Recommended readings

- Dominique Paret, *Multiplexed Networks for Embedded Systems: CAN, LIN, FlexRay, Safe-by-Wire...*, ISBN: 978-0-470-03416-3, 434 pages, WILEY, UK, 2007.

- Wolfhard Lawrenz, CAN System Engineering: From, Theory to Practical Applications, ISBN-10: 0387949399, ISBN-13: 978-0387949390, 468 pages, Springer, 1997

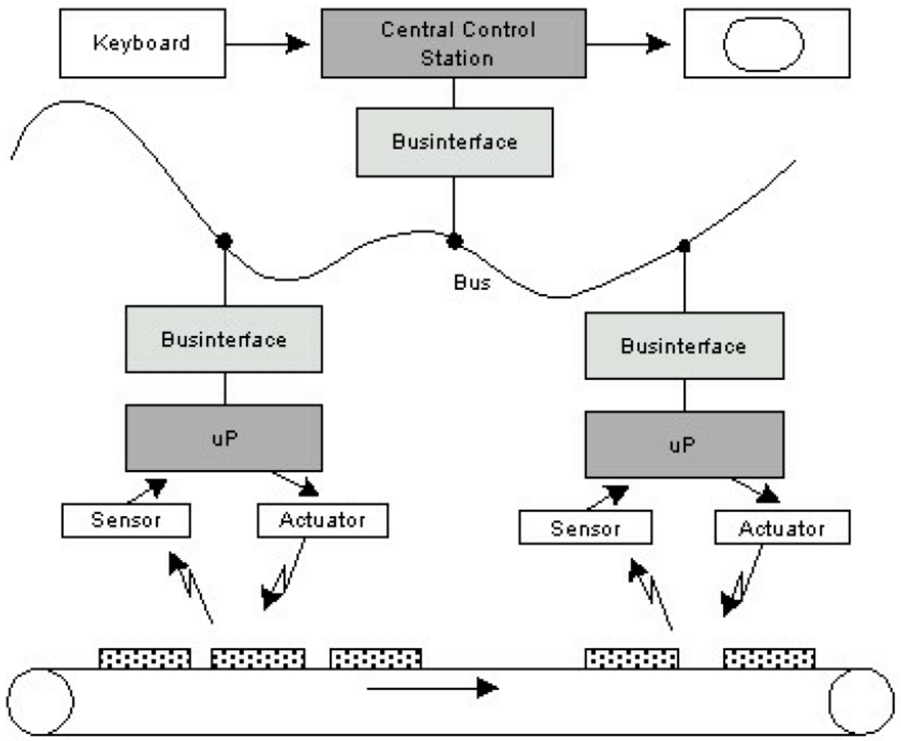
- Konrad Etschberger, Controller Area Network, IXXAT Automation GmbH (August 22, 2001), ISBN-10:3000073760, ISBN-13: 978-3000073762, 430 pages, 2001.

Lecture 3.

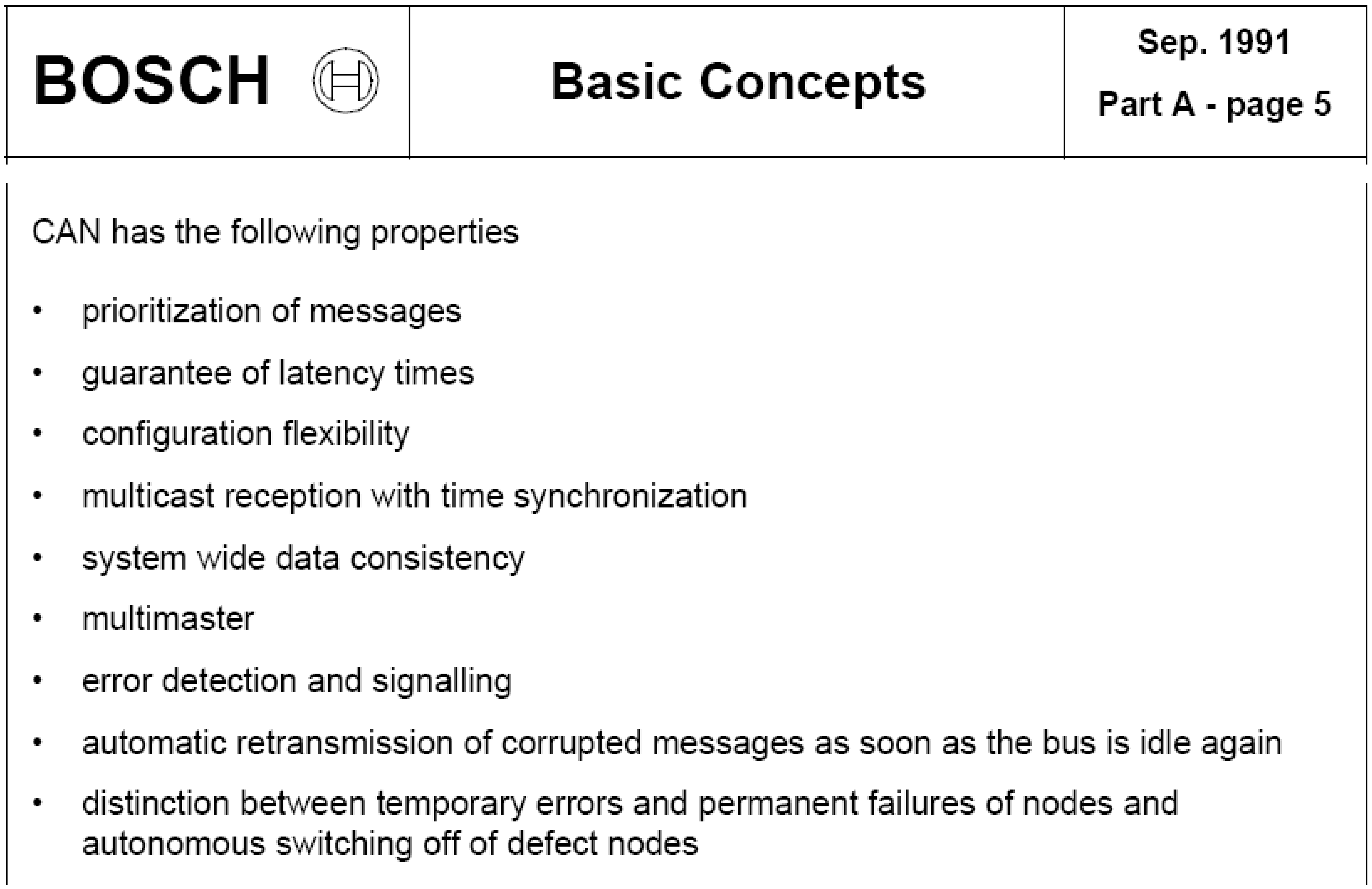
Data Link Layer in CAN: Recessive and Dominant bits, Arbitration, Frame Structure: standard and extended frames (CAN 2.0A and CAN 2.0 B), Bit Stuffing, Error Detection, Error Management (Error passive, error active and bus-off states)

General Setting: Distributed Control System (DCS)

- Desired features:
  - Communication is message oriented and not target oriented (the same message is likely to be of interest for multiple endpoints)
  - Supports broadcast, multicast: a controller can use the sensors from other controller as its own
  - Real-time: rapidly changing information has to be transmitted faster than other (e.g. engine load changes rapidly than temperature)
  - Entire bus capacity is used to sent the most relevant information at some point (DoS due to traffic overload is not possible)
  - Does not require physical destination addresses
  - Easy to add nodes without hardware or software modifications to existing nodes



With respect to these, the CAN Standard from Bosch specifies



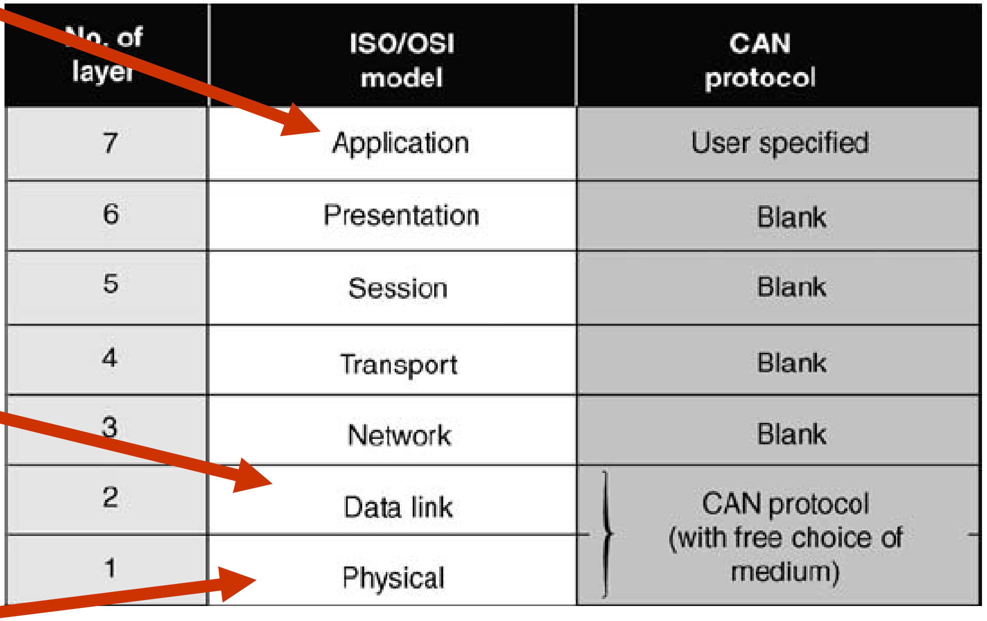
BOSCH	Basic Concepts	Sep. 1991 Part A - page 5

CAN has the following properties

- prioritization of messages
- guarantee of latency times
- configuration flexibility
- multicast reception with time synchronization
- system wide data consistency
- multimaster
- error detection and signalling
- automatic retransmission of corrupted messages as soon as the bus is idle again
- distinction between temporary errors and permanent failures of nodes and autonomous switching off of defect nodes

Comparison of CAN architecture with the OSI/ISO

- Application (can implement in software absent layers)



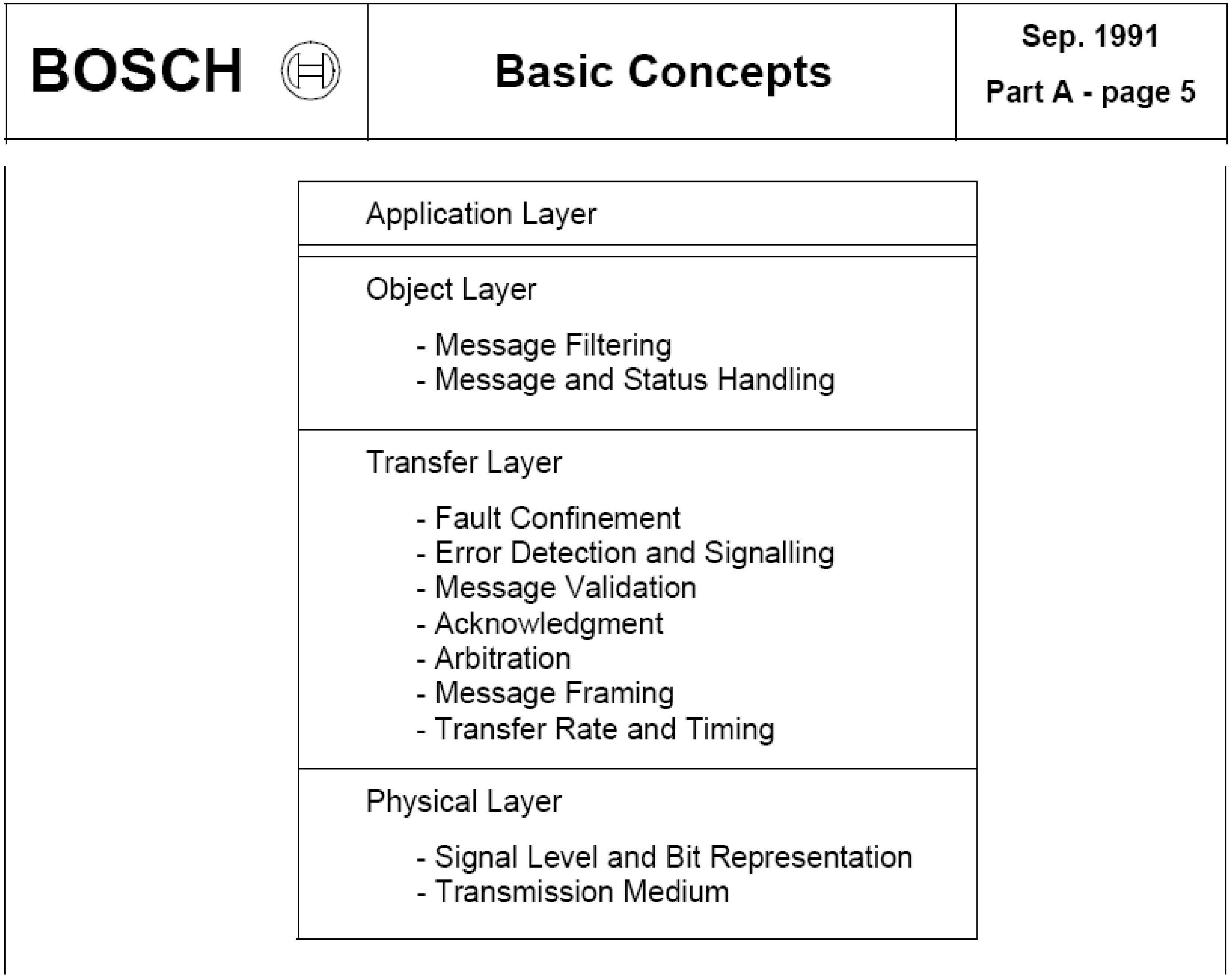
No. of layer	ISO/OSI model	CAN protocol
7	Application	User specified
6	Presentation	Blank
5	Session	Blank
4	Transport	Blank
3	Network	Blank
2	Data link	CAN protocol (with free choice of medium)
1	Physical	

- Frame setup, error control, flow control, multiplexing

(picture c.f. Parret, 2007)

- Electrical and mechanical specifications

CAN layers, more precise…



BOSCH	Basic Concepts	Sep. 1991 Part A - page 5

Application Layer

Object Layer

- Message Filtering
- Message and Status Handling

Transfer Layer

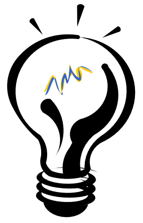
- Fault Confinement
- Error Detection and Signalling
- Message Validation
- Acknowledgment
- Arbitration
- Message Framing
- Transfer Rate and Timing

Physical Layer

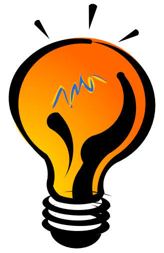
- Signal Level and Bit Representation
- Transmission Medium

Bit representation: dominant and recessive bits

- Dominant bits can overwrite recessive bits
- In CAN: 0 – dominant, 1 – recessive
- Some physical examples:
  - Trivial: Optical example

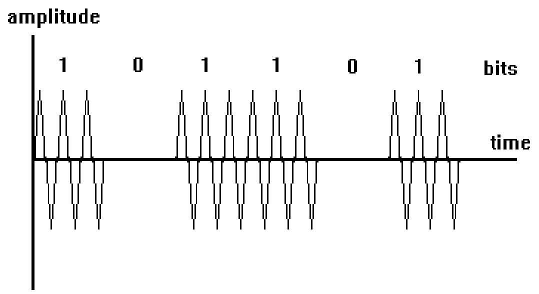


Light off – recessive



Light on – dominant

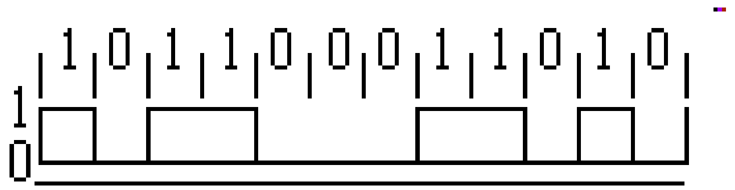
  - More technical: ASK (Amplitude Shift Keying)



Signal present – dominant   Signal absent – recessive

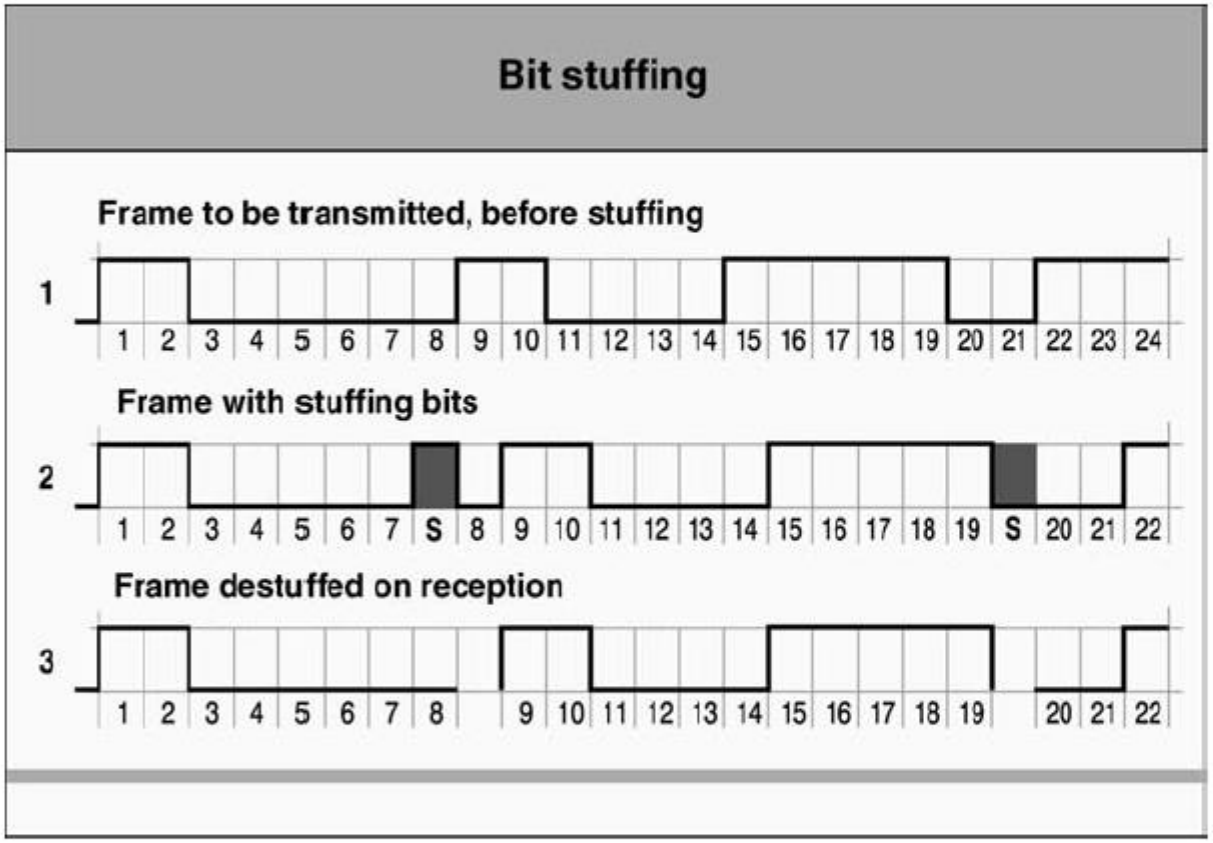
NRZ - encoding

- CAN uses Non-return-to-Zero:
  - “0” is represented as one physical level
  - “1” is represented as another physical level
- Main deficiency
  - As physical level remains constant over a bit, transmitting long sequences of bits leads to loss of synchronization



Bit Stuffing

- CAN bit stuffing rule: after 5 consecutive bits of identical value 1 bit of the opposite value is added
- De-stuffing has to be done at reception

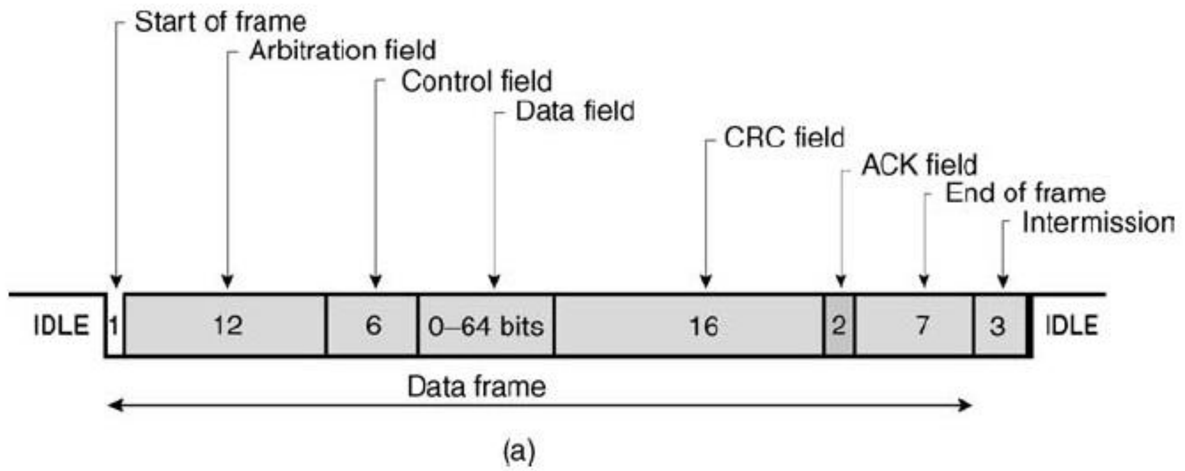


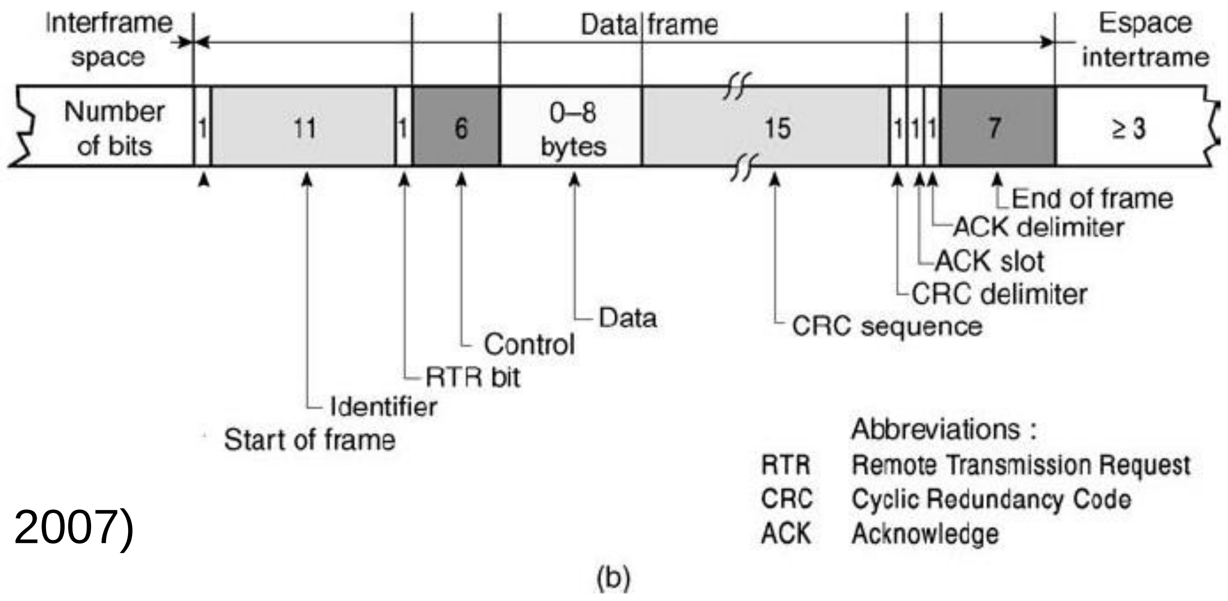
(picture c.f. Parret, 2007)

CAN Frame

- The frame is an envelope for a message (or for a part of a message)
- As the data field in a frame is at most 8 bytes a message can be distributed over more than 1 frame
- There are 4 different kinds of frames:
  - **Data frame** - carries data from a transmitter to the receivers
  - **Remote frame** - is transmitted by a bus unit to request the transmission of the data frame with the same id
  - **Error frame** - is transmitted by any unit on detecting a bus error
  - **Overload frame** - is used to provide for an extra delay between the preceding and the succeeding data or remote frame
- Data frames and remote frames are separated from preceding frames by an **interframe space**
- Two frame formats CAN 2.0A – CAN standard frame and CAN 2.0B – CAN extended frame

CAN 2.0A standard frame





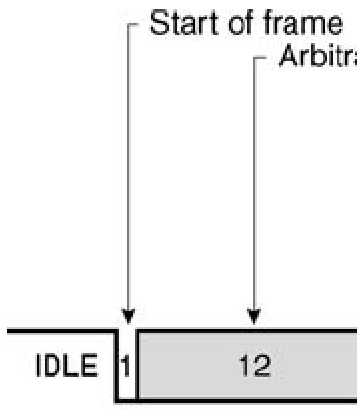
Abbreviations :

RTR	Remote Transmission Request
CRC	Cyclic Redundancy Code
ACK	Acknowledge

(picture c.f. Parret, 2007)

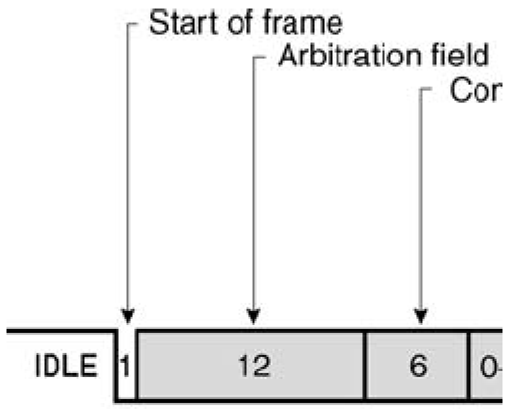
Start of Frame

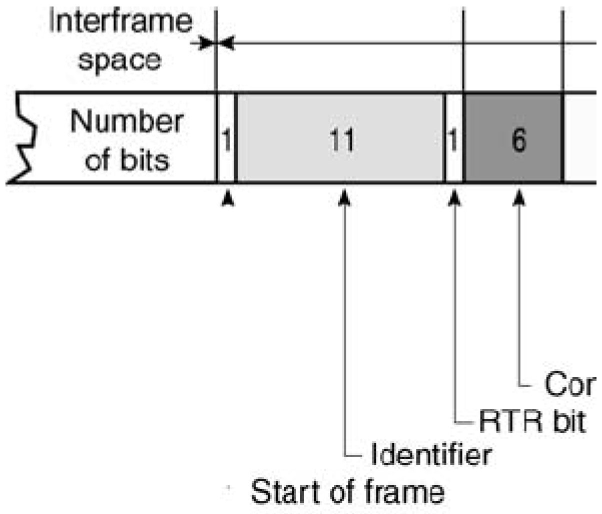
- Used for synchronization
- A single dominant bit
- Can start only if bus was idle



Arbitration Field

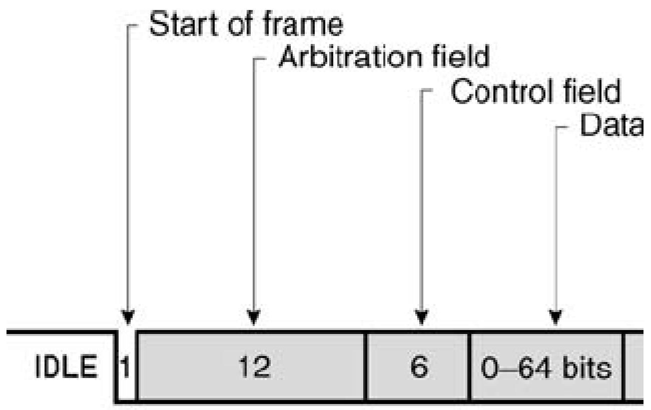
- Identifier 11 bit:
  - Lower the value higher the priority
  - The first 7 bits must not be all recessive => 2032 maximum combinations
- Remote Transmission Request RTR bit:
  - dominant in data frames
  - recessive in remote frames

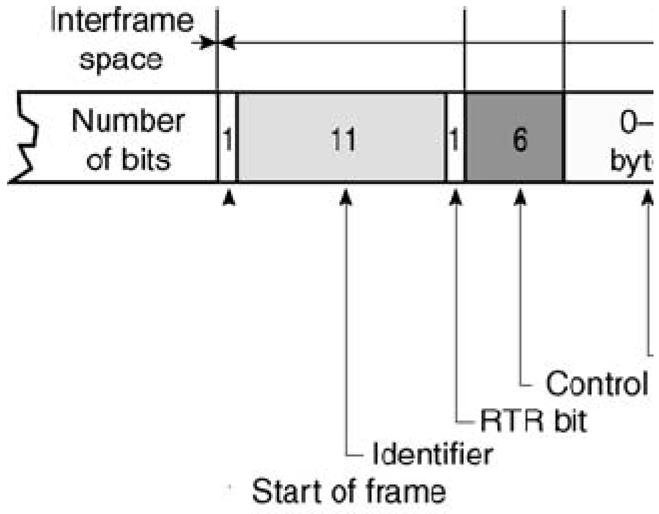


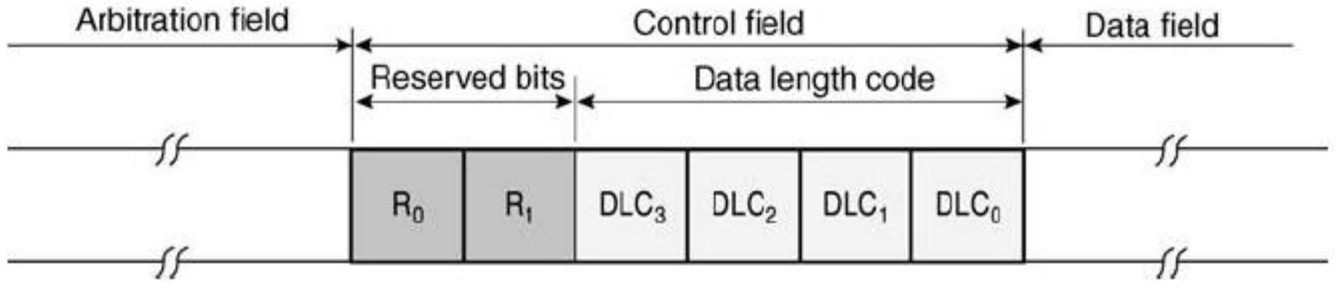


Control Field

- The first 2 bits reserved for upward compatibility
  - Identifier Extension bit – dominant is Standard CAN frame is used
  - r0 bit - reserved
- The last 4 bits indicate the data length

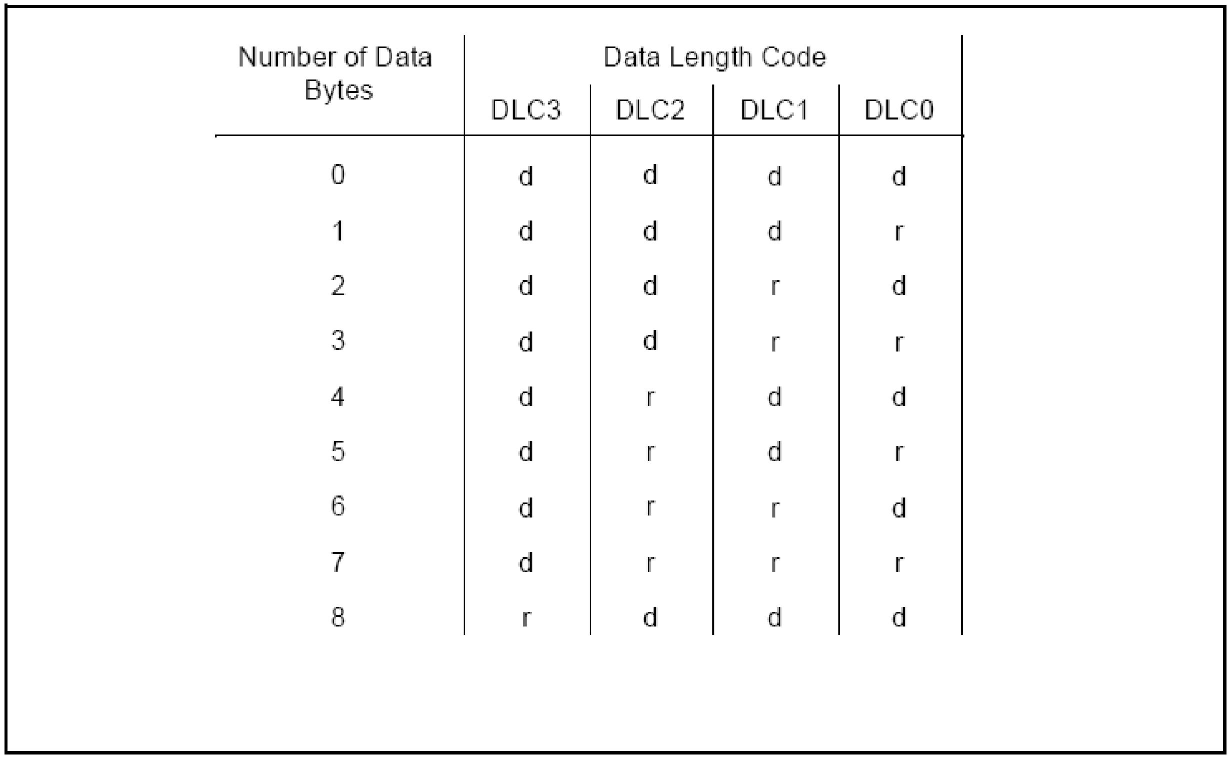






Note that 4 bits indicate 16 possible data lengths and only 9 lengths are used in CAN

(picture c.f. CAN Specification v. 2.0)

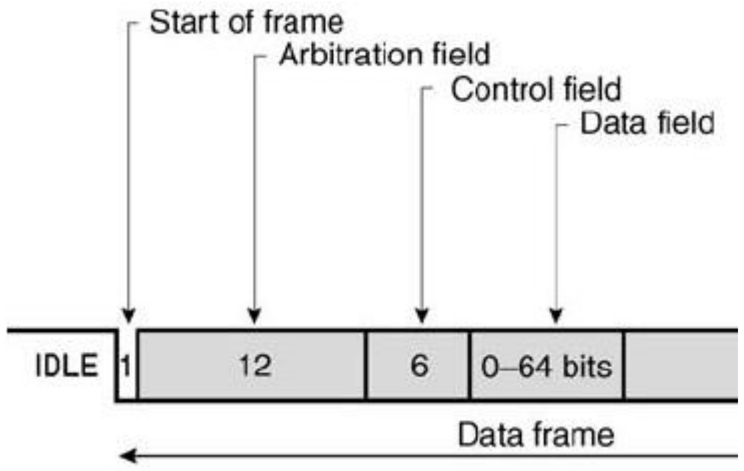


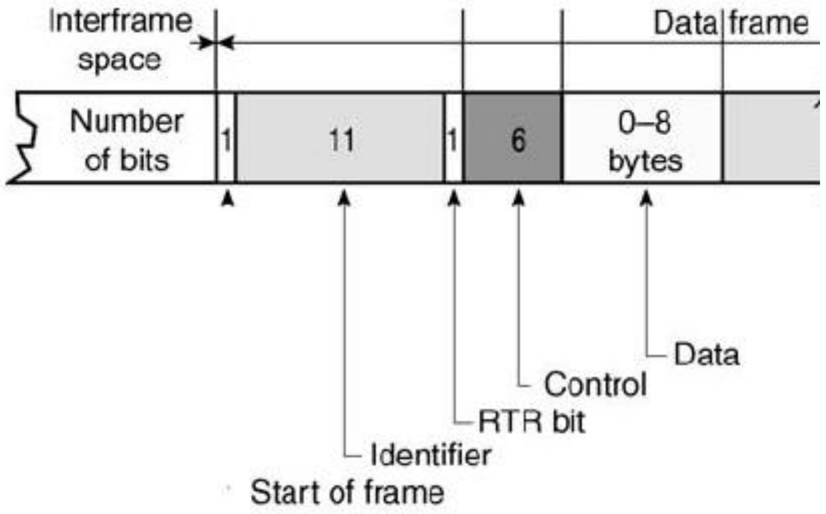
Number of Data Bytes	Data Length Code			
	DLC3	DLC2	DLC1	DLC0
0	d	d	d	d
1	d	d	d	r
2	d	d	r	d
3	d	d	r	r
4	d	r	d	d
5	d	r	d	r
6	d	r	r	d
7	d	r	r	r
8	r	d	d	d

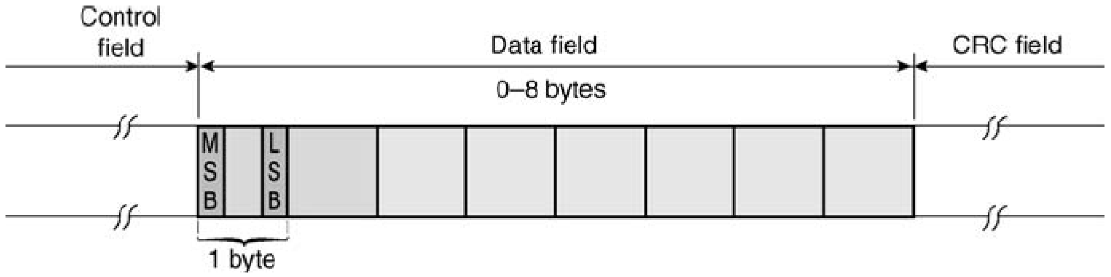
DATA FRAME: admissible numbers of data bytes: {0,1,....,7,8}.
Other values may not be used.

Data Field

- At most 8 bytes of actual data

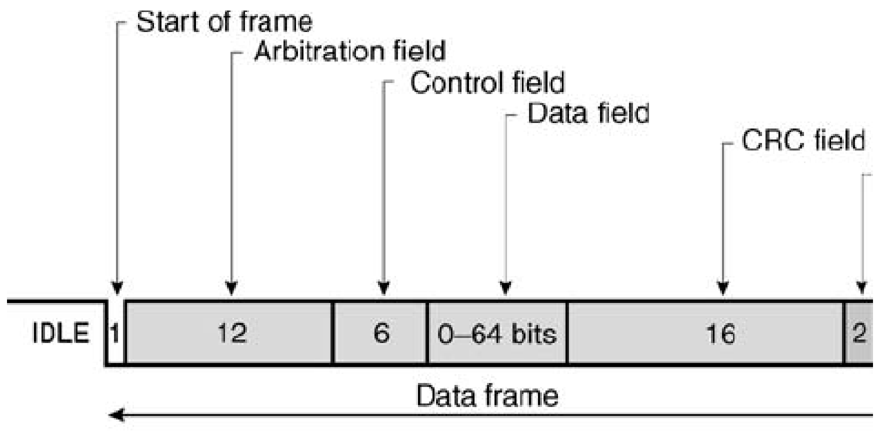


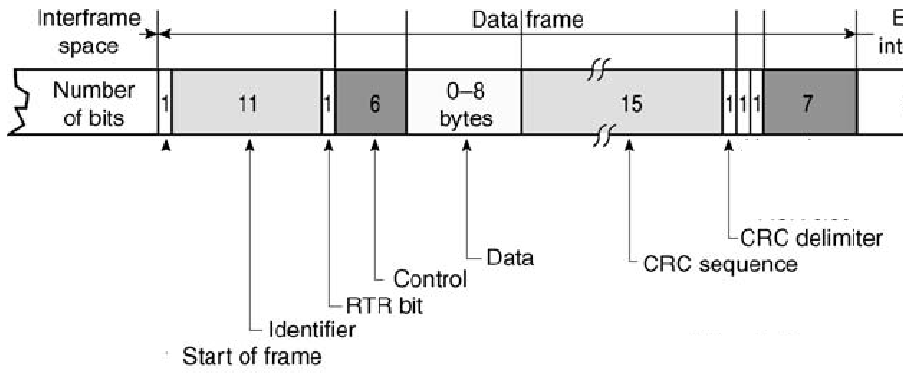




CRC Field

- 15 bits with Hamming distance 6
- 1 bit CRC delimiter (always recessive)
- Must detect:
  - up to 5 randomly distributed (independent) errors in a message are detected
  - burst errors of length less than 15 in a message are detected
  - errors of any odd number in a message are detected
- Total residual error probability for undetected corrupted messages: less than message error rate $4.7 * 10^{-11}$





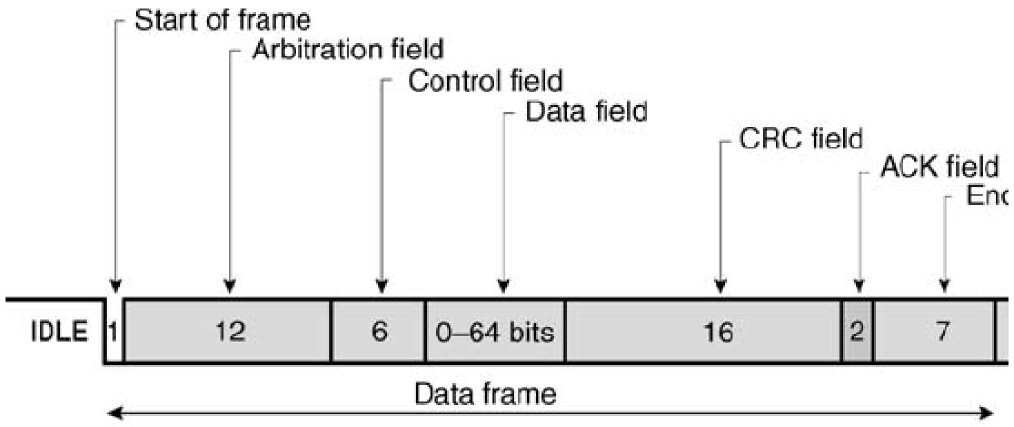
BCH (Bose and Ray-Chaudhuri)

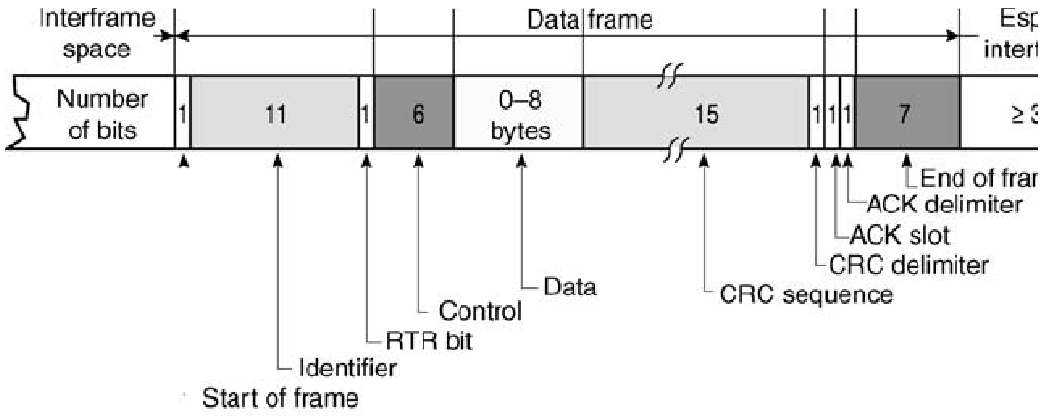
- Also known as CRC-15
- Close to optimal
- The following polynomial is used:

$$P(x) = x^{15} + x^{14} + x^{10} + x^{8} + x^{7} + x^{4} + x^{3} + 1$$

ACK Field

- 1 ACK slot recessive, overwritten by a dominant bit
- 1 ACK delimiter, always recessive





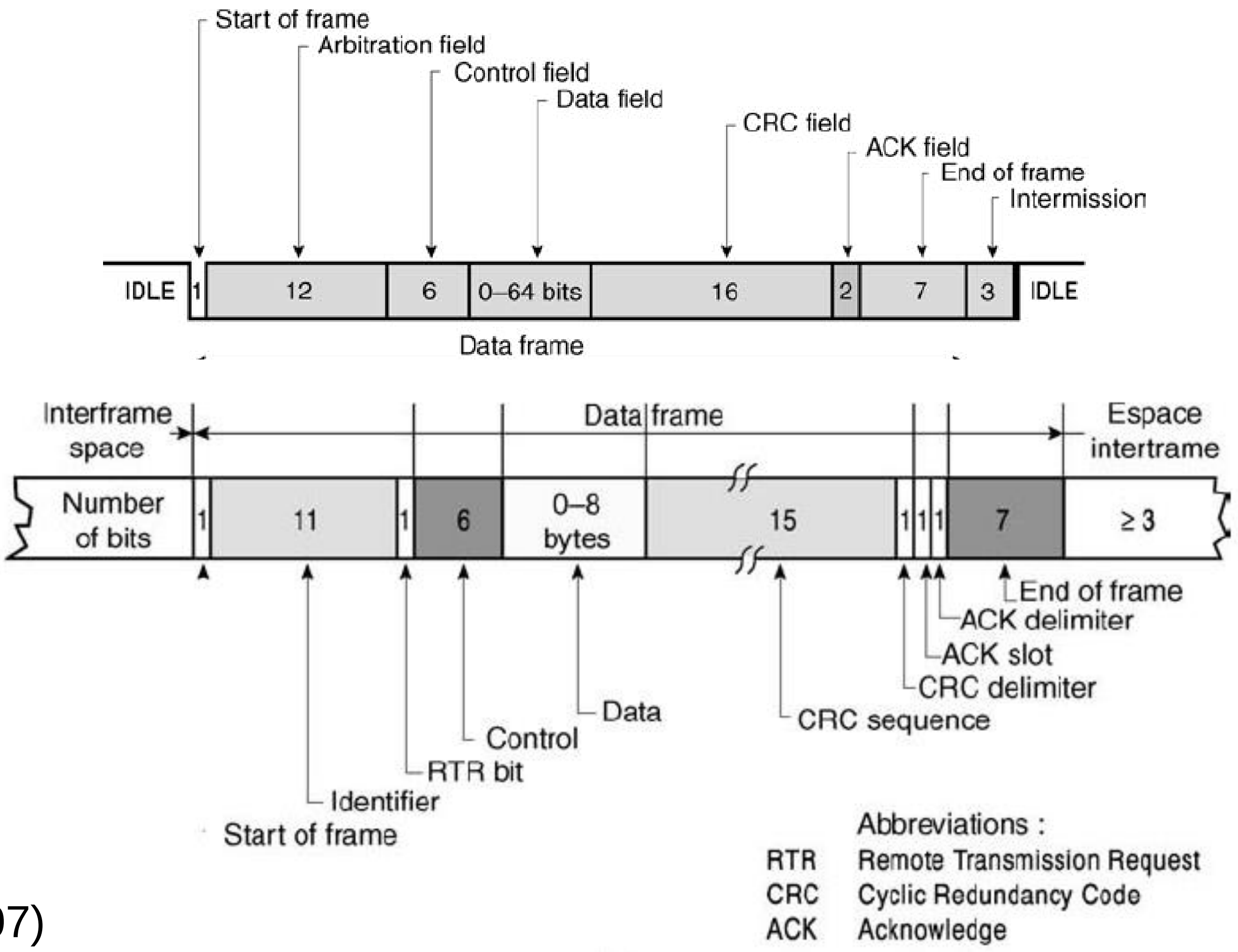
(picture c.f. Parret, 2007)

ACK management

- Sender writes a recessive bit in the slot
- Each receiver that correctly received the message writes a dominant bit
- Consequence: the sender knows if there is at least one receiver had correctly received it (does this imply that the message was correctly transmitted?)
- Still, this does not mean the receiver actually uses this information !
- How to decide if the message was received by the intended node ? Use layer 7

End of Frame Field (EOF)

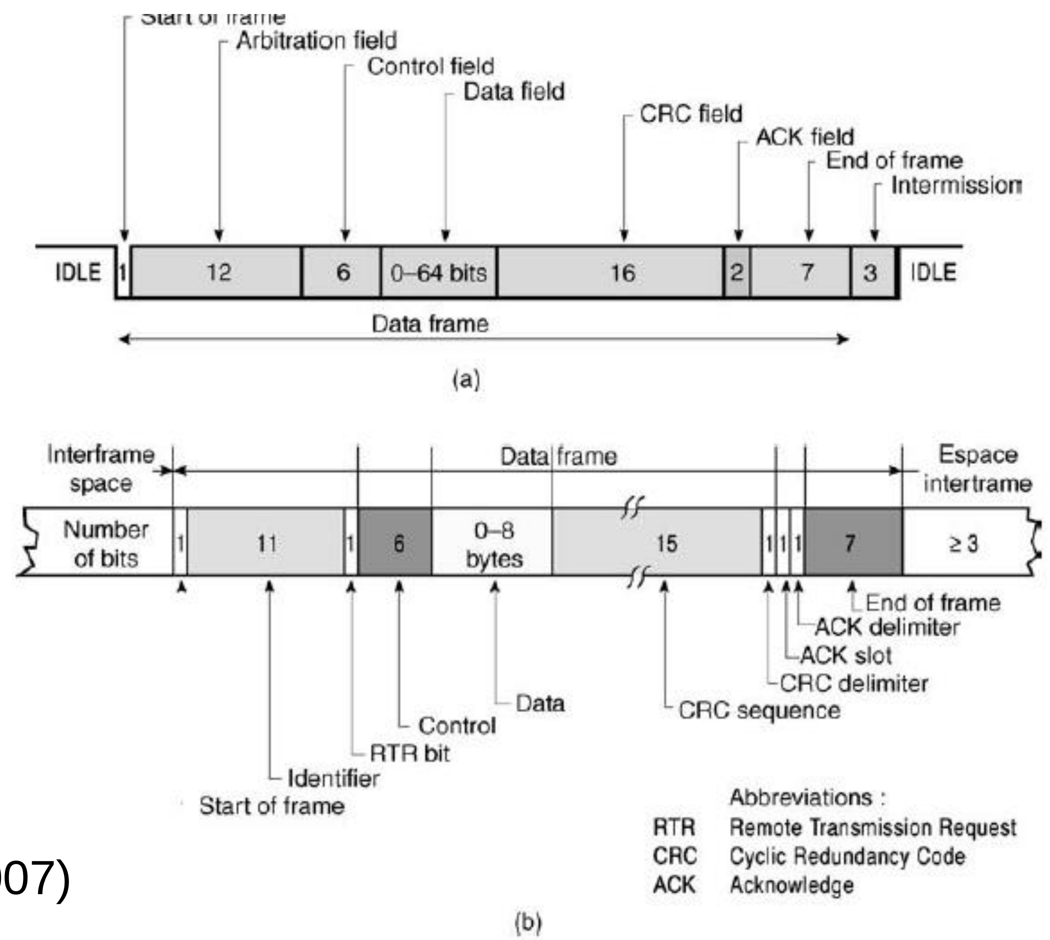
- Just 7 recessive bits



(picture c.f. Parret, 2007)

Interframe Space (IFS)

- IFS - 3 recessive bits
- A successfully ACK is followed by 11 recessive bits

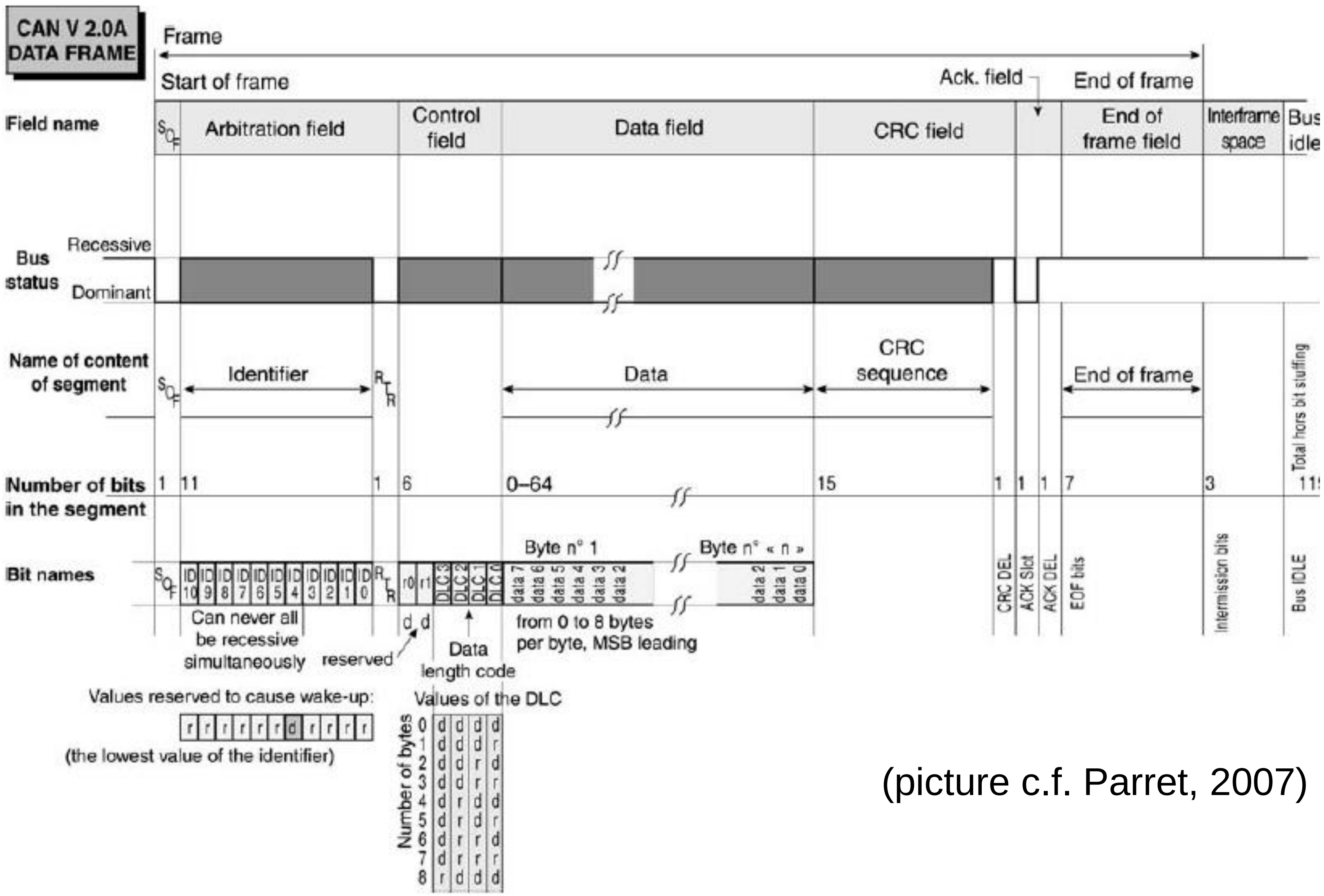


(picture c.f. Parret, 2007)

More?

- Idle, recessive bits >= 0

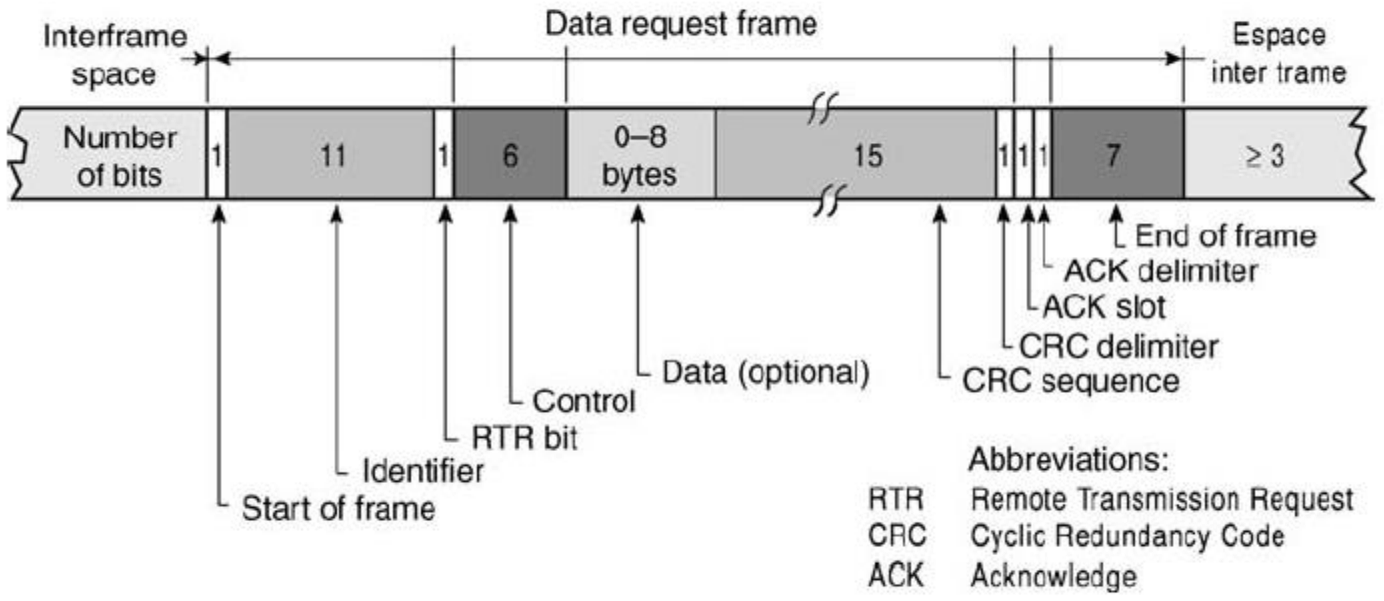
See the big picture



(picture c.f. Parret, 2007)

Remote Frame

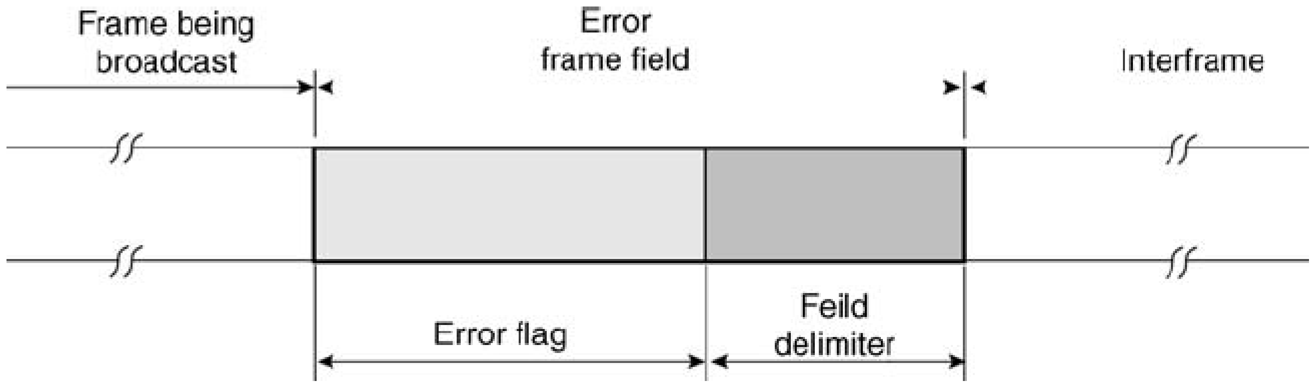
- RTR bit is recessive
- Data is 0 bytes



(picture c.f. Parret, 2007)

Error Frame

- Error flag 6 bits:
  - Active error – 6 dominant bits (no bit stuffing => violates bit stuffing=>chain reaction)
  - Passive error (may overlap with active error)– 6 recessive bits (no bit stuffing)
- Error delimiter 8 bits
- Transmission of error frame starts simultaneously with the next bit after the detection of an error
- CRC error frames start after the ACK delimiter



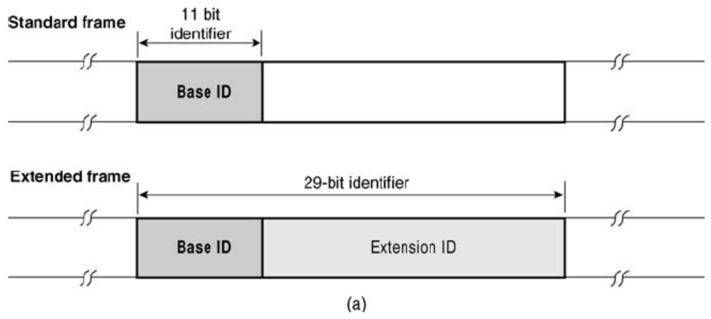
(picture c.f. Parret, 2007)

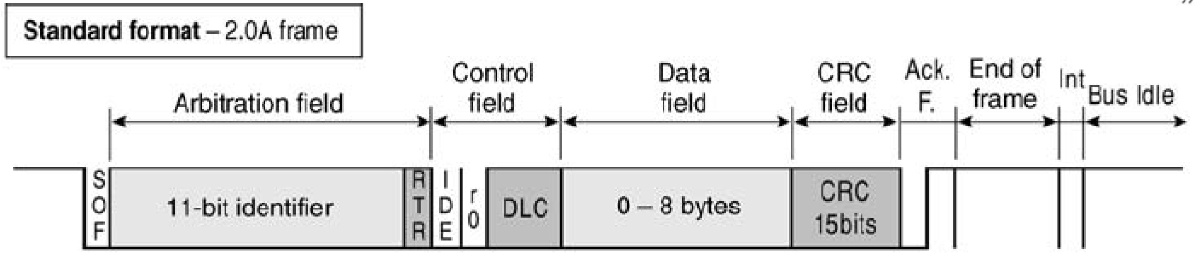
Overload Frame

- Used to indicate that a unit cannot receive additional frames due to overload
- Identical to an active error frame (again 6 consecutive bits => chain reaction)
- At most 2 consecutive overload frames can be used
- According to CANSpec there are 2 situation for OFs:
  1) The internal conditions of a receiver, which requires a delay of the next DATA FRAME or REMOTE FRAME.
  2) Detection of a ’dominant’ bit during INTERMISSION.
- CANSpec: The start of an OVERLOAD FRAME due to OVERLOAD condition 1 is only allowed to be started at the first bit time of an expected INTERMISSION, whereas OVERLOAD FRAMEs due to OVERLOAD condition 2 start one bit after detecting the ’dominant’ bit
- Remark CANSpec: In case that there is a ’dominant’ bit detected during the 3rd bit of INTERMISSION locally at some node, the other nodes will not interpret the OVERLOAD FLAG correctly, but interpret the first of these six ’dominant’ bits as START OF FRAME. The sixth ’dominant’ bit violates the rule of bit stuffing causing an error condition

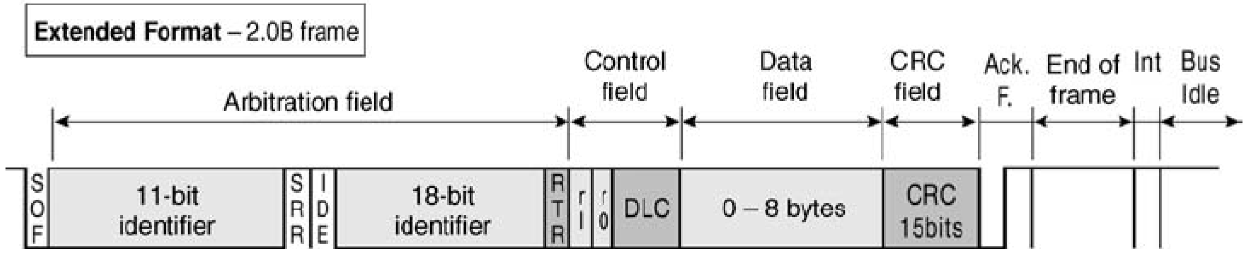
CAN 2.0 B frame

- Extended CAN, uses 29 bit identifiers
- SRR – substitute remote request bit
- IDE is recessive
- Remark: Since the SRR bit is received before the IDE bit, a receiver cannot decide instantly whether it receives a RTR or a SRR bit. That means only the IDE bit decides whether the frame is a Standard Frame or an Extended Frame.
- … see pictures below



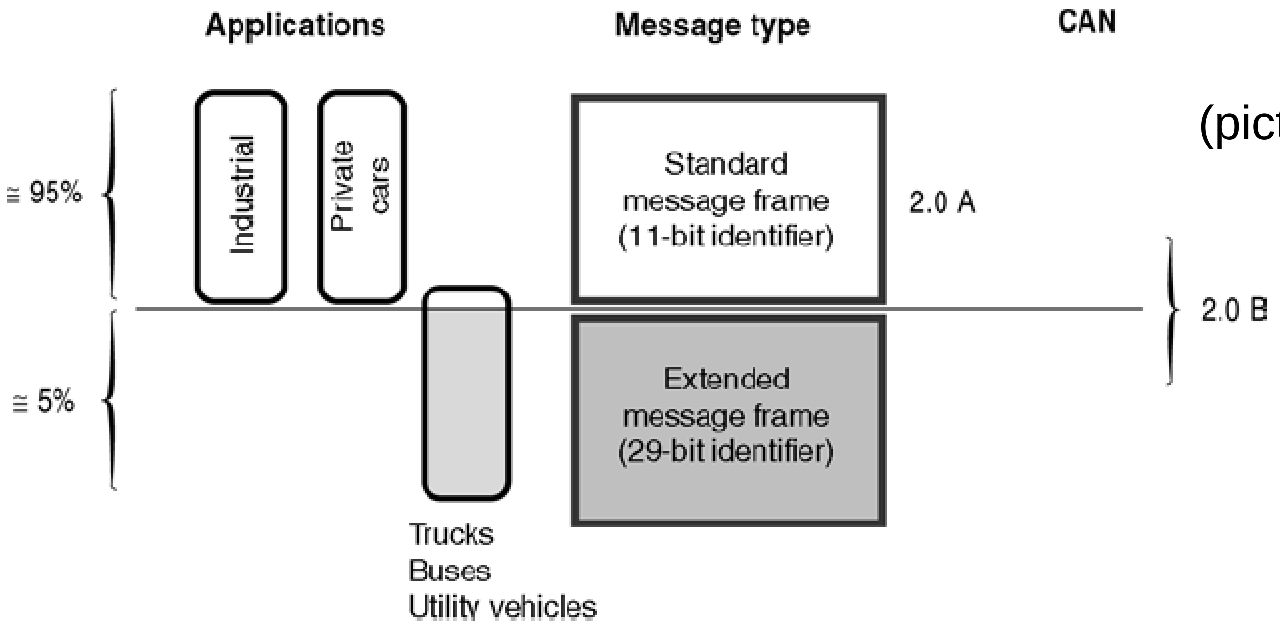


(picture c.f. Parret, 2007)



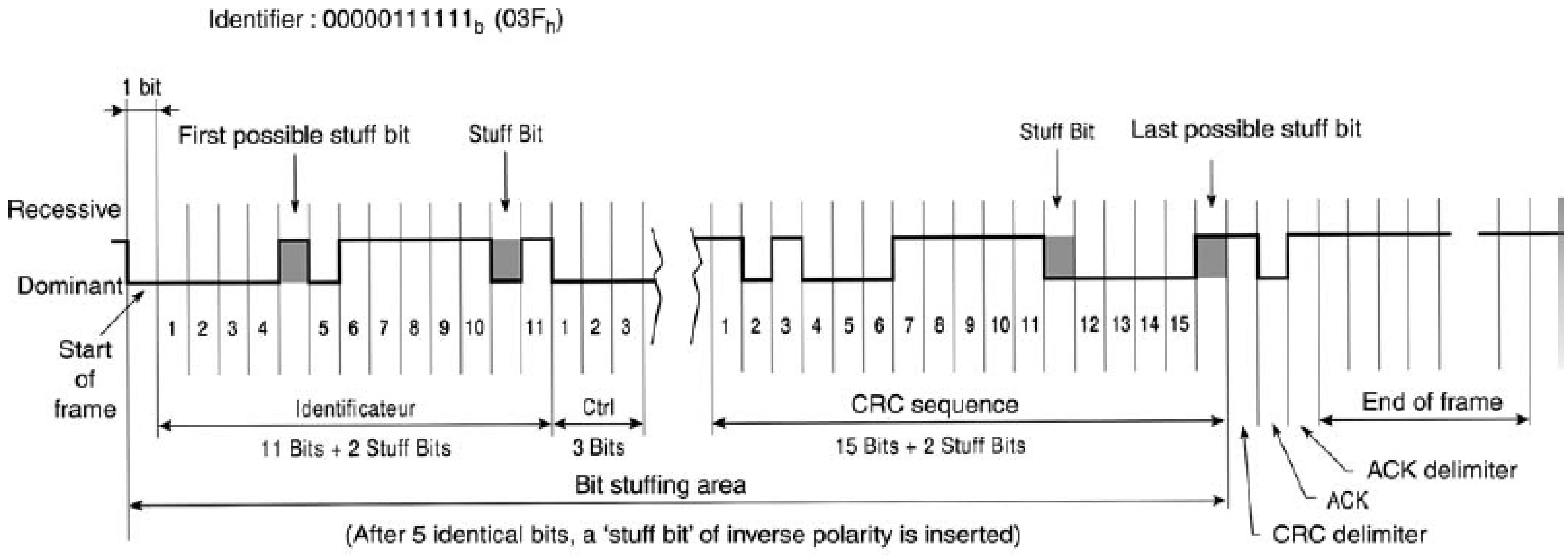
CAN 2.0 A – CAN 2.0 B interoperability

- CAN 2.0 B controllers must support standard CAN frames
- CAN 2.0 B Active – can read/write CAN 2.0B frames
- CAN 2.0 B Passive – can read CANB 2.0 frames
- CAN 2.0 B can work with standard or extended frames at the same time



(picture c.f. Parret, 2007)

CAN bit stuffing Revisited



(picture c.f. Parret, 2007)

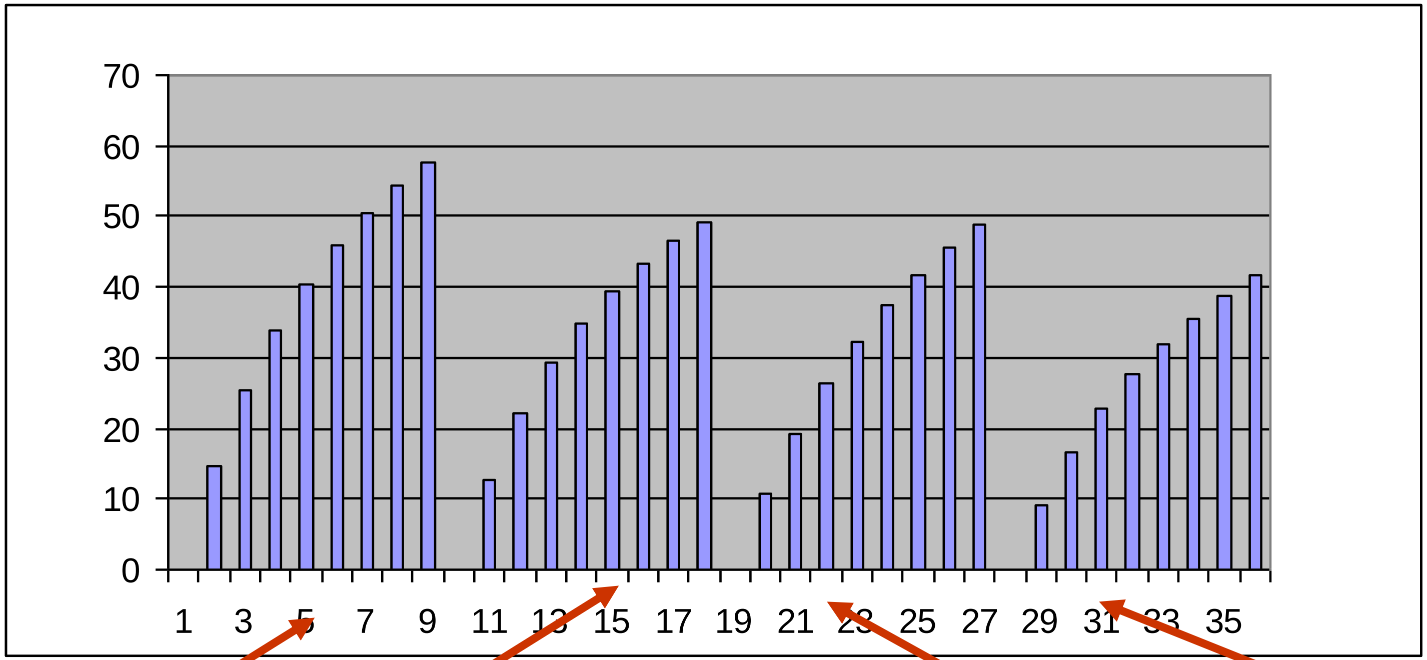
CAN « Bit Stuffing »

Figure 2.6

- The maximum size of the stuffing area: 98 bits => maximum 19 stuff bits

CAN frame overhead

- Overhead: Data size / Frame size (consider also bit stuffing)



CAN 2.0 A (without stuffed bits)

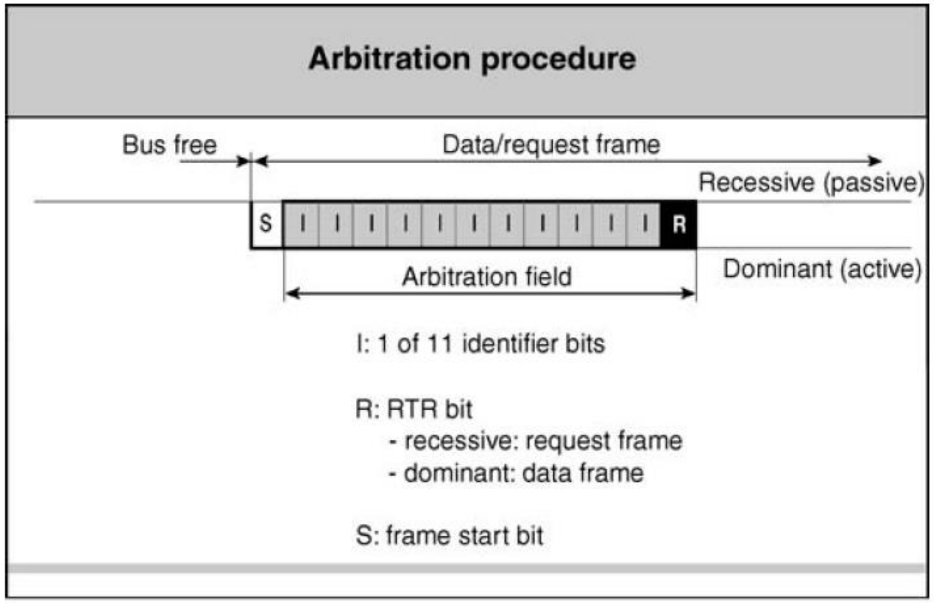
CAN 2.0 A (with stuffed bits)

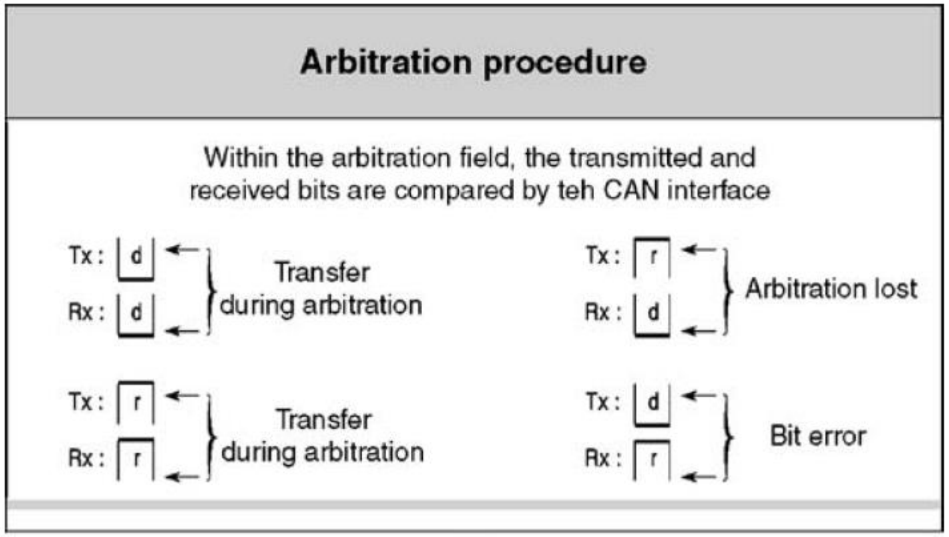
CAN 2.0 B (without stuffed bits)

CAN 2.0 B (with stuffed bits)

Arbitration procedure

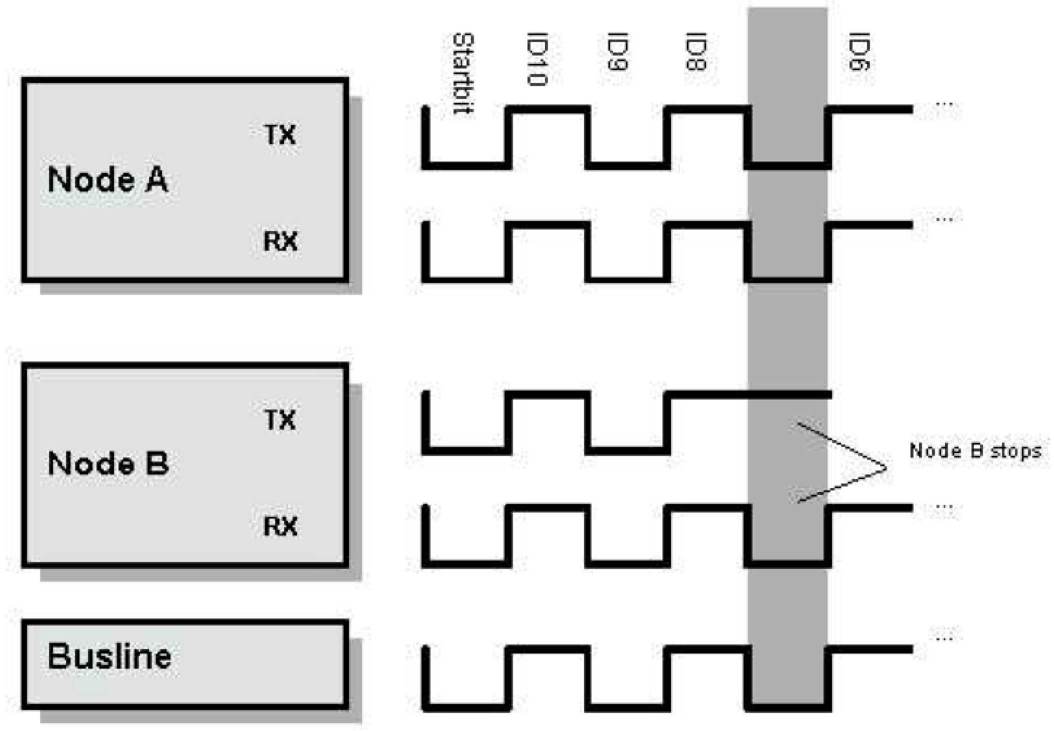
- **High priority messages must have low identifiers !!!**




Arbitration procedure
Within the arbitration field, the transmitted and received bits are compared by teh CAN interface
Tx : d
Rx : d
Transfer during arbitration
Tx : r
Rx : d
Arbitration lost
Tx : r
Rx : r
Transfer during arbitration
Tx : d
Rx : r
Bit error

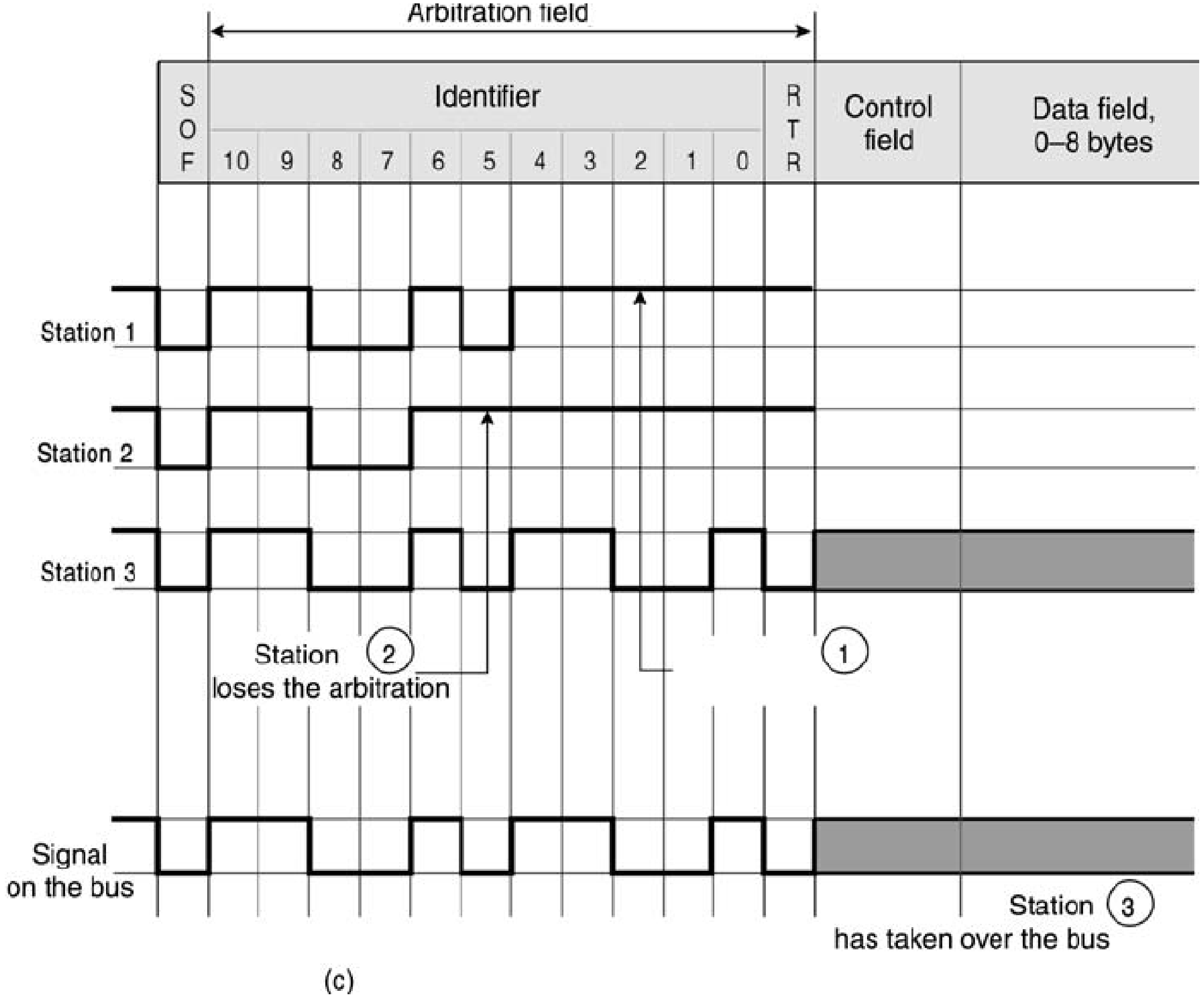
(picture c.f. Lawrenz, 2007)

Example …



(picture c.f. Lawrenz, 2007)

… and another example

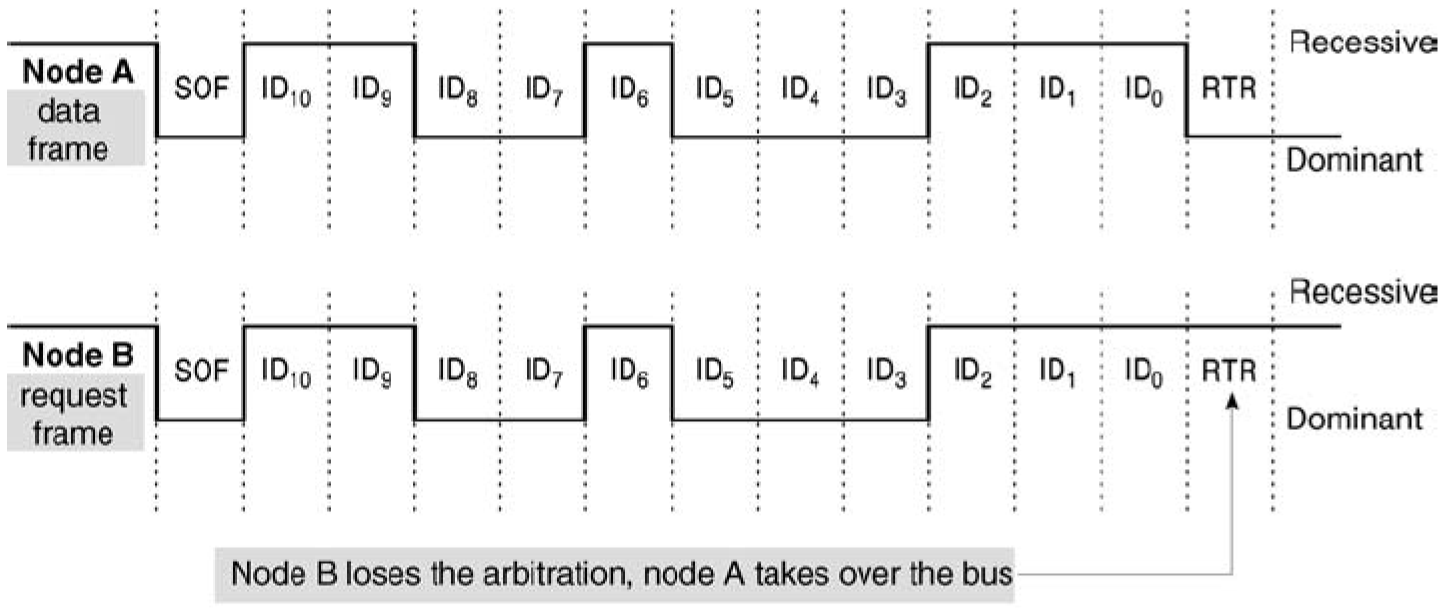


(picture c.f. Parret, 2007)

Arbitration: Data Frame vs. Remote Frame

- Data frame wins

Standard frame (data/request), with identical identifiers

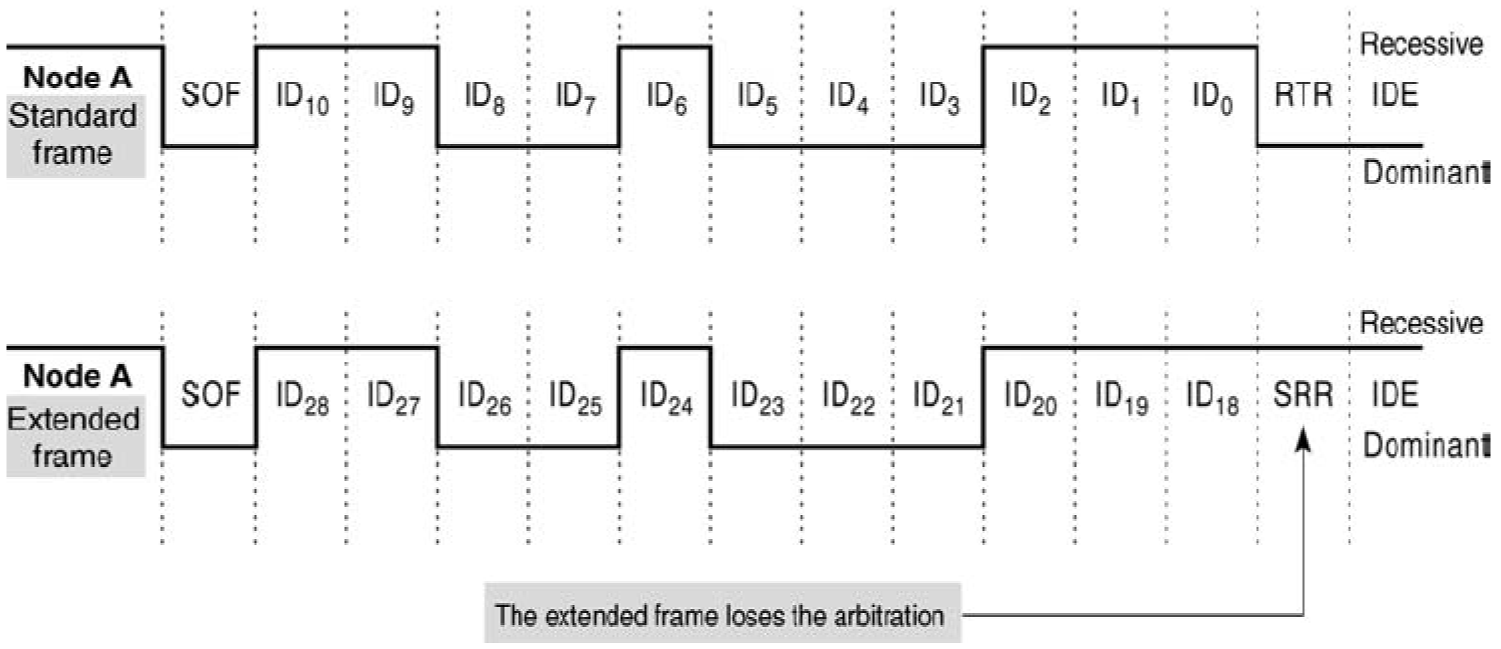


(b)

Arbitration: Standard Frame vs. Extended Frame

- Standard frame wins

Arbitration between a standard frame and an extended frame



Arbitration type
ND + CSMA/CA + AMP

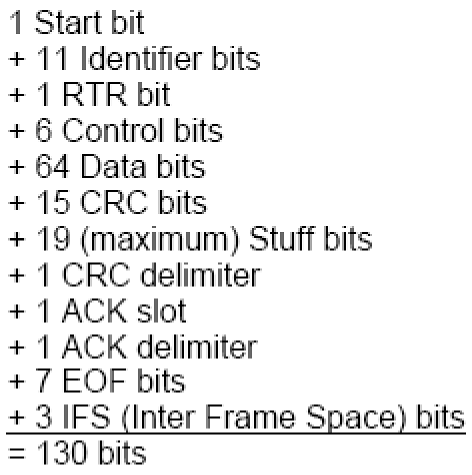
i.e. Non-Destructive + Carrier Sense Multiple Access / Collision Avoidance + Arbitration on Message Priority

Timing Considerations

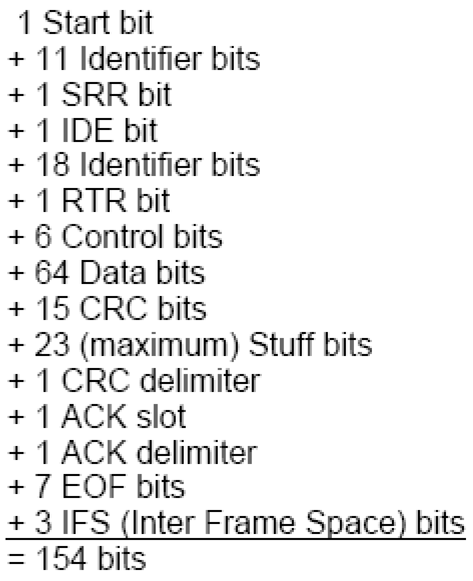
- Can we compute the maximum delay at which a particular message will arrive ?

Timing Considerations

- Can we compute the maximum delay at which a particular message will arrive ?
- Only for the highest priority message



```
1 Start bit
+ 11 Identifier bits
+ 1 RTR bit
+ 6 Control bits
+ 64 Data bits
+ 15 CRC bits
+ 19 (maximum) Stuff bits
+ 1 CRC delimiter
+ 1 ACK slot
+ 1 ACK delimiter
+ 7 EOF bits
+ 3 IFS (Inter Frame Space) bits
= 130 bits
```



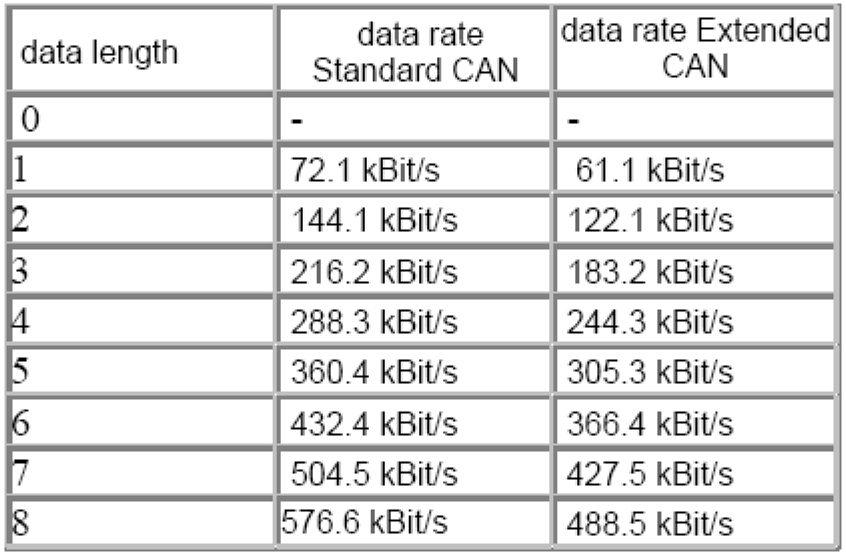
```
1 Start bit
+ 11 Identifier bits
+ 1 SRR bit
+ 1 IDE bit
+ 18 Identifier bits
+ 1 RTR bit
+ 6 Control bits
+ 64 Data bits
+ 15 CRC bits
+ 23 (maximum) Stuff bits
+ 1 CRC delimiter
+ 1 ACK slot
+ 1 ACK delimiter
+ 7 EOF bits
+ 3 IFS (Inter Frame Space) bits
= 154 bits
```

Questions

- Which is the worst time interval at which an interrupt for a receive routine is triggered ?
- How fast will the highest priority message arrive ?

Data rate revisited

- Data rate = useful data rate from the entire frame at 1Mbps



data length	data rate Standard CAN	data rate Extended CAN
0	-	-
1	72.1 kBit/s	61.1 kBit/s
2	144.1 kBit/s	122.1 kBit/s
3	216.2 kBit/s	183.2 kBit/s
4	288.3 kBit/s	244.3 kBit/s
5	360.4 kBit/s	305.3 kBit/s
6	432.4 kBit/s	366.4 kBit/s
7	504.5 kBit/s	427.5 kBit/s
8	576.6 kBit/s	488.5 kBit/s

Error Detection

- Five types of errors are defined on CAN Standard:
  1) **BIT ERROR** - A unit that is sending a bit on the bus also monitors the bus. A BIT ERROR has to be detected at that bit time, when the bit value that is monitored is different from the bit value that is sent. An **exception** is the sending of a ’recessive’ bit during the stuffed bit stream of the **ARBITRATION** FIELD or during the **ACK SLOT**. Then no BIT ERROR occurs when a ’dominant’ bit is monitored. A TRANSMITTER sending a PASSIVE ERROR FLAG and detecting a ’dominant’ bit does not interpret this as a BIT ERROR.
  2) **STUFF ERROR** - A STUFF ERROR has to be detected at the bit time of the 6th consecutive equal bit level in a message field that should be coded by the method of bit stuffing.
  3) **CRC ERROR** - The CRC sequence consists of the result of the CRC calculation by the transmitter The receivers calculate the CRC in the same way as the transmitter. A CRC ERROR has to be detected, if the calculated result is not the same as that received in the CRC sequence.
  4) **FORM ERROR** - A FORM ERROR has to be detected when a fixed-form bit field contains one or more illegal bits.
  5) **ACKNOWLEDGMENT ERROR** - An ACKNOWLEDGMENT ERROR has to be detected by a transmitter whenever it does not monitor a ’dominant’ bit during the ACK SLOT.

When is a message valid ?

- The point of time at which a message is taken to be valid, is different for the transmitter and the receivers of the message:
- Transmitter:
  - The message is valid for the transmitter, if there is no error until the end of END OF FRAME. If a message is corrupted, retransmission will follow automatically and according to prioritization. In order to be able to compete for bus access with other messages, retransmission has to start as soon as the bus is idle.
- Receivers:
  - The message is valid for the receivers, if there is no error until the last but one bit of END OF FRAME.

- Question: Why are these rules different for senders and receivers ?

Error Handling

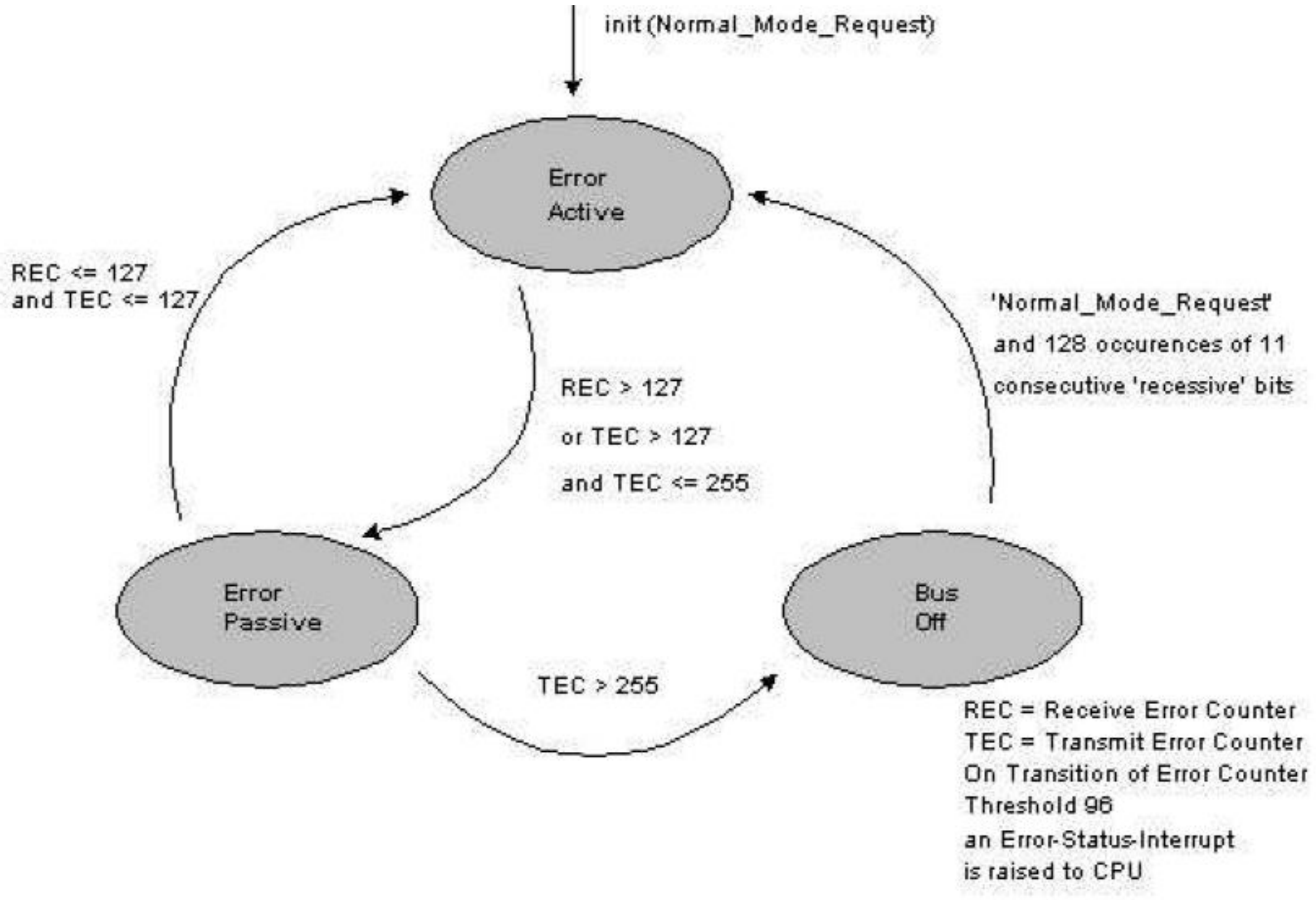
- The following steps are done:
  1) Error detected
  2) Error frame transmitted
  3) Message discarded by all nodes
  4) Error counters incremented
  5) Message retransmitted

Error Limitation

- To prevent a permanently disturbed bus each CAN controller has 3 states:
  1) **Error active** - an ’error active’ unit can normally take part in bus communication and sends an ACTIVE ERROR FLAG when an error has been detected.
  2) **Error passive** - an ’error passive’ unit must not send an ACTIVE ERROR FLAG. It takes part in bus communication but when an error has been detected only a PASSIVE ERROR FLAG is sent. Also after a transmission, an ’error passive’ unit will wait before initiating a further transmission. (See SUSPEND TRANSMISSION)
  3) **Bus off** - a ’bus off’ unit is not allowed to have any influence on the bus. (E.g. output drivers switched off.)

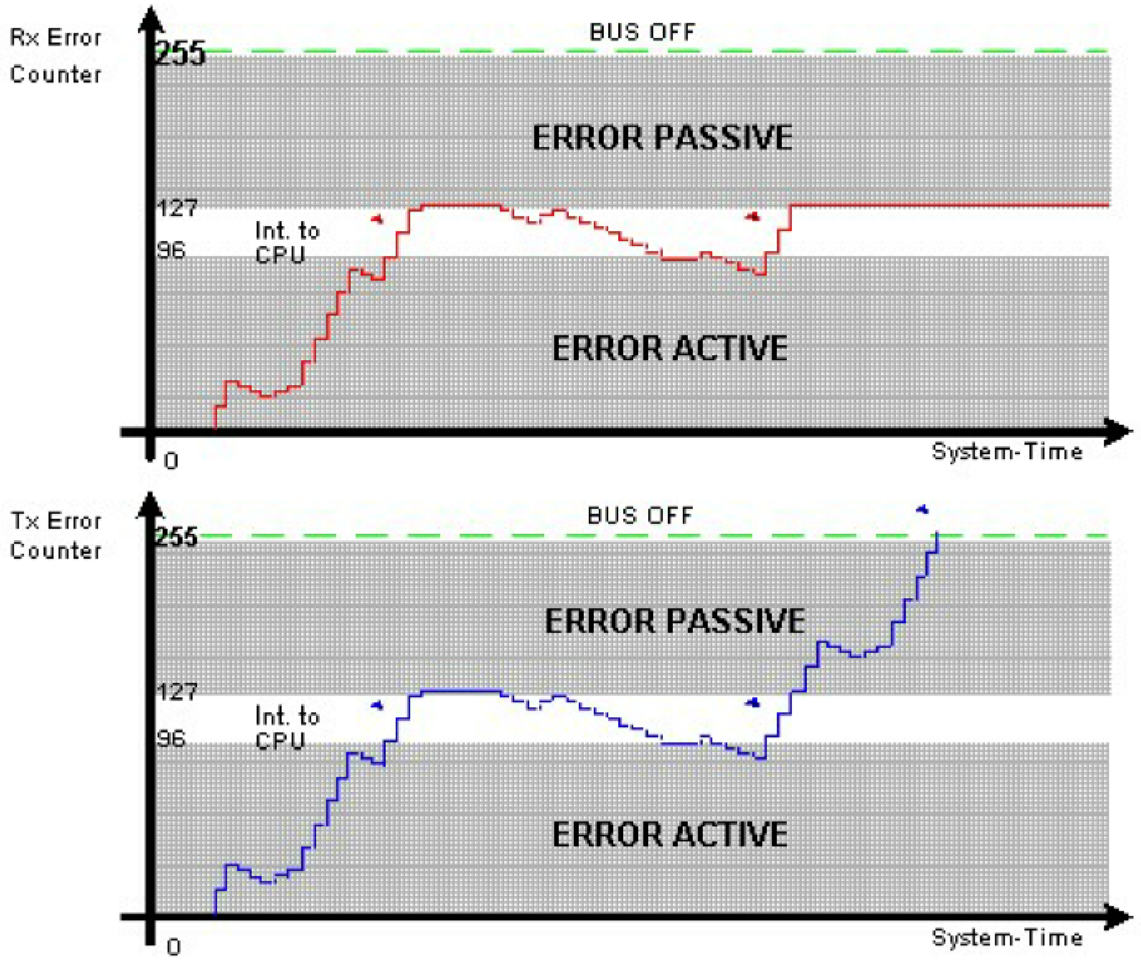
CAN node states

- Error passive and error active



init (Normal_Mode_Request)
Error Active
REC <= 127 and TEC <= 127
REC > 127 or TEC > 127 and TEC <= 255
'Normal_Mode_Request' and 128 occurences of 11 consecutive 'recessive' bits
Error Passive
Bus Off
TEC > 255
REC = Receive Error Counter
TEC = Transmit Error Counter
On Transition of Error Counter Threshold 96 an Error-Status-Interrupt is raised to CPU

■ Evolution of error counters (example)



Tx Error counter modifications (brief and incomplete)

- + 8 when an error in the transmitted message is detected
- - 1 when a message is correctly transmitted and acknowledged (not applied when counter is 0)

Rx error counter modifications (in brief)

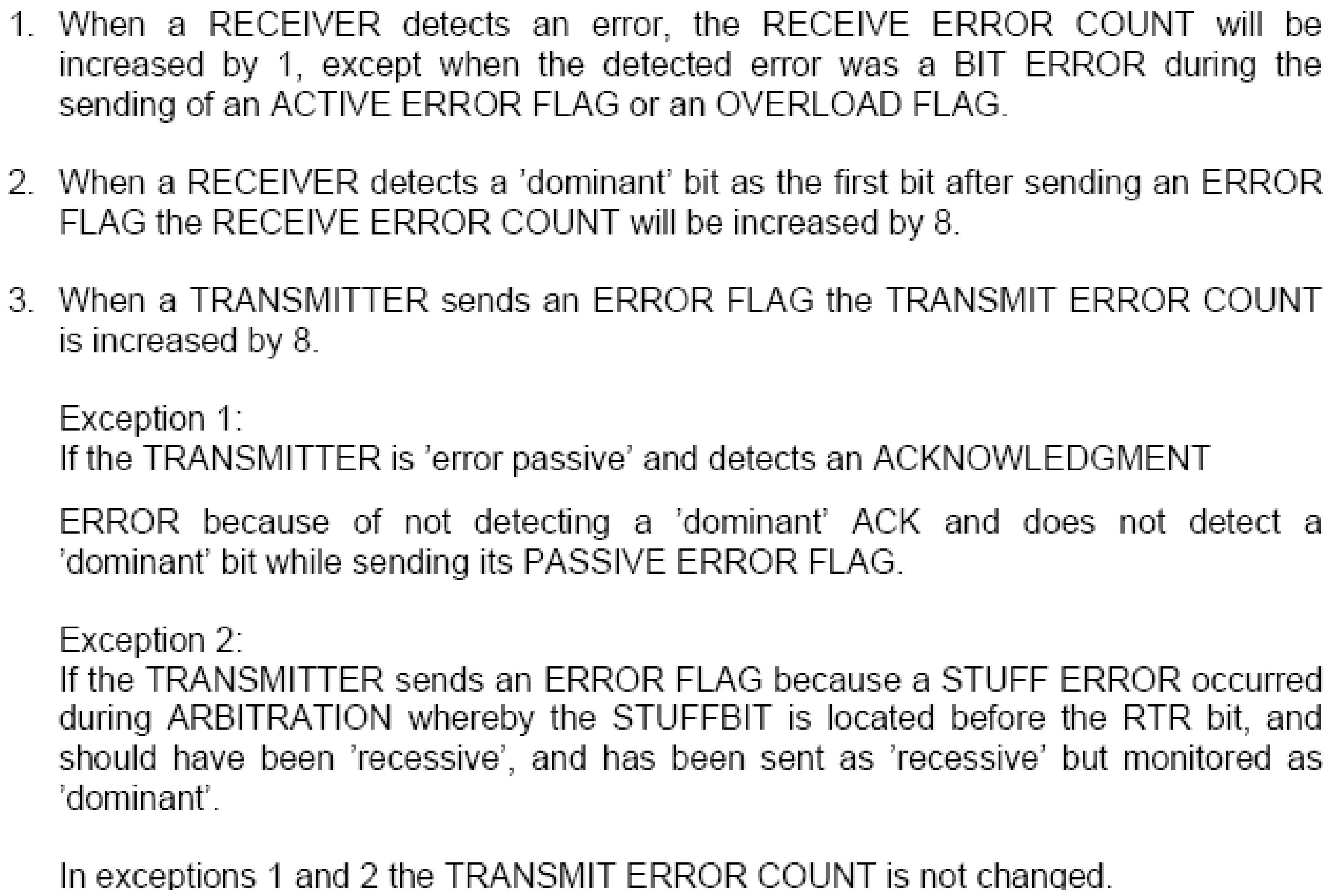
- + 1 when it detects an error
- + 8 when it is the only unit that detected the error
- - 1 when a frame is successfully received

Efficiency of Error Management

- Main idea: locate faulty nodes => faulty nodes always accumulate more points then other nodes

Error Counters Modification Rules

- According to CAN Spec (note that more than 1 rule can be applied)



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.

   Exception 1:
   If the TRANSMITTER is 'error passive' and detects an ACKNOWLEDGMENT

   ERROR because of not detecting a 'dominant' ACK and does not detect a 'dominant' bit while sending its PASSIVE ERROR FLAG.

   Exception 2:
   If the TRANSMITTER sends an ERROR FLAG because a STUFF ERROR occurred during ARBITRATION whereby the STUFFBIT is located before the RTR bit, and should have been 'recessive', and has been sent as 'recessive' but monitored as 'dominant'.

   In exceptions 1 and 2 the TRANSMIT ERROR COUNT is not changed.

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.

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.

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.

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 ACTIVE 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.

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.

Example 1

- What happens if a node A gets a receive error due to a local disturbance ?

Example 1

- Consider a node A gets a receive error due to a local disturbance:
  - A increments its Rx counter by 1 with rule 1)
  - A sends an active error flag
  - Other nodes detect a bit stuffing error on the 6th bit of the error flag and send an error flag
  - A detects a dominant after its error flag and increments Rx counter by 8 according to rule 2)

Example 2

- What happens if node A transmits when he is the only node on-line ?

Example 2

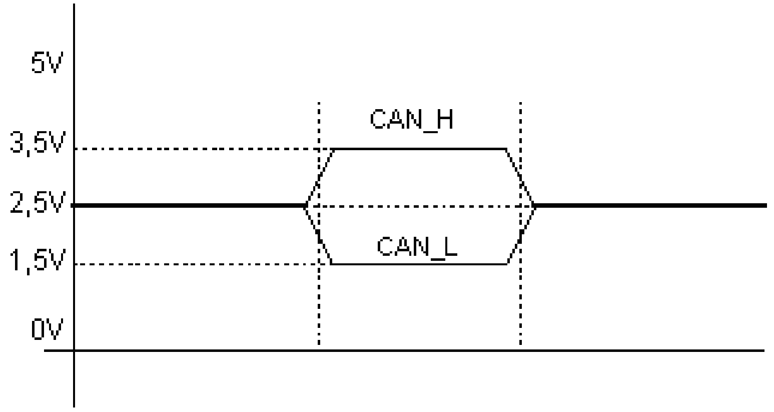
- Consider a node A transmits when he is the only node on-line:
  - A does not get an acknowledgment
  - A increment its Tx counter and get in error passive
  - A cannot get to BUS OFF due to exception 1 in rule 3

Lecture 4.

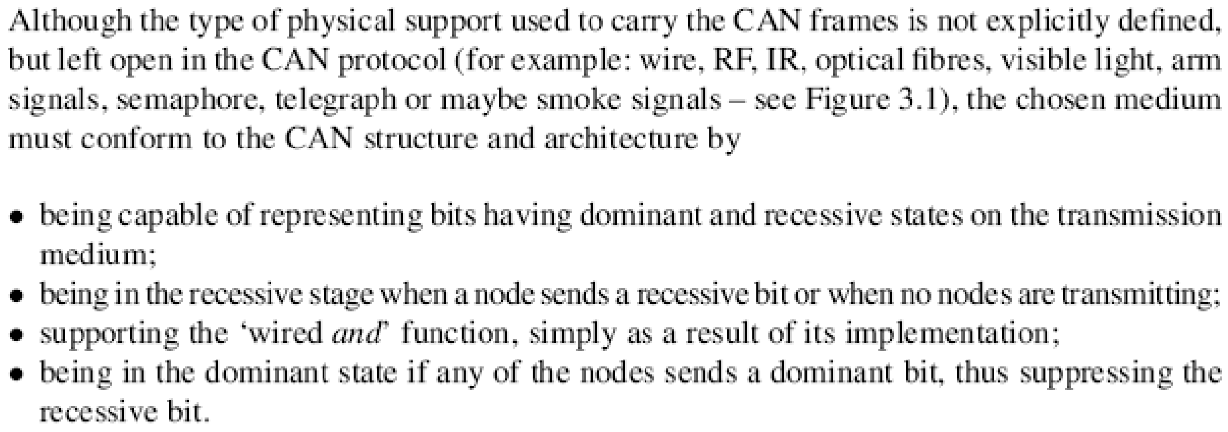
Physical Layer in CAN: Implementation aspects and bit timing procedures, hardware synchronization and resynchronization, high speed and fault tolerant CAN, theoretical limitations of CAN speed, oscillator tolerance for CAN networks

ISO 11898-5 specifies the CAN physical layer

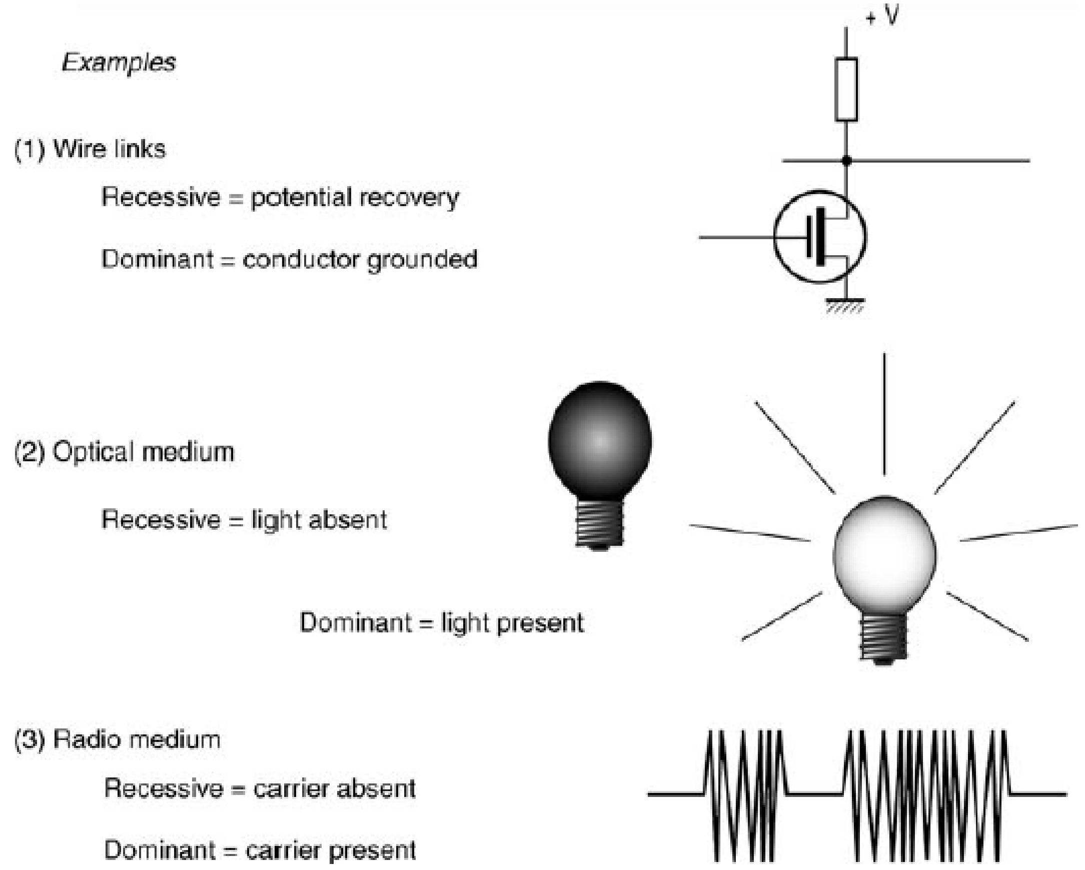
- Recessive condition: if CAN-H is not higher than CAN-L plus 0.5 V
- Dominant condition: if CAN-H is at least 0.9 V higher than CAN-L
- Each node must be capable to produce output voltage between 1.5V and 3.0V



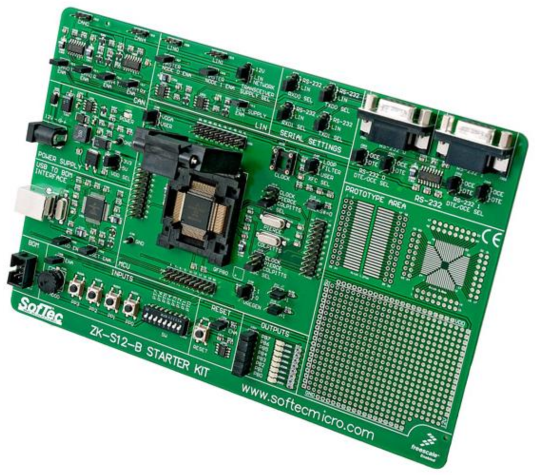
Parret, 2007:


Although the type of physical support used to carry the CAN frames is not explicitly defined, but left open in the CAN protocol (for example: wire, RF, IR, optical fibres, visible light, arm signals, semaphore, telegraph or maybe smoke signals – see Figure 3.1), the chosen medium must conform to the CAN structure and architecture by

- being capable of representing bits having dominant and recessive states on the transmission medium;
- being in the recessive stage when a node sends a recessive bit or when no nodes are transmitting;
- supporting the 'wired *and*' function, simply as a result of its implementation;
- being in the dominant state if any of the nodes sends a dominant bit, thus suppressing the recessive bit.

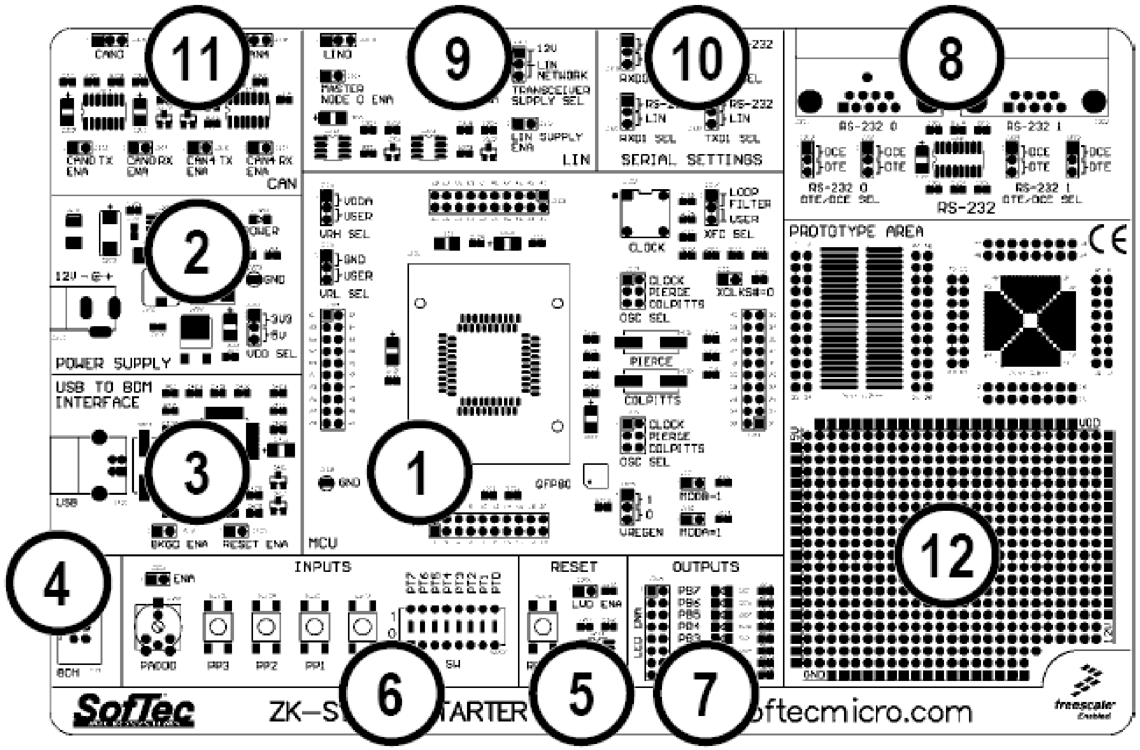

Examples
(1) Wire links
Recessive = potential recovery
Dominant = conductor grounded
+ V
(2) Optical medium
Recessive = light absent
Dominant = light present
(3) Radio medium
Recessive = carrier absent
Dominant = carrier present

The ZK-S12-B Starter Kit



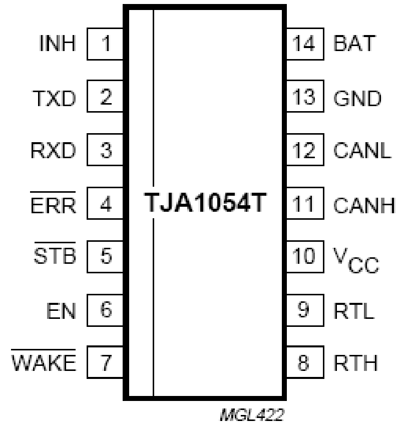
=

11. The “CAN” section contains two fault-tolerant (up to 125 Kbaud) CAN transceivers. The TX and RX signals of CAN nodes CAN0 and CAN4 can be disconnected (by removing the respective **“CANx TX ENA”** and **“CANx RX ENA”** jumpers) from the microcontroller’s respective pins. Two 3x1 male header connectors are provided to interface to an external CAN bus.

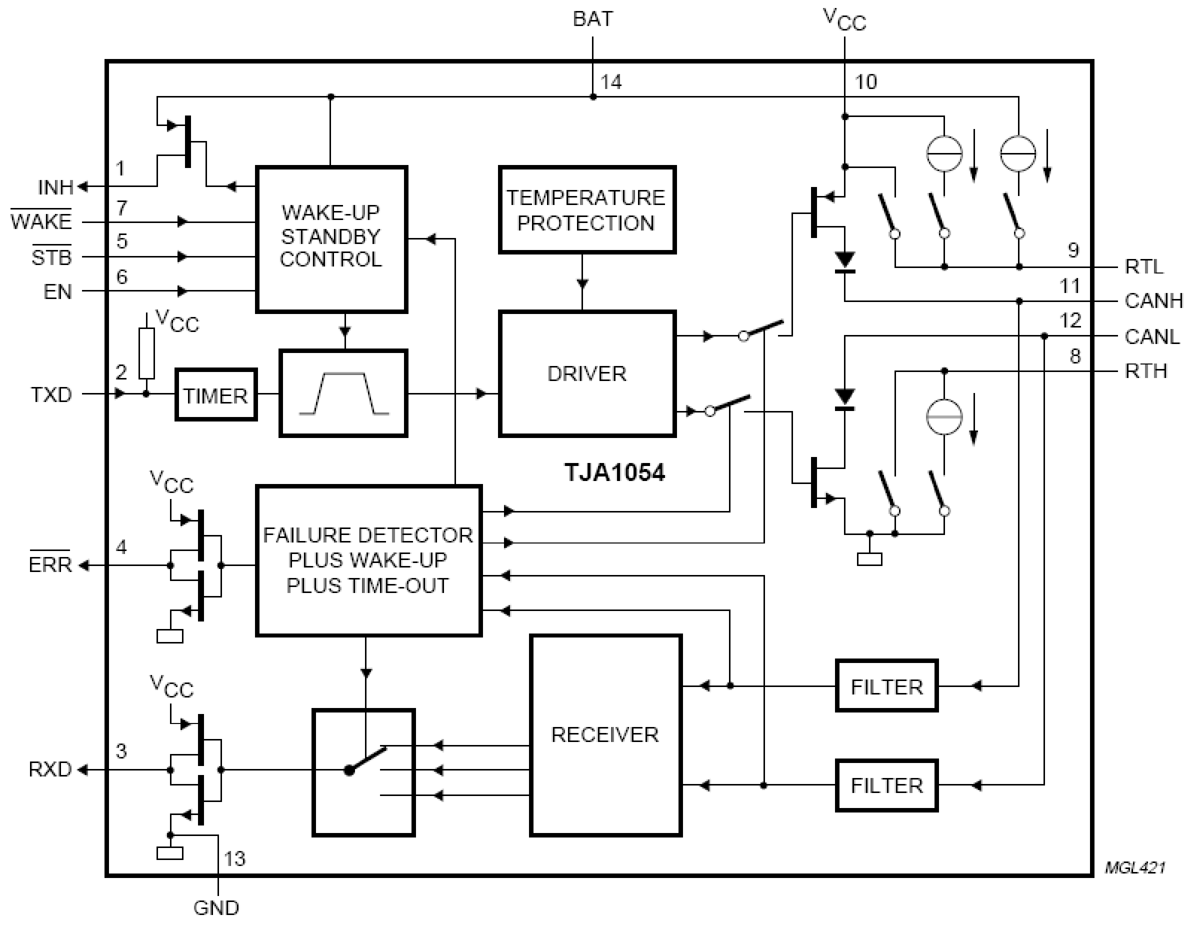


Fault-tolerant CAN transceiver

- Philips Double Wire TJA1054 CAN Transceiver

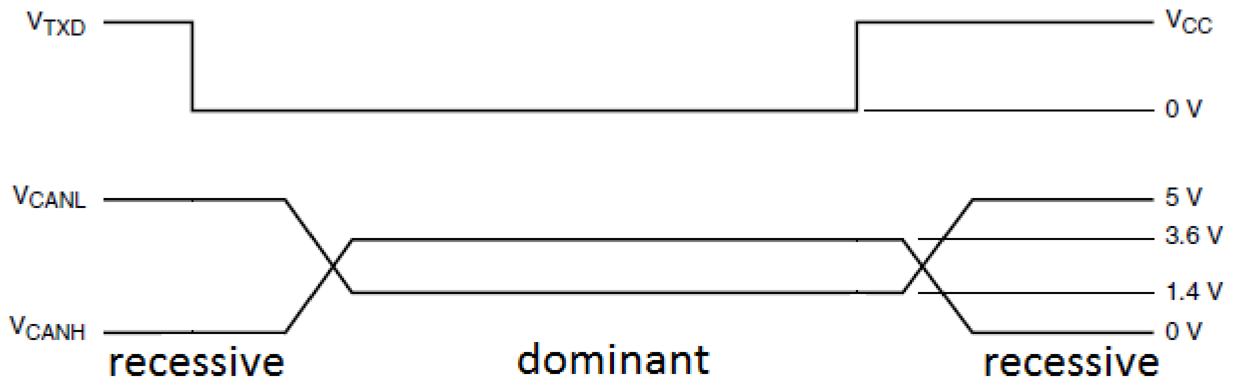


=



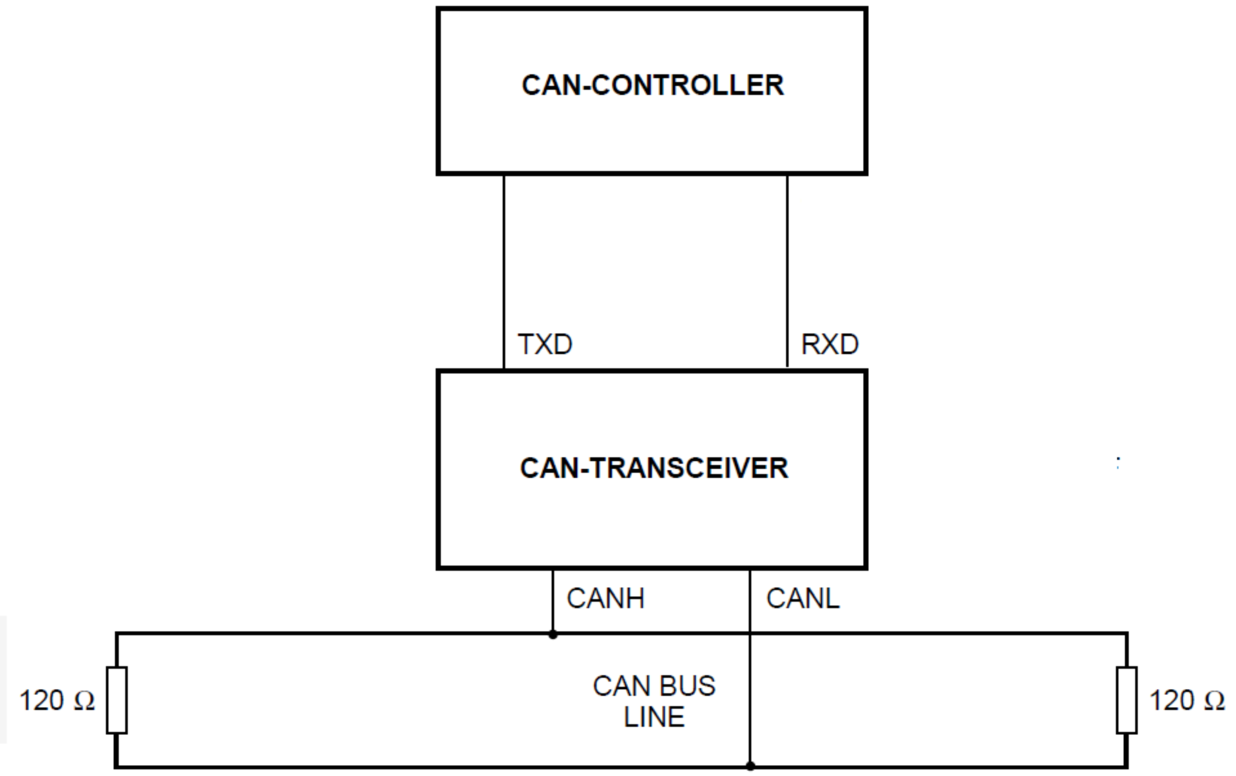
Fault tolerant transceivers specifications

- Low speed, data rates up to 125 kbit/s and uo to 32 nodes
- Detect and handle the following bus error conditions:
  - Interruption on CAN-H
  - Interruption on CAN-L
  - Short circuit of CAN-H with VCC
  - Short circuit of CAN-H with GND
  - Short circuit of CAN-L with VCC
  - Short circuit of CAN-L with GND



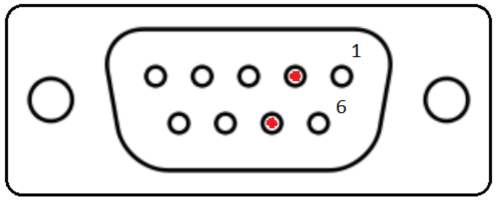
Bus wiring

- 120 ohm termination resistor (not needed for TJA1054)



CAN Connector according to CIA recommendation

- Can-in-Automation (CIA) DS-102 – CAN physical layer for industrial applications
- 9 pin D-SUB connector



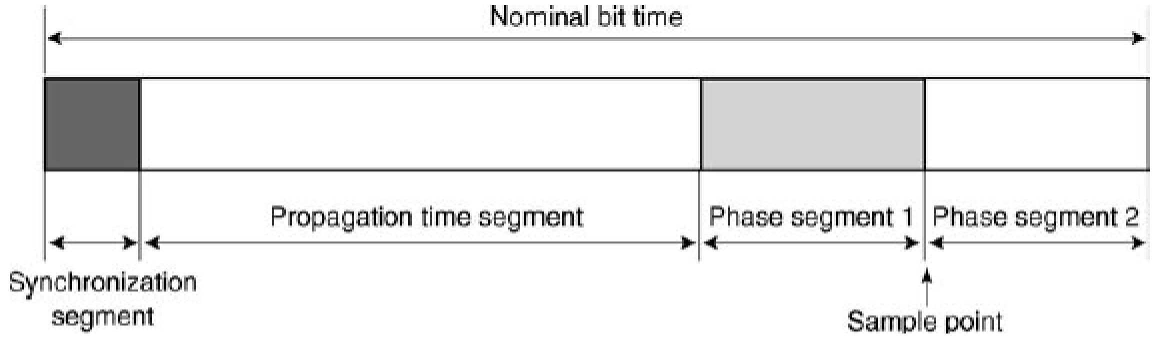
Pin	Signal	Description
1	-	Reserved
2	CAN_L	CAN_L bus line (dominant low)
3	CAN_GND	CAN Ground
4	-	Reserved
5	(CAN_SHLD)	Optional CAN Shield
6	(GND)	Optional CAN Ground
7	CAN_H	CAN_H bus line (dominant high)
8	-	Reserved (error line)
9	(CAN_V+)	Optional CAN external positive supply (dedicated for supply of transceiver and optocouplers, if galvanic isolation of the bus nodes applies)

Bit Time and Nominal Bit Time

- Bits are encoded with NRZ – main issue: lack of synchronization
- Bit Time = the period of time for which the bit is present on the bus
- Nominal Bit Time = the theoretical period of time for which the bit is present on the bus

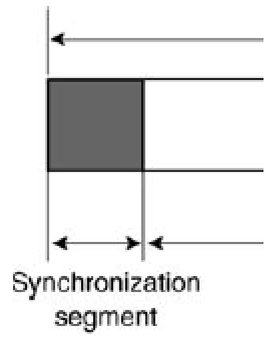
$$\text{NominalBitTime} = \frac{1}{\text{NominalBitRate}}$$

- Nominal bit time is divided in 4 parts



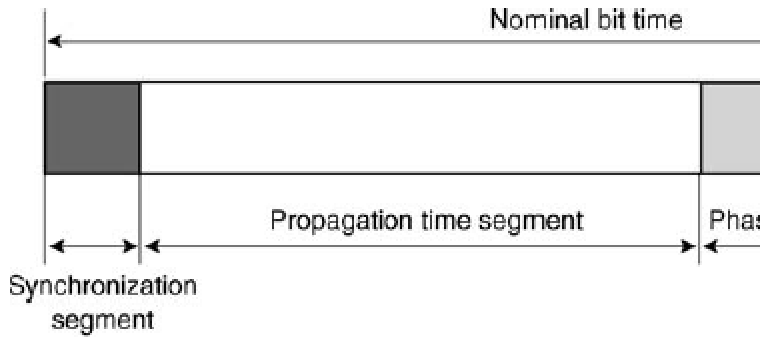
Synchronization segment

- Used to synchronize all nodes, the leading edge of an incoming bit should appear within this segment



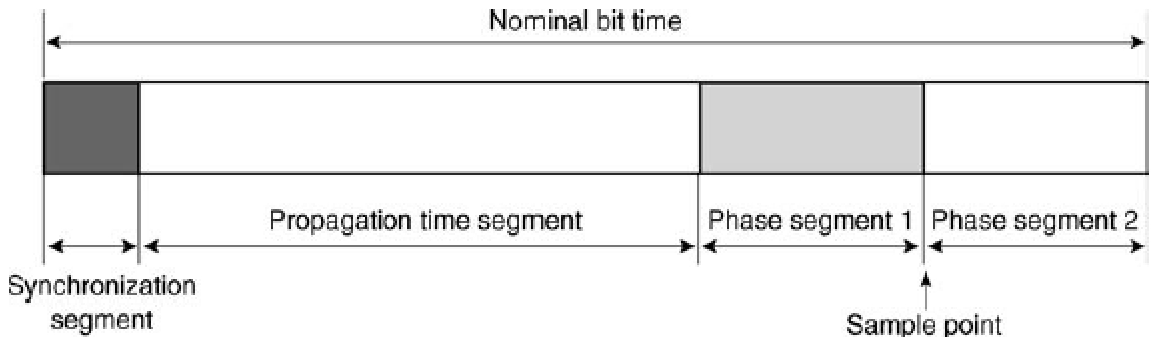
Propagation Time Segment

- Used to compensate the delays on the network (twice the transmission plus propagation time)



Phase Buffer Segment 1 and 2

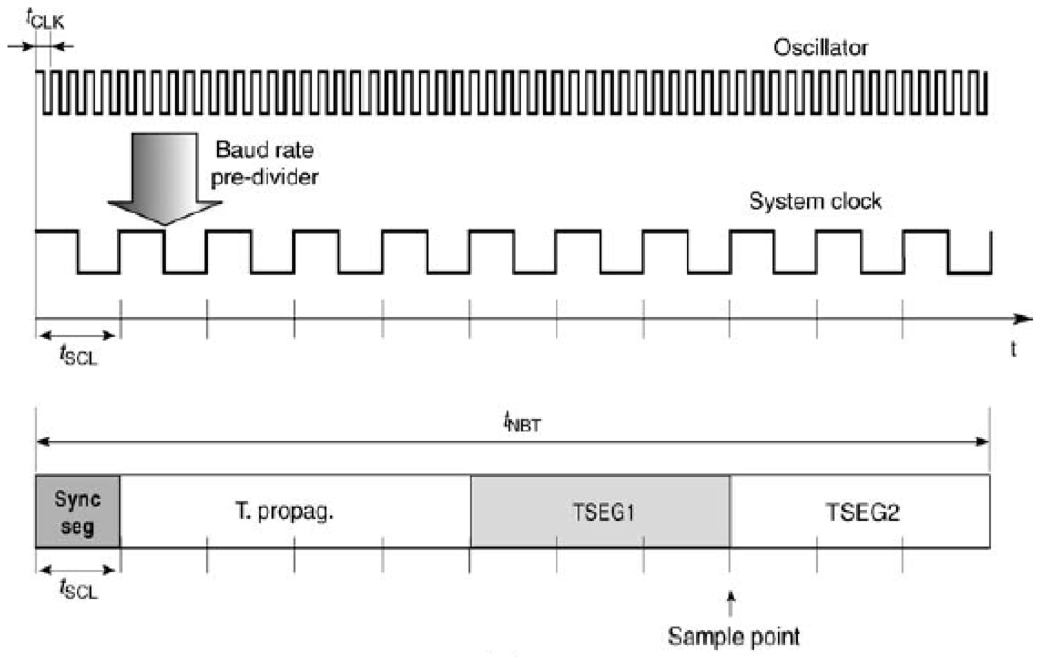
- Used to compensate for edge phase errors (mostly due to small frequency variations in the oscilators)
- Designed to be extended or shortened by resynchronization procedure
- Sample Point = the point at which the bus level is read (located at the boundary between TSGE1 and TSEG2), two relevant issues:
  - Must be as late as possible - to overcome delays and have maximum confidence on the value
  - Must not be too late – time is also needed to compute the actual value
  - => The Sampling Point must be towards the end of a bit period
- Information Processing Time = the time segment starting with the Sample Point (TSEG 2)
- Multiple sampling is available on some circuits: a but can be sampled more than once (e.g., odd number of time and use majority logic)



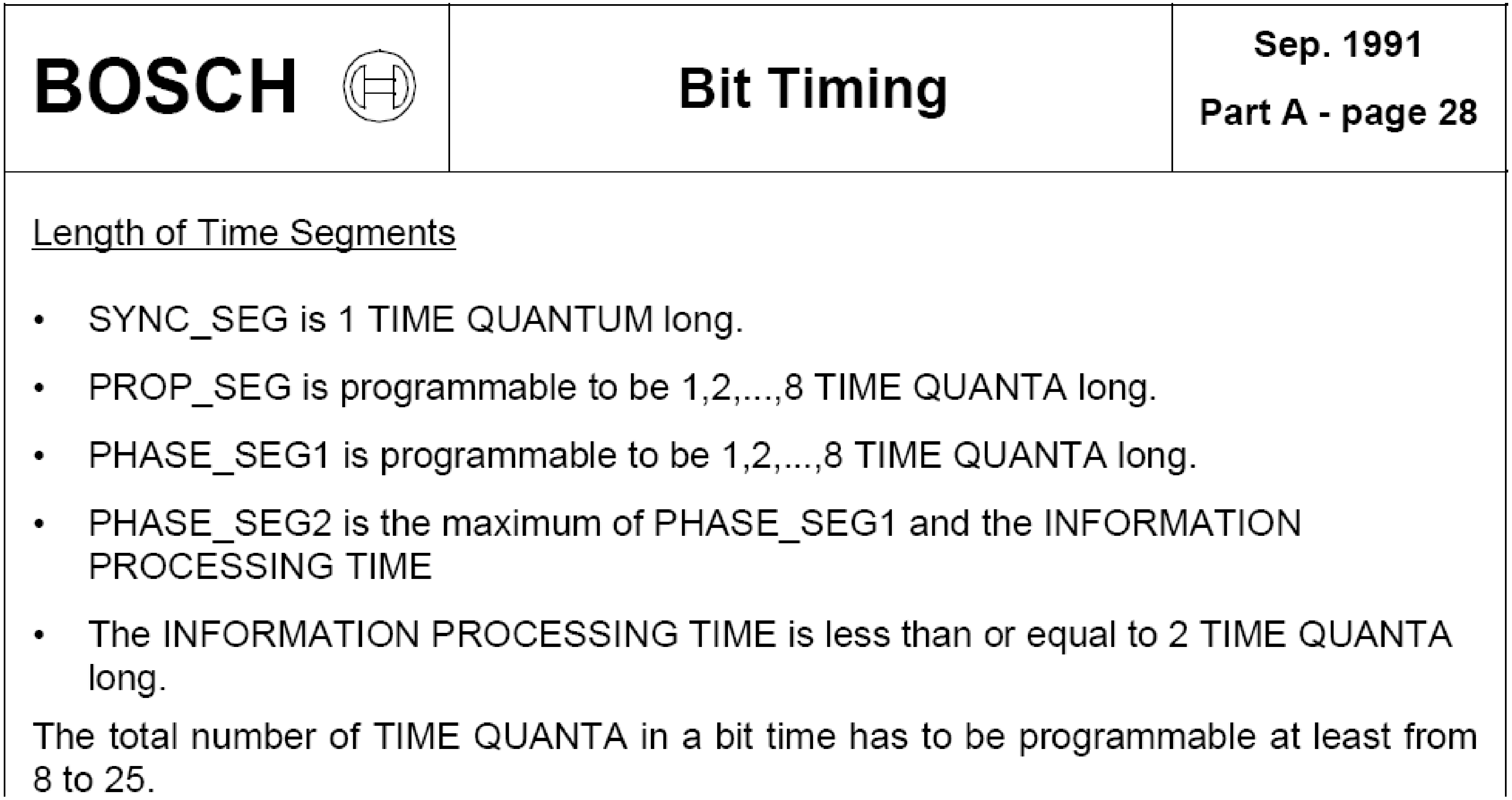
Computing the Nominal Bit Time

- Minimum Time Quantum – generally derived from the station's clock
- Time Quantum of the Bit Time – derived from the minimum time quantum by the use of dividers (division in the range from 1 to 32)

$$\mathrm{TimeQuantum} = m \times \mathrm{MinimumTimeQuantum}$$



Length of Time Segments



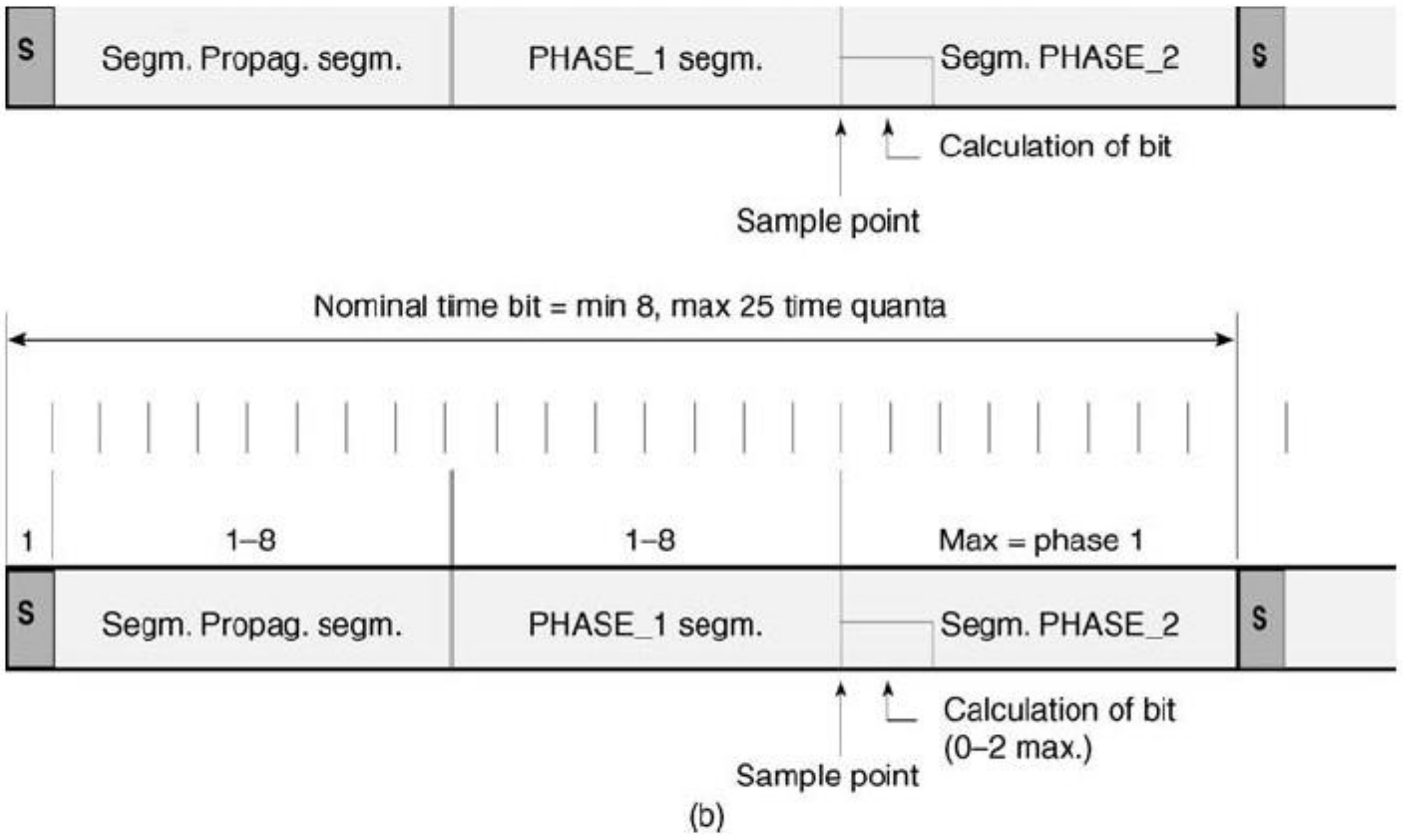
BOSCH	Bit Timing	Sep. 1991 Part A - page 28

Length of Time Segments

- SYNC_SEG is 1 TIME QUANTUM long.
- PROP_SEG is programmable to be 1,2,...,8 TIME QUANTA long.
- PHASE_SEG1 is programmable to be 1,2,...,8 TIME QUANTA long.
- PHASE_SEG2 is the maximum of PHASE_SEG1 and the INFORMATION PROCESSING TIME
- The INFORMATION PROCESSING TIME is less than or equal to 2 TIME QUANTA long.

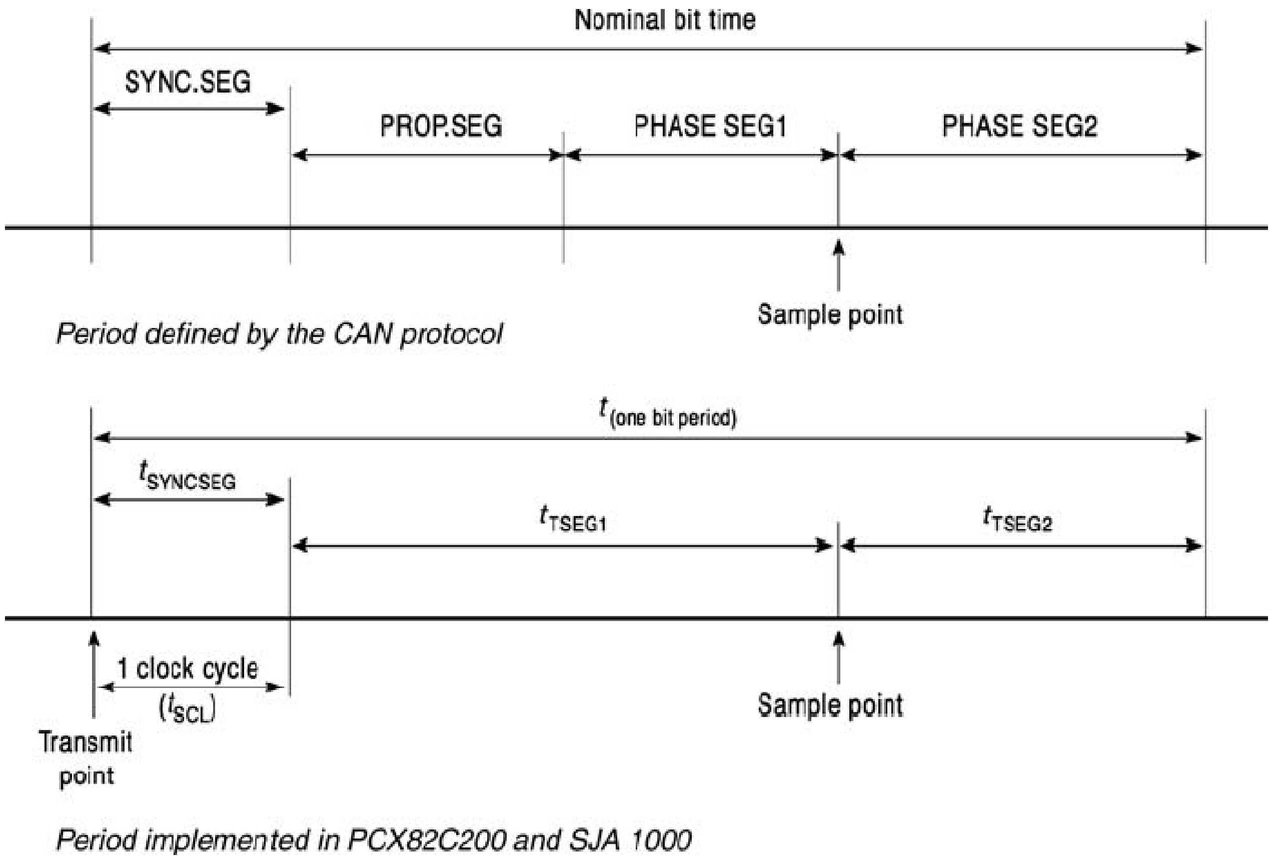
The total number of TIME QUANTA in a bit time has to be programmable at least from 8 to 25.

An example from [Parret, 2007]



And another example …

- Note that on some microcontrollers one can set only TSEG1 and TSEG2, in this case TSEG1 must include the propagation time

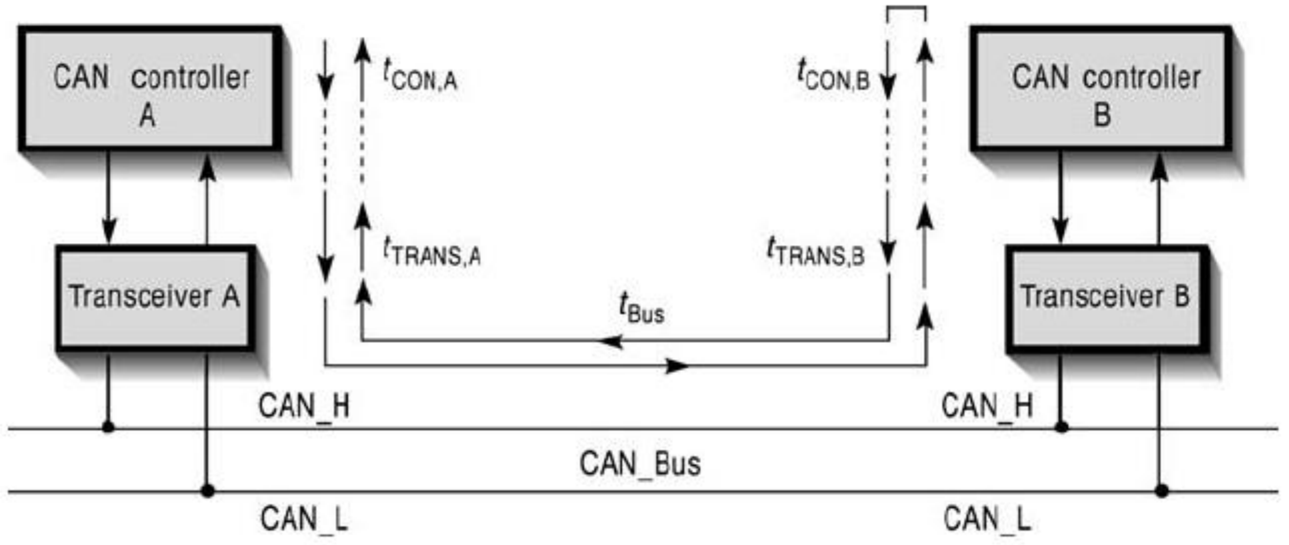


Exact computation for time segments length

- Synchronization Segment – 1 time quanta
- Propagation Speed Segment – depends on the transmission delay caused by:
    - Medium
    - Signal speed on the medium
    - Length of the network
    - Topology
    - Electrical parameters
    - Quality of signal
- TSEG1 and TSEG2 – depend on clock frequencies and their tolerances, fluctuations, temperature (remember that these segments are used to compensate for errors, etc.)

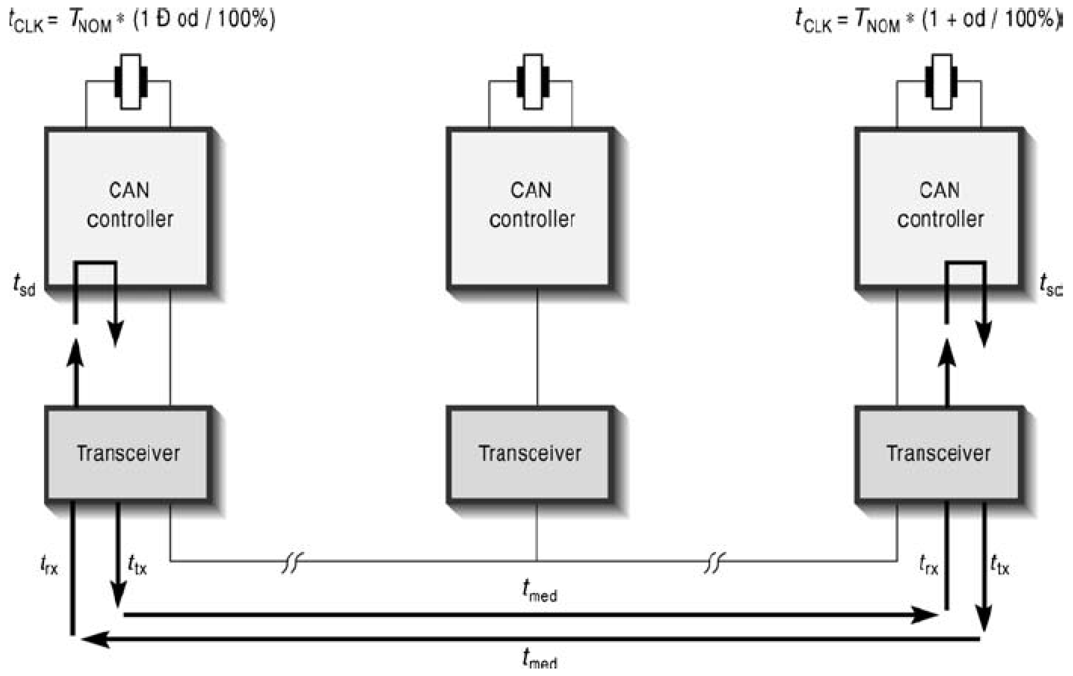
Computing Propagation Time

- The longest distance between 2 CAN nodes determines the propagation delay
- To assure correct arbitration the bit time must be at least twice the delay required to send a bit from A to B. Why ?



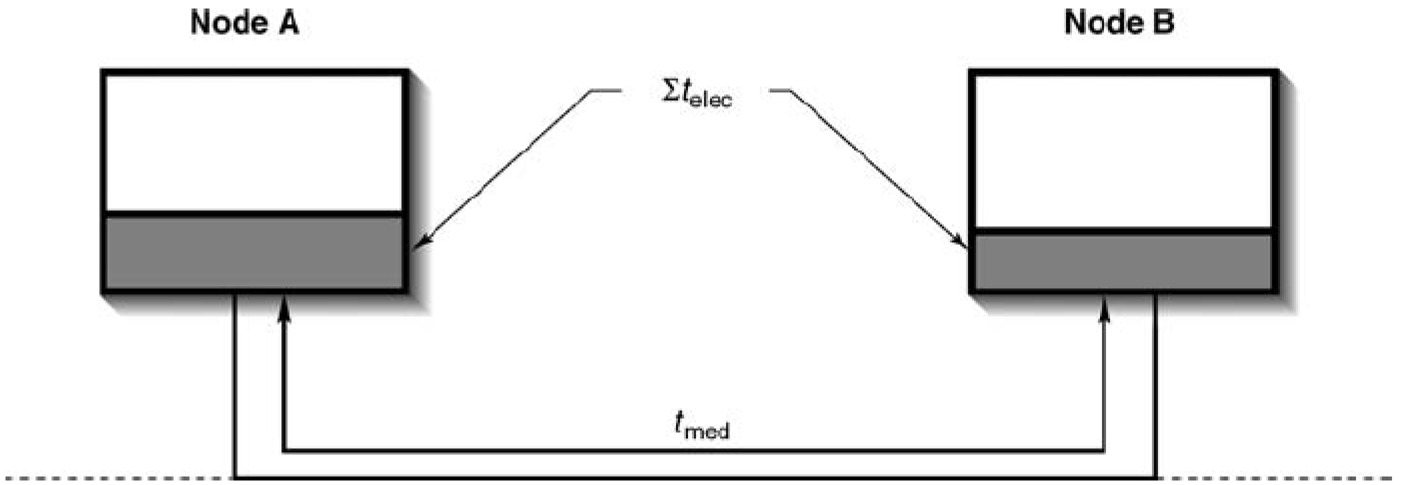
Detailed computation of propagation time

- The following delays must be added:
  - The delay required by the controller to output the signal
  - The delay required by the transceiver to generate the signal
  - The time to transport the signal
  - The time required by the reception transceiver to send the signal to the controller
  - The time required by the controller to process the signal



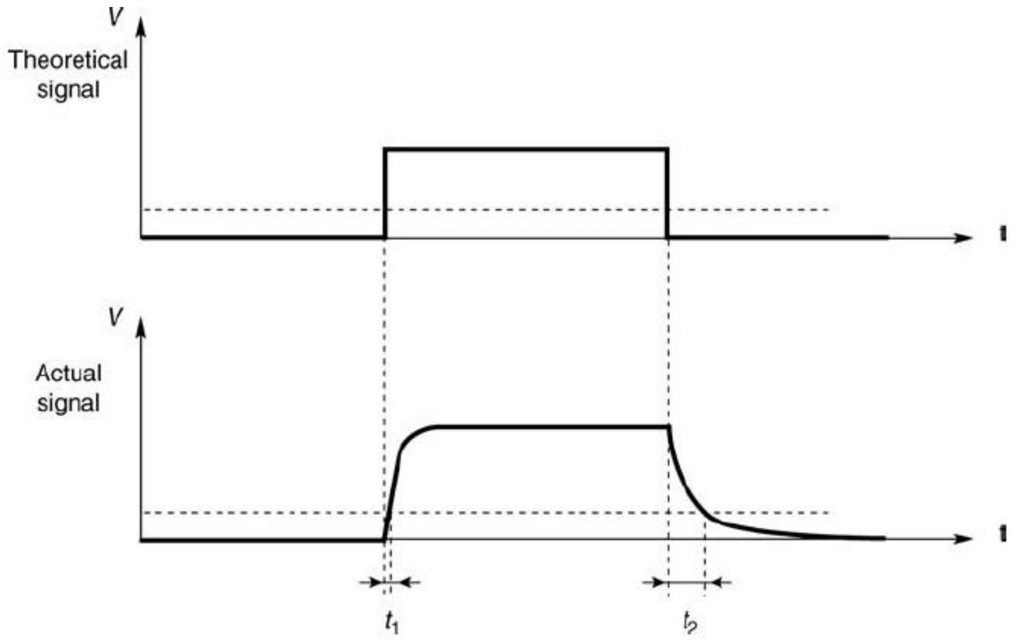
- Can be further synthesized in two types of delays

$$T_{res} = T_{med} + T_{elec}$$



- The delayed caused by the electronic components has the following components:
  - Tsd – the delay caused by the CAN controller
  - Trx, Ttx – the delay caused by the transceiver
  - Tqual – the delay caused by imperfections in the signal

$$T_{elec} = T_{sd} + T_{tx} + T_{rx} + T_{qual}$$



- The delayed caused by the medium is determined by the length of the medium and the propagation speed:

$$T_{med} = \frac{L}{v_{prop}}$$

- Therefore the minimum value of the propagation segment is:

$$\left(seg_{prop}\right)_{\min} \geq 2\left(T_{med} + T_{sd} + T_{tx} + T_{rx} + T_{qual}\right)$$

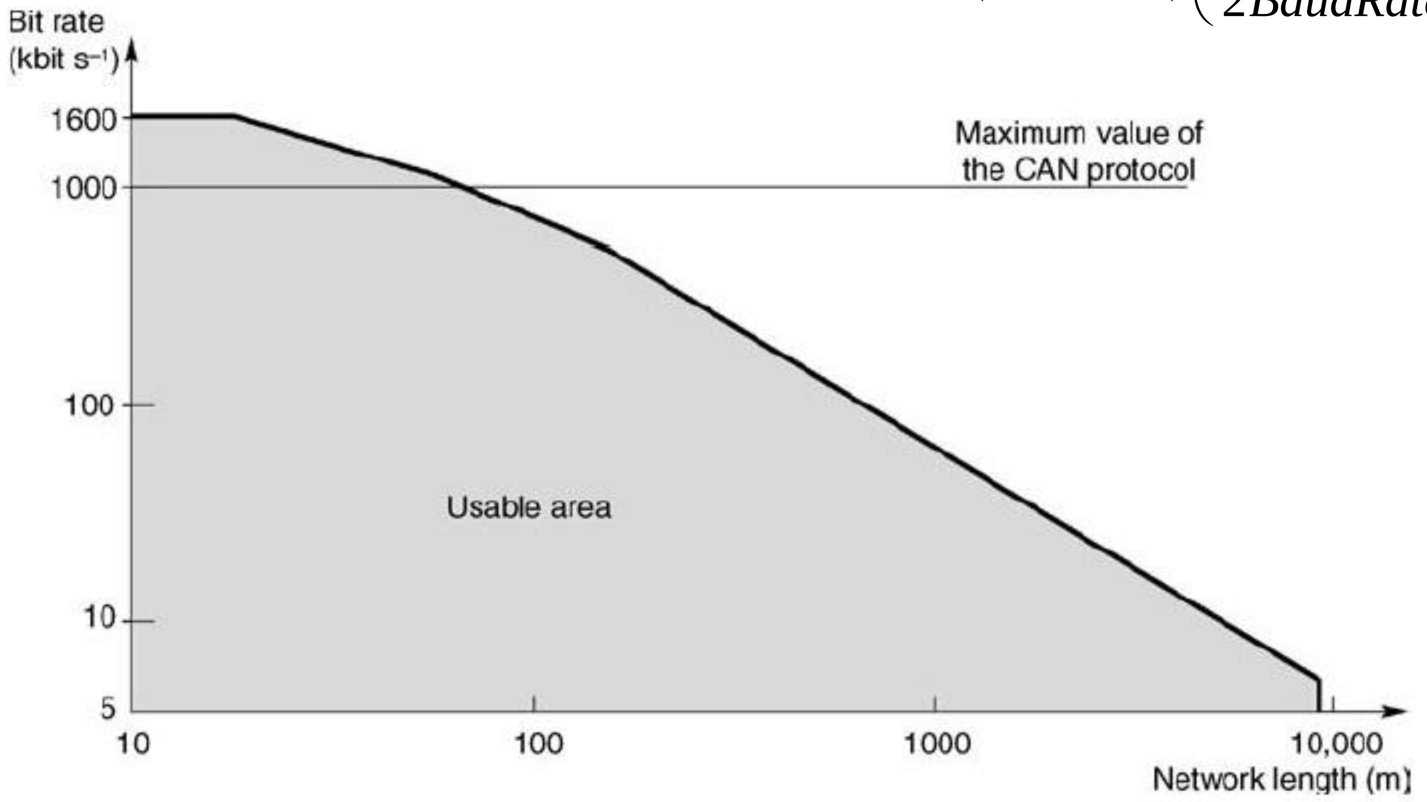
Relations between maximum bit-rate and length of the network

- Question 1: what is the maximum distance at a particular bit-rate on some medium ?
- Question 2: what is the maximum bit-rate on a given network length and medium ?

- We now that the propagation segment must be at least twice the delay, assume that this segment is only x percents of the bit time, then we have:

$$\left(seg_{prop}\right)_{\min} = xT_{bit} \geq 2T_{prop} = 2\left(T_{med} + T_{sd} + T_{tx} + T_{rx} + T_{qual}\right)$$

$$\Rightarrow L \leq \left(0.2mns^{-1}\right)\left(\frac{x}{2BaudRate} - T_{elec}\right)$$



Bit Synchronization

- Hardware Synchronization
  - Internal bit-time started in SYNC_SEG
  - Forces the edge that caused Hard Synchronization to lie in SYNC_SEG

- Resynchronization

Synchronization rules

- According to CAN standard:
  1. Only one synchronization allowed during one bit time
  2. An edge is used only if the value after the edge differs from the previous sample point
  3. HARD Synchronization is performed only after recessive-dominant transition during BUS IDLE
  4. All other recessive-dominant transitions are used for RESYNCHRONIZATION

Resynchronization

- PHASE_SEG1 may be lengthen or PHASE_SEG1 may be shortened
- The amount of lengthening/shortening is given by RESYNCHRONIZATION JUMP WIDTH
- RESYNCHRONIZATION JUMP WIDTH = min(4, PHASE_SEG1)
- The maximum length between 2 transitions that can be used for resynchronization is 29 bit times

Phase Error

- Phase Error of an Edge = difference in time quanta between the position of the edge and SYNC_SEG:
  - $e=0$ lies in SYNC_SEG
  - $e>0$ lies before the SAMPLEPOINT
  - $e<0$ lies after the SAMPLEPOINT

Resynchronization Rules

- Two cases
  i) If PHASE ERROR is smaller (or equal) then RSJW the effect is the same of a HARD Synchronization
  ii) If PHASE ERROR greater then RSJW then
     - If PE > 0 then PHASE_SEG1 lengthened by RSJW
     - If PE < 0 then PHASE_SEG2 is shortened by RSJW

Effects of the oscillator frequencies

- The actual time on each node is (here od = oscillator drifts)

$$T_{\min} = T_{nom}\left(1 - od/100\right) \leq T_{nom} \leq T_{\max} = T_{nom}\left(1 + od/100\right)$$

Estimation of RJW

- If x is the oscillator tolerance then for one bit the synchronization jump must be (explain why 2):

$$RJW = 2 \cdot T_{bit} \cdot \frac{x}{100}$$

- CAN standard specifies that resynchronization must be feasible after 29 bits, therefore we have:

$$RJW = 58 \cdot T_{bit} \cdot \frac{x}{100}$$

- The minimum bit-time is:

$$T_{bit}^{\min} = 1 + segProp + 1tq + RJW_{\max}$$

- Therefore we have

$$RJW_{\max}\left(1 - \frac{58}{100}x\right) = \frac{58}{100}x\left(2t_q + segProp\right)$$

Question

- What is the maximum oscillator tolerance for a node in a CAN network?