UNIVERSITY OF KARACHI
DEPARTMENT OF COMPUTER SCIENCE (UBIT)
COURSE INSTRUCTOR: DR AAMIR AMIR

PROJECT REPORT

NAME: MUHAMMAD MUZAMMIL ZIA.

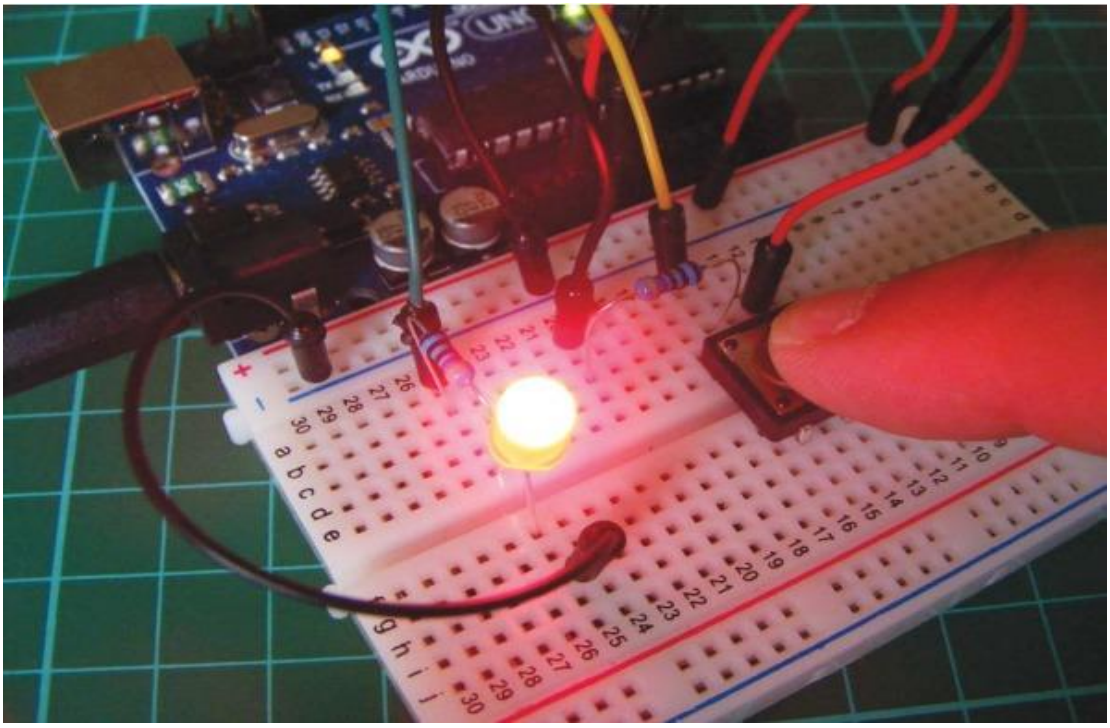
COURSE: SYSTEM DESIGN WITH MICROPROCESSOR.

CLASS: BSCS 6th SEMESTER.

SECTION: B

SEAT #: EB-21102073

Project # 1
Push Button Controlled LED



COMPONENTS:

1. LED
2. BreadBoard
3. Arduino
4. Jumper Wires
5. four-pin push button
6. 10k ohm resistor
7. 220 ohm resistor

Pin Configuration:

Push Button	Arduino
Pin A	GND and pin 2 via 10k-ohm resistor
Pin B	+5v

LED

- positive leg to Arduino pin 13 via a 220-ohm resistor
- negative leg to GND

CODE:

```
const int buttonPin = 2; // Pin connected to pushbutton
const int ledPin = 13; // Pin connected to LED
int buttonState = 0; // Give pushbutton a value

void setup() {
     pinMode(ledPin, OUTPUT); // Set LED pin as output
     pinMode(buttonPin, INPUT); // Set pushbutton pin as input
}

void loop() {
   buttonState = digitalRead(buttonPin); // Read input from pin 2
   if (buttonState == HIGH) {
        digitalWrite(ledPin, HIGH); // Turn on LED
  }
  else {
    digitalWrite(ledPin, LOW); // turn off LED
  }
}
```

LEARNING:

Learned breadboarding fundamentals for component connections. Acquired skills in configuring digital input/output pins using **pinMode()**. Explored conditional logic with **digitalRead()** and **digitalWrite()**, enabling control of real-world components based on push button states. Developed practical understanding of safety measures, including resistor use for LED protection.

PROJECT # 02
BLINKING LED

COMPONENTS:

1. Red LED
2. Arduino

Pin Configuration:

- DIGITAL PIN NO 13 (+ve)
- GND (-ve)

CODE:

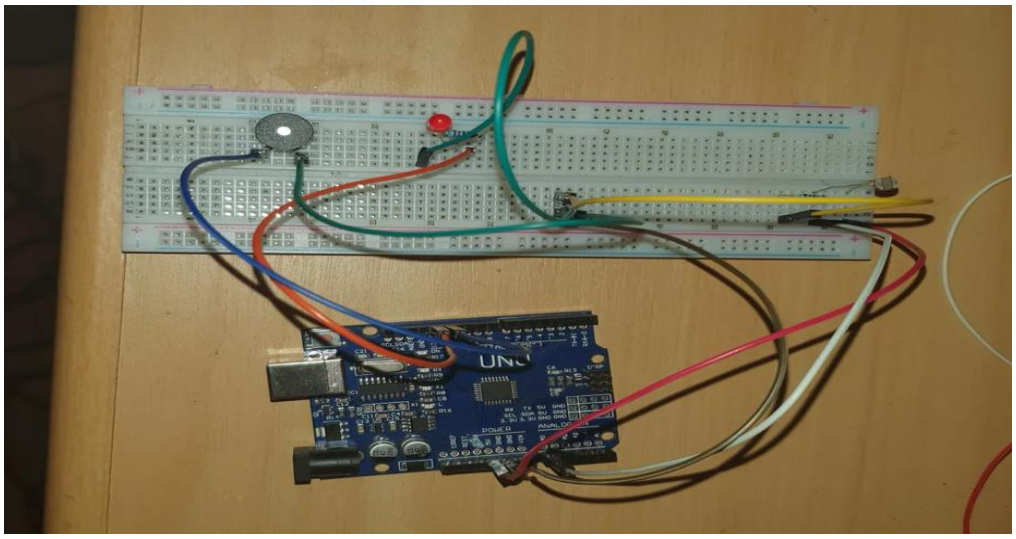
```
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on
  delay(1000);                      // wait for a second

  digitalWrite(LED_BUILTIN, LOW);   // turn the LED off
  delay(1000);}                     // wait for a second
```

LEARNING:

Learned how to upload program to Arduino boards via Arduino IDE, we can choose from multiple templates then customize those, and then upload. Additionally, learned the practical application of the delay function in Arduino programming for time-controlled operations.

PROJECT # 03

light-triggered alarm



Components:

- Buzzer
- LED
- LDR(Light Dependent Resistor)
- 10k ohm resistor
- Wires

Pin Configuration:

- LED connected to Pin 13
- Buzzer connected to Pin 12
- LDR connected to Analog Pin A0

CODE:

```
const int ledPin = 13;

const int buzzerPin = 12;

const int ldrPin = A0;

void setup () {
```

```
    Serial.begin(9600);
    pinMode(ledPin, OUTPUT);
    pinMode(buzzerPin, OUTPUT);
    pinMode(ldrPin, INPUT);
}
void loop() {
    int ldrStatus = analogRead(ldrPin);
    if (ldrStatus >= 400) {
        tone(buzzerPin, 100);
        digitalWrite(ledPin, HIGH);
        delay(100);
        noTone(buzzerPin);
        digitalWrite(ledPin, LOW);
        delay(100);
        Serial.println("----------- ALARM ACTIVATED -----------");
    }
    else {
        noTone(buzzerPin);
        digitalWrite(ledPin, LOW);
    Serial.println("ALARM DEACTIVATED");
    }
```

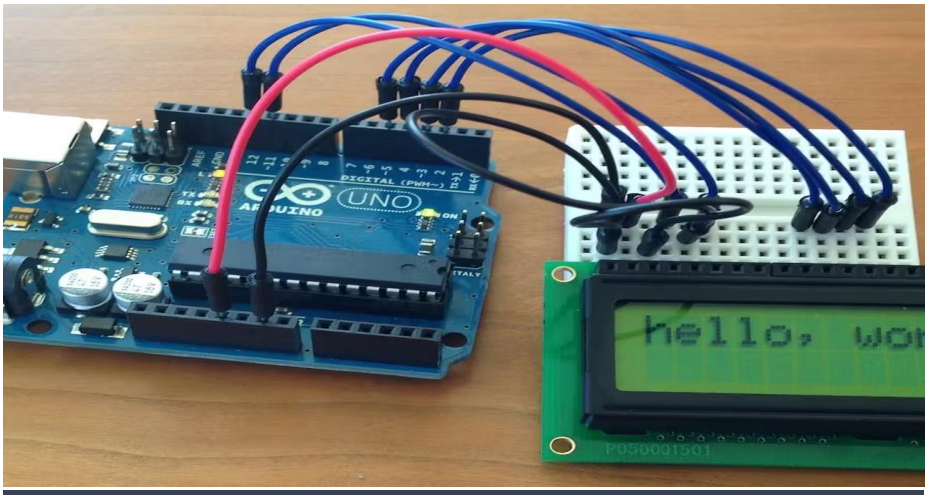
```
}
```

LEARNING:

Learned to incorporate a Light Dependent Resistor (LDR) into a circuit to detect light levels. This alarm consist of both visual (LED on pin 13) and sound indicator (buzzer on pin 12), so learned how to use **tone()** function to generate sound. Also used **serial.println()** to print info about status of alarm.

PROJECT # 04

Hello World Printing On LCD Screen



Components:

- Arduino
- LCD
- wires

Pin Configuration:

LCD	ARDUINO
RS	12
E	11
D4	5
D5	4
D6	3
D7	2

CODE:

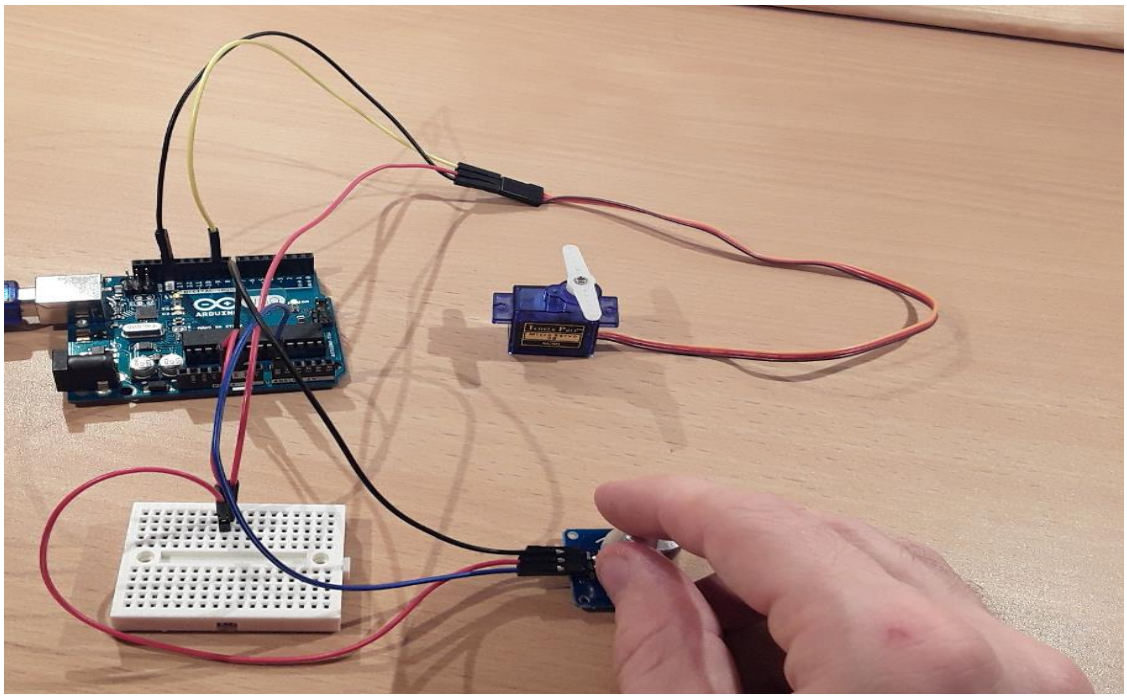
```
#include <LiquidCrystal.h>
```

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

void setup(){
    // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  // initialize the serial communications:
  Serial.begin(9600);
}

void loop()
{
  // when characters arrive over the serial port...
  if (Serial.available()) {
    // wait a bit for the entire message to arrive
    delay(100);

    lcd.clear();
    // read all the available characters
    while (Serial.available() > 0) {
      // display each character to the LCD
      lcd.write(Serial.read());
    }
  }

{
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 2);
  // print the number of seconds since reset:
  lcd.print(millis()/1000);

  }
}
```

LEARNING:

In this project, understand how the computer program (Arduino code) talks to and manages the electronic screen (LCD display). Learn to move around and show letters on the screen using commands like "lcd.setCursor" and "lcd.write()". Also, get the hang of starting up the screen properly using "lcd.begin" to make sure it works well with the Arduino code.

PROJECT # 05
SERVO MOTOR WITH POTENTIOMETER



Components:

- Potentiometer
- Arduino
- Servo motor
- Wires
- BreadBoard

Pin Configuration:

- **Potentiometer**

Two outer pins are connected to power (+5V) and ground
The middle pin is connected to input 0 on the board

- **Servo motor**

three wires: power wire connected to the 5V pin on the Arduino, ground wire is connected to a ground pin on the board, signal pin is orange and connected to pin 9

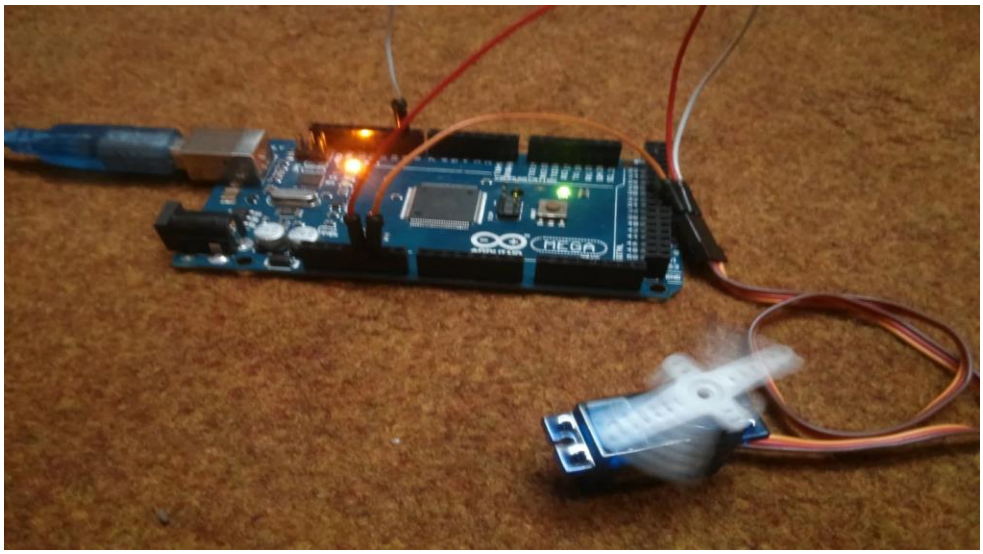
CODE:

```
#include <servo.h>
int servoPin = 9;

int potPin = A0;

void setup() {

  Servo1.attach(servoPin);}

void loop() {

  int reading = analogRead(potPin);

  int angle = map(reading, 0, 1023, 0, 180);

  Servo1.write(angle);}
```

LEARNING:

Understand how a potentiometer functions, with its two outer pins connected to power and ground, and the middle pin connected to an input on the Arduino board. Learn the wiring configuration for a servo motor, with its power wire connected to the 5V pin, ground wire to a ground pin, and the signal wire (orange) connected to pin 9 on the Arduino. Integrate the Servo library in the code (#include <Servo.h>) to enable the use of servo-related functions.

PROJECT # 06
SERVO MOTOR



Components:

- Arduino Uno
- Servo motor

Pin Configuration:

- Servo motor

Three pins: one is usually the ground, power wire to 5V on the Arduino and third to a digital pin on the Arduino.

CODE:

```
#include <Servo.h>
// Declare the Servo pin
int servoPin = 3;
// Create a servo object
Servo Servo1;
void setup() {
  // We need to attach the servo to the used pin number
  Servo1.attach(servoPin);
}
void loop(){
```

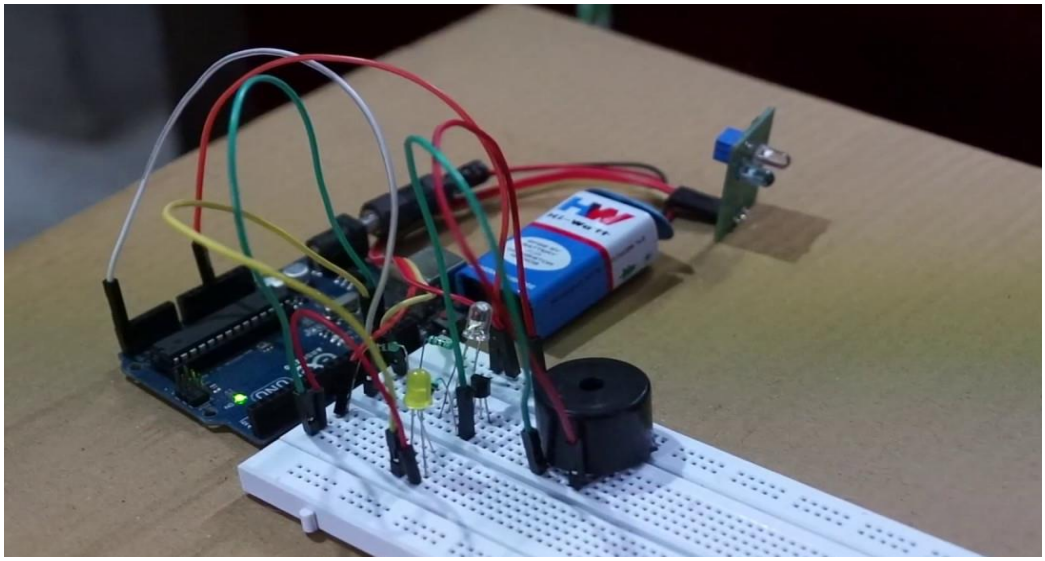
```
  // Make servo go to 0 degrees
  Servo1.write(0);
  delay(1000);
  // Make servo go to 90 degrees
  Servo1.write(90);
  delay(1000);
  // Make servo go to 180 degrees
  Servo1.write(180);
  delay(1000);
}
```

LEARNING:

Learned to use the servo motor with servo.h library, also learned to control it’s rotation using servo.write()

PROJECT # 07

Obstacle Detector By IR Sensor



Components:

- IR sensor
- Buzzer
- 220 ohm resistor
- Jumper wires

Pin Configuration:

Arduino	IR sensor
+5v	Vcc
GND	GND
D2	OUT Pin

Arduino	Buzzer
AO pin	positive
GND	negative

CODE:

```
int val = 0 ;
 void setup()
 {
 Serial.begin(9600); // sensor buard rate
 pinMode(2,INPUT);  // IR sensor output pin connected
 pinMode(6,OUTPUT);  // LED
 pinMode(7,OUTPUT);  // BUZZER
 }
 void loop()
 {
 val = digitalRead(2);  //  IR sensor output pin connected
 Serial.println(val);  // see the value in serial monitor in Arduino IDE
 delay(500);
 if(val == 1 )
 {
 digitalWrite(6,HIGH);  // LED ON
 digitalWrite(7,HIGH);  // BUZZER ON
 }
 else
 {
 digitalWrite(6,LOW);  // LED OFF
 digitalWrite(7,LOW);  // BUZZER OFF
 }
 }

// IR Sensor with Arduino

 void setup()
 {
 Serial.begin(9600); // sensor baud rate
 pinMode(6,OUTPUT);  // LED
 pinMode(7,OUTPUT);  // BUZZER
 }
 void loop()
 {
 int s1=analogRead(A0); //ANALOG PIN FOR SENSOR
 Serial.println(s1);   // see the value in serial monitor in Arduino IDE
 delay(100);
 if(s1>200 )
```

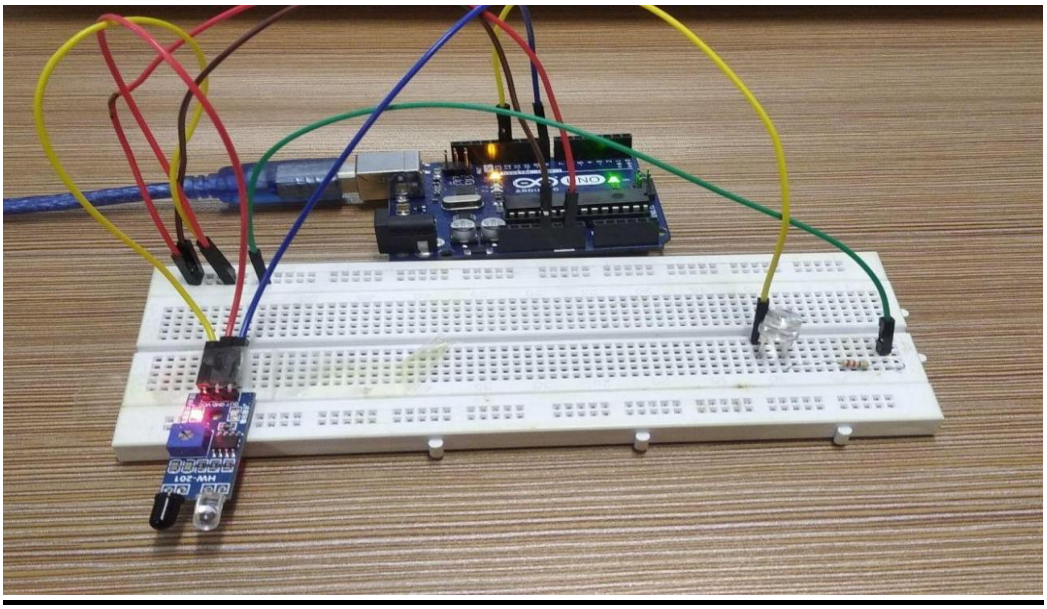
```
{
digitalWrite(6,HIGH);  // LED ON
digitalWrite(7,HIGH);  // BUZZER ON
}
else
{
digitalWrite(6,LOW);  // LED OFF
digitalWrite(7,LOW);  // BUZZER OFF
}

}
```

LEARNING:

This project is about making an Arduino respond to an IR sensor. The sensor checks for infrared signals or reflections. When it detects something, the Arduino makes a light (LED) and a sound (buzzer) go on.

PROJECT # 08

IR SENSOR WITH LED



COMPONENTS:

- IR sensor
- LED
- Arduino
- wires

CODE:

```
int irPin = 2;  // IR sensor output pin

int ledPin = 6; // LED pin

void setup() {

  Serial.begin(9600); // Initialize serial communication

  pinMode(irPin, INPUT);  // IR sensor pin as input

  pinMode(ledPin, OUTPUT); // LED pin as output
```

```
}

void loop() {

  int irValue = digitalRead(irPin); // Read the output of the IR sensor

  Serial.println(irValue); // Print the value to the serial monitor

  if (irValue == HIGH) {

    digitalWrite(ledPin, HIGH); // Turn on the LED if IR sensor detects something

  } else {

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

  }

  delay(500); // Add a delay to avoid rapid readings

}
```

LEARNING:

Understand how to connect and interact with an Infrared (IR) sensor, which detects the presence or absence of infrared signals. Learn to control a Light Emitting Diode (LED) using an Arduino board based on the readings from the IR sensor.

PROJECT # 09

Ultrasonic Sensor for Distance Measurement

Components:

- Ultrasonic sensor
- Arduino Uno
- Jumper wires
- LCD

PIN CONFIGURATION:

- Ultrasonic Sensor :

Trigger Pin (trig): Connected to digital pin 2.
Echo Pin (echo): Connected to digital pin 3.

- I2C LCD:

SDA (Serial Data): Connected to the A4 pin on the Arduino Uno.
SCL (Serial Clock): Connected to the A5 pin on the Arduino Uno.

CODE:

```
#include <Wire.h>
```

```
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

int trig = 2;

int echo = 3;

long duration, distance;

void setup()

{

  pinMode(trig, OUTPUT);

  pinMode(echo, INPUT);

  lcd.begin( );  // You might need to use lcd.begin(columns, rows) instead

  lcd.clear();

  lcd.backlight();

  lcd.setCursor(2, 0);

  lcd.print("Distance");

  lcd.setCursor(1, 1);

  lcd.print("UltraSonic");

  delay(2000);

  lcd.clear();

}

void loop()

}
```

```
digitalWrite(trig, HIGH);

delayMicroseconds(2);

digitalWrite(trig, LOW);

delayMicroseconds(10);

digitalWrite(trig, HIGH);

duration = pulseIn(echo, HIGH);

distance = (duration / 2) * 0.0346;

lcd.setCursor(2, 0);

lcd.print(distance);

lcd.setCursor(4, 0);

lcd.print("cm");}
```

LEARNING:

In this Arduino project, an ultrasonic distance sensor is interfaced with the Arduino board along with Liquid Crystal Display (LCD). This code continuously measures and displays the distance on the LCD in centimeters. The setup utilizes the Wire library for I2C communication with the LCD and the pulseIn function to measure the duration of the echo pulse from the ultrasonic sensor.