# Serial Bus Systems in the Automobile

## Part 2:

## Reliable data exchange in the automobile with CAN

**The relentless pace of globalization has brought growing competitive pressure to bear on automotive OEMs and suppliers, which in turn leads to one innovative offensive after another. Electronics plays a decisive role here: Increasingly complex electronic systems provide for a high level of safety and comfort in car driving. The CAN (Controller Area Network) serial bus system makes a crucial contribution here with to its specific properties. It assures reliable data exchange even under harsh environmental conditions for example. This technical article is intended to serve as an introduction to CAN technology.**

### CAN standard, implementation and interface

The CAN technology developed by Bosch [1] has been standardized since 1993 and exists as ISO Standard 11898 which is organized in several parts (Figure 1). The first part contains the CAN protocol and covers the entire data link layer (framing, addressing, bus access, data assurance) and part of the physical layer (physical signaling) of the standardized reference model for data communication (ISO 7498). In the meantime a large number of cost-effective CAN controllers have become available which implement the CAN protocol in hardware.

The second part describes the CAN High-Speed physical layer, and the third part the CAN Low-Speed physical layer. These two parts cover the Physical Layer of ISO 7498 (including physical bus interface, data rates and voltage levels). The CAN High-Speed physical layer is used primarily in powertrain and chassis applications. It is essentially implemented by the CAN High-Speed transceiver, which supports a maximum data rate of 1 MBit/s. The CAN Low-Speed transceiver with a maximum data rate of 125 KBit/s is generally used for the CAN Low-Speed physical layer that is primarily used in the body/convenience area.

Accordingly the CAN interface (Figure 2) consists of a CAN controller and a CAN transceiver. While the CAN controller handles the CAN protocol, the CAN transceiver assumes the task of physically coupling the CAN controller to the CAN bus operated in differential signal mode. Differential signal transmission enhances noise immunity and requires two

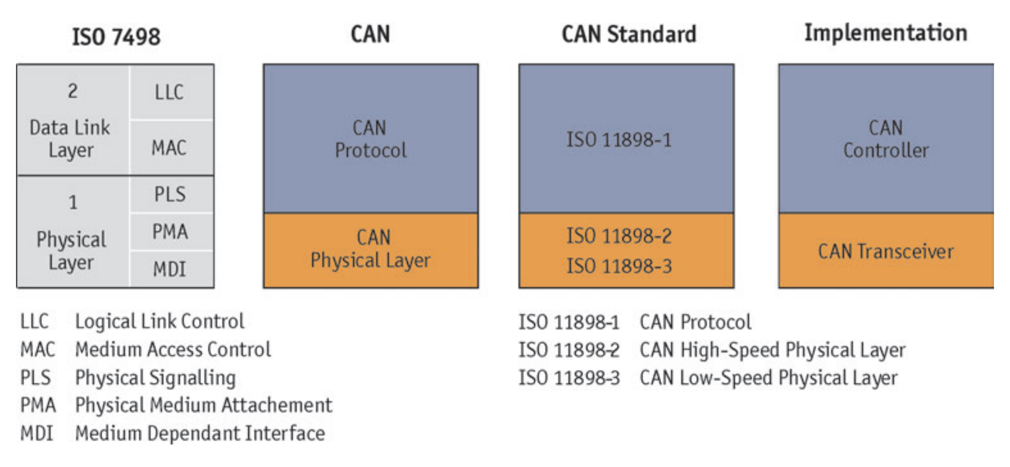


**Figure 1: CAN in the reference model for data communication (ISO 7498), CAN standard and implementation.**

communication lines (CAN-High and CAN-Low line), which are terminated with the characteristic line impedance to avoid reflections at the ends.

## Message distribution

Message addresses and message filters are used in a CAN network to organize nodes and messages. Message addresses, commonly referred to as Identifiers (ID) do not identify the CAN target nodes, rather they identify the messages themselves, so that in principle all CAN messages are available to be received by all CAN nodes (message distribution). By means of a filter each CAN node selects those CAN messages from the message stream that are relevant to it (receiver-selective system). The 11 bit wide ID permits specification of up to 2048 CAN messages in a CAN network.

Message distribution offers the following advantages:

- Cost savings by shared use of sensors,
- Easy implementation and synchronization of distributed processes, and above all:
- High flexibility with regard to the configuration.

This is because omitting node addresses makes it possible to integrate other bus nodes without having to modify the hardware or software of existing bus nodes. However this is only true if the added bus node is exclusively a receiver.

## Event control

The messages that are transmitted in a CAN network and their sequence do not depend on a time progression, rather they depend on the occurrence of special events. Each CAN node is in principle authorized to access the CAN bus immediately after an event occurs. Given its relatively short message length of max. 130 bits in standard format and its high data transfer rate of up to 1 MBit/s, this method enables quick reactions to asynchronous events. This is an important precondition for real-time capable data transmission in the millisecond range (1 to 10 ms), which is primarily a requirement of powertrain and chassis applications.

Since CAN communication is not based on any time schedule the message traffic is not determined until runtime, and this implies that it carries the inherent risk of collisions. This risk increases with increasing bus load, and it calls the real-time capability of the system into question. To assure real-time data transmission in spite of random bus access, the CSMA/CA bus access method is utilized in the CAN network.

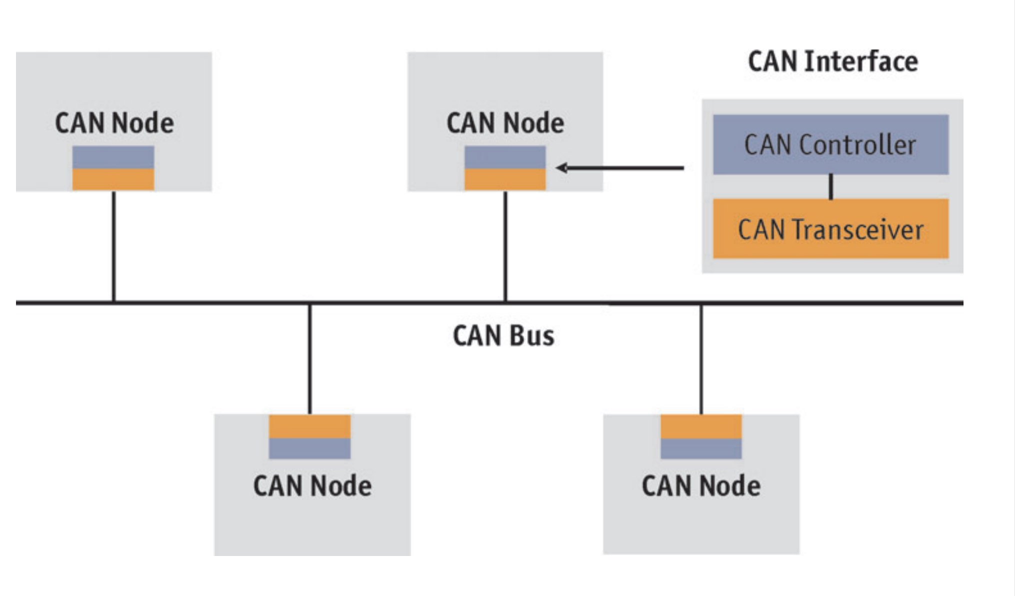


**Figure 2: CAN interface and CAN network.**

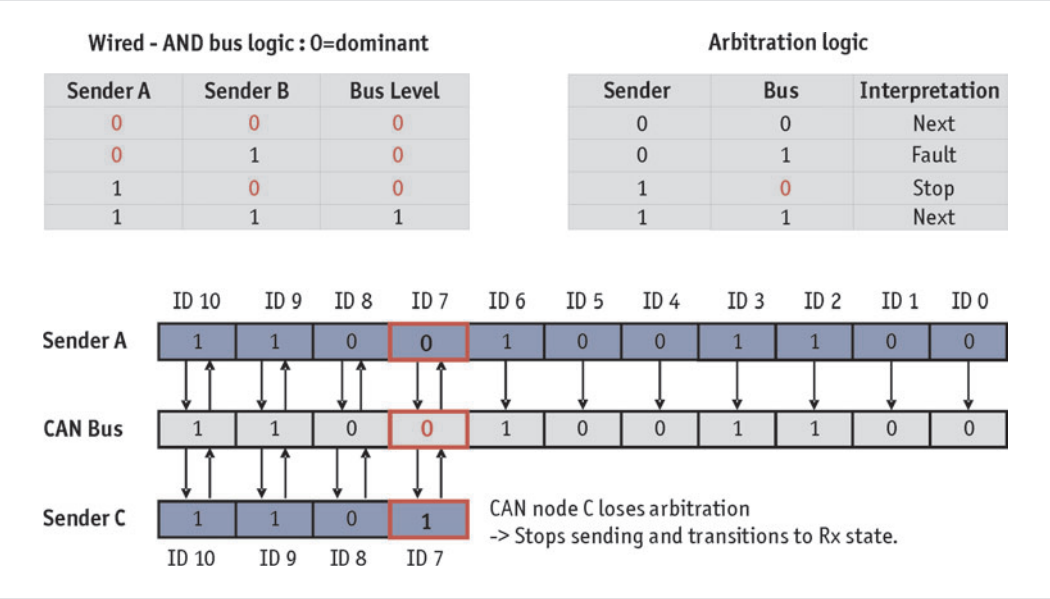
## CSMA/CA bus access method

Bus access begins when a CAN node wishing to send first listens to the CAN bus (Carrier Sense – CS). If the CAN bus is available the CAN node may begin to transmit its message immediately. On the other hand, if it detects bus activity it must postpone its send request until the CAN bus is available and the currently running message transmission has been completed; in addition it must wait a duration of three bit times (ITM Intermission). An ongoing message transmission is not interrupted in this method – bus access is nondestructive.

If there are multiple CAN nodes wishing to send, bitwise arbitration (Figure 3) prevents collisions from occurring in spite of simultaneous bus access (Multiple Access – MA). In the framework of bitwise arbitration all CAN nodes wishing to send place the IDs of the CAN messages they wish to send bitwise on the bus, from the highest to the lowest significant bit. The wired-AND bus logic (0=dominant) that forms the basis of the CAN network ensures that there is always an unambiguous bus level. After adding on an ID bit, each CAN node compares the bus level with the level it sent. The arbitration logic decides whether a CAN node may continue to send or whether it must stop sending. At the end of the arbitration phase the CAN node that gets send authorization is the node transmitting the CAN message with the least significant identifier. Lower priority CAN nodes first switch to the Rx state and access the CAN bus for a renewed send attempt as soon as the bus is free again.

Not only do the bus and arbitration logic prevent collisions (Collision Avoidance – CA), they also provide priority-controlled bus access: The lower the significance of the identifier, the higher the priority of the CAN message, and this results in faster bus access. The CAN message with the smallest identifier (ID=0) will therefore be transmitted without delay.

If the bus load is not too high, this type of random, nondestructive, priority-driven bus access facilitates correct and very fast bus access. On the one hand, it should be noted that delays grow with increasing bus load, above all delays of low priority CAN messages. In the worst case a situation may arise in which CAN messages arrive too late at receivers or are suppressed entirely. On the other hand, the CSMA/CA bus access method produces a reciprocal relationship between network extension and maximum data rate. During bitwise arbitration a recessively sending CAN node must be able to



Wired - AND bus logic : 0=dominant

| Sender A | Sender B | Bus Level |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Arbitration logic

| Sender | Bus | Interpretation |
|---|---|---|
| 0 | 0 | Next |
| 0 | 1 | Fault |
| 1 | 0 | Stop |
| 1 | 1 | Next |

| | ID 10 | ID 9 | ID 8 | ID 7 | ID 6 | ID 5 | ID 4 | ID 3 | ID 2 | ID 1 | ID 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Sender A | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| CAN Bus | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| Sender C | 1 | 1 | 0 | 1 | | | | | | | |

**Figure 3: Wired-AND bus logic, arbitration logic and bitwise arbitration based on example of two CAN nodes.**

reliably detect a dominant level. The bit time interval should therefore be sized such that signal propagation times on the CAN bus are fully compensated. A length extension to a network therefore necessitates a longer bit time interval, which in turn defines a maximum usable data rate.

## Data transmission

It is primarily Data frames that are responsible for data transmission in the automobile (Figure 4). While in fact Remote frames also exist for requesting data, they are hardly ever used since data transmission in the automobile is not typically request-based, rather it is primarily provided on the information generator's own initiative. The two types of frames have identical layouts; the only difference is that the data field is omitted in the Remote frame.

A basic prerequisite for transmitting Data and Remote frames is synchronism between sender and receiver. Since a clock line has been omitted for reasons of cost and effort, synchronism is achieved by signal edges and a well-defined resynchronization mechanism. Each message transmission begins with transmission of the dominant synchronization bit (SOF – Start of Frame) and this generates the first signal edge (Bus-Idle exhibits a recessive bus level). The receiver ensures synchronization over the entire transmission by evaluating each arriving signal edge and adapting its own bit timing as necessary. The bit stuffing method ensures that a complementary bit (stuff bit) appears at the latest after five homogeneous bits, thereby providing a signal edge.

Following the SOF is the ID, which may be either 11 bits (Standard-ID) or 29 bits (Extended-ID) in length. The standard format dominates in the automotive field. The extended format typically plays a role in conjunction with higher level protocols such as SAE J1939. The ID format being used is indicated by the IDE (Identifier Extension) bit. Another bit switch (RTR bit – Remote Transmission Request) indicates whether the frame is a Data or Remote frame.

The 64 bit wide data field is available for transmitting useful information, in which the exact number of useful bytes is indicated by a DLC (Data Length Code). Following the data field is the so-called CRC sequence (CRC – Cyclic Redundancy Check). The sender generates the CRC sequence based on all bits to be transmitted, a generator polynomial and a well-defined algorithm. Independent of the message filtering the same process occurs at the receiving end with the arriving bits. The two sequences are compared, and the acknowledgment is made after the recessive CRC delimiter in the Acknowledge slot (ACK slot). At the end of a Data frame, after the recessive ACK delimiter, comes the seven bit long and recessive EOF (End of Frame).

## Data protection

The probability that corrupted CAN messages will remain undetected is extraordinarily low. It is estimated to be 4.7 x $10^{-11}$ [2]. Responsible for this are error detection mechanisms defined in the CAN protocol. On the receiver side, besides the message-filtering-independent CRC that is capable

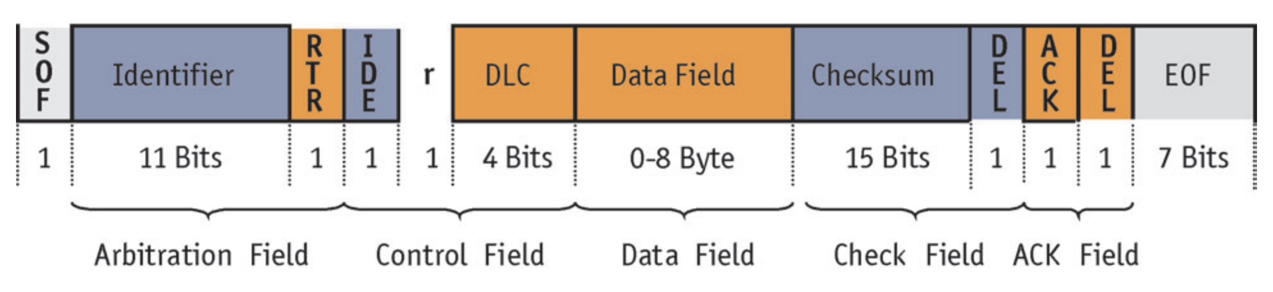


**Figure 4: Structure of the Data frame.**

of detecting up to five errors within a CAN message, checks are also made of the format (Form Check) and bit stuffing rule (Stuff Check). The sender performs bit monitoring and evaluates the ACK slot.

If one assumes an error rate of $10^{-3}$ in a CAN network, then given an annual operating time of 1000 hours, a data rate of 500 KBit/s, a mean bus load of 25 percent and a mean message length of 80 bits, statistically speaking a corrupted CAN message will remain undetected by the CAN protocol just once every 4000 years. What is understood as the error rate is the ratio of corrupted CAN messages to the number of all transmitted CAN messages.

As soon as an error detection mechanism signals a transmission error, the CAN node detecting the error terminates message transmission by placing an error flag (six dominant bits) on the CAN bus. The error flag intentionally violates the bit stuffing rule so that network-wide each CAN node perceives what until then was a local error and responds by terminating the message transmission, i.e. by appending an error flag. This method assures network-wide data consistency which is so important in distributed applications.

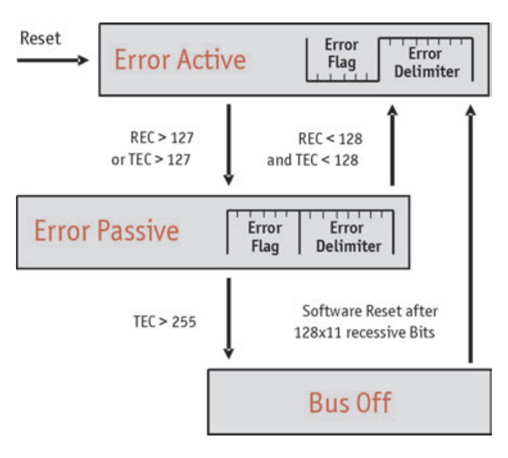


**Figure 5:**
**Network node monitoring using two error counters and three node states.**

Error correction consists of repetition of the aborted CAN message by the same sender as soon as the CAN bus is free again (after the error delimiter and ITM). In designing the system it must be considered that the CSMA/CA bus access method does not guarantee immediate repetition. The error recovery time depends on the priority of the message and the bus load.

## Node monitoring

Error signaling by error flag gives each CAN node the capability of terminating ongoing message transmissions. Since this also applies to defective CAN nodes, such nodes are capable of bringing the entire CAN communication to a standstill. To prevent this each CAN node has network node monitoring (Figure 5) that can disconnect (Bus off) a node found to be defective based on error counters and rules for controlling the error counters.

## Acknowledgment of received CAN messages

In a CAN network each message transmission is acknowledged simultaneously by all receivers in the ACK slot (in-frame acknowledgement), independent of message filtering. A dominant level signifies to a positive acknowledgment, and a recessive level signifies a negative acknowledgment. Since the sender places a recessive level in the ACK slot, just one positive acknowledgment is sufficient to confirm a correct message transmission. Because of this node-neutral positive acknowledgement, negatively acknowledging CAN nodes are overwritten and remain unheard. Therefore they send an error flag after the ACK delimiter.
If not a single positive acknowledgment is received, that is the ACK slot is not overwritten by any receiver, the sender detects an ACK error and aborts the ongoing message transmission by sending an error flag.

## Outlook

Until just a few years ago CAN was the most sought after bus technology in the automotive industry. The relentless electronification of the vehicle has caused CAN to encounter limits. Vehicle developers are questioning the suitability of the CAN bus especially in bandwidth-intensive, real-time, critical and highly safety-critical motor vehicle applications such as the “Lane Keeping Assistance” driver assistance system, but also in cost-sensitive convenience applications.

Therefore besides CAN two other bus technologies have become established over the course of time for use in the automobile or they are on an ideal course in that direction. We are talking about LIN and FlexRay here. LIN (Local Interconnected Network) is already being used for cost-effective networking of sensors and actuators in the convenience area. FlexRay is on the verge of being implemented in real-time and safety-critical automotive applications due to its time-triggered communication method, a data rate of up to 20 MBit/sec and the possibility of sending over two communication channels. In its first production application worldwide FlexRay will be implemented in an active suspension control system on the new BMW X5.

## Reliable ECU networking and data exchange

The specialists at Vector Informatik [3] support automotive OEMs and suppliers, not only in CAN networking but also in the LIN, FlexRay and MOST bus systems. For customer projects we offer universal tool chains of design and development tools, optionally with software components and base software for AUTOSAR control modules. These products are supplemented by customer support, consulting services and tools for process management in various application areas that enable customized adaptations to specific requirements. These services are rounded out by a comprehensive training program covering Vector tools, software components and serial bus systems.

For an introduction to ECU networking or data exchange in the automobile the Stuttgart-based company offers the one-day seminar “Serial bus systems in the automobile”. Fundamentals seminars on CAN, LIN, FlexRay and MOST convey the basic knowledge needed to quickly gain familiarity with the many different development activities related to automotive electronics [4].

The first part of this series of articles [5] addressed serial bus systems in the automobile in general terms. Upcoming articles three through five will discuss the LIN, FlexRay and MOST serial bus systems. Interested readers will find supplemental and in-depth information on these topics that has already been published at the Internet site of the Vector Academy [4].

**Literature and Internet links:**

[1] www.bosch.com
[2] Unruh, J., Mathony, H.J., Kaiser, K.H.: Error Detection Analysis of Automotive Communication Protocols, SAE International Congress 1990.
[3] www.vector-informatik.de
[4] www.vector-academy.de
[5] Mayer, E.: Serielle Bussysteme im Automobil – Architektur, Aufgaben und Vorteile [“Serial bus systems in the automobile – Architecture, tasks and advantages”]. Elektronik Automotive 7/2006, pp. 70ff.

**Author:**

**Eugen Mayer** (Graduate Engineer with Technical Teaching Certificate), after completing his vocational training to become a communications technician, studied electronics at the Technical College in Ravensburg/Weingarten, Germany, and studied electrical engineering and vocational teaching at the University of Stuttgart. Since 1999 he has been working at Vector Informatik where he is employed as a Senior Trainer.
Tel. +49-711/80670-574, Fax +49-711/80670-111,
E-Mail: eugen.mayer@vector-informatik.de