School of
Electronics and Communication Engineering

Minor Project I Report

on

Development of a CAN Protocol-Based Automotive Dashboard System

By:

1. **Nitish P** USN: 01FE21BEC033
2. **Tilak Badiger** USN: 01FE21BEC026
3. **Ganesh Kurdekar** USN: 01FE21BEC012
4. **Malatesh Kalled** USN: 01FE21BEC004

Semester: VI, 2023-2024

Under the Guidance of

Prof. Nivedita Shettar

K.L.E SOCIETY'S
KLE Technological University,
HUBBALLI-580031
2023-2024

SCHOOL OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that project entitled **"Development of a CAN Protocol-Based Automotive Dashboard System"** is a bonafide work carried out by the student team of **"Tejaswini Divatagi (01FE21BEC035) , Tilak Badiger (01FE21BEC026) , Ganesh Kurdekar (01FE21BEC012), Gaddam Sharmila , (01FE21BEC027) "**. The project report has been approved as it satisfies the requirements with respect to the mini project work prescribed by the university curriculum for BE (VI Semester) in School of Electronics and Communication Engineering of KLE Technological University for the academic year 2023-2024.

Nivedita Shettar	**Suneeta V. Budihal**	**B. S. Anami**
Guide	**Head of School**	**Registrar**

External Viva:

Name of Examiners	**Signature with date**
1.	
2.	

ACKNOWLEDGMENT

Every project's success is largely attributable to the work of numerous individuals who have consistently provided their insightful counsel or extended a helping hand. We really appreciate the motivation, assistance, and direction provided by everyone who helped to make this effort a success. We would like to use this opportunity to express our gratitude to Dr. Suneeta V. Budihal, Head of the School of Electronics and Communications, for providing us with a learning environment that encouraged the development of our practical abilities, which helped our project succeed.We would also want to take this time to thank Prof. Nivedita Shettar for his constant supervision, direction, and provision of the information we needed to complete the project.We are grateful and pleased to have received ongoing support, encouragement, and guidance from SoECE's teaching and non-teaching faculty, which allowed us to successfully complete our project.We are grateful that the School of Electronics and Communications at KLE Technological University gave us the tools we needed to finish this project.

-The project Team 24

ABSTRACT

The creation of a cutting-edge dashboard system for automobiles that will improve driver awareness and vehicle monitoring. The central processing unit of the system is an ESP32 microcontroller, which is coupled to a number of sensors. To give real-time information on critical vehicle parameters, the dashboard specifically includes an ultrasonic sensor, a fuel level sensor, and a temperature sensor.The sensor data is processed by the ESP32 microcontroller and sent to a CAN receiver module over a Controller Area Network (CAN) bus. Decoding the CAN signals, the receiving module—which is likewise built around an ESP32 microcontroller—transmits the data to an LCD and a buzzer.While the buzzer acts as an audible alert system for impend- ing dangers, the LCD displays vital information like interior temperature,fuel level and object proximity. This integration takes advantage of the ESP32's powerful networking and processing capabilities, ensuring a dependable and efficient communication link inside the vehicle's electronic environment. The created system intends to improve driving safety and convenience by providing fast and accurate information via a simple interface.

Contents

List of Figures

Chapter 1

Introduction

The automobile industry is constantly striving to improve vehicle safety, efficiency, and driving convenience through technological breakthroughs. The dashboard system is a crucial component of modern automobiles, providing drivers with vital information about the vehicle's status and environment. This research looks at the design and execution of a revolutionary car dashboard that uses modern sensor technologies and communication protocols to provide real-time data to the driver.

At the heart of this system are three primary sensors: a temperature sensor, a fuel level sensor, and an ultrasonic sensor. The temperature sensor monitors the engine's operating temperature, ensuring it remains within safe limits. The fuel level sensor provides accurate readings of the fuel tank, helping to prevent unexpected shortages. The ultrasonic sensor detects obstacles in the vehicle's vicinity, enhancing situational awareness and aiding in parking maneuvers.

These sensors are connected to an ESP32 microcontroller, known for its versatility and powerful processing capabilities. The ESP32 collects and processes the sensor data, which is then transmitted over a Controller Area Network (CAN) bus. The CAN bus is a robust vehicle communication system that allows microcontrollers and devices to communicate with each other without a host computer.

The transmitted data is received by another ESP32 microcontroller configured as a CAN receiver. This microcontroller decodes the data and displays it on an LCD screen, providing the driver with a clear and immediate visual representation of the vehicle's status. Additionally, a buzzer is integrated into the system to deliver auditory alerts for critical warnings, ensuring the driver's attention is promptly captured.

By utilizing the ESP32's advanced features and the reliable CAN bus protocol, this au- tomotive dashboard system promises enhanced safety, reliability, and user-friendliness. This introduction sets the stage for a detailed examination of the system's design, implementation, and potential benefits in subsequent sections of the report.diverse settings.

1.1 Motivation

The motivation behind developing an advanced automotive dashboard system is to significantly enhance vehicle safety,efficiency and driver convenience by leveraging modern technology.This system addresses these issues by continuously monitoring critical vehicle parameters to prevent overheating,fuel shortages,collisions, thereby ensuring safer driving conditions. Additionally,it provides clear visual and auditory alerts for immediate driver awareness,improving response

times in critical situations.The use of advanced communication protocols and microcontroller technology showcases the potential for creating intelligent,responsive vehicle systems.

1.2 Objectives

1. Integrate temperature, fuel level, and ultrasonic sensors with ESP32 microcontrollers.
2. Establish communication between sensors and ESP32 boards using the Controller Area Network (CAN) protocol.
3. Utilize the ESP32 board as a CAN transmitter to relay sensor data.
4. Configure a CAN receiver connected to another ESP32 board to receive and decode CAN signals.
5. Implement processing of sensor data within the ESP32 microcontrollers for real-time analysis.
6. Display critical vehicle parameters such as temperature, fuel level, and obstacle proximity on an LCD display.
7. Provide auditory alerts using a buzzer for immediate hazard notifications.
8. Ensure continuous monitoring of essential metrics to enhance driving safety and convenience.
9. Demonstrate the effectiveness of modern sensor technology and communication protocols in improving vehicle performance and driver awareness.
10. Enhance the user experience by delivering clear and timely information through the dashboard system.

1.3 Societal Context

In today's global context, achieving sustainability goals is paramount, with the United Nations' Sustainable Development Goals (SDGs) playing a crucial role. These goals provide a vital blueprint for addressing interconnected global challenges, including poverty, hunger, health, education, gender equality, clean water, and climate action.

• SDGs prioritize inclusivity, equity, and leaving no one behind, aiming to ensure that all people can enjoy peace and prosperity by 2030.

• They promote long-term sustainability for present and future generations, emphasizing responsible consumption and production patterns.

• SDGs faster economic growth, poverty alleviation, and environmental protection through integrated and holistic approaches to development.

1.3.1 SDG Connect

SDG 11: Sustainable Cities and Communities: An automotive dashboard equipped with sensors contributes to SDG 11 by promoting road safety and enhancing urban mobility.

By reducing traffic accidents and congestion, the dashboard helps create safer and more sustainable cities and communities.

Target 11.2: By 2030, provide access to safe, affordable, accessible, and sustainable transport systems for all, improving road safety, notably by expanding public transport, with special attention to the needs of those in vulnerable situations, women, children, persons with disabilities, and older persons.

Indicator 11.2.1: Proportion of population that has convenient access to public transport, by sex, age, and persons with disabilities.

1.4 Literature survey

- The article called ”Car Design and How CAN bus Works- A Review” by S. N. Chikhale gives a deep dive into the Controller Network (CAN) bus protocol used in cars, focusing on how it’s crucial for self-driving and semi-self-driving cars. The paper about how important it is have a good communication to connect cars and computers, allowing for remote control of car settings. It says that the CAN bus is the most studied communi- cation method in the car industry, providing a simple, quick, and reliable way for cars to talk to each other - essential for modern car technologies. The part about system de- sign explains how wireless interfaces are added to cars, with CAN supporting applications like monitoring tire pressure, linking smartphones, and sharing GPS data. It introduces the Vehicular Collision Warning Communication (VCWC) protocol that needs a strong and safe communication system like CAN. The article explains the network design of the CAN system, including components like a CAN transceiver, controller, and Single Board Computer (SBC) that gather data from different points and wirelessly send it using the ECANDC algorithm for safety. More details about CAN show its origins in streamlining car wiring and its potential for Internet of Things (IoT) uses. CAN is a standard serial bus connecting Electronic Control Units (ECUs) without making car wiring too complicated. It also talks about how important it is to have controls connecting drivers with their semi-hybrid cars, with Raspberry Pi handling car info and functions. The section on system operations describes how Raspberry Pi works with the CAN controller and transceiver, using sensors to check car conditions and giving priority to certain parts for better control. The paper finishes by saying how effective the CAN bus protocol is for controlling cars and how it could be used for even more things like IoT applications. The references point out sources that helped understand and develop CAN technology. Overall, this article gives a detailed overview of how vital the CAN bus protocol is for cars, discussing its past developments and potential future uses mentioned in important quotes from the document.

- The journal article ”Vehicle Control System Implementation Using CAN Protocol” by S. Vijayalakshmi in the International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering talks about a new digital driving system created to make it easier for drivers to interact with semi-autonomous vehicles. This system uses an ARM-based data acquisition module that changes control data into digital form using Analog-to-Digital Converters (ADCs) displayed on an LCD. By using the Controller Area Network (CAN) protocol, data transfer is made more efficient, and emergency communications are possible through a GSM module. Various components like a CAN bus controller, ARM microcontroller, LCD display, GSM module, and sensors work together at different data transfer rates for various vehicle systems. The chosen ARM7TDMI-S microprocessor is known for its performance and low power usage, perfect for embedded control applications. With software written in Embedded C and debugged with MPLAB X IDE, a master node (ARM controller) and two slave nodes (PIC microcontrollers) collect data from sensors and communicate via the CAN bus. This system keeps an eye on vehicle conditions like pressure, temperature, fuel level, and speed, giving real-time feedback to the driver and sending alerts to the vehicle owner in emergencies. In conclusion, this paper highlights the cost-effectiveness and reliability of employing an embedded system for vehicle control using CAN bus technology to enable easy sharing of data between nodes for better collaboration in automotive technology advancements.

- The paper written by R. Manoj Prasanth, S. Raja, and L. Saranya from Christ the King Engineering College shares insights on a Vehicle Control System utilizing the Controller Area Network (CAN) protocol for an Intelligent Braking System (IBS). It highlights the

growing importance of electrical components in vehicles and introduces a digital driving system for semi-autonomous vehicles. This system uses a PIC-based data acquisition module to convert analog data to digital and showcase it on an LCD screen. By incorporating ultrasonic sensors for distance measurement and an RFID module for critical zone detection, the system enables a priority-based IBS.The paper elaborates on the efficient operation and advanced functionalities of the PIC18F458 microcontroller and CAN bus module that can manage different data transfer rates with message prioritization and error handling capabilities. Both simulation outcomes and a physical model illustrate the system's ability to generate alarms effectively for scenarios like low fuel, obstacle detection, and overspeed, providing real-time feedback on the driver's panel.In conclusion, the paper underscores the significance of the CAN bus system in enhancing vehicle safety, comfort, and efficiency while also reducing fuel consumption and emissions. This is backed by the authors' proficiency in automotive sensor control through CAN-based techniques.

1.5 Problem statement

Design and implement an automotive dashboard that enhances the driving experience, provides real-time data, and ensures ease of control for the driver using CAN protocol.

1.6 Organization of the report

Commencing with an introduction to the project in Chapter 1.The report systematically delves into various aspects of the endeavor.Chapter 2 comprehensively addresses the intricacies of system design, providing a detailed exploration of the algorithm.Moving forward, Chapter 3 meticulously covers the planning and implementation aspects of the design. In Chapter 4, the focus shifts to the presentation and analysis of results. Finally, Chapter 5 encapsulates the report with a conclusive summary, offering insights into the project's outcomes, implications, and laying the groundwork for future exploration and development in the identified scope.

Chapter 2

System design

Chapter 2 emphasizes system design, which is the process of defining a system's modules, in- terfaces, components, and data to meet specified requirements. It can be thought of as systems theory applied to product development. Conceptual, logical, and physical design are all aspects of system design.

2.1 Functional block diagram

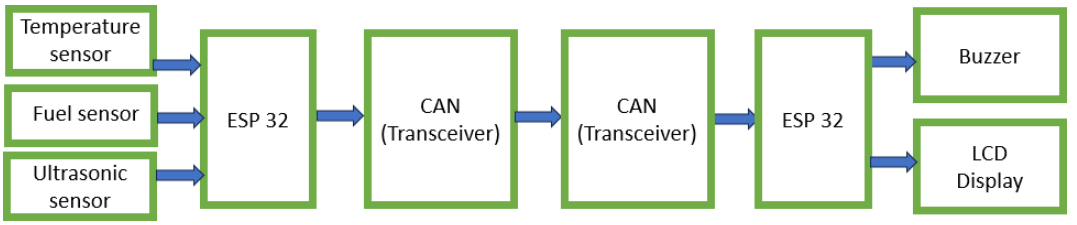

Figure 2.1: Functional Block Diagram of CAN protocol based Automative Dashboard

The block diagram illustrates a sophisticated monitoring system leveraging multiple sensors, two ESP32 microcontrollers, and a CAN (Controller Area Network) communication protocol. The system begins with three distinct sensors: a temperature sensor, a fuel sensor, and an ultrasonic sensor. These sensors are interfaced with the first ESP32 microcontroller, which gathers data such as environmental temperature, fuel levels, and distance measurements. The ESP32 processes this data locally before transmitting it further.

Once the data is processed by the first ESP32, it is sent to the CAN transmitter. The CAN bus is a robust vehicle bus standard designed to allow microcontrollers and devices to communicate

with each other without a host computer. The CAN transmitter encodes the sensor data and sends it over the network to the CAN receiver. The CAN receiver, connected to the second ESP32 module, decodes the incoming data and forwards it to the second microcontroller for further processing and decision-making.

The second ESP32 module is responsible for taking the necessary actions based on the received data. This might involve triggering a buzzer to alert users to certain conditions, such as a high temperature or low fuel level, thereby providing an immediate audible alert. Additionally, the data is also displayed on an LCD screen, offering a clear and accessible visual representation of the sensor readings. This combination of audible and visual outputs ensures that users are promptly informed of important system statuses, enhancing both safety and operational efficiency.

2.2 Technical Requirements

2.2.1 Hardware

The hardware components required for the system design include:

- ESP 32 Board
- LM35 temperature sensor
- Beeper : Piezo
- Voltage regulator
- Ultrasonic sensor HC-SR04
- CAN Transreceiver
- water level sensor
- LCD display 16x2
- connecting wires and PCB

2.2.2 Software

The software requirements for the system include:

Embedded C for programming the ESP 32 and CAN Transreceiver.

SPI.h library - for SPI communication.

mcp2515.h library - for interfacing with the MCP2515 CAN controller.

Wire.h library- for I2C communication.

LiquidCrystal I2C.h library - for controlling the I2C LCD display.

2.2.3 Bill of Material

The Bill of Materials (BOM) lists all the components and their respective quantities needed for the project.

Figure 2.2: Bill Of Materials

Sl. No	Item	Quantity	Approximate Cost
1	ESP32-WROOM-32 Module	2	1000 /-
2	LCD Display	1	100 /-
3	LM 35 Temperature sensor	1	100 /-
4	Water sensor	1	200 /-
5	ESP 32 Developer kit	1	500 /-
6	CAN Transreceiver MCP2515	2	150 /-
7	PCB Board	1	50 /-

2.3 Design Considerations

2.3.1 Memory

The design of a CAN Protocol-Based Automotive Dashboard System leveraging the ESP32 microcontroller involves careful consideration of memory resources. The ESP32 provides 448 KB of internal flash memory and 520 KB of SRAM, which are utilized for storing firmware, sensor data, and intermediate computational results. Efficient memory management techniques, such as circular buffering for sensor data storage, ensure optimal utilization of available memory.

2.3.2 Timing

Timing considerations are crucial in the design of a CAN-based automotive dashboard using the ESP32 microcontroller to ensure real-time performance and accurate data acquisition. The ESP32 operates at a clock speed of 240 MHz, providing the necessary processing power for real-time data sampling and computations. Timing constraints are addressed by implementing interrupt-driven routines for sensor data acquisition, ensuring that speed, engine temperature, and brake status readings are captured at precise intervals. The timing requirements also extend to the communication interfaces, such as the CAN bus, used for transmitting data within the vehicle's network and displaying information on the dashboard.

2.3.3 Power

Power management is a critical aspect of the CAN-based automotive dashboard design using the ESP32 microcontroller, as it directly impacts the overall efficiency and reliability of the system. The ESP32 is designed to operate at low power, making it suitable for automotive applications. Power considerations include optimizing the power consumption of the microcontroller and connected sensors, ensuring minimal energy usage during data acquisition and processing. The system design also includes power supply units and voltage regulation circuits to maintain stable operation under varying load conditions. Effective power management strategies contribute to the extended lifespan of both the dashboard system and the vehicle's electronic components.

Chapter 3

Implementation details

Chapter 3 provides a detailed design and implementation plan for the CAN protocol-based automotive dashboard system. It includes a block diagram illustrating the process flow. The chapter serves as a practical guide for users, offering step-by-step instructions for executing various processes

3.1 Specifications and final system architecture

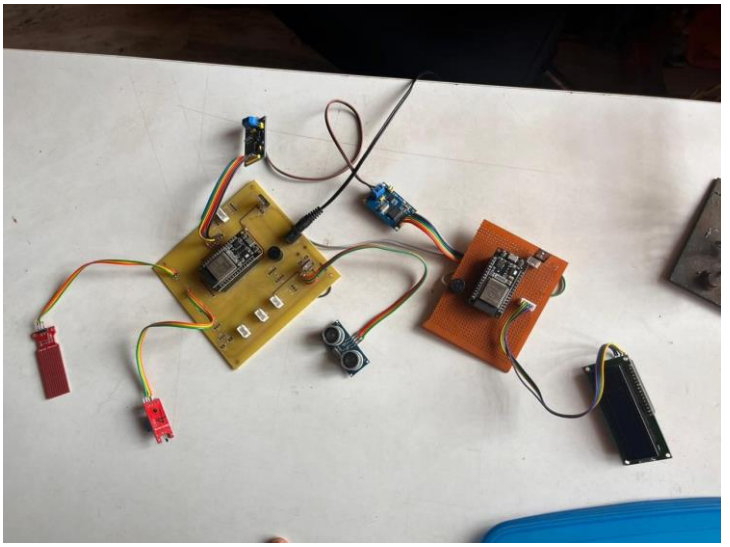

Figure 3.1: Hardware implementation with ESP32, CAN module, and sensors

COMPONENTS:

- Board-ESP32
- CAN-Module-MCP2515
- Sensors-ultrasonic sensor,Fuel Sensor,Temperature sensor
- LCD 16X2
- Buzzer

3.2 Algorithm

Transmitter

- Initialize Serial Communication:

  Begin serial communication at 9600 baud rate.

- Initialize Pin Modes:

  Set TRIG pin as OUTPUT.

  Set buzz pin as OUTPUT.

  Set ECHO pin as INPUT.

  Set TEMP pin as INPUT.

  Set WATER pin as INPUT.

  Set buzz pin to LOW initially.

- Initialize LCD:

  Turn on the LCD backlight.

  Set the cursor to the first row and print ”WEL COME”.

  Set the cursor to the second row and print ”CAN TRANSMITTER”.

  Delay for 1 second.

- Initialize CAN Communication:

  Reset the MCP2515 CAN controller.

  Set the bitrate of the CAN controller to 125KBPS.

  Set the CAN controller to normal mode.

  Print Initialization Message:

  Print ”Write to CAN” on the serial monitor.

  Main Loop

- Read Temperature:

  Call the ReadTEMP function to read the temperature sensor value and store it in variable t.

- Read Oil Sensor:

  Call the ReadOilSensor function to check the oil sensor status and set variable o accordingly.

- Measure Distance:

  Call the Distance function to measure the distance using the ultrasonic sensor and store it in variable distance.

- Prepare CAN Message:

  Set CAN message ID to 0x0F6.

  Set the data length code (DLC) to 8.

  Assign the sensor values (t, us, o, distance) to the first 4 bytes of the CAN message.

  Set the remaining 4 bytes of the CAN message to 0x00.

- Send CAN Message:

  Send the prepared CAN message using the MCP2515 CAN controller.

  ReadTEMP Function

- Read Temperature Sensor Value:

  Read the analog value from the TEMP pin. Convert the analog value to a desired scale (e.g., dividing by 4 for an 8-bit range) and store in variable t.

  ReadOilSensor Function

- Read Oil Sensor Status:

  Check the digital value from the WATER pin.

- Update Oil Status:

  If the value is HIGH (1), set o to 1 and print ”Oil Status: HIGH” on the serial monitor.

  If the value is LOW (0), set o to 0 and print ”Oil Status: LOW” on the serial monitor.

  Activate Buzzer (if Oil Status is LOW):

- If the oil status is LOW, activate the buzzer in a pattern:

  Turn the buzzer ON. Delay for 200 milliseconds.

  Turn the buzzer OFF. Delay for 200 milliseconds.

  Repeat this pattern 4 times. Distance Function

- Trigger Ultrasonic Sensor:

  Set the TRIG pin to LOW for 2 microseconds.

  Set the TRIG pin to HIGH for 10 microseconds.

  Set the TRIG pin back to LOW.

  Measure Echo Pulse:

  Measure the duration of the HIGH pulse from the ECHO pin.

- Calculate Distance:

  Calculate the distance in centimeters using the formula: distance = duration * 0.034 / 2. Store the calculated distance in variable distance

  Receiver

- Initialize Serial Communication:

  Begin serial communication at 9600 baud rate.

  Initialize Pin Modes:

  Set TRIG pin as OUTPUT.

  Set buzz pin as OUTPUT.

  Set ECHO pin as INPUT.

  Set TEMP pin as INPUT.

  Set WATER pin as INPUT.

  Set buzz pin to LOW initially.

- Initialize LCD:

  Initialize the LCD.

  Turn on the LCD backlight.

  Set the cursor to the first row and print ”WEL COME”.

  Set the cursor to the second row and print ”CAN TRANSMITTER”.

  Delay for 1 second.

- Initialize CAN Communication:

  Reset the MCP2515 CAN controller.

  Set the bitrate of the CAN controller to 125KBPS.

  Set the CAN controller to normal mode.

- Print Initialization Message:

  Print ”Write to CAN” on the serial monitor.

  Main Loop

- Read Temperature:

  Call the ReadTEMP function to read the temperature sensor value and store it in variable t.

- Read Oil Sensor:

  Call the ReadOilSensor function to check the oil sensor status and set variable o accordingly.

- Measure Distance:

  Call the Distance function to measure the distance using the ultrasonic sensor and store it in variable distance.

- Prepare CAN Message:

  Set CAN message ID to 0x0F6.

  Set the data length code (DLC) to 8.

  Assign the sensor values (t, us, o, distance) to the first 4 bytes of the CAN message.

  Set the remaining 4 bytes of the CAN message to 0x00.

- Send CAN Message:

  Send the prepared CAN message using the MCP2515 CAN controller.

  ReadTEMP Function

- Read Temperature Sensor Value:

  Read the analog value from the TEMP pin. Convert the analog value to a desired scale (e.g., dividing by 4 for an 8-bit range) and store in variable t.

  ReadOilSensor Function

- Read Oil Sensor Status:

  Check the digital value from the WATER pin.

- Update Oil Status:

  If the value is HIGH (1): Set o to 1.

  Print ”Oil Status: HIGH” on the serial monitor. Update the LCD to show ”Oil Level: HIGH”.

  If the value is LOW (0): Set o to 0.

  Print ”Oil Status: LOW” on the serial monitor. Update the LCD to show ”Oil Level: LOW”.

- Activate a buzzer for a short period:

  Turn the buzzer ON.

  Delay for 200 milliseconds.

  Turn the buzzer OFF.

  Delay for 200 milliseconds.

  Repeat this pattern 4 times.

  Distance Function

- Trigger Ultrasonic Sensor:

  Set the TRIG pin to LOW for 2 microseconds.

  Set the TRIG pin to HIGH for 10 microseconds.

  Set the TRIG pin back to LOW.

  Measure Echo Pulse:

  Measure the duration of the HIGH pulse from the ECHO pin.

- Calculate Distance:

  Calculate the distance in centimeters using the formula: distance = duration * 0.034 / 2.

  Store the calculated distance in variable distance.

- Update LCD and Buzzer if Near Vehicle:

- If us is 1 (indicating a near vehicle):

  Clear the LCD.

  Set the cursor to the first row and print ” Near Vehicle ”.

  Activate the buzzer for a short period: Turn the buzzer ON.

  Delay for 200 milliseconds.

  Turn the buzzer OFF.

  Delay for 200 milliseconds.

  Repeat this pattern 3 times

3.3 Flowchart

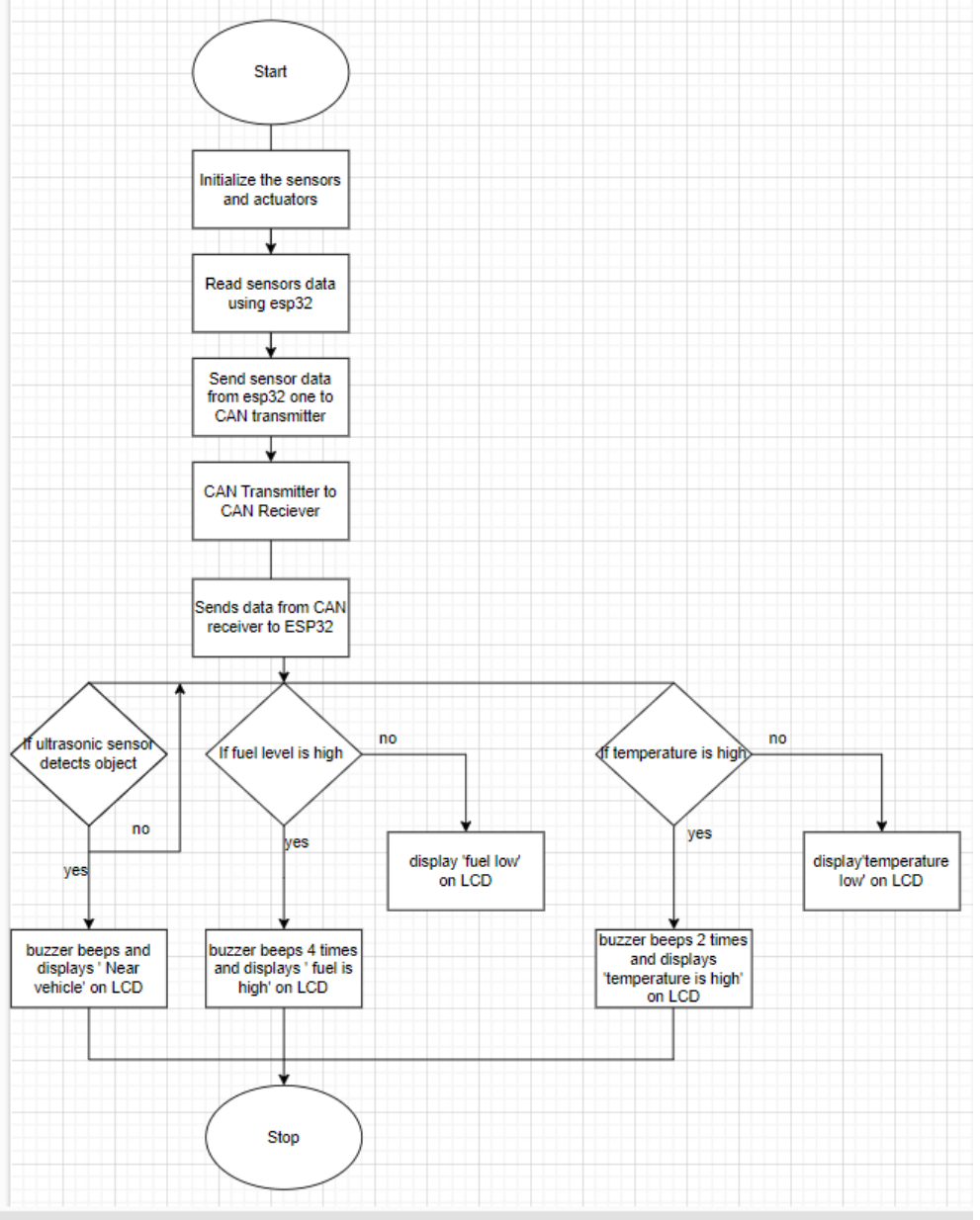

Figure 3.2: Hardware implementation with ESP32,CAN module and sensors Flowchart

3.4 Optimization

In the CAN-based automotive dashboard using the ESP32 microcontroller, optimization focuses on efficient data processing and display. Various algorithms are employed to ensure accurate monitoring of vehicle parameters such as speed, engine temperature, and brake status. Once

developed and evaluated, these algorithms are deployed to the ESP32, enabling the dashboard system to perform real-time monitoring and display directly within the vehicle. By continuously assessing vehicle performance, the dashboard provides timely insights for optimizing driving behavior and vehicle maintenance.

3.5 Debugging

Debugging the CAN-based automotive dashboard using the ESP32 involves several steps to ensure proper functioning. This includes verifying sensor connections, ensuring accurate data acquisition, and troubleshooting data transmission issues on the CAN bus. Tools like serial monitors and debugging software are used to trace and resolve issues during system development.

3.6 Challenges and Issues Faced

During the development of the CAN-based automotive dashboard using the ESP32, several challenges were encountered. These included ensuring accurate synchronization of sensor data sampling, managing memory constraints on the ESP32 microcontroller, and optimizing algorithms for deployment on a resource-constrained device. Additionally, maintaining reliable communication over the CAN bus posed significant challenges, which were addressed through robust error-handling mechanisms and communication protocols.

Chapter 4

Results and discussions

4.1 Results

The picture shows a screen used in a car dashboard. It's showing a distance of 11 cm with a warning "Near Vehicle." Figure 4.1 This is probably for an alert system that senses objects nearby and warns the driver. The screen is lit up in blue, showing white text for easy reading. It's connected to the system with colored wires, indicating it's part of a bigger system with sensors and a microcontroller like an ESP32 for processing data and communicating in real-time. The alarm system uses colorful wires, showing it's part of a big network. The wires probably connect to a microcontroller like the ESP32, known for its powerful processing and Wi-Fi capabilities. The ESP32 is vital in this system, handling real-time data from sensors on the vehicle. These sensors could be ultrasonic sensors keeping an eye out for obstacles and other vehicles. When something is too close, the system shows a warning on the dashboard screen. An 11 cm distance and "Near Vehicle" warning tell the driver to act fast to avoid a crash.

The picture in Figure 4.2 shows a system designed to keep an eye on two important things: temperature and oil level. Right now, both things are saying "HIGH". This means the temperature is too high and the oil level is too much. These readings are super important to make sure the system works right and stays safe, whether it's a car engine or big machines. The bright blue display makes it easy for people to check quickly and fix any problems fast.

When the temperature is too high, it means the system might get too hot. This could happen if it runs for a long time, doesn't cool down enough, or if something's wrong with the cooling system. If it gets too hot, parts could get messed up, break, or stop working altogether. On the other hand, if there's too much oil, it can make things worse by raising pressure inside, causing leaks, bubbles, and making things less slippery. This can also cause overheating and wear out moving parts quicker.

To watch over all this stuff, there are sensors and a computer brain (maybe an ESP32) that work together. The sensors keep checking the temperature and oil level all the time and send this info to the brain. The brain then shows the info on the display andThe picture in Figure 4.2 shows a system made to watch over two important things: temperature and oil level. Right now, both things are marked as "HIGH," meaning the temperature went over a set limit the oil level is too high. These readings are super important to make sure the system works right and stays safe, whether it's in a car or big machines. The bright blue display with white words makes it easy for people to see what's going on and take action quickly to avoid any damage.

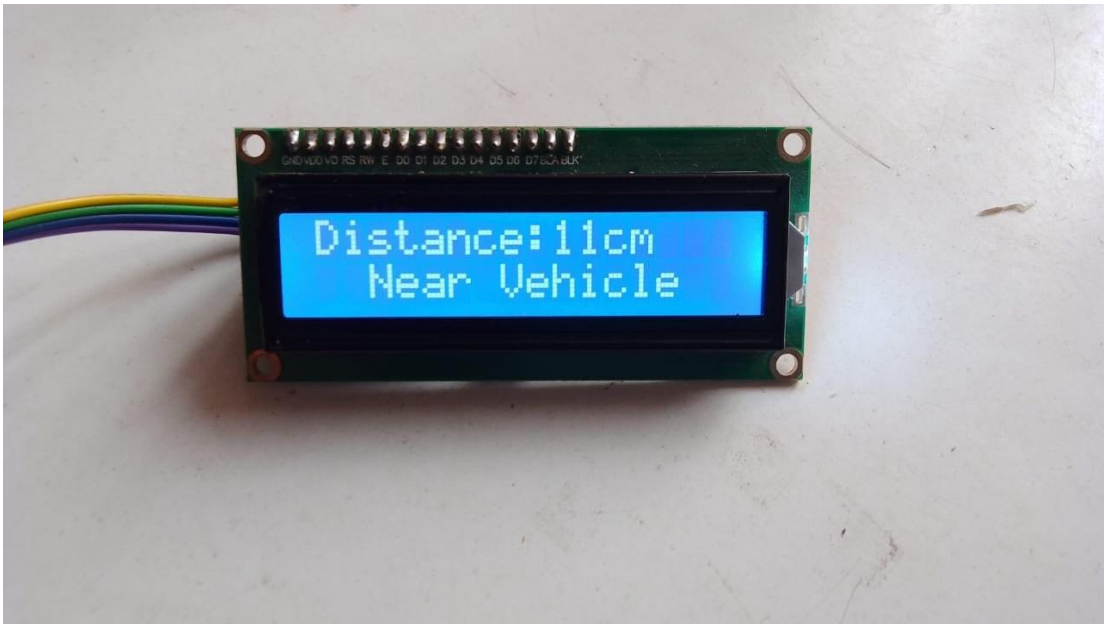

Figure 4.1: Output showing the distance and alerting the driver.

Figure 4.2: Output showing the temperature status of the engine and fuel level.

4.2 Analysis

4.2.1 Timing

The timing analysis focuses on the duration taken by different stages of the CAN-based automotive dashboard operation, including sensor data acquisition, data processing, and communication over the CAN bus. Accurate timing is crucial to ensure real-time monitoring and display of vehicle parameters such as speed, engine temperature, and brake status. This analysis helps in identifying bottlenecks and optimizing the system to maintain seamless performance. By ensuring precise timing, the dashboard can provide timely and reliable information to the driver, enhancing safety and vehicle efficiency.

4.2.2 Memory

Efficient memory management is crucial when designing a CAN-based dashboard using the ESP32 microcontroller. Techniques like circular buffering are essential for handling real-time sensor data effectively, ensuring continuous data processing without exceeding memory limits. Optimizing storage and processing of sensor data involves choosing appropriate data structures and algorithms to fit within the ESP32's memory constraints, minimizing fragmentation, and employing efficient coding practices. It's also important to streamline model parameter stor- age and dashboard functionalities to avoid unnecessary memory usage, possibly implementing data compression where feasible. By prioritizing memory efficiency throughout development, developers can maintain responsiveness and reliability while maximizing the utilization of the ESP32's memory resources for a robust CAN-based dashboard solution.

4.2.3 Power

Minimizing power consumption is paramount when developing a CAN-based dashboard using the ESP32 microcontroller, as it directly influences the efficiency of the system. Each component's power usage, from sensors to the microcontroller and communication interfaces, is meticulously measured and analyzed. The objective is to optimize power consumption without compromising the accuracy and reliability of battery monitoring and prediction capabilities. Techniques such as low-power modes for the ESP32, efficient handling of sensor data through periodic sampling or interrupt-driven mechanisms, and careful selection of communication protocols within the CAN framework are implemented to achieve this goal. By prioritizing power efficiency throughout the design process, developers ensure that the CAN-based dashboard on the ESP32 operates reliably while conserving energy, thus enhancing the overall effectiveness of the battery management system.

Chapter 5

Sustainable Development Goals (SDGs)

5.1 Introduction of SDG

The United Nations approved the Sustainable Development Goals (SDGs), sometimes referred to as the Global Goals, in 2015 as a global call to action to end poverty, safeguard the environment, and guarantee that by 2030 all people live in peace and prosperity. By 2030, the Sustainable Development Goals (SDGs) aim to eradicate poverty, safeguard the environment,and guarantee prosperity and peace for all people. They are a global call to action. The 17 Sustainable Development Goals (SDGs), which were endorsed by every UN member state in 2015, provide a broad framework for addressing the most important issues affecting both the environment and humankind. These objectives cover a wide range of topics, including social inclusion, environmental sustainability, and economic progress.

5.2 Causal Loop

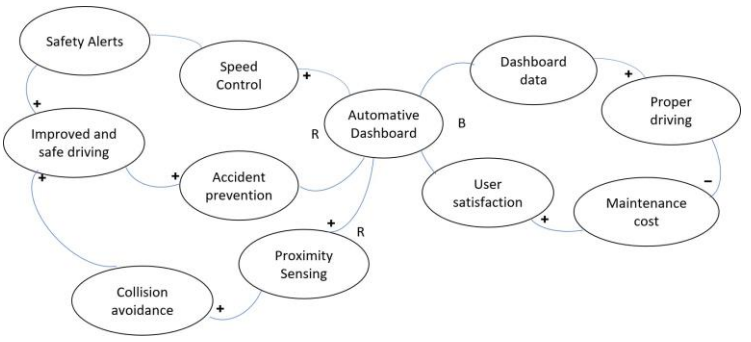

Figure 5.1: Causal Loop

Figure 5.2: SDG

5.3 Project related SDG Goal

SDG 11: Sustainable Cities and Communities: An automotive dashboard equipped with sensors contributes to SDG 11 by promoting road safety and enhancing urban mobility.

By reducing traffic accidents and congestion, the dashboard helps create safer and more sustainable cities and communities.

Target 11.2: By 2030, provide access to safe, affordable, accessible, and sustainable transport systems for all, improving road safety, notably by expanding public transport, with special attention to the needs of those in vulnerable situations, women, children, persons with disabilities, and older persons.

Indicator 11.2.1: Proportion of population that has convenient access to public transport, by sex, age, and persons with disabilities.

Our work focuses on making roads safer, supporting eco-friendly transportation methods, and enhancing city travel, in line with Goal 11.2 of the Sustainable Development Goals (SDGs). This goal wants to offer safe, affordable, reachable, and sustainable transport options to everyone by 2030, especially focusing on those in vulnerable situations like women, kids, persons with disabilities, older individuals.

To make this happen, our project combines new technologies and creative ideas to build a better safer transportation system. By using high-tech sensors, real-time data, and smart systems, we aim to decrease accidents, improve traffic flow boost the safety of everyone on the road. These steps directly help with Indicator 11.2.1 which centers on how many people have easy access to public transport.

Our project supports sustainable transportation by encouraging eco-friendly modes like electric vehicles, bicycles, public transit. We add features for seamless integration of these transport options, making it easier to choose greener alternatives. This helps reduce carbon emissions creates a cleaner, healthier urban environment.

Chapter 6

Conclusion and future scope

6.1 Conclusion

In conclusion, implementing an automotive dashboard system that integrates a temperature sensor, ultrasonic sensor, and fuel level sensor to read real-time data from one ESP32 and transmit it to another ESP32 via the CAN protocol offers a highly efficient and reliable solution. This setup allows for effective monitoring and display of critical vehicle information on an LCD, while also providing timely alerts through a buzzer. By leveraging the CAN protocol for data communication, the system ensures robust and secure data transfer, minimizing latency and enhancing the overall driving experience. This innovative approach not only improves vehicle safety and maintenance but also demonstrates the potential for further advancements in automotive dashboard technology.

6.2 Future scope

As vehicle technology advances, these systems are becoming increasingly important for making driving safer, more efficient, and more enjoyable.One major area of development is the integration of advanced safety features. CAN protocol-based dashboards will work closely with systems that help prevent accidents and protect passengers.CAN protocol-based dashboards can provide this information in a clear and accessible way, helping drivers make the most of their vehicle's capabilities.Personalization is another key trend. Future dashboards will offer customizable displays that adapt to individual driver preferences. This could include augmented reality features that overlay navigation instructions directly onto the windshield, making it easier to follow directions without taking your eyes off the road.CAN protocol-based systems will be able to monitor the health of various vehicle components in real time, predicting when maintenance is needed before problems occur. This will help drivers avoid unexpected breakdowns and extend the life of their vehicles.Security is also a top priority. As vehicles become more connected, protecting data from cyber threats is crucial. CAN protocol-based dashboards will ensure secure data communication, safeguarding both vehicle and personal information.Overall, the future of CAN protocol-based automotive dashboard systems is bright. These advancements will keep dashboards at the forefront of automotive technology, continually improving the driving experience for everyone.

Bibliography

[1] Prasanth, R. Manoj, S. Raja, and L. Saranya. ”Vehicle Control Using CAN Protocol For Implementing the Intelligent Braking System (IBS).” International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering 3.3 (2014).

[2] Salunkhe, A. A., Pravin P. Kamble, and Rohit Jadhav. ”Design and implementation of CAN bus protocol for monitoring vehicle parameters.” 2016 IEEE international conference on recent trends in electronics, information communication technology (RTEICT). IEEE, 2016.

[3] Vijayalakshmi, S. ”Vehicle control system implementation Using CAN protocol.” International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering 2.6 (2013): 2532-2538.

[4] Chikhale, Shahu N. ”Automobile design and implementation of can bus protocol-A review.” IJRDO-Journal of Electrical And Electronics Engineering 4.1 (2018): 01-05.