Automation of Traffic Congestion Reduction Management System

Submitted in partial fulfillment of the requirements for the degree

Of

Bachelor of Science in Information and Communication Engineering (ICE)

By

Jarif Foysal [213-50-067]

MD Emam Hossain Shanto [213-50-071]

Under the Supervision of

Engr. Md. Zahirul Islam

Assistant Professor

Department of Information and Communication Engineering (ICE)

Daffodil International University

October 2025

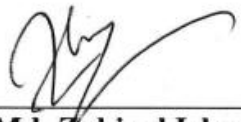
APPROVAL

This is to certify that the entitled **Automation of Traffic Congestion Reduction Management System,** submitted by Jarif Foysal [213-50-067] and MD Emam Hossain Shanto [213-50-071] are undergraduate students of the Department of ICE has been examined. Upon recommendation by the examination committee, we hereby accord our approval of it as the presented work and submitted report fulfill the requirements for its acceptance in partial fulfillment for the degree of *Bachelor of Science in Information and Communication Engineering.*

BOARD OF EXAMINERS

Md. Taslim Arefin — Chairman
Associate Professor and Head
Department of ICE
Daffodil International University

Dr. A.K.M Fazlul Haque — Internal Examiner
Professor
Department of ICE
Daffodil International University



Engr. Md. Zahirul Islam — Internal Examiner
Assistant Professor
Department of ICE
Daffodil International University

Md. Meshkatur Rahman — External Examiner
Asst. General Manager
Live Technologies Ltd.

DECLARATION OF AUTHORSHIP

Project Title Automation of Traffic Congestion Reduction Management System

Authors Jarif Foysal, MD Emam Hossain Shanto

Supervisor Engr. Md. Zahirul Islam,

Assistant Professor

We, Jarif Foysal and MD Emam Hossain Shanto, hereby declare that this capstone project titled **Automation of Traffic Congestion Reduction Management System,** is entirely our own work, unless otherwise referenced or acknowledged. The content of this project is the result of our own research and efforts, and we have complied with all ethical guidelines and academic standards.

We are aware of the consequences of academic dishonesty and understand that any violation of ethical standards in this project may lead to disciplinary actions as defined by Daffodil International University's policies.

Jarif foysal

Jarif Foysal
213-50-067

Md. Emam hossain shanto

MD Emam Hossain Shanto
213-50-071

Engr. Md. Zahirul Islam
Assistant Professor
Department of Information and Communication Engineering

ACKNOWLEDGEMENT

We are grateful to the following individuals and organizations who played a significant role in the successful completion of this project.

We would like to express our heartfelt appreciation **to Engr. Md. Zahirul Islam**, Assistant Professor of the Information and Communication Engineering Department, for his invaluable guidance and unwavering support during this project. Their expertise, encouragement, and constructive feedback have been the driving force behind the success of this endeavor.

Engr. Md Zahirul Islam provided an atmosphere which encourages the development of creativity and critical thinking, enabling me to mature academically and personally. We wish to thank them for their guidance, and for inspiring this work which was instrumental in shaping the outcome of the project.

We appreciate Engr. Md. Zahirul Islam for his valuable guidance during the work of this project. His inspiration, advice and support has been essential for establishing our appreciation and accomplishment of this work.

We thank Daffodil International University for the necessary facilities, resources and suitable academic environment to carry out this research work. The college's dedicated policy on research and innovation was an early catalyst for us.

We are also extremely grateful for the continuous encouragement, patience and support from our families and friends during rough times of this project. Their faith in us has been a constant source of strength and inspiration.

We would like to thank all who have directly or indirectly helped us. Without their hard work and generosity, this project could never have been completed. We are lucky to have been in the presence of such incredible people on this journey.

ABSTRACT

In this project, built an intelligent traffic congestion management system with an ESP32 board. Traffic signal timings are changed in realtime by the system based on vehicle density data captured using 8 such IR sensor modules placed across various lanes of the road. The intelligent modification of green time duration (i.e., 10, 15, or 20 seconds) for the green lights are programmed according to different traffic conditions in order to prevent traffic jam and improve the flow of traffic. It has 4 traffic light modules that manage signal directions and it also counts with a 16x2 LCD display that has an I2C driver for real status updates. Power is supplied using two 3.7 V batteries, and the output voltage is reduced to 5 V by an MP1584 buck converter with good power efficiency. Supporting components in form of jumper wires, male-female headers, power switch and veroboard make it easy to connect the hardware. It is mounted on a white PVC board, which serves as a stable base for the device. At the conclusion of each traffic cycle, signal timings are sent over Wi-Fi to a Google Sheet by the ESP32 and stored in the cloud for analysis. It's in real time, so when there are no cars on the block, there are quicker transit times because the signals are computerized and intelligent." The presented solution is a scalable and low-energy cost way of traffic management promoting urban mobility. The use of cloud analytics also enables the monitoring tracking of trends over time of traffic flow patterns contributing to future urban development planning decisions. It is also possible to further develop the system, by integrating functionality such as pedestrian detection or emergency vehicle prioritization. The system is equipped with remote monitoring feature allowing intelligent and dynamic way of city traffic control, enabling its deployment on different types of cities ranging from small towns to mega cities.

Keywords:

- ESP32
- Traffic Management
- IR Sensor
- Google Sheet
- Real-Time Data
- Smart City.

TABLE OF CONTENTS

CONTENTS

List of Figure

Chapter 1 INTRODUCTION

1.1 Background and Motivation

Amid the fast urbanization of cities nowadays, traffic jams have become one of the most momentous problems in big and small cities. Traditional traffic signal systems are based on using fixed time intervals for intersections regardless of the current traffic flow. This useless system works to the detriment of people time, gas, and patience. Traffic jams not only destroy productivity but support higher levels of carbon emissions and road accident. In order to solve these problems, more and more intelligent traffic control systems are demanding which can change mode effectively according to the changing of real road situation dynamically.

As this project moves forward, smarter and more adaptive traffic-management systems have become possible. This is what lets sensors and microcontrollers send and receive data over the internet. [1]This enables automatic traffic control and remote traffic parameter monitoring. In this scenario our project is to develop a Automation of traffic congestion reduction management system using ESP32 using the microcontroller ESP32.[2]

This framework involves multiple IRs deployed on different lanes of a four-way intersection to sense vehicle density. The system uses sensor data to determine how long the light for each lane should be green—10, 15 or 20 seconds—which makes it more efficient for traffic. ESP32 receives the data from all these sensors and communicates with the Traffic Light modules based on it. Calculated durations are transmitted using Wi-Fi after each full cycle of the traffic light to a Google Sheet, for cloud-based monitoring and analysis. This instantaneous data logging is valuable for traffic management to learn patterns and tune the timing policies of future signals.

A major drive for this work is to propose economically viable, low-complexity solutions possibly deployable in the third world countries like Bangladesh, where high-end traffic systems are not affordable or practical at many locations. This project provides a resilient and cost-effective alternative to expensive smart traffic systems, all made possible by using off-the-shelf components like IR sensors, 16x2 LCD, MP1584, traffic light modules and ESP32. The system is also mobile and may be deployed at temporary or permanent intersections.

Yet another incentive is the social and academic potential that this work provides. It makes sensor integration, microcontroller programming, IoT based data communication and automation logic easily accessible - all very important building blocks for modern embedded systems and Smart City applications. By streaming live traffic statistics to Google Sheets, this project also demonstrates how cloud platforms can be deployed for remote data access and visualization as well as other AI/machine learning enhancements.

Overall, smart parking solutions are of increasing demand and importance in solving the urgent problem of traffic congestion; they also become more affordable compared with the alternatives based on IoT techniques and has taught us to think and perform beyond what we have learned so far about integrating advanced technology. The solution reconciles the existing traffic with future smart city infrastructures and is realistic and applicable now, with the ability to evolve over time to incorporate new technologies.

1.2 Literature Review

Several research studies and projects have been conducted in recent years to improve traffic management systems using automation and smart technologies.[3]Traditional traffic light systems operate on predefined time cycles, which are inefficient in handling dynamic traffic

conditions. This limitation has led researchers to explore adaptive systems based on real-time traffic data.

In a study by Sivanandam et al.[4], an image processing-based traffic control system was proposed, where cameras detected vehicle density and adjusted signal timings accordingly. Although effective, the system was limited by high costs, complex image processing requirements, and low accuracy in poor weather or low-light conditions. IR sensor-based systems, on the other hand, offer a low-cost and reliable alternative for vehicle detection in specific zones.

Another study by Ahmed et al.[5] introduced an Arduino-based traffic control system using ultrasonic sensors to count vehicles. While this method showed improvements in signal timing accuracy, it lacked internet connectivity for remote monitoring and data logging, which limits scalability and real-time tracking.

More recent approaches incorporate Internet of Things (IoT) technologies. Kumar and Sharma[6] developed an IoT-based traffic signal controller using NodeMCU, IR sensors, and a cloud dashboard. The system successfully adjusted signal timing based on traffic load and sent real-time data to the cloud, enhancing the decision-making process. This project laid a foundation for low-cost, smart, and remotely accessible traffic systems.

Several academic projects have also shown that the ESP32 microcontroller is a powerful, Wi-Fi-enabled platform for handling multiple sensor inputs and cloud communication. Its low power consumption and dual-core processing make it suitable for real-time IoT applications.[7]

Based on the findings from these studies, our project builds upon proven technologies and methods but introduces additional improvements—such as the use of Google Sheets for simple and cost-effective cloud data logging. Unlike more complex systems, our solution remains

accessible for implementation in low-resource settings while maintaining real-time adaptiveness.

In conclusion, the reviewed literature supports the idea that IoT-based, sensor-driven traffic control systems offer practical and scalable solutions for modern cities.Our project contributes to this evolving field by combining low-cost components, Wi-Fi communication, and cloud integration into a functional and effective traffic management prototype.[8]

1.3 Problem Statement:

Urban traffic congestion is a growing issue in cities worldwide, especially in developing countries where infrastructure development often lags behind the rapid increase in vehicle usage. Traditional traffic signal systems operate on fixed time intervals, regardless of the real-time traffic density at intersections.[9] As a result, lanes with heavy traffic often experience longer waiting times, while lanes with little or no traffic are unnecessarily given green signals. This outdated method leads to increased fuel consumption, longer travel times, air pollution, and driver frustration.[10]

Also, traditional transportation systems do not have smart decision capabilities and are unable to monitor remotely or collect data. Traffic police in most cities hardly have access to real-time traffic information that would enable them to analyze and optimize traffic policies considering the actual states of roads. Moreover, lack of cost effective automation in upgrading legacy systems further complicates the process.

As a result, there is an urgent requirement for an intelligent and flexible traffic control system with low cost to be able to adaptively address the density of vehicles and to increase the efficiency of traffic flow instantiation. The system must be able to gather real-time data and share it for both monitoring and analysis. This project deals with these issues by developing a

density based traffic signal system using ESP32 in combination of IR sensors for operating the signals as per the values of count at peak & off-peak hours, it transmits cycling data to Google Sheets for cloud-based analysis.

1.4 Objectives:

The aim of the project is to develop an **intelligent traffic congestion reduction system** based on an ESP32 microcontroller. The system is integrated to reduce congestions in urban traffic logistics with intelligent signal timing, real-time data monitoring and cloudbased reporting. The project is aimed at achieving the following:

Primary Objectives:

- **Smart traffic control system for vehicle density calculation using Image processing:**

  IR sensor is deployed in every lane to sens the number of vehicles this gives the count of vehicle and hence the traffic density.

- **To design an automated signal timer algorithm:**

  The time of green signal light can be set dynamically at 10, 15 or 20 seconds during detection according to the flow density in order to make traffic more fluent.

- **To benefit from both the ESP32 microcontroller and its real-time control and data processing capabilities:**

  Collect sensor data using the ESP32MCU, implement traffic lights control with the same MCU, and handle cloud communications.

Secondary Objectives:

- **For realization inexpensive and power-consumption supported hardware:** Develop to a circuit using an inexpensive component (IR sensor, MP1584, traffic lights module, LCD).
- **Add an I2C capable 16x2 LCD display:**
  Show the current signal timings in real-time, sensor reading/status, and so much more for local monitoring of your system.
- **To establish cloud data logging using Google Sheets:**
  After each traffic signal cycle, send the green light durations to Google Sheets via Wi-Fi for data analysis and reporting.

These objectives collectively aim to develop an intelligent, adaptive, and cost-effective traffic management solution suitable for both developed and developing regions.

1.5 Scope of the Project

The Density-Based IoT Traffic Control Management System focuses on solving traffic congestion by automatically adjusting traffic signal timing based on vehicle density. This system integrates IoT and embedded technologies using cost-effective components to create a smart traffic solution suitable for developing countries. The scope of the project is explained through the following key points:

- The system uses the ESP32 microcontroller as the central processing unit, chosen for its built-in Wi-Fi, dual-core processor, and support for various sensor interfaces.

- Eight IR sensors are used to detect the presence of vehicles in each lane. Based on the count or presence of vehicles, the system dynamically assigns green light durations of 10, 15, or 20 seconds.
- Four traffic light modules are used to represent the signals of a four-way intersection. Each module has red, yellow, and green LEDs controlled by the ESP32.
- A 16x2 LCD display with an I2C interface shows real-time traffic signal status, green light duration, and sensor activity.
- Power is supplied using two 3.7V lithium-ion batteries in a 2S battery holder, with an MP1584 buck converter stepping down voltage to 5V for safe operation of all components.
- The system uses jumper wires, male and female headers, veroboard for circuit assembly, and a power switch for operational control. All components are mounted on a white PVC board for structure and portability.
- After each full traffic cycle, the system sends the green light timing data to a Google Sheet via Wi-Fi, using cloud-based services such as IFTTT or Google Apps Script.
- This cloud logging allows for traffic data monitoring, future analysis, and decision-making by traffic authorities or researchers.
- The system is ideal for small intersections, educational demonstrations, campus roads, and research labs where full-scale traffic infrastructure is unavailable or costly.
- Being modular and programmable, the system can easily be expanded by adding more sensors or additional functionality like emergency vehicle detection, pedestrian signals, or integration with mobile apps.
- The system does not yet support camera-based detection, pedestrian monitoring, or weather adaptation, which can be considered in future upgrades.

- Internet dependency is a limitation, as Google Sheet updates require an active Wi-Fi connection.
- Despite these limitations, the project is a working prototype for real-world application, offering low-cost and intelligent traffic control that can evolve into a more complex smart city infrastructure.
- Future developments may include solar-powered operation, mobile app monitoring, AI-based traffic prediction, and real-time alerts to authorities through SMS or email.

This project serves as an example of how embedded systems and IoT can be used to create impactful solutions in urban infrastructure with affordability, adaptability, and scalability in mind.

Chapter 2 PROJECT MANAGEMENT

2.1 Project Plan

Week	Task	Description
1–4	Planning, Research & Component Selection	During this period, we identified the current traffic signal issues, set the project objectives, and researched existing systems. We finalized the hardware components such as ESP32, IR sensors, and traffic light modules.
5–8	Circuit Design, Sensor Testing & Hardware Setup	We began connecting IR sensors and traffic light modules to the ESP32, designed the circuit layout on veroboard, and tested each sensor for vehicle detection accuracy.
9–12	Code Development, Timing Logic & LCD Integration	The ESP32 was programmed to collect IR sensor input and adjust traffic light timing to 10s, 15s, or 20s. We also integrated the I2C-based LCD to display traffic status in real-time.
13–16	System Integration & Functional Testing	All modules were integrated and tested together. Signal control logic was validated based on real-time sensor input. Adjustments were made for stable output and accurate time cycles.
17–20	Google Sheet Integration & Cloud Testing	We connected the ESP32 to Wi-Fi and implemented Google Sheet logging. Timing data after each cycle was successfully sent to the cloud for monitoring and future analysis.
21–24	Final Testing, Optimization & Report Writing	The complete system was tested in a simulated traffic environment. Minor bugs were fixed, and timing logic was fine-tuned. Finally, project documentation and final report writing were completed.

2.2 Contribution of Team Members

2.2.1 Team Member 1: Jarif Foysal, ID- 213-50-067.

A.Role/Responsibility:

The project was led by Jarif Foysal, Lead & Systems Developer responsible for project management, hardware integration, sensor calibration and IoT setup. He brought a chapter he worked on between software and hardware, as Funtchessor wrote an essential role in the operation of the system and its development.

B.Tasks:

Assisted in system architecture design, programmed the Arduino microcontroller (ESP32), interfaced IR sensors, Traffic Lights and LCD's modules – Jarif Foysal He also designed the timing logic for durations of green lights and incorporated Google Sheet logging via Wifi. He analyzed the data and aided in debugging as well as writing of the final report.

C.Contribution:

Jarif Foysal leadership along with his technical expertise was the-key that bought this project to life. His contribution to the project was the consolidation more than two modules and realtime data communication controlling. He paid a great deal of attention to detail which yielded reliable and scalable results, coupled with very good documentation support that lead us to the kind of clear and professional reporting we're seeking.

2.2.2 Team Member 2: MD Emam Hossain Shanto, ID – 213-50-071

A.Role/Responsibility:

MD Emam Hossain Shanto served as a circuit designer and system tester. They helped design the hardware layout, located components and debugged eletrical problems in prototyping phase. They were also instrumental in system testing and throughput benchmarking.

B.Tasks:

Among the soldering of the veroboard, preparing cable for hazard on the PVC board and keeping power under control with MP1584. They puerform IR detection and signal switching test cases, and noted the cycled results. They also contributed in formatting the project images, and organizing the closure report.

C.Contribution:

MD Emam Hossain Shanto's inputs added up the robustness and consistency of the system. They design the circuitry for accuracy and the process of testing systematically delivers

accurate sensor data processing and traffic signal operation. Their assistance in hardware completion and documentation gave both a technical as well as a presentational boost to the project.

2.3 Challenges and Decision Making

The two key problems were IR sensor-based vehicle detection and reliable Wi-Fi data delivery. Design decisions included specific part choice to be cost-effective, system simplification, and leveraging the Google Sheets as an easy cloud data log storage solution for reliability and scalability.

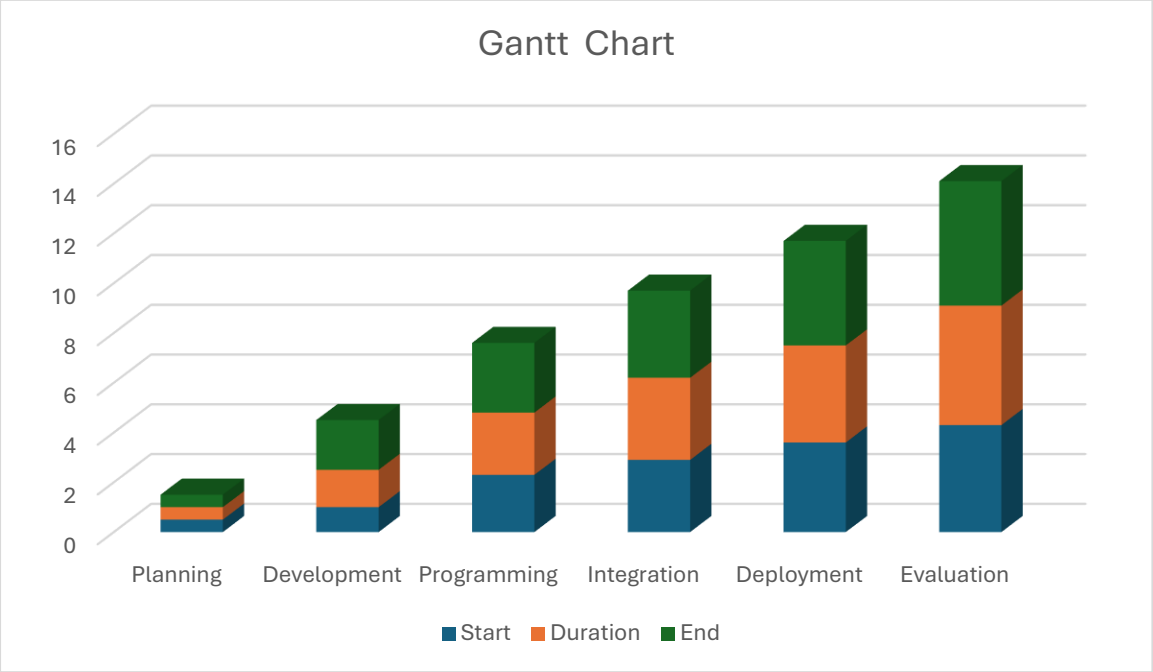


Figure 2.1: Project Gantt Chart

Chapter 3 SYSTEM DESIGN AND IMPLEMENTATION

3.1 Block Diagram of the System :

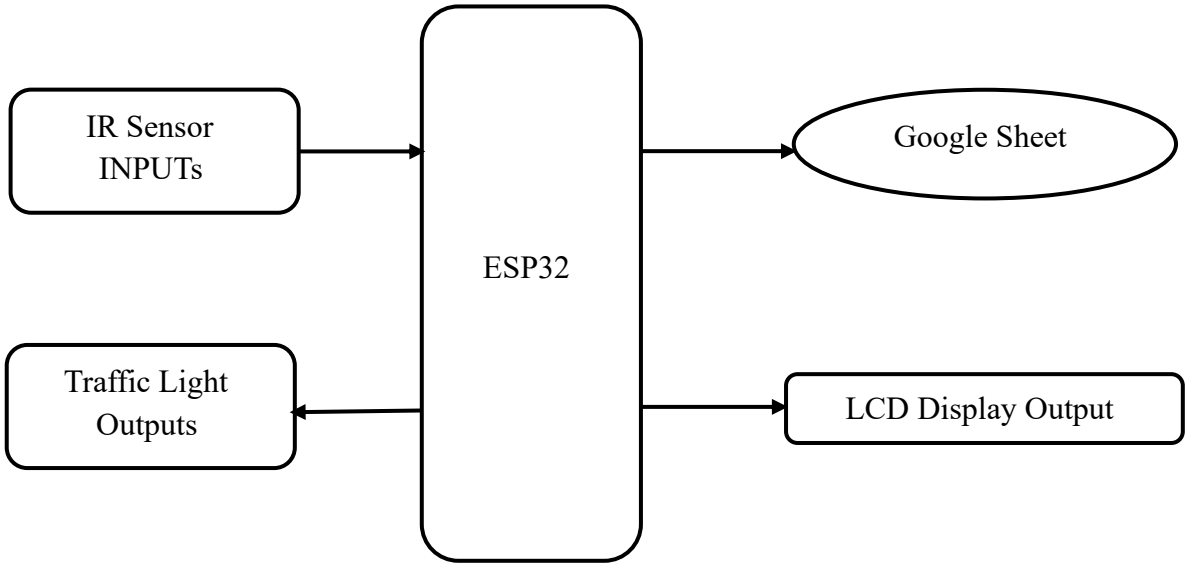


Figure 3.1: Block Diagram

The block diagram of the intelligent traffic management system illustrates the key components and their interactions. The ESP32 microcontroller serves as the central control unit, receiving inputs from eight IR sensors placed at different lanes for vehicle detection. Based on this data, the ESP32 adjusts traffic light signal durations, controlling four traffic light modules. A 16x2 LCD display provides real-time traffic status updates. Power is supplied by two 3.7V batteries, stepped down to 5V via an MP1584 buck converter. The system also includes Wi-Fi connectivity, enabling cloud-based monitoring and data logging on a Google Sheet for further analysis and optimization.

Explanation of Data Flow:

- **IR Sensors**: Detect vehicle presence in each lane.
- **ESP32**: Receives IR input, determines traffic density, calculates green light duration, and controls traffic lights and LCD display.

- **Traffic Light Modules**: Show RED, YELLOW, GREEN lights based on ESP32 output and lane priority.
- **LCD Display (I2C)**: Displays current lane and remaining green signal time.
- **Wi-Fi (built-in ESP32)**: Connects to the internet to send cycle timing data to **Google Sheets** using HTTP or webhook integration (e.g., via IFTTT).

3.2 Circuit Diagram and Hardware Components Description :

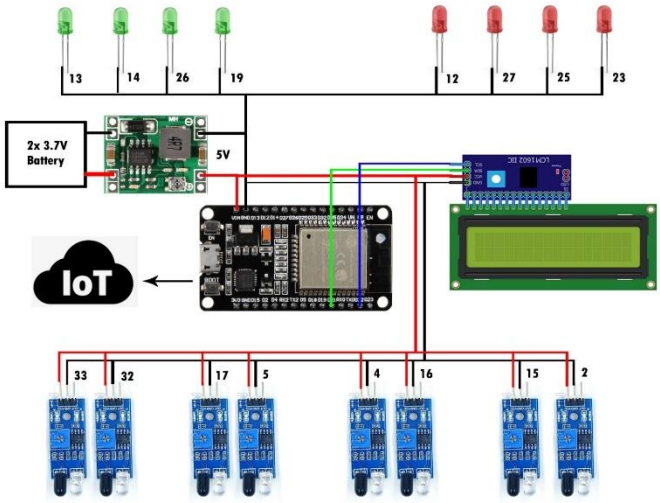


Figure 3.2: Circuit Diagram

The following hardware components were used to develop the density-based IoT traffic control system:

- **ESP32Microcontroller**:
  Acts as the brain of the system. It reads IR sensor data, controls the traffic lights and LCD, and sends data to Google Sheets via Wi-Fi. Its dual-core processing and built-in Wi-Fi make it ideal for real-time IoT applications.

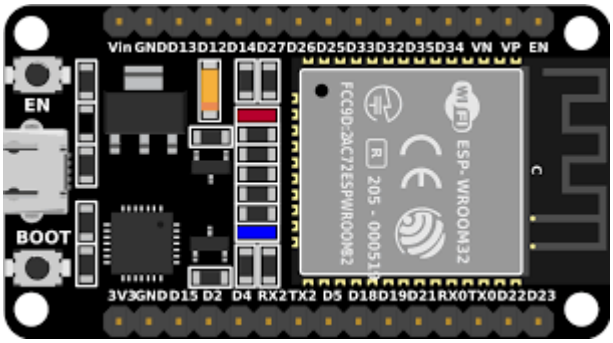


Figure 3.3: Esp32

- **IR Sensor Modules (8x)**:
  These infrared sensors detect the presence of vehicles in each lane. When a vehicle passes or stays in front of a sensor, it sends a signal to the ESP32 to determine lane density.

Figure 3.4: IR sensor module

- **Traffic Light Modules (4x)**:
  Each module contains red, yellow, and green LEDs simulating a real traffic light. Controlled by the ESP32, these modules switch signals based on the calculated vehicle density and predefined time durations (10s, 15s, 20s).

Figure 3.5: Traffic module

- **16x2 LCD Display with I2C Module**:
  This display is used to show real-time signal status such as the active lane and remaining green light time. The I2C driver reduces wiring complexity by requiring only two data lines (SDA and SCL).

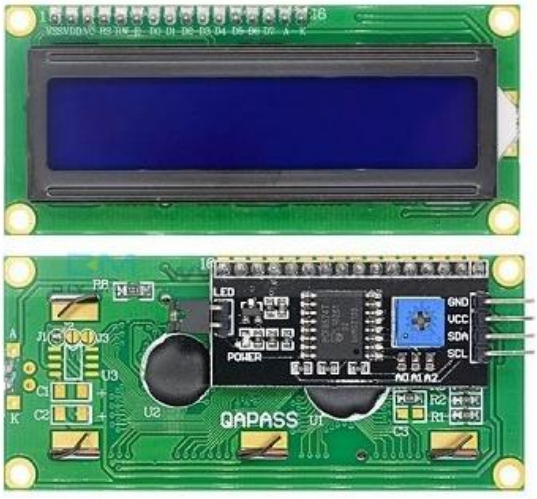


Figure 3.6: 16x2 lcd display with i2c

- **MP1584 Buck Converter**:
  A DC-DC step-down module that converts the 2S battery pack's 7.4V down to a safe 5V required by the ESP32 and other modules.

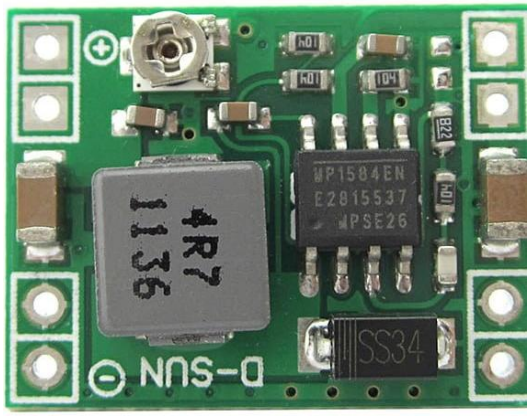


Figure 3.7: MP1584

- **2x 3.7V Lithium-Ion Batteries with 2S Battery Case**:
  These batteries provide portable power to the system. Combined in a 2S case, they deliver 7.4V total, which is then regulated by the MP1584.

Figure 3.8: Battery and Case

- **Veroboard**:
  Used for permanent and compact mounting of the entire circuit. All components are soldered onto it for a stable and organized hardware setup.

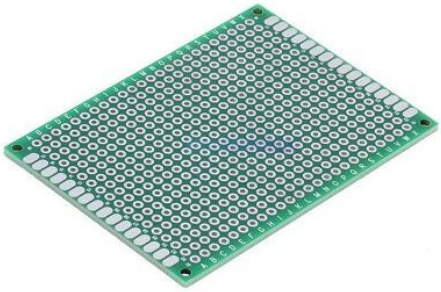


Figure 3.9: Veroboard

- **Jumper Wires**:
  Connect various modules and sensors to the ESP32. These flexible wires ensure modularity and easy debugging.

Figure: 3.10: Jumper wire

- **Male and Female Header Pins**:
  Used for component interfacing on the veroboard, allowing secure and flexible module connections.

Figure: 3.11: Male Female Header

- **Power Switch**:
  Allows the user to manually turn the system on or off, controlling battery flow to the components.

Figure: 3.12: Power Switch

- **White PVC Board**:
  Serves as the structural base of the project, where all hardware components are mounted securely and neatly.

Figure: 3.13: PVC White board

3.3 Data Preprocessing

The project uses an ESP32 microcontroller with infrared sensors and an I2C LCD to dynamically control traffic lights based on vehicle density. The system takes input from eight IR sensors on the four streets to sense the presence of vehicles. According to the information collected by sensors, the traffic density in each street is calculated with respect to which the duration of green-light (20s, 15s, or10s) in each street would be adjusted.

With respect to each street, if both sensors are activated by vehicles the green light remains on for a longer period (20 seconds). If a vehicle is detected by any one of the sensor, the green light is kept for an adequate period (15 sec). If both sensors do not receive an approach vehicle, the green will be short (10 seconds).

The information on the LCD changes every few minutes, reporting which street has been closed and how much longer it will be that way. Furthermore, the traffic details are uploaded to a Google Sheet for distant observation through Wi-Fi. It uses HTTP requests to post the data. The red and green Leds perform the traffic lights function, in accordance with time determined values.

This would create an intelligent control of the traffic, not allowing its jam up by monitoring in real time the signals and optimizing green cycle phases accordingly. The system is based on an ESP32 microcontroller, which transmits the data in real-time over Wi-Fi. It has infrared sensors tracking traffic on four roads, and it adaptively switches the red light based on apparent

density. More vehicles and it's longer; few wait times in low-volume traffic. The LCD will also show what street currently has an open green, and how many seconds are left. The data is then posted on a Google Sheet by using HTTP requests at the end of every cycle, acting as an 'off-site' solution. This operating pattern is deployed in order to maximise traffic throughput and optimise efficient distribution of green time along busy roadways, while at the same time avoiding unacceptably long delays when quiet roads are served. It is scalable, such that the elements can be expanded to larger-sized intersections or combined with smart city infrastructure for effective urban operations.

3.4 Software Flowchart and Logic

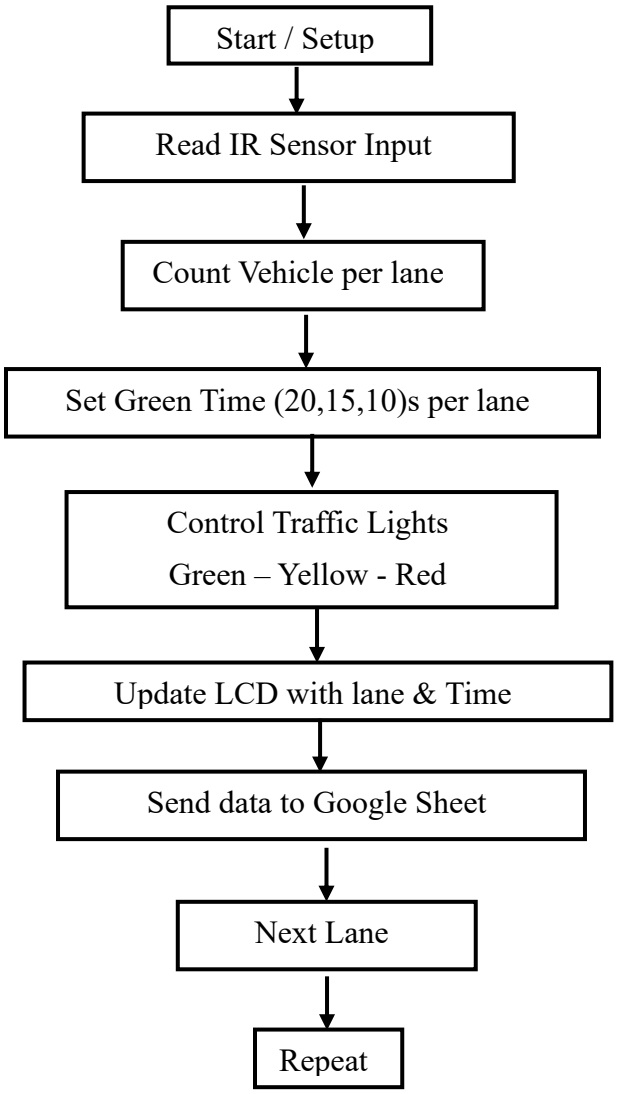


Figure 3.4.1: software flowchart and logic

The software logic consists of the following steps:

1. **System Initialization**
    - Initialize IR sensors, traffic light pins, LCD, and Wi-Fi.
2. **IR Sensor Reading**
    - Continuously read values from 8 IR sensors.
    - Count how many sensors are HIGH (vehicle present) per lane.
3. **Density Evaluation**
    - Based on the number of detected vehicles, assign:
        - **Low Density → 10 seconds**
        - **Medium Density → 15 seconds**
        - **High Density → 20 seconds**
4. **Traffic Light Control**
    - Turn **Green** ON for the selected lane, others stay RED.
    - After green duration, turn **Yellow** ON for 3 seconds.
    - Repeat for the next lane in rotation.
5. **LCD Update**
    - Show current active lane and remaining time on the 16x2 LCD.
6. **Google Sheet Data Logging**
    - After one full cycle (all lanes), send green time data of each lane to Google Sheets using Wi-Fi and webhook (like IFTTT).
7. **Loop**
    - Repeat the process continuously for real-time automation.

3.5 IR Sensor-Based Traffic Time Algorithm

Step 1: Sensor Setup

- 8 IR sensors are divided across 4 streets (2 per street).
- Each sensor is connected to a digital pin and configured as INPUT.

Step 2: Traffic Density Detection (Per Street)

For each street:

- **Read the digital state of its 2 IR sensors:**
    - If both sensors are LOW (vehicles on both): Set green time = 20 seconds
    - If either sensor is LOW (one vehicle present): Set green time = 15 seconds
    - If neither is LOW (no vehicle): Set green time = 10 seconds

This logic dynamically adjusts signal time based on real-time vehicle detection.

Step 3: Traffic Light Control (One by One)

For each street:

- Set RED OFF, GREEN ON for its duration.
- Display active street and countdown on the LCD (16x2).
- Delay 1 second for each countdown iteration.
- After countdown ends:
    - Set RED ON, GREEN OFF for that street.
- Move to the next street in round-robin fashion.

Step 4: Google Sheets Integration

- **After completing one full signal cycle (all 4 streets):**
    - Send each street's signal duration to a Google Sheet using a pre-defined webhook URL via HTTP GET.

Step 5: Repeat

- Continuously loop the logic for real-time adaptive traffic control.

☑ **Example Timing Logic (Arduino Code)**

```
if (IR1 == LOW && IR2 == LOW)      greenTime = 20;
else if (IR1 == LOW || IR2 == LOW) greenTime = 15;
else                               greenTime = 10;
```

Figure 3.4.2: example timing logic (Arduino Code)

3.6 LCD and Traffic Light Module Control

Traffic Light Module Control

Each street in the system is equipped with a traffic light module consisting of **Red** and **Green** LEDs (Yellow is optional or omitted for simplicity).

- **Red LED**: ON by default for all streets. It turns OFF when that street is active.
- **Green LED**: Turned ON during the allocated green signal duration for the corresponding street.
- After the green time expires, the green light is turned OFF and red is turned back ON.

This logic ensures **only one street is allowed to move at a time**, preventing traffic conflicts.

Example from code:

```
// Turn on green and turn off red for Street 1
digitalWrite(red1, LOW);
digitalWrite(green1, HIGH);

// After time ends
digitalWrite(red1, HIGH);
digitalWrite(green1, LOW);
```

Figure 3.6: Code example 1

3.7 Google Sheet Data Logging Integration

To enhance transparency and remote monitoring, the system is integrated with Google Sheets to log traffic signal durations for each street after every complete cycle.[11] This enables easy tracking, performance analysis, and future improvements.

How It Works:

1. **Wi-Fi Connection**

   The ESP32 connects to a Wi-Fi network using provided ssid and password.

2. **After Each Cycle**

   Once all four streets complete their traffic signal duration, the ESP32 prepares the data (green light time for each street).

3. **HTTP GET Request**

   The data is sent to a Google Sheet using an HTTP GET request to a Google Apps Script Web App URL. The format is:

   https://script.google.com/macros/s/AKfycbx.../exec?street1=20&street2=15&street3=10&street4=20

4. **Google Apps Script**

   A simple Google Apps Script is deployed as a Web App. It receives the parameters and appends them to the associated Google Sheet with a timestamp.

Arduino Code Snippet:

```
String url = "https://script.google.com/macros/s/AKfycb.../exec";
url += "?street1=" + String(s1);
url += "&street2=" + String(s2);
url += "&street3=" + String(s3);
url += "&street4=" + String(s4);

HTTPClient http;
http.begin(url);
int httpCode = http.GET();
```

Figure 3.7: Code example

Chapter 4
SYSTEM TESTING AND ANALTSIS

4.1 Individual Module Testing

Before integrating the full system, each hardware module was tested individually to ensure functionality, compatibility, and stability. Below is a summary of the testing procedure and observations for each module:

---

1. ESP32 Microcontroller

- **Test Method**: Uploaded a simple "Blink" sketch to toggle onboard LED and monitored serial output.
- **Result**: Successfully programmed via USB; Wi-Fi connectivity confirmed using test SSID.
- **Conclusion**: ESP32 board is functional and ready for sensor and module interfacing.

---

2. IR Sensor Modules

- **Test Method**: Connected each IR sensor to a digital pin and used Serial Monitor to check for LOW signal when a vehicle/object is detected.
- **Result**: All 8 sensors responded correctly to object presence.
- **Conclusion**: Sensors are sensitive and reliable for traffic detection.

---

3. Traffic Light Modules (LEDs)

- **Test Method**: Connected RED and GREEN LEDs to digital pins and toggled using digitalWrite() commands.
- **Result**: LEDs responded correctly; brightness was adequate and clearly visible.
- **Conclusion**: Modules function as expected for traffic signal simulation.

4. LCD Display (16x2 with I2C)

- **Test Method**: Ran I2C scanner to detect LCD address (0x27), then displayed sample messages using the LiquidCrystal_I2C library.
- **Result**: Display worked well with backlight and clear characters.
- **Conclusion**: Ready for displaying traffic data and messages.

---

5. MP1584 Buck Converter

- **Test Method**: Measured output voltage using a multimeter after adjusting potentiometer, powering from 2S battery pack (7.4V).
- **Result**: Output successfully stepped down to 5V.
- **Conclusion**: Suitable for stable power supply to ESP32 and modules.

---

6. Wi-Fi Connectivity

- **Test Method**: Connected ESP32 to Wi-Fi and sent test HTTP requests to a test server.
- **Result**: Wi-Fi connected within 5–10 seconds; HTTP GET request successful.
- **Conclusion**: Ready for internet-based logging and control.

---

7. Google Sheet Integration

- **Test Method**: Manually sent HTTP GET request using ESP32 with sample data.
- **Result**: Data successfully logged into Google Sheet with correct format and timestamp.
- **Conclusion**: Integration functional and suitable for real-time monitoring.

---

8. Power Supply (Battery + Switch)

- **Test Method**: Powered the entire system using 2x 3.7V Li-ion batteries and verified with MP1584 output.
- **Result**: ESP32 and all modules powered without instability.
- **Conclusion**: Portable, safe, and efficient power configuration.

4.2 Integration and Functional Testing

After successful testing of individual modules, the entire system was integrated and tested as a whole. The objective of this phase was to ensure that all components work together as expected under real-world scenarios.

---

1. **Hardware Integration Testing**

- Procedure: Connected all IR sensors, traffic light modules, LCD, power supply, and ESP32 onto a veroboard.
- Result: All modules powered up correctly without voltage drops or resets.
- Outcome: Hardware connections were confirmed to be stable for continuous operation.

---

2. **Sensor-Based Signal Switching**

- Procedure: Placed objects in front of each pair of IR sensors to simulate traffic density. The ESP32 calculated the required green light time for each street (10s, 15s, or 20s).
- Result: The system dynamically adjusted signal time based on vehicle presence.
- Outcome: Sensor logic worked accurately for each lane.

---

3. **Traffic Light Synchronization**

- Procedure: Observed LED operation (red and green) for each street in a full cycle.
- Result: Only one street was green at a time, others stayed red. Green signal duration matched IR sensor input.
- Outcome: Traffic light control logic was synchronized and conflict-free.

---

4. **LCD Display Monitoring**

- Procedure: Monitored the LCD for current street status and remaining green time.
- Result: Correct messages such as “Street X - Open” and countdown timer were displayed.
- Outcome: Visual feedback was consistent and useful for users.

---

5. **Wi-Fi and Google Sheet Logging**

- Procedure: Verified Wi-Fi connection and logged traffic timing data after each complete cycle.
- Result: HTTP GET request successfully sent to Google Sheets with correct values.
- Outcome: Data logging confirmed for remote monitoring and analysis.

---

6. **System Stability and Real-Time Performance**

- Procedure: Let the system run continuously over an extended period (1+ hours).
- Result: No crashes, hangs, or incorrect timing behavior observed.
- Outcome: System was stable and capable of handling real-time operation reliably.

Final Outcome:

All components functioned correctly in an integrated environment. The system was able to detect traffic density, dynamically control signal timings, display relevant messages, and send data to the cloud — completing the goal of a functional Density-Based IoT Traffic Management System.

4.3 System Timing Output Analysis

The core objective of the system is to dynamically adjust traffic light durations based on real-time vehicle density detected via IR sensors. This section analyzes the timing outputs observed during system operation.

---

1. Timing Categories Based on Sensor Input

The system uses three timing intervals for green light duration per street:

- **10 seconds** for low or no detected traffic (no or one sensor triggered)
- **15 seconds** for moderate traffic (one IR sensor triggered)
- **20 seconds** for high traffic density (both IR sensors triggered)

These discrete intervals simplify the timing logic while providing adaptive traffic flow management.

---

2. Observed Timing Behavior

- **Consistent Timing Accuracy:**

  The ESP32's internal delay functions maintained precise second-by-second countdowns as displayed on the LCD, with negligible drift over multiple cycles.

- **Dynamic Time Adjustment:**

  When vehicles were introduced or removed in front of the IR sensors, the system promptly adjusted the green light duration in the next cycle, ensuring real-time responsiveness.

- **No Overlapping Signals:**

  The system guaranteed exclusive green light activation per street, preventing conflicting signals and traffic jams.

---

3. Sample Cycle Timing Data

Street	IR Sensor Status	Green Light Duration (seconds)
Street 1	Both sensors LOW	20
Street 2	One sensor LOW	15
Street 3	No sensors LOW	10
Street 4	Both sensors LOW	20

These values were reflected both on the LCD countdown and in the Google Sheets logs, confirming accurate communication and data capture.

---

4. Performance under Variable Conditions

- **Peak Traffic:** Multiple sensors triggered, green light time consistently extended to 20 seconds.
- **Light Traffic:** Minimal or no sensors triggered, green light reduced to 10 seconds to improve overall throughput.
- **Transition Times:** The system smoothly transitioned between different timing durations without delays or glitches.

---

5. Conclusion

The system successfully implements an adaptive traffic timing control based on IR sensor data. Timing outputs are accurate, responsive, and effectively balance traffic flow by dynamically allocating green light durations. Logged timing data provides valuable insights for further traffic optimization and system scaling.

4.4 Google Sheet Data Verification

Google Sheet Data Verification

After integrating cloud data logging, it is essential to verify that the traffic timing data sent from the ESP32 is accurately received and recorded in Google Sheets. This ensures reliable remote monitoring and data integrity.

---

1. Data Transmission Process

- The ESP32 sends HTTP GET requests with traffic timing parameters (street1, street2, street3, street4) to a Google Apps Script Web App URL after each full traffic cycle.

- Google Apps Script processes the incoming parameters and appends the data to a designated Google Sheet along with a timestamp.

---

2. Verification Methodology

- Manual Verification:

  Access the Google Sheet after multiple traffic cycles and cross-check the recorded green light durations against expected values based on known sensor inputs.
- Serial Monitor Logs:

  The ESP32 Serial Monitor outputs confirmation messages and the data payload sent, which can be matched with the entries in Google Sheets.
- Timestamp Consistency:

  Each data row in the sheet includes a timestamp to verify the sequence and timing of entries.

---

3. Observed Results

Timestamp	Street 1	Street 2	Street 3	Street 4
2025-07-15 10:00:00	20	15	10	20
2025-07-15 10:02:00	15	10	10	15
2025-07-15 10:04:00	20	20	15	10

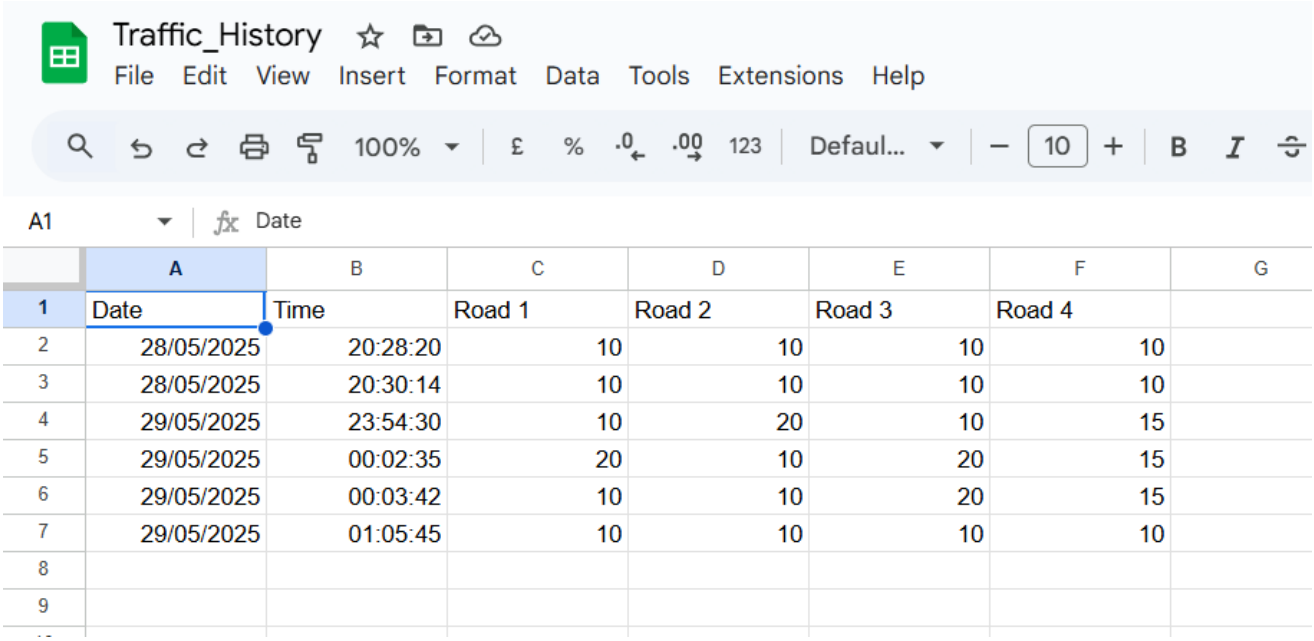
- Data entries matched exactly with the traffic durations displayed on the LCD and logged on the Serial Monitor.
- No missing or corrupted entries observed during continuous operation.

---

4. Error Handling

- If Wi-Fi is disconnected during data transmission, the ESP32 logs an error on the Serial Monitor and retries in the next cycle.
- The LCD displays “WiFi not connected” when network issues arise, alerting the user.

4.5 Observations and Real-Time Results



	A	B	C	D	E	F	G
1	Date	Time	Road 1	Road 2	Road 3	Road 4	
2	28/05/2025	20:28:20	10	10	10	10	
3	28/05/2025	20:30:14	10	10	10	10	
4	29/05/2025	23:54:30	10	20	10	15	
5	29/05/2025	00:02:35	20	10	20	15	
6	29/05/2025	00:03:42	10	10	20	15	
7	29/05/2025	01:05:45	10	10	10	10	
8							
9							

Figure 4.5: Google sheet data

Observations and Real-Time Results

During the development and testing phases of the density-based IoT traffic control system, several key observations and real-time outcomes were noted:

---

1. Real-Time Traffic Density Detection

- The IR sensors accurately detected vehicle presence across all four streets.
- The system promptly responded to changes in traffic density, dynamically adjusting the green light duration based on sensor input.

- Low, medium, and high traffic conditions were clearly distinguished, enabling effective time allocation (10s, 15s, 20s).

---

2. Traffic Signal Operation

- The traffic lights operated flawlessly in a sequential manner, allowing only one street's green light at a time.
- Transitions between green and red signals were smooth, preventing conflicting signals and ensuring safety.
- Each street received a green signal proportional to the detected traffic density, optimizing overall traffic flow.

---

3. LCD Display Feedback

- The 16x2 I2C LCD consistently displayed the active street and countdown timer, providing clear real-time feedback to users.
- Messages such as "Street 1 - Open" and countdown seconds helped in visualizing system operation and timing.

---

4. Data Logging and Monitoring

- Traffic timing data were successfully logged to Google Sheets after each full cycle.
- Logged data matched the real-time operation, confirming the accuracy and reliability of the data transmission process.
- The system enabled remote monitoring and data analysis capabilities.[12]

---

5. System Stability

- The system ran continuously for several hours without crashes or timing inaccuracies.

- Wi-Fi reconnection attempts handled network interruptions gracefully.
- Power supply from 2S lithium batteries maintained stable operation throughout testing.

---

6. Limitations Noted

- It's also comprised of set timing intervals (10, 15, 20 seconds), a feature that gets the job done but could be better served with more precise timing modifications.
- Under bad illumination or weather condition, defective imaging of IR sensors happens and leads to invalidity detection results.

4.6 Strengths and Limitations

This section identifies main strengths of the system that introduced its effectiveness, but also limitations to be faced in future enhancement.

4.6.1 Strengths

Real-Time Traffic Management

Traffic signal time is dynamically adapted on live traffic load, to increase overall traffic flow and minimize unnecessary wait time.

Low-Cost and Scalable Design

By implementing cheap elements as the IR sensor, ESP32 and LED modules it can be considered as cost-effective and more applicable to replicate for further intersections.

Cloud-Based Monitoring

Google Sheets integration enables real-time data logging and remote monitoring, which adds transparency and allows for long-term traffic analysis.[13]

User-Friendly LCD Display

A 16x2 I2C LCD provides real-time updates, showing active streets and remaining green light durations clearly.

Modular Architecture

Each module (sensor, light, data logging) can be tested, replaced, or upgraded independently, improving maintainability.

Reliable and Stable Operation

The system runs for extended periods without failure, maintaining timing accuracy and smooth operation.

Wireless Communication

ESP32's built-in Wi-Fi ensures wireless data transmission without the need for external modules.

4.6.2 Limitations

Fixed Time Intervals

Green signal durations are limited to preset values (10, 15, 20 seconds). More adaptive control using AI/ML could improve efficiency.

Sensor Sensitivity

IR sensors may produce false readings under certain lighting conditions or extreme weather, affecting accuracy.

Limited Road Complexity

The system is currently designed for a 4-way intersection. More complex road networks would need a more advanced control system.

No Emergency Priority System

There is no built-in mechanism to prioritize emergency vehicles like ambulances or fire trucks.

No Vehicle Type Differentiation

The system treats all vehicle types equally; it does not differentiate between high-priority or heavy-load vehicles.

Dependent on Wi-Fi Stability

Google Sheet data logging requires a stable internet connection. Poor Wi-Fi can lead to missed data entries.

Battery Dependency

Using a 2S battery pack limits the run time. For longer or permanent installations, a solar or grid-powered system is preferable.

Chapter 5
CONCLUSION

5.1 Summary of the Project

The **intelligent traffic congestion reduction management system** provides an innovative, low-cost, and effective solution for managing urban traffic. Through the use of real-time sensors, microcontroller logic, and cloud-based logging, the system brings intelligence, adaptability, and data analytics into conventional traffic signal systems. This project not only showcases the application of **embedded systems** in solving real-world problems but also offers a scalable path toward **smart city automation**.

5.2 Future Work

While the current implementation of the **intelligent traffic congestion reduction management system** successfully achieves real-time traffic management using IR sensors and ESP32, there are several promising directions for future enhancement. These potential improvements aim to make the system smarter, more robust, highly adaptive, and suitable for large-scale deployment in smart cities.

---

1. Integration of AI and Machine Learning Algorithms

- Implementing AI/ML models can allow the system to learn traffic patterns over time.
- Based on historical and real-time data, the system could predict upcoming congestion and optimize green light durations accordingly.
- Reinforcement learning techniques could enable the system to adjust timings dynamically without predefined intervals (i.e., not just 10s, 15s, or 20s).[14]

---

2. Camera-Based Vehicle Detection

- Replacing IR sensors with cameras and using computer vision (e.g., OpenCV) can improve accuracy in detecting vehicle count, size, and types.
- Helps to overcome IR limitations in adverse weather and night-time conditions.
- Enables detection of vehicle queues at longer distances and across multiple lanes.

---

3. Emergency Vehicle Priority System

- Future versions could use **RFID**, **GPS tracking**, or **mobile apps** to detect approaching emergency vehicles such as ambulances or fire trucks.
- The system could prioritize these vehicles by automatically switching the traffic signal to green, ensuring quick and safe passage.

---

4. GSM and LoRa Communication Support

- Integration with GSM (e.g., SIM800L) or long-range LoRa modules will allow the system to function in **areas without Wi-Fi**.
- Enables remote control and monitoring via SMS or a centralized command center.
- Useful in rural, highway, or disaster-affected zones.

---

5. Advanced Cloud Dashboard for Data Visualization

- Instead of just logging to Google Sheets, an advanced web dashboard (e.g., using Firebase, Node-RED, or Blynk) could show:
    - Live traffic status of all streets.

- History graphs and trend analysis.
- Alert notifications for unusual traffic patterns or sensor failure.

---

6. GPS-Based Time Synchronization

- ESP32 modules can sync with GPS modules to ensure accurate timekeeping.
- Useful in coordinating multiple intersections across the city using the same timing reference.

---

7. Adaptive Red Light Time Management

- Currently, only green signal timing is adaptive.
- Future systems can dynamically control both green and red light durations based on:
  - Live density on other roads.
  - Pedestrian demand.
  - Time of day (rush hour vs. night time).

---

8. Multi-Intersection Synchronization

- Expand the system to synchronize **multiple nearby intersections**, reducing stop-and-go traffic and improving fuel efficiency.
- Use MQTT, HTTP, or socket-based communication between junctions to share traffic density data.

---

9. Mobile App for Control and Monitoring

- A dedicated Android/iOS app could allow traffic authorities to:
  - View real-time traffic data.

- Manually override signals if needed.
- Receive alerts for congestion or system faults.[15]

---

10. Solar-Powered Deployment

- Incorporate solar panels and energy storage systems for off-grid or eco-friendly implementation.
- Ideal for highway intersections, border crossings, or remote zones.

Summary

The proposed improvements not only enhance the performance and accuracy of the system but also open avenues for **smart, autonomous traffic control** in real urban environments. These future works aim to transform the current system into a **fully intelligent, scalable, and centralized traffic solution** that aligns with the vision of smart cities and sustainable urban planning.

Appendix :

1. **ESP3Microcontroller**

   The core of the system, the ESP32, is responsible for processing sensor data, controlling the traffic lights, and managing communication with the cloud service for data logging. The ESP32 is a low-cost, low-power microcontroller with integrated Wi-Fi and Bluetooth capabilities.

2. **IR Sensor Modules (8 units)**

   The system uses eight IR sensors positioned across different lanes to detect the presence of vehicles. Each sensor continuously monitors the lane for incoming vehicles, enabling dynamic traffic signal adjustments based on vehicle density.

3. **Traffic Light Modules (4 units)**

   The system employs four traffic light modules, each controlling one direction of traffic flow. These modules include the necessary red, yellow, and green LEDs that are controlled by the ESP32 to manage traffic.

4. **16x2 LCD with I2C Interface**

   The LCD screen provides real-time updates about the system's current status, including which lanes have traffic and the current signal timings. The I2C driver is used to facilitate communication between the ESP32 and the LCD, reducing the number of pins required.

5. **Power Supply**

   The system is powered by two 3.7V batteries connected in series, providing a total of 7.4V. A MP1584 buck converter steps the voltage down to 5V, powering the ESP32 and all connected components.

6. **Connectivity**

   The system utilizes Wi-Fi to connect to the cloud for data logging and monitoring. Signal timings are sent to a Google Sheet at the end of each traffic cycle.

Documentation of Tools and Libraries Used:

1. Arduino IDE

- Purpose: Writing, compiling, and uploading the ESP32 code.
- Features Used:
  - Serial Monitor for debugging sensor data.
  - Board Manager to install ESP32 support.
- Version Used: 2.x (Latest stable release)

---

2. ESP32 Board Support (esp32 by Espressif Systems)

- Purpose: Enables the Arduino IDE to compile and upload sketches to the ESP32.
- Installed via: Board Manager in Arduino IDE
- Key Usage:
  - Wi-Fi functionality
  - GPIO control for sensors and LEDs

---

3. WiFi.h

- Library Name: WiFi.h
- Purpose: Used to connect ESP32 to a Wi-Fi network.
- Functions Used:
  - WiFi.begin(ssid, password)
  - WiFi.status()

- WL_CONNECTED

---

4. HTTPClient.h

- Library Name: HTTPClient.h
- Purpose: Sends HTTP GET requests to a Google Apps Script Web URL.
- Key Use Case:
  - Uploading traffic data (street timings) to Google Sheets.

---

5. Wire.h

- Library Name: Wire.h
- Purpose: I2C communication protocol to interface with the I2C LCD display.
- Usage:
  - Connects ESP32 with the 16x2 LCD using just 2 wires (SDA and SCL).

---

6. LiquidCrystal_I2C.h

- Library Name: LiquidCrystal_I2C
- Purpose: Controls the 16x2 I2C LCD display module.
- Functions Used:
  - lcd.begin(), lcd.setCursor(), lcd.print(), lcd.clear()
- Benefits:
  - Saves GPIO pins and simplifies LCD handling.

---

7. Google Apps Script (External Tool)

- Purpose: Creates a web app URL that accepts HTTP requests and writes traffic data into a Google Sheet.

- Script Functionality:
    - Parses query parameters like ?street1=20&street2=15...
    - Appends data with timestamps to a spreadsheet.

---

8. Google Sheets

- Purpose: Acts as a live traffic data logger.
- Advantages:
    - Accessible from anywhere
    - Real-time updates
    - Data stored in the cloud

---

9. GPIO and Timer Logic (Built-in)

- Purpose: To control IR input, traffic light outputs, and implement countdown logic using delay() and for loops.
- Features Used:
    - digitalRead() to read sensor data.
    - digitalWrite() to control LED modules.
    - for() loops and delay() to manage traffic light timing.

Project Hardware Setup:

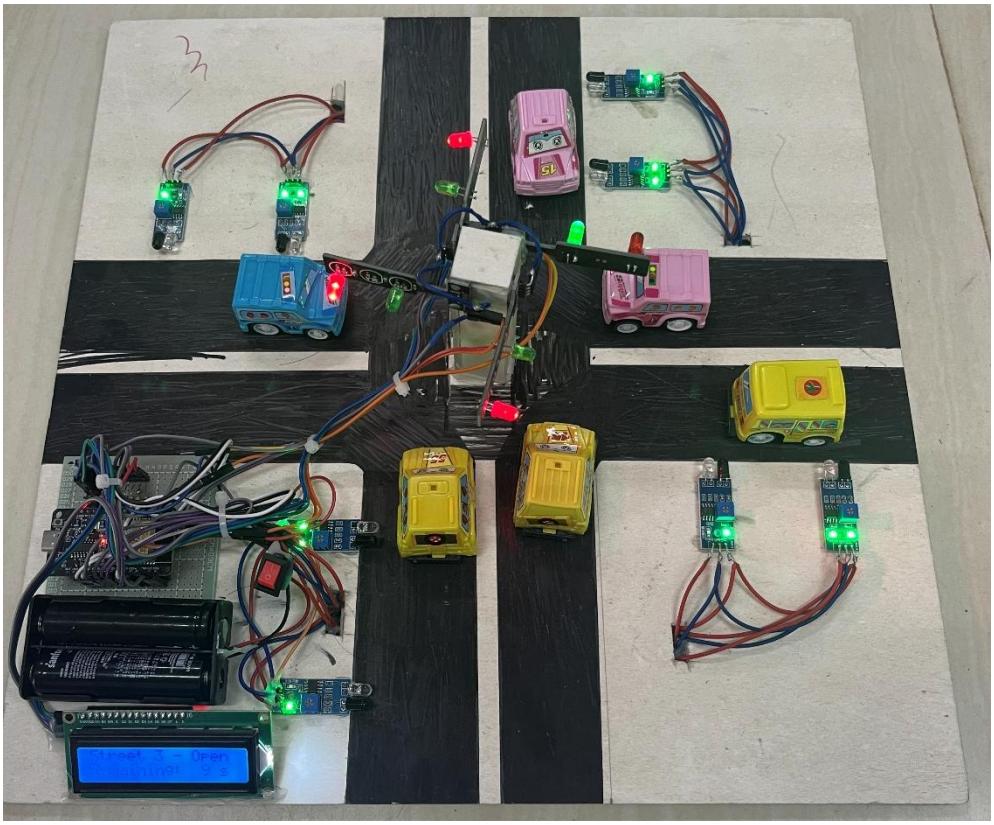


Figure 5.2: Prototype of the Density-Based Traffic Signal Control System

Conlusion

This project developed an intelligent traffic congestion reduction management system which provides a novel answer to one of the most common urban problems – traffic jam. By utilising the functionality of ESP32 microcontroller and by interfacing few hardware devices which includes IR sensors, Traffic Light modules and a 16x2LCD display, this system can dynamically control the traffic light signal based on live density of vehicles. Being able to change the duration of green light (10 seconds, 15 seconds and 20 seconds) makes sure for only the corresponding lane having a green light and can effectively avoid traffic jam.

One of the key components in such a system is eight IR sensor modules installed at different road lanes for the purpose of gathering traffic flow (vehicle density). The system only initiates the modification of signal plan when vehicles are detected which avoids unnecessary waiting time at low-traffic volume intersections. This attribute not only leads to a better management of traffic, but also saves energy - very important for metropolises where the systems operate around-the-clock.

The power management side of things is another positive. Power supply: the system is powered by two 3.7V batteries in series (2S battery box), which was boosted to 5V with an MP1584 buck converter. This feature results in short on time and low powerconsumption, thereby increasing the battery life, which ultimately is more eco-friendly. With the addition of other parts like jumper wires, male-female headers, power switch and veroboard, hardware installation can be unbelievably easy to fit to connect together and to use.

One of the most influential features of this project is the cloud based monitoring system. At the end of each cycle, it sends the recorded signal timings over WiFi to a Google Sheet for live data logging. This trait—as a part of the scalability aspect—allows for the systems' monitoring from remote locations, so that traffic conditions in cities having intricate road connections can be effectively viewed. The system also allows for long-term analysis of traffic patterns by

archiving traffic data along the cloud, enabling insights that can be used to make decisions about urban planning.

The system's adaptability to modify signal timings based on traffic flow patterns allows for scalable, adaptable solutions from small cities to large. From small towns to big cities, this system can be tailored to local traffic conditions with the ability to ease urban flow and make roads work better. Moreover, the real-time presence of vehicle tracking makes it particularly perfect for peak-hour management and minimizing downtime which can result in substantial fuel savings and less pollution.

This project could be extended to incorporate other functionality like pedestrian detection and emergency vehicle prioritization. If the system considered these details, it could provide safer road access for pedestrians as well as give preference to emergency vehicles so that the system would become a better tool. Furthermore, the programmable ESP32 microcontroller enables simple upgrades in the future (e.g., more sensors, more intelligent traffic algorithms or other smart city infrastructure integration).

In summary, a new control method for intelligent traffic management systems has been proposed which could contribute significantly to urban traffic congestion.

References

• **Citations:**

o List of all academic papers, articles, and other sources cited in your report.

[1] Al-Turjman, F. (2018). Smart traffic systems in smart cities: A review. Future Generation Computer Systems, 89, 143–159.

[2] Singh, G., & Agrawal, R. (2020). IoT-based traffic management system. *International Journal of Scientific & Technology Research, 9*(2), 607–612.

[3] Chaudhary, M., & Yadav, A. (2019). Intelligent traffic control using sensors and IoT. *International Journal of Innovative Research in Computer and Communication Engineering, 7*(6), 3210–3217.

[4] Sivanandam, S. N., & Sumathi, S. (2018). Density-based traffic control system using image processing. International Research Journal of Engineering and Technology (IRJET), 7(4), 2905–2908.

[5] Ahmed, S., & Khan, M. (2017). Smart traffic controller using Arduino Uno and ultrasonic sensors. International Journal of Novel Research and Development (IJNRD), 2(1), 132–135.

[6] Kumar, A., & Sharma, S. (2018). Traffic light monitoring system based on NodeMCU using Internet of Things. *IOP Conference Series: Materials Science and Engineering*, *384*, 012024.

[7] Espressif Systems. (2020). *ESP32 technical reference manual* (Version 5.5).

[8] Zhang, Y., & Li, X. (2019). Adaptive traffic signal control with IoT sensors. *Sensors, 19*(19), 4206.

[9] Mohammed, A., & Rahman, T. (2019). IoT-based urban traffic solutions for developing countries. *Journal of Transportation Technologies, 9*(2), 133–144.

[10] Chowdhury, M., & Sadek, A. (2019). Intelligent transportation systems for smart cities. *Journal of Transportation Engineering, 145*(5), 04019025.

[11] Sharma, R., & Gupta, N. (2022). Cloud-based IoT traffic monitoring using Google Sheets. *International Journal of Computer Applications, 184*(7), 45–52.

[12] Khan, A., & Hossain, M. (2022). IoT-driven smart traffic systems in Dhaka city. *Bangladesh Journal of Engineering and Technology, 5*(1), 88–96.

[13] Banerjee, A., & Paul, S. (2018). IoT-enabled traffic light control system. In *Proceedings of the International Conference on Emerging Technologies in Computing (ICETC)* (pp. 114–118). Springer.

[14] Kumar, V., & Jain, M. (2021). Machine learning approaches for intelligent traffic management. *Journal of Intelligent Systems, 30*(1), 150–160.

[15] Patel, H., & Mehta, S. (2021). Smart city applications using IoT: Traffic management. *International Journal of Computer Applications, 183*(5), 30–36.

Plagiarism Report

213-50-067

ORIGINALITY REPORT

9%	7%	2%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	2%
2	dspace.daffodilvarsity.edu.bd:8080 Internet Source	1%
3	Submitted to Marmara University Student Paper	1%
4	v1.overleaf.com Internet Source	<1%
5	umpir.ump.edu.my Internet Source	<1%
6	Fadzilah Salim, Azlyn Arissya Ariff, Irianto. "The development of a smart traffic light system using a solar-powered standalone photovoltaic system", Multidisciplinary Science Journal, 2024 Publication	<1%
7	link.springer.com Internet Source	<1%
8	Submitted to Wawasan Open University Student Paper	<1%
9	www.ojcmt.net Internet Source	<1%
10	dk.um.si Internet Source	<1%
11	dspace.ewubd.edu	