VEHICLE CAN Implementation

Contents

[1. What is CANBus? 3](#_Toc212820314)

[Technical Features: 3](#_Toc212820315)

[Frame Structure: 3](#_Toc212820316)

[2. What is CAN in Automotive? 3](#_Toc212820317)

[Some ECUs in Automotive Sector: 4](#_Toc212820318)

[3. CAN Reading 5](#_Toc212820319)

[a. Differential Signaling Concept: 6](#_Toc212820320)

[b. 2. Voltage Levels (Classical CAN): 6](#_Toc212820321)

[c. Why Differential Works Well: 6](#_Toc212820322)

[d. Bus Termination: 6](#_Toc212820323)

[4. CAN in Embedded 6](#_Toc212820324)

[CAN Basics in Embedded 6](#_Toc212820325)

[5. CAN Implementation 7](#_Toc212820326)

[Steps for Implementation: 7](#_Toc212820327)

[6. References 8](#_Toc212820328)

Vehicle CAN Implementation

# What is CANBus?

CAN (Controller Area Network) is a high reliability serial communication protocol used in automotive and industrial applications. Developed by Bosch in the 1980s, it allows microcontrollers and devices to communicate quickly and securely without a host computer and data loss. It has multiple features alongside.

## Technical Features:

* Multi Master Architecture: Any node can initiate communication if the bus is idle. Which come from the importance levels that put for data transferring.
* Message-Oriented Protocol: Data is transmitted in frames with identifiers, not addressed to specific nodes.
* Bitwise Arbitration: Higher priority messages (lower ID value) access the bus first without collisions.
* Error Detection: Includes CRC, bit stuffing, ACK, and form checks.
* Speed & Length: Classical CAN supports up to 1 Mbps for up to 40 m

## Frame Structure:

Messages that CANBus holds is identifies as,

1. Start of Frame (SOF) – 1 bit
2. Identifier (ID) – 11 bits (Standard) or 29 bits (Extended)
3. Control Field – 6 bits (Data length code, flags)
4. Data Field – 0–8 bytes (classical CAN), up to 64 bytes in CAN FD
5. CRC Field – 15 bits
6. ACK Field – 2 bits
7. End of Frame (EOF) – 7 bits

# What is CAN in Automotive?

In modern vehicles, CAN networks interconnect ECUs, sensors, and actuators. It provides real time communication for every system. It works well both safety critical and non-safety critical systems.

## Some ECUs in Automotive Sector:

Electronic Control Unit (ECU): ECU is a general term for embedded controllers in a vehicle that manage specific functions by processing sensor inputs and commanding actuators. Each ECU typically has a microcontroller, memory, and communication interfaces. It builds with its own communicative parts. Can control engine timing, fuel injection, emissions, HVAC, or lighting depending on its specialization. Modern vehicles may have 50+ ECUs communicating over multiple CAN buses. The ECU performs real-time closed-loop control, often implementing PID controllers or LUTs (lookup tables)

Transmission Control Module (TCM): Specialized ECU that manages automatic transmission behavior. Controls gear shifts (upshift/downshift) using hydraulic solenoids or electronic actuators. Implements adaptive shift strategies for fuel efficiency or performance (this TCM works in mostly combustion vehicles) Inputs of the controllers are vehicle speed, engine RPM, throttle position, brake status, gear selector position.

Anti-lock Braking System (ABS): ECU that prevents wheel lock-up during braking, improving vehicle stability and control. Reads wheel speed sensors (typically 4x sensors). Detects impending wheel lock-up Modulates brake pressure using hydraulic valves to prevent skidding. ABS ECU operates at very high speeds (milliseconds response) and communicates wheel speed and brake status over CAN to other systems like stability control or traction control.

A car driving on a road

AI-generated content may be incorrect.When you abruptly hit the brakes, there's a possibility that one or more of your car's wheels could lock up, making it challenging to have control over your car. This is known as a wheel lock, and it can cause your car to skid. Back in the day, drivers were taught to physically slow it down when this happened to regain control.

Advantages in Automotive Systems Usage of CANBus:

* Reduced wiring complexity: One bus replaces point to point wiring.
* Deterministic Communication: High-priority messages are guaranteed bus access.
* Fault Tolerance: Error counters and retransmissions prevent message loss.
* Scalability: Additional nodes can be added with minimal reconfiguration.

# A diagram of a diagram AI-generated content may be incorrect.CAN Reading

Message Reception Process:

1. Physical Layer: Signal received via a CAN transceiver (e.g., MCP2551) from the differential bus lines CAN\_H and CAN\_L.
2. Data Link Layer: CAN controller checks. Messages failing error checks are discarded and retransmitted.
3. Application Layer: Decoding the payload according to the message ID and interpreting sensor or actuator data.

Example: Reading Engine RPM from CAN:

CAN ID = 0x0C8, Data[0:1] = RPM value (uint16\_t) → RPM = (Data[0] << 8 | Data[1])/4

Tools for CAN Reading:

* Hardware: CAN transceivers, logic analyzers, OBD-II adapters
* Software: Vector CANalyzer/CANoe, Python-CAN libraries, STM32 HAL CAN APIs

CANHigh and CANL- Differential Signaling:

## Differential Signaling Concept:

* CAN uses a differential pair: two wires called CANHigh and CANLow
* The signal is represented as a voltage difference between these two lines, not relative to ground. That is big confirmation.
* This design provides high noise immunity, which is essential in automotive environments with electrical interference from motors, ignition coils, or other electronics.

## 2. Voltage Levels (Classical CAN):

Recessive (Logic 1): 2.5V CAN\_H and 2.5V CAN\_L. As differential here comes 0V.

## Why Differential Works Well:

* Noise that appears equally on both lines (common-mode noise) is ignored by the CAN controller because it detects only the voltage difference.
* This makes CAN robust in electrically noisy environments like vehicles or factories.

## Bus Termination:

* Typical CAN bus has 120 Ω resistors at each end to match the characteristic impedance.
* Prevents signal reflections, ensures clean differential voltage, and stabilizes the bus.

# CAN in Embedded

## CAN Basics in Embedded

* CAN requires a transceiver to convert MCU logic levels to differential signals.
* Many MCUs such as STM32, PIC32, NXP, TI have integrated CAN controllers.
* Hardware Abstraction Layer (HAL): MCU-specific registers for CAN initialization, filtering, and interrupts, especially in STM32.
* CAN Driver: Handles message transmission, reception, and error management
* Middleware/Protocol Layer: Optional higher-layer protocols like CANopen or J1939 for structured communication

CAN Basic Configuration in STM32

CAN\_HandleTypeDef hcan1;

hcan1.Instance = CAN1;

hcan1.Init.Prescaler = 16;

hcan1.Init.Mode = CAN\_MODE\_NORMAL;

hcan1.Init.SyncJumpWidth = CAN\_SJW\_1TQ;

hcan1.Init.TimeSeg1 = CAN\_BS1\_1TQ;

hcan1.Init.TimeSeg2 = CAN\_BS2\_1TQ;

HAL\_CAN\_Init(&hcan1);

# CAN Implementation

## Steps for Implementation:

1. You should identify all nodes and required message flows as network design.
2. Assign standard/extended IDs; prioritize critical messages. You should do it by hand or by library example.
3. In most MCU implementation the structure goes like this: Microcontroller + transceiver + termination resistors (typically 120 Ω at bus ends).
4. Software:

* Configure CAN controller (bit timing, mode, bitwise mathematics)
* Implement TX/RX interrupt or polling mechanisms
* Error handling and logging

1. Bench testing with CAN simulator or in vehicle testing with diagnostic tools is a must for validation.

# References

**Linking Received Packet to the Transmitter Through Physical-Fingerprinting of Controller Area Network**

“Linking Received Packet to the Transmitter Through Physical-Fingerprinting of Controller Area Network,” ResearchGate, 2017 (or year of publication).

**L9615 CAN Bus Transceiver Datasheet**

STMicroelectronics, L9615 CAN BUS Transceiver Datasheet, 2013. [STMicroelectronics](https://www.st.com/resource/en/datasheet/l9615.pdf)

**CAN bus — Wikipedia**

“CAN bus,” Wikipedia, accessed [date you accessed], [https://en.wikipedia.org/wiki/CAN\_bus](https://en.wikipedia.org/wiki/CAN_bus?utm_source=chatgpt.com).

Bosch, *CAN Specification Version 2.0*, 1991