Faculty of Engineering and TechnologyElectrical and

Computer Engineering Department

ENCS4380 - interface

Design and Implementation of a Distance-BasedWarning System using Arduino

Students names :

Std1 : Jana sawalmeh - 1212467

Std2 : Asma'a Shejaeya - 1210084

Std3 : Doaa Hatu - 1211088

Std4 : Rebal zabade - 1210162

1 Introduction

In this report, we will design, implement, and explain a Distance-Based Warning System using an Arduino microcontroller. The system will employ an ultrasonic sensor to measure distances to nearby objects and provide feedback via LEDs, dynamically adjusting the thresholds based on potentiometer input.

2 Components

The following components are used in the project:

- Arduino Uno
- HC-SR04 Ultrasonic Sensor
- Potentiometer (10k)
- Green LED
- Yellow LED
- Red LED
- Resistors (220)
- Breadboard and Jumper Wires
- Power Supply (USB or battery)

3 System Design

3.1 Circuit Design using TinkerCad

The circuit consists of connecting the ultrasonic sensor and LEDs to the Arduino. The potentiometer will control the distance ranges. A basic schematic diagram is shown in Figure , which you can create using Tinkercad or drawing software.

3.2 Operation Principles

The ultrasonic sensor sends out a sound pulse and measures the time it takes for the echo to return. The distance to the object is calculated using the formula:

$$\text{Distance} = \frac{\text{Time} \times \text{Speed of Sound}}{2}$$

Based on the measured distance, the system activates the corresponding LED:

- **Green LED:** Distance $>$ 20 cm

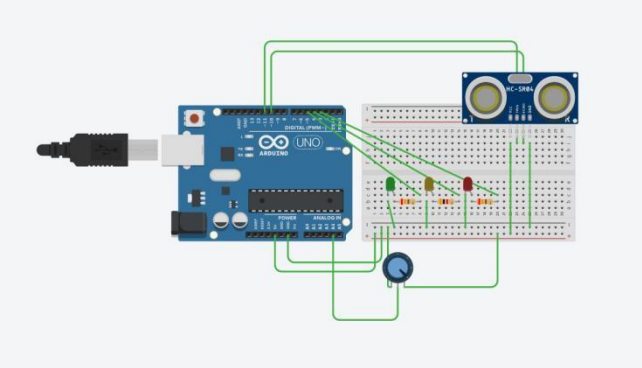

Figure 1: Circuit Diagram of the Distance-Based Warning System

- **Yellow LED:** 10 cm $<$ Distance $\leq$ 20 cm
- **Red LED:** Distance $\leq$ 10 cm

4 Implementation

The physical circuit is built on a breadboard following the schematic. The Arduino code is implemented to read the ultrasonic sensor values, process potentiometer input, and control the LEDs accordingly.

4.1 Arduino Code

Here is the sample Arduino code for the project:

```
// jana sawalmeh , rebal Zabzda , assma Fares , Doaa Hatu
// Pin assignments
const int trigPin = 11;
const int echoPin = 10;
const int xPin = A4; // Joystick X-axis pin
```

```
const int greenLEDPin = 5;
const int yellowLEDPin = 4;
const int redLEDPin = 3;

// Distance thresholds (modifiable by joystick)
int greenThreshold = 20; // cm
int yellowThreshold = 10; // cm
void setup() {
  // Initialize pins
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(greenLEDPin, OUTPUT);
```

```
  pinMode(yellowLEDPin, OUTPUT);
  pinMode(redLEDPin, OUTPUT);

  // Begin Serial communication for debugging
  Serial.begin(9600);
}

void loop() {
  // Read distance from the ultrasonic sensor
  int distance = getDistance();

  // Adjust distance thresholds based on joystick
  adjustThresholds();

  // Display distance and thresholds for debugging
  Serial.print("Distance:-");
  Serial.print(distance);
  Serial.print("-cm-|-Green-Threshold:-");
  Serial.print(greenThreshold);
  Serial.print("-cm-|-Yellow-Threshold:-");
  Serial.print(yellowThreshold);
  Serial.println("-cm");

  // Control LEDs based on distance and thresholds
  if (distance > greenThreshold) {
    setLEDs(HIGH, LOW, LOW); // Green LED on
  } else if (distance > yellowThreshold) {
    setLEDs(LOW, HIGH, LOW); // Yellow LED on
  } else {
    setLEDs(LOW, LOW, HIGH); // Red LED on
  }

  // Debugging to check which LED is supposed to be on
```

```
  Serial.print("Green-LED-State:-");
  Serial.print((distance > greenThreshold) ? "ON" : "OFF");
  Serial.print("-|-Yellow-LED-State:-");
  Serial.print((distance > yellowThreshold && distance <=
      greenThreshold) ? "ON" : "OFF");
  Serial.print("-|-Red-LED-State:-");
  Serial.println((distance <= yellowThreshold) ? "ON" : "OFF")
      ;

  delay(100); // Short delay for stability
}

// Function to get distance from ultrasonic sensor
int getDistance() {
  // Send a pulse to trigger the sensor
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
```

```
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // Measure the time of the echo pulse
  long duration = pulseIn(echoPin, HIGH);

  // Calculate distance in cm (speed of sound = 34300 cm/s)
  int distance = duration * 0.034 / 2;
  return distance;
}

// Function to adjust distance thresholds based on joystick
    position
void adjustThresholds() {
  int xValue = analogRead(xPin);

  // Map joystick X value to a range of -2 to +2 (for
      sensitivity)
  int adjustment = map(xValue, 0, 1023, -2, 2);

  // Adjust thresholds within a safe range
  greenThreshold = constrain(greenThreshold + adjustment, 20,
      20);
  yellowThreshold = constrain(yellowThreshold + adjustment,
      10, 10);
}

// Function to control LED states
void setLEDs(bool greenState, bool yellowState, bool redState)
    {
  digitalWrite(greenLEDPin, greenState);
  digitalWrite(yellowLEDPin, yellowState);d
```

```
  igitalWrite(redLEDPin, redState);
}
```

4.2 Arduino implementation

the componats that was implemented using tinkercad we implement it using an actual componants in ka3kash lab here the result and we will upload a video for results

5 Discussion

In this section, discuss the specifications of the components used, the challenges faced during the circuit design and implementation, and how the system can be improved.

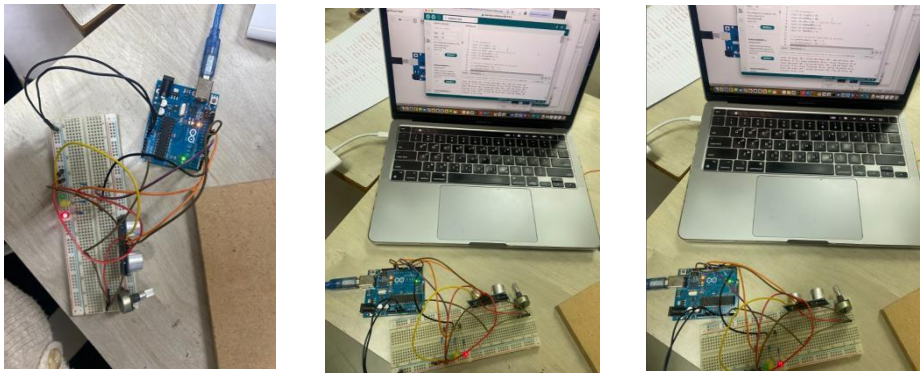

Figure 2: Actual implementation for Tinkercad circuit

6 Conclusion

The Distance-Based Warning System effectively measures distance and provides visual feedback using LEDs based on preset thresholds. The incorporation of a potentiometer for dynamic adjustment showcases the flexibility of the Arduino platform for real-world applications.

7 References

- Arduino Documentation: https://www.arduino.cc/
- HC-SR04 Ultrasonic Sensor Datasheet
- Potentiometer Specifications