**Parking system alert Arduino buzzers with ultrasonic -TickerCad Circuit(IOT)**

The goal of this project is to design a **parking alert system** that uses an ultrasonic sensor to detect the distance of an obstacle, such as a wall or another vehicle, and activates a buzzer when the vehicle gets too close. The system can be easily implemented using a microcontroller such as an **Arduino Uno**, and can be tested using **Tinkercad Circuits**, a free, web-based simulation tool.

The core of this system includes the following components:

* **Arduino Uno**: The brain of the system, which reads data from the ultrasonic sensor and controls the buzzer.
* **Ultrasonic Sensor (HC-SR04)**: Measures the distance between the sensor and an obstacle using sound waves.
* **Buzzer**: Produces a sound to alert the driver when the vehicle approaches too close to an obstacle.

### 2. ****Components Used****

* **Arduino Uno**: Acts as the main controller for the system.
* **HC-SR04 Ultrasonic Sensor**: Measures the distance of an object using ultrasonic waves.
  + **Trigger Pin (Trig)**: Used to trigger the ultrasonic pulse.
  + **Echo Pin (Echo)**: Used to receive the reflected pulse and calculate the distance.
* **Passive Buzzer**: Generates an audible alert when the distance is below a certain threshold.
* **Breadboard**: For wiring and connecting components.
* **Jumper Wires**: To establish connections between components.

### 3. ****Working Principle****

The system operates on the **principle of echo-location**, similar to how bats navigate in the dark. The ultrasonic sensor sends out high-frequency sound waves that bounce off nearby objects. The time it takes for the echo to return is measured, and using this information, the distance to the object is calculated. If the measured distance falls below a certain threshold (e.g., 10 cm), the system activates the buzzer to alert the user.

The **ultrasonic sensor** works as follows:

1. The **Trigger Pin** sends a high pulse (sound wave) for 10 microseconds.
2. The sound travels until it hits an object and is reflected back to the sensor.
3. The **Echo Pin** listens for the returning sound wave.
4. The Arduino calculates the distance based on the time it took for the sound to return.

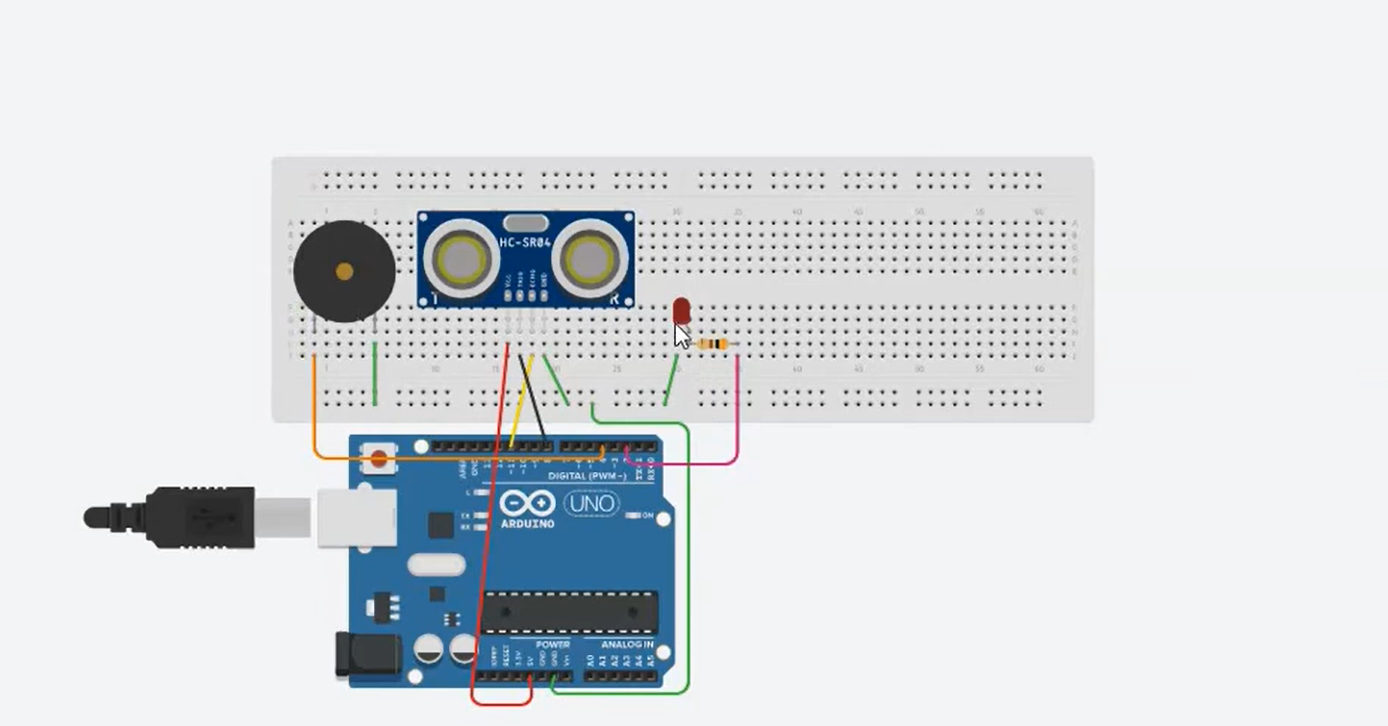
When an object is closer than the threshold distance, the **buzzer** is activated, generating a warning sound.

### 4. ****Circuit Diagram and Connections****

In the **Tinkercad Circuits** environment, the components are connected as follows:

1. **Ultrasonic Sensor (HC-SR04)**:
   * **VCC** pin to Arduino **5V**.
   * **GND** pin to Arduino **GND**.
   * **Trig** pin to Arduino **Pin 9**.
   * **Echo** pin to Arduino **Pin 10**.
2. **Buzzer**:
   * **Positive terminal (Anode)** of the buzzer to Arduino **Pin 8**.
   * **Negative terminal (Cathode)** of the buzzer to **GND**.
3. **LED (optional)**:
   * Connect the **anode** of the LED to **Pin 7** through a current-limiting resistor (e.g., 220 ohms).
   * Connect the **cathode** of the LED to **GND**.

The **Arduino Uno** reads the distance data from the ultrasonic sensor and controls the buzzer depending on the calculated distance.



### 5. ****Code Explanation****

The system uses a simple Arduino program to operate. Here's a breakdown of the program:

#### Arduino Code for Parking Alert System

cpp

Copy code

// Pin definitions

const int trigPin = 9; // Pin connected to Trig of HC-SR04

const int echoPin = 10; // Pin connected to Echo of HC-SR04

const int buzzerPin = 8; // Pin connected to the buzzer

const int ledPin = 7; // Pin connected to the LED

// Variables for calculating distance

long duration;

int distance;

void setup() {

// Initialize serial communication at 9600 bits per second

Serial.begin(9600);

// Set pin modes

pinMode(trigPin, OUTPUT); // Trig is an output pin

pinMode(echoPin, INPUT); // Echo is an input pin

pinMode(buzzerPin, OUTPUT); // Buzzer is an output pin

pinMode(ledPin, OUTPUT); // LED is an output pin

// Initial states

digitalWrite(ledPin, LOW); // LED off initially

}

void loop() {

// Clear the trigPin by setting it LOW

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Trigger the sensor by sending a 10us HIGH pulse

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Read the echoPin, and measure the time it takes for the pulse to return (in microseconds)

duration = pulseIn(echoPin, HIGH);

// Calculate the distance in cm

distance = duration \* 0.034 / 2;

// Print the distance to the Serial Monitor

Serial.print("Distance: ");

Serial.print(distance);

Serial.println(" cm");

// Check if the object is within 10 cm

if (distance < 10 && distance > 0) {

// Object is close, activate buzzer and LED

tone(buzzerPin, 1000); // Generate a 1kHz sound (adjust frequency as needed)

digitalWrite(ledPin, HIGH); // Turn on the LED

} else {

// Object is far, deactivate buzzer and LED

noTone(buzzerPin); // Stop the tone

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

}

// Wait a little before taking the next reading

delay(200); // 200ms delay

}

### Code Breakdown:

1. **Pins Setup**:
   * The ultrasonic sensor’s trigPin and echoPin are defined and set up.
   * The buzzer is connected to buzzerPin, and the LED is connected to ledPin.
2. **Distance Measurement**:
   * The **trigger pin** sends a 10-microsecond pulse to generate the ultrasonic wave.
   * The **echo pin** listens for the reflected wave and calculates the time it took to return.
   * Using the time difference, the distance is calculated.
3. **Buzzer and LED Activation**:
   * If the distance is less than 10 cm, the buzzer generates a tone using the tone() function (1 kHz in this example), and the LED turns on.
   * If the distance is greater than 10 cm, the buzzer and LED are turned off.

### 6. ****How It Works in Tinkercad****

Once you upload the code to the Arduino in Tinkercad and start the simulation:

1. **Ultrasonic Sensor**: Detects the distance of an object in front of it.
2. **Buzzer**: Emits a sound when the object is closer than 10 cm.
3. **Serial Monitor**: Displays the real-time distance between the sensor and the object.
4. **LED (Optional)**: Can be added as a visual alert when the object is too close.

The system continuously measures the distance, updates the Serial Monitor, and alerts the user with a sound when the object enters the danger zone.

### 7. ****Conclusion****

The parking alert system using Arduino, an ultrasonic sensor, and a buzzer is a simple but effective system to help prevent collisions in tight parking spaces. It provides a practical application of sensors and microcontroller programming in real-world scenarios. Using **Tinkercad Circuits** for simulation makes it easy to design, test, and visualize the system before implementing it in hardware.

This project demonstrates the basic principles of distance measurement using ultrasonic waves and sound-based alert systems, providing a foundation for more advanced vehicle safety systems.