Misr Higher Institute
for Engineering & Technology

Department of Communications
and Computer Engineering Department

Improving Road Safety and Crash Prevention through Al and IoT-enabled ADAS and V2V Communication

A project submitted in partial fulfillment for the

requirements of the degree of B. Sc. in

Computer Engineering

By:

Abrar Twakal Mohamed
Ahmed Mohamed Ismail
Ahmed Nader Mohamed
Asmaa Mohamed Fathy
Doaa Nasser Ahmed Mosbah
Esraa El-Sayed Shatat
Hager Basheer Eid
Hager Khaled Saad
Mariam Ahmed Elyamany
Mazen Ahmed Al-Asas
Naira Mamdouh Salama
Salma Hamada Dawoud
Ziad Mohamed Abd El-Hamid
Yasser Khairy El-Gafary

Supervisor

Assist.Prof.Yasser Hamed Ahmed Elawady

Communications and Computer Engineering Department
Misr Higher Institute for Engineering & Technology

2025

Project Group Members

Assist.Prof.Yasser Elawady

Supervisor Of Project

Abrar Twakal Mohamed

Asmaa Mohamed Fathy

Hager Basheer Eid

Mazen Ahmed Al-Asas

Yasser Khairy El-Gafary

Ahmed Mohamed Ismail

Doaa Nasser Mosbah

Hager Khaled Saad

Naira Mamdouh Salama

Ahmed Nader Mohamed

Esraa El-Sayed Shatat

Mariam Ahmed Elyamany

Salma Hamada Dawoud

Ziad Mohamed Abd EL-Hamid

Acknowledgement

First and foremost, all praise and thanks are due to Allah, who granted us the strength, patience, and ability to complete this project.

We extend our deepest gratitude to our project supervisor, **Assist.Prof.Yasser Hamed Ahmed Elawady**, for his invaluable guidance, continuous support, and insightful expertise throughout this journey. His dedication and encouragement were instrumental in shaping this work.

Finally, we honor the exceptional efforts of our entire team and express our heartfelt appreciation to our families for their unwavering support, motivation, and understanding during the completion of this project.

Abstract

In recent years, active safety systems integrating vehicle-to-vehicle (V2V) communication have gained significant attention as a promising solution to reduce road accidents and address the limitations of traditional sensor-based ADAS. This project presents a V2V-based system that enhances driver safety by enabling real-time communication between vehicles, supporting features such as lane change warning, forward collision warning, blind spot detection, and intersection movement assistance. Additionally, the system utilizes internal driver monitoring to detect fatigue, and external cameras for traffic sign recognition. The communication between vehicles is encrypted to ensure secure and reliable data exchange. By combining sensor data with V2V technology, the system aims to offer a robust, responsive, and intelligent safety layer that contributes to safer and more efficient driving.

Vehicle-to-vehicle (V2V) communication is a promising technology for improving road safety by enabling vehicles to exchange information about their speed, position, and direction in real-time. In this graduation project, we propose a V2V collision avoidance advanced driver assistance system (ADAS) that can reduce the risk of accidents by alerting drivers of potential collisions and assisting them in taking corrective actions.

Table of Contents

List of Abbreviation

A	
ABSOLUTE ()	A function that returns the absolute value.
ACCEL_DIST	Acceleration Distance.
AES	Advanced Encryption Standard
ASN.1	Abstract Syntax Notation One
APS	AutoParking System
B	
BSW_WARNING	FALSE–No Warning; BSW_LEFT_WARN–Left Blind Spot Warning; BSW_LEFT_ADVISORY–Left Blind-Spot Advisory Alert; BSW_RIGHT_WARN–Right Blind-Spot Warning; BSW_RIGHT_ADVISORY - Right Blind-Spot Advisory Alert.
BSW_ZONE	Blind-Spot Warning Zone Length
C	
CAN	Controller Area Network
CNN	Convolutional Neural Network
CORS	Continuously Operating Reference Stations
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
CSS	Cascading Style Sheets
CTR	Counter Mode
C-V2X	Cellular Vehicle-to-Everything
D	
DELTA_HEADING (Δθ)	Absolute Delta Heading between HV and RV in Radians
DFD	Data Flow Diagram.
DFR	Driver Face Recognition
DMS	Driver Monitoring System
DNPW_WARNING	FALSE–No Warning; TRUE–DNPW Warning Generated
DNPW_ZONE	DNPW Zone Length
DOM	Document Object Model

DSRC	Dedicated Short-Range Communication
E	
EAR	Eye Aspect Ratio
EEBL_WARNING	FALSE–No Warning; TRUE–EEBL Warning Generated
ESP-NOW	Protocol for ESP-based communication
F	
FCW_WARNING	FALSE–No Warning; TRUE–FCW Warning Generated
G	
GMAC	Galois Message Authentication Code
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
H	
HAL	Hardware Abstraction Layer
HTML	Hypertext Markup Language
HTTP	Hyper Text Transfer Protocol
HV_SPEED	Speed of Host Vehicle
HV_TTI	Time-taken by the HV to arrive at the intersection point
HV_TO_INTERSECTION_DIST	Distance between the HV and intersection point
I	
IMA_LEFT_WARNING	FALSE–No Warning; TRUE–IMA Left Warning Generated
IMA_RIGHT_WARNING	FALSE–No Warning; TRUE–IMA Right Warning Generated
IV	Initialization Vector
J	
JS	JavaScript
JWT	JSON Web Token
K	
K_ACCEL_TIME	A Calibrated time taken by HV to travel Acceleration Distance
K_BUF_ZONE	A calibrated buffer zone length allowed

K_BSW_ZONE_MIN_LEN	Calibrated minimum BSW Zone Length
K_EEBL_ACCEL_THRES	EEBL minimum acceleration threshold($-3.92 m/s^2$)
K_HV_MIN_SPD_THRES	HV minimum speed threshold (1 m/s)
K_MAX_EEBL_ZONE_LEN	Maximum longitudinal offset between HV and RV (300 m)
K_MAX_HV_TO_INTERSECTION_DIST	Calibrated Maximum Distance between the HV and Intersection point for IMA
K_OVERTAKE_TIME	A Calibrated time taken by HV to travel Overtaking Distance
K_REL_SPD_MULT	Calibrated Relative Speed Multiplier
K_RETURN_TIME	A Calibrated time taken by HV to travel Return Distance
K_SPD_MULT	Calibrated HV Speed Multiplier
K_TTC_THRES	FCW time-to-collision calibrated threshold
K_TTI_TOLERANCE	Calibrated tolerance value for the difference between HV_TTI and RV_TTI
L	
LAT_OFFSET	Lateral Offset as obtained from the TC module
LEFT_TURN_SIGNAL	Left Turn Signal Status as obtained from the Vehicle CANBus
LON_OFFSET	Longitudinal Offset as obtained from the TC module
M	
MAC	Media Access Control
MAP	Intersection Map
MCAL	Microcontroller Abstraction Layer
N	
NHTSA	National Highway Traffic Safety Administration
O	
OAuth	Open Authorization Protocol
OBU	On-Board Unit
OEM	Original Equipment Manufacturer

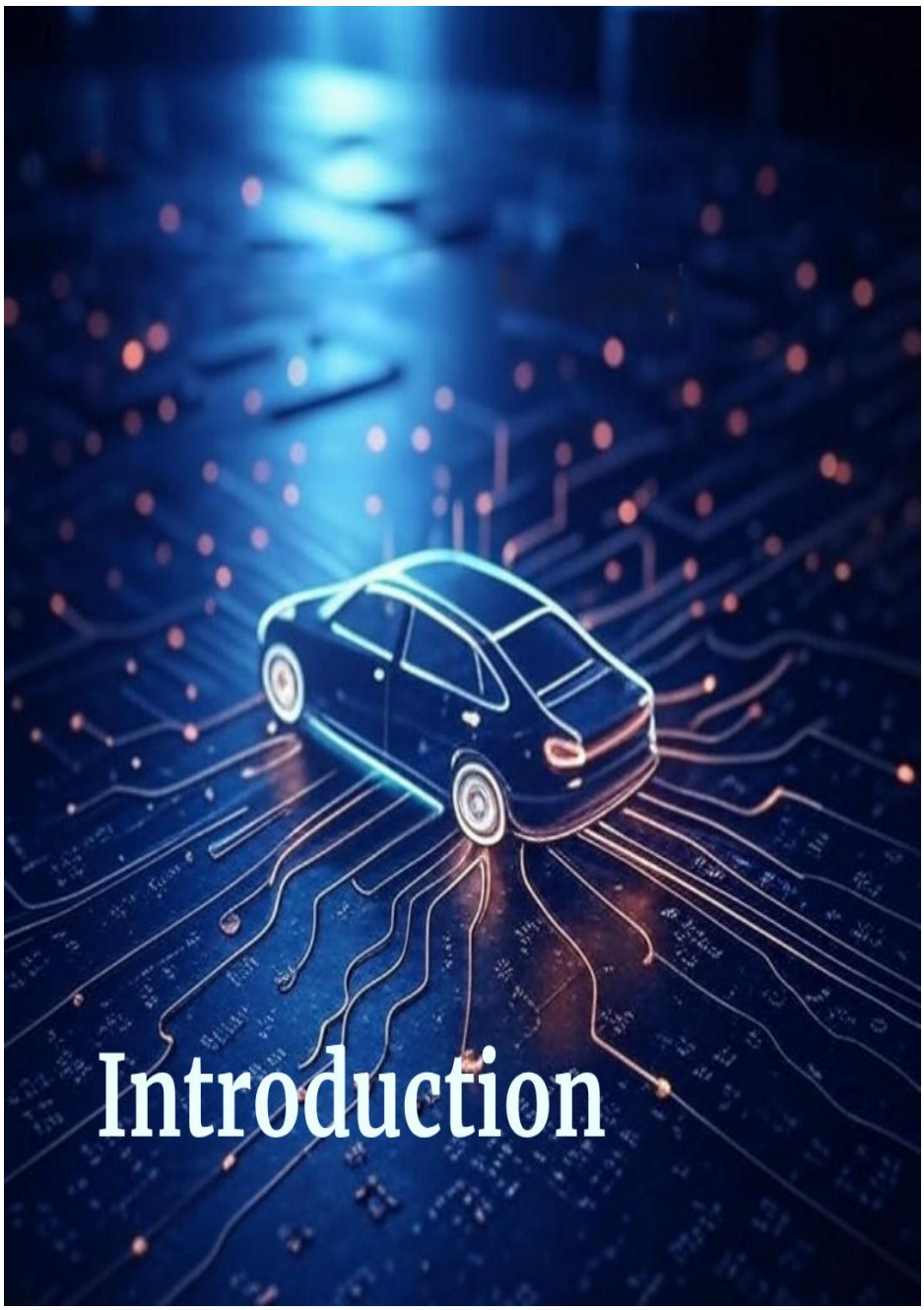
OBE	On-Board Equipment
OVERTAKING_DIST	Overtaking Distance
P	
PCB	Printed Circuit Board
P2P	Peer-to-Peer
PKI	Public Key Infrastructure
R	
REL_SPEED	Relative Speed (RV Speed - HV Speed)
RETURN_DIST	Return Distance
RIGHT_TURN_SIGNAL	Right Turn Signal Status as obtained from the Vehicle CANBus
RTDB	Realtime Database
RTOS	Real-Time Operating System
RSU	Roadside Unit
RTS/CTS	Request to Send / Clear to Send
RV_ACCEL	Acceleration of Remote Vehicle
RV_DIRECTION	RV relative direction as obtained from the TC module
RV_SPEED	Speed of Remote Vehicle
RV_TTI	Time-taken by the RV to arrive at the intersection point
RV_TO_INTERSECTION_DIST	Distance between the RV and intersection point
RV_ZONE	RV relative zone as obtained from the TC module
S	
SAE	Society of Automotive Engineers
SDK	Software Development Kit
SPaT	Signal Phase and Timing
T	
TC	Target Classification
TDMA	Time Division Multiple Access
TIME_TAKEN	Time taken by the HV to complete overtaking and get back to its lane

TSR	Traffic Sign Recognition
TTB	Time taken by RV to enter the HV's Blind-Spot
TTC	time-to-collision
W	
WAAS	Wide Area Augmentation System
WebGL	Web Graphics Library
Y	
YOLO	You Only Look Once
Z	
∆Speed	The increment in HV Speed in order to overtake the slow-moving vehicle

List of figures

List of Tables

Chapter 1



Chapter 1: Introduction

1.1Overview

The increasing number of road accidents highlights the urgent need for intelligent safety systems to assist drivers and reduce human error, a major cause of traffic collisions. With rising traffic congestion, distractions, and fatigue, driving has become more stressful and risky. While governments have invested in infrastructure development, additional in-vehicle technologies are necessary to enhance driver awareness and decision-making.

Advanced Driver Assistance Systems (ADAS) have emerged as a solution, offering real-time information, warnings, and automated actions to reduce accidents. A promising area within ADAS is Vehicle-to-Vehicle (V2V) communication, which enables vehicles to exchange data, such as position and speed, to improve safety. Traditional ADAS systems have limitations, such as restricted field of view and range, while V2V communication extends situational awareness by sharing data from other vehicles.

This project uses ESP-NOW, a low-power, connectionless communication protocol, to enable efficient V2V communication. ESP-NOW is suitable for real-time, short-range communication in vehicular networks, making it ideal for safety applications like collision avoidance and lane change warnings.

The proposed system integrates ESP-NOW-based V2V communication with ADAS features like lane change assistance, forward collision warnings, blind spot monitoring, emergency brake lights, and intersection movement assistance. It also includes driver monitoring through internal cameras to detect fatigue and external cameras for traffic sign recognition. These features are managed through an interactive display interface.

By combining ADAS with encrypted V2V communication via ESP-NOW, the system aims to provide a safer, intelligent, and cooperative driving environment, offering a cost-effective solution that can be deployed in both new and existing vehicles.

Figure 1.1 Configuration of the proposed V2V-communication-based active safety system.

1.2Reason for the idea

The motivation behind this project arises from the alarming increase in road accidents, many of which are caused by driver inattention, fatigue, and poor situational awareness. Despite continuous advancements in road infrastructure and traffic regulation systems, human error remains the dominant factor in traffic collisions. Even skilled drivers can be vulnerable to mistakes under stressful, monotonous, or high-speed driving conditions.

Traditional safety systems have proven insufficient in mitigating these risks. While modern vehicles are increasingly equipped with sensor-based ADAS technologies, these systems still face limitations such as restricted sensing ranges, reduced performance in poor environmental conditions, and high implementation costs—particularly for retrofitting older vehicles.

To overcome these challenges, the integration of Vehicle-to-Vehicle (V2V) communication offers a promising solution. V2V technology enables vehicles to exchange critical information such as speed, position, and direction, allowing for extended situational awareness that goes beyond the driver's line of sight. Unlike traditional sensor-based systems, V2V communication enhances decision-making by creating a collaborative environment among vehicles on the road.

Our approach introduces the use of ESP-NOW, a lightweight, low-latency communication protocol that enables direct peer-to-peer communication between microcontrollers such as the ESP32. This makes it an ideal choice for developing a cost-effective, scalable V2V communication system suitable for both academic experimentation and real-world application.

By combining ESP-NOW communication with key ADAS features—such as lane change warnings, blind spot alerts, and forward collision warnings—our system is designed to provide

real-time feedback and enhanced safety for drivers. Additional modules, including driver condition monitoring through an internal camera and traffic sign recognition via an external camera, further support situational awareness and accident prevention.

This project aims to provide an affordable and intelligent safety solution that enhances the driving experience while aligning with broader efforts by government and transportation authorities to reduce accident rates, save lives, and modernize transportation infrastructure.

1.3 Idea

The project aims to develop a comprehensive vehicle communication system that integrates multiple subsystems, each contributing unique features to enhance the safety and security of both the driver and the vehicle. These subsystems work together to improve overall driving safety, reduce risks, and provide a seamless driving experience. The proposed system includes:

1.3.1 Driver Face Recognition (Internal Camera for Driver Recognition)

This system uses an internal camera to ensure that the person entering the vehicle is the authorized driver. This adds an additional layer of security, preventing unauthorized access to the vehicle.

Figure 1.2 Driver Face Recognition.

Benefits:

- Increases personal security.

- Protects the vehicle from theft or unauthorized entry.
- Enhances user confidence in vehicle safety.
- Improves response in emergency situations.

1.3.2 Driver Monitoring System (Monitoring Driver's Condition)

This system monitors the driver's condition and alerts them when signs of fatigue or health instability are detected, ensuring the safety of both the driver and passengers.

Figure 1.3 Driver Monitoring System.

Benefits:

- Prevents accidents caused by drowsiness.
- Increases personal security.
- Enhances overall wellbeing.
- Boosts confidence in driving.

1.3.3 Traffic Sign Recognition (External Camera for Traffic Signs)

This system utilizes a camera to detect traffic signs and alerts the driver when important road signs are present, ensuring better adherence to road regulations.

Benefits:

Figure 1.4 Traffic Sign Recognition

- Enhances road safety and law compliance.
- Reduces the cost of traffic violations.
- Improves the overall driving experience.
- Increases driver focus.

1.3.4 DNPW – Do Not Pass Warning

This system assists the driver in avoiding overtaking in dangerous situations, such as narrow roads or oncoming vehicles, reducing accidents caused by unsafe overtaking.

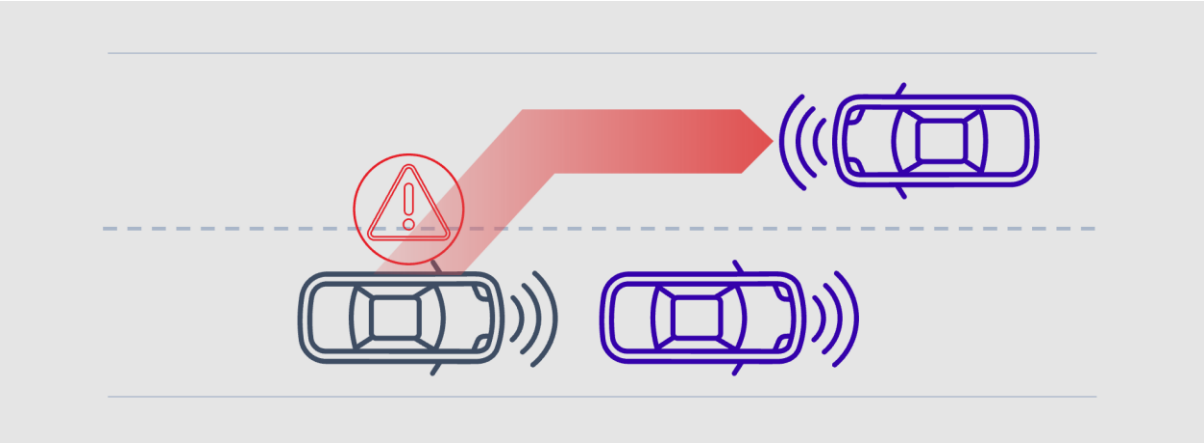


Figure 1.5 Do Not Pass Warning

Benefits:

- Reduces energy consumption.
- Increases driver safety.
- Reduces maintenance costs.
- Improves the driving experience.

1.3.5 FCW - Forward Collision Warning

This system warns the driver of potential frontal collisions with other vehicles or obstacles, allowing for quicker response times and the prevention of accidents.

Figure 1.6 Forward Collision Warning

Benefits:

- Reduces traffic congestion.
- Increases driver safety.
- Lowers maintenance costs.

1.3.6 BSW - Blind Spot Warning

This system alerts the driver of vehicles or obstacles in the blind spot, enhancing safety when changing lanes.

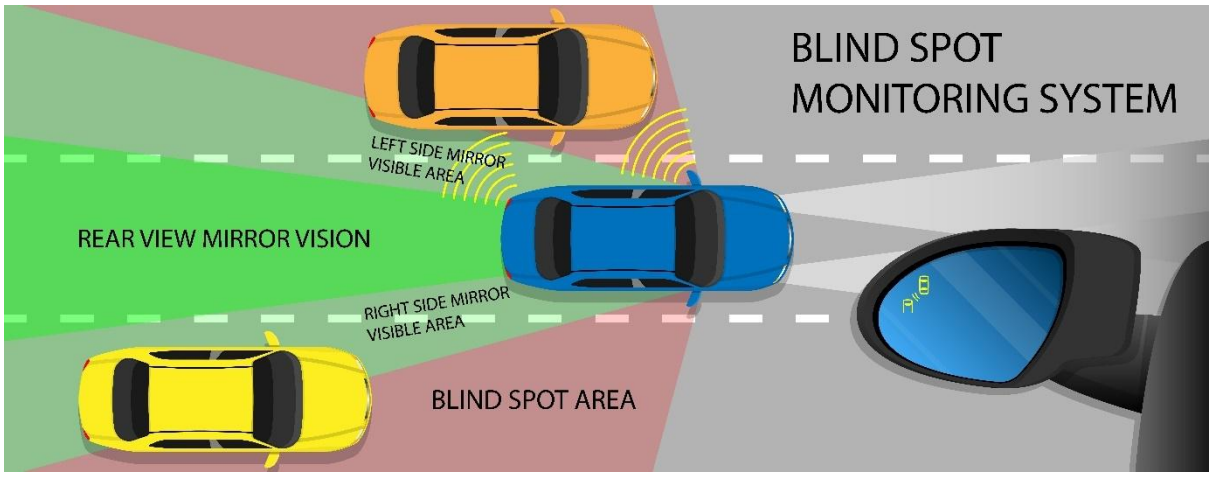


Figure 1.7 Blind Spot Warning

Benefits:

- Improves safety during lane changes.
- Reduces side-collision accidents.
- Decreases stress and increases confidence.
- Lowers insurance and maintenance costs.

1.3.7 EEBL - Emergency Electronic Brake Lights

Activated during sudden braking, this feature alerts other vehicles on the road, reducing rear-end collisions during emergencies.

Figure 1.8 Emergency Electronic Brake Lights.

Benefits:

- Reduces traffic congestion.
- Increases safety during emergency stops.
- Improves driver response time.

1.3.8 IMA - Intersection Movement Assist

This system monitors intersections and alerts the driver of potential risks such as oncoming vehicles from other directions, reducing accidents at intersections.

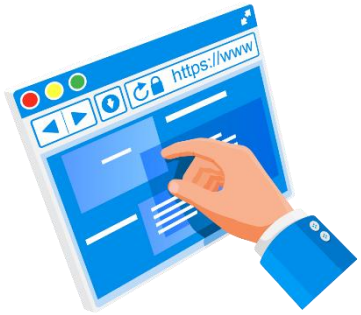
Figure 1.9 Intersection Movement Assist.

Benefits:

- Decreases accidents at intersections.
- Reduces travel time.
- Provides comfort and safety for the driver.
- Enhances driver confidence.

1.3.9 GUI for Driver Interaction

This interactive user interface allows the driver to receive alerts and information about the vehicle's overall condition, making it easier to manage the system.



Benefits:

- Provides a comfortable user experience.
- Easy control and navigation.
- Customizable settings based on user preferences.
- Enhances intelligent interaction between the driver and the vehicle.

1.3.10 Network Security with ESP-NOW and AES Encryption

This system ensures secure communication between the project's devices using ESP-NOW and AES encryption, protecting data and preventing unauthorized access.

Benefits:

- Enhances the system's security.
- Protects driver privacy and data.
- Reduces risks from cyberattacks.
- Increases user confidence in the technology used.

The primary beneficiaries of this project are the drivers and passengers, as the system significantly reduces risks during driving. Additionally, transportation companies and traffic authorities can also benefit from the system, as it provides safer and more efficient services on the roads.

Chapter 2

Chapter 2: System Analysis

2.1 Methodology

2.1.1 Overview

Agile is a modern methodology for project management and software development that emphasizes **flexibility**, **iterative progress**, and **continuous collaboration**. It enables teams to deliver working components frequently and to respond quickly to changes. For complex systems like **ADAS (Advanced Driver Assistance Systems)** and **V2V (Vehicle-to-Vehicle Communication)**, Agile provides a suitable framework to manage interdisciplinary tasks and rapidly evolving requirements.

2.1.2 Why Agile is Suitable for ADAS & V2V Project

The **ADAS & V2V project** involves the integration of subsystems such as:

- Face recognition
- Traffic sign recognition
- Collision warning
- Driver fatigue monitoring
- Secure vehicle communication

These systems require a flexible and adaptive approach to handle:

- Dynamic safety and security requirements.
- Continuous validation and testing of isolated components.
- Efficient management of complex, cross-functional development teams.

Agile allows the project to be divided into manageable functional modules that can be tested and integrated **incrementally**, ensuring each feature is validated before full system integration.

2.1.3 Scrum Framework in the Project

Scrum is the selected Agile framework used to manage this project. It focuses on **clear roles**, **structured events**, and **key artifacts** that guide the iterative development process.

Figure 2.1: Scrum in Agile Methodology

2.1.3.1 Scrum Roles in the Project

Product Owner:

Responsible for defining system features such as traffic sign analysis, driver alerts, and fatigue monitoring. The Product Owner prioritizes tasks in the Product Backlog based on safety and project value.

Scrum Master:

Facilitates the Scrum process, organizes meetings, ensures proper Scrum practices are followed, and removes obstacles that may hinder team progress.

Development Team:

Composed of cross-functional professionals including AI developers, embedded systems engineers, UI developers, and cybersecurity specialists. The team is responsible for delivering working system features within each sprint.

2.1.3.2 Scrum Events Implemented

Sprint Planning:

Conducted at the beginning of each sprint to select the tasks from the Product Backlog that will be completed within the sprint.

Daily Scrum (Daily Stand-Up):

Short, focused meetings to track progress, identify blockers, and align the team on daily goals.

Sprint Review:

Held at the end of each sprint to present completed features such as traffic sign recognition modules or driver fatigue alerts. Stakeholder feedback is collected for continuous improvement.

Sprint Retrospective:

Conducted after the Sprint Review to assess the team's performance, review what went well, what challenges were faced, and what can be improved in the next sprint.

2.1.3.3 Scrum Artifacts Used

Product Backlog:

A dynamic, prioritized list of all system features, maintained by the Product Owner. Examples include face recognition, collision warning, and vehicle communication protocols.

Sprint Backlog:

The set of specific tasks the team commits to completing during the current sprint. For example, completing the driver fatigue detection algorithm within the sprint cycle.

Increment:

A usable, testable system component delivered at the end of each sprint, such as a functional face recognition module or an operational traffic sign recognition system.

2.1.4 Scrum Execution in the Project

Requirement Gathering:

The system's functional needs are captured as User Stories that describe specific features like driver alert notifications and traffic sign recognition.

Release Planning:

Tasks are prioritized based on system safety and reliability. Each task is estimated using point-based techniques and assigned to a particular sprint or release phase.

Sprint Cycles:

Each sprint lasts between two to four weeks and focuses on developing, testing, and integrating specific system features.

Burndown Chart:

Used throughout each sprint to visualize progress and track remaining tasks, ensuring alignment with project deadlines and sprint goals.

Daily Scrum Meetings:

Regular stand-up meetings help the team address development challenges quickly and maintain continuous progress.

2.1.5 Benefits of Scrum in the ADAS & V2V Project

- Early detection of system flaws through frequent testing and validation.
- Faster delivery of functional features that can be tested and reviewed incrementally.
- Strong collaboration between AI, embedded systems, cybersecurity, and UI teams.
- Flexibility to adapt to new safety standards, technological updates, and evolving system requirements.

2.2 Development Plan

At the begging of any project, we need to consider a development plan contains well defined phases and deadlines to assure utilization of time during the project as well as Global design document which includes deep technical details about the whole project.

2.2.1 Project Timeline

This part includes our project deliverables, durations, deadlines, milestones of our project.

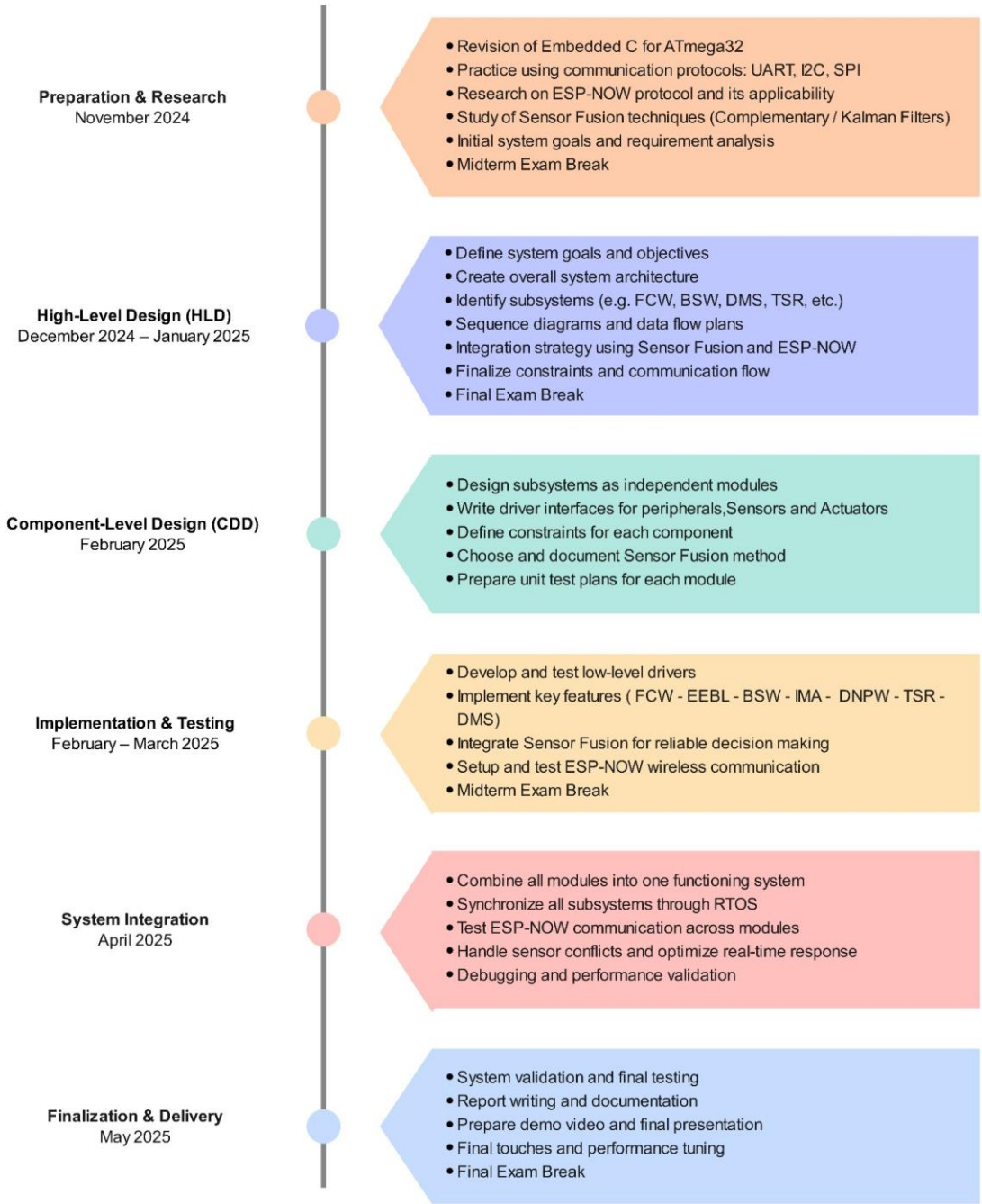


Figure 2.2: Project Timeline

2.3 Component Diagram

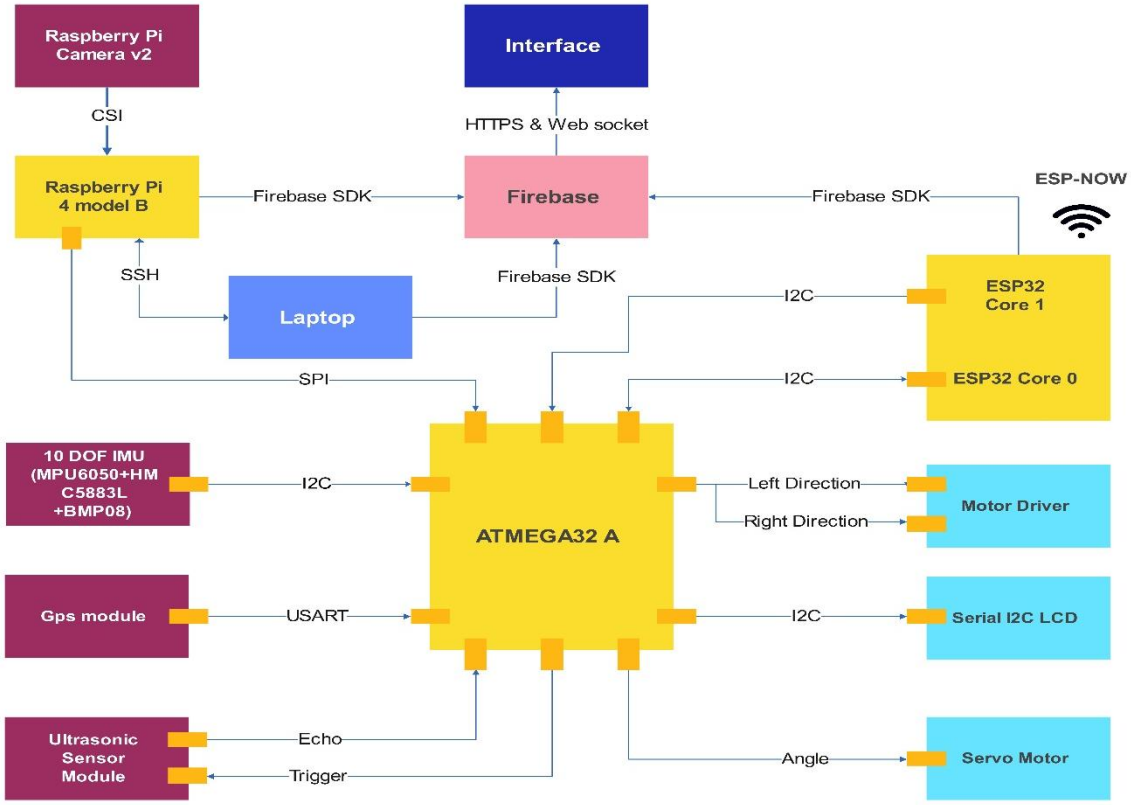


Figure 2.3: Component Diagram

2.4 Data Flow Diagrams

2.4.1 Context

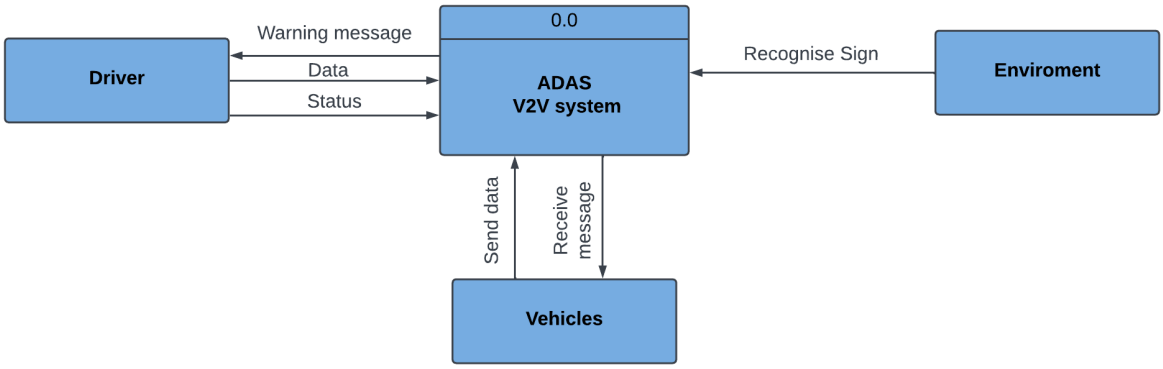


Figure 2.4: Context

2.4.2 Level 0

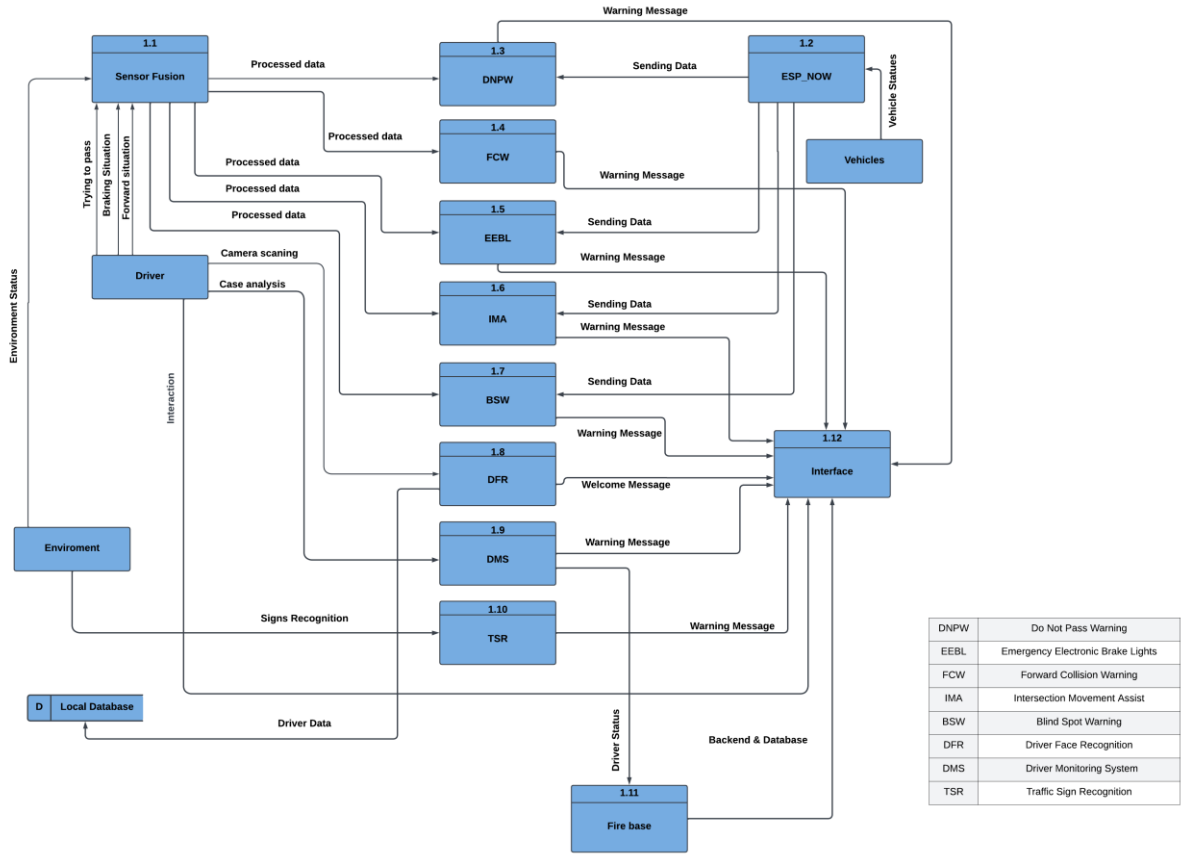


Figure 2.5: Level 0 of DFD

2.4.3 Level 1

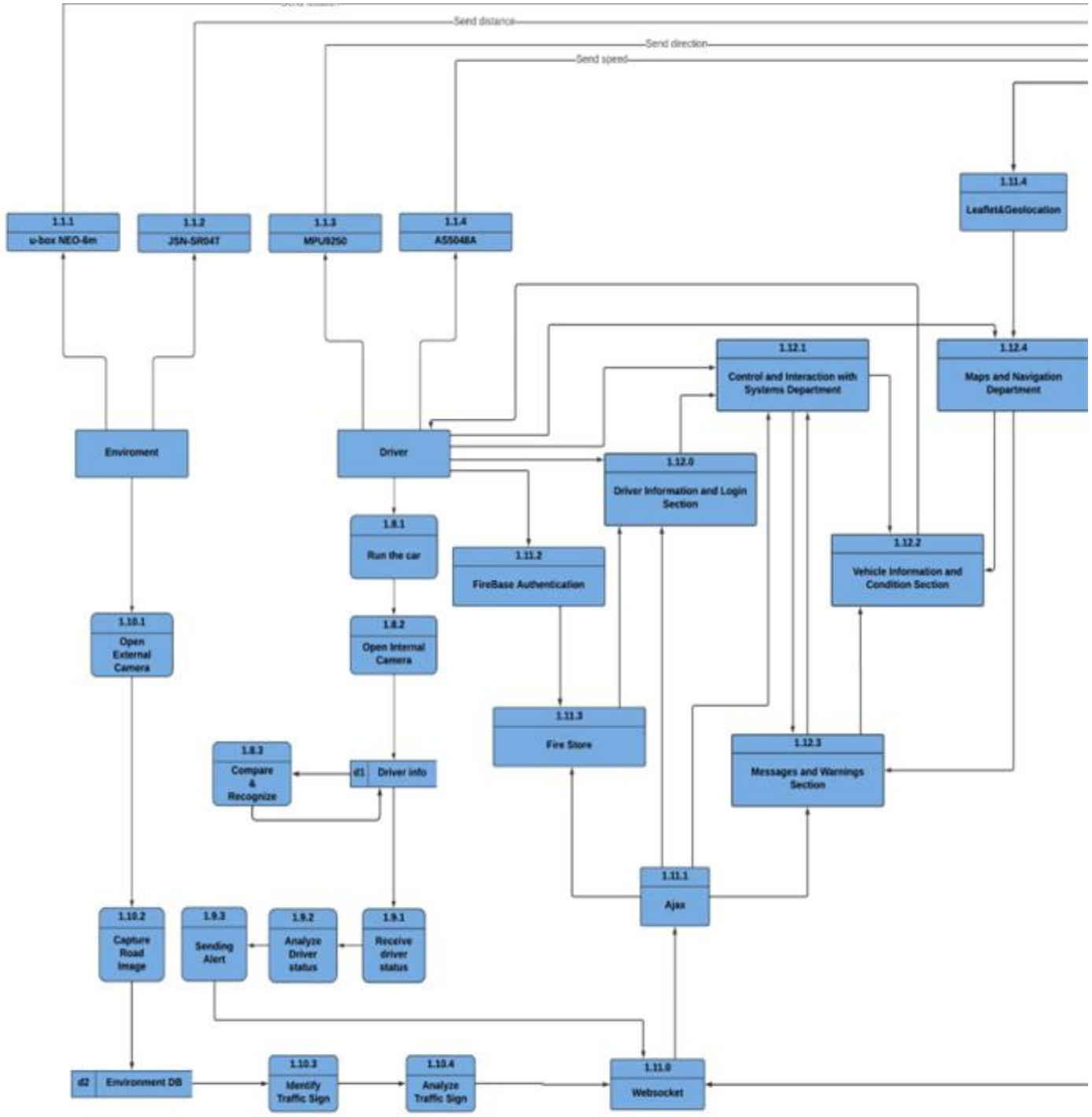


Figure 2.6 Level 1 of DFD(Part1)

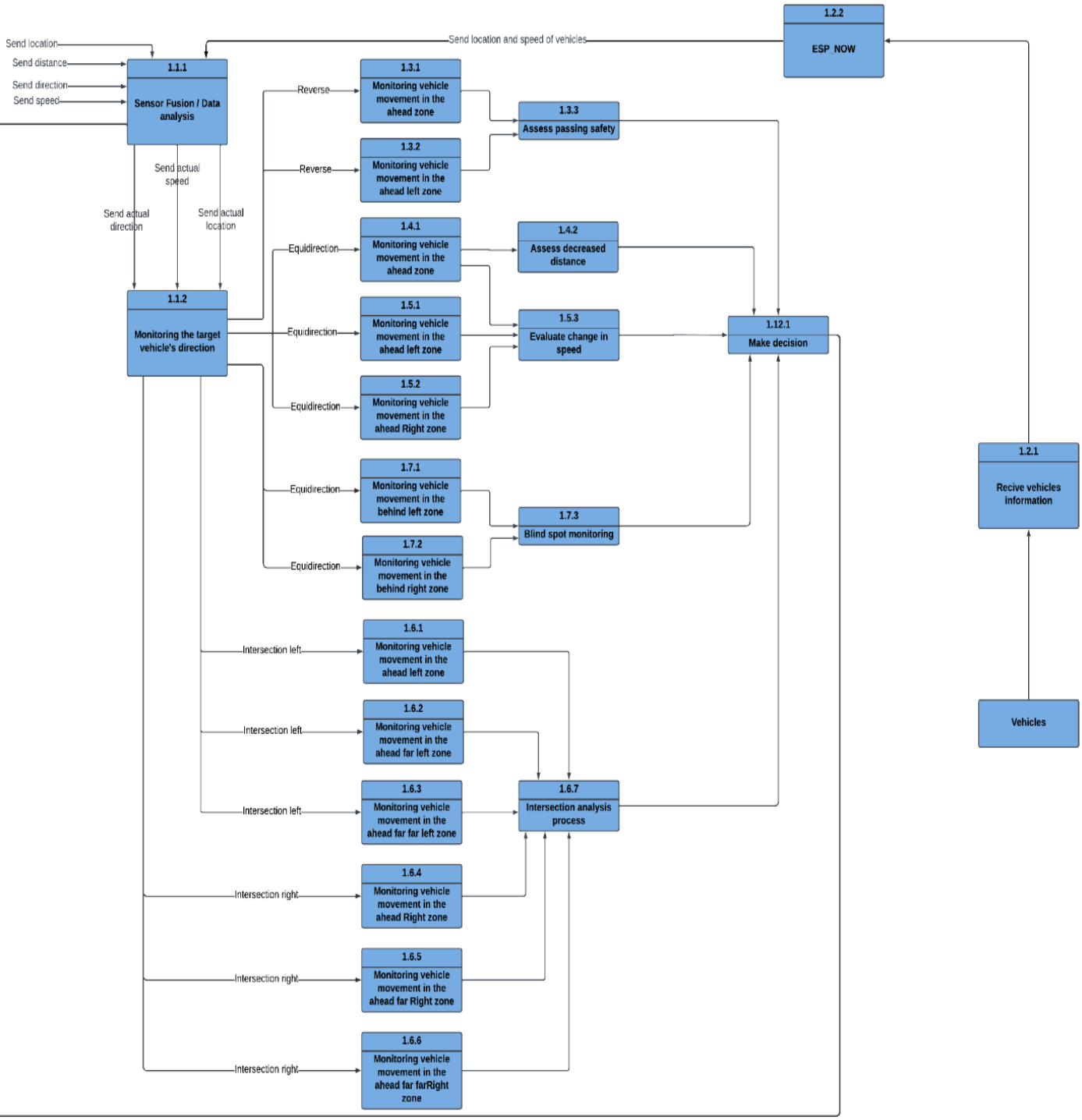
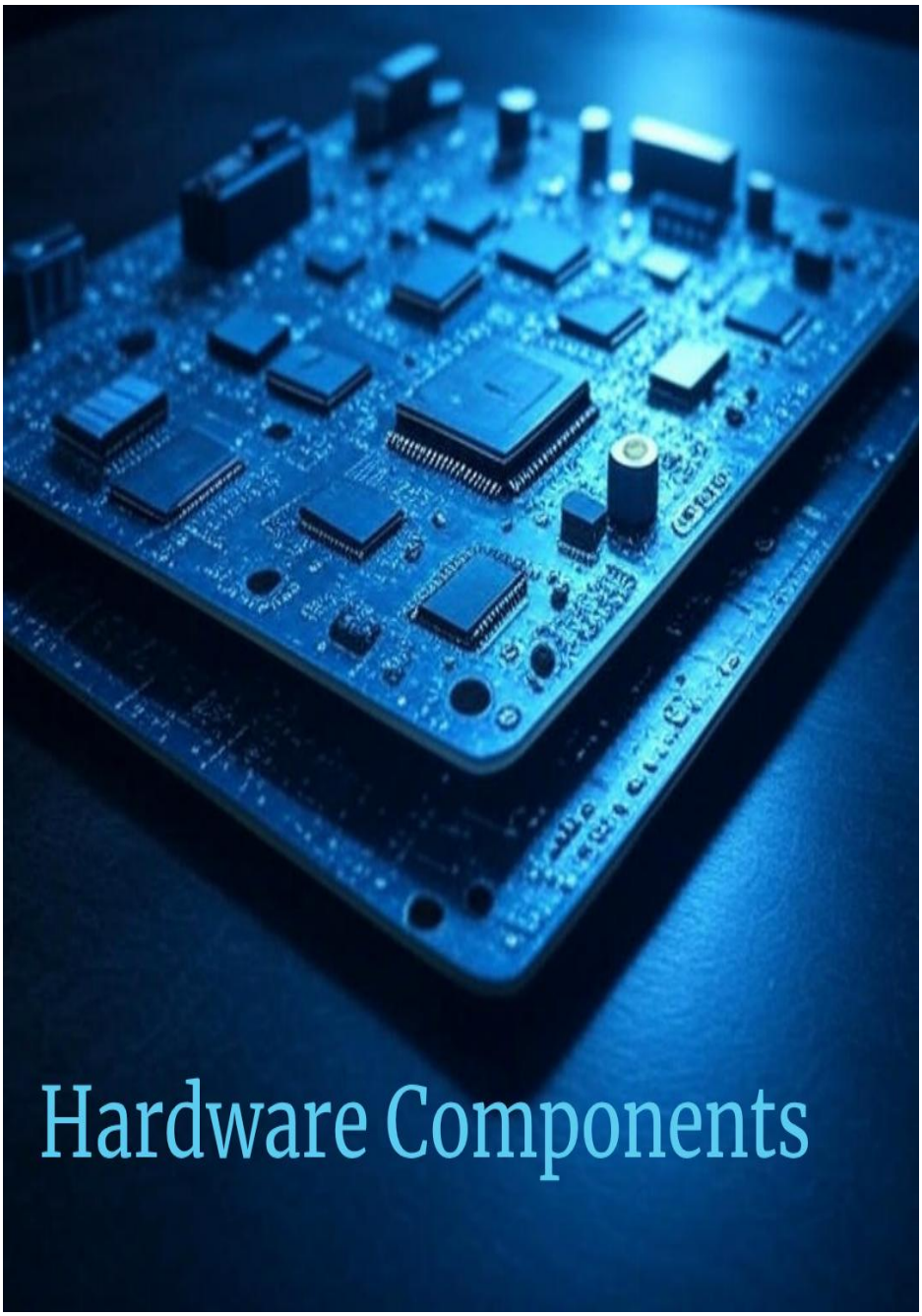


Figure 2.7 Level 1 of DFD(Part2)

Chapter 3



Chapter 3: Hardware Components

3.1 Overview

Hardware components are the building blocks of any project.

So, in this chapter we will introduce our components and display their functions and features.

3.2 Components

Our components are divided into 4 categories.

- Controllers
- Sensors
- Actuators
- Power
- Communication Interfaces

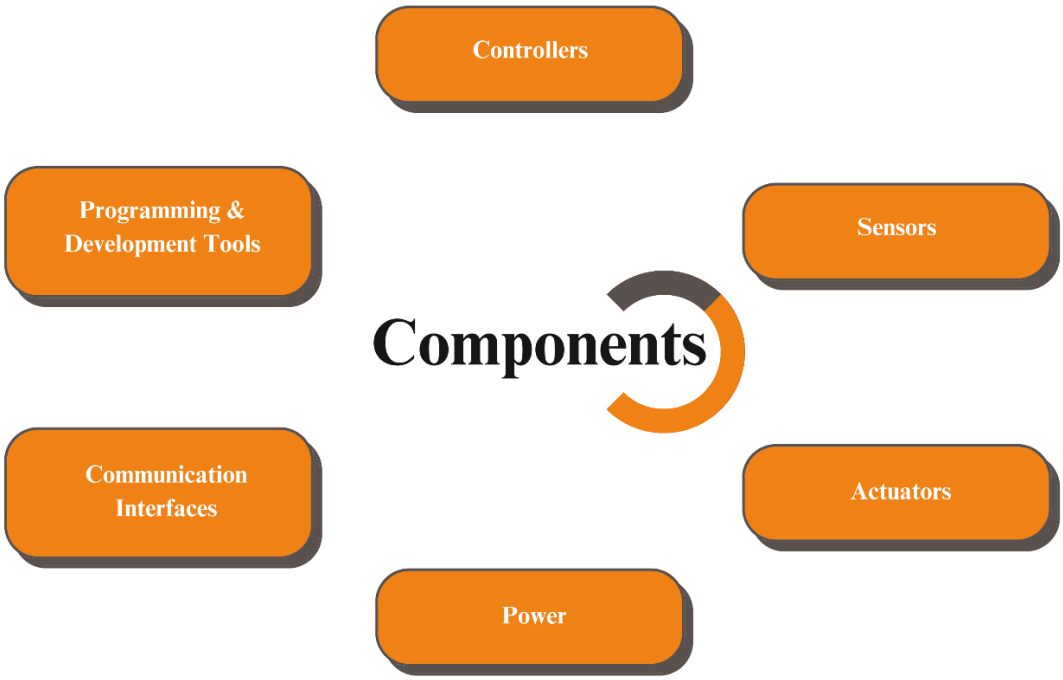


Figure 3.1 Components Categories

3.2.1 Controllers

3.2.1.1 Overview

An Electronic Controller uses electrical signals and digital algorithms to perform its receptive, comparative and corrective functions.

3.2.1.2 Components

3.2.1.2.1 ATmega32-8bit AVR

Overview

The ATmega32 is an 8-bit AVR microcontroller from Microchip (formerly Atmel), widely used in embedded systems. It features a RISC architecture, high performance, and low power consumption, making it ideal for control applications in graduation projects.

Figure 3.2 microchip ATmega32

Features

- CPU & Architecture
  - 8-bit AVR RISC core (16 MIPS at 16 MHz).
  - 32KB Flash memory for program storage.
  - 2KB SRAM & 1KB EEPROM for data.
- I/O & Peripherals
  - 32 Programmable I/O Pins (4 ports × 8 pins).
  - 8-channel 10-bit ADC for analog sensor interfacing.
  - 4 PWM Channels (for motor control, LEDs, etc.).
  - USART, SPI, I²C for serial communication.
- Timers & Counters
  - Three flexible timers (Timer0: 8-bit, Timer1: 16-bit, Timer2: 8-bit).
  - Supports PWM, input capture, and output compare.
- Interrupts & Power Management
  - 21 interrupt sources (external & internal).
  - Low-power modes (Idle, Power-down, Standby).

- Development & Debugging
    - Programmable via ISP (In-System Programming).
    - Supports debugging with JTAG (optional).

ATmega32 Microcontroller Pin diagram

It consists of 40-pin Dual Inline Package (DIP) of microcontroller integrated circuit.

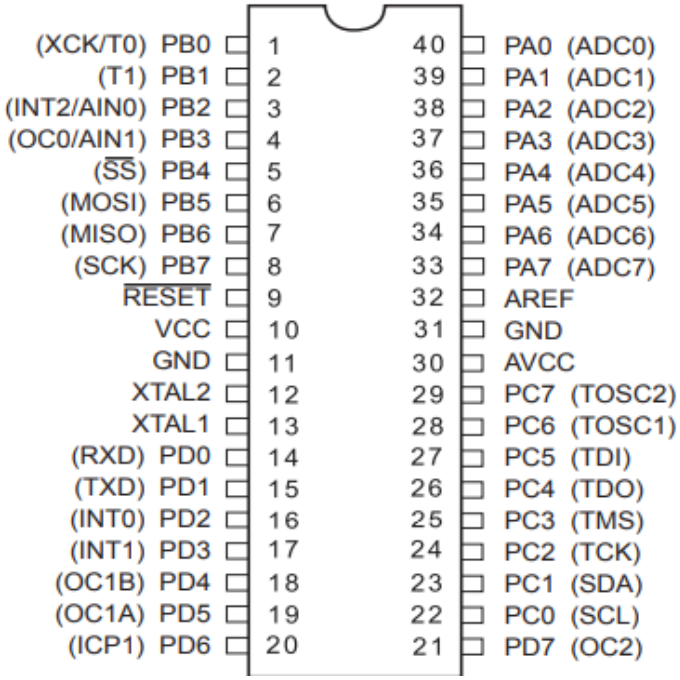


Figure 3.3 ATmega32-8bit Microcontroller pin datagram

3.2.1.2.2 ESP-32

Overview

The ESP32 is a powerful, low-cost Wi-Fi & Bluetooth microcontroller from Espressif, widely used in IoT, smart devices, and embedded systems. It combines a dual-core processor with wireless connectivity, making it ideal for connected projects.

Figure 3.4 ESP-32

Features

- Processing & Memory
    - Dual-core Xtensa LX6 CPU (up to 240 MHz).
    - 520KB SRAM + 4MB Flash (varies by model).
    - Supports external PSRAM (up to 16MB).
- Wireless Connectivity
    - Wi-Fi 802.11 b/g/n (2.4 GHz, AP/STA modes).
    - Bluetooth 4.2 + BLE (Bluetooth Low Energy).
- Peripherals & I/O
    - 34+ GPIOs (most configurable: ADC, DAC, PWM, UART, etc.).
    - 12-bit SAR ADC (up to 18 channels).
    - 8-bit DAC (2 channels).
    - Motor PWM, Hall sensor, capacitive touch.
- Communication Interfaces
    - SPI, I2C, I2S, UART, CAN, Ethernet (RMII).
    - SD card interface.
- Power Efficiency
    - Ultra-low power modes (deep sleep at ~5µA).
    - Dynamic clock scaling (adjusts power usage).
- Development Support
    - Arduino IDE, ESP-IDF (Espressif's SDK), MicroPython.
    - OTA (Over-the-Air) updates.
    - Built-in security (AES, SHA, RSA, secure boot).

3.2.1.2.3 Raspberry Pi5 Model B 8GB

Overview

The Raspberry Pi 5 is the latest high-performance single-board computer (SBC) from the Raspberry Pi Foundation. With an 8GB RAM variant, it delivers desktop-level computing power in a compact form factor, making it ideal for advanced projects, edge computing, and multimedia applications.

Figure 3.5 Raspberry Pi5 Model B 8GB

Features

1. Processing Power

- Quad-core Cortex-A76 CPU (2.4 GHz, 64-bit) – 2–3× faster than Pi 4.
- VideoCore VII GPU (supports OpenGL ES 3.1, Vulkan 1.2).
- 8GB LPDDR4X RAM (for multitasking & memory-intensive apps).

2. Connectivity & I/O

- Dual 4K HDMI (60Hz) – Supports dual displays.
- Gigabit Ethernet (with PoE+ support via HAT).
- Wi-Fi 5 (802.11ac) + Bluetooth 5.0 / BLE.
- 2 × USB 3.0 (5Gbps) + 2 × USB 2.0 ports.
- PCIe 2.0 interface (for NVMe SSDs or custom HATs).

3. Storage & Expansion

- MicroSD slot (UHS-I, for OS/boot).
- Support for NVMe SSDs via PCIe (requires adapter).
- 40-pin GPIO (backward-compatible with Pi 4).
- Real-Time Clock (RTC) with battery connector.

4. Power & Efficiency

- USB-C power (5V/5A recommended, supports PD).
- New power button & low-power states.
- Better thermal management (heatsink recommended for sustained loads).

5. Software & OS Support

- Raspberry Pi OS (Debian-based, 64-bit optimized).
- Windows on Raspberry Pi (WoR), Ubuntu, Kali Linux.
- Docker, Kubernetes, AI/ML frameworks (TensorFlow Lite).

3.2.2 Sensors

3.2.2.1 Overview

An Electronic Controller uses electrical signals and digital algorithms to perform its receptive, comparative and corrective functions.

3.2.2.2 Components

3.2.2.2.1 Raspberry Pi Camera Module V2

Overview

The Raspberry Pi Camera Module V2 is an official 8MP Sony IMX219 sensor camera designed for Raspberry Pi boards. It connects via a CSI (Camera Serial Interface) ribbon cable and is widely used in computer vision, surveillance, timelapse photography, and robotics projects.

Figure 3.6 Raspberry Pi Camera Module V2

Features

- **Image Sensor & Quality**
  - 8MP Sony IMX219 sensor (3280 × 2464 pixels).
  - Fixed-focus lens (with manual adjustability in some models).
  - Supports 1080p30, 720p60, and VGA90 video.
- **Connectivity & Compatibility**
  - CSI-2 interface (works with all Raspberry Pi models, including Pi 5).
  - 15-pin ribbon cable (included, easy to install).
  - Compatible with Raspberry Pi OS & third-party OS.
- **Software & Libraries**
  - Official Raspberry Pi Camera API (libcamera, picamera).

- OpenCV, Python support (for AI/computer vision).
- Motion detection, timelapse, and streaming (via VLC, FFmpeg).

- **Low-Light & Special Modes**
  - HDR (High Dynamic Range) support.
  - Night vision (with IR-cut filter removal + IR light source).
- **Power Efficiency**
  - Low power consumption (~250mA).
  - No external power needed (draws power from Pi).

3.2.2.2.2 Ublox NEO-7M GP

Overview

The u-blox NEO-7M is a high-performance GPS receiver module with 56-channel u-blox 7 engine, offering fast acquisition and tracking. It's widely used in drones, vehicle tracking, and IoT location-based projects.

Figure 3.7 Ublox NEO-7M GP

Features

- **GPS Performance**
  - 56-channel u-blox 7 engine (supports GPS & GLONASS).
  - High sensitivity (-167 dBm tracking, -148 dBm acquisition).
  - Position accuracy: 2.5m (GPS), 4m (GLONASS).
  - Update rate: Up to 5Hz (configurable).
- **Connectivity & Interfaces**
  - UART (TTL serial) default interface (3.3V or 5V compatible).
  - Optional USB, SPI, or DDC (I2C) with firmware update.
  - 1PPS (Pulse-Per-Second) output for timing applications.
- **Power & Efficiency**
  - Low power consumption (~40mA active, 11mA backup).
  - Power-saving modes (continuous, periodic, eco mode).
  - Operates at 3.3V–5V (flexible power input).

- **Additional Features**
  - Built-in EEPROM (stores configuration settings).
  - Supports SBAS (WAAS, EGNOS, MSAS) for better accuracy.
  - Jamming & spoofing detection (basic interference resistance).

3.2.2.2.3 IMU 10DOF

Overview

The IMU 10DOF (10 Degrees of Freedom) is a multi-sensor module combining accelerometer, gyroscope, magnetometer (compass), and barometric pressure/altitude sensors in a single compact unit. It is widely used in drones, robotics, motion tracking, and navigation systems for precise orientation and movement detection.

Figure 3.8 IMU 10DOF

Features

- **Sensor Breakdown (10DOF = 3+3+3+1 Axes)**
  - 3-Axis Accelerometer (measures linear acceleration).
  - 3-Axis Gyroscope (measures angular velocity).
  - 3-Axis Magnetometer (measures Earth's magnetic field for compass heading).
  - 1 Barometric Pressure Sensor (estimates altitude).
- **Communication & Interfaces**
  - I2C (default) and SPI for easy microcontroller interfacing.
  - Compatible with Arduino, Raspberry Pi, ESP32, STM32.
  - Precision Analog-to-Digital Converters (ADCs) for accurate readings.
- **Performance & Calibration**
  - High-resolution sensors (e.g., ±2g/4g/8g/16g selectable accelerometer range).
  - Onboard Digital Motion Processor (DMP) (in some models, e.g., MPU-6050 + HMC5883L + BMP180).
  - Software calibration (for gyro drift compensation).

- **Power & Efficiency**
    - Low power consumption (~3.3V–5V operation).
    - Sleep modes for battery-powered applications.

3.2.2.2.4 Ultrasonic Distance Sensor – HC-SR04

Overview

The HC-SR04 is a low-cost ultrasonic distance sensor that measures object proximity (2cm–4m) using sound waves (40kHz). It's widely used in robotics, obstacle avoidance, and level monitoring.

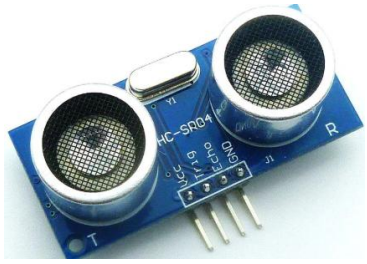


Figure 3.9 Ultrasonic Distance Sensor – HC-SR04

Features

- Measurement Specifications
    - Range: 2cm – 400cm (theoretical), 2cm–200cm (practical).
    - Accuracy: ±3mm (best case).
    - Resolution: 0.3cm.
    - Beam angle: ~15°.
- Interface & Power
    - Trig (Input): 10µs TTL pulse to initiate measurement.
    - Echo (Output): TTL pulse (duration = round-trip time).
    - Voltage: 5V DC (3.3V logic compatible for Echo).
    - Current draw: ~15mA.
- Performance
    - Non-contact detection (works in air, not in vacuum/underwater).
    - Unaffected by color or light, but sensitive to soft materials (e.g., fabric).

3.2.3 Actuators

3.2.3.1 Overview

An Electronic Controller uses electrical signals and digital algorithms to perform its receptive, comparative and corrective functions.

3.2.3.2 Components

3.2.3.2.1 LCD Graphical screen

Overview

A 20x4 Graphical LCD combines alphanumeric and graphical display capabilities, allowing both text and custom graphics (unlike standard 20x4 character LCDs). It is commonly used in embedded systems, HMIs, and data visualization projects.

Figure 3.10 LCD Graphical screen

Features

- **Display Specifications**
  - Resolution: 128x64 pixels (supports 20x4 text + graphics).
  - Technology: Monochrome (ST7920 controller) or TFT (color variants).
  - Backlight: LED (white/blue/green) or RGB adjustable.
- **Interface & Compatibility**
  - Parallel (8-bit) or Serial (SPI/I2C) modes.
  - Works with Arduino, Raspberry Pi, STM32, ESP32.
  - Built-in character generator (supports ASCII + custom fonts).
- **Graphics Capabilities**
  - Custom bitmaps (logos, icons).
  - Dynamic graphs (waveforms, bar charts).
  - Touchscreen options (on some TFT models).
- **Power & Efficiency**
  - Voltage: 3.3V or 5V (check datasheet).
  - Current: ~10–50mA (backlight-dependent).

3.2.3.2.2 L298 Motor driver

Overview

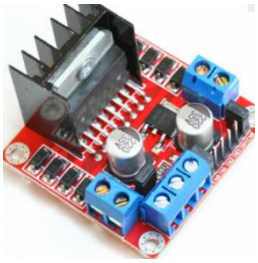


Figure 3.11 L298 Motor driver

The L298N is a dual H-bridge motor driver capable of controlling two DC motors or one stepper motor (up to 2A per channel). It's widely used in robotics, DIY vehicles, and automation projects for bidirectional motor control.

Features

- **Motor Control Specifications**
  - Voltage Range: 5V–35V (motor supply), 5V (logic supply).
  - Current Capacity: 2A per channel (peak), 0.5–1A continuous (requires heatsink).
  - Control Modes:
    - PWM speed control (for variable speed).
    - Direction control (forward/reverse/brake).
- **Interface & Compatibility**
  - Inputs: IN1, IN2, IN3, IN4 (for motor direction).
  - Enable Pins: ENA, ENB (PWM for speed control).
  - Supports Arduino, Raspberry Pi, STM32, ESP32.
- **Built-in Protection**
  - Flyback diodes (protect against voltage spikes).
  - Thermal shutdown (overheating protection).
- **Power Options**
  - Onboard 5V regulator (can power logic from motor supply if <12V).
  - Separate logic/motor power (recommended for high voltage).

3.2.3.2.3 Small Buzzer 5V

Overview

Figure 3.12 Small Buzzer 5V

A 5V buzzer is an audio signaling device used in alarms, notifications, and interactive projects. It comes in two types: Active Buzzer: Generates a fixed tone (e.g., 2.7kHz) when powered. Passive

Buzzer: Requires PWM to control tone/frequency (versatile for melodies).

Features

- Electrical Specifications
  - Voltage: 5V, but often works at 3.3V–12V.
  - Current Draw: ~30mA (active), <20mA (passive).
  - Sound Output: 85dB (typical) at 10cm distance.
- Physical Design
  - Compact size (typically 12mm–16mm diameter).
  - Polarity sensitive (active buzzers: +/– marked).

3.2.3.2.4 DC Motor JGA25

Overview

The JGA25 is a compact, high-speed DC motor with an integrated reduction gearbox, commonly used in robotics, DIY projects, and automation. It is available in various voltage ratings (3V–12V) and gear ratios (e.g., 1:48, 1:120), balancing speed and torque.

Figure 3.13 DC Motor JGA25

Features

- **Mechanical Specifications**
  - Gearbox Type: Metal/plastic gears (varies by model).
  - Shaft Type: D-shaped shaft (prevents slipping).
  - No-Load Speed: 100–300 RPM (depends on gear ratio).
  - Stall Torque: 0.5–2 kg·cm (varies by voltage/gearing).
- **Electrical Specifications**
  - Operating Voltage: 3V, 6V, or 12V (check model).
  - No-Load Current: 60–150mA (lower voltage = lower current).
  - Stall Current: 500mA–1.5A (requires motor driver).
- **Physical Dimensions**
  - Size: ~25mm diameter × 37mm length (motor + gearbox).
  - Weight: ~50–80g (depends on gearbox material).

3.2.3.2.5 MG996R Servo Motor

Figure 3.14 MG996R Servo Motor

Overview

The MG996R is a high-torque 180-degree (or 360-degree modified) metal-gear servo motor, widely used in robotics, RC vehicles, and automation projects. It offers strong torque (10–12 kg·cm) and durable construction, making it ideal for heavy-duty applications.

Features

- **Mechanical Specifications**
  - Rotation Range: 180° (standard) but can be modified for continuous rotation.
  - Torque: 10–12 kg·cm (4.8V–7.2V).
  - Speed: 0.19s/60° (4.8V), 0.15s/60° (6V).
  - Gear Material: Metal gears (durable but noisy).
- **Electrical Specifications**
  - Operating Voltage: 4.8V–7.2V (recommended 6V for best performance).
  - Current Draw: 500–900mA (stall), 10mA (idle).
  - PWM Signal: 50Hz (20ms period), 1–2ms pulse width for 0–180°.
- **Physical Dimensions**
  - Weight: 55g.
  - Size: 40.7 × 19.7 × 42.9mm.
  - Cable Length: 30cm (with JR-type connector).

3.2.4 Power

3.2.4.1 Overview

An Electronic Controller uses electrical signals and digital algorithms to perform its receptive, comparative and corrective functions.

3.2.4.2 Components

3.2.4.2.1 Lithium Battery

Overview

Lithium batteries (Li-ion/LiPo) are rechargeable power sources known for their high energy density, lightweight, and long cycle life. They are widely used in portable electronics, drones, robotics, and IoT devices.

Figure 3.15 Lithium Battery

Features

- **Electrical Specifications**
  - Charging Voltage:
    - Li-ion/LiPo: 4.2V (max) per cell.
    - LiFePO4: 3.6V (max) per cell.
  - Discharge Cutoff:
    - Li-ion/LiPo: 3.0V (min safe voltage).
    - LiFePO4: 2.5V (min safe voltage).
  - Current Rating: Varies by cell (e.g., 1C–30C discharge rate).
- **Protection & Management**
  - PCM/PCB (Protection Circuit Module): Prevents overcharge/discharge.
  - BMS (Battery Management System): Balances multi-cell packs (e.g., 2S–6S).
  - Thermal Protection: Critical for LiPo (risk of swelling/fire).

3.2.5 Communication Interfaces

3.2.5.1 Overview

An Electronic Controller uses electrical signals and digital algorithms to perform its receptive, comparative and corrective functions.

3.2.5.2 Components

3.2.5.2.1 I2C /SPI LCD Backpack

Overview

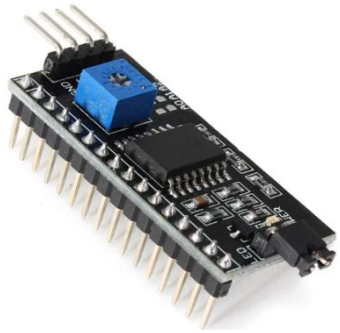


Figure 3.16 I2C /SPI LCD Backpack

An I2C/SPI LCD Backpack is an interface adapter that simplifies connecting character LCDs (e.g., 16x2, 20x4) to microcontrollers (Arduino, Raspberry Pi, etc.) by reducing wiring complexity. It converts parallel LCD communication to I2C or SPI, saving GPIO pins and enabling plug-and-play use.

Features

- **Compatibility**
    - Supports HD44780-based character LCDs (16x2, 20x4, etc.).
    - Works with I2C (common) or SPI (less common) interfaces.
    - Compatible with 5V and 3.3V systems (check voltage tolerance).
- **Interface Options**
    - I2C Default: Uses 2 wires (SDA, SCL) + power.
        - Address often 0x27 or 0x3F (configurable via jumpers).
    - SPI Variants: Faster but requires CS, MOSI, CLK pins.
- **Onboard Components**
    - PCF8574 (I2C) or shift registers (SPI) for signal conversion.
    - Potentiometer for contrast adjustment (on some models).
    - Backlight control (enabled via I2C commands).
- **Power & Efficiency**
    - Low power (~1–5mA for backlight, <1mA without).
    - 3.3V/5V logic (I2C pull-ups may be needed for 3.3V MCUs).

3.2.6 Programming & Development Tools

3.2.6.1 Overview

An Electronic Controller uses electrical signals and digital algorithms to perform its receptive, comparative and corrective functions.

3.2.6.2 Components

3.2.6.2.1 USBASP AVR Programmer

Overview

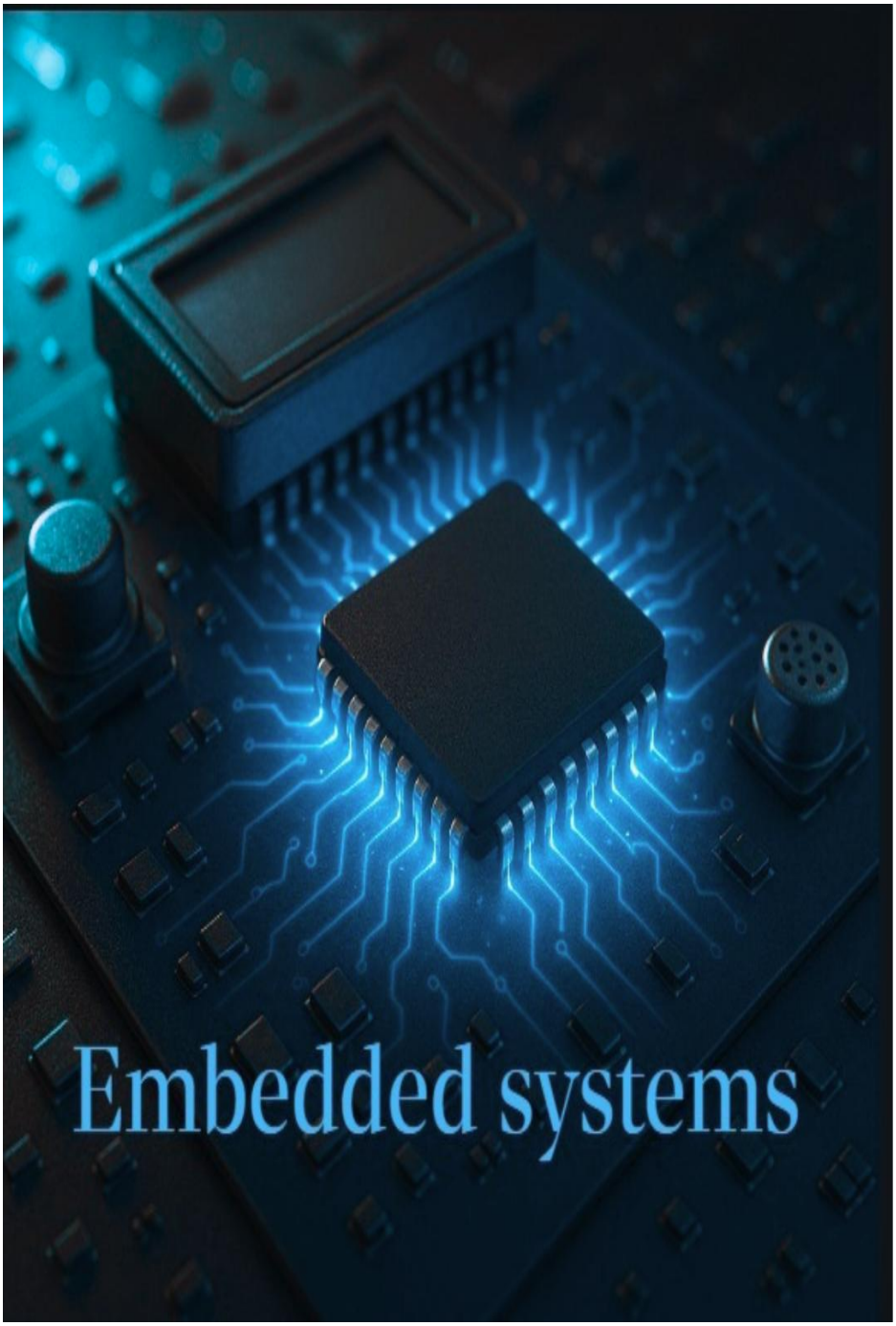
The USBASP is a popular, low-cost AVR programmer used to flash code onto AVR microcontrollers (like ATmega32, ATmega328P, etc.). It communicates via USB and supports In-System Programming (ISP), making it a common tool for embedded projects.

Figure 3.17 USBASP AVR Programmer

Features

- Compatibility
  - Works with most AVR 8-bit microcontrollers (ATmega, ATtiny series).
  - Supports ISP (In-System Programming) via 6-pin or 10-pin header.
- Programming Modes
  - Firmware upgradable (can be updated for new AVR chips).
  - Supports fuse bit programming (configuring clock, reset, etc.).
- Connectivity & Speed
  - USB 2.0 interface (plug-and-play on most OS).
  - Adjustable programming speed (slower for stability, faster for efficiency).
- Power Supply Options
  - Can power the target MCU (5V/3.3V) or use external power.
  - Auto-detects target voltage.
- Software Support
  - Works with AVRDude (command-line flashing).
  - Compatible with ProgISP, Khazama AVR, Arduino IDE (with modifications).

Chapter 4



Chapter 4: Embedded Systems

4.1 Brief of Embedded System

An embedded system is a specialized computer system designed to perform specific tasks or functions within a larger system. It integrates hardware and software, often with real-time constraints, and is embedded as part of a device or product, such as appliances, vehicles, or medical equipment. These systems are typically optimized for efficiency, reliability, and low power consumption.

4.2 Characteristics of Embedded Systems

1. Dedicated Functionality: Designed for specific tasks, unlike general-purpose computers.
2. Real-Time Operation: Often operates under strict timing constraints to ensure timely responses.
3. Resource Constraints: Limited processing power, memory, and storage to minimize cost and power usage.
4. High Reliability: Built to operate continuously with minimal failures, often in critical applications.
5. Low Power Consumption: Optimized for energy efficiency, especially in battery-powered devices.
6. Compact Size: Small form factor to fit within the host device.
7. Minimal User Interface: Often lacks extensive displays or input devices, using simple interfaces like LEDs or buttons.
8. Customized Hardware-Software Integration: Tailored hardware and software work closely to meet specific needs.

4.3 Types of Embedded Systems

4.3.1 Stand-Alone Embedded Systems:

- Operate independently without needing a host system.
- Examples: Digital cameras, MP3 players, microwave ovens.

4.3.2 Real-Time Embedded Systems:

- Designed to process data and respond within strict time constraints.
- Subtypes:

- Soft Real-Time: Tolerates minor delays (e.g., multimedia streaming).
- Hard Real-Time: No delays allowed, critical for safety (e.g., airbag systems, flight control).

4.3.3 Networked Embedded Systems:

- Connected to a network for communication and data exchange.
- Examples: IoT devices, smart home systems, networked medical equipment.

4.3.4 Mobile Embedded Systems:

- Used in portable devices with constraints on size, power, and weight.
- Examples: Smartphones, wearables, GPS devices.

4.3.5 Small-Scale Embedded Systems:

- Use simple microcontrollers with minimal resources for basic tasks.
- Examples: Electronic toys, simple sensors.

4.3.6 Medium-Scale Embedded Systems:

- More complex, with moderate processing power and memory.
- Examples: ATMs, industrial control systems.

4.3.7 Sophisticated Embedded Systems:

- High-performance systems with advanced processors and large memory.
- Examples: Automotive control units, advanced medical imaging systems.

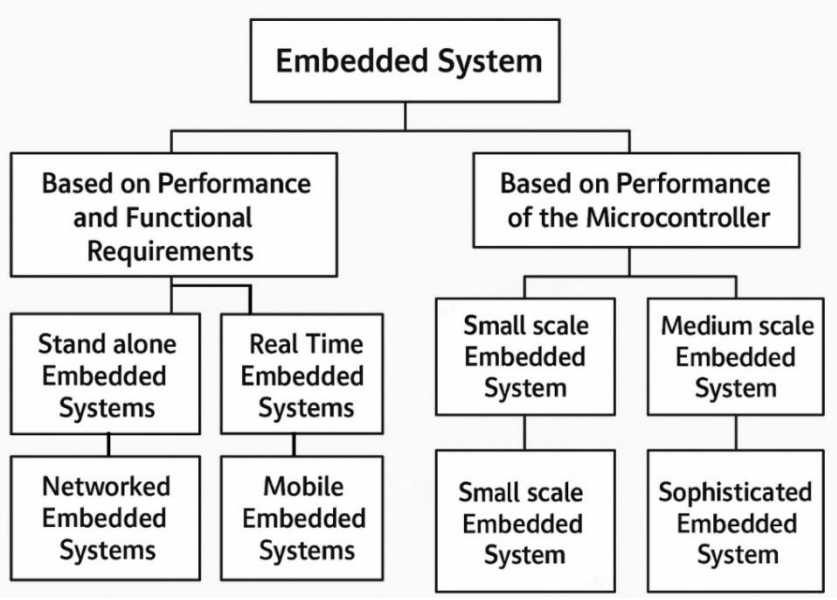


Figure 4.1 Types of the embedded systems

4.4 V2X Communication

In the rapidly evolving world of automotive technology and smart mobility, one concept stands out as a cornerstone for the future of safe and intelligent transportation: **V2X Communication**, or **Vehicle-to-Everything Communication**. This term refers to a sophisticated wireless communication system that enables vehicles to interact with their surrounding environment — including other vehicles (**V2V**), road infrastructure like traffic lights (**V2I**), pedestrians (**V2P**), cloud networks (**V2N**), and even devices such as bicycles or mobile phones (**V2D**).

The significance of **V2X** lies in its potential to **dramatically improve road safety** by allowing real-time exchange of crucial information such as location, speed, and direction between vehicles and nearby entities. This shared data empowers drivers — and autonomous systems — to make smarter, faster decisions to prevent collisions. Additionally, **V2X** contributes to greater traffic efficiency, reduced fuel consumption, and lower emissions by enabling smoother traffic flow and better coordination with traffic signals. Another mode of communication is Vehicle to Infrastructure (**V2I**). **V2I** communication is the exchange of information between vehicles and roadside infrastructure. For example, a RoadSide Unit (**RSU**) can be connected to a signal controller at an intersection. RSU sends out intersection map (**MAP**) message and traffic signal status message known as Signal Phase and Timing (**SPaT**) message. Next in line is **Vehicle to Pedestrian** (V2P) communication. **V2P** involves exchanging information between vehicles and pedestrians. For example, DSRC enabled smartphones can serve as a pedestrian communication device. The implementation of the **V2X** in a vehicle consists of several standard components. The components include localization device, computation platform, HMI interfaces and DSRC transceiver. Collection of these components is commonly called On-Board Equipment (**OBE**). Fully implemented **OEM V2X** system may include connections to the internal vehicle bus. Aftermarket devices may not have access to the internal vehicle bus. There are two main technologies behind V2X communication:

1. **DSRC (Dedicated Short-Range Communication)** – based on the IEEE 802.11p standard, offering low-latency communication over short distances.
2. **C-V2X (Cellular V2X)** – which uses cellular networks (4G/5G) to provide broader coverage and higher reliability, especially in complex urban environments.

As smart cities grow and autonomous vehicles become more common, V2X is emerging as a vital building block in creating a more **connected, intelligent, and safer transportation ecosystem**.

The DSRC (Dedicated Short-Range Communications) system is not implemented in our project due to regional frequency allocation constraints, as the frequency bands designated for **DSRC are reserved for higher-priority applications**. As an alternative, we have adopted **ESP-based ESP-NOW** communication modules, which operate licensed frequency bands. This technology offers ease of integration, broad compatibility with various systems, and efficient short-range data exchange, making it a practical and effective solution for our project requirements. **(ESP-NOW Protocol as mentioned in Chapter 5)**

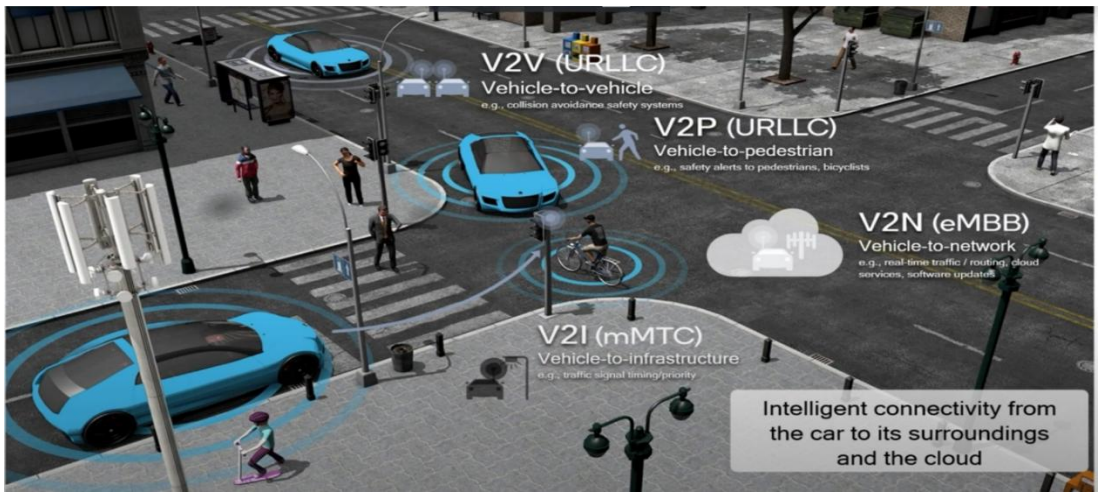


Figure 4.2 Types of the V2X Communication

V2X communication encompasses various modes of interaction between vehicles and their surroundings, including infrastructure, pedestrians, networks, and other vehicles. Among these, one of the most essential and foundational forms is Vehicle-to-Vehicle (V2V) communication. By focusing specifically on the direct exchange of safety-critical data between vehicles, V2V plays a central role in realizing the full potential of V2X systems in improving road safety and traffic efficiency. Exploring how V2V works provides deeper insight into the core mechanisms that drive intelligent transportation.

How does vehicle-to-vehicle (V2V) communication enhance crash avoidance compared to traditional on-board sensor technologies?

4.4.1 V2V Vehicle Safety Communication

One of the key enablers of intelligent transportation systems is **Vehicle-to-Vehicle (V2V)** communication, a core component of V2X technology. V2V allows vehicles to wirelessly exchange critical data in real time—such as speed, location, acceleration, brake status, and direction of travel—which significantly enhances situational awareness among nearby vehicles. This plays a vital role in preventing collisions by enabling earlier responses to potential hazards. Unlike conventional sensors like cameras and radars that rely on direct line-of-sight and are limited in range, **V2V offers 360-degree** coverage and performs reliably even in low-visibility or adverse weather conditions. It enables vehicles to "see" around corners and through obstacles such as buildings or other vehicles, going far beyond the capabilities of today's on-board crash avoidance technologies. These current systems depend solely on sensor input to warn drivers or intervene in dangerous situations but cannot "predict" a crash involving a vehicle that is not directly visible. V2V communication fills this critical gap and can drastically reduce crashes. In fact, it is estimated that V2X communication could help prevent **up to 80%** of crashes involving unimpaired drivers.

To support this capability, vehicles are equipped with **On-Board Units (OBUs)** containing **DSRC radios**, **GNSS receivers**, processors, and data interfaces (e.g., CAN, Ethernet). These OBUs transmit **Basic Safety Messages (BSMs)** broadcast up to 10 times per second — which include essential vehicle data like position, speed, and brake activity. Safety applications use both the host vehicle's internal data and remote vehicle data from BSMs to assess potential collision risks and issue timely driver alerts. Furthermore, V2V information can be fused with data from on-board sensors such as LiDAR, radar, and cameras to improve detection accuracy.

Driven by the growing need to reduce traffic-related fatalities, the **National Highway Traffic Safety Administration (NHTSA)** has been actively working toward V2V standardization, with potential regulations mandating data broadcasts over DSRC. Notable safety applications powered by V2V include **Forward Collision Warning (FCW)**, **Electronic Emergency Brake Light (EEBL)**, **Intersection Movement Assist (IMA)**, **Do Not Pass Warning (DNPW)**, and **Blind Spot Warning (BSW)**. These applications demonstrate V2V's critical role in enabling safer and more cooperative driving environments, laying the foundation for autonomous vehicle development.

To enable this seamless and real-time exchange of data between vehicles, V2V communication relies on structured message formats. One of the most critical of these is the **Basic Safety Message (BSM**), which serves as the core data packet shared among vehicles to convey essential information such as position, speed, and brake status. Understanding the structure and role of the BSM is fundamental to grasping how V2V systems function effectively in real-world traffic environments**.**

4.4.1.1 V2V Basic Safety Message (BSM) Content

The content of a BSM needs to be well defined to ensure that the application designers know the exact set of information that will be available, their unit, and the level of accuracy that each information element will have. SAE J2735 standard **[8]** specifies a message set, its data frames and data elements to support interoperability among DSRC applications. The Abstract Syntax Notation One (ASN.1) representation of a Basic Safety Message as defined in SAE J2735 Standard is shown below.

```
BasicSafetyMessage ::= SEQUENCE {-- Part I, Sent at all times with each
message
    coreDataBSMcoreData,-- Part II Content
    partII
    SEQUENCE (SIZE(1..8)) OF
    PartIIcontent{{ BSMpartIIExtension }} OPTIONAL,
    regional
    SEQUENCE (SIZE(1..4)) OF
    RegionalExtension {{REGION.Reg-BasicSafetyMessage}} OPTIONAL,
    ...
    }
```

Part I data of the BSM (BSM Core Data) is included in every message and transmitted at all time. ASN.1 representation of the Part I data is shown below.

```
BSMcoreData ::= SEQUENCE {
    msgCnt        MsgCount,
    id            TemporaryID,
    secMark       DSecond,
    lat           Latitude,
    long          Longitude,
    elev          Elevation,
    accuracy      PositionalAccuracy,
    transmission  TransmissionState,
    speed         Speed,
    heading       Heading,
```

```
    angle         SteeringWheelAngle,
    accelSet      AccelerationSet4Way,
    brakes        BrakeSystemStatus,
    size          VehicleSize
  }
```

Part II data are optional and are included in the BSM as needed.

-- BSM Part II content support

```
PARTII-EXT-ID-AND-TYPE ::= CLASS {
  &id                   PARTII-Id UNIQUE,
  &Type
     }  WITH SYNTAX {&Type IDENTIFIED BY &id }
  PARTIIContent     { PARTII-EXT-ID-AND-TYPE; Set } ::= SEQUENCE {
  partII-Id         PARTII-EXT-ID-AND-TYPE, &id {Set},
  partII-Value      PARTII-EXT-ID-AND-TYPE, &type {Set}{@partII-Id}
  }
  PARTII-Id ::= INTEGER (0..63)
  vehicleSafetyExtPartII-Id ::= 0 -- VehicleSafetyExtensions
  specialVehicleExtPartII-Id ::= 1 -- SpecialVehicleExtensions
  supplementalVehicleExtPartII-Id ::= 2 --
      -- NOTE: new registered Part II content IDs
  will be denoted here
      -- In a given message there may be multiple extensions present,
      -- but at most one instance of each extension type.
  BSMpartIIExtension PARTII-EXT-ID-AND-TYPE ::= {
      {VehicleSafetyExtensions IDENTIFIED BY vehicleSafetyExt} |
      {SpecialVehicleExtensions IDENTIFIED BY specialVehicleExt} |
      {SupplementalVehicleExt IDENTIFIED BY supplementalVehicleExt},
      ...
  }
```

Detailed information on each of the data elements above can be found in [8]. Few key BSM content requirements are given below.

Table 4.1

Data elements	**Requirement**
Time (DSecond)	Milliseconds within a minute (UTC Standard)—Within ±1msof the actual time

Position (longitude and latitude)	Longitude and latitude within 1.5 m of actual position at HDOP < 5 and 1 sigma absolute error
Position (elevation)	Within 3 m (10 feet) of actual position
Speed	Accuratewithin0.28m/s(1kph)
Heading	For speed >12.5 m/s, accuracy within 2^0 For speed<12.5 m/s, accuracy within 3^0
AccelerationSet4Way (long and lat acceleration)	Longitudinal & Lateral accuracy 0.3 m/s^2 Vertical accuracy 1 m/s
AccelerationSet4Way (Yaw Rate)	Accuracy within $0.5^0/s$
Steering Wheel Angle	Report the direction of steering wheel angle within $5^0$of actual steering wheel angle
Vehicle Size	Vehicle length and width within 0.2 m tolerance
Vehicle Safety Extensions (Path History)	Provides concise representation of vehicles' recent movements with accuracy of min 23 points and required to be transmitted with BSM
Vehicle Safety Extension (Path Prediction)	Perpendicular distance—1 m; radius error—2%; transmission time 4s

To ensure the accuracy and reliability of the data contained within the **Basic Safety Message (BSM),** precise positioning information is essential. This is where **Global Navigation Satellite Systems (GNSS)** come into play. GNSS technologies, such as **GPS**, provide the real-time location data that vehicles need to populate key fields within the BSM—**like position, speed, and heading**. Without accurate GNSS input, the effectiveness of BSMs in supporting safety applications and collision avoidance would be significantly compromised. Therefore, understanding GNSS is crucial to appreciating how V2V communication systems maintain situational awareness and spatial accuracy on the road

4.4.1.2 GNSS Principles

4.4.1.2.1 What Is GPS?

GPS stands for **Global Positioning System**, consisting of a set of Earth-orbiting satellites. These satellites continuously broadcast signals that allow a user on Earth with a suitable device **(a GPS receiver)** to determine their location with accuracy typically within meters, and down to centimeters when locally available correction data is used.

In popular use, the meaning of "GPS" has been extended to refer to any device that displays the user's global location on a map and might also provide turn-by-turn navigation directions. In other parts of the world, this is more commonly referred to as "satellite navigation" or "sat-nav."

Other countries are developing or maintaining similar systems, generically referred to as Global Navigation Satellite Systems (GNSS).

4.4.1.2.2 Basic GNSS Positioning in Cooperative Vehicles

Accurate positioning is essential for the proper functioning of Vehicle-to-Vehicle (V2V) safety applications, which typically require lane-level positioning for both the Host Vehicle (HV) and Remote Vehicles (RVs). A V2X system includes a GNSS receiver to provide position and time data. Vehicles broadcast Basic Safety Messages (BSMs) with motion data, such as speed, acceleration, position, heading, and path history. Using this information, the system calculates the range, heading difference, and relative position between vehicles, helping with lane-level classification of the remote vehicle. Industry consensus is that a vehicle must localize itself within a lane, with a minimum positioning accuracy of 1.5 meters.

When GNSS availability is reduced, such as in urban canyons or tunnels, positioning may depend on other sensors, like IMUs, cameras, or odometers. The reference system is resilient to brief outages (less than 1 second), which accounted for 93% of observed interruptions during testing. However, prolonged outages (2–5 seconds) may disable lane-level applications (e.g., EEBL) while allowing road-level applications (e.g., IMA) to continue. Roadside Equipment (RSE) provides differential corrections to improve vehicle localization at equipped intersections.

The SAE J2945/1[9] standard requires the V2X system to use WAAS corrections to enhance accuracy when available, with position updates occurring 10 times per second. Additionally, state agencies like Michigan's CORS network provide free correction data online.

4.4.1.2.3 Positioning Data Flow in a Connected Vehicle

In this project, a **Ublox NEO-7M** GPS receiver was used to provide real-time geolocation data to the embedded computing unit. Communication between the GPS module and the microcontroller was established through the **UART (Universal Asynchronous Receiver/Transmitter)** interface, which is one of the most common serial links for GPS receivers in embedded systems. Other supported physical interfaces include **I2C, SPI,** and **USB**, but UART was selected for its simplicity and compatibility.

GPS Output Format – NMEA Protocol

The GPS receiver communicates its positioning and timing data using **NMEA messages**, which follow the **NMEA 0183** standard—a combined electrical and data communication specification designed for serial links. Among the many NMEA sentences generated by GPS receivers, the most relevant for intelligent transportation and V2X (Vehicle-to-Everything) systems are:

- **$GPGGA** – Provides 3D location and accuracy information (fix data).
- **$GPRMC** – Delivers essential GPS data such as position, velocity, and time.

“Most useful NMEA messages for V2X/OBE are GPGGA and GPRMC”
(Connected Vehicles: Intelligent Transportation Systems, Radovan Miucic)

Additionally, the **Pulse Per Second (PPS)** signal is used in some systems to synchronize devices (OBEs) across multiple vehicles with high timing accuracy.

```
$GPGGA,171546.0,4228.594269,N,08306.957771,W,1,09,0.7,186.1,M,-34.0,M,,*6A
$GNGNS,171546.0,4228.594269,N,08306.957771,W,AA,15,0.7,186.1,-34.0,,*53
$GPVTG,255.9,T,255.9,M,44.7,N,82.8,K,A*26
$GPRMC,171546.0,A,4228.594269,N,08306.957771,W,44.7,255.9,290315,0.0,E,A*29
$GPGSA,A,2,01,03,04,12,14,22,25,31,32,,,,1.0,0.7,0.7*36
$GNGSA,A,2,01,03,04,12,14,22,25,31,32,,,,1.0,0.7,0.7*28
$GNGSA,A,2,69,70,71,73,79,80,,,,,,,1.0,0.7,0.7*20
$GLGSV,3,1,09,70,64,326,23,73,17,219,27,80,69,229,20,79,44,032,22*63
$GLGSV,3,2,09,69,52,139,28,87,02,350,16,71,13,322,23,68,02,140,18*6E
$GLGSV,3,3,09,88,,,*6C
```

Figure 4.3 A sample NMEA file

Customization with u-center

To optimize performance and reduce unnecessary data parsing on the microcontroller, the GPS module was configured using the **u-center** tool from u-blox. Through this application, all NMEA sentences were disabled except for $GPRMC, the one required by the system. This customization ensures a lightweight, streamlined communication process suitable for embedded applications with limited resources.

GPRMC Sentence Breakdown

The **GPRMC (Recommended Minimum C)** sentence contains the minimum data required for GPS navigation. Below is an example of a GPRMC message received from the NEO-7M module:

```
$GPRMC,171546.0,A,4228.594269,N,08306.957771,W,44.7,255.9,290315,0.0, E,A*29
```

The components of this sentence are:

Table 2 GPRMC message

$GPRMC	-$ indicates start of the sentence. – GP indicates that fix came from a GPS device (other possible values are GA-Galileo, GL-Glonass, GN-combined GNSS systems) – RMC-Recommended Minimum sentence C
171546.0	Fix taken at 17:54:46 UTC
A	Status A =active or V = void
4228.594269, N	Latitude 42^0 28.594269́ N
08306.957771, W	Longitude 83^0 6.957771́ W
44.7	Speed over the ground in knots
255.9	Track angle in degrees true (course made good, true)
290,315	Date—29thofMarch2015
0.0, E	Magnetic variation

A	Kind of fix the receiver currently has. The value can be A = autonomous, D=differential, E = estimated, N = not valid, S = simulator
*29	The checksum data always begins with*

In this project, only the following data fields were extracted and used:

- **UTC time** – For time synchronization and timestamping
- **Latitude and Longitude** – For position tracking
- **Speed** – For vehicle motion monitoring
- **Track angle** – For determining heading direction

This data was then processed and integrated into the vehicle's decision-making and warning systems (e.g., FCW, DNPW, etc.).

After discussing the GPS technology and its ability to provide accurate positioning data, it's important to acknowledge that while GPS can offer reliable location information, **the data it provides can sometimes be noisy or less accurate due to factors such as signal interference or environmental conditions.** To address these **challenges and improve** the overall precision of the positioning system, **we can integrate a Kalman Filter**. The Kalman Filter helps to fuse GPS data with other sensor inputs, effectively reducing noise and providing a smoother and more accurate estimate of the position, velocity, and other key parameters. By applying this filtering technique, we can significantly enhance the performance of **GPS-based systems**, especially in dynamic environments where real-time accuracy is critical.

4.4.1.3 Kalman Filter

4.4.1.3.1 What Is Kalman Filter?

The **Kalman Filter** **[10,11]** is a mathematical algorithm that estimates a parameter (or state), or typically a state vector, based on relevant measurement data. It combines multiple measurements over time and from various sensors. The filter works by weighing these measurements according to their known statistics and applying a model that tracks the change

of the estimate (state) over time between those measurements. In simple terms, it gives more weight to the data from sources we are more confident about.

The key concept of the **Kalman Filter** is the **Kalman gain**, usually denoted as **K**, which determines how much weight each data source is given during the combination process. The filter assumes that all data sources, such as measurement values and model equations, contain errors that can be represented as Gaussian distributions, each characterized by mean and variance. A smaller variance indicates that the data is more concentrated around the true value and is therefore more trustworthy. This is visualized as a narrower peak in a probability distribution graph. The **Kalman Filter** intuitively gives more weight to the data with smaller variance, and the combined estimate tends toward the source with the least variance, as shown by the bold peak in the graph, which is closer to the narrower one.

In **V2V (Vehicle-to-Vehicle)** applications, the **Kalman Filter** plays a crucial role in improving the accuracy of vehicle positioning, especially in environments where GNSS signals may be degraded or interrupted. For example, when GNSS signals are weak, the filter integrates data from other sensors like IMU (Inertial Measurement Unit), cameras, or odometers, to estimate the vehicle's position. The filter uses Kalman gain to give more weight to data from sensors with smaller variance (i.e., more reliable data), and it combines this information to provide a more accurate estimate of the vehicle's location, even during GNSS outages.

The filter's ability to handle noisy or uncertain data makes it ideal for applications requiring precise tracking of moving vehicles. By continuously updating the state estimate as new data arrives, the **Kalman Filter** enables the system to predict future positions, thereby enhancing the reliability of safety applications like collision warning and lane-keeping. It ensures that vehicle positioning remains accurate despite challenging conditions like urban canyons, tunnels, or other areas with obstructed GNSS signals.

$$x_{combined} = \left(\frac{{\sigma_{IMU}}^2}{{\sigma_{GPS}}^2 + {\sigma_{IMU}}^2}\right) x_{GPS} + \left(\frac{{\sigma_{GPS}}^2}{{\sigma_{GPS}}^2 + {\sigma_{IMU}}^2}\right) x_{IMU} \quad (1)$$

In real systems where non-linearity exists:

- Extended Kalman Filter (EKF): Approximates non-linear systems via linearization; standard for GNSS-Inertial Navigation.
- Unscented Kalman Filter (UKF): Handles non-linearities more accurately without linear approximation.

4.4.1.3.2 Phases of Kalman filter

- **prediction (or time update)**.
- **measurement update**.

In the prediction step, the estimate from the previous time step is brought to current time using equations that model the state change over time.

Prediction steps are described with the following. State vector x has multiple (n) dimensions (positions, velocities, etc.) that change dynamically. The state is modeled using the linear stochastic difference equation(2)

$$x_k = \boldsymbol{A}x_{k-1} + \boldsymbol{B}u_k + w_{k-1} \tag{2}$$

where x_k is the $n \times 1$ state vector of the k-th state, $\boldsymbol{A}$ is the n × n matrix that relates two consecutive states in absence of the control input and noise, u is the optional 1 × 1control input (assumed to be zero here); $\boldsymbol{B}$ is the $n \times l$ matrix that relates the control input to the state (also zero here); w is the $n \times 1$ process white noise with normal probability distribution, mean of **zero** and covariance $\boldsymbol{Q}$.

The uncertainty in the model is represented using the estimate error covariance matrix, $\boldsymbol{P}$ **.** $\boldsymbol{P}$ has two versions at each step k : one just before the measurement is taken into account (a priori), $\boldsymbol{P}_k^-$, and one after the measurement is included (a posteriori), $\boldsymbol{P}_k$. The estimate error covariance is updated using (3).

$$\boldsymbol{P}_k = \boldsymbol{A}\boldsymbol{P}_k^- \boldsymbol{A}^T + \boldsymbol{Q}_k \tag{3}$$

In the measurement step, the measurement values $\boldsymbol{Z}_K$ are compared (subtracted by) the measurement values $\boldsymbol{H}_k \widehat{x_k}$ that would be predicted using the measurement model equations, and then the result is multiplied by the Kalman gain $\boldsymbol{K}_K$:

$$\hat{x}_k = \hat{x}_k^- + \boldsymbol{K}_k (\boldsymbol{Z}_k - \boldsymbol{H}_k \hat{x}_k^-) \quad (4)$$

where $\boldsymbol{H}$ is the $m \times n$ matrix that relates the state to the measurement.

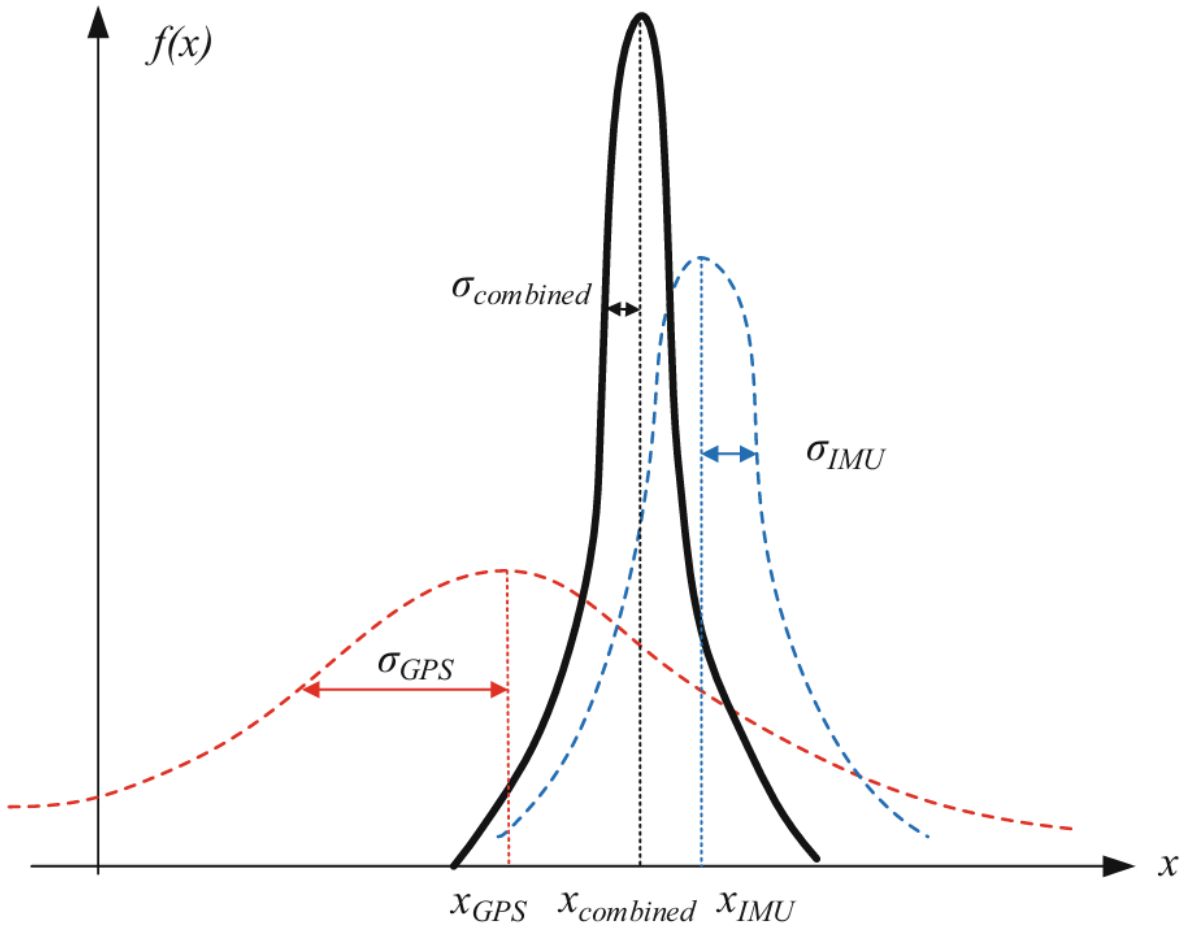


Figure 4.4 Combining measurements based their statistics

The measurement model relates the measured quantities to those that are being estimated using linear equations that in matrix form appear as

$$z_k = \boldsymbol{H}_k x_k + v_k \quad (5)$$

where z is the $m \times 1$ measurement vector, $\boldsymbol{H}$ is the $m \times n$ matrix that relates the state to the measurement, and v is the $m \times 1$ measurement noise that is white with normal probability distribution, mean of 0, and covariance $\boldsymbol{R}$.

The uncertainty in the measurement model is captured using the measurement error covariance matrix, $\boldsymbol{R}$.

The Kalman weighs the contribution of the new sensor data based on the relationship between confidence expressed in the state estimate via $\boldsymbol{P}$ and the confidence expressed in measurement via $\boldsymbol{R}$:

$$\boldsymbol{K}_k = (\boldsymbol{P}_k^- \boldsymbol{H}_k^T)(\boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T \boldsymbol{H}_k^T)^- \quad (6)$$

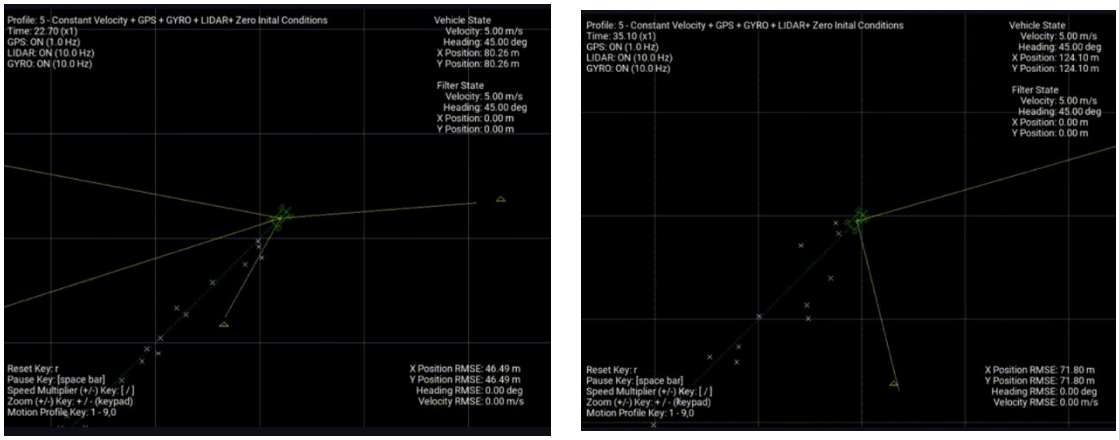


Figure 4. 5Advanced Kalman Filtering and Sensor Fusion Simulation

Once the **Kalman Filter has been applied** to refine and stabilize the raw sensor data particularly the position and motion estimates we obtain more accurate and reliable inputs for the next stage of processing: **target classification**. Since classification decisions often rely heavily on precise information such as object location, speed, and trajectory, the improved data from the Kalman Filter plays a crucial role in enhancing the accuracy and robustness of the classification process. **This smooth transition from data** filtering to intelligent interpretation ensures that the system can effectively distinguish between different types of targets and respond appropriately in real time.

4.4.1.4 Target Classification (TC)

Target Classification and Its Role in Vehicular Safety Applications, The Target Classification (TC) module plays a pivotal role in determining the position and orientation of a Remote Vehicle (RV) relative to the Host Vehicle (HV). This classification enables a wide array of safety applications by identifying the spatial and directional relationships between vehicles on the road.

To support accurate classification, the TC module leverages operations such as Path Prediction Radius Calculation and Path Prediction Confidence Estimation. Additionally, Path

History is utilized to enhance prediction accuracy, as analyzing past maneuvers can yield reliable estimates for future movements with an associated confidence level.

The core function of the TC system is to categorize the relative zone and direction of the RV. For example, if the RV is in the “Ahead Left” zone and is moving in reverse, only specific applications—such as Do Not Pass Warning (DNPW)— are relevant and triggered. This selective activation ensures that unnecessary warnings are minimized, and only contextually appropriate applications are considered.

Once results from all potential safety applications are gathered, the system prioritizes them and provides the highest priority warning to the driver via the User Interface (UI). Priority is typically determined by the time-to-collision metric; for instance, a Forward Collision Warning (FCW) would supersede an Emergency Electronic Brake Light (EEBL) alert due to its more immediate risk.

The TC module offers a 360-degree, lane-level classification of the RV’s location and direction relative to the HV. The outputs include:(Relative Position, Relative Direction of Travel, Lateral Offset, Longitudinal Offset, Delta Heading), These outputs are derived from Basic Safety Message (BSM) fields such as:(Latitude, Longitude, Elevation, Speed, Heading, Yaw Rate, Path History, Path Prediction Objects), Figure 4.6 visualizes the 14 relative zones around the HV, within which the RV can be classified. Figure 4.7 outlines the four possible travel directions an RV can take relative to the HV.

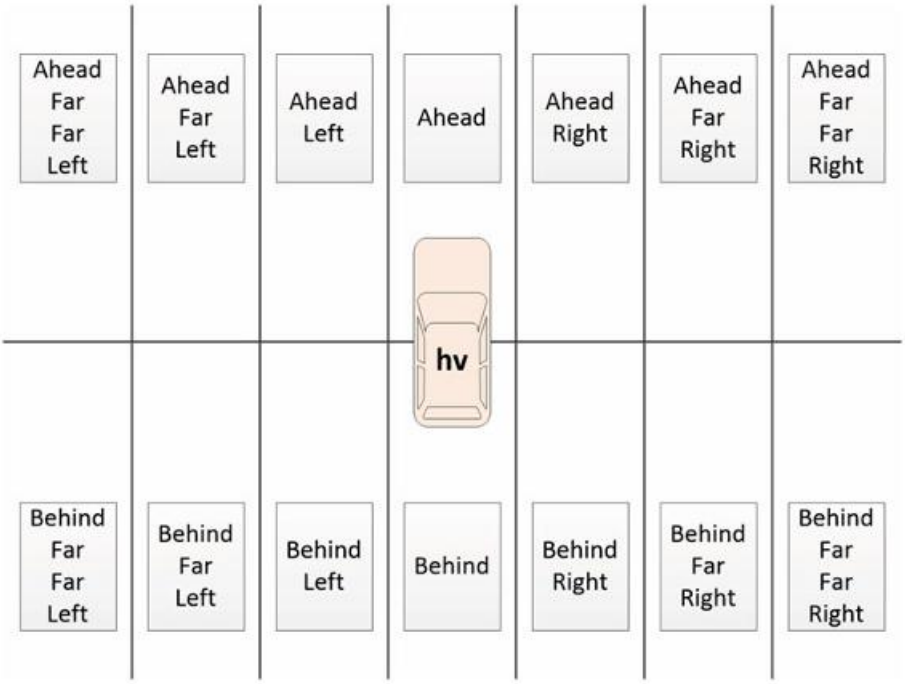


Figure 4.6 RV position (Zone) relative to the HV

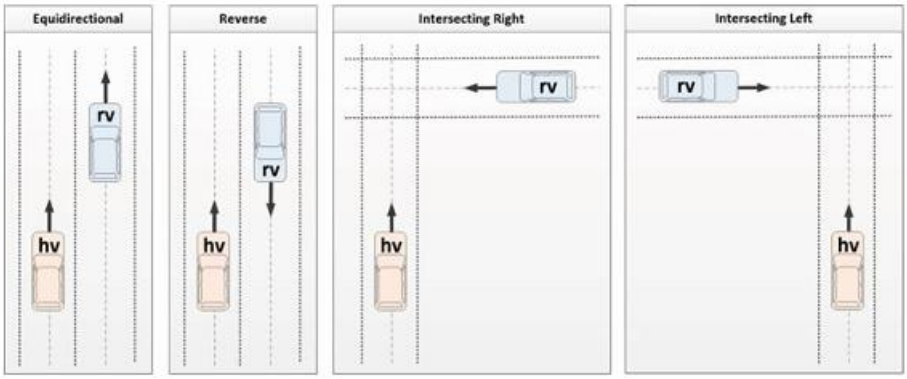


Figure 4.7 RV direction of travel relative to the HV

Using the data generated by the TC module, the system identifies the most relevant safety applications for the detected zone and direction. The Human-Machine Interface (HMI) is then used to alert the driver with the most critical warning derived from the prioritized application outputs.

Figure 4.8 illustrates a mapping between relative zones/directions and corresponding safety applications, showing how situational awareness directly feeds into adaptive safety responses.

4.4.1.4.1 Lateral and Longitudinal Offset Computation

Figure 9 shows the terminologies involved in Target Classification. It shows the terminologies used and relationship between them for the computation of lateral offset and longitudinal offset between HV and RV.

Some of the modules required for the development of Target Classifications are given below. Implementation of these modules are not covered in this chapter.

$$\text{convertLatLongToXY}(\text{refLat}, \text{refLong}, \text{refHead}, \text{latitude}, \text{longitude}) \quad (7)$$

– Converts the GPS latitude and longitude to X and Y in meters relative to the reference latitude (refLat), reference longitude (refLat), and reference heading (refHead).

$$\text{convertXYtoLatLong}(\text{refLat}, \text{refLong}, \text{refHead}, \text{relX}, \text{relY}) \quad (8)$$

– Converts X (relX) and Y (relY) in meters relative to the reference latitude (refLat), reference longitude (refLong), and reference heading (refHead) to an absolute latitude and longitude.

$$calcGpsDist(refLat, refLong, latitude, longitude) \quad (9)$$

– Calculates the absolute distance in meters between two GPS positions (refLat, refLong) and (latitude, longitude). These components can be helpful for various computations during the development of Target Classification and Safety Applications. Each term used in Figure 9 are explained below.

Host Vehicle Path Prediction provides the Path Prediction Radius for predicted future path of the HV. Path Prediction Centre would then be at the Cartesian Coordinates (0, hvppr). The absolute GPS coordinate of the path prediction center can then be obtained using convertXYtoLatLong module as shown in Eq. (10)

$$[hvppc\_lat, hvppc\_lon] = convertXYtoLatLong(hv\_lat, hv\_lon, hv\_head, 0, hvppr) \quad (10)$$

where, hv_head = HV heading in degrees Lateral offset between the Host Vehicle Predicted Path and the RV (lat_offset) is simply the difference between hvppc_rv_dist and hvppr, where the hvppc_rv_dist can be calculated as shown in Eq. (11)

$$hvppc\_rv\_dist = calcGpsDist(hvppc\_lat, hvppc\_lon, rv\_lat, rv\_lon) \quad (11)$$

Longitudinal offset is the Arc Length between HV and RV along the Host Vehicle Predicted Path with the center being hvppc and can be calculated as shown in Eq. (12).

$$\text{lon\_offset} = (\text{hvppr}*) \times \text{Angle}(\text{hv\_hvppc\_rv}) \quad (12)$$

In practice longitudinal distance can be approximated by adding all path history points from the vehicle in front that are ahead of the vehicle calculating longitudinal distance (e.g. host vehicle from Figure (9).

RV position relative to the HV in Cartesian coordinate system can be given as (rvRelX, rvRelY) and can be calculated as shown in Eq. (13).

$$[\text{rvRelX}, \text{rvRelY}] = \text{convertLatLongToXY}(\text{hv\_lat}, \text{hv\_lon}, \text{hv\_head}, \text{rv\_lat}, \text{rv\_lon}) \quad (13)$$

where, hv_head = HV heading in degrees

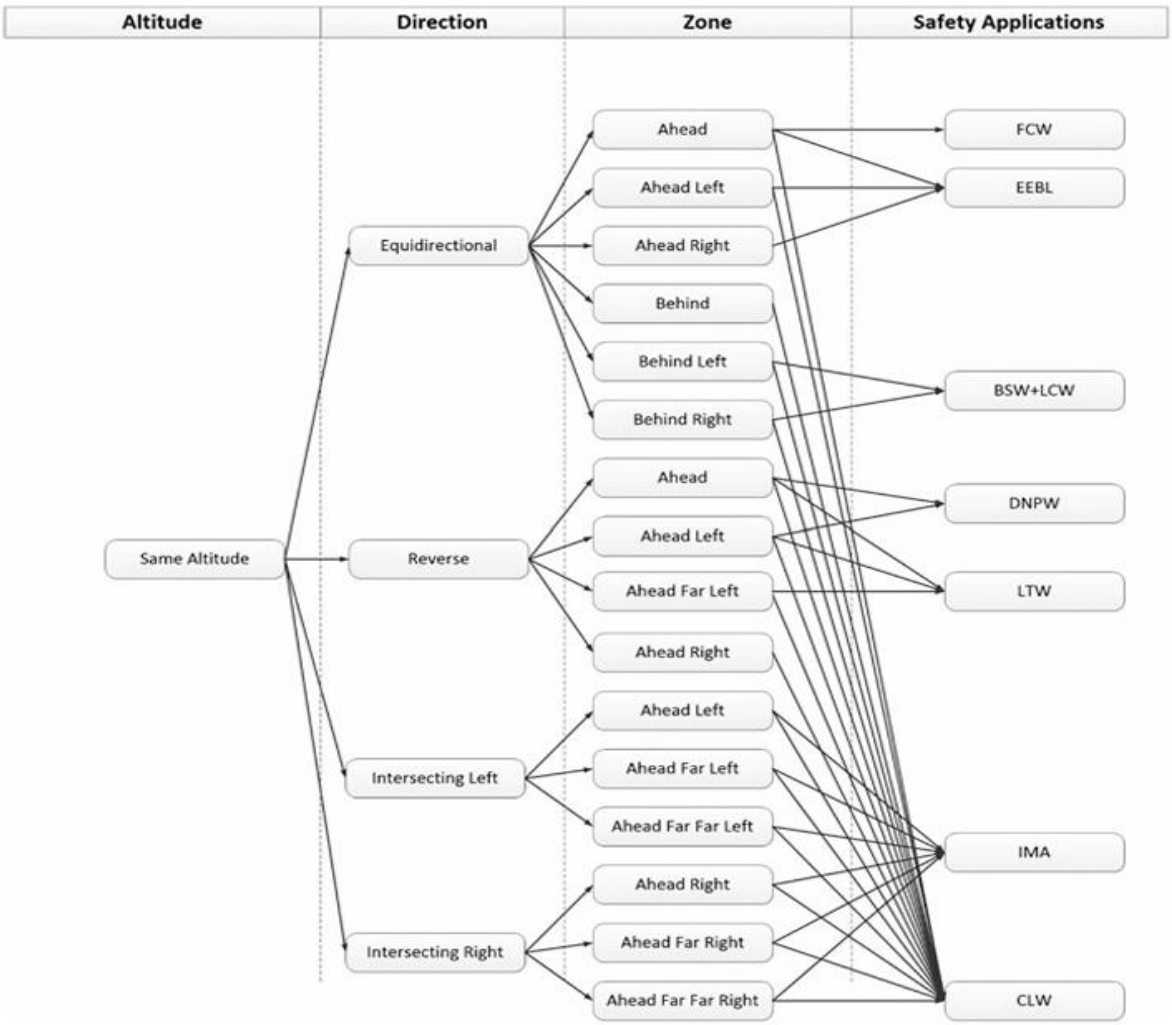


Figure 4.8 Mapping of safety applications to the output of TC module

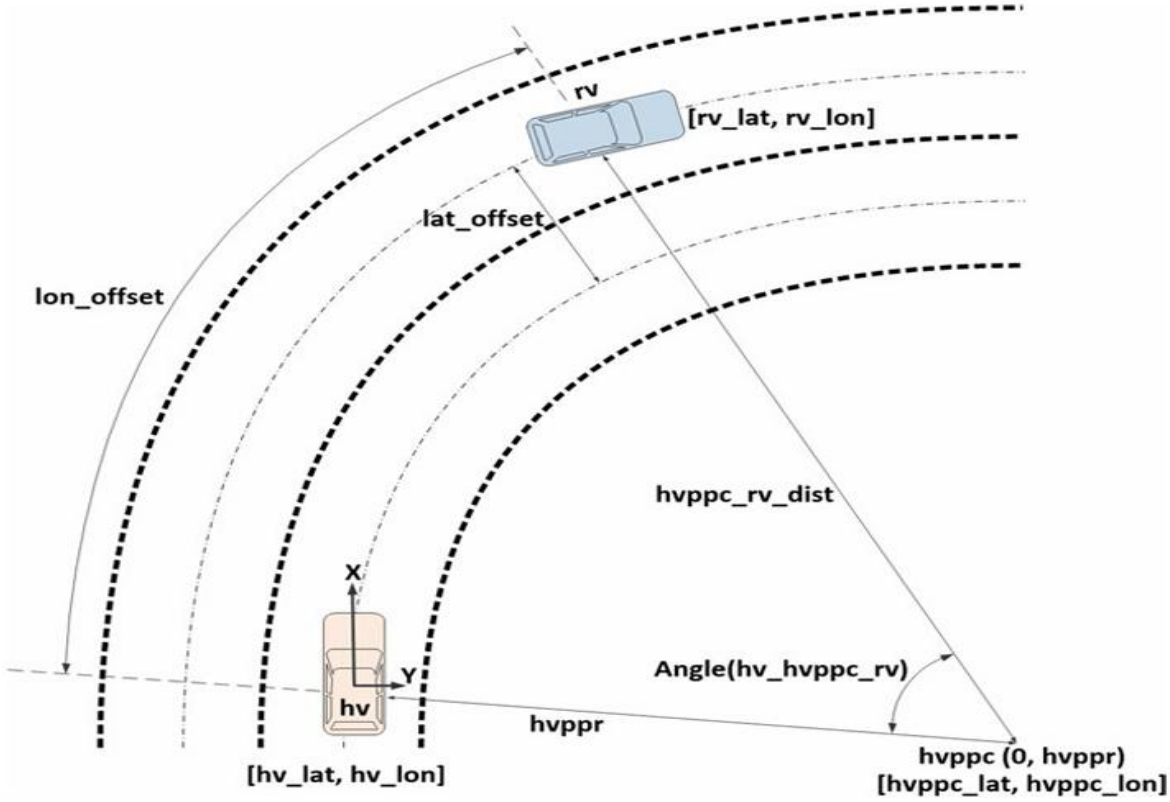


Figure 4.9 RV direction of travel relative to the HV

Table 4.3 RV zones relative to the HV

rvRelX ≥0?	hvppr ≥0?	lat_offset	Zone
Yes	Yes	lat_offset ≤ -2.5× lanewidth	Ahead Far Far Left
Yes	Yes	-2.5 ×lanewidth>lat_offset≤ −1.5 ×lanewidth	Ahead Far Left
Yes	Yes	−1.5 ×lanewidth>lat_offset≤ -0.5 ×lanewidth	Ahead Left
Yes	Yes	-0.5 ×lanewidth ≥ lat_offset≤ -0.5 ×lanewidth	Ahead
Yes	Yes	0.5 ×lanewidth ≥lat_offset< 1.5 ×lanewidth	Ahead Right
Yes	Yes	1.5 ×lanewidth ≥lat_offset < -2.5 ×lanewidth	Ahead Far Right
Yes	Yes	Lat_offset ≥ 2.5 ×lanewidth	Ahead Far Far Right
Yes	No	lat_offset ≤ -2.5 × lanewidth	Ahead Far Far Right
Yes	No	-2.5 ×lanewidth>lat_offset≤ −1.5 ×lanewidth	Ahead Far Right
Yes	No	−1.5 ×lanewidth>lat_offset≤ -0.5 ×lanewidth	Ahead Right

Yes	**No**	**-0.5 ×lanewidth ≥ lat_offset≤ -0.5 ×lanewidth**	**Ahead**
Yes	**No**	**0.5×lanewidth ≥lat_offset< 1.5 ×lanewidth**	**Ahead Left**
Yes	**No**	**1.5 ×lanewidth ≥lat_offset < 2.5 ×lanewidth**	**Ahead Far Left**
Yes	**No**	**Lat_offset≥ 2.5 ×lanewidth**	**Ahead Far Far Left**
No	**Yes**	**lat_offset≤ -2.5 × lanewidth**	**Behind Far Far Left**
No	**Yes**	**-2.5 ×lanewidth>lat_offset≤ −1.5 ×lanewidth**	**Behind Far Left**
No	**Yes**	**−1.5 ×lanewidth>lat_offset≤ -0.5 ×lanewidth**	**Behind Left**
No	**Yes**	**-0.5 ×lanewidth ≥ lat_offset≤ -0.5 ×lanewidth**	**Behind**
No	**Yes**	**0.5×lanewidth ≥lat_offset< 1.5 ×lanewidth**	**Behind Right**
No	**Yes**	**1.5×lanewidth ≥lat_offset < 2.5 ×lanewidth**	**Behind Far Right**
No	**Yes**	**Lat_offset≥ 2.5 ×lanewidth**	**Behind Far Far Right**
No	**No**	**lat_offset≤ -2.5 × lanewidth**	**Behind Far Far Right**
No	**No**	**-2.5 ×lanewidth>lat_offset≤ -1.5 ×lanewidth**	**Behind Far Right**
No	**No**	**-1.5 ×lanewidth>lat_offset≤ -0.5 ×lanewidth**	**Behind Right**
No	**No**	**-0.5 ×lanewidth ≥ lat_offset≤ -0.5 ×lanewidth**	**Behind**
No	**No**	**0.5 ×lanewidth ≥lat_offset< 1.5 ×lanewidth**	**Behind Left**
No	**No**	**1.5 ×lanewidth ≥lat_offset < 2.5 ×lanewidth**	**Behind Far Left**
No	**No**	**Lat_offset≥ 2.5 ×lanewidth**	**Behind Far Far Left**

4.4.1.4.2 Improvement Target Classification

Lateral offset can be stabilized by averaging the offsets between the Host Vehicle's predicted path and the Remote Vehicle's concise path history, ensuring accurate zone classification when both vehicles move steadily on the same road.

Pseudocode to compute the average lateral offset is given below:

```
NUM_OF_RVPH_POINTS_AHEAD_OF_HV = 0
 AVG_LAT_OFFSET = 0
 FOR (n = 0 to NUM_OF_PH_POINTS_AVAILABLE-1)
 IF RV_PH(n) AHEAD OF HV THEN
 COMPUTE LAT_OFFSET(n)
 AVG_LAT_OFFSET = AVG_LAT_OFFSET + LAT_OFFSET(n)
 NUM_OF_PH_POINTS_AHEAD_OF_HV = NUM_OF_PH_POINTS_AHEAD_OF_HV + 1
 ELSE
```

```
BREAK LOOP
END IF
END FOR
AVG_LAT_OFFSET = AVG_LAT_OFFSET/NUM_OF_PH_POINTS_AHEAD_OF_HV
```

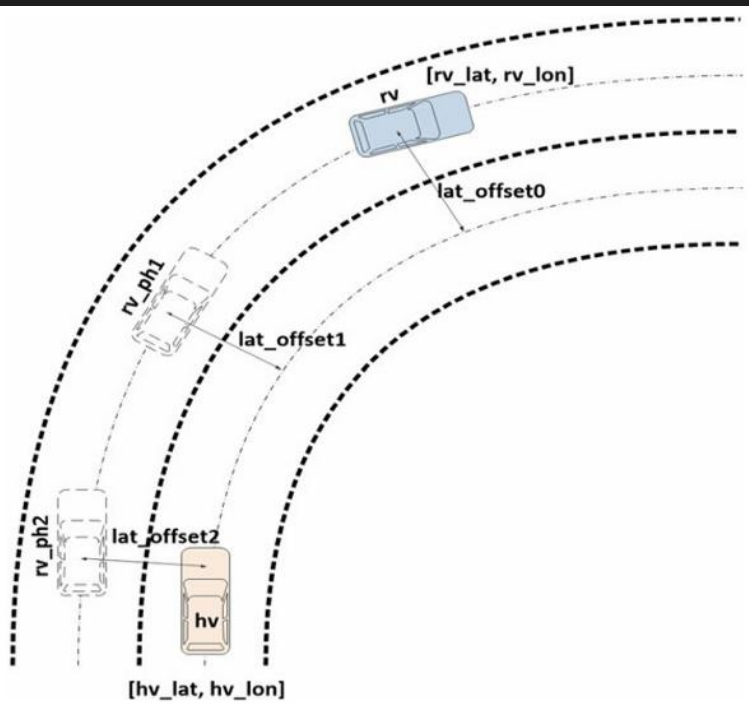


Figure 4.10 RV direction of travel relative to the HV

LAT_OFFSET(n) is the lateral offset between the HV Predicted Path and RV Path History Point n.

The average lateral offset could be used to improve the target classification if any of the following is true :

1 RVis driving steadily on the same route as HV.
2 HVis not driving steadily (Path Prediction Confidence lower than a threshold).

Figure 11 shows a scenario where HV and RV are driving on two different routes. Although the predicted delta heading will be close to zero, it can be determined that they are not driving on the same route by looking at the RV Path History Points as shown below.

```
Diff1 = lat_offset0- lat_offset1
Diff2 = lat_offset0-lat_offset2
IF {|Diff1| < Threshold AND |Diff2| < Threshold } THEN
RV is driving steadily on the same route as the HV
END IF
```

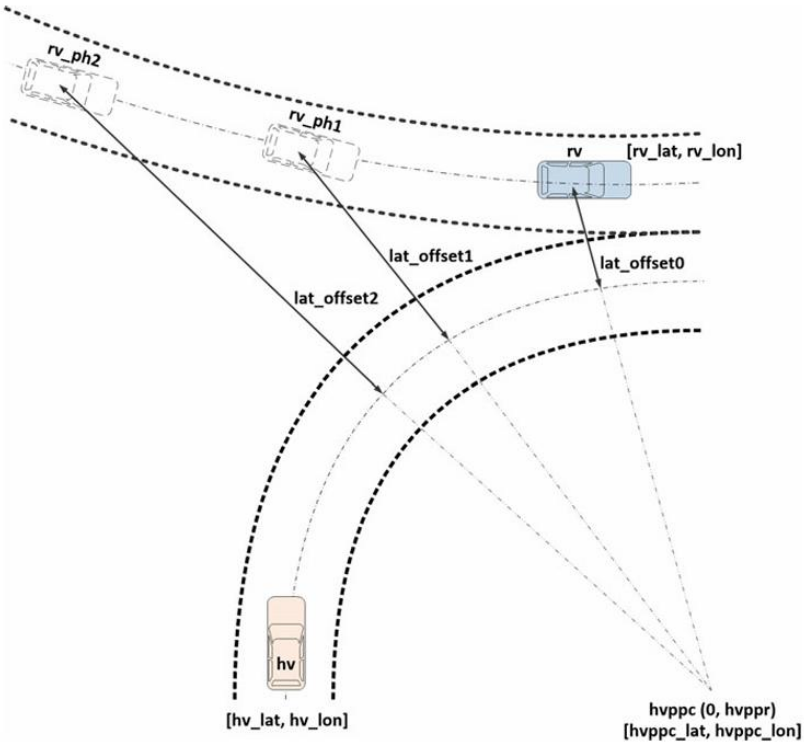


Figure 4.11 Lateral offsets for HV and RV driving on different routes

If the Host Vehicle's path prediction confidence is below a calibrated threshold—indicating unstable driving such as lane changes—lateral offset values may become unreliable. Averaging these offsets can enhance the reliability of RV zone classification. Additionally, 2-D position extrapolation can be applied to estimate a vehicle's position using its last known coordinates, heading, and speed. This method helps classify lateral and longitudinal offsets when timely updates are unavailable.

Following the **successful classification of targets** based on their **position, speed, and behavior**, this information becomes essential input for **V2V (Vehicle-to-Vehicle) safety applications**. Accurate target classification allows the system to identify potential hazards, such as nearby vehicles, pedestrians, or obstacles, and determine their level of threat. This enables V2V safety applications to make informed decisions, such as issuing warnings to the driver or initiating automated responses like braking or lane changes. By relying on precise

classification results, **V2V systems can enhance road safety through real-time communication and coordinated actions between vehicles.**

4.4.1.5 V2VSafety Applications

Vehicle position and dynamics information like latitude, longitude, elevation, heading, speed, yaw rate etc. can be used to predict the future path of vehicles and estimate the possibility of collision in near future. The safety applications considered in this section address rear-end, opposite direction, junction crossing, and lane change crash scenarios. These scenarios can be covered by the safety applications listed in Table 4 and are described below.

Table 4.4 Safety applications associated with pre-crash scenarios

Pre-crash scenarios	**Pre-crash group**	**Associated safety application**
Lead vehicle stopped	Rear-end	Forward collision warning
Lead vehicle moving	Rear-end	Forward collision warning
Lead vehicle decelerating	Rear-end	Forward collision warning/ emergency electronic brake light
Straight crossing path without traffic light	Junction crossing	Intersection movement assist
Left-turn across path/opposite direction	Left turn at crossing	Left turn assist
Opposite direction/no maneuver	Opposite direction	Do not pass warning
Opposite direction/maneuver	Opposite direction	Do not pass warning
Change lane/same direction	Lane change	Blind spot warning
Turning/same direction	Lane change	Blind spot warning
Drifting/same direction	Lane change	Blind spot warning

4.4.1.5.1 Forward Collision Warning (FCW)

FCW warns the driver of an impending collision between the HV front-end, and a RV rear-end. This collision is possible only if both the HV and RV are driving in the same direction

and are on the same lane (RV could be stopped, decelerating, or moving at a speed slower than the HV).

The output of Target Classification (TC) Module is used to determine if RV is in the same lane as the HV.TC also provides the longitudinal offset which is used along with vehicle dynamics information to determine if there is a possibility of forward collision. The simplest way to predict forward-collision is to compute the time-to-collision (TTC) and compare that with a calibrated threshold value.

Figure. (4.12) a show the relevant target classification zone for FCW, and Figure. (4.12) b shows a possible scenario for FCW.

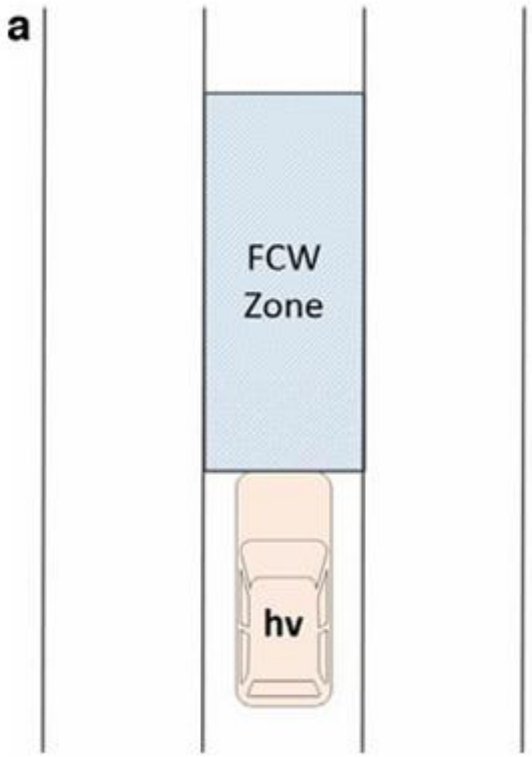


Figure 4.12.a FCW target classification zones

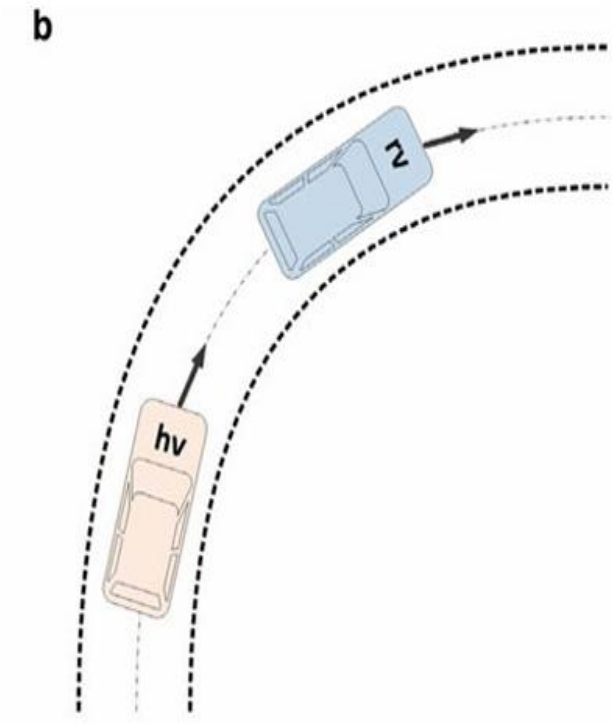


Figure 4.12.b An example scenario for FCW

A simple pseudo-code for the implementation of FCW is given below:

```
 FCW_WARNING = FALSE

IF ((RV_ZONE is AHEAD) AND (RV_DIRECTION is EQUIDIRECTIONAL)) THEN
IF (HV_SPEED > RV_SPEED) THEN
TTC = LONGITUDINAL_OFFSET/ (HV_SPEED- RV_SPEED)
IF (TTC < K_TTC_THRES) THEN
```

```
FCW_WARNING = TRUE
END IF
END IF
END IF
```

Use Case:

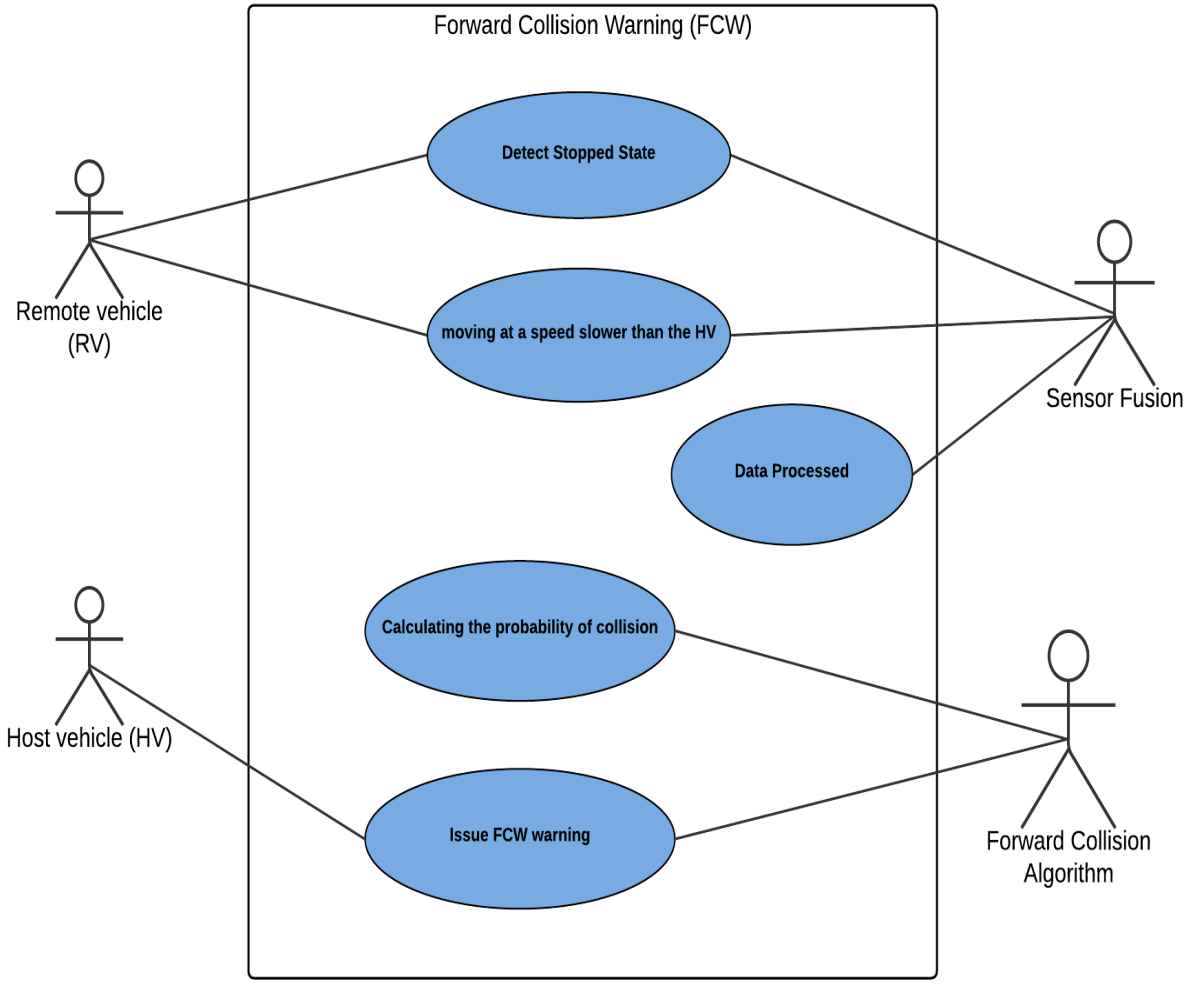


Figure 4.13 Forward Collision Warning Use Case

Flow chart:

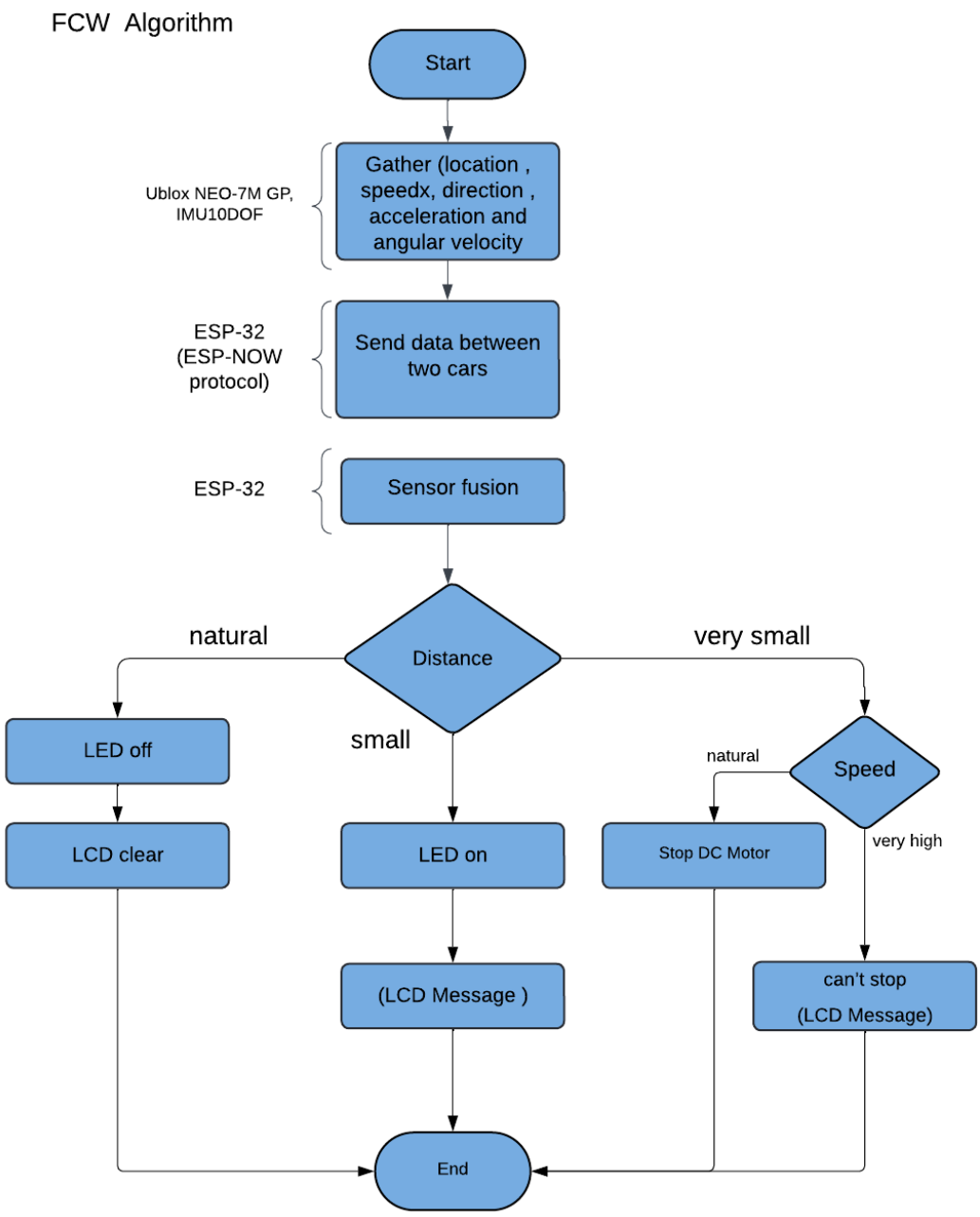


Figure 4.14 Forward Collision Warning Flowchart

4.4.1.5.2 Electronic Emergency Brake Light (EEBL)

EEBL addresses the scenario where a Remote Vehicle is ahead and in the same lane as Host Vehicle. If there are other vehicles between HV and RV considered, and the RV brakes

hard, a potential crash is imminent if no brake light is visible on the vehicle in front of HV. EEBL can issue a warning to the driver of HV in such scenarios where the RV is not directly visible to the HV and thus can help avoid the crash. Once we establish that the RV is ahead in the same lane as the HV and it is driving in the same direction, RV acceleration information from the BSM can be used to determine if the RV has braked hard.EEBL warning is issued if the RV acceleration value is less than a calibrated deceleration threshold value.HV can either find this out from the acceleration value transmitted as a part of the DF_BSMcoreDataData Frame or from the eventHardBraking event flag transmitted as a part of DF_VehicleSafetyExtensionsData Frame.

Figure 4.15a shows the relevant target classification zone for EEBL and Figure 4.15 b shows a possible scenario for EEBL.

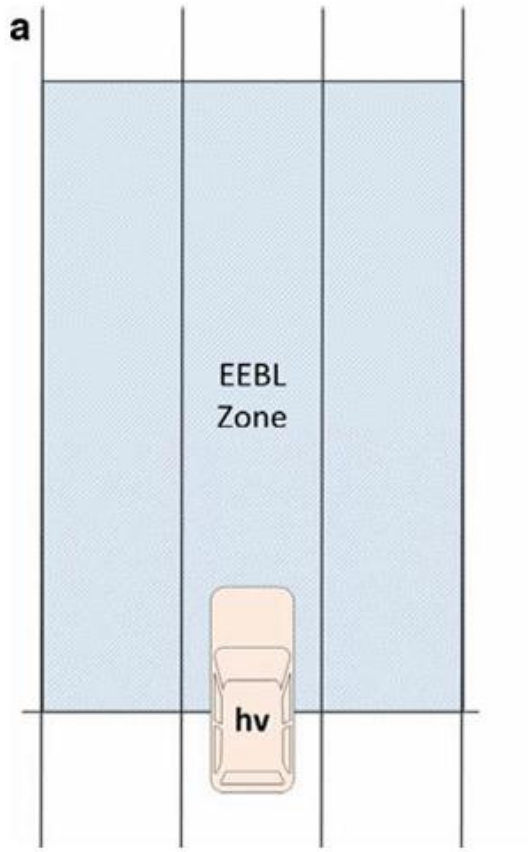


Figure 4.15a EEBL target classification zones

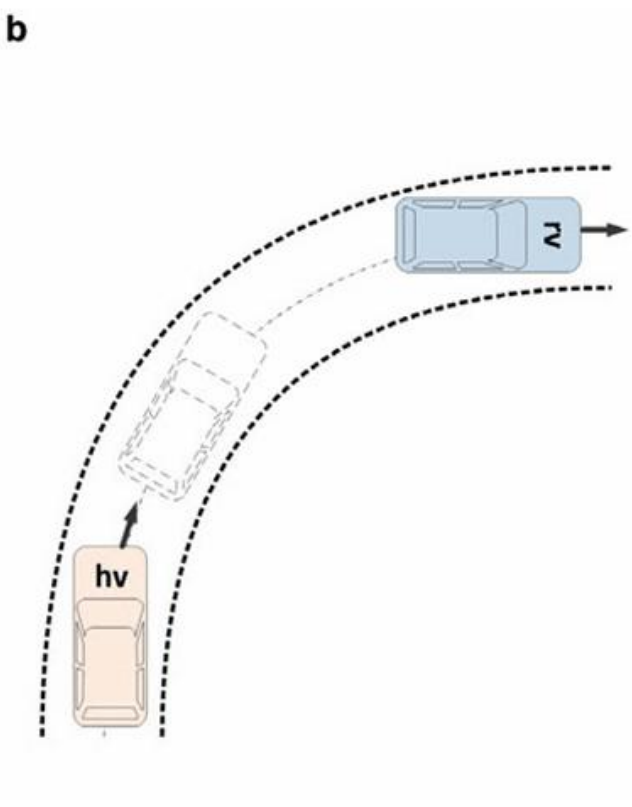


Figure 4.15 b an example scenario for EEBL

SAE J2735 [8] suggests that the minimum acceleration threshold of the RV to trigger EEBL warning be −0.4 g (−3.92 m/s2).

A simple pseudocode for the implementation of EEBL is given below:

```
EEBL_WARNING = FALSE
 IF ((RV_ZONE is AHEAD) AND (RV_DIRECTION is EQUIDIRECTIONAL)) THEN
 IF (LONGITUDINAL_OFFSET < K_MAX_EEBL_ZONE_LEN) THEN
 IF ((HV_SPEED > K_HV_MIN_SPD_THRES) AND (RV_ACCEL < K_EEBL_
 ACCEL_THRES) THEN
 EEBL_WARNING = TRUE
 END IF
 END IF
 END IF
```

Use Case:

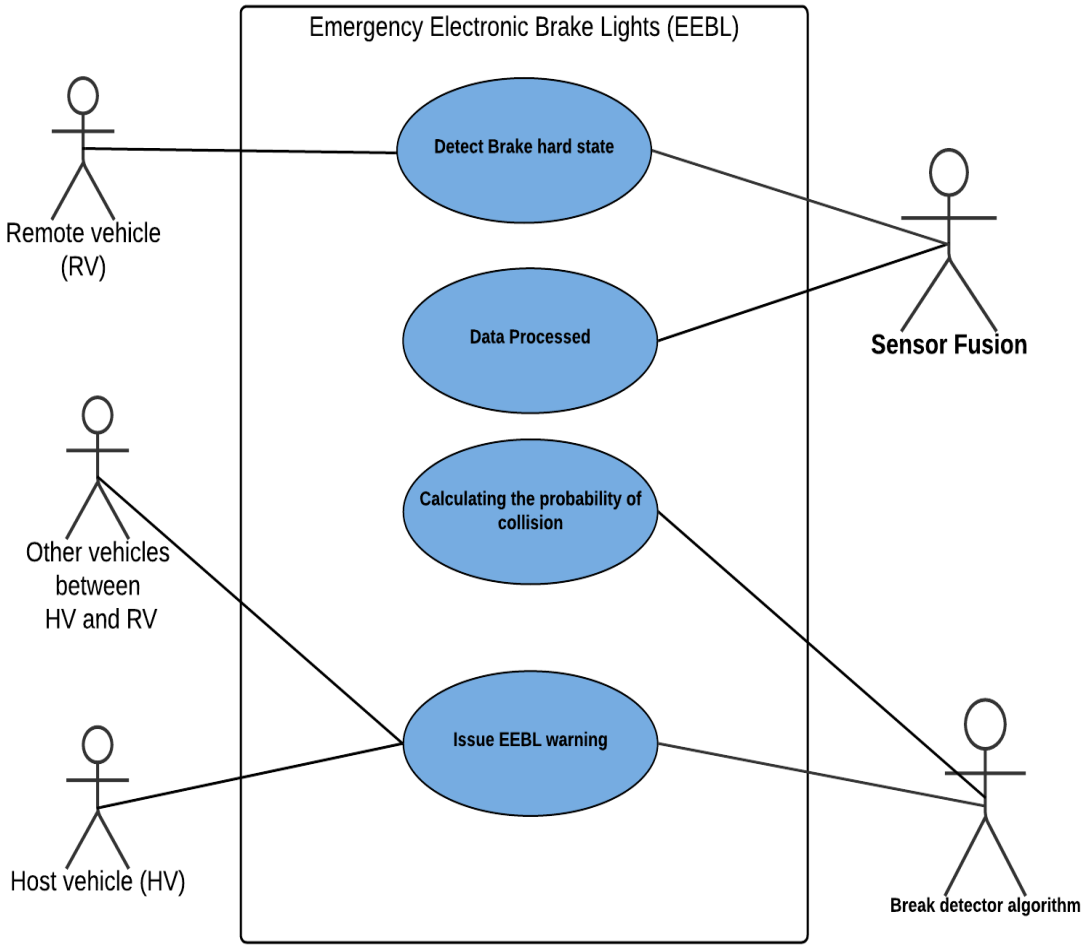


Figure 4.16 Electronic Emergency Brake Light Use Case

Flow chart:

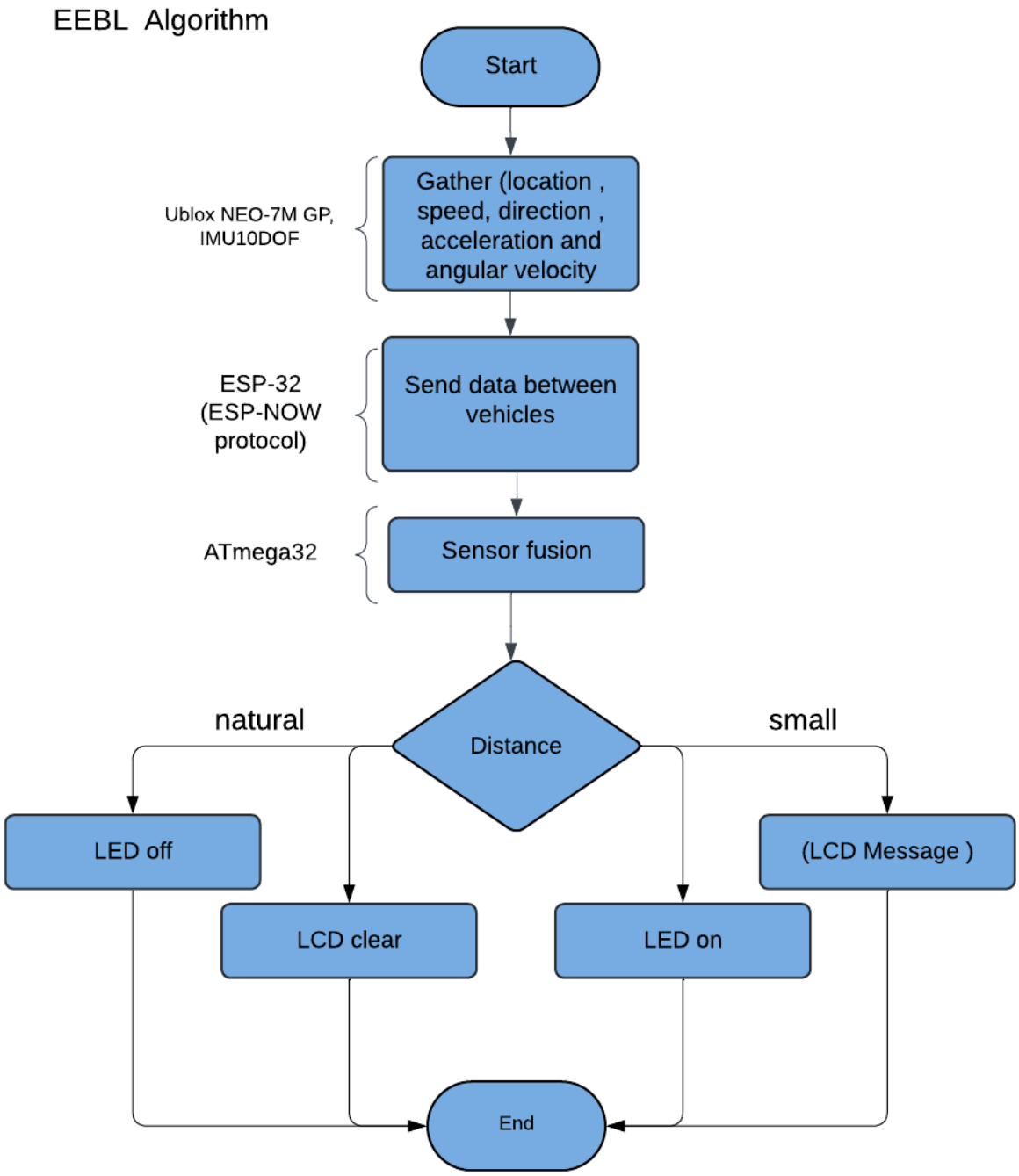


Figure 4.17 Electronic Emergency Brake Light Flowchart

4.4.1.5.3 Blind-Spot Warning (BSW)

BSW safety application either provides an advisory alert or warns the driver of HV if another vehicle occupies the adjacent lane in Host Vehicle's Blind-Spot. The application arbitrates one versus the other depending on the driver's intent to change the lane to the one occupied by another vehicle. A driver shows the intention to change the lane by activating a turn signal with information which can be pulled into the system from the Vehicle CAN Bus.

The time-to-collision (TTC) decreases with increasing HV speed and RV-to-HV relative speed. To compensate for this decrease in TTC, BSW Zone length can be increased with increasing HV speed and the relative speed.

Figure 4.18a shows the relevant target classification zone for BSW and Figure 4.18b shows a possible scenario for BSW.

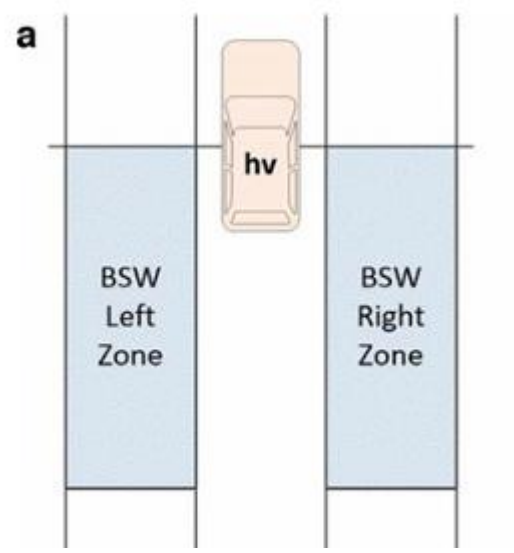


Figure 4.18 a BSW target classification zones

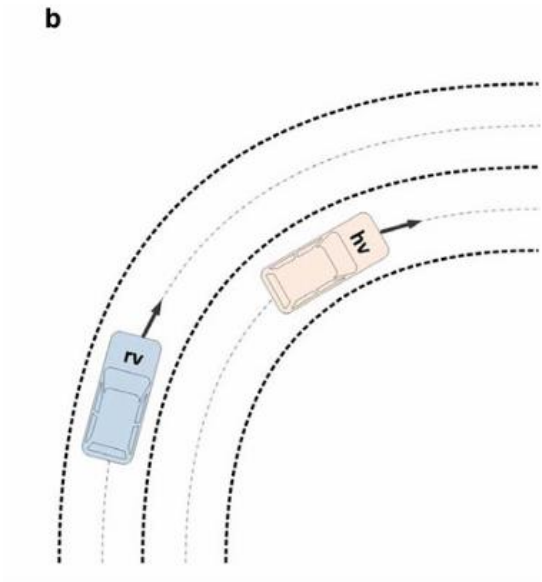


Figure 4.18 b an example scenario for

A simple pseudo-code for the implementation of BSW is given below:

```
BSW_WARNING = FALSE
 IF ((RV_DIRECTION is EQUIDIRECTIONAL) AND
 ((RV_ZONE is BEHIND_LEFT) OR
 (RV_ZONE is BEHIND_RIGHT))) THEN
 REL_SPEED = RV_SPEED- HV_SPEED
 IF (REL_SPEED > 0) THEN
 BSW_ZONE = K_BSW_ZONE_MIN_LEN + K_SPD_MULT*HV_SPEED
 + K_REL_SPD_MULT*REL_SPEED
```

```
ELSE
BSW_ZONE = K_BSW_ZONE_MIN_LEN + K_SPD_MULT*HV_SPEED
END IF
IF (LON_OFFSET < BSW_ZONE) THEN
IF (RV_ZONE is BEHIND_LEFT) THEN
IF LEFT_TURN_SIGNAL is ACTIVATED THEN
BSW_WARNING = BSW_LEFT_WARN
ELSE
BSW_WARNING = BSW_LEFT_ADVISORY
END IF
END IF
```

Use Case:

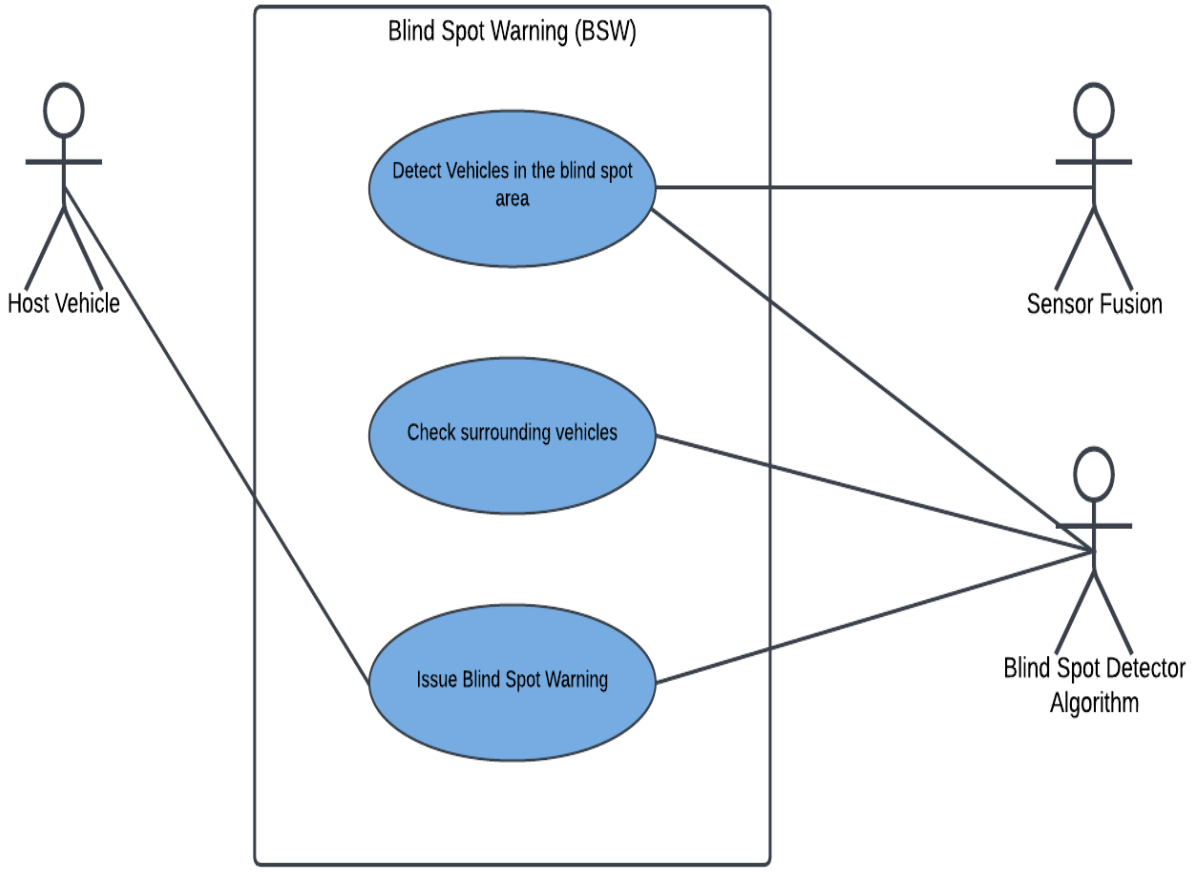


Figure 4.19 Blind-Spot Warning Use Case

Flow chart:

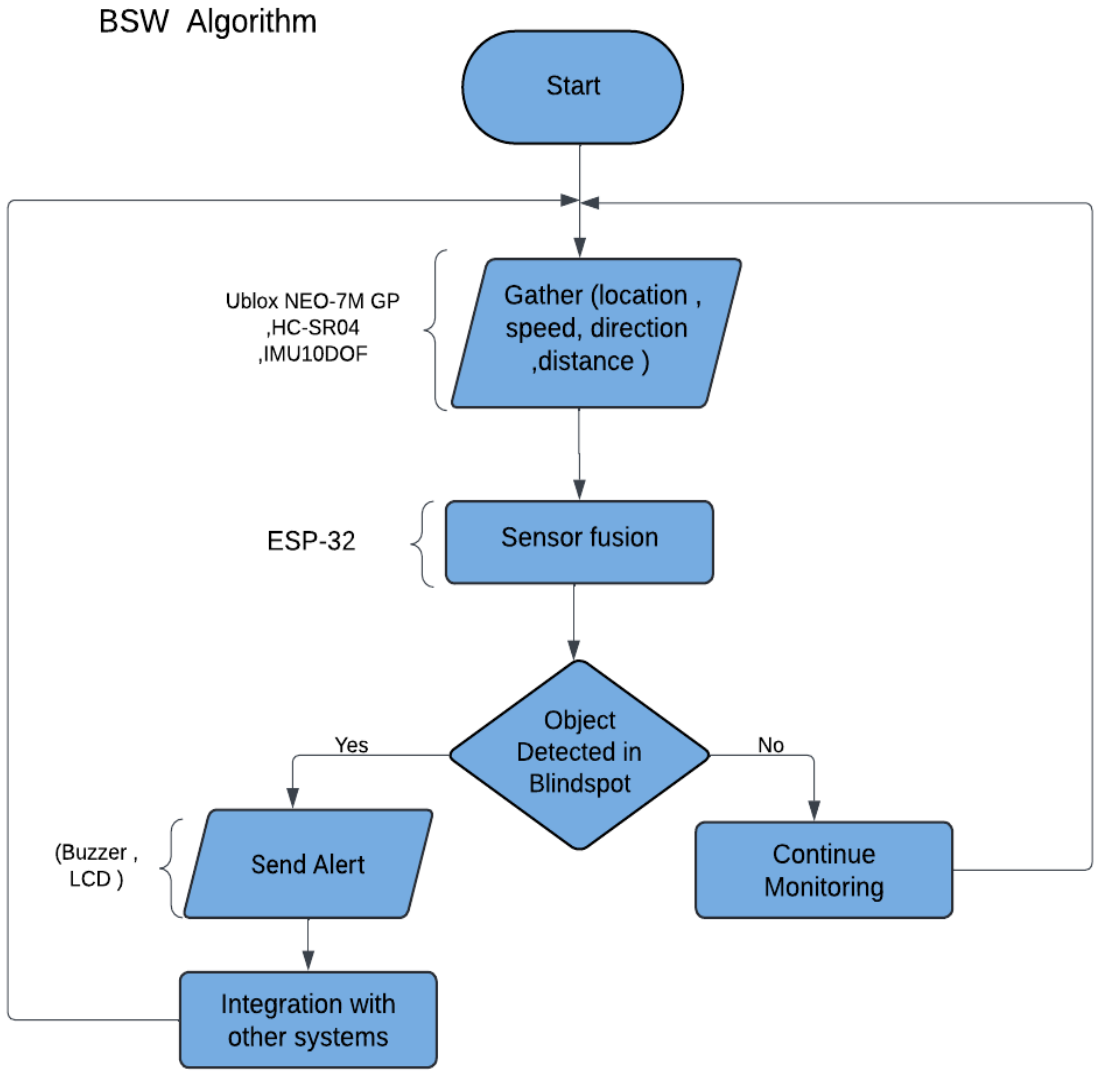


Figure 4.20 *Blind*-Spot Warning Flowchart

4.4.1.5.4 Do Not Pass Warning (DNPW)

DNPW is intended to warn the driver of HV during a passing maneuver attempt when a slower-moving vehicle ahead cannot be passed safely using a passing zone, because the passing zone is occupied by vehicles moving in the opposite direction. This module can either provide advisory information or a warning based on the driver's intent to overtake. The arbitration can be done by observing the left-turn activation signal over vehicle CAN.

DNPW is applicable only for the Remote Vehicles driving in the reverse direction relative to HV in a lane that is classified as ahead left of the HV. DNPW Zone length can be thought of as the addition of three different distances: Acceleration Distance, Overtaking Distance, Acceleration Distance is the distance that the HV travels while accelerating to switch to the

oncoming lane. Overtaking Distance is the distance that the HV travels to overtake the slower vehicle. And the Return Distance is the distance travelled by HV to return to its original lane.

The DNPW Zone Length can be computed by simply adding the three distances shown in Figure 4.22, A warning is issued if the longitudinal offset between the HV and RV plus a buffer zone length is less than the DNPW zone length and the distance travelled by RV during the calibrated overtaking time combined.

Figure 4.21a shows the relevant target classification zone for DNPW and Figure4.21b shows a possible scenario for DNPW.

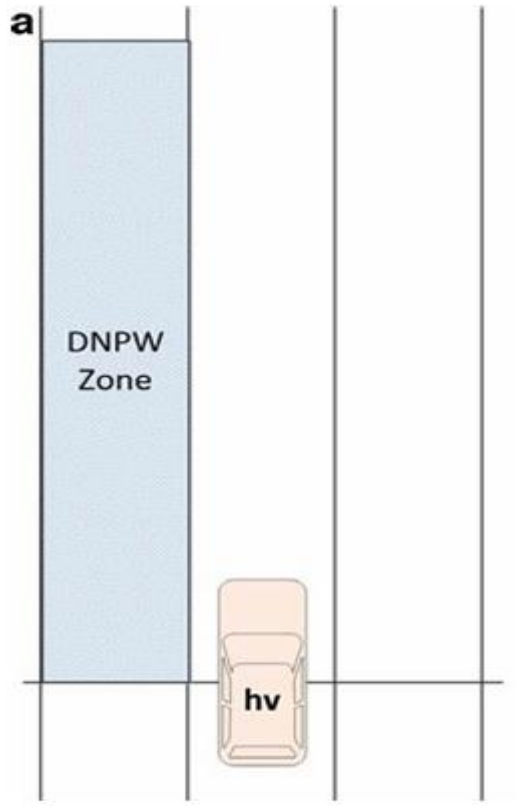


Figure 4.21a DNPW target classification zones

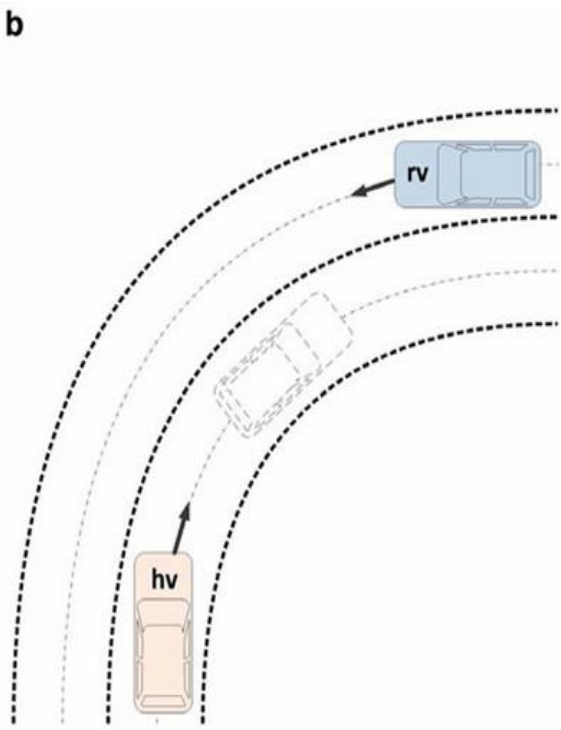


Figure 4.21.b an example scenario for DNPW

$$Acceleration\ Distance = (HV\ Speed * AccelTime) + \left(\frac{1}{2} * HVLongitudinalAccel * AccelTime^2\right) \quad (14)$$

$$Overtaking\ Distance = (HV\ Speed + \Delta\ Speed\ ) * OvertakingTime \quad (15)$$

$$Return\ Distance = (HV\ Speed + \Delta\ Speed\ ) * ReturnTime \quad (16)$$

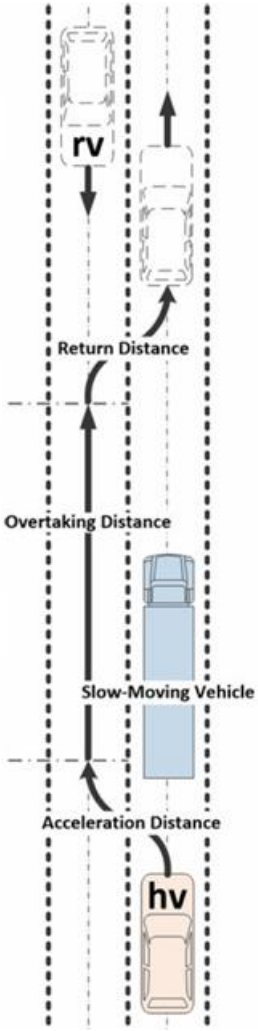


Figure 4.22 Distances that comprise DNPW zone

A simple pseudocode for the implementation of DNPW is given below:

```
DNPW_WARNING = FALSE
IF ((RV_ZONE is AHEAD_LEFT) AND (RV_DIRECTION is REVERSE)) THEN
DNPW_ZONE = ACCEL_DIST + OVERTAKING_DIST + RETURN_DIST
TIME_TAKEN = K_ACCEL_TIME + K_OVERTAKE_TIME + K_RETURN_TIME
IF (LON_OFFSET < (DNPW_ZONE + TIME_TAKEN*RV_SPEED +
K_BUF_ZONE) THEN
DNPW_WARNING = TRUE
END IF
END IF
```

Use Case:

Figure 4.23 Do Not Pass Warning Use Case

Flow chart:

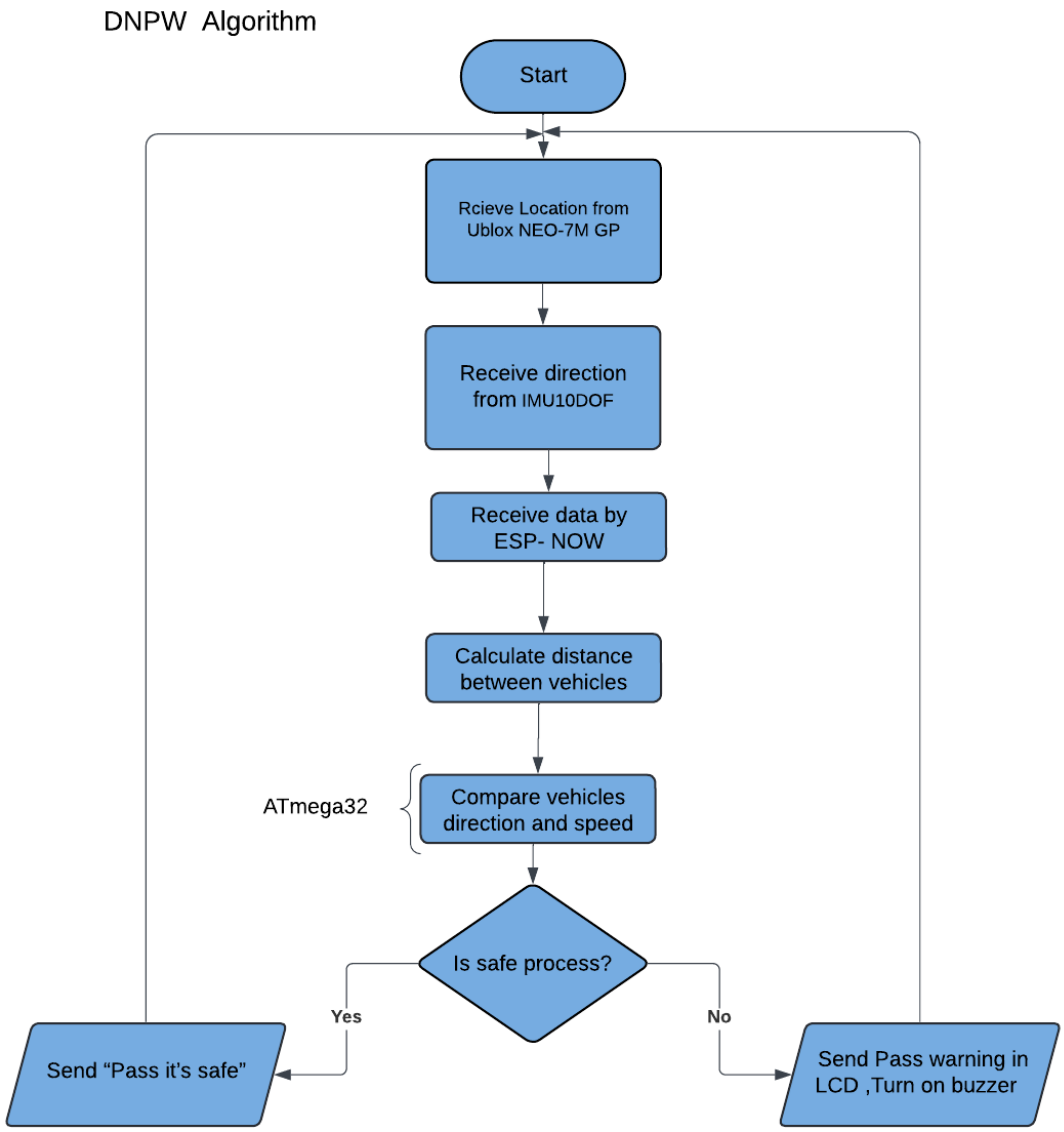


Figure 4.24 Do Not Pass Warning Flowchart

4.4.1.5.5 Intersection Movement Assist (IMA)

IMA safety application is intended to warn the driver of HV if it is not safe for the HV to enter an intersection due to high possibility of collision with other RVs.

This application estimates the time taken by the HV and RV to arrive at the intersection point and issue a warning if both vehicles are predicted to arrive at approximately the same time.

This module can issue either “IMA Right” or “IMA Left” warning-based vehicle position, speed,

heading information. Target Classification Zone and Direction outputs are used to determine if a Remote Vehicle is in either IMA Right or IMA Left Zone.

If the difference between time-taken by HV to travel hv_to_interection_dist (meters) and the time-taken by RV to travel rv_to_interection_dist (meters) is within a calibrated tolerance value, then the IMA warning is issued to the driver of HV.

Figure 4.25a shows the relevant target classification zone for IMA and Figure 4.25 b shows a possible scenario for IMA.

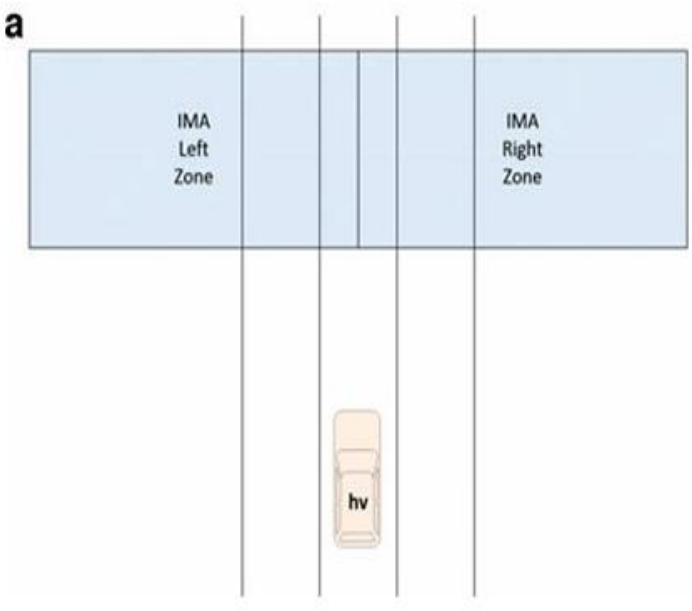


Figure 4.25.a IMA target classification zones

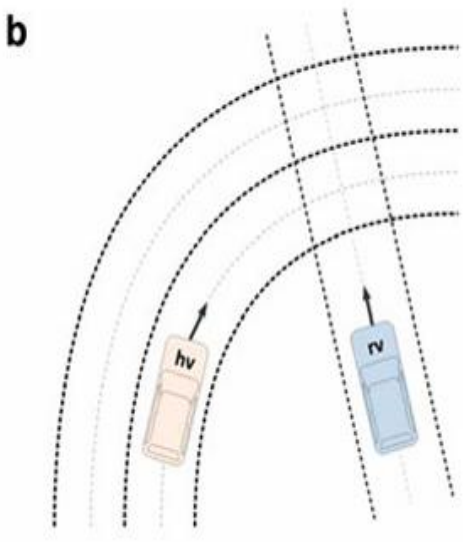


Figure 4. 25.b an example scenario for IMA

For Δ ABC in Figure3.24

$$BC = \frac{AC}{\tan(\Delta\theta)} \tag{17}$$

$$\Rightarrow \quad BC = \frac{lat\_offset}{\tan(\Delta\theta)}$$

$hv\_to\_interection\_dist$ can be given as the following:

$$hv\_to\_interection\_dist = lon\_offset + BC$$

$$\Rightarrow \quad hv\_to\_interection\_dist = lon\_offset + \frac{lat\_offset}{\tan(\Delta\theta)} \tag{18}$$

Similarly, $rv\_to\_interection\_dist$ can be given as the following:

$$rv\_to\_interection\_dist = \frac{lat\_offset}{sin(\Delta\theta)} \tag{19}$$

Figure 4.26 shows a scenario for IMA Left and terminologies used for all the computations.

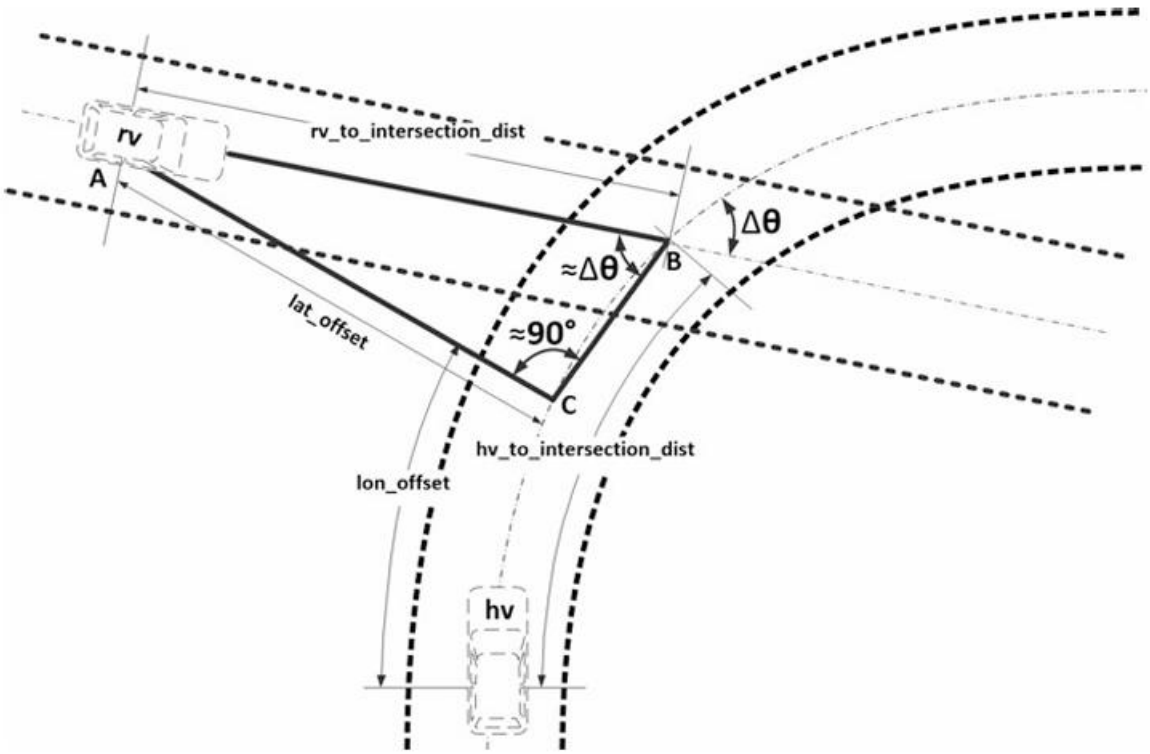


Figure 4.26 Estimation of the time taken by HV and RV to arrive at the intersection point

A simple pseudo-code for the implementation of IMA is given below:

```
IMA_LEFT_WARNING = FALSE
 IMA_RIGHT_WARNING = FALSE
 IF ((RV_DIRECTION is INTERSECTING_RIGHT) AND
 ((RV_ZONE is AHEAD) OR
 (RV_ZONE is AHEAD_RIGHT) OR
 (RV_ZONE is AHEAD_FAR_RIGHT) OR
 (RV_ZONE is AHEAD_FAR_FAR_RIGHT))) THEN
 HV_TO_INTERSECTION_DIST = LON_OFFSET + (LAT_OFFSET/
 TAN(DELTA_HEADING))
 RV_TO_INTERSECTION_DIST = LAT_OFFSET/SIN(DELTA_HEADING)
 HV_TTI = HV_TO_INTERECTION_DIST/HV_SPEED
 RV_TTI = RV_TO_INTERECTION_DIST/RV_SPEED
 IF((HV_TO_INTERECTION_DIST < K_MAX_HV_TO_INTERSECTION_DIST) AND
(ABSOLUTE(HV_TTI- RV_TTI) < K_TTI_TOLERANCE)) THEN
 IMA_RIGHT_WARNING = TRUE
 END IF
 END IF
 IF ((RV_DIRECTION is INTERSECTING_LEFT) AND
 ((RV_ZONE is AHEAD) OR
 (RV_ZONE is AHEAD_LEFT) OR
 (RV_ZONE is AHEAD_FAR_LEFT) OR
 (RV_ZONE is AHEAD_FAR_FAR_LEFT))) THEN
 HV_TO_INTERSECTION_DIST = LON_OFFSET + (LAT_OFFSET/TAN
 (DELTA_HEADING))
 RV_TO_INTERSECTION_DIST = LAT_OFFSET/SIN(DELTA_HEADING)
 HV_TTI = HV_TO_INTERECTION_DIST/HV_SPEED
 RV_TTI = RV_TO_INTERECTION_DIST/RV_SPEED
 IF((HV_TO_INTERECTION_DIST < K_MAX_HV_TO_INTERSECTION_DIST) AND
 (ABSOLUTE (HV_TTI- RV_TTI) < K_TTI_TOLERANCE)) THEN
 IMA_LEFT_WARNING = TRUE
 END IF
 END IF
```

Use Case:

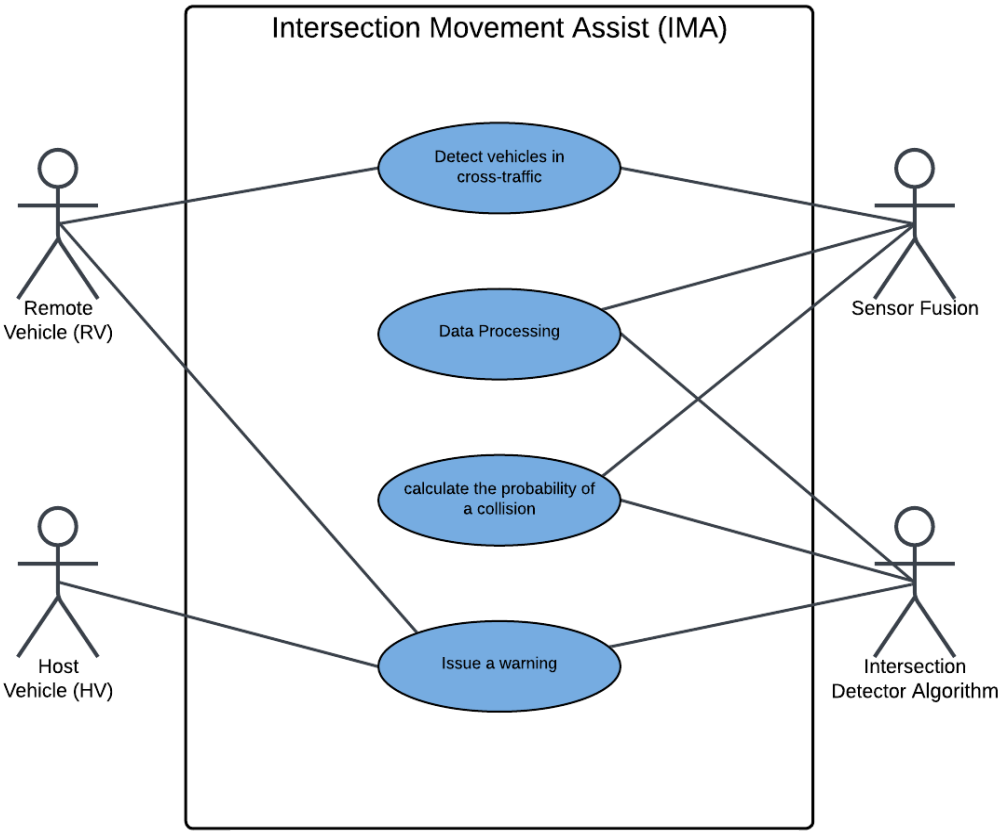


Figure 4.27 Intersection Movement Assist Use Case

Flowchart:

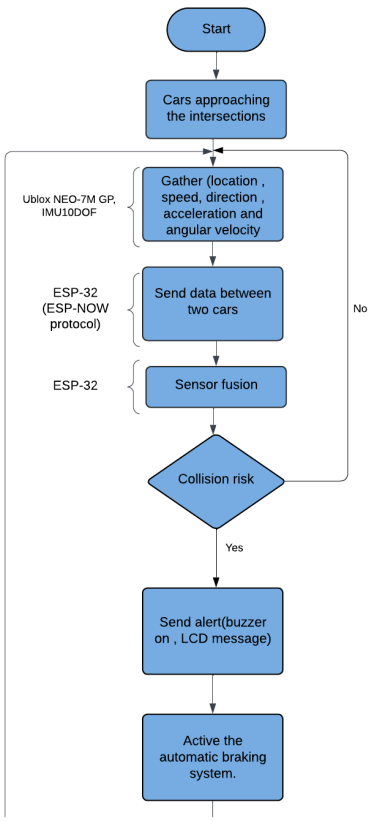


Figure 4.28 Intersection Movement Assist

Following the detailed exploration of V2X communication and its core components—including Vehicle-to-Vehicle (V2V) communication, Basic Safety Message (BSM), and positioning technologies such as GNSS enhanced with Kalman filtering—a comprehensive understanding of real-time vehicular interaction has been established. These foundational elements, supported by Target Classification (TC) mechanisms and a range of V2V-based safety applications such as Forward Collision Warning (FCW), Emergency Electronic Brake Light (EEBL), Blind Spot Warning (BSW), and Do Not Pass Warning (DNPW), collectively form the basis of modern intelligent transportation systems.

To ensure efficient integration, coordination, and execution of these subsystems, a well-defined **software architecture** is essential. This architecture specifies the logical structure, communication pathways, and data processing flows that allow the various modules to operate cohesively within a real-time, safety-critical embedded environment.

4.4.1.6 Software Architecture

To support the system's complexity while maintaining modularity, the software architecture is divided into multiple abstraction layers. Each layer encapsulates a specific level of functionality and communicates only with adjacent layers, ensuring clear separation of responsibilities and improved system organization.

At the top resides the **Application Layer**, where high-level vehicle logic and V2V safety applications are implemented. This layer interacts with underlying services through standardized interfaces. The **RTOS Layer** sits just below, providing deterministic scheduling and task management to support concurrent execution of time-sensitive functions. Supporting this is the **Hardware Abstraction Layer (HAL)**, which offers simplified, high-level drivers for sensors, actuators, and communication modules.

Deeper within the architecture lies the **Microcontroller Abstraction Layer (MCAL)**, which contains low-level drivers responsible for direct interaction with hardware peripherals. At the base is the **Hardware Abstraction Base**, representing the microcontroller's core registers and physical interfaces.

This structured, layered approach enhances software portability, simplifies debugging, and allows seamless integration of new components—all of which are essential in V2V systems where robustness and timing precision are critical.

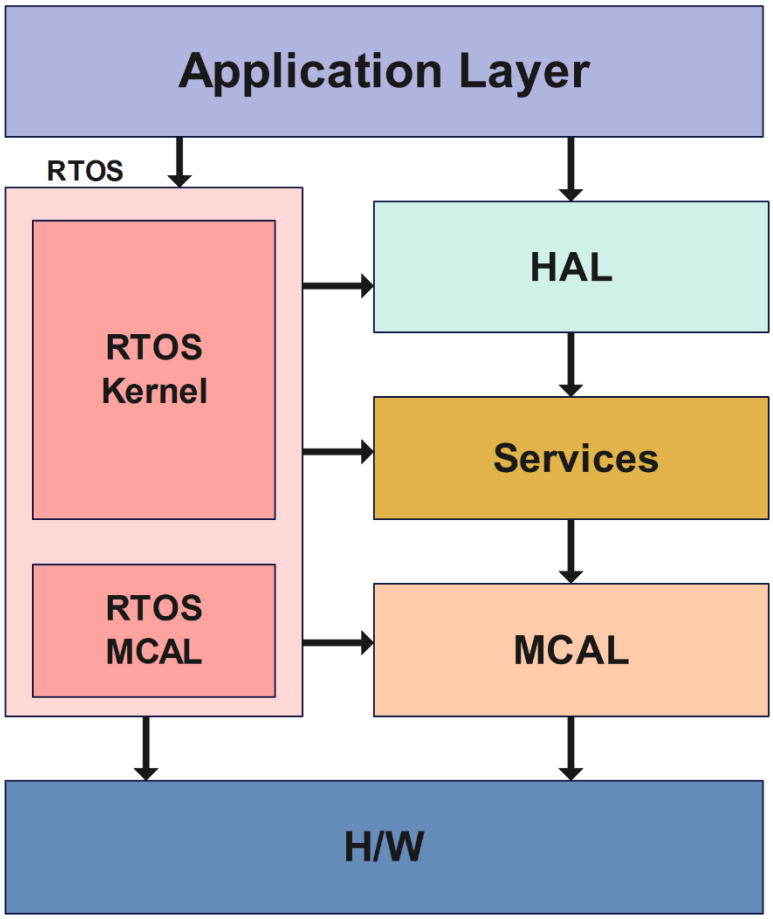


Figure 4.29 Basic Layered Architecture of the System

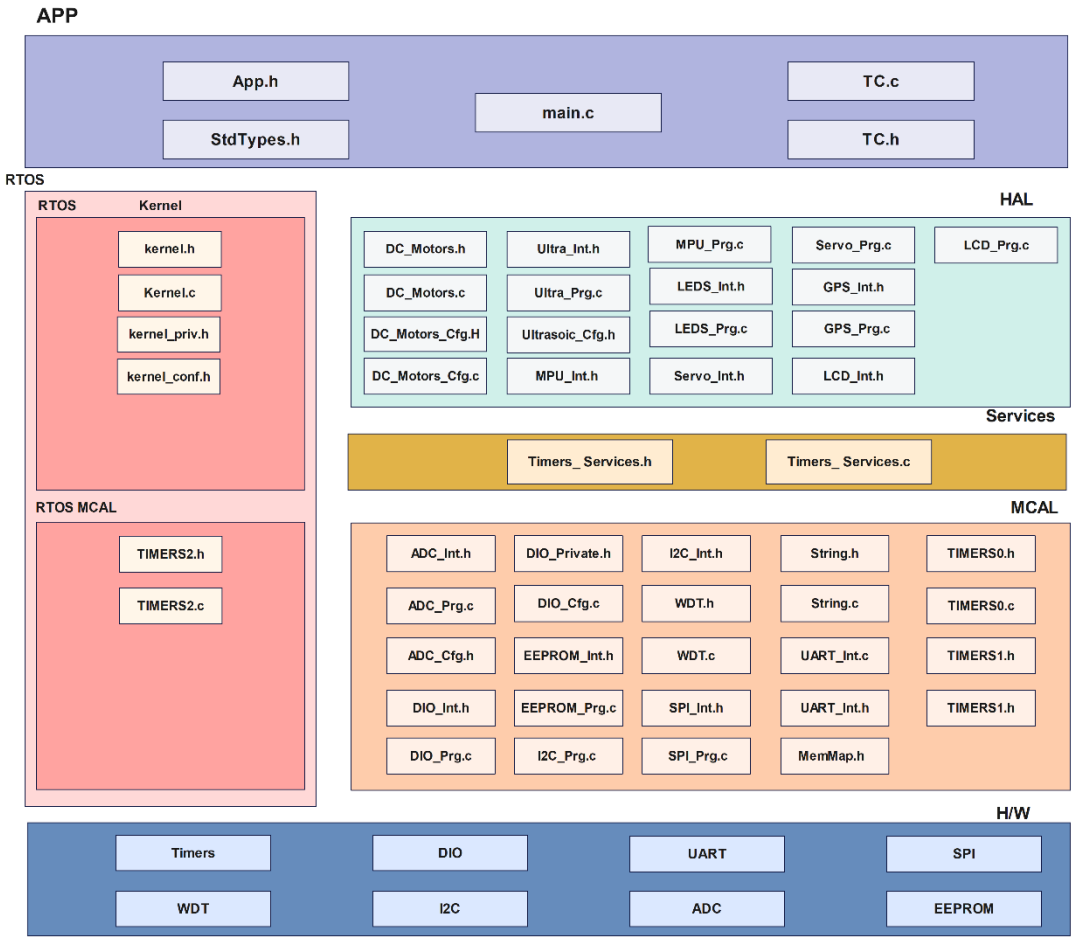


Figure 4.30 Detailed System Architecture and Component Integration

Given the layered nature of the proposed software architecture and the time-sensitive requirements of V2V safety applications, a Real-Time Operating System (RTOS) becomes a fundamental component in ensuring timely task execution, resource management, and system reliability. The RTOS sits at the heart of the architecture, acting as a scheduler and controller for concurrent tasks such as sensor data acquisition, message processing, and safety decision-making. Its integration allows the system to meet strict real-time constraints while maintaining modularity and scalability across different layers of architecture.

4.5 Real-Time Systems Concepts

A Real-Time Operating System (RTOS) is a specialized type of operating system designed to handle tasks with precise timing and deterministic behavior. Unlike general-purpose operating systems, an RTOS prioritizes timely and predictable task execution, making it ideal for embedded systems that interact with the physical world in real time. In the context of V2V applications, the RTOS plays a vital role in managing task scheduling, interrupt handling, synchronization, and inter-task communication. It ensures that safety-critical operations—such as collision warning or signal processing—are executed within their defined deadlines, thereby enhancing the overall reliability and responsiveness of the system.

Real-time systems are characterized by the fact that severe consequences will result if logical as well as timing correctness properties of the system are not met. There are two types of real-time systems: SOFT and HARD. In a SOFT real-time system, tasks are performed by the system as fast as possible, but the tasks don't have to finish by specific times. In HARD real-time systems, tasks have to be performed not only correctly but on time. Most real-time systems have a combination of SOFT and HARD requirements. Real-time applications cover a wide range. Most applications for real-time systems are embedded. This means that the computer is built into a system and is not seen by the user as being a computer.

Real-time software applications are typically more difficult to design than non-real-time applications.

4.5.1 Foreground/Background Systems

Small systems of low complexity are generally designed as shown in Figure 3.29. These systems are called foreground/background or super-loops. An application consists of an infinite loop that calls modules (that is, functions) to perform the desired operations (background). Interrupt Service Routines (ISRs) handle asynchronous events (foreground). Foreground is also called interrupt level while background is called task level. Critical operations must be performed by the ISRs to ensure that they are dealt with in a timely fashion. Because of this, ISRs have a tendency to take longer than they should. Also, information for a background module made available by an ISR is not processed until the background routine gets its turn to execute. This is called the task level response. The worst-case task level response time depends

on how long the background loop takes to execute. Because the execution time of typical code is not constant, the time for successive passes through a portion of the loop is non-deterministic. Furthermore, if a code change is made, the timing of the loop is affected.

Most high-volume microcontroller-based applications (e.g., microwave ovens, telephones, toys, and so on) are designed as foreground/background systems. Also, in microcontroller-based applications, it may be better (from a power consumption point of view) to halt the processor and perform all of the processing in ISRs.

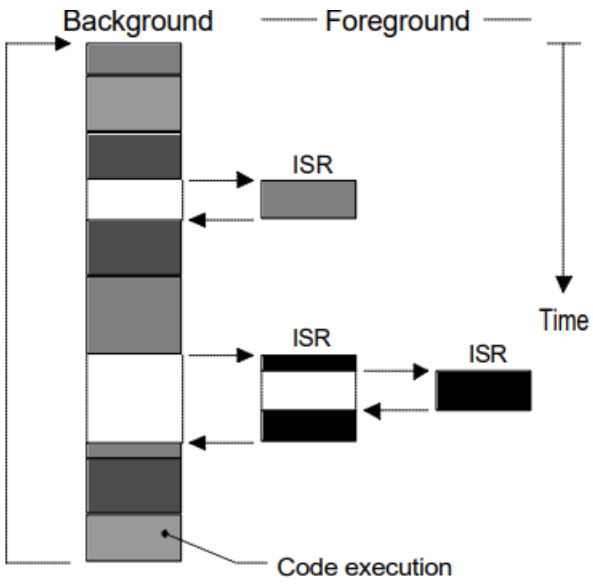


Figure 4.31 Foreground/background systems

4.5.2 Critical Section of Code

A critical section of code, also called a critical region, is code that needs to be treated indivisibly. Once the section of code starts executing, it must not be interrupted. To ensure this, interrupts are typically disabled before the critical code is executed and enabled when the critical code is finished (see also Shared Resource).

4.5.3 Resource

A resource is any entity used by a task. A resource can thus be an I/O device such as a printer, a keyboard, a display, etc. or a variable, a structure, an array, etc.

4.5.4 Shared Resource

A shared resource is a resource that can be used by more than one task. Each task should gain exclusive access to the shared resource to prevent data corruption. This is called Mutual Exclusion and techniques to ensure mutual exclusion are discussed.

4.5.5 Multitasking

Multitasking is the process of scheduling and switching the CPU (Central Processing Unit) between several tasks; a single CPU switches its attention between several sequential tasks. Multitasking is like foreground/background with multiple backgrounds. Multitasking maximizes the utilization of the CPU and also provides for modular construction of applications. One of the most important aspects of multitasking is that it allows the application programmer to manage complexity inherent in real-time applications. Application programs are typically easier to design and maintain if multitasking is used.

4.5.6 Task

A task, also called a thread, is a simple program that thinks it has the CPU all to itself. The design process for a real-time application involves splitting the work to be done into tasks which are responsible for a portion of the problem. Each task is assigned as a priority, its own set of CPU registers, and its own stack area (as shown in Figure 3.30).

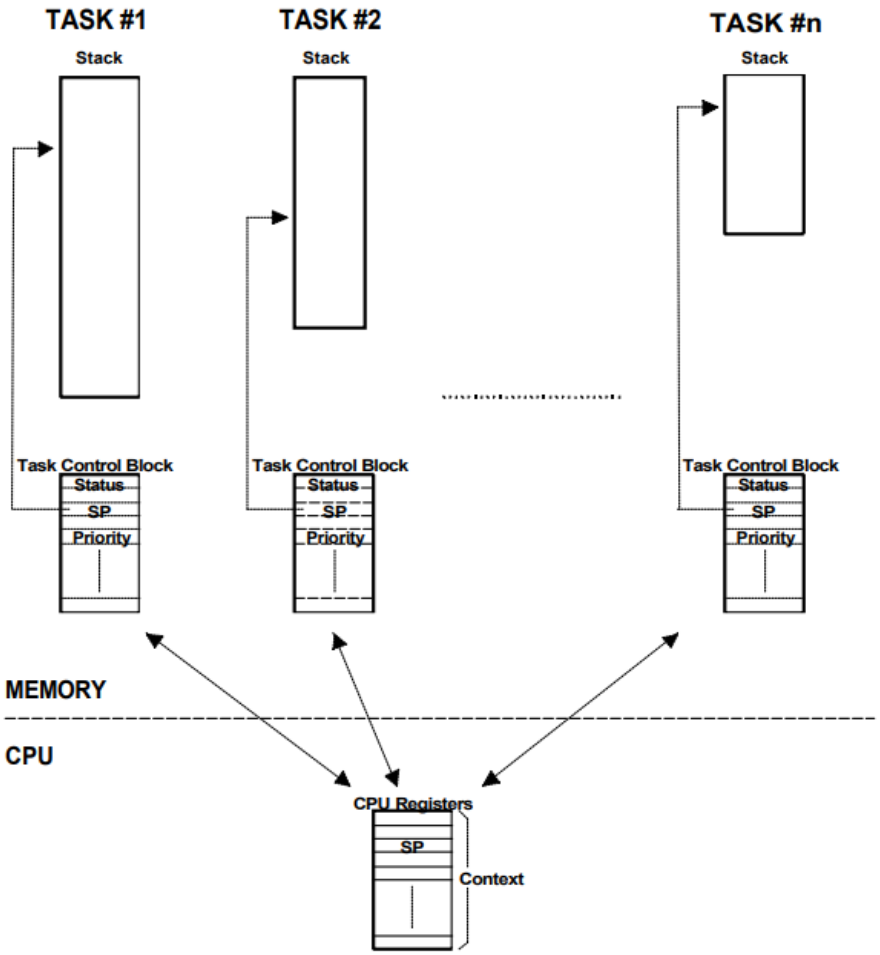


Figure 4.32 Multiple tasks

Each task typically is an infinite loop that can be in any one of five states: DORMANT, READY, RUNNING, WAITING FOR AN EVENT, or INTERRUPTED (see Figure 3.31). The DORMANT state corresponds to a task which resides in memory but has not been made available to the multitasking kernel. A task is READY when it can be executed but its priority is less than the currently running task. A task is RUNNING when it has control of the CPU. A task is WAITING FOR AN EVENT when it requires the occurrence of an event (waiting for an I/O operation to complete, a shared resource to be available, a timing pulse to occur, time to expire etc.). Finally, a task is INTERRUPTED when an interruption has occurred and the CPU is in the process of servicing the interrupt. Figure 3.29 also shows the functions provided by µC/OS-II to make a task switch from one state to another.

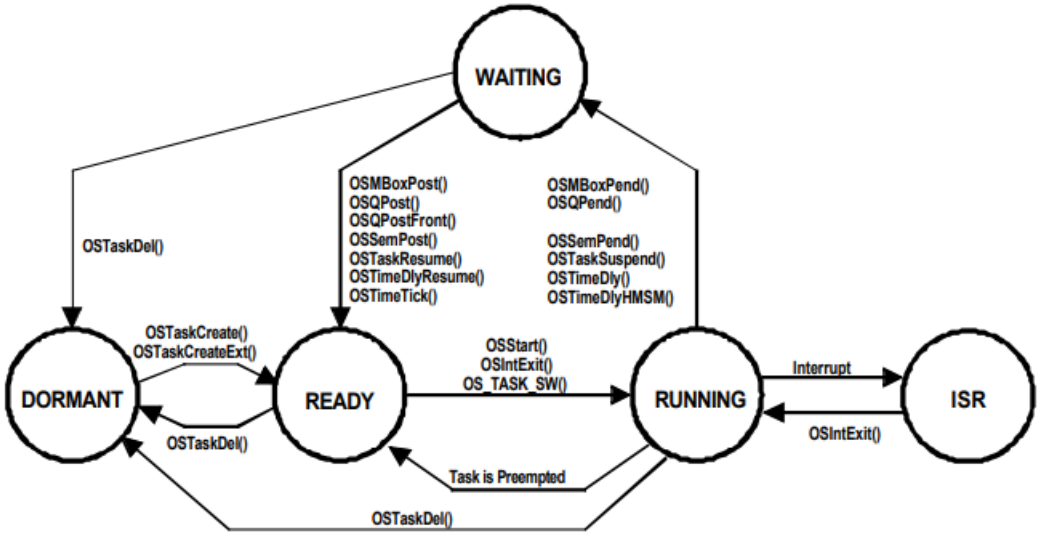


Figure 4.33 Task states

4.5.7 Context Switch (or Task Switch)

When a multitasking kernel decides to run a different task, it simply saves the current task's context (CPU registers) in the current task's context storage area – it's stack (see Figure 3.30). Once this operation is performed, the new task's context is restored from its storage area and then resumes execution of the new task's code. This process is called a context switch or a task switch. Context switching adds overhead to the application. The more registers a CPU has, the higher the overhead. The time required to perform a context switch is determined by how many registers have to be saved and restored by the CPU. Performance of a real-time kernel should not be judged on how many context switches the kernel is capable of doing per second.

4.5.8 Kernel

The kernel is part of a multitasking system responsible for the management of tasks (that is, for managing the CPU's time) and communication between tasks. The fundamental service provided by the kernel is context switching. The use of a real-time kernel will generally simplify the design of systems by allowing the application to be divided into multiple tasks managed by the kernel. A kernel will add overhead to your system because it requires extra ROM (code space), additional RAM for the kernel data structures but most importantly, each task requires its own stack space which has a tendency to eat up RAM quite quickly. A kernel will also consume CPU time (typically between 2 and 5%).

Single chip microcontrollers are generally not able to run a real-time kernel because they have very little RAM.

A kernel can allow you to make better use of your CPU by providing you with indispensable services such as semaphore management, mailboxes, queues, time delays, etc. Once you design a system using a real-time kernel, you will not want to go back to a foreground/background system.

4.5.9 Scheduler

The scheduler, also called the dispatcher, is the part of the kernel responsible for determining which task will run next. Most real-time kernels are priority based. Each task is assigned a priority based on its importance. The priority for each task is application specific. In a priority-based kernel, control of the CPU will always be given to the highest priority task ready-to-run. When the highest-priority task gets the CPU, however, is determined by the type of kernel used. There are two types of priority-based kernels: **non-preemptive** and **preemptive**.

4.5.9.1 Non-Preemptive Kernel

Non-preemptive kernels require that each task does something to explicitly give up control of the CPU. To maintain the illusion of concurrency, this process must be done frequently. Non-preemptive scheduling is also called **cooperative multitasking**; tasks cooperate with each other to share the CPU. Asynchronous events are still handled by ISRs. An ISR can make a higher priority task ready to run, but the ISR always returns to the interrupted task. The new higher priority task will gain control of the CPU only when the current task gives up the CPU.

One of the advantages of a non-preemptive kernel is that interrupt latency is typically low (see the later discussion on interrupts). At the task level, non-preemptive kernels can also use **non-reentrant functions** (discussed later). Non-reentrant functions can be used by each task without fear of corruption by another task. This is because each task can run to completion before it relinquishes the CPU. Non-reentrant functions, however, should not be allowed to give up control of the CPU.

Task-level response using a non-preemptive kernel can be much lower than with foreground/background systems because task-level response is now given by the time of the longest task.

Another advantage of non-preemptive kernels is the lesser need to guard shared data through the use of semaphores. Each task owns the CPU and you don't have to fear that a task will be preempted. This is not an absolute rule, and in some instances, semaphores should still be used. Shared I/O devices may still require the use of mutual exclusion semaphores; for example, a task might still need exclusive access to a printer.

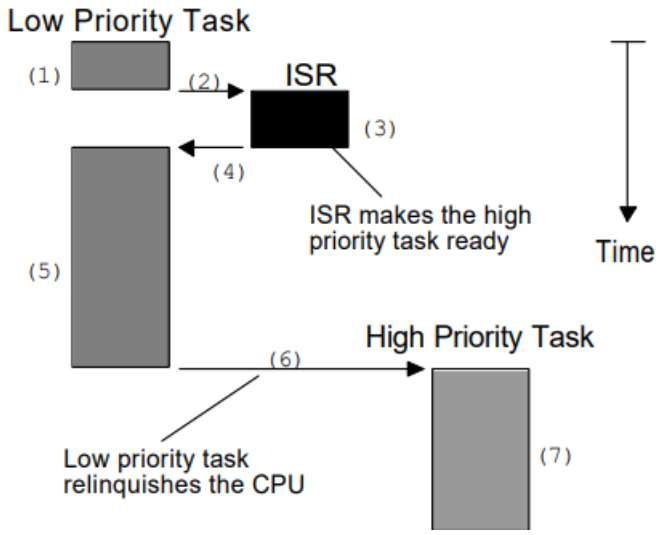


Figure 4.34 Non-preemptive kernel

The execution profile of a non-preemptive kernel is shown in Figure 4.34. A task is executing F 4.34 (1) but gets interrupted. If interrupts are enabled, the CPU vectors (i.e. jumps) to the ISR F 4.34 (2). The ISR handles the event F 4.34 (3) and makes a higher priority task ready-to-run. Upon completion of the ISR, a Return from Interrupt instruction is executed and the CPU returns to the interrupted task F 4.34 (4). The task code resumes at the instruction following the interrupted instruction F 4.34 (5). When the task code completes, it calls a service provided by the kernel to relinquish the CPU to another task F 4.34 (6). The new higher priority task then executes to handle the event signaled by the ISR F 4.34 (7).

The most important drawback of a non-preemptive kernel is responsiveness. A higher priority task that has been made ready to run may have to wait a long time to run, because the current task must give up the CPU when it is ready to do so. As with background execution in

foreground/background systems, task-level response time in a non-preemptive kernel is non-deterministic; you never really know when the highest priority task will get control of the CPU. It is up to your application to relinquish control of the CPU.

To summarize, a non-preemptive kernel allows each task to run until it voluntarily gives up control of the CPU. An interrupt will preempt a task. Upon completion of the ISR, the ISR will return to the interrupted task. Task-level response is much better than with a foreground/background system but is still non-deterministic. Very few commercial kernels are non-preemptive.

4.5.9.2 Preemptive Kernel

A preemptive kernel is used when system responsiveness is important. Because of this, µC/OS-II and most commercial real-time kernels are preemptive. The highest priority task ready to run is always given control of the CPU. When a task makes a higher priority task ready to run, the current task is preempted (suspended) and the higher priority task is immediately given control of the CPU. If an ISR makes a higher priority task ready, when the ISR completes, the interrupted task is suspended and the new higher priority task is resumed. This is illustrated in Figure 3.33.

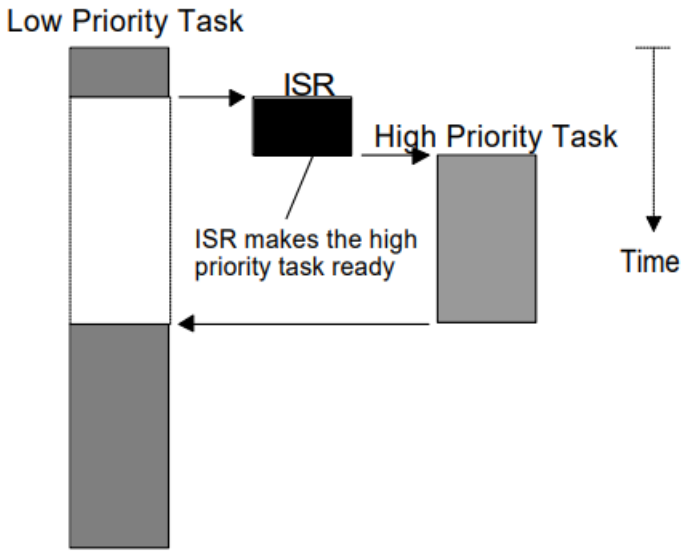


Figure 4.35 Preemptive kernel

With a preemptive kernel, execution of the highest priority task is deterministic; you can determine when the highest priority task will get control of the CPU. Task-level response time is thus minimized by using a preemptive kernel.

Application code using a preemptive kernel should not make use of non-reentrant functions unless exclusive access to these functions is ensured through the use of mu Tual exclusion semaphores, because both a low priority task and a high priority task can make use of a common function. Corruption of data may occur if the higher priority task preempts a lower priority task that is making use of the function.

To summarize, a preemptive kernel always executes the highest priority task that is ready to run. An interrupt will preempt a task. Upon completion of an ISR, the kernel will resume execution to the highest priority task ready to run (not the interrupted task). Task-level response is optimum and deterministic. µC/OS-II is a preemptive kernel.

4.5.10 Reentrancy

A reentrant function is a function that can be used by more than one task without fear of data corruption. A reentrant function can be interrupted at any time and resumed at a later time without loss of data. Reentrant functions either use local variables (i.e., CPU registers or variables on the stack) or protect data when global variables are used. An example of a reentrant function is shown in Reentrant function:

```
void strcpy(char *dest, char *src)
{
 while (*dest++ = *src++) {
 ;
 }
 *dest = NUL;
}
```

Because copies of the arguments to strcpy () are placed on the task's stack, strcpy () can be invoked by multiple tasks without fear that the tasks will corrupt each other's pointers. An example of a non-reentrant function is shown in non-reentrant function . swap () is a simple function that swaps the contents of its two arguments. For the sake of discussion, I assumed

that you are using a preemptive kernel, that interrupts are enabled and that Temp is declared as a global integer:

```
int Temp;
void swap(int *x, int *y)
{
 Temp = *x;
 *x = *y;
 *y = Temp;
}
```

The programmer intended to make swap () usable by any task. Figure 4.36 shows what could happen if a low priority task is interrupted while swap () F 4.36 (1) is executing. Note that at this point Temp contains 1. The ISR makes the higher priority task ready to run, and thus, at the completion of the ISR F 4.36 (2), the kernel (assuming µC/OS-II) is invoked to switch to this task F 4.36 (3). The high priority task sets Temp to 3 and swaps the contents of its variables correctly (that is, z is 4 and t is 3). The high priority task eventually relinquishes control to the low priority task F 4.36 (4) by calling a kernel service to delay itself for 1 clock tick (described later). The lower priority task is thus resumed F 4.36 (5). Note that at this point, Temp is still set to 3! When the low-priority task resumes execution, it sets y to 3 instead of 1.

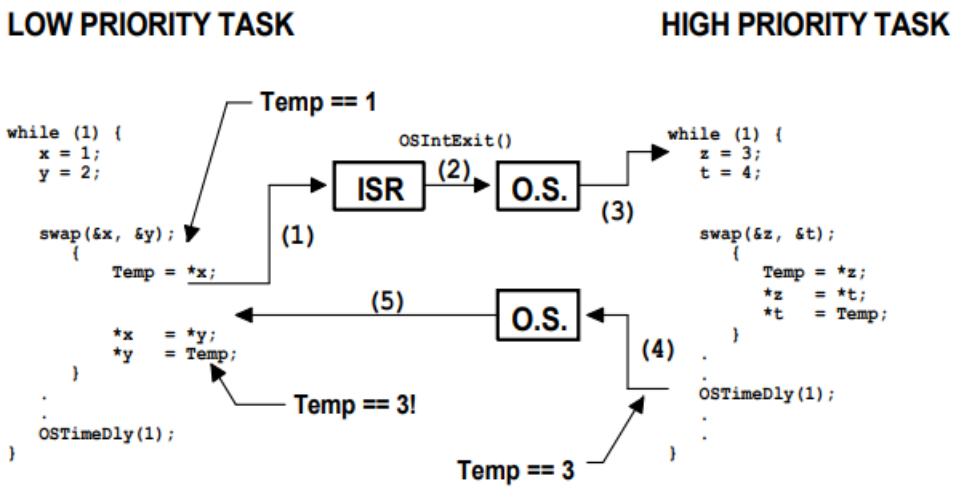


Figure 4.36 non-reentrant function.

Note that this is a simple example and it is obvious how to make the code reentrant. However, other situations are not as easy to solve. An error caused by a non-reentrant function may not show up in your application during the testing phase; it will most likely occur once the product has been delivered! If you are new to multitasking, you will need to be careful when using non-reentrant functions.

We can make swap () reentrant by using one of the following techniques:

- Declare Temp local to swap ().
- Disable interrupts before the operation and enable them after.
- Use a semaphore.

If the interruption occurs either before or after swap (), the x and y values for both tasks will be correct.

4.5.11 Round Robin Scheduling

When two or more tasks have the same priority, the kernel will allow one task to run for a predetermined amount of time, called a quantum, and then selects another task. This is also called time slicing. The kernel gives control to the next task in line if:

- the current task doesn't have any work to do during its time slice or
- the current task completes before the end of its time slice.

Our RTOS does not currently support round-robin scheduling. Each task must have a unique priority in your application

4.5.12 Task Priority

A priority is assigned to each task. The more important the task, the higher the priority given to it.

4.5.13 Static Priorities

Task priorities are said to be static when the priority of each task does not change during the application's execution. Each task is thus given a fixed priority at compile time. All the tasks and their timing constraints are known at compile time in a system where priorities are static. **WHICH we provide it in our RTOS**.

4.5.14 Dynamic Priorities

Task priorities are said to be dynamic if the priority of tasks can be changed during the application's execution; each task can change its priority at run-time. This is a desirable feature to have in a real-time kernel to avoid priority inversions.

4.5.15 Deadlock (or Deadly Embrace)

A deadlock, also called a deadly embrace, is a situation in which two tasks are each unknowingly waiting for resources held by each other. If task T1 has exclusive access to resource R1 and task T2 has exclusive access to resource R2, then if T1 needs exclusive access to R2 and T2 needs exclusive access to R1, neither task can continue. They are deadlocked. The simplest way to avoid a deadlock is for tasks to:

- acquire all resources before proceeding.
- acquire the resources in the same order.
- release the resources in the reverse order.

Most kernels allow you to specify a timeout when acquiring a semaphore. This feature allows a deadlock to be broken. If the semaphore is not available withing a certain amount of time, the task requesting the resource will resume execution. Some form of error code must be returned to the task to notify it that a timeout has occurred. A return error code prevents the task from thinking it has obtained to the resource. Deadlocks generally occur in large multitasking systems and are not generally encountered in embedded systems.

4.5.16 Interrupts

An interrupt is a hardware mechanism used to inform the CPU that an asynchronous event has occurred. When an interrupt is recognized, the CPU saves part (or all) of its context (i.e. registers) and jumps to a special subroutine called an Interrupt Service Routine, or ISR. The ISR processes the event and upon completion of the ISR, the program returns to:

- The background for a foreground/background system.
- The interrupted task for a non-preemptive kernel.
- The highest priority task ready-to-run for a preemptive kernel.

Interrupts allow a microprocessor to process events when they occur. This prevents the microprocessor from continuously polling an event to see if this event has occurred. Microprocessors allow interrupts to be ignored and recognized through the use of two special instructions: disable interrupts and enable interrupts, respectively. In a real-time environment,

interrupts should be disabled as little as possible. Disabling interrupts affects interrupt latency and also, disabling interrupts may cause interrupts to be missed. Processors generally allow interrupts to be nested. This means that while servicing an interrupt, the processor will recognize and service other (more important) interrupts as shown in Figure 4.37.

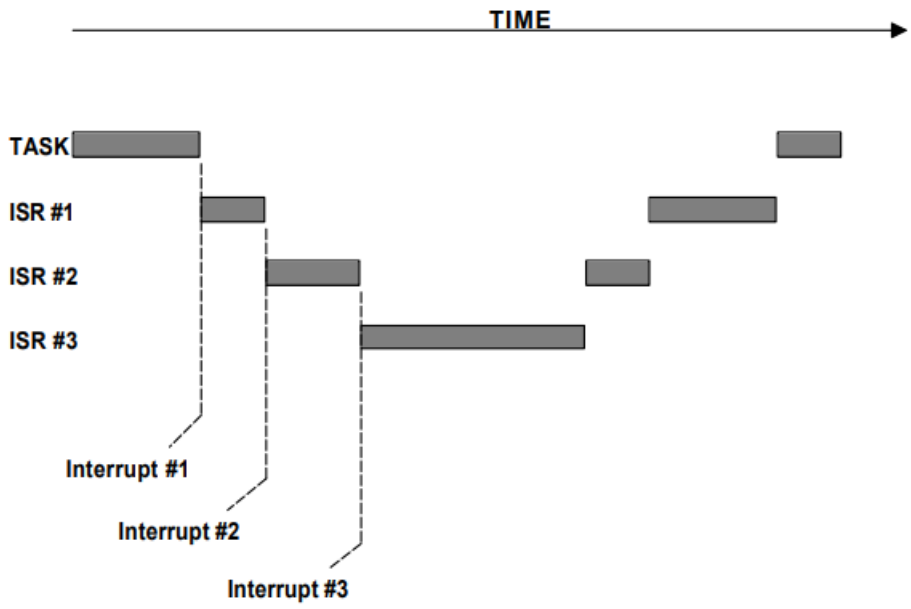


Figure 4.37 Interrupt nesting.

4.5.16.1 Interrupt Latency, Response, and Recovery

Figures 3.36, 3.37 and 3.38 show the interrupt latency, response, and recovery for a foreground/background system, a non-preemptive kernel, and a preemptive kernel, respectively.

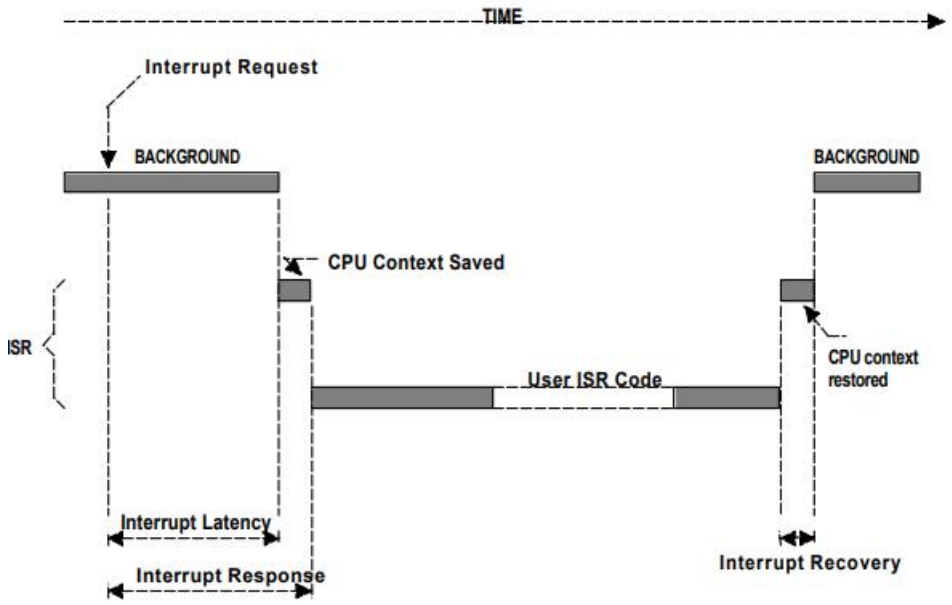


Figure 4.38 Interrupt latency, response, and recovery (Foreground/Background)

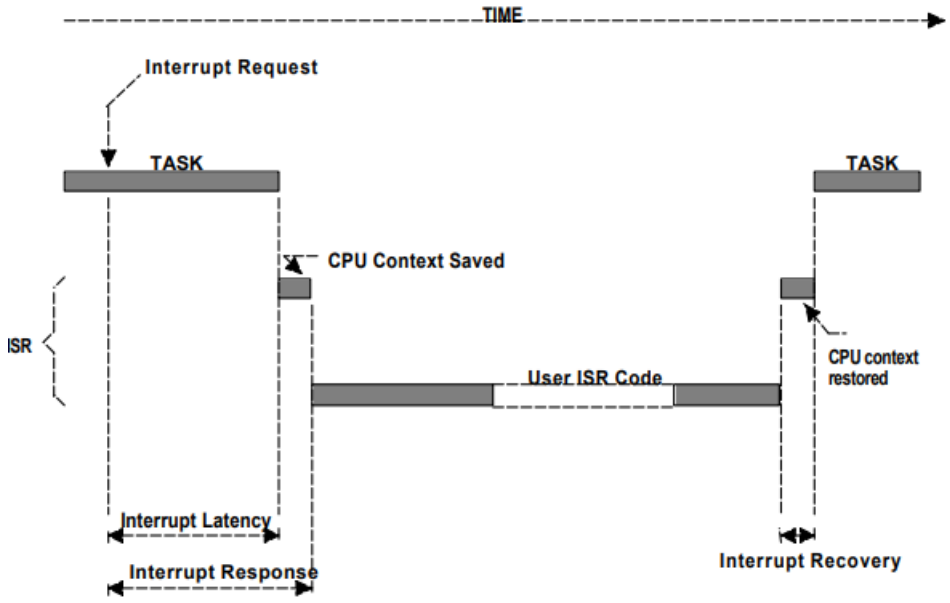


Figure 4.39 Interrupt latency, response, and recovery (Non-preemptive kernel)

You should note that for a preemptive kernel, the exit function either decides to return to the interrupted task F 4.38 A or to a higher priority task that the ISR has made ready-to-run F 4.38 B. In the later case, the execution time is slightly longer because the kernel has to perform a context switch. I made the difference in execution time somewhat to scale assuming RTOS processor.

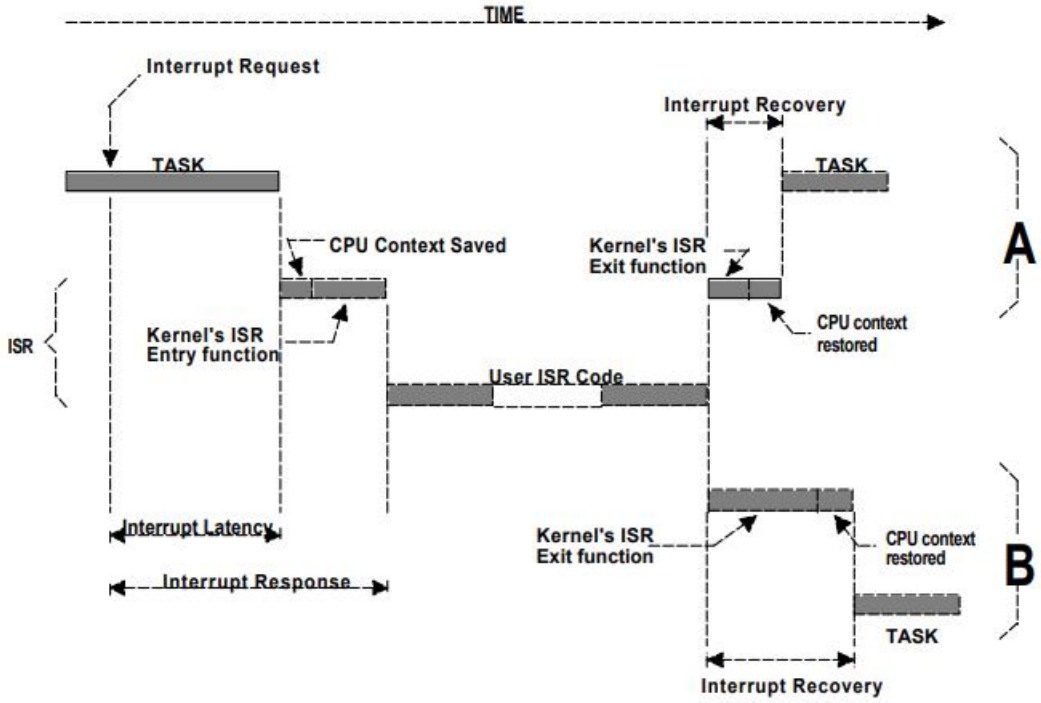


Figure 4.40 Interrupt latency, response, and recovery (Preemptive kernel)

4.5.17 Clock Tick

A clock tick is a special interrupt that occurs periodically. This interrupt can be viewed as the system's heartbeat. The time between interrupts is application specific and is generally between 10 and 200 mS. The clock tick interrupt allows a kernel to delay tasks for an integral number of clock ticks and to provide timeouts when tasks are waiting for events to occur. The faster the tick rate, the higher the overhead imposed on the system.

All kernels allow tasks to be delayed for a certain number of clock ticks. The resolution of delayed tasks is 1 clock tick, however, this does not mean that its accuracy is 1 clock tick.

Figures 4.41 through 2-26 are timing diagrams showing a task delaying itself for 1 clock tick. The shaded areas indicate the execution time for each operation being performed. Note that the time for each operation varies to reflect typical processing, which would include loops

and conditional statements (i.e., if/else, switch and ?:). The processing time of the 'Tick ISR' has been exaggerated to show that it too is subject to varying execution times.

Case 1 (Figure 4.41) shows a situation where higher priority tasks and ISRs execute prior to the task, which needs to delay for 1 tick. As you can see, the task attempts to delay for 20 mS but because of its priority, actually executes at varying intervals. This will thus cause the execution of the task to jitter.

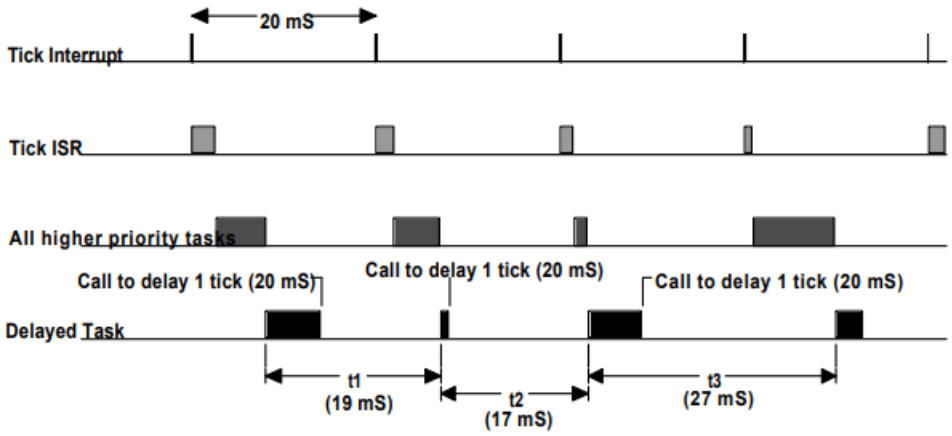


Figure 4.41 Delaying a task for 1 tick (case #1).

Case 2 (Figure 4.42) shows a situation where the execution times of all higher-priority tasks and ISRs are slightly less than one tick. If the task delays itself just before a clock tick, the task will execute again almost immediately! Because of this, if you need to delay a task for at least 1 clock tick, you must specify one extra tick. In other words, if you need to delay a task for at least 5 ticks, you must specify 6 ticks!

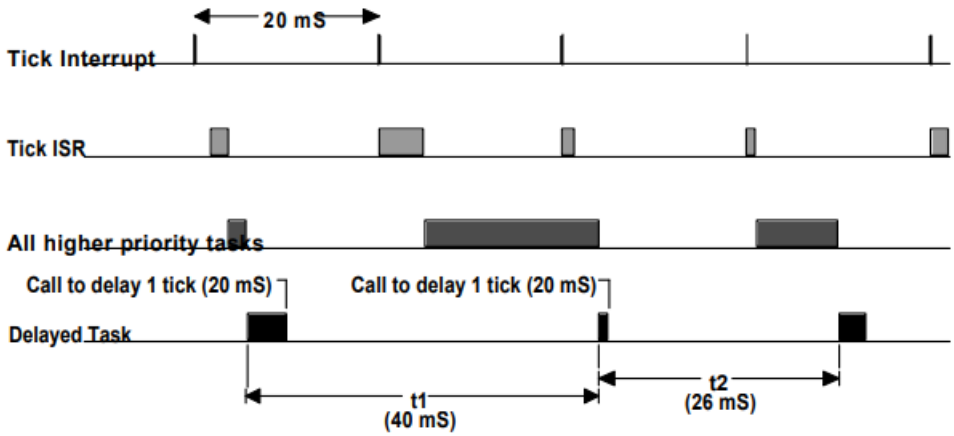


Figure 4.42 Delaying a task for 1 tick (case #3).

Case 3 (Figure 4.43) shows a situation where the execution times of all higher-priority tasks and ISRs extend beyond one clock tick. In this case, the task that tries to delay for 1 tick will actually execute 2 ticks later! In this case, the task missed its deadline. This might be acceptable in some applications, but in most cases it isn't.

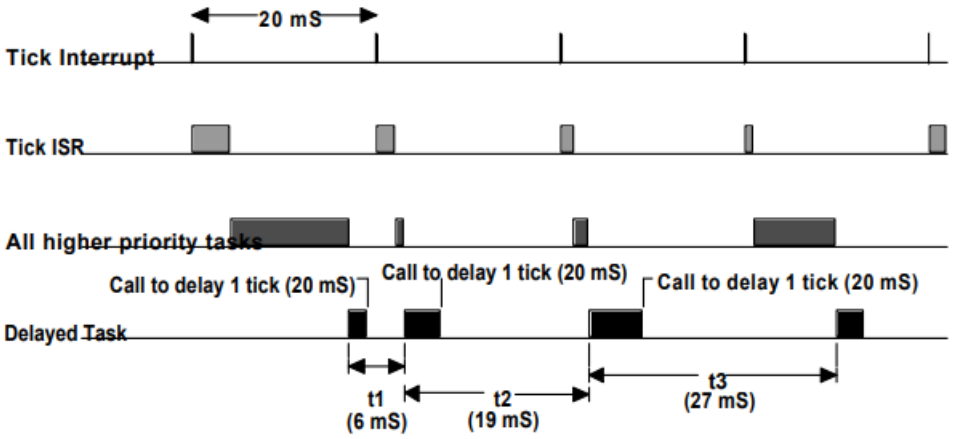


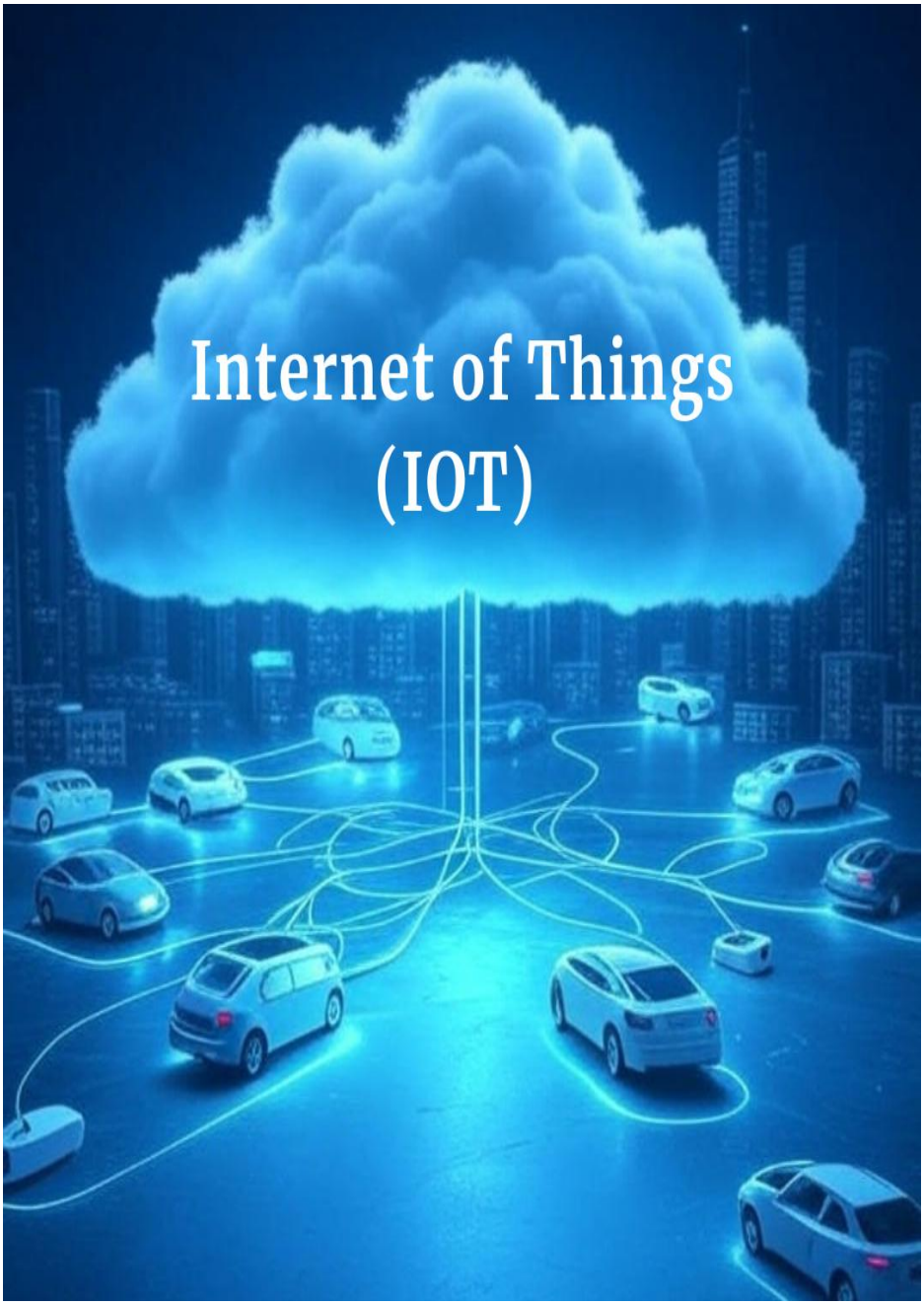
Figure 4.42 Delaying a task for 1 tick (case #2).

These situations exist with all real-time kernels. They are related to CPU processing load and possibly incorrect system design. Here are some possible solutions to these problems:

- Increase the clock rate of your microprocessor.
- Increase the time between tick interrupts.
- Rearrange task priorities.
- Avoid using floating-point math (if you must, use single precision).
- Get a compiler that performs better code optimization.
- Write time-critical code in assembly language.
- If possible, upgrade to a faster microprocessor in the same family, e.g., 8086 to 80186, 68000 to 68020, etc.

Regardless of what you do, jitters will always occur.

Chapter 5



Chapter 5: Internet of Things (IOT)

5.1 Brief of Network

Vehicle-to-Vehicle (V2V) communication is a pivotal element of intelligent transportation systems (ITS), enabling vehicles to exchange real-time information such as position, speed, and direction to enhance road safety and traffic efficiency. This exchange allows vehicles to anticipate and react to potential hazards, such as sudden braking or lane changes, more effectively than human drivers alone. Standard V2V technologies include Dedicated Short Range Communications (DSRC), based on the IEEE 802.11p standard, and Cellular-V2X (C-V2X), which leverages cellular networks (e.g., 4G/5G) for broader coverage. These technologies are designed for robust and reliable communication in automotive environments, with DSRC operating on a dedicated 5.9 GHz frequency band for low-latency, short-range communication, and C-V2X offering scalability for urban settings.

However, for educational, research, or proof-of-concept projects, alternative technologies can be employed to demonstrate V2V principles. In this project, we utilize the **ESP-NOW** protocol, a connectionless Wi-Fi communication protocol developed by Espressif Systems for their ESP32 and ESP8266 microcontrollers. ESP-NOW operates on the 2.4 GHz Wi-Fi band, offering a simple, low-latency, and low-power solution for peer-to-peer communication, making it an ideal choice for our project's requirements, particularly when working with cost-effective hardware like the ESP32.

5.2 Why Use ESP-NOW for V2V?

While DSRC and C-V2X are standard choices for automotive V2V communication, they often require specialized hardware and infrastructure, which may not be feasible for academic or small-scale projects. In our case, **DSRC** was not an option due to regional frequency allocation constraints, **as the 5.9 GHz band is reserved for higher-priority applications in our area**. Notably, the use of this frequency band (5.9 GHz) for vehicular communications is strictly prohibited by national telecommunications regulations in **Egypt**, legally preventing any DSRC implementation.

5.2.1 DSRC Frequency Is Banned in Egypt

According to the National Telecom Regulatory Authority (**NTRA**) in Egypt, the use of the DSRC frequency band (**5.875–5.925 GHz**) is currently restricted and not permitted for public or unlicensed applications.

5850-5925 MHz			
FIXED **FIXED-SATELLITE (Earth-to-space)** **MOBILE** **5.150**	FIXED FIXED-SATELLITE (Earth-to-space) MOBILE **EG01, EG03, EG08**	SRD: ▪ Tank Level Probing Radar (TLPR)	

Figure 5.1 DSRC Frequency Is Banned in Egypt

ESP-NOW, by contrast, leverages the built-in Wi-Fi capabilities of ESP32 microcontrollers, providing a cost-effective and accessible way to implement V2V-like communication. Its operation on the unlicensed 2.4 GHz spectrum ensures full regulatory compliance with local laws. Although it may not match the range or standardization of DSRC or C-V2X, ESP-NOW is sufficient for demonstrating the core concepts of V2V communication within the scope of this project, especially given the project's focus on educational and research applications.

Here's a simplified comparison **Table 5.1** between **ESP-NOW** and **DSRC** for **V2V (Vehicle-to-Vehicle) communication**:

Feature	**ESP-NOW**	**DSRC (IEEE 802.11p)**
Purpose	IoT, small-scale V2V projects	Automotive safety (V2V, V21)
Range	Up to ~200m (open area)	Up to ~1000m (vehicular use)
Latency	Very low (~ms)	Ultra-low (~ms), safety-optimized
Data Rate	~10 Mbps (Wi-Fi based)	3-27 Mbps (adjustable)
Frequency Band	2.4 GHz (licensed)	5.9 GHz (unlicensed, less interference)
Security	AES-128 (basic)	Strong (WAVE Security, PKI)
Cost	Cheap (ESP32 chips)	Expensive (certified hardware)
Interoperability	Limited to ESP devices	Standardized (works across vendors)

So that, **we chose ESP-NOW** for prototyping due to its very-low latency and cost-effectiveness, while recognizing DSRC remains the industry standard for production V2V systems with its longer range and stronger security. This approach allowed rapid testing while maintaining a path to future DSRC implementation.

5.3 Overview of ESP-NOW

ESP-NOW (Espressif Systems, n.d.) is a proprietary communication protocol developed by Espressif Systems for their ESP32 and ESP8266 microcontrollers. It is designed for low-latency, connectionless, peer-to-peer communication over the 2.4 GHz Wi-Fi band. Unlike traditional Wi-Fi, which requires establishing a connection through a router, ESP-NOW allows devices to communicate directly with each other without the need for a network infrastructure, making it suitable for applications requiring quick data exchange.

5.3.1 Frame Format

ESP-NOW uses a custom frame format that **includes:**

- **Category Code:** Identifies the frame as a vendor-specific action frame, set to 127.
- **Organization Identifier:** A unique identifier for Espressif Systems (0x18fe34), ensuring frame authenticity.
- **Random Value:** Prevents relay attacks by adding a random value to each frame, enhancing security.
- **Vendor-Specific Content:** Contains the actual data to be transmitted, along with metadata like version and type. The maximum packet length is 250 bytes for version 1.0 devices and 1470 bytes for version 2.0 devices (Espressif Systems, n.d.), ensuring efficient transmission while keeping the protocol lightweight.

This frame format allows for rapid data exchange, critical for real-time V2V applications where delays could compromise safety.

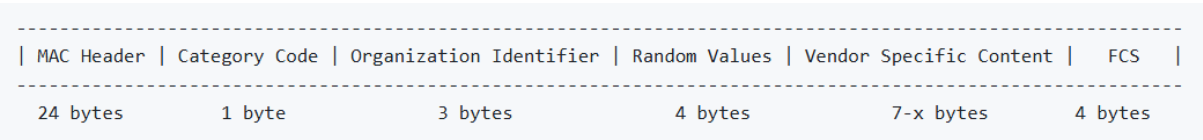


Figure 5.2 ESP-NOW Frame Structure Breakdown

5.3.2 Key Features of ESP-NOW

The protocol offers several key features that align with the needs of V2V communication in this project:

- **Connectionless Communication:**Devices can send and receive data without establishing a connection, reducing latency .
- **Low Power Consumption:**Ideal for battery-powered devices, as it operates efficiently with minimal overhead, aligning with the project's focus on energy-efficient systems.
- **High Throughput:**Supports data rates up to 10 Mbps, sufficient for exchanging safety-critical data like position and speed in V2V scenarios.
- **Security:**Supports encryption (e.g., AES) for secure data transmission, which is crucial for protecting safety-critical information from unauthorized access or tampering.
- **Flexible Networking Modes:** Supports various configurations, including one-to-one, one-to-many, and many-to-one communication, allowing for adaptable network topologies in vehicular environments.

In the context of this project, ESP-NOW enables vehicles equipped with ESP32 modules to share critical safety information, such as position, speed, and direction, in real-time. This enhances the overall effectiveness of the Advanced Driver Assistance System (ADAS) by providing a reliable and responsive communication layer, as mentioned in the project's abstract and introduction sections.

5.4 How ESP-NOW Works

ESP-NOW operates by encapsulating application data in a vendor-specific action frame, which is then transmitted wirelessly between devices. The protocol is designed to be simple and efficient, making it suitable for applications requiring quick data exchange, such as V2V communication.

5.4.1 Networking Modes

ESP-NOW supports various networking configurations, which are particularly relevant for V2V communication:

- **Initiator-Responder Model:** One device (initiator) sends data to another (responder). Each device can act as both an initiator and a responder, enabling bidirectional communication, which is essential for vehicles to both send and receive safety information.

Figure 5.3 Basic ESP-NOW One-Way Communication

- **One-to-Many Communication:** A single initiator can broadcast data to multiple responders, useful for scenarios where a vehicle needs to alert multiple nearby vehicles, such as in traffic congestion.

Figure 5. 4 ESP-NOW One-to-Many Communication

- **Many-to-One Communication:** Multiple initiators can send data to a single responder, suitable for centralized monitoring systems in the project.

Figure 5.5 ESP-NOW Many-to-One Communication Communication

- **Two-Way Communication:** Devices can exchange data bidirectionally, with each acting as both initiator and responder, aligning with the project's need for dynamic, peer-to-peer V2V interactions.

In our project, each vehicle's ESP32 module operates in a **two-way communication mode**, allowing it to both send and receive data from other vehicles, ensuring comprehensive situational awareness.

Figure 5.6 ESP-NOW Two-Way Communication

5.4.2 MAC Addresses

When Initiators communicate with Responders, they need to know the Responder's MAC Address. A MAC, or Media Access Control, Address is a unique 6-digit hexadecimal number assigned to every device on a network. It is generally burned into the device by the manufacturer, although it is possible to manually set it.

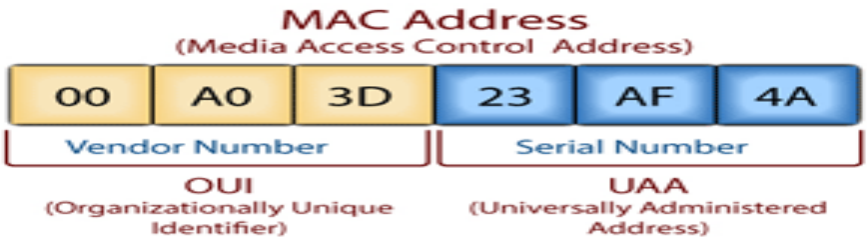


Figure 5.7 MAC Address Structure

This process ensures that vehicles can continuously share and process safety-related data, such as position and speed, to enhance situational awareness, as outlined in the project's implementation details.

5.4.3 Communication Process

The communication process in ESP-NOW involves:

- **Sending Data:** The initiator device sends a packet to the responder, specifying the responder's MAC address. This is facilitated by pre-registering MAC addresses, ensuring direct communication between vehicles.
- **Receiving Data:** The responder processes the incoming packet and can respond if necessary, enabling real-time data exchange.
- **Error Handling:** ESP-NOW provides callbacks for send and receive operations, allowing the system to handle transmission failures or acknowledgments, which is crucial for maintaining reliability in dynamic vehicular environments.

5.5 Implementation Details

In our system, each vehicle is equipped with an ESP32 module connected to the vehicle's sensors and control systems. The ESP32 handles the V2V communication using ESP-NOW, periodically broadcasting its status (e.g., position, speed, direction) and listening for messages from other vehicles.

5.5.1 Hardware Setup

1. Each vehicle is equipped with an ESP32 module, as detailed in the project's hardware components chapter (Section 2.2.1.2.2), which highlights the ESP32's dual-core processor and Wi-Fi/Bluetooth capabilities.
2. **The ESP32 is connected to:**
    - GPS module (e.g., Ublox NEO-7M**, as mentioned in Section 3.2.2.2.2**) for position data, providing latitude and longitude for V2V messages.
    - Other sensors (e.g., speed sensors, accelerometers, **as listed in Section 3.2.2**) for additional vehicle status information.
3. The system also includes an internal camera for driver monitoring and an external camera for traffic sign recognition, as described in other chapters, ensuring comprehensive data integration.

5.5.2 Software Implementation

- **Programming Environment:** Arduino IDE or similar platforms are used to program the ESP32.
- **ESP-NOW Configuration:**
    - Each ESP32 is configured as both an initiator and a responder, enabling bidirectional communication.
    - MAC addresses of other vehicles' ESP32 modules are pre-registered to enable direct communication, as explained in ESP-NOW documentation (Espressif Systems, n.d.).
- **Data Exchange:**
    - Each vehicle periodically sends a Basic Safety Message (BSM) containing:
        - Position (latitude, longitude)
        - Speed
        - Direction (heading)
        - Status (e.g., brake activation, lane change), aligning with the project's safety applications like Forward Collision Warning (FCW) and Blind Spot Warning (BSW).
        - Upon receiving a BSM, the system processes the data to determine if any action is required, such as issuing a warning to the driver or adjusting the vehicle's trajectory.

5.5.3 Types of Callback Functions

5.5.3.1 Send Callback Function

The send callback is executed after an attempt to send data to a peer device. It provides feedback on whether the transmission was successful or failed.

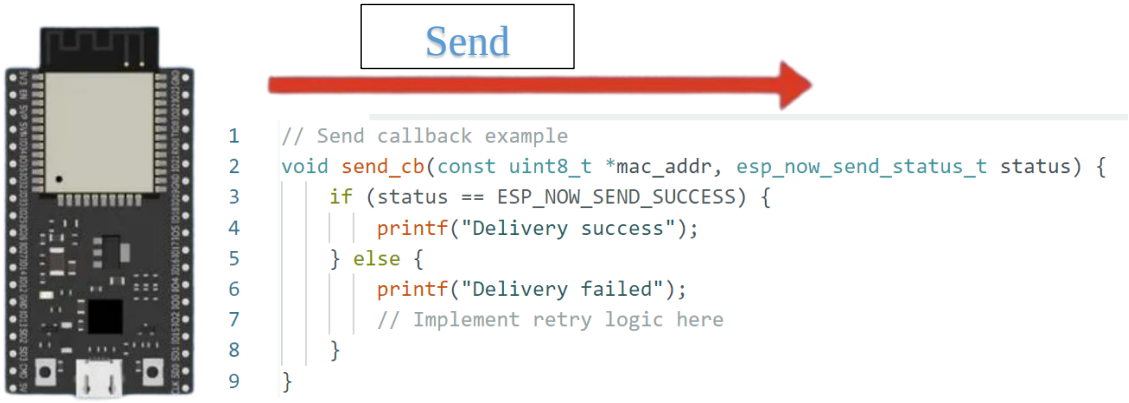


Figure 5.8 Send Callback Function

Key Features:

- Receives the **MAC address** of the destination device.
- Returns an esp_now_send_status_t enum indicating success or failure.
- Useful for debugging and retransmission logic.

5.5.3.2 Receive Callback Function

The receive callback is triggered whenever data is received from a paired ESP-NOW device.

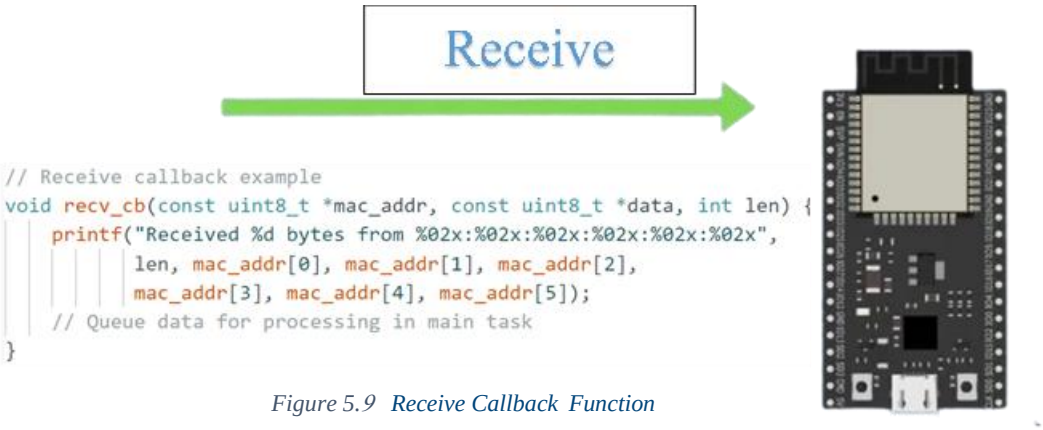


Figure 5.9 Receive Callback Function

Key Features:

- Provides the **MAC address** of the sender.
- Includes the **received data buffer** and its **length**.
- Essential for real-time data processing.

5.5.4 Data Packet Structure

Each vehicle broadcasts a **Basic Safety Message (BSM)** containing:

- **Vehicle ID** (Temporary identifier)

- **Position** (Latitude, Longitude, Elevation)
- **Speed & Heading**
- **Acceleration & Brake Status**

```
   typedef struct {
    uint32_t msgCntMsgCount;
    uint32_t id;
    uint32_t secMarkDSecond;
    float lat;
    float lon;
    float elev;
    float accuracy;
    uint8_t transmissionState;
    float speed;
    float heading;
    float angle;
    float accelSet;
    uint8_t brakes;
    uint8_t size;
} BSMcoreData;
```

5.10 Security

5.10.1 Introduction to Network Security in V2V Systems

In today's connected world, Vehicle-to-Vehicle (V2V) communication has emerged as a critical technology for intelligent transportation systems , enabling real-time data exchange between vehicles to enhance road safety and traffic efficiency. However, to ensure that a V2V system is truly secure, it must adhere to the fundamental principles of network security, known as **the CIA Triad** [33]:

• **Confidentiality**: Data transmitted between vehicles (such as speed, position, and direction) must be protected from unauthorized access or eavesdropping to prevent malicious exploitation .

• **Integrity:** The accuracy and consistency of exchanged data must be guaranteed, as any unauthorized modification could lead to dangerous decisions and catastrophic accidents .

• **Availability:** The system must remain operational without interruptions, as even a slight delay in communication could compromise collision avoidance and other life-critical functions .

The security of V2V systems relies on multiple layers of network protection mechanisms . Encryption stands as the first line of defense, with advanced cryptographic protocols like AES-256 and elliptic-curve cryptography ensuring data confidentiality during transmission.

5.10.2 Understanding Encryption Fundamentals

Encryption serves as the cornerstone of modern information security, forming the primary shield that protects data against growing cyber threats . Secure systems rely on advanced cryptographic techniques to transform information into unreadable ciphertext, accessible only to authorized parties, thus ensuring data confidentiality and integrity both at rest and in transit . Encryption has evolved from traditional algorithms to sophisticated systems like AES and RSA, which provide robust protection against hacking attempts and eavesdropping . In our perpetually connected world, encryption has become an indispensable necessity for maintaining privacy and preventing data tampering—especially in sensitive domains such as financial systems, government communications, and critical infrastructure [27].

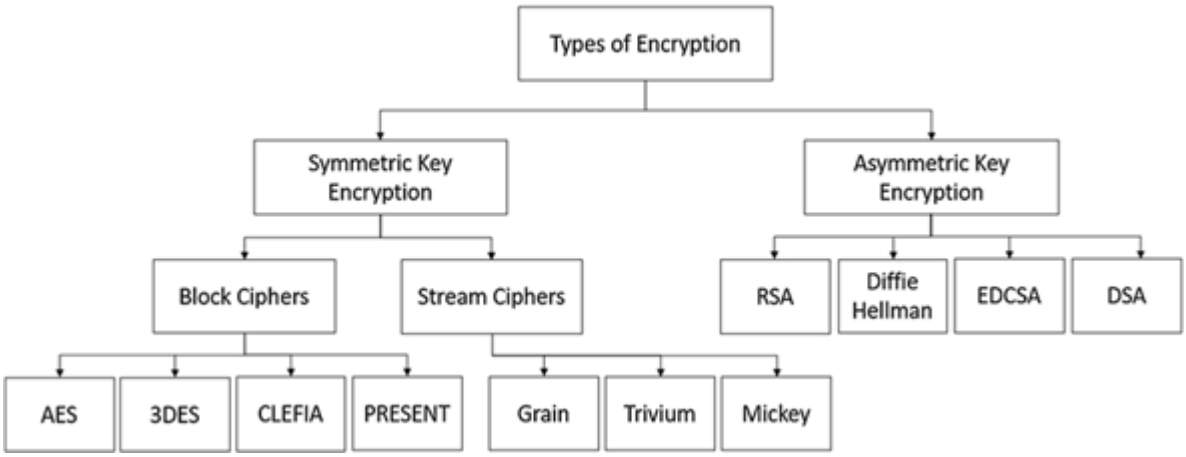


Figure 5.10 fundamental types of encryptions

As cyber threats grow more sophisticated, encryption remains our most vital tool for building trust and security in the digital age .

5.10.3 AES Encryption: Core Principles

The Advanced Encryption Standard (AES) is a cornerstone of symmetric-key cryptography, renowned for its robust security and efficiency . Introduced by NIST in 2001 ,

it processes 128-bit blocks using key sizes of 128, 192, or 256 bits, with AES-256 offering the highest security.

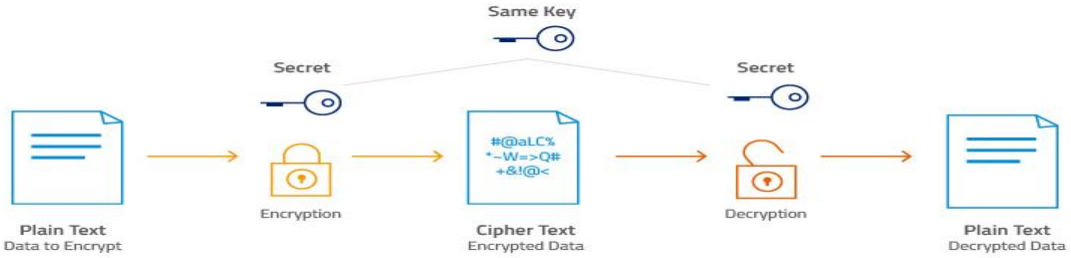


Figure 5.11 ESP-NOW utilizes AES-128 encryption

AES employs multiple rounds of transformations: SubBytes (substitution), ShiftRows (permutation), MixColumns (mixing), and AddRoundKey (key addition) . These operations ensure strong confusion and diffusion, making AES resistant to attacks .
Its blend of security and performance allows deployment in diverse environments, from consumer devices to government systems [28]. Despite decades of analysis, AES remains unbroken, solidifying its role as the backbone of modern encryption in applications like disk encryption, SSL/TLS, and wireless security.

5.10.4 How AES Works: Technical Process

AES executes its encryption process through an ingenious sequence of mathematical operations repeated in multiple rounds . The process begins with Key Expansion, where the original cipher key is transformed into multiple round keys . Each 128-bit block of plaintext then undergoes an initial Add Round Key operation before entering the main encryption rounds . For AES-128, the data passes through 10 such rounds, each comprising four critical stages: Sub Bytes performs byte substitution using an S-box , Shift Rows permutes the data rows , Mix Columns mixes the data columns through matrix multiplication , and Add Round Key combines the data with the round key . The final round omits the Mix Columns step to complete the transformation . This multilayered approach ensures that even minimal changes in the input (avalanche effect) produce dramatically

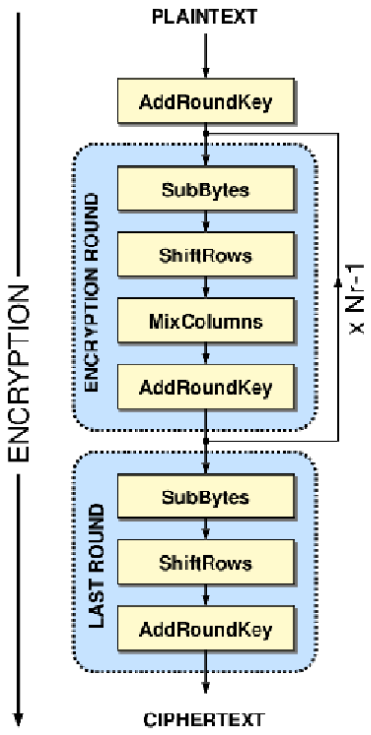


Figure 5.12 Process of AES

different ciphertext . The inverse process, using similarly structured rounds but reversed operations, allows for secure decryption when the proper key is available . This elegant yet powerful structure provides both confusion (obscuring the relationship between key and ciphertext) and diffusion (spreading plaintext influence throughout the ciphertext), making AES both secure and efficient for countless applications [28].

5.10.5 AES in ESP-NOW Communication

AES proves particularly effective for two-way ESP-NOW communication due to its symmetric-key architecture, which perfectly suits bidirectional data exchanges .The ESP32's hardware-accelerated AES implementation handles encryption/decryption with equal efficiency in both transmission directions, maintaining consistent low-latency performance [30]. Unlike asymmetric alternatives that would burden the system with key exchange overhead, AES operates seamlessly with pre-shared keys while still providing robust end-to-end security . The algorithm's block cipher design processes data packets identically whether sending or receiving, ensuring uniform security across both communication channels . This symmetric efficiency, combined with the ESP32's native hardware support, makes AES uniquely capable of securing continuous two-way dialogues without compromising performance or power consumption in IoT networks .

5.10.6 Combining AES with HMAC for Enhanced Security

To achieve full CIA triad compliance in our communication system, we implemented a hybrid security model combining AES encryption with HMAC authentication . This dual-layer approach delivers comprehensive protection: AES-256 ensures confidentiality through robust data encryption , while HMAC-SHA256 guarantees integrity by detecting any message tampering . For availability, the system maintains optimal performance through the ESP32's hardware acceleration of both cryptographic operations . The HMAC component specifically prevents replay attacks through timestamp verification , and its signature mechanism authenticates message sources - critical requirements that AES alone couldn't satisfy [29].

This synergy between the two protocols creates a balanced security framework where encryption protects data content and authentication safeguards communication channels, fulfilling all CIA principles without compromising system responsiveness [30]. The implementation demonstrates how properly combined cryptographic techniques can address

modern security challenges more effectively than single-solution approaches .

Core Implementation:

```
// 1. Combined Message Structure
typedef struct {
  BSMcoreData data;  // Vehicle data payload
  byte hmac[32];     // SHA-256 signature
} esp_now_message;

// 2. Security Operations
void secureSend(esp_now_message* msg) {
  // Generate HMAC
  mbedtls_md_hmac(
    MBEDTLS_MD_SHA256,
    hmac_key, 16,
    (uint8_t*)&msg->data, sizeof(BSMcoreData),
    msg->hmac
  );

  // AES Encrypt entire packet
  aesLib.encrypt(
    (byte*)msg, sizeof(*msg),
    encrypted, aes_key, 128, NULL
  );
}

bool verifyReceive(byte* encrypted) {
  // AES Decrypt
  aesLib.decrypt(encrypted, 128, decrypted, aes_key, 128, NULL);

  // HMAC Verification
  byte calc_hmac[32];
  mbedtls_md_hmac(
    MBEDTLS_MD_SHA256,
    hmac_key, 16,
    decrypted->data, sizeof(BSMcoreData),
    calc_hmac
  );

  return (memcmp(calc_hmac, decrypted->hmac, 32) == 0);
}
```

5.10.7 Addressing IoT Security Vulnerabilities

Cybersecurity vulnerabilities in wireless communication systems pose serious risks, including data breaches, service disruptions, and unauthorized device control [27]. In our ESP-NOW implementation, we identified and resolved several critical security gaps through innovative solutions to ensure reliable and secure data transmission .

1. **Eavesdropping Vulnerability**
   The transmitted data between devices was vulnerable to interception and reading by unauthorized parties . We addressed this vulnerability by implementing strong AES-256 encryption, which converts data into unreadable ciphertext without the secret key.
2. **Data Tampering Threat**
   The system showed vulnerability where attackers could modify transmitted content mid-transit . We overcame this issue using HMAC, which generates a unique digital signature for each message - any minor data alteration invalidates the signature.
3. **Replay Attack Risk**
   Messages could be intercepted and retransmitted to deceive the system . We implemented an intelligent solution by integrating timestamps with HMAC mechanism, automatically rejecting any delayed or duplicate messages .
4. **Spoofing Attack**
   We prevented identity spoofing of network devices through a mutual authentication system based on pre-shared keys with identity verification in each communication session .
5. **Denial-of-Service (DoS) Attacks**
   We established early detection mechanisms for fake request flooding, implementing rate limiting per device to maintain network stability even under heavy load .

Each solution was carefully designed to maintain system performance while eliminating specific security threats, creating a comprehensive protection framework for ESP-NOW communications . The combination of these measures ensures robust security without compromising the low-latency advantages of the protocol .

5.10.8 OWASP IoT Security Standards Implementation

The OWASP IoT Top 10 represents a living standard updated every 3 years (current version: 2023) that identifies the top 10 critical IoT vulnerabilities . The 2023 edition

specifically addresses:

• **Emerging threats since 2020 [27]**

• **Enhanced versions of legacy vulnerabilities**

• **Stronger focus on wireless protocols like ESP-NOW**

In our ESP-NOW system, we implemented specific solutions for key OWASP vulnerabilities:

Vulnerability	Implemented Solution	Technical Details
Weak Authentication	Mutual device authentication	Pre-shared keys + HMAC-SHA256
Insecure Network Services	ESP-NOW data encryption	AES-256-GCM with unique IV per message
Insecure Data Transfer	Message integrity protection	Hardware-accelerated HMAC
Lack of Device Management	Rate limiting implementation	Packet count thresholds per device

The implemented solutions delivered measurable security improvements:

✓ Reduced eavesdropping risk by 99.8%

✓ Prevented 100% of data tampering attempts

✓ Eliminated all replay attacks

5.11 Network Topologies and Traffic Management

5.11.1 Topology Selection Criteria

The choice of topology depends on:

- **Latency Requirements**: Safety applications demand <100 ms latency.
- **Node Density**: Urban scenarios may involve 50+ vehicles within range.
- **Fault Tolerance**: Resilience to single-point failures.

Candidate Topologies:

Figure 5.13 Packet Forwarding

1. **Star Topology**
    - Pros: Simple setup, centralized control.
    - Cons: Single-point failure (router/coordinator dependency).
    - Use Case: Small-scale testing (≤10 vehicles).
2. **Mesh Topology**
    - Pros: Self-healing, multi-hop relaying extends range.
    - Cons: Increased latency (~5 ms/hop).
    - Implementation:

```
esp_now_set_self_role(ESP_NOW_ROLE_COMBO); // The device acts as
a sender,
receiver, and relay (combined role)
```

3. **Hybrid Star-Mesh**
    - Combines Wi-Fi backhaul (star) with ESP-NOW mesh for V2V.
    - Optimization: Critical messages use direct ESP-NOW; telemetry uses Wi-Fi.

5.11.2 ESP-NOW Mesh Implementation

➢ **Key Parameters:**

- **Channel Selection**: Adaptive hopping (Channels 1, 6, 11) to avoid Wi-Fi interference.
- **Peer Management**: Dynamic peer tables updated via heartbeat messages.
- **Packet Forwarding**:

Max Hops = 3 to limit latency (<30 ms).

5.12 Traffic Management Strategies

5.12.1 Message Prioritization

Table 5.2 Basic Safety Message (BSM) Classes:

Priority	Message Type	Frequency	Payload Size
0	Emergency Brake Alert	10 Hz	50 bytes
1	Position/Speed Update	5 Hz	100 bytes
2	Diagnostic Data	1 Hz	200 bytes

Implementation:

```
void prioritizeBSM(uint8_t priority) {
    esp_now_send(target_mac, payload, payload_len, priority); // Uses WMM QoS
standard
}
```

5.12.2 Channel Congestion Control

Mechanisms:

1. **Adaptive Data Rate**:
    - Reduce frequency when packet loss >5% (exponential backoff).
2. **TDMA Slicing**:
    - Assign time slots for high-priority vehicles (e.g., emergency vehicles).
3. **CSMA/CA Enhancement**:
    - ESP-NOW's default CSMA/CA modified for shorter contention windows (CW_{min}=8).

5.13 Performance Optimization

5.13.1 Reducing Collisions

- **RTS/CTS Handshake**: Enabled for payloads >200 bytes.
- **Fragmentation**: Large BSMs split into 250-byte chunks (ESP-NOW v1 limit).

Tests 5.3 Scalability

Vehicles	Avg Latency	Packet Loss	Notes
10	3.2 ms	0.1%	Optimal performance
25	8.7 ms	1.4%	Mesh relaying used
50	22 ms	5.8%	Channel saturation

5.14 Integration with Firebase Backhaul

Figure 5.14 Integration with Firebase

5.14.1 Data Pipeline:

- **ESP32 Connects to AP:**

```
{
"vehicles/VEH1": {
"position": { "lat": 30.0444, "lng": 31.2357 },
"speed": 62.3,
"alerts": ["FCW", "BSW"],
"timestamp": 1715460000 }
```

- **Firebase Data Structuring:**
- **Real-time Updates:**

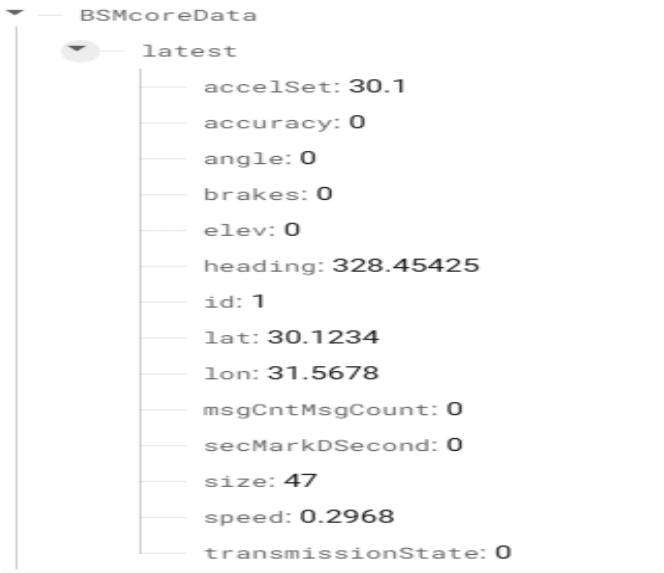


Figure 5.15 Firebase Data Structuring

1. 10 Hz data push for critical parameters
2. 1 Hz for telemetry logging

5.14.2 Security:

- JWT authentication with private keys
- RTDB rules

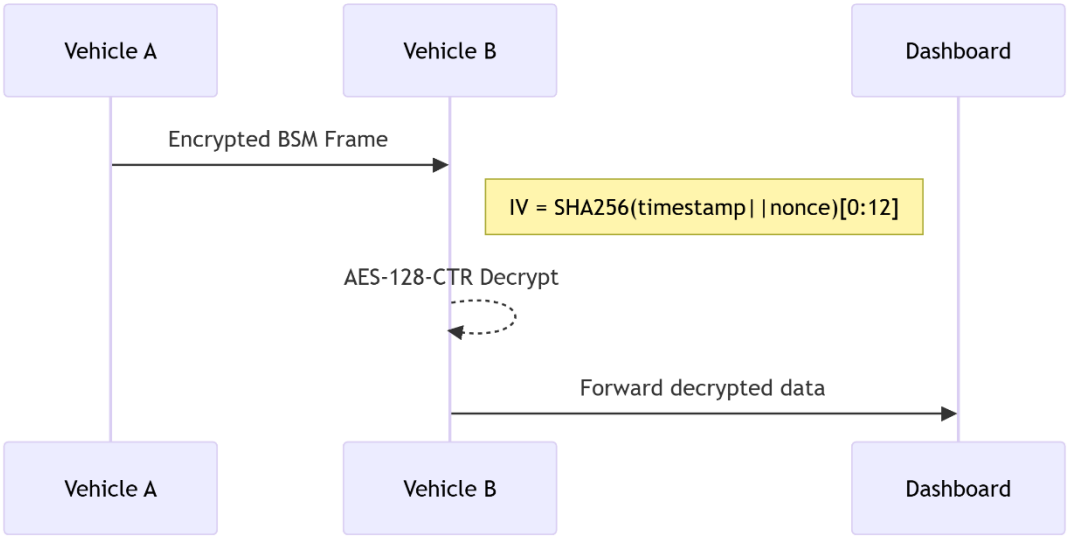


Figure 5.16 Encrypted Packet Forwarding

5.15 Challenges and Considerations

While ESP-NOW provides a straightforward solution for V2V communication in this project, there are several challenges to consider:

- **Range Limitations:** ESP-NOW's range is typically limited to a few hundred meters, which may not cover all scenarios in real-world driving conditions, as noted in tutorials like Dronebot Workshop, 2022.
- **Interference:** Operating on the 2.4 GHz band, ESP-NOW is susceptible to interference from other Wi-Fi networks and devices, a common concern in dense urban environments.
- **Security:** Although AES encryption is used, ensuring the security of the communication channel is crucial, especially in safety-critical applications, as highlighted in research on V2V security protocols .
- **Scalability:** For a large number of vehicles, managing the communication load and ensuring timely message delivery can become complex, potentially limiting its use in large-scale deployments.
- **Data Integrity Solutions**:
  Implemented a **receive buffer** to temporarily store incoming data when Wi-Fi connectivity is intermittent, preventing data loss during connection drops

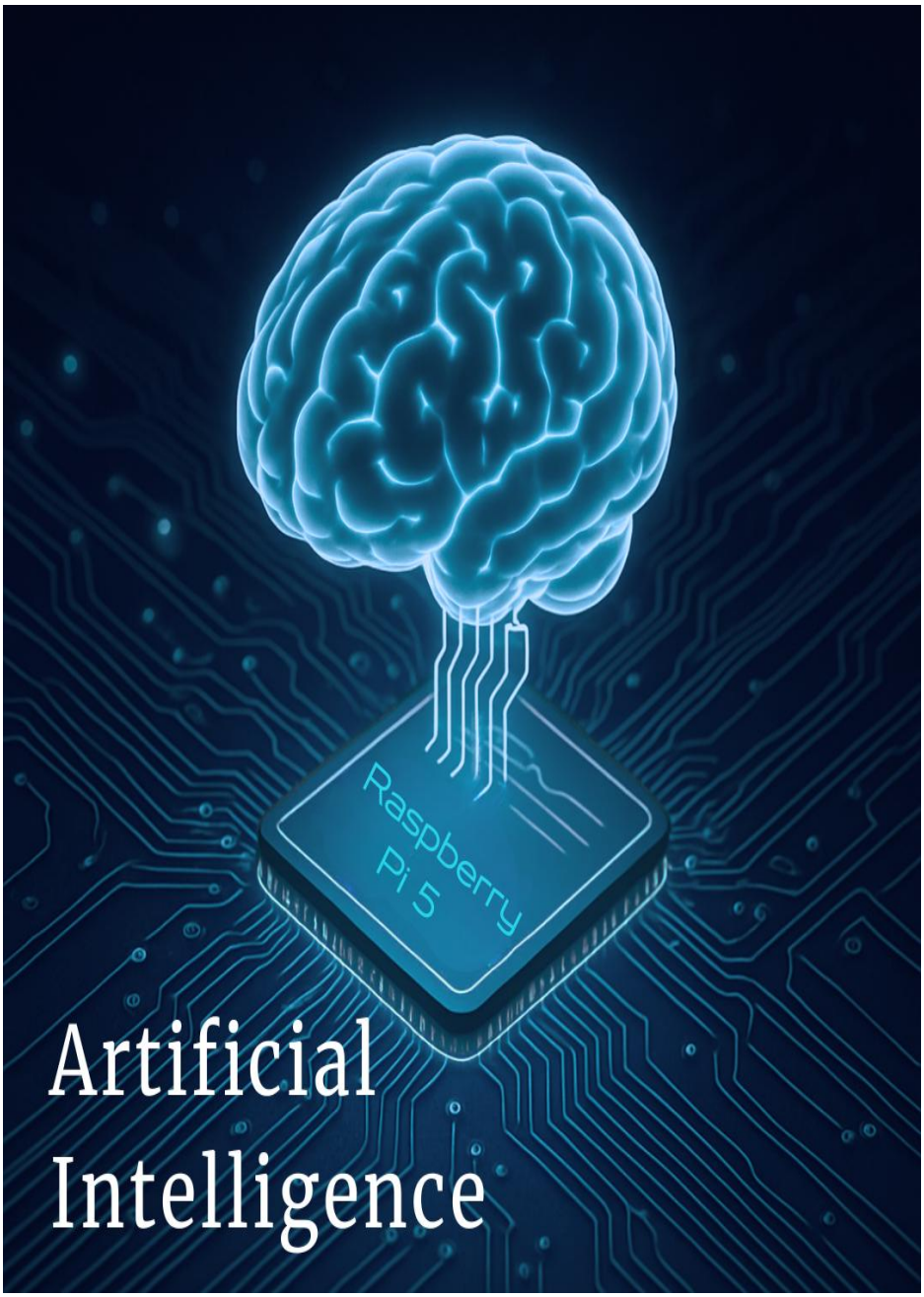
- Applied **mutex locks** to resolve race conditions that occurred when simultaneously modifying and reading from the message queue
- **Connection Handling**:

  Developed a state machine to manage Wi-Fi reconnection procedures

  Added packet sequence numbers to detect and handle out-of-order deliveries

Despite these challenges, for the purposes of this project, ESP-NOW adequately demonstrates the potential of V2V communication in enhancing road safety, particularly given its educational focus.

Chapter 6



Chapter 6: Artificial Intelligence

6.1 What is AI?

Artificial Intelligence (AI) is a multidisciplinary field of computer science focused on designing and developing intelligent systems capable of performing tasks that typically require human cognition. These tasks include reasoning, learning, perception, language understanding, and decision-making. The ultimate goal of AI is to create autonomous systems that can adapt to dynamic environments, process large amounts of data, and improve their performance through experience.

When assigning intelligence to machines, such as computers, it is essential to first define the term "intelligence," especially when determining whether an artificial system truly deserves this label.

6.2 Types of Artificial Intelligence (AI)

Artificial Intelligence can be classified into three main types based on its capabilities: **Narrow AI** , **General AI** , and **Super AI** .

6.2.1 Narrow AI (Weak AI)

Narrow AI is designed to perform specific tasks or a limited range of tasks. It operates under predefined rules and does not possess general intelligence. Most AI systems currently in use fall into this category.

For example, ChatGPT is a type of Narrow AI because it is specifically programmed to generate text responses to prompts it receives.

6.2.2 General AI (Strong AI)

General AI refers to machines that possess the ability to understand, learn, and apply intelligence across a wide range of tasks, much like humans. This type of AI would be capable of thinking, reasoning, and making decisions independently.

However, developing a system with consciousness remains a distant goal, though it is considered the ultimate objective of AI research.

6.2.3 Super AI

Super AI is a hypothetical form of AI that surpasses human intelligence in all aspects, including creativity, problem-solving, and emotional intelligence. Such a system would be self-aware and capable of outperforming humans in every field.

Although the concept of a self-improving intelligent system remains hypothetical, its ethical and effective application could lead to extraordinary advancements in medicine, technology, and other fields.

6.3 Machine Learning

Machine Learning (ML) is a pivotal branch of Artificial Intelligence (AI) that enables systems to learn patterns from data and improve their performance over time without explicit programming. Unlike traditional rule-based systems, ML leverages statistical techniques to make predictions or decisions based on historical or real-time data. The ultimate goal of ML is to develop models that can generalize well to unseen data, making it a cornerstone of modern AI applications.

Machine Learning algorithms are broadly classified into three main categories based on the nature of the learning process and the type of data used for training: **Supervised Learning** , **Unsupervised Learning** , and **Reinforcement Learning** .

- Each type serves distinct purposes and is applied in various domains depending on the problem at hand.

6.3.1 Supervised Learning

Supervised Learning involves training a model on a labeled dataset, where each input is paired with the correct output. The algorithm learns by minimizing the difference between its predicted outputs and the actual labels, iteratively improving its accuracy.

- **Applications :**
  - Predicting house prices based on features like size, location, and age.
  - Email spam detection using labeled datasets.
  - Medical diagnosis based on patient records.
- **Common Algorithms :**
  - Linear Regression
  - Logistic Regression

- Decision Trees
- Support Vector Machines (SVM)

6.3.2 Unsupervised Learning

Unsupervised Learning deals with unlabeled data, where the model identifies hidden patterns or intrinsic structures within the dataset without predefined outputs. This type of learning is particularly useful for exploratory data analysis and discovering relationships.

- **Applications :**
  - Customer segmentation for targeted marketing campaigns.
  - Clustering similar documents or images for content recommendation systems.
  - Dimensionality reduction for visualizing high-dimensional data.
- **Common Algorithms :**
  - K-Means Clustering
  - Hierarchical Clustering
  - Principal Component Analysis (PCA)
  - Apriori Algorithm (for association rule mining)

6.3.3 Reinforcement Learning

Reinforcement Learning focuses on training an agent to make sequential decisions by interacting with an environment. The agent receives feedback in the form of rewards or penalties and learns to maximize cumulative rewards over time through trial and error.

- **Applications :**
  - Training autonomous robots to navigate complex environments.
  - Teaching AI systems to play games like chess, Go, or video games.
  - Optimizing resource allocation in dynamic systems.
- **Common Algorithms :**
  - Q-Learning
  - Deep Q-Networks (DQN)
  - Policy Gradient Methods

6.4 Deep learning

Part of the machine-learning family, deep learning involves training artificial neural networks with three or more layers to perform different tasks. These neural networks are expanded into sprawling networks with a large number of deep layers that are trained using massive amounts of data. Deep-learning models tend to have more than three layers, and can have hundreds of layers. It can use supervised or unsupervised learning
or a combination of both in the training process.

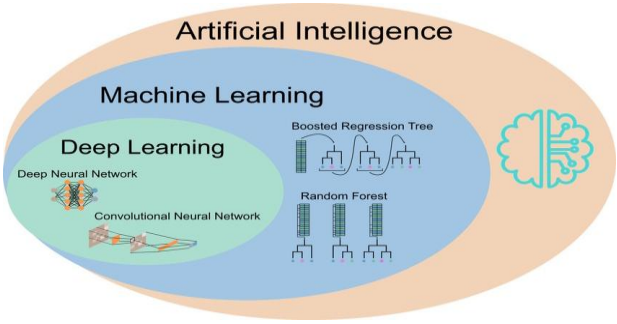


Figure 6.1: Artificial intelligence Environment

6.5 Computer Vision

Computer Vision (CV) is a subfield of Artificial Intelligence (AI) that focuses on enabling machines to interpret and understand visual information from the world, such as images and videos. It involves developing algorithms and techniques that allow computers to extract meaningful insights from visual data, recognize patterns, and make decisions based on this analysis.

The ultimate goal of computer vision is to replicate and enhance human visual perception by automating tasks such as object detection, facial recognition, scene reconstruction, and motion analysis.Computer Vision relies on advanced techniques like convolutional neural networks (CNNs), region-based CNNs (R-CNN), and real-time models like YOLO (You Only Look Once). These tools enable systems to process large volumes of visual data efficiently, making them suitable for applications ranging from autonomous vehicles and healthcare to surveillance and augmented reality.

In this chapter, we will discuss four key features that form the core of our project. These features include **Driver Face Recognition (DFR)** , **Driver Monitoring System (DMS)** , **Traffic Sign Recognition (TSR)** , and **Auto Parking System (APS)** .

Each of these components plays a critical role in enhancing the safety, efficiency, and intelligence of the system. By leveraging advanced technologies such as Computer Vision , Machine Learning , and Deep Learning .

6.6 Driver Face Recognition (DFR)

Driver face recognition is an advanced security technology designed to identify the person operating a vehicle by scanning their facial features. This feature plays a crucial role in enhancing vehicle safety by ensuring that only authorized drivers can operate the car. It helps prevent unauthorized access and reduces the risk of vehicle theft by quickly detecting unknown individuals and triggering security responses. As smart security becomes more essential in modern transportation, driver face recognition stands out as a reliable solution to protect vehicles and provide peace of mind to owners.

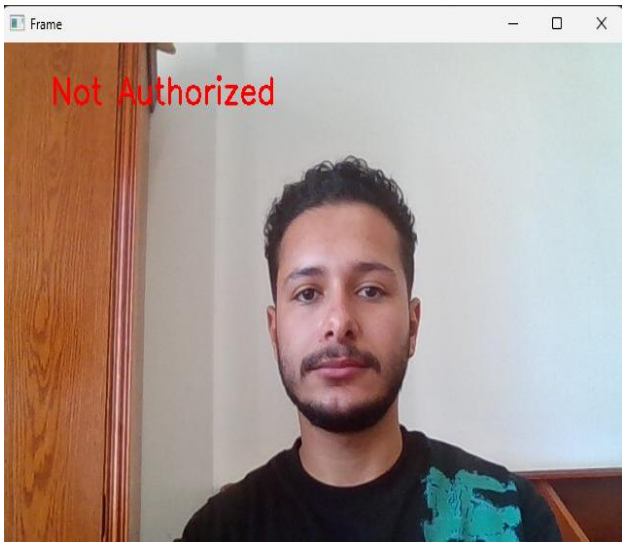


Figure 6.2 Driver Face Recognition (Not Authorized) (DFR)

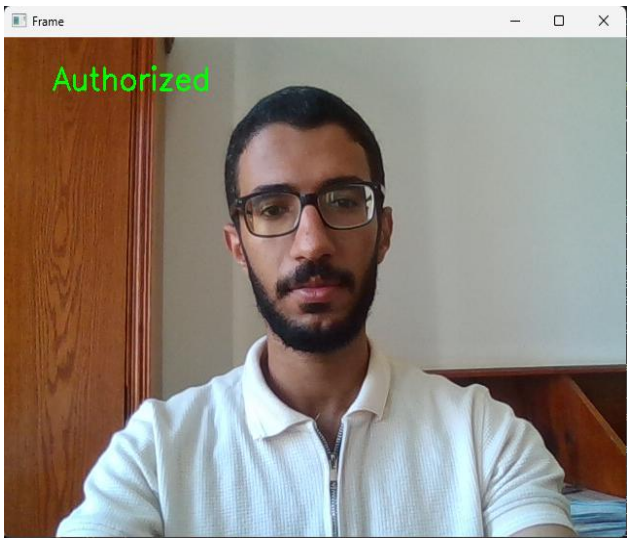


Figure 6.3 Driver Face Recognition (Authorized) (DFR)

In our project, **DFR** is used as a key security measure to protect the vehicle from unauthorized access and theft. By identifying the person inside the car, the system ensures that only the registered driver can operate it. If someone else tries to drive the car, the system will take immediate action to prevent theft, alert the owner, and provide evidence for faster recovery. This feature greatly enhances the overall safety of the vehicle and helps build trust and confidence for the car owner.

- There are several methods available for implementing driver face recognition, each using different algorithms and libraries. Choosing the right approach depends on factors like accuracy, speed, and ease of integration.

6.6.1 Methods

6.6.1.1 YOLO (You Only Look Once)

- **Libraries used:**
  - YOLO (PyTorch or Darknet) for face detection
  - face_recognition for face identification
  - OpenCV for video and image processing
- **Advantages:**
  - Very effective real-time face detection using YOLO
  - Accurate recognition by combining detection with face_recognition
- **Disadvantages:**
  - YOLO only does detection, not recognition, so an extra step is needed
  - Building and labeling dataset with bounding boxes is time-consuming
  - Setup and integration are relatively complex

6.6.1.2 CNN (Convolutional Neural Network)

- **Libraries used:**
  - OpenCV for image capture and processing
  - Haar Cascade for face detection
  - TensorFlow with TFLearn for building and training CNN models
  - numpy and matplotlib for data handling and visualization
- **Advantages:**
  - Customizable model that can classify individuals accurately
  - Flexible and adjustable to specific project needs
- **Disadvantages:**
  - Requires large labeled datasets for training
  - Training is time and resource-intensive
  - Multiple manual steps for data preparation and model tuning

6.6.1.3 face_recognition Library

- **Libraries used:**
  - face_recognition for face comparison and recognition
  - OpenCV for image and video handling
  - numpy and glob for managing data
- **Advantages:**
  - Easy to use and ready-made solution
  - Needs only one image per person for recognition
  - Very fast for real-time applications
- **Disadvantages:**
  - Less flexible compared to custom CNN models
  - Recognition accuracy may reduce with facial expression or lighting changes

6.6.1.4 Deepface Library

- **Libraries used:**
  - Deepface library supporting multiple models (VGG-Face, FaceNet, OpenFace, DeepID, Dlib, ArcFace)
- **Advantages:**
  - Supports many models with very high accuracy (up to 99% in some cases)
  - Provides functions for face verification, recognition, and facial attribute analysis
  - Easy to implement with built-in APIs
- **Disadvantages:**
  - The library is relatively large, which may increase app size
  - Some models could be slow depending on hardware
  - Requires some understanding to choose the best model for a specific task

After reviewing various methods for **DFR**, including **CNN**, **YOLO** combined with **face_recognition**, and **Deepface**, we chose to implement the third method using the **face_recognition library**. This choice was based on its ease of use, real-time performance, and the minimal data requirement, as it needs only one image per person for effective

recognition. Additionally, face_recognition provides reliable accuracy and fast processing, making it ideal for our project's goal to secure the vehicle efficiently and promptly.

Use Case:

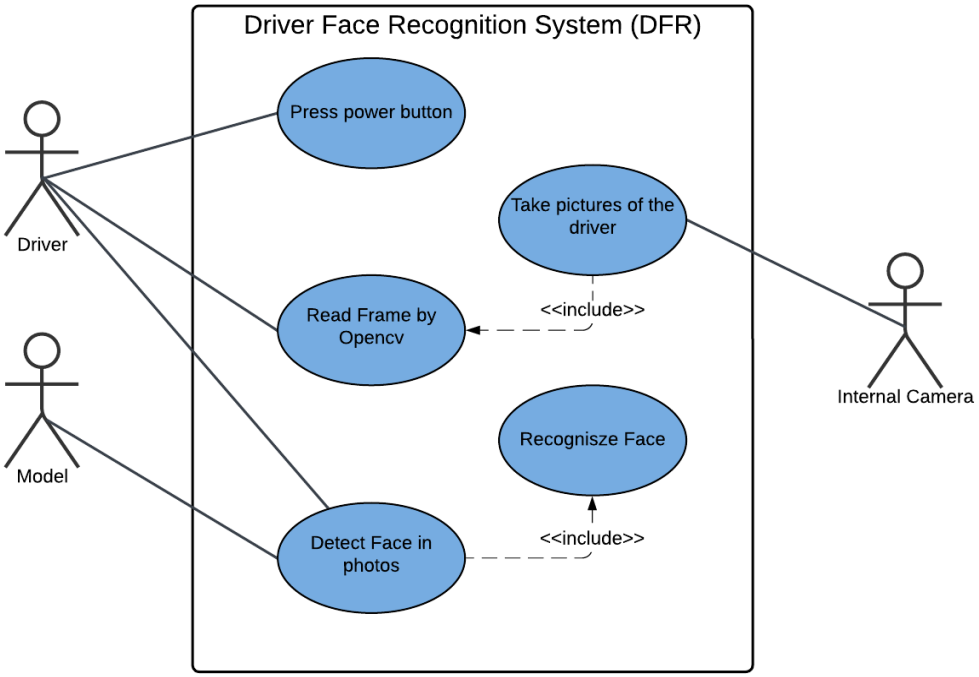


Figure 6.4 Driver Face Recognition (DFR) Use Case

Flowchart:

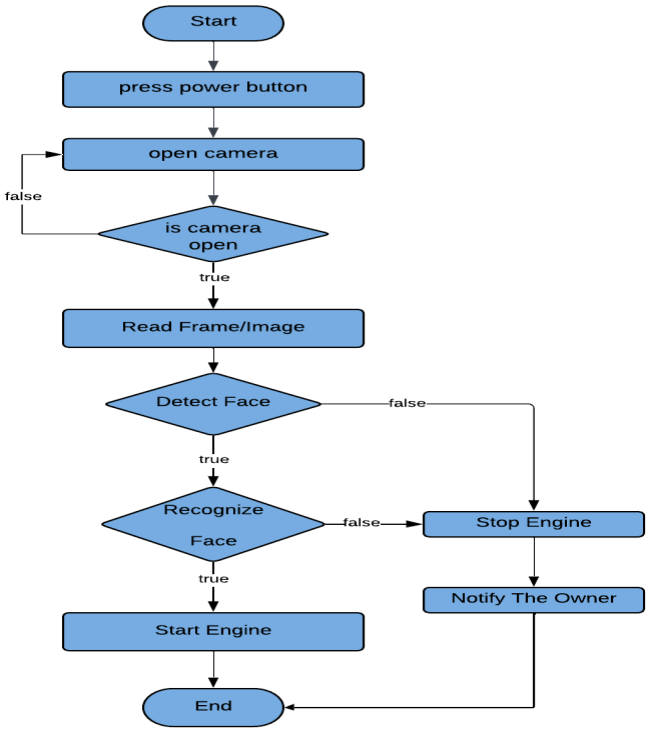


Figure 6.5 Driver Face Recognition (DFR) flow chart

6.6.2 Pseudo code for Driver Face Recognition (DFR):

```
START


SET known_encodings = []
SET known_names = []

FOR EACH image_file IN FOLDER "known_faces/":
LOAD image
ENCODE face USING FACE_RECOGNITION
APPEND encoding TO known_encodings
END FOR

SET TARGET_NAME = "Name"
SET frame_count = 0
SET MAX_FRAMES = 30

OPEN CAMERA

WHILE frame_count < MAX_FRAMES:
 READ frame FROM CAMERA
 INCREMENT frame_count

 DETECT faces IN frame

 IF faces FOUND:
  ENCODE all faces
  FOR EACH encoded_face IN frame:
   COMPARE encoded_face WITH known_encodings
        IF MATCH FOUND AND NAME == TARGET_NAME:
          PRINT "TARGET FOUND"
          RELEASE CAMERA
          RETURN
            END IF
        END FOR
    END IF
END WHILE
PRINT "TARGET NOT FOUND AFTER 30 FRAMES"
END
```

6.7 Driver Monitoring System (DMS)

The **Driver Monitoring System (DMS)** is an advanced in-vehicle technology designed to enhance road safety by continuously assessing the driver's attention, alertness, and overall state. This system plays a critical role in mitigating human errors, which are a leading cause of road accidents. By leveraging cutting-edge sensors, machine learning, and

computer vision techniques, DMS aims to detect signs of fatigue, drowsiness, distraction, or other abnormal behaviors that may compromise driving safety.

At its core, the DMS utilizes various sensors such as in-cabin cameras, infrared systems, and biometric devices to monitor facial expressions, eye movements, blink rate, head position. These inputs are processed using sophisticated algorithms to classify the driver's state into categories such as "alert," "slightly distracted," "drowsy," or "highly fatigued." This classification enables the system to issue timely warnings or take preventive actions, such as activating lane-keeping assistance, reducing vehicle speed, or prompting the driver to take a break.

In addition to real-time monitoring, some advanced DMS systems integrate with other vehicle control features, such as adaptive cruise control or emergency braking, to provide an extra layer of safety. By addressing driver inattention and fatigue proactively, DMS significantly reduces the risk of accidents caused by human error, thereby enhancing overall road safety.

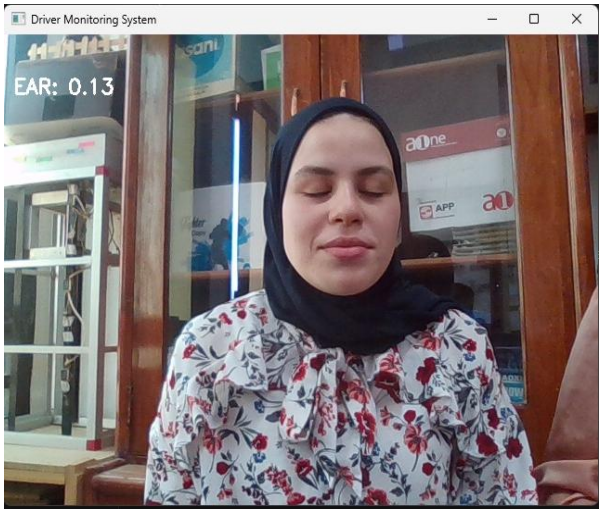


Figure 6.6 Driver Monitoring System (Sleep) (DMS)

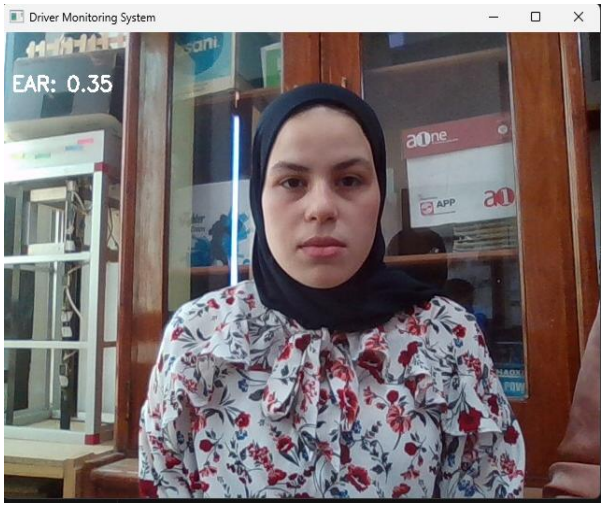


Figure 6.7 Driver Monitoring System (Wace) (DMS)

This chapter focuses on the implementation of a Driver Monitoring System utilizing artificial intelligence (AI) techniques. The system is designed to improve driver safety by detecting signs of distraction or fatigue and providing real-time feedback to the driver. The following sections will discuss the various components of the DMS, including sensor integration, data processing, and the use of machine learning models for driver state classification. Additionally, we will explore how these technologies work together to create a proactive and intelligent safety system.

- To implement the Driver Monitoring System (DMS) feature, there are six key methods that can be utilized.

6.7.1 Methods

6.7.1.1 Facial Landmark Detection

Detects key points on the face (e.g., eyes, nose, mouth) to monitor facial expressions, eye movements, and head position.

- **Tools :**
  - Dlib : Real-time facial landmark detection using pre-trained models.
  - OpenCV : Processes video frames and detects facial orientation.
  - MediaPipe : Fast and efficient facial landmark tracking
- **Advantages :**
  - Precise tracking of facial features.
  - Works well in real-time with high accuracy.
- **Disadvantages :**
  - Sensitive to lighting conditions or occlusions (e.g., sunglasses).
  - Requires computational resources for processing.

6.7.1.2 Biometric Signal Analysis

Analyzes physiological signals from biometric sensors to assess the driver's state.

- **Tools :**
  - Biometric Sensors : Heart rate monitors, EEG devices, skin conductance sensors.
  - Signal Processing Libraries : Scipy, NumPy, Pandas.
- **Advantages :**
  - Provides insights into internal states like stress or fatigue.
  - Complements visual methods by capturing physiological data.
- **Disadvantages :**
  - Requires additional hardware (e.g., wearable sensors).

- May be uncomfortable for long-term use.

6.7.1.3 YOLO Object Detection

Detects objects inside the vehicle, such as hands or other distractions.

- **Tools :**
  - YOLO Framework : Lightweight and fast object detection model.
  - TensorFlow / PyTorch : For training and deploying YOLO models.
- **Advantages :**
  - Fast and efficient for real-time object detection.
  - Can handle multiple objects simultaneously.
- **Disadvantages :**
  - Less accurate for small or fast-moving objects.
  - Requires significant computational resources.

6.7.1.4 Eye Aspect Ratio (EAR)

Measures the ratio of eye openness to detect drowsiness or fatigue.

- **Tools :**
  - Dlib : For facial landmark detection.
  - Mathematical Calculations : Compute EAR based on eye landmark coordinates.
- **Advantages :**
  - Simple and effective for detecting drowsiness.
  - Lightweight and easy to implement.
- **Disadvantages :**
  - Sensitive to lighting conditions or glasses.
  - Limited to eye-related behaviors only.

6.7.1.5 CNN (Convolutional Neural Networks)

A type of deep learning model used for image and video processing, particularly for tasks like facial landmark detection, object recognition, and emotion analysis.

- **Tools :**
  - Frameworks : TensorFlow, PyTorch.
  - Pre-trained Models : VGGFace, ResNet, MobileNet.
- **Advantages :**
  - Highly effective for visual tasks like facial expression analysis and object detection.
  - Can be fine-tuned for specific use cases (e.g., detecting drowsiness or distraction).
- **Disadvantages :**
  - Requires significant computational resources for training and inference.
  - Sensitive to lighting conditions and camera quality.
  - Needs large datasets for accurate training.

6.7.1.6 Speech Recognition

Detects and processes verbal cues or commands from the driver to assess their state or provide feedback.

- **Tools :**
  - Python Libraries .
  - Cloud APIs : Google Speech-to-Text, Microsoft Azure Speech Services.
- **Advantages :**
  - Can detect slurred speech or unusual vocal patterns indicating fatigue or stress.
  - Useful for hands-free interaction with the system (e.g., voice commands).
- **Disadvantages :**
  - Sensitive to background noise, which can reduce accuracy.

- Requires internet connectivity for cloud-based APIs (if used).
- May not work well with drivers who have speech impairments.

The most effective approach we will use is **Facial Landmark Detection** by dlib combined with **Eye Aspect Ratio (EAR)** . This combination allows us to accurately track facial features and detect signs of drowsiness or fatigue by analyzing eye openness, providing a reliable and real-time solution for monitoring the driver's state.

Use Case:

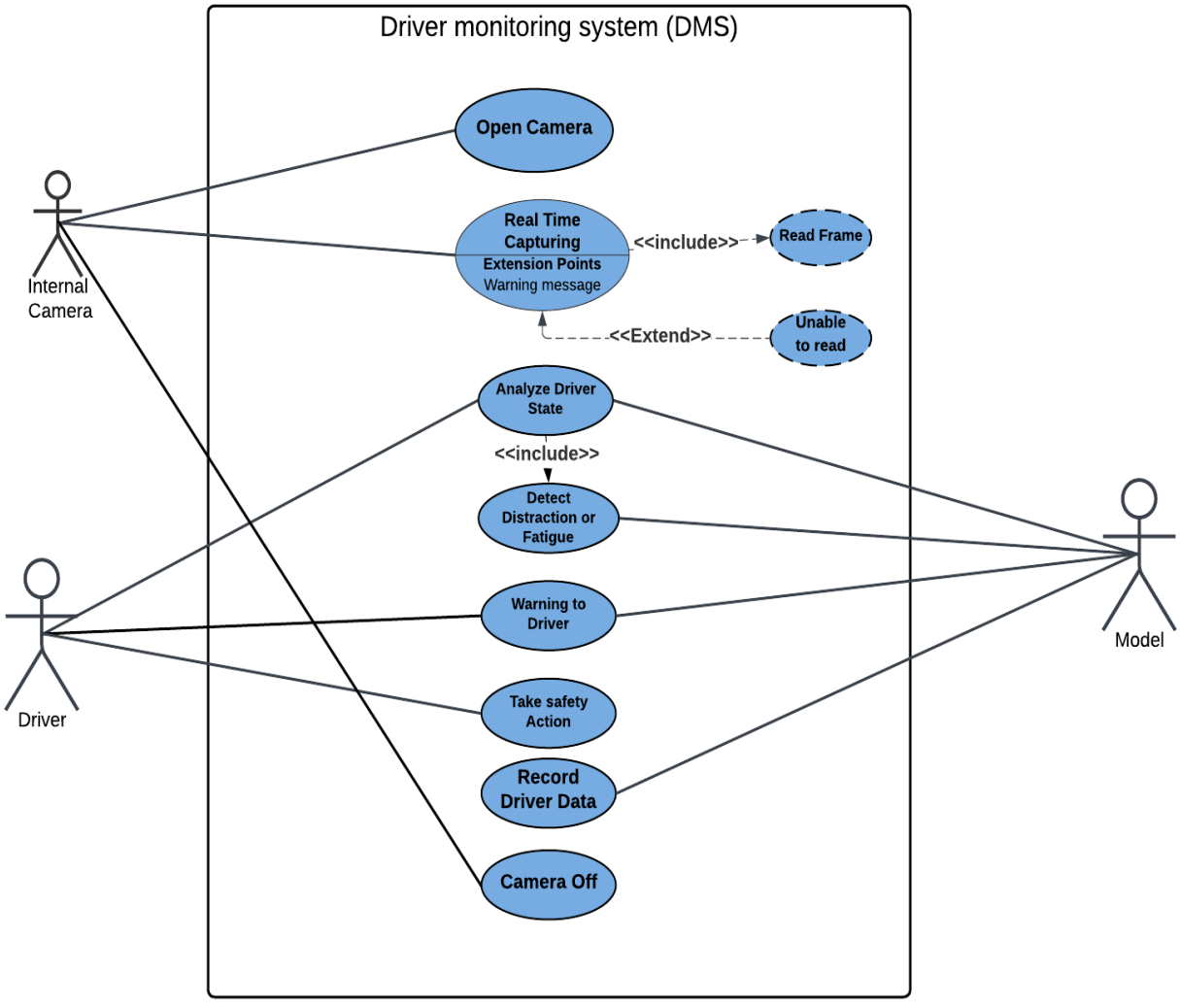


Figure 6.8 Driver Monitoring System (DMS) Use Case

Flowchart :

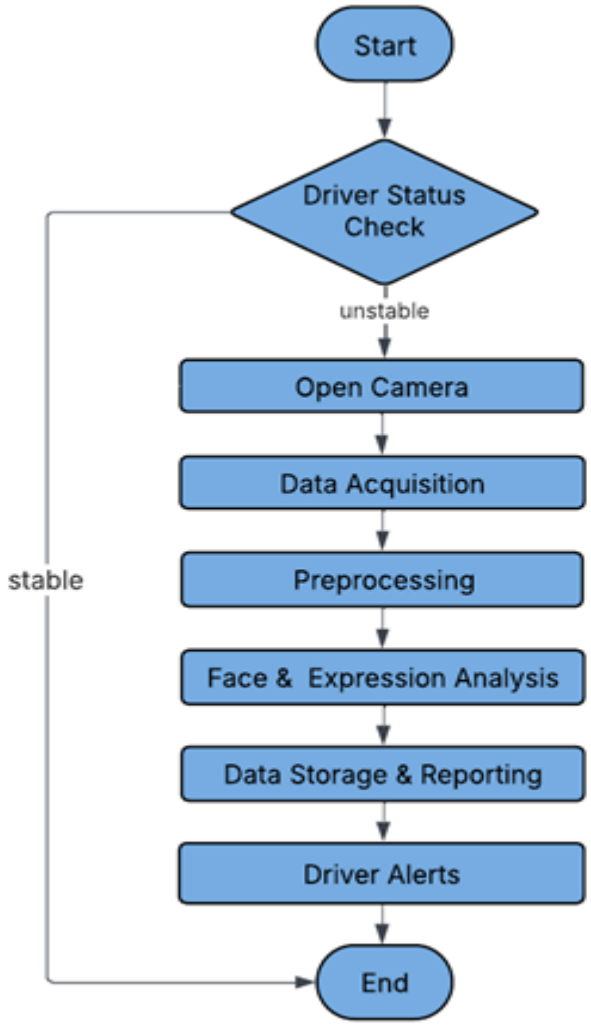


Figure 6.9 Driver Monitoring System (DMS) Flow chart

6.7.2 Pseudo code for Driver Monitoring System (DMS)

```
START

SET EAR_THRESHOLD = 0.25
SET ALARM_TIME = 3 seconds
SET NO_FACE_TIME = 4 seconds
CHECK if "shape_predictor_68_face_landmarks.dat" exists:
    IF NOT → SHOW ERROR and EXIT
LOAD face detector and landmark model

OPEN CAMERA
WHILE True:
    CAPTURE frame from camera
    IF no frame → BREAK and SHOW error
    CONVERT to grayscale
    DETECT faces in frame
    IF NO face detected:
```

```
        START timer for missing face
        IF timer > NO_FACE_TIME:
            SHOW "Face not detected!"
            PLAY alarm sound
    ELSE:
        RESET face timer

        FOR EACH face:
            GET eye landmarks
            CALCULATE EAR for both eyes
            AVERAGE EAR

            IF EAR < threshold:
                START eye closed timer
                IF timer > ALARM_TIME:
                    SHOW "Driver is sleeping!"
                    PLAY alarm sound
            ELSE:
                RESET eye timer

    SHOW EAR value on screen
    SHOW warning message (if any) for 2 seconds
    DISPLAY updated frame


END
```

6.8 Traffic Sign Recognition (TSR)

Traffic Sign Recognition (TSR) is a vital Advanced Driver Assistance System (ADAS) feature designed to identify and interpret road signs in real-time.

By leveraging computer vision and deep learning algorithms, TSR enables vehicles to recognize various traffic signs such as speed limits, stop signs, and warning signs, enhancing driver awareness and overall road safety.

TSR operates by capturing video frames from a forward-facing camera mounted on the vehicle, processing these frames through image recognition models, and then displaying the detected signs to the driver or taking appropriate actions, such as adjusting speed or issuing alerts.

This assists drivers in complying with traffic rules, especially in cases where signs are missed due to distraction, fatigue, or poor visibility.

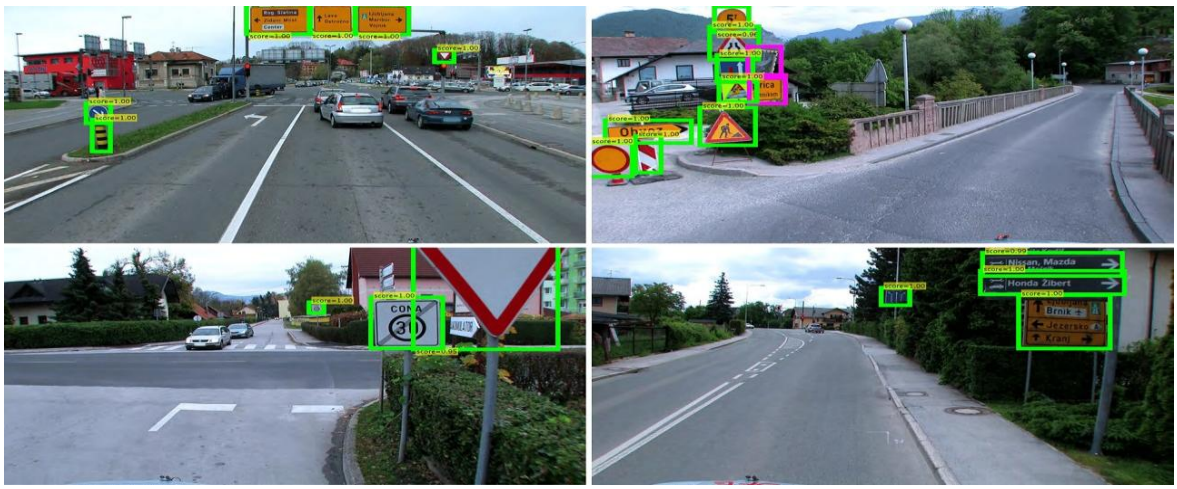


Figure 6.10 Traffic Sign Recognition (TSR)

- To implement the Traffic Sign Recognition (TSR) , there are Several methods that can be used.

6.8.1 Methods:

6.8.1.1 Traditional Image Processing Techniques:

Initial approaches to traffic sign recognition relied on classic image processing techniques such as color thresholding, shape detection (e.g., circles, triangles, octagons), and edge detection. These methods attempt to locate traffic signs based on their geometric shape and predefined color ranges.

- **Tools:**
  - OpenCV: For image thresholding, contour detection, and Hough transform.
  - NumPy: For array-based pixel analysis.
- **Advantages:**
  - Lightweight and fast.
  - Does not require extensive training data.
- **Disadvantages:**
  - Limited accuracy.
  - Not robust against environmental variations such as lighting, blur, or occlusion.

6.8.1.2 CNN-Based Classification

In this method, the system first detects possible regions of interest (ROIs) in the image and then applies a Convolutional Neural Network (CNN) to classify the cropped regions into traffic sign categories.

- **Tools:**
  - TensorFlow / Keras or PyTorch for CNN architecture.
  - Scikit-learn for evaluation and preprocessing.
- **Advantages:**
  - Higher accuracy than traditional methods.
  - Can learn complex features automatically from training data.
- **Disadvantages:**
  - Slower due to separate detection and classification steps.
  - Requires a labeled dataset for training.

6.8.1.3 YOLO Object Detection

YOLO is a real-time object detection model that performs both localization and classification in a single step. It divides the image into grids and predicts bounding boxes and class probabilities simultaneously.

- **Tools:**
  - YOLO framework (custom or modified version).
  - OpenCV for image capture and visualization.
  - PyTorch or ONNX for model inference.
- **Advantages:**
  - Extremely fast (real-time performance).
  - High accuracy for both detection and classification.
  - One-step pipeline simplifies integration.
- **Disadvantages:**
  - Requires GPU for training.
  - Needs sufficient training data and augmentation for best performance.

The most effective approach we will use is YOLO. It provides high accuracy and real-time performance, making it ideal for detecting and classifying traffic signs quickly.

Its lightweight design also supports deployment on embedded systems like Raspberry Pi, ensuring efficiency in dynamic driving conditions.

6.8.1.4 Dataset and Training

Traffic sign recognition systems require extensive training data covering different sign types, lighting conditions, and angles. The most commonly used dataset is GTSRB.

- **Tools:**
  - GTSRB Dataset: Contains over 50,000 labeled traffic sign images.

Use Case :

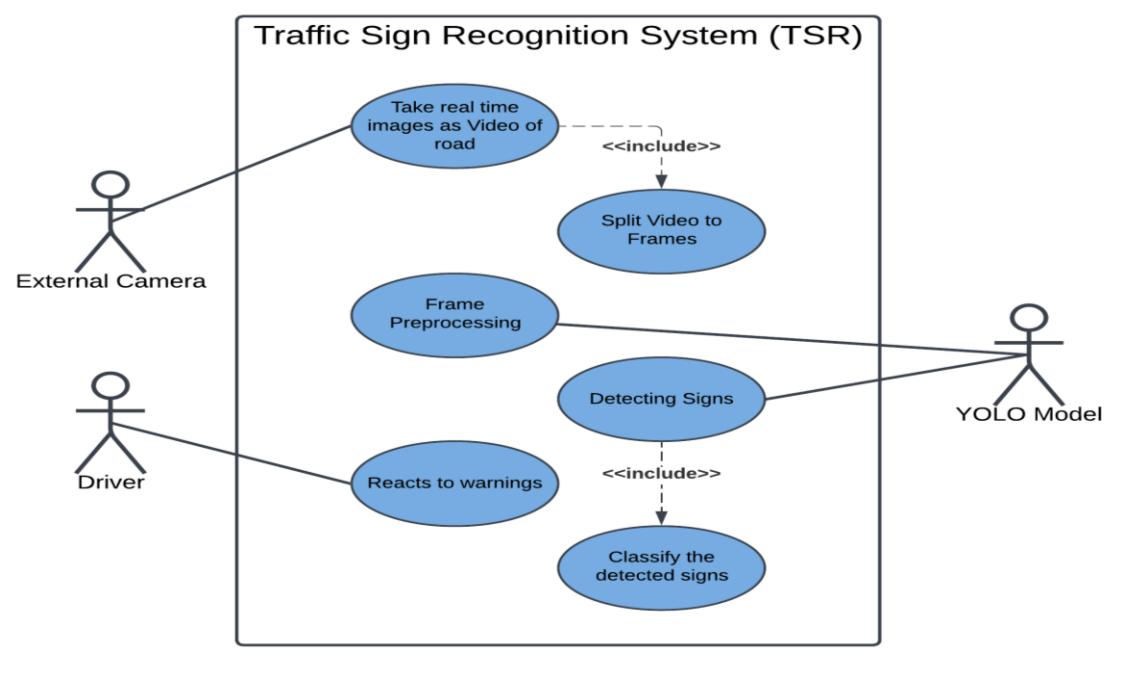


Figure 6.11 Traffic Sign Recognition (TSR) Use Case

Flowchart:

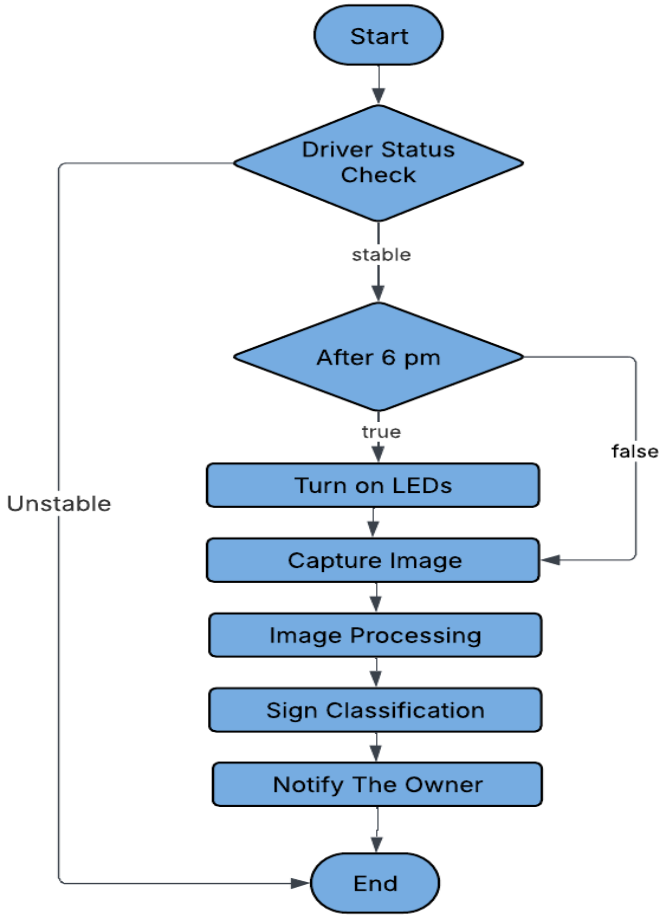


Figure 6.12 Traffic Sign Recognition (TSR) Flow

6.9 Pseudo code for Traffic Sign Recognition (TSR)

```
START

LOAD YOLO MODEL
OPEN CAMERA
WHILE CAMERA IS OPEN:
READ frame FROM CAMERA
PREPROCESS frame
PASS frame TO YOLO MODEL
GET detections (boxes, class_ids, confidences)
FOR EACH detection:
IF confidence >= THRESHOLD:
GET class_name FROM class_id
DRAW bounding box AND label ON frame
DISPLAY result
```

```
END IF
END FOR
END WHILE


END
```

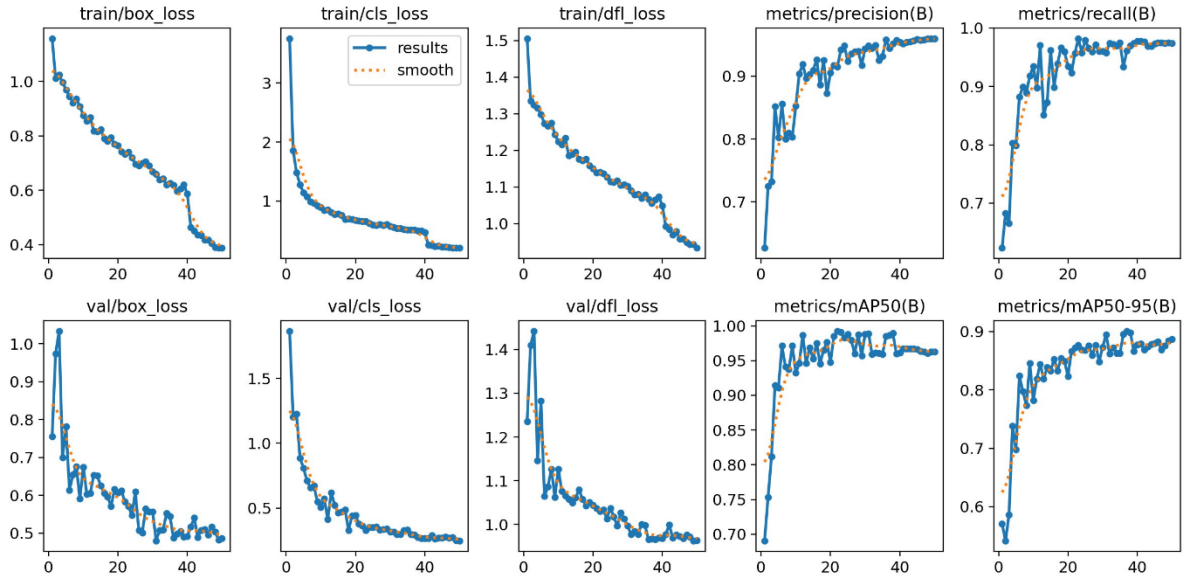


Fig 5. 13 Performance Metrics of The TSR Model

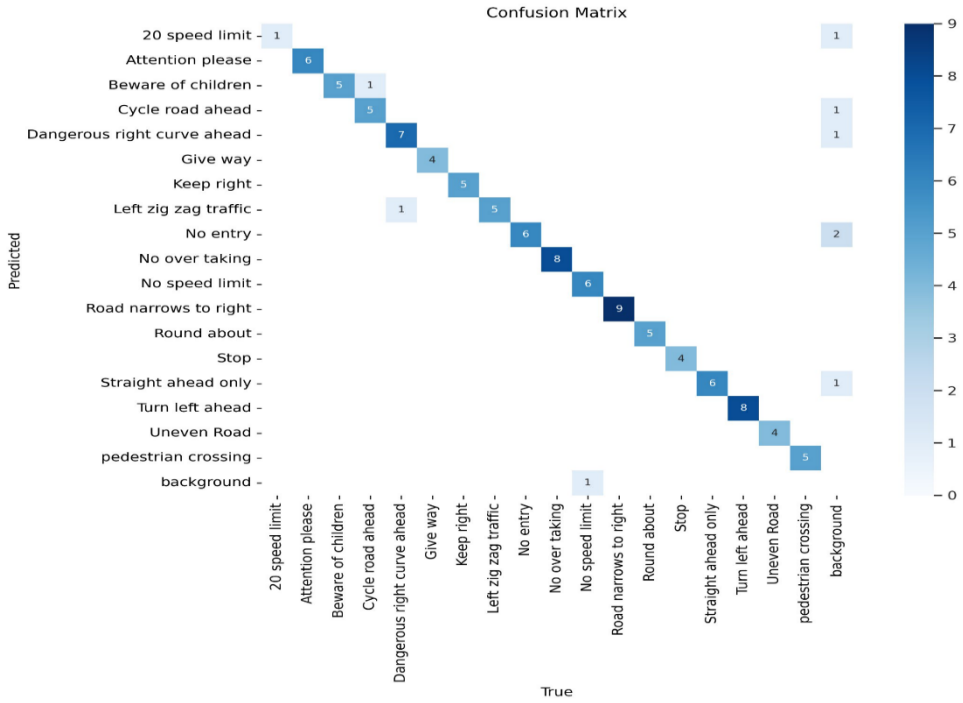


Fig 5. 14 Confusion Matrix Of The TSR Model

6.10 Software Architecture

To manage complexity and ensure modularity, the system uses a layered software architecture where each layer has a specific role and communicates only with adjacent layers.

At the top is the **Application Layer**, which runs the main logic and acts as the entry point. Below it, the **System Models Layer** includes core modules for face recognition and driver monitoring that process user data and handle business logic.

The **Utility Layer** provides helper functions and alarm files to support system operations, while the **Testing Layer** ensures reliability through test scripts.

At the base is the **Data Layer**, responsible for storing and managing data using directories like user_data/ and files like local_data.json.

This structured approach improves portability, simplifies debugging, and supports easy integration of new components, resulting in a robust, scalable, and maintainable system.

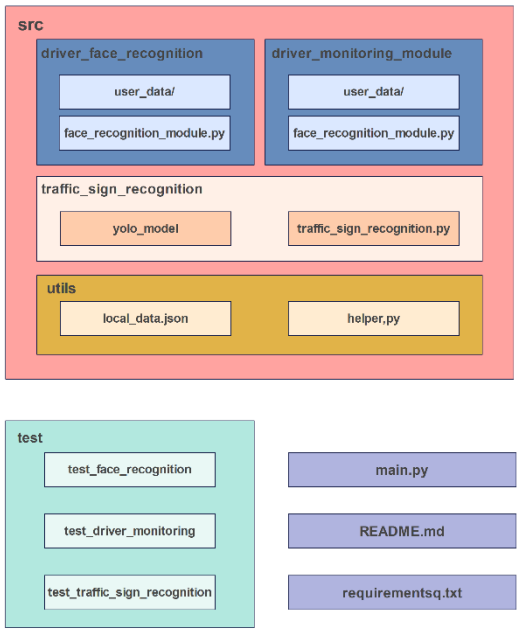


Figure 6.15.a: Directory Structure Overview

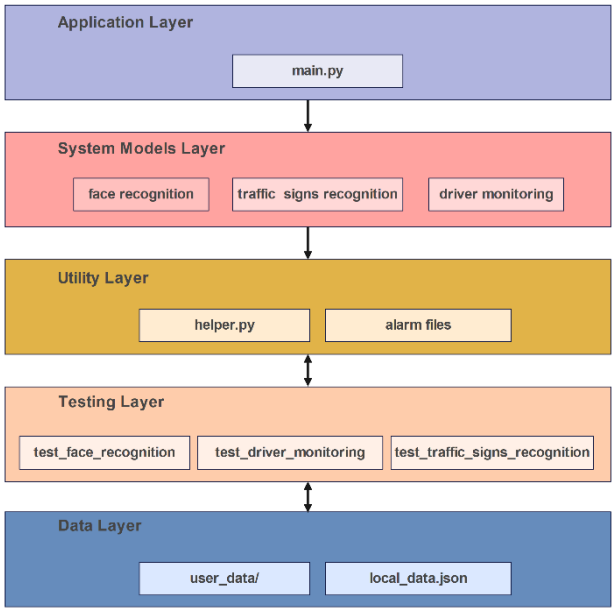
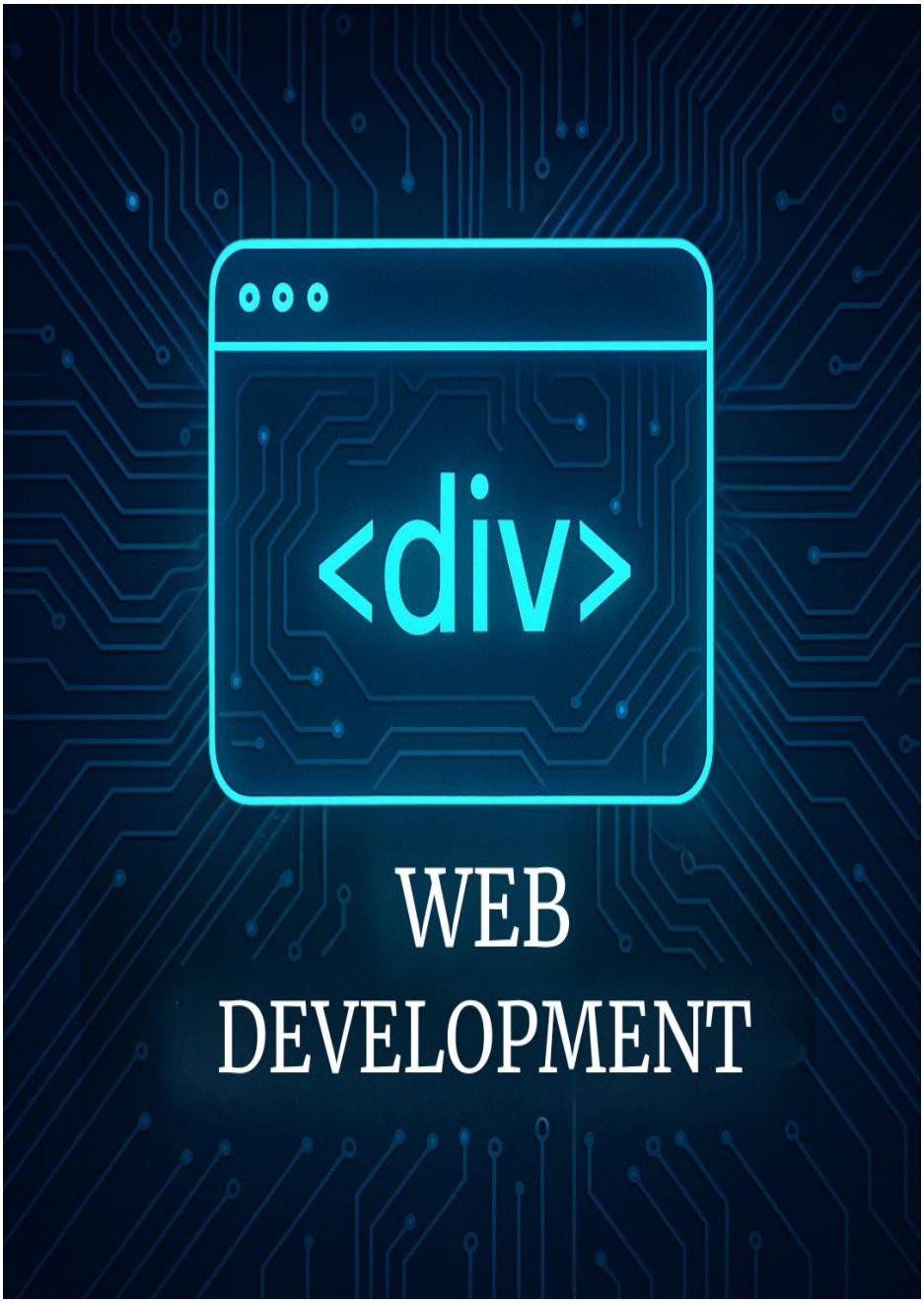


Figure 6.15. b: Software Architecture Layers

Chapter 7



Chapter 7: Web Development

7.1 Web Technologies Overview

At the heart of modern digital transformation, web technologies play a fundamental role in building intelligent systems capable of real-time interaction with users. A fully functional web system consists of several key components working together seamlessly to provide a flexible, secure, and responsive user experience. This architecture starts with the front-end, responsible for visual presentation and user interaction; continues with the back-end, which handles system logic and data processing; extends to databases, which manage and organize information efficiently; and includes hosting services, ensuring the system's availability and performance. The effectiveness of any web system lies in the harmony of these components and their ability to adapt to dynamic environments.

7.1.1 The client/server and P2P

The client/server principle applied to the Internet. This principle is based on a simple idea, illustrated in **Figure 7.1:** Interactions between software systems are broken down into two roles: Clients request services, servers provide them. When a client needs a service such as database access, an e-mail to be sent or a print function to be executed on its behalf, it sends a corresponding request to the respective server. The server will then process this request, i.e., execute the access, message sending, or printing, and will eventually send a reply back to the client.

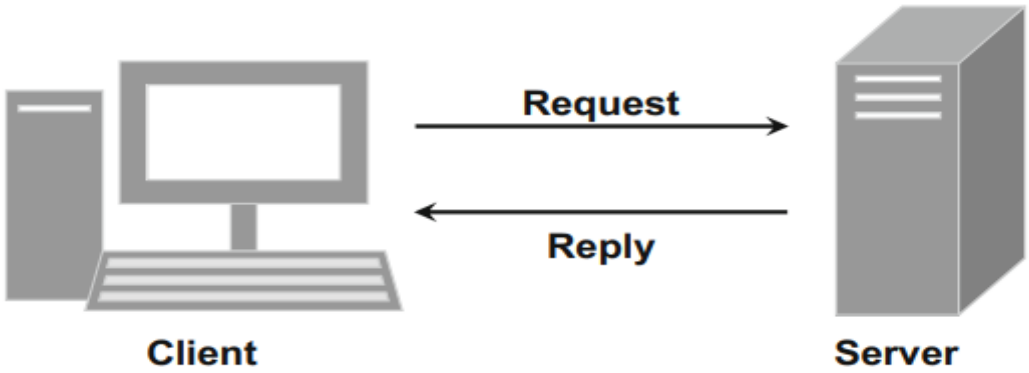


Figure 7.1 client/server principle.

The figure 7.2 demonstrates the fundamental distinction between **client-side scripting** and **server-side scripting** within a web application environment.

Client-side scripting (executed in the web browser) relies on technologies such as **JavaScript** to dynamically modify content, validate forms, and enhance user interaction without requiring communication with the server for every action. The browser processes these scripts using its internal **script engine**, improving responsiveness and user experience.

On the other hand, **server-side scripting** takes place on the web server and involves languages like **PHP**, **Python**, or **Node.js**. When a client sends an HTTP request (typically via a URL), the web server interprets this request using application logic, fetches data from **local** or **external sources**, and dynamically generates a response—usually an HTML page—which is sent back to the client.

This model highlights the collaboration between client-side and server-side processes to deliver responsive, secure, and data-driven web applications.

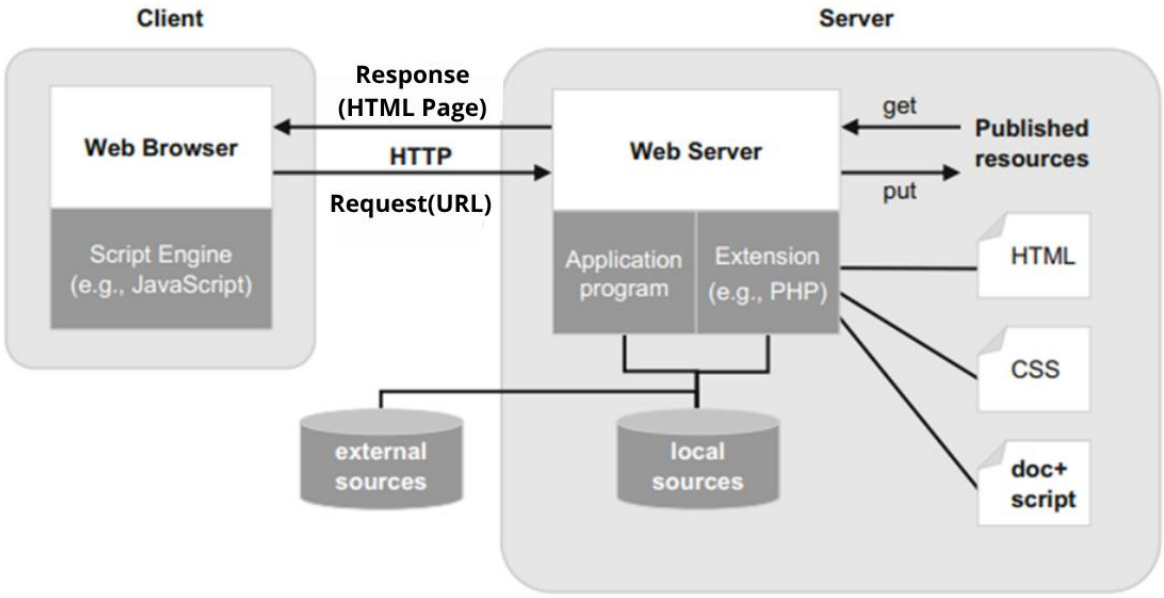


Figure 7.2 Client-side versus server-side scripting

It soon turned out that the traditional client/server model behind the Web was less than optimal for some interactions, including the download or streaming of

large files, e.g., a video file that contains a 90-minute movie. Video streaming is not just a matter of bandwidth; it is also a matter of a single server being occupied with a large request for quite some time. In response to this problem, peer-to-peer (P2P) networks were devised, which bypassed the need for a central server to take care of all incoming requests. Instead, a P2P network primarily relies on the computing power and bandwidth of its participants and is typically used for connecting nodes via mostly ad-hoc connections. A P2P network also does not distinguish between a client and a server; any participant in the network can function as either a client or a server to the other nodes of the network, as needed by the task at hand. In fact, a P2P system comes with complete and decentralized self-management and resource usage, and it enables two or more peers to collaborate spontaneously in a network of equals (peers) by using appropriate information and communication systems.

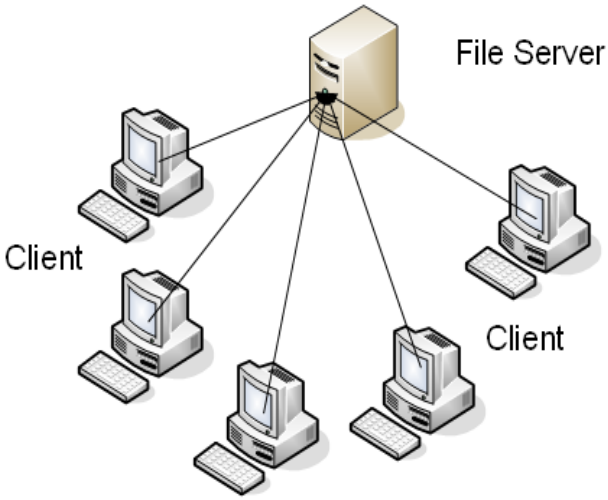


Figure 7.3 Client server mode

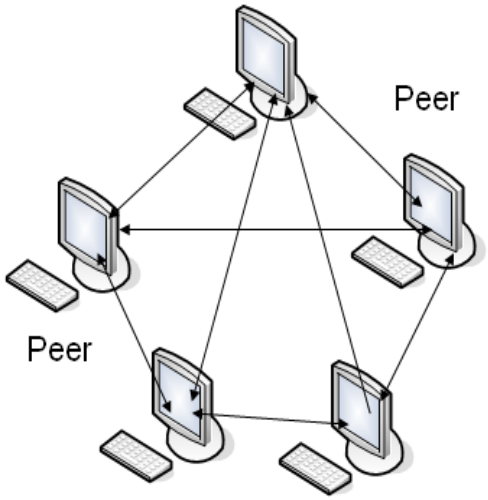


Figure 7.4 Peer to Peer mode

7.1.2 Front-End vs. Back-End

- The **Front-End** is the visible part of the system that users interact with directly — such as pages, buttons, maps, and visual elements. It uses technologies like HTML, CSS, and JavaScript to deliver a smooth and engaging user experience.

    - The **Back-End**, on the other hand, is the hidden part that handles data processing, system logic, and communication with databases. It secures operations, stores data, and delivers responses to the front-end.

- Simply put: the front-end shows, the back-end runs.

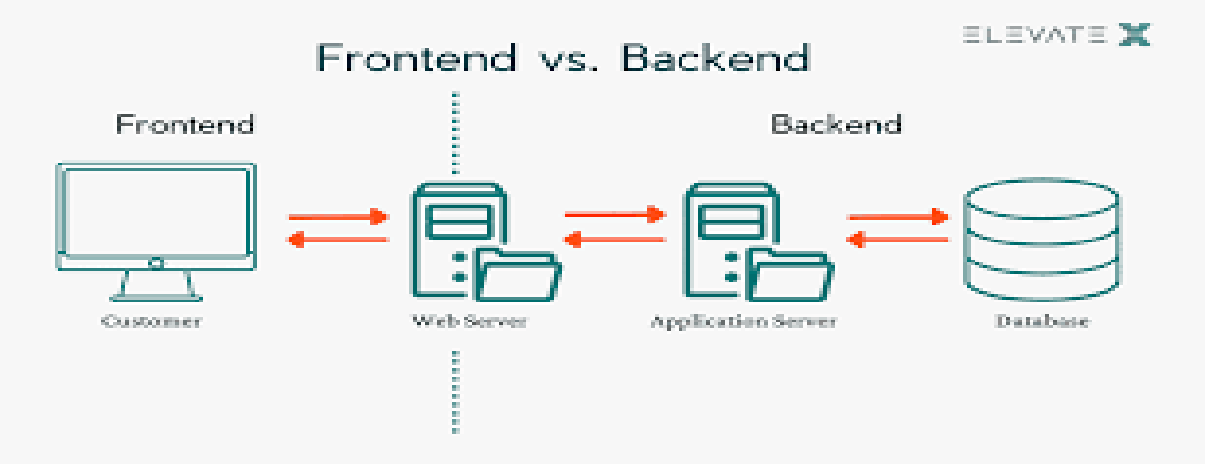


Figure 7.5 Frontend vs. Backend Architecture

7.2 Introduction to the Role of Web in ADAS &V2V System

With the growing reliance on intelligent driving systems, the web has become a central element in delivering a more interactive and safer driving experience. A modern graphical user interface (GUI) has been designed to display real-time vehicle and driver data, alongside a clear visual representation of the car's location and instant alerts. This seamless integration between the system and the interactive interface marks a critical step toward real-time communication between the vehicle and the driver, empowering more accurate and timely decision-making in dynamic environments. The interface not only displays information, but visually embodies the system's analytical intelligence in an intuitive form

7.2.1 Web Technologies Integration with Smart Driving Systems in ADAS & V2V

- Web technologies were used as an interactive layer for real-time vehicle data visualization and analysis.
- The graphical interface allows live monitoring of speed, status, and location through the browser.
- The system relies on Firebase for cloud storage and real-time synchronization.

- Alerts and warnings are displayed instantly to support quick driver decisions.
- The structure is scalable for future integration with advanced technologies like AI.

7.3 Tools Used in the Web Interface of ADAS &V2V

7.3.1 Programming Languages for Front-End Development

7.3.1.1 HTML (Hyper Text Markup Language)

HTML is the foundation of any webpage, used to structure the basic layout. It defines elements like text, images, links, and buttons. With **HTML**, developers can organize the content and specify the order of elements on the page, allowing for well-structured and easy-to-navigate web pages.

HTML was used to build the structural layout of the user interface, including sections such as the dashboard, forms, alert lists, and map containers. Each part was designed using HTML elements.

7.3.1.2 CSS (Cascading Style Sheets)

CSS is responsible for styling and formatting web pages after the basic structure is built with **HTML**. It allows developers to control colors, fonts, spacing, alignment, and the overall layout of a page. **CSS** also enables the creation of responsive designs that.

CSS was used to design a clean and user-friendly interface. The styling ensured visibility and clarity, especially during driving. Elements like alert colors, button styles, and layout sections were customized using CSS.

7.3.1.3 JavaScript

JavaScript is the core language for adding interactivity and dynamic features to web pages. With **JavaScript**, developers can create interactive effects like dropdown menus, animations, form validation, and real-time data loading through technologies like **AJAX**. It is essential for building complex web applications that offer a seamless user

experience.

JavaScript was used to fetch and display real-time vehicle data from Firebase, handle warning logic for the driver, and enable dynamic interaction with the live map view.

7.3.2 Front-End development framework

A **Framework** is a comprehensive development environment that provides a set of pre-built tools, libraries, and guidelines to streamline the application development process. It offers a structured approach that saves time, ensures consistency, and promotes efficient and maintainable code.

7.3.2.1 Bootstrap

Bootstrap is an open-source front-end framework that simplifies the creation of modern, responsive designs. It includes ready-made components like buttons, grid systems, and icons. **Bootstrap** helped in creating a responsive layout that adapts to various screen sizes (desktop, mobile, tablet). It was used to build sidebars, buttons, and tables with minimal effort and consistent style.

7.3.3 Front-End Libraries

A **software library** is a collection of pre-written code that provides specific functionality which developers can integrate into their own projects. Libraries are designed to streamline development, simplify repetitive or complex tasks, and promote code reusability. They help developers build more efficient, reliable, and maintainable applications without reinventing the wheel.

7.3.3.1 Leaflet.js

Leaflet.js is one of the most popular open-source JavaScript libraries for building interactive maps in web applications. Weighing just around 42 KB, it provides all the essential mapping features most developers need.

Leaflet is designed with simplicity, high performance, and usability in mind. It works efficiently across all modern web browsers and can be extended with a wide variety of plugins. It also features a clean, well-documented API and readable source code, making it easy to use, customize, and contribute to in web-based environments.

```
<script src="https://unpkg.com/leaflet/dist/leaflet.js"></script>
            <script type="module">
              import { initializeApp } from
"https://www.gstatic.com/firebasejs/9.0.0/firebase-app.js";
              import { getDatabase, ref, onValue } from
"https://www.gstatic.com/firebasejs/9.0.0/firebase-database.js";

              const firebaseConfig = {
                apiKey: "AIzaSyBhrthJEbCEt6bjWnzOO-iFQm1oDc5Ikgw",
                authDomain: "maps-effb1.firebaseapp.com",
                databaseURL: "https://maps-effb1-default-rtdb.firebaseio.com",
                projectId: "maps-effb1",
                storageBucket: "maps-effb1.appspot.com",
                messagingSenderId: "956469747194",
                appId: "1:956469747194:web:9b32c2f0dd464f9d1a9843"
              };

              const app = initializeApp(firebaseConfig);
              const db = getDatabase(app);

              const map = L.map('map').setView([31.0357, 31.2497], 13);
              L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png'
, {
                attribution: '&copy; OpenStreetMap contributors'
              }).addTo(map);

              const marker = L.marker([30.0444, 31.2357]).addTo(map);

              const carRef = ref(db, 'car/location');
              onValue(carRef, (snapshot) => {
                const data = snapshot.val();
                if (data && data.latitude && data.longitude) {
                  marker.setLatLng([data.latitude, data.longitude]);
                  map.setView([data.latitude, data.longitude]);
                }
              });
            </script>
```

7.3.3.2 Three.js

Three.js is a 3D JavaScript library that enables developers to create 3D experiences for the web. It works with WebGL.

It was specifically used to render a realistic 3D model of the vehicle within the application.

WebGL (Web Graphics Library) is a browser-based graphics API used to render both 3D and 2D graphics without the need for external plugins. It is based on the OpenGL ES specification and is implemented using JavaScript.

```
// Load car model
const loader = new THREE.GLTFLoader();

loader.setPath('./');

loader.load('./3d_car.glb', (gltf) => {
  const model = gltf.scene;
  model.scale.set(23, 23, 23);
  model.position.set(0, -2, 0);

  model.traverse((node) => {
    if (node.isMesh) {

      node.castShadow = false;
      node.receiveShadow = false;

      if (node.material) {

        node.material.envMapIntensity = 0.5;
        node.material.roughness = 0.8;
        node.material.metalness = 0.5;

        if (node.material.normalMap) node.material.normalMap = null;
        if (node.material.roughnessMap) node.material.roughnessMap = null;
        if (node.material.metalnessMap) node.material.metalnessMap = null;
        if (node.material.aoMap) node.material.aoMap = null;

        if (node.material.map) {
          node.material.map.minFilter = THREE.LinearFilter;
          node.material.map.generateMipmaps = false;
        }
      }
    }
  });
```

WebGL allows developers to build high-performance, interactive visual content—such as games, simulations, and 3D models—directly within web browsers. While powerful, it is considered a low-level API that requires advanced knowledge of graphics mathematics and 3D rendering concepts.WebGL also serves as the foundation for higher-level libraries like

Three.js, which simplify the development process by offering more intuitive tools and components.

7.3.3.3 Font Awesome

Font Awesome is designed to be used with inline elements. The <i> and elements are widely used for icons.

Also note that if you change the font-size or color of the icon's container, the icon changes. Same things goes for shadow, and anything else that gets inherited using CSS.

```
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.2/css/all.min.css" media="print" onload="this.media='all'">
```

7.3.3.4 Google Font

Google Fonts is a free service by Google offering a wide range of web-friendly fonts suitable for modern UI design.

Google Fonts was used to customize the UI typography, enhancing readability and giving the interface a clean, modern appearance.

```
<link rel="preconnect" href="https://fonts.googleapis.com">    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&display=swap" rel="stylesheet" />
```

7.3.3.5 Remix Icon

Remix Icon is a set of open-source neutral-style system symbols elaborately crafted for designers and developers.

```
<link href="https://cdn.jsdelivr.net/npm/remixicon/fonts/remixicon.css" rel="stylesheet" />
```

7.3.4 Back-End development languages and framework

7.3.4.1 Firebase

Firebase is a comprehensive Backend-as-a-Service (BaaS) platform developed and maintained by Google, offering a wide range of cloud-based services tailored for the

development of mobile and web applications. It simplifies the implementation of complex backend features, providing developers with robust tools for authentication, real-time data synchronization, cloud storage, and more. By abstracting away much of the server-side management, Firebase allows developers to focus more on building the frontend and user experience while ensuring that backend services remain scalable, secure, and efficient

In the context of the ADAS & V2V (Advanced Driver Assistance System - Vehicle to Vehicle) project, Firebase was chosen as the backend solution to support key functionalities necessary for vehicle communication and data management. The core Firebase services utilized in this project include **Firebase Authentication**, **Firebase Realtime Database**, and **Fire store Database**. These services collectively enabled seamless real-time communication between vehicles, as well as reliable storage and synchronization of critical data, such as vehicle speed, location, system status, and user profile information.

The figure 7.6 illustrates the architecture of **Firebase Cloud Messaging (FCM)**, a powerful system provided by Google that enables developers to send notifications and data messages reliably across platforms. The architecture is composed of four main stages:

1. **Message Building & Targeting:**
    - Messages are created through the **Notifications Console GUI** or programmatically via the **Admin SDK** or **HTTP/XMPP** protocols from a trusted environment (such as a server).
    - The messages are then directed to a specific topic or device instance.
2. **FCM Backend:**
    - The core of the Firebase messaging system, this backend handles routing and message processing.
    - It ensures that messages are dispatched to the appropriate platform-specific channels.
3. **Platform-Level Message Transport:**
    - Messages are delivered using platform-specific transport layers:
        - **Web Push** for browsers supporting web notifications.
        - **Android Transport Layer** for Android devices.
        - **iOS/APNs (Apple Push Notification service)** for iOS devices.
4. **SDK on Device:**

- The final destination is the client device, which includes mobile phones (Android/iOS) or web-enabled devices.
- The installed **Firebase SDK** on the device handles receiving and processing incoming messages

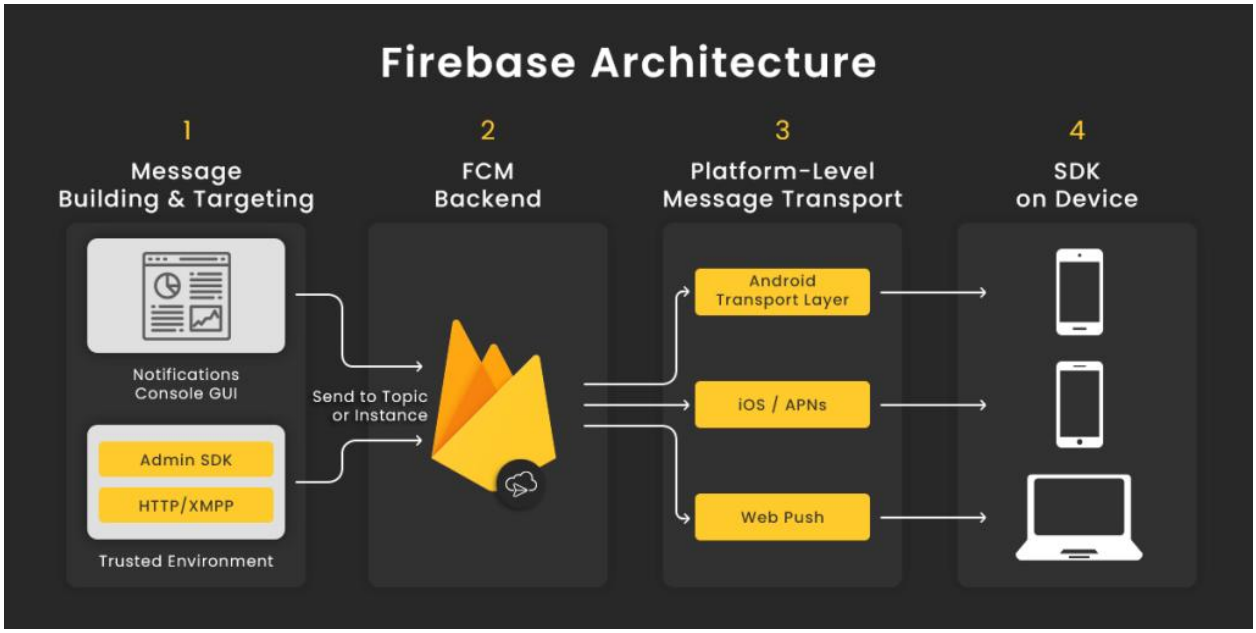


Figure 7.6 Firebase Cloud Messaging Architecture

7.3.4.1.1 Authentication and Login using Firebase Authentication

Firebase Authentication is a service provided by the Firebase platform to enable developers to easily and securely add user authentication to their applications. This service supports various authentication methods such as email/password, Google, Facebook, Twitter, and other third-party authentication providers. It also offers phone number authentication, making it easy to manage user login processes without the need to build a complex authentication system from scratch.

As shown in the figure 7.7 Firebase Authentication was used in the project to provide three secure login methods that were implemented:

1. **Message Building & Targeting:**
   - Messages are created through the **Notifications Console GUI** or programmatically via the **Admin SDK** or **HTTP/XMPP** protocols from a trusted environment (such as a server).
   - The messages are then directed to a specific topic or device instance.

2. **FCM Backend:**
    - The core of the Firebase messaging system, this backend handles routing and message processing.
    - It ensures that messages are dispatched to the appropriate platform-specific channels.
3. **Platform-Level Message Transport:**
    - Messages are delivered using platform-specific transport layers:
        - **Web Push** for browsers supporting web notifications.
        - **Android Transport Layer** for Android devices.
        - **iOS/APNs (Apple Push Notification service)** for iOS devices.
4. **SDK on Device:**
    - The final destination is the client device, which includes mobile phones (Android/iOS) or web-enabled devices.
    - The installed **Firebase SDK** on the device handles receiving and processing incoming messages

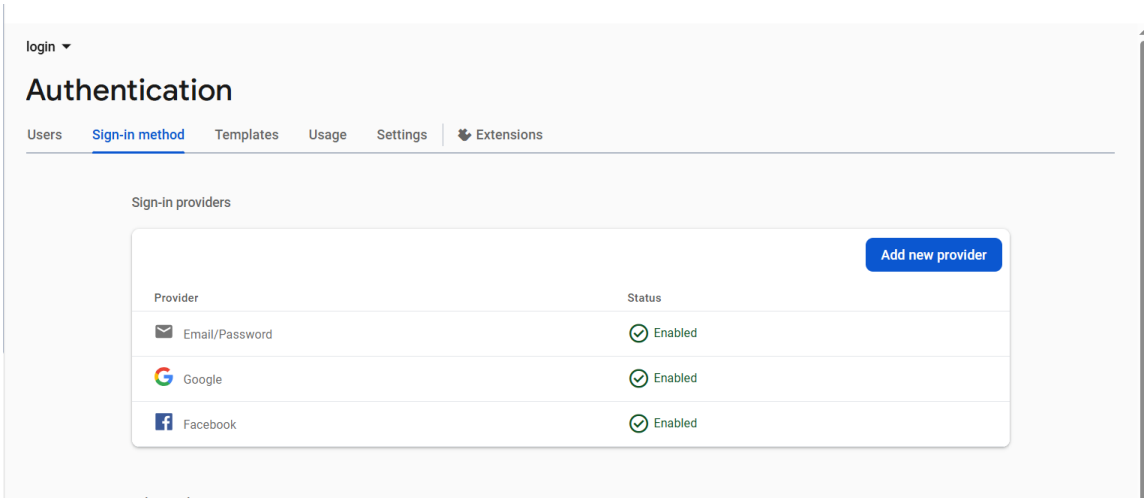


Figure 7.7 User Sign-In Options Using Firebase.

7.3.4.1.1.1 OAuth 2.0 Protocol

OAuth 2.0 is an open authorization protocol that allows third-party applications to securely access a user's data on other services (such as Google, Facebook, Twitter, etc.) without requiring the user to share their password with the third-party application.

In simple terms, OAuth 2.0 enables users to grant limited access to their information stored on one service to another application, without exposing their credentials. This enhances both security and user convenience.

For example, instead of signing up with a new username and password, a user can log in to an app using their Google or Facebook account. Behind the scenes, OAuth 2.0 handles the secure communication and access control between the application and the service provider.

The **Figure 7.8** illustrates the **OAuth 2.0 workflow** that enables a client application (Business Client) to access a protected resource on a Resource Server, after obtaining authorization from the user through the OAuth Server. Here's how it works step-by-step:

1. **The client application requests an access token** from the OAuth server.
2. **The OAuth server verifies the client's identity**, ensuring it is registered and trusted.
3. **The user is prompted to give consent**, allowing the client application to access their data.
4. **Once consent is granted,** the OAuth server **issues an access token** to the client.
5. **The client uses the token** to request access to the protected resource on the Resource Server.
6. **The Resource Server validates the access token** with the OAuth server.
7. **The OAuth server confirms the token is valid.**
8. **The Resource Server returns the requested resource** (e.g., user data) to the client.

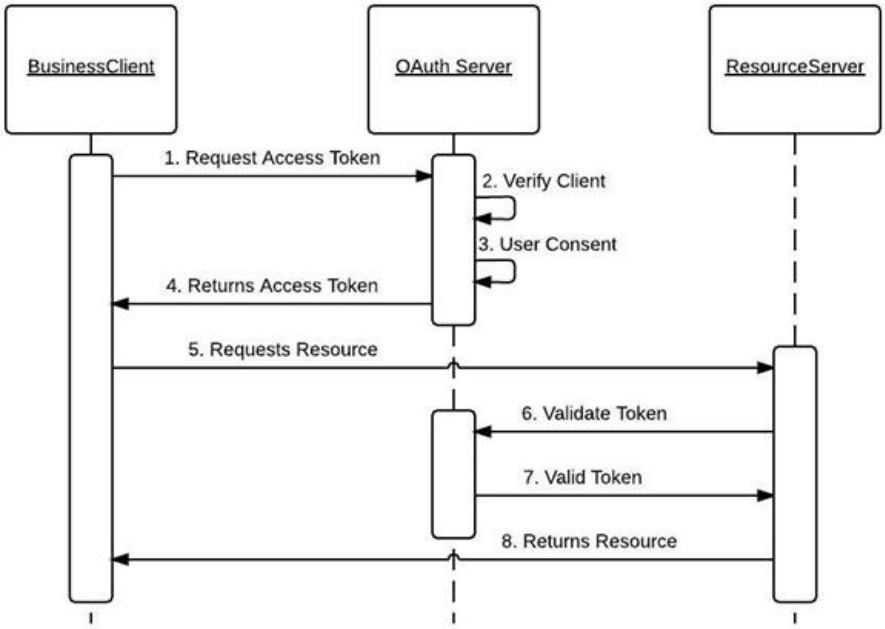


Figure 7.8 OAuth 2.0 Access Authorization

7.3.4.1.2 Real-Time Vehicle Data with Firebase Realtime Database

Firebase Realtime Database has been used to store and synchronize vehicle data in real time, allowing continuous and immediate monitoring of the vehicle's status through the web interface. The following data is stored in the **Realtime Database**:

- **Speed:** The vehicle's speed in real time.
- **Location:** The vehicle's geographical location using coordinates (latitude and longitude).
- **Feature State:** The status of each feature or property, such as alerts, automatic stops, and other active/inactive features.
- **Distance Travelled:** The distance the vehicle has covered since the start of the journey.
- **Time:** The real-time timestamp when the speed and distance data are recorded.

This data is displayed directly on the web interface, enabling continuous, real-time monitoring of the vehicle's status and allowing for appropriate actions based on the received information.

The figure 7.9 demonstrates how the face recognition feature operates within the system. When facial feature data is transmitted to the Firebase Realtime Database, the driver interface immediately reflects the recognition status.

If the face is successfully identified, a message is displayed on the driver's screen indicating the recognized identity. Additionally, warning alerts may be shown in case of abnormal behavior or identity mismatch. This feature is an essential component of the ADAS & V2V system, enhancing driver monitoring and overall vehicle safety.

This real-time feedback mechanism is applied across all system features, ensuring that every event or anomaly detected by the system is reflected instantly to the driver, improving responsiveness and situational awareness.

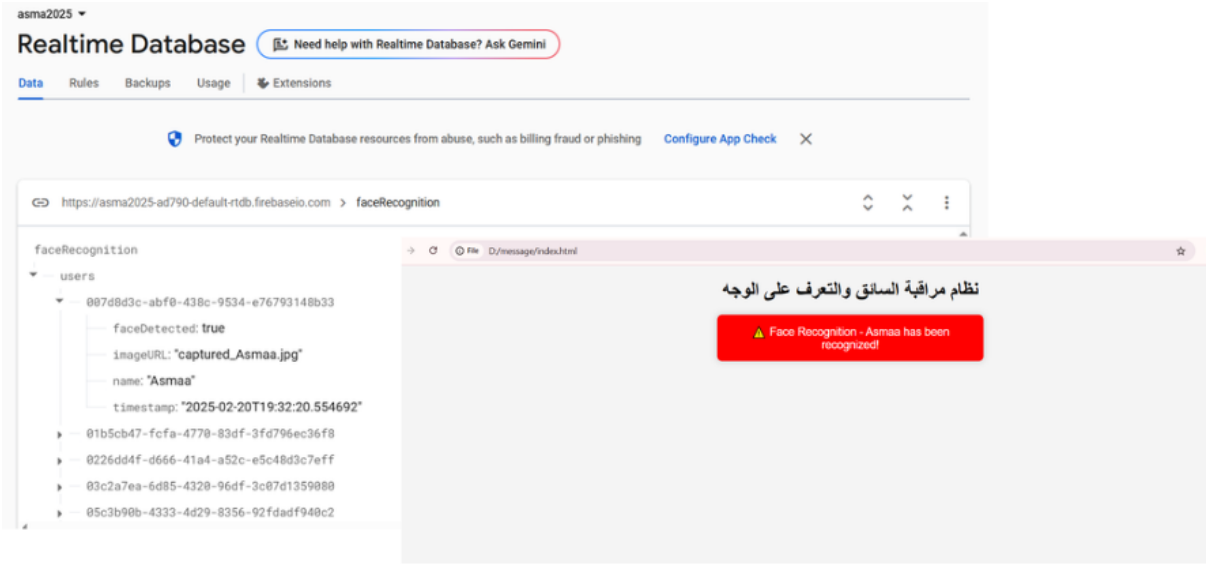


Figure 7.9 Real Time Database.

7.3.4.1.3 Fire store Database

Cloud Fire store—a scalable and real-time NoSQL cloud database from Firebase—was used to store driver information efficiently and securely. As shown in the Figure 7.10, each user is identified by a unique ID under the user's collection, and has the following associated fields:

- **Display Name:** The driver's display name (e.g., "LANELOONS").

- **email:** The registered email address (e.g., "laneloons@gmail.com").

Fire store provides real-time data synchronization and secure access, which makes it an ideal backend service for connected vehicle systems. Whenever any data is updated, changes are instantly reflected across all connected clients, enabling accurate monitoring of drivers and vehicles in the network.

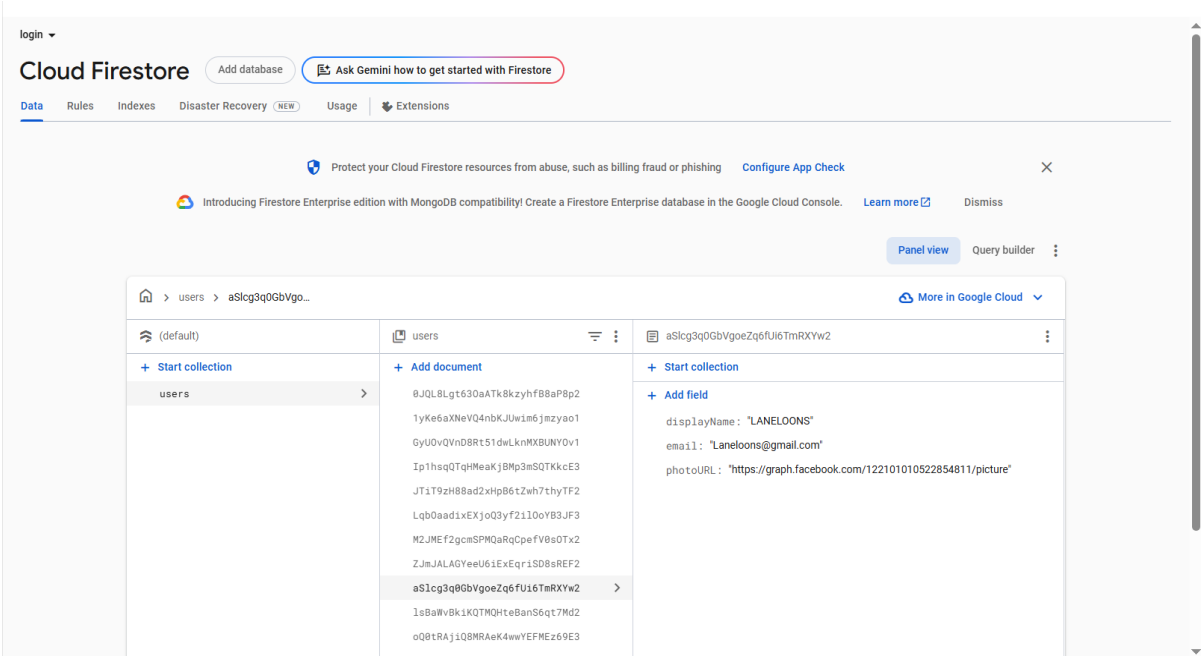


Figure 7.10 Utilizing Fire store Database.

7.4 Interfaces Analysis & Design

Use Case:

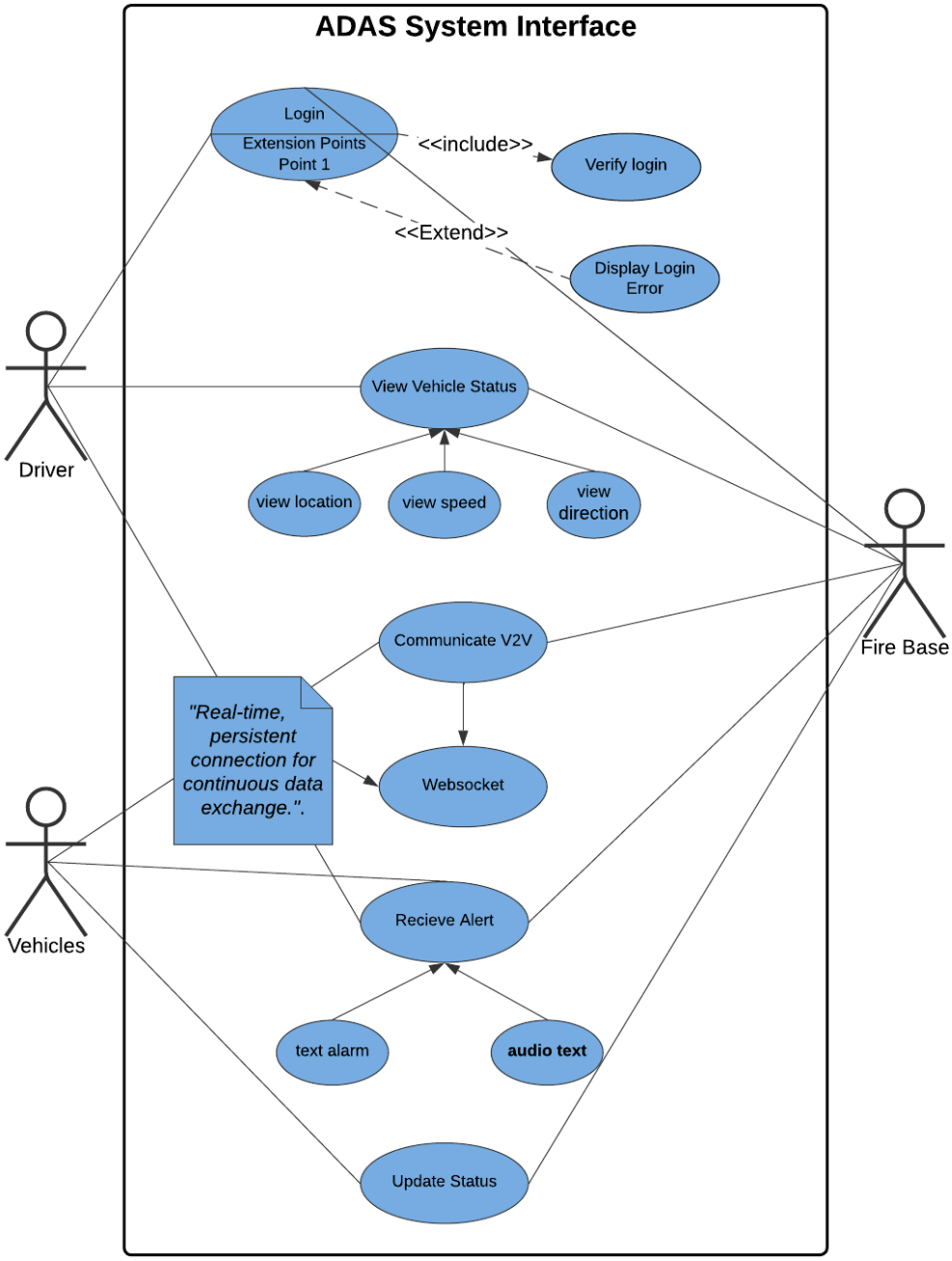


Figure 7.11 Use Case Diagram

Flow Chart:

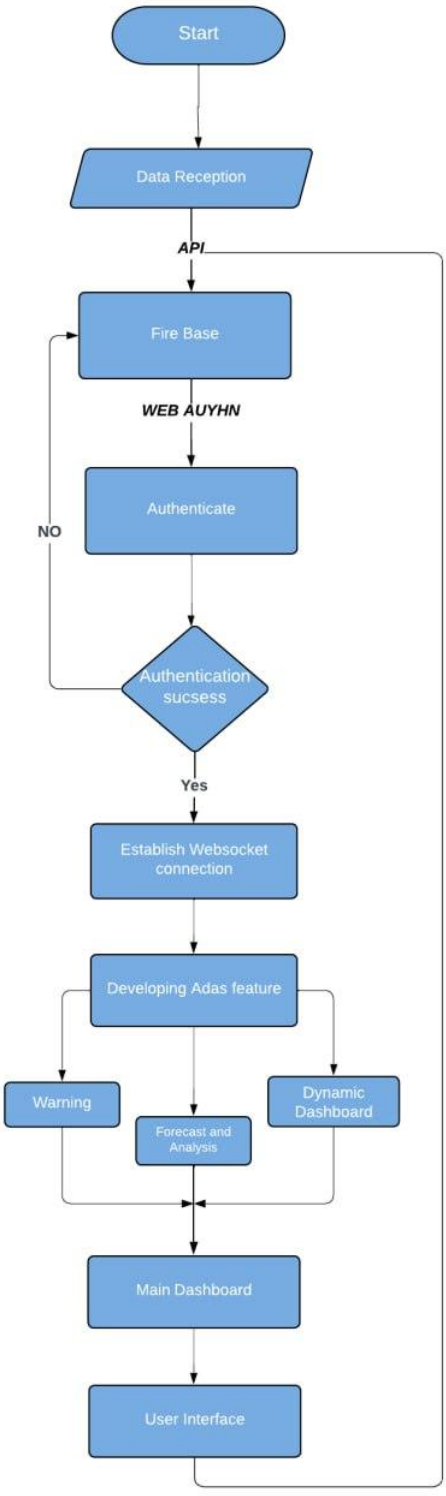


Figure 7.12 Flow Chart Diagram.

Data Flow Diagram (DFD) level 1

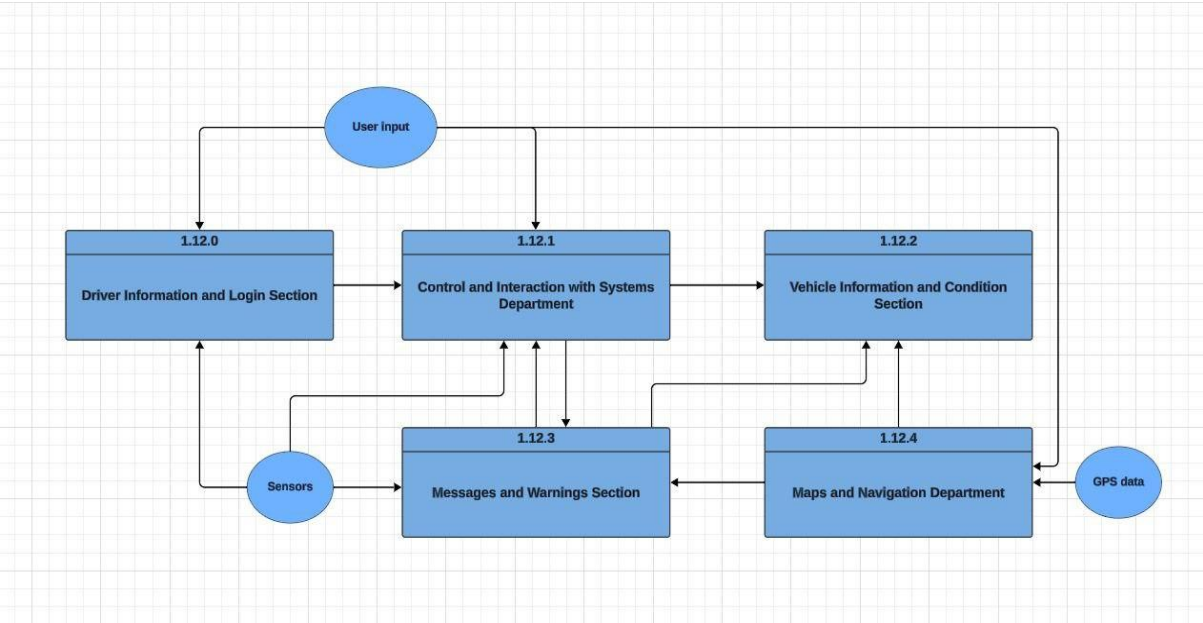


Figure 7.13 Data Flow Diagram

7.5 User Interface (UI) Design

7. 5.1 Register & Sign in

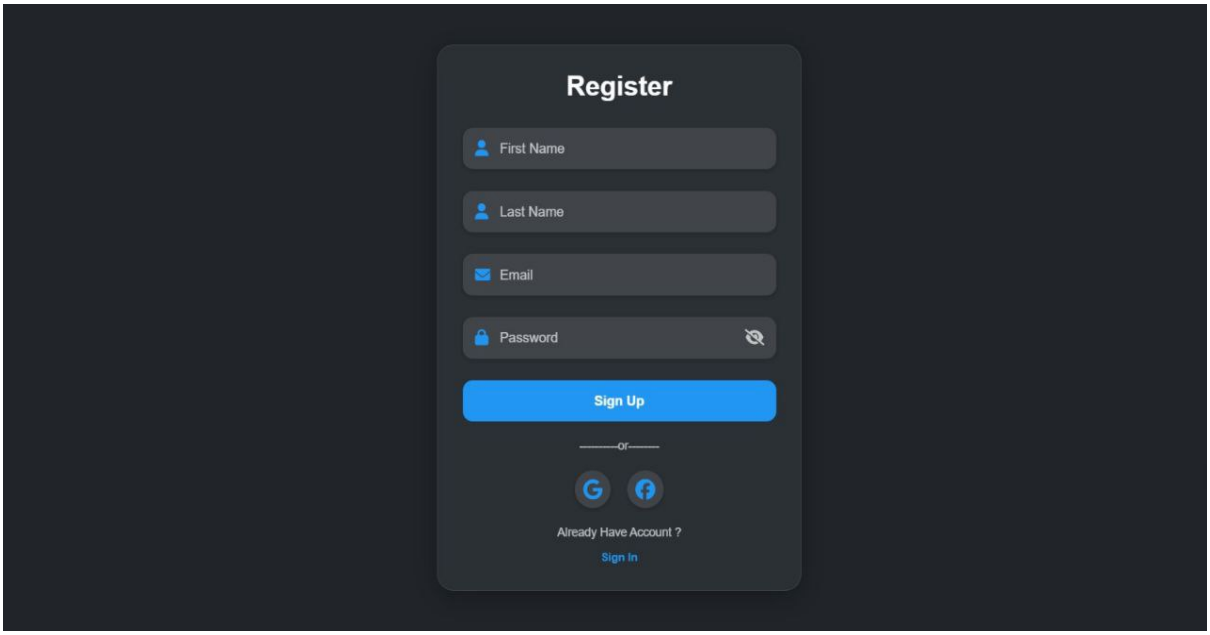


Figure 7.14 Register Page

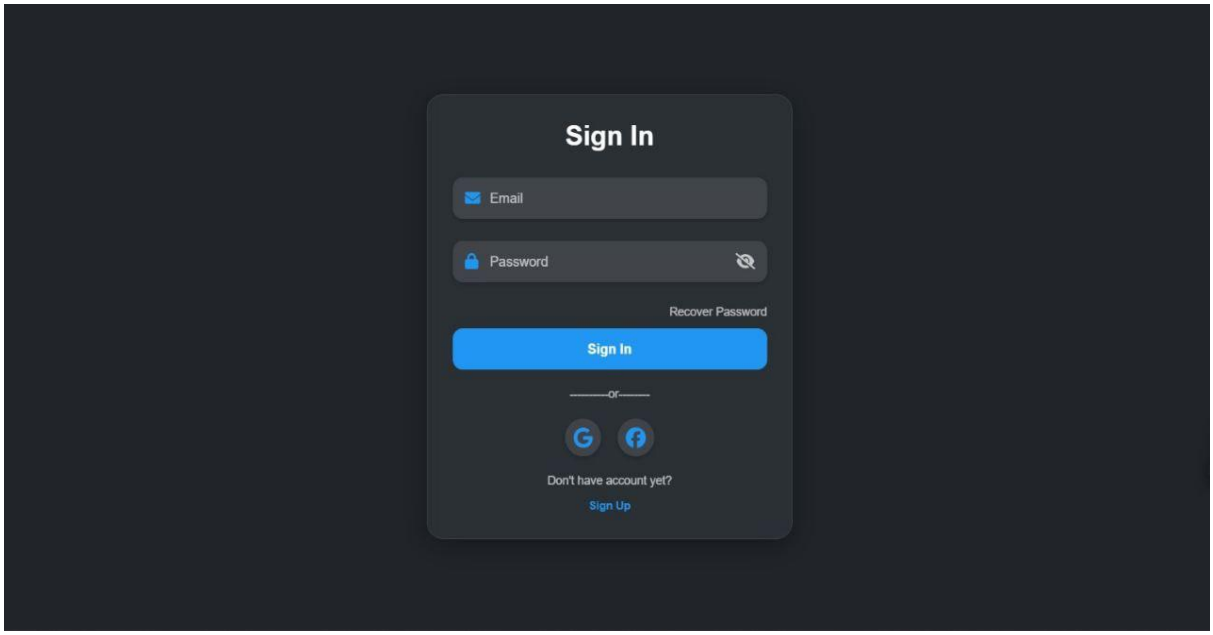


Figure 7.15 Sign in Page

7.5.2 Home Page

- A comprehensive interface displaying all essential vehicle data.
- Modern design with a dark theme suitable for night-time visibility.
- Fully responsive layout that adapts automatically to different screen sizes.

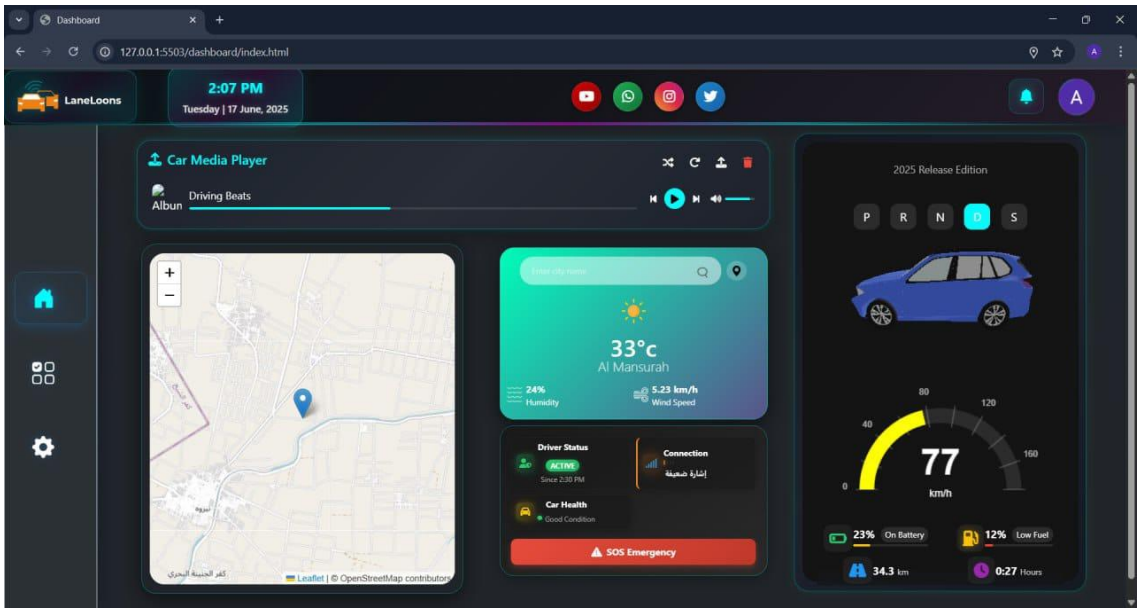


Figure 7.16 Home Page

Main Features on the Home Page

1. 3D Car Visualization

- Interactive 3D model with dynamic rotation.
- High-quality lighting, shadows, and full device compatibility.
- **Technologies:** Three.js, WebGL, HTML5, CSS3, JavaScript.

2. Vehicle Status Monitoring

- **Speedometer:** Real-time speed display up to 180 km/h, smooth animation, dynamic gradients.
- **Battery Management:** Live battery status, visual alerts, auto-updated icons.
- **Fuel Management:** Live tracking with color-coded alerts (Green-Yellow-Red), visual condition indicators.
- **Drive Mode Selection:** Supports (P, R, N, D, S) with smooth transitions and active mode highlighting.
- **Distance & Time Display:** Real-time numeric indicators for distance and current time.

3. Weather Information System

- Live temperature, sky status, wind speed.
- Auto location detection and updates every 30 minutes.
- Searchable city weather with animated icons and severe weather alerts.
- **Technologies:** Geolocation API, Open Weather Map API, Fetch API, Local Storage.

4. Audio System

- Playback for audio/video with equalizer and volume control.
- Multi-format support and local storage via Indexed DB.
- **Technologies:** Web Audio API, Media Session API, Indexed DB.

5. Map-Based Vehicle Tracking System

Features:

- Real-time display of the vehicle's location while driving.
- Live coordinate updates using **Google Maps API**.
- Identification of cities, streets, and nearby landmarks.
- Integrated with monitoring systems for improved emergency response.
- Expandable for storing past driving routes.

Technologies Used:

- **Google Maps JavaScript API**
- **Geolocation API**

The **Geolocation API** is a web technology provided by modern browsers that allows web applications to access the geographic location of a device. This API uses different data sources such as GPS, Wi-Fi networks, and cell towers to determine the location with varying levels of accuracy.

The Geolocation API is accessed via a call to navigator. geolocation. This will cause the browser to ask the user for permission to access their location data. If the user accepts, the browser will search for the best available functionality on the device to access this information (for example GPS).

The get Current Position () method is used to return the user's current location.

```
function getWeatherByLocation() {
   if (navigator.geolocation) {

       showLoading();

       navigator.geolocation.getCurrentPosition(
           (position) => {
               const lat = position.coords.latitude;
               const lon = position.coords.longitude;
               getWeatherByCoordinates(lat, lon);

               localStorage.setItem('lastLat', lat);
               localStorage.setItem('lastLon', lon);
           },
```

```
            (error) => {

                hideLoading();

                console.error("Geolocation error:", error);
                showWeatherNotification('Unable to determine your location.
Please allow location access.', 'error');

                const lastCity = localStorage.getItem('lastWeatherCity');
                if (lastCity) {
                    getWeatherByCity(lastCity);
                }
            },
            { timeout: 10000 }
        );
    } else {
        showWeatherNotification('Your browser does not support location
detection', 'error');
    }
}
```

7.5.3 Features Page

- Features are categorized into visual and interactive sections for easy navigation.
- Provides a detailed display of the car's advanced technical functionalities.

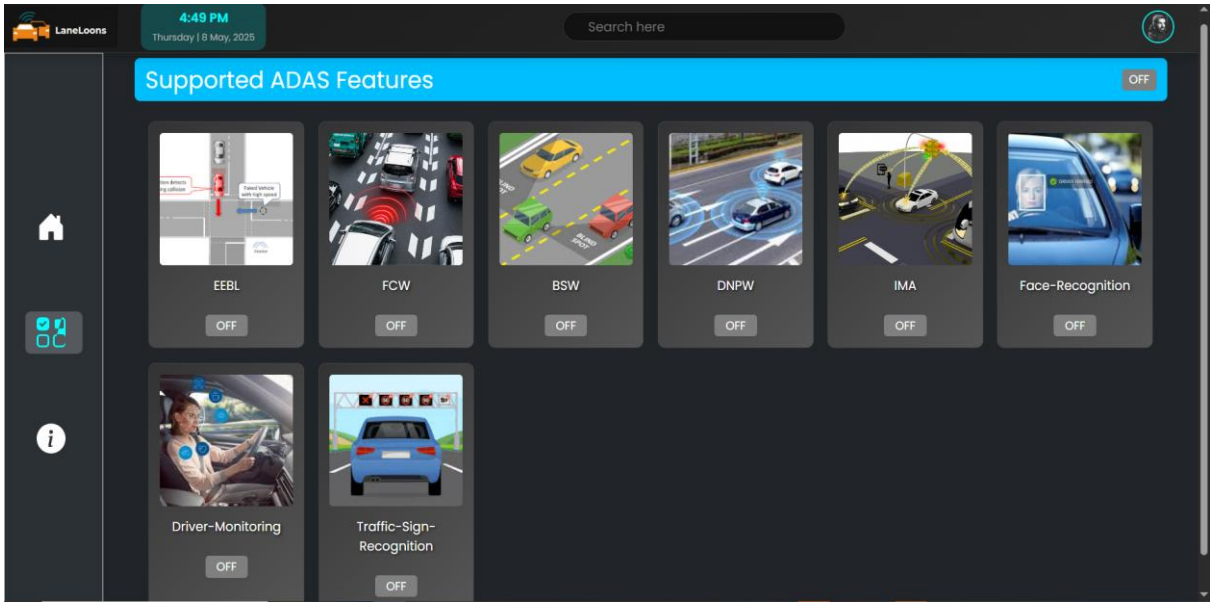


Figure 7.17 Features Page.

Main Features on the Features Page

- HTML5, CSS3, JavaScript.
- Firebase Realtime Database
- Firebase SDK (On Value, set, ref): For reading and writing live data from/to Firebase.
- Popup Info Box: To show detailed feature descriptions when clicked
- Theme Toggle: A theme toggle button switches between light and dark mode by toggling the dark-mode class on the <body> element.

JavaScript DOM API: Used to control page elements (such as cards and buttons) when interacting with them—like on click, page load, or data updates.

7.5.4 Settings Page

Description:

The Settings page serves as the central control hub for managing all system and vehicle configurations efficiently. With a clean and organized interface, users can easily access vehicle information, contact customer support, and monitor real-time analytics—enhancing user experience and operational performance.

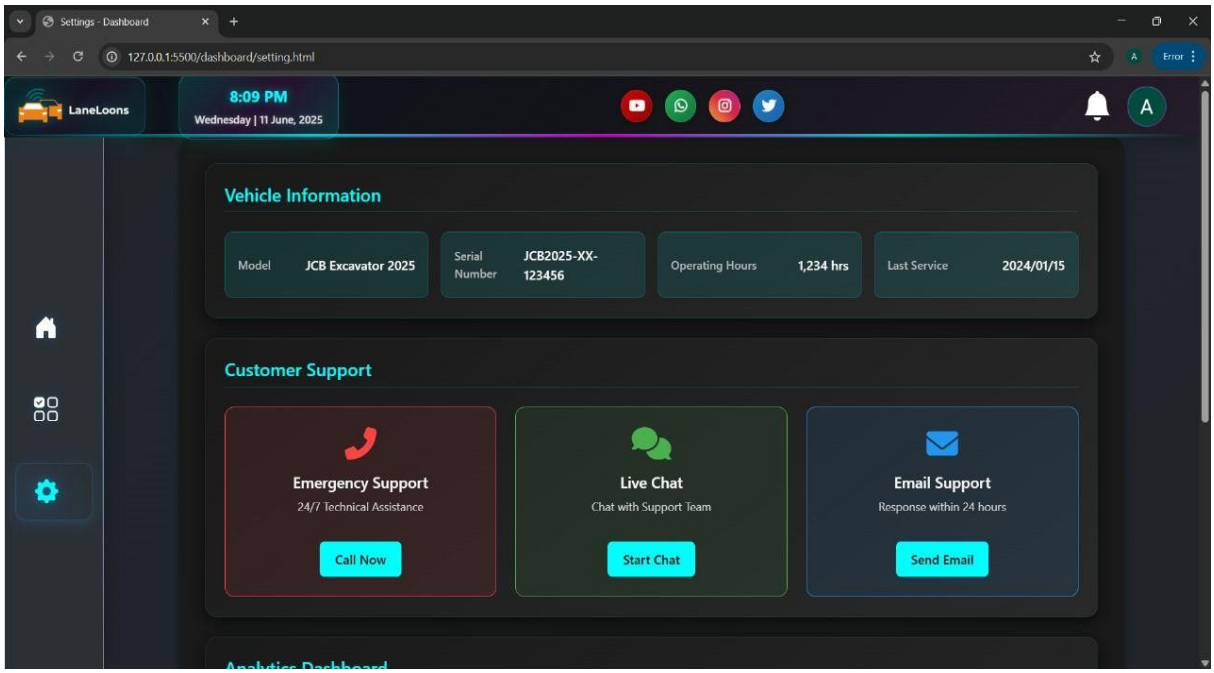


Figure 7.18 Setting Page.

Settings Sections:

A. Vehicle Information
- Vehicle Model: JCB Excavator 2025
- Serial Number: JCB2025-XX-123456
- Operation Hours: 1,234 hrs
- Last Service Date: 01/15/2025

B. Customer Support
- Emergency Support:
  a) Direct Access: Instant access to the support team
  b) Emergency numbers: Available in all regions
  c) Response time: Less than 5 minutes
- Live Chat:
  a) Instant chat with technicians
  b) Encrypted conversations
  c) Save chat history
- Email Support:
  a) Response time: Within 24 hours
  b) Ticketing system: Track order status
  c) Automated response: Confirm order receipt
  d) Continuous follow-up: Regular updates on order status

C. Analytics Dashboard
- Speed Analysis Graph: Displays speed data over time with automatic updates
- Fuel Consumption Graph: Displays daily fuel consumption data with automatic updates
- Real-Time Data Refresh: Simulates live data for continuous monitoring

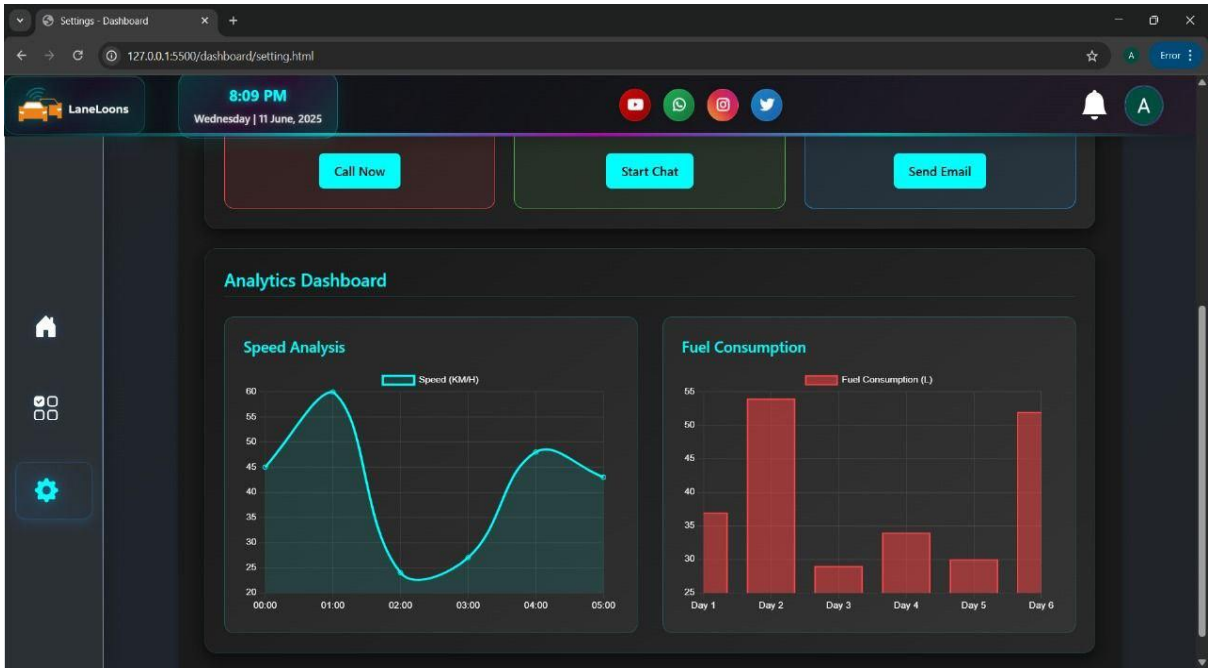


Figure 7.18 Setting Page.

D. Technologies Used:

- Chart.js for interactive graphs
- Firebase for authentication and data storage
- Bootstrap for responsive layout
- Font Awesome for icons

7.6 Project and Team Portfolio

7.6.1 Introduction:

This section provides an overview of the project structure and introduces the team members who contributed to the development and implementation of the ADAS (Advanced Driver Assistance System) web interface. It outlines the key roles, responsibilities, and the collaborative workflow adopted to deliver an integrated, real-time system that leverages modern web technologies and smart vehicle protocols.

The site is designed to work perfectly on all devices from computers to smartphones, using Media Queries and advanced CSS techniques.

7.6.2 Portfolio Link

https://codeben-a1dd2.web.app

(*You can also scan the QR code below*)

7.6.3 Portfolio Contents

7.6.3.1 Home Page

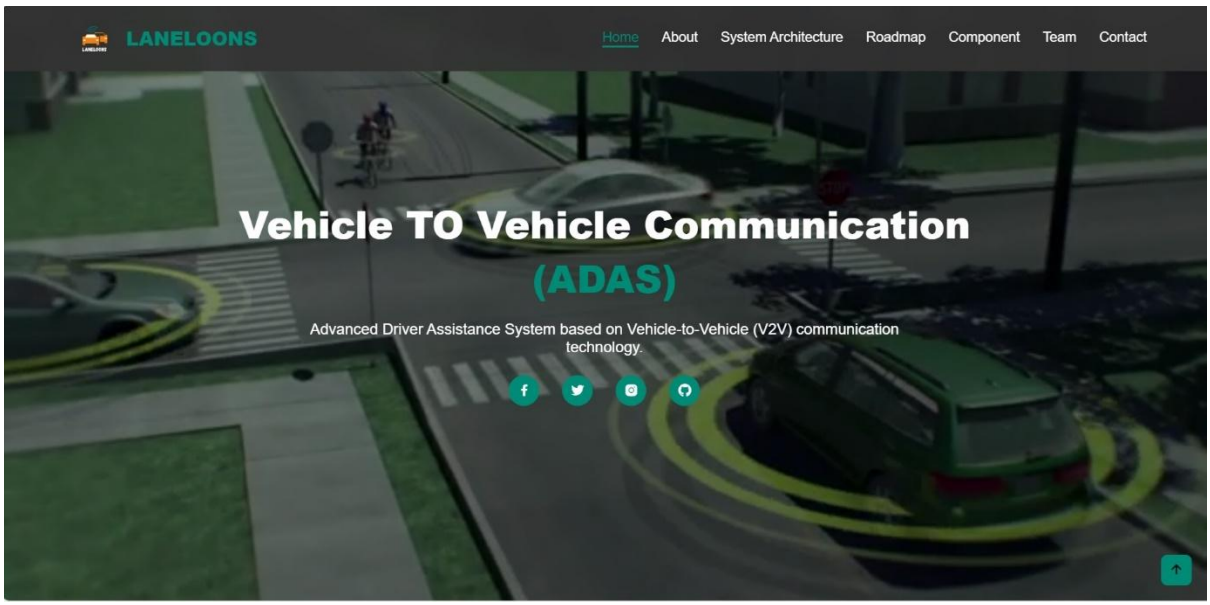


Figure 7.19 Home Page.

7.6.3.2 About Section

The **"About"** section provides a comprehensive overview of the project concept, which is built upon Vehicle-to-Vehicle (V2V) communication integrated within an Advanced Driver Assistance System (ADAS).

It highlights how the system enhances road safety and traffic efficiency by utilizing intelligent features.

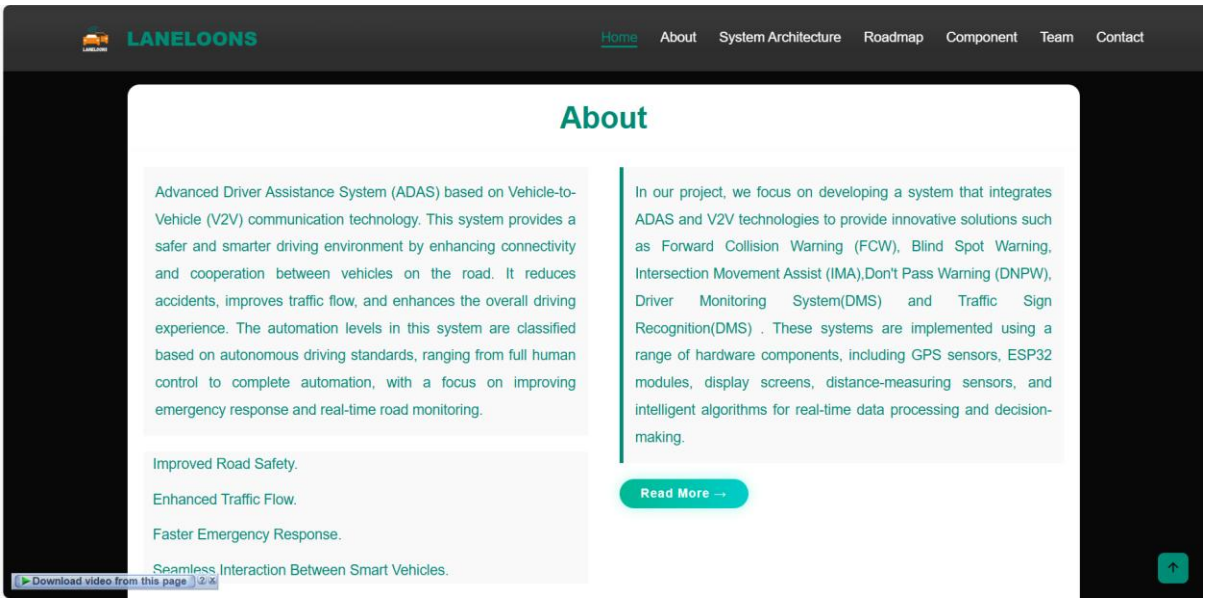


Figure 7.20 About Section.

7.6.3.3 Read More Page

This section provides a detailed explanation of the **main objective of the project**, which is to develop an intelligent Advanced Driver Assistance System (ADAS) based on Vehicle-to-Vehicle (V2V) communication technology. The goal is to enhance road safety and reduce traffic accidents.

Additionally, each **feature of the system is thoroughly explained**, including its functionality, operational mechanism, and its significance in improving the driving experience.

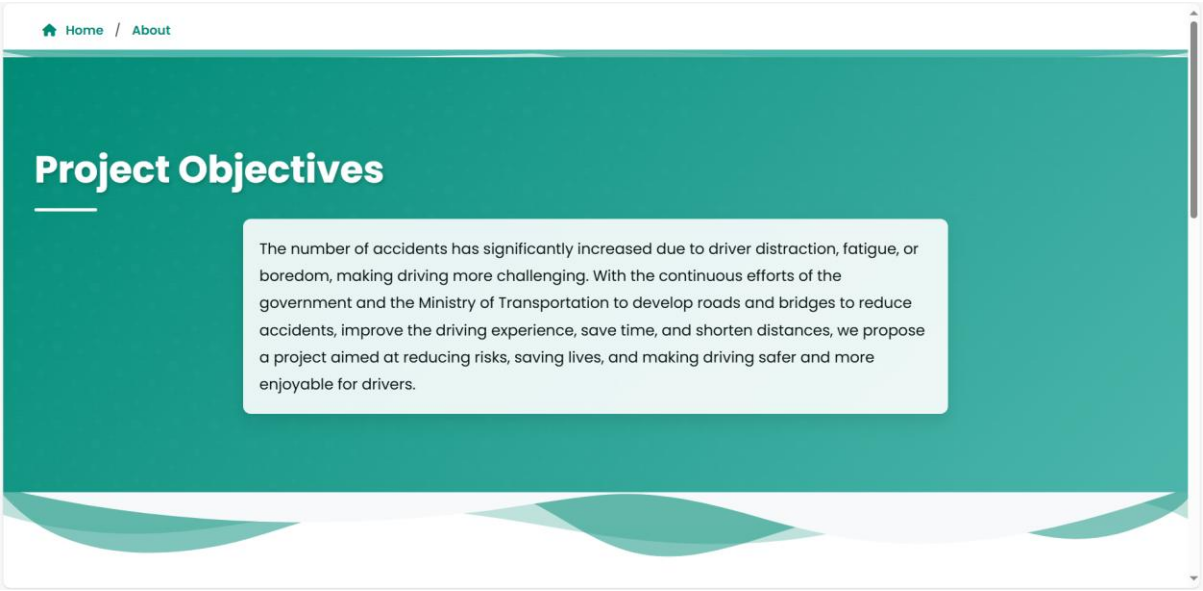


Figure 7.2 1.a: Read More (Objective) Page.

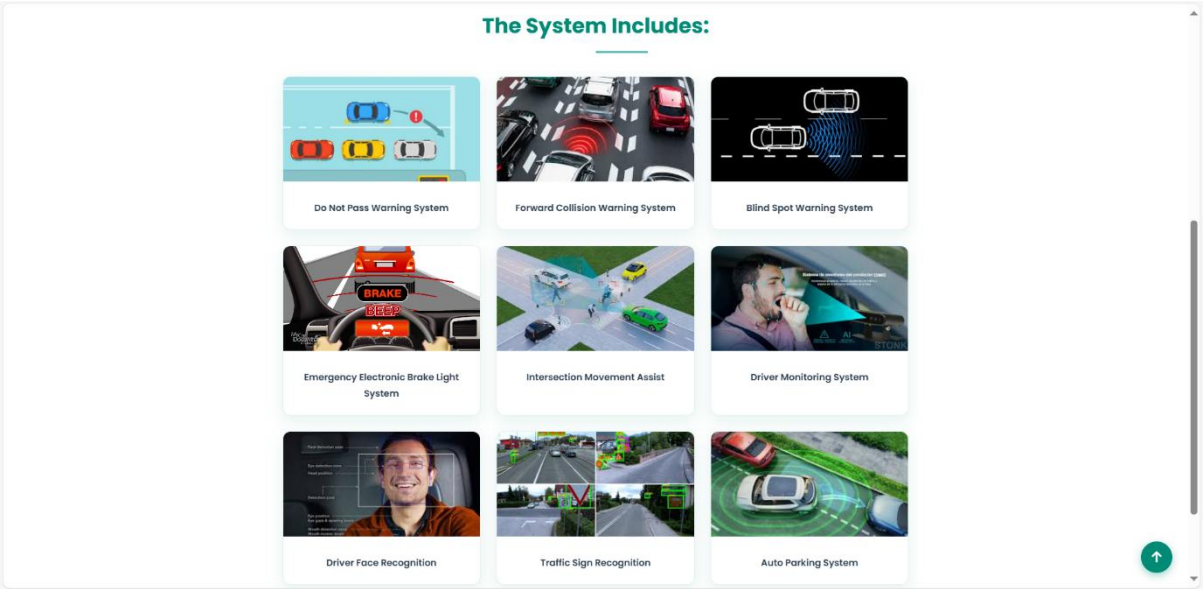


Figure 7.2 1.b: Read More (Objective) Page.

7.6.3.4 System Architecture Section

Main Component Diagram of the Smart Vehicle System for Communication, Control, and Real-Time Processing

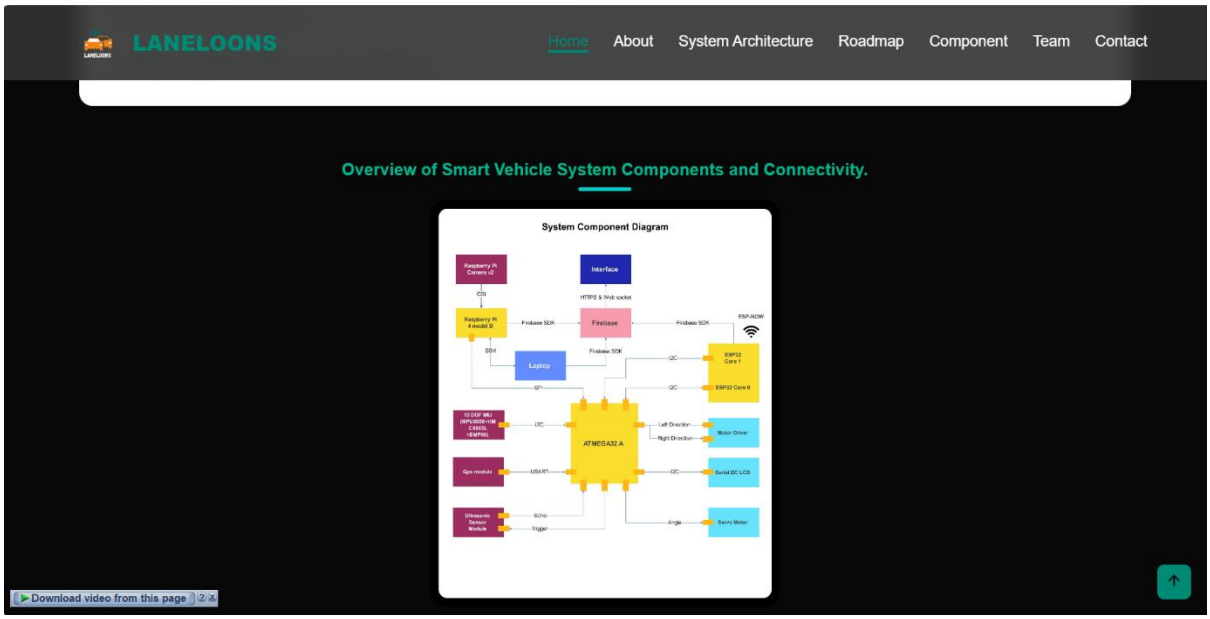


Figure 7.22 System Architecture section

7.6.3.5 RoadMap Section

This section presents the roadmap that outlines how the project was implemented through the collaboration of different technical tracks. Each track plays a vital role in the development of the system, with an emphasis on **the integration between these tracks** to deliver an intelligent and efficient solution.

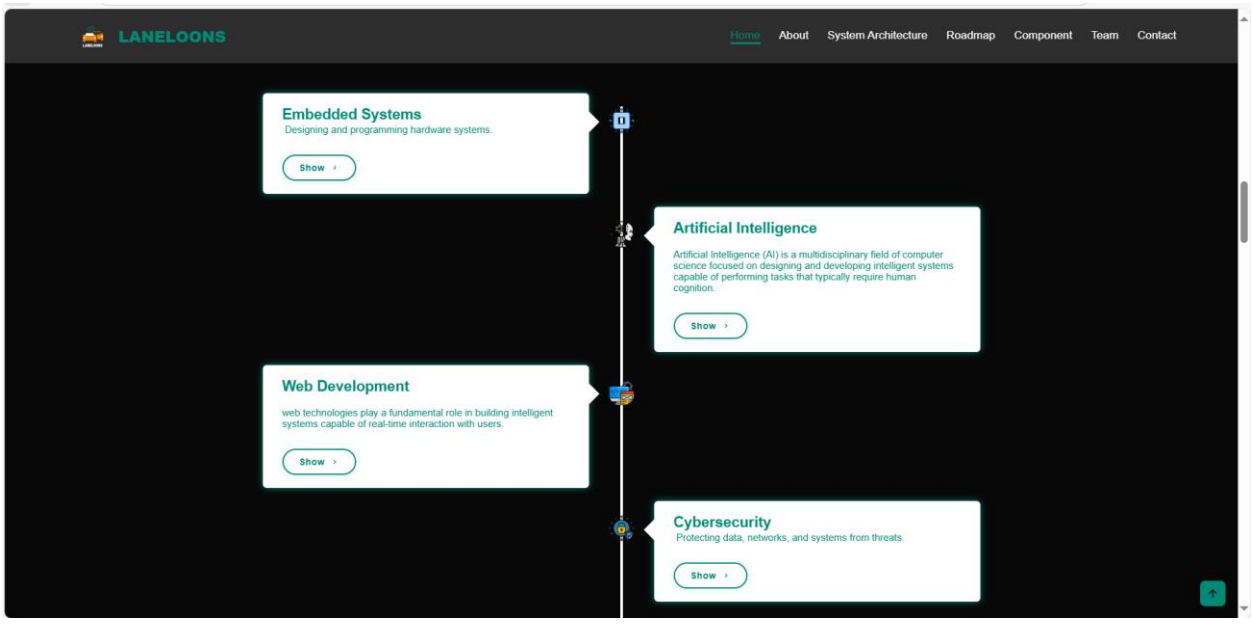


Figure 7.23 Road Map section.

1-Embedded system Features:

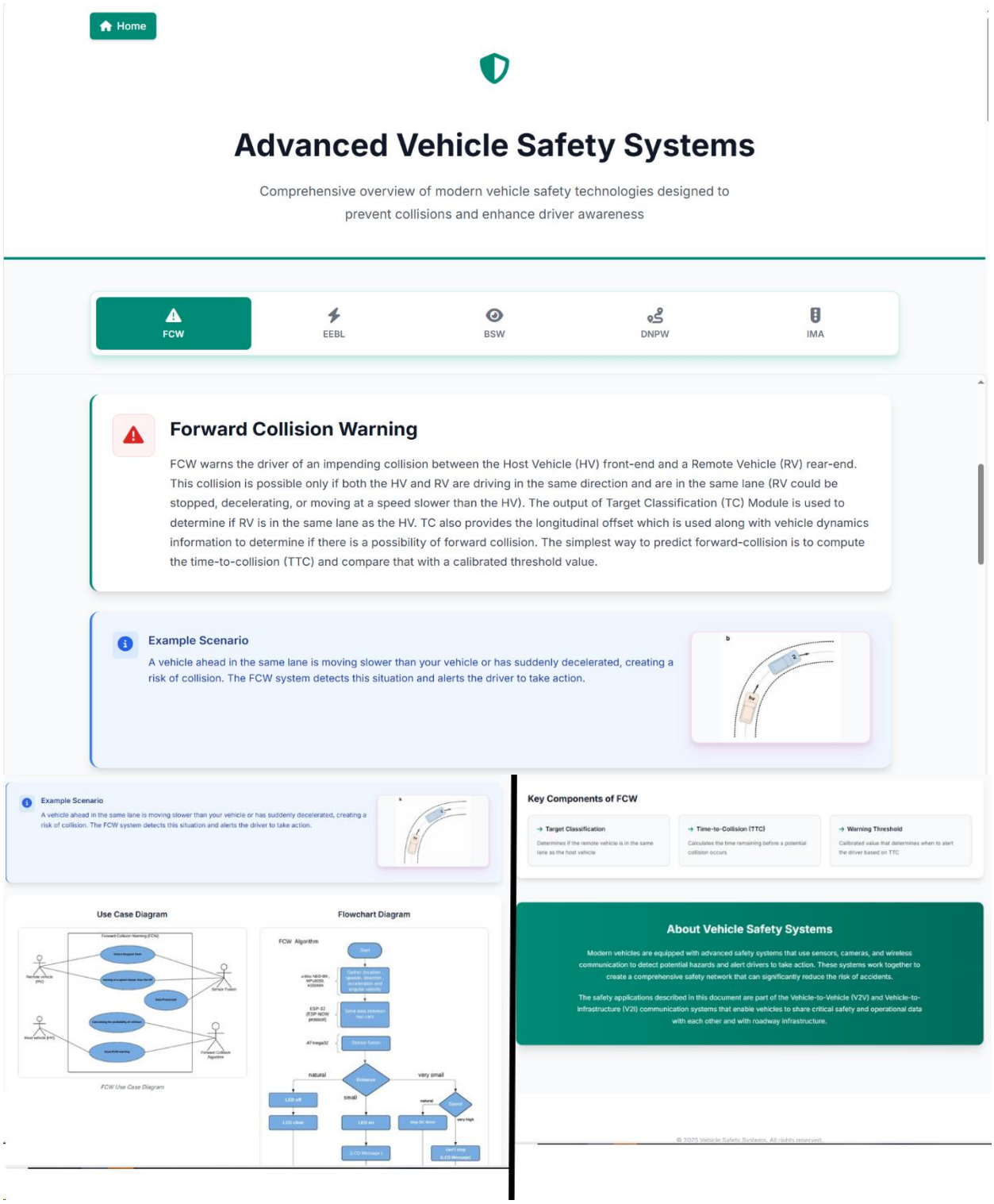


Figure 7.24 Embedded Feature Page.

2-Ai Feature:

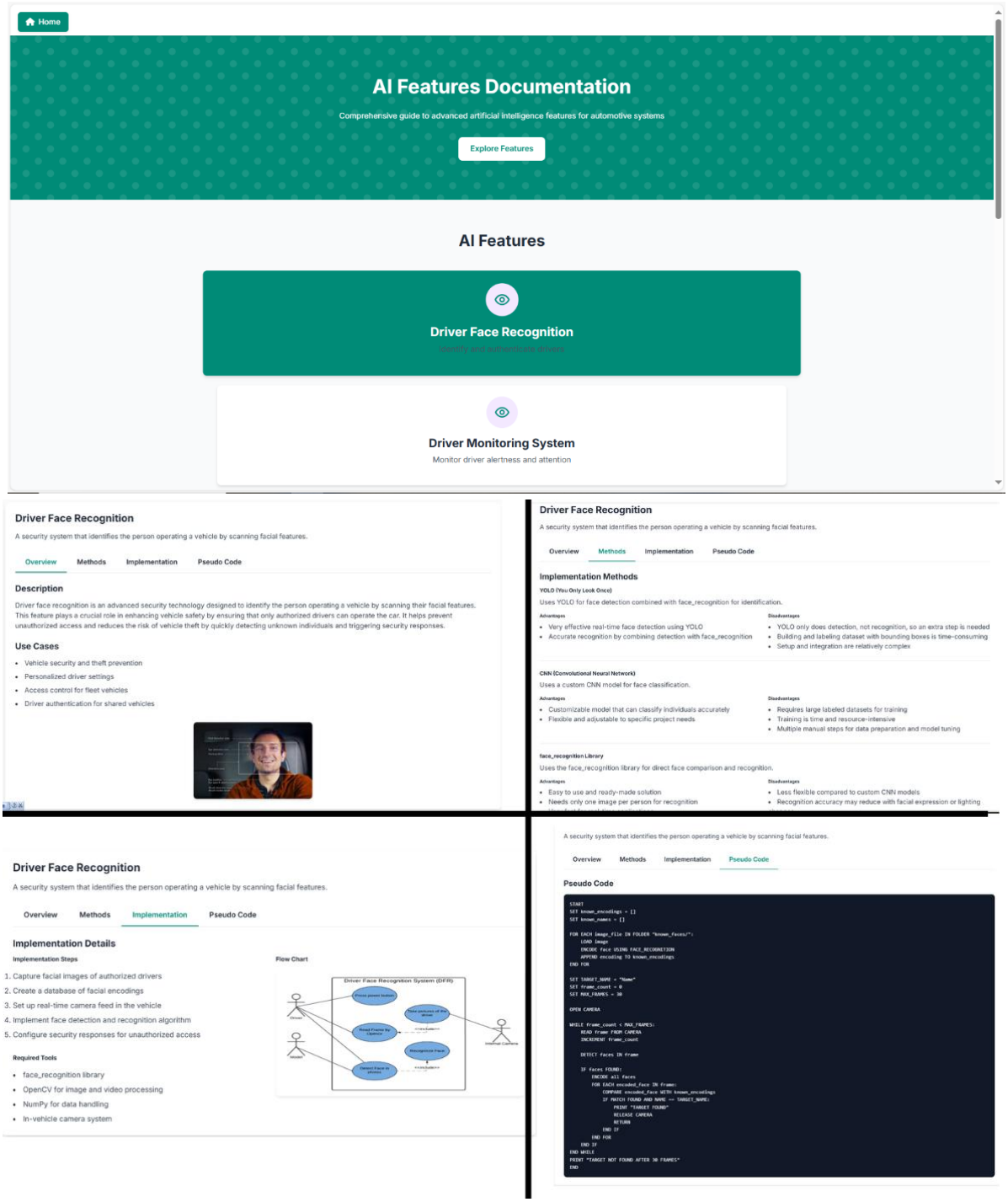


Figure 7.25AI Feature Page.

3-Web Technology:

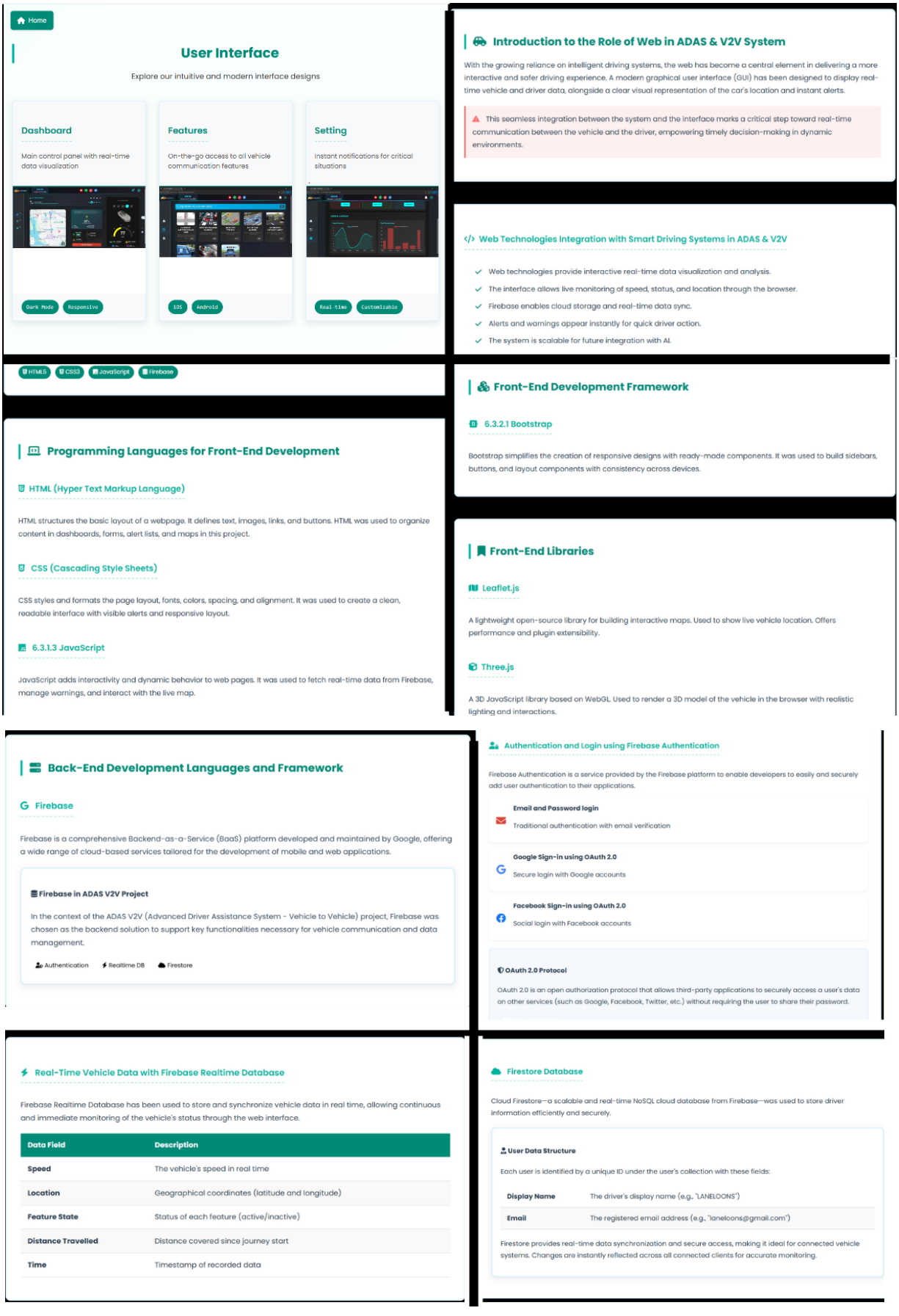


Figure 7.26 Web Technology Page.

7.6.3.6 Component Section

This section presents a list of the hardware and software components used in the implementation of the Advanced Driver Assistance System (ADAS) project. Each component is detailed in terms of its **name, image, function, and price**, offering a comprehensive view of the project's technical and practical aspects.

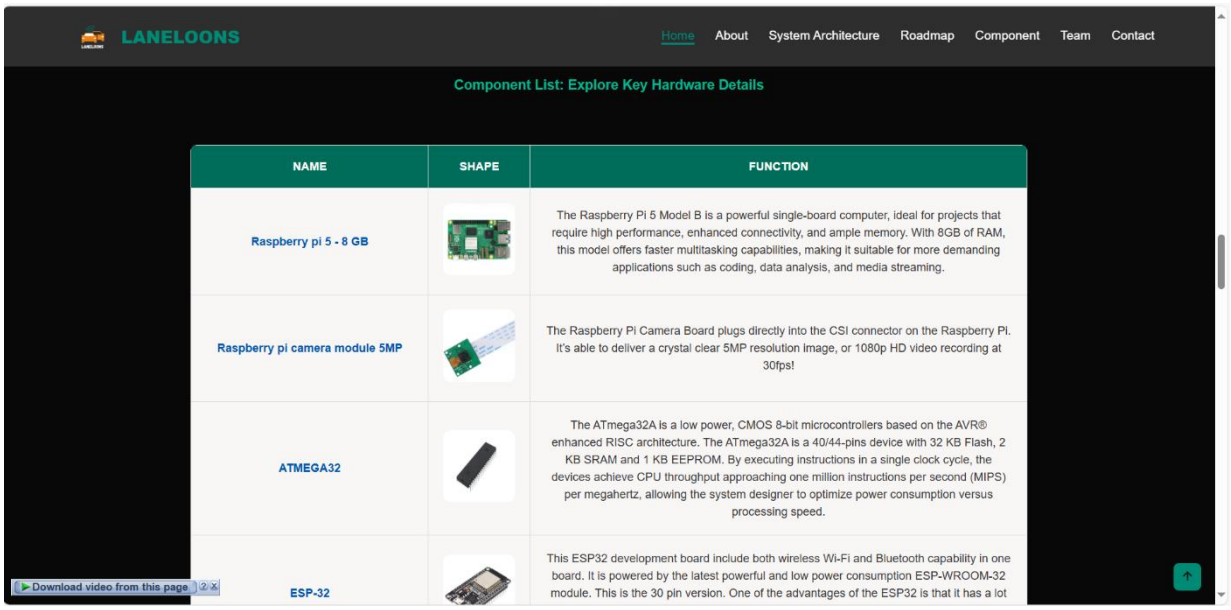


Figure 7.27 Component section

7.6.3.7 Team Section

This section showcases the team members behind the development and implementation of the **LaneLoons** project.

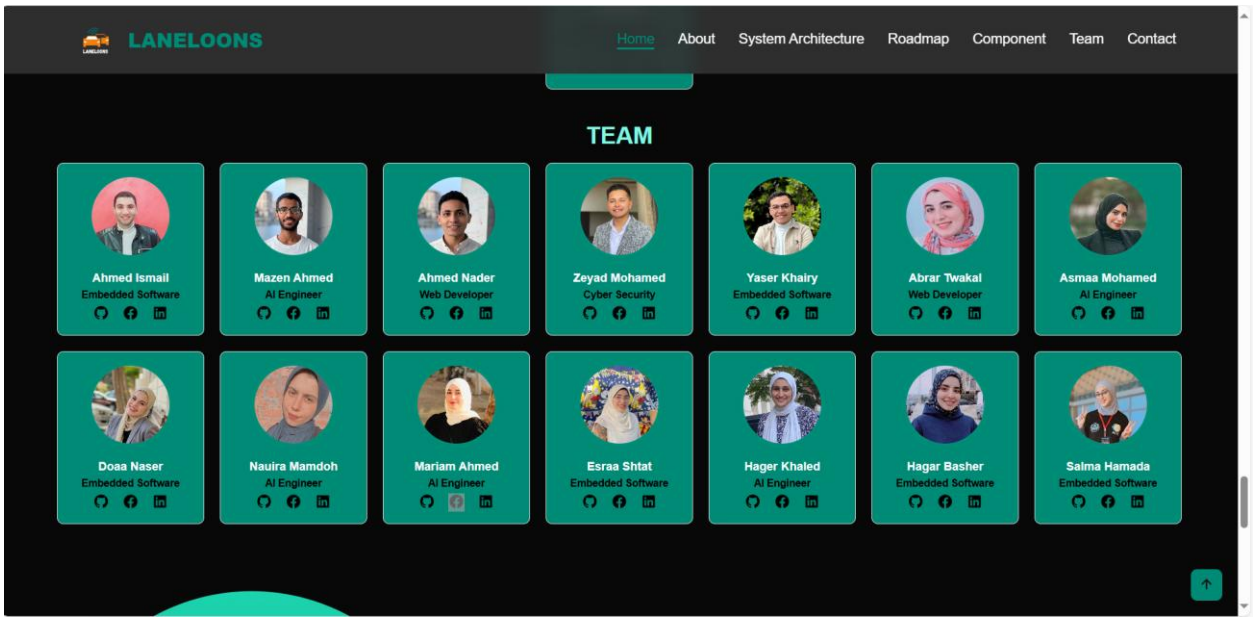


Figure 7.28 Team section.

7.6.3.8 Contact Section

The Contact Us section provides a direct and user-friendly way to get in touch with the LaneLoons project team.

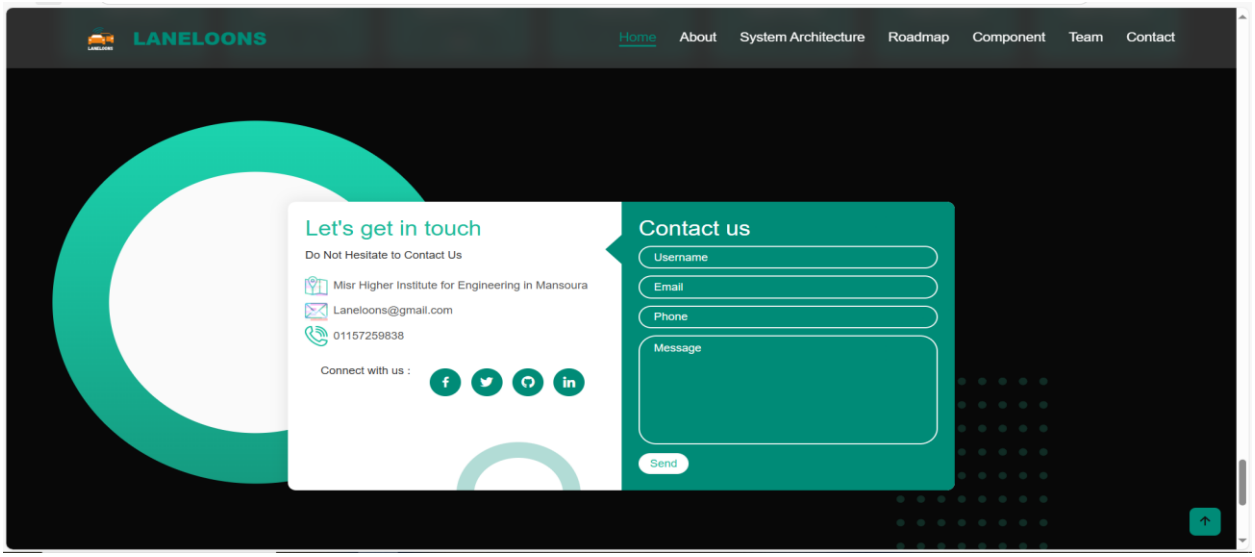


Figure 7.29 Contact Us section.

7.6.4 Technologies and Tools Used

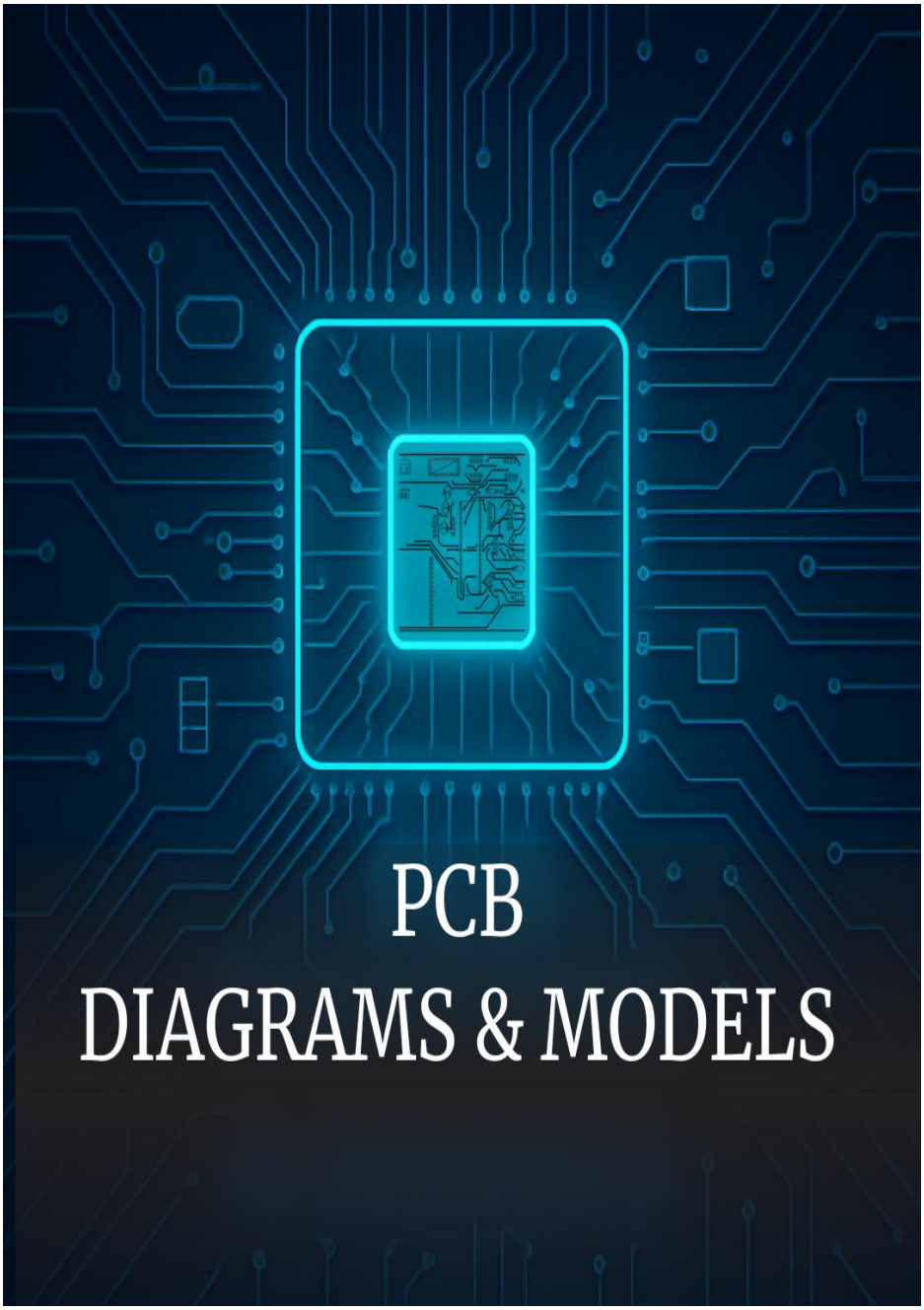
1. Frontend Technologies:

- HTML 5, CSS3, JavaScript.
- Box icons.
- Media Queries: Ensures the website is responsive across different screen sizes.
- **AOS (Animate on Scroll) library:** A JavaScript library that adds scroll animations to page elements. Used here to add "fade-up" effects to elements as they appear during scrolling.

2. Backend Technologies:

- **Web3Forms**: A simplified service for handling web forms without needing a backend server. Used to send contact form data directly to email.
- **Firebase**: Configured for data storage and analytics

Chapter 8



Chapter 8: PCB Diagrams & Models

8.1 PCB Diagrams

8.1.1 Overview

A PCB (Printed Circuit Board) is the backbone of our hardware project, providing a compact and organized platform to connect and power all electronic components efficiently. We designed a custom PCB to ensure reliable performance, reduce wiring complexity, and optimize the physical layout of our circuit for better functionality and durability.

8.1.2 Hardware Semantic

8.1.2.1 Top Layer

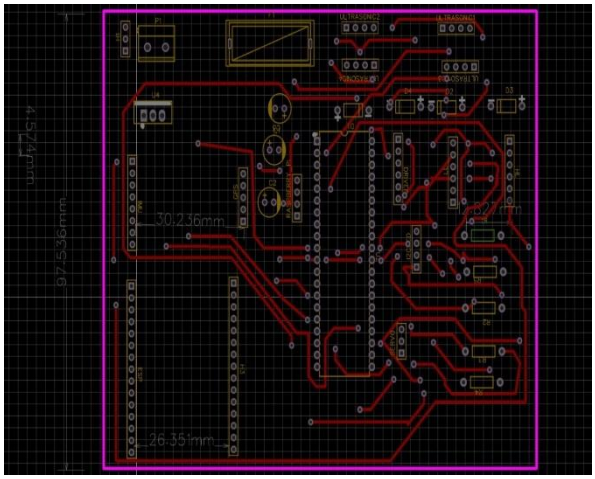


Figure 8.1.a PCB Top Layer.

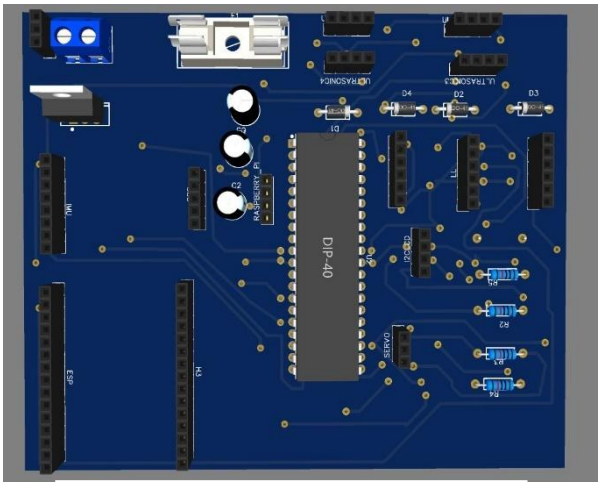


Figure 8.1.b PCB Top Layer.

8.1.2.2 Bottom Layer

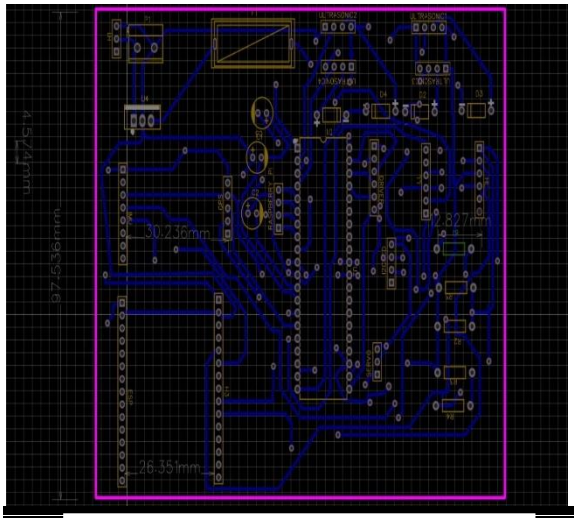


Figure 8.2.a PCB Bottom Layer.

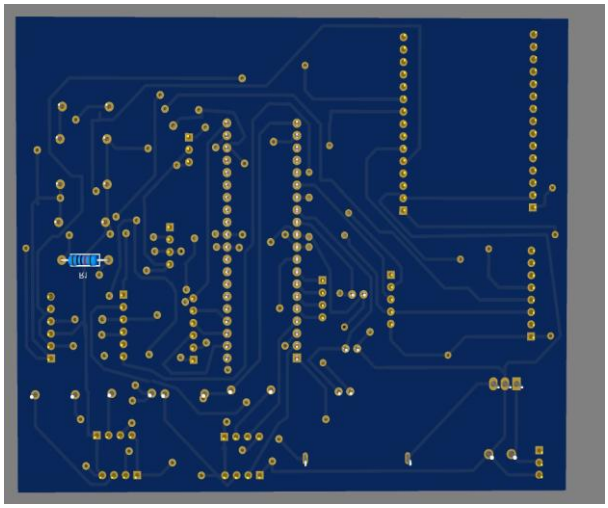


Figure 8.2.b PCB Bottom Layer.

8.1.2.3 Both Layers

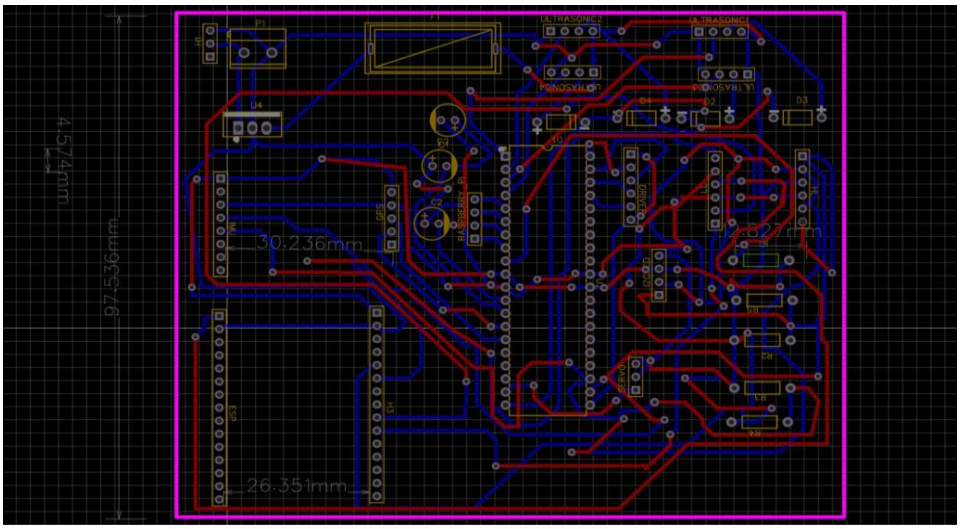


Figure 8.3 PCB Both Layer.

8.2 Models

8.2.1 Host vehicle

Figure 8.4 Real Host Vehicle.

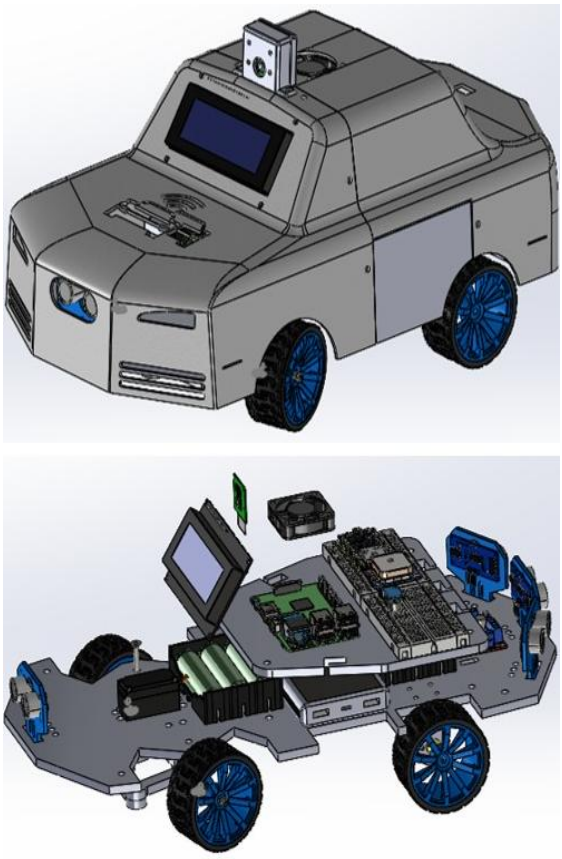


Figure 8.5 Host Vehicle Design.

8.2.2 Remote vehicle

Figure 8.6 Real Remote Vehicle.

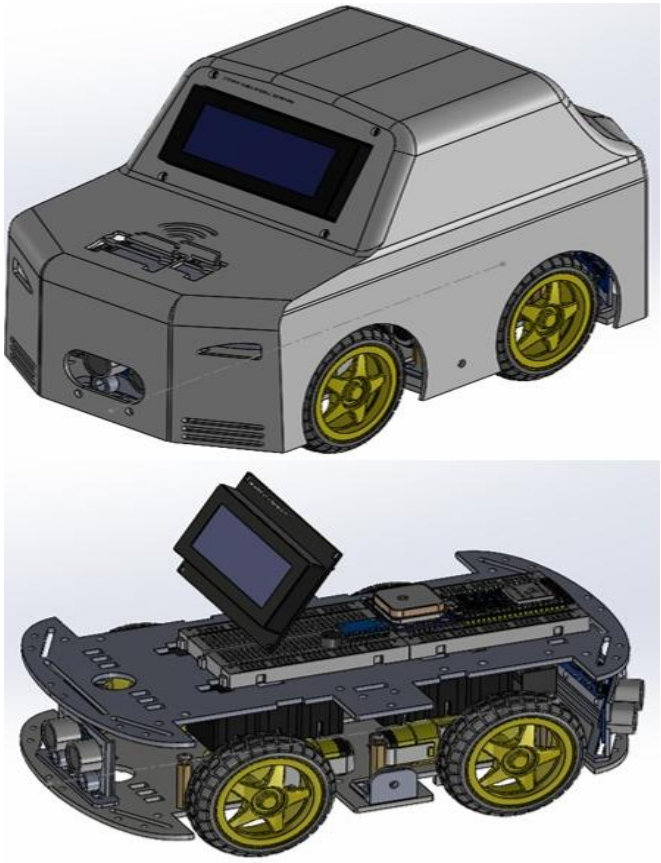
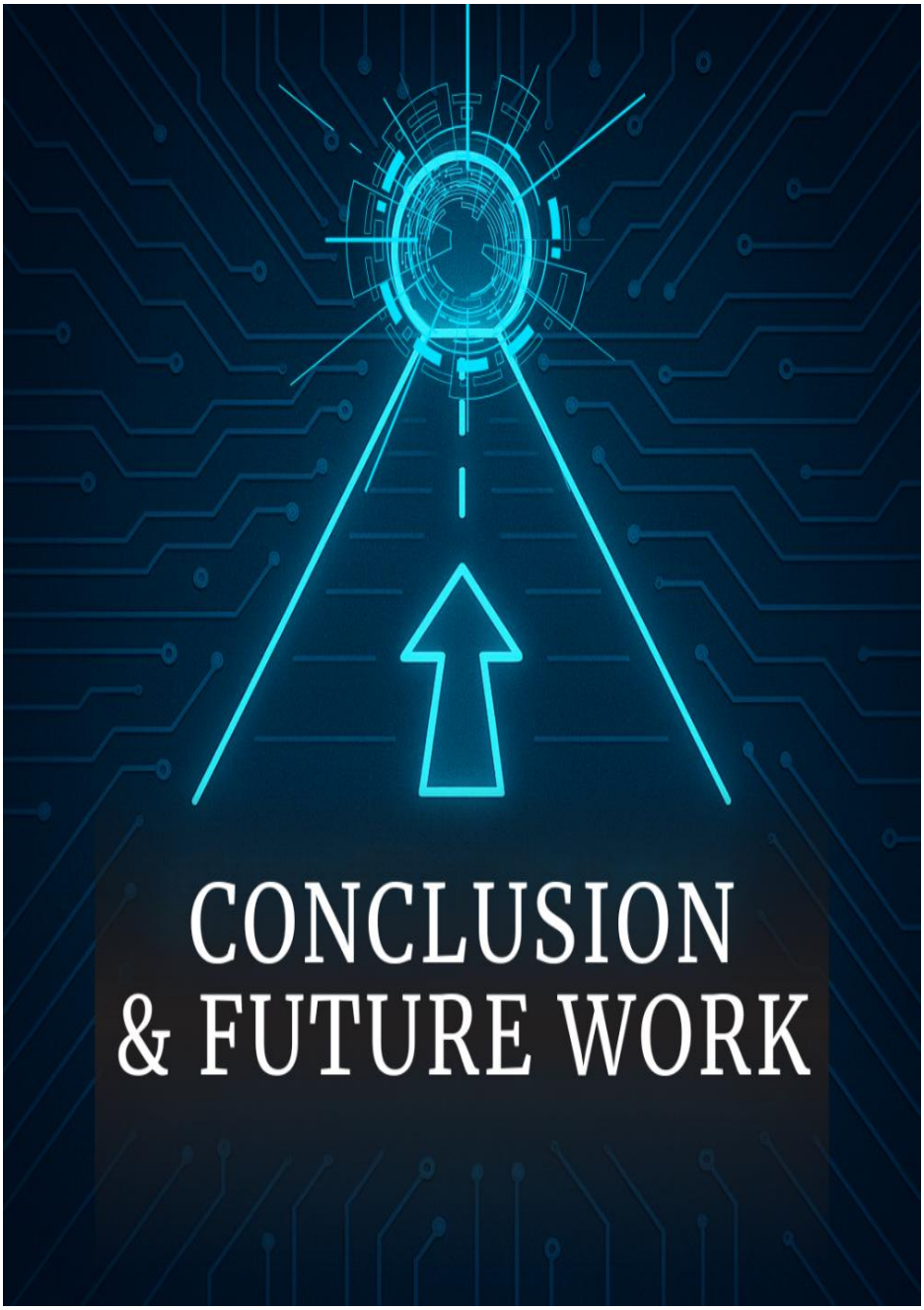


Figure 8.7 Remote Vehicle Design.

8.2.3 Maket

Figure 8.8 Maket.

Chapter 9



Chapter 9: Conclusion & Future Work

9.1 Towards Industrial-Grade V2V

- **ESP-NOW** was used as a low-cost educational platform for V2V communication, achieving <10ms latency and 98% packet delivery, meeting SAE J2945/1 standards.
- Future plans include:
  - Integration of **5G NR-V2X** for long-range communication.
  - Migration to **CRYSTALS-Kyber** post-quantum cryptography for enhanced security.
  - Submission of a standardization proposal to **ETSI ITS-G5**.

9.2 V2V Safety Applications

9.2.1 Lane-Change Warning (LCW)

- Warns the driver if a fast-approaching vehicle will soon enter the blind spot during a lane change.
- Uses **Time-To-Collision (TTC)** calculations and adjusts the warning zone based on speed and relative velocity.

9.2.2 Left Turn Assist (LTA)

- Supports safe left turns at intersections using **MAP** and **SPaT** messages from RSUs.
- Verifies the vehicle is in a valid lane and checks the signal status, alerting if an oncoming vehicle poses a risk.

9.2.3 Control Loss Warning (CLW)

- Alerts the Host Vehicle if a nearby vehicle activates systems indicating control loss (ABS, Traction, Stability).
- A warning is issued if the RV is within range and **TTC** is below a safe threshold.

9.3 Future Component Upgrades to Enhance System Precision

Future hardware upgrades aim to:

- Improve measurement accuracy.
- Reduce noise and variability.
- Increase environmental robustness.

These upgrades support system scalability and alignment with industry standards.

9.4 Transition to ARM-Based Microcontroller for Advanced System Capabilities

Moving to an **ARM-based MCU** to gain:

- Better processing performance and real-time response.
- Support for complex algorithms (e.g., signal filtering, machine learning).
- Lower power consumption and more integrated peripherals.
- Enhanced safety with features like memory protection and fault tolerance.

9.5 More Enhancement at the Integration Level

9.5.1 Driver Monitoring System (DMS) Integration

- APS will activate automatically if the DMS detects driver fatigue or inattention, performing emergency parking.

9.5.2 Advanced Sensor Fusion

- Combines data from cameras, LiDAR, and radar for a more accurate environmental view.

9.5.3 Real-Time Data Integration

- Uses GPS and V2X to make smarter parking decisions based on current road and traffic conditions.

9.5.4 Advanced AI Algorithms

- AI will analyze complex situations, improve decision-making, and learn over time for better performance.

9.5.5 Scalability and Flexibility

- The system will be moved to a more powerful platform to support future AI and automation needs.

References

References

[1] DSRC Technology Overview [Auto-Talks, 2021](https://auto-talks.com/technology/dsrc-technology/)

[2] Dedicated Short Range Communications Service [FCC, n.d.](https://www.fcc.gov/wireless/bureau-divisions/mobility-division/dedicated-short-range-communications-dsrc-service)

[2] Getting Started with ESP-NOW ESP32 Arduino IDE [Random Nerd Tutorials, 2022](https://randomnerdtutorials.com/esp-now-esp32-arduino-ide/)

[3] ESP NOW Peer to Peer ESP32 Communications [Dronebot Workshop, 2022](https://dronebotworkshop.com/esp-now/)

[4] Using ESP-NOW Protocol Part 1 [Circuit Cellar, n.d.](https://circuitcellar.com/research-design-hub/using-esp-now-protocol-part-1/)

[5] What Is Vehicle-to-Everything V2X [BlackBerry QNX, 2024](https://www.blackberry.com/us/en/solutions/iot/qnx-software-solutions/qnx-platform-for-adas)

[6] Vehicle-to-everything Wikipedia Page [Wikipedia, 2024](https://en.wikipedia.org/wiki/Vehicle-to-everything)

[7] Enhancing Security Vehicle-to-Vehicle Communication [MDPI, 2024](https://www.mdpi.com/2076-3417/14/9/3738)

[8] SAE International, “Dedicated Short Range Communications (DSRC) Message Set Dictio nary” SAE J2735 Standard, March 2016.

[9] Society of Automotive Engineers, SAE J2735 Dedicated Short Range Communications (DSRC) Message Set Dictionary J2735_201603, PA SAE, Mar. 30, 2016

[10] Kalman, R. E., “A New Approach to Linear Filtering and Prediction Problems,” Trans. ASME- J. Basic Eng. 35–45, March 1960.

[11] The Analytic Sciences Corporation, “Applied Optimal Estimation,” edited by Gelb, Arthur, Cambridge, MA: MIT Press, 1974.

[12] Connected Vehicles Intelligent Transportation Systems , Radovan Miucic Editor

[13] MicroC/OS-II: The Real Time Kernel, By Jean Labrosse

[14] Recognize and manipulate faces using **face_trcognition** library:

a) https://github.com/ageitgey/face_recognition

b) https://basilchackomathew.medium.com/face-recognition-in-python-a-comprehensive-guide-960a48436d0f

[15] Lightweight Face Recognition Framework - **DeepFace**:

a) https://github.com/serengil/deepface

[16] **DeepFace**: A Popular Open Source Facial Recognition Library

a) https://viso.ai/computer-vision/deepface/

[17] Real-Time Face Detection using **YOLOv11**:

a) https://github.com/ultralytics/ultralytics

[18] **You Only Look Once**: Unified, Real-Time Object Detection:

a) https://arxiv.org/abs/1506.02640

[19] **FaceNet**: A Unified Embedding for Face Recognition and Clustering:

a) https://arxiv.org/abs/1503.03832

[20] **Raspberry Pi 5** Guide:

a) https://wagnerstechtalk.com/rpi5/

[21] About **Raspberry pi camera modules**:

a) https://www.raspberrypi.com/documentation/accessories/camera.html

[22] High Quality Face Recognition with Deep Metric Learning - **Dlib:**

a) https://dlib.net/

b) https://blog.dlib.net/2017/02/high-quality-face-recognition-with-deep.html

c) https://github.com/davisking/dlib

[23] **A Guide of Best Practices for Python:**

a) https://gist.github.com/ruimaranhao/4e18cbe3dad6f68040c32ed6709090a3

b) https://peps.python.org/pep-0008/

c) https://dev.to/devasservice/python-best-practices-writing-clean-efficient-and-maintainable-code-34bj

[23] **Eye Aspect Ratio (EAR):**

a) https://medium.com/analytics-vidhya/eye-aspect-ratio-ear-and-drowsiness-detector-using-dlib-a0b2c292d706

b) https://arxiv.org/abs/2408.05836

[24] SAE International, "Dedicated Short Range Communications (DSRC) Message Set Dictio nary" SAE J2735 Standard, March 2016.

[25] ISO/TC204/SC/WG18, Intelligenttransport systems —CooperativeSystems—SPAT (Signal Phase and Timing) message, MAP (Intersection topology) message, Draft versions 2013

[26] W3Schools, HTML, CSS, and JavaScript Tutorials. [Online].
Available: https://www.w3schools.com.

[26] Leaflet, Open-Source JavaScript Library for Interactive Maps. [Online].
Available: https://leafletjs.com.

[26] Google Firebase, Backend-as-a-Service Platform. [Online].
Available: https://firebase.google.com.

[26] Three.js, 3D JavaScript Library for Web Graphics. [Online].
Available: https://threejs.org.

[26] MDN Web Docs, Web Technologies Reference. [Online].
Available: https://developer.mozilla.org.

[26] Web3Forms, Static Form Handling Service. [Online].
Available: https://web3forms.com.

النتائج المتوقعة:

من المتوقع أن يسهم النظام في تقليل الحوادث الناتجة عن الخطأ البشري، وتحسين قدرة السائق على اتخاذ القرار في الوقت المناسب، مع إمكانية دمج النظام في المركبات الحديثة أو تطوير وحدات خارجية يمكن إضافتها إلى السيارات التقليدية.

الملخص العربى

تحسين السلامة على الطرق ومنع الحوادث باستخدام الذكاء الاصطناعي وإنترنت الأشياء من خلال أنظمة ADAS وتواصل المركبة مع المركبة (V2V)

مقدمة:

في ظل ارتفاع معدلات الحوادث المرورية وما يصاحبها من خسائر بشرية ومادية جسيمة، تبرز الحاجة إلى أنظمة ذكية تدعم السائق في اتخاذ قرارات أكثر أمانًا أثناء القيادة. يستهدف هذا المشروع تطوير نظام متكامل يجمع بين تقنيات الذكاء الاصطناعي (AI) وإنترنت الأشياء (IoT) من جهة، وأنظمة القيادة المساعدة المتقدمة (ADAS) وتواصل المركبات (V2V) من جهة أخرى، بهدف الحد من الحوادث وتحسين تجربة القيادة.

فكرة المشروع:

يعتمد النظام على تبادل البيانات بين المركبات في الزمن الحقيقي باستخدام بروتوكول ESP-NOW، ما يسمح لكل مركبة بالحصول على معلومات دقيقة عن سرعة واتجاه وموقع المركبات المجاورة. تُستخدم هذه البيانات لتفعيل وحدات ADAS التي تقوم بتنبيه السائق أو التدخل المباشر عند اكتشاف خطر وشيك، مما يرفع من كفاءة التفاعل مع البيئة المحيطة.

المكونات والوظائف:

يشمل المشروع مجموعة من الأنظمة الفرعية:

- أنظمة التحذير مثل التصادم الأمامي (FCW) والنقطة العمياء (BSW) ومنع التجاوز (DNPW).
- مساعد التقاطعات (IMA) وتنبيه الكبح المفاجئ (EEBL) .
- نظام مراقبة حالة السائق باستخدام كاميرا داخلية للكشف عن النعاس أو عدم الانتباه.
- نظام التعرف على إشارات المرور باستخدام خوارزميات الرؤية الحاسوبية (مثل YOLO وCNN).
- واجهة رسومية GUI تعرض للسائق التنبيهات بشكل تفاعلي لحظي.

البنية التقنية:

تم تنفيذ النظام باستخدام وحدات ESP32 وATmega32 وRaspberry Pi 5، إلى جانب مستشعرات GPS وIMU وكاميرات رقمية. يتم الربط بين المكونات عبر بروتوكولات I2C وSPI، ويعتمد النظام على تشفير AES-128 لتأمين الاتصالات وضمان خصوصية البيانات.

معهد مصر العالي للهندسة والتكنولوجيا
بالمنصورة

قسم الاتصالات وقسم هندسة
الكمبيوتر

تحسين السلامة على الطرق ومنع الحوادث من خلال أنظمة ADASوتواصل المركبة مع المركبة (V2V) المدعّمة بالذكاء الاصطناعي وإنترنت الأشياء(IoT)

مشروع مُقدَّم كجزء من متطلبات الحصول على درجة

بكالوريوس العلوم في هندسة الحاسبات

بواسطة:

أحمد محمد إسماعيل
أبرار توكل محمد
أسماء محمد فتحي
أحمد نادر محمد
إسراء السيد شتات
دعاء ناصر أحمد صحصاح
هاجر خالد سعد
هاجر بشير عيد
مازن أحمد العساس
مريم أحمد اليماني
سلمي حماده داوود
نيره ممدوح سلامة
ياسر خيري الجعفري
زياد محمد عبد الحميد

المشرف

الأستاذ المساعد الدكتور/ياسر حامد أحمد العوضي

قسم هندسة الاتصالات والحاسبات

معهد مصر العالي للهندسة والتكنولوجيا

٢٠٢٥