Distance Measurement & Object Detection using Ultrasonic Sensor with Raspberry Pi Pico W

TEAM MEMBERS

- **ALDRICH MARIAN A**
- **BARATH M**
- **ASHISH C**
- **AKASH SAI**

AIM:

To design and develop a distance measurement and object detection system using an ultrasonic sensor with Raspberry Pi Pico W. The system measures the distance between the sensor and an object in real time and triggers alerts when objects are within a specified range.

COMPONENTS REQUIRED:

- Raspberry Pi Pico W
- Ultrasonic Sensor (HC-SR04)
- Buzzer
- LED
- Resistors
- Breadboard

Raspberry Pi Pico W:

A compact microcontroller with wireless connectivity that supports real-time data processing. It is used to trigger the ultrasonic sensor and interpret distance readings for object detection.

Ultrasonic Sensor (HC-SR04):

An ultrasonic sensor that sends high-frequency sound waves and measures the time it takes for the echo to return. It calculates the distance by using the speed of sound, helping detect objects in front of the sensor.

Buzzer:

An output device that emits sound alerts when objects are detected within a predefined range. It helps warn users of nearby obstacles.

LED:

A light-emitting diode that visually signals when an object is within the danger zone. It can be used in conjunction with the buzzer for better awareness.

Resistors:

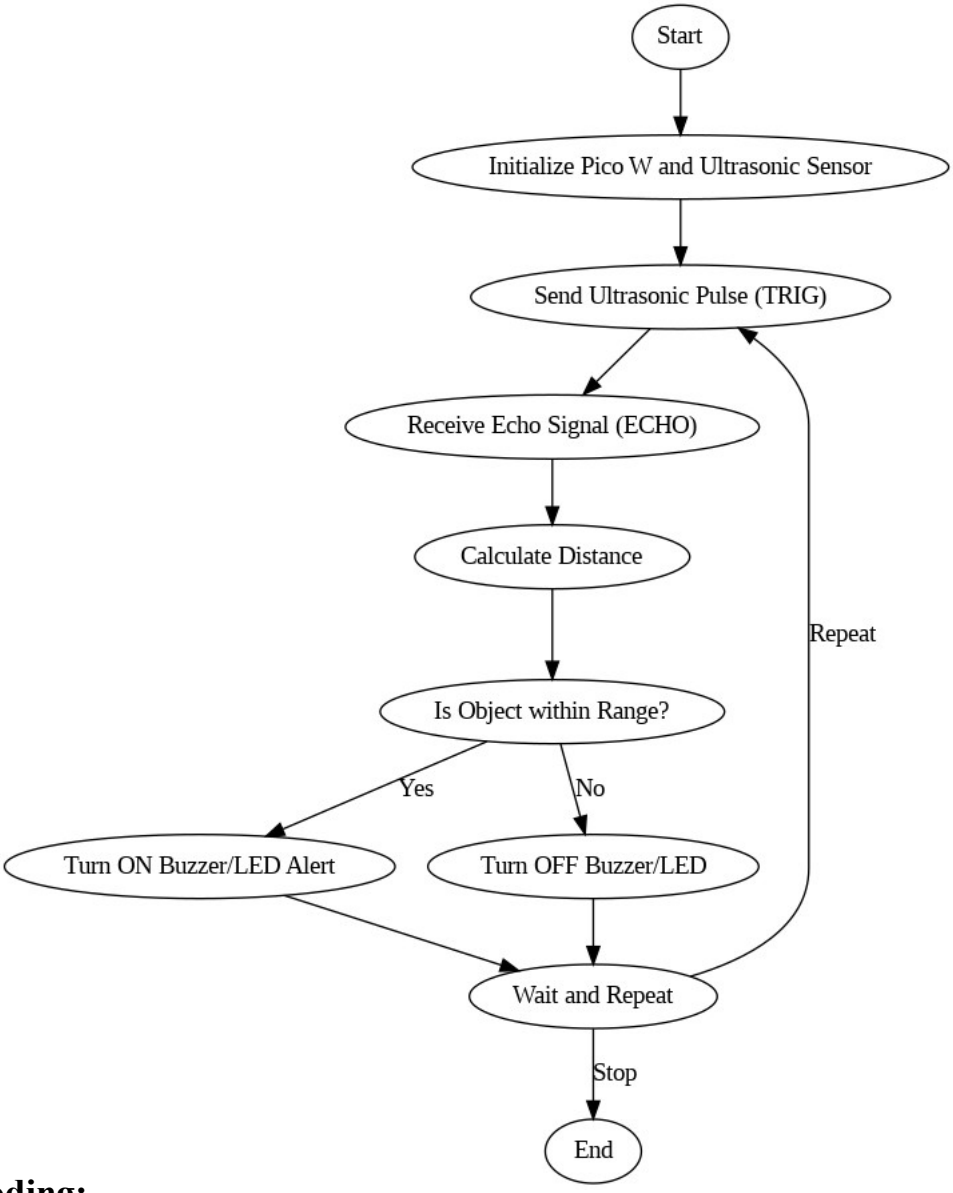
Resistors limit the flow of electric current in a circuit to protect components. They ensure proper operation by controlling voltage and current levels.

Pin Table:

Component	Pin on Component	Raspberry Pi Pico W Pin	Description
Ultrasonic Sensor	VCC	3.3V or 5V	Power supply to the sensor
Ultrasonic Sensor	GND	GND	Ground connection
Ultrasonic Sensor	TRIG	GPIO 15	Sends ultrasonic pulse
Ultrasonic Sensor	ECHO	GPIO 14	Receives echo signal
Buzzer (optional)	+	GPIO 13	Audible alert output
Buzzer (optional)	–	GND	Ground connection
LED (optional)	+	GPIO 12	Visual alert output

Circuit Connection:

Flowchart:



Coding:

```
from machine import Pin, time_pulse_us
import time
# Define pins
TRIG = Pin(15, Pin.OUT)
ECHO = Pin(14, Pin.IN)
BUZZER = Pin(13, Pin.OUT)
LED = Pin(12, Pin.OUT)

# Threshold distance in cm
DIST_THRESHOLD = 10

def measure_distance():
    # Send pulse
    TRIG.low()
    time.sleep_us(2)
    TRIG.high()
    time.sleep_us(10)
    TRIG.low()

    # Measure echo time
    try:
        pulse_time = time_pulse_us(ECHO, 1, 30000)  # Timeout 30ms
    except OSError as ex:
        print("Timeout!")
        return None

    # Calculate distance (speed of sound: 34300 cm/s)
    distance = (pulse_time / 2) / 29.1
```

```
    return distance

def main():
    print("Starting Distance Measurement System...")
    while True:
        distance = measure_distance()

        if distance is not None:
            print("Distance: {:.2f} cm".format(distance))

            if distance < DIST_THRESHOLD:
                print("Object Detected!")
                BUZZER.high()
                LED.high()
            else:
                BUZZER.low()
                LED.low()
        else:
            print("Failed to read distance")
            BUZZER.low()
            LED.low()

        time.sleep(1)

# Run the main function
if __name__ == "__main__":
    main()
```

Execution: