Training on CAN

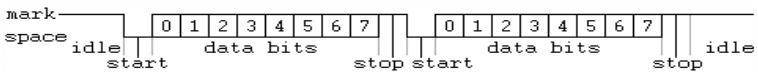
By S. Thaminmun Ansari, Proprietor, Yahasiya Software

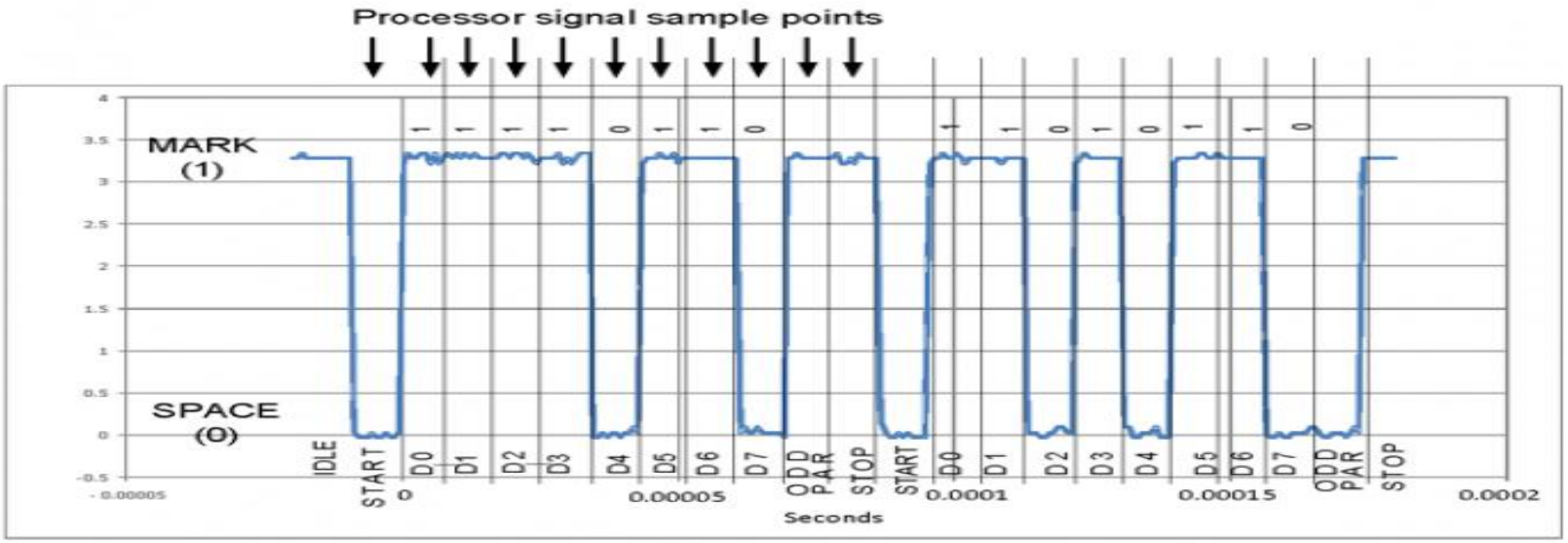
Agenda

- Exploring CAN in details

History of CAN

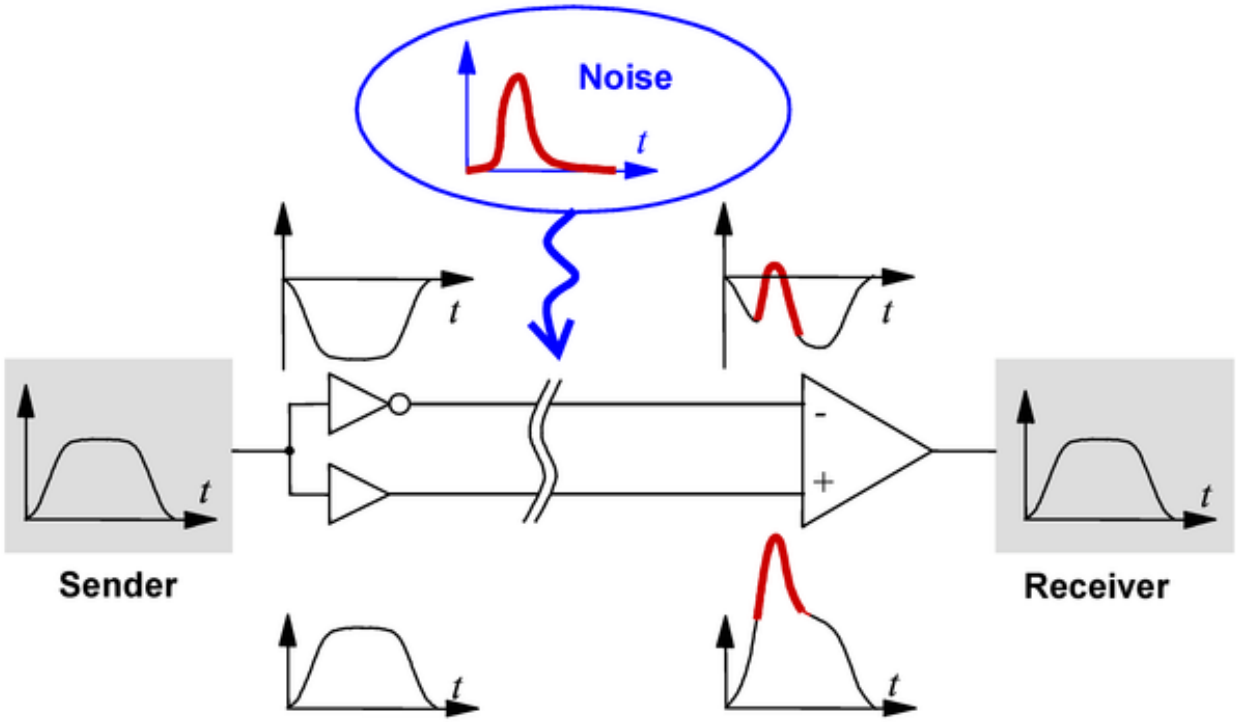
- Controller Area Network
  - ✓ Invented by Robert Bosch GmbH in 1980 for automotive applications
  - ✓ Asynchronous Serial Bus





History of CAN

- Simple 2-wire differential bus



History of CAN

- ✓ Absence of node addressing
  - Message identifier specifies contents and priority
  - Lowest message identifier has highest priority
- ✓ Non-destructive arbitration system by CSMA with collision detection

Multi-master / Broadcasting concept

- ✓ The broadcasting of messages in a CAN Bus network is based on a producer-consumer principle
- ✓ In a multi-master network nodes may transmit data at any time. Each node "listens" to the network bus and will receive every transmitted message.
- ✓ The CAN protocol supports message-filtering, i.e. the receiving nodes will only react to data that is relevant to them.

History of CAN

- ✓ Sophisticated error detection & handling system

History of CAN

- Design goal was to make automobiles more reliable, safer, and more fuel efficient.
- The widely used CAN specification is the version 2.0 made in 1991.
- In 2012 Bosch released CAN FD 1.0 or CAN with Flexible Data-Rate. This specification uses a different frame format that allows a different data length as well as optionally switching to a faster bit rate after the arbitration is decided.
- CAN FD is compatible with existing CAN 2.0 networks so new CAN FD devices can coexist on the same network with existing CAN devices.

What is CAN ?

- The CAN is an ISO standard (ISO 11898) for serial communication
- Today CAN has gained widespread use:
  - ✓ Industrial Automation
  - ✓ Automotive, …etc.
- The CAN standard includes:
  - ✓ Physical layer
  - ✓ Data-link layer
    - Some message types
    - Arbitration rules for bus access
    - Methods for fault detection and fault confinement

Why CAN ?

- Mature Standard
  - ✓ CAN protocol more than 41 years
  - ✓ Numerous CAN products and tools on the market
- Hardware implementation of the protocol
  - ✓ Combination of error handling and fault confinement with high transmission speed (up to 1Mb/s)
- Cost Effective
- Simple Transmission Medium
  - ✓ Twisted pair of wires is the standard, but also just one wire will work
  - ✓ Other links works, too: Opto - or radio links
- Excellent Error Handling
  - ✓ CRC error detection mechanism
- Fault Confinement
  - ✓ Built-in feature to prevent faulty node from the system
- Most used protocol in industrial and automotive world

General Characteristics of CAN(1 of 3)

- Carrier Sense Multiple Access with Collision Detection (CSMA/CD)
  - Every node on the network must monitor the bus (carrier sense) for a period of no activity before trying to send a message on the bus.
  - Once the bus is idle, every node has equal opportunity to transmit a message.
  - If two nodes happen to transmit simultaneously, a nondestructive arbitration method is used to decide which node wins.

General Characteristics of CAN (2 of 3)

- Message-Based Communication
  - Each message contains an identifier.
    - Identifiers allow messages to arbitrate and also allow each node to decide whether to work on the incoming message.
    - The lower the value of the identifier, the higher the priority of the identifier.
  - Each node uses one or more filters to compare the incoming messages to decide whether to take actions on the message.
  - CAN protocol allows a node to request data transmission from other nodes.
  - There is no need to reconfigure the system when a new node joins the system.

Sleep Mode / Wake-up

- To reduce the system's power consumption, a CAN-device may be set into sleep mode without any internal activity and with disconnected bus drivers.
- The sleep mode is finished with a wake-up by any bus activity or by internal conditions of the system

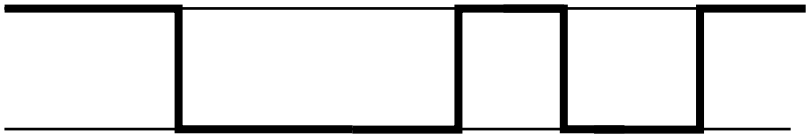
General Characteristics of CAN (3 of 3)

- Error Detection and Fault Confinement
  - The CAN protocol requires each node to monitor the CAN bus to find out if the bus value and the transmitted bit value are identical.
  - The CRC checksum is used to perform error checking for each message.
  - The CAN protocol requires the physical layer to use bit stuffing to avoid long sequence of identical bit value.
  - Defective nodes are switched off from the CAN bus.

CAN Bus Logic

‘1’ Recessive (r)

‘0’ Dominant (D)



Two logic states on the CAN bus

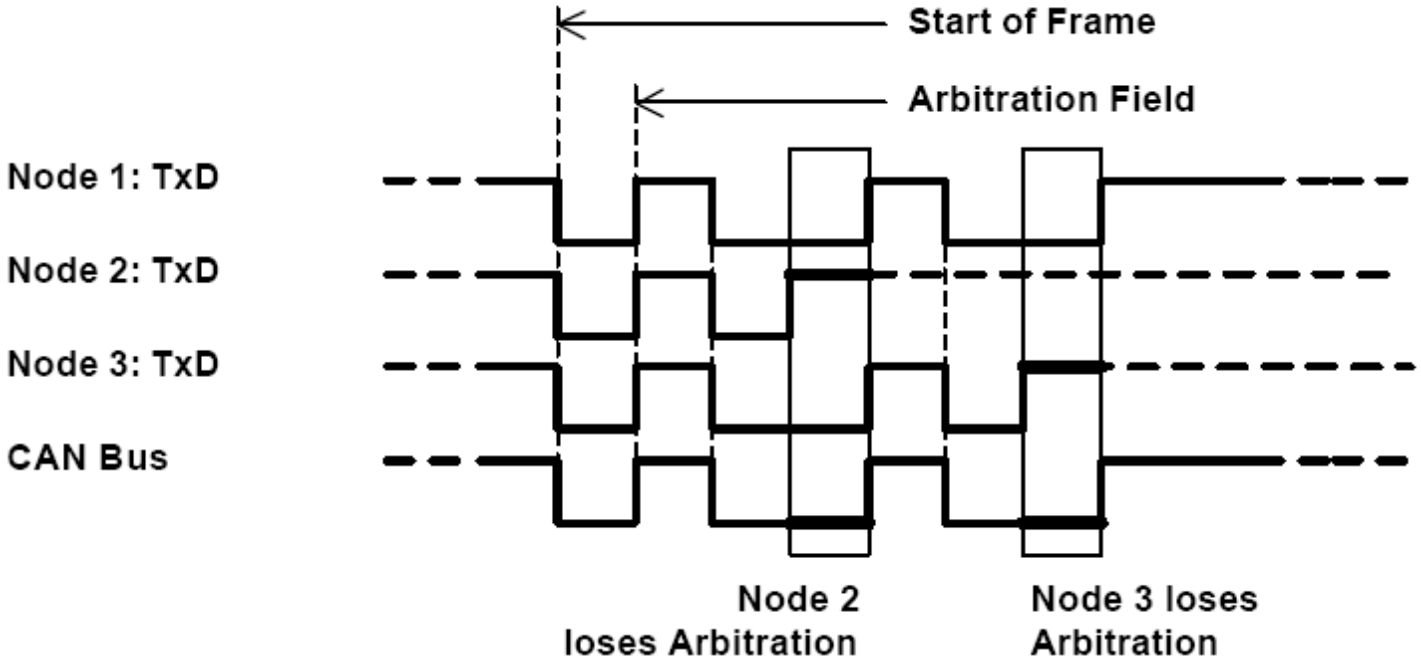
Node 1	Node 2	Node 3	Bus
D	D	D	D
D	D	r	D
D	r	D	D
D	r	r	D
r	D	D	D
r	D	r	D
r	r	D	D
r	r	r	r

Wired-AND Function

Bus in dominant state

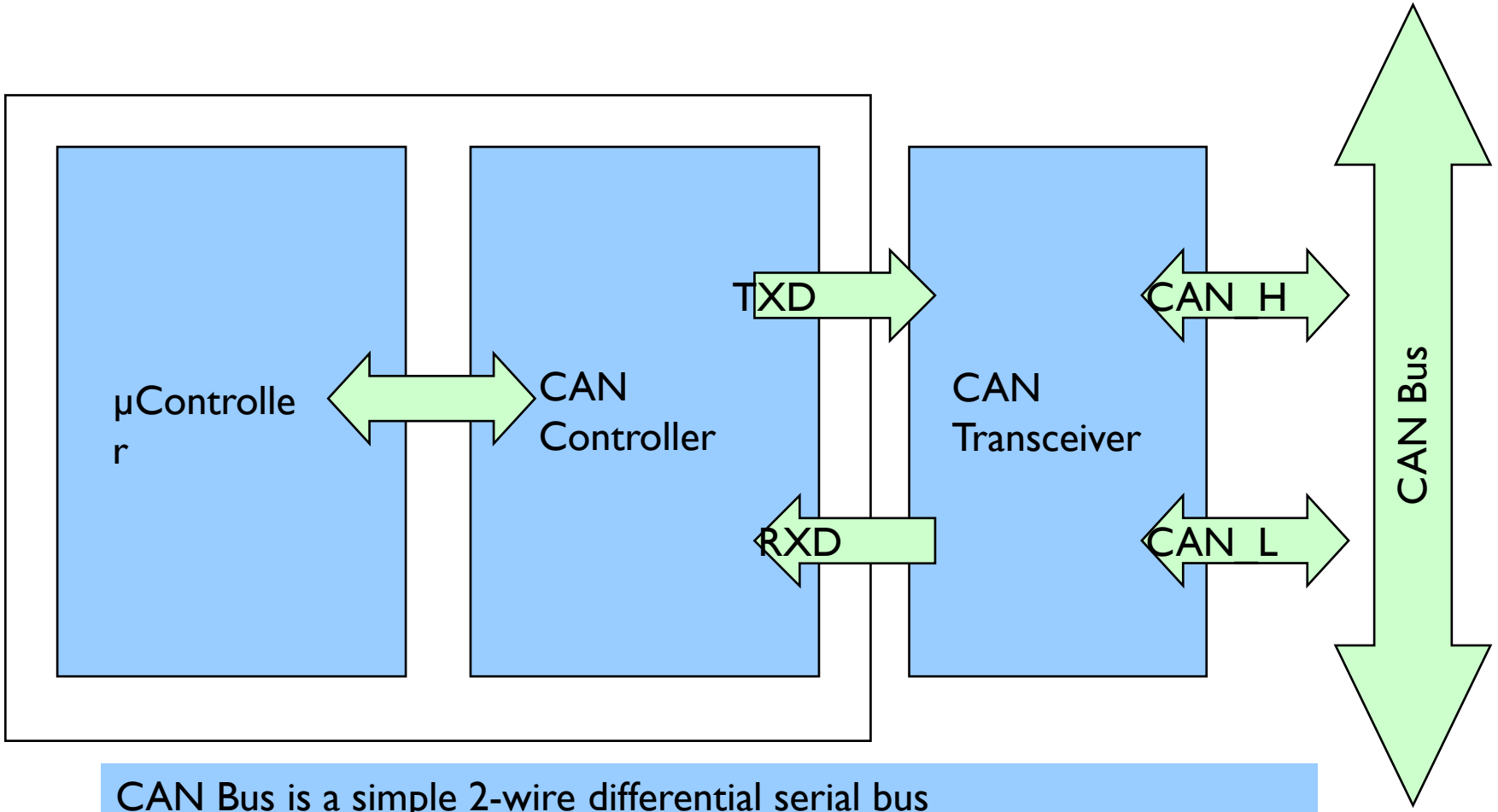
Bus in recessive state or idle

CAN Bus Access and Arbitration: CSMA/CD and AMP



- CSMA/CD: Carrier Sense Multiple Access / Collision Detection
- AMP: Arbitration by Message Priority

Typical CAN Node

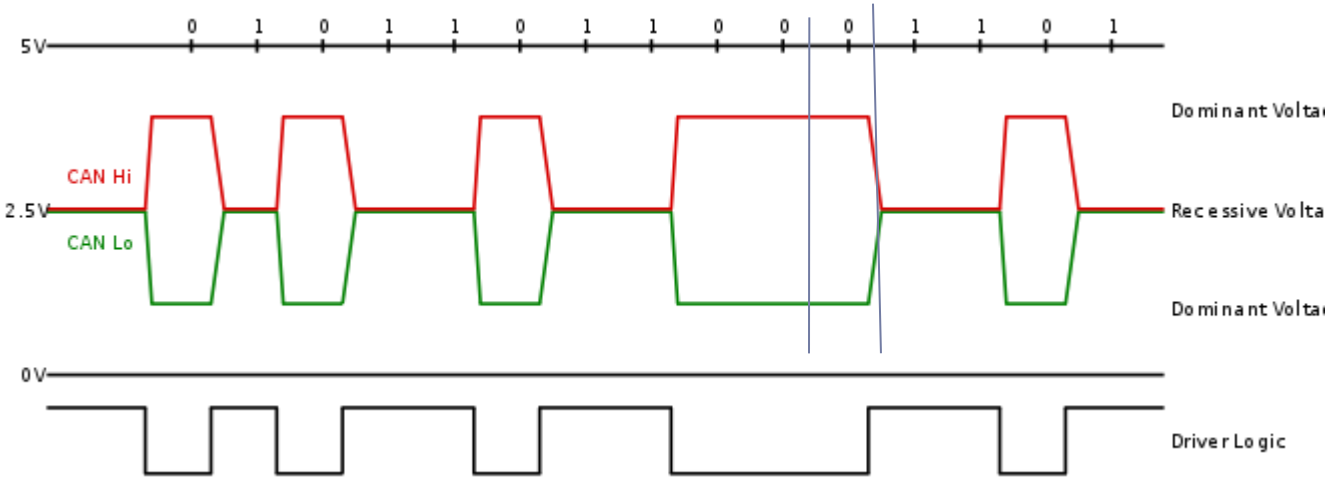


CAN Bus is a simple 2-wire differential serial bus

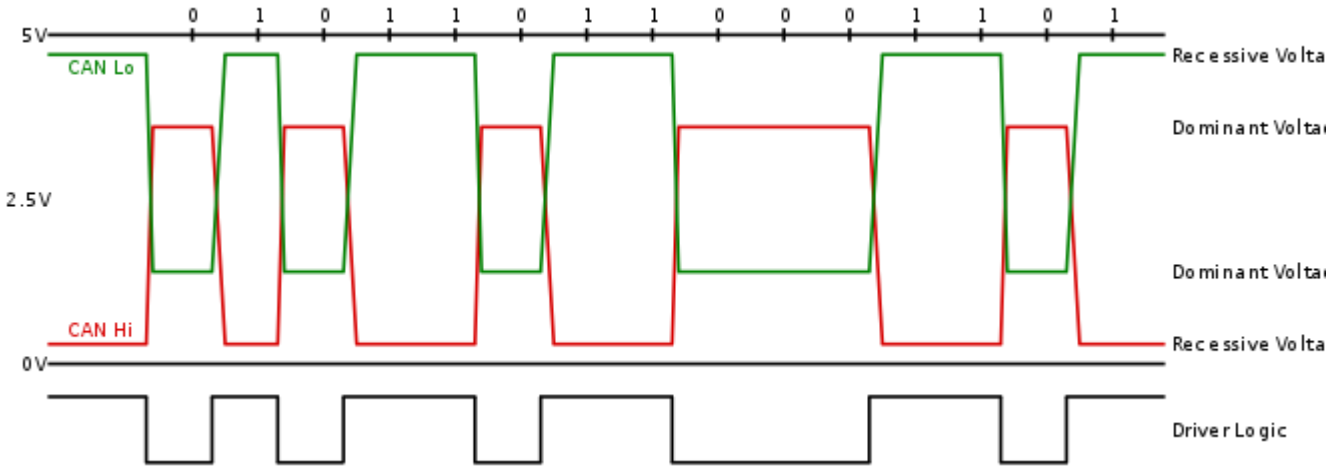
CAN Bus is terminated on each side by a 120 Ohm resistor

CAN bus State

- Two states of CAN bus
  - Recessive: high or logic 1
  - Dominant: low or logic 0



CAN bus State



Types of CAN frame

- Data frame
- Remote frame
- Error frame
- Overload frame

Data Frame

- A data frame consists of seven fields: start-of-frame, arbitration, control, data, CRC, ACK, and end-of-frame.

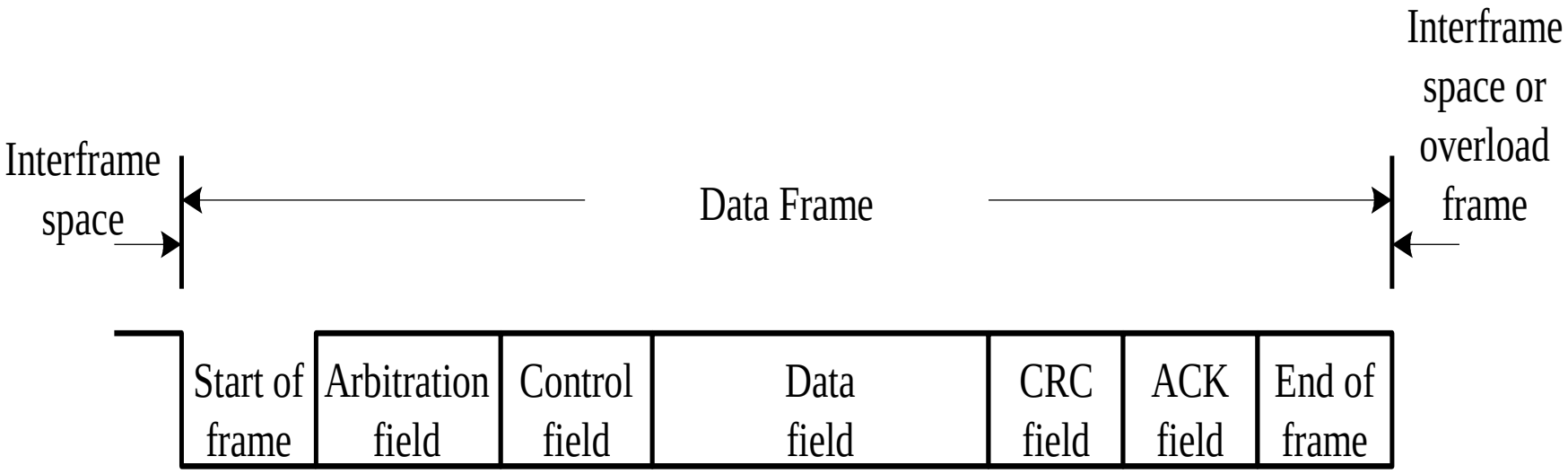


Figure 13.2 CAN Data frame

Start of Frame

- A single dominant bit to mark the beginning of a data frame.
- All nodes have to
- synchronize to the
- leading edge caused
- by this field.

Arbitration Field

- There are two formats for this field: standard format and extended format.

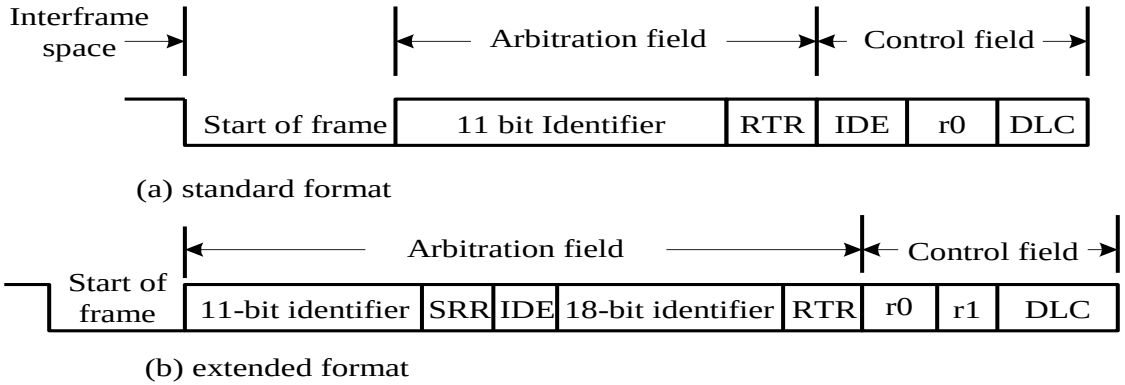


Figure 13.3 Arbitration field

- The identifier of the standard format corresponds to the base ID in the extended format.
- The RTR bit is the remote transmission request and must be 0 in a data frame.
- The SRR bit is the substitute remote request and is recessive.
- The IDE field indicates whether the identifier is extended and should be recessive in the extended format.
- The extended format also contains the 18-bit extended identifier.

Control Field

- Contents are shown in figure 13.4.
- The first bit is IDE bit for the standard format but is used as reserved bit r1 in extended format.
- r0 is reserved bit.
- DLC3…DLC0 stands for data length and can be from 0000 (0) to 1000 (8).

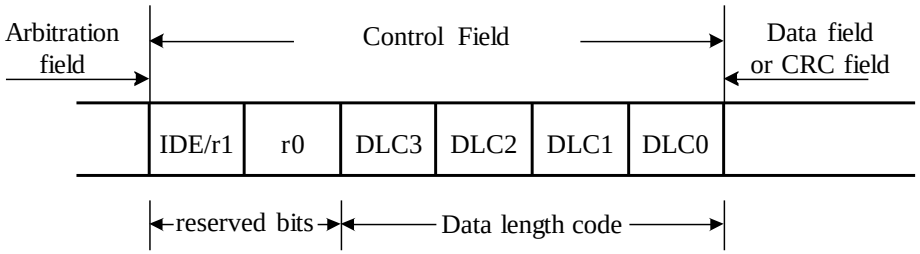


Figure 13.4 Control field

Data Field

- May contain 0 to 8 bytes of data

CRC Field

- It contains the 16-bit CRC sequence and a CRC delimiter.
  - CRC_RG = 0; // initialize shift register
  - REPEAT
  - CRCNXT = NXTBIT EXOR CRC_RG(14);
  - CRC_RG(14:1) = CRC_RG(13:0); // shift left by
  - CRC_RG(0) = 0; // 1 position
  - IF CRCNXT THEN
  - CRC_RG(14:0) = CRC_RG(14:0) EXOR (4599hex);
  - ENDIF
  - UNTIL (CRC SEQUENCE starts or there is an ERROR condition)
- The CRC delimiter is a single **recessive** bit.

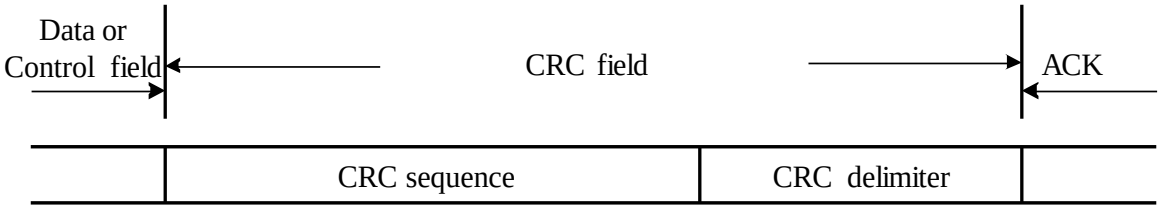


Figure 13.5 CRC field

ACK Field

- Consists of two bits
- The first bit is the **acknowledgement bit**.
  - This bit is set to recessive by the transmitter, but will be reset to dominant if a receiver acknowledges the data frame.
- The second bit is the **ACK delimiter** and is recessive.

Remote Frame

- Used by a node to request other nodes to send certain type of messages
- Has six fields as shown in Figure 13.7
  - These fields are identical to those of a data frame with the exception that the RTR bit in the arbitration field is **recessive** in the remote frame.
- If a DATA FRAME and a REMOTE FRAME with the same IDENTIFIER are initiated at the same time, the DATA FRAME prevails over the REMOTE FRAME

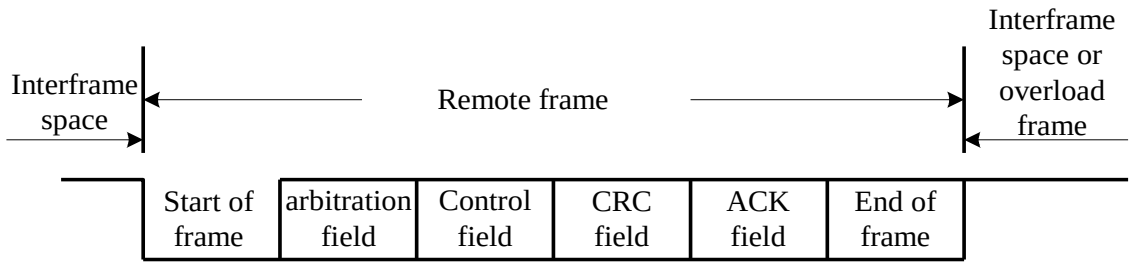


Figure 13.7 Remote frame

Error Frame

- This frame consists of two fields.
  - The first field is given by the superposition of error flags contributed from different nodes.
  - The second field is the error delimiter.
- Error flag can be either active-error flag or passive-error flag.
  - Active error flag consists of six consecutive dominant bits.
  - Passive error flag consists of six consecutive recessive bits.
- The error delimiter consists of eight recessive bits.

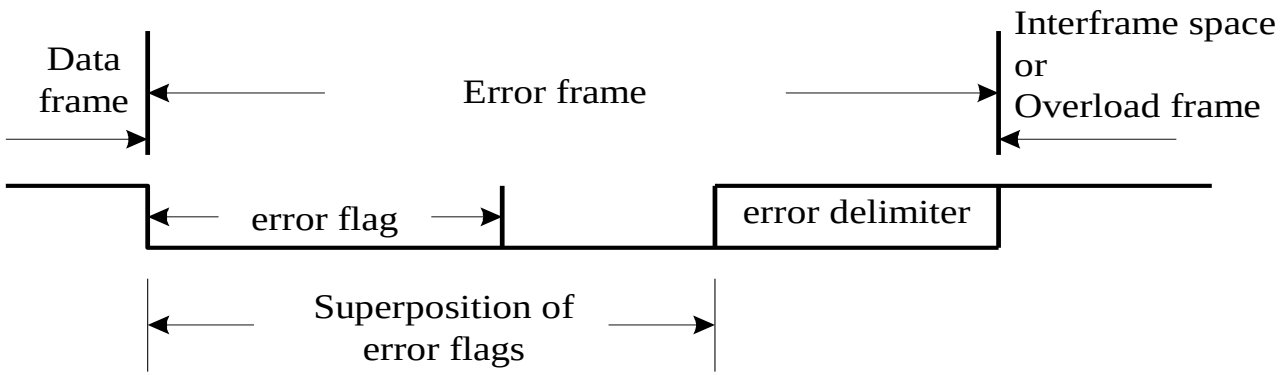


Figure 13.8 Error frame

Overload Frame

- Consists of two bit fields: overload flag and overload delimiter
- Three different overload conditions lead to the transmission of the overload frame:
  - Internal conditions of a receiver require a delay of the next data frame or remote frame.
  - At least one node detects a dominant bit during intermission.
  - A CAN node samples a dominant bit at the eighth bit (i.e., the last bit) of an error delimiter or overload delimiter.
- Format of the overload frame is shown in Figure 13.9.
- The overload flag consists of six dominant bits.
- The overload delimiter consists of eight recessive bits.

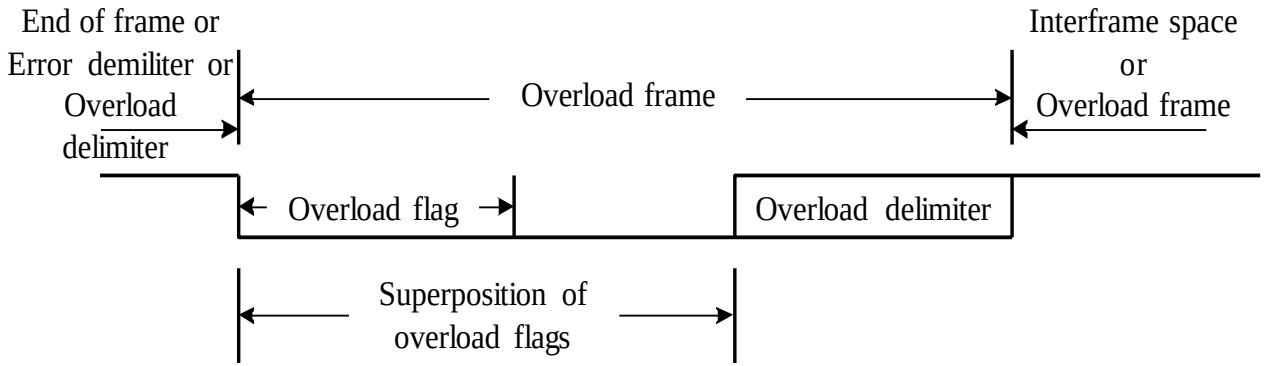


Figure 13.9 Overload frame

Inter frame Space

- Data frames and remote frames are separated from preceding frames by the inter frame space.
- Overload frames and error frames are not preceded by an inter frame space.
- The formats for inter frame space is shown in Figure 13.10 and 13.11.

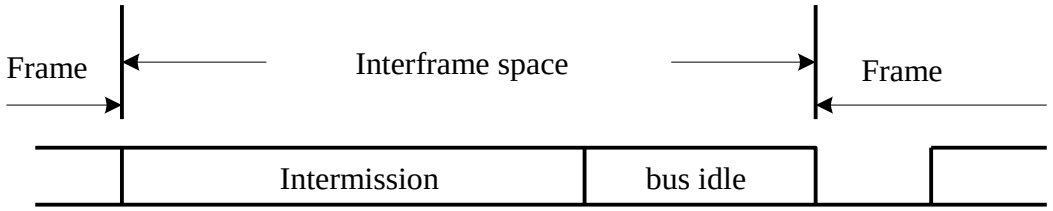


Figure 13.10 Interframe space for non error-passive nodes or receiver of previous message

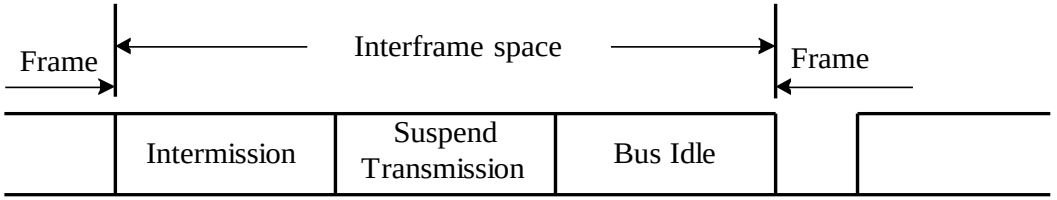


Figure 13.11 Interframe space for error-passive nodes

Inter frame Space

- The intermission subfield consists of three recessive bits.
- During intermission no node is allowed to start transmission of the data frame or remote frame.
- The period of bus idle may be of arbitrary length.
- After an error-passive node has transmitted a frame, it sends eight recessive bits following intermission, before starting to transmit a new message or recognizing the bus as idle.

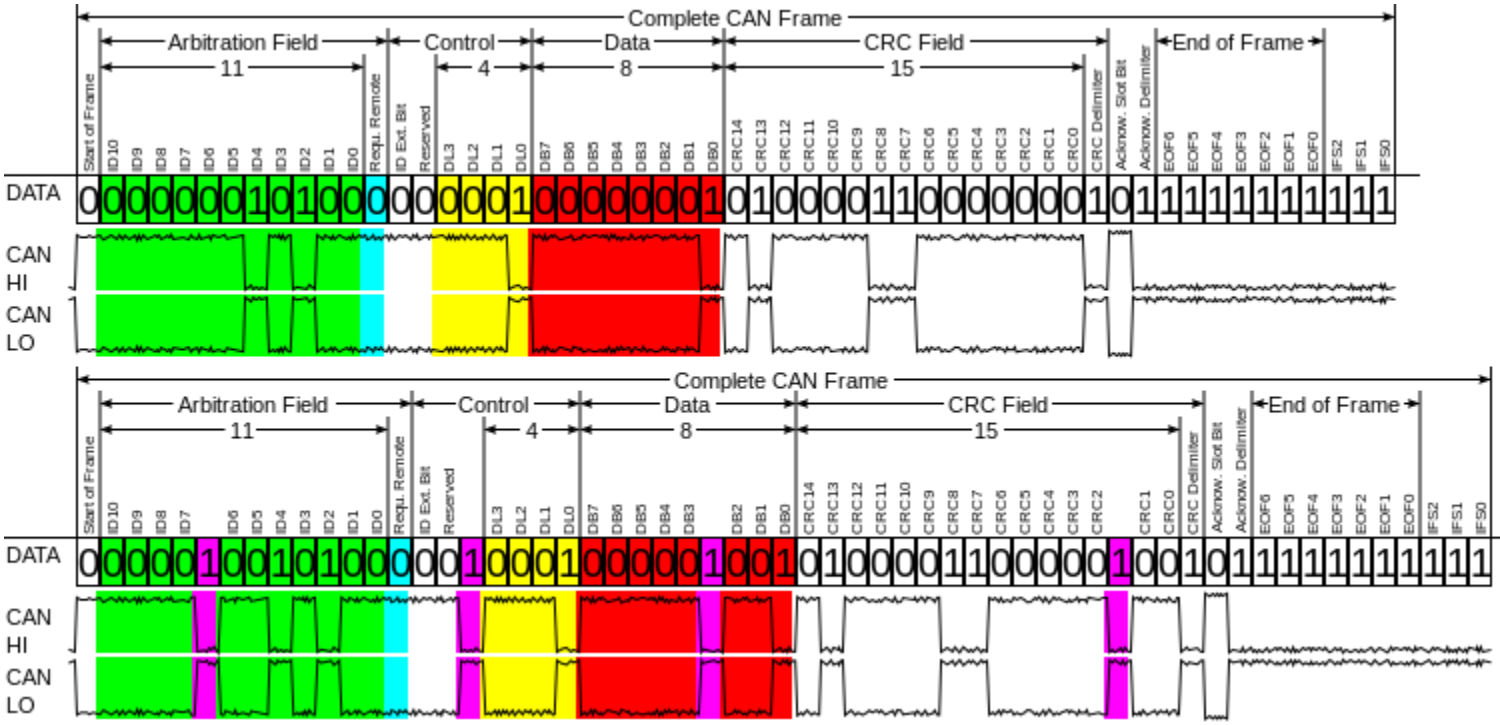
Message Filtering

- A node uses filter (s) to decide whether to work on a specific message.
- Message filtering is applied to the whole identifier.
- A node can optionally implement mask registers that specify which bits in the identifier are examined with the filter.
- If mask registers are implemented, every bit of the mask registers must be programmable.

Bit Stream Encoding

- The frame segments including start-of-frame, arbitration field, control field, data field, and CRC sequence are encoded by bit stuffing.
- Whenever a transmitter detects five consecutive bits of identical value in the bit stream to be transmitted, it inserts a complementary bit in the actual transmitted bit stream.
- The remaining bit fields of the data frame or remote frame (CRC delimiter, ACK field and end of frame) are of fixed form and not stuffed.
- The error frame and overload frame are also of fixed form and are not encoded by the method of bit stuffing.
- The bit stream in a message is encoded using the non-return-to-zero (NRZ) method.
- In the non-return-to-zero encoding method, a bit is either recessive or dominant.

Bit stuffing Example



Errors (1 of 3)

- Error handling
  - CAN recognizes five types of errors.
- Bit error
  - A node that is sending a bit on the bus also monitors the bus.
  - When the bit value monitored is different from the bit value being sent, the node interprets the situation as an error.
  - There are two exceptions to this rule:
    - A node that sends a recessive bit during the stuffed bit-stream of the arbitration field or during the ACK slot detects a dominant bit.
    - A transmitter that sends a passive-error flag detects a dominant bit.

Errors (2 of 3)

- Stuff error
  - Six consecutive dominant or six consecutive recessive levels occurs in a message field.
- CRC error
  - CRC sequence in the transmitted message consists of the result of the CRC calculation by the transmitter.
  - The receiver recalculates the CRC sequence using the same method but resulted in a different value. This is detected as a CRC error.

Errors (3 of 3)

- Form error
  - Detected when a fixed-form bit field contains one or more illegal bits
- Acknowledgement error
  - Detected whenever the transmitter does not monitor a dominant bit in the ACK slot
- Error Signaling
  - A node that detects an error condition and signals the error by transmitting an error flag
    - An error-active node will transmit an active-error flag.
    - An error-passive node will transmit a passive-error flag.

Fault Confinement

- A node may be in one of the three states: error-active, error-passive, and bus-off.
- A CAN node uses an error counter to control the transition among these three states.
- CAN protocol uses 12 rules to control the increment and decrement of the error counter.
- When the error count is less than 128, a node is in error-active state.
- When the error count equals or exceeds 128 but not higher 255, the node is in error-passive state.
- When the error count equals or exceeds 256, the node is in bus off state.
- An error-active node will transmit an active-error frame when detecting an error.
- An error-passive node will transmit a passive-error frame when detecting an error.
- A bus-off node is not allowed to take part in bus communication.

Twelve Rules to update Error Count

1. When a RECEIVER detects an error, the RECEIVE ERROR COUNT will be increased by 1, except when the detected error was a BIT ERROR during the sending of an ACTIVE ERROR FLAG or an OVERLOAD FLAG.
2. When a RECEIVER detects a 'dominant' bit as the first bit after sending an ERROR FLAG the RECEIVE ERROR COUNT will be increased by 8
3. When a TRANSMITTER sends an ERROR FLAG the TRANSMIT ERROR COUNT is increased by 8.

Twelve Rules to update Error Count

4. If an TRANSMITTER detects a BIT ERROR while sending an ACTIVE ERROR FLAG or an OVERLOAD FLAG the TRANSMIT ERROR COUNT is increased by 8.
5. If an RECEIVER detects a BIT ERROR while sending an ACTIVE ERROR FLAG or an OVERLOAD FLAG the RECEIVE ERROR COUNT is increased by 8.

Twelve Rules to update Error Count

6. Any node tolerates up to 7 consecutive 'dominant' bits after sending an ACTIVE ERROR FLAG, PASSIVE ERROR FLAG or OVERLOAD FLAG.

After detecting the 14th consecutive 'dominant' bit (in case of an ACTIVE ERROR FLAG or an OVERLOAD FLAG) or after detecting the 8th consecutive 'dominant' bit following a PASSIVE ERROR FLAG, and after each sequence of additional eight consecutive 'dominant' bits every TRANSMITTER increases its TRANSMIT ERROR COUNT by 8 and every RECEIVER increases its RECEIVE ERROR COUNT by 8.

Twelve Rules to update Error Count

7. After the successful transmission of a message (getting ACK and no error until END OF FRAME is finished) the TRANSMIT ERROR COUNT is decreased by 1 unless it was already 0.
8. After the successful reception of a message (reception without error up to the ACK SLOT and the successful sending of the ACK bit), the RECEIVE ERROR COUNT is decreased by 1, if it was between 1 and 127. If the RECEIVE ERROR COUNT was 0, it stays 0, and if it was greater than 127, then it will be set to a value between 119 and 127.

Twelve Rules to update Error Count

9. A node is 'error passive' when the TRANSMIT ERROR COUNT equals or exceeds 128, or when the RECEIVE ERROR COUNT equals or exceeds 128. An error condition letting a node become 'error passive' causes the node to send an PASSIVE ERROR FLAG.
10. . A node is 'bus off' when the TRANSMIT ERROR COUNT is greater than or equal to 256
11. An 'error passive' node becomes 'error active' again when both the TRANSMIT ERROR COUNT and the RECEIVE ERROR COUNT are less than or equal to 127.

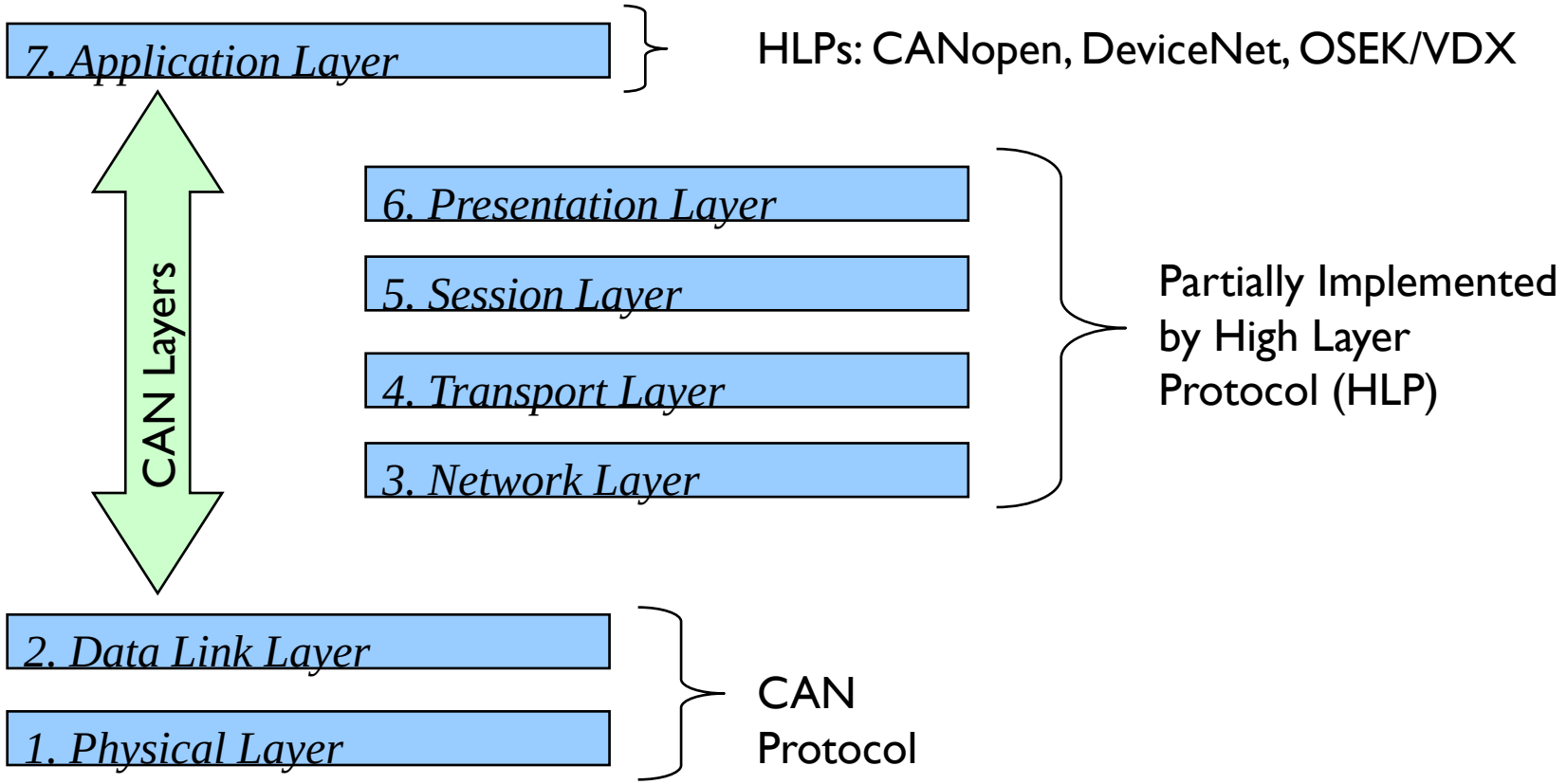
Twelve Rules to update Error Count

12. An node which is 'bus off' is permitted to become 'error active' (no longer 'bus off') with its error counters both set to 0 after 128 occurrence of 11 consecutive 'recessive' bits have been monitored on the bus.

CAN Message Bit Timing

- The setting of a bit time in a CAN system must allow a bit sent out by the transmitter to reach the far end of the CAN bus and allow the receiver to send back acknowledgement and reach the transmitter.
- The number of bits transmitted per second is defined as the nominal bit rate.

ISO-OSI Reference Model



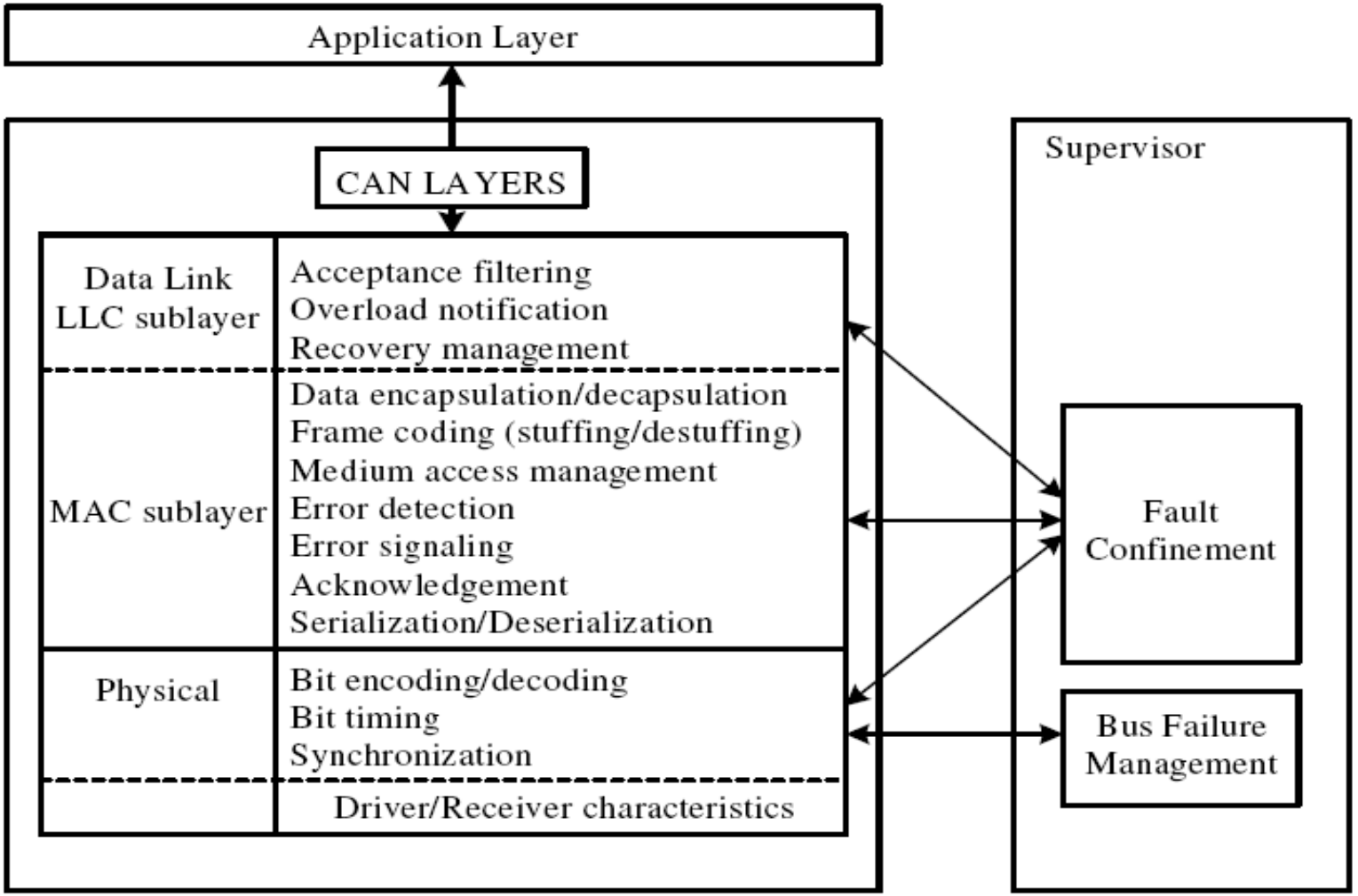
Layered Approach in CAN (1 of 3)

- Only the logical link and physical layers are described.
- Data link layer is divided into two sublayers: logical link control (LLC) and medium access control (MAC).
  - LLC sublayer deals with message acceptance filtering, overload notification, and error recovery management.
  - MAC sublayer presents incoming messages to the LLC sublayer and accepts messages to be transmitted forward by the LLC sublayer.
  - MAC sublayer is responsible for message framing, arbitration, acknowledgement, error detection, and signaling.
  - MAC sublayer is supervised by the fault confinement mechanism.

Layered Approach in CAN

- The physical layer defines how signals are actually transmitted, dealing with the description of bit timing, bit encoding, and synchronization.
- CAN bus driver/receiver characteristics and the wiring and connectors are not specified in the CAN protocol.
- System designer can choose from several different media to transmit the CAN signals.

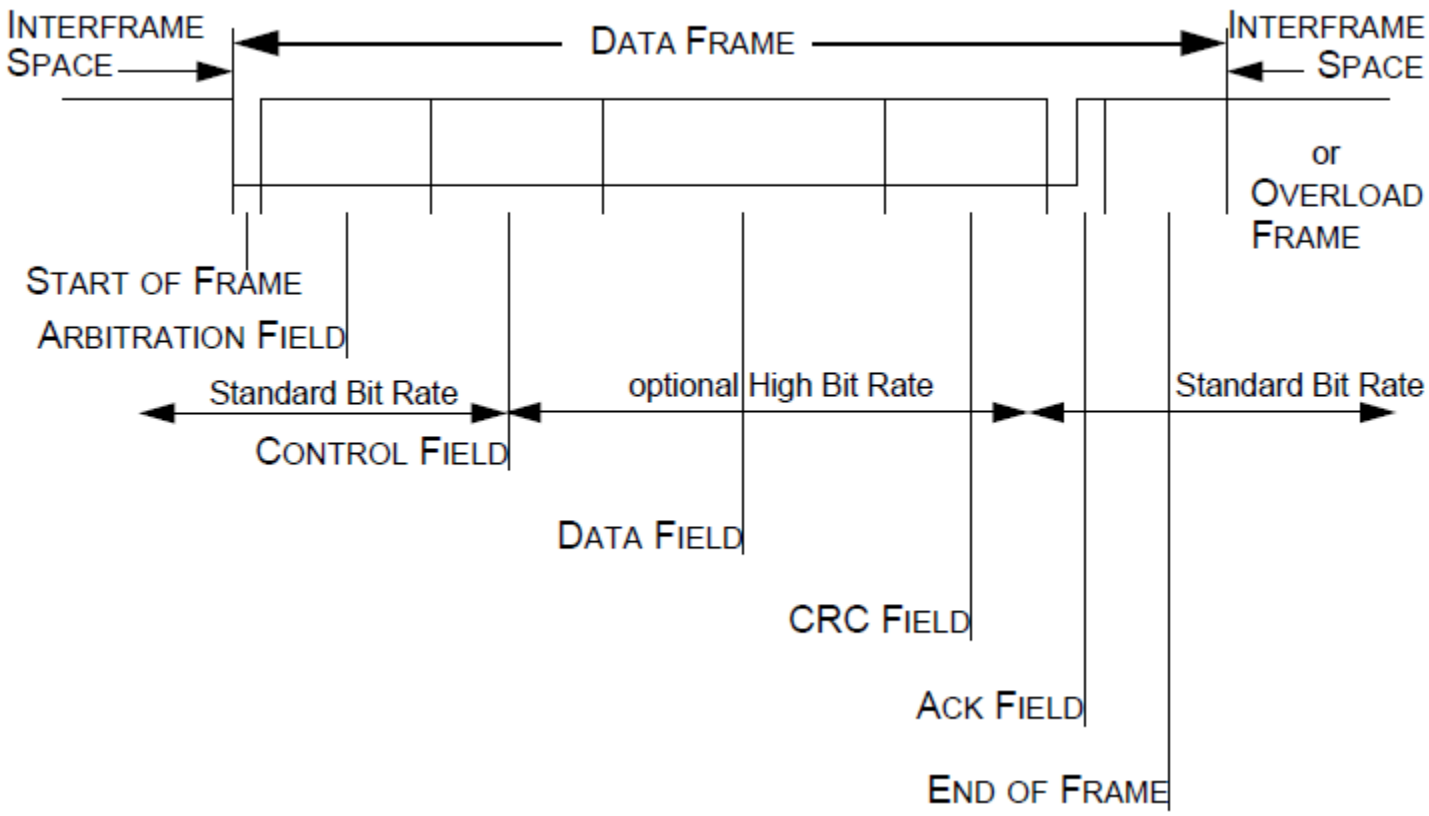
Layered Approach in CAN



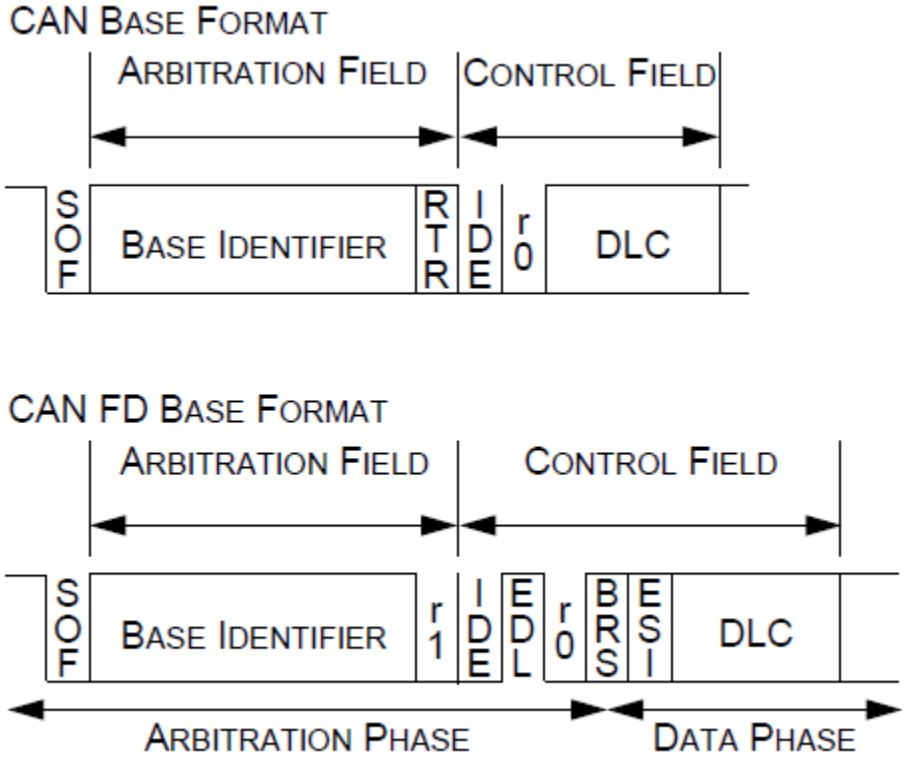
CAN FD

- CAN FD is an extension to the original CAN bus protocol that was specified in ISO 11898-1
- Developed in 2011 and released in 2012 by Bosch, CAN FD was developed to meet the need to increase the data transfer rate up to 5 times faster and with larger frame /message sizes for use in modern automotive Electronic Control Units (ECU)s.

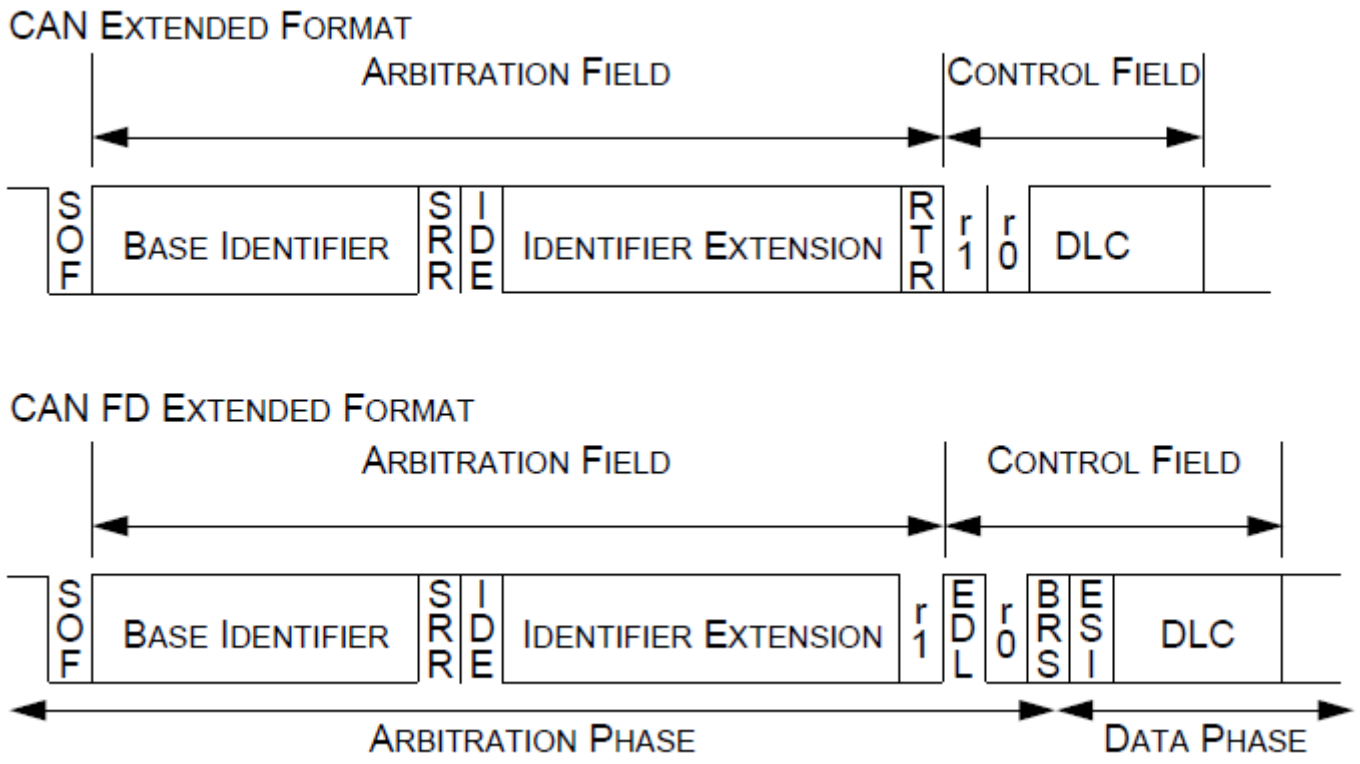
CAN FD Transmission Details



CAN STD Vs FD Data Frame Format



CAN Ext Vs CAN FD Ext



EDL (Extended Data Length)

- The EXTENDED DATA LENGTH (EDL) bit is *recessive. It only exists in CAN FD format* frames,
- it distinguishes between CAN format and CAN FD format frames.
- In a CAN format frame, the *dominant bit r0 is transmitted instead of EDL.*
- *In frames with 11-bit identifiers,* EDL comes after the IDE bit,
- in frames with 29-bit-identifier, it comes after the r1 bit. EDL is always followed by the *dominant bit r0, which is reserved for future expansion* of the protocol

BRS (Bit Rate Switch)

- The BIT RATE SWITCH (BRS) bit decides whether the bit rate is switched inside a CAN FD format frame.
- If the bit is transmitted *recessive, the bit rate is switched from the* standard bit rate of the ARBITRATION PHASE to the preconfigured alternate bit rate of the DATA PHASE.
- If it is transmitted *dominant, the bit rate is not switched. BRS does not exist* in CAN format frames.

ESI (Error State Indicator)

- The ERROR STATE INDICATOR (ESI) flag is transmitted *dominant by error active nodes,recessive by error passive nodes.*
- *ESI does not exist in STD CAN frames.*

DLC -DATA LENGTH CODE CAN FD

	Number of Data Bytes	Data Length Code			
		DLC3	DLC2	DLC1	DLC0
Codes in CAN and CAN FD Format	0	0	0	0	0
	1	0	0	0	1
	2	0	0	1	0
	3	0	0	1	1
	4	0	1	0	0
	5	0	1	0	1
	6	0	1	1	0
	7	0	1	1	1
CAN Format	8	1	0/1	0/1	0/1
Codes in CAN FD Format	8	1	0	0	0
	12	1	0	0	1
	16	1	0	1	0
	20	1	0	1	1
	24	1	1	0	0
	32	1	1	0	1
	48	1	1	1	0
	64	1	1	1	1

Q & A

Thank You