SMART PARKING

PHASE 3 DEVELOPMENT PART I

Project Logic:

The project's logic revolves around an ultrasonic sensor emitting a high-frequency sound wave, which travels through the air until it encounters an object, at which point the system measures the time it takes for the sound wave to travel to the object and back. By calculating the distance using the known speed of sound, the system determines whether the area is "In Use" when an object is nearby or "Free" when no object is detected. The occupancy status is visually communicated on the OLED display, and the entire process is continuously monitored in a loop, allowing for real-time updates and responsiveness to changes in the environment. This setup is commonly used in applications like presence detection, parking space monitoring, or any scenario where knowing the occupancy status of an area is crucial.

Components:

A. Raspberry Pi Pico
B. Ultrasonic Distance Sensors (3x)
C. OLED Display (SSD1306)

Connections:

Raspberry Pi Pico:

- SDA (Pin 0) connected to the OLED display's SDA pin (Data)
- SCL (Pin 1) connected to the OLED display's SCL pin (Clock)

Ultrasonic Sensor 1:

- Trigger1 (Pin 26) connected to the trigger pin of the first ultrasonic sensor.
- Echo1 (Pin 27) connected to the echo pin of the first ultrasonic sensor.

Ultrasonic Sensor 2:

- Trigger2 (Pin 21) connected to the trigger pin of the second ultrasonic sensor.
- Echo2 (Pin 20) connected to the echo pin of the second ultrasonic sensor.

Ultrasonic Sensor 3:

- Trigger3 (Pin 19) connected to the trigger pin of the third ultrasonic sensor.
- Echo3 (Pin 18) connected to the echo pin of the third ultrasonic sensor.

OLED Display (SSD1306):

- VCC (3.3V) connected to the Raspberry Pi Pico's 3.3V power supply.
- GND connected to the Raspberry Pi Pico's GND (ground).
- SDA connected to the SDA pin (Pin 0) on the Raspberry Pi Pico.
- SCL connected to the SCL pin (Pin 1) on the Raspberry Pi Pico.

Python Script:

```
from machine import Pin, I2C
from ssd1306 import SSD1306_I2C
import utime

trigger1 = Pin(26, Pin.OUT)
echo1 = Pin(27, Pin.IN)
trigger2 = Pin(21, Pin.OUT)
echo2 = Pin(20, Pin.IN)
trigger3 = Pin(19, Pin.OUT)
echo3 = Pin(18, Pin.IN)

i2c = I2C(0, sda=Pin(0), scl=Pin(1), freq=400000)
oled = SSD1306_I2C(128, 64, i2c, addr=0x3C)
def ultra(trigger, echo):
    trigger.low()
    utime.sleep_us(2)
    trigger.high()
    utime.sleep_us(5)
    trigger.low()
    while echo.value() == 0:
        signaloff = utime.ticks_us()
    while echo.value() == 1:
        signalon = utime.ticks_us()
    timepassed = signalon - signaloff
    distance = (timepassed * 0.0343) / 2
    return distance

def display_status(distance):
    if distance > 200:
        status = "Free"
    else:
        status = "In Use"
    return status

while True:
    distance1 = ultra(trigger1, echo1)
    distance2 = ultra(trigger2, echo2)
    distance3 = ultra(trigger3, echo3)

    status1 = display_status(distance1)
    status2 = display_status(distance2)
    status3 = display_status(distance3)

    oled.fill(0)
    oled.text("Slot 1: " + status1, 0, 10)
    oled.text("Slot 2: " + status2, 0, 30)
    oled.text("Slot 3: " + status3, 0, 50)
    oled.show()
```

Code Overview:

Importing Libraries: The code starts by importing the necessary libraries:

machine: This library provides access to the hardware interfaces and pins on the Raspberry Pi Pico.

I2C: It's used for configuring the I2C communication interface.

ssd1306.SSD1306_I2C: This library is used to interact with the SSD1306 OLED display via the I2C interface.

utime: It provides functions for time-related operations.

Hardware Configuration: The code sets up the pins for the three ultrasonic sensors (trigger and echo pins) and configures the I2C interface for the OLED display. It also initializes the OLED display with the specified I2C address (0x3C).

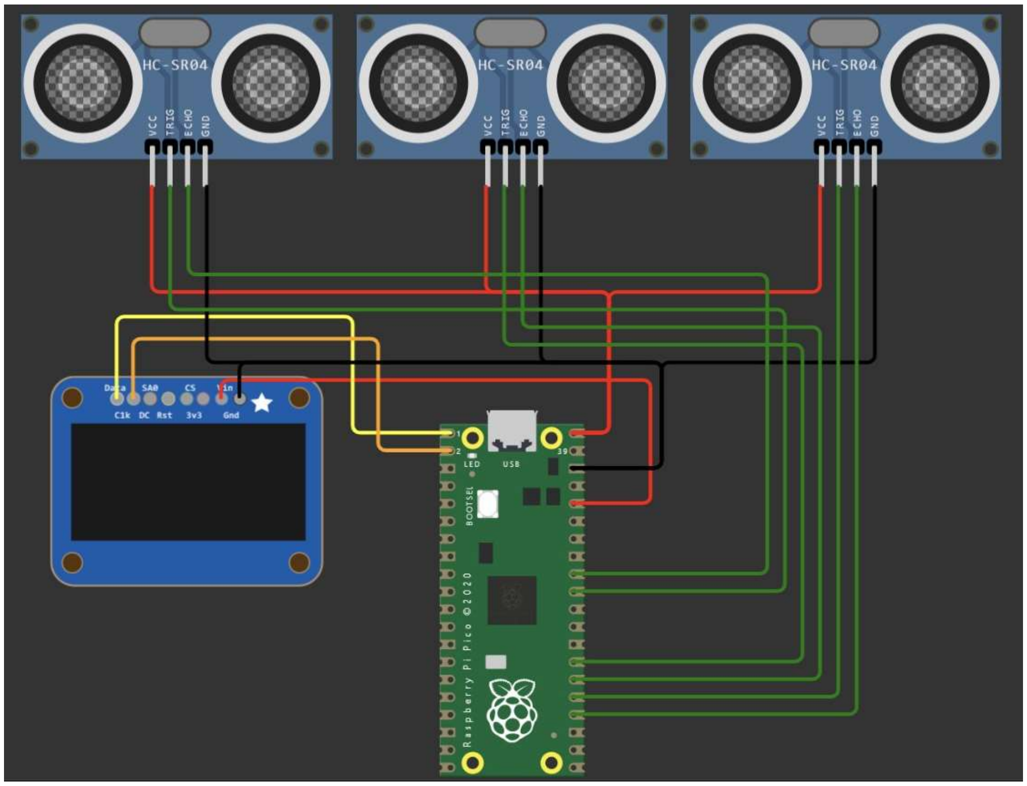
ultra() Function: This function takes a trigger pin and an echo pin as input and measures the distance using an ultrasonic sensor connected to these pins. It sends a trigger signal, measures the time it takes for the echo signal to return, and calculates the distance based on the time.

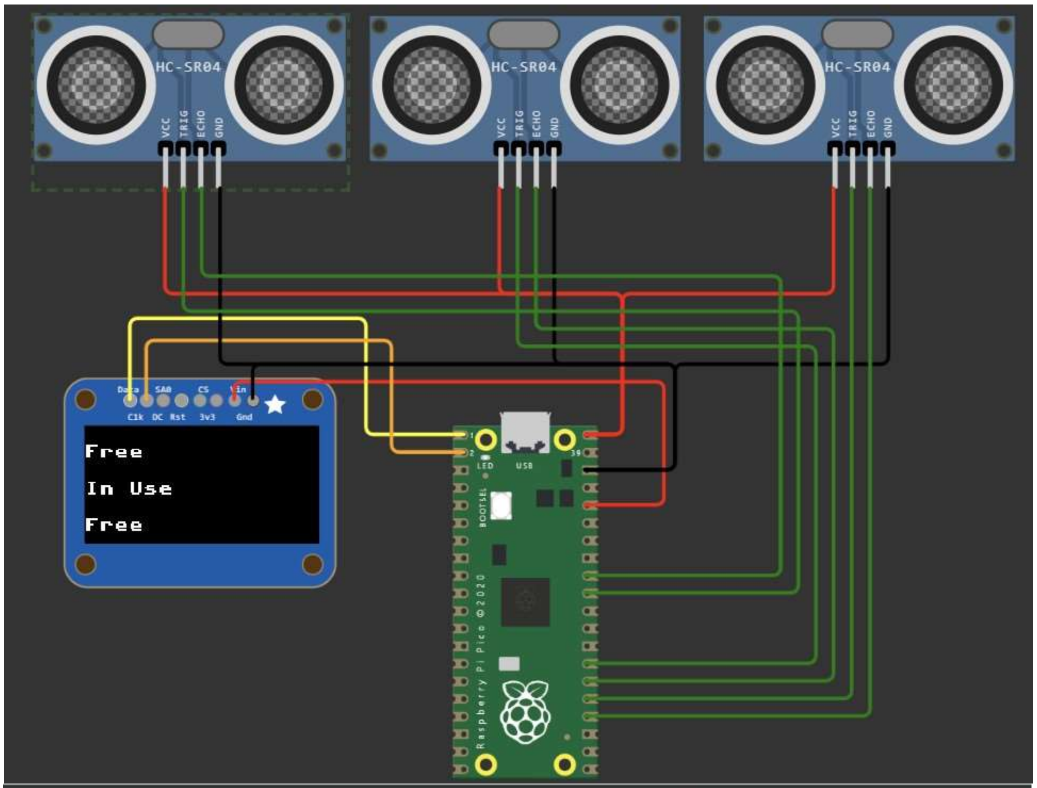
display_status() Function: This function takes the distance as input and determines the status, which can be "Free" if the distance is greater than 200 or "In Use" otherwise.

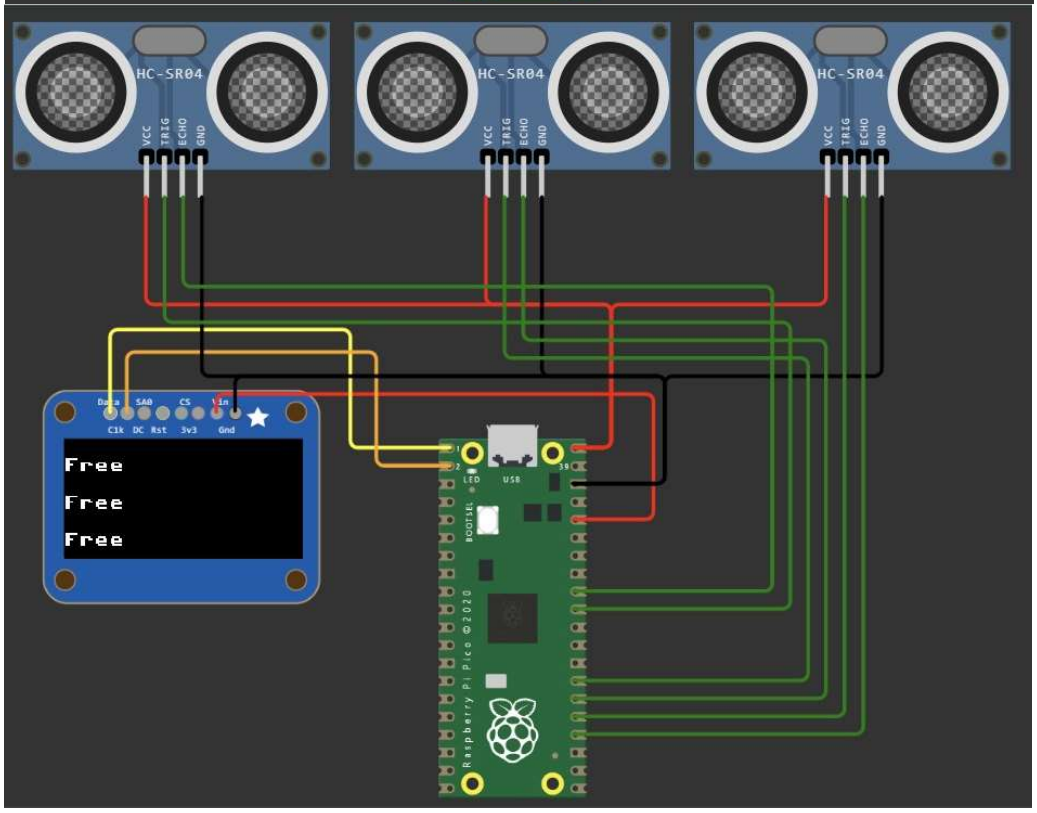
Main Loop: The main loop continuously reads the distances from the three ultrasonic sensors using the ultra() function. It then calculates the status for each sensor using the display_status() function based on the distance measured.

OLED Display: For each sensor, it clears the OLED display, displays the status on separate lines (vertical positions), and updates the display using oled.show(). This allows you to see the real-time status of each sensor on the OLED display.

Circuit Diagram:




HC-SR04
HC-SR04
HC-SR04
VCC
TRIG
ECHO
GND
Data
SA0
CS
Vin
Clk
DC
Rst
3v3
Gnd
Free
In Use
Free
LED
USB
BOOTSEL
Raspberry Pi Pico ©2020


HC-SR04
HC-SR04
HC-SR04
VCC
TRIG
ECHO
GND
Data
SA0
CS
Vin
Clk
DC
Rst
3v3
Gnd
Free
Free
Free
LED
USB
BOOTSEL
Raspberry Pi Pico ©2020