ACS
COLLEGE OF ENGINEERING

#207, Kambipura, Mysore road, Banglore-74

Department of Computer Science & Engineering

Internet of Things

(For VII Semester BE)

SUBJECT CODE: BCS701

LAB MANUAL

Academic Year 2025-26

Sponsored by

MOOGAMBIGAI CHARITABLE & EDUCATION TRUST

BANGALORE-560074

Experiment 1: Develop a program to blink 5 LEDs back and forth.

Objective

To develop a program in Arduino to blink 5 LEDs back and forth using Tinkercad simulator.

Components/Equipment Required

- Arduino Uno (1)
- Breadboard (1)
- LEDs (5)
- Resistors (220Ω) (5)
- Jumper wires
- Computer with internet and Tinkercad account

Circuit Connections

ARDUINO -- RESISTORS +LEDs + GND

Pin 2 ----/\/\----->|---- GND

Pin 3 ----/\/\----->|---- GND

Pin 4 ----/\/\----->|---- GND

Pin 5 ----/\/\----->|---- GND

Pin 6 ----/\/\----->|---- GND

(>| represents LED symbol, /\/\ is resistor)

Working Principle

The program turns on LEDs one by one in sequence and then in reverse sequence, creating a back and forth blinking effect. It uses a for-loop to iterate through the pins connected to the LEDs.

Procedure

1. Connect 5 LEDs with current-limiting resistors to digital pins 2–6 of Arduino.
2. Open Tinkercad and create a new circuit.
3. Place components and make the necessary wiring connections.
4. Write the Arduino program as below.
5. Start simulation and observe the LEDs blinking back and forth.

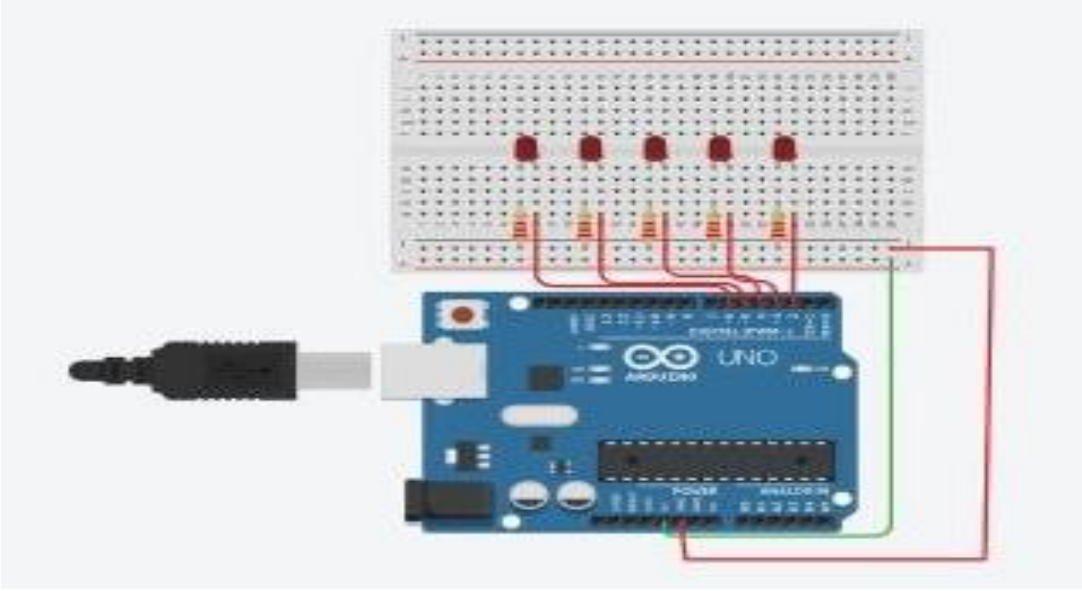
Program Code:-

```
int ledPins[] = {2, 3, 4, 5, 6};
int numLeds = 5;
 void setup()
{   for (int i = 0; i < numLeds; i++) {
    pinMode(ledPins[i], OUTPUT);
  }
}
 void loop() {
  for (int i = 0; i < numLeds; i++) {
digitalWrite(ledPins[i],          HIGH);
delay(200);
    digitalWrite(ledPins[i], LOW);
  }
  for (int i = numLeds - 2; i > 0; i--) {      digitalWrite(ledPins[i],
HIGH);     delay(200);
    digitalWrite(ledPins[i], LOW);
  }
}
```

Output/Result:-

The 5 LEDs blink sequentially from left to right and then right to left, repeatedly.

Circuit Diagram / Block Diagram



Experiment 2: Develop a program to interface a relay with Arduino board

Objective

To develop a program for interfacing a relay module with Arduino board and control it using a digital output pin.

Components/Equipment Required

- Arduino Uno [1]
- Relay Module [1]
- Breadboard [1]
- Connecting Wires [5]
- Power Supply [1]

Connections

Relay Pin	Connects To
DC+	Arduino **5V**
DC-	Arduino **GND**

IN	Arduino **Digital Pin 7**
Relay Pin	**Connects To**
COM	**+5V** from Arduino or Breadboard
NO	**Green LED anode** (with 220Ω resistor to GND)
NC	**Red LED anode** (with 220Ω resistor to GND)

LEDs

- **Green LED** ○ Anode → Relay NO
  ○ Cathode → 220Ω resistor → GND
- **Red LED** ○ Anode → Relay NC ○ Cathode → 220Ω resistor → GND

Working Principle

The relay module is connected to a digital pin of the Arduino. When the digital pin is set HIGH or LOW, it energizes or de-energizes the relay coil, thereby switching the connected device ON or OFF.

Procedure

1. Connect the relay module IN pin to Arduino digital pin 7.
2. Connect the VCC and GND of the relay module to Arduino 5V and GND respectively.
3. Ensure proper relay coil power supply.
4. Write and upload the code to Arduino via Arduino IDE.
5. Observe the relay switching action in Tinkercad simulation.

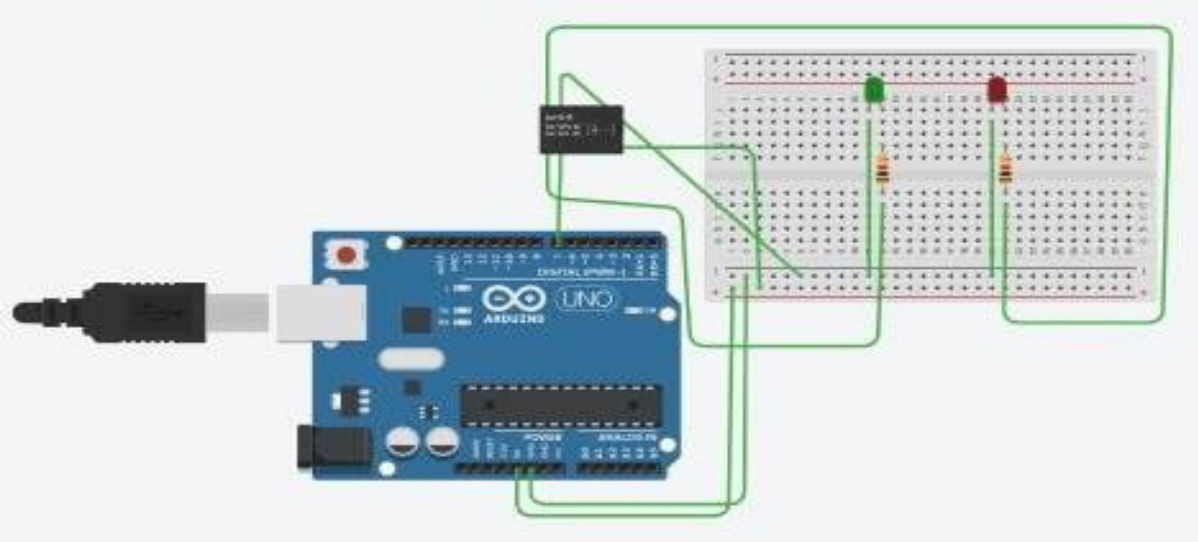
Program Code

```
int relayPin = 7; // Relay control pin
 void setup() {
  pinMode(relayPin, OUTPUT); // Set pin as output
}  void loop()
{
  digitalWrite(relayPin, HIGH); // Turn relay ON   delay(1000);
// Wait 1 second
  digitalWrite(relayPin, LOW); // Turn relay OFF   delay(1000);
// Wait 1 second
}
```

Output/Result

The relay toggles ON and OFF every second, which can be observed as clicking sound or LED indicator blinking on relay module.

Circuit Diagram / Block Diagram



Experiment 3: Develop a program to deploy an intrusion detection system using Ultrasonic and sound sensors.

Objective

To develop a system that detects intrusions using ultrasonic and sound sensors and alerts accordingly.

Components/Equipment Required

Arduino Uno [1]

Ultrasonic Sensor [1]

Sound Sensor [1] (slide switch)

Jumper Wires [As required]

Breadboard [1]

Circuit Connections

Arduino UNO

[HC-SR04]

VCC → 5V

GND → GND

Trig → Pin 7

Echo → Pin 6

[Sound Sensor]

VCC → 5V

GND → GND

OUT → Pin 5

[Buzzer]

\+ → Pin 8

\- → GND

Working Principle

The ultrasonic sensor measures the distance to detect obstacles, while the sound sensor detects sound intensity. When a moving object or abnormal sound is detected, the system triggers an alert using the Arduino.

Procedure

1. Connect ultrasonic sensor to Arduino pins.
2. Connect sound sensor to analog pin.
3. Power the circuit with Arduino.
4. Upload the detection code.
5. Observe serial monitor for intrusions.

Program Code

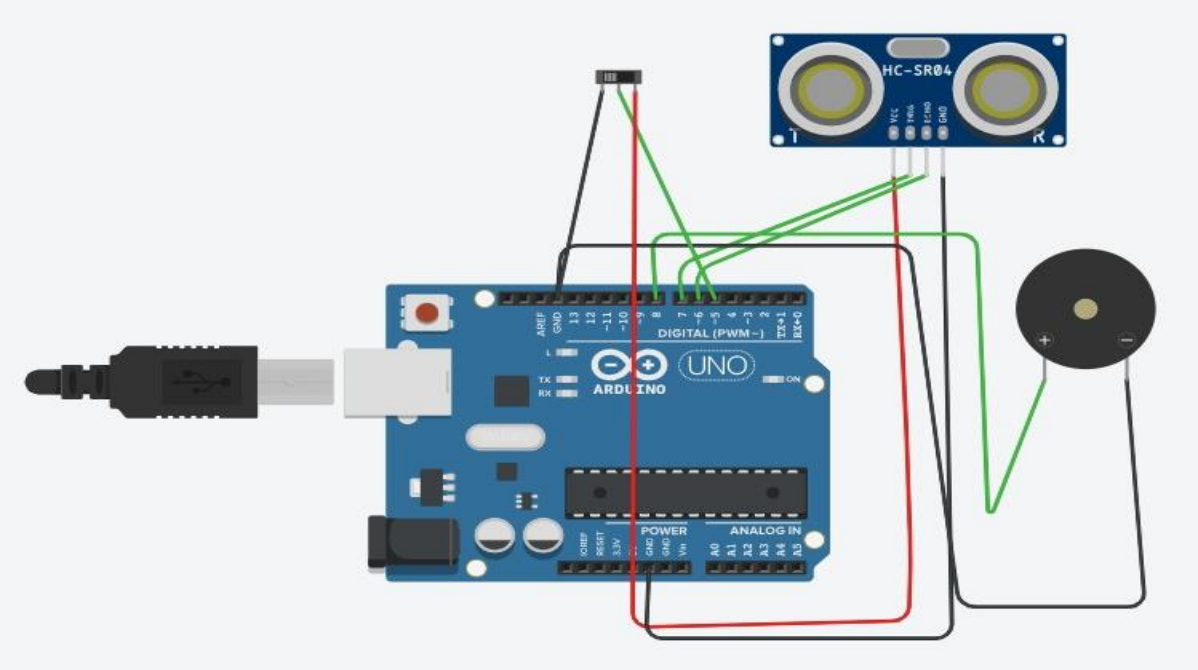
```
#define trigPin 7
#define echoPin 6
#define soundPin 5
#define buzzerPin 8
long duration; int distance; int soundState;
void setup()
      {
      pinMode(trigPin,   OUTPUT);          pinMode(echoPin,   INPUT);
      pinMode(soundPin, INPUT);   pinMode(buzzerPin, OUTPUT);
      Serial.begin(9600);
      }
void loop()
      {
      // Measure distance   digitalWrite(trigPin, LOW);
      delayMicroseconds(2);   digitalWrite(trigPin, HIGH);
      delayMicroseconds(10);   digitalWrite(trigPin, LOW);
      duration = pulseIn(echoPin, HIGH);
      distance = duration * 0.034 / 2;
      soundState = digitalRead(soundPin);
      Serial.print("Distance: ");
      Serial.print(distance);
      Serial.print(" cm | Sound: ");
      Serial.println(soundState);
      if (distance < 50 || soundState == HIGH)
      {
      digitalWrite(buzzerPin, HIGH);  // Alarm ON
      }
```

```
      else
      {
      digitalWrite(buzzerPin, LOW);   // Alarm OFF
      }
   delay(200);
}
```

Output/Result

When an object is detected within 20 cm or sound level exceeds threshold, a warning is printed on the serial monitor.

Circuit Diagram / Block Diagram



Experiment 4: Control a DC motor with Arduino board.

Objective

To control the speed and direction of a DC motor using Arduino and a motor driver.

Components/Equipment Required

Arduino Uno [1]

DC Motor [1]

L293D Motor Driver [1]

Power Supply [1]

Jumper Wires [As required]

Connections

L293D Motor Driver IC

L293D Pin	Connection
Pin 1 (EN1)	Arduino Digital Pin 9
Pin 2 (IN1)	Arduino Digital Pin 2
Pin 3 (OUT1)	DC Motor Terminal 1
Pin 4 (GND)	GND (Breadboard)
L293D Pin	Connection
Pin 5 (GND)	GND (Breadboard)
Pin 6 (OUT2)	DC Motor Terminal 2
Pin 7 (IN2)	Arduino Digital Pin 3
Pin 8 (Vcc2)	Motor Power Supply (Optional: 9V battery)
Pin 16 (Vcc1)	Arduino 5V
Pin 15 (IN4)	Not Used
Pin 10 (IN3)	Not Used
Pin 9 (EN2)	Not Used

DC Motor

- Connected between L293D OUT1 (Pin 3) and OUT2 (Pin 6).

Potentiometer

- One end → 5V
- Other end → GND
- Center pin → Arduino Analog Pin A0

Working Principle

The motor driver is controlled by digital signals from the Arduino. PWM is used for speed control, and digital pins for direction.

Procedure

1. Connect DC motor to L293D outputs.
2. Connect control pins of L293D to Arduino digital pins.
3. Provide external power to motor driver.
4. Upload code to control motor speed and direction.

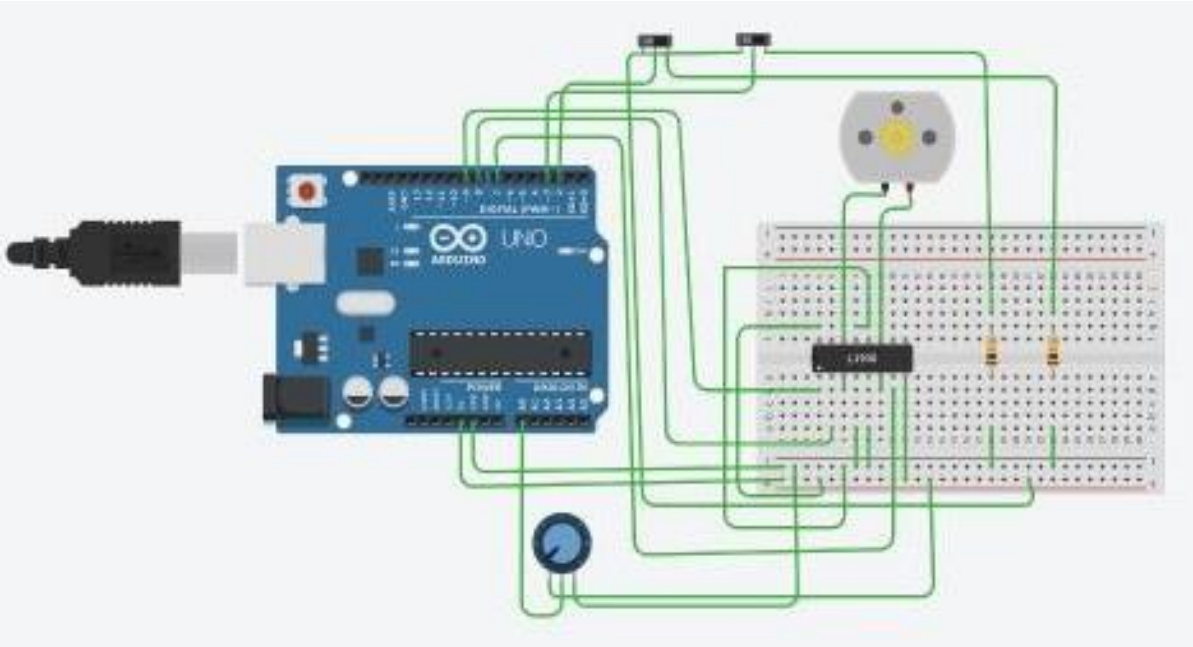
Program Code

```
 int motorPin1 = 3; int motorPin2 = 4; void
setup() {    pinMode(motorPin1,
OUTPUT);     pinMode(motorPin2,
OUTPUT);
} void loop() {
  digitalWrite(motorPin1, HIGH);
digitalWrite(motorPin2, LOW);   delay(2000);
  digitalWrite(motorPin1, LOW);
digitalWrite(motorPin2, HIGH);   delay(2000);
}
```

Output/Result:-

Motor rotates in one direction for 2 seconds and then reverses for 2 seconds repeatedly.

Circuit Diagram / Block Diagram



Experiment 5: Smart Street Light using LDR

Objective:

To develop a program to deploy a smart street light system using LDR sensor.

Components/Equipment Required:

- Arduino Uno [1]
- LDR Sensor [1]
- LED [1]
- Resistor 10kΩ [1]
- Breadboard [1]
- Jumper Wires [As required]

Circuit Connections Arduino

UNO Pins:

Pin	Connected To
5V	Breadboard positive rail (+)
GND	Breadboard negative rail (–)
A0	Middle pin of the **photoresistor (LDR)**
Pin	**Connected To**
D8	Anode of **LED** (via resistor ~220–330Ω)

Photoresistor (LDR):

- One side of LDR → 5V (positive rail).
- Other side of LDR → Analog pin **A0** AND one side of a **fixed resistor (~10kΩ)**.
- Other side of the fixed resistor → GND (negative rail).

LED:

- Long leg (anode) of LED → Through a resistor (~220Ω) → Pin **D8**.
- Short leg (cathode) of LED → GND.

Breadboard power:

- Breadboard + rail connected to Arduino 5V.
- Breadboard – rail connected to Arduino GND.

Working Principle:

The LDR sensor detects light intensity. If ambient light is low, the streetlight (LED) turns on automatically.

Procedure:

1. Connect LDR in a voltage divider circuit to Arduino analog pin.
2. Connect LED to digital pin with resistor.
3. Write logic to switch LED based on LDR value.
4. Upload code and observe LED response to light.

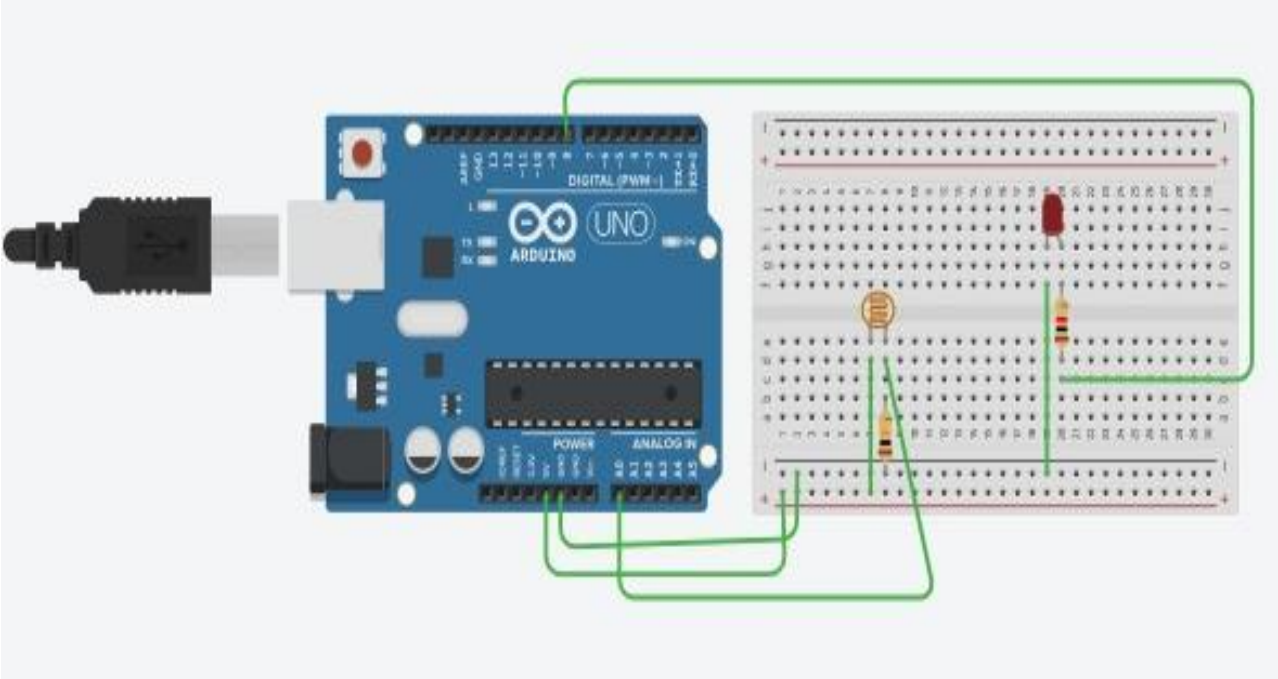
Program Code:

```
int ldrPin = A0;
int ledPin = 8;
void setup()
      {
      pinMode(ledPin, OUTPUT);
      Serial.begin(9600);
      }
void loop()
      {
      int ldrValue = analogRead(ldrPin);
      Serial.println(ldrValue);
      if (ldrValue < 500)
      {
      digitalWrite(ledPin, HIGH);
      }
      else
      {
      digitalWrite(ledPin, LOW);
      }
  delay(1000);
}
```

Output/Result:

LED lights up when ambient light is low.

Circuit Diagram / Block Diagram:



Experiment 6: Waste Classification using Moisture Sensor (DHT22)

Objective:

To develop a program to classify dry and wet waste using a moisture sensor.

Components/Equipment Required:

- Arduino Uno [1]
- DHT22 Sensor [1]
- Breadboard [1]
- Jumper Wires [As required]
- LED (Red and Green) [2]
- Resistors 220Ω [2]

Connections:

LCD (16×2) — connected in 4-bit mode

LCD Pin	Connected to Arduino Pin
VSS	GND
VDD	+5V
VO	Middle pin of potentiometer (for contrast)
RS	**7**
LCD Pin	**Connected to Arduino Pin**
RW	GND
E	**8**
D4	**9**
D5	**10**
D6	**11**
D7	**12**
A (LED+)	+5V via 220Ω resistor
K (LED-)	GND

Potentiometer (10kΩ) for LCD contrast:

- One end: GND
- Other end: +5V
- Middle (wiper): VO of LCD

Soil Moisture Sensor

Assuming you have the typical 3-pin soil sensor module:

Sensor Pin	Connected to Arduino Pin
VCC	+5V
GND	GND
AO (Analog Output)	**A0**

Power

- Arduino powered via USB or external 9V adapter.
- Breadboard power rails connected to Arduino 5V & GND for distribution.

Working Principle:

The DHT22 sensor measures humidity to distinguish between wet and dry waste. If humidity is above a set threshold, it is classified as wet waste.

Procedure:

1. Connect DHT22 data pin to Arduino digital pin.
2. Install and include DHT library.
3. Read humidity and display LED for classification.
4. Upload and test with samples.

Program Code:

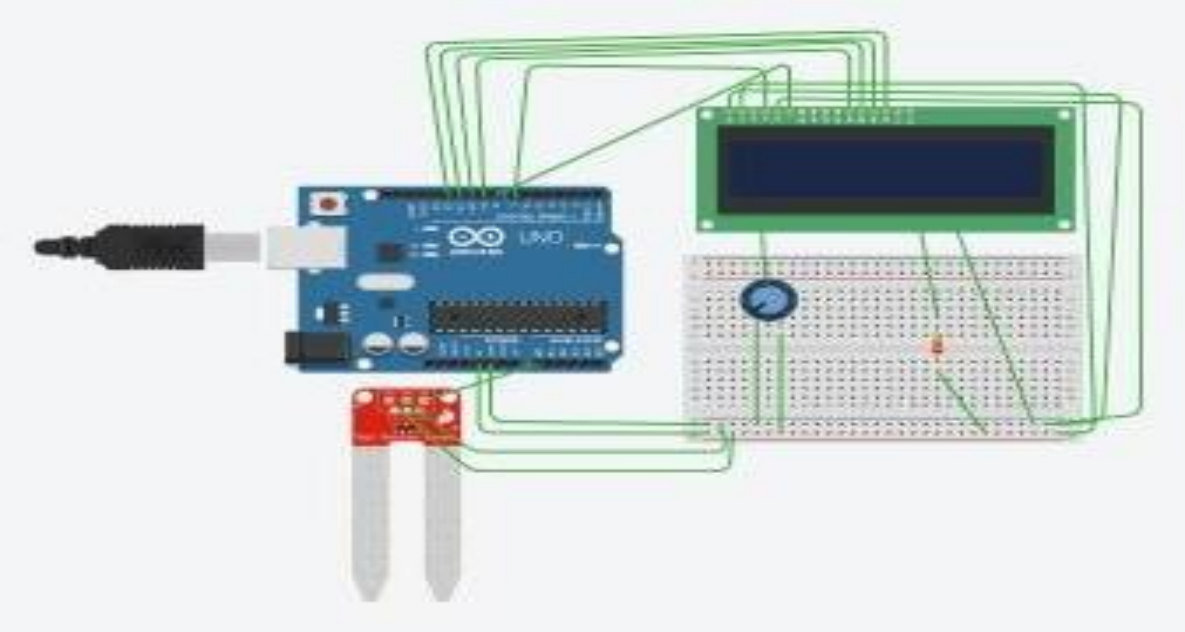
```
// C++ code
#include <LiquidCrystal.h>
// LCD pins: RS, E, D4, D5, D6, D7
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);
#define SOIL_PIN A0  // Soil moisture sensor pin
int soilValue = 0;   // Stores moisture reading
int threshold = 600; // Tune as per experiment
void setup()
      {
      Serial.begin(9600);
      lcd.begin(16, 2);
      lcd.print("Dry/Wet Classifier");
      delay(2000);
      lcd.clear();
      }
void loop()
      {
      soilValue = analogRead(SOIL_PIN);
      Serial.print("Soil Moisture Value: ");
      Serial.println(soilValue);
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Moisture: ");
      lcd.print(soilValue);
      lcd.setCursor(0, 1);
      if (soilValue > threshold)
      {
            lcd.print("Waste: DRY");
            Serial.println("Waste is: DRY");
      }
      else
      {
      lcd.print("Waste: WET");
```

```
      Serial.println("Waste is: WET");
      }
 Serial.println("--------------------");
 delay(2000);
}
```

Output/Result:

Red LED lights for wet waste and green LED for dry waste based on humidity level.

Circuit Diagram / Block Diagram:



Experiment 7: Read pH Values of Various Substances

Objective:

To develop a program to read the pH value of various substances like milk, lime, and water.

Components/Equipment Required:

- Arduino Uno [1]
- pH Sensor Module [1]
- Breadboard [1]
- Jumper Wires [As required]

Flowchart:

Connections:

Pin on Arduino UNO	Connected to
5V	Potentiometer VCC pin
GND	Potentiometer GND pin
A0	Potentiometer middle pin (wiper)

Working Principle:

The pH sensor measures the hydrogen ion concentration in the solution and outputs a corresponding voltage, which is read and calibrated to a pH value.

Procedure:

1. Connect the pH sensor module to analog pin A0.
2. Calibrate the sensor using buffer solutions.
3. Read analog voltage and convert to pH using formula.
4. Upload and test with different substances.

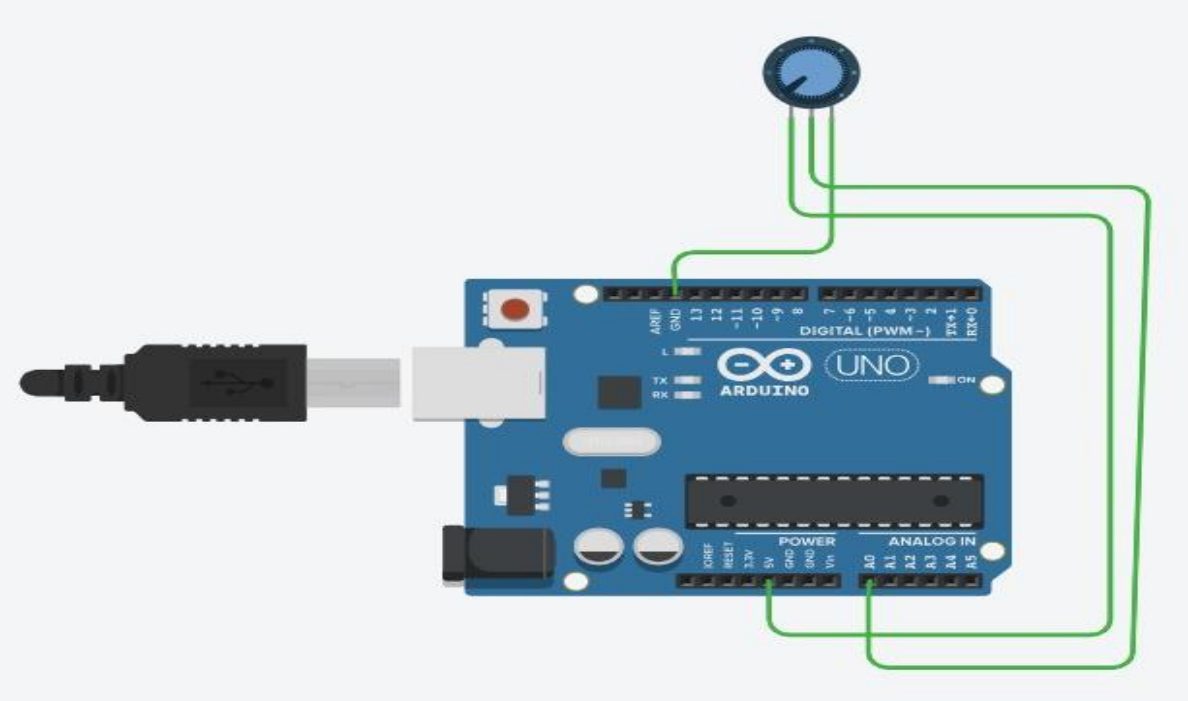
Program Code:

```
int potPin = A0;
float pH_value;
void setup()
{
  Serial.begin(9600);
  Serial.println("Turn potentiometer to simulate pH of milk, lime,
or water.");
}
void loop()
{
  int analogValue = analogRead(potPin); // 0-1023
  pH_value=map(analogValue, 0, 1023, 0, 140) / 10.0; // Map to 0-14
  Serial.print("Simulated pH value: ");
  Serial.println(pH_value, 1); // 1 decimal place
  delay(1000);
}
```

Output/Result:

Displays the voltage and pH value of the tested substance.

Circuit Diagram / Block Diagram:



Experiment 8: Gas Leakage Detection System

Objective:

To develop a program to detect gas leakage in the surrounding environment.

Components/Equipment Required:

- Arduino Uno [1]
- MQ-2 Gas Sensor [1]
- Buzzer [1]
- LED [1]
- Resistor 220Ω [1]
- Breadboard [1]
- Jumper Wires [As required]

Connections:

MQ-2 Gas Sensor

MQ-2 Pin	Connect To
VCC	5V on Arduino
GND	GND on Arduino
A0	A0 on Arduino

Buzzer (Optional Alert)

Buzzer Pin	Connect To
+	Digital Pin **8**
-	GND

LED (Optional Indicator)

| **LED Anode (+)** | Digital Pin **9** via 220Ω resistor |

| **LED Cathode (-)** | GND |

Working Principle:

The MQ-2 sensor detects gases like LPG, methane, and smoke. When concentration crosses a threshold, it activates an alert system.

Procedure:

1. Connect sensor analog pin to Arduino A0.
2. Connect LED and buzzer to digital pins.
3. Set threshold for gas concentration.
4. Test using a lighter gas (brief exposure).

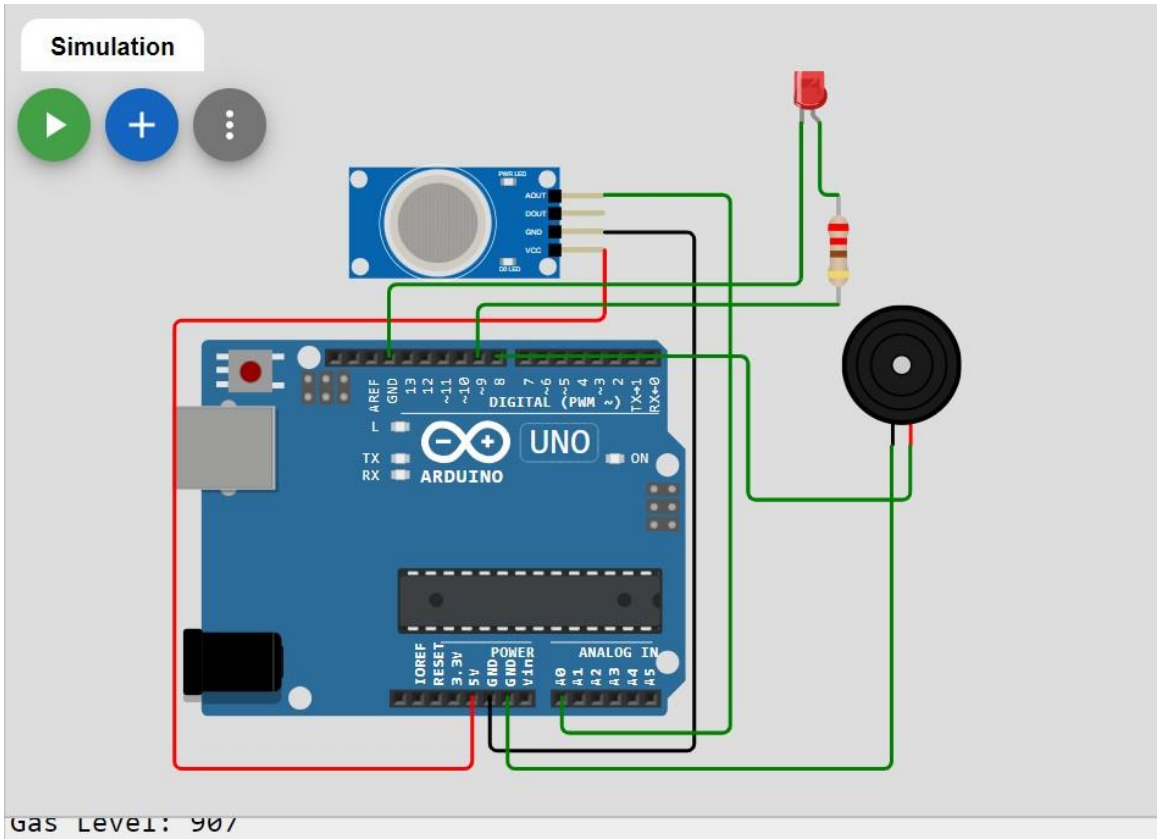
Program Code:

```
const int gasPin = A0;      // Analog pin connected to MQ-2 A0
const int buzzerPin = 8;    // Buzzer connected to D8
const int ledPin = 9;       // Optional LED on D9
void setup()
  {  Serial.begin(9600);
      pinMode(buzzerPin, OUTPUT);
      pinMode(ledPin, OUTPUT);
  }
void loop()
  { int gasValue = analogRead(gasPin);
  Serial.print("Gas Level: ");
  Serial.println(gasValue);
  if (gasValue > 400)
  {   // Threshold for gas leakage
   digitalWrite(buzzerPin, HIGH);
   digitalWrite(ledPin, HIGH);
   }
   else
   { digitalWrite(buzzerPin, LOW);
    digitalWrite(ledPin, LOW);
  }
  delay(1000);
}
```

Output/Result:

Buzzer and LED turn ON when gas concentration exceeds threshold.

Circuit Diagram / Block Diagram:



Experiment 9: Weather Station Readings using Arduino

Objective:

To develop a program to demonstrate weather station readings using Arduino.

Components/Equipment Required:

- Arduino Uno [1]
- DHT11/DHT22 Sensor [1]
- LCD Display (16x2) [1]
- Potentiometer [1]
- Breadboard [1]
- Jumper Wires [As required]

LCD (16x2) to Arduino UNO

LCD Pin	Function	Arduino Pin
1 (VSS)	Ground	GND
2 (VDD)	+5V Power	5V
3 (V0)	Contrast	Middle pin of potentiometer
4 (RS)	Register Select	Digital **12**
5 (RW)	Read/Write	GND
6 (E)	Enable	Digital **11**
11 (D4)	Data bit 4	Digital **5**
12 (D5)	Data bit 5	Digital **4**
13 (D6)	Data bit 6	Digital **3**
14 (D7)	Data bit 7	Digital **2**
15 (A/LED+)	Backlight +	5V (use 220Ω resistor)
16 (K/LED-)	Backlight -	GND

Potentiometer Connections (for contrast)

Pot Pin	Connect To
1 (left)	GND
2 (middle)	LCD pin 3 (V0)
3 (right)	5V

DHT22 Sensor to Arduino UNO

DHT22 Pin	Function	Connect To
1 (VCC)	Power	**5V**
2 (DATA)	Data	**Digital Pin 7**
3 (NC)	Not Used	Leave unconnected
4 (GND)	Ground	**GND**
Extra	Pull-up	10kΩ resistor between **pin 1** and **pin 2** of DHT22

Working Principle:

The DHT sensor measures temperature and humidity, and the values are displayed on the LCD screen. It simulates a basic weather station.

Procedure:

2. Connect DHT sensor to digital pin of Arduino.
3. Connect LCD with potentiometer to control contrast.
4. Install DHT and LiquidCrystal libraries.
5. Display real-time temperature and humidity.

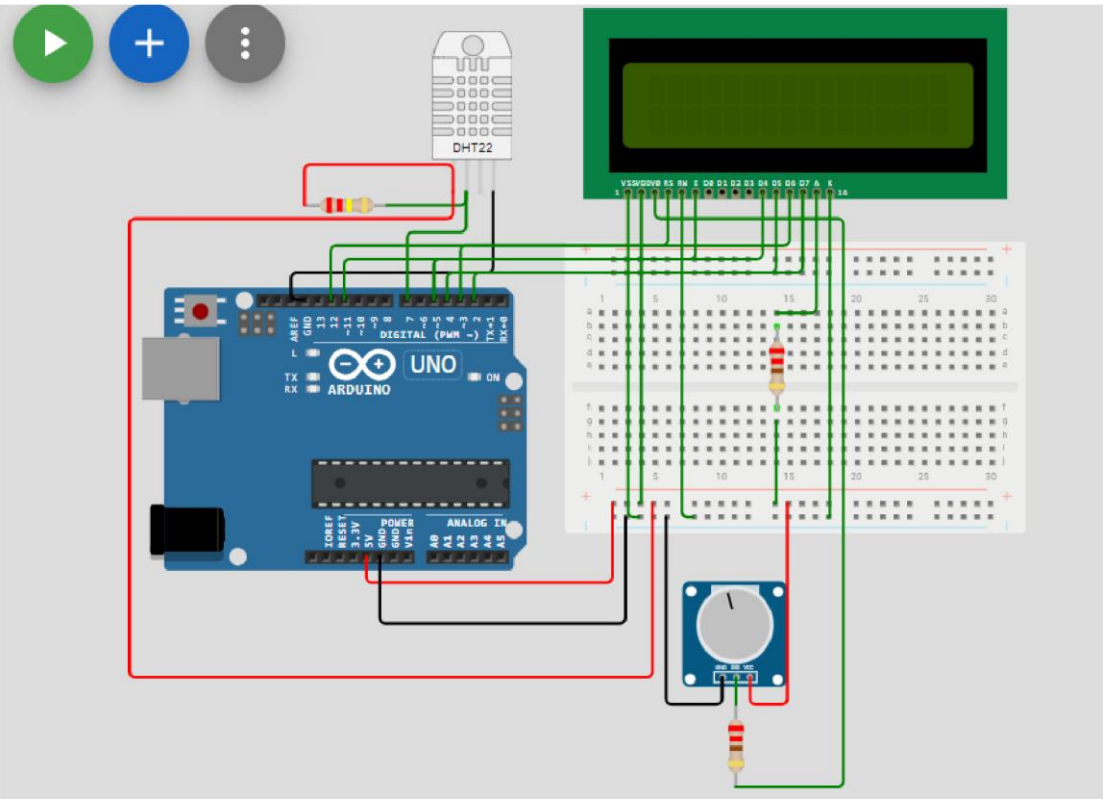
Program Code:

```
#include <LiquidCrystal.h>
#include <DHT.h>
 // Define pins
#define DHTPIN 7                    // Data pin of DHT22
#define DHTTYPE DHT22               // DHT22 sensor
DHT dht(DHTPIN, DHTTYPE);       // LCD pin connections: RS, E, D4, D5, D6, D7
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
 void setup()
{ lcd.begin(16, 2);       // Initialize 16x2 LCD
  dht.begin();           // Initialize DHT sensor
  lcd.print("Weather Station");
  delay(2000);           // Display title for 2 seconds   lcd.clear();
}
 void loop()
   {      float temp = dht.readTemperature();  // Read temperature in °C
           float humid = dht.readHumidity();    // Read humidity %
           // Check if reading failed
           if (isnan(temp) || isnan(humid))
           {    lcd.setCursor(0, 0);     lcd.print("Sensor error");
                return; }
   // Display temperature
           lcd.setCursor(0, 0);
           lcd.print("Temp: ");
           lcd.print(temp, 1);
// Show 1 decimal
           lcd.print(" C");
   // Display humidity
           lcd.setCursor(0, 1);
           lcd.print("Humidity: ");
           lcd.print(humid, 1);
           lcd.print(" %");
           delay(2000);  // Refresh every 2 seconds
}
```

Output/Result:

Displays temperature and humidity on the LCD screen.

Circuit Diagram / Block Diagram:



Experiment 10: UART Protocol String Transmission

Objective:

To develop a program to setup a UART protocol and pass a string through the protocol.

Components/Equipment Required:

- Arduino Uno [2]
- Jumper Wires [As required]

Connections

Wiring (Basic UART)

Master pin	Slave pin
TX (pin 1) →	RX (pin 0)
RX (pin 0) ←	TX (pin 1)
GND ↔	GND

Working Principle:

UART (Universal Asynchronous Receiver/Transmitter) protocol enables serial communication between two Arduino boards using TX and RX pins.

Procedure:

1. Connect TX of Arduino1 to RX of Arduino2 and vice versa.
2. Connect ground pins.
3. Upload sender code on one Arduino.
4. Upload receiver code on other Arduino.
5. Observe serial monitor for received data.

Program Code (Sender, Master):

```
void setup()
{
  Serial.begin(9600);  // Master TX/RX + Serial Monitor
}
void loop()
{
  Serial.println("Hello Slave!");   delay(500);
  if (Serial.available())
{
    String reply = Serial.readStringUntil('\n');
    Serial.print("Slave replied: ");
    Serial.println(reply);
  }    delay(1000);
}
```

Program Code (Receiver,Slave):

```
void setup()
{
  Serial.begin(9600);  // Slave TX/RX
}
void loop()
{
   if(Serial.available())
   {  String msg = Serial.readStringUntil('\n');
      Serial.print("Got: ");
      Serial.println(msg);
       Serial.println("ACK from Slave");
  }
}
```

Output/Result:

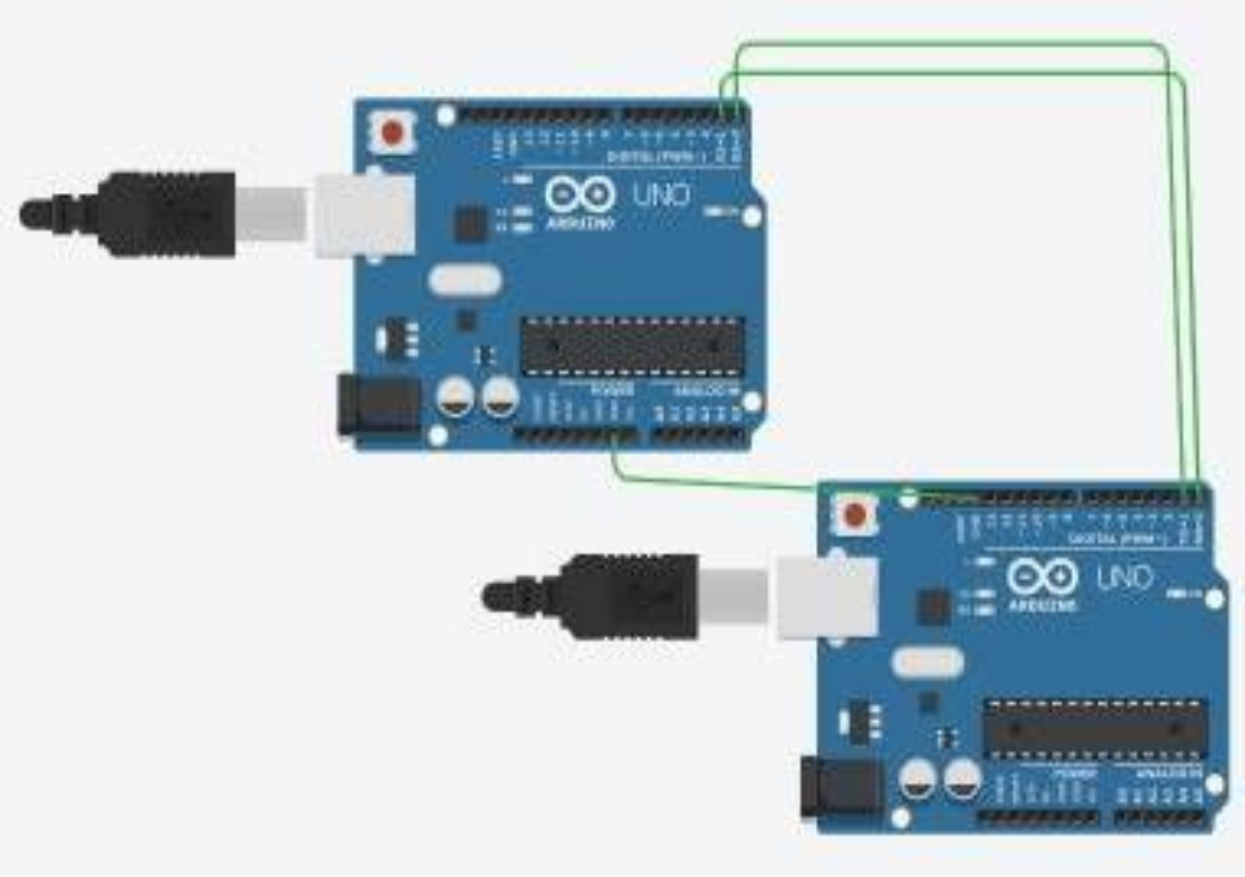
Second Arduino displays string sent from the first.

Hello Slave!

Slave replied: Got: Hello Slave! Hello Slave!

Slave replied: ACK from Slave

Circuit Diagram / Block Diagram:



Experiment 11: Water Level Detection using Ultrasonic Sensor.

Objective:

To develop a water level depth detection system using an Ultrasonic sensor.

Components/Equipment Required:

- Arduino Uno [1]
- Ultrasonic Sensor HC-SR04 [1]
- LCD [1]
- [As required]
- Resistor 220Ω [1]
- Breadboard [1]
- Jumper Wires

Connections

HC-SR04

Sensor Pin	Connect To
VCC	5V
GND	GND
TRIG	Digital Pin 9
Sensor Pin	**Connect To**
ECHO	Digital Pin 10

16x2 LCD Display

LCD Pin	Connect To
VSS	GND
VDD	5V
VO	Middle pin of 10K pot (for contrast)
RS	Digital Pin 12
RW	GND
E	Digital Pin 11
D4	Digital Pin 5
D5	Digital Pin 4
D6	Digital Pin 3
D7	Digital Pin 2
A	5V (via 220Ω resistor)
K	GND

Working Principle:

The ultrasonic sensor measures distance between the water surface and sensor. The water level is calculated by subtracting this distance from the total height of the tank.

Procedure:

1. Connect the ultrasonic sensor's trig and echo pins to Arduino.
2. Power the sensor.
3. Define height of the container.
4. Calculate distance using time-of-flight formula.
5. Display or trigger alert based on water level.

Program Code:

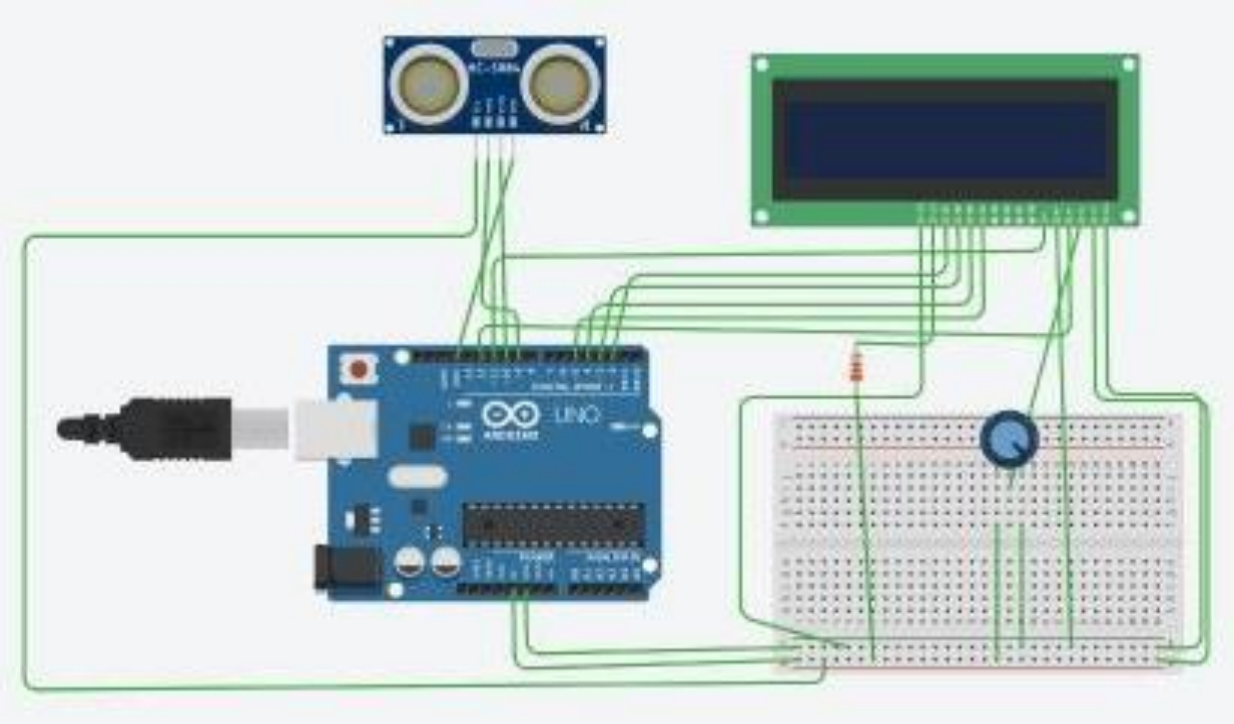
```
// C++ code
#include <LiquidCrystal.h>
// Initialize the LCD: RS, E, D4, D5, D6, D7
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// HC-SR04 sensor pins
const int trigPin = 9;
const int echoPin = 10;

// Tank configuration
const int tankHeight=30;//in cm (adjust based on actual tank height)
long duration;
float distance, waterLevel;
 void setup()
 {
  lcd.begin(16, 2); // 16x2 LCD
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  lcd.setCursor(0, 0);
  lcd.print(" Water Level Sys ");
  delay(2000);
  lcd.clear();
```

```
}
void loop()
{
  // Trigger ultrasonic pulse
      digitalWrite(trigPin, LOW);
      delayMicroseconds(2);
      digitalWrite(trigPin, HIGH);
      delayMicroseconds(10);
      digitalWrite(trigPin, LOW);
  // Read echo duration
duration = pulseIn(echoPin, HIGH);
distance = duration * 0.034 / 2;
  // Calculate water level
if (distance > tankHeight || distance < 0)
   {
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print(" Out of Range ");
      delay(1000);
      return;
  }
  waterLevel = tankHeight - distance;
   // Display on LCD
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("Water Level:");
      lcd.setCursor(0, 1);
      lcd.print(waterLevel, 1);
      lcd.print(" cm");
   delay(1000);
}
```

Output/Result:

Displays water level in cm and triggers LED if below 10 cm. **Circuit Diagram / Block**

Diagram:



Experiment 12: Interfacing Keypad Module with Arduino

Objective:

To develop a program to simulate interfacing with the keypad module and record the keystrokes.

Components/Equipment Required:

- Arduino Uno [1]
- 4x4 Matrix Keypad [1]
- Breadboard [1]
- Jumper Wires [As required]

Connections

Keypad pin 1 → Arduino pin 2

Keypad pin 2 → Arduino pin 3

Keypad pin 3 → Arduino pin 4

Keypad pin 4 → Arduino pin 5

Keypad pin 5 → Arduino pin 6

Keypad pin 6 → Arduino pin 7

Keypad pin 7 → Arduino pin 8

Keypad pin 8 → Arduino pin 9

Working Principle:

The keypad has rows and columns wired to specific digital pins. When a key is pressed, a specific row and column short, which is detected by scanning logic.

Procedure:

1. Connect keypad to digital pins (8 wires total).
2. Install Keypad library.
3. Write logic to scan keypress.
4. Display pressed key on Serial Monitor.

Program Code:

```
#include <Keypad.h>

// Define the rows and columns of the keypad
const byte ROWS = 4;
const byte COLS = 4;
 // Define the keymap
char keys[ROWS][COLS] = {  {'1', '2', '3', 'A'},
                           {'4', '5', '6', 'B'},
                           {'7', '8', '9', 'C'},
                           {'*', '0', '#', 'D'}
};
 // Connect keypad ROW0, ROW1, ROW2, ROW3 to Arduino pins byte
rowPins[ROWS] = {2, 3, 4, 5};
 // Connect keypad COL0, COL1, COL2, COL3 to Arduino pins byte
colPins[COLS] = {6, 7, 8, 9};

// Create the keypad object
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS,
COLS );
void setup()
{
  Serial.begin(9600);
  Serial.println("Keypad ready. Press keys...");
}
void loop()
{
  char key = keypad.getKey();
  if (key)
  {  // If a key is pressed
    Serial.print("Key pressed: ");
    Serial.println(key);
  }
}
```

Output/Result:

Displays each key pressed on Serial Monitor.

Circuit Diagram / Block Diagram: