INDEX

Sl. No	Name of Experiment	Date of Experiment	Marks	Signature of Faculty

Experiment No. : 01 Date : 24/04/24

Blink

Aim : Turn a LED on & off every second

Hardware required : Arduino UNO Board, LED, resistor, jump wires, Bread board

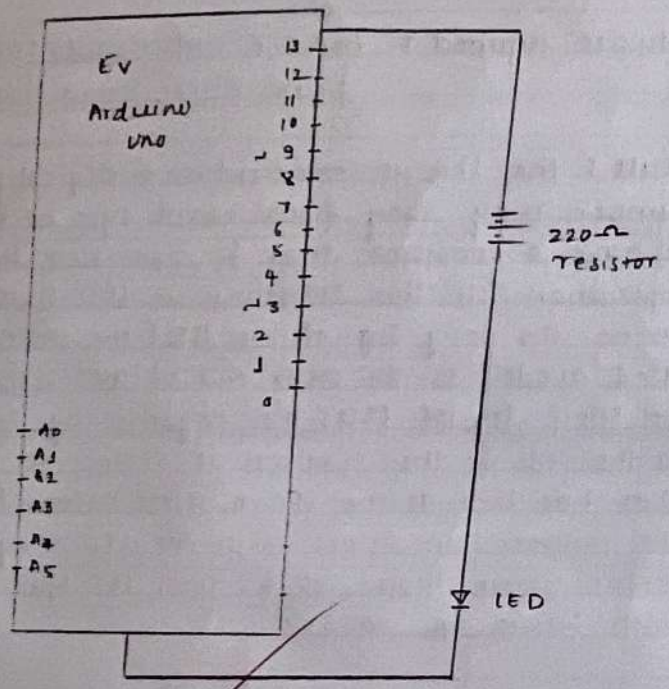
Circuit : The LED is connected to a digital pin & its number may vary from board type to board type We have a constant that is specified in every board description file This constant is LED-BUICTIN Constant Connect the long leg of the LED [the positive leg is called anode] to the other end of the resistor Connect the short leg of LED (The negative leg, called Cathode] to the GND in the diagram as shown on UNO Board that has D-B as the LED-BUICTIN value; The value of the resistor in series with the LED may be of a diff-erent value than 220Ω; then LED will light up also with value up to 1KΩ

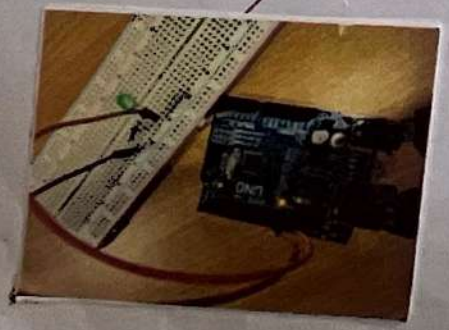
Code :

```
/* Blink
Modified 24th April 2024 Gunavarshini
*/
#define led pin 13
Void setup ()
{
pinmode (led-pin, OUTPUT);
}
Void loop ()
{
```

Schematics

OUTPUT 01





Experiment No. :..... Date :.....

```
digitalWrite(led_pin, HIGH);
delay(2000);
digitalWrite(led_pin; LOW);
delay(2000);
}
```

Result: We could see the LED blinking with a delay.

O/P:

Experiment No. : 02 Date : 8/5/24

3 LED BIINK

Aim : Alternate blinking of LED's with arduino board

Hardware Required : Arduino UNO Board, Bread board, 3 LED's, Jump Wires, Ohm resistor x 2

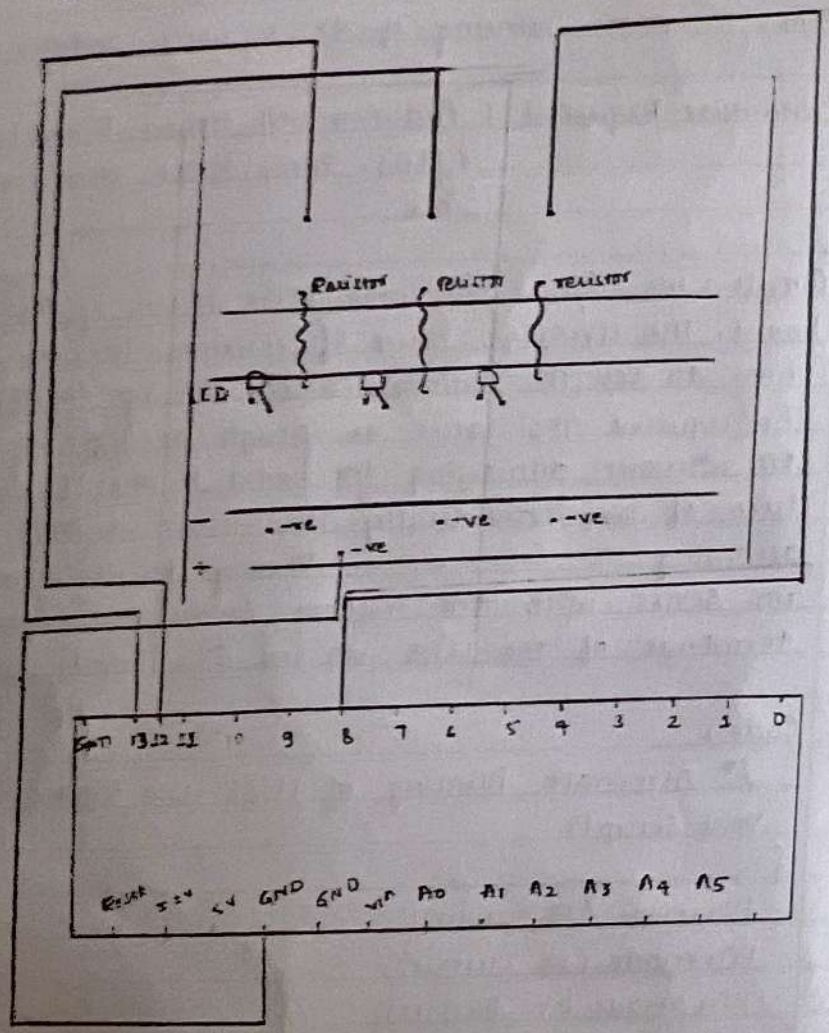
Circuit : We connect the three LED's to pins of the arduino board. The limiting value of resistance between 220-330 Ohms to set the optimal current through the LED's. The required resistance is enough to light up an LED without damaging the board & the LED, will turn of lon individually. The circuit is connected according to Schematics. Simply LED is connected in series with the resistor. Connect the negative terminal of the LEDs to the GND (Ground)

Code :

```
/* Alternate blinking of LED's with Arduino Board */
Void Setup()
{
Pinmode (13, output);
Pin mode (12, output);
pin mode (8, output);
}
Void loop ()
{
digital Write (13, HIGH);
delay (1000);
digital Write (13, low);
```

Schematic

OUTPUT 02

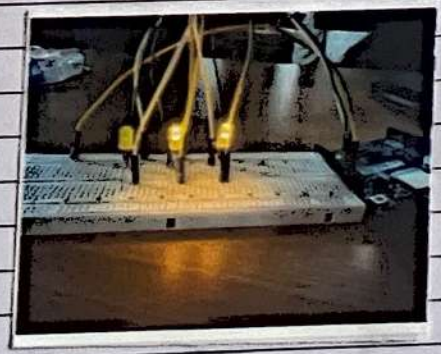


Blinding of 3 leds with Arduino UNO

Experiment No. : _____ Date : _____

```
delay (1000);
digital Write (12, HIGH);
delay (1000);
digital Write (12, LOW);
delay (1000);
digital Write (8, HIGH);
delay (1000);
digital Write (8, LOW);
delay (1000);
}
```

Result : The 3 LEDs blink one after another in Series after a delay



Experiment No.: 03 Date: 15/5/24

LED with Push button

Aim: LED Blinking with a push button using Arduino

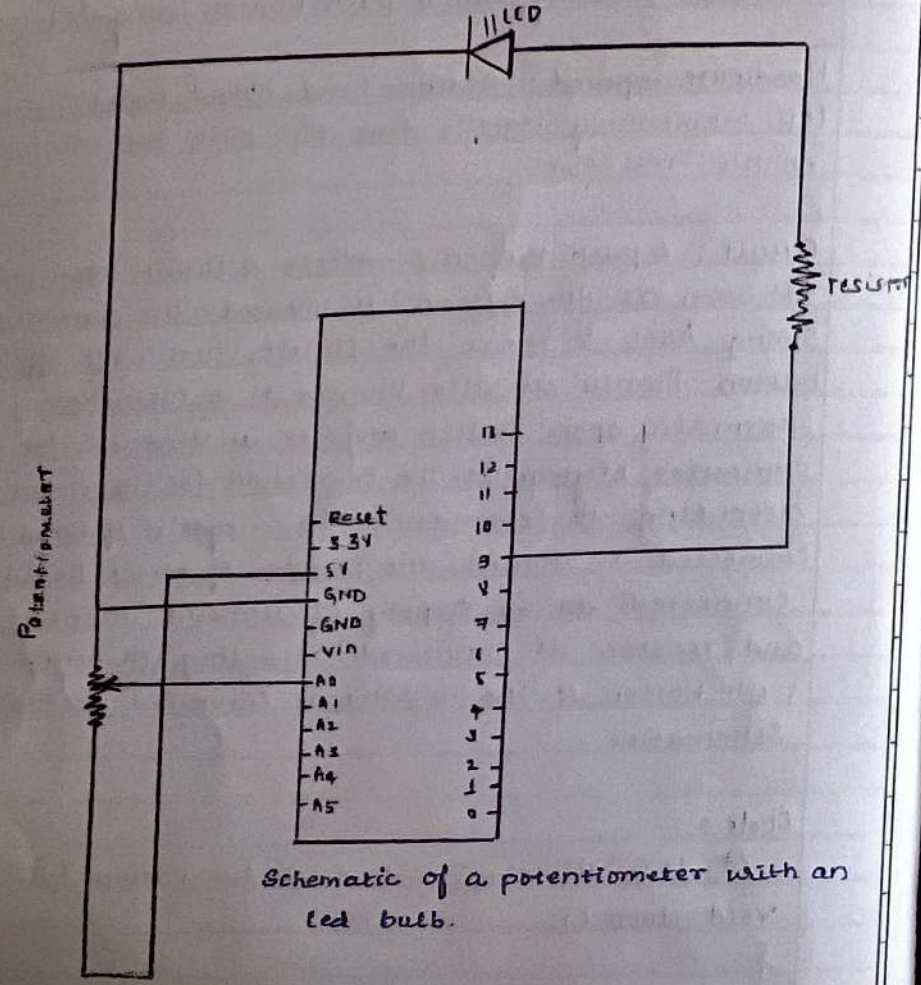
Hardware required: Arduino hard, bread board, jumper wires LED, Momentary tactile four-pin push button, two ohmic resistors.

Circuit: A push button completes a circuit, when pressed. As Soon as the button is released, the connection will Spring back & break the circuit, turning it off. The push button switch is also known as a mementary or normally open switch and it is used in, for examples, Computer keyboards. The Connection for the circuit is done according to Schematics. The negative of bread board is connected to ground, the positive of bread board is Connected to 5V supply of arduino uno board. The led and resistor is Connected according to Schematics and push botton & the resistor is Connected according to Schematics.

Code:

```
/* LED Blinking with push bo button */
Void loop ()
{
int potentiometer value = analogRead (POTENTIOMETER_PIN);
int brightness = potentiometer value ;
Analog write (LED_PIN, brightness);
}
```

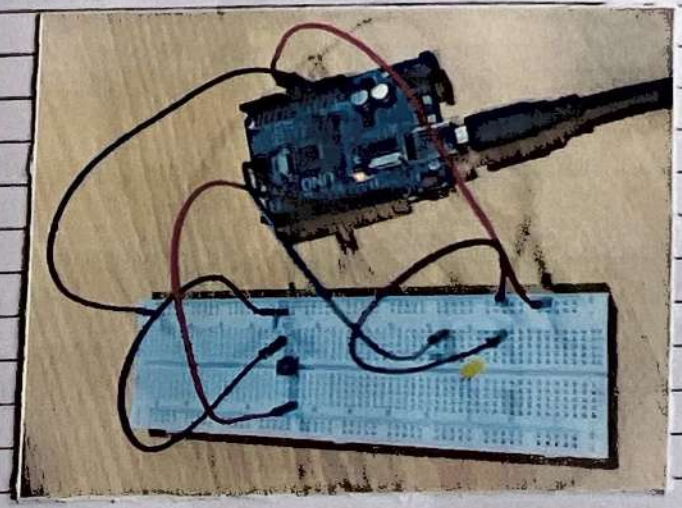
OUTPUT : 04



Schematic of a potentiometer with an led bulb.

Experiment No. :..... Date :.....

Result : As the potentiometer's regulator was turned clockwise & anticlockwise the brightness of LED increased and decreased



Experiment No. : 04 Date : 22/5/24

POTENTIOMETER

Aim: Control LED brightness with a potentiometer

Hardware required: Arduino, Uno board, bread board, LED, Ohm resistor, potentiometer, jumper wires

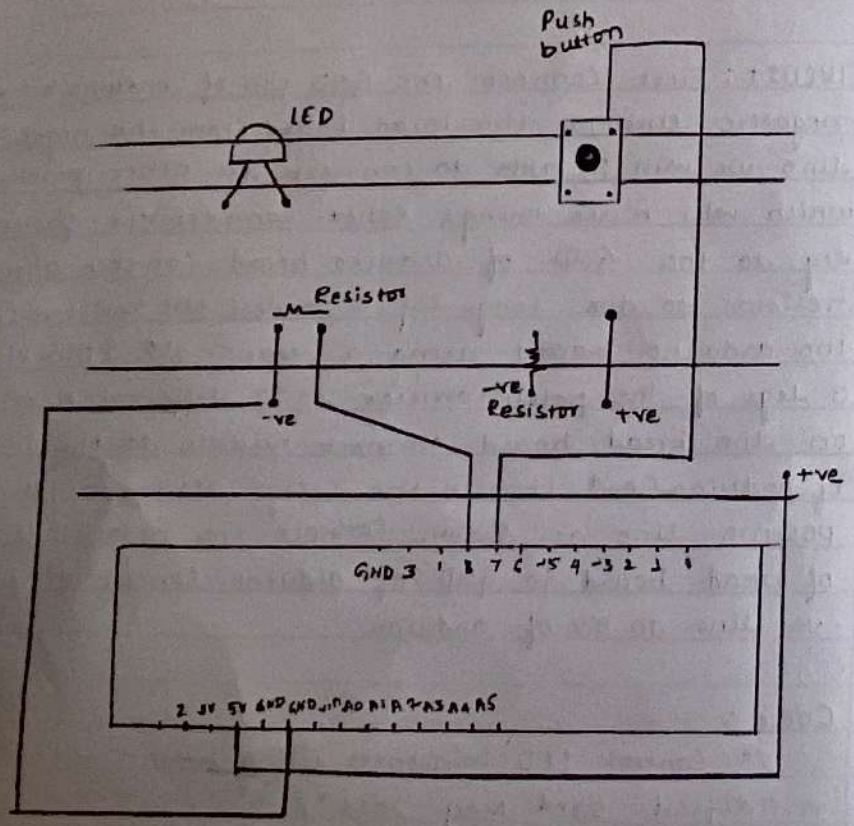
CIRCUIT: First connect the GND pin of arduino to the negative line on the bread board from this negative line we will be able to connect all other grounds, which will make things easier. Connect the shorter leg to the GND of arduino board. Connect ohmic resistor to the long leg & connect the resistor to the arduino board using a jumper to 9 pin. Plug 3 legs of the potentiometer to I different lines on the bread board. Connect middle to the pin of arduino [A0]. Connect the 2 legs to the negative & positive line as shown. Connect the negative line of bread board to GND of arduino. Connect the positi -ve line to 5V of arduino.

Code:

```
/* Control LED brightness using potentiometer
modified 22nd May 2024 */

# define LED. PIN 9
# define POTENTIOMETER PIN A0
void setup()
{
pinMode (LED PIN - DUTPUT);
}
```

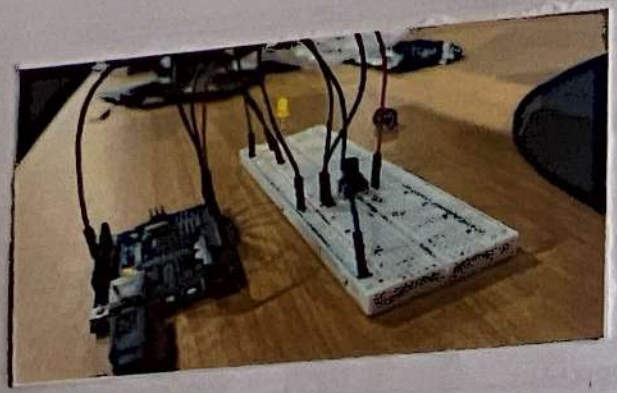
OUTPUT : 03



Experiment No. :..... Date :.....

```
#define LED-PIN 8
# define BUTTON-PIN 7
 void setup()
 {
  pinMode [LED-PIN, OUTPUT);
  pin Mode (BUTTON PIN, INPUT);
  }

void loop()
{
 if (digital Read (Button-PIN == HIGH))
 {
digital write (LED-PIN, HIGH);
 }
 else
 {
  digital write (LED-PIN, LOW);
 }
}
```



Experiment No. : 04 Date : 22/5/24

Buzzer

Aim: To Create a circuit using an Arduino board that produces different sounds using buzzer.

Hardware required: Arduino UNO Board, Buzzer, Jumper wires, USB cables, Bread board

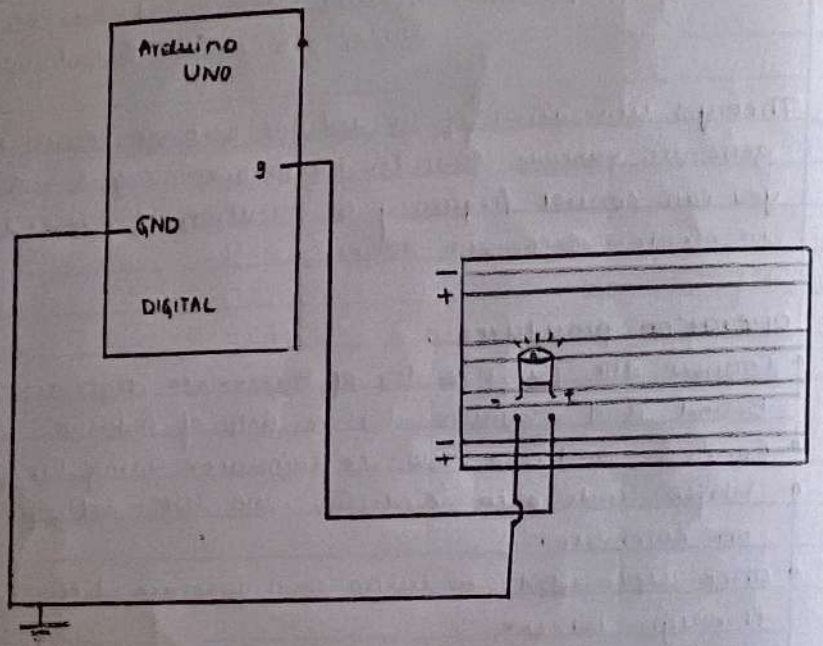
Theory: Utilization of an arduino UNO to control Buzzer generate various sounds, by programming the arduino. you can control frequency & duration of buzzer's output to create different tones.

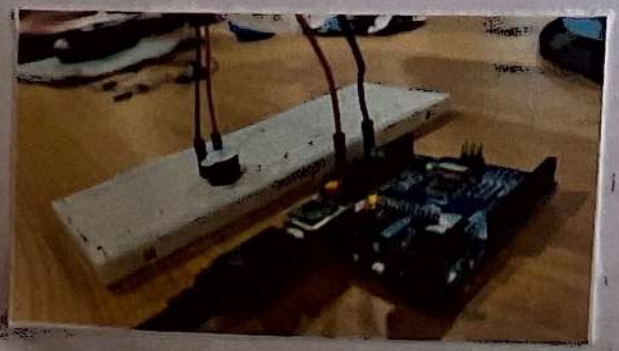
Operation procudures:

- Connect the pasitive leg of Buzzer to digital pin 9 on Board and negative leg to GND of Arduino board.
- Connect arduino UNO to computer using USB
- Write code in arduino UNO IDE - Software opened on computer.
- once uploaded, arduino will generate different sounds through buzzer.

Code:

```
void setup () // Set digital pin 9 as an output
 {
  pinmode (9, output);
 }
void loop()
 {
 tone (9, 2500, 200);
 delay (1000);
 }
```





Experiment No. :..... Date :.....

Conclusion :

By following the steps allow and uploading the provided Code to arduino UNO Board, if we can succesfully creats project that produces different Buzzer Sounds The Frequency of duration of sounds can be adjusting by modifying Code

Experiment No. : 05 Date : 29/5/24

05. Displaying the temperature and Humidity Readings using Arduino with DHT11 Sensor.

Aim : To design and develop a program using Arduino to interface with a DHT11 Sensor and display the temperature and humidity readings

i) Arduino Board
ii) DHT11 Sensor
iii) Jumper Wires
iv) Bread board

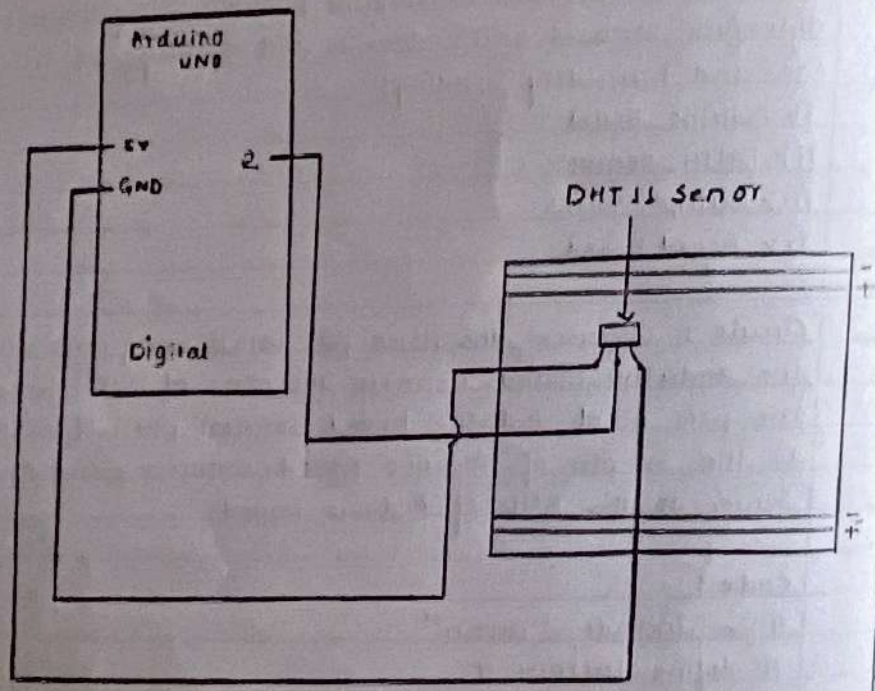
Circuit : Connect the three pin DHT11, using three wires, with the arduino board. Connect the pin2 of DHT11 Sensor of the pin 7 of Arduino board, Connect pin 1 of DHT11 Sensor to the 5V pin of Arduino board. Connect pin 3 of DHT11 Sensor to the GND of Arduino board.

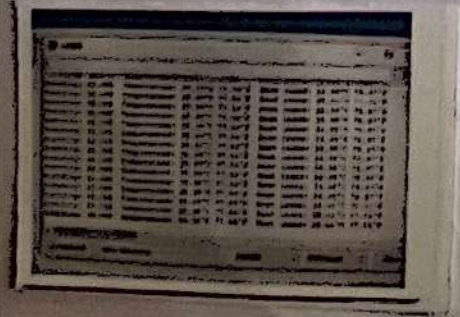
Code :

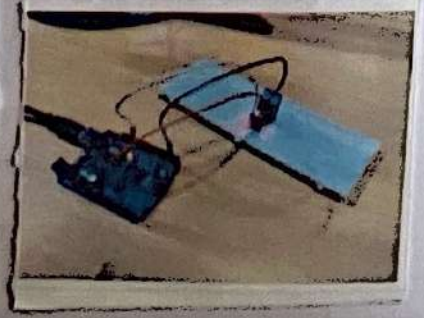
```
#De. include "DHT.h"
# define DHTPIN 7
# define DHT TYPE DHT11

DHT dht (DHTPIN, DHTTYPE);

Void Setup () {
 Serial. begin (9600);
 Serial. println (F("DHT * * Test ("));
 dht. begin ();
 }
```







Experiment No. :..... Date :.....

```
void loop() {
  delay(2000);
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  float f = dht.readTemperature(true);

  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println(F("Failed to read from DHT Sensor!"));
    return;
  }
  float hif = dht.computeHeatIndex(f,h);
  float hic = dht.computeHeatIndex(t,h,false);

  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("% Temperature: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(hf);
  Serial.print(F("°F HeatIndex: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
}
```

Result :

The above experiement displays temperature and humidity on the screen with DHT11 Sensor.

Experiment No.: 06 Date 14/06/24

06. Display temperature Reading using Arduino with LM35 Sensor.

Aim: To design & develop a program using Arduino to interface with a LM35 sensor & display the temperature readings

Hardware required:

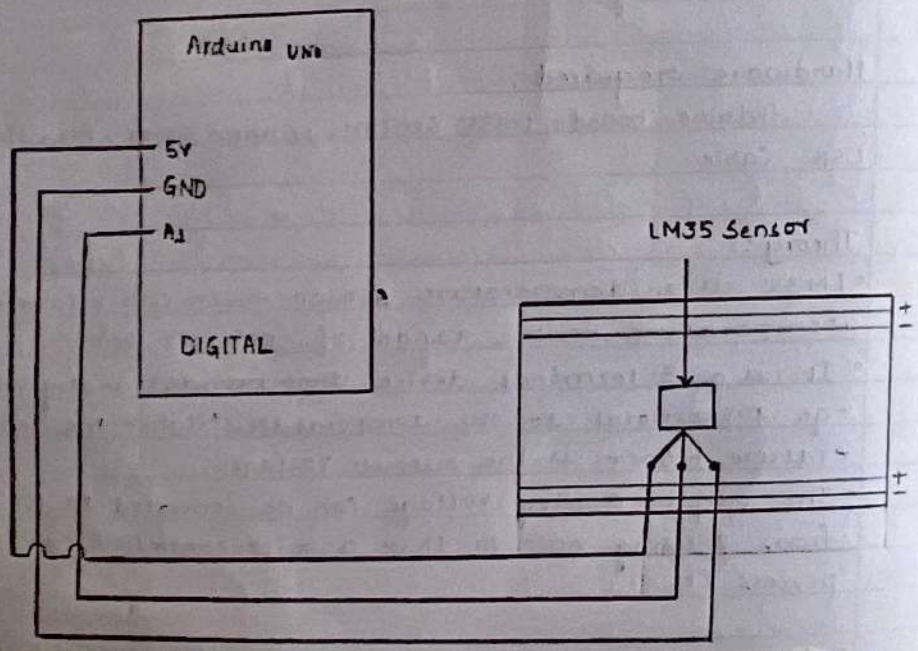
Arduino board, LM35 Sensor, jumper wires, breadboard USB Cable

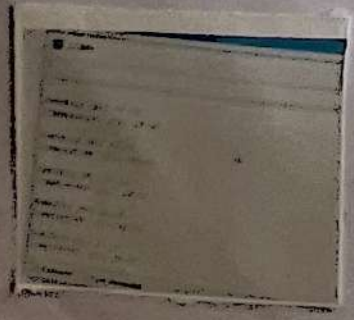
Theory:

- LM35 is a temperature sensor which can measure temperature in the range of -55°c to 300°c.
- It is a 3-terminal device that provides analog voltage proportional to the temperature. Higher the temperature, higher is the output voltage
- The output analog voltage can be converted to digital from 0 using ADC or that a Microcontroller can process it.

Procedure:

1. Connect the three pin LM35 Sensor using three wires with arduino board
2. Connect the pin 2 of LM35 sensor to the analog pin A1 of the arduino board.
3. Connect the pin 1 of Sensor to 5V of the Arduino board
4. Connect to pin 3 of Sensor to the ground i.e. GND of the Arduino board.
5. Write and upload the code to arduino.





Experiment No. :..... Date :.....

Code :

```
#define SensorPin A0
    void setup() {
    Serial.begin (9600);
    }
    void loop (){
int reading = analogRead (SensorPin);
float voltage = reading (5.0/1024.0);
float temperatureC = voltage 100;
Serial.print ("Temperature : ");
Serial.print (temperature C);
Serial.print ("\x(2\xBO");
Serial.print ("C1");

float temperatureF = (temperature (x*9.0/5.0)+32.0);
Serial.print (temperature F);
Serial.print ("/x2/xBO");
Serial.print ("F");

delay (1000);
}
```

Result :

The above experiment displays the temperature in Celcius & Fahrenheit can the server with the LM35 Sensor using Arduino.

Experiment No. : 07 Date : 21/06/24

07 Controlling multiple LED's sequentially using an Arduino.

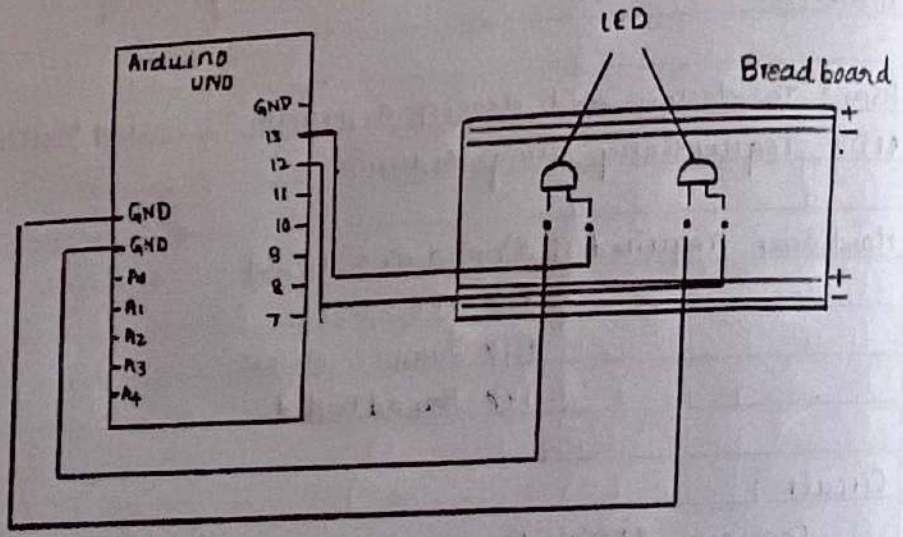
Aim : To design and develop a program to control multiple LED's sequentially using Arduino.

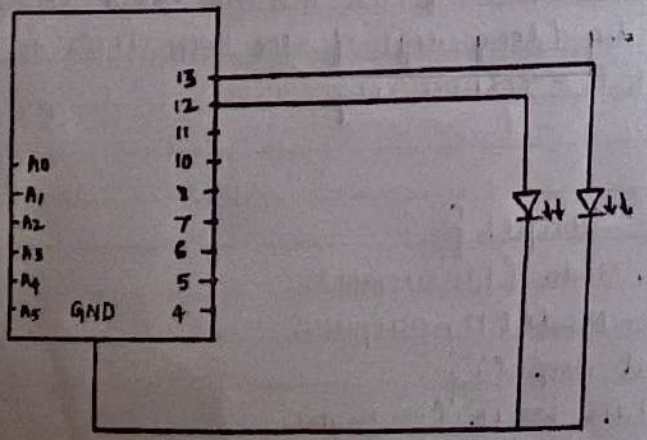
Hardware required : i) Arduino board
ii) LED's
iii) Jumper wires
iv) Bread board.

Circuit :
Connect LED's to the Arduino board using jumper wires Connect the negative leg (short leg) of both the LED's to the GND of the Arduino board. Connect the positive leg (long leg) of the both LED's to the digital pin 12 & 13 respectively

Code :

```
void setup() {
pin_Mode (13, output);
Pin - Mode (12, output);
void loop () {
digital - write (13, High);
 delay (1000);
digital write (13, low);
 delay (1000);
digital write (12, High);
 delay (1100);
digital write (12, low);
```





Experiment No. :..... Date :.....

```
    delay (1000);
}
```

Result :

This experiment controls the blinking of multiple LED's requently using an Arduino UNO.

Experiment No. : 08 Date :

~~Display temperature Reading using Arduino with LM35~~

08. Design & develop a program using ESP8266 Node MCU to interface with DHT11 Sensor & display the temperature & humidity reading.

Aim :

The aim of this experiment is to design & develop a program using an ESP8266 Node MCU board to interface with DHT11 Sensor & display the temperature & humidity readings.

Components used :

* ESP8266 Node MCU board
* DHT11 temperature & humidity Sensor
* Bread board
* Jumper wires
* USB cable

Theory :

The DHT11 Sensor is a digital temperature & humidity sensor that communicates with Node MCU using one-wire protocol The program will read the sensor data & display the temperature & humidity readings on the Serial Monitor

Procedure :

1) Connect the VCC pin of the DHT11 sensor to the 3.3V pin on Node MCU
2) Connect the GND pin of the DHT11 Sensor to the GND pin on Node MCU

Experiment No. :_____ Date :_____

3) Connect the data pin of the DHT11 sensor to a digital i/o pin (e.g D2) on node MCU

4) upload the Arduino sketch to the Node MCU using Arduino IDE

5) Open the Serial monitor in the Arduino IDE to view temperature & humidity readings

Code :

```
# include "DHT.h"
# define DHTPIN D2
# define DHTTYPE DHT11
 DHT dht (DHTPIN, DHTTYPE);

void setup ()
{
 Serial.begin (9600);
 Serial.println (F ("DHTxx test !"));
 dht.begin ();
}
void loop ()
{
delay (2000);
float h = dht.readHumidity ();
float t = dht.readTemperature ();
float f = dht.read Temperature (true);
if (isnanch) || isnan(t) || isnan (f)
{
Serial.println (F(" Failed to read from DHT Sensor
 !"));
return ;
```

Experiment No. :..... Date :.....

```
}
float hif = dht.computeHeatIndex(f,h);
float hic = dht.computeHeatIndex(t,h,false);

Serial.print(F("Humidity : "));
Serial.print(h);
Serial.print(F("%. Temperature : "));
Serial.print(t);
Serial.print(F(" C"));
Serial.print(f);
Serial.print(F("F Heat Index : "));
Serial print(hic);
Serial.print(F("C "));
Serial.print(hif);
Serial.println(F("F"));
}
```

Conclusion :

The DHT11 sensor provides accurate temperature & humidity readings

Experiment No. : 09 Date :

09. Design and develop a program to control an LED ON and OFF using an ESP8266 Node MCU

Aim :

The aim of this experiment is to design & develop a program to control an LED connected to an ESP8266 Node MCU board.

Components used :

* ESP8266 Node MCU board
* LED [Light - Emitting Diode]
* 220 - ohm resistor
* Breadboard
* Jumper wires

Theory :

* The ESP8266 Node MCU board has multiple GPIO [General purpose Input output] pins that can be used to control external devices like LED's.
* We will use one of these GPIO pins to connect the LED with a current limiting resistor

Procedure :

1) Connect the anode of LED to one end of the 220-Ω resis-tor
2) Connect the other ends of the resistor to GPIO pin [e.g. D0] on the ESP8266 Node MCU board
3) Connect the Cathode of the LED to the ground on the bread board
4) Upload the Arduino Sketch to the ESP8266 Node MCU

Experiment No. :_____ Date :_____

board using the Arduino IDE

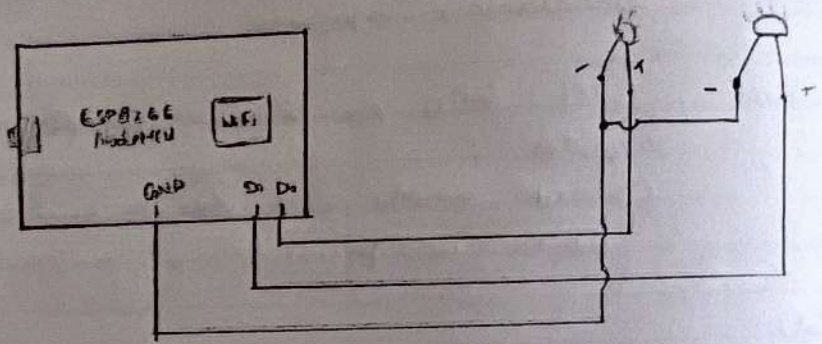
5) Observe the LED, which should be turn on & off automatically as per the program's logic

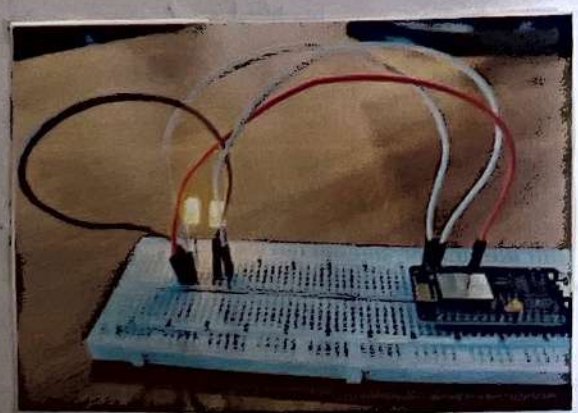
Code :

```
#define LED_PIN D0
 void setup ()
{
 Pin Mode (LED_PIN , Output );
}
void loop()
{
digital write (LED_PIN , HIGH);
delay (1000);
digital write (LED_PIN, LOW);
delay (1000);
}
```

Conclusion :

In this experiment, we successfully designed & developed a program to control an LED using ESP8266 Node MCU board. This simple experiment demonstrates how to use the GIPO pins of the ESP 8266 Node MCU to control external devices





Experiment No. : 09 Date :

Two LED NodeMCU

Aim: Control two LEDs by NodeMCU.

Hardware: ESP8266 NodeMCU, 2 LED, Jumper Wire, Bread Board.

Circuit: Both cathode to GND.
One anode to GPIO D0 and other to GPIO D1.
Connect the GND pin to ground rail.

Code:

```
#define LED D0
#define LED D1
void setup(){
    pinMode(LED, OUTPUT);
    pinMode(LED2, OUTPUT);
}
void loop(){
    digitalWrite(LED, HIGH);
    digitalWrite(LED1, HIGH);
    delay(1000);
    digitalWrite(LED, LOW);
    digitalWrite(LED1, LOW);
    delay(1000);
}
```

Conclusion: Both LEDs turn on and off automatically as per code.