digitalWrite(gled,LOW);
  delay(5000);

  digitalWrite(rled,LOW);
  digitalWrite(yled,HIGH);
  digitalWrite(gled,LOW);
  delay(2000);

  digitalWrite(rled,LOW);
  digitalWrite(yled,LOW);
  digitalWrite(gled,HIGH);
  delay(5000);
}
```

OUTPUT


SIM1
AREF
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX > 1
RX < 0
RESET
5V
GND
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
ARDUINO
SIMULINO
DIGITAL (PWM~)
ATMEL ATMEGA328P
www.arduino.cc
blogembarcado.blogspot.com
SIMULINO UNO


SIM1
AREF
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX > 1
RX < 0
RESET
5V
GND
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
ARDUINO
SIMULINO
DIGITAL (PWM~)
ATMEL ATMEGA328P
www.arduino.cc
blogembarcado.blogspot.com
SIMULINO UNO


SIM1
AREF
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX > 1
RX < 0
RESET
5V
GND
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
ARDUINO
SIMULINO
DIGITAL (PWM~)
ATMEL ATMEGA328P
www.arduino.cc
blogembarcado.blogspot.com
SIMULINO UNO

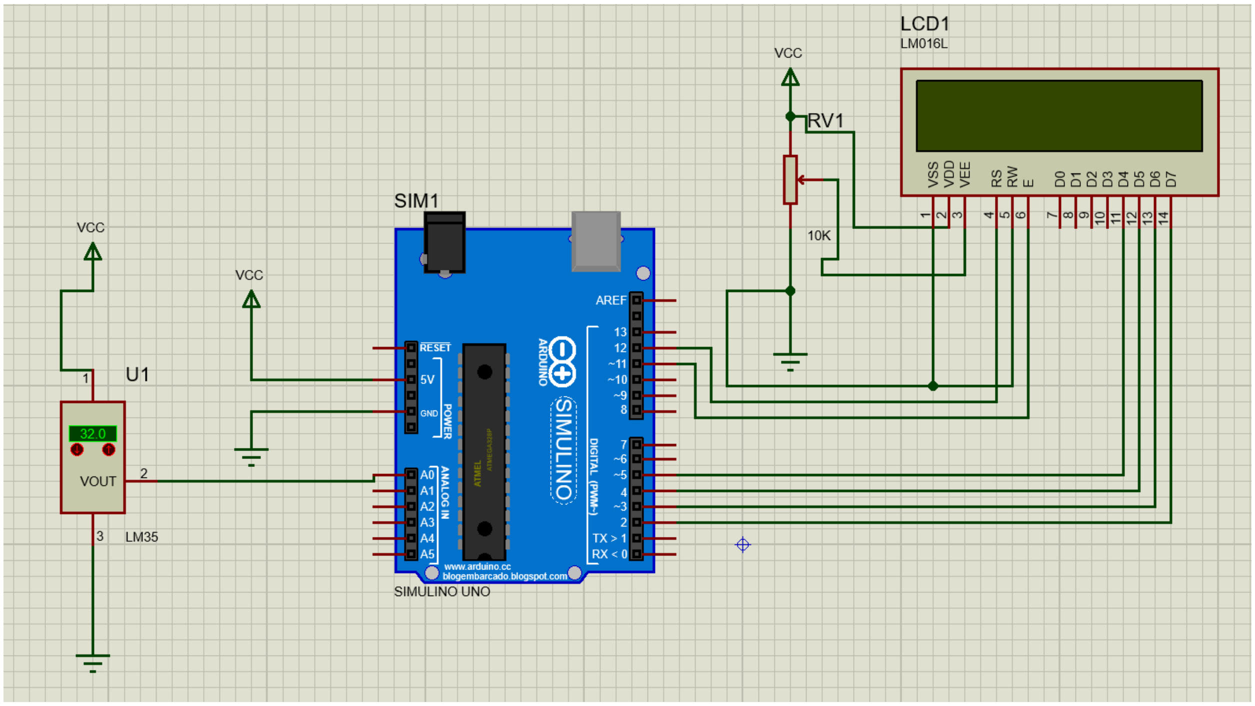
4 i. Interfacing of temperature sensor LM35 with Arduino
ii. Interfacing Servo Motor with the Arduino

i. Interfacing of temperature sensor LM35 with Arduino

Requirements

1. Arduino Uno
2. LM35 – Temperature Sensor
3. LCD – 16 x 2 Display
4. Resistor – 3005P-1-103
5. Connecting wires

Circuit Connection

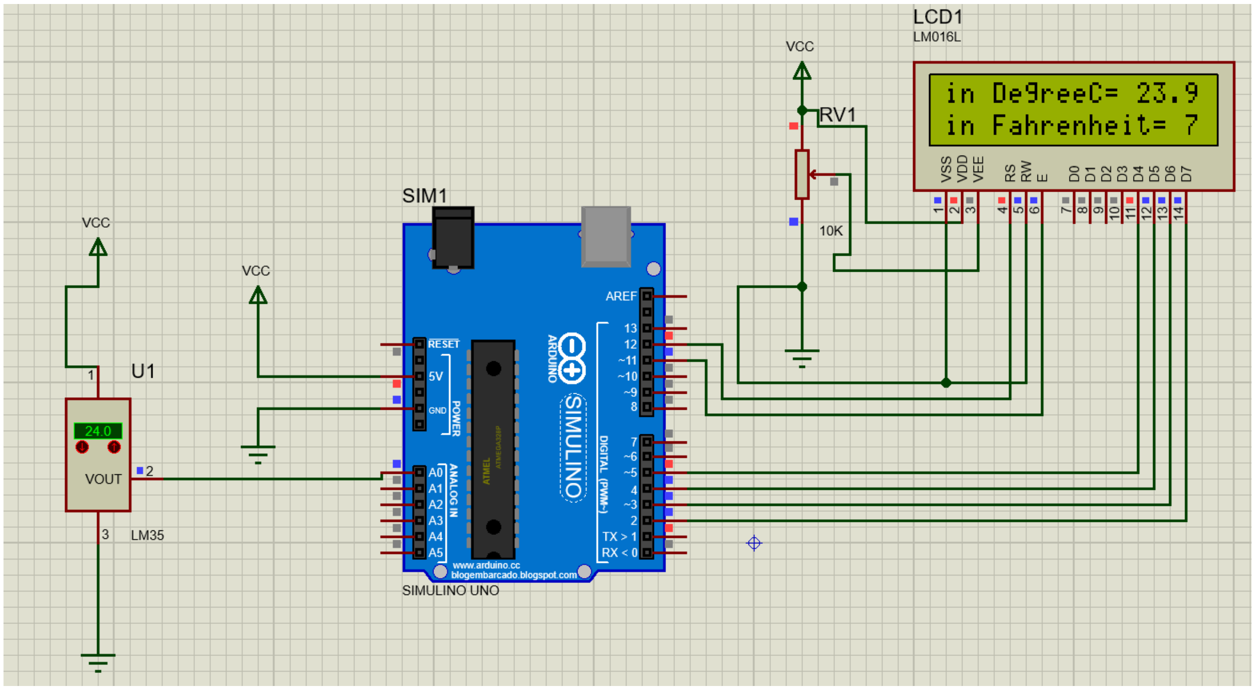


Program

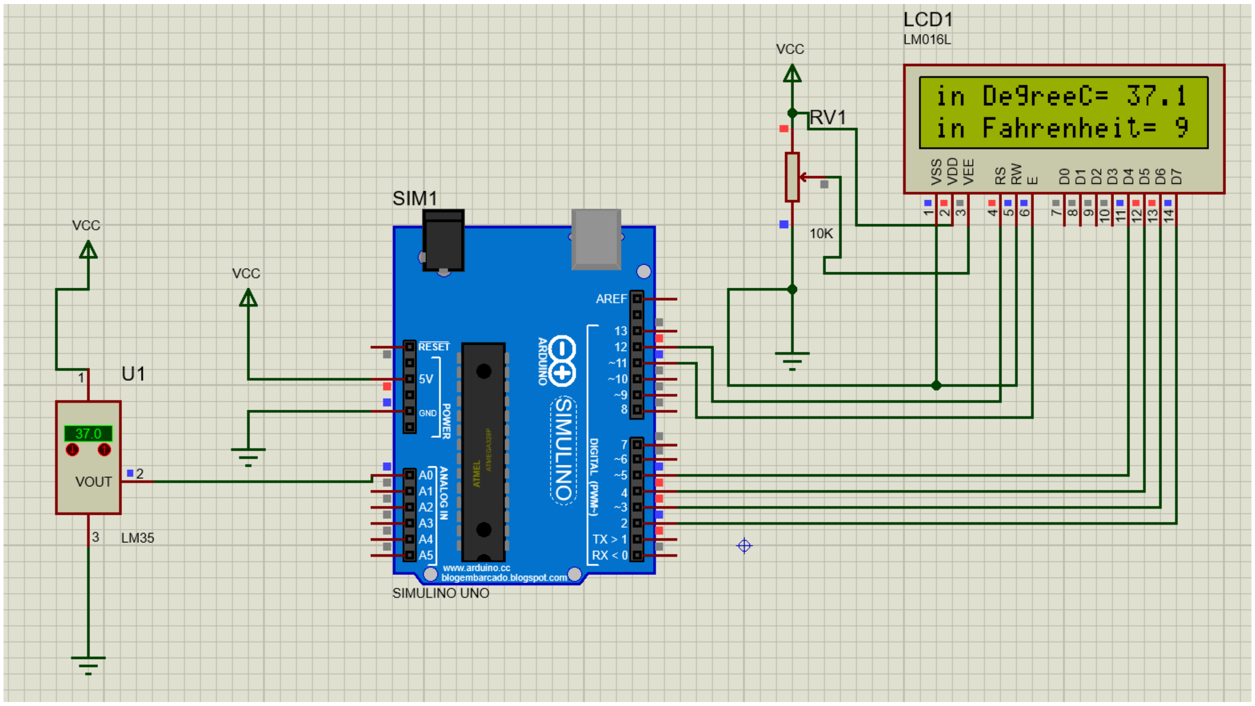
```
#include<LiquidCrystal.h>
LiquidCrystal lcd(12,11,5,4,3,2);
const int sensor=A0;
float tempc;
float tempf;
float vout;
```

```
void setup()
{
 pinMode(sensor,INPUT);
 Serial.begin(9600);
 lcd.begin(16,2);
   delay(500);
}
void loop()
{
vout=analogRead(sensor);
vout=(vout*500)/1023;
tempc=vout;
tempf=(vout*1.8)+32;
lcd.setCursor(0,0);
lcd.print("in DegreeC= ");
lcd.print(tempc);
lcd.setCursor(0,1);
lcd.print("in Fahrenheit= ");
lcd.print(tempf);
delay(1000);
}
```

OUTPUT



Increase or Decrease the Temperature values it will display an LCD Monitors

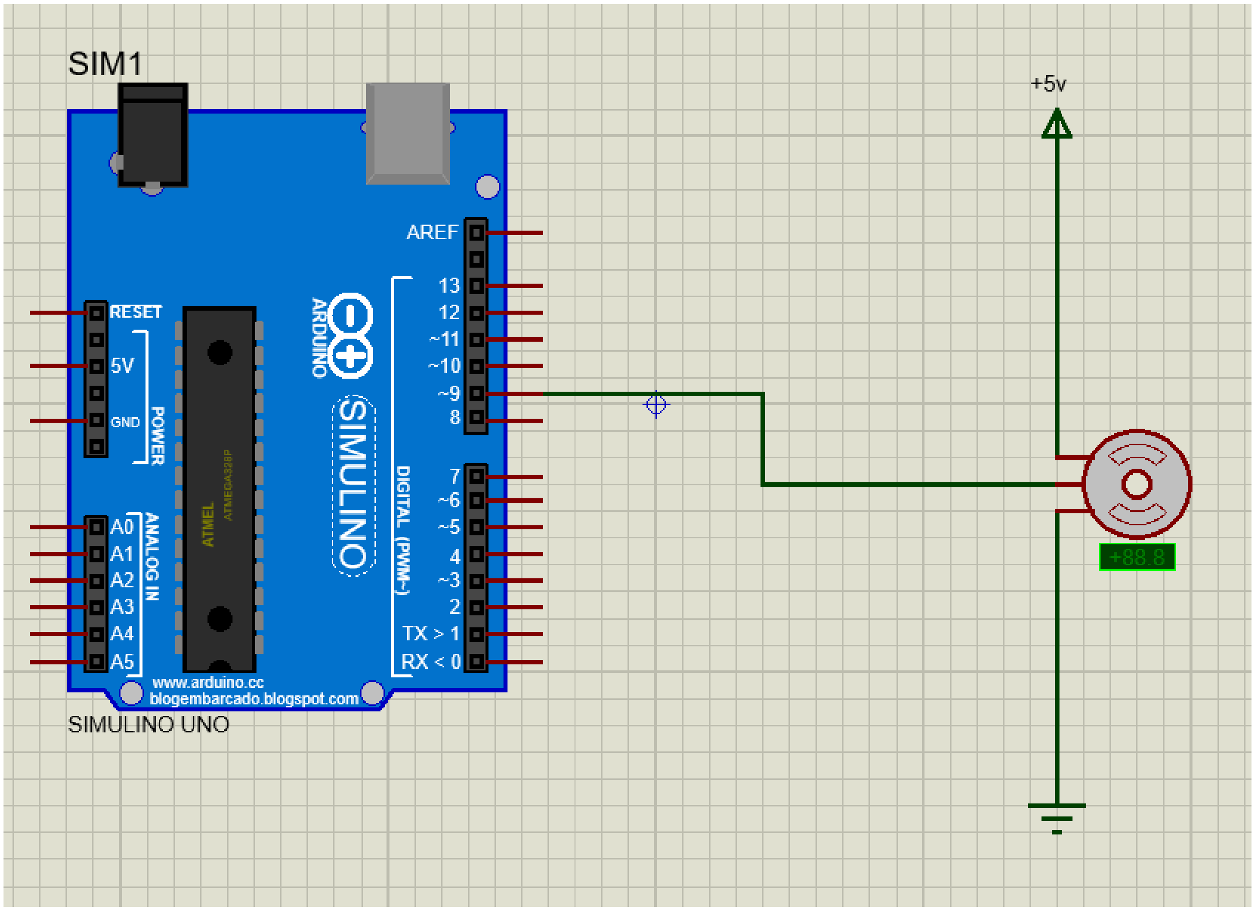


ii.. Interfacing Servo Motor with the Arduino

Requirements

1. Arduino Uno
2. Servo-Motor – PWMServo
3. Connecting wires

Circuit Connection



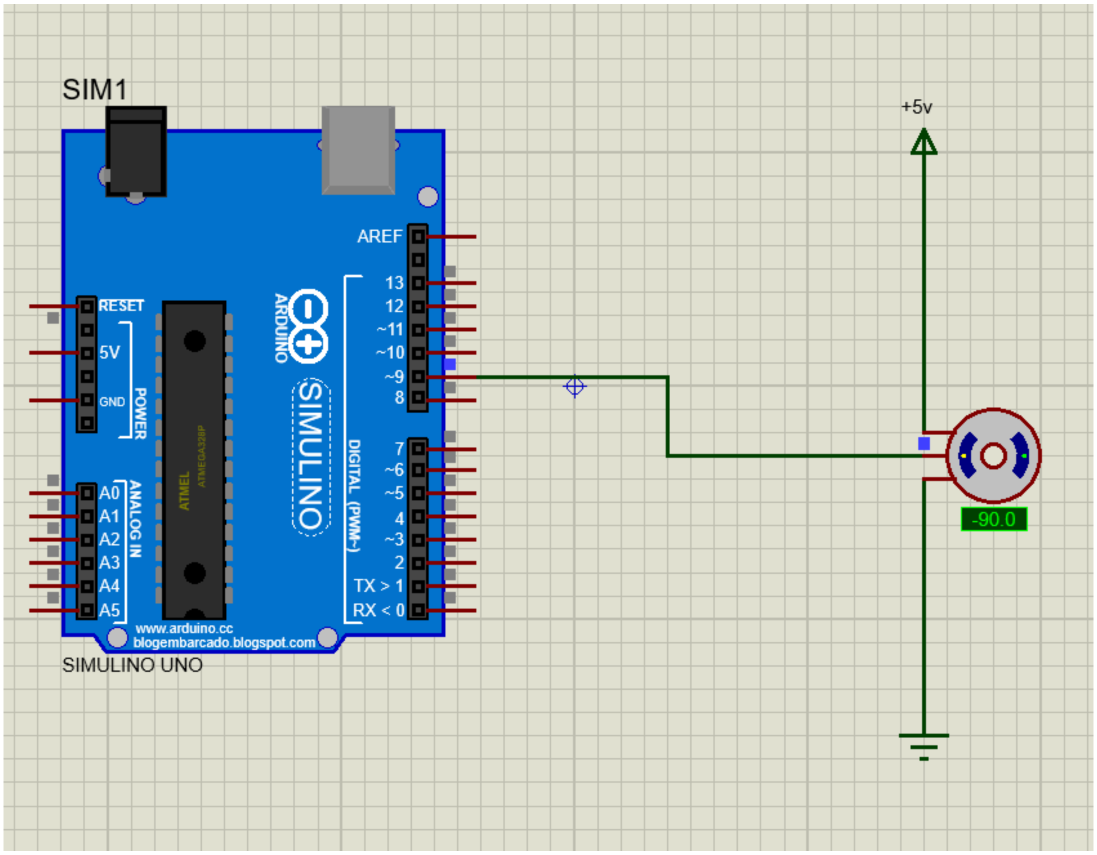
Program

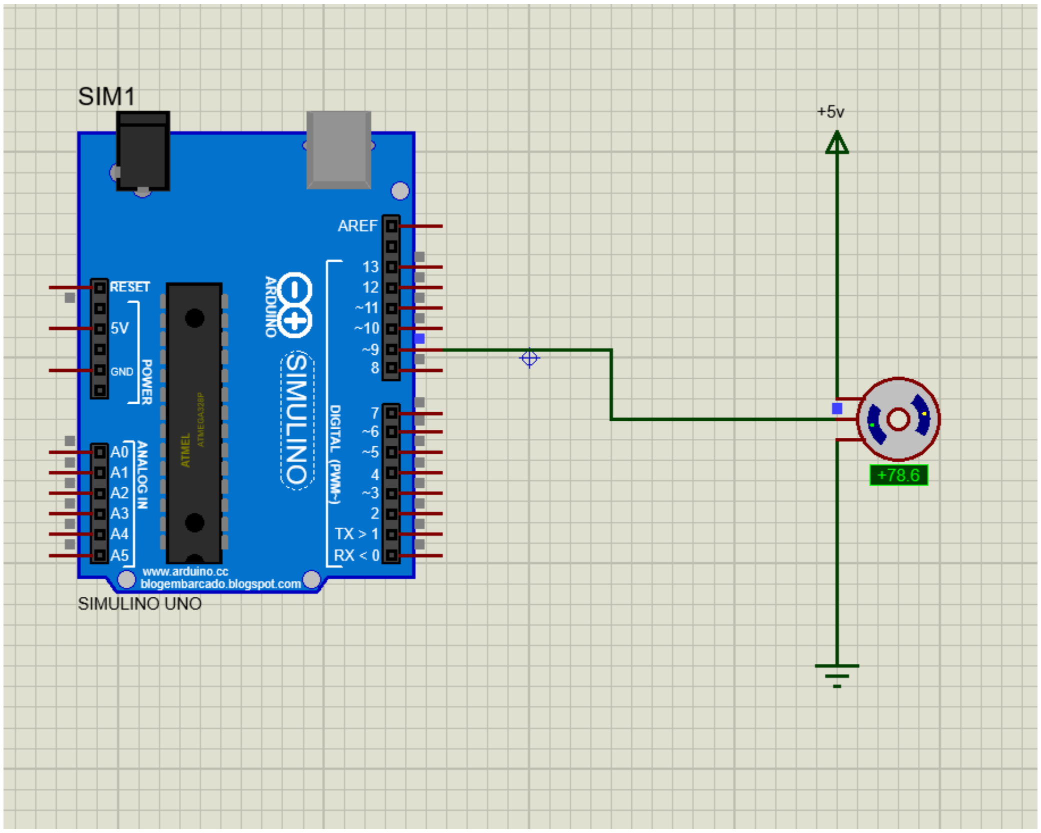
```
#include <Servo.h>
Servo myservo;
int pos = 0;
void setup() {
  // put your setup code here, to run once:
  myservo.attach(9);
}
```

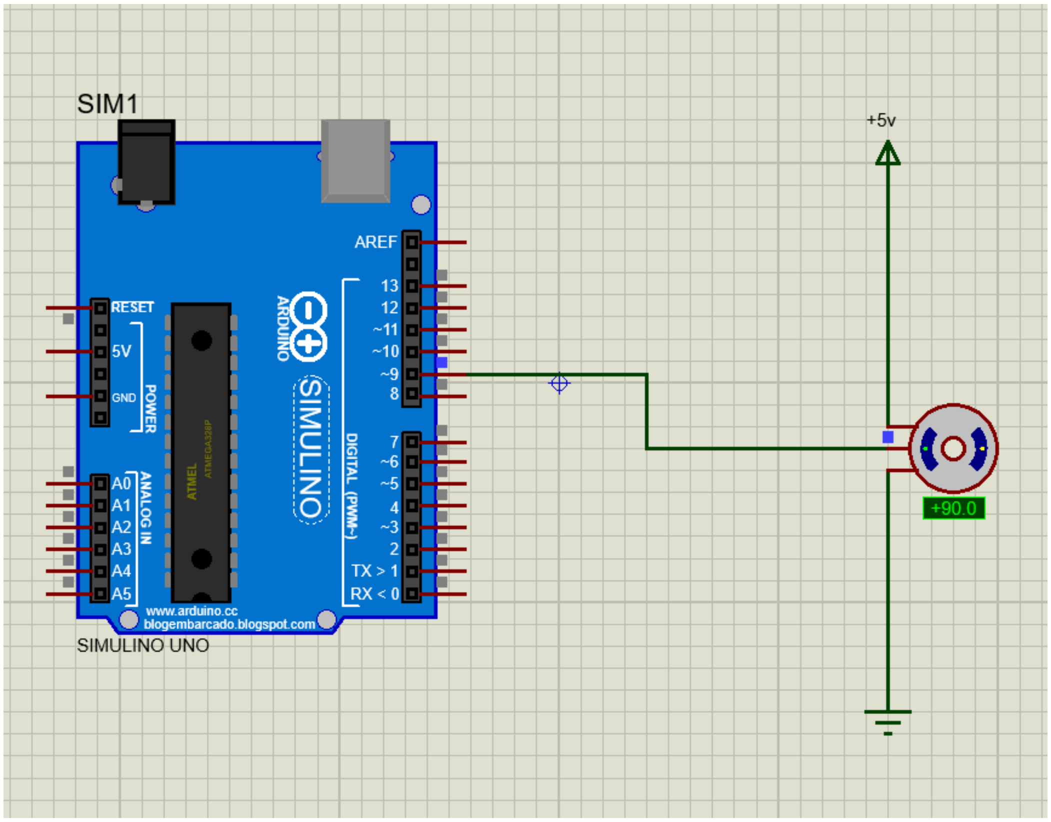
```
void loop() {
  // put your main code here, to run repeatedly:
  for (pos = 0; pos <= 180; pos += 1) {
    myservo.write(pos);
    delay(150);
  }
  for (pos = 180; pos >= 0; pos -= 1) {
    myservo.write(pos);
    delay(150);
  }
}
```

OUTPUT

Wait for some amount of time, Automatically the motor will rotate




SIM1
+5v
AREF
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX > 1
RX < 0
DIGITAL (PWM~)
RESET
5V
GND
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
ATMEL
ATMEGA328P
ARDUINO
SIMULINO
www.arduino.cc
blogembarcado.blogspot.com
SIMULINO UNO
+78.6


SIM1
+5v
AREF
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX > 1
RX < 0
DIGITAL (PWM~)
RESET
5V
GND
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
ATMEL
ATMEGA328P
ARDUINO
SIMULINO
www.arduino.cc
blogembarcado.blogspot.com
SIMULINO UNO
+90.0

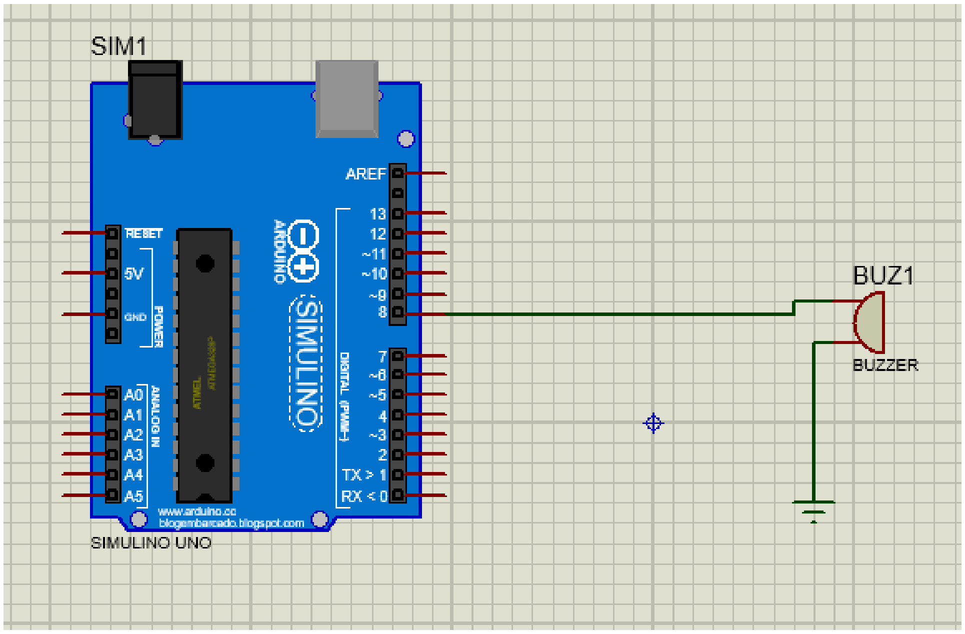
5 i. Interfacing of the Active Buzzer with Arduino
ii. Interfacing of the Relay with Arduino.

a. Interfacing of the Active Buzzer with Arduino

Requirements

1. Arduino Uno
2. Buzzer – Active one
3. Connecting wires

Circuit Connection



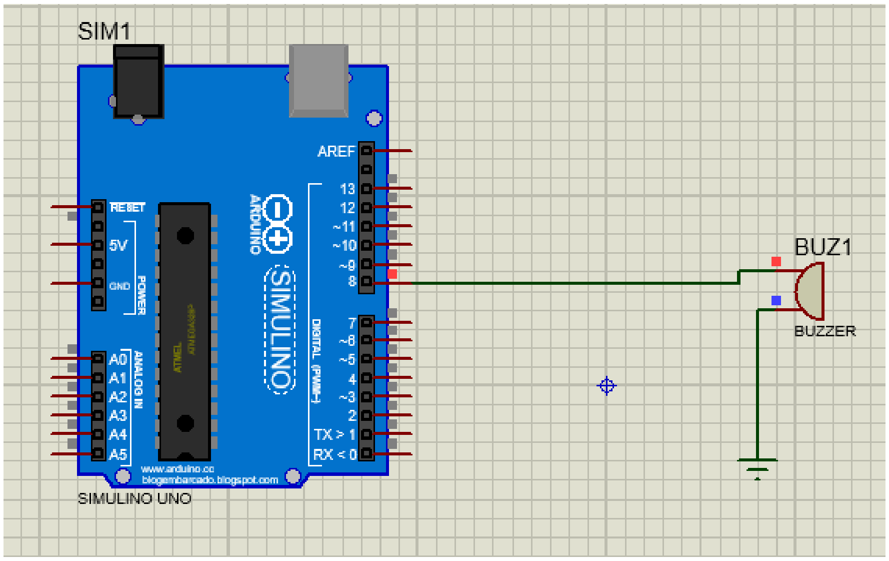
Note:

Double Click on the buzzer set the operating voltage to +5V and load Resistance to 100

Program

```
#define NOTE_A3  220

#define NOTE_B3  247

#define NOTE_C4  262

#define NOTE_G3  196


int melody[] = {

  NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4

};

int noteDurations[] = {
  4, 8, 8, 4, 4, 4, 4, 4
};
void setup() {
  // put your setup code here, to run once:
  for (int thisNote = 0; thisNote < 8; thisNote++) {
  int noteDuration = 1000 / noteDurations[thisNote];
    tone(8, melody[thisNote], noteDuration);
  int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
   noTone(8);
  }
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

OUTPUT

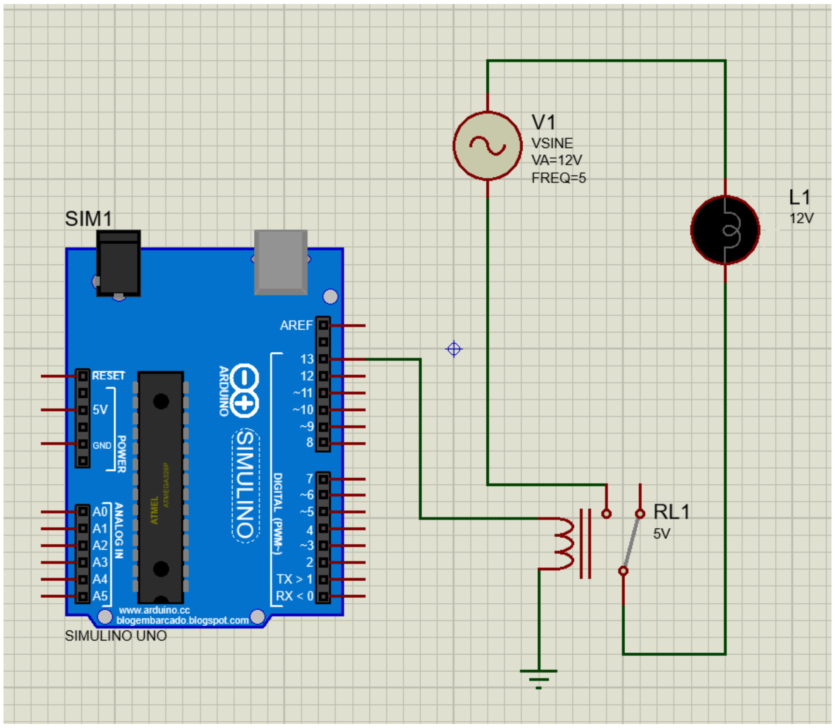


b. Interfacing of the Relay with Arduino.

Requirements

1. Arduino Uno
2. LAMP
3. Relay- Relay [Active- Animated Relay Model]
4. VSINE- sine wave AC voltage Source
5. Connecting wires

Circuit Connection:



Instruction: 1. Double click on Relay – Change component value to 5V

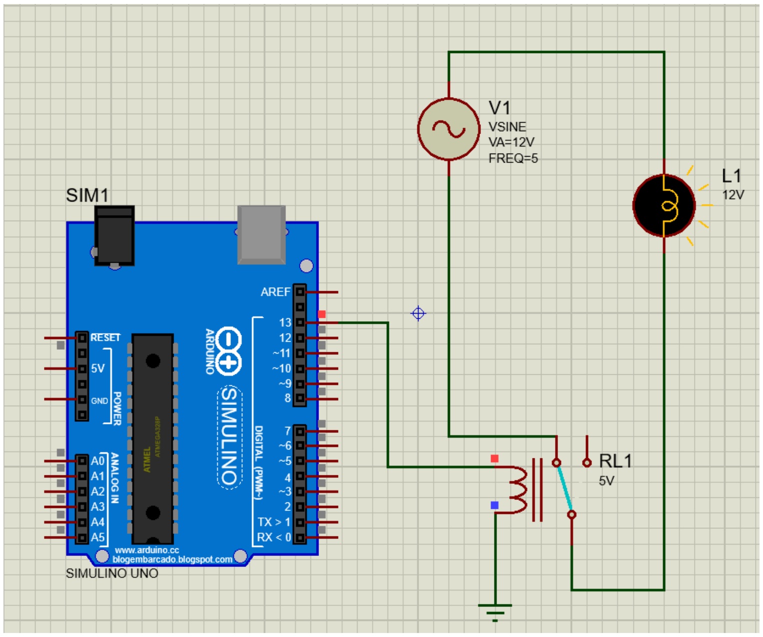
2. Double click on VSINE – set the amplitude to 12V and Frequency to 5

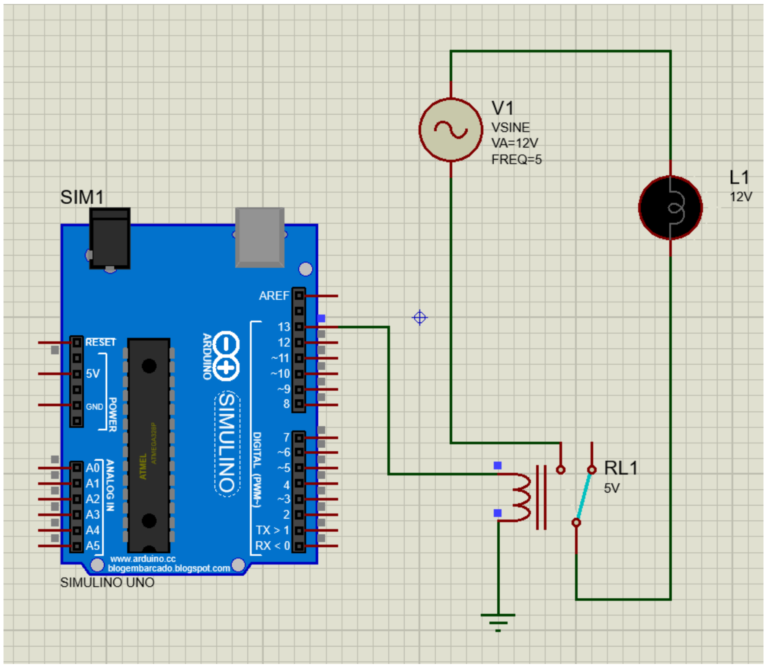
Program:

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


// the loop function runs over and over again forever
```

```
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

OUTPUT



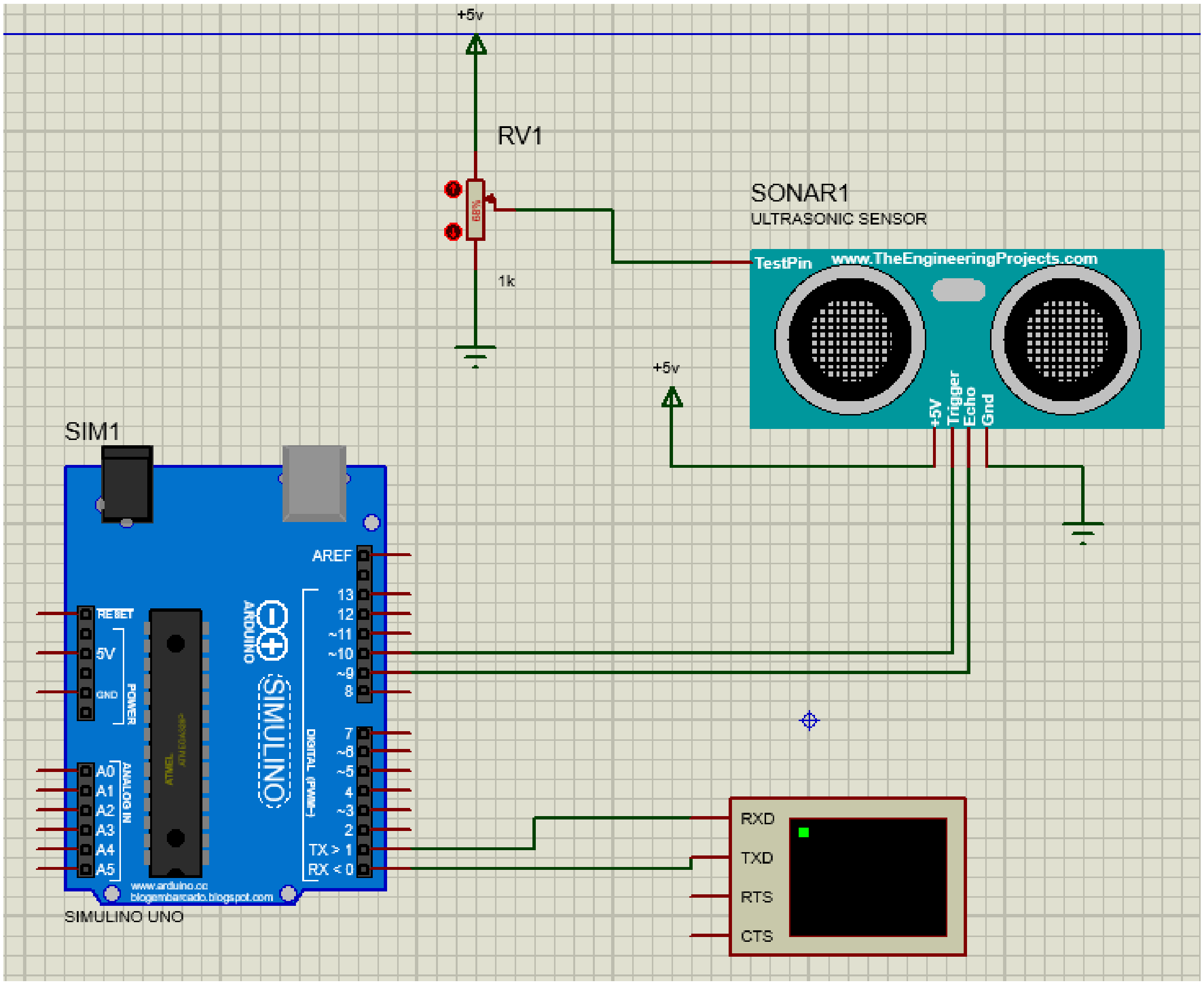


6. Calculate the distance to an object with the help of an ultrasonic sensor and display it.

Requirements

1. Arduino Uno
2. Ultrasonic Sensor
3. Virtual Display
4. Potentiometer - POT-HG
5. Connecting wires

Circuit Connection



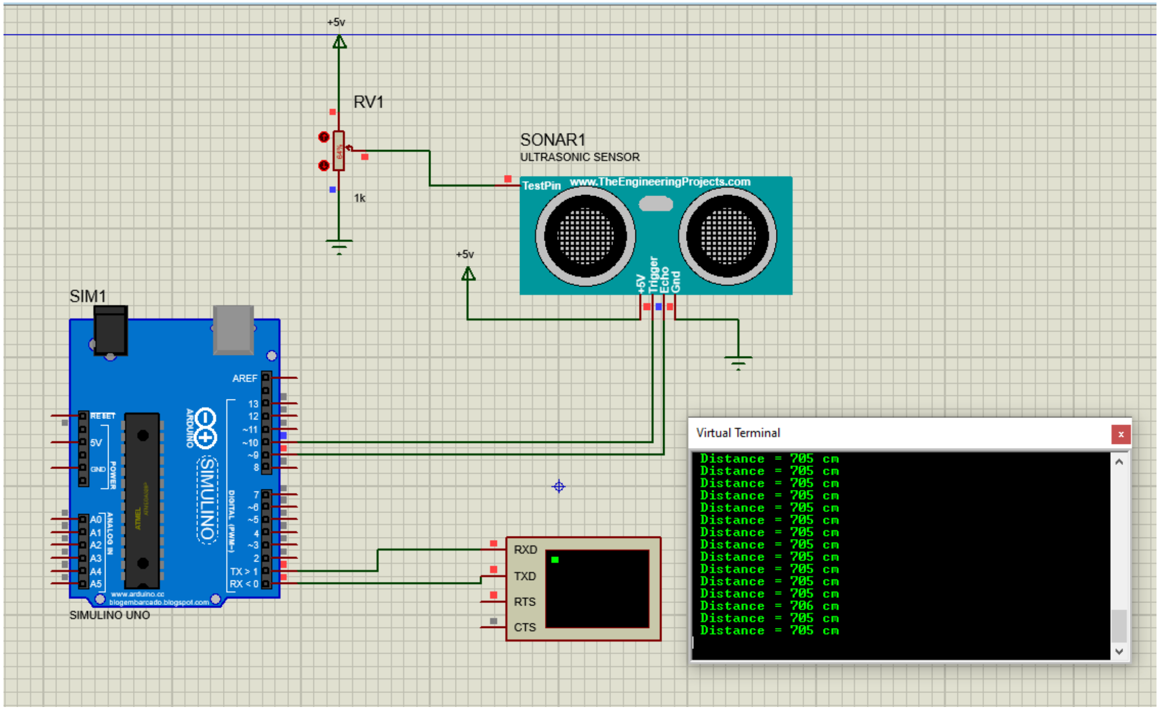
Note: Update the library file location of Ultrasonic sensor by double click select the file location were file name UltraSonicTEP.HEX is present

Program:

```
int trig = 10;
   int echo = 9;
   long duration;
   int cm;
void setup() {
  pinMode(trig, OUTPUT);
  pinMode(echo,INPUT);
  Serial.begin(9600);
}
void loop() {
   // Sketch to calculate ultrasonic distance
        digitalWrite(trig, LOW);
       delayMicroseconds(10);
       digitalWrite(trig, HIGH);
       delayMicroseconds(10);
       digitalWrite(trig, LOW);
       delayMicroseconds(10);
       duration = pulseIn(echo, HIGH);
       cm = (duration/2) * 0.034;
   // Print Value on Serial Monitor
```

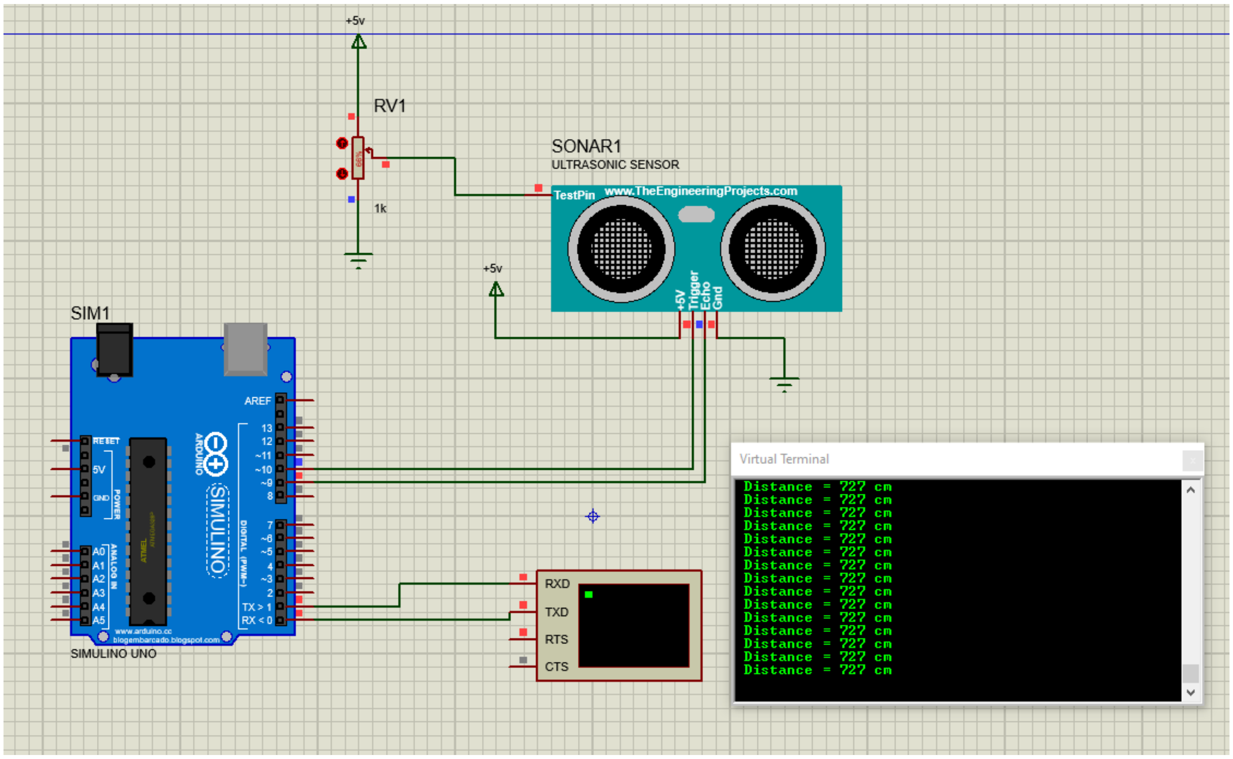
```
        Serial.print(" Distance = ");

        Serial.print(cm);

        Serial.println(" cm");

}
```

OUTPUT:

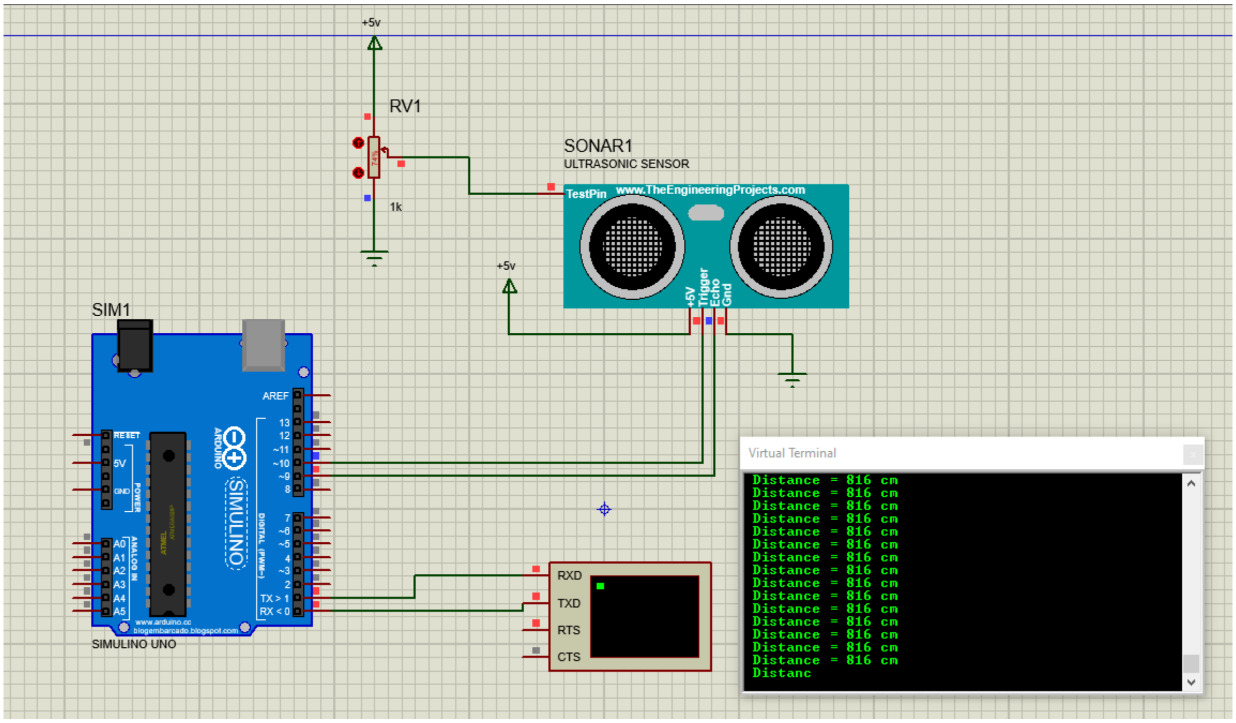


```
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 705 cm
Distance = 706 cm
Distance = 705 cm
Distance = 705 cm
```

Increase the distance by controlling the Potentiometer.



```
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
Distance = 727 cm
```



```
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distance = 816 cm
Distanc
```

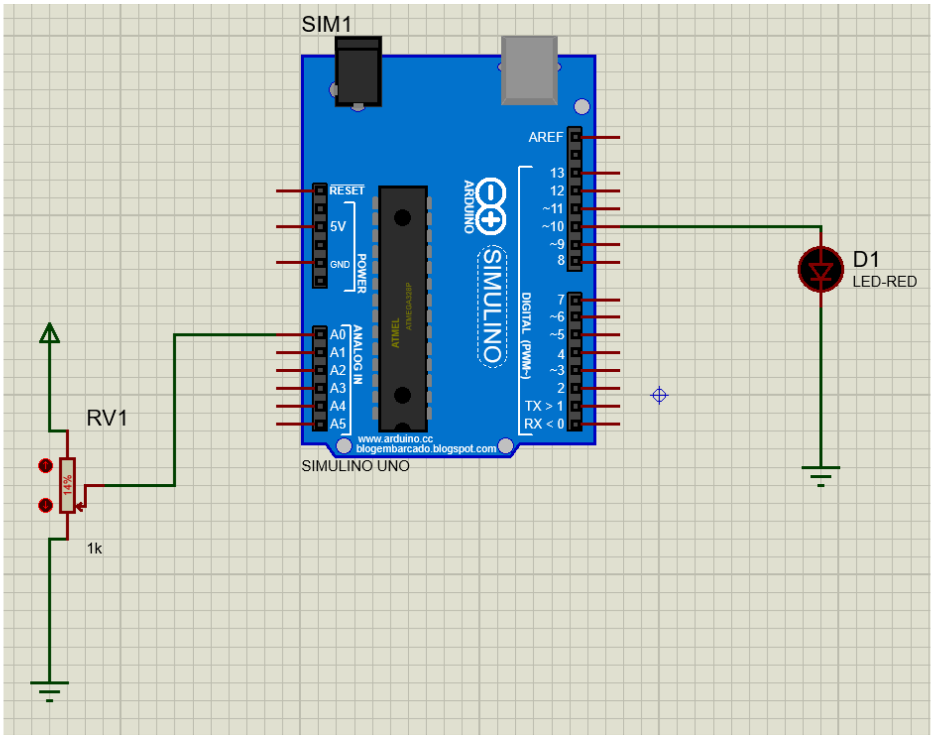
7 i. Controlling the LED blink rate with the potentiometer interfacing with Arduino
ii. ON/OFF control based on light Intensity –LDR sensor.

i. Controlling the LED blink rate with the potentiometer interfacing with Arduino

Requirements

1. Arduino Uno
2. LED-Red/ Bulb
3. Potentiometer - POT-HG
4. Connecting wires

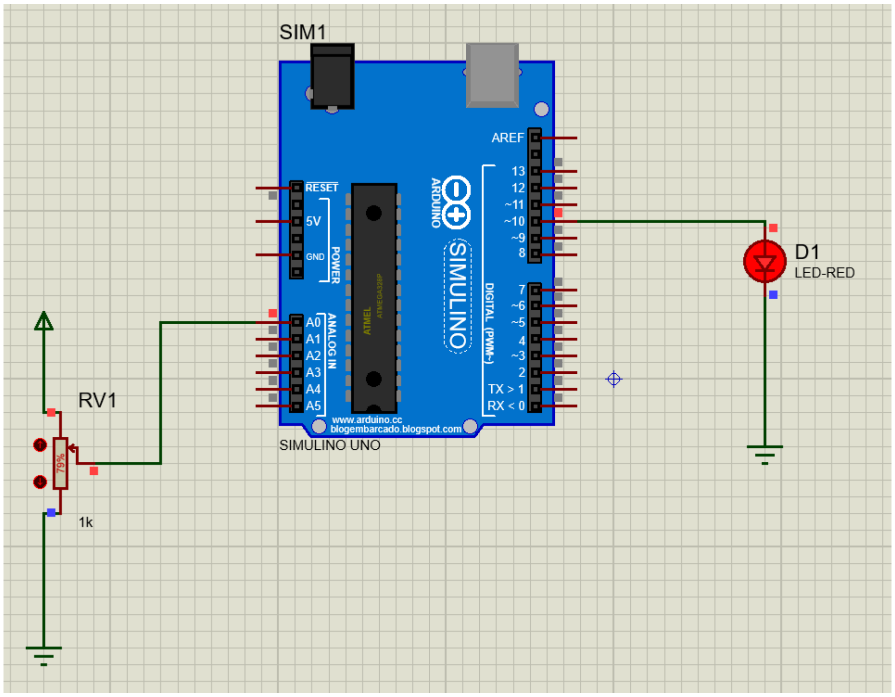
Circuit Connection

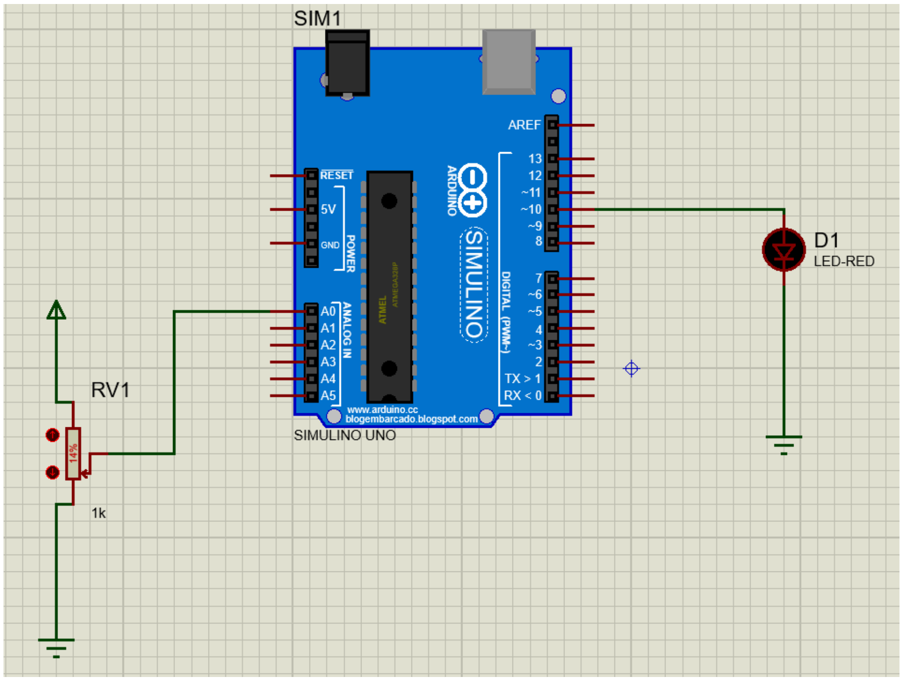


Program:

```
int potpin=A0;
int ledpin=10;
int potv=0;
void setup() {
  // put your setup code here, to run once:
  pinMode(ledpin, OUTPUT);
  pinMode(potpin, INPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  potv=analogRead(potpin);
  digitalWrite(ledpin,(potv/4));
  delay(10);
}
```

OUTPUT:



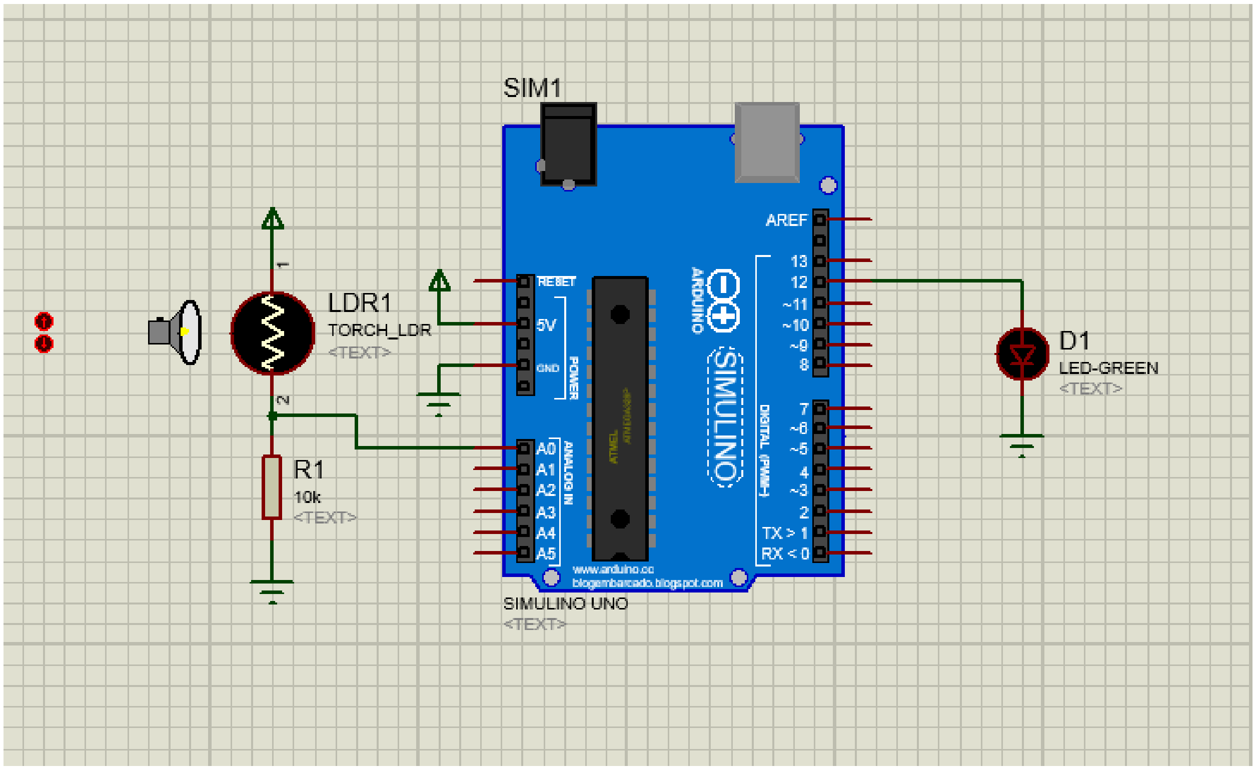


ii. **ON/OFF control based on light Intensity –LDR sensor**

Requirements

1. Arduino Uno
2. LED-Red/ Bulb
3. LDR-Torch_LDR
4. Resistor
5. Connecting wires

Circuit Connection



Program:

```
void setup()
{
 //Serial.begin(9600);  // initialize serial communication at 9600 bits per second:
 pinMode(12,OUTPUT);
}
void loop()
{
 // read the input on analog pin 0:
 int LDR_Value = analogRead(A0);
```

```
  if(LDR_Value < 500)
  {
   digitalWrite(12,HIGH);  // Turn ON LED
  }
  else
  digitalWrite(12,LOW);  // Turn OFF LED
}
```

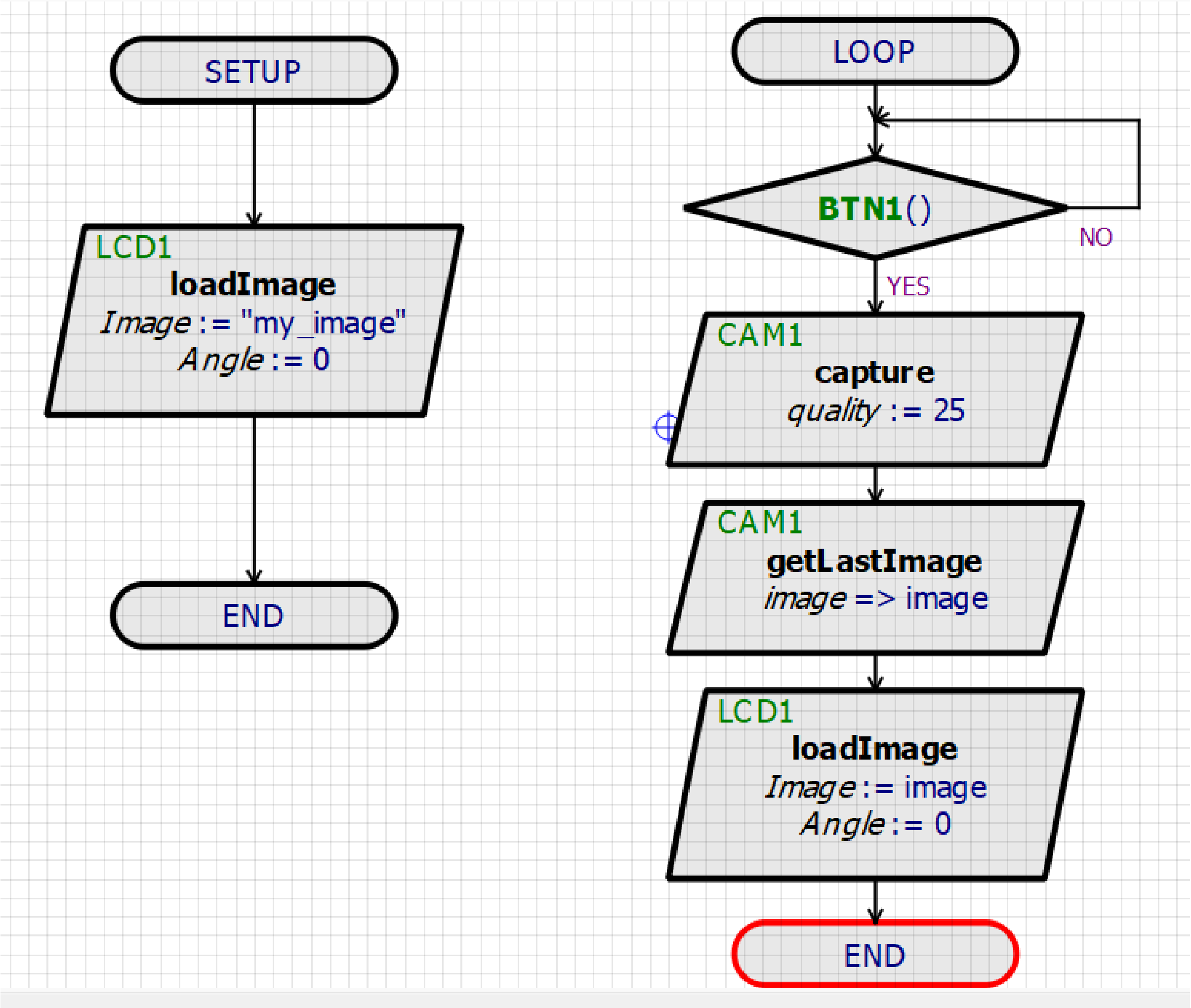
OUTPUT:

8. Interfacing the regular USB webcam with the device and capture the image.

Requirements

1. Raspberry pi
2. Pi camera- picamera
3. Button- break through
4. LCD- TFT LCD
5. Image with extension of .png

Visual Designers



Note:

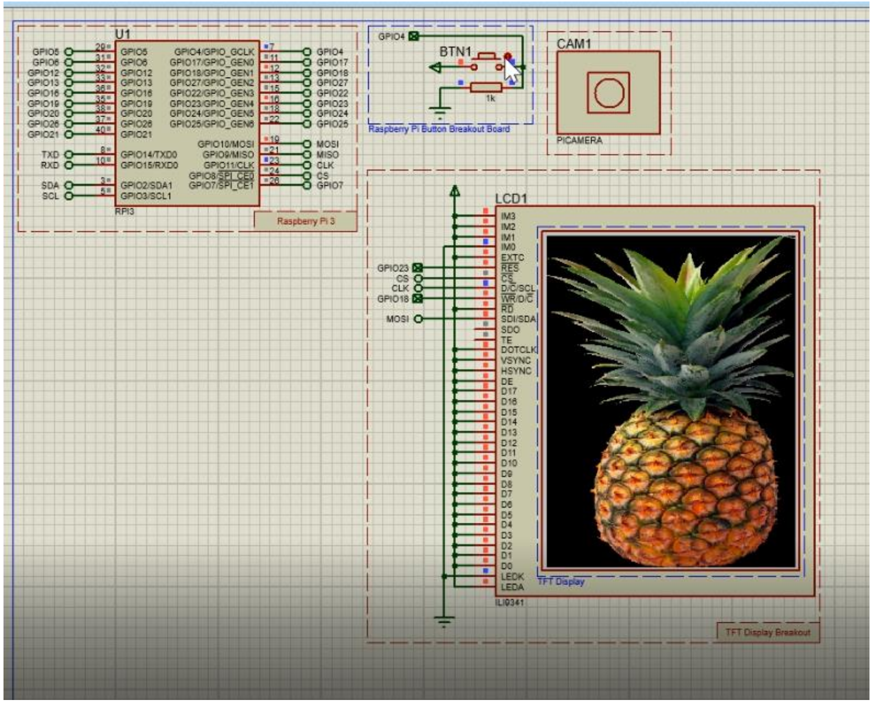
1. 1. Take any image with extension of *.png keep it in the desktop
2. Add the components from left hand side peripherals – right click-> add peripherals

In category

a. select →Breakout Peripherals -----→ in that choose Momentary Action push Button click to add button
b. select →Camera---→ Raspberry Pi camera Module click to add button
c. Display→Displays ---→ select the TFT display click to add button

Drag and drop the Function of the above module to the setup and loop

OUTPUT



9. **Build a circuit using Arduino based weather reporting over IOT**

Requirements

1. Arduino Uno
2. Soil Moisture sensor
3. Rain sensor
4. DHT11
5. LDR-LDR – Generic model
6. LCD- LM044L (20x4 Alphanumeric lcd)
7. Capacitor -CAP
8. Potentiometer - POT-HG
9. Inductor – Inductor Primitive
10. Logicstate – Active
11. Connecting wires

Circuit Connection

Note:

1. Add library files for the soil and rain sensors – by double click on the sensor upload the *.hex library of respective sensors
2. ***In Arduino IDE Upload DHT Library***

Program:

```
// include the library code:
#include <LiquidCrystal.h> //library for LCD

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(13, 12, 11, 10, 9, 8);
#include "dht.h"
#define dht_apin A1
dht DHT;

// defines pins numbers
const int SensorPin = A0;
const int SENSOR_PIN = 2;
void setup()
{
 pinMode (SENSOR_PIN, INPUT);
 lcd.begin(20, 4); // set up the LCD's number of columns and rows:
 lcd.setCursor(0,0);
 lcd.print("Weather Report");
}

void loop()
{
 int Val = analogRead(SensorPin);
 int Sensor_Val = digitalRead(SENSOR_PIN);
   // Prints Message on the LCD
 int Moisture = map(Val,0,1023,0,100);
 DHT.read11(dht_apin);
 lcd.setCursor(0,1);
 lcd.print("Soil Moisture: ");
 lcd.print(Moisture);
 lcd.print("%   ");
 if (Sensor_Val == HIGH) //If Sensor Detected the Rain
 {
   //digitalWrite(RLED_PIN, HIGH);
```

```
    //digitalWrite(GLED_PIN, LOW);

    lcd.setCursor(0, 3);
    lcd.print("   Rain Detected           ");
    delay(100);
    lcd.setCursor(0, 3);
    lcd.print("   Rain Detected.          ");
    delay(100);
    lcd.setCursor(0, 3);
    lcd.print("   Rain Detected..         ");
    delay(100);
    lcd.setCursor(0, 3);
    lcd.print("   Rain Detected...        ");
    delay(100);
  }
  else
  {
    lcd.setCursor(0, 3);
    lcd.print("      NO RAIN           ");
    //digitalWrite(RLED_PIN, LOW);
    //digitalWrite(GLED_PIN, HIGH);
  }


int Temp = DHT.temperature;
  lcd.setCursor(0,1);
  lcd.print("  TM:");
  lcd.print(Temp);
  lcd.print("C  ");

  //Humidity Sennsing
  int Humidity = DHT.humidity;
  lcd.setCursor(11,1);
  lcd.print("HM:");
  lcd.print(Humidity);
  lcd.print("%  ");
}
```

OUTPUT: