import machine

import time

# Constants

TRIGGER\_PULSE\_DURATION = 10 # microseconds

SOUND\_SPEED = 340.29 # m/s

BLINK\_DURATION = 0.5 # seconds

# Ultrasonic sensor pin configurations

ultrasonic\_pins = [

{"trigger": 5, "echo": 18},

{"trigger": 17, "echo": 16},

{"trigger": 4, "echo": 15}

]

# Sound sensor pins

sound\_sensor\_pins = [34, 35, 32]

# Traffic light pin configurations

traffic\_light\_pins = [

{"red": 19, "yellow": 21, "green": 22},

{"red": 23, "yellow": 25, "green": 26},

{"red": 27, "yellow": 14, "green": 13}

]

# Setup ultrasonic sensors

ultrasonics = []

for pins in ultrasonic\_pins:

trigger\_pin = machine.Pin(pins["trigger"], machine.Pin.OUT)

echo\_pin = machine.Pin(pins["echo"], machine.Pin.IN)

ultrasonics.append({"trigger": trigger\_pin, "echo": echo\_pin})

# Setup sound sensors

sound\_sensors = [machine.Pin(pin, machine.Pin.IN) for pin in sound\_sensor\_pins]

# Setup traffic light pins

traffic\_lights = []

for pins in traffic\_light\_pins:

red = machine.Pin(pins["red"], machine.Pin.OUT)

yellow = machine.Pin(pins["yellow"], machine.Pin.OUT)

green = machine.Pin(pins["green"], machine.Pin.OUT)

traffic\_lights.append({"red": red, "yellow": yellow, "green": green})

# Function to measure distance using ultrasonic sensor

def measure\_distance(trigger, echo):

trigger.off()

time.sleep\_us(2)

trigger.on()

time.sleep\_us(TRIGGER\_PULSE\_DURATION)

trigger.off()

timeout = time.ticks\_us()

while echo.value() == 0:

if time.ticks\_diff(time.ticks\_us(), timeout) > 1000000:

return None

start = time.ticks\_us()

while echo.value() == 1:

if time.ticks\_diff(time.ticks\_us(), start) > 1000000:

return None

end = time.ticks\_us()

duration = time.ticks\_diff(end, start)

distance = (duration / 2) \* (SOUND\_SPEED / 1000000)

return distance

# Detect emergency vehicle based on sound

def detect\_emergency():

return any(sensor.value() == 1 for sensor in sound\_sensors)

# Reset all traffic lights to off

def reset\_all\_traffic\_lights():

for light in traffic\_lights:

light["red"].off()

light["yellow"].off()

light["green"].off()

# Set traffic light to a specific state

def set\_traffic\_light(index, state):

for i, light in enumerate(traffic\_lights):

if i == index:

if state == "red":

light["red"].on()

light["yellow"].off()

light["green"].off()

elif state == "yellow":

light["red"].off()

light["yellow"].on()

light["green"].off()

elif state == "green":

light["red"].off()

light["yellow"].off()

light["green"].on()

else:

# All other lights red

light["red"].on()

light["yellow"].off()

light["green"].off()

# Blink specific traffic light

def blink\_traffic\_light(index, state, duration):

end\_time = time.time() + duration

while time.time() < end\_time:

set\_traffic\_light(index, state)

time.sleep(BLINK\_DURATION)

reset\_all\_traffic\_lights()

time.sleep(BLINK\_DURATION)

# Main loop

def main():

while True:

distances = [measure\_distance(u["trigger"], u["echo"]) for u in ultrasonics]

for i, distance in enumerate(distances):

if distance is None:

print(f"[Sensor {i}] Measurement failed.")

continue

print(f"[Sensor {i}] Distance: {distance:.2f} meters")

if distance < 0.2:

blink\_traffic\_light(i, "red", 5)

print(f"🔴 Congestion at sensor {i}")

elif distance < 0.5:

blink\_traffic\_light(i, "yellow", 5)

print(f"🟡 Warning at sensor {i}")

else:

blink\_traffic\_light(i, "green", 5)

print(f"🟢 Clear road at sensor {i}")

if detect\_emergency():

print("🚨 Emergency vehicle detected!")

else:

print("✅ No emergency vehicle detected.")

time.sleep(1)

main()