Experiment No.: 03

Name of the Experiment: Hands-On Projects with Arduino Microcontroller

Objectives:

- Understand the basics of Arduino and its interfacing.
- Build 20 hands-on mini projects using Arduino.
- Improve practical skills with sensors, actuators, and displays.

Project No.: 01

Project Name: LED Brightness Control with POT

Overview: This project demonstrates how to control the brightness of an LED using a potentiometer. The analog signal from the POT is read by Arduino and used to adjust LED brightness using PWM.

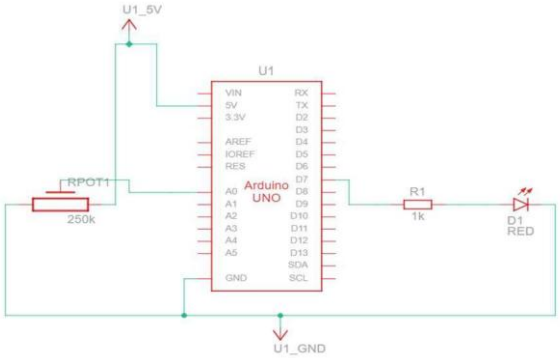


Fig.03.1.1: Schematic Diagram LED Brightness Control with POT Circuit

Required Apparatus:

Table 3.1.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	LED	5mm	1
3	Resistor	1k ohm	1
4	Potentiometer	100k ohm	1
5	Breadboard	400 tie-points	1
6	Jumper Wires	Male-to-Male, 20 cm	Several

Circuit Diagram:

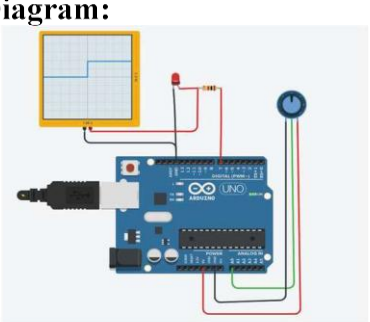


Fig.03.1.2: Circuit Diagram of LED Brightness Control with POT

Experimental Setup:

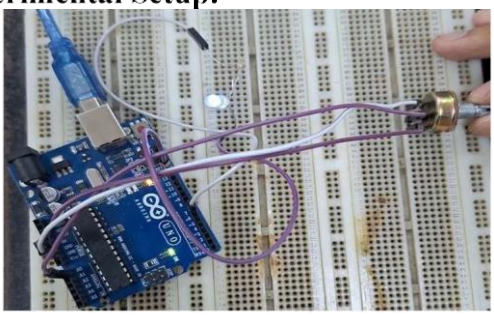


Fig.03.1.3: Experimental setup of the LED Brightness Control with POT

Code:

```
//2101130
void setup()
{
  pinMode(7, OUTPUT);
}
void loop()
{
  analogWrite(7, analogRead(A0));
}
```

Discussions:

This project's main purpose was to increase or decrease brightness of LED. It's essential in many Works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 02

Project Name: Traffic Light with Button Input

Overview:

This project simulates a traffic light system with a push-button to simulate a pedestrian request. It helps understand basic traffic light logic and input handling with pull-down resistors

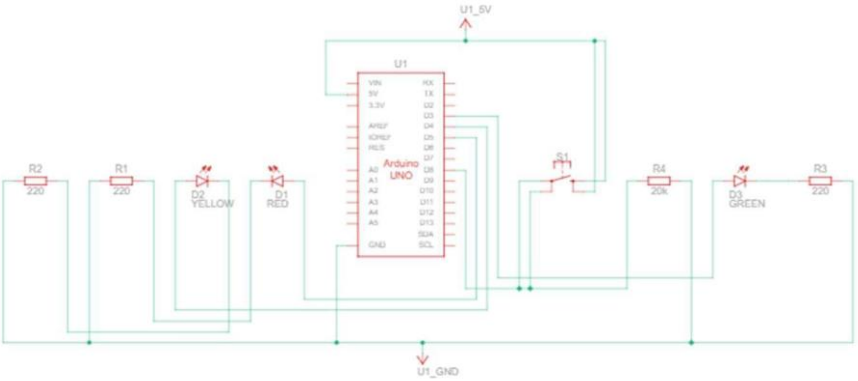


Fig.3.2.1: Schematic Diagram of Traffic Light with Button Input Circuit

Required Apparatus:

Table 3.2.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	LEDs	Red, Yellow,Green	1 each
3	Resistors	1k ohm	3
4	Push Button	Normally Open (NO)	1
5	Resistor	10k ohm	1
6	Breadboard	400 tie-points	1
7	Jumper Wires	Male-to-Male, 20 cm	Several

Circuit Diagram:

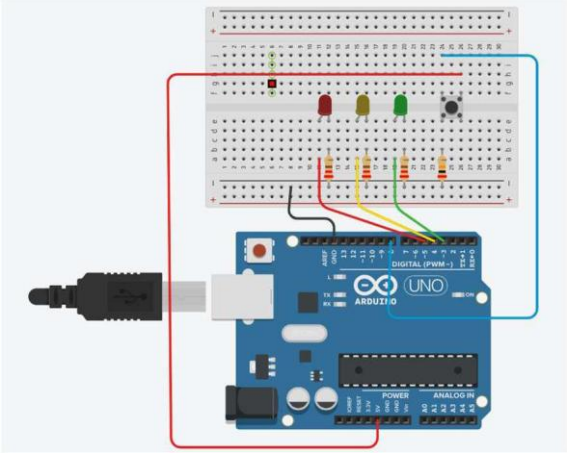


Fig.03.2.2: Circuit Diagram of Traffic Light with Button Input

Experimental Setup:

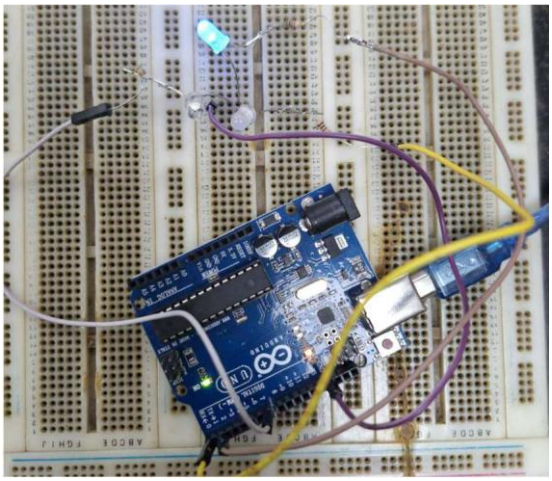


Fig.03.2.3: Experimental setup of the Traffic Light with Button Input

Code:

```
void setup() {
    pinMode(3, OUTPUT);  // declare pin 3 (green LED) as output
    pinMode(4, OUTPUT);  // declare pin 4 (yellow LED) as output
    pinMode(5, OUTPUT);  // declare pin 5 (red LED) as output
    pinMode(8, INPUT);   // declare pin 8 (button) as input
}
void loop() {
    // Main code to run repeatedly:
    if (digitalRead(8) == LOW) {
        digitalWrite(3, HIGH);  // Light up the LEDonpin 3
        digitalWrite(4, LOW);   // Switch off the LED on pin 4
        digitalWrite(5, LOW);   // Switch off the LED on pin 5
        delay(10000);           // Wait for 10 seconds
        digitalWrite(3, LOW);   // Switch off the LED on pin 3
        digitalWrite(4, HIGH);  // Light up the LED on pin 4
        digitalWrite(5, LOW);   // Switch off LED on pin 5
        delay(3000);            // Wait for 3 seconds
        digitalWrite(3, LOW);   // Switch off the LED on pin 3
        digitalWrite(4, LOW);   // Switch off the LED on pin 4
        digitalWrite(5, HIGH);  // Light up LED on pin 5
        delay(10000);           // Wait for 10 seconds
    }
    if (digitalRead(8) == HIGH) {  //if the button is pressed:
        digitalWrite(3, HIGH);       // Light up the LED on pin 3
        digitalWrite(4, LOW);        // Switch off the LED on pin 4
        digitalWrite(5, LOW);        // Switch off the LED on pin 5
        delay(500);                  // Wait half a second
        digitalWrite(3, LOW);        // Switch off the LED on pin 3
        delay(500);                  // Wait half a second
    }
}
```

Discussions:

This project's main purpose was to create a traffics light control system. It's essential in many Works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 03

Project Name: Digital Dice with LCD Display

Overview:

This project implements a digital dice using an Arduino, a push-button, and an I2C LCD. When the button is pressed, the Arduino generates a random number between 1 and 6 and displays the result on the LCD. It serves as an introduction to generating random numbers, using input buttons with pull-up resistors, and outputting data to an I2C LCD module.

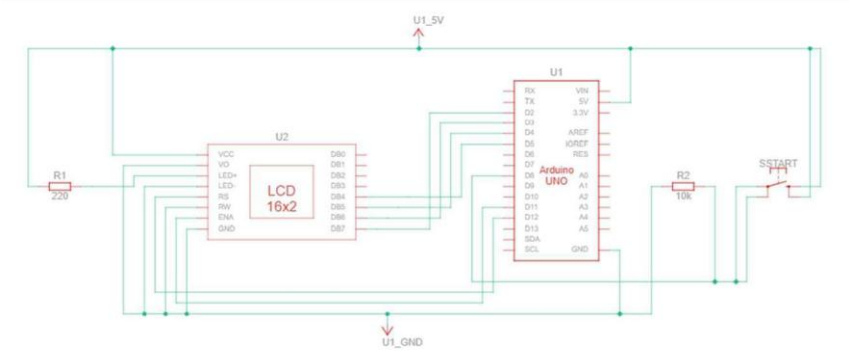


Fig.03.3.1: Schematic Diagram of Digital Dice with LCD Display

Required Apparatus:

Table 3.3.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Push Button	SPST	1
3	I2C LCD Display	16x2	1
4	Breadboard	400 tie-points	1
5	Jumper Wires	Male-to-Male, 20 cm	Several
6	resistor	10k Ohm	1

Circuit Diagram:

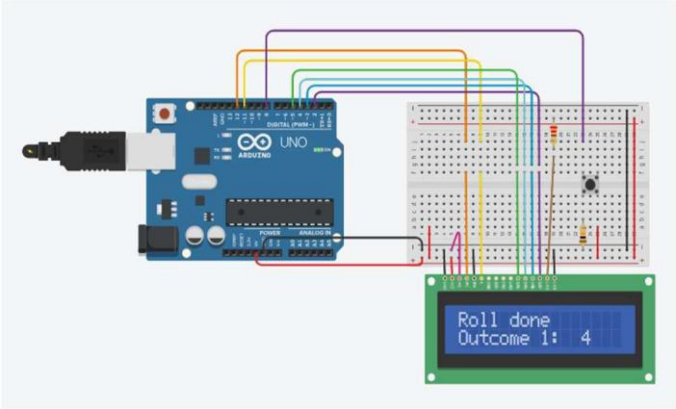


Fig.03.3.2: Circuit Diagram of Digital Dice with LCD Display

Experimental Setup:

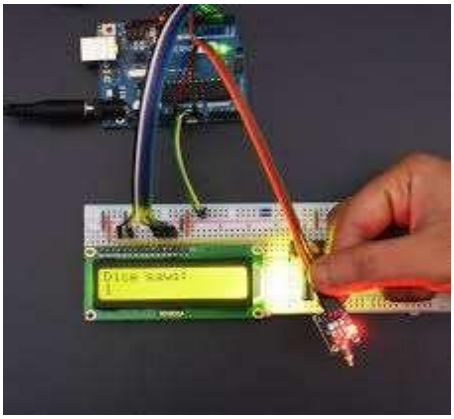


Fig.03.3.3: Experimental setup of Digital Dice with LCD Display

Code:

```
//2101127
#include <LiquidCrystal.h>

#define BUTTON 8
#define LCD_RS 12
#define LCD_EN 11
#define LCD_D4 5
#define LCD_D5 4
#define LCD_D6 3
#define LCD_D7 2
#define LCD_COLS 16
#define LCD_ROWS 2
#define ROLL_STEP 250
#define ROLL_STEPS 6

LiquidCrystal lcd(LCD_RS, LCD_EN, LCD_D4,
LCD_D5, LCD_D6, LCD_D7);
int rollCount = 0;
void setup() {
  pinMode(BUTTON, INPUT);
  lcd.begin(LCD_COLS, LCD_ROWS);
  lcd.setCursor(0, 0);
  lcd.print("Dice game!");
  randomSeed(analogRead(0));
}
```

```
void rollDie() {
  lcd.setCursor(0, 1);
  lcd.print("Rolling        ");
  for (int i = 0; i < ROLL_STEPS;
i++) {
    lcd.print(".");
    delay(ROLL_STEP);
  }
  int value = random(1, 7);
  rollCount++;
lcd.setCursor(0, 0);
  lcd.print("Roll done      ");
  lcd.setCursor(0, 1);
  lcd.print("Outcome ");
  lcd.print(rollCount);
  lcd.print(": ");
  lcd.print(value);
}

void loop() {
  if (digitalRead(BUTTON) == HIGH) {
    rollDie();
  }
}
```

Discussions:

This project's main purpose was to create digital dice with LCD display. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 04

Project Name: Mood Light with POT.

Overview:

This project creates a mood light whose brightness or color can be adjusted using a potentiometer. It demonstrates the use of analog input to control LED output. Ideal for learning PWM and analog signal processing with Arduino.

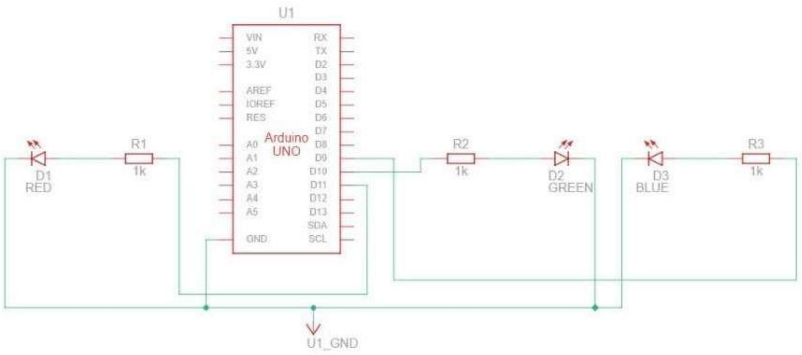


Fig.03.4.1: Schematic Diagram of Mood Light with POT.

Required Apparatus:

Table 3.4.1: Table For Required Apparatus

Sl. No.	Name of Apparatus	Specifications	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Breadboard	---	1
3	RGB LED	Common cathode, 4-pin	1
4	Potentiometer	10kΩ	3
5	Resistor	220Ω	3
6	Jumper Wires	---	~15

Circuit Diagram:

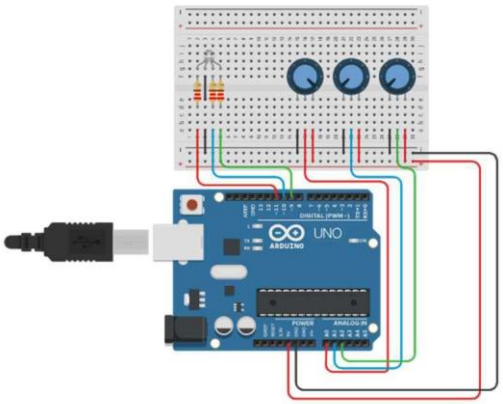


Fig.03.4.2: Circuit Diagram of Mood Light with potentiometers.

Experimental Setup:

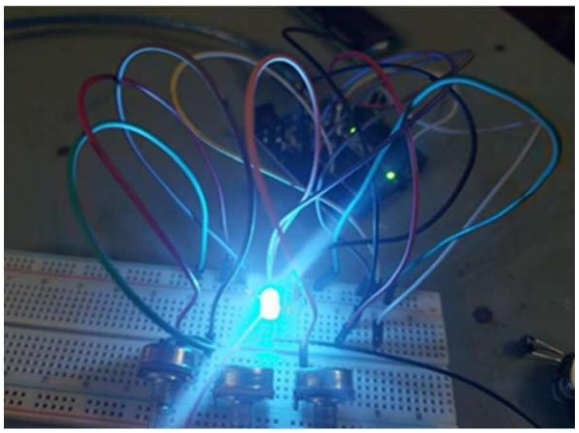


Fig.03.4.3: Experimental setup of of Mood Light with potentiometers.

Code:

```
int pinR = 11;
int pinB = 10;
int pinG = 9;

int potenR;
int potenB;
int potenG;

void setup()
{
  pinMode(11, OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(9, OUTPUT);
  Serial.begin(9600);
}
void loop()
{
  potenR   map(analogRead(A0),
1023, 0, 255);
  potenB   map(analogRead(A1),
1023, 0, 255);
  potenG   map(analogRead(A2),
1023, 0, 255);
  analogWrite(pinR, potenR
  analogWrite(pinB, potenB
  analogWrite(pinG, potenG
  delay(50);
Serial.print(" Red: ");
  Serial.print(potenR);
  Serial.print(" Blue
  Serial.print(potenB);
    ri   print   Gre
  Serial.println(potenG);
  delay(100);
}
```

Discussions:

This project's main purpose was to create Mood Light with POT. It's essential in many Works. Though some problems occurred due to faulty wirings. All problems were overcome by proper and careful wirings and arrangement.

Project No.: 05

Project Name: Display temperature and humidity using DHT11 sensor and track min/max values on LCD.

Overview:

This project logs temperature and humidity using the DHT11 sensor and displays it on an I2C LCD. It also keeps track of minimum and maximum values over time.

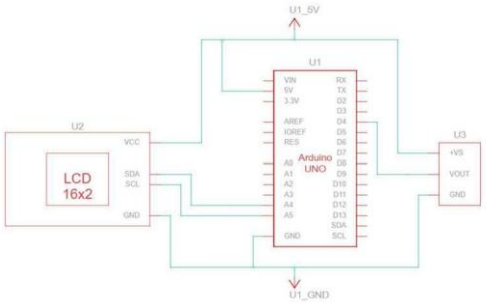


Fig.03.5.1: Schematic diagram of temperature and humidity measure on LCD using DHT11

Required Apparatus:

Table 3.5.1: Table for Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	DHT11 Sensor	0-50^{0}C	1
3	I2C LCD Display	16x2	1
4	Breadboard	--	1
5	Jumper Wires	--	Several

Circuit Diagram:

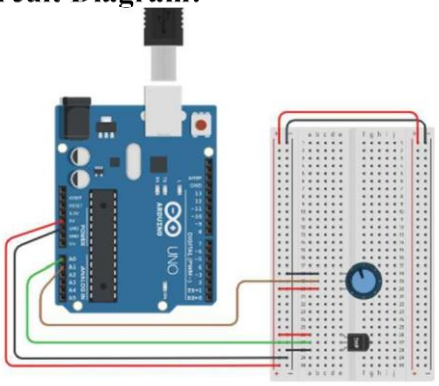


Fig.03.5.2: Circuit diagram of temperature and humidity measure on LCD using DHT11

Experimental Setup:

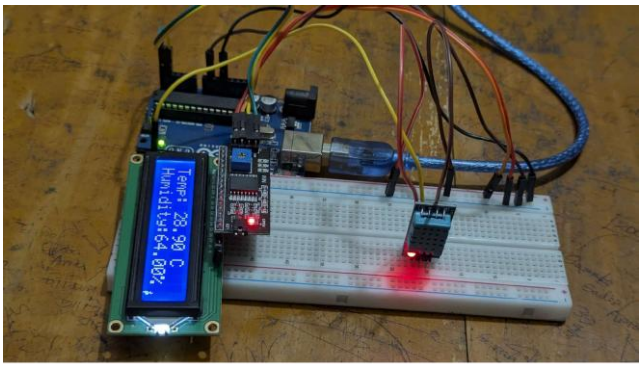


Fig.03.5.3: Experimental setup of the temperature and humidity measure on LCD using DHT11

Code:

```
#include <Wire.h>
     //Include I2C library
#include <LiquidCrystal_I2C.h>
// Include LCD I2C library
#include <DHT.h>
// Include DHT library
// Initialize I2C LCD with 16
columns and 2 rows
LiquidCrystal_I2C lcd(0x27, 16,
2);       // LCD address 0x27,
16x2 display
// Initialize DHT11 sensor
#define DHTPIN 2
// DHT11 data pin connected to
D2
#define DHTTYPE DHT11
// Define sensor type
DHT dht(DHTPIN, DHTTYPE);
void setup() {
lcd.begin(16, 2);       //
Initialize LCD with 16 columns,
2 rows
lcd.backlight();       // Turn
on LCD backlight
dht.begin();           //
Initialize DHT sensor
lcd.setCursor(0, 0);
lcd.print("Temperature &");
lcd.setCursor(0, 1);
lcd.print("Humidity Test...");
delay(5000);
lcd.clear();}
void loop() { // Read temperature and
humidity
float humidity = dht.readHumidity();
float temperature =
dht.readTemperature(); // Check if
readings are valid
if (isnan(humidity) ||
isnan(temperature)) {
lcd.clear();
lcd.print("Sensor Error");
lcd.setCursor(0, 1);
lcd.print("Check Connection");
delay(1000);
return;
} // Display temperature and humidity
on LCD
lcd.clear();
lcd.setCursor(0, 0); // First row
lcd.print("Temp: ");
lcd.print(temperature);
lcd.print(" C");
lcd.setCursor(0, 1); // Second row
lcd.print("Humidity:");
lcd.print(humidity);
lcd.print("%");
delay(2000); // Refresh every 2 seconds
}
```

Discussions:

This project's main purpose was to detect or display temperature and humidity. It's essential in many works, specially in scientific labs, big pharma and industrial areas. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 06

Project Name: Ultrasonic Distance Meter on LCD

Overview:

This project uses the HC-SR04 ultrasonic sensor to measure distance and displays the result on an LCD. The project provides a practical application of time-of-flight distance measurement.

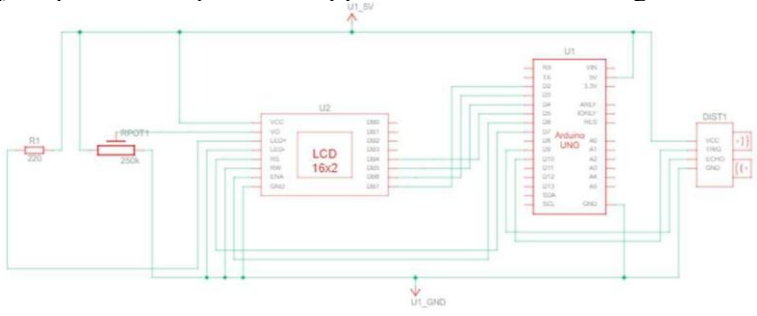


Fig.03.6.1: Schematic Diagram of Ultrasonic Distance Meter on LCD

Required Apparatus:

Table 3.6.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Ultrasonic Sensor	HC-SR04	1
3	I2C LCD Display	16x2	1
4	Breadboard	--	1
5	Jumper Wires	--	Several
6	POT	250k Ohm	1

Circuit Diagram:

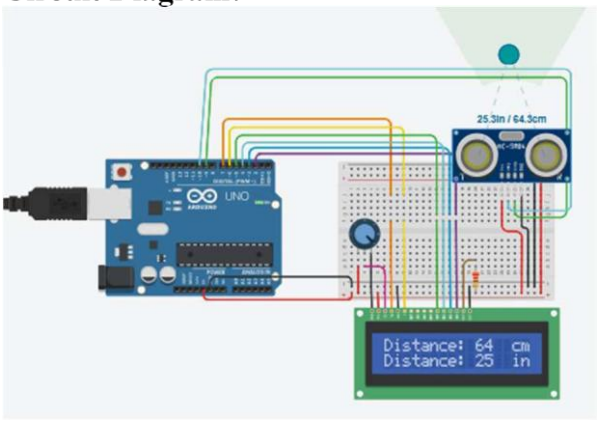


Fig.03.6.2: Circuit Diagram of Ultrasonic Distance Meter on LCD

Experimental Setup:

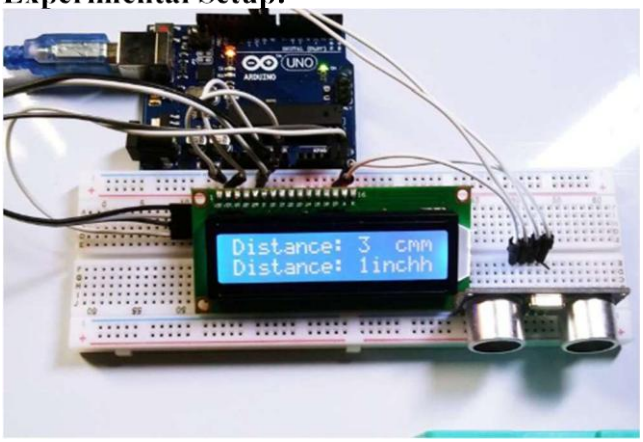


Fig.03.6.3: Experimental setup of the Ultrasonic Distance Meter on LCD

Code:

```
#include <LiquidCrystal.h>
#define ECHO 9
#define TRIGGER 10
#define TIEMPO_MUESTREO 1000
#define PULSO_TRIGGER 1
int DURACION;
float DISTANCIA;        // in cm
float DISTANCIA_INCH;  // in inch

LiquidCrystal lcd(7, 6, 5, 4, 3, 2);

void setup() {
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  Serial.begin(9600);
  lcd.begin(16, 2);
}
void loop() {
  digitalWrite(TRIGGER, HIGH);
  delay(PULSO_TRIGGER);
  digitalWrite(TRIGGER, LOW);

  DURACION = pulseIn(ECHO, HIGH);
  DISTANCIA = 0.01716 * DURACION;
DISTANCIA_INCH = DISTANCIA / 2.54;
  Serial.print("Distance (cm): ");
  Serial.print(DISTANCIA);
  Serial.print("  |  Distance (inch):
");

  Serial.println(DISTANCIA_INCH);
  lcd.setCursor(0, 0);
  lcd.print("Distance:     cmm");

  lcd.setCursor(0, 1);
  lcd.print("Distance:     inch");

  // Print cm value before 'cmm'
  lcd.setCursor(10, 0);
  lcd.print("    ");  // clear space
  lcd.setCursor(10, 0);
  lcd.print((int)DISTANCIA); // Rounded
value
  lcd.setCursor(10, 1);
  lcd.print("    ");  // clear space
  lcd.setCursor(10, 1);
  lcd.print((int)DISTANCIA_INCH);

  delay(TIEMPO_MUESTREO);
```

Discussions:

This project's main purpose was to measure distance by transducers. It's essential in many Works, specially in automated car industry. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 07

Project Name: Soil Moisture Level with Display

Overview: This project monitors soil moisture in real-time using a sensor and displays the data on an LCD or OLED. It helps users identify when soil is too dry or wet, making it ideal for smart farming and gardening.

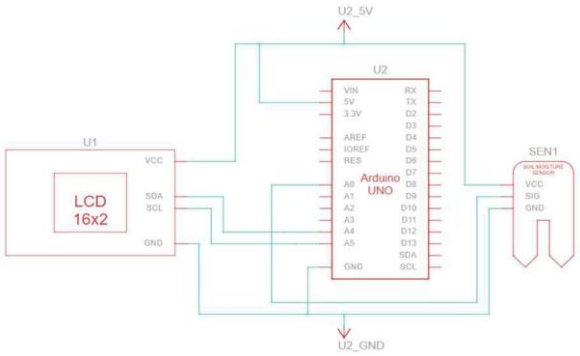


Fig.03.7.1: Schematic diagram of Soil Moisture Level with Display.

Required Apparatus:

Table 3.7.1: Table for Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Soil Moisture Sensor	YL-69	1
3	I2C LCD Display	16x2	1
4	Jumper Wires	--	Several

Circuit Diagram:

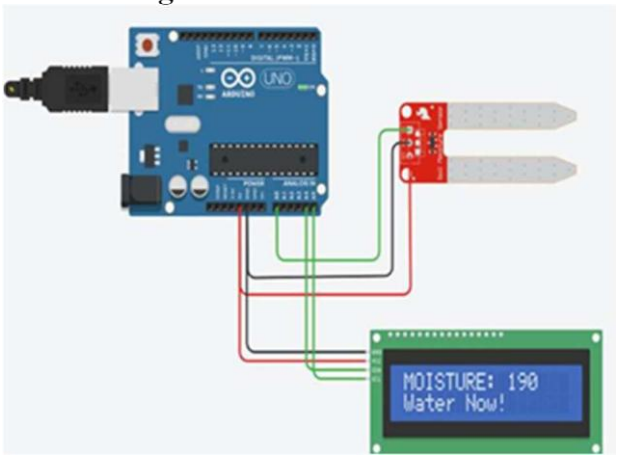


Fig.03.7.2: Circuit diagram of Soil Moisture Level with Display.

Experimental Setup:

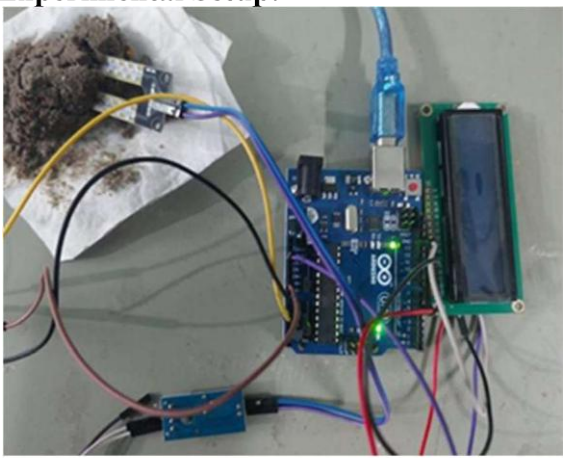


Fig.03.7.3: Experimental setup of the Soil Moisture Level with Display.

Code:

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x26, 16, 2);
void setup() {
Serial.begin(115200);
  lcd.init();
  lcd.backlight();
}
void loop() {
  int level = analogRead(A0);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("MOISTURE: ");
  lcd.print(level);
  lcd.setCursor(0, 1);

  if (level < 400) {
    lcd.print("Water Now!");
  } else if (level < 700) {
    lcd.print("Moist OK");
  } else {
    lcd.print("Wet, No Water");
  }
  delay(1000);
}
```

Project No.: 08

Project Name: POT-Controlled Servo Motor

Overview: In this project, a potentiometer was used to control the rotation angle of a servo motor by mapping the analog input value to the servo's position. The Arduino interpreted the potentiometer value and adjusted the servo angle accordingly using the servo.write() function.

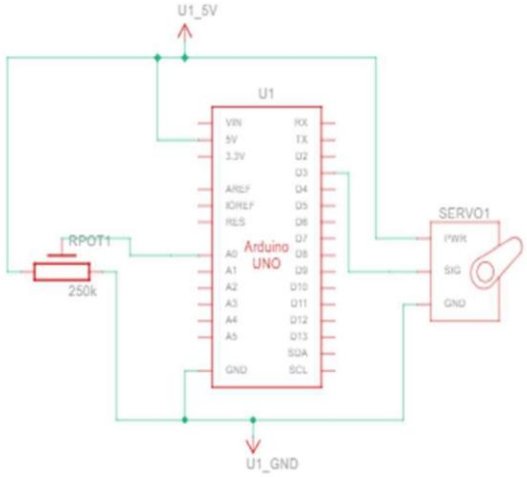


Fig. 3.8.1: Schematic Diagram of POT-Controlled Servo Motor Circuit

Required Apparatus:

Table 3.8.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1.	Arduino Uno	ATmega328P Microcontroller	1
2.	Potentiometer	10kΩ Linear Taper	1
3.	Servo Motor	5V, Standard Torque (e.g., 9g)	1
4.	Breadboard	400 tie-points	1
5.	jumper wires	Male-to-Male, 20 cm	-

Circuit Diagram

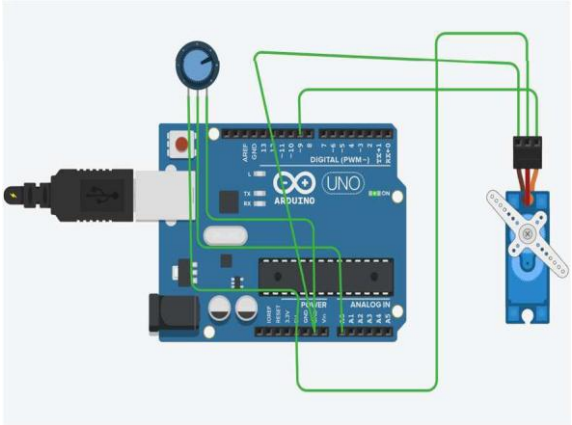


Fig. 3.8.2: POT-Controlled Servo Motor

Experimental Setup:

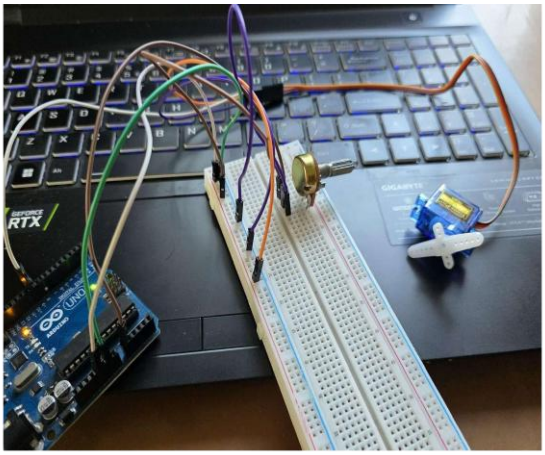


Fig. 3.8.3: Experimental setup of the experiment

Code:

```
#include <Servo.h>
Servo myServo;
const int potPin = A0;
const int servoPin = 9;
void setup() {
  myServo.attach(servoPin);
  Serial.begin(9600);
}
void loop() {
  int potValue = analogRead(potPin);
  int angle = map(potValue, 0, 1023, 0, 180);
  myServo.write(angle);
  delay(15);
                    }
```

Discussion:

This project's main purpose was to create a POT controlled servo motor. It's essential in many Works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 09

Project Name: LCD Thermometer

Overview:

This project displays real-time temperature readings on an LCD using a temperature sensor like the LM35 or DHT11. It's a simple way to learn sensor interfacing and data visualization with Arduino.

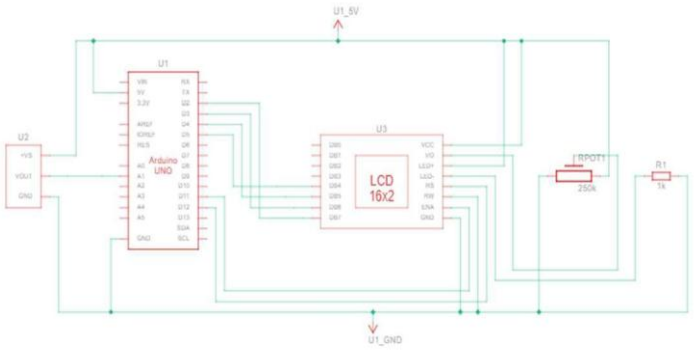


Fig. 3.9.1: Schematic Diagram of LCD Thermometer

Required Apparatus:

Table 3.9.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1.	Arduino Uno	ATmega328P Microcontroller	1
2.	LM35 Temperature Sensor	voltage range of 4V to 30V	1
3.	I2C LCD	16x2	1
4.	Breadboard	400 tie-points	1
5.	Jumper wires	Male-to-Male, 20 cm	10+

Circuit Diagram:

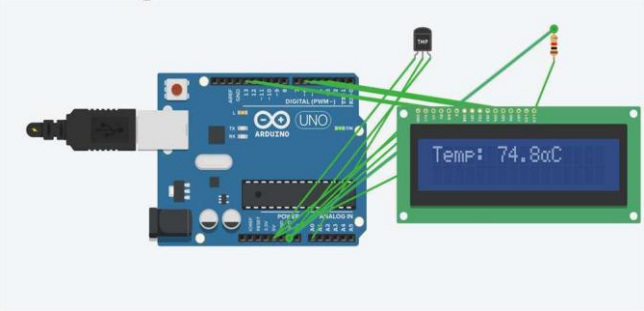


Fig. 3.9.1: LCD Thermometer

Experimental Setup:

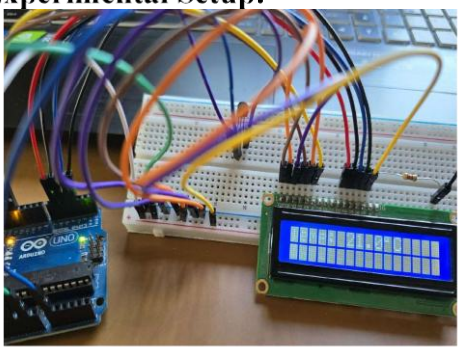


Fig. 3.9.3: Experimental setup of the LCD Thermometer

Code:

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
const int sensorPin = A0;
LiquidCrystal_I2C lcd(0x27, 16, 2);
void setup() {
  lcd.init();
  lcd.backlight();
}
void loop() {
  int sensorValue = analogRead(sensorPin);
  float voltage = sensorValue * (5.0 / 1023.0);
  float temperatureC = voltage * 100;
lcd.setCursor(0, 0);
  lcd.print("Temp: ");
  lcd.print(temperatureC);
  lcd.print(" C");
  delay(1000);
}
```

Discussions:

This project's main purpose was to create a digital thermometer. It's essential in medical science. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 10

Project Name: Clap Switch with LED Indicator

Overview:

In this project, a sound sensor was used to detect a clap, which acted as a trigger to turn an LED on or off. The optional pushbutton allowed the LED to be toggled manually, adding an extra layer of control to the system.

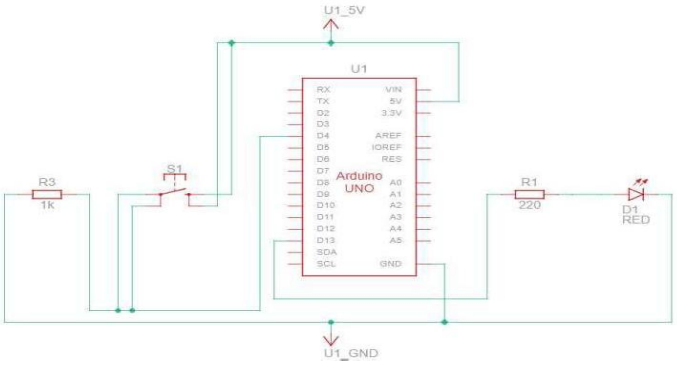


Fig.03.10.1: Schematic diagram of Clap Switch with LED Indicator.

Required Apparatus:

Table 3.10.1: Table for Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Sound Sensor	LM393, 3.3-5V DC	Sound Sensor
3	LED	Green	1
4	Pushbutton	~12V DC, 50-100mA	1
5	Resistor	220Ω, 10kΩ	1
6	Breadboard	--	1
7	Jumper Wires	--	Several

Circuit Diagram:

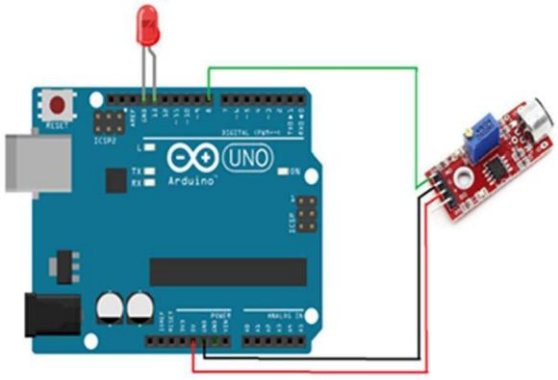


Fig.03.10.2: Circuit diagram of Clap Switch with LED Indicator.

Experimental Setup:

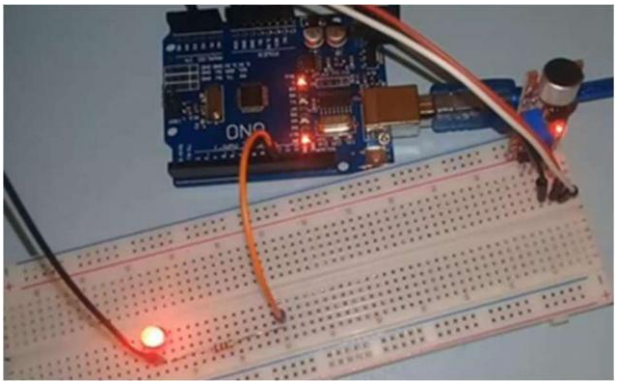


Fig.03.10.3: Experimental setup of Clap Switch with LED Indicator.

Code:

```
const int soundPin = 8;
const int ledPin = 13;
bool ledState = false;
void setup() {
  pinMode(soundPin, INPUT);
  pinMode(ledPin, OUTPUT);
}
void loop() {
  if (digitalRead(soundPin) == HIGH) {
    ledState = !ledState;
    digitalWrite(ledPin, ledState);
    delay(300);
  }
}
```

Discussions:

This project's main purpose was to create a clap switch with LED indicator. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 11

Project Name: Pushbutton Counter on LCD

Overview:

This project counts button presses using a digital input and displays the result on an LCD. It includes a pull-up resistor to maintain stable input levels.

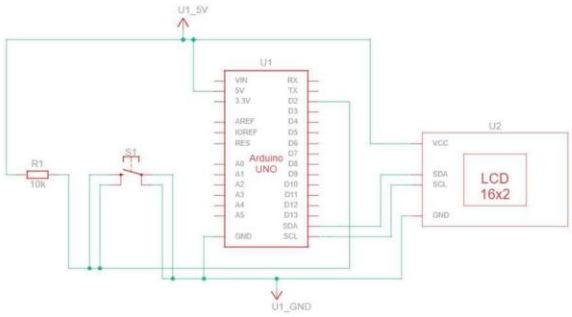


Fig.03.11.1: Schematic Diagram of Pushbutton Counter on LCD Circuit

Required Apparatus:

Table 3.11.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Push Button	--	1
3	Resistor	10k ohm	1
4	I2C LCD Display	16x2	1
5	Breadboard	400 tie-points	1
6	Jumper Wires	Male-to-Male, 20 cm	Several

Circuit Diagram:

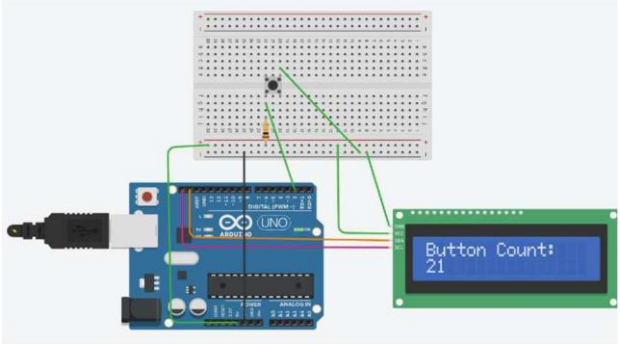


Fig.03.11.2: Circuit Diagram of Pushbutton Counter on LCD

Experimental Setup:

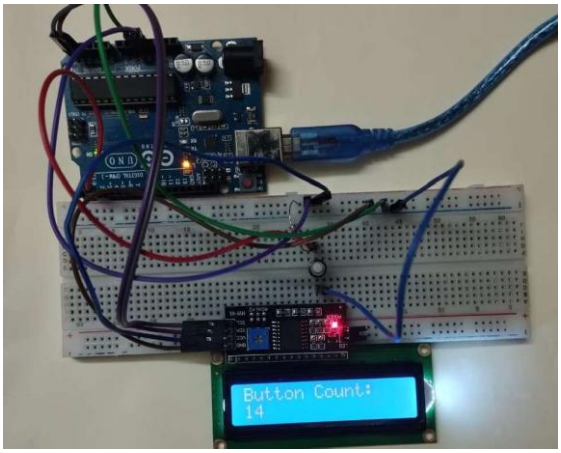


Fig.03.11.3: Experimental setup of the Pushbutton Counter on LCD

Code:

```
#include <Wire.h>
#include <LiquidCrystal_I2C h>

LiquidCrystal_I2C lcd(0x27, 16, 2);

const int buttonPin = 2;
int buttonState;
int lastButtonState = HIGH;
unsigned long lastDebounceTime = 0;
unsigned long debounceDelay = 50;

int pressCount = 0;

void setup() {
  pinMode(buttonPin, INPUT);
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Button Count:");
  lcd.setCursor(0, 1);
  lcd.print(pressCount);
}
```

```
void loop() {
  int reading = digitalRead(buttonPin);


  if (reading != lastButtonState) {
    lastDebounceTime = millis();
  }


  if ((millis() - lastDebounceTime) > debounceDelay) {
    if (reading != buttonState) {
      buttonState = reading;
      if (buttonState == LOW) {
        pressCount++;
        lcd.setCursor(0, 1);
        lcd.print("                ");
        lcd.setCursor(0, 1);
        lcd.print(pressCount);
      }
    }
  }

  lastButtonState = reading;
}
```

Discussions:

This project's main purpose was to create a pushbutton counter on LCD display. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 12

Project Name: Mini Piano with Buzzer and Buttons

Overview:

This project simulates a mini piano using push buttons and a buzzer. Each button triggers a different tone, allowing users to play simple melodies. It's ideal for learning tone generation and digital input handling with Arduino.

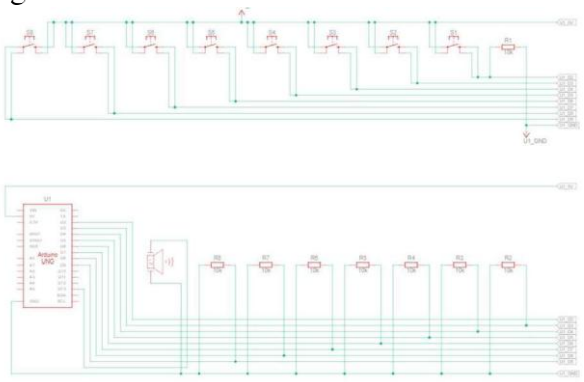


Fig.03.12.1: Schematic diagram of Mini Piano with Buzzer and Buttons

Required Apparatus:

Table 3.12.1: Table for Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Buzzer	5v	1
3	Multiple Pushbuttons	---	1
4	Resistors	10kΩ	1
5	Breadboard	400 tie-points	1
6	Jumper Wires	Male-to-Male, 20 cm	Several

Circuit Diagram:

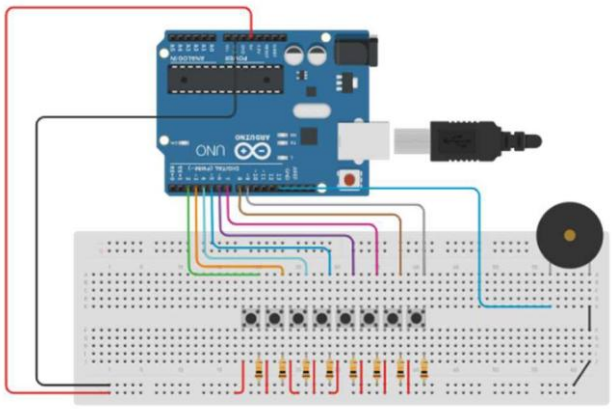


Fig.03.12.2: Circuit diagram of Mini Piano with Buzzer and Buttons

Experimental Setup:

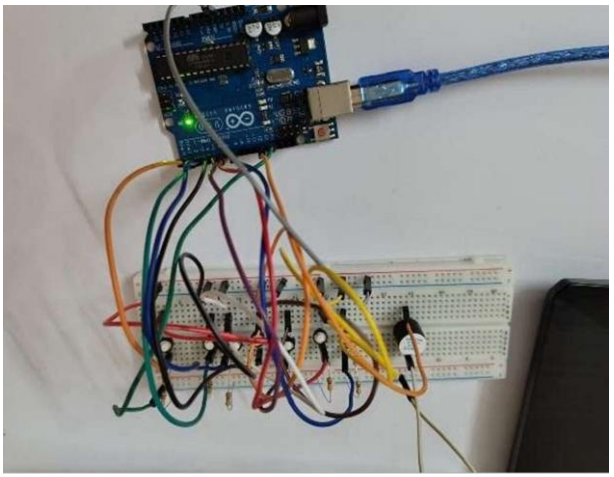


Fig.03.12.3: Experimental setup of the Mini Piano with Buzzer and Buttons

Code:

```
int buzzer = 5;
int button1 = 12;
int button2 = 11;
int button3 = 10;
int button4 = 9;
int button5 = 8;
int button6 = 7;
void setup() {
  pinMode(buzzer, OUTPUT);
  pinMode(button1, INPUT);
  pinMode(button2, INPUT);
  pinMode(button3, INPUT);
  pinMode(button4, INPUT);
  pinMode(button5, INPUT);
  pinMode(button6, INPUT);
}
void loop() {
  if (digitalRead(button1) == LOW)
{
    tone(buzzer, 500, 300);
    delay(200);
    noTone(buzzer);
  }
  if (digitalRead(button2) == LOW)
{
    tone(buzzer, 1000, 300);
    delay(200);
    noTone(buzzer);
  }
if (digitalRead(button3) == LOW) {
    tone(buzzer, 1500, 300);
    delay(200);
    noTone(buzzer);
  }

  if (digitalRead(button4) == LOW) {
    tone(buzzer, 2000, 300);
    delay(200);
    noTone(buzzer);
  }

  if (digitalRead(button5) == LOW) {
    tone(buzzer, 2500, 300);
    delay(200);
    noTone(buzzer);
  }

  if (digitalRead(button6) == LOW) {
    tone(buzzer, 3000, 300);
    delay(200);
    noTone(buzzer);
  }
}
```

Discussions:

This project's main purpose was to create a piano buzzer. It's essential in music, phonetics and linguistic studies Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 13

Project Name: Gas Leak Detector with Alarm.

Overview:

This project detects the presence of flammable gases using a gas sensor and triggers an alarm when gas levels exceed a threshold. It's useful for safety applications and teaches sensor interfacing and threshold-based alerts.

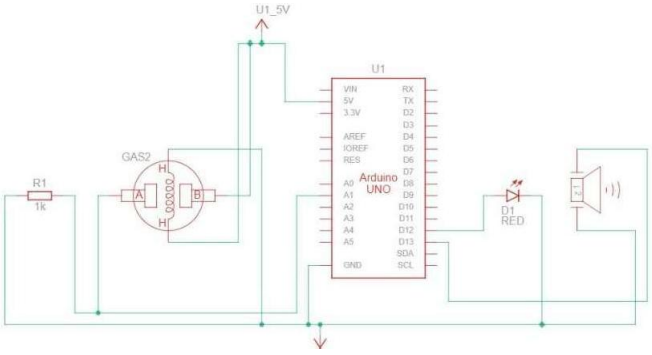


Fig.03.13.1: Schematic diagram of Gas Leak Detector with Alarm.

Required Apparatus:

Table 3.13.1: Table For Required Apparatus

Sl. No.	Name of Apparatus	Specifications	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Breadboard	400 tie-points	1
3	Buzzer	5v	1
4	I2C LCD	16*2	3
5	Jumper Wires	Male-to-Male, 20 cm	~15

Circuit Diagram:

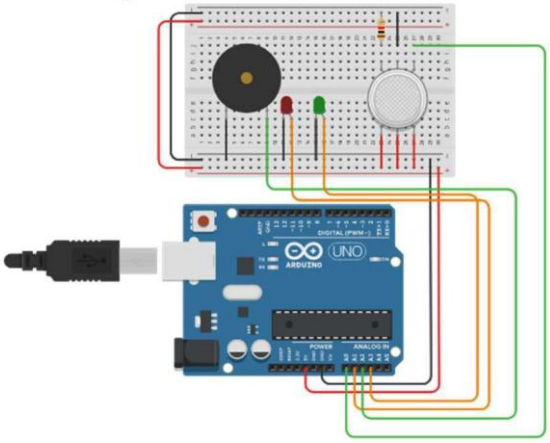


Fig.03.13.2: Circuit Diagram of gas leak detector with alarm.

Experimental Setup:

Fig.03.13.3: Experimental setup of gas leak detector with alarm.

Code:

```
int LED = A1;
int LED1 = A3;
int gas_pin = A0;
int buzzer_pin = A2;
void setup()
{
  Serial.begin(9600);
  pinMode (buzzer_pin, OUTPUT);
  pinMode (gas_pin, INPUT);
}
void loop() {
      float sensorValue,gas_pin;
      sensorValue =
analogRead(gas_pin); // read analog
input pin 0
  if(sensorValue >= 300)
  {
    digitalWrite(LED,HIGH);
    digitalWrite(LED1,LOW);
digitalWrite (buzzer_pin, HIGH);
    //Serial.println();
    Serial.print(sensorValue);
    Serial.println(" |SMOKE
DETECTED|");
  }
  else
  {
     digitalWrite(LED,LOW);
    digitalWrite(LED1,HIGH);
    digitalWrite (buzzer_pin, LOW);
    Serial.println();
    Serial.println("Sensor Value: ");
    Serial.print(sensorValue);
    //Serial.print(" |Safe Mode|");
  }
  delay(1000);
}
```

Discussions:

This project's main purpose was to create gas leak detector. It's essential in smart homes. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 14

Project Name: Water Level Indicator with LEDs & LCD

Overview:

This project uses multiple float switches or analog water level sensors to detect different levels of water in a tank. It then lights up LEDs corresponding to each level and also displays the water level numerically on a 16x2 I2C LCD. It's a practical implementation for water management systems

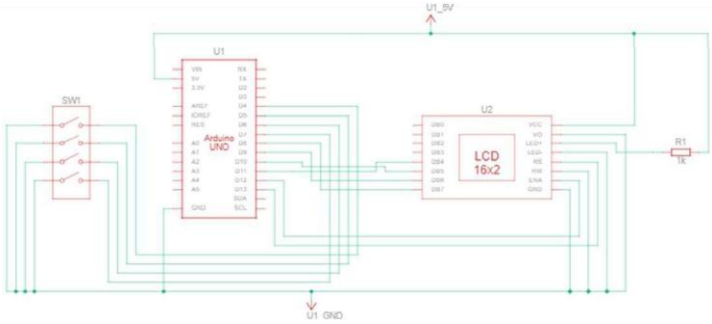


Fig.03.14.1: Schematic Diagram of Water Level Indicator with LEDs & LCD

Required Apparatus:

Table 3.14.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Water Level Sensor	Operating voltage: DC3_5V	1
3	I2C LCD Display	16x2	1
4	LED	--	1
5	Breadboard	400 tie-points	1
6	Jumper Wires	Male-to-Male, 20 cm	Several
7	Resistor	1k Ohm	1

Circuit Diagram:

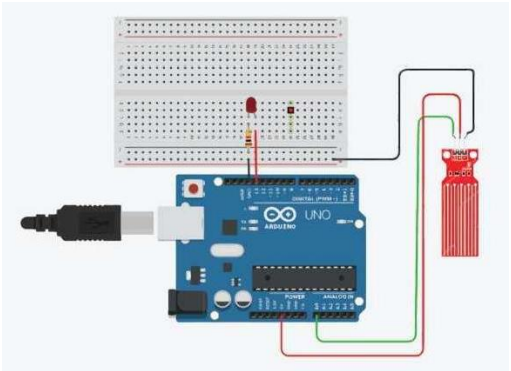


Fig.03.14.2: Circuit Diagram of Water Level Indicator with LEDs & LCD(Laptop)

Experimental Setup:

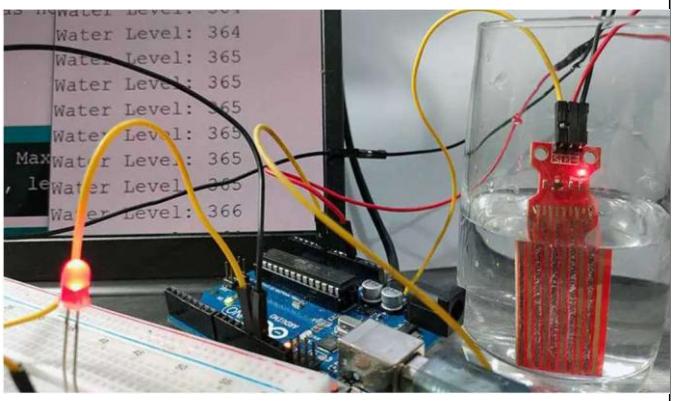


Fig.03.14.3: Experimental setup of Water Level Indicator with LEDs & LCD

Code:

```
int waterSensorPin = A0;
int ledPin = 13;

void setup() {
    Serial.begin(9600);
    pinMode(ledPin, OUTPUT);
}
void loop() {
  int sensorValue =
analogRead(waterSensorPin);
  Serial.print("Water Level: ");
  Serial.println(sensorValue);

  if (sensorValue > 300) {
     digitalWrite(ledPin, HIGH);
} else {
    digitalWrite(ledPin, LOW);
  }
  delay(1000);
}
```

Discussions:

This project's main purpose was to create a water level indicator. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 15

Project Name: IR Remote Controlled LED System

Overview:

This project uses an IR remote to wirelessly control LEDs via an IR receiver and Arduino. It demonstrates remote signal decoding and digital output control for basic home automation.

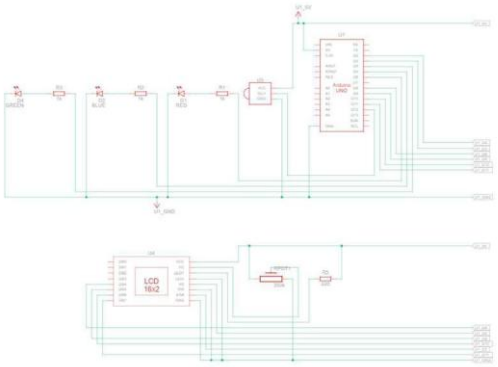


Fig.03.15.1: Schematic Diagram of IR Remote Controlled LED System

Required Apparatus:

y

Table 3.15.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantit
1	Arduino Uno	ATmega328P Microcontroller	1
2	IR Remote	--	1
3	I2C LCD Display	16x2	1
4	Breadboard	400 tie-points	2
5	Jumper Wires and LEDs	--	Several
6	POT	250k Ohm	1

Circuit Diagram:

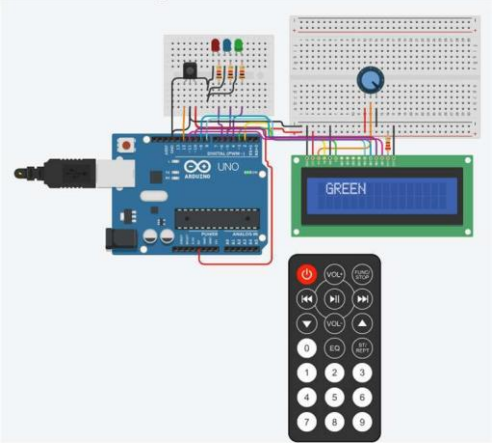


Fig.03.15.2: Circuit Diagram of IR Remote Controlled LED System

Experimental Setup:

Fig.03.15.3: Experimental setup of IR Remote Controlled LED System

Code:

```
#include <IRremote h>
//Define Pins
int redLed = 5;
int yellowLed = 4;
int greenLed = 3;
int blueLed = 2;
int RECV_PIN = 11;
//IR Library stuff
IRrecv irrecv(RECV_PIN);
decode_results results;


void setup()
{
  //Set Led Pins
  pinMode(redLed, OUTPUT);
  pinMode(yellowLed, OUTPUT);
  pinMode(greenLed, OUTPUT);
  pinMode(blueLed, OUTPUT);

  //Enable serial usage and IR
signal in
  Serial.begin(9600);
  Serial.println("Enabling
IRin");
  irrecv.enableIRIn();
  Serial.println("Enabled
IRin");
}
void loop() {
  if (irrecv.decode(&results)) {
unsigned int value = results.value;
Serial.println(value);
    switch (value) {
      case 2295:
            digitalWrite(redLed, HIGH);
            delay(500);
            digitalWrite(redLed, LOW);
            break;

      case 34935:
            digitalWrite(yellowLed, HIGH);
            delay(500);
            digitalWrite(yellowLed, LOW);
            break;

      case 18615:
            digitalWrite(greenLed, HIGH);
            delay(500);
            digitalWrite(greenLed, LOW);
            break;

      case 10455:
            digitalWrite(blueLed, HIGH);
            delay(500);
            digitalWrite(blueLed, LOW);
    }


    irrecv.resume(); // Receive the next
value
  }
}
```

Discussions:

This project's main purpose was to create a IR remote control LED. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 16

Project Name: Light Level Meter with LDR and LCD.

Overview:

This project measures ambient light using an LDR (Light Dependent Resistor) and displays the intensity on an LCD. It helps in understanding analog sensor input and real-time data visualization with Arduino.

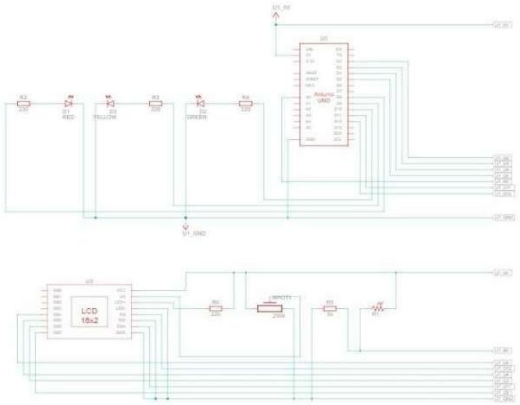


Fig.03.16.1: Schematic Diagram of Light Level Meter with LDR and LCD.

Required Apparatus:

Table 3.16.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	LDR	5v	1
3	I2C LCD Display	16x2	1
4	Breadboard	400 tie-points	2
5	Jumper Wires	Male-to-Male, 20 cm	Several

Circuit Diagram:

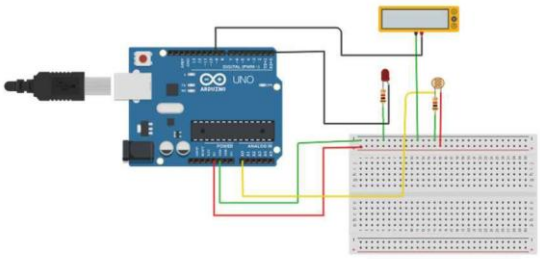


Fig.03.16.2: Circuit Diagram of Light Level Meter with LDR and LCD.

Experimental Setup:

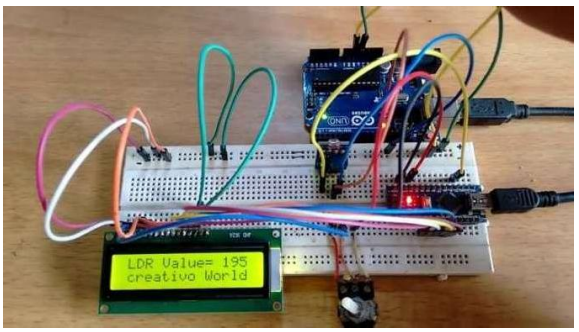


Fig.03.16.3: Experimental setup of Light Level Meter with LDR and LCD.

Code:

```
int sensorValue = 0;
void setup()
{
pinMode(A0, INPUT);
Serial.begin(9600);
pinMode(9, OUTPUT);
}
void loop()
{
sensorValue = analogRead(A0);
Serial.println(sensorValue);
analogWrite(9, map(sensorValue,0,1023,0,255));
delay(1000);
}
```

Discussions:

This project's main purpose was to create Light level meter with LDR and LCD. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 17

Project Name: Real-Time Analog Value Display

Overview:

Reads analog input (from a potentiometer or LDR) and displays the values in real-time using either Serial Plotter or an LCD

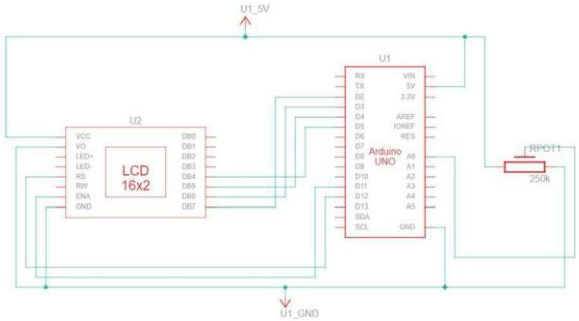


Fig.03.17.1: Schematic diagram of Real-Time Analog Value Display

Required Apparatus:

Table 3.17.1: Table for Required Apparatus

Sl. No.	Components	Specification	Quantit
1	Arduino Uno	ATmega328P Microcontroller	1 y
2	POT (Potentiometer)	250K	1
3	I2C LCD Display	16x2	1
4	Breadboard	400 tie-points	1
5	Jumper Wires	Male-to-Male, 20 cm	Several

Circuit Diagram:

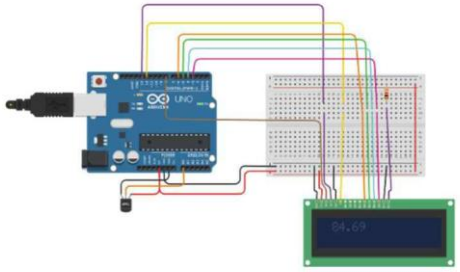


Fig.03.17.2: Circuit diagram of Real-Time Analog Value Display

Experimental Setup:

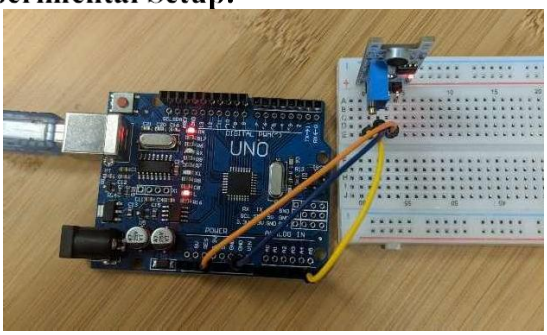


Fig.03.17.3: Experimental setup of Real-Time Analog Value Display

Code:

```
#include <LiquidCrystal h>
int value;
//LiquidCrystal lcd(RS, Enable,
DB4, DB5, DB6, DB7)
LiquidCrystal lcd(12, 11,5, 4,
3, 2);
void setup(){
  lcd.begin(16, 2);
  pinMode(A0, INPUT);
  Serial.begin(9600);
}
void loop(){
  lcd.print("Sensor Value");
  value = analogRead(A0);
  lcd.setCursor(0, 1);
  lcd.print(value);
  delay(1000);
  lcd.clear();
  Serial.println(value);
}
```

Discussions:

This project's main purpose was to create a Real time analog value display. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 18

Project Name: Two-Button Digital Lock

Overview:

This simple digital lock uses two buttons to input a combination.
access is granted and shown on an LED.

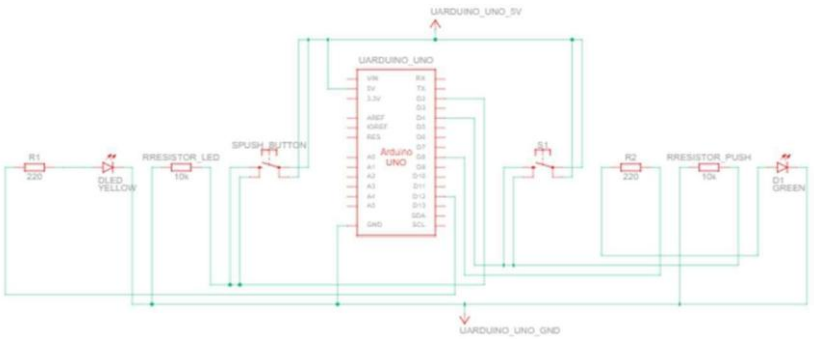


Fig.03.18.1: Schematic Diagram of Two-Button Digital Lock

Required Apparatus:

Table 3.18.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	Push Buttons	--	2
3	Resistors	10k ohm	2
4	Resistors	220 ohm	2
5	Breadboard	400 tie-points	1
6	LED	5mm	2
7	Jumper Wires	Male-to-Male, 20 cm	Several

Code:

```
int buttonState = 0;
void setup()
{
  pinMode(2, INPUT);
  pinMode(8, OUTPUT);
  pinMode(4, INPUT);
  pinMode(12, OUTPUT);
}
void loop()
{
  buttonState = digitalRead(2);
  if (buttonState == HIGH)
  {
    digitalWrite(8, HIGH);
  }
  else
  {

    digitalWrite(8, LOW);
  }
  buttonState = digitalRead(4);
  if (buttonState == HIGH)
  {
    digitalWrite(12, HIGH);
  }
  else
  {
    digitalWrite(12, LOW);
  }
  delay(20);
 }
```

Circuit Diagram:

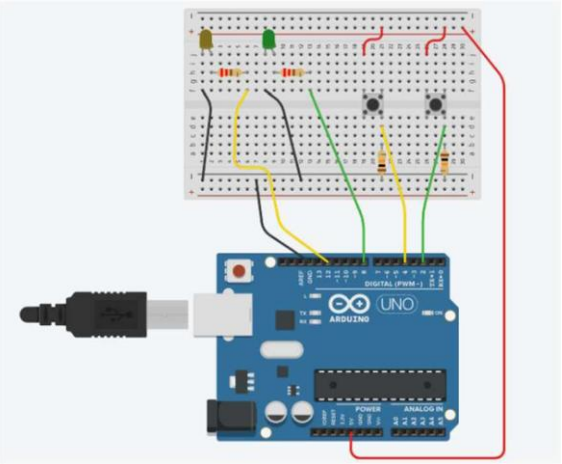


Fig.03.18.2: Circuit Diagram of Two-Button Digital Lock

Experimental Setup:

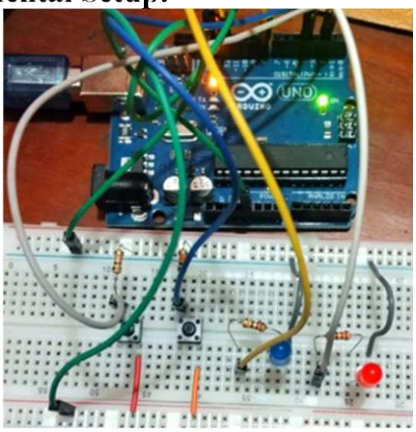


Fig.03.18.3: Experimental setup of the Two-Button Digital Lock

Discussions:

This project's main purpose was to create a 2 button digital clock. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 19

Project Name: Automatic Night Lamp with LDR

Overview:

This project uses an LDR to detect darkness and automatically switches on an LED. It functions as a basic night lamp system.

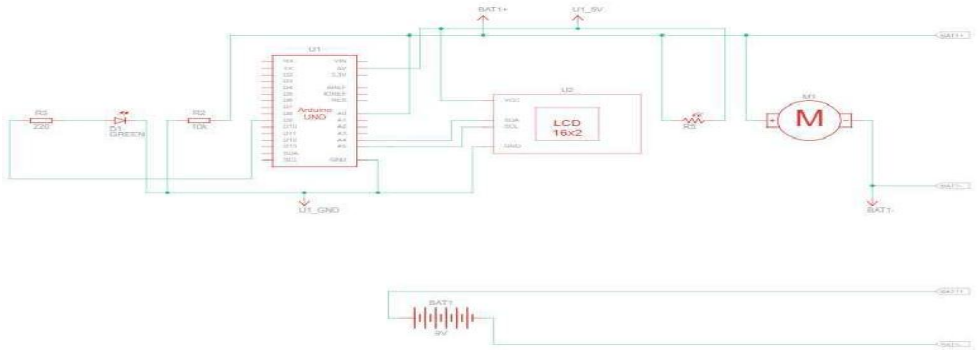


Fig.03.19.1: Schematic Diagram of Automatic Night Lamp with LDR

Required Apparatus:

Table 3.19.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1	Arduino Uno	ATmega328P Microcontroller	1
2	LDR	5V	1
3	LED	5mm	1
4	Resistor	10k ohm	1
5	Resistor	220 ohm	1
6	Breadboard	400 tie-points	1
7	Jumper Wires	Male-to-Male, 20 cm	Several

Code:

```
//2101130
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

// Use correct address (replace
0x27 if needed)
LiquidCrystal_I2C lcd(0x27, 16,
2);
const int ldrPin = A0;
const int ledPin = 9;
void setup() {
```

Circuit Diagram:

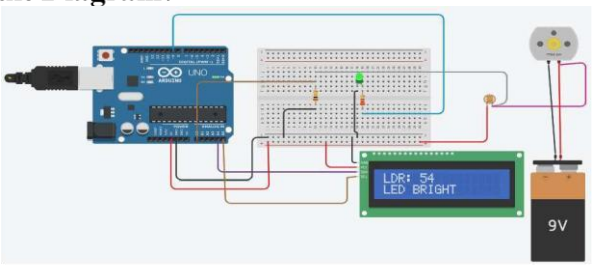
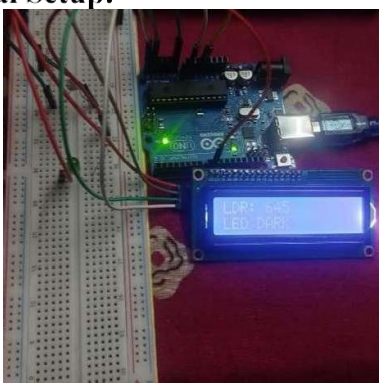


Fig.03.19.2: Circuit Diagram of Automatic Night Lamp with LDR

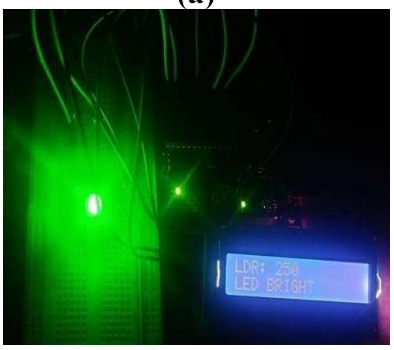
```
pinMode(ledPin, OUTPUT);
  lcd.init();
  instead of lcd.begin()
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Initializing...");
  delay(2000);
  lcd.clear();
}
void loop() {
  int ldrValue =
analogRead(ldrPin);

  lcd.setCursor(0, 0);
  lcd.print("LDR: ");
  lcd.print(ldrValue);
  lcd.print("    ");
  lcd.setCursor(0, 1);
  if (ldrValue < 500) {
    digitalWrite(ledPin, HIGH);
    lcd.print("LED
BRIGHT      ");
  } else {
    digitalWrite(ledPin, LOW);
    lcd.print("LED DARK    ");
  }
  delay(500);
}
```

Experimental Setup:



(a)



(b)

Fig.03.19.3: Experimental setup of the Automatic Night Lamp with LDR (a) Night Lamp off ,(b) Night Lamp on

Discussions:

This project's main purpose was to create a automatic night lamp with LDR. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.

Project No.: 20

Project Name: Temperature Controlled Fan

Overview:

This project uses a temperature sensor to monitor the room temperature and automatically turn on a fan (DC motor or relay) when the temperature exceeds a set threshold. It's useful for learning environmental control systems.

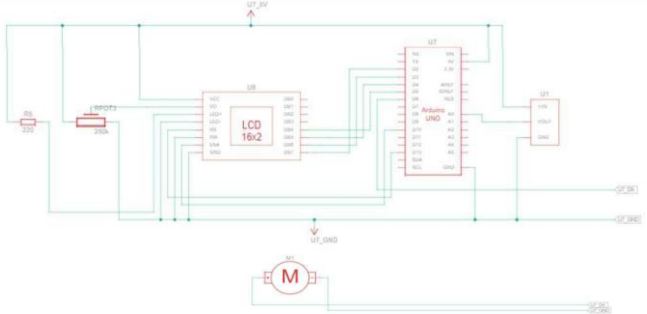


Fig.03.20.1: Schematic Diagram of Temperature Controlled Fan

Required Apparatus:

Table 3.20.1: Table For Required Apparatus

Sl. No.	Components	Specification	Quantity
1.	Arduino Uno	ATmega328P Microcontroller	1
2.	DHT11 Sensor	0°C to 50°C	1
3.	DC Motor	3V – 6V	1
4.	Relay Module	5V	1
5.	Breadboard	400 tie-points	1
6.	jumper wires	Male-to-Male, 20 cm	10

Code:

```
const int sensorPin = A0;
const int fanPin = 9;
float temperature;
int threshold = 30;
void setup() {
  pinMode(fanPin, OUTPUT);
}
void loop() {
  int sensorValue =
analogRead(sensorPin);
  float voltage = sensorValue *
(5.0 / 1023.0);

  temperature = voltage * 100;
  if (temperature >= threshold)
{
    digitalWrite(fanPin, HIGH);
  } else {
    digitalWrite(fanPin, LOW);
  }
  delay(500);
}
```

Circuit Diagram:

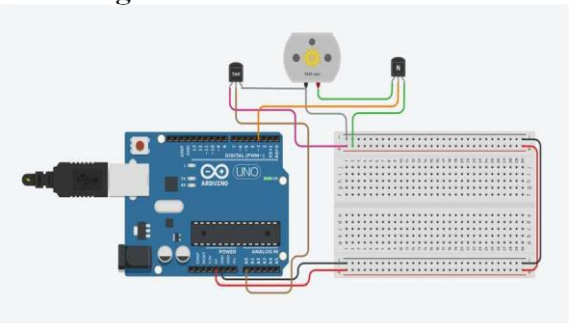


Fig. 3.20.2: Temperature Controlled Fan

Experimental Setup:

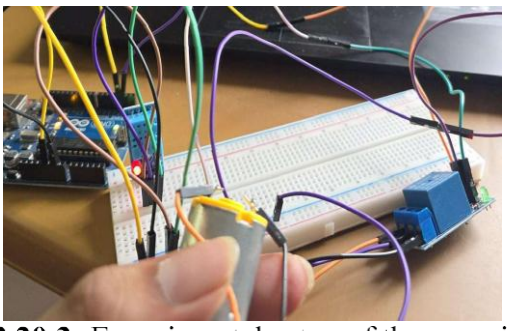


Fig. 3.20.3: Experimental setup of the experiment

Discussions:

This project's main purpose was to create a temperature controlled fan. It's essential in many works. Though some problems occurred due to faulty wirings. All problems were overcome by Proper and careful wirings and arrangement.