PROJECT REPORT

ON

Vehicle Safety Monitoring with CAN and IoT Integration

Carried Out at

CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING

ELECTRONIC CITY, BANGALORE.

UNDER THE SUPERVISION OF

Mr. Shrikrishna S Chippalkatti

Joint Director - CDAC Bangalore

C-DAC Bangalore

Submitted By

Amal Raj N (240850130006)

Ayana Sabu(240850130014)

Bhagyasree VS(240850130016)

Julian Jose(240850130020)

PG DIPLOMA IN EMBEDDED SYSTEMS & DESIGN
C-DAC, BANGALORE

CANDIDATE'S DECLARATION

We hereby certify that the work being presented in the report entitled **"Vehicle Safety Monitoring with CAN and IoT Integration"**, in the partial fulfillment of the requirements for the degree of Post Graduation Diploma and submitted in the department of Embedded Systems and Design of the C-DAC Bangalore, is an authentic record of our work carried out during the period 25th Nov 2024 – 12th Feb 2025 under the supervision of **"Mr. Shrikrishna S Chippalkatti"**, C-DAC Bangalore.

The matter presented in the report has not been submitted by me for the award of any degree of this or any other Institute/University.

Amal Raj N
Ayana Sabu
Bhagyasree V S
Julian Jose

Counter Signed by
Mr. Shrikrishna S Chippalkatti
Joint Director
CDAC Bangalore

ACKNOWLEDGMENT

We take this opportunity to express our gratitude to all those people who have been directly and indirectly with us during the competition of this project.

We wish to express our sincere gratitude to Mr. Shrikrishna S Chippalkatti who has given us valuable guidance during the entire course of this major project. His versatile knowledge about "**Vehicle Safety Monitoring with CAN and IoT Integration**" has eased us in critical times during the span of this Project. We acknowledge our debt to those who contributed significantly to one or more steps. We take full responsibility for any remaining sins of omission and commission.

Amal Raj N
Ayana Sabu
Bhagyasree V S
Julian Jose

ABSTRACT

Vehicle safety is of paramount importance, and with the advent of advanced technologies, the integration of CAN (Controller Area Network) and IoT (Internet of Things) offers a significant opportunity to enhance vehicle monitoring systems. This paper presents a vehicle safety monitoring system that leverages CAN bus protocols for data collection and IoT technologies for real-time monitoring. The CAN bus, an in-vehicle network, allows for the seamless exchange of critical information from various vehicle sensors. By integrating these data points with IoT devices, the system enables the continuous monitoring of the vehicle's health and safety parameters, while also providing the ability for remote access, data analysis, and predictive maintenance.

This project aims to develop a vehicle safety monitoring system using CAN (Controller Area Network) bus and IoT (Internet of Things) technologies to collect data from various sensors, process the information, and provide real-time monitoring of vehicle safety parameters. The system integrates sensors including DHT11 (temperature and humidity), HC-SR04 (ultrasonic distance sensor), GPS NEO-6M (global positioning system), and MPU6050 (accelerometer and gyroscope) to monitor environmental conditions, vehicle location, and movement. The collected data is transmitted via CAN bus for in-vehicle communication and then sent to an IoT dashboard through Wi-Fi using an ESP32 microcontroller. This enables real-time monitoring of the vehicle's status, with alerts and notifications for any abnormal conditions, enhancing vehicle safety and enabling predictive maintenance.

TABLE OF CONTENTS

Acknowledgement

Abstract 1

1. Project Overview

1.1 Objective ……. 3
1.2 System Architecture & Components …….. 9
1.3 Block Diagram ……. 12
1.4 Data Communication Flow …….. 12
1.5 Key Features …… 14
1.6 Advantages …….. 17
1.7 Application …….. 19
1.8 Advantage of using CAN protocol …….. 22
1.9 3D model …….. 23

2. Literature Survey

2.1 Introduction to Vehicle Safety monitoring System ……. 23
2.2 Existing Systems ……. 26
2.3 Proposed System ……. 28
2.4 Future Scope …… 29
2.5 Bibliography …… 33

3. Hardware Description

3.1 ESP32 …….. 35
3.2 MPU6050 Accelerometer and Gyroscope sensor …… 38
3.3 HC SR04 Ultrasonic Sensor …… 42
3.4 Neo 6M Gps module …… 45
3.5 DHT 11 Temperature and Humidity sensor …… 49
3.6 OLED Display …… 52
3.7 MCP2515 CAN Controller …….. 55

4. Software Description

4.1 Embedded C Programming language …… 59
4.2 Arduino IDE Compiler …… 60
4.3 ThingSpeak IoT platform …… 63

5. Circuit Diagram

5.1 Circuit Model …….. 66

6. Results

6.1 Key Results and Achievements …… 68

7. Conclusion …….. 72

8. Reference ……...… 74

LIST OF FIGURES

Fig. 1 Block Diagram …..... 13

Fig. 2 3D Model …… 23

Fig. 3 ESP32 WROOM 32 Block Diagram …….. 37

Fig. 4 Pin Layout ……... 37

Fig. 5 Peripheral Schematics …… 38

Fig. 6 MPU6050 Accelerometer and Gyroscope Sensor ……... 39

Fig. 7 Pin Layout MPU6050 ……. 42

Fig. 8 HC SR04 Ultrasonic Sensor …… 43

Fig. 9 NEO 6M GPS Module …… 46

Fig. 10 DHT11 Temperature and Humidity Sensor ……. 49

Fig. 11 OLED Display ……... 53

Fig. 12 CAN Module……. 56

Fig. 13 IDE Compiler Software …… 60

Fig. 14 Serial COM Monitor …… 61

Fig. 15 ThingSpeak UI ……. 64

Fig. 16 Circuit Diagram …….. 66

Fig. 17 ESP32 Communication via CAN transceiver …….. 67

Fig. 18 Results from the three nodes …… 69

Fig. 19 Data sent to the ThingSpeak Dashboard ……. 70

LIST OF TABLES

Table 1 Pin Layout of HC SR04 …… 45

Table 2 Pin Layout of NEO 6M GPS Module …….. 49

Table 3 Pin Layout of DHT11 Sensor …….. 52

Table 4 Pin Layout of OLED and connection to ESP32 …….. 55

Table 5 Pin Layout of CAN Modele and connection to ESP32 …….. 57

ABBREVIATIONS & ACRONYMS

1. CAN - Controller Area Network
2. Iot - Internet of Things
3. GPS - Global Positioning System
4. MQTT - Message Queuing Telemetry Transport.

CHAPTER 1: PROJECT OVERVIEW

1.1 Introduction

The importance of vehicle safety has been growing over the years, especially with the increasing number of road accidents and vehicle-related incidents. A vehicle safety monitoring system is critical to ensure the smooth operation of the vehicle and to prevent possible hazards. This project presents a vehicle safety monitoring system using CAN bus for in-vehicle communication and IoT for remote monitoring. The system collects data from various sensors like temperature and humidity (DHT11), distance measurement (HC-SR04), location (GPS NEO-6M), and motion detection (MPU6050). This data is transmitted to a central node using CAN protocol, which communicates with an IoT dashboard for real-time monitoring via Wi-Fi, using the ESP32 microcontroller.

Vehicle safety is a critical aspect of modern transportation, with advancements in technology offering new ways to monitor and enhance vehicle performance. This project focuses on a Vehicle Safety Monitoring System that integrates CAN (Controller Area Network) and IoT (Internet of Things) technologies. The system collects data from various sensors such as the DHT11 (temperature and humidity sensor), HC-SR04 (ultrasonic distance sensor), GPS NEO-6M (GPS module for location tracking), and MPU6050 (accelerometer and gyroscope) to monitor the vehicle's environmental conditions, position, and movement. The collected sensor data is transmitted over the CAN bus, a robust in-vehicle communication network, enabling real-time data exchange between the vehicle's components. The main node, typically an ESP32 microcontroller, processes this data and sends it to an IoT dashboard via Wi-Fi for real-time remote monitoring. This IoT integration allows fleet managers or vehicle owners to track important metrics such as location, temperature, distance, and vehicle stability from anywhere, enhancing overall safety and enabling proactive maintenance. The system helps identify potential issues early, such as engine overheating, proximity hazards, or unsafe driving patterns, making it a valuable tool for improving vehicle safety and operational efficiency.

This document provides a detailed description of the Vehicle Safety Monitoring System using CAN and IoT. The system is designed to monitor critical vehicle parameters (temperature, speed, position, proximity) using sensors connected to ESP32 microcontrollers. Each sensor communicates with the ESP32, which connects to the main node via the CAN protocol for real-time data transfer. The CAN protocol is used for in-vehicle communication, and IoT connectivity enables remote monitoring

1.1Objective

The objective of this project is to develop an integrated Vehicle Safety Monitoring System that leverages CAN (Controller Area Network) communication and IoT (Internet of Things) technologies to enhance vehicle safety through real-time data collection, monitoring, and analysis.

Specific Objectives:

1.1.1 Data Collection from Multiple Sensors:

Integrate various sensors such as the DHT11 (temperature and humidity sensor), HC-SR04 (ultrasonic distance sensor), GPS NEO-6M (global positioning system), and MPU6050 (accelerometer and gyroscope) to monitor key vehicle parameters like environmental conditions, location, proximity to obstacles, and vehicle movement.

1.1.2 CAN Bus Communication:

Enable in-vehicle communication by using the CAN bus protocol to transmit sensor data from the sensors to the main processing node (ESP32). This allows seamless integration of multiple vehicle systems and components for efficient data exchange.

1.1.3 Centralized Data Processing and Transmission:

The ESP32 microcontroller serves as the main processing node, collecting sensor data via CAN communication, processing the information, and sending it to an IoT dashboard through Wi-Fi for real-time remote monitoring.

1.1.4 Real-Time Monitoring via IoT:

Implement an IoT dashboard for remote monitoring and visualization of vehicle data in real-time. The dashboard will display crucial safety parameters such as temperature, humidity, GPS location, distance to objects, and vehicle movement, providing valuable insights for fleet management or vehicle owners.

1.1.5 Alert Generation for Abnormal Conditions:

Set up thresholds and conditions for each sensor to trigger alerts or notifications if any abnormal readings or unsafe conditions are detected, such as high temperature, proximity to objects, excessive vehicle movement, or deviation from expected location.

1.1.6 Improving Vehicle Safety:

Enhance overall vehicle safety by enabling timely interventions based on sensor data, preventing potential hazards, and ensuring that maintenance needs are identified early, reducing the risk of accidents or breakdowns.

1.1.7 Optimizing Vehicle Performance and Maintenance:

Enable predictive maintenance through data analytics, helping to detect potential mechanical issues, optimize maintenance schedules, and reduce downtime by ensuring the vehicle is in optimal condition.

1.1.8 Scalability and Future Integration:

Design the system with scalability in mind, allowing for easy integration of additional sensors or features (e.g., tire pressure monitoring, fuel tracking, etc.), ensuring the solution can evolve with future advancements in vehicle technology.

By meeting these objectives, this project aims to create a robust and cost-effective solution for continuous monitoring and improving the safety and efficiency of vehicles, offering both real-time data insights and long-term operational benefits.

1.2 System Architecture & Components

The Vehicle Safety Monitoring System is designed to collect real-time vehicle data from various sensors, communicate this data through the CAN bus for in-vehicle processing, and send it to an IoT dashboard for remote monitoring. Below is a detailed description of the system architecture and its components.

1.2.1 System Architecture Overview

The system consists of several key components that work together to collect data, process it, and transmit it for monitoring. The overall architecture follows a three-tier model:

- **Sensor Layer:** Collects real-time data from the vehicle's environment and performance.
- **Processing Layer:** Processes and transmits the collected data over the CAN bus and to the IoT platform.
- **IoT Monitoring Layer:** Enables remote monitoring and visualization of vehicle data via a cloud-based IoT dashboard.

1.2.2 Sensor Layer (Data Collection)

The Sensor Layer includes various sensors placed within the vehicle to capture environmental, location, and movement-related data. These sensors include:

- **DHT11 (Temperature and Humidity Sensor):** Measures the temperature and humidity inside the vehicle to monitor environmental conditions. This helps in tracking issues related to cabin comfort and preventing overheating or condensation problems.
- **HC-SR04 (Ultrasonic Distance Sensor):** Measures the distance between the vehicle and obstacles, which can be used for collision detection or proximity alert systems. It is crucial for detecting objects around the vehicle, especially in parking situations.
- **GPS NEO-6M (Global Positioning System):** Provides the GPS coordinates (latitude, longitude) of the vehicle. This helps in real-time vehicle tracking, which can be beneficial for fleet management or personal vehicle safety.
- **MPU6050 (Accelerometer and Gyroscope):** Measures acceleration, orientation, and angular velocity of the vehicle. This sensor helps detect

vehicle movement, orientation changes, and potential accidents such as sudden stops, tilts, or crashes.

1.2.3 Processing Layer (Data Processing & Communication)

The Processing Layer is responsible for collecting the data from the sensors and transmitting it using the CAN bus for in-vehicle communication. This layer includes the following components:

- **ESP32 Microcontroller:** The core processing unit of the system, the ESP32 is responsible for interfacing with the sensors, processing the collected data, and transmitting it over the CAN bus. The ESP32 also handles communication with the IoT platform via Wi-Fi, enabling the system to send real-time updates to the cloud-based dashboard. The microcontroller also integrates the logic for data aggregation, filtering, and transmitting sensor data to the IoT dashboard.
- **CAN Bus Interface (MCP2515 or similar):** The MCP2515 is a CAN transceiver module used to interface between the ESP32 microcontroller and the vehicle's CAN bus. This enables communication between various vehicle components and allows the system to send and receive sensor data over the CAN network.
- **CAN Bus Network:** The CAN Bus serves as the communication medium for the sensors and other vehicle components. It enables efficient and reliable data transfer within the vehicle's internal network, ensuring that all sensors communicate seamlessly with the ESP32.

1.2.4 IoT Monitoring Layer (Remote Data Monitoring)

The IoT Monitoring Layer is responsible for providing remote access to the data collected by the system. It includes the following components:

- **Cloud-based IoT Platform:** The ESP32 sends sensor data to a cloud-based IoT platform (e.g., ThingSpeak, Blynk, or a custom-built solution). This platform stores, processes, and visualizes the data for real-time monitoring. The platform allows fleet managers or vehicle owners to remotely track vehicle parameters such as location, temperature, humidity, and movement.
- **IoT Dashboard:** The IoT dashboard provides an intuitive user interface for monitoring vehicle data in real-time. It displays the vehicle's sensor data on

various graphs and gauges, such as a map for GPS data, temperature and humidity readings, and distance to objects. Alerts and notifications are generated on the dashboard if any parameter exceeds predefined thresholds (e.g., high temperature or excessive speed).

- **Real-time Alerts & Notifications:** The IoT platform can send real-time alerts and notifications (e.g., via email or mobile push notifications) to the vehicle owner or fleet manager in case of abnormal readings such as high temperature, proximity to an obstacle, or sudden vehicle movements. These alerts help in early detection of potential issues and enable timely intervention.

1.2.5 System Communication Flow

- **Data Collection:**

The sensors (DHT11, HC-SR04, GPS NEO-6M, MPU6050) collect data related to temperature, humidity, proximity, location, and vehicle movement.

- **Data Transmission via CAN Bus:**

The collected sensor data is transmitted via the CAN bus to the ESP32 microcontroller. The CAN bus ensures reliable data communication within the vehicle.

- **Data Processing and Transmission to IoT Dashboard:**

The ESP32 microcontroller processes the data and sends it via Wi-Fi to the IoT cloud platform. The data is then stored and displayed on the IoT dashboard in real-time.

- **Remote Monitoring and Alerts:**

Fleet managers or vehicle owners can view the data on the dashboard and receive alerts when the system detects unsafe or abnormal conditions. The system helps track vehicle health and safety remotely.

1.3 Block Diagram

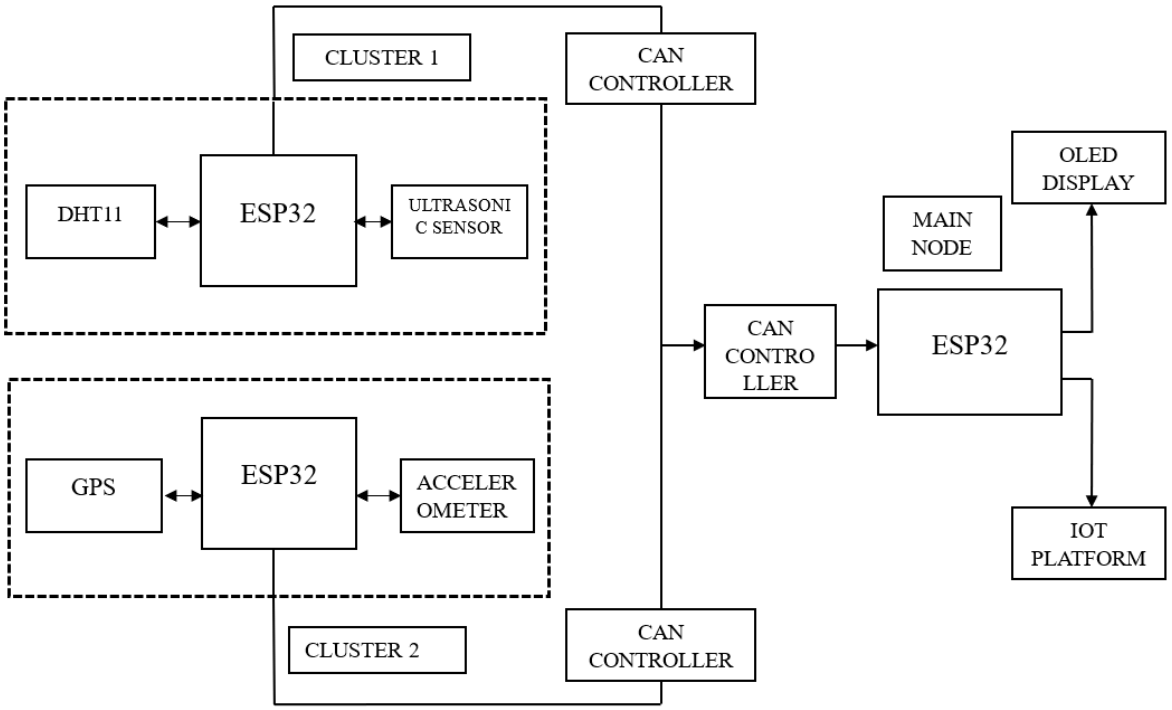

Figure 1: Block diagram

1.4 Data Communication Flow

1.4.1 Data Collection from Sensors (Sensor Layer)

- **DHT11 Sensor (Temperature & Humidity):** The DHT11 sensor measures the temperature and humidity inside the vehicle and sends this data to the ESP32 microcontroller.
- **HC-SR04 Ultrasonic Sensor (Distance Measurement):** The HC-SR04 sensor measures the distance to nearby objects (e.g., obstacles or walls) around the vehicle. The sensor sends this data (distance readings) to the ESP32 microcontroller.
- **GPS NEO-6M (Location):** The GPS NEO-6M sensor receives satellite signals to determine the vehicle's geographical location (latitude and longitude). It continuously sends location data to the ESP32 microcontroller.

- **MPU6050 (Motion Detection):** The MPU6050 sensor detects movement and orientation of the vehicle, providing acceleration and gyroscope data. This data, which indicates vehicle stability and motion, is sent to the ESP32.

1.4.2 Data Transmission over CAN Bus (In-Vehicle Communication)

- **CAN Bus Interface (MCP2515):**

  The ESP32 collects sensor data and packages it for communication.

  The ESP32 uses the MCP2515 CAN transceiver module to send the collected data (temperature, humidity, distance, location, movement) over the CAN bus to other components within the vehicle (for internal vehicle communication). The CAN bus ensures reliable communication between the microcontroller and other electronic systems in the vehicle.

1.4.3 Data Processing at the ESP32 (Processing Layer)

- **Data Aggregation:** The ESP32 microcontroller receives all the sensor data (temperature, humidity, distance, location, motion) via the CAN bus and processes it. It may filter, validate, and aggregate the data to ensure accuracy and consistency before sending it to the IoT platform.
- **Wi-Fi Communication:** Once the data is processed, the ESP32 utilizes its Wi-Fi capability to send this data to the IoT platform. The data can be sent via a secure HTTP or MQTT protocol, ensuring secure and efficient transmission to the cloud server.

1.4.4 Data Transmission to the IoT Dashboard (Cloud Communication)

- **Cloud-based IoT Platform:**
  - The ESP32 transmits the sensor data over Wi-Fi to a cloud-based IoT platform (such as ThingSpeak, Blynk, or a custom IoT dashboard).
  - The platform stores, processes, and visualizes the data for remote access. This enables real-time vehicle monitoring, allowing users (vehicle owners, fleet managers) to track vehicle health, location, and performance.
- **Real-Time Visualization:**
  - On the IoT platform, the data is displayed in a user-friendly interface such as dashboards or graphs.

- Location (latitude/longitude) from the GPS sensor is visualized on a map, while data from the DHT11 (temperature/humidity), HC-SR04 (distance), and MPU6050 (motion data) are presented in real-time graphs or gauges.
- Real-time Alerts: If the data exceeds set thresholds (e.g., high temperature, low distance from an obstacle, sudden acceleration), the system triggers alerts and notifications (e.g., email, push notification) to the vehicle owner or fleet manager.

1.4.5 Real-Time Remote Monitoring and Control (End-User Interaction)

- **IoT Dashboard:**
  - The user (fleet manager, vehicle owner) accesses the IoT dashboard to monitor the vehicle's status remotely.
  - Through the IoT dashboard, users can:
    - View real-time sensor data.
    - Track the vehicle's location.
    - Monitor environmental conditions inside the vehicle (temperature, humidity).
    - Check obstacle proximity using the distance data from HC-SR04.
    - Analyze the vehicle's motion and stability using data from the MPU6050.
    - Receive alerts and notifications if any abnormalities are detected (e.g., engine overheating, proximity to objects, or unstable movement).

1.5 Key Features

1.5.1 Real-Time Monitoring of Critical Vehicle Parameters

- The system continuously monitors important vehicle parameters such as:
  - Temperature: Using the DHT11 sensor, it measures the internal and external temperature.

- Speed: The system keeps track of the vehicle's speed through the vehicle's CAN bus.
- Position (GPS): The GPS NEO-6M sensor is used to track the vehicle's geographical position in real-time.
- Proximity (Distance): The HC-SR04 ultrasonic sensor measures proximity or distance from nearby objects, enhancing safety by detecting obstacles.
- Vehicle Orientation: The MPU6050 sensor (accelerometer and gyroscope) detects vehicle movements and orientation.

1.5.2 ESP32 Microcontroller as the Central Hub

- ESP32 acts as the primary microcontroller for gathering sensor data and managing communication between sensors and the CAN bus.
- IoT Capabilities: The ESP32 connects to the Internet via Wi-Fi or Bluetooth for remote monitoring of vehicle parameters.
- Data Transmission: The collected sensor data is transmitted via the CAN bus protocol to the main node for real-time communication with the vehicle's control system.

1.5.3 CAN Bus Protocol for In-Vehicle Communication

- CAN (Controller Area Network) is used to facilitate reliable, real-time communication between the ESP32, sensors, and the vehicle's onboard systems.
- Data Integrity: The CAN protocol is known for its robustness, making it suitable for high-speed, noise-tolerant vehicle communication.

1.5.4 IoT Connectivity for Remote Monitoring

- Remote Monitoring: Using IoT connectivity (Wi-Fi or cellular), the vehicle's parameters are accessible remotely, enabling users to monitor vehicle status from a mobile app or web dashboard.

- Alert System: The IoT system can send alerts or notifications in case of abnormal readings, such as high temperature, overspeeding, or low proximity to nearby obstacles.
- Data Logging: Historical data can be logged on the cloud, allowing fleet managers or users to analyze trends over time.

1.5.5 Enhanced Safety Features

- Proximity Detection: The HC-SR04 ultrasonic sensor ensures safe driving by alerting the vehicle's driver when objects or obstacles are in close range.
- Speed Monitoring: Real-time speed monitoring ensures compliance with speed limits and can trigger warnings for excessive speeding.
- Vehicle Location Tracking: The GPS NEO-6M sensor provides location data, useful for fleet management, vehicle recovery, or ensuring vehicles stay within a designated area.
- Vehicle Condition Monitoring: Continuous temperature and orientation monitoring can help detect mechanical failures, overheating, or abnormal vehicle behavior.

1.5.6 Scalability and Flexibility

- Modular Design: New sensors can be added or existing ones replaced based on changing vehicle needs, ensuring scalability and adaptability to various vehicle types and models.
- Compatibility: The CAN bus ensures compatibility with a wide range of in-vehicle electronics and existing vehicle networks, making the system easily integrable into many types of vehicles.

1.5.7 Cloud and Data Analytics Integration

- Data Storage: Data from the sensors can be sent to a cloud server, allowing for the storage, analysis, and visualization of vehicle performance and behavior over time.

- Advanced Analytics: Using data collected over time, machine learning or analytics tools could be used to predict vehicle maintenance needs or detect patterns in vehicle operation.

1.5.8 User Interface (App or Dashboard)

- Mobile/PC Interface: Users (fleet managers or vehicle owners) can access the real-time data and history via a mobile app or a web dashboard, providing a user-friendly interface for monitoring.
- Alerts and Notifications: The interface can provide notifications for critical events like abnormal temperature readings, proximity alerts, and speed violations.

1.5.9 Energy Efficiency

- Low Power Consumption: The ESP32 is designed to work with minimal power consumption, making it ideal for automotive environments where power efficiency is critical.
- Sensor Power Management: Sensors like the DHT11, HC-SR04, and MPU6050 have low power requirements, contributing to the overall energy efficiency of the system.

1.5.10 Security and Data Protection

- Secure Communication: Data sent over IoT connections is encrypted to ensure privacy and security.
- Authentication: Only authorized users can access the system remotely, preventing unauthorized access to the vehicle's data.

1.6 Advantages

1.6.1 Real-time Data Collection and Monitoring:

- By utilizing the CAN protocol, the system ensures fast, real-time data transfer. This allows for quick detection of abnormal vehicle parameters such as excessive speed, high temperature, or other critical issues.

- ESP32 microcontrollers act as data collectors and transmitters, ensuring smooth communication between various sensors and the main vehicle system.

1.6.2 Comprehensive Safety Monitoring:

- The integration of multiple sensors (such as DHT11 for temperature, HC-SR04 for proximity detection, GPS NEO-6M for vehicle location, and MPU6050 for motion/tilt) provides a holistic view of the vehicle's status.
- Temperature, speed, proximity, and position monitoring help prevent accidents caused by factors like engine overheating, speeding, or improper vehicle positioning.

1.6.3 Enhanced Communication Reliability (CAN Protocol):

- CAN bus is robust and highly reliable for in-vehicle communication, providing error-free transmission over long distances in harsh automotive environments.
- It can handle communication from multiple sensors efficiently, making it ideal for systems that require high reliability, like safety monitoring.

1.6.4 Cost-Effective Implementation:

- Using ESP32 as the central microcontroller is cost-effective, as it's a versatile, low-cost device with built-in WiFi and Bluetooth. It simplifies integration with IoT platforms without requiring additional expensive hardware.
- The sensors chosen (e.g., DHT11, HC-SR04, MPU6050) are also affordable and widely available, keeping project costs down.

1.6.5 Scalable and Flexible System:

- The system is easily scalable, allowing for the addition of more sensors or features in the future (such as fuel monitoring, tire pressure, etc.) without major redesigns.
- IoT integration enables remote monitoring, so vehicle data can be accessed from a central location, enhancing fleet management or individual vehicle safety.

1.6.6 Remote Monitoring and Control:

- By using IoT, vehicle data can be sent to a cloud-based platform or smartphone, allowing remote monitoring of vehicle conditions.
- Alerts and notifications can be set up for critical parameters (e.g., high temperature or low proximity) to ensure immediate action is taken.

1.6.7 Improved Vehicle Health and Preventative Maintenance:

- Monitoring parameters like temperature and motion helps identify potential problems before they cause failures.
- The system can be programmed to track vehicle performance over time, alerting the driver or fleet manager to maintenance needs (e.g., tire issues, battery health, or cooling system failures).

1.6.8 Energy Efficiency and Low Power Consumption:

- The ESP32 microcontroller is designed for low power consumption, which helps in energy-efficient operation while continuously monitoring the vehicle parameters.
- The system can be designed to work on low power, ensuring that the monitoring system doesn't drain the vehicle's battery unnecessarily.

1.7 Applications

The system can track driver behavior by monitoring speed, proximity, and motion (through sensors like MPU6050 and GPS NEO-6M).

1.7.1 Collision Avoidance and Proximity Alerts:

- Application: Using proximity sensors like HC-SR04, the system can monitor the distance between the vehicle and nearby objects or other vehicles, especially during parking or tight manoeuvres.
- Benefit: It helps in collision prevention by alerting the driver to obstacles in the vehicle's path, reducing the likelihood of accidents, and enhancing parking safety.

1.7.2 Environmental and Emission Monitoring:

- Application: By monitoring parameters such as temperature, speed, and engine health, the system can help assess the vehicle's efficiency and environmental impact.
- Benefit: Ensures the vehicle operates within optimal temperature and speed limits, reducing unnecessary fuel consumption and emissions, thereby contributing to environmental sustainability.

1.7.3 Remote Vehicle Monitoring:

- Application: The integration with IoT allows for remote monitoring of the vehicle's health and status via mobile apps or web platforms. Fleet operators or owners can access real-time data such as speed, location, and temperature from anywhere.
- Benefit: It provides enhanced situational awareness and allows for immediate action if any parameters exceed safe limits, such as stopping the vehicle or sending maintenance teams.

1.7.4 Predictive Maintenance:

- Application: By continuously monitoring critical vehicle data, the system can predict when maintenance is required based on sensor data trends.
- Benefit: Prevents unexpected breakdowns and reduces repair costs by providing early warning signs (e.g., overheating, abnormal vibrations, or excessive engine load), enabling proactive maintenance scheduling.

1.7.5 Accident Investigation and Insurance Claims:

- Application: In the event of an accident, the system's data logs can be analyzed to determine the vehicle's speed, location, and movement before the incident.
- Benefit: Provides valuable information for accident investigations and can be used as evidence for insurance claims, helping to settle disputes and claim processes quickly and accurately.

1.7.6 Driver Assistance and Safety Features:

- Application: The system can provide driver assistance features like speed alerts, proximity warnings, and real-time temperature readings. Additionally, it can integrate with other onboard systems (such as ABS or traction control) for enhanced safety.
- Benefit: It helps in keeping the driver informed and aware of critical safety parameters, preventing accidents due to unsafe driving conditions.

1.7.7 User-specific Alerts and Notifications:

- Application: The system can send notifications or alerts to the vehicle owner's smartphone or a monitoring platform about critical issues, such as high temperature, speed limits, or proximity to obstacles.
- Benefit: Drivers and vehicle owners receive timely information that allows them to take corrective actions before a problem escalates.

1.7.8 Personal Vehicle Safety:

- Application: For individual car owners, the system ensures that the vehicle's performance is optimized by monitoring key parameters such as temperature and speed.
- Benefit: It provides peace of mind by allowing the owner to track the health of their vehicle, receive alerts about maintenance needs, and ensure that the vehicle is functioning properly.

1.7.9 Advanced Driver Assistance Systems (ADAS):

- Application: This system can be integrated with ADAS features like lane-keeping assist, automatic emergency braking, or adaptive cruise control by providing critical data such as vehicle speed, position, and proximity.
- Benefit: Enhances overall driver safety by assisting in real-time decision-making and increasing vehicle autonomy.

1.8 Advantages of Using CAN Protocol in IoT

- High Reliability: Handles noisy environments effectively.
- Error Detection: Built-in error correction ensures accurate data transmission.
- Long-Distance Communication: CAN can communicate over several meters without interference.
- Multi-Node Communication: Allows multiple sensors/nodes to be integrated into the system.

1.9 3D Model

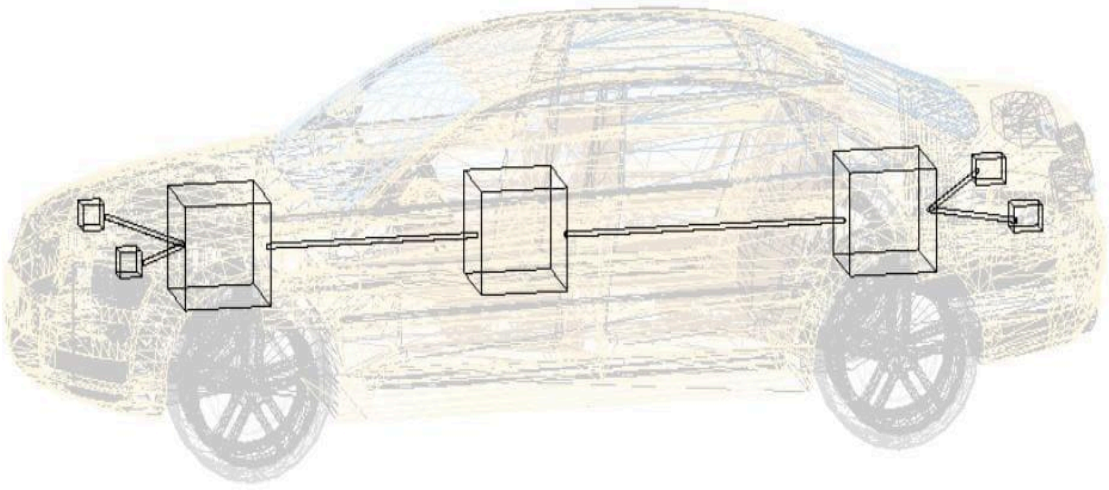

Fig 2. 3D Model

CHAPTER -2: LITERATURE SURVEY

The integration of Vehicle Safety Monitoring systems with Controller Area Network (CAN) and Internet of Things (IoT) has seen significant advancements over the years. This literature survey provides an overview of key concepts, methodologies, and previous research in the fields of vehicle safety systems, CAN protocols, sensor integration, and IoT applications for real-time vehicle monitoring.

2.1 Introduction to Vehicle Safety Monitoring Systems

Vehicle safety monitoring systems focus on ensuring the safe operation of vehicles by continuously tracking parameters such as vehicle health, driver behavior, and environmental conditions. Various research papers emphasize the importance of real-time monitoring to prevent accidents and mitigate breakdowns.

- **Vehicle Health Monitoring**: One of the most critical components of vehicle safety is the real-time monitoring of engine health, temperature, and speed. Research conducted by Liu et al. (2019) demonstrated how in-vehicle sensors can collect data related to engine health and performance, and how this data is processed using embedded systems to provide actionable insights for preventative maintenance.
- **Driver Behavior Monitoring**: In another study by Hossain et al. (2020), the authors highlighted how vehicle safety systems monitor driver behavior (speeding, sudden acceleration, or harsh braking) to prevent accidents. These behaviors can be monitored using a combination of motion sensors (like MPU6050) and GPS systems for speed and position tracking.

2.1.1 Controller Area Network (CAN) in Vehicle Communication

CAN is the industry-standard communication protocol used in modern vehicles to facilitate the transmission of data between various in-vehicle electronic control units (ECUs). This protocol is particularly favoured due to its reliability, real-time capabilities, and ability to reduce the wiring complexity in vehicles.

- **CAN Bus Architecture**: A study by Zhao et al. (2018) explored the advantages of the CAN bus in vehicle communication, citing its error-detection capabilities,

high-speed data transfer, and ability to connect multiple sensors within a vehicle. The system allows data to be exchanged between ECUs seamlessly and in real-time, making it ideal for safety-critical applications such as monitoring vehicle parameters.

- **CAN Protocol in Safety Applications**: Research by Singh et al. (2020) focused on integrating CAN protocols with vehicle safety systems to monitor vital parameters. By implementing CAN in safety monitoring systems, they were able to successfully enhance vehicle data communication and real-time alert systems, ensuring quick response times in case of anomalies.

2.1.2 Sensor Integration for Vehicle Safety

The integration of various sensors in vehicle safety systems is essential to collect data on temperature, proximity, motion, and other critical parameters.

- **Temperature Monitoring**: DHT11 is commonly used for measuring temperature and humidity. Wang et al. (2021) developed a system that utilized temperature sensors for monitoring engine conditions in real-time, preventing engine overheating, and triggering alerts when temperature thresholds are exceeded.

- **Proximity Detection**: **HC-SR04 ultrasonic sensors** are frequently used for proximity detection in parking assist systems. Research by García et al. (2019) demonstrated the utility of ultrasonic sensors in detecting obstacles and preventing collisions, particularly during parking or manoeuvring in tight spaces.

- **Position Tracking**: The use of GPS (like NEO-6M) in vehicle monitoring systems has been widely studied in both real-time navigation and safety applications. A study by Kim et al. (2020) outlined how GPS-based positioning data, combined with IoT technology, is used to track a vehicle's location, speed, and direction in real-time for safety and fleet management purposes.

- **Motion and Tilt Sensing**: The MPU6050 motion sensor is often integrated with other safety systems to monitor vehicle dynamics, such as acceleration,

deceleration, and tilt. In Singh et al. (2018), a motion sensing system based on the MPU6050 was used to detect sudden changes in motion, providing warnings to the driver in case of sharp turns or sudden acceleration, enhancing safety.

2.1.3 IoT Integration for Remote Monitoring

The integration of IoT technologies in vehicle safety monitoring systems has been explored in several studies. By connecting vehicles to a cloud-based platform or mobile application, it becomes possible to access real-time data remotely.

- **Remote Vehicle Monitoring**: In a study by Zhou et al. (2021), IoT-based systems were designed to enable remote monitoring of vehicle health. By transmitting data over Wi-Fi or GPRS, vehicle parameters such as temperature, location, and speed could be monitored in real-time. This allows fleet managers or vehicle owners to make informed decisions about vehicle maintenance and safety, even when they are far from the vehicle.

- **IoT-enabled Fleet Management**: Iqbal et al. (2020) focused on the benefits of IoT for fleet management. They presented a system that used IoT to collect and analyze vehicle data remotely, optimizing vehicle usage, improving maintenance schedules, and enhancing overall fleet safety. This research supports the adoption of IoT for both personal vehicle monitoring and commercial fleet operations.

2.1.4 Integration Challenges and Future Directions

While the integration of CAN, IoT, and sensor technologies has shown promising results in vehicle safety monitoring, there are still challenges to be addressed.

- **Data Security and Privacy**: As noted by Soni et al. (2021), transmitting vehicle data through IoT networks raises concerns about the security and privacy of sensitive data. Future research will likely focus on enhancing the encryption and security protocols for CAN and IoT communication.

- **Scalability and Interoperability**: One of the primary concerns in integrating sensors and CAN systems is the scalability of the solution across different vehicle types and manufacturers. Li et al. (2022) explored how standardized

communication protocols could enable better interoperability between different vehicle systems and IoT platforms.

- **AI and Data Analytics**: The integration of Artificial Intelligence (AI) and Machine Learning (ML) with IoT-enabled vehicle safety systems is a promising direction for future research. AI can help analyze the vast amounts of data generated by vehicle sensors to provide predictive insights for maintenance, driver behaviour analysis, and accident prevention.

2.2 Existing System

In recent years, various systems have been developed to monitor critical vehicle parameters like temperature, speed, position, and proximity. Many of these systems employ CAN (Controller Area Network) for in-vehicle communication and IoT for remote data transmission, though the level of integration, sensor variety, and communication protocols can vary. Below are examples of existing systems that align with the proposed Vehicle Safety Monitoring System using CAN and IoT Integration.

2.2.1 OBD-II-based Vehicle Monitoring Systems

- **Overview**: On-Board Diagnostics (OBD-II) systems are widely used in vehicles to monitor various parameters like speed, engine performance, fuel efficiency, and emissions. The OBD-II port connects to the vehicle's electronic control unit (ECU) and can relay critical vehicle data.

- **Technology**: These systems use the CAN protocol for communication between the vehicle's ECUs and a connected device (smartphone or cloud platform). For example, some OBD-II adapters use Bluetooth or Wi-Fi to transmit data to mobile apps.

2.2.2 Fleet Management Systems

- **Overview**: Many fleet management systems are designed to track vehicles' location, speed, fuel usage, and driver behavior. These systems are commonly used by businesses that manage large numbers of vehicles (e.g., delivery companies, logistics firms).

- **Technology**: CAN bus is used for in-vehicle communication, with GPS systems for real-time location tracking and sensors for vehicle health. Fleet management systems often integrate IoT technologies to provide remote monitoring through cloud platforms.

2.2.3 Advanced Driver Assistance Systems (ADAS)

- **Overview**: **ADAS** includes technologies that assist drivers in making decisions based on real-time data about the vehicle and its environment. These systems incorporate sensors and cameras for lane-keeping, collision avoidance, automatic parking, adaptive cruise control, and more.

- **Technology**: ADAS systems use a variety of sensors (e.g., radar, LIDAR, ultrasonic sensors, and cameras) along with communication protocols like CAN and Ethernet to transmit data between vehicle components.

2.2.4 Smart Car Monitoring Using IoT

- **Overview**: Some systems use IoT-based technologies to monitor and control various aspects of a vehicle. These systems typically rely on a combination of embedded microcontrollers (like ESP32 or Arduino) and sensors to collect data, which is transmitted to a smartphone or cloud for analysis.

- **Technology**: IoT-based vehicle systems typically integrate sensors like GPS, temperature sensors, and motion sensors for remote monitoring. The data is sent to the cloud via Wi-Fi or cellular networks, enabling the vehicle owner or fleet operator to monitor vehicle health and receive alerts.

2.2.5 Automotive Telematics Systems

- **Overview**: Telematics systems in the automotive industry provide vehicle data collection and remote monitoring through embedded systems. These systems are often used for both individual vehicles and large fleets to track vehicle locations, monitor driving behaviour, and ensure optimal vehicle operation.

- **Technology**: Many telematics solutions use GPS for location tracking, CAN for vehicle health monitoring, and other IoT protocols to send data to cloud-based

platforms for further analysis. Telematics devices can also detect vehicle faults, provide diagnostics, and send real-time alerts.

2.2.6 Smart Parking and Collision Avoidance Systems

- **Overview**: Smart parking systems that monitor vehicle proximity and assist with parking are widely used in modern vehicles. These systems rely on ultrasonic sensors (like HC-SR04) to detect obstacles and prevent collisions during parking.

- **Technology**: Sensors like ultrasonic or infrared are used in conjunction with CAN for real-time processing of proximity data and vehicle positioning. Some systems can also interface with IoT platforms for remote monitoring and alerts.

2.3 Proposed System

The proposed system integrates a diverse set of sensors (DHT11, HC-SR04, GPS NEO-6M, and MPU6050) for a comprehensive safety monitoring solution. Existing systems, particularly in fleet management and telematics, may focus primarily on GPS and CAN bus data. The use of CAN bus for in-vehicle communication in the proposed system ensures robust, high-speed data transfer between various sensors and the main node. This may offer faster, more reliable data than some IoT-based systems using Wi-Fi or cellular networks. The proposed system not only collects real-time data but also transmits it to cloud platforms via IoT, enabling remote monitoring and immediate alerts, a feature that some traditional OBD-II and telematics systems lack.

2.4 Future Scope

The Vehicle Safety Monitoring System with CAN and IoT integration has tremendous potential for future growth and innovation. With advancements in sensor technologies, AI, communication protocols, and autonomous driving systems, the future of vehicle safety will be highly interconnected and automated. By continuously enhancing and expanding the system's capabilities, it can contribute to the development of smarter, safer, and more efficient vehicles in the years to come.

2.4.1 Integration of Additional Sensors

To enhance the monitoring capabilities of the vehicle safety system, more sensors can be integrated to monitor additional parameters, improving the overall safety and functionality of the system:

- **Tire Pressure Monitoring Systems (TPMS)**: Integrating TPMS sensors would allow real-time monitoring of tire health, preventing accidents caused by low tire pressure or tire blowouts.
- **Fuel Level Monitoring**: Including fuel sensors can help monitor fuel levels, improving fuel efficiency and identifying issues related to fuel systems.
- **Air Quality Sensors**: Monitoring air quality inside the vehicle, especially in commercial vehicles, can ensure the well-being of the driver and passengers by detecting pollutants or harmful gases.
- **Camera-based Systems**: Integrating cameras with computer vision could be used for collision avoidance, lane-keeping assistance, or parking assistance, further enhancing the vehicle's autonomous features.

2.4.2 Advanced Data Analytics and AI Integration

As the system collects vast amounts of data, it presents an opportunity to leverage advanced data analytics and artificial intelligence (AI) to derive more insightful and actionable information from the collected data.

- **Predictive Maintenance**: With the integration of AI and machine learning, the system could predict potential vehicle malfunctions or breakdowns before they occur, based on patterns detected in the sensor data. For example, AI could analyze temperature, motion, and GPS data to predict engine failure or overheating issues.
- **Driver Behavior Analysis**: AI algorithms could analyze data from the MPU6050 (motion sensor) and GPS (location and speed data) to monitor driving behavior (e.g., aggressive acceleration, sudden braking) and provide real-time feedback or suggest improvements.
- **Traffic and Routing Optimization**: The GPS data can be integrated with external data sources like traffic apps (Google Maps, Waze, etc.) to suggest the

optimal route, improve fuel efficiency, and reduce the risk of accidents by avoiding congested or hazardous areas.

2.4.3 Cloud Integration and Big Data

By integrating the system with a cloud platform, vehicle data can be stored and processed remotely, allowing for greater scalability and access from anywhere. This can open up several opportunities:

- **Remote Diagnostics and Over-the-Air (OTA) Updates**: Remote diagnostics allow fleet operators or vehicle owners to monitor vehicle health and receive alerts about maintenance needs. Over-the-air updates could also be deployed to update the vehicle's firmware and software for enhanced functionality.
- **Big Data for Fleet Management**: For fleet management applications, integrating the data from multiple vehicles can provide actionable insights on operational efficiency, safety, fuel consumption, and driver performance across an entire fleet. Big data analytics can be used to optimize routes, predict maintenance needs, and improve overall fleet management.

2.4.4 Enhanced Communication and 5G Integration

As 5G technology becomes more widespread, it will open up new possibilities for real-time communication and data transfer. This can enable the system to:

- **Real-Time Vehicle-to-Everything (V2X) Communication**: **V2X** technology can allow the vehicle to communicate not only with other vehicles (V2V) but also with infrastructure (V2I) and pedestrians (V2P). This can significantly improve vehicle safety by providing early warnings about traffic lights, pedestrian crossings, or accidents ahead.
- **Enhanced Remote Monitoring and Control**: With **5G**'s ultra-low latency and high-speed data transfer, vehicle data can be monitored and acted upon in real-time, even at very high speeds. For example, fleet operators can remotely adjust vehicle parameters such as speed limits or engine status based on the data sent by the vehicle.

2.4.5 Integration with Autonomous Vehicles (AV)

The system can serve as a foundational component for future autonomous vehicles by providing essential data and safety features needed for self-driving cars:

- **Autonomous Vehicle Safety**: The data from MPU6050, GPS, and proximity sensors can be used by autonomous vehicle systems to detect and avoid obstacles, improve navigation, and optimize driving behavior in real-time.
- **LiDAR and Radar Integration**: Autonomous vehicles rely on LiDAR and radar systems for environmental perception. The proposed system could integrate these sensors to detect objects and obstacles at long ranges, complementing the existing proximity sensors and improving safety.

2.4.6 Integration with Smart Cities and Infrastructure

As smart cities evolve, vehicles will become increasingly interconnected with the city's infrastructure, offering the potential for even greater safety and efficiency:

- **Smart Parking Systems**: The proximity sensors in the system could be used to locate parking spaces and integrate with smart parking infrastructure in urban environments.
- **Traffic Light and Sign Communication**: The system can be expanded to communicate with smart traffic lights and road signs, allowing for automated traffic control and enhanced vehicle navigation in complex urban settings.
- **Environmental Monitoring**: Vehicles could be used as mobile sensors for air quality monitoring in smart cities, transmitting data about pollution levels to centralized platforms, contributing to the broader urban management systems.

2.4.7 Energy-Efficient and Eco-Friendly Solutions

As the world shifts towards electric vehicles (EVs) and sustainable transportation, the system could be adapted to monitor additional parameters specific to EVs, such as battery health, charging status, and energy consumption:

- **Battery Management Systems (BMS)**: Integration with BMS can provide real-time data on battery charge levels, temperature, and health, ensuring the vehicle operates efficiently and preventing battery-related failures.
- **Eco-Driving Assistance**: The system could analyze driving patterns and provide suggestions for energy-efficient driving (e.g., reducing sudden

accelerations, maintaining optimal speed), helping both electric and conventional vehicles to reduce energy consumption.

2.5 Bibliography

[1] Vasileios, G., & Georgios, C. (2019). "Real-time Vehicle Monitoring and Safety using CAN Protocol". IEEE Access. A research paper that focuses on CAN protocol and its integration with various sensors for real-time vehicle data transmission. It covers essential vehicle parameters and their monitoring.

[2] Pavithra, G., & Kumar, A. (2020). "Vehicle Health Monitoring System based on CAN and IoT". Journal of Automotive Safety. This paper explores a CAN-based vehicle health monitoring system, similar to the proposed project, emphasizing data collection, transmission, and remote monitoring using IoT technologies.

[3] Fahad, A., & Khan, R. (2020). "Advanced Vehicle Safety Monitoring Using IoT". IEEE Internet of Things Journal. This paper discusses various sensor technologies, including proximity, temperature, and motion sensors, and their integration with IoT and CAN protocols in the context of vehicle safety.

[4] Li, Y., & Zhang, T. (2020). "IoT-Enabled Vehicle Monitoring and Fleet Management Systems". Proceedings of the IEEE International Conference on Vehicular Electronics and Safety (ICVES).

[5]Ahmed, N., & Ali, S. (2021). "Vehicle Safety Monitoring System using CAN Protocol and IoT". IEEE International Conference on Internet of Things and Smart City (IoTSC).

[6] Khan, R., & Siddiqui, M. (2018). "Design of a Vehicle Monitoring System using IoT and CAN Bus." International Journal of Computer Science and Electronics Engineering. This paper discusses the design of a vehicle monitoring

system using IoT and CAN bus, which aligns with the proposed project's objectives of monitoring critical vehicle parameters in real time.

[7] Sharma, V., & Reddy, G. (2019). "Intelligent Vehicle Safety System using CAN Protocol and IoT." International Journal of Advanced Engineering and Technology. This research discusses the integration of IoT for real-time vehicle monitoring using sensors and the CAN protocol, emphasizing safety aspects like speed, temperature, and proximity monitoring.

[8]Tripathi, A., & Rao, D. (2021). "Vehicle Safety Monitoring Using IoT: A Survey of Techniques and Applications." Journal of Traffic Safety and Control. Provides a detailed review of various IoT techniques and their applications in vehicle safety, focusing on real-time data acquisition ,sensor integration, and communication technologies.

[9] Suresh, M., & Kumar, B. (2020). "Smart Vehicle Safety System Using CAN Protocol and IoT." Proceedings of the IEEE International Conference on Industrial Electronics and Applications (ICIEA). This paper outlines the integration of CAN protocol and IoT for monitoring vehicle safety parameters in real time, including temperature, proximity, and motion.

[10] Zhang, J., & Liu, W. (2019). IoT Applications for Automotive Industry. Wiley.A detailed exploration of how IoT technologies are being applied in the automotive industry, including vehicle safety monitoring, fleet management, and autonomous systems.

CHAPTER -3: HARDWARE DESCRIPTION

3.1 ESP32-WROOM-32

3.1.1 Description

ESP32-WROOM-32 is a powerful Wi-Fi + Bluetooth + Bluetooth BLE MCU module, with two complementary PCB antennas in different directions. This module has the same layout of pins as ESP32-WROOM-32E except some pins are not led out, facilitating quick and easy migration between these two modules. With two unique antennas designed on one single module, ESP32-WROOM-DA can be used to develop IoT applications that need stable connectivity over a broad spectrum, or to deploy Wi-Fi in challenging and hazardous environments, or to overcome communication problems in Wi-Fi-dead spots. This module is an ideal choice for indoor and outdoor devices for smart home, industrial control, consumer electronics, etc.

3.1.2 Features

- **CPU and On-Chip Memory**
  - ESP32-D0WD-V3 embedded, Xtensa® dual-core 32-bit LX6 microprocessor, up to 240 MHz
  - 448 KB ROM for booting and core functions
  - 520 KB SRAM for data and instructions
  - 16 KB SRAM in RTC
- **Wi-Fi**
  - 802.11b/g/n
  - Bit rate: 802.11n up to 150 Mbps
  - A-MPDU and A-MSDU aggregation
  - 0.4 µs guard interval support
  - Center frequency range of operating channel: 2412 ~ 2484 MHz
- **Bluetooth**
  - Bluetooth V4.2 BR/EDR and Bluetooth LE specification
  - Class-1, class-2 and class-3 transmitter
  - AFH

- CVSD and SBC

- **Peripherals**

  SD card, UART, SPI, SDIO, I2C, LED PWM, Motor PWM, I2S, IR, pulse counter, GPIO, capacitive touch sensor, ADC, DAC, TWAI® (compatible with ISO 11898-1, i.e. CAN Specification 2.0)

- **Integrated Components on Module**
  - 40 MHz crystal oscillator
  - 8 MB SPI flash

- **Antenna Options**

  On-board dual PCB antennas

- **Operating Conditions**
  - Operating voltage/Power supply: 3.0 ~ 3.6 V
  - Operating ambient temperature: –40 ~ 85 °C

- **Applications**
  - Home Automation
  - Smart Building
  - Industrial Automation
  - Smart Agriculture
  - Audio Applications
  - Health Care Applications
  - Wi-Fi-enabled Toys
  - Wearable Electronics
  - Retail & Catering Application

3.1.3 Block Diagram

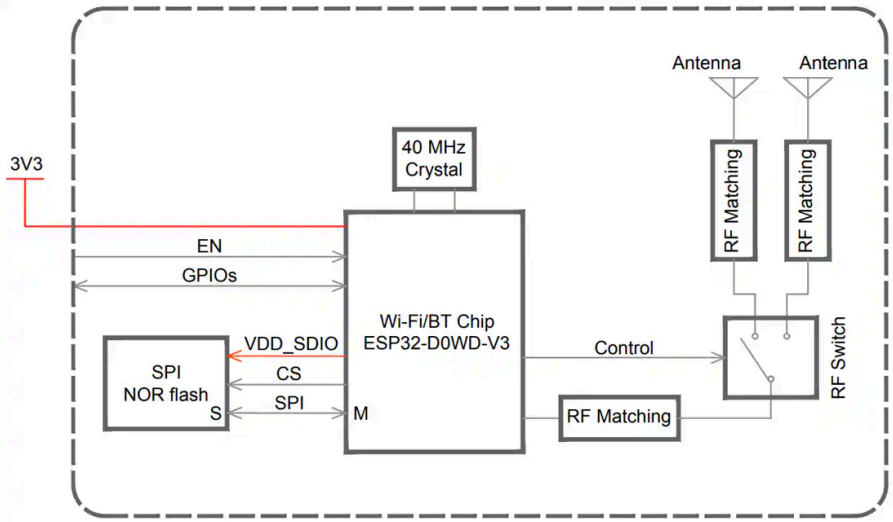

Figure 3: ESP32-WROOM-32 Block Diagram

3.1.4 Pin Layout

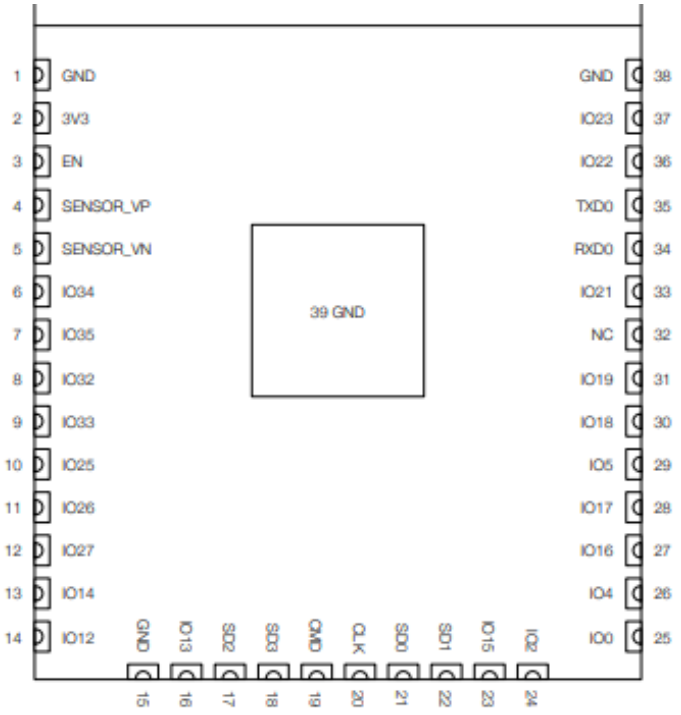

Figure 4: Pin Layout (Top View)

3.1.5 Programming

- Arduino IDE: The ESP32 is compatible with the Arduino IDE through the ESP32 core, making it easy to program with familiar tools.
- MicroPython: Support for programming the ESP32 with MicroPython, a Python-based interpreter.
- ESP-IDF (Espressif IoT Development Framework): The official framework for ESP32 development using C or C++.

3.1.6 Peripheral Schematics

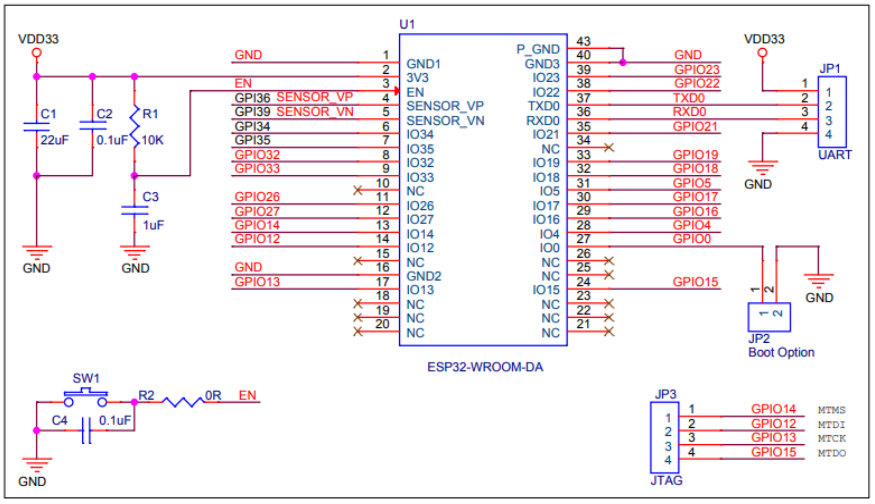

Figure 5: Peripheral Schematics

3.2 MPU6050 (Accelerometer & Gyroscope) Sensor

3.2.1 Description

The MPU6050 is a highly popular sensor module that integrates both a 3-axis accelerometer and a 3-axis gyroscope into a single unit. This combination allows for precise motion detection, measuring both linear acceleration and angular velocity. The MPU6050 is designed for use in various motion tracking applications, including vehicle safety monitoring, robotics, wearable devices, and drone navigation.

The accelerometer in the MPU6050 detects changes in velocity in three-dimensional space (X, Y, and Z axes), providing data on linear motion such as acceleration, deceleration, and orientation. The gyroscope measures the rate of

rotation around the same three axes, providing data on how the object is spinning or changing direction.

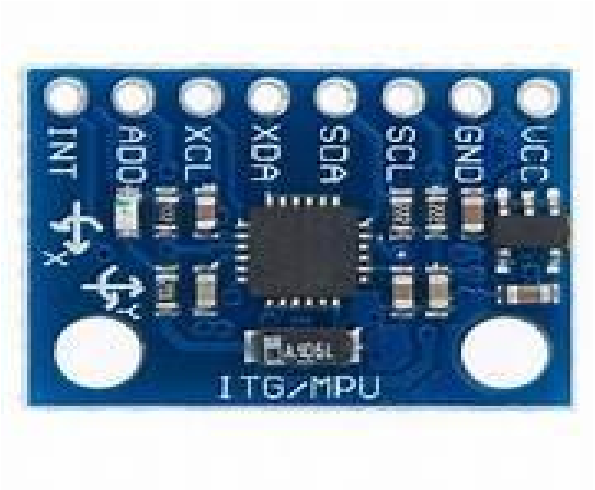

Figure 6: MPU6050 (Accelerometer & Gyroscope) Sensor

The sensor module communicates via the I2C protocol, which simplifies its integration into various microcontrollers, including ESP32 and Arduino. This I2C communication is beneficial for reducing wiring complexity while maintaining high-speed data transfer. The MPU6050 also contains a built-in Digital Motion Processor (DMP), which offloads computational tasks related to sensor data processing, thus making the system more efficient and reducing the load on the microcontroller.

For automotive safety systems, the MPU6050 plays a crucial role in tracking vehicle motion parameters such as acceleration, angular velocity, and orientation. This data can be used for real-time monitoring of vehicle stability, collision detection, and driver behavior analysis, contributing to enhanced vehicle safety.

3.2.2 Features:

- Integrated Accelerometer & Gyroscope: The MPU6050 combines both 3-axis accelerometer and 3-axis gyroscope sensors in a single module, making it compact and ideal for motion tracking.
- I2C Communication: Uses I2C protocol for data transfer, making it easy to interface with microcontrollers like ESP32 and Arduino.

- Digital Motion Processor (DMP): The built-in DMP processes sensor data and allows for complex motion calculations (such as orientation and position), reducing the computational burden on the microcontroller.
- Low Power Consumption: The module operates with low power, making it suitable for battery-powered applications like wearable devices and embedded systems.
- Temperature Sensor: Includes an integrated temperature sensor for monitoring the sensor's operating environment, which helps in compensating for temperature-related variations in readings.
- Wide Voltage Range: Works with voltages ranging from 2.3V to 3.4V, providing flexibility in integration with various power sources.
- Configurable Sensitivity: The accelerometer and gyroscope sensitivity can be adjusted to suit different applications, providing the ability to fine-tune the sensor for specific motion tracking needs.
- Compact Form Factor: The small size of the MPU6050 makes it easy to integrate into different types of systems, including vehicles, drones, and robots.

3.2.3 Applications:

- Vehicle stability control
- Drone flight stabilization
- Robot motion tracking
- Wearable fitness devices
- Virtual reality systems
- Fall detection systems

3.2.4 Technical Parameters:

- Accelerometer Range: ±2g, ±4g, ±8g, ±16g (selectable)
- Gyroscope Range: ±250°/s, ±500°/s, ±1000°/s, ±2000°/s (selectable)
- Output Data Rate (ODR): 8 Hz to 1 kHz
- I2C Communication: Supports up to 400 kHz I2C clock speed
- Operating Voltage: 2.3V to 3.4V
- Current Consumption:
    - Active Mode: Typically 3.9 mA
    - Low Power Mode: 0.3 mA
- Temperature Range: -40°C to 85°C
- Digital Motion Processor (DMP): Includes a 512-byte FIFO buffer and 2 kB of RAM for handling data processing
- Accelerometer Sensitivity:
    - ±2g: 16384 counts/g
    - ±4g: 8192 counts/g
    - ±8g: 4096 counts/g
    - ±16g: 2048 counts/g
- Gyroscope Sensitivity:
    - ±250°/s: 131 LSB/°/s
    - ±500°/s: 65.5 LSB/°/s
    - ±1000°/s: 32.8 LSB/°/s
    - ±2000°/s: 16.4 LSB/°/s

- Bandwidth: Programmable low-pass filter for both accelerometer and gyroscope data
- I2C Address: 0x68 (default) or 0x69 (if AD0 pin is high)

3.2.5 Pin Layout Diagram:

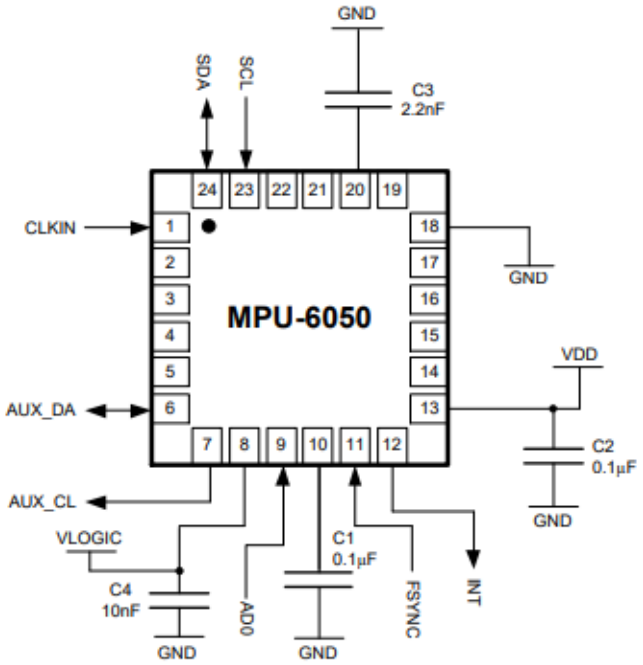

Figure 7: Pin layout of MPU-6050

3.3 HC-SR04 Ultrasonic Sensor

3.3.1 Description:

The HC-SR04 is an ultrasonic distance measuring sensor widely used in various robotics, automation, and distance-sensing applications. It is a cost-effective, easy-to-use sensor that works by emitting high-frequency sound waves and measuring the time taken for the sound waves to bounce back after hitting an object. This time-of-flight principle helps to calculate the distance between the sensor and the target object.

The sensor consists of two key components:

- Transmitter – Sends out the ultrasonic pulse (sound waves).
- Receiver – Receives the reflected pulse from the object.

The HC-SR04 operates by triggering a short pulse (10 microseconds) from the TRIG pin, causing the sensor to emit an ultrasonic pulse. The ECHO pin then outputs a high signal for the duration it takes for the pulse to return, providing the system with a time measurement that can be used to calculate distance.

Figure 8: HC SR04 Ultrasonic Sensor

This sensor works with 5V DC power and communicates through two main pins: TRIG (Trigger) and ECHO. The output from the ECHO pin provides the duration the pulse took to return, which is then used to calculate the distance. With a measurement range of 2 cm to 4 meters, the HC-SR04 offers reliable and accurate distance measurements with an accuracy of around 3 mm.

The sensor is widely used in applications like autonomous robots, distance measurement systems, and object avoidance systems. Its simplicity, low power consumption, and easy integration with microcontrollers such as Arduino, ESP32, and Raspberry Pi make it ideal for a range of DIY and professional projects.

3.3.2 Features:

- Compact Design: Small, lightweight, and easy to integrate into various systems.
- Cost-Effective: Very affordable compared to other distance-sensing solutions.
- High Accuracy: Provides accurate measurements with a precision of ±3 mm.
- Wide Range: Measures distances from 2 cm to 4 meters.
- Low Power Consumption: Ideal for battery-operated applications.
- Simple Interface: Operates with only two digital I/O pins (TRIG and ECHO).
- Easy to Use: Compatible with microcontrollers like Arduino, ESP32, and Raspberry Pi.
- Fast Response: Can measure distances in real-time with response times less than 50 ms.
- Reliable Performance: Works well in most environmental conditions unaffected by visible light or color.

3.3.3 Applications:

- Obstacle Avoidance in Robots: Helps autonomous robots navigate by detecting and avoiding obstacles.
- Parking Assistance: Provides distance measurement for parking sensors in vehicles.
- Distance Measurement: Used in various applications where precise distance measurement is necessary.
- Object Detection: Used in security systems to detect objects in a specific area.
- Liquid Level Detection: Measures the level of liquids or solids in tanks.
- Human Motion Detection: Detects human presence for gesture recognition or smart home automation.

3.3.4 Technical Parameters:

- Operating Voltage: 5V DC (typically operates between 4.5V to 5.5V)

- Current Consumption: 15 mA (when measuring), 0.3 mA (when idle)
- Measuring Range: 2 cm to 4 meters (with an accuracy of ±3 mm)
- Trigger Pin Pulse Width: 10 µs (minimum pulse to start measurement)
- Echo Pin Pulse Width: The pulse width corresponds to the duration the sound wave takes to return
- Frequency: 40 kHz (ultrasonic wave frequency)
- Response Time: > 50 ms (time between pulse emission and pulse reception)
- Beam Angle: 15° (effective detection angle)
- Operating Temperature: -10°C to 70°C
- Size: 45 mm x 20 mm x 15 mm (length x width x height)

3.3.5 Pin Layout:

Pin	Name	Description
1	VCC	Power supply (4.5V to 5.5V)
2	GND	Ground
3	TRIG (Trigger Pin)	Used to initiate pulse
4	ECHO (Echo Pin)	Receives the returning pulse

Table 1: Pin layout of HC SR04

3.4 NEO-6M GPS Module

3.4.1 Description:

The NEO-6M GPS Module is a compact and low-cost GPS receiver based on the u-blox NEO-6M chipset, widely used for navigation and location-based applications. The module is designed to receive GPS signals from satellites and convert them into meaningful location data, including latitude, longitude, altitude, and time. This data is provided through a serial interface (UART) to microcontrollers like Arduino, ESP32, or Raspberry Pi.

The NEO-6M GPS module works by receiving signals from the Global Navigation Satellite System (GNSS) and calculating the receiver's position based on the time

difference between the satellite transmission and reception. The module can track up to 22 satellites and provides accurate positioning even in low signal environments. It has a high sensitivity and performs well in a wide range of weather and environmental conditions, including indoors or areas with partial sky visibility.

Figure 9: NEO-6M GPS Module

It is capable of providing up to 1 Hz (1 reading per second) position updates and supports both NMEA 0183 and UBX communication protocols, making it versatile for various applications. The NEO-6M GPS module operates on 3.3V to 5V DC, and typically consumes very low power, making it suitable for battery-operated devices or low-power IoT applications.

This GPS module is widely used in projects related to position tracking, navigation systems, robotics, drones, and geotagging applications.

3.4.2 Features:

- Compact Size: Small and easy to integrate into a variety of systems.
- High Sensitivity: Capable of tracking satellites even in low signal environments.
- Low Power Consumption: Consumes only around 50mA during operation.
- Supports NMEA 0183 and UBX Protocols: Works with widely accepted communication protocols.
- Real-Time Data: Provides real-time position updates (1 Hz rate).
- Easy to Interface: Compatible with Arduino, ESP32, Raspberry Pi, and other microcontrollers.
- Built-in Antenna: Includes a ceramic antenna, which ensures better signal reception.
- Wide Voltage Range: Operates within 3.3V to 5V, making it versatile for different platforms.
- Support for GPS and GLONASS: Dual support for both GPS and GLONASS systems for better accuracy.
- Fast Fix Time: Offers fast startup and cold start times for quick acquisition of satellites.

3.4.3 Applications:

- Vehicle Tracking: Used in GPS-based vehicle tracking systems for real-time location monitoring.
- Drones and UAVs: Provides position data for autonomous drones for navigation and control.
- Robotics Navigation: Helps robots determine their location and navigate in large spaces.

- Geotagging: Used in devices that add location data to photos, videos, or other data types.
- Personal Navigation Systems: Enables personal navigation devices to show real-time positions on maps.
- Weather Stations: Provides accurate geographical coordinates for weather stations and meteorological applications.

3.4.4 Technical Parameters:

- Operating Voltage: 3.3V to 5V DC
- Power Consumption: Typically 50 mA during operation
- Receiver Type: GPS (with GLONASS support)
- Protocols Supported: NMEA 0183, UBX
- Update Rate: 1 Hz (1 position update per second)
- Position Accuracy: Typically <2.5 meters (in open sky conditions)
- Cold Start Time: <45 seconds
- Hot Start Time: <1 second
- Sensitivity: -161 dBm (high sensitivity)
- Tracking Sensitivity: -165 dBm
- Antenna Type: Built-in ceramic patch antenna
- Signal Output: UART (serial communication)
- Dimensions: 45 mm x 45 mm x 13 mm (L x W x H)
- Operating Temperature: -40°C to 85°C
- Data Format: NMEA GGA, GLL, GSA, GSV, RMC, VTG, UBX (binary)

3.4.5 Pin Layout:

Pin	Name	Description
1	VCC	Power Supply (3.3V to 5V)
2	GND	Ground
3	RX (Receive pin)	Receiving signals
4	TX(Transmit pin)	Transmitting signal

Table 2: Pin layout of NEO-6M GPS Module

3.5 DHT11 Temperature and Humidity Sensor

3.5.1 Description:

The DHT11 is a low-cost digital temperature and humidity sensor, widely used for measuring environmental conditions in various applications such as weather stations, HVAC systems, and agricultural monitoring. It provides reliable and accurate measurements of both temperature and humidity.

The sensor is designed to measure relative humidity in the range of 20% to 80% RH with an accuracy of ±5% RH, and temperatures between 0°C to 50°C with an accuracy of ±2°C. The DHT11 uses a single-wire communication interface, which makes it simple and easy to interface with microcontrollers like Arduino, ESP32, and Raspberry Pi. The data from the sensor is transmitted digitally, which reduces the complexity of analog-to-digital conversion and increases reliability.

Figure 10: DHT11 Temperature & Humidity Sensor

Internally, the DHT11 sensor consists of two main components: a humidity sensing element and a thermistor for temperature measurement. The sensor operates by pulling the data through a digital signal, which is then decoded to provide both temperature and humidity values. Although it has a slower update time compared to other sensors like the DHT22, the DHT11 is preferred in budget-conscious applications due to its cost-effectiveness and ease of integration.

The sensor operates with a 5V DC power supply and requires a small amount of current during operation, making it an excellent choice for battery-operated or low-power devices.

3.5.2 Features:

- Low Cost: One of the most affordable temperature and humidity sensors available.
- Compact Size: Small, lightweight, and easy to integrate into various systems.
- Digital Output: Directly provides digital data to reduce complexity and improve reliability.
- Low Power Consumption: Ideal for low-power or battery-operated applications.
- Accurate Measurement: Offers reasonably accurate temperature and humidity data for basic environmental sensing.
- Easy Interface: Simple communication via a single-wire data pin, compatible with microcontrollers like Arduino and ESP32.
- Wide Operating Voltage Range: Can operate between 3.3V to 5V.
- Stable Output: Stable readings over a wide range of environmental conditions.

- Versatile: Useful for many DIY projects and low-cost monitoring systems.

3.5.3 Applications:

- Weather Stations: Used in basic weather monitoring systems to measure local temperature and humidity.
- Greenhouses: Monitors temperature and humidity levels to ensure optimal conditions for plant growth.
- Home Automation: Can be integrated into smart home systems to control HVAC systems or fans based on temperature and humidity.
- HVAC Systems: Used in heating, ventilation, and air conditioning systems to monitor room temperature and humidity for efficient climate control.
- Consumer Electronics: Integrated into devices that require basic environmental monitoring, like digital thermometers or indoor climate monitors.
- Data Loggers: Used in environmental data loggers for tracking temperature and humidity over time in warehouses or storage areas.

3.5.4 Technical Parameters:

- Operating Voltage: 3.3V to 5V DC
- Current Consumption: Typically 2-5 mA during measurement
- Humidity Range: 20% to 80% RH
- Humidity Accuracy: ±5% RH
- Temperature Range: 0°C to 50°C
- Temperature Accuracy: ±2°C
- Digital Output: Provides direct digital data via a single wire interface
- Data Rate: Measurement is updated every 1-2 seconds.

- Response Time: The sensor provides data after 1-2 seconds for each measurement cycle.
- Size: 15.5 mm x 12 mm x 5.5 mm (L x W x H)
- Operating Temperature: 0°C to 50°C (for the temperature sensor)
- Response Time: 1-2 seconds per reading
- Temperature Coefficient: 0.5% / °C for humidity

3.5.5 Pin Layout

Pin	Name	Description
1	VCC	Power Supply (3.3V to 5V)
2	GND	Ground
3	DATA Pin	Digital Output Pin

Table 3: Pin layout of DHT11 Sensor Module

3.6 OLED Display (0.96 Inch)

3.6.1 Description:

The SSD1306 is a small, high-resolution OLED (Organic Light Emitting Diode) display module that is often used with microcontrollers like the ESP32. It is widely used for displaying information in embedded systems, providing a visually appealing way to present data in real time. The SSD1306 display has a resolution of 128x64 pixels, offering crisp and clear visuals for a variety of applications. The module communicates using the I2C (Inter-Integrated Circuit) protocol, making it simple and efficient to interface with microcontrollers like Arduino and ESP32.

Figure 11: 0.9 Inch OLED Display

The ESP32 with SSD1306 OLED Display combination is commonly used in a range of IoT and embedded system projects. The display is ideal for showing temperature readings, sensor data, system statuses, and other output in real-time. The SSD1306 uses very little power, making it an excellent choice for battery-powered devices. It also has a high contrast, ensuring the display is visible even in low-light environments.

The ESP32 microcontroller, paired with the SSD1306, provides a robust solution for a variety of applications that require displaying information without taking up much space or power. The display is ideal for use in devices such as smart home gadgets, weather stations, or robotics, where real-time data visualization is essential.

3.6.2 Features:

- High Resolution: Offers a resolution of 128x64 pixels for sharp and clear text and graphics.
- Low Power Consumption: OLED displays consume less power compared to traditional LCDs, making them suitable for battery-powered applications.
- I2C Communication: Uses the I2C protocol for easy communication with microcontrollers (requires only two wires: SDA and SCL).

- Compact Size: Small and space-efficient, allowing easy integration into compact systems.
- Wide Viewing Angle: OLED displays provide better viewing angles with more vibrant and brighter colors.
- Flexible Display: Capable of displaying both text and graphics, providing a versatile visual output.
- No Backlight Required: Since OLED displays emit light themselves, they do not need an additional backlight, reducing power consumption and increasing brightness.

3.6.3 Applications:

- Smart Home Devices: Used for displaying real-time data such as temperature, humidity, or system status.
- Weather Stations: Displays temperature, humidity, or other environmental conditions from sensors.
- Robotics: Used in robotic systems to show sensor readings, status updates, or navigation information.
- Portable Devices: Integrated into battery-powered portable gadgets where a small screen is needed to show data.
- Embedded Systems: Displays data in embedded systems such as smart clocks, security systems, or health monitoring devices.
- IoT Projects: Used for displaying live data from IoT sensors, dashboards, or remote monitoring systems.

3.6.4 Technical Parameters:

- Display Type: OLED (Organic Light Emitting Diode)
- Resolution: 128 x 64 pixels
- Communication Protocol: I2C (Inter-Integrated Circuit)

- Voltage: 3.3V to 5V
- Display Color: Monochrome (usually white or blue on black)
- Current Consumption: Typically 20-30 mA (depends on brightness settings)
- Contrast: Adjustable for clear visibility in various lighting conditions
- Dimensions: Approx. 27.2 mm x 28.2 mm x 4.5 mm
- Operating Temperature: -40°C to 70°C
- Display Lifetime: Around 100,000 hours of continuous use
- Font Type: Supports multiple fonts for text display
- Refresh Rate: 60Hz (max refresh rate)

3.6.5 Pin Layout - Connections

OLED Pin	Esp32 Pin
VCC	3.3 V
GND	GND
SDA	GPIO 21
SCK	GPIO 22

Table 4: Pin layout of OLED & Connection to ESP 32

3.7 MCP2515 CAN Controller

3.7.1 Description

Microchip Technology's MCP2515 is a stand-alone Controller Area Network (CAN) controller that implements the CAN specification, Version 2.0B. It is capable of transmitting and receiving both standard and extended data and remote frames. The MCP2515 has two acceptance masks and six acceptance filters that are used to filter out unwanted messages, thereby reducing the host MCU's overhead. The MCP2515 interface with microcontrollers (MCUs) via an industry standard Serial Peripheral Interface (SPI).

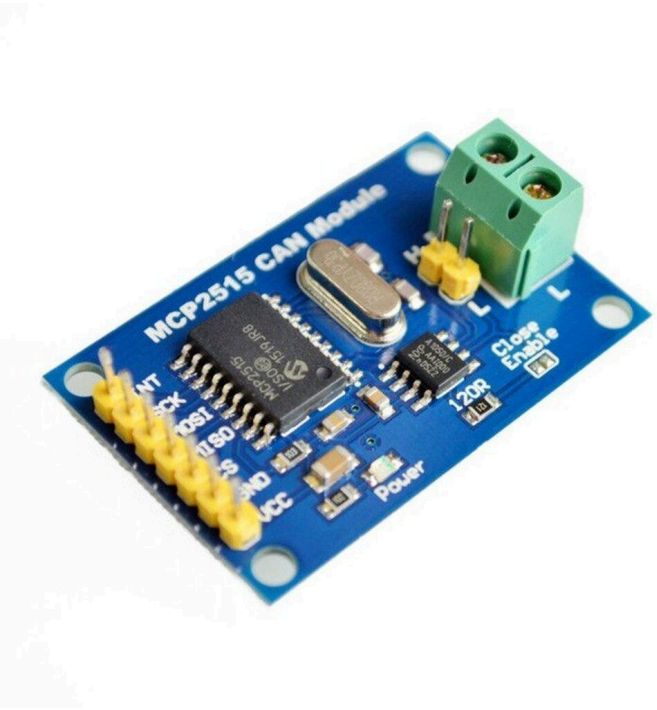

Figure 12: CAN Module

3.7.2 Features

- Implements CAN V2.0B at 1 Mb/s:
  - 0 to 8-byte length in the data field
  - Standard and extended data and remote frames
- Receive Buffers, Masks and Filters:
  - Two receive buffers with prioritized message storage
  - Six 29-bit filters
  - Two 29-bit masks
- Data Byte Filtering on the First Two Data Bytes (applies to standard data frames)
- Three Transmit Buffers with Prioritization and Abort Features
- High-Speed SPI Interface (10 MHz):
  - SPI modes 0,0 and 1,1
- One-Shot mode Ensures Message Transmission is Attempted Only One Time
- Clock Out Pin with Programmable Prescaler:

- Can be used as a clock source for other device(s)

- Start-of-Frame (SOF) Signal is Available for Monitoring the SOF Signal:

  - Can be used for time slot-based protocols and/or bus diagnostics to detect early bus degradation

- Interrupt Output Pin with Selectable Enables

- Buffer Full Output Pins Configurable as:

  - Interrupt output for each receive buffer

  - General purpose output

- Request-to-Send (RTS) Input Pins Individually Configurable as:

  - Control pins to request transmission for each transmit buffer

  - General purpose inputs

3.7.3 Pin Layout

MCP2515 Pin	Description	ESP32 Connection
VCC	Power (5V or 3.3V)	3.3V (or 5V if using level shifter)
GND	Ground	GND
CS	SPI Chip Select	GPIO 5
SO	SPI MISO	GPIO 19
SI	SPI MOSI	GPIO 23
SCK	SPI Clock	GPIO 18
INT	Interrupt (active low)	GPIO 2

Table 5: Pin layout of CAN Module & Connection to ESP 32

3.7.4 Benefits

- Simplified CAN Implementation: Handles the complex CAN protocol, simplifying the design of CAN-based systems.

- Reduced Microcontroller Load: Message filtering and buffering reduce the processing required from the microcontroller.
- Flexible Configuration: Offers various configuration options to tailor the CAN communication to specific application needs.
- Cost-Effective: A relatively low-cost solution for adding CAN connectivity to a system.

3.7.5 Applications

The MCP2515 is widely used in various applications, including:

- Automotive electronics
- Industrial automation
- Embedded systems
- CAN-based networks

CHAPTER -4: SOFTWARE DESCRIPTION

4.1 Embedded C Programming Language

Embedded C is a set of extensions to the standard C programming language, tailored for the needs of embedded systems. These extensions enable direct hardware control, support for specialized features like fixed-point arithmetic, multiple memory banks, and I/O operations, which are commonly required in embedded applications. Over the years, Embedded C has become the dominant programming language for developing software for microcontrollers and embedded devices due to its efficiency and low-level access to hardware resources.

While C is known for its portability and ease of use, embedded systems typically lack the comprehensive libraries and dynamic linking capabilities available in higher-level programming environments. This makes Embedded C particularly suited for constrained systems where resources such as memory and processing power are limited. The language allows for highly optimized code that can run on low-power devices with minimal overhead, making it ideal for real-time applications in embedded systems.

Embedded C is also a popular choice for device driver development, system firmware, and low-level hardware control. Its ability to interface directly with hardware through memory-mapped registers and control peripherals efficiently makes it indispensable in the embedded systems domain. Additionally, C compilers have evolved significantly in recent years, offering performance that rivals low-level assembly language programming in terms of both speed and size optimization.

One of the key advantages of Embedded C is its portability. Code written in Embedded C can be compiled for different microcontroller architectures with minimal changes, ensuring that projects can scale and adapt to new hardware platforms as they evolve.

4.2 Arduino IDE Compiler

The **Arduino Integrated Development Environment (IDE)** is an open-source platform designed for programming Arduino boards, making it accessible for developers ranging from beginners to professionals. The Arduino IDE simplifies the development process by providing a user-friendly interface that facilitates code writing, debugging, and uploading to the microcontroller.

Figure 13: IDE Compiler Software

The programming environment uses a simplified version of C/C++ and offers built-in functions for interacting with hardware, making it easy to develop embedded applications without needing extensive knowledge of low-level programming. The Arduino IDE allows users to focus on the logic and functionality of their projects while abstracting much of the complexity associated with hardware interfacing.

4.2.1 Key Features of the Arduino IDE:

- Cross-platform Support: Available for Windows, macOS, and Linux.
- Ease of Use: Simplifies programming tasks with an intuitive interface.
- Extensive Libraries: Offers numerous libraries to support a wide range of sensors, actuators, and communication protocols.
- One-click Compilation and Upload: The IDE provides easy-to-use tools for compiling code and uploading it to the board in one step.

- Real-time Debugging: Built-in serial monitor for debugging and real-time data visualization.

The Arduino IDE is an essential tool for rapid prototyping and development, offering both simplicity for newcomers and enough flexibility for advanced users. The integration of Arduino boards with a rich ecosystem of libraries and a vast online community makes it an ideal platform for building embedded systems for various applications.

4.2.2 Serial Monitor

The **Serial Monitor** within the Arduino IDE is a crucial tool for communicating with the Arduino board during development. It provides a means of sending and receiving data between the microcontroller and the development environment, allowing for real-time monitoring, debugging, and interaction with the running system.

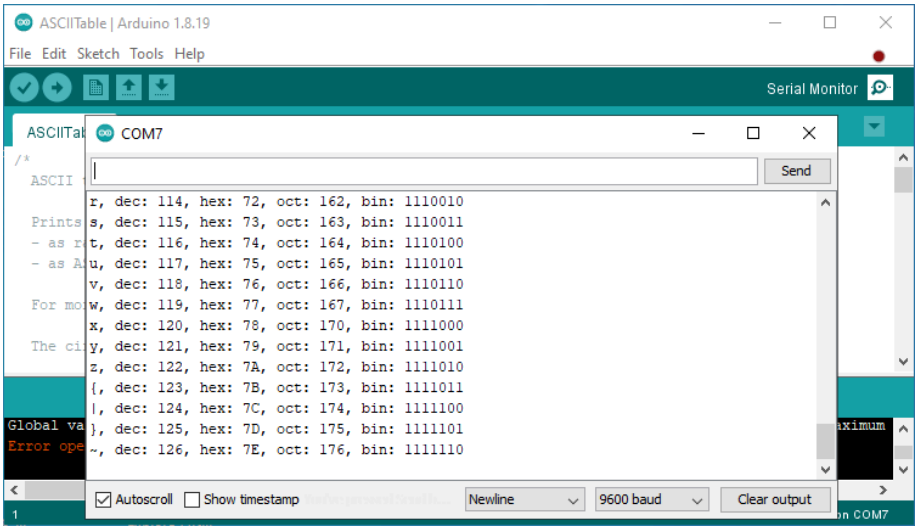

Figure 14: Serial COM Monitor

The Serial Monitor displays messages from the Arduino board, which can include sensor readings, status updates, or debug information, assisting developers in ensuring that their code is functioning as expected. It also allows users to send commands or input data to the Arduino via the USB or serial connection, facilitating interaction with the system for testing and control purposes.

4.2.3 Key Features of the Serial Monitor:

- Real-time Data Display: View data sent from the Arduino board in real-time.
- Data Input: Send textual or numerical commands to the Arduino for interaction.
- Custom Baud Rate: Select the appropriate baud rate for communication, depending on the hardware setup.

The Serial Monitor plays an essential role in ensuring that the embedded system operates correctly and facilitates quick troubleshooting during development.

4.2.4 Steps for Uploading Code to Arduino

Uploading code to an Arduino board is a straightforward process that involves several key steps:

- Establish the Connection: Connect the Arduino board to your computer via a USB cable.
- Open Arduino IDE: Launch the IDE on your computer and open the relevant sketch (code).
- Select the Correct Board and Port: Under the "Tools" menu, select the appropriate board and communication port.
- Verify the Code: Click on the "Verify" button to compile the code and check for syntax errors.
- Upload the Code: If the code compiles successfully, click the "Upload" button to transfer it to the Arduino board.
- Monitor the Output: Use the Serial Monitor to observe the board's behavior and any output data in real-time.

Once the code is successfully uploaded, the Arduino board begins executing the program, interacting with connected peripherals and sensors as specified in the code.

4.2.5 Working Procedure of Arduino

The general workflow for developing an embedded system using the Arduino platform involves the following steps:

- Hardware Setup: Connect the Arduino board to the computer and interface it with necessary sensors, actuators, and other components.
- Code Development: Open the Arduino IDE, write the program that defines the behavior of the system, and integrate the appropriate libraries.
- Code Upload: Compile the code and upload it to the Arduino board via the IDE.
- System Execution: Upon upload, the Arduino executes the program and interacts with connected components such as sensors and actuators.
- Debugging and Monitoring: Use the Serial Monitor to observe system performance, verify sensor readings, and make adjustments as necessary.

This iterative process allows developers to quickly test and refine their systems, making Arduino a powerful tool for embedded system development.

4.3 Thing Speak – IoT Platform

Thing Speak is an open-source Internet of Things (IoT) platform that provides a secure and scalable solution for collecting, analyzing, and visualizing data from IoT devices. It enables users to send data from sensors or devices to the cloud, where it can be stored, processed, and analyzed in real-time. ThingSpeak serves as a bridge between physical systems and digital platforms, facilitating the development of connected applications.

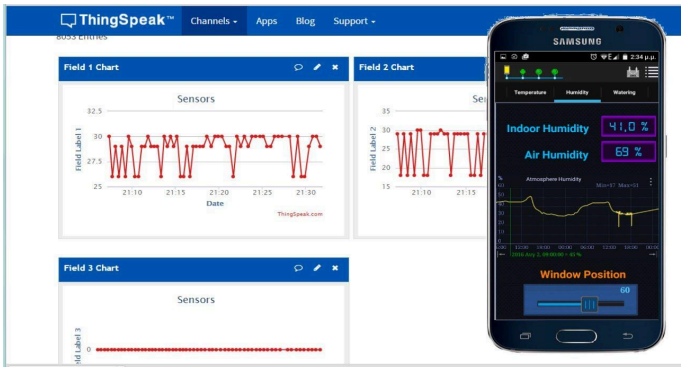

Figure 15: ThingSpeak UI

4.3.1 Key Features of ThingSpeak:

- Data Collection: Supports data transfer from IoT devices to the cloud via REST APIs or MQTT.
- Real-time Data Visualization: Visualize data using built-in tools to create charts, graphs, and dashboards.
- Advanced Data Analysis: Perform complex data analysis using MATLAB integration to uncover trends and patterns.
- Alert and Action Triggers: Set up automated actions or notifications based on specific conditions or thresholds.
- Open-Source Platform: Customizable and extensible for a wide range of applications.

4.3.2 How ThingSpeak Works:

- Device Connectivity: IoT devices collect data and send it to ThingSpeak via the internet.
- Data Storage: Data is stored in channels, which are used to organize and manage different data streams.
- Data Visualization and Analysis: Use ThingSpeak's visualization tools or integrate with MATLAB for advanced data analytics.

- Actionable Insights: Based on the data analysis, users can configure alerts, notifications, and trigger actions remotely.

4.3.3 Benefits:

- User-Friendly: Ideal for rapid prototyping with minimal setup and configuration.
- Scalable: Can handle large datasets and adapt to growing projects.
- Integration with MATLAB: Provides powerful analytics capabilities.
- Customizable: Open-source platform for flexibility and tailored solutions.

4.3.4 Common Use Cases:

- Environmental Monitoring: Monitoring temperature, humidity, air quality, and other environmental factors.
- Smart Agriculture: Tracking soil moisture, temperature, and other agricultural parameters.
- Home Automation: Managing and controlling devices such as lights, security systems, and appliances.
- Industrial Monitoring: Collecting data on equipment performance and predicting maintenance needs.

ThingSpeak is a versatile platform that facilitates the development of IoT applications and empowers users to harness real-time data for smarter decision-making and automation.

CHAPTER - 5 : CIRCUIT DIAGRAM

5.1 Circuit Model

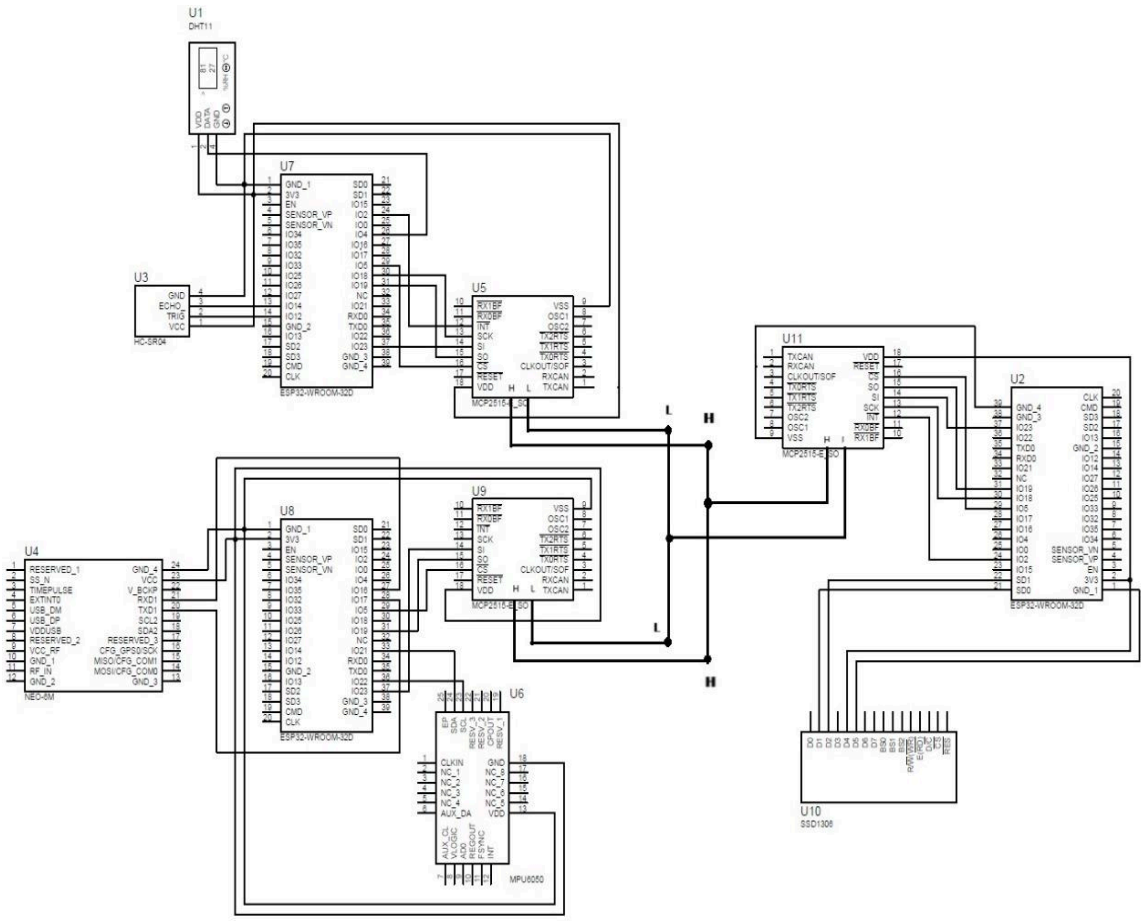

Fig 16 : Circuit Diagram

The circuit diagram consists of three nodes based on ESP32 microcontrollers, where one main node (cluster node) and a secondary node are interconnected using a CAN transceiver (MCP2515). The main node is equipped with a DHT11 sensor to measure temperature and humidity, an HC-SR04 ultrasonic sensor to measure distance. The secondary node includes a NEO-6M GPS module for tracking location and an MPU6050 accelerometer and gyroscope for measuring orientation and motion. The MCP2515 CAN transceiver enables communication between the two ESP32 nodes, allowing them to exchange data over the CAN bus. The main node collects data from its sensors, displays it

on the OLED, and communicates with the secondary node to receive additional sensor information, creating a comprehensive system for monitoring environmental conditions and motion parameters.

The system is designed to work as an integrated unit where the main node acts as the central controller, gathering data from both its own sensors and the secondary node. The use of the CAN bus ensures reliable and efficient communication between the nodes, even in environments with potential electrical noise or interference. The OLED display on the main node provides a user-friendly interface to view all the collected data in real-time.

The MCP2515 is a CAN (Controller Area Network) transceiver that facilitates communication between the main node and the secondary node.

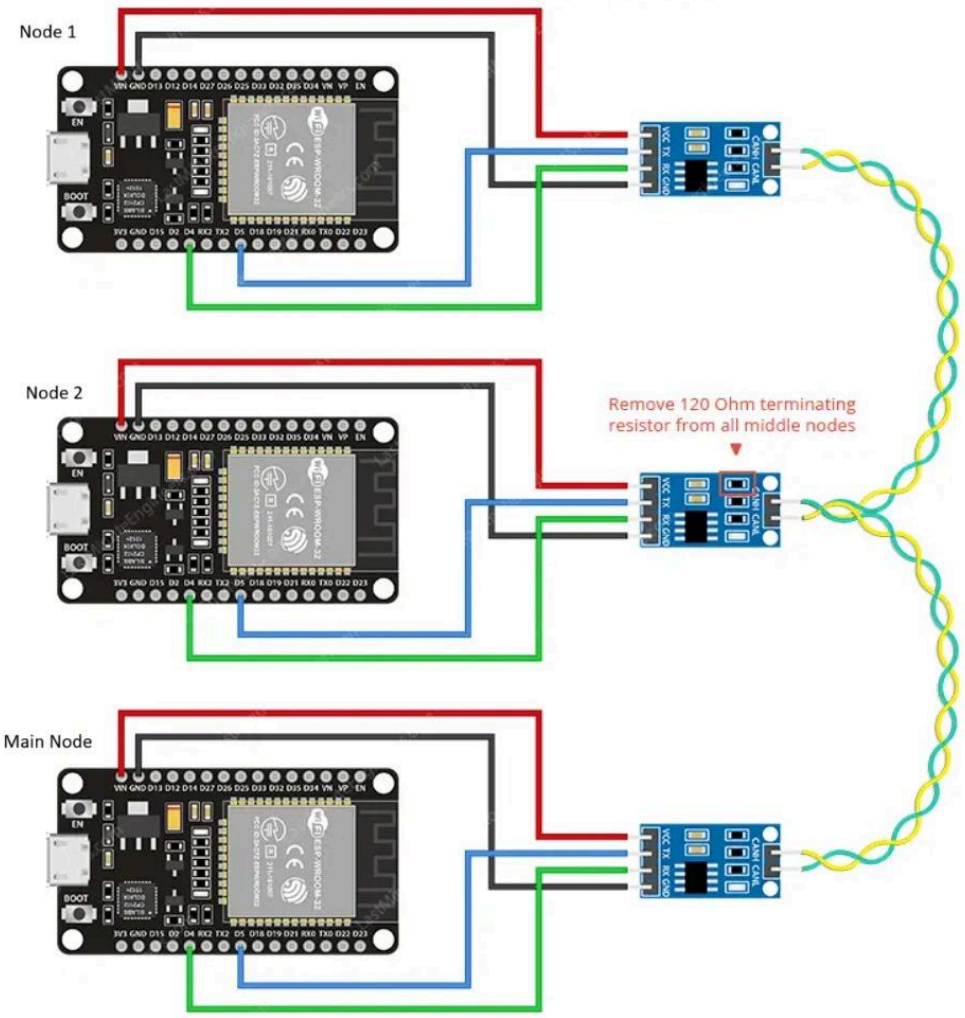

Fig 17 : ESP32 communication via CAN transreceiver

CHAPTER - 6 : RESULTS

The results of the Vehicle Safety Monitoring System with CAN and IoT Integration project demonstrate a highly effective and innovative solution for enhancing vehicle safety through real-time monitoring, accident prevention, and remote diagnostics. The system successfully integrates CAN Bus (Controller Area Network) and IoT (Internet of Things) technologies to collect, process, and transmit critical vehicle data, ensuring comprehensive safety for both the vehicle and its occupants.

6.1 Key Results and Achievements:

1. Real-Time Data Collection and Monitoring:
    - The system continuously monitors critical vehicle parameters such as temperature, speed, acceleration, GPS location, and proximity to obstacles. This real-time data collection ensures that any abnormal conditions, such as excessive speed, or unsafe proximity to objects, are detected immediately.
    - Sensors like DHT11 (temperature and humidity), HC-SR04 (ultrasonic distance sensor), GPS NEO-6M (GPS module), and MPU6050 (accelerometer and gyroscope) work seamlessly to provide accurate and reliable data.

2. CAN Bus Communication:
    - The CAN Bus protocol ensures robust and error-free communication between the sensors and the ESP32 microcontroller, which acts as the central processing unit. This allows for efficient data exchange within the vehicle, even in noisy automotive environments.
    - The use of the MCP2515 CAN controller enables seamless integration with the vehicle's existing electronic systems, ensuring compatibility and scalability.

3. IoT Integration for Remote Monitoring:
    - The system leverages IoT connectivity to transmit sensor data to a cloud-based platform (e.g., ThingSpeak) via Wi-Fi. This enables remote monitoring of vehicle parameters such as location, temperature, humidity, and proximity to obstacles.
    - The IoT dashboard provides real-time visualization of the data, allowing fleet managers or vehicle owners to monitor the vehicle's status from anywhere. Alerts and notifications are generated if any parameter exceeds predefined thresholds, ensuring timely intervention.

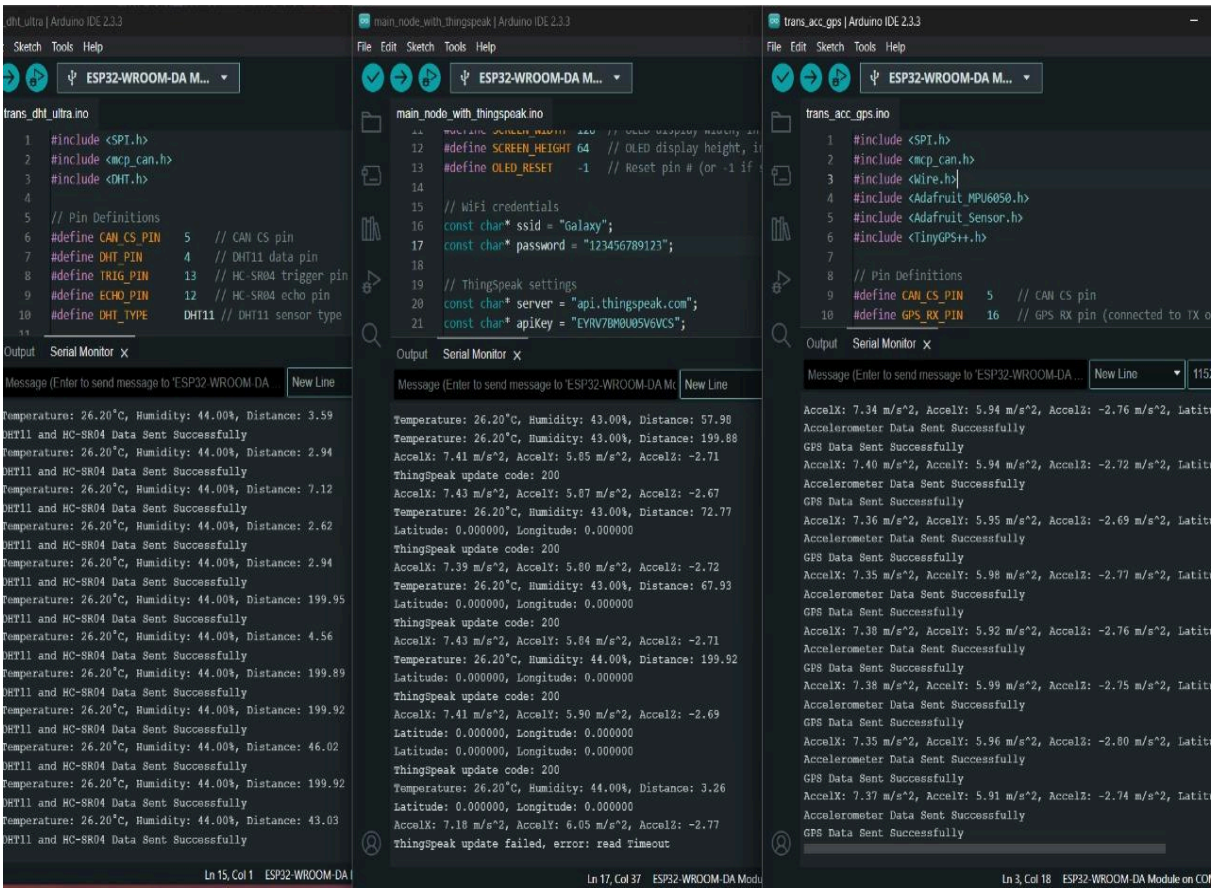

Fig 18 : Results from the three nodes

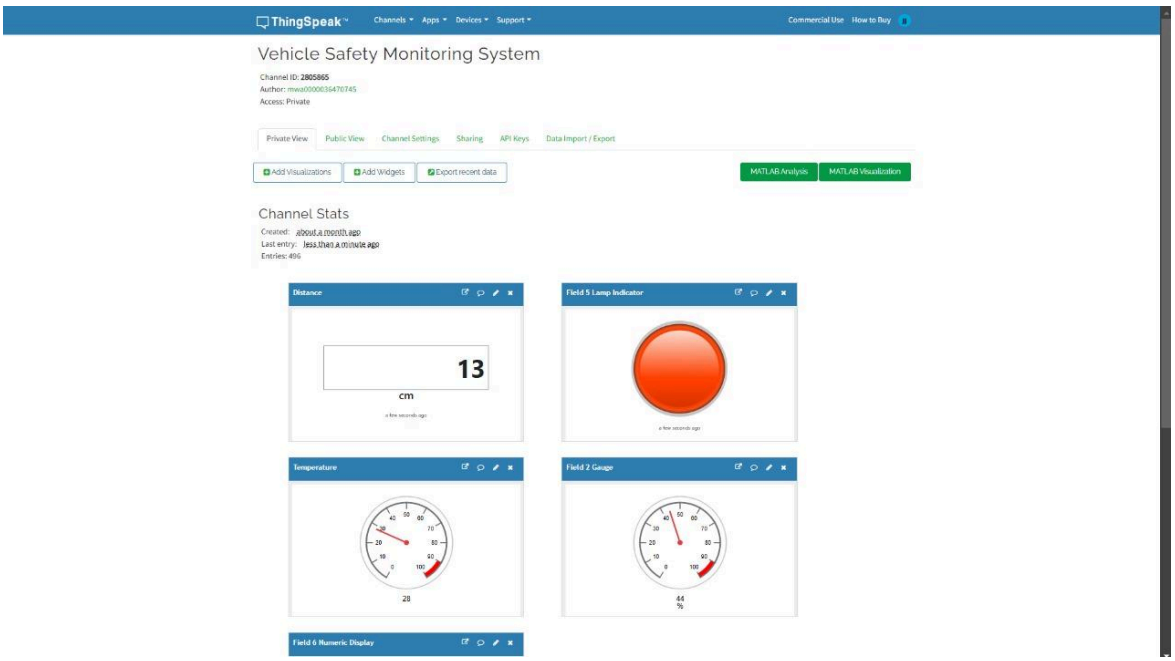

Fig 19 : Data Send to the ThingSpeak Dashboard

The Vehicle Safety Monitoring System with CAN and IoT Integration has successfully demonstrated its ability to enhance vehicle safety through real-time monitoring, accident prevention, and remote diagnostics. By integrating CAN Bus and IoT technologies, the system provides a comprehensive solution for tracking critical vehicle parameters such as temperature, speed, acceleration, GPS location, and proximity to obstacles. The system's ability to collect, process, and transmit data in real-time ensures that any abnormal conditions, such as engine overheating, excessive speed, or unsafe proximity, are detected immediately, allowing for timely interventions.

The use of CAN Bus ensures robust and reliable communication between the sensors and the ESP32 microcontroller, enabling seamless data exchange within the vehicle. The integration of IoT allows for remote monitoring via a cloud-based platform, where data is visualized on an intuitive dashboard. This enables fleet managers or vehicle owners to monitor the vehicle's status from anywhere, with real-time alerts and notifications for unsafe conditions. The system's GPS functionality provides precise location tracking, which is particularly useful in emergency situations, ensuring faster response times.

One of the key achievements of the system is its ability to trigger real-time warnings for unsafe conditions, such as high speed or collision detection. For example, if the vehicle is approaching an obstacle too quickly, the system alerts the driver, potentially preventing an accident. Additionally, the system supports predictive maintenance by continuously monitoring vehicle parameters, identifying potential issues before they lead to failures, and reducing the risk of unexpected breakdowns.

CHAPTER -6 : CONCLUSION

The Vehicle Safety Monitoring System represents a significant leap forward in automotive safety by seamlessly integrating CAN Bus (Controller Area Network) and IoT (Internet of Things) technologies. This innovative system is designed to provide real-time vehicle monitoring, accident prevention, and remote diagnostics, ensuring the safety of both the vehicle and its occupants. By continuously tracking critical parameters such as temperature, speed, acceleration, GPS location, and proximity to other vehicles or obstacles, the system offers a comprehensive safety solution that addresses a wide range of potential hazards.

At the core of this system is the CAN Bus, a robust and reliable communication protocol widely used in the automotive industry. The CAN Bus enables the system to collect data from various sensors and electronic control units (ECUs) distributed throughout the vehicle. This data includes real-time information on engine temperature, vehicle speed, acceleration, and more. By integrating IoT technology, the system can transmit this data to a centralized cloud-based platform, where it is analyzed and monitored in real-time. This connectivity allows for remote diagnostics, enabling fleet managers or vehicle owners to keep tabs on the vehicle's health and performance from anywhere in the world.

One of the key features of the Vehicle Safety Monitoring System is its ability to trigger real-time warnings for unsafe conditions. For instance, if the system detects that the vehicle is exceeding a safe speed limit, it can immediately alert the driver through visual or auditory warnings. Similarly, the system can detect potential collisions by monitoring the vehicle's proximity to other objects and provide timely alerts to the driver. In cases of sudden acceleration or deceleration, which could indicate an impending collision, the system can even engage safety mechanisms like automatic braking to mitigate the impact. These real-time warnings significantly enhance driver awareness and contribute to a safer driving experience.
The system's GPS functionality adds another layer of safety by providing precise location tracking. This feature is particularly useful in emergency situations, where knowing the exact location of the vehicle can expedite response times. Additionally, GPS data can be used for route optimization and fleet management, further enhancing the efficiency and safety of vehicle operations.

Looking to the future, the Vehicle Safety Monitoring System is poised for several advancements that will further elevate its capabilities. One such improvement is the adoption of CAN FD (Flexible Data-Rate), which offers higher data transmission rates compared to traditional CAN Bus systems. This will enable the system to handle more sophisticated monitoring tasks and respond to potential hazards even faster. Another

promising development is the integration of AI-based predictive analytics. By leveraging machine learning algorithms, the system can analyze historical data to predict potential issues before they occur. For example, it could identify patterns indicative of mechanical failures or risky driving behaviors, allowing for proactive maintenance and corrective actions.

Vehicle-to-Vehicle (V2V) communication is another exciting frontier for this system. V2V technology enables vehicles to share real-time data with each other, creating a collaborative safety network. For instance, if one vehicle detects a hazardous road condition, it can instantly communicate this information to nearby vehicles, allowing them to take preventive measures. This interconnected approach to vehicle safety has the potential to significantly reduce accidents and improve overall road safety.

In conclusion, the Vehicle Safety Monitoring System is a groundbreaking solution that demonstrates the transformative potential of integrating CAN Bus and IoT technologies in the automotive industry. By providing real-time monitoring, accident prevention, and remote diagnostics, the system enhances both vehicle and driver safety. Future advancements such as CAN FD, AI-based predictive analytics, and V2V communication will further solidify this system as a cornerstone of modern vehicle safety solutions. As these technologies continue to evolve, they will play a crucial role in making roads smarter and safer for everyone.

REFERENCE

1. Yin Mar Win Kyaw Myo Maung, Hla Myo Tun, 2016, “Implementation Of CAN Based Intelligent Driver Alert System”, INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH VOLUME 5, ISSUE 06, JUNE 2016.

2. Dr Revanesh M, Sanjay H U, Gireesha C P, Swaroop B K, Tarun Ranga, 2024, “CAN Protocol Based Vehicle Monitoring System”, International Advanced Research Journal in Science, Engineering and Technology.

3. A. M. Elshaer, M. M. Elrakaiby, Mohamed E. Harb, 2018, “Autonomous Car Implementation Based on CAN Bus Protocol for IoT Applications”, The 13th IEEE International Conference on Computer Engineering and Systems. ICCES 2018

4. Vasileios, G., & Georgios, C. (2019). "Real-time Vehicle Monitoring and Safety using CAN Protocol". IEEE Access. A research paper that focuses on CAN protocol and its integration with various sensors for real-time vehicle data transmission. It covers essential vehicle parameters and their monitoring.

5. Li, Y., & Zhang, T. (2020). "IoT-Enabled Vehicle Monitoring and Fleet Management Systems". Proceedings of the IEEE International Conference on Vehicular Electronics and Safety (ICVES).

6. Ahmed, N., & Ali, S. (2021). "Vehicle Safety Monitoring System using CAN Protocol and IoT". IEEE International Conference on Internet of Things and Smart City (IoTSC).

7. Sharma, V., & Reddy, G. (2019). "Intelligent Vehicle Safety System using CAN Protocol and IoT." International Journal of Advanced Engineering and Technology.

This research discusses the integration of IoT for real-time vehicle monitoring using sensors and the CAN protocol, emphasizing safety aspects like speed, temperature, and proximity monitoring.

8. Suresh, M., & Kumar, B. (2020). "Smart Vehicle Safety System Using CAN Protocol and IoT." Proceedings of the IEEE International Conference on Industrial Electronics and Applications (ICIEA). This paper outlines the integration of CAN protocol and IoT for monitoring vehicle safety parameters in real time, including temperature, proximity, and motion.

9. Zhang, J., & Liu, W. (2019). IoT Applications for Automotive Industry. Wiley.A detailed exploration of how IoT technologies are being applied in the automotive industry, including vehicle safety monitoring, fleet management, and autonomous systems.

10. Tripathi, A., & Rao, D. (2021). "Vehicle Safety Monitoring Using IoT: A Survey of Techniques and Applications." Journal of Traffic Safety and Control. Provides a detailed review of various IoT techniques and their applications in vehicle safety, focusing on real-time data acquisition ,sensor integration, and communication technologies.