Zed-King

GROUP OF INSTITUTIONS

ISO 9001 : 2015 Certified

M4-R5.1 Practical Marathon

Introduction to Internet of Things and It's Application

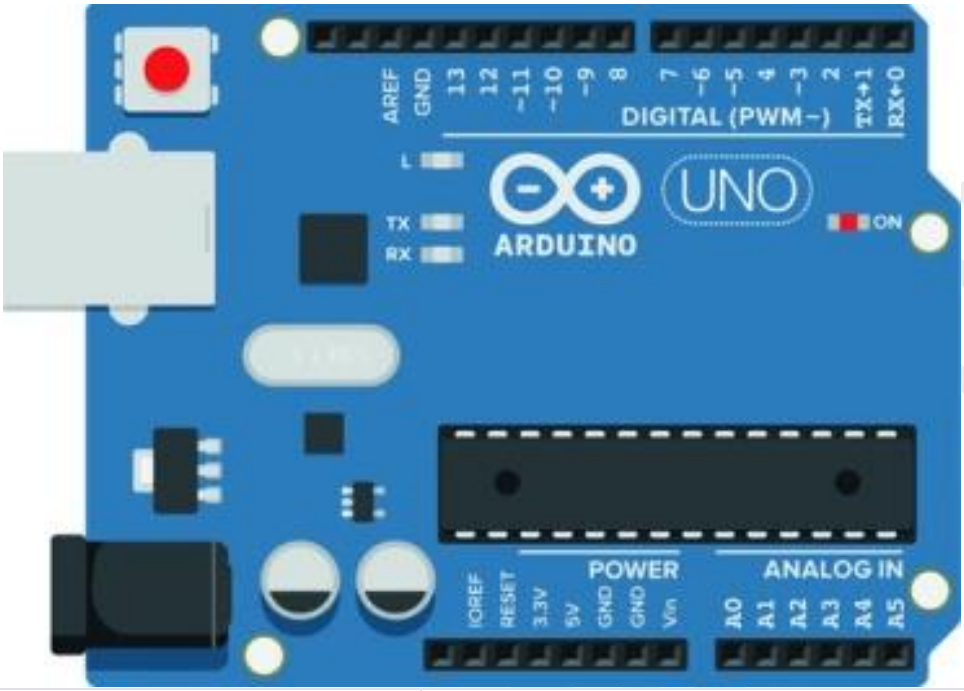




Zed-King Institute

Basic Components

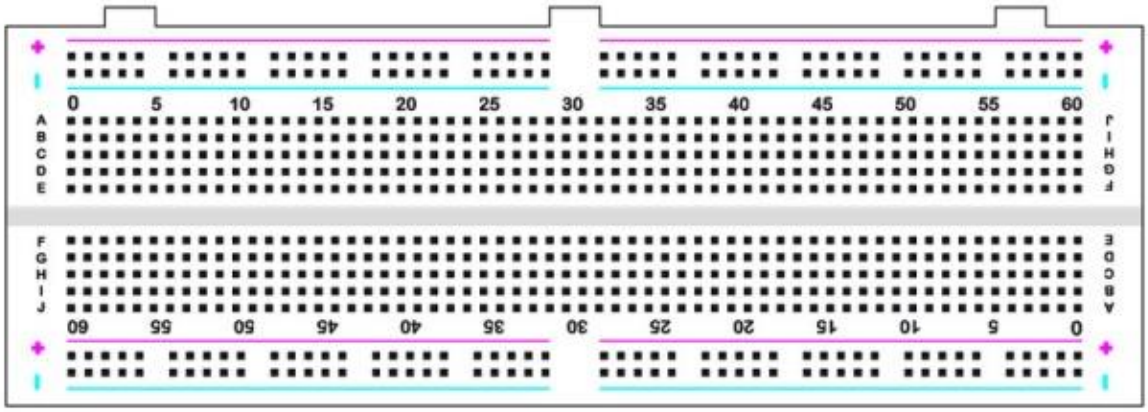
1. Arduino UNO R3:



Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analogue Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz

LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

2. Breadboard:



A breadboard (sometimes called protoboard) is essentially the foundation to construct and prototype electronics. A breadboard allows for easy and quick creation of temporary electronic circuits or to carry out experiments with circuit design. Breadboards enable developers to easily connect components or wires.

3. Jumper Wires:

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering. Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed.

Types of Jumper Wires

Jumper wires typically come in three versions: male-to-male, male-to-female and female-to-female. The difference between each is in the end point of the wire.

Zed-King

Male to Male



Female to Female



Male to Female



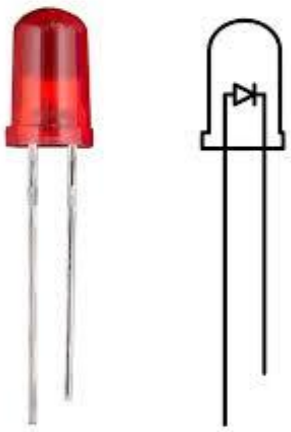
4. Registers:

A passive electrical component with two terminals that are used for either limiting or regulating the flow of electric current in electrical circuits. The main purpose of resistor is to reduce the current flow and to lower the voltage in any particular portion of the circuit.



5. LED:

The LED, also known as a Light Emitting Diode, is a component that emits light. The Grove Red LED module is a digital component that uses only the output, not the input.



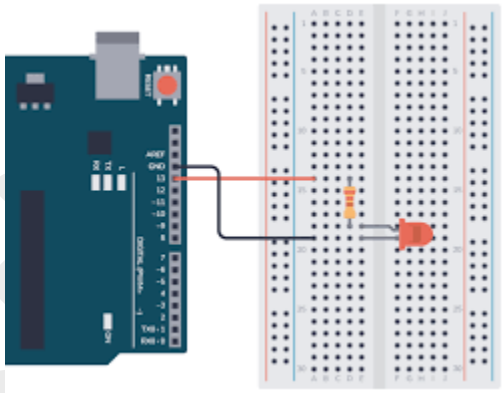
Zed-King Institute

Practical -1

Object Code:-

Write a program to Blink default Light Emitting Diode(LED) on Arduino board with the delay of 2 sec.

Components Needed:



1. **Arduino Uno**
2. **LED**
3. **Resistor**
4. **Breadboard and jumper wires**

Wiring:

Connect the Resistor and LED:

- ✓ Insert the longer leg (anode) of the LED into the breadboard.
- ✓ Insert one end of the resistor into the same row as the LED's anode.
- ✓ Insert the other end of the resistor into a GND (ground) row on the breadboard.

Connect to Arduino:

- ✓ Insert a jumper wire from the Arduino's digital pin 13 (or any other digital pin) to the same row as the LED's anode.
- ✓ Insert a jumper wire from the GND pin on the Arduino to a GND row on the breadboard.

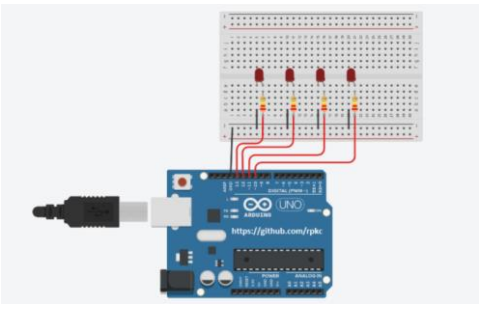
Zed-King Institute

Arduino Code:

```
// Blink an LED connected to pin 13, with a 2-second delay.

const int ledPin = 13; // the number of the LED pin

void setup() {
  // initialize the digital pin as an output.
  pinMode(ledPin, OUTPUT);
}

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

Practical -2

Object Code: Write a program to interface LEDs on pin no. 10,11,12,13 and blink alternatively at the delay of 1 sec.



Components Needed:

Zed-King Institute

1. **Arduino Uno**
2. **LED**
3. **Resistor**
4. **Breadboard and jumper wires**

Wiring:

Connect the LEDs:

- Connect the anode (longer leg) of each LED to the corresponding pin (10, 11, 12, 13) on the Arduino.
- Connect the cathode (shorter leg) of each LED to a 220-ohm resistor.
- Connect the other end of each resistor to ground on the Arduino.

Arduino Code:

```
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
```

Zed-King Institute

```
}
void loop()
{
 digitalWrite(10, HIGH);
  digitalWrite(11, LOW);
  digitalWrite(12, LOW);
  digitalWrite(13, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(10, LOW);
  digitalWrite(11, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(13, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(10, LOW);
  digitalWrite(11, LOW);
  digitalWrite(12, HIGH);
```

Zed King Institute

```
    digitalWrite(13, LOW);
    delay(1000); // Wait for 1000 millisecond(s)
    digitalWrite(10, LOW);
    digitalWrite(11, LOW);
    digitalWrite(12, LOW);
    digitalWrite(13, HIGH);
    delay(1000); // Wait for 1000 millisecond(s)
}
```

Practical -3

Object Code:-Write a program to run pattern(s) on LEDs connect at pins 10,11,12,13. Pattern example:

on, off, off, off
off, on, off, off
off, off, on, off
off, off, off, on
on, on, off, off
off, on, on, off
off, off, on, on

Components Needed:

Zed-King Institute

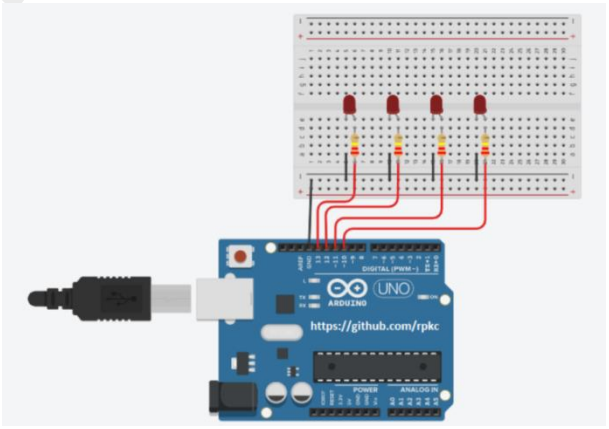
1. **Arduino Uno**
2. **LEDs**
3. **Resistor**
4. **Breadboard and jumper wires**

Wiring:

Connect the LEDs:

- Connect the anode (longer leg) of each LED to the corresponding pin (10, 11, 12, 13) on the Arduino.
- Connect the cathode (shorter leg) of each LED to a 220-ohm resistor.
- Connect the other end of each resistor to ground on the Arduino.

Arduino Code:




```
void setup()
{
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
}
```

Zed-King Institute

```
void loop()
{
 digitalWrite(10, HIGH);
 digitalWrite(11, LOW);
 digitalWrite(12, LOW);
  digitalWrite(13, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(10, LOW);
  digitalWrite(11, HIGH);
  digitalWrite(12, LOW);
  digitalWrite(13, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(10, LOW);
  digitalWrite(11, LOW);
  digitalWrite(12, HIGH);
  digitalWrite(13, LOW);
```

Zed-King Institute

```
delay(1000); // Wait for 1000 millisecond(s)
digitalWrite(10, LOW);
digitalWrite(11, LOW);
digitalWrite(12, LOW);
digitalWrite(13, HIGH);
delay(1000); // Wait for 1000 millisecond(s)
digitalWrite(10, HIGH);
digitalWrite(11, HIGH);
digitalWrite(12, LOW);
digitalWrite(13, LOW);
delay(1000); // Wait for 1000 millisecond(s)
digitalWrite(10, LOW);
digitalWrite(11, HIGH);
digitalWrite(12, HIGH);
digitalWrite(13, LOW);
delay(1000); // Wait for 1000 millisecond(s)
```

Zed King Institute

```
    digitalWrite(10, LOW);

    digitalWrite(11, LOW);

    digitalWrite(12, HIGH);

    digitalWrite(13, HIGH);

    delay(1000); // Wait for 1000 millisecond(s)

}
```

Practical - 4

Object Code:-

Write a program to run traffic light system on LEDs connect at pins with the delay of 1 min.

Components Needed:

5. **Arduino Uno**
6. **LEDs**
7. **Resistor**
8. **Breadboard and jumper wires**

Wiring:

Connect the LEDs:

- Connect the anode (longer leg) of each LED to the corresponding pin (8,9,10) on the Arduino.
- Connect the cathode (shorter leg) of each LED to a 220-ohm resistor.
- Connect the other end of each resistor to ground on the Arduino.

Zed-King Institute

Arduino Code:

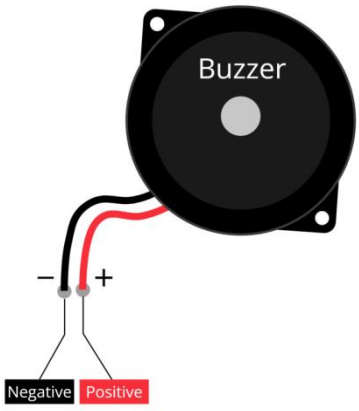
```
void setup()
{
  pinMode(10, OUTPUT);  // for red light
  pinMode(11, OUTPUT);  // for yellow light
  pinMode(12, OUTPUT);  // for green light
}
void loop()
{
digitalWrite(10, HIGH);
 digitalWrite(11, LOW);
 digitalWrite(12, LOW);
  delay(1000); // Wait for 60000 millisecond(s)(1min)
  digitalWrite(10, LOW);
  digitalWrite(11, HIGH);
  digitalWrite(12, LOW);
```

Zed King Institute

```
  delay(60000); // Wait for 60000 millisecond(s)(1 min)
 digitalWrite(10, LOW);
 digitalWrite(11, LOW);
 digitalWrite(12, HIGH);
  delay(60000); // Wait for 60000 millisecond(s)(1min)
}
```

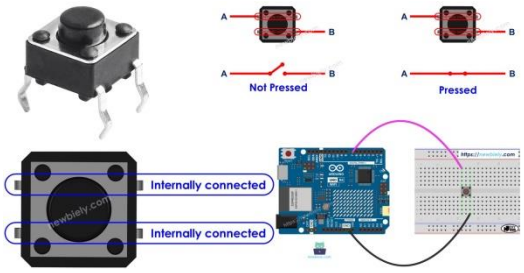
6. Buzzer:

A buzzer has two pins: a positive pin and a negative pin. The positive pin is usually marked with a plus sign (+) or is longer than the negative pin. The negative pin is usually marked with a minus sign (-) or is shorter than the positive pin.



7. Button:

The buttons are similar to switches that create and break electrical connections in the circuits. The button plays a transition between ON and OFF state. A single press turns the state ON, while another press turns the state OFF. It means that the button connects the two points in a circuit when we press them.



Zed-King Institute

Practical - 5

Object Code:-

Write a program to interface buzzer with Arduino board to buzz on/off with the delay of 1sec.

Components Needed

1. **Arduino UNO**
2. **Buzzer**
3. **Jumper Wires**
4. **Breadboard**

Wiring:

- **Connect the buzzer's positive terminal to a digital pin on your Arduino board.** (Let's assume you connect it to pin 13 for this example).
- **Connect the buzzer's negative terminal to ground on the Arduino board.**

Arduino Code:

```
const int buzzerPin = 13;  // Define the buzzer pin

void setup() {
pinMode(buzzerPin, OUTPUT);  // Set the buzzer pin as output
}
```

Zed-King Institute

```
void loop() {
  digitalWrite(buzzerPin, HIGH);  // Turn the buzzer ON
  delay(1000);                    // Wait for 1 second
  digitalWrite(buzzerPin, LOW);  // Turn the buzzer OFF
  delay(1000);                    // Wait for 1 second
}
```

Alternate Code:-

```
const int buzzerPin = 13;  // Define the buzzer pin

void setup() {
pinMode(buzzerPin, OUTPUT);  // Set the buzzer pin as output
}

void loop() {
  tone(13,1000);  // buzzer beep
  delay(1000);    // Wait for 1 second
  noTone(13);  // buzzer OFF
  delay(1000); // Wait for 1 second Turn the
}
```

Practical - 6

Object Code:

Write a program to interface LED and Buzzer With

Zed-King Institute

Arduino Board keeping the condition that whenever LED Glows, there is a Buzzer ringing and when LED is off then buzzer is also put on off.

Components Needed:

1. **Arduino board** (like Uno, Nano, etc.)
2. **LED**
3. **Resistor** (220Ω for the LED)
4. **Buzzer**
5. **Breadboard and jumper wires**

Circuit Diagram:

1. **LED**:
   - Connect the anode (longer leg) of the LED to a digital output pin (e.g., pin 9) through a 220Ω resistor.
   - Connect the cathode (shorter leg) of the LED to GND.
2. **Buzzer**:
   - Connect one pin of the buzzer to a digital output pin (e.g., pin 8).
   - Connect the other pin of the buzzer to GND.

Arduino code:-

```
// Define the pin numbers for the LED and

Buzzer const int ledPin = 9;

const int buzzerPin

= 8; void setup() {

  // Set LED and Buzzer pins
```

Zed-King Institute

```
  as output pinMode(ledPin,
  OUTPUT);
  pinMode(buzzerPin,
  OUTPUT);
}
void loop() {
  // Turn on LED and Buzzer
  digitalWrite(ledPin, HIGH);
  digitalWrite(buzzerPin,
  HIGH);
  // Keep them on for 2
  seconds delay(2000);
  // Turn off LED and Buzzer
  digitalWrite(ledPin, LOW);
  digitalWrite(buzzerPin,
  LOW);
  // Keep them off for 2
  seconds delay(2000);
}
```

Zed-King Institute

Practical - 7

OBJECT CODE:-

Write a program to interface button and LED, so that LED blinks/glow when button is pressed.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **LED** (any standard 5mm LED)
3. **220-ohm Resistor** (to limit current and protect the LED)
4. **Pushbutton** (momentary pushbutton switch)
5. **10k-ohm Resistor** (for the pull-down resistor)
6. **Breadboard** (optional, for easy connections)
7. **Jumper Wires**

Circuit Diagram:

1. **Connect the LED**:
    - **Anode (longer leg)** of the LED to a digital pin on the Arduino (e.g., pin 9).
    - **Cathode (shorter leg)** of the LED to one end of the 220-ohm resistor.
    - **Other end of the resistor** to the Arduino GND (ground) pin.
2. **Connect the Pushbutton**:
    - One terminal of the pushbutton to a digital pin on the Arduino (e.g., pin 7).
    - The other terminal of the pushbutton to the Arduino GND (ground) pin.
    - Connect a 10k-ohm resistor between the button pin (pin

Zed-King Institute

7) and

- GND to act as a pull-down resistor.

Arduino code:-

```
// Define the pin for the LED and
the button const int ledPin = 9;
const int buttonPin= 7;
void setup() {
 // Set the LED pin as an output
 pinMode(ledPin, OUTPUT);
 // Set the button pin as an input
 pinMode(buttonPin, INPUT);
}
void loop() {
 // Read the state of the button
 int buttonState = digitalRead(buttonPin);
 // Check if the button is pressed if (buttonState == HIGH)
 {
  // Turn the LED on
  digitalWrite(ledPin, HIGH);
 } else {
```

Zed-King Institute

```
    digitalWrite(ledPin, LOW);
  }
}
```

Practical - 8

Object code:-

Write a program to interface Button, Buzzer and LED, Whenever the Button is pressed the Buzzer gives beep for 100ms and LED status is toggled.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **LED** (any standard 5mm LED)
3. **220-ohm Resistor** (for the LED, to limit current and protect the LED)
4. **Buzzer** (active buzzer preferred for simplicity)
5. **Pushbutton** (momentary pushbutton switch)
6. **10k-ohm Resistor** (for the pull-down resistor)
7. **Breadboard** (optional, for easy connections)
8. **Jumper Wires**

Circuit Diagram:

1. **Connect the LED**:
   - **Anode (longer leg)** of the LED to a digital pin on the Arduino (e.g., pin 9).
   - **Cathode (shorter leg)** of the LED to one end of a

Zed-King Institute

220-ohm resistor.
    - **Other end of the resistor** to the Arduino GND (ground) pin.
2. **Connect the Buzzer**:
    - **Positive Terminal** of the buzzer to a digital pin on the Arduino (e.g., pin 8).
    - **Negative Terminal** of the buzzer to the Arduino GND (ground) pin.
3. **Connect the Pushbutton**:
    - One terminal of the pushbutton to a digital pin on the Arduino (e.g., pin 7).
    - The other terminal of the pushbutton to the Arduino GND (ground) pin.
    - Connect a 10k-ohm resistor between the button pin (pin 7) and GND to act as a pull-down resistor.

Arduino code:-

```
// Define the pins for the LED, buzzer, and
button const int ledPin = 9;
const int buzzerPin = 8;
const int buttonPin = 7;
// Variable to keep track of the
LED state bool ledState = true;
void setup() {

 // Set the LED and buzzer pins as outputs

 pinMode(ledPin, OUTPUT);

 pinMode(buzzerPin, OUTPUT);
```

Zed-King Institute

```
  // Set the button pin as an input
  pinMode(buttonPin, INPUT);
}

void loop() {

  // Read the state of the button

  int buttonState = digitalRead(buttonPin);

  // Check if the button is pressed
  if (buttonState == HIGH) {

    // Beep the buzzer for 100 milliseconds
    digitalWrite(buzzerPin, HIGH);

    delay(100);
    digitalWrite(buzzerPin, LOW);

    // Toggle the LED state

    ledState = !ledState;
    digitalWrite(ledPin, ledState ? HIGH : LOW);

    // Debounce delay to avoid multiple toggles from a single button
    press

    delay(100);
  }

}
```

Practical - 9

Object code:-

Zed-King Institute

Write a program to interface Button, Buzzer and LED, Whenever the Button is pressed the Buzzer gives beep for 350ms and LED status is toggled.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **LED** (any standard 5mm LED)
3. **220-ohm Resistor** (for the LED, to limit current and protect the LED)
4. **Buzzer** (active buzzer preferred for simplicity)
5. **Pushbutton** (momentary pushbutton switch)
6. **10k-ohm Resistor** (for the pull-down resistor)
7. **Breadboard** (optional, for easy connections)
8. **Jumper Wires**

Circuit Diagram:

1. **Connect the LED**:
    - **Anode (longer leg)** of the LED to a digital pin on the Arduino (e.g., pin 9).
    - **Cathode (shorter leg)** of the LED to one end of a 220-ohm resistor.
    - **Other end of the resistor** to the Arduino GND (ground) pin.
2. **Connect the Buzzer**:
    - **Positive Terminal** of the buzzer to a digital pin on the Arduino (e.g., pin 8).
    - **Negative Terminal** of the buzzer to the Arduino GND (ground) pin.
3. **Connect the Pushbutton**:

Zed King Institute

- One terminal of the pushbutton to a digital pin on the Arduino (e.g., pin 7).
- The other terminal of the pushbutton to the Arduino GND (ground) pin.
- Connect a 10k-ohm resistor between the button pin (pin 7) and GND to act as a pull-down resistor.

Arduino code:-

```
// Define the pins for the LED, buzzer, and
button const int ledPin = 9;
const int buzzerPin = 8;
const int buttonPin = 7;
// Variable to keep track of the
LED state bool ledState = true;
void setup() {

  // Set the LED and buzzer pins as outputs

  pinMode(ledPin, OUTPUT);

  pinMode(buzzerPin, OUTPUT);


  // Set the button pin as an input
  pinMode(buttonPin, INPUT);
}

void loop() {

  // Read the state of the button

  int buttonState = digitalRead(buttonPin);
```

Zed-King Institute

```
  // Check if the button is pressed
  if (buttonState == HIGH) {

    // Beep the buzzer for 350 milliseconds
    digitalWrite(buzzerPin, HIGH);

    delay(350);
    digitalWrite(buzzerPin, LOW);

    // Toggle the LED state

    ledState = !ledState;
    digitalWrite(ledPin, ledState ? HIGH : LOW);

    // Debounce delay to avoid multiple toggles from a single button
    press

    delay(200);
  }

}
```

LCD Display:

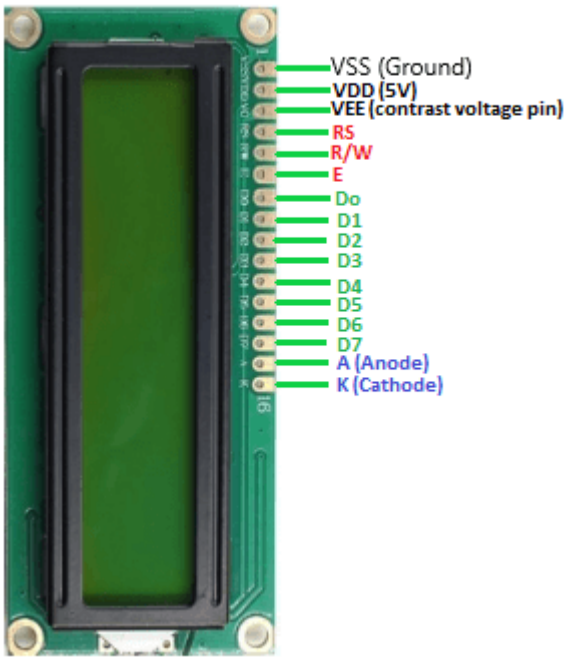
The LCD (**Liquid Crystal Display**) is a type of display that uses the liquid crystals for its operation. The library that allows us to control the LCD display is called **Liquid Crystal Library**

LCD Structure

The LCD display has a 16-pin interface.
The structure of the LCD is shown below:

Zed-King Institute

LCD



The 16-pins present on the LCD display are discussed below:

- **RS**

The **Register Select (RS)** pin controls the memory of the LCD in which we write the data. We can select either the **data register** or the **instruction register**.

- **R/W**

The Read/Write pin selects the reading or writing mode.

- **E**

The **Enable (E)** mode is used to enable the writing to the registers. It sends the data to the data pins when the mode is HIGH.

- **D0 to D7**

  - These are eight data pins numbered as D0, D1, D3, D3, D4, D5, D6, and D7. We can set the state of the data pin either HIGH or LOW.
  - These are eight data pins numbered as D0, D1, D3, D3, D4, D5, D6, and D7.

Zed-King Institute

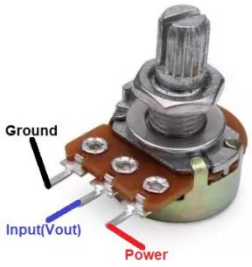
We can set the state of the data pin either HIGH or LOW.

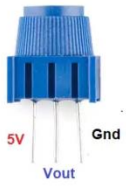
- Pin 1 of the LCD is the Ground pin, and pin 2 is the Vcc or the voltage source pin.
- The pin 3 of the LCD is the **VEE** or the contrast pin. For example, we can connect the potentiometer's output to the VEE and can adjust the contrast of the LCD.
- The A and K pins are also called as **Backlight** pins (Bklt+ and Bklt-).

The LCDs can be controlled in 4-bit or 8-bit modes, which requires 7 and 11 Input/Output pins from the particular Arduino board.

Potentiometer:

Potentiometers adjust the LCD brightness.





Practical -10

Object code:-

Write a program in Arduino to interface 2*16 LCD to Display characters “congratulations”.

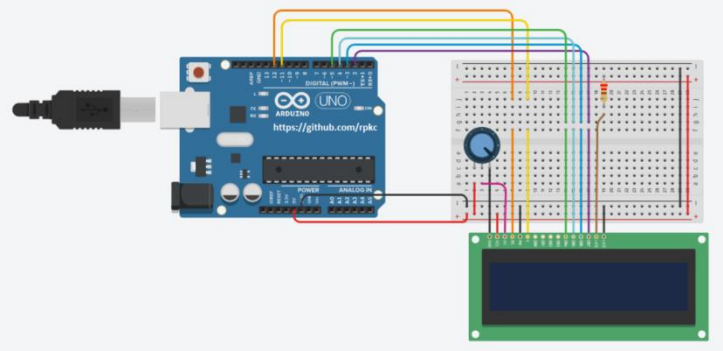
Hardware Setup:

1. **LCD Connections:**
   - **VSS** (Ground) to Arduino GND
   - **VDD** (Power) to Arduino 5V
   - **V0** (Contrast) to the center pin of a potentiometer (with the other two pins

Zed-King Institute

connected to 5V and GND) for contrast adjustment

- **RS** (Register Select) to Arduino pin 12
- **RW** (Read/Write) to Arduino GND (to write only)
- **E** (Enable) to Arduino pin 11
- **D0** to **D3** (Data Pins 0-3) not used in 4-bit mode
- **D4** to **D7** (Data Pins 4-7) to Arduino pins 10, 9, 8, 7 respectively
- **A (LED+)**(Anode for backlight) to Arduino 5V (through a resistor if needed)
- **K(LED-)** (Cathode for backlight) to Arduino GND



Arduino code:-

```
#include <LiquidCrystal.h>

// Initialize the LCD with numbers of the interface pins

LiquidCrystal lcd(12, 11, 10, 9, 8, 7);

void setup() {

 // Set up the LCD's number of columns and rows
```

Zed-King Institute

```
  lcd.begin(16, 2);

  // Print a message to the LCD

  lcd.setCursor(0, 0); // Set cursor to column 0, row 0

  lcd.print("congratulations");

}

void loop() {

  // Nothing to do here}
```

Keypad:

A **keypad** is a small input device used to enter data or control devices. It consists of a grid of buttons arranged in rows and columns. When you press a button, it connects a specific row and column to send a signal to a microcontroller or device.

Pinout Information:

A typical keypad, like a 4x4 matrix, has 8 pins:

- 4 **row pins**: These are connected to the rows of buttons.
- 4 **column pins**: These are connected to the columns of buttons.

Practical -11

Object code:-

Write a program to interface LCD and keypad with Arduino Board and display the key pressed from keypad

Zed-King Institute

on LCD.

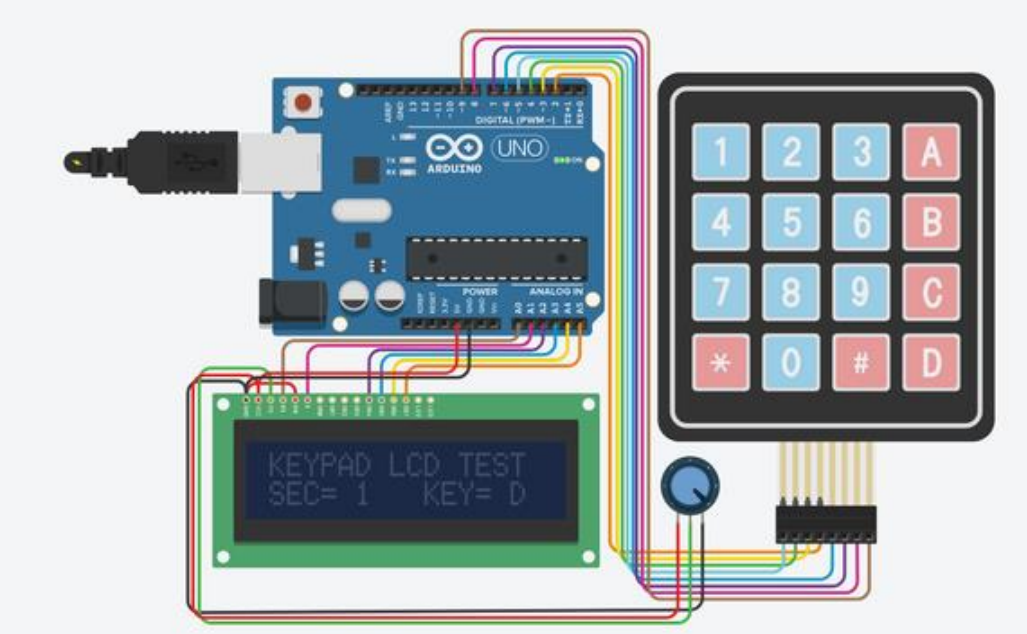
Components Needed

1. Arduino Uno (or similar board)
2. 16x2 LCD
3. 4x4 Matrix Keypad
4. Breadboard and jumper wires

Wiring Diagram

1. **LCD Wiring (using the LiquidCrystal library)**:
   - **VSS** to GND
   - **VDD** to 5V
   - **VO** to a 10k ohm potentiometer (to adjust contrast) and the other end to GND
   - **RS** to digital pin A0
   - **RW** to GND
   - **E** to digital pin A1
   - **D0-D3** to GND (not used for 4-bit mode)
   - **D4** to digital pin A2
   - **D5** to digital pin A3
   - **D6** to digital pin A4
   - **D7** to digital pin A5
   - **A** (LED+) to 5V
   - **K** (LED-) to GND
2. **Keypad Wiring**:
   - Connect the 4 rows (R1, R2, R3, R4) to digital pins 5, 4, 3, and 2 respectively.
   - Connect the 4 columns (C1, C2, C3, C4) to digital pins 6, 7, 8, and 9 respectively.

Zed King Institute



Arduino Code:-

```
#include<LiquidCrystal.h>

#include <Keypad.h>
// LCD pin configuration
LiquidCrystal lcd(A0,A1,A2,A3,A4,A5);
const byte ROWS = 4; // Four rows
const byte COLS = 4; // Four columns
char keys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'},
};
```

Zed-King Institute

```
byte rowPins[ROWS] = {5, 4, 3, 2}; // Connect to the row pins of the keypad

byte colPins[COLS] = {6, 7, 8, 9}; // Connect to the column pins of the keypad

Keypad kpd = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);

void setup() {
  lcd.begin(16, 2); // Initialize the LCD with 16 columns and 2 rows
  lcd.print("Keypad Input:");
}
void loop() {
  char key = kpd.getKey();


    lcd.setCursor(0, 1); // Move to the start of the second row
    lcd.print(“Sec=”); // Display the pressed key
  lcd.print(millis()/1000);

  if (key)
  {
  lcd.setCursor(9,1);
  lcd.print(“Key=”);
  lcd.print(key);
  }
  }
}
```

LM35 Temperature Sensor:

The **LM35** is a temperature sensor that provides an analog output directly proportional to the temperature in Celsius. It’s widely used because of its simplicity, accuracy, and low power consumption.

Zed-King Institute

Pinout:

The LM35 typically has **3 pins**:

1. **Vcc**: Power supply pin (usually 5V or 9V).
2. **Output**: The analog voltage output that corresponds to the temperature.
3. **Ground (GND)**: The reference ground pin.

Practical -12

Object code:-

Write a program to interface LM 35/TMP temperature sensor and display the readings on LCD.

Hardware Required

1. **LM35 Temperature Sensor**
2. **16x2 LCD Display (using HD44780 driver)**
3. **Arduino Board (e.g., Arduino Uno)**
4. **Resistors (10kΩ for the LCD potentiometer)**
5. **Breadboard and Jumper Wires**
6. **Power Source (e.g., USB connection to the Arduino)**

Connections

1. **LM35 Sensor**
   - **VCC**: Connect to the 5V pin on the Arduino.
   - **GND**: Connect to the GND pin on the Arduino.
   - **OUT**: Connect to an analog input pin on the Arduino (e.g., A0).
2. **16x2 LCD Display**

Zed-King Institute

- **VSS**: Connect to GND.
- **VDD**: Connect to 5V.
- **VO**: Connect to the middle pin of a 10kΩ potentiometer. The ends of the potentiometer connect to 5V and GND to adjust contrast.
- **RS**: Connect to pin 12 on Arduino.
- **RW**: Connect to GND.
- **E**: Connect to pin 11 on Arduino.
- **D0-D3**: Connect to GND (not used in 4-bit mode).
- **D4-D7**: Connect to pins 5, 4, 3, and 2 on Arduino respectively.
- **A (Anode)**: Connect to 5V (for backlight).
- **K (Cathode)**: Connect to GND (for backlight).

Arduino Code:-

```
#include <LiquidCrystal.h>

// Initialize the LCD with the numbers of the interface pins

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// LM35 Sensor connected to Analog pin A0

const int sensorPin = A0;

void setup() {

 // Set up the LCD's number of columns and rows

 lcd.begin(16, 2);

 // Print a startup message
```

Zed-King Institute

```
    lcd.print("Temperature:");
  }
  void loop() {
    // Read the analog value from the sensor

    int sensorValue = analogRead(sensorPin);
 // Convert the analog value to voltage (assuming a 5V reference volt)
float voltage = sensorValue * (5.0 / 1023.0);
// Convert the volt to temp  in Celsius (TMP gives 10mV per degree C)
    float temperatureC = voltage * 100.0;
    // Print the temperature to the LCD
    lcd.setCursor(0, 1); // Move cursor to the second line
    lcd.print("Temp: ");
    lcd.print(temperatureC);
    lcd.print(" C");
    // Wait a bit before repeating
    delay(1000);
  }
```

MQ135 Sensor:

The **MQ135** is a gas sensor designed to detect various harmful gases like ammonia

Zed King Institute

(NH3), carbon dioxide (CO2), benzene, alcohol, and others. It is commonly used in air quality monitoring systems.

Pinout:

The MQ135 sensor typically has **4 pins**:

1. **Vcc**: Power supply pin (typically 5V).
2. **GND**: Ground pin (connected to the system ground).
3. **Analog Output (A0)**: This pin provides an analog voltage that is proportional to the concentration of the detected gas.
4. **Digital Output (D0)**: This pin provides a digital signal (high or low) based on a threshold value set for the gas concentration.

Practical -13

Object Code:-

Write a program to Sense the air index value from MQ135 and print it's value on LCD.

Components Needed

1. **Arduino** (e.g., Arduino Uno)
2. **MQ135 Gas Sensor**
3. **16x2 LCD with I2C Interface**
4. **Breadboard and Connecting Wires**

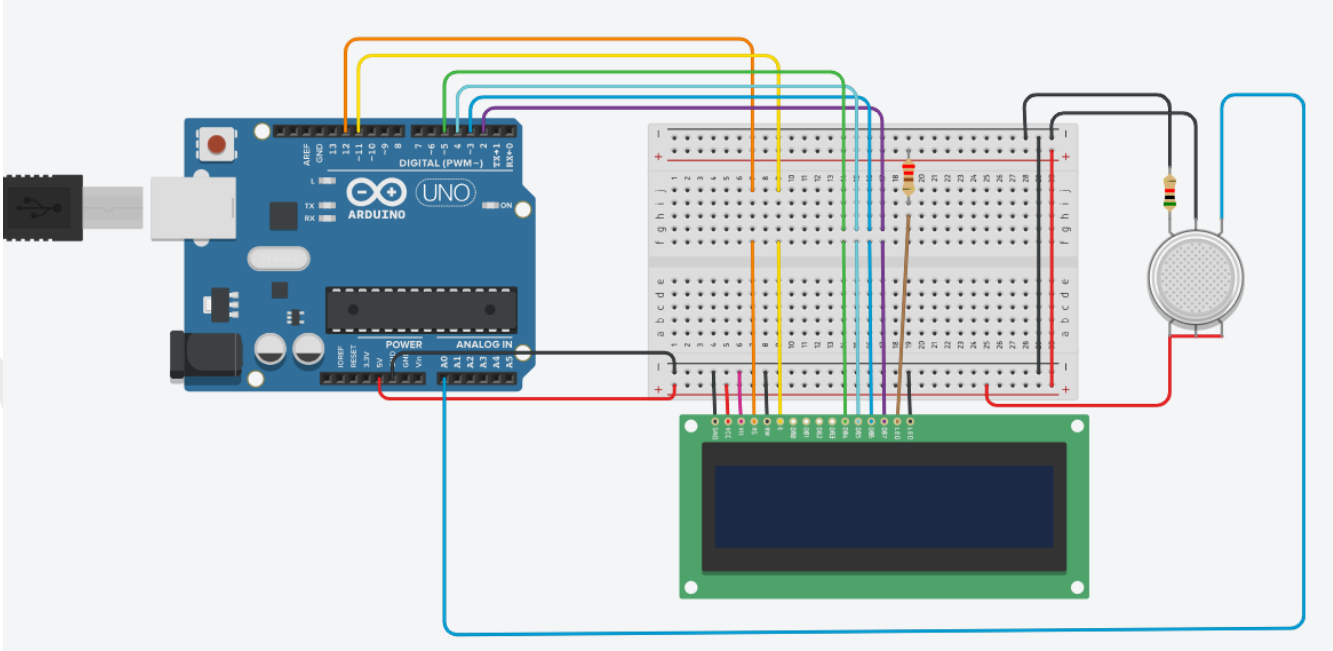
Wiring:-

1. **MQ135 Gas Sensor:**
    - **B1,B2,H2** to Arduino 5V
    - **A2** to Arduino A0
    - **H1** to Arduino GND
    - **A1** to Arduino GND through register

Zed King Institute

2. LCD :

- RS pin to digital pin 12
- Enable pin to digital pin 11
- D4 pin to digital pin 5
- D5 pin to digital pin 4
- D6 pin to digital pin 3
- D7 pin to digital pin 2
- R/W pin to ground
- VSS pin to ground
- VCC pin to 5V
- Contrast pin to ground



Arduino Code:-

Zed-King Institute

```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int sensor = A0;
int val = 0;
int limit = 40;

void setup() {
   // set up the LCD's number of columns and rows:

 lcd.begin(16, 2);

 // Print a message to the LCD.

 lcd.print("Air Monitoring");
 lcd.setCursor(0, 1);
 lcd.print("System");
 delay(500);
 lcd.clear();
}
void loop() {

 val = analogRead(sensor);
 val = map(val, 306, 750, 0, 100);

/*Mapping the output voltage values from the Gas Sensor to values in the
range from 0 to 100.*/
```

Zed-King Institute

```
//If a harmful gas is detected.

   if (val > limit) {
    lcd.setCursor(0, 0);
    lcd.print("Harmful Gas is");  //Print a message to the LCD.
    lcd.setCursor(0, 1);
    lcd.print("Detected");
   }

   else {
    lcd.setCursor(0, 0);
    lcd.print("No Harmful Gas");  //Print a message to the LCD.
    lcd.setCursor(0, 1);
    lcd.print("Air is Clear");
    delay(100);

     }
}
```

Practical -14

Object Code:-

Write a program for interfacing Air quality sensor pollution (e.g.MQ135) to Arduino board and display dots generated by sensor on LCD.

Components Needed

5. **Arduino** (e.g., Arduino Uno)
6. **MQ135 Gas Sensor**

Zed-King Institute

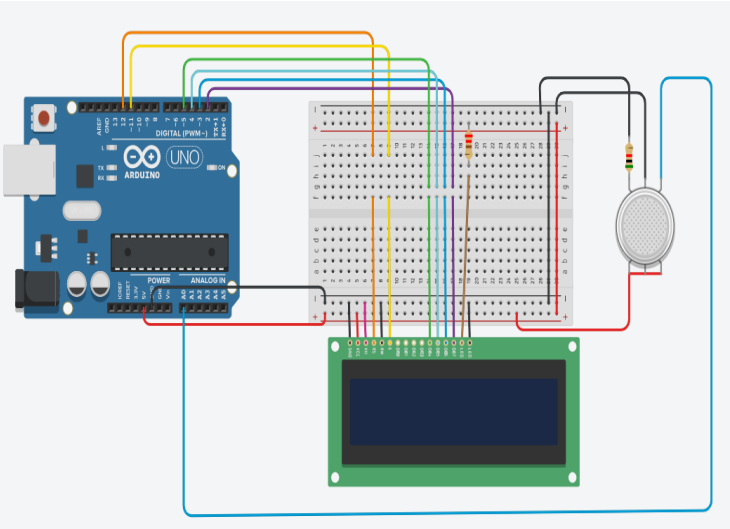
7. **16x2 LCD with I2C Interface**
8. **Breadboard and Connecting Wires**

Wiring:-

3. **MQ135 Gas Sensor:**
    - **B1,B2,H2** to Arduino 5V
    - **A2** to Arduino A0
    - **H1** to Arduino GND
    - **A1** to Arduino GND through register

4. **LCD :**

- **RS pin to digital pin 12**
- **Enable pin to digital pin 11**
- **D4 pin to digital pin 5**
- **D5 pin to digital pin 4**
- **D6 pin to digital pin 3**
- **D7 pin to digital pin 2**
- **R/W pin to ground**
- **VSS pin to ground**
- **VCC pin to 5V**
- **Contrast pin to ground**



Zed-King Institute

Arduino Code:

```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int sensor = A0;

void setup() {
   // set up the LCD's number of columns and rows:

 lcd.begin(16, 2);

 // Print a message to the LCD.

 lcd.print("Air Monitoring");
 lcd.setCursor(0, 1);
 lcd.print("System");
 delay(1000);
 lcd.clear();
}
void loop() {

 int airquality = analogRead(sensor);  // Read sensor value
 int dot = map(airquality, 0, 1023, 0, 16);  //Map Sensor value to no. of dots

lcd.clear(); // clear the display before printing new data

for (int i=0;i<dot;i++) {
```

Zed-King Institute

```
    lcd.print("."); //Print dots based on air quality
  }
  lcd.print("Air is clear");
  delay(5000); // Wait 5 seconds before the next reading

  }
```

DHT11/DHT22:

The **DHT11** and **DHT22** are popular digital temperature and humidity sensors used in many electronics projects, especially for measuring environmental conditions.

Pinout:

1. **VCC**: Power supply (3.3V to 6V).
2. **Data**: The digital data output pin.
3. **Ground (GND)**: Ground pin.

Practical -15

Object Code:-

Write a program to interface LCD and DHT11, displaying the value read from sensor DHT on LCD

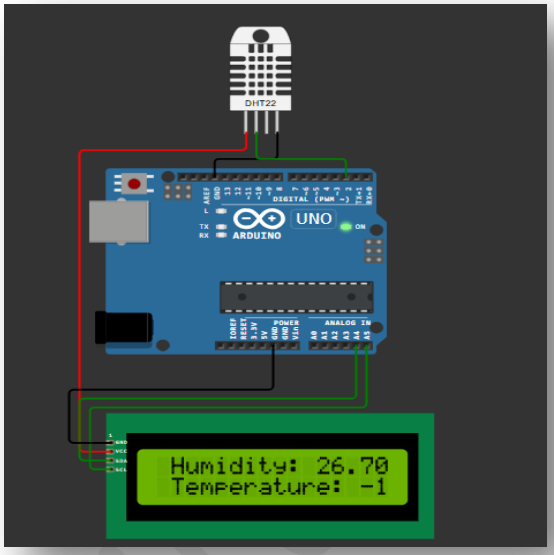
Components Needed

1. **Arduino Board** (e.g., Arduino Uno)
2. **DHT11 Sensor**
3. **16x2 LCD Display**
4. **10k Ohm Potentiometer** (for LCD contrast adjustment)
5. **Breadboard and Jumper Wires**

Connections:

Zed-King Institute

DHT11 Sensor:



- **VCC** -> Connect with LCD VCC pin
- **GND** -> GND on Arduino
- **DATA** -> Digital Pin 2 on Arduino

16x2 LCD(I2C):

- **GND** -> GND on Arduino
- **VCC**-> Connect with DHT11 VCC
- **SDA** -> A4 on Arduino
- **SCL**-> A5 on Arduino)

Arduino Code:

```
#include <DHT.h>

#include <LiquidCrystal_I2C.h>

// Define pins for DHT11 sensor


#define DHTPIN 2

#define DHTTYPE DHT22

// Initialize DHT sensor
DHT dht(DHTPIN, DHTTYPE);


LiquidCrystal_I2C  lcd(0x27,20,4);

void setup() {

 pinMode(DHTPIN, INPUT);

  lcd.init();
```

Zed-King Institute

```
  lcd.backlight();
  // Initialize the DHT sensor
  dht.begin();
}
void loop() {
  // Wait a few seconds between measurements
  delay(2000);
lcd.setCursor(1, 0);
float h = dht.readHumidity();
float t = dht.readTemperature();
if (isnan(h) || isnan(t)) {
  lcd.print("Failed to read from DHT sensor ");
    return;
  }
  lcd.print("Humidity: ");
  lcd.print(h);
  lcd.setCursor(1, 1);
  lcd.print("Temperature: ");
  lcd.print(t);
  delay(100);
```

Zed-King Institute

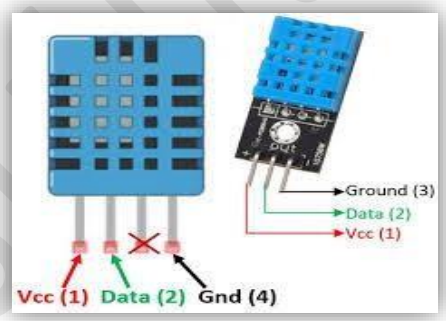
}

Practical -16

Object Code:-

Write a program to measure humidity using a DHT11 Humidity sensor and circuit using Arduino Uno Board.

Components Needed



- **Arduino Uno**
- **DHT11 Humidity Sensor**
- **Breadboard and jumper wires**
- **10k ohm resistor (usually needed for some sensors, but not always required for the DHT11)**

Wiring the DHT11 Sensor

1. DHT11 Pinout:

- **VCC**: Connects to 5V on the Arduino.
- **GND**: Connects to GND on the Arduino.
- **DATA**: Connects to a digital pin on the Arduino (commonly used pin is D2).
- **NC**: Not connected (usually not used).

2. Connections:

- Connect the **VCC** pin of the DHT11 to the **5V** pin on the Arduino.
- Connect the **GND** pin of the DHT11 to a **GND** pin

Zed-King Institute

on the Arduino.

- Connect the **DATA** pin of the DHT11 to **digital pin 2** on the Arduino (you can use another digital pin if you change the pin number in the code).

Arduino Code:

```
#include <DHT.h>

// Define the pin where the DATA pin of the DHT11 is connected

#define DHTPIN 2

// Define the type of sensor used

#define DHTTYPE DHT11

// Initialize the DHT

sensor DHT dht(DHTPIN,

DHTTYPE);

void setup() {

 // Start the serial

 communication

 Serial.begin(9600);

 // Initialize the DHT sensor

 dht.begin();

}

void loop() {
```

Zed-King Institute

```
  // Wait a few seconds between measurements
  delay(2000);
  // Read humidity (percentage)
  float humidity = dht.readHumidity();
  // Check if any reads failed and exit early (to try
  again) if (isnan(humidity)) {
    Serial.println("Failed to read from DHT
    sensor!"); return;
  }
  // Print the humidity value to the Serial
  Monitor Serial.print("Humidity: ");
  Serial.print(humidity);
  Serial.println(" %");
}
```

LDR:

An LDR **(Light Dependent Resistor), also known as a** photoresistor**, is a type of resistor whose resistance decreases as the intensity of light falling on it increases. In simple terms, it "senses" light and changes its resistance based on how bright the light is.**

Working:

- When there is **low light** (darkness), the resistance of the LDR is **high**, which means it doesn't allow much current to pass through.
- When there is **high light** (brightness), the resistance of the LDR is **low**, allowing more current to pass through.

Zed-King Institute

Pinout:

An LDR is a simple component and typically has **2 pins**:

1. **Pin 1**: Connected to the power supply or one side of the circuit.
2. **Pin 2**: Connected to the rest of the circuit where the variable resistance will influence the current flow.

Practical -17

Object code:-

Write a program to interface light dependent resistor (LDR) with LED using an Arduino development board.

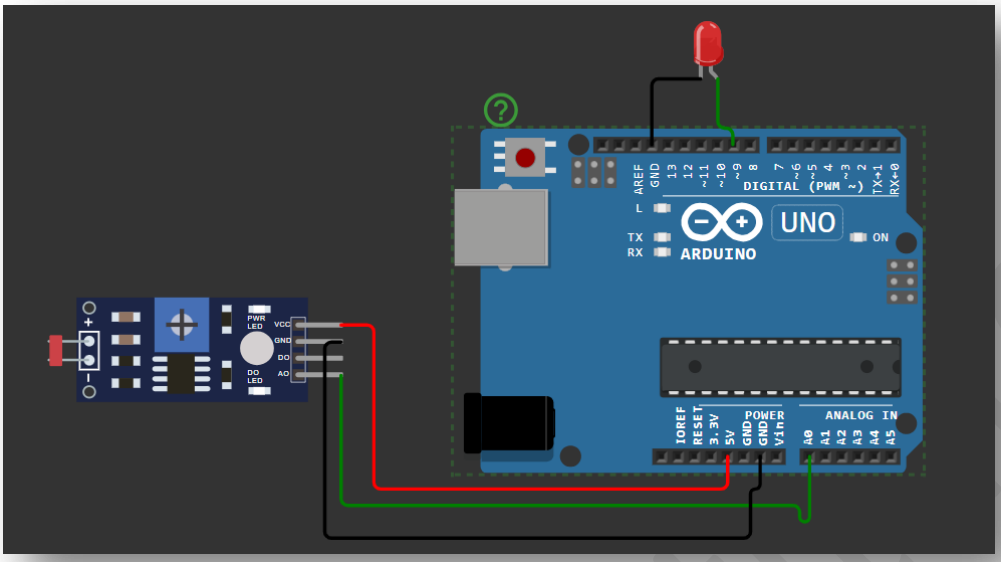
Components Needed

1. Arduino Uno
2. Light Dependent Resistor (LDR)
3. LED
4. 10k ohm resistor (for LDR)
5. Breadboard & Jumper Wires

Wiring Diagram

- **LED**
  - **Connect Anode to Arduino pin no. 9**
  - **Connect Cathode to Arduino GND**
- **Photo resistor (LDR)**
  - **Connect Terminal 1 to Arduino GND with resistor also with A0**
  - **Connect Terminal 2 to Arduino 5V.**

Zed-King Institute



Arduino Code:

```
int sensorValue = 0;

void setup()

{

  pinMode(A0, INPUT);

  pinMode(9, OUTPUT);

}

void loop()

{

  sensorValue = analogRead(A0); // for reading the value from sensor

   analogWrite(9, map(sensorValue, 0, 1023, 255, 0));
```

Zed-King Institute

```
  // glow LED according to the LDR sensor
  delay(100); // wait for 100 milliseconds
}
```

IR Sensor:

An **IR (Infrared) sensor** is a device that detects infrared radiation, typically used to sense the presence, distance, or motion of objects. It works by emitting infrared light (which is invisible to the human eye) and then detecting the reflection of that light from objects.

Pinout (for basic IR sensor module):

1. **VCC**: Power supply pin (typically 3.3V or 5V).
2. **GND**: Ground pin.
3. **OUT**: Output pin, which sends a signal (usually HIGH or LOW) when it detects an object or infrared light.

Practical -18

Object Code

Write a program to interface light dependent resistor (LDR) with an IR sensor using an Arduino development board.

Components Needed

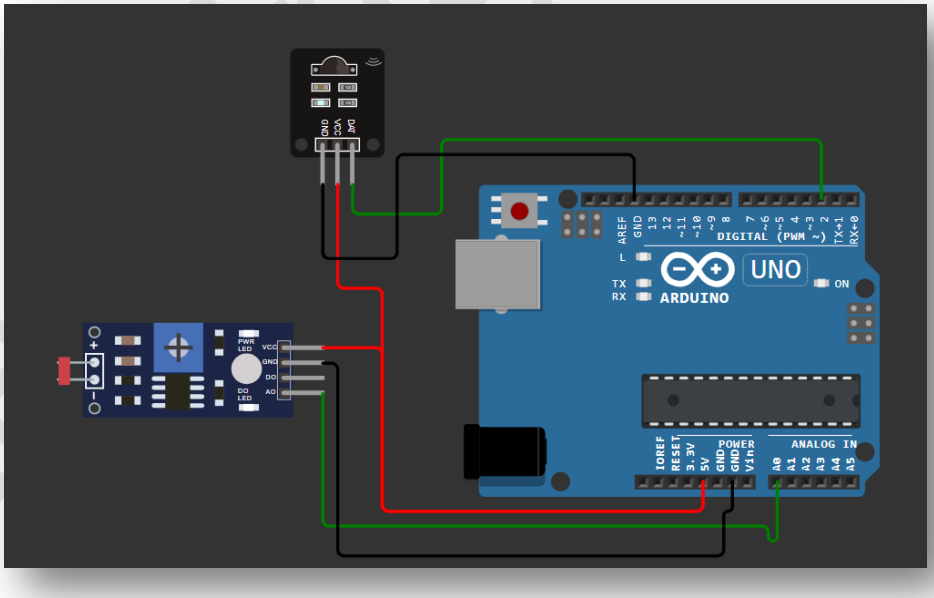
- Arduino Uno
- Light Dependent Resistor (LDR)
- IR Sensor

Zed-King Institute

- 10k ohm resistor (for LDR)
- Breadboard and jumper wires

Wiring Diagram

1. **LDR Circuit**:
    - Connect VCC to **5V** on the Arduino.
    - Connect GND to **GND** on the Arduino.
    - Connect the junction between the LDR and resistor to **analog pin A0** on the Arduino.
2. **IR Sensor Circuit**:
    - Connect the **VCC** pin of the IR sensor to **5V** on the Arduino.
    - Connect the **GND** pin of the IR sensor to **GND** on the Arduino.
    - Connect the **OUT** pin of the IR sensor to a **digital pin (e.g., D2)** on the Arduino.



Zed-King Institute

Arduino Code

```
// Define pin numbers
#define LDR_PIN A0      // Analog pin for LDR
#define IR_SENSOR_PIN 2  // Digital pin for IR sensor

void setup() {
  // Start serial communication
  Serial.begin(9600);

  // Initialize the IR sensor pin as input
  pinMode(IR_SENSOR_PIN, INPUT);
}

void loop() {
  // Read LDR value (analog)
  int ldrValue = analogRead(LDR_PIN);

  // Read IR sensor value (digital)
  int irValue = digitalRead(IR_SENSOR_PIN);

  // Print LDR value to Serial Monitor
  Serial.print("LDR Value: ");
  Serial.print(ldrValue);

  // Print IR sensor status to Serial Monitor
  Serial.print(" | IR Sensor Value: ");
  if (irValue == HIGH) {
  Serial.println("Object detected");
  }
  else {
    Serial.println("No object detected");
  }

  // Delay between readings
```

Zed-King Institute

```
  delay(500);
};
```

Practical -19

Object code:-

Write a program to interface LEDs at any two PWM pins and exhibit LED fading .

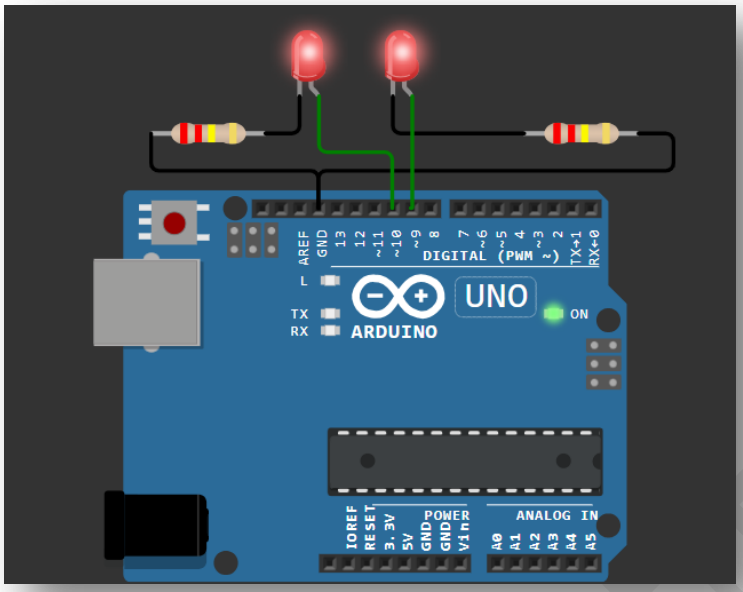
Components Required:

1. Arduino Uno
2. Two LEDs
3. Two current-limiting resistors (typically 220Ω or 330Ω)
4. Breadboard and jumper wires

Wiring:

1. Connect the anode of each LED to a PWM-9, 10 respectively on the Arduino
2. Connect the cathode of each LED to the ground (GND) through a current-limiting resistor.

Zed-King Institute



Arduino Code:-

```
int led = 9;

int led2=10;      // the PWM pin the LED is attached to

int brightness = 0;  // how bright the LED is

int fadeAmount = 5;  // how many points to fade the LED by

// the setup routine runs once when you press reset:

void setup() {

  // declare pin 9 & 10 to be an output:

  pinMode(led, OUTPUT);

  pinMode(led2, OUTPUT);

}

// the loop routine runs over and over again forever:
```

Zed-King Institute

```
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);
  analogWrite(led2, brightness);
  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;
  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

Practical-20

Object code:-

Write a program to interface Light Dependent Resistor (LDR) and display the values read on the Serial monitor after delay of 2 seconds each.

Components Needed:

1. **Arduino board** (like Uno, Nano, etc.)
2. **LDR (Light Dependent Resistor)**
3. **Resistor** (10kΩ is commonly used for the voltage divider with

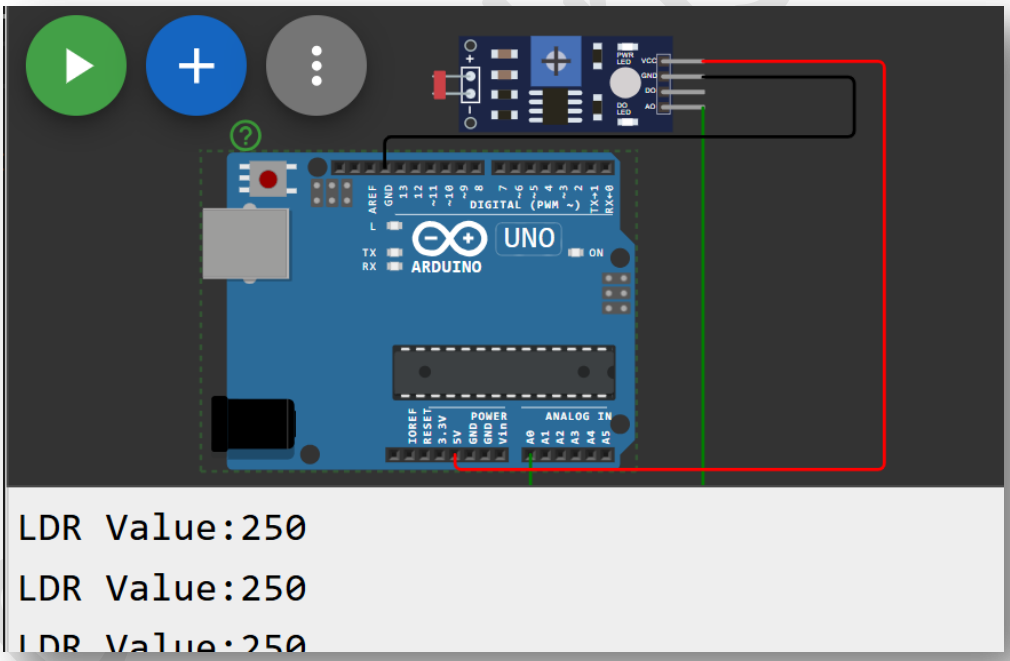
Zed-King Institute

the LDR)

4. **Breadboard and jumper wires**

Circuit Diagram:

1. **LDR** and **10kΩ resistor** in a voltage divider configuration.
2. Connect one end of the LDR to 5V (or 3.3V depending on your Arduino).
3. Connect the other end of the LDR to one end of the 10kΩ resistor.
4. Connect the other end of the 10kΩ resistor to GND.
5. Connect the junction between the LDR and the resistor to the analog input pin A0 on the Arduino



Arduino code:-

```
// Define the analog pin where the LDR is
int ldrPin = A0; // Analog pin A0
```

Zed-King Institute

```
void setup() {
  // Start serial communication at 9600 baud rate
  Serial.begin(9600);
}

void loop() {
  // Read the value from the LDR
  int ldrValue =analogRead(ldrPin);

  // Print the value to the Serial Monitor
  Serial.print("LDR Value: ");
  Serial.println(ldrValue);

  // Wait for 2 seconds
  delay(2000);
}
```



Zed-King Institute

Practical-21

Object code:-

Write a program to sense obstacle closer than 1 meter near an IR sensor interfaced With Arduino Uno.

Components Needed:

1. **Arduino Uno board**
2. **IR Distance Sensor** (e.g., Sharp GP2Y0A21YK0F)
3. **Breadboard and jumper wires**

Circuit Diagram:

1. **IR Distance Sensor**:
   - Connect the **VCC** pin of the IR sensor to the **5V** pin of the Arduino.
   - Connect the **GND** pin of the IR sensor to the **GND** pin of the Arduino.
   - Connect the **OUT** pin of the IR sensor to an analog input pin on the Arduino (e.g., **A0**).

Arduino Sketch:

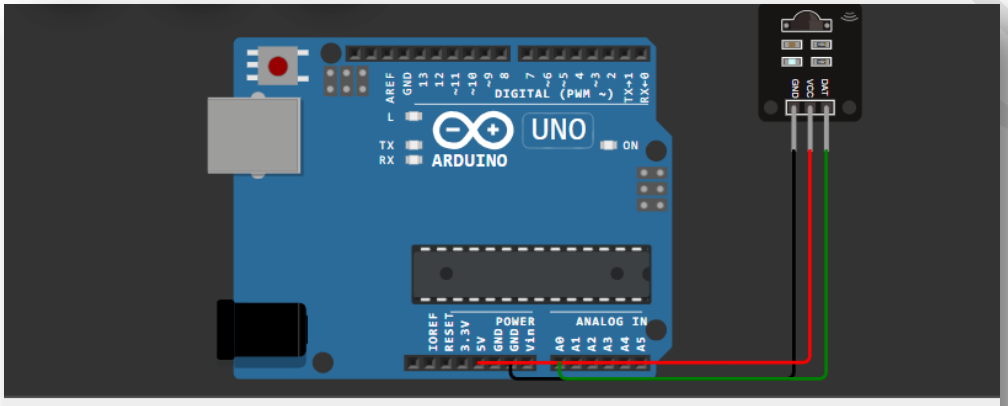
```
// Define the analog pin where the IR sensor is connected

const int sensorPin = A0;

// Define the threshold distance (in centimeters) for obstacle detection

const int thresholdDistance = 100; // 1 meter = 100 centimeters

void setup() {
```

Zed-King Institute

```
  // Start serial communication at 9600 baud rate
  Serial.begin(9600);
}
void loop() {
  // Read the value from the IR sensor
  int sensorValue = analogRead(sensorPin);
  // Convert the sensor value to distance
  // Note: The conversion factor needs to be adjusted based on your specific
// sensor
  // For the Sharp GP2Y0A21YK0F, a typical conversion might look like this:
  // Distance (cm) = 27.86 / (sensorValue / 4.8 - 11.8)
  // This is just an example and may need calibration for your sensor
  float distance = 27.86 / (sensorValue / 4.8 - 11.8);
  // Print the distance to the Serial
  Monitor Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println(" cm");
  // Check if the distance is less than the threshold
  if (distance < thresholdDistance) {
    Serial.println("Obstacle detected within 1 meter!");
```

Zed-King Institute

```
}
// Wait for 500 milliseconds before taking another
reading delay(500);
}
```




```
Distance: 0.14 cm
Obstacle detected within 1 meter!
```

Zed-King

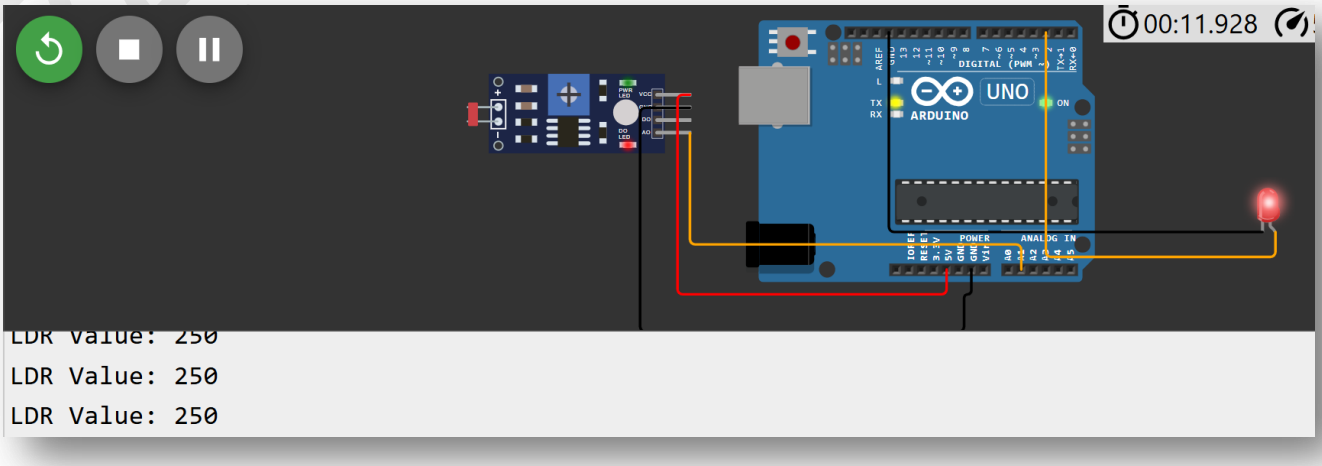
Practical:-22

Object code:-

Write a program to Design interface LDR at pin A1 and led to pin A2 of Arduino uno and turning the LED on as and when the intensity of natural Light goes down.

Hardware Setup:

1. **LDR Connections:**
   - **LDR**: Connect one end of the LDR to the analog input pin A1.
   - **Other End of LDR**: Connect it to one end of a resistor (10kΩ is a common choice).
   - **Resistor**: Connect the other end of the resistor to GND.
   - **LDR-VCC Junction**: Connect the junction between the LDR and resistor to the A1 pin on the Arduino.
   - **Resistor to GND**: Connect the resistor's other end to the Arduino GND.
2. **LED Connections:**
   - **LED Anode**: Connect the long leg (anode) of the LED to digital pin A2 through a current-limiting resistor (220Ω or 330Ω).
   - **LED Cathode**: Connect the short leg (cathode) of the



Zed King Institute

LED to GND.

Arduino code:-

```
// Define the pin numbers
const int ldrPin = A1; // Analog pin connected to LDR

const int ledPin = 2; // Digital pin 2 connected to LED
// Define the light intensity threshold
const int threshold = 500; // Adjust this value based on your environment
void setup() {
  // Initialize the LED pin as an output
  pinMode(ledPin, OUTPUT);
  // Initialize serial communication for debugging
  Serial.begin(9600);
}
void loop() {
  // Read the value from the LDR

  int ldrValue = analogRead(ldrPin);
  // Print the LDR value to the Serial Monitor
  Serial.print("LDR Value: ");
  Serial.println(ldrValue);
  // Compare the LDR value to the threshold
  if (ldrValue < threshold) {
    // Turn on the LED if light intensity is below the threshold
```

Zed-King Institute

```
    digitalWrite(ledPin, HIGH);
  }
  else
  {
    // Turn off the LED if light intensity is above the threshold
    digitalWrite(ledPin, LOW);
  }
  // Wait a short period before reading the value again
  delay(100);
}
```

Relay Module:

A **relay module** is an electronic switch used to control high-voltage devices using low-voltage signals. It is commonly used to interface microcontrollers (like Arduino or Raspberry Pi) with electrical devices such as motors, lights, or other appliances that require a higher voltage or current.

Pinout of a typical 5V Relay Module:

1. **VCC**: Power supply pin (typically 5V or 12V, depending on the relay).
2. **GND**: Ground pin.
3. **IN**: Control input pin. When a HIGH signal is applied to this pin, the relay is activated (closes the switch).
4. **COM (Common)**: Common terminal for the switch.
5. **NO (Normally Open)**: The terminal that is connected to **COM** when the relay is activated.
6. **NC (Normally Closed)**: The terminal that is connected to **COM** when the relay is not activated (the default state).

Zed-King Institute

Practical :-23

Object code:-

Write a program to Control a lamp operating at 220v connected to a relay module.

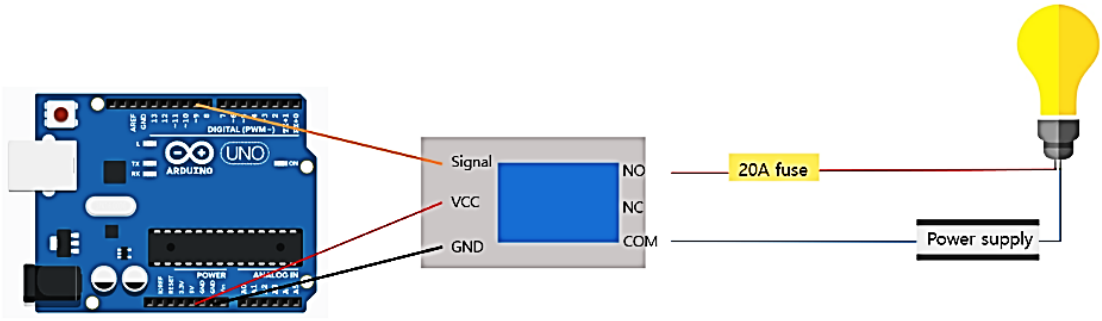
Components Needed

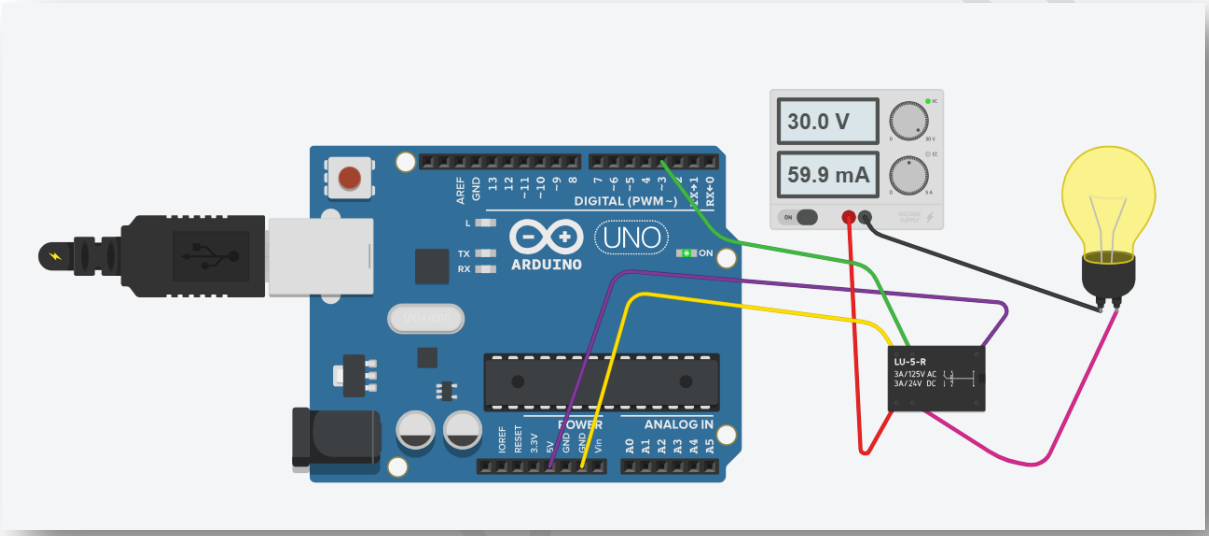
1. **Arduino (e.g., Arduino Uno)**
2. **Relay Module (compatible with Arduino, usually rated for 220V AC)**
3. **Lamp**
4. **Power source (220V AC)**
5. **Connecting wires**

Wiring:-

1. **Relay Module, Power Supply & Light Bulb:**
    - VCC(Terminal 1) to Arduino 5V
    - GND(Terminal 6) to Arduino GND
    - IN1(Terminal 5) to Arduino digital pin (e.g., pin 9)
    - Terminal 7 to Positive pin of power supply
    - Terminal 8 to Terminal 2 of Light Bulb(lamp)
    - Terminal 1 of Light Bulb to negative pin of power supply.
2. **Arduino:**
    - Connect to your computer via USB for programming and power.

Zed-King Institute





Arduino Code:-

```
// Define the pin connected to the relay module
const int relayPin = 9; // Connect this to the IN1 pin on the relay module

void setup() {

pinMode(relayPin, OUTPUT);   // Set relayPin as an OUTPUT

}

void loop() {

  digitalWrite(relayPin, HIGH);  // Turn the lamp on
```

Zed-King

```
  delay(1000); // Lamp stays on for 5 seconds
  digitalWrite(relayPin, LOW);  // Turn the lamp off
  delay(1000); // Lamp stays off for 5 seconds
}
```

Practical:-24

Object Code:

Write a program to print “hello world” on the serial monitor when send message “hello world” from Arduino.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **USB Cable** to connect the Arduino to your computer
3. **Arduino IDE** installed on your computer for programming the Arduino

Arduino code:-

```
// Define the incoming message
const String targetMessage = "hello world";
void setup() {
  // Initialize the serial communication at 9600 baud
  Serial.begin(9600);
}
void loop() {
```

Zed-King Institute

```
  // Check if data is available to read
  if (Serial.available() > 0) {
    // Read the incoming message
    String incomingMessage = Serial.readStringUntil('\n');
    // Remove any leading or trailing whitespace
    incomingMessage.trim();
    // Compare the incoming message with the target message
    if (incomingMessage == targetMessage) {
      // Print "hello world" to the Serial Monitor
      Serial.println("hello world");
    }
  }
}
```

Practical:-25

Object code:-

Write a program demonstrating the use of analog output (pulse with modulation-PWM) to fade LED.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **LED** (any standard 5mm LED)
3. **220-ohm Resistor** (to limit current and protect the LED)
4. **Breadboard** (optional, for easy connections)
5. **Jumper Wires**

Zed-King Institute

Circuit Diagram:

1. **Connect the LED**:
   - **Anode (longer leg)** of the LED to a PWM-capable digital pin on the Arduino (e.g., pin 9).
   - **Cathode (shorter leg)** of the LED to one end of the 220-ohm resistor.
   - **Other end of the resistor** to the Arduino GND (ground) pin.

Arduino Program:

```
int led1=10;      // the PWM pin the LED is attached to

int brightness = 0;  // how bright the LED is

int fadeAmount = 5;  // how many points to fade the LED by

// the setup routine runs once when you press reset:

void setup() {

  // declare pin 10 to be an output:

    pinMode(led1, OUTPUT);

}

// the loop routine runs over and over again forever:

void loop() {

  // set the brightness of pin 10:

  analogWrite(led1, brightness);

  // change the brightness for next time through the loop:

  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the ends of the fade:
```

Zed-King Institute

```
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(30);
}
```

Practical:-26

Object Code:

Write a program to print even number in range of 100 to 200 on serial monitor.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **USB Cable** to connect the Arduino to your computer
3. **Arduino IDE** installed on your computer for programming the Arduino

Arduino CODE:-

```
void setup() {
  // Initialize the serial communication at 9600
  baud Serial.begin(9600);
}

void loop() {
```

Zed-King Institute

```
// Print even numbers from 100 to 200

for (int num = 100; num <= 200; num++) {
if (num % 2 == 0) {
   Serial.println(num);
  }
 }
 // Stop the loop after printing the
 numbers while (true) {
 }
}
```

Practical:-27

Object Code:

Write a program to Design an interface showing the street traffic signal programmed to keep red Light for 30 secs and green light for 60 secs.

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **3 LEDs** (Red, Yellow, Green)
3. **220-ohm Resistors** (for each LED, to limit current and protect the LEDs)
4. **Breadboard** (optional, for easy connections)

Zed-King Institute

5. **Jumper Wires**

Circuit Diagram:

1. **Connect the LEDs**:
    - **Red LED**:
        - **Anode (longer leg)** of the Red LED to a digital pin on the Arduino (e.g., pin 9).
        - **Cathode (shorter leg)** to one end of a 220-ohm resistor.
        - **Other end of the resistor** to the Arduino GND (ground) pin.
    - **Yellow LED**:
        - **Anode (longer leg)** of the Yellow LED to a digital pin on the Arduino (e.g., pin 10).
        - **Cathode (shorter leg)** to one end of a 220-ohm resistor.
        - **Other end of the resistor** to the Arduino GND (ground) pin.
    - **Green LED**:
        - **Anode (longer leg)** of the Green LED to a digital pin on the Arduino (e.g., pin 11).
        - **Cathode (shorter leg)** to one end of a 220-ohm resistor.
        - **Other end of the resistor** to the Arduino GND (ground) pin.

Arduino code:-

```
// Define the pins for the LEDs

const int redPin = 9;

const int yellowPin = 10;

const int greenPin = 11;

void setup() {
```

Zed-King Institute

```
  // Set the LED pins as outputs
  pinMode(redPin, OUTPUT);
  pinMode(yellowPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
}
void loop() {
// Turn on the red light for 30 seconds
digitalWrite(redPin, HIGH);
digitalWrite(yellowPin, LOW);
digitalWrite(greenPin, LOW);
  delay(30000); // delay for 30 seconds
  // Turn on the yellow light for 5 seconds (transition between red and green)
  digitalWrite(redPin, LOW);
  digitalWrite(yellowPin, HIGH);
  digitalWrite(greenPin, LOW);
  delay(5000); //delay for  5 seconds
  // Turn on the green light for 60 seconds
  digitalWrite(redPin, LOW);
  digitalWrite(yellowPin, LOW);
  digitalWrite(greenPin, HIGH);
  delay(60000); // 60 seconds
  // Turn on the yellow light for 5 seconds (transition between green and red)
  digitalWrite(redPin, LOW);
  digitalWrite(yellowPin, HIGH);
```

Zed King Institute

```
  digitalWrite(greenPin, LOW);
  delay(5000); // 5 seconds
}
```

Practical:-28

Object code:-

Write a program in Arduino to interface a temperature of a Room .

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **Temperature Sensor**:
   - **LM35** (analog temperature sensor)
   - **DHT11** or **DHT22** (digital temperature and humidity sensor)
3. **Breadboard** (optional, for easy connections)
4. **Jumper Wires**

Using LM35 (Analog Temperature Sensor)

LM35 provides an analog voltage output that is directly proportional to the temperature. The output voltage increases by 10mV per degree Celsius.

Circuit Diagram for LM35:

- **VCC** of LM35 to **5V** on Arduino.
- **GND** of LM35 to **GND** on Arduino.
- **VOUT** of LM35 to **analog pin A0** on Arduino.

Zed-King Institute

Arduino code:-

```
// Define the pin for the LM35 sensor

const int lm35Pin = A0;

void setup() {
  // Initialize serial communication at 9600 baud
  Serial.begin(9600);
}

void loop() {
  // Read the analog value from the LM35
  int sensorValue = analogRead(lm35Pin);
  // Convert the analog value to temperature in Celsius
  float voltage = sensorValue * (5.0 / 1023.0); // Convert to voltage
  float temperatureC = voltage * 100; // Convert voltage to temperature in C
  // Print the temperature to the Serial Monitor
  Serial.print("Temperature: ");
  Serial.print(temperatureC);
  Serial.println(" C");

  // Wait for 2 seconds
  delay(2000);
}
```

Practical:-29

Object code:-

Zed-King Institute

Write a program to interface DHT sensor with LED the program will switch on LED once temperature rises above 25 degrees ?

Components Required:

1. **Arduino Board** (e.g., Arduino Uno, Arduino Nano, etc.)
2. **DHT11 Temperature and Humidity Sensor** (or DHT22)
3. **LED** (any standard 5mm LED)
4. **220-ohm Resistor** (for the LED, to limit current and protect the LED)
5. **Breadboard** (optional, for easy connections)
6. **Jumper Wires**

Circuit Diagram:

1. **Connect the DHT11/DHT22**:
    - **VCC** of DHT11/DHT22 to **5V** on Arduino.
    - **GND** of DHT11/DHT22 to **GND** on Arduino.
    - **DATA** of DHT11/DHT22 to a digital pin on Arduino (e.g., pin 7).
    - **Pull-up resistor** (10k ohm) between **VCC** and **DATA** pin (optional but recommended).
2. **Connect the LED**:
    - **Anode (longer leg)** of the LED to a digital pin on the Arduino (e.g., pin 9).
    - **Cathode (shorter leg)** of the LED to one end of a 220-ohm resistor.
    - **Other end of the resistor** to the Arduino GND (ground) pin.

Arduino code:-

Zed-King Institute

```
#include <DHT.h>

// Define the type of DHT sensor
#define DHTTYPE DHT11 // Change to DHT22 if you are using DHT22

DHT dht(7, DHTTYPE);   // Initialize DHT sensor on digital pin 7

const int ledPin = 9;      // Define the pin for the LED

void setup() {

  // Initialize serial communication at 9600 baud

  Serial.begin(9600);

  // Initialize the DHT sensor

  dht.begin();

  // Set the LED pin as an output

  pinMode(ledPin, OUTPUT);

}

void loop() {

  // Wait a few seconds between measurements

  delay(2000);

  // Read temperature as Celsius

  float temperatureC = dht.readTemperature();

  // Check if any reads failed and exit early


  if (isnan(temperatureC)) {

    Serial.println("Failed to read from DHT sensor!");
```

Zed-King Institute

```
    return;
  }
  // Print the temperature to the Serial Monitor
  Serial.print("Temperature: ");
  Serial.print(temperatureC);
  Serial.println(" C");
  // Turn the LED on if the temperature is above 25 degrees Celsius

  if (temperatureC > 25) {
    digitalWrite(ledPin, HIGH);
  } else {
    digitalWrite(ledPin, LOW);
  }
}
```

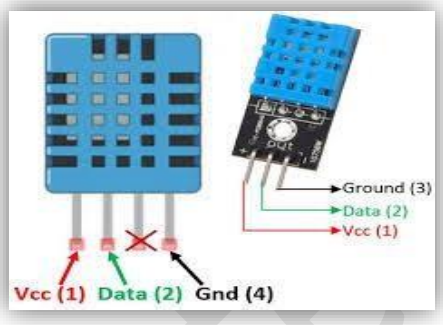
Practical:-30

Object code:-

Write an arduino program to interface a temperature monitoring system in a public environment which will display information on 16*2 LCD. The display should show "Fever" if temp. is greater than 60.6 degrees. and "No Fever" if the temp. is below 60.6 degrees.(use of DHT11 or LM35 temperature sensor is permissible).

Zed-King Institute

Components Needed



- **Arduino Uno**
- **DHT11 Humidity Sensor**
- **Breadboard and jumper wires**
- **10k ohm resistor (usually needed for some sensors, but not always required for the DHT11)**

Wiring the DHT11 Sensor

3. DHT11 Pinout:

- **VCC**: Connects to 5V on the Arduino.
- **GND**: Connects to GND on the Arduino.
- **DATA**: Connects to a digital pin on the Arduino (commonly used pin is D2).
- **NC**: Not connected (usually not used).

4. Connections:

- Connect the **VCC** pin of the DHT11 to the **5V** pin on the Arduino.
- Connect the **GND** pin of the DHT11 to a **GND** pin on the Arduino.
- Connect the **DATA** pin of the DHT11 to **digital pin 2** on the Arduino (you can use another digital pin if you change the pin number in the code).

Arduino Code:

```
#include <DHT.h>
```

Zed-King Institute

```
#include <LiquidCrystal.h>

// Pin configurations

#define DHTPIN 7        // Pin where DHT11 data pin is connected

#define DHTTYPE DHT11     // Define sensor type (DHT11)

#define LM35PIN A0       // If using LM35, connect to analog pin A0

DHT dht(DHTPIN, DHTTYPE);  // Initialize DHT sensor

// LCD pin setup (adjust as per your connection)

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {

  // Start serial communication

  Serial.begin(9600);


  // Initialize DHT sensor

  dht.begin();

    // Set up LCD

  lcd.begin(16, 2);

  lcd.setCursor(0, 0);

  lcd.print("Temp. Monitoring");

  delay(2000); // Wait for 2 seconds

}

void loop() {
```

Zed-King Institute

```
// Reading temperature from DHT11 sensor
float temperature = dht.readTemperature();
// Uncomment the following lines if you use LM35 sensor instead of DHT11
// float voltage = analogRead(LM35PIN) * (5.0 / 1023.0);
// float temperature = voltage * 100;  // LM35 outputs 10mV per degree
// Check if the reading is valid
if (isnan(temperature)) {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Failed to read");
  lcd.setCursor(0, 1);
  lcd.print("sensor data!");
  delay(2000);
  return;
}
  // Print temperature on serial monitor for debugging
Serial.print("Temperature: ");
Serial.print(temperature);
Serial.println(" C");
// Clear the display before showing new info
lcd.clear();
```

Zed-King Institute

```
  // Display temperature status on LCD
  lcd.setCursor(0, 0);
  if (temperature > 60.6) {
    lcd.print("Fever");
  } else {
    lcd.print("No Fever");
  }
  // Wait for 2 seconds before reading the temperature again
  delay(2000);
}
```

Practical:-31

Object code:-

Write a program to display scrolling text “NIELIT HQ” scrolling from left to write and right to left on 16*2 LCD display.

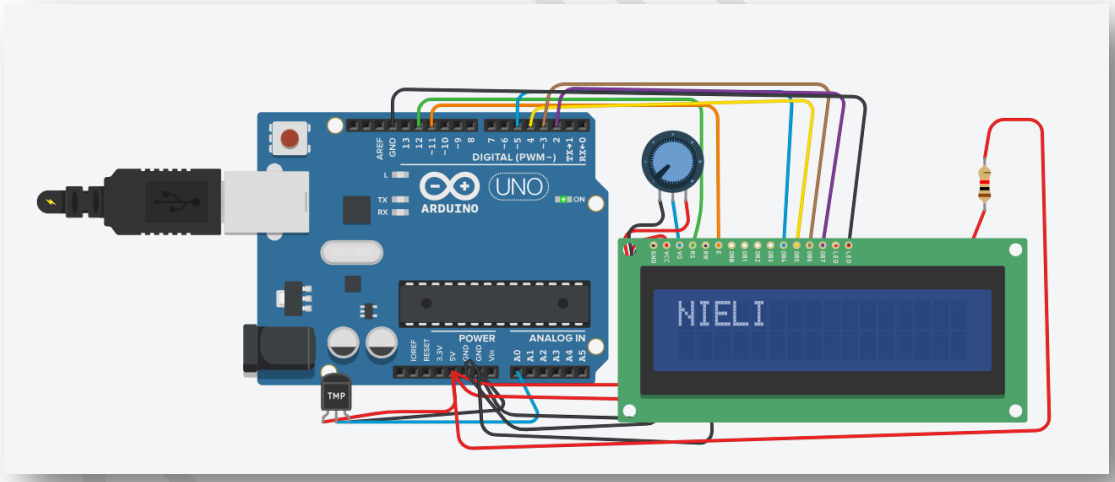
Hardware Required:

1. Arduino board (e.g., Uno, Nano)
2. 16x2 LCD
3. Jumper wires
4. Breadboard

Circuit Connections:

Zed-King Institute

1. **16x2 LCD:**
   - VSS to GND
   - VDD to 5V
   - VO to potentiometer (used to adjust brightness)
   - RS to Pin 12 on Arduino
   - RW to GND
   - EN to Pin 11 on Arduino
   - D4 to Pin 5 on Arduino
   - D5 to Pin 4 on Arduino
   - D6 to Pin 3 on Arduino



   - D7 to Pin 2 on Arduino

Arduino Code:

```
#include <LiquidCrystal.h>

// LCD pin configuration (adjust as per your setup)
```

Zed-King Institute

```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);


String message = "NIELIT HQ";  // Message to scroll
void setup() {
  // Initialize the LCD
  lcd.begin(16, 2);
}
void loop() {
  // Scroll text from left to right
  for (int position = 0; position < message.length(); position++) {
    lcd.clear();
    lcd.setCursor(0, 0);  // Set cursor at the beginning
    lcd.print(message.substring(position));
    delay(300);  // Adjust delay for scrolling speed
  }
  delay(1000);  // Pause before reversing the direction
    // Scroll text from right to left
  for (int position = message.length(); position >= 0; position--) {
    lcd.clear();
    lcd.setCursor(0, 0);  // Set cursor at the beginning
    lcd.print(message.substring(0, position));
    delay(300);  // Adjust delay for scrolling speed
  }
```

Zed-King Institute

```
  delay(1000);  // Pause before starting over
}
```

O-Level (M4-R5.1)
Internet of Things and its Applications by
Ms.Maffi (IOT Instructor)
Zed-King Group of Institutions, Kaithal (Haryana)

Free Online Computer Classes on YouTube Channel
https://www.youtube.com/@zedkinginstitute

For free PDF Notes
Our Telegram Channel: t.me/boost/Zedking_Institute

Zed-King Institute

Our Social Media Platforms

















Zed-King Institute