CAN Protocol

Alex Eagles

Contents

Table of Figures

What is CAN Protocol

CAN stands for **Controller Area Network** protocol. It is a protocol that was developed by Robert Bosch in around 1986. The **CAN** protocol is a standard designed to allow the microcontroller and other devices to communicate with each other **without** any host computer. The feature that makes the CAN protocol unique among other communication protocols is the **broadcast** type of bus. Here, broadcast means that the information is transmitted to **all** the nodes. The node can be a sensor, microcontroller, or a gateway that allows the computer to communicate over the network through the **USB** cable or **ethernet** port. The **CAN** is a **message-based** protocol, which means that message carries the **message identifier**, and based on the identifier, **priority is decided**. There is **no** need for node identification in the **CAN** network, so it becomes very easy to insert or delete it from the network. It is a **serial half-duplex** and **asynchronous** type of communication protocol. The **CAN** is a **two-wired** communication protocol as the **CAN** network is connected through the two-wired bus. Initially, it was mainly designed for communication within the vehicles, but it is now used in many other contexts. Like **UDS**, and **KWP 2000**, **CAN** also be used for the on-board diagnostics.

How does CAN work

The **CAN** bus is a **decentralized** communication protocol. Its decentralized approach makes it **ideal** for applications in automotive and industrial systems where reliability and real-time performance are essential.

In a **CAN** network, all nodes are connected via **twisted-pair** wiring or optical fiber cables. Each node has its **own microcontroller** responsible for processing incoming messages and sending outgoing ones. Data is **broadcasted** by a node on the shared bus, allowing all other nodes to receive it.

The primary stages of the communication process are:

Arbitration: To prevent collisions when multiple nodes attempt to transmit **simultaneously**, **CAN** uses an **arbitration** process based on message priority. The **lower** the identifier value of a message, the higher its priority.

Error detection: Built-in error detection mechanisms ensure data integrity within **CAN** networks. These include cyclic redundancy checks **(CRC)**, frame check sequences (**FCS**), and **acknowledgment** bits from receiving nodes.

Fault confinement: If any node detects an error or malfunctions during transmission, it will enter an "**error passive**" state until proper operation resumes. This prevents faulty transmissions from affecting overall system functionality.

This combination of features allows **CAN** buses to maintain **high** levels of efficiency while ensuring reliable communication between different components in complex systems like vehicles or factory automation equipment.

CAN Frame Structure

- **SOF**: **SOF** stands for the **start** of frame, which indicates that the new frame is entered in a network. It is of **1** bit.
- **Identifier**: A standard data format defined under the **CAN 2.0 A** specification uses an **11-bit** message identifier for a**rbitration**. Basically, this message identifier sets the priority of the data frame.
- **RTR**: **RTR** stands for **Remote Transmission Request**, which defines the frame type, whether it is a data frame or a remote frame. It is of **1-bit**.
- **Control field**: It has user-defined functions.
    - **IDE**: An IDE bit in a **control field** stands for identifier extension. A **dominant IDE (0)** bit defines the 11-bit

standard identifier, whereas **recessive IDE (1)** bit defines the **29-bit** extended identifier.

  - **DLC**: **DLC** stands for **Data Length Code**, which defines the data length in a data field. It is of **4 bits**.
  - **Data field**: The data field can contain up to **8** bytes.

- **CRC field**: The data frame also contains a **cyclic redundancy check** field of **15** bit, which is used to detect the **corruption** if it occurs during the transmission time. The sender will compute the **CRC before** sending the data frame, and the receiver also computes the **CRC** and then compares the computed **CRC** with the **CRC** received from the sender. If the **CRC does not match**, then the receiver will **generate the error**.
- **ACK field**: This is the **receiver's acknowledgment**. In other protocols, a separate packet for an acknowledgment is sent after receiving all the packets, but in case of **CAN** protocol, **no** separate packet is sent for an acknowledgment.
- **EOF**: **EOF** stands for end of frame. It contains **7** consecutive recessive bits known End of frame.

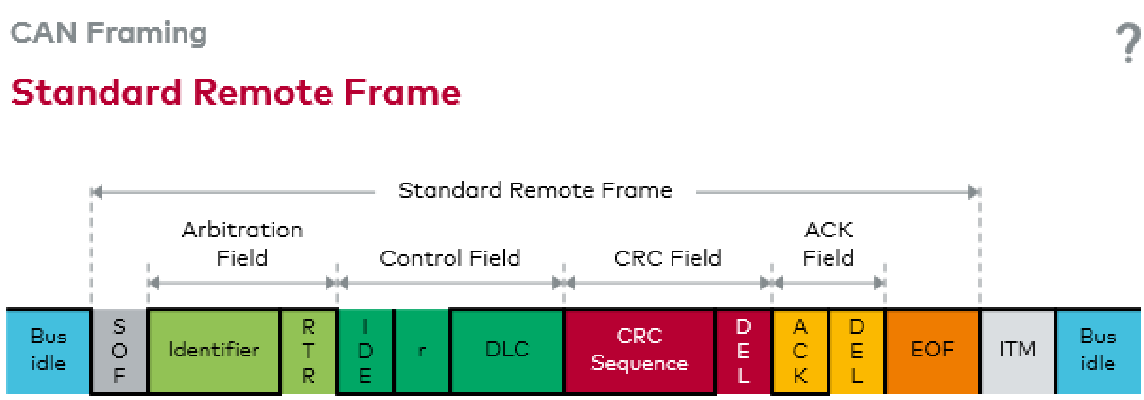

Figure 1 CAN Frame Structure

CAN Node Structure

A **CAN** network consists of multiple of **CAN** nodes. In the below case, we have considered three **CAN** nodes, and named them as **node A**, **node B**, and **node C**. **CAN** node consists of three elements which are given below:

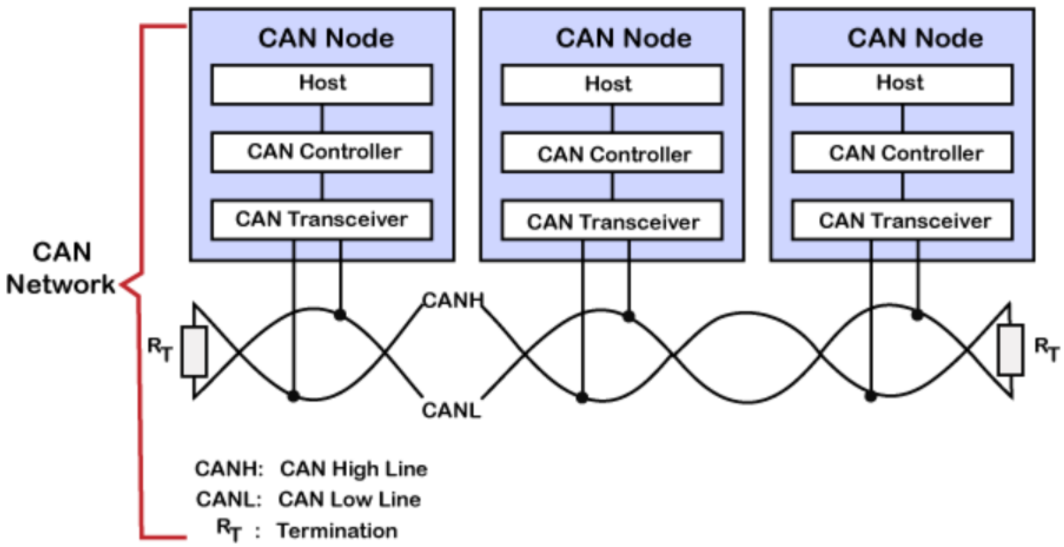

Figure 2 Nodes in CAN

- **Host**
  A **host** is a microcontroller or microprocessor which is running some application to do a specific job. A host decides what the received message means and what message it should send next.
- **CAN Controller**
  CAN controller deals with the communication functions described by the **CAN** protocol. It also triggers the **transmission**, or the **reception** of the **CAN** messages.
- **CAN Transceiver**
  CAN transceiver is responsible for the transmission or the reception of the data on the **CAN** bus. It converts the data signal into the

stream of data collected from the **CAN** bus that the **CAN** controller can understand.

In the above diagram, unshielded twisted pair cable is used to transmit or receive the data. It is also known as **CAN** bus, and **CAN** bus consists of two lines, i.e., **CAN low line** and **CAN high line**, which are also known as **CANH** and **CANL**, respectively. The transmission occurs due to the differential voltage applied to these lines. The **CAN** uses twisted pair cable and differential voltage because of its environment. For example, in a car, motor, ignition system, and many other devices can cause **data loss and data corruption due to noise**. The twisting of the two lines also reduces the magnetic field.

CAN Differential Signals

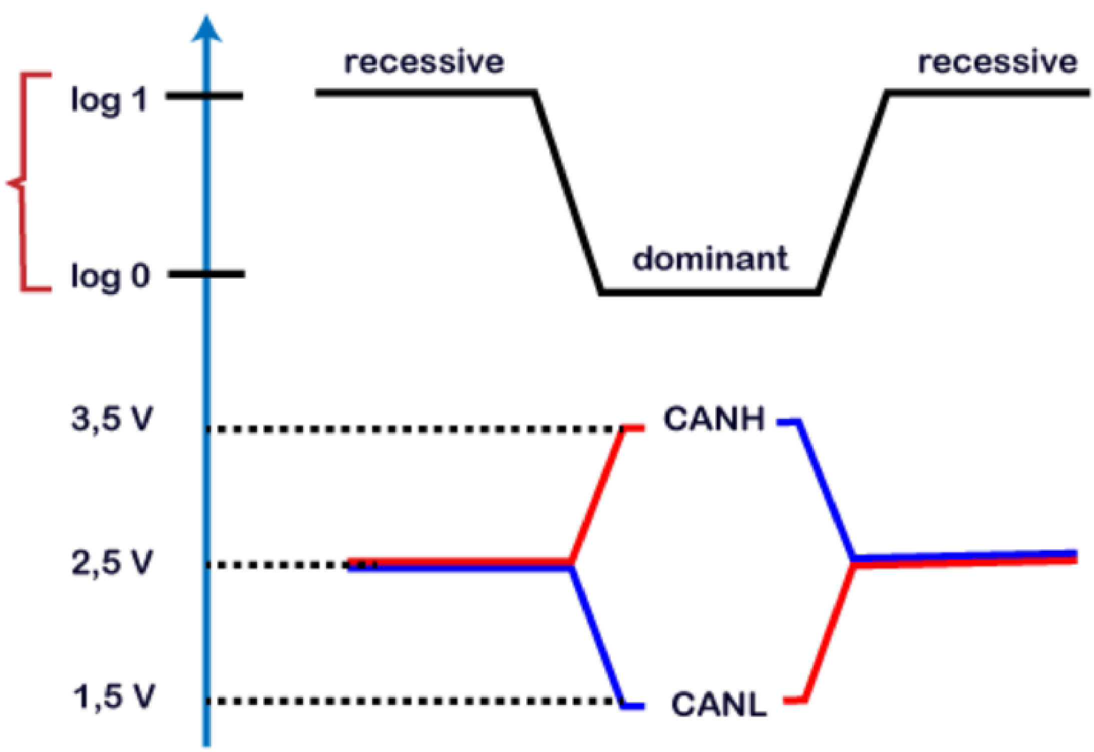

Figure 3 CAN Differential Signals

With the help of differential voltage, we will determine how 0 and 1 are transmitted through the **CAN** bus. The above figure is the voltage graph that shows the voltage level of **CAN low** and **CAN high**. In **CAN** terminology, logic **1** is said to be **recessive** while logic **0** is **dominant**. When **CAN** high line and **CAN** low line are applied with **2.5** volts, then the actual differential voltage would be **zero** volt. A zero volt on **CAN** bus is read by the **CAN** transceiver as a **recessive** or logic 1. A z**ero** volt on **CAN** bus is an **ideal** state of the bus. When **CAN** high line is pulled up to **3.5** volt and the **CAN** low line is pulled down to **1.5** volt, then the bus's actual differential voltage would be **2** volts. It is treated as a **dominant** bit or logic **0** by the **CAN** transceiver. If the bus state is reached to **dominant** or logic **0** then it would become impossible to move to the recessive state by any other node.

Key points learnt from the CAN characteristics

- Logic **1** is a **recessive** state. To transmit **1** on **CAN** bus, **both CAN high and CAN low** should be applied with **2.5V**.
- Logic **0** is a **dominant** state. To transmit **0** on **CAN** bus, **CAN high** should be applied at **3.5V** and **CAN low** should be applied at **1.5V**.
- The ideal state of the bus is **recessive**.
- If the node reaches the **dominant** state, it **cannot** move back to the recessive state by any other node.

CAN Applications

Automotive Industry:

- Used in passenger vehicles, heavy-duty vehicles, and agricultural machinery.
- Enables communication among ECUs (Engine Control Units) without one-to-one wiring.

- Commonly used for engine management, body electronics, in-vehicle diagnostics (ISO-15765), entertainment systems, and electric/hybrid vehicle systems (e.g., battery management, servo controllers).
- Extensively utilized in agriculture machines like excavators and forklifts, and truck-based cranes.

Public Transport:

- Employed in high-speed trains for braking systems, door control, diagnostics, and passenger information systems.
- Used in aircraft for engine control systems, flight sensors, and navigation systems.
- Applied in road transport for traffic management and surveillance, and in maritime electronics.

Industrial Automation:

- Integral to control systems in manufacturing equipment, robotics, packaging, textile processing, and assembly lines.
- CAN-based protocols like CANopen, DeviceNet, and Smart Distributed Systems are commonly used.

Building Automation:

- Utilized for controlling elevators, HVAC systems, automatic doors, lighting, and other sub-systems in smart buildings.

Medical Electronics:

- Applied in medical devices like X-ray machines, CT scanners, and for networking in intensive care units and operating theaters due to its reliability and error detection capabilities.

Embedded Electronics:

- Used as a system bus in consumer appliances like washing machines, vending machines, and audio-video systems.

Figure 4 CAN Applications

CAN Advantages and Disadvantages

CAN Advantages:

- Short, high message frequency, more than 10,000/s.
- High bandwidth utilization.
- Reasonable transmission speeds.
- Support for higher-layer protocols like CANopen (standardized protocol for devices and applications from different manufacturers) and J1939 (standard for heavy-duty vehicles).

CAN Disadvantages

- Because of electrical loading, the number of connected devices is limited to a maximum of **64** nodes.

- Cable length is limited to **40** meters (a touch over 131 feet) in length, which is not a problem for most use cases, but could limit some applications.
- According to the standard, the maximum speed is 1 Mbit/second, although this limitation has been solved in CAN FD, which offers 5 Mbit/s.
- **CAN** produce excessive electric noise.
- While **CAN** reduces some costs, software development and maintenance expenses can be high.