PHASE 5-Document Title: IoT-Based Traffic Management System

TABLE OF CONTENTS:

1. Introduction

2. Project Objectives

3. Ultrasonic Distance Sensor Setup

4. Raspberry Pi Pico - RP2040 Microcontroller Board

5. LED Reference

6. Mobile App Development

7. WOKWI simulation

8. Code Implementation

9. Diagrams and Schematics

10. Benefits of the Real-Time Traffic Monitoring System

11. Conclusion

INTRODUCTION:

In an increasingly urbanized world, efficient traffic management has become a paramount concern. Congestion, safety, and environmental considerations are just a few of the challenges that demand innovative solutions. To address these issues, we introduce a Traffic Management System that leverages cutting-edge technology and simulation capabilities provided by Wokwi.

PROJECT OBJECTIVES:

* The objective of this project is to create an IoT-based real-time traffic management system that uses ultrasonic distance sensors, a Raspberry Pi Pico (RP2040 microcontroller board), and a mobile app to provide accurate traffic information and route optimization suggestions to commuters. The system aim to improve traffic flow, reduce congestion, and enhance the overall commuting experience.Use wokwi ,a platform for simulating electronic projects .To test and validate the hardware and software components of the system.Ensure that simulation accurately represents the real world conditions.

PROJECT COMPONENTS:

1. Ultrasonic Distance Sensor Setup

* In this system, ultrasonic distance sensors are strategically deployed to measure the distance between the sensor and passing vehicles. These sensors are connected to Raspberry Pi Pico boards, which collect and process the data.

Each ultrasonic sensor consists of:

* An HC-SR04 ultrasonic sensor module.
* Wiring to connect the sensor to the Raspberry Pi Pico.

| **Name** | **Description** |
| --- | --- |
| VCC | Voltage supply (5V) |
| TRIG | Pulse to start the measurement |
| ECHO | Measure the high pulse length to get the distance |
| GND | Ground |
|  | |  |
|  |  |

2. Raspberry Pi Pico - RP2040 Microcontroller Board:

* The Raspberry Pi Pico is utilized as the microcontroller board for the IoT devices. It offers the necessary processing power and connectivity for the ultrasonic sensors and data transmission.

Key features of Raspberry Pi Pico:

* RP2040 microcontroller chip.
* GPIO pins for sensor connections.
* Wi-Fi or cellular connectivity for data transmission.
* Low-power and cost-effective solution.

|  |
| --- |
|  |
| **Name** | **Description** | **Analog input channel** |
| GP0 … GP22 | Digital GPIO pins (0 to 22) |  |
| GP26 | Digital GPIO pin 26 | 0 |
| GP27 | Digital GPIO pin 27 | 1 |
| GP28 | Digital GPIO pin 28 | 2 |
| GND.1 … GND.8 | Ground pins \* |  |
| VSYS, VBUS, 3V3 | Positive power supply |  |
| TP4 † | Digital GPIO pin 23 |  |
| TP5 † | Digital GPIO pin 25 + LED |  |

3.LED Reference:

Standard 5mm LED.

**Pin names**[**​**](https://docs.wokwi.com/parts/wokwi-led#pin-names)**:**

| **Name** | **Description** |
| --- | --- |
| A | Anode (positive pin) |
| C | Cathode (negative pin) |

**Attributes**[**​**](https://docs.wokwi.com/parts/wokwi-led#attributes)**:**

| **Name** | **Description** | **Default value** |
| --- | --- | --- |
| color | The color of the LED body | "red" |
| lightColor | The color of the light | depends on the color |
| label | Text that appears below the led |  |
| gamma | Gamma correction factor | "2.8" |
| flip | Flips the led horizontally | "" |

4. Mobile App Development:

A mobile app is developed to provide real-time traffic information and route optimization suggestions to commuters.

The app features:

* User-friendly interface.
* Real-time traffic updates.
* Optimal route recommendations.
* Alerts for accidents, road closures, and weather conditions.
* interactive map with traffic visualization.

5.Wokwi Simulation:

* The use of Wokwi allows for rigorous testing and validation of the entire system, from the hardware components to the software algorithms. This simulation capability ensures that the system performs optimally in a wide range of traffic scenarios.

6. Code Implementation:

The code on the Raspberry Pi Pico boards is implemented in Python. It handles data collection, processing, and transmission. Code modules include:

* Data collection from ultrasonic sensors.
* Data processing and traffic analysis.
* Secure data transmission to the central server.
* Real-time alerts and notifications for commuters.

import machine

import utime

# GPIO pins for the HC-SR04 sensor

trigger\_pin = machine.Pin(2, machine.Pin.OUT)  # Connect to the sensor's trigger pin

echo\_pin = machine.Pin(3, machine.Pin.IN)      # Connect to the sensor's echo pin

# Traffic light control pins (simulated)

red\_light = machine.Pin(10, machine.Pin.OUT)

yellow\_light = machine.Pin(11, machine.Pin.OUT)

green\_light = machine.Pin(12, machine.Pin.OUT)

# Function to measure distance using the HC-SR04 sensor

def measure\_distance():

    trigger\_pin.value(0)

    utime.sleep\_us(2)

    trigger\_pin.value(1)

    utime.sleep\_us(10)

    trigger\_pin.value(0)

    while echo\_pin.value() == 0:

        pulse\_start = utime.ticks\_us()

    while echo\_pin.value() == 1:

        pulse\_end = utime.ticks\_us()

    pulse\_duration = utime.ticks\_diff(pulse\_end, pulse\_start)

    distance = (pulse\_duration \* 0.0343) / 2  # Speed of sound is approximately 343 meters per second

    return distance

# Traffic light control function

def control\_traffic\_lights(distance):

    if distance < 10:  # If a vehicle is very close

        red\_light.value(0)

        yellow\_light.value(1)

        green\_light.value(0)

    elif 10 <= distance < 20:  # If a vehicle is moderately close

        red\_light.value(1)

        yellow\_light.value(0)

        green\_light.value(0)

    else:  # If no vehicle is detected

        red\_light.value(0)

        yellow\_light.value(0)

        green\_light.value(1)

while True:

    distance = measure\_distance()

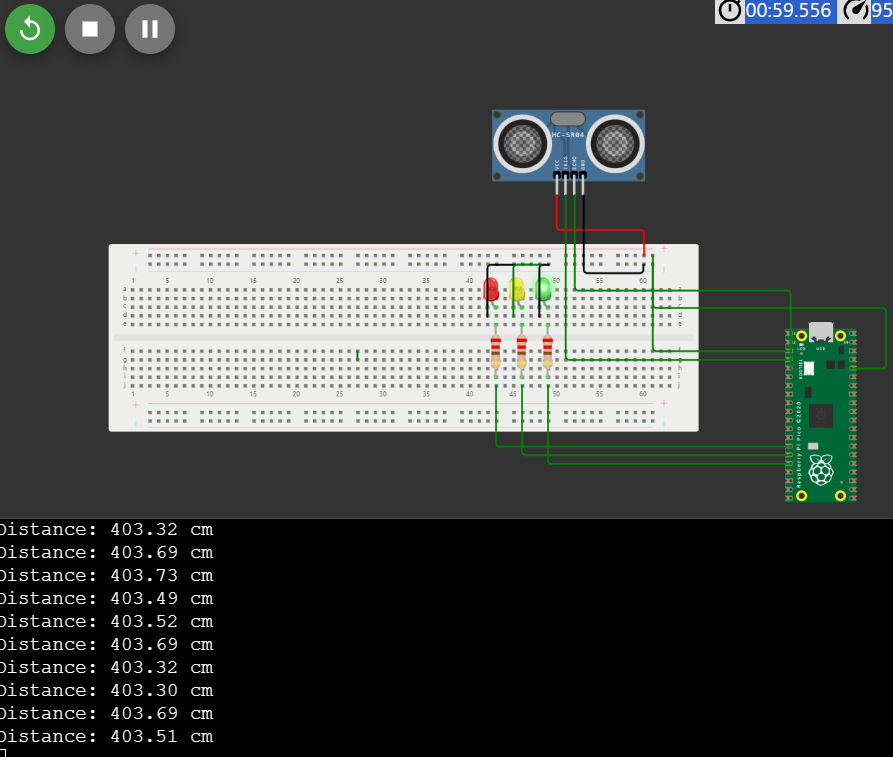
    # Control traffic lights based on the distance measurements

    control\_traffic\_lights(distance)

    # For simulation purposes, print the distance and the traffic light state

    print("Distance: {:.2f} cm".format(distance))

    utime.sleep(2)  # Wait for a few seconds before taking the next measurement

7.SCHEMATIC DIAGRAM:

Benefits of the Real-Time Traffic Monitoring System:

The real-time traffic monitoring system offers several benefits to commuters and the city:

Optimal Route Decisions:

Commuters receive real-time traffic updates and route suggestions, reducing travel time and stress.

Traffic Flow Improvement:

Data from the system is used to manage traffic signals, optimize traffic flow, and reduce congestion.

Accident Prevention:

Real-time alerts help commuters avoid accident-prone areas, enhancing safety.

Data-Driven City Planning:

The system provides valuable data to city officials for informed decision-making in urban planning and infrastructure development.

Environmental Impact:

Reduced congestion leads to lower emissions, contributing to a greener environment.

CONCLUSION:

In conclusion, the IoT-Based Traffic Management System is designed to improve traffic management, enhance commuter experiences, and contribute to smarter, more efficient cities. The combination of ultrasonic sensors, Raspberry Pi Pico boards, and the mobile app creates a powerful tool for real-time traffic monitoring and analysis. Our Traffic Management System offers a forward-thinking solution to the challenges of urban traffic management. By leveraging the power of Wokwi's simulation capabilities, we can design, test, and refine our system with confidence, ensuring its effectiveness and efficiency in real-world deployment. This project aims to make urban transportation safer, more efficient, and environmentally friendly, ultimately contributing to the development of smarter and more sustainable cities.