GOVERNMENT OF TAMILNADU

DIRECTORATE OF TECHNICAL EDUCATION, CHENNAI

NAAN MUDHALVAN SCHEME (TNSDC) SPONSORED

STUDENTS DEVELOPMENT PROGRAMME

ON

IoT AND ITS APPLICATIONS

HOST INSTITUTION

xxxxx

COIMBATORE – 04

TRAINING PARTNER

ENTHU TECHNOLOGY SOLUTIONS INDIA PVT LTD

DATE:

Name	Roll No
Name 1	1
Name 2	2
Name 3	3
Name 4	4
Name 5	5

Table Of Contents

Abstract

In urban areas, the efficient management of traffic flow is essential to reduce congestion and ensure road safety. The Four-Way Traffic Light Controller System is designed to manage traffic at an intersection with four roads, ensuring an organized flow of vehicles while minimizing the likelihood of accidents.

This project utilizes microcontroller-based technology to control the traffic lights at each intersection, cycling through red, green, and yellow lights in a predefined sequence. The system is programmed to handle different traffic scenarios, including peak and non-peak hours, by adjusting the duration of the lights accordingly.

The controller system is further enhanced by integrating it with Wi-Fi connectivity to ThingzMate Cloud, allowing for remote monitoring and control. This connection enables real-time updates on the status of the traffic lights and allows for the modification of light sequences or timings based on live traffic data or emergency situations.

The system also features an uplink for remote communication and an authorization mechanism to ensure secure access. Through this project, we aim to improve the efficiency of traffic management systems, reduce the likelihood of congestion, and enhance overall road safety.

By combining traditional traffic light control with modern IoT capabilities, this project demonstrates a scalable solution that can be adapted for various urban traffic environments.

Introduction

In modern urban environments, the management of traffic at intersections is crucial for maintaining an efficient flow of vehicles and ensuring the safety of both drivers and pedestrians. Traffic congestion and accidents are common problems at busy intersections, often resulting from poorly coordinated traffic signals. A four-way intersection, where vehicles from four different directions converge, requires a carefully designed control system to manage the movement of traffic effectively.

The Four-Way Traffic Light Controller System is developed to address these challenges by automating the control of traffic lights at such intersections. The system uses a microcontroller to sequence the traffic lights in a manner that optimizes traffic flow, reducing wait times and minimizing the risk of collisions. The lights are programmed to follow a cycle of red, yellow, and green phases, each with adjustable durations to accommodate varying traffic densities at different times of the day

Incorporating Internet of Things (IoT) technology, the system is connected to the ThingzMate Cloud platform, allowing for remote monitoring and control. This connectivity enables traffic authorities to adjust the light sequences in real-time based on live traffic conditions or to prioritize emergency vehicles. Additionally, the system includes security features such as an authorization mechanism to prevent unauthorized access and ensure the integrity of the traffic control operations.

By integrating traditional traffic signal control with modern IoT capabilities, this system offers a robust and scalable solution for managing four-way intersections in busy urban areas. It not only improves traffic efficiency but also enhances road safety, making it a vital component of smart city infrastructure.

Hardware and Software Requirements

Hardware Requirements

1.ESP32 Microcontroller

2.LED

3.BreadBoard

4.USB Cable

5.Jumper Wires

Software Requirements

1.Wokwi Simulator

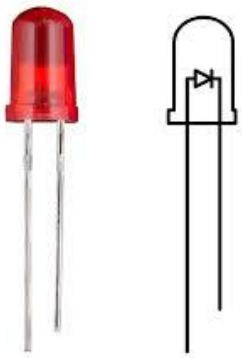
2.Arduino IDE

3.Thingzmate Cloud

ESP32 Microcontroller

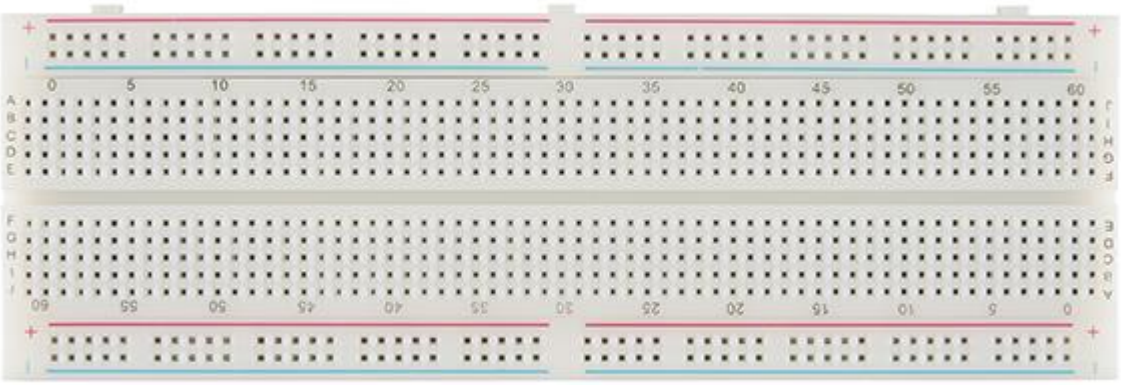
The ESP32 microcontroller is a powerful and versatile chip used in this project to control the four-way traffic light system. It features integrated Wi-Fi and Bluetooth capabilities, allowing for seamless communication with the ThingzMate Cloud for remote monitoring and control. With its dual-core processor and multiple GPIO pins, the ESP32 efficiently handles the timing and sequencing of the traffic lights, making it ideal for real-time IoT applications.

LED



In this project, standard LEDs are used to represent the traffic lights at the four-way intersection. These LEDs are controlled by the ESP32 microcontroller, which turns them on and off according to the programmed traffic light sequence. The LEDs provide a simple yet effective way to visualize the traffic signals, making them essential for testing and demonstrating the functionality of the traffic light controller system.

BreadBoard



The breadboard is an essential component in this project, used to prototype the circuit connections for the four-way traffic light controller system. It allows for easy placement and rearrangement of the LEDs, resistors, and connections to the ESP32 microcontroller without the need for soldering. The breadboard's flexibility makes it ideal for quickly testing and modifying the circuit design as the project develops.

USB-Cable

The USB cable is a critical tool in this project, used to connect the ESP32 microcontroller to a computer for power supply, programming, and debugging. It enables the transfer of code and data between the development environment and the microcontroller, facilitating the upload of firmware and real-time communication during the development process. The USB connection also allows for serial monitoring, providing valuable insights into the system's performance and behavior.

Jumper Wires

Jumper wires are essential in this project, used to connect the ESP32 microcontroller to the components on the breadboard. These wires provide a flexible and reliable way to link the microcontroller's GPIO pins to the LEDs, resistors, and other circuit elements, enabling proper signal and power flow. Their ease of use allows for quick modifications and testing during the prototyping stage.

Wokwi Simulator

Wokwi Simulator is a powerful online tool used in this project to simulate the four-way traffic light controller system before physical implementation. It allows for the virtual testing of the ESP32 microcontroller, LEDs, and other components, enabling real-time visualization and debugging of the circuit and code. By using Wokwi, developers can ensure the system's functionality and make necessary adjustments without needing the actual hardware setup.

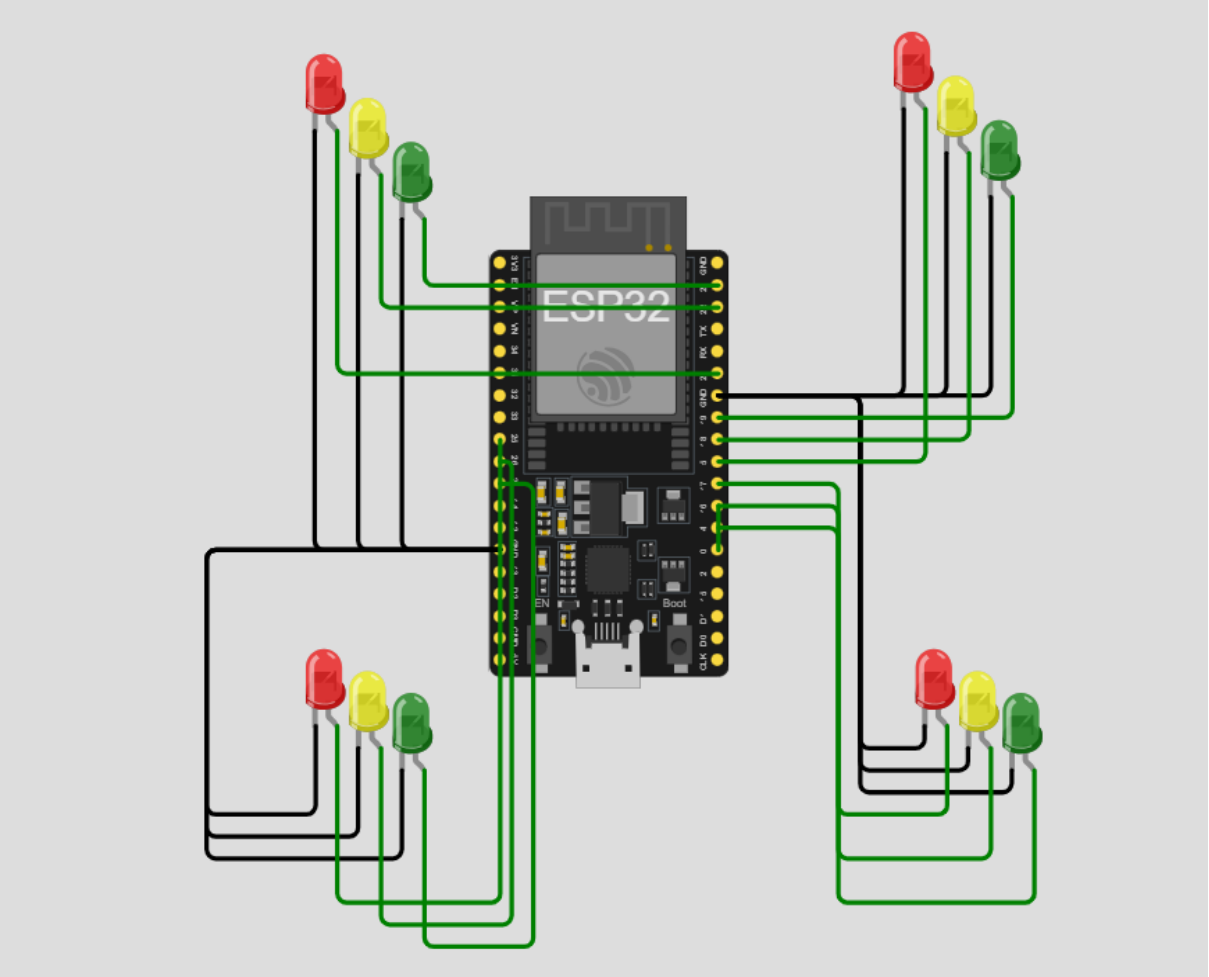
Arduino IDE

Arduino IDE is the primary development environment used in this project for programming and uploading code to the ESP32 microcontroller. It provides an easy-to-use interface for writing, compiling, and debugging the code that controls the four-way traffic light system. With its extensive library support and compatibility with ESP32, the Arduino IDE streamlines the development process, allowing for efficient code iteration and testing.

Thingzmate Cloud

ThingzMate enables real-time simulation of a 4-way traffic light control system, allowing you to monitor and manage traffic lights via cloud connectivity.It provides tools for configuring traffic light sequences, ensuring accurate simulation of traffic flow and light transitions.With ThingzMate, you can easily visualize and control traffic light statuses, optimizing traffic management and ensuring efficient simulation scenarios.

Block Diagram



Code

```
#include <WiFi.h>
#include <HTTPClient.h>

// Wi-Fi credentials
#define WIFI_SSID "tyrant"
#define WIFI_PASSWORD "speed123"

// ThingzMate Cloud credentials and server URL
const char  *serverUrl  = "https://console.thingzmate.com/api/v1/device-types/esp3211/devices/esp3211/uplink"; // Replace with your actual server endpoint
String AuthorizationToken = "Bearer 5a38d738c842f32d6091b9a13fec23a5"; // Replace with your actual token

// Define pins for traffic lights
int red1 = 5;
int yellow1 = 18;
int green1 = 19;

int red2 = 21;
int yellow2 = 22;
int green2 = 23;

int red3 = 25;
int yellow3 = 26;
int green3 = 27;

int red4 = 14;
```

```
int yellow4 = 12;
int green4 = 13;

void setup() {
  Serial.begin(115200);
  delay(4000); // Delay to let serial settle

  // Connect to WiFi
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  Serial.println("Connected to WiFi");

  // Initialize traffic light pins as outputs
  pinMode(red1, OUTPUT);
  pinMode(yellow1, OUTPUT);
  pinMode(green1, OUTPUT);

  pinMode(red2, OUTPUT);
  pinMode(yellow2, OUTPUT);
  pinMode(green2, OUTPUT);

  pinMode(red3, OUTPUT);
  pinMode(yellow3, OUTPUT);
  pinMode(green3, OUTPUT);
```

```
  pinMode(red4, OUTPUT);
  pinMode(yellow4, OUTPUT);
  pinMode(green4, OUTPUT);
}

void loop() {
  // Sector 1 Go
  digitalWrite(red1, LOW);
  digitalWrite(yellow1, LOW);
  digitalWrite(green1, HIGH);
  digitalWrite(red2, HIGH);
  digitalWrite(yellow2, LOW);
  digitalWrite(green2, LOW);
  digitalWrite(red3, HIGH);
  digitalWrite(yellow3, LOW);
  digitalWrite(green3, LOW);
  digitalWrite(red4, HIGH);
  digitalWrite(yellow4, LOW);
  digitalWrite(green4, LOW);
  sendStatusToServer("Sector 1: GO", "Sector 2: STOP", "Sector 3: STOP", "Sector 4: STOP");
  delay(4000);

  // Sector 1 Yellow
  digitalWrite(red1, LOW);
  digitalWrite(yellow1, HIGH);
  digitalWrite(green1, LOW);
  sendStatusToServer("Sector 1: YELLOW", "Sector 2: STOP", "Sector 3: STOP", "Sector 4: STOP");
```

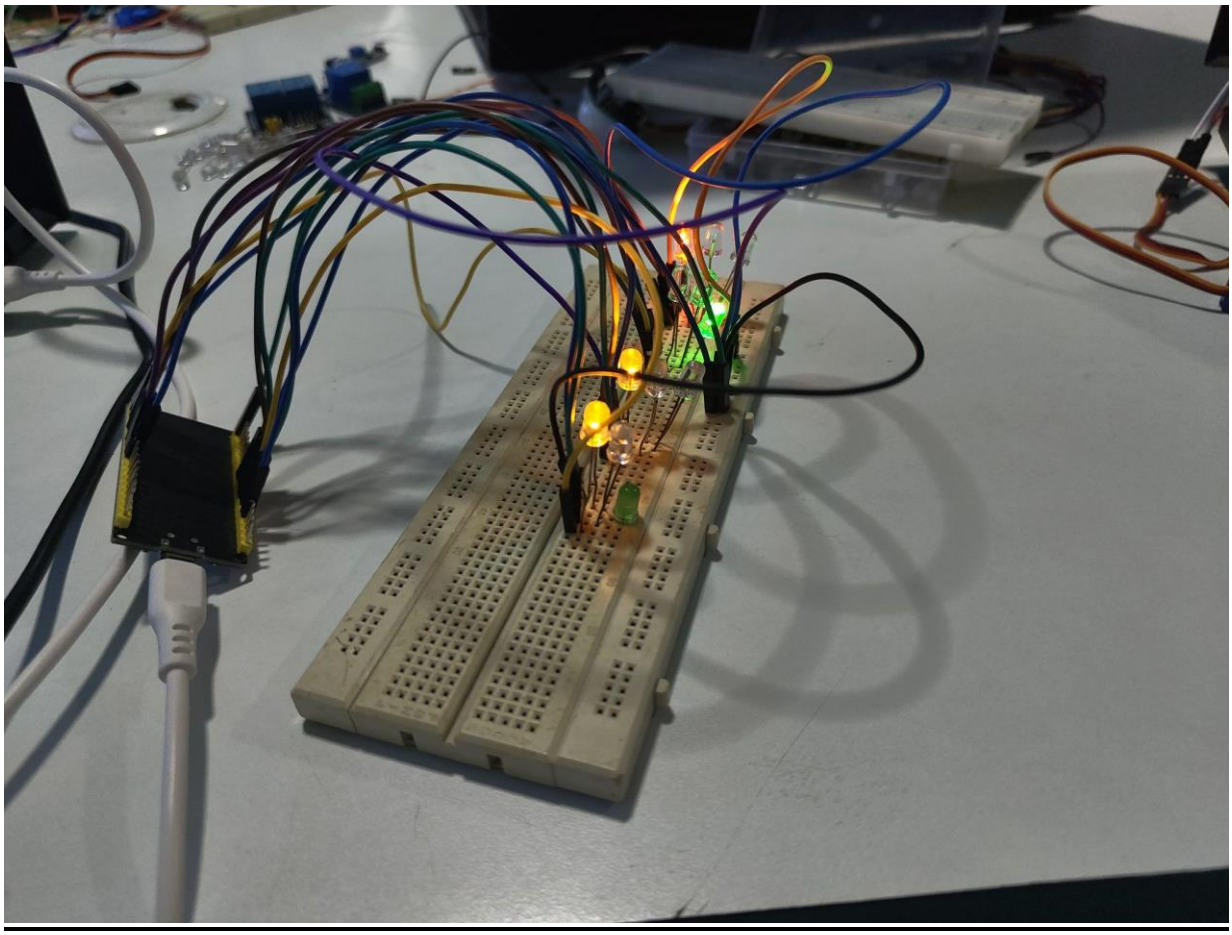
```
  delay(1000);

  // Sector 2 Go
  digitalWrite(red1, HIGH);
  digitalWrite(yellow1, LOW);
  digitalWrite(green1, LOW);
  digitalWrite(red2, LOW);
  digitalWrite(yellow2, LOW);
  digitalWrite(green2, HIGH);
  digitalWrite(red3, HIGH);
  digitalWrite(yellow3, LOW);
  digitalWrite(green3, LOW);
  digitalWrite(red4, HIGH);
  digitalWrite(yellow4, LOW);
  digitalWrite(green4, LOW);
  sendStatusToServer("Sector 1: STOP", "Sector 2: GO", "Sector 3: STOP", "Sector 4: STOP");
  delay(4000);

  // Sector 2 Yellow
  digitalWrite(red2, LOW);
  digitalWrite(yellow2, HIGH);
  digitalWrite(green2, LOW);
  sendStatusToServer("Sector 1: STOP", "Sector 2: YELLOW", "Sector 3: STOP", "Sector 4: STOP");
  delay(1000);

  // Sector 3 Go
  digitalWrite(red1, HIGH);
```

```
  digitalWrite(yellow1, LOW);
  digitalWrite(green1, LOW);
  digitalWrite(red2, HIGH);
  digitalWrite(yellow2, LOW);
  digitalWrite(green2, LOW);
  digitalWrite(red3, LOW);
  digitalWrite(yellow3, LOW);
  digitalWrite(green3, HIGH);
  digitalWrite(red4, HIGH);
  digitalWrite(yellow4, LOW);
  digitalWrite(green4, LOW);
  sendStatusToServer("Sector 1: STOP", "Sector 2: STOP", "Sector 3: GO", "Sector 4: STOP");
  delay(4000);

  // Sector 3 Yellow
  digitalWrite(red3, LOW);
  digitalWrite(yellow3, HIGH);
  digitalWrite(green3, LOW);
  sendStatusToServer("Sector 1: STOP", "Sector 2: STOP", "Sector 3: YELLOW", "Sector 4: STOP");
  delay(1000);

  // Sector 4 Go
  digitalWrite(red1, HIGH);
  digitalWrite(yellow1, LOW);
  digitalWrite(green1, LOW);
  digitalWrite(red2, HIGH);
  digitalWrite(yellow2, LOW);
```

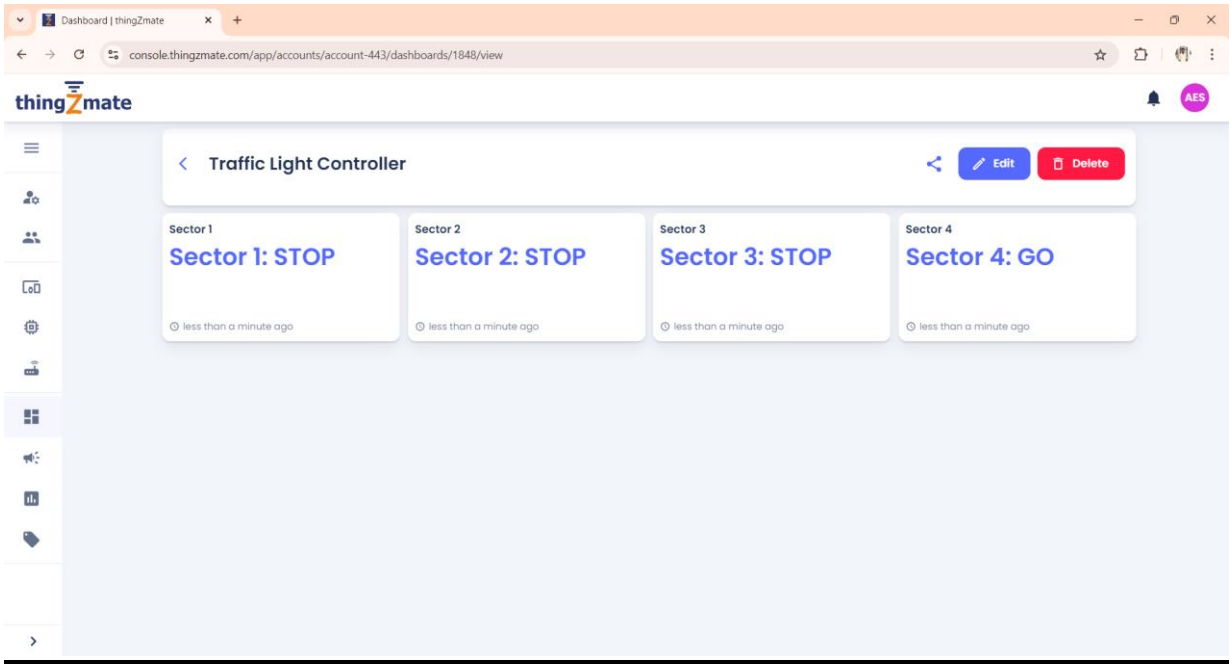
```
  digitalWrite(green2, LOW);
  digitalWrite(red3, HIGH);
  digitalWrite(yellow3, LOW);
  digitalWrite(green3, LOW);
  digitalWrite(red4, LOW);
  digitalWrite(yellow4, LOW);
  digitalWrite(green4, HIGH);
  sendStatusToServer("Sector 1: STOP", "Sector 2: STOP", "Sector 3: STOP", "Sector 4: GO");
  delay(4000);

  // Sector 4 Yellow
  digitalWrite(red4, LOW);
  digitalWrite(yellow4, HIGH);
  digitalWrite(green4, LOW);
  sendStatusToServer("Sector 1: STOP", "Sector 2: STOP", "Sector 3: STOP", "Sector 4: YELLOW");
  delay(1000);
}

void sendStatusToServer(String sector1Status, String sector2Status, String sector3Status, String sector4Status) {
  HTTPClient http;
  http.begin(serverUrl);
  http.addHeader("Content-Type", "application/json");
  http.addHeader("Authorization", AuthorizationToken);

  // Create JSON payload
  String payload = "{\"sector1\":\"" + sector1Status + "\",\"sector2\":\"" + sector2Status + "\",\"sector3\":\"" + sector3Status + "\",\"sector4\":\"" + sector4Status + "\"}";
```

```
  // Send POST request
  int httpResponseCode = http.POST(payload);

  if (httpResponseCode > 0) {
    String response = http.getString();
    Serial.println("HTTP Response code: " + String(httpResponseCode));
    Serial.println(response);
  } else {
    Serial.print("Error code: ");
    Serial.println(httpResponseCode);
  }

  http.end(); // Free resources
}
```

Output Results



Cloud Output



Conclusion

In conclusion, a 4-way traffic light control system is essential for managing complex intersections efficiently, enhancing traffic flow, and ensuring safety. By employing advanced technologies like ThingzMate, which provides real-time simulation and cloud-based control, we can optimize traffic management strategies, adapt to varying traffic conditions, and improve overall transportation infrastructure. This integration not only streamlines the control process but also allows for proactive adjustments and more informed decision-making, ultimately contributing to smoother and safer roads.