

The CAN protocol



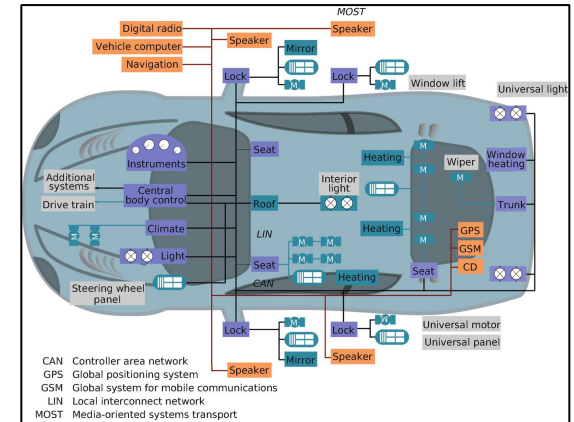
By: Yehia M. Abu Eita

Outlines

- **What is CAN bus?**
- **Why CAN bus?**
- **CAN Node**
- **CAN bus connection**
- **CAN communication principle**
- **CAN standards**
- **CAN arbitration**
- **CAN frames**
- **CAN Logical Error detection and handling**
- **CAN Error tracking**

What is CAN bus?

- CAN is referred to a **C**ontroller **A**rea **N**etwork.
- It is a **serial communication protocol** that efficiently supports distributed real-time control with a **very high-level of security**.
- It is one of the most famous protocols used in automotive industry.

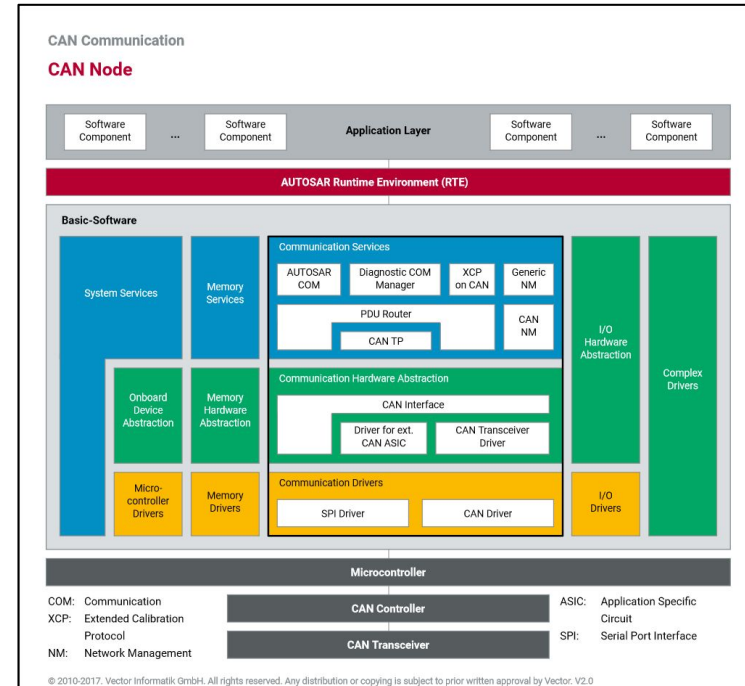


Why CAN bus?

- **Low cost**
 - ECUs Communicate via a single CAN interface.
- **Centralized**
 - Central error diagnosis and configuration across all the ECUs.
- **Robust**
 - CAN bus is robust towards the failure of sub-systems and electromagnetic interference.
- **Efficient**
 - CAN messages are prioritized based on IDs, the highest priority IDs are non-interrupted messages.
- **Flexible**
 - Each ECU contains a chip that receives all the transmitted messages, decide relevance and act accordingly.

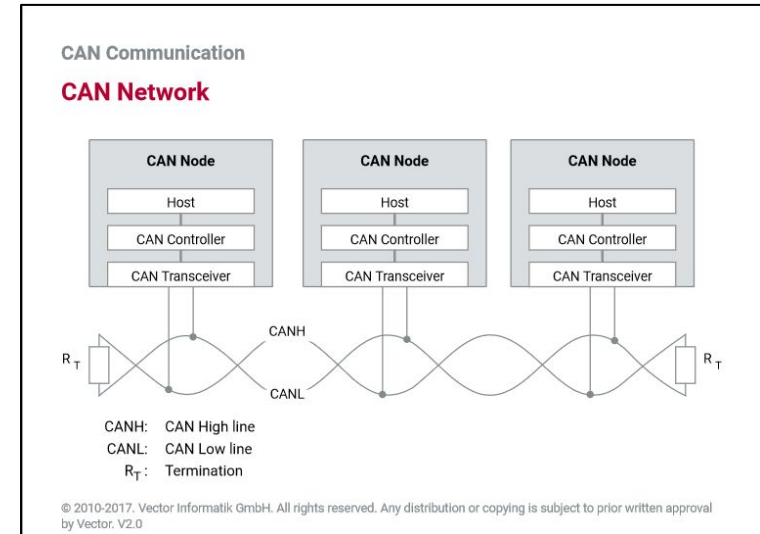
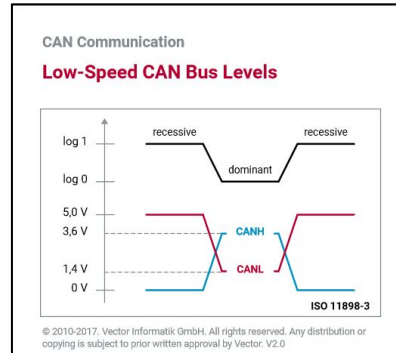
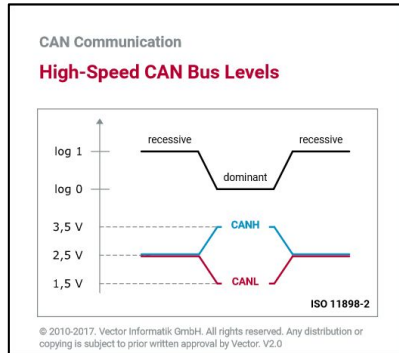
CAN node

- CAN Transceiver
- CAN Controller
- Host



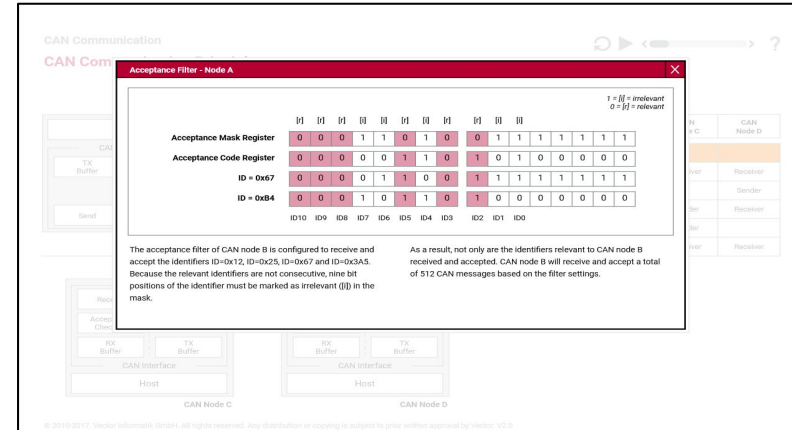
CAN bus connection

- CANH
- CANL



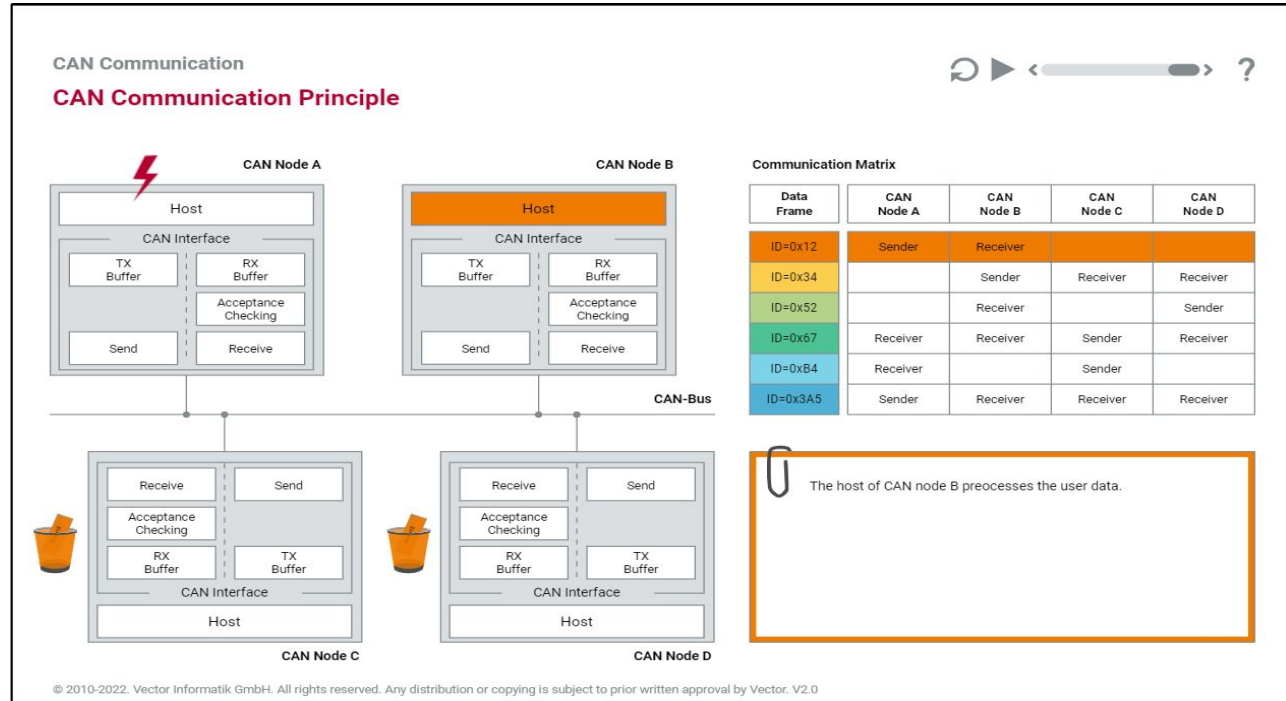
CAN communication principle

- Each CAN node has an **acceptance filter**.
- The acceptance filter contains **message IDs** for each node.
- If the **message ID** is **not** existed, **no response**.
- If the **message ID exists**, the **node will respond**.



CAN communication principle

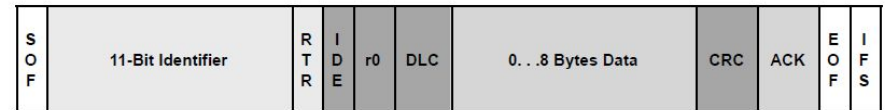
- Use case



CAN standards

- **Standard CAN 2.0A:**

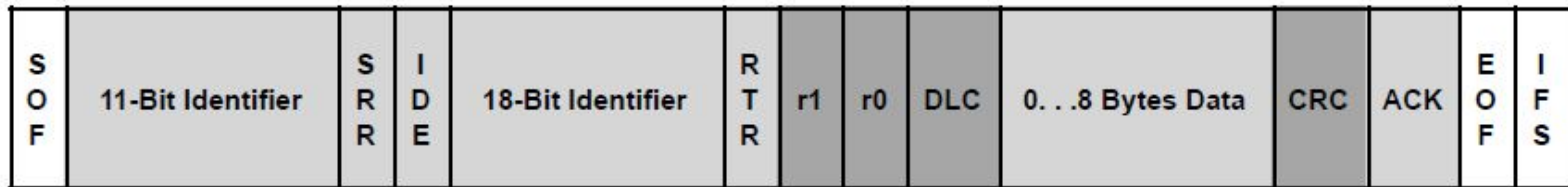
- **SOF** —The single dominant start of frame (SOF).
- **Identifier** —The Standard CAN 11-bit identifier establishes the priority of the message.
- **RTR** — The single remote transmission request (RTR).
- **IDE**— A dominant single identifier extension.
- **r0** — Reserved bit (for possible use by future standard amendment).
- **DLC** — The 4-bit data length code.
- **Data** — Up to 64 bits of application data may be transmitted.
- **CRC** — The 16-bit (15 bits plus delimiter).
- **ACK** — ACK is 2 bits, (1 bit plus delimiter).
- **EOF** — This end-of-frame (EOF) 7-bit field.
- **IFS** — This is at least 3 recessive bits inter-frame space.



CAN standards

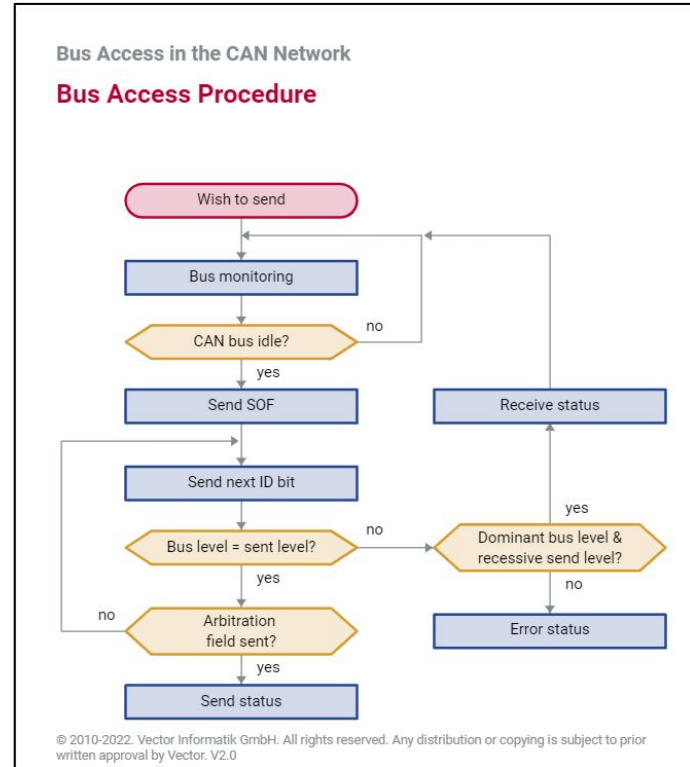
- **Extended CAN 2.0B:**

- **SRR** — The substitute remote request (SRR) bit replaces the RTR bit in the standard message location as a placeholder in the extended format.
- **IDE** — A recessive bit in the identifier extension (IDE) indicates that there are more identifier bits to follow. The 18-bit extension follows IDE.
- **r1** — Following the RTR and r0 bits, an additional reserve bit has been included ahead of the DLC bit.



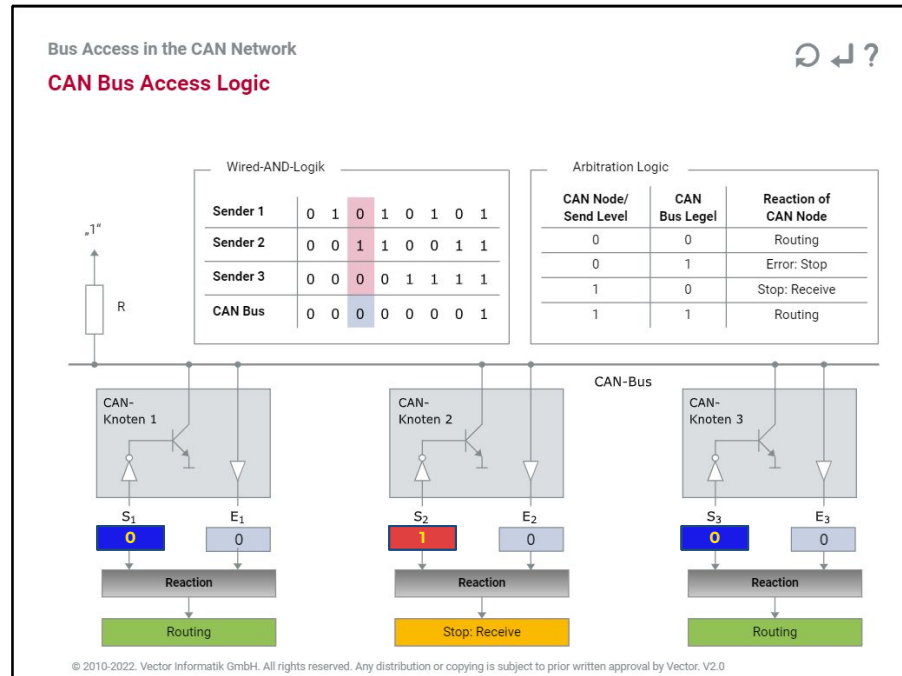
CAN arbitration

- Bus Access Procedure



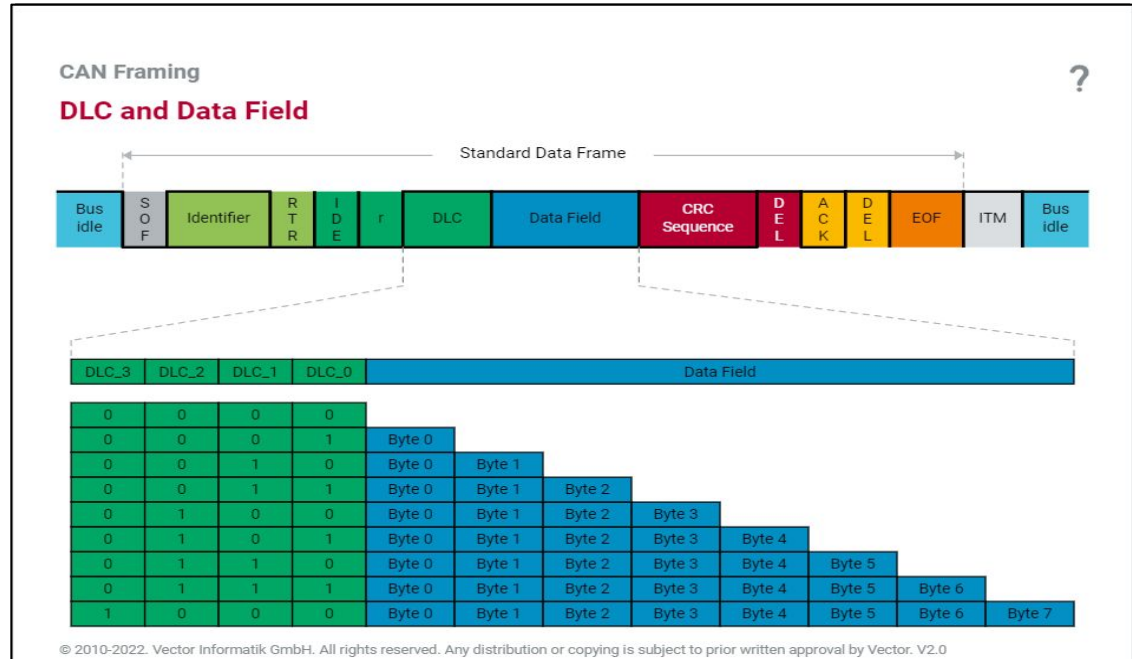
CAN arbitration

- Bus Access Logic



CAN frames

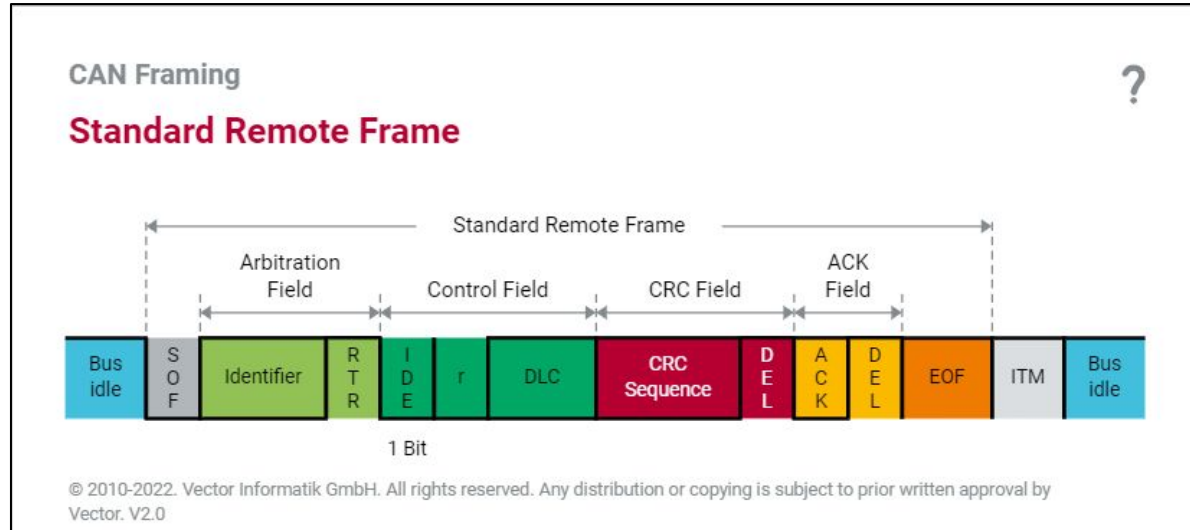
- **Data frame:**
 - For transmitting user data, **ISO 11898-1** prescribes the so-called data frame.
 - It can transport a **maximum** payload of **eight bytes**.



CAN frames

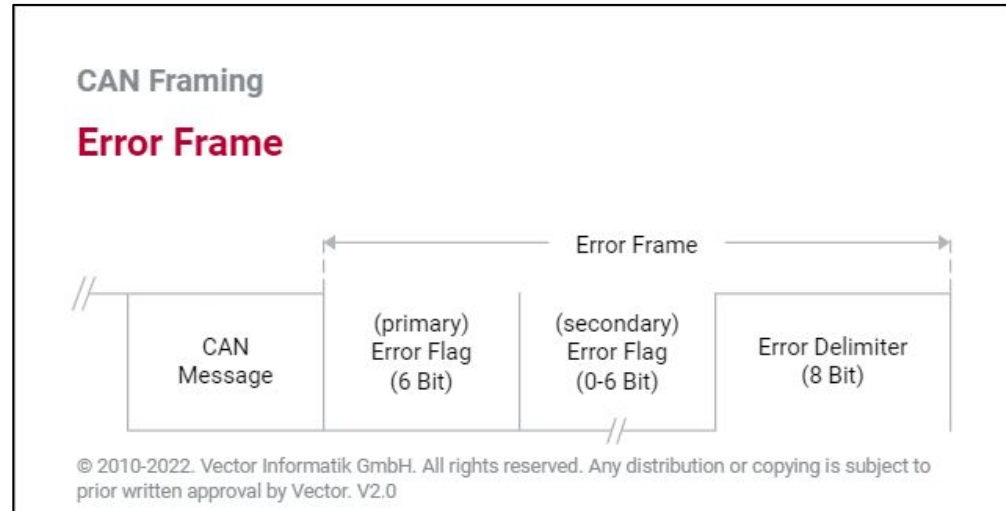
- **Remote frame**

- It is a frame type with which user data, i.e. **data frames, can be requested** from **any other CAN node**.
- **Except for its missing data field**, a remote frame has the same structure as a data frame.



CAN frames

- **Error frame**
 - The error frame is available to **indicate errors detected during communication.**
 - An ongoing **erroneous data transmission** is **terminated** and an error frame is issued.



CAN Logical Error detection and handling

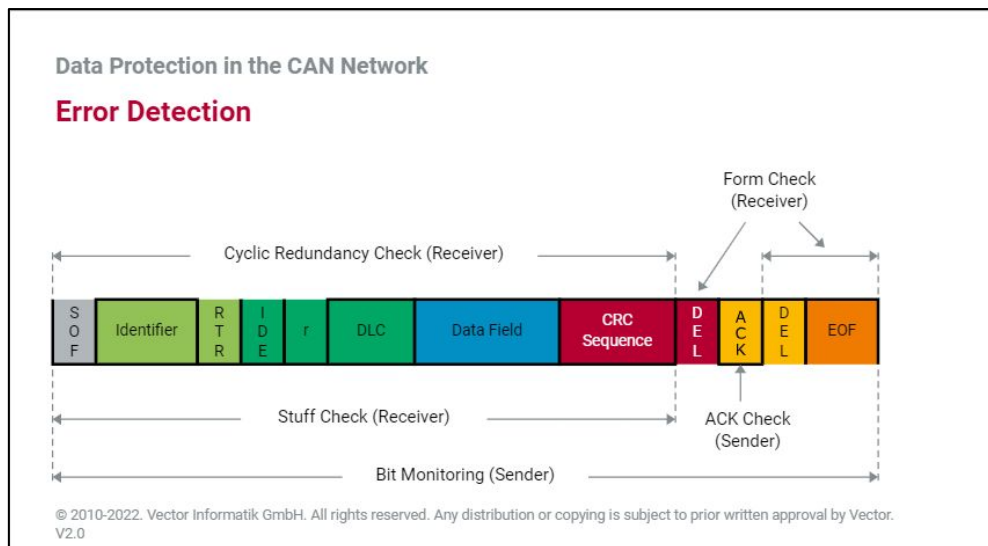
- **Logical Error Detection:**

- **Sender tasks:**

- **Bit Monitoring**
 - **ACK Check**

- **Receiver tasks:**

- **Stuff Check**
 - **Cyclic Redundancy Check**
 - **Form Check**



CAN Logical Error detection and handling

- **Logical Error Detection:**

- **Sender tasks:**

- **Bit Monitoring**

CAN Data Protection

Error Detection

Bit Monitoring	▶
Stuff Check	▶
Form Check	▶
Cyclic Redundancy Check	▶
ACK Check	▶

Bit Monitoring

- ▶ Sender Task
- ▶ Compares every bit placed on the CAN bus with actual bus level
- ▶ Discrepancy indicates a bit monitoring error and results in error handling
- ▶ Exceptions during Arbitration Field and Ack Slot

CAN Logical Error detection and handling

- **Logical Error Detection:**

- **Receiver tasks:**

- **Stuff Check**

CAN Data Protection

Error Detection

Bit Monitoring	▶
Stuff Check	▶
Form Check	▶
Cyclic Redundancy Check	▶
ACK Check	▶

Stuff Check

- ▶ Receiver Task
- ▶ Compares arriving bit stream for a sequence of six homogeneous bits
- ▶ Comparison in the so-called bit stuffing area (from SOF up to and including CRC sequence)
- ▶ Detection of a sixth homogeneous bit indicates a bit stuffing error and results in error handling

?

© 2010-2022. Vector Informatik GmbH. All rights reserved. Any distribution or copying is subject to prior written approval by Vector. V2.0

CAN Logical Error detection and handling

- **Logical Error Detection:**

- **Receiver tasks:**

- **Form Check**

CAN Data Protection

Error Detection

Bit Monitoring	▶
Stuff Check	▶
Form Check	▶
Cyclic Redundancy Check	▶
ACK Check	▶

Form Check

- ▶ Receiver Task
- ▶ Comparison of the arriving bit stream with the message format
- ▶ Detection of a dominant delimiter bit (CRC delimiter, ACK delimiter) or a dominant bit within EOF indicates a format error and results in error handling

?

© 2010-2022. Vector Informatik GmbH. All rights reserved. Any distribution or copying is subject to prior written approval by Vector. V2.0

CAN Logical Error detection and handling

- **Logical Error Detection:**
 - **Receiver tasks:**
 - **Cyclic Redundancy Check**

CAN Data Protection

Error Detection

Bit Monitoring	▶
Stuff Check	▶
Form Check	▶
Cyclic Redundancy Check	▶
ACK Check	▶

Cyclic Redundancy Check

- ▶ Receiver Task
- ▶ Utilizes the arriving bit stream and generator polynomial for the Cyclic Redundancy Check defined in ISO 11898-1
- ▶ Detection of a CRC error results in error handling

?

© 2010-2022. Vector Informatik GmbH. All rights reserved. Any distribution or copying is subject to prior written approval by Vector. V2.0

CAN Logical Error detection and handling

- **Logical Error Detection:**

- **Sender tasks:**

- **ACK Check**

CAN Data Protection

Error Detection

Bit Monitoring	▶
Stuff Check	▶
Form Check	▶
Cyclic Redundancy Check	▶
ACK Check	▶

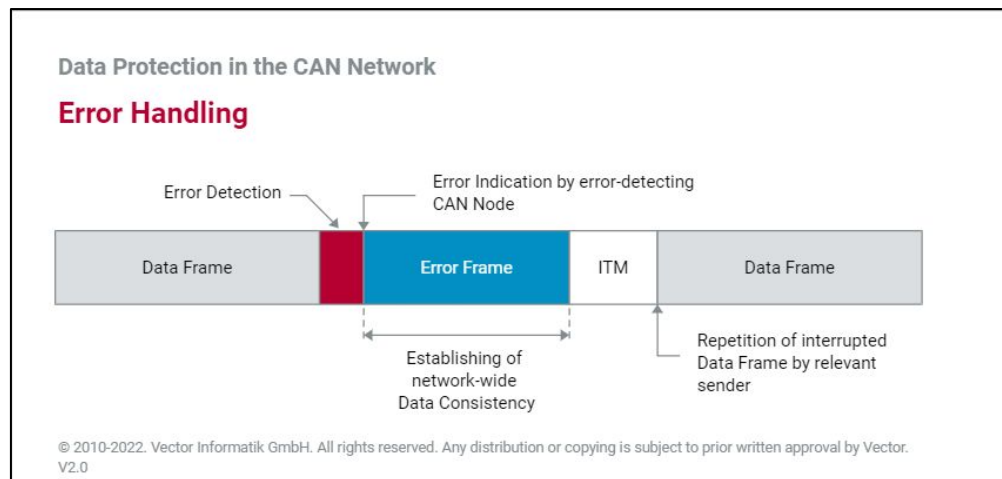
ACK Check

- ▶ Sender Task
- ▶ Comparison of the recessive bit placed on the CAN bus with the bus level in the ACK slot
- ▶ Acknowledge error (ACK error) is detected if the recessive level placed by the sender is not overwritten
- ▶ Detection of an ACK error results in error handling

CAN Logical Error detection and handling

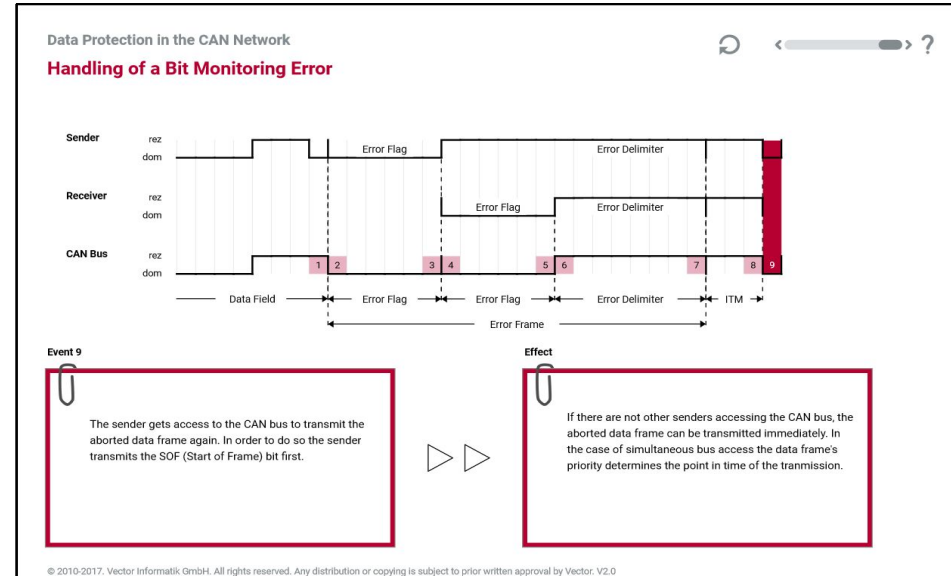
- **Error handling:**

- If a CAN node is experiencing a **local disturbance**, it **must inform all CAN nodes**.
- An error signal (**error flag**) is transmitted which is made up of **six dominant bits**.
- This is an **intentional violation** of the **bit stuffing rule**, and it generates a bit **stuffing error**.
- All other CAN nodes will also transmit an error flag (**secondary error flag**).



CAN Logical Error detection and handling

- **Error handling:**
 - **Handling of a bit Monitoring Error**



CAN Error tracking

- **Error Active**

- After the start, it is the normal state.
- The CAN controller sends **six dominant bits** (active error flag) after detecting an error.
- When (TEC>127; REC>127), the CAN controllers **switch** over to the “**Error Passive**” state.

- **Error Passive**

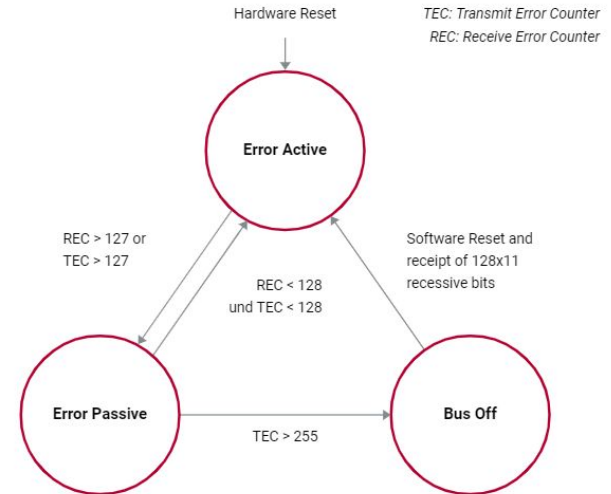
- CAN controllers can only indicate a detected error by sending **six recessive bits**.
- In addition, when **sending two consecutive data or remote frames**, CAN controllers that are in the “Error Passive” state **must wait** the “**Suspend Transmission Time**” (8 bits).

- **Bus Off**

- If a CAN controller **fails** or if there are **extreme accumulations of errors**, a state transition is made to the **Bus Off** state. The CAN controller **disconnects** from the CAN bus.

Data Protection in the CAN Network

Error Tracking



Summary

- You are learned what is CAN bus, its characteristics, its standards, its frame formats, and its error detection, handling and tracking techniques.
- CAN is high reliable, multimaster, low cost, flexible, and depending on message IDs not node addressing.