DISTANCE MEASUREMENT AND OBJECT DETECTION USING ULTRASONIC SENSORS WITH RASPBERRY PI PICO

TEAM MEMBERS:

- ***JOSEPH RIJAN A***
- ***INDHUCHUDAN V S***
- ***MAHESH A***
- ***POORNESH V***

AIM:

To measure the distance and detect objects using ultrasonic sensors with Raspberry Pi Pico W.

TOOLS/HARDWARE REQUIRED:

- Raspberry Pi Pico W
- Ultrasonic Sensor HC-SR04
- Indicator LEDs
- Buzzer
- Jumper Wires
- MicroPython

THEORY:

A distance measuring system is used to measure the distance between two objects by transmitting high-frequency waves and calculating the time taken for the echo to return. In this project,

- **Raspberry Pi**
  - A small, affordable, and highly capable single-board computer.
  - Can run a full-fledged operating system and support various programming languages.
- **Ultrasonic Sensor (e.g., HC-SR04)**
  - A sensor that uses high-frequency sound waves to measure distance.
  - Can detect objects and calculate their distance from the sensor.
  - The sensor has TRIG and ECHO pins: TRIG sends the signal, and ECHO receives the reflected pulse.
  - Range varies between 2cm – 4m.

To calculate the distance, we use the following formula:

Distance = Time x 0.034/2 = 0.017m/s

The system can detect objects within a certain range and alert the user if an obstacle is too close. Applications include obstacle avoidance, smart parking systems, and automation.

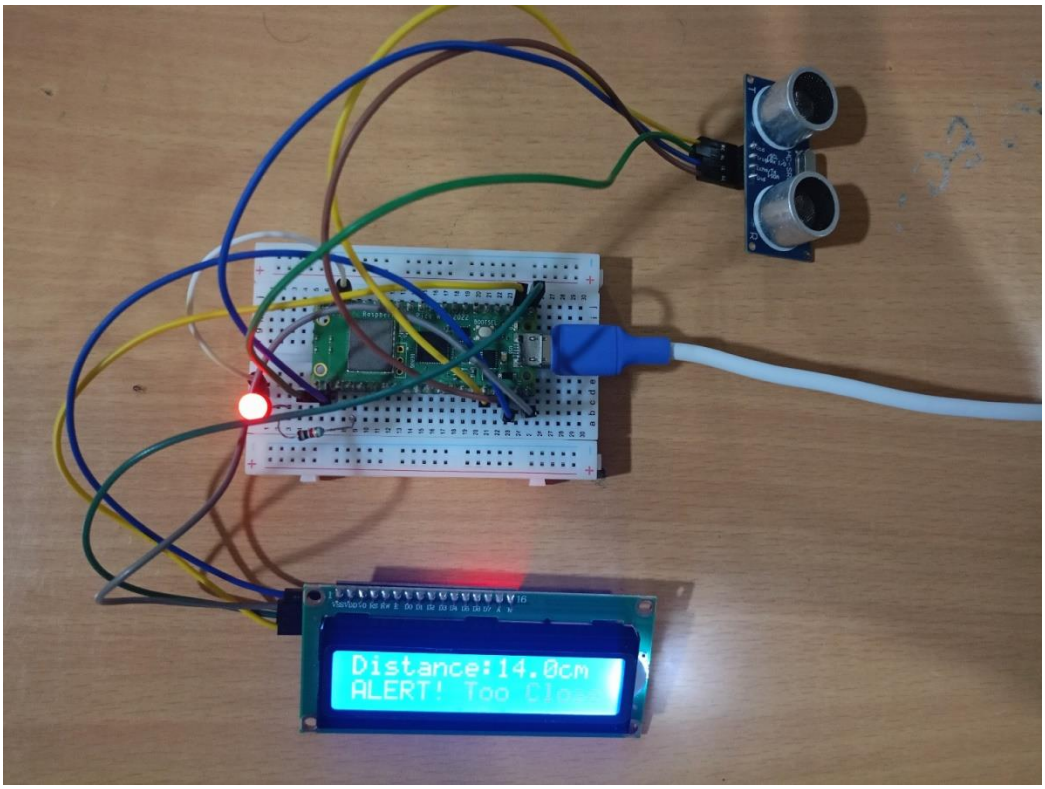
PIN CONNECTIONS:

Component	Pico W Pin
HC-SR04 VCC	VBus
HC-SR04 GND	GND
HC-SR04 ECHO	Pin 16
HC-SR04 TRIG	Pin 17
LCD SDA	GPIO 0
LCD SCL	GPIO 1

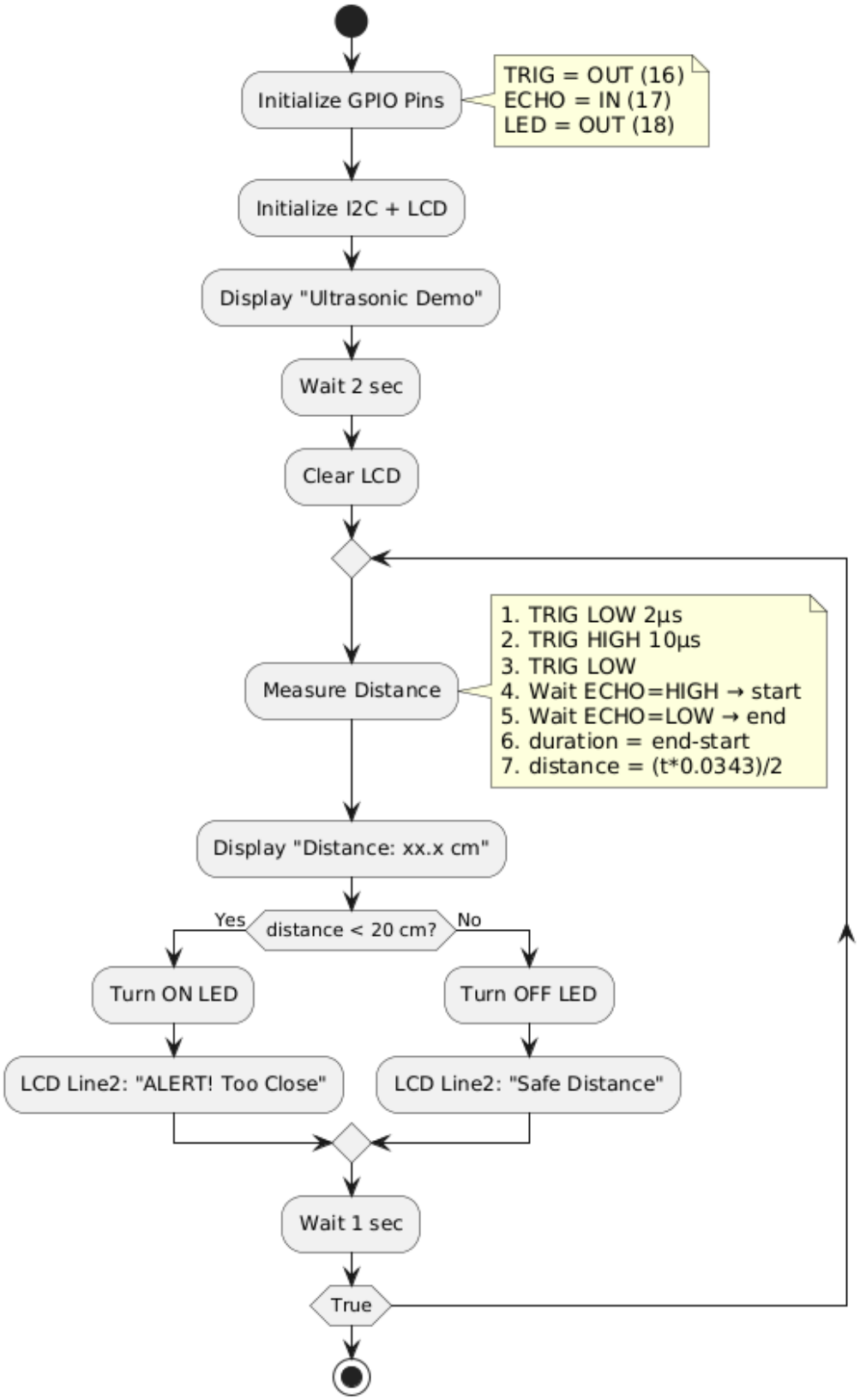
PROCEDURE:

- Connect the HC-SR04 sensor, LEDs, and buzzer to the Raspberry Pi Pico's GPIO pins.
- Set up the trigger as output and echo as input in MicroPython.
- Send a 10µs pulse from the trigger to start measurement.
- Measure the time it takes for the echo to return.
- Calculate distance using the formula.
- Turn ON LEDs and buzzer if an object is detected within a set range.
- Repeat the process continuously for real-time detection.

CIRCUIT CONNECTION:



FLOWCHART:


Initialize GPIO Pins
TRIG = OUT (16)
ECHO = IN (17)
LED = OUT (18)
Initialize I2C + LCD
Display "Ultrasonic Demo"
Wait 2 sec
Clear LCD
Measure Distance
1. TRIG LOW 2µs
2. TRIG HIGH 10µs
3. TRIG LOW
4. Wait ECHO=HIGH → start
5. Wait ECHO=LOW → end
6. duration = end-start
7. distance = (t*0.0343)/2
Display "Distance: xx.x cm"
distance < 20 cm?
Yes
No
Turn ON LED
Turn OFF LED
LCD Line2: "ALERT! Too Close"
LCD Line2: "Safe Distance"
Wait 1 sec
True

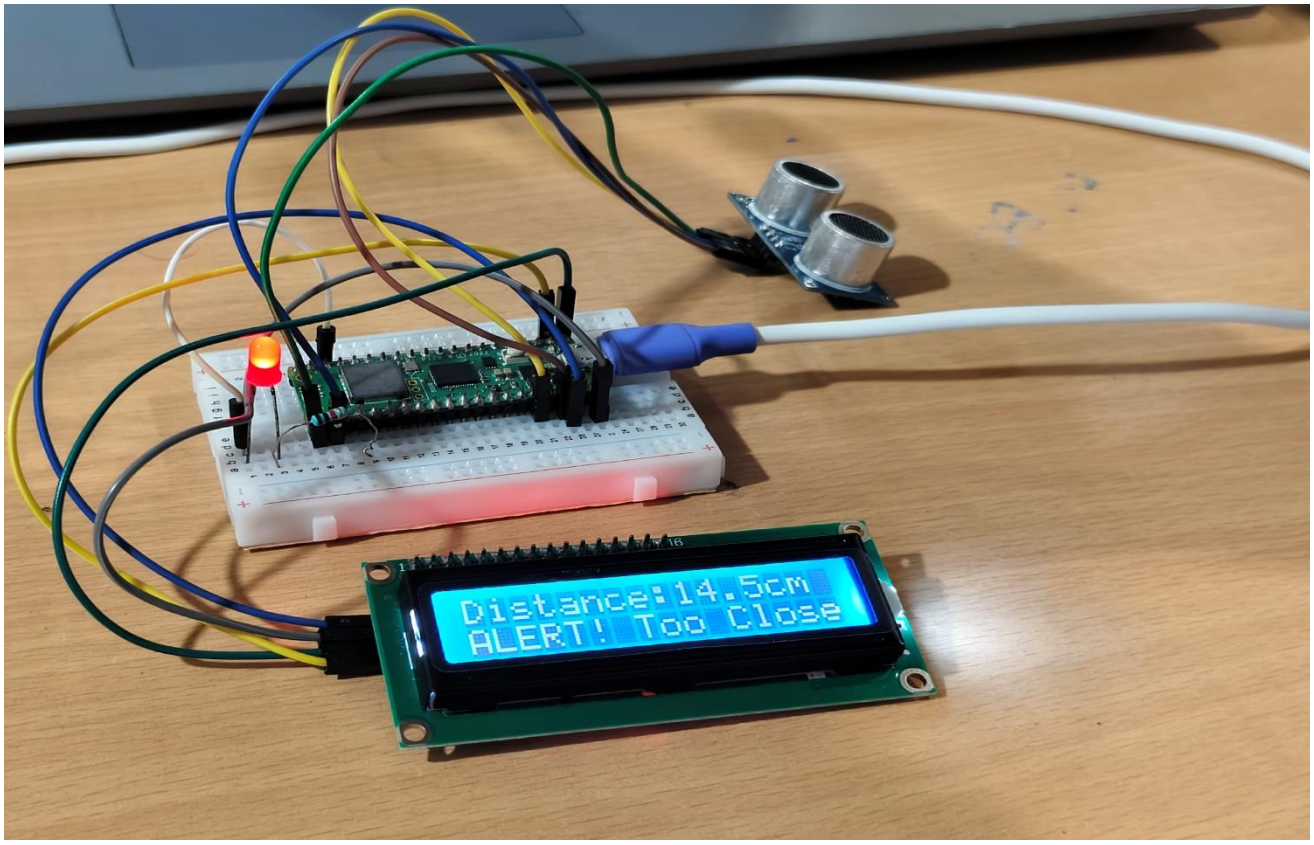
PROGRAM:

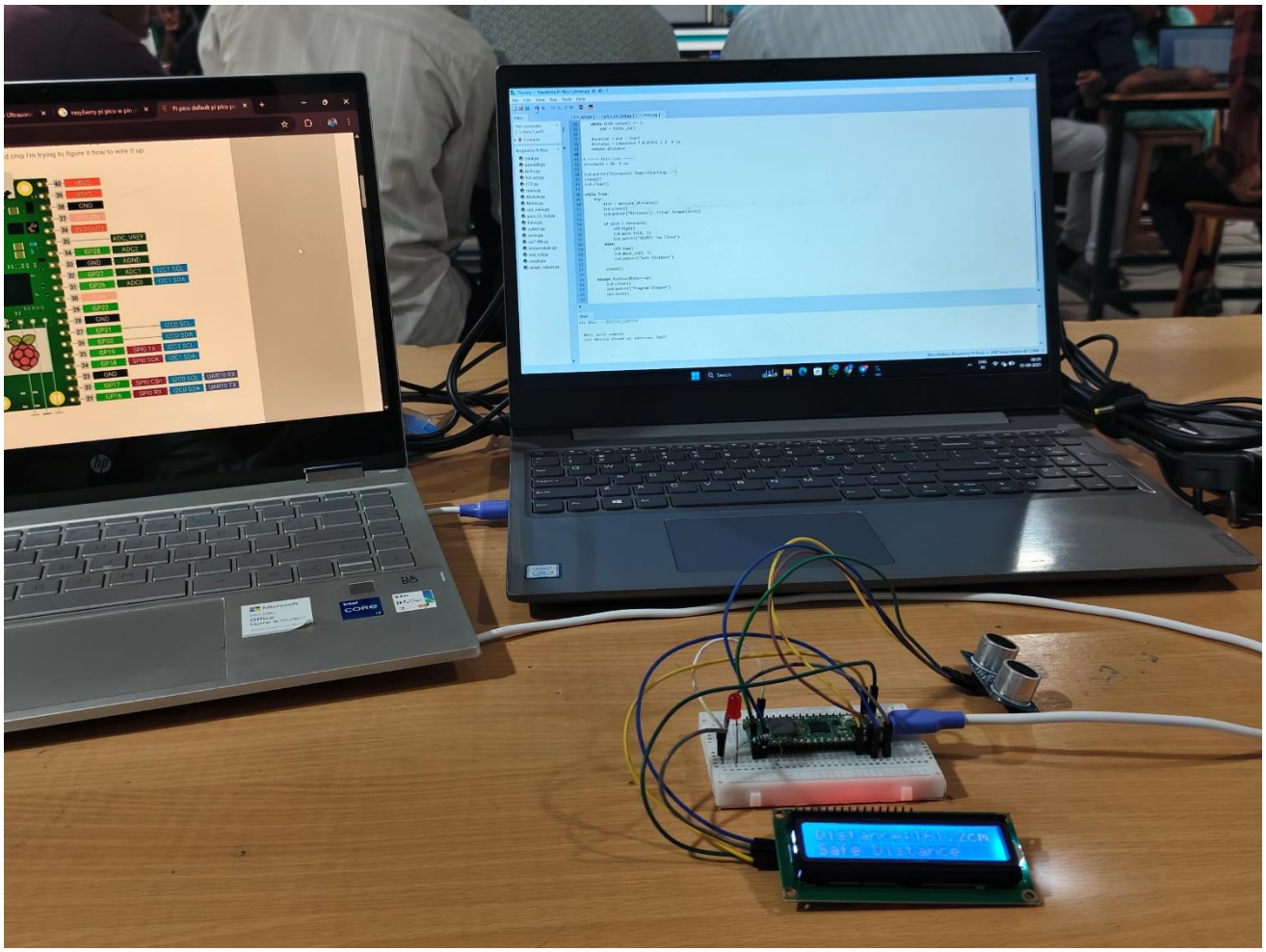
```
from machine import Pin, I2C
from time import sleep, sleep_us, ticks_us
import sys
from pico_i2c_lcd import I2cLcd

TRIG = Pin(16, Pin.OUT)
ECHO = Pin(17, Pin.IN)
LED = Pin(18, Pin.OUT)

i2c = I2C(0, scl=Pin(1), sda=Pin(0), freq=400000)
devices = i2c.scan()

if len(devices) == 0:
   print("No I2C device found!")
   sys.exit()
else:
   I2C_ADDR = devices[0]  # Auto-detect first device (commonly 0x27 or 0x3F)
   print("I2C device found at address:", hex(I2C_ADDR))

lcd = I2cLcd(i2c, I2C_ADDR, 2, 16)

def measure_distance():
   TRIG.low()
   sleep_us(2)
   TRIG.high()
   sleep_us(10)
   TRIG.low()

   while ECHO.value() == 0:
      start = ticks_us()
   while ECHO.value() == 1:
      end = ticks_us()

   duration = end - start
   distance = (duration * 0.0343) / 2  # cm
   return distance

threshold = 20  # cm

lcd.putstr("Ultrasonic Demo\nStarting...")
sleep(2)
lcd.clear()

while True:
   try:
```

```
        dist = measure_distance()
        lcd.clear()
        lcd.putstr("Distance:{:.1f}cm".format(dist))

        if dist < threshold:
            LED.high()
            lcd.move_to(0, 1)
            lcd.putstr("ALERT! Too Close")
        else:
            LED.low()
            lcd.move_to(0, 1)
            lcd.putstr("Safe Distance")

        sleep(1)

    except KeyboardInterrupt:
        lcd.clear()
        lcd.putstr("Program Stopped")
        sys.exit()
```

EXECUTION:

The distance measurement and object detection system using the Raspberry Pi Pico W and the HC-SR04 ultrasonic sensor works by sending a short pulse from the Trigger pin of the sensor and measuring the time it takes for the echo to return. The Raspberry Pi Pico W is connected to the sensor's VCC and GND for power, and its GPIO pins are used to control the Trigger and Echo signals. The measured time is then converted into distance using the speed of sound formula. If an object is detected within a certain threshold distance, indicator LEDs turn on and a buzzer sounds to alert the user. The program is written in Micro Python, which initializes the GPIO pins, sends pulses, reads the response, and calculates the distance. The output is displayed in real-time on the console, while the LEDs and buzzer provide immediate feedback. This setup effectively detects objects and measures distances, making it useful for robotics, obstacle detection, and safety applications.


Distance:14.5cm
ALERT! Too Close



RESULT:

The system measures distances up to 4 meters using the ultrasonic sensor and Raspberry Pi Pico W. When an object is within a set range, LEDs light up and the buzzer sounds as an alert. Distance readings are shown in real-time, making it useful for obstacle detection and safety applications.