A

Project Report on

“Vehicle speed control and accident avoidence system using CAN communication”

Submitted in the partial fulfillment of the requirements for the PG Diploma in

EMBEDDED SYSTEMS & DESIGN (PG - DESD)

by

Mayur Maruti Patil	250844230054
Aishwarya Dadasaheb Amale	250844230009
Poonam Sundar Puri	250844230064
Anuj Santosh Palkar	250844230053

Sunbeam Institute of Information Technology, Pune & Karad
Year: Aug – 2025

Sunbeam Institute of Information Technology, Pune & Karad.

CERTIFICATE

This is to certify,

Mayur Maruti Patil	250844230054
Aishwarya Dadasaheb Amale	250844230009
Poonam Sundar Puri	250844230064
Anuj Santosh Palkar	250844230053

have satisfactorily completed Project and presented a report on topic titled "**Vehicle speed control and accident avoidence system using CAN communication**" at Sunbeam Institute of Information Technology in partial fulfilment of requirement of PG Diploma in Embedded Systems & Design (PG-DESD) academic year 2025.

Date: 31/01/2026

ACKNOWLEDGEMENT

This project, "**Vehicle speed control and accident avoidence system using CAN communication"** was a great learning experience for us, and we are submitting this work to Sunbeam Institute of Information Technology. We are very grateful to **Ms. Utkarsha Ma'am** for her valuable guidance throughout the project. Her continuous support and encouragement helped us overcome various challenges and technical difficulties during the course of the project work.

We would also like to express our sincere thanks to **Mr. Devendra Dhande (Course Coordinator, PG-DESD)** for providing all the necessary support and coordination, including access to required hardware, internet facilities, and extended lab hours, which were essential for the successful completion of this project and throughout the course duration at Sunbeam Institute of Information Technology, Pune.

INDEX

ABSTRACT

Road accidents are a major concern due to over-speeding, delayed driver response, and lack of real-time safety feedback. This project presents a **Vehicle Speed Control and Accident Avoidance System** implemented using the **STM32F407G microcontroller** and **CAN (Controller Area Network) communication**. The system uses a **dual-node architecture (TX and RX nodes)** connected through **MCP2551 CAN transceivers**.

An **ultrasonic sensor (HC-SR04)** and an **IR obstacle sensor** are used to detect obstacles and unsafe proximity conditions. Based on sensor inputs, the TX node transmits safety messages over the CAN bus. The RX node receives these messages and controls vehicle speed using an **L298N motor driver** and **BO DC motor**. A **buzzer** provides audible alerts during hazardous conditions.

The system demonstrates reliable, real-time communication suitable for automotive applications and effectively reduces vehicle speed or stops the motor to avoid accidents

1. INTRODUCTION

About Project:

Modern road transportation systems face increasing safety challenges due to rising vehicle density, higher speeds, and human-related factors such as delayed reaction time, distraction, and poor judgment in critical situations. Over-speeding and failure to respond promptly to obstacles are among the leading causes of road accidents, especially in urban and highway environments. To overcome these limitations, intelligent electronic systems capable of real-time sensing, decision-making, and control are essential.

With advancements in embedded systems, microcontrollers, and automotive communication protocols, it has become possible to design compact and efficient safety systems that assist the driver and, in certain conditions, take automatic corrective actions. Modern vehicles rely heavily on Electronic Control Units (ECUs) that continuously monitor vehicle parameters and environmental conditions to improve safety, reliability, and performance.

This project, **Vehicle Speed Control and Accident Avoidance System using STM32F407G and CAN Communication**, focuses on developing a prototype-level intelligent safety system that can detect obstacles and control vehicle speed automatically. The STM32F407G microcontroller is used due to its high processing capability, real-time performance, and built-in CAN controller, making it suitable for automotive applications.

The system adopts a distributed architecture consisting of a Transmitter (TX) node and a Receiver (RX) node interconnected through the **Controller Area Network (CAN)** protocol. The TX node collects data from sensors such as the ultrasonic sensor and IR sensor to detect obstacles and unsafe proximity conditions. Based on this sensor data, the TX node generates safety commands and transmits them over the CAN bus.

The RX node receives these CAN messages and performs appropriate control actions, such as reducing motor speed or stopping the vehicle using the L298N motor driver. Audible alerts are generated using a buzzer to warn the driver during dangerous situations. By using CAN communication, the system ensures reliable, fast, and priority-based data transmission, which is crucial in safety-critical automotive environments.

Overall, this project demonstrates how embedded systems and CAN-based communication can be effectively used to enhance vehicle safety by minimizing human dependency and improving reaction time during potential accident scenarios.

Modern vehicles increasingly rely on electronic control units (ECUs) and real-time communication to enhance safety. Over-speeding and delayed reaction to obstacles are major causes of accidents. By integrating sensors, microcontrollers, and a robust communication protocol, intelligent speed control and accident avoidance can be achieved.

This project focuses on using **CAN communication**, which is widely adopted in automotive systems for its reliability, priority-based messaging, and noise immunity. The STM32F407G microcontroller processes sensor data and communicates safety commands between nodes to control vehicle speed and generate alerts.

1.1 Project Scope

- Real-time obstacle detection using ultrasonic and IR sensors
- Vehicle speed control based on obstacle distance
- CAN-based communication between transmitter and receiver node
- Motor speed control using PWM
- Audible warning generation using a buzzer
- Modular and scalable system design

1.1 Project scope:

The scope of the *Vehicle Speed Control and Accident Avoidance System using STM32F407G and CAN Communication* is to develop a prototype embedded safety system that controls vehicle speed and helps prevent accidents through real-time obstacle detection. The system uses ultrasonic and IR sensors to continuously monitor the vehicle's surroundings and detect unsafe proximity conditions.

The STM32F407G microcontroller processes sensor data and automatically determines appropriate control actions based on predefined safety thresholds. When a potential collision risk is detected, the system reduces vehicle speed or stops the motor without requiring driver intervention. Safety commands are transmitted between two microcontroller nodes using CAN communication, simulating real automotive ECU interaction.

The project also provides audible alerts using a buzzer to warn the driver during hazardous situations. Although implemented as a prototype using basic sensors and a DC motor, the system architecture is modular and scalable, allowing future integration with automotive-grade sensors and advanced braking systems.

1.2 CAN Protocol:

The CAN bus was developed by BOSCH as a multi-master, message broadcast system that specifies a maximum signaling rate of 1 megabit per second (bps). Unlike a traditional network such as USB or Ethernet, CAN does not send large blocks of data point-to-point from node A to node B under the supervision of a central bus master. In a CAN network, many short messages like temperature or RPM are broadcast to the entire network, which provides for data consistency in every node of the system. CAN is an International Standardization Organization (ISO)-defined serial communications bus originally developed for the automotive industry to replace the complex wiring harness with a two-wire bus.

The specification calls for high immunity to electrical interference and the ability to self-diagnose and repair data errors.

CAN Frame Format:

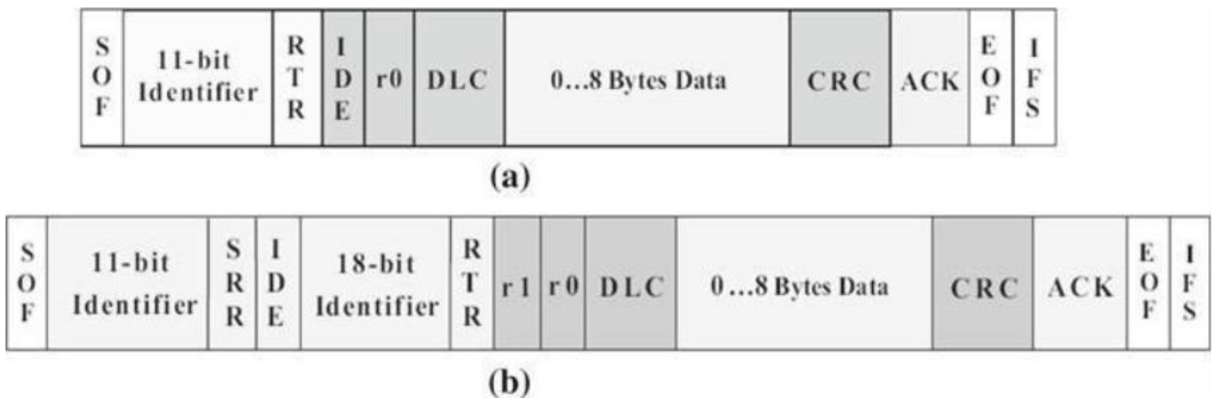


Fig 1.1 CAN Frame Format

The following figure shows the CAN protocols' frame format.

- SOF: The single dominant start of frame (SOF) bit marks the start of a message and is used to synchronize the nodes on a bus after being idle.
- Identifier: The standard CAN 11-bit identifier establishes the priority of the message. The lower the binary value, the higher its priority.
- RTR: The single remote transmission request (RTR) bit is dominant when information is required from another node. All nodes receive the request, but the identifier determines the specified node. The responding data is also

received by all nodes and used by any node interested. In this way, all data being used in a system is uniform.

- IDE: A dominant single identifier extension (IDE) bit means that a standard CAN identifier with no extension is being transmitted.
- ro: Reserved bit (for possible use by future standard amendment).
- DLC: The 4-bit data length code (DLC) contains the number of bytes of data being transmitted.
- Data: up to 64 bits of application data may be transmitted.
- CRC: The 16-bit (15 bits plus delimiter) cyclic redundancy check (CRC) contains the checksum (number of bits transmitted) of the preceding application data for error detection.
- ACK: Every node receiving an accurate message overwrites this recessive bit in the original message with a dominant bit, indicating an error-free message has been sent. Should a receiving node detect an error and leave this bit recessive, it discards the message, and the sending node repeats the message after re-arbitration. In this node acknowledges (ACK) the integrity of its data. ACK is 2 bits; one is the acknowledgement bit and the second is a delimiter. way, each
- EOF: This end-of-frame (EOF), 7-bit field marks the end of a CAN frame (message) and disables bit-stuffing, indicating a stuffing error when dominant. When 5 bits of the same logic level occur in succession during normal operation, a bit of the opposite logic level is stuffed into the data.
- IFS: This 7-bit inter-frame space (IFS) contains the time required by the controller to move a correctly received frame to its proper position in a message buffer area.

The Extended CAN Frame format consists of these additional bits:

- SRR: The substitute remote request (SRR) bit replaces the RTR bit in the standard message location as a placeholder in the extended format.
- IDE-A recessive bit in the identifier extension (IDE) indicates that more identifier bits follow. The 18-bit extension follows IDE.
- r1: Following the RTR and r0 bits, an additional reserve bit has been included ahead of the DLC bit.

System Requirements

Hardware requirements are as follows:

- STM32F407G Discovery Board (TX Node)
- STM32F407G Discovery Board (RX Node)
- MCP2551 CAN Transceivers (2 Nos.)
- Ultrasonic Sensor (HC-SR04)
- IR Obstacle Sensor Module
- L298N Motor Driver Module
- BO DC Motor
- Buzzer
- Battery / Power Supply
- Connecting Wires and Breadboard

Software requirements are as follows:

- STM32CubeIDE
- STM32 HAL Drivers
- Embedded C Language
- CAN Peripheral Drivers
- PWM and Timer Configuration

2. HARDWARE REQUIREMENTS

1. STM32F407 Discovery Board

The STM32F407G-DISC1 Discovery Kit is designed to help both beginners and experienced users explore the features of the STM32F407/417 line and develop applications. It's based on the STM32F407VGT6 microcontroller and includes an embedded ST-LINK/V2 debug tool, MEMS sensors, an audio DAC, LEDs, push buttons, and a USB OTG micro-AB connector.

Key Features and Specifications:

- Microcontroller: STM32F407VGT6 featuring a 32-bit ARM Cortex-M4F core, 1 MB Flash, and 192 KB RAM in an LQFP100 package. The core runs at up to 168 MHz and includes a floating-point unit (FPU) and Digital Signal Processing (DSP) instructions for high-performance embedded applications.
- On-board ST-LINK/V2: With a selection mode switch, it can be used as a standalone ST-LINK/V2 with an SWD connector for programming and debugging. Newer boards use ST-LINK/V2-A.
- Power Supply: Can be powered through a USB bus or an external 5 V supply voltage. It also provides an external application power supply at 3 V and 5 V.
- MEMS: Includes a LIS302DL or LIS3DSH ST MEMS 3-axis accelerometer and an MP45DT02 ST MEMS audio sensor, which is an omnidirectional digital microphone.
- LEDs: Eight LEDs are present: LD1 (red/green) for USB communication, LD2 (red) for 3.3 V power on, four user LEDs (orange, green, red, and blue)
- Push Buttons: It has two push buttons for user input and reset.
- USB OTG FS: Comes with a micro-AB connector.
- Debug and Programming: The ST-LINK circuitry allows you to flash code onto the microcontroller and debug it using host software like Keil or STM Cube IDE.

The board comes in two versions, a newer one labeled "DISC1" and an older one without this designation. The board identification number MB997D is for the newer board, while MB997C is for older versions.

The STM32F4 Discovery board allows users to develop and design applications, offering modules for communication and interface design without relying on third-party devices. It incorporates modern system modules and peripherals like DAC, ADC, and UART.

It supports a wide choice of Integrated Development Environments (IDEs) including IAR Embedded Workbench®, MDK-ARM, and STM32CubeIDE

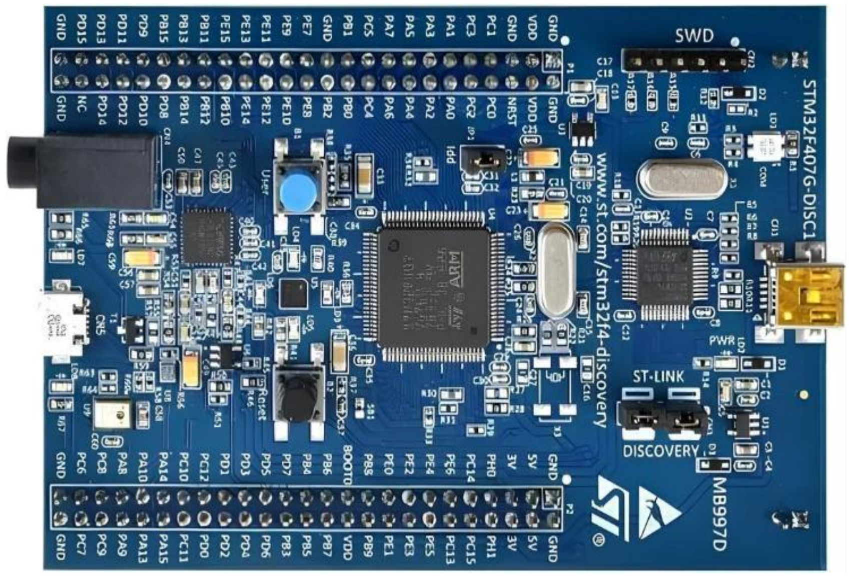


Fig 2.1 STM32F407 Discovery Board

2. **MCP2551 CAN Transceivers:**

The MCP2551 is a high-speed CAN, fault-tolerant device that serves as the interface between a CAN protocol controller and the physical bus. The MCP2551 device provides differential transmit and receive capability for the CAN protocol controller, and is fully compatible with the ISO-11898 standard, including 24V requirements. It will operate at speeds of up to 1 Mb/s. Typically, each node in a CAN system must have a device to convert the digital signals generated by a CAN controller to signals suitable for transmission over the bus cabling (differential output). It also provides a buffer between the CAN controller and the high-voltage spikes that can be generated on the CAN bus by outside sources.

Some of its features are:

- Supports 1 Mb/s operation
- Implements ISO-11898 standard physical layer requirements

- Suitable for 12V and 24V systems
- Detection of ground fault (permanent Dominant) on TXD input
- Power-on Reset and voltage brown-out protection
- An unpowered node or brown-out event will not disturb the CAN bus
- Low current standby operation
- Protection against damage due to short-circuit conditions (positive or negative battery voltage)
- Up to 112 nodes can be connected
- High-noise immunity due to differential bus implementation
- Temperature ranges: - Industrial (I): -40°C to +85°C - Extended (E): -40°C to +125°C

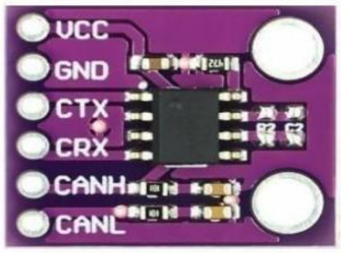


Fig 2.2 MCP2551

3. IR Sensor (Infrared Obstacle Detection Sensor)

Purpose of IR Sensor in the Project

In the *Vehicle Speed Control and Accident Avoidance System*, the IR sensor is used to detect nearby obstacles at short distances and provide quick response during critical situations. While the ultrasonic sensor measures distance over a longer range, the IR sensor ensures fast detection of obstacles very close to the vehicle, such as sudden objects or immediate hazards. The sensor helps the system make timely decisions to reduce vehicle speed or stop the motor, thereby preventing collisions.

Working Principle

The IR sensor works on the principle of infrared light reflection. It consists of two main components: an IR transmitter (IR LED) and an IR receiver (photodiode/phototransistor).

The IR transmitter continuously emits infrared rays. When an object or obstacle comes in front of the sensor, these infrared rays are reflected back toward the receiver. The IR receiver detects the reflected rays and converts them into an electrical signal. The strength of the reflected signal depends on the distance and surface of the obstacle.

The sensor module includes a comparator circuit that compares the received signal with a preset threshold. If the reflected IR intensity exceeds this threshold, the sensor output changes state, indicating the presence of an obstacle. This output signal is read by the STM32F407G microcontroller through a GPIO pin.Once an obstacle is detected, the microcontroller sends a control message via CAN communication to reduce the motor speed or stop the vehicle and activates the buzzer to alert the driver.

Pin Configuration

- **VCC Pin**
  +5V power supply
- **GND Pin**
  Ground
- **OUT Pin**
  Digital output indicating obstacle detection

OUT Pin Behavior:

- Logic LOW → Obstacle detected
- Logic HIGH → No obstacle detected

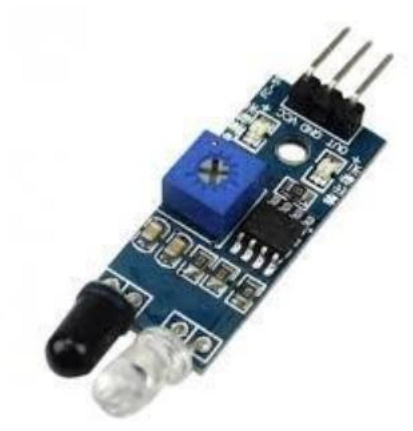


Fig. 2.3 TCRT5000

4.Ultrasonic Sensor (HC-SR04)

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules include ultrasonic transmitters, receiver and control circuit. The basic principle of work:
1) Using IO trigger for at least 10us high level signal,
2) The Module automatically sends eight 40 kHz and detects whether there is a pulse signal back.
3) IF the signal back, through high level, time of high output IO duration is the time from sending ultrasonic to returning.
Test distance = (high level time velocity of sound (340M/S)/2

Wire connecting direct as following:
• 5V Supply
• Trigger Pulse Input
• Echo Pulse Output
• 0V Ground

Working Voltage	DC 5 V
Working Current	15 mA
Working Frequency	40 KHz
Max Range	4m
Min Range	2cm
Measuring Angle	15 degrees
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in Proportion
Dimension	45*20*15mm

Table 2.2. Electric parameters of HC-SR04

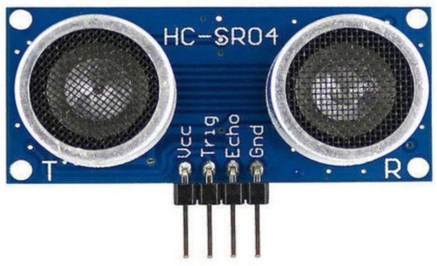


Fig 2.4. HCSR04

Timing diagram:

The timing diagram is shown below. You only need to supply a short 10 µs pulse to the trigger input to start the ranging, and then the module will send out an 8-cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion. You can calculate the range through the time interval between sending the trigger signal and receiving the echo signal. Formula: us / 58 = centimeters or us / 148 inches; or: the range = high-level time * velocity (340 m/s) / 2; we suggest using over a 60 ms measurement cycle in order to prevent the trigger signal from reaching the echo signal.

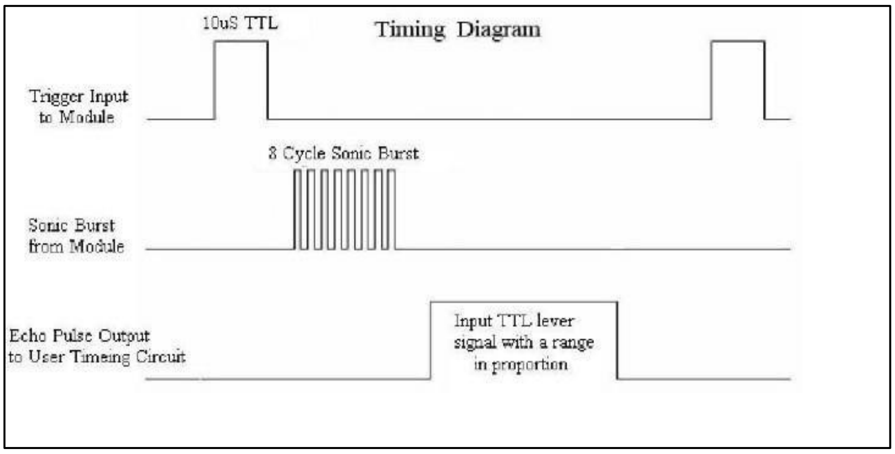

10uS TTL
Timing Diagram
Trigger Input to Module
8 Cycle Sonic Burst
Sonic Burst from Module
Echo Pulse Output to User Timeing Circuit
Input TTL lever signal with a range in proportion

Fig 2.5. Timing Diagram of HCSR-04

5.DC Motor

Almost every mechanical movement that we see around us is accomplished by an electric motor. Electric machines are a means of converting energy. Motors take electrical energy and produce mechanical energy. Electric motors are used to power hundreds of devices we use in everyday life. Motors come in various sizes. Huge motors that can take loads of 1000s of horsepower are typically used in the industry. Some examples of large motor applications include elevators, electric trains, hoists, and heavy metal rolling mills. Examples of small motor applications include motors used in automobiles, robots, hand power tools, and food blenders. Micro-machines are electric machines with parts the size of red blood cells and find many applications in medicine. Electric motors are broadly classified into two different categories: DC (Direct Current) and AC (Alternating Current). Within these categories are numerous types, each offering unique abilities that suit them well for specific applications. In most cases, regardless of type, electric motors consist of a stator (stationary field) and a rotor (the rotating field or armature) and operate through the interaction of magnetic flux and electric current to produce rotational speed and torque. DC motors are distinguished by their ability to operate from direct current. There are different kinds of D.C. motors, but they all work on the same principles. In this chapter, we will study their basic principle of operation and their characteristics. It's important to understand motor characteristics so we can choose the right one for our application requirements. The learning objectives for this chapter are listed below.

DC Motor Basic Principles

- Energy Conversion

If electrical energy is supplied to a conductor lying perpendicular to a magnetic field, the interaction of current flowing in the conductor and the magnetic field will produce mechanical force (and therefore, mechanical energy).

- Value of Mechanical Force

There are two conditions that are necessary to produce a force on the conductor. The conductor must be carrying current and must be within a magnetic field. When these two conditions exist, a force will be applied to the conductor, which will attempt to move the conductor in a direction perpendicular to the magnetic field. This is the basic theory by which all DC motors operate. The force exerted upon the conductor can be expressed as follows.

$F = B * i * l$ Newton

where B is the density of the magnetic field, l is the length of the conductor, and i is the value of the current flowing in the conductor. The direction of motion can be found using Fleming's Left Hand Rule.

The first finger points in the direction of the magnetic field (first - field), which goes from the North Pole to the South Pole. The second finger points in the direction of the current in the wire (second - current). The thumb then points in the direction the wire is thrust or pushed while in the magnetic field (thumb: torque or thrust).

Fig 2.6. DC Motor

6. Motor Driver IC (L293D)

The L293D is a dual H-bridge motor driver IC used to drive DC motors and stepper motors. Microcontrollers cannot directly drive motors due to low current output; therefore, the L293D acts as an interface between the STM32 microcontroller and the DC motors.

Working Principle

- The L293D contains two H-bridges, allowing control of two DC motors independently.
- Direction of motor rotation is controlled by logic signals from the microcontroller.
- Motor speed is controlled using PWM (Pulse Width Modulation).
- The IC provides current amplification and protects the microcontroller.

Fig 2.8. L298

Key Specifications

Parameter	**Value**
Motor Supply Voltage (Vs)	4.5V – 36V
Logic Supply Voltage (Vss)	5V
Output Current	600 mA per channel
Peak Output Current	1.2 A
Number of Motors	2 DC motors
Protection	Built-in flyback diodes

7.Buzzer

A buzzer is an audio signaling device used to generate audible alerts. It plays a crucial role in warning the driver during unsafe conditions.

Working Principle

- When voltage is applied, the buzzer produces sound
- Driven using a **transistor** to protect the microcontroller
- Different beep patterns represent different alerts
-

Key Specifications

Parameter	**Value**
Operating Voltage	**5V**
Type	**Active buzzer**
Sound Level	~85 dB
Current Consumption	~30 mA

Fig 2.10 Buzzer

3. SOFTWARE REQUIREMENTS

1. STM32CubeIDE.

STM32CubeIDE is a free, all-in-one integrated development environment (IDE) from STMicroelectronics, designed for developing applications on STM32 microcontrollers and microprocessors. It is part of the STM32Cube software ecosystem and is based on the Eclipse/CDT framework. STM32CubeIDE integrates features from STM32CubeMX for configuration and project management, along with a GNU C/C++ compiler toolchain and GDB debugger.

Key features:

- Integration of STM32CubeMX: Allows for the selection of STM32 microcontrollers, pin assignments, clock and peripheral configuration, and initialization code generation.
- Eclipse-based: Supports Eclipse plug-ins and uses the GNU C/C++ toolchain for ARM and GDB debugger.
- Debugging Tools: Offers advanced debugging features, including CPU core, peripheral register, and memory views, live variable watch, system analysis, real-time tracing, and CPU fault analysis.
- Build and Stack Analyzers: Includes build and stack analyzers that provide information on project status and memory requirements.
- Project Import: Supports importing projects from Atollic TrueSTUDIO and AC6 System Workbench for STM32 (SW4STM32).
- Multi-OS Support: Compatible with Windows, Linux, and MacOS.

STM32CubeIDE helps developers with chip selection, project configuration, code generation, editing, compiling, debugging, and burning. It uses a workspace to manage projects, where each workspace is a folder containing project folders and a “. metadata" folder with project information. For dual-core architecture MCUs, such as STM32H7, STM32L5, and STM32MP1 series, the STM32CubeIDE project has a hierarchical structure with "root" and "child" projects for each core.

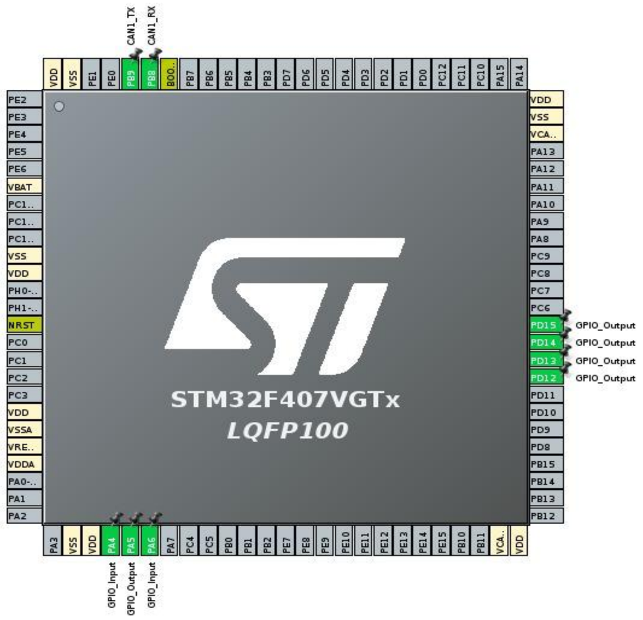

CAN1_TX
CAN1_RX
PB9
PB8
STM32F407VGTx
LQFP100
PD15 GPIO_Output
PD14 GPIO_Output
PD13 GPIO_Output
PD12 GPIO_Output
PA4 GPIO_Input
PA5 GPIO_Output
PA6 GPIO_Input

Fig . Tx

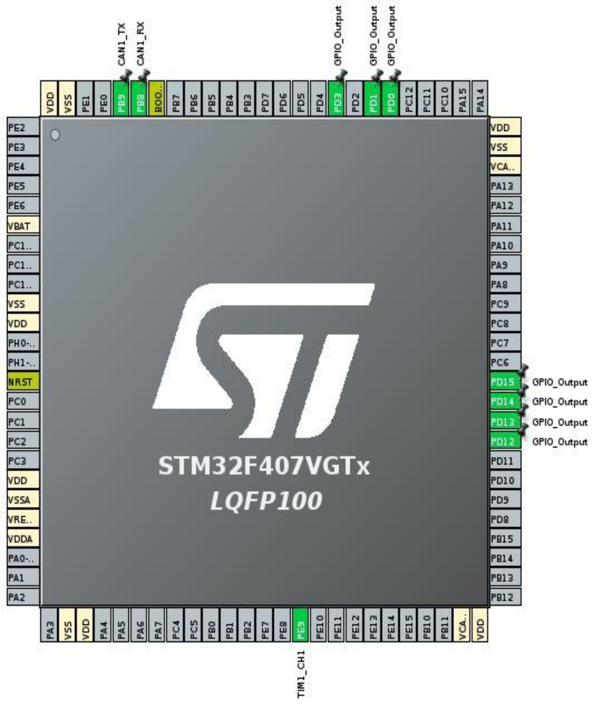

CAN1_TX
CAN1_RX
PB9
PB8
PD3 GPIO_Output
PD1 GPIO_Output
PD0 GPIO_Output
STM32F407VGTx
LQFP100
PD15 GPIO_Output
PD14 GPIO_Output
PD13 GPIO_Output
PD12 GPIO_Output
PE9 TIM1_CH1

fig. Rx

2. STM32 HAL Drivers

STM32 HAL (Hardware Abstraction Layer) Drivers are software libraries provided by STMicroelectronics to simplify peripheral configuration and control. HAL drivers offer a high-level API that abstracts low-level hardware registers, making embedded development easier, safer, and more portable.

Advantages

- Faster development
- Improved code readability
- Portability across STM32 devices
- Reduced hardware dependency

3. Embedded C Language

Introduction

Embedded C is an extension of the C programming language used for developing firmware in microcontroller-based systems. It allows direct control of hardware resources while maintaining structured and readable code.

Role in the Project

Embedded C is used to:

- Read and process sensor data
- Implement safety decision logic
- Control motors and buzzer
- Handle CAN communication
- Manage interrupts and real-time events

Advantages

- High execution speed
- Efficient memory usage
- Hardware-level control
- Widely supported in embedded systems

4. CAN Protocol Stack

Introduction

The Controller Area Network (CAN) protocol is a robust serial communication protocol widely used in automotive and industrial systems. A CAN protocol stack defines how messages are formatted, transmitted, received, and prioritized.

Role in the Project

In this system, CAN is used to:

- Transmit safety alerts
- Ensure reliable and real-time communication
- Prioritize emergency messages using CAN IDs
- Enable scalability for future modules

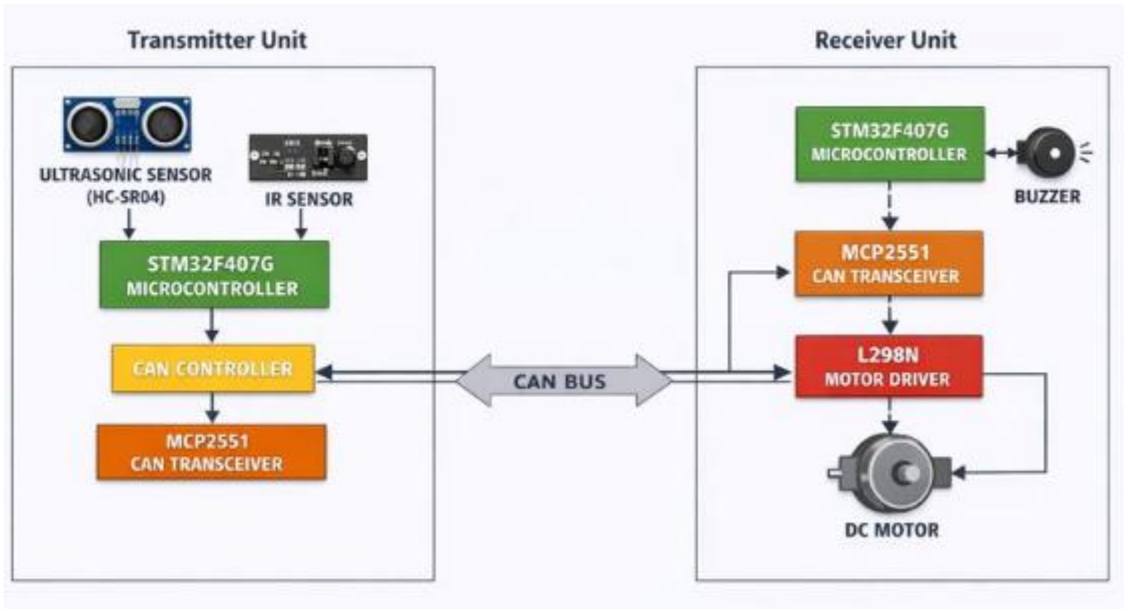
Key Features

- Message-based communication
- Multi-master architecture
- Built-in error detection and correction
- High noise immunity

4. DESIGN AND IMPLEMENTATION

- The **Vehicle Speed Control and Accident Avoidance System using STM32F407G and CAN Communication** is designed to monitor the vehicle's surroundings and control vehicle speed in real time to prevent accidents. The system integrates obstacle detection sensors with the STM32F407G microcontroller, which processes sensor data and makes speed control decisions. All critical control commands are transmitted using the CAN protocol**,** ensuring reliable and prioritized communication.
- The system follows a modular architecture in which sensors such as the ultrasonic and IR sensors continuously detect obstacles and unsafe distances. Based on predefined thresholds, the transmitter node sends control commands over the CAN bus. The receiver node receives these commands and controls the motor using a motor driver while activating a buzzer to alert the driver during hazardous conditions. This design reflects real automotive ECU communication and provides an effective solution for intelligent speed control and accident avoidance.

Block Diagram:



Working of the System

1. When the system is powered ON, both the **Transmitter (TX)** and **Receiver (RX)** STM32F407G microcontrollers initialize their peripherals, including sensors, CAN module, timers, and GPIOs.
2. The **ultrasonic sensor** continuously measures the distance between the vehicle and obstacles ahead by transmitting ultrasonic pulses and receiving the echo signal.
3. The **IR sensor** detects nearby obstacles at short distances and provides fast response for critical situations.
4. Sensor data is processed by the **TX node microcontroller**, where it is compared with predefined safety distance thresholds.
5. Based on the distance measurement:
   - Safe distance → Normal vehicle speed
   - Warning distance → Reduced vehicle speed
   - Critical distance → Emergency stop command
6. The TX node transmits the corresponding control command over the **CAN bus** using the CAN controller and **MCP2551 CAN transceiver**.
7. The **RX node** receives the CAN message through its MCP2551 transceiver and decodes the command.
8. According to the received command, the RX node controls the **L298N motor driver** using PWM signals to regulate the speed of the **DC motor**.
9. In dangerous or emergency conditions, the motor is stopped completely to avoid collision.
10. A **buzzer** connected to the RX node is activated to provide an audible alert to warn the driver.
11. Once the obstacle is cleared and safe conditions are restored, the system resumes normal motor operation.

- **Implementation Methodology**

1. Initialize the STM32F407G microcontroller peripherals using STM32 HAL drivers, including GPIO, timers, PWM, CAN controller, and communication interfaces.
2. Continuously acquire data from the ultrasonic sensor and IR obstacle sensor to monitor the distance between the vehicle and nearby obstacles.
3. Process the sensor data in real time and compare the measured distance values with predefined safety thresholds stored in the microcontroller.
4. Detect unsafe driving conditions based on sensor inputs:
    - **Safe distance** → Normal vehicle operation
    - **Reduced distance** → Warning condition
    - **Critical short distance** → High collision risk
5. Based on the detected condition, generate appropriate control actions:
    - Reduce motor speed or stop the motor using the L298N motor driver
    - Activate the buzzer to alert the driver
    - Transmit control and alert messages to the receiver node using CAN communication
6. At the receiver (RX) node, receive CAN messages and execute motor control actions accordingly.
7. Resume normal vehicle operation only when the obstacle is cleared and safe distance conditions are restored

5.ADVANTAGES

1. **Improved Vehicle Safety**
The system continuously monitors obstacles and automatically controls vehicle speed, significantly reducing the risk of accidents caused by delayed driver response.

2.**Real-Time Speed Control**
Obstacle detection and speed regulation are performed in real time, allowing immediate corrective action such as speed reduction or emergency stopping.

3.**Reliable CAN Communication**
The use of CAN protocol ensures fast, priority-based, and noise-immune communication between control units, making the system suitable for automotive environments.

4.**Reduced Driver Dependency**
Automatic decision-making minimizes dependence on driver reaction time during critical situations.

5.**Modular and Scalable Architecture**
The distributed TX–RX design allows easy expansion by adding more sensors or CAN nodes without major changes.

6. **Cost-Effective Implementation**
The system uses low-cost sensors and standard embedded components, making it economical for prototype and educational use.

7. **Fast Response Time**
Combination of ultrasonic and IR sensors provides quick detection of both long-range and short-range obstacles.

8. **Simple and Efficient Design**
The system is easy to implement, understand, and maintain, making it suitable for academic and industrial training projects.

6. Disadvantages

1. **Limited Sensing Range**
   The ultrasonic and IR sensors have limited detection range and may not perform accurately at very high vehicle speeds.
2. **Environmental Sensitivity**
   Sensor performance can be affected by environmental conditions such as dust, rain, fog, sunlight, and surface reflectivity.
3. **Prototype-Level Implementation**
   The system is implemented as a prototype using basic components and does not use automotive-grade sensors or braking mechanisms.
4. **No Advanced Object Identification**
   The system cannot distinguish between different types of obstacles, as it does not use camera or vision-based recognition.
5. **Calibration Requirement**
   Sensors require proper calibration for accurate distance measurement and obstacle detection.
6. **Limited Decision Intelligence**
   The system operates on fixed threshold values and does not use adaptive or AI-based decision-making.
7. **Power Dependency**
   Continuous operation of sensors and CAN communication depends on stable power supply, which may affect performance in low-power conditions.

7. FUTURE SCOPE

1. **Integration of Additional CAN Nodes**
The system can be extended by adding more CAN nodes such as braking control units, dashboard displays, and steering modules to simulate a complete in-vehicle network.

2. **Use of Automotive-Grade Sensors**
Replacing prototype sensors with automotive-grade ultrasonic, radar, or LiDAR sensors can improve accuracy, reliability, and durability.

3. **Automatic Emergency Braking System**
The speed control mechanism can be upgraded to control actual braking actuators for automatic emergency braking.

4. **Advanced Obstacle Detection**
Camera-based vision systems and sensor fusion techniques can be integrated for better obstacle classification and detection accuracy.

5. **Adaptive Speed Control**
Speed control logic can be enhanced using adaptive algorithms or AI-based decision-making instead of fixed thresholds.

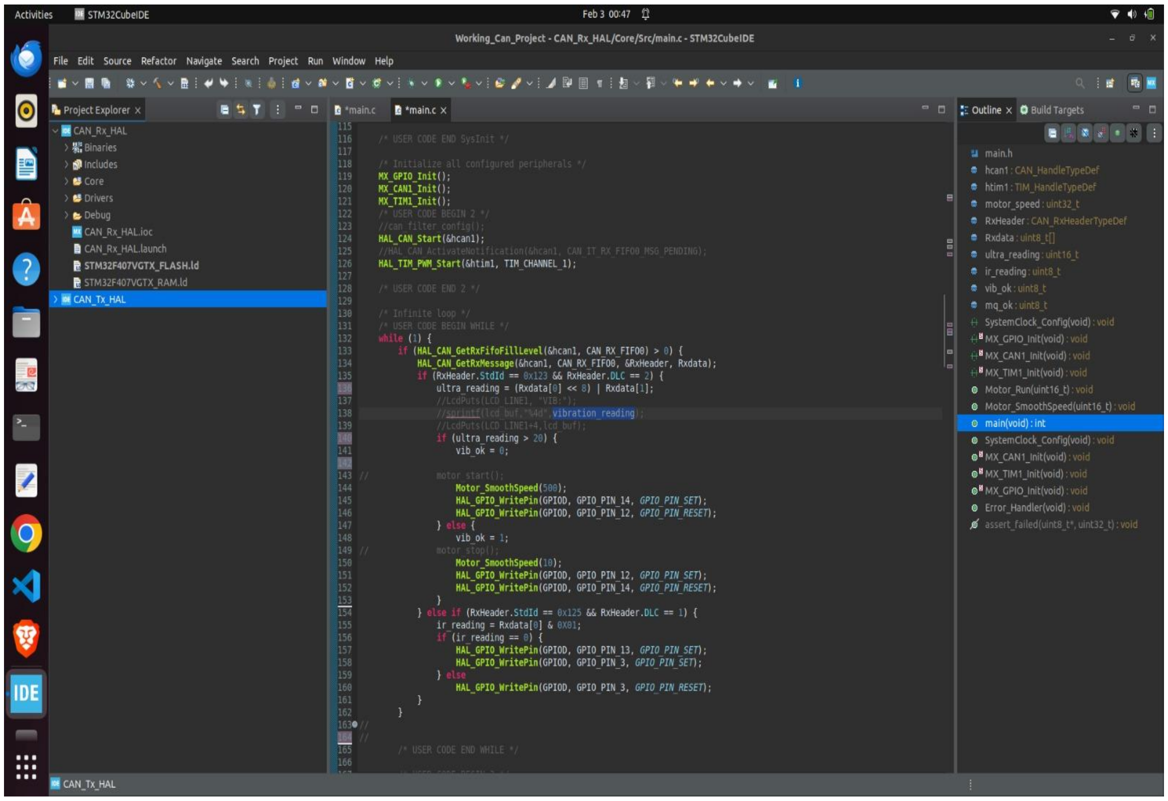
6. **Data Logging and Diagnostics**
Vehicle and safety event data can be logged in non-volatile memory for post-analysis and fault diagnostics.

7. **Integration with Vehicle Dashboard**
CAN messages can be displayed on a digital dashboard to provide real-time alerts and system status to the driver.

8. **Compliance with Automotive Safety Standards**
The system can be upgraded to meet functional safety standards such as ISO 26262 for real-world automotive applications

8.CONCLUSION

- The *Vehicle Speed Control and Accident Avoidance System using STM32F407G and CAN Communication* successfully demonstrates an intelligent embedded solution for improving vehicle safety through real-time obstacle detection and automatic speed control. By integrating ultrasonic and IR sensors with the STM32F407G microcontroller, the system continuously monitors the vehicle's surroundings and detects unsafe proximity conditions.
- The use of CAN communication enables reliable, fast, and priority-based data exchange between distributed control units, closely resembling real automotive ECU networks. Based on sensor inputs, the system automatically reduces vehicle speed or stops the motor and alerts the driver using a buzzer, thereby minimizing the risk of collisions caused by delayed human response.
- Overall, the project proves that CAN-based embedded systems can effectively enhance road safety while maintaining a modular and scalable architecture. With further enhancements such as automotive-grade sensors and advanced control algorithms, the proposed system can be extended for real-world vehicle safety applications.



Activities
STM32CubeIDE
Feb 3 00:47
Working_Can_Project - CAN_Rx_HAL/Core/Src/main.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
CAN_Rx_HAL
Binaries
Includes
Core
Drivers
Debug
CAN_Rx_HAL.ioc
CAN_Rx_HAL.launch
STM32F407VGTX_FLASH.ld
STM32F407VGTX_RAM.ld
CAN_Tx_HAL
*main.c
*main.c
Outline
Build Targets
main.h
hcan1 : CAN_HandleTypeDef
htim1 : TIM_HandleTypeDef
motor_speed : uint32_t
RxHeader : CAN_RxHeaderTypeDef
Rxdata : uint8_t[]
ultra_reading : uint16_t
ir_reading : uint8_t
vib_ok : uint8_t
mq_ok : uint8_t
SystemClock_Config(void) : void
MX_GPIO_Init(void) : void
MX_CAN1_Init(void) : void
MX_TIM1_Init(void) : void
Motor_Run(uint16_t) : void
Motor_SmoothSpeed(uint16_t) : void
main(void) : int
SystemClock_Config(void) : void
MX_CAN1_Init(void) : void
MX_TIM1_Init(void) : void
MX_GPIO_Init(void) : void
Error_Handler(void) : void
assert_failed(uint8_t*, uint32_t) : void
CAN_Tx_HAL

```c

    /* USER CODE END SysInit */

    /* Initialize all configured peripherals */
    MX_GPIO_Init();
    MX_CAN1_Init();
    MX_TIM1_Init();
    /* USER CODE BEGIN 2 */
    //can_filter_config();
    HAL_CAN_Start(&hcan1);
    //HAL_CAN_ActivateNotification(&hcan1, CAN_IT_RX_FIFO0_MSG_PENDING);
    HAL_TIM_PWM_Start(&htim1, TIM_CHANNEL_1);

    /* USER CODE END 2 */

    /* Infinite loop */
    /* USER CODE BEGIN WHILE */
    while (1) {
        if (HAL_CAN_GetRxFifoFillLevel(&hcan1, CAN_RX_FIFO0) > 0) {
            HAL_CAN_GetRxMessage(&hcan1, CAN_RX_FIFO0, &RxHeader, Rxdata);
            if (RxHeader.StdId == 0x123 && RxHeader.DLC == 2) {
                ultra_reading = (Rxdata[0] << 8) | Rxdata[1];
                //LcdPuts(LCD_LINE1, "VIB:");
                //sprintf(lcd_buf,"%4d",vibration_reading);
                //LcdPuts(LCD_LINE1+4,lcd_buf);
                if (ultra_reading > 20) {
                    vib_ok = 0;

//              motor_start();
                    Motor_SmoothSpeed(500);
                    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_SET);
                    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12, GPIO_PIN_RESET);
                } else {
                    vib_ok = 1;
//              motor_stop();
                    Motor_SmoothSpeed(10);
                    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_12, GPIO_PIN_SET);
                    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_14, GPIO_PIN_RESET);
                }
            } else if (RxHeader.StdId == 0x125 && RxHeader.DLC == 1) {
                ir_reading = Rxdata[0] & 0X01;
                if (ir_reading == 0) {
                    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_13, GPIO_PIN_SET);
                    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_3, GPIO_PIN_SET);
                } else
                    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_3, GPIO_PIN_RESET);
            }
        }
//
//
        /* USER CODE END WHILE */

```

3. REFERENCE

[1] Robert Bosch GmbH, **“CAN Specification Version 2.0,”** Bosch, Germany, 1991.

[2] STMicroelectronics, **“STM32F407xx Datasheet,”** STMicroelectronics, 2023.

[3] STMicroelectronics, **“STM32F4 Discovery Board User Manual,”** STMicroelectronics.

[4] Microchip Technology Inc., **“MCP2551 High-Speed CAN Transceiver Datasheet,”** Microchip Technology.

[5] Texas Instruments, **“L293D Quadruple Half-H Motor Driver Datasheet,”** Texas Instruments.

[6] Muhammad Ali Mazidi, Rolin D. McKinlay, and Danny Causey, ***STM32 ARM Programming for Embedded Systems,*** Pearson Education.

[6] Raj Kamal, ***Embedded Systems: Architecture, Programming and Design,*** McGraw-Hill Education.

[7] Manoj Prasanth R., Raja S., and Saranya L., **“Vehicle Control Using CAN Protocol for Implementing Intelligent Braking System,”** *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, Vol. 3, Issue 3, 2014.