ADVANCED INSTRUMENT CLUSTER DEVELOPMENT USING STM32H735IG-DK

Embedded Software Engineer

in

Automotive Domain

By

SANDHYA LUKKA

Embedded Software Engineer in Automotive Domain

ESTAR TECHNOLOGIES OPC PRIVATE LIMITED

4th -Floor, AWFIS, Vasavi MPM Grand, Yella Reddy Guda, Ameerpet
Hyderabad, Telangana,

500073

Abstract

This project utilizes the STM32H7 microcontroller to streamline the integration of odometers and speedometers, traditionally managed by separate controllers, into a unified system. By leveraging the STM32H7's advanced features, including high-performance cores and integrated peripherals, the project aims to eliminate compatibility issues and synchronization problems common with multiple controllers. This approach enhances real-time data processing accuracy, enabling precise calculation of distance traveled and vehicle speed. Additional functionalities like TripA and TripB readings enrich the instrument cluster's utility, providing customizable display layouts and critical vehicle condition indicators for enhanced user experience and safety.

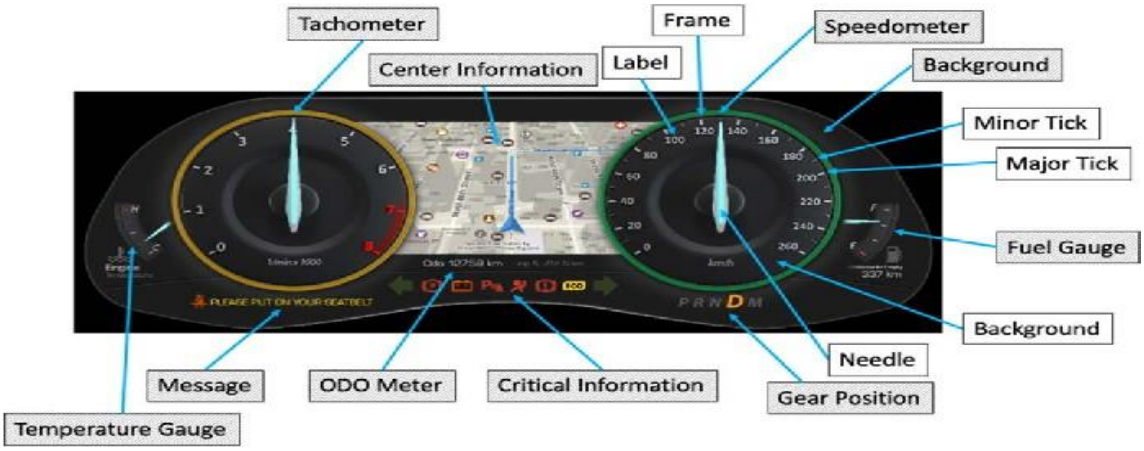
Introduction

The project focuses on developing an advanced instrument cluster for vehicles using the STM32H735G-DK development board. This cluster incorporates essential features such as an odometer and a speedometer. Leveraging the STM32H7 microcontroller's high process- ing power and comprehensive peripheral support, the development process involves both hardware and software components. In terms of software development, the project utilizes the STM32CubeIDE as the primary Integrated Development Environment (IDE). Configu- ration of microcontroller peripherals is streamlined using the STM32CubeMX tool, which generates initialization code to set up interfaces like GPIOs, UARTs and timers. The graph-ical user interface (GUI) for the instrument cluster is designed using graphics libraries like TouchGFX , enabling the creation of visually appealing displays with animations, images, and text. Sensor integration is a crucial aspect of the project, facilitating accurate moni- toring of vehicle parameters. Sensors such as wheel speed sensors and odometer sensors are interfaced with the microcontroller to provide data on the vehicle's speed and distance traveled.

1.1 Overview of Instrument Cluster

An instrument cluster, or dashboard, is essential in vehicles, located behind the steering wheel to provide critical information such as speed, fuel level, engine temperature, odometer readings, and warning lights. It interfaces with various sensors and modules, like wheel speed sensors and fuel level sensors, to deliver real-time data. Modern clusters often feature digital displays with enhanced customization, improved visibility, and additional functionalities like navigation and entertainment controls. Communicating via the Controller Area Network (CAN) bus, instrument clusters ensure seamless integration and data exchange, enhancing driver safety, comfort, and overall vehicle performance.

Figure 1.1: Digital Instrument Cluster



1.2 Objective and scope of the project

The objective of developing an instrument cluster with the STM32H735G-DK is to create a reliable and user-friendly interface for displaying essential vehicle information to the driver. This includes real-time monitoring of parameters like vehicle speed, engine RPM, and fuel level, integrating with vehicle systems for enhanced control and safety, offering customization options, ensuring reliability under harsh conditions, and complying with automotive safety standards. Ultimately, the goal is to enhance the driving experience, improve vehicle performance, and ensure safety on the road.

The project involves developing an instrument cluster with odometer and speedometer functionalities using the STM32H735 microcontroller. This encompasses:

- Hardware Design:

  Selecting sensors, display modules, and communication interfaces compatible with the microcontroller for accurate tracking of vehicle speed and distance..

- Software Development:

  Creating algorithms to process sensor data, calculate odometer readings, and update vehicle speed in real-time using appropriate programming languages and development tools.

- Integration:

  Integrating hardware components with the microcontroller and developing firmware for seamless communication between sensors, display modules, and the micro controller.

- User Interface Design:

  Designing a user-friendly interface for the instrument cluster display to present odometer readings, vehicle speed, and other information clearly to the driver.

- Testing and Validation:

  Conducting comprehensive testing to ensure accuracy, reliability, and compliance

with automotive safety standards, testing under various conditions to verify odometer and speedometer performance.

Proposed Solution

2.1 Basic Components of Instrument Cluster

- STM32H735 Microcontroller:

  This is the central processing unit of your system. It handles tasks such as reading sensor data, processing it, and controlling the display output.

- Speed Sensor:

  This sensor detects the speed of the vehicle. It could be device like a Hall effect sensor that generates pulses as the wheels rotate. These pulses are counted by the microcontroller to calculate the speed.

- Odometer Sensor:

  Similar to the speed sensor, the odometer sensor measures the distance traveled by the vehicle. It could be a sensor that counts wheel rotations or detects motion to calculate the distance covered.

- Display Module:

  This component visually presents the speed and distance information to the driver. It could be an LCD display, showing real-time speed and distance data in a user-friendly format.

- Memory:

  Non-volatile memory stores settings, trip data, or other relevant information that needs to be retained even when the power is turned off. This could include EEP-ROM.

- Input Devices:

Buttons or switches allow the user to interact with the instrument cluster. For example, they might be used to toggle between trip meters or reset trip data.

- Peripheral Interfaces:

  These interfaces facilitate communication between the microcontroller and other components. For instance, CAN interfaces may be used to communicate with sensors and display modules.

- Power Supply:

  A stable power supply is essential for powering the microcontroller and other components. This could be provided by the vehicle's electrical system/Battery.

- Development Tools:

  Tools like STM32CubeIDE, debugger, and programmer are used for developing, debugging, and programming the firmware that controls the behavior of the microcontroller and integrates all the components into a cohesive system.

2.2 Flow Chart

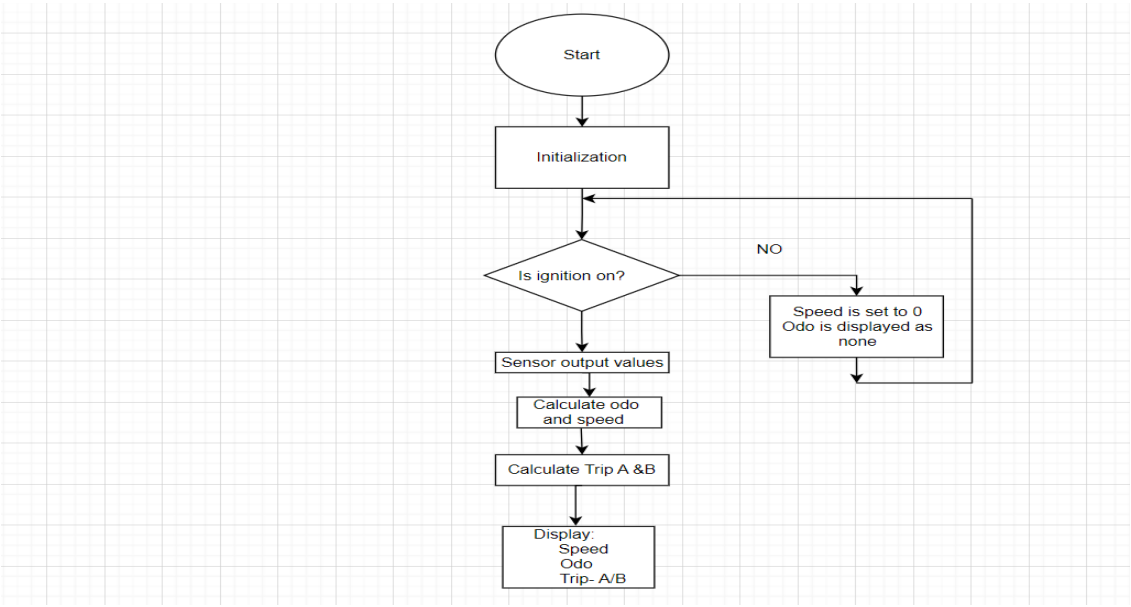


Figure 2.1: Flow Chart

Based on Figure 2.1 this flow Chart is designed to implement the Applications (Odometer,Speedometer) of Instrument Cluster Using STM32H735.It illustrates the series of event starting from the initialization to the display unit,above figure represents the basic flow chart of entire process.

- INITIALIZATION
- This involved the following tasks:
- Set the initial value of the odometer to zero.
- Ensure that the odometer mechanism is calibrated correctly to accurately track distance traveled.
- Verify that the odometer's sensor inputs (such as wheel rotation sensors) are functioning properly.

If applicable, enable the option to reset the odometer manually or automatically during specific intervals or maintenance procedures.

- Speedometer
- Initialization:
- Calibrate the speedometer to ensure accurate readings.
- Test the speedometer against known distances or GPS readings to verify its accuracy.
- Configure the speedometer to display readings in the desired units (e.g., miles per hour, kilometers per hour).
- Ensure that the speedometer is properly connected to the vehicle's sensors or source of speed data, such as wheel speed sensors or GPS modules.
- IS IGNITION ON?

  If the ignition is on, the system activates the input sensor to capture pulses, which are

then displayed on the output. However, if the ignition is off, the speed value display remains static at zero, and the odometer shows ’none’.

- CALCULATE ODO AND SPEED
- Odometer Calculation:
- The odometer tracks the total distance traveled by the vehicle.
- It receives input from sensors monitoring wheel rotations.
- Calculation is based on the no of pulses received per 100 meters and converting it into distance units (e.g., miles, kilometers).
- Odometer readings accumulate over time and are displayed to the user.
- Speedometer Calculation: The speedometer measures the vehicle’s instantaneous speed.
- Sensors monitor wheel speed, engine speed, or other relevant parameters.
- Calculation involves determining the rate of change of distance over time (e.g., miles per hour, kilometers per hour).
- Speedometer readings are continuously updated and displayed to the user in real-time.
- CALCULATE TRIPA AND TRIPB :

  The Trip readings displays the distance traveled during a particular trip since the last reset.The trip readings can be calculated by subtracting the odometer sensor’s value before reset from the present odometer value.

- DISPLAY SPEED,ODO,TRIP A/B
- The vehicle’s display provides essential information:
- Real-time speed display shows the current speed in kilometers per hour (km/h) or miles per hour (mph).

- The odometer continuously accumulates total distance traveled, presented in kilometers or miles, offering insights into overall vehicle usage.

- Two trip meters (TripA and TripB) independently track distances traveled for specific trips. Drivers can reset readings to monitor new trips, aiding in tracking travel distances for various purposes.

2.3 Block Diagram

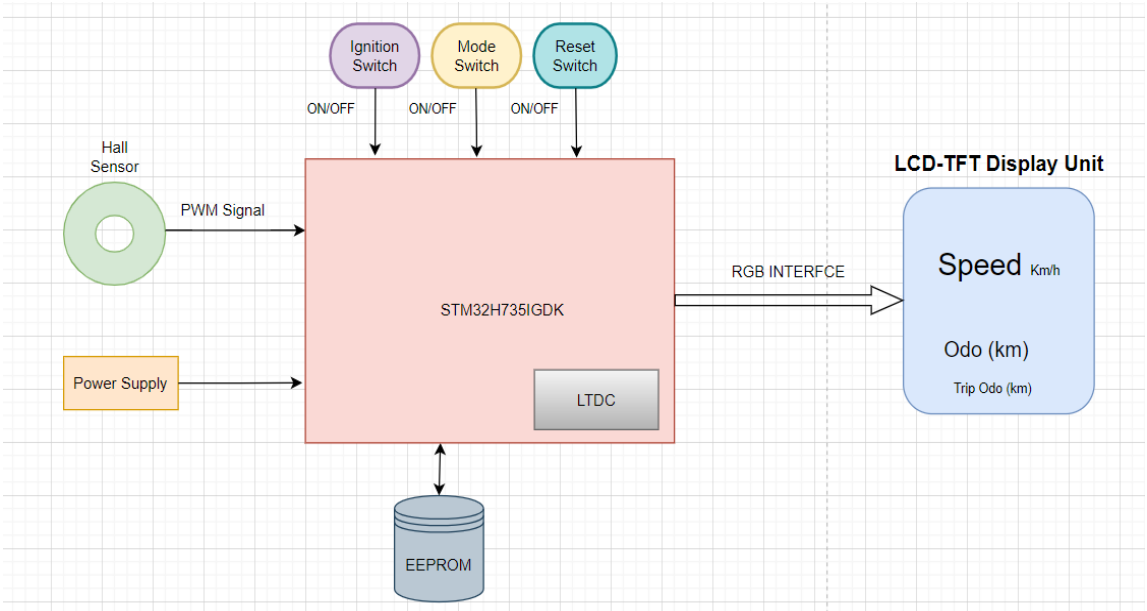


Figure 2.2: Block Diagram

Conclusion

In summary, the development of the advanced instrument cluster using the STM32H735IG-DK microcontroller demonstrates a meticulous process focused on functionality and user satisfaction. By integrating essential features like the odometer and speedometer with dynamic data handling capabilities, the cluster ensures clear and intuitive presentation of vehicle information. This structured approach not only enhances system responsiveness but also prioritizes driver engagement and safety, making it a robust solution for automotive applications.