CAN

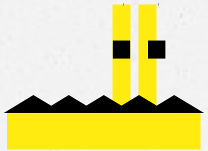
Main Features:

- CAN stands for Controller Area Network
- Serial communication protocol
- Supports real time control
- High level of error handling.
- Multi-Master Multi-Slave communication.
- Message based protocol .

CAN

CAN Application:

- Motor vehicles
- Utility vehicles
- Industrial automation.



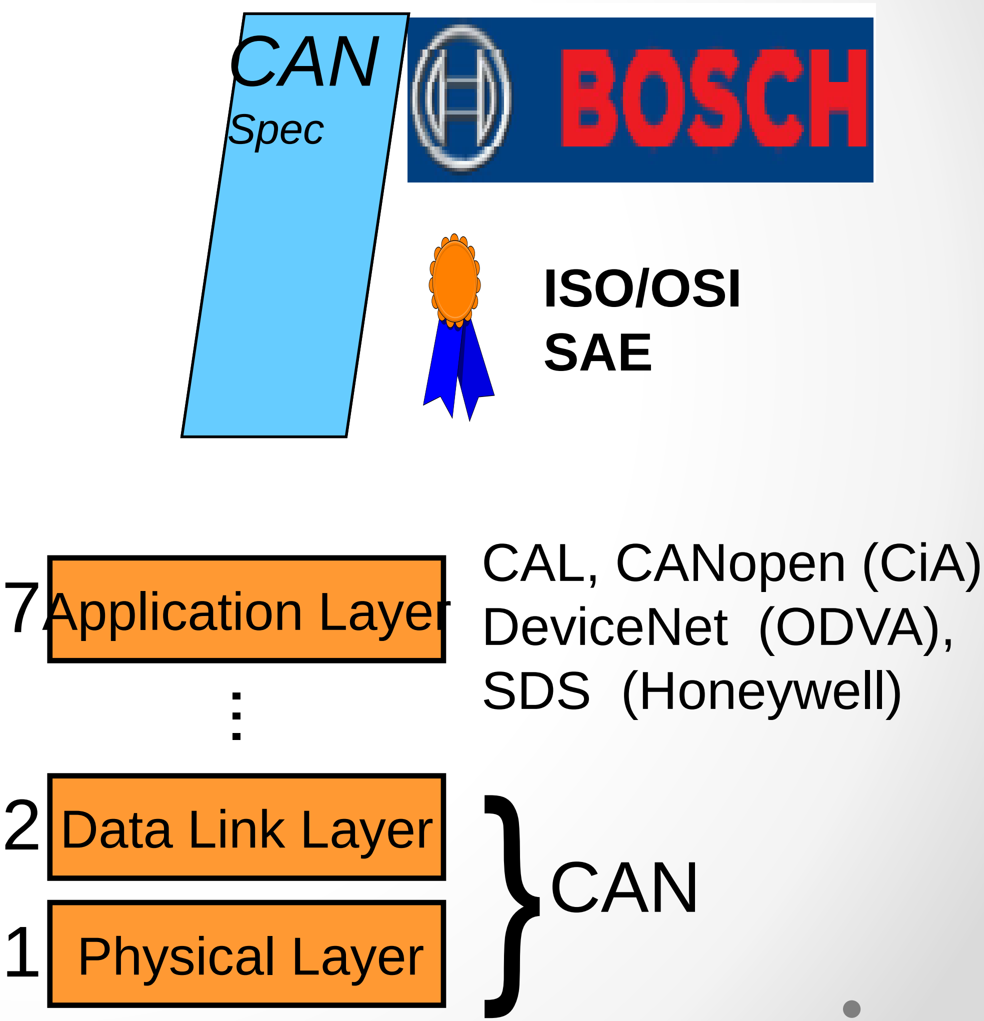
- Other applications for CAN are:

  trains, medical equipment, building automation, household appliances, and office automation.

CAN

CAN History:

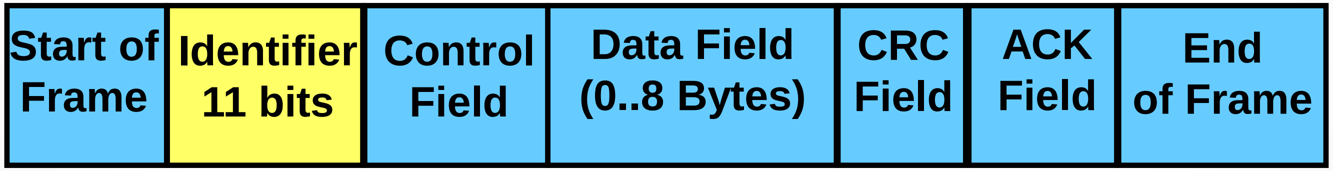
- Initially developed for motor vehicles by Robert Bosch GmbH, Germany, starting 1983, also holding the CAN license.
- CAN is internationally standardized by ISO and SAE in 1986 .



CAN History:

Standard CAN

- The original CAN specifications (Versions 1.0, 1.2 and 2.0A) specify an 11 bit message identifier.
- 16 messages with the lowest priority (2032-2047) are reserved.



Start of Frame	Identifier 11 bits	Control Field	Data Field (0..8 Bytes)	CRC Field	ACK Field	End of Frame

CAN History:

Extended CAN

- CAN specification Version 2.0B updated to meet the SAE J1939 standard .
- 29-bit identifier is made up of the 11-bit identifier ("Base ID") and the 18-bit Extended Identifier ("ID Extension").
- CAN v2.0B still allows message identifier lengths of 11 bits to be used.

Start of Frame	**Identifier 29 bits**	**Control Field**	**Data Field (0..8 Bytes)**	**CRC Field**	**ACK Field**	**End of Frame**

CAN

CAN Specification :

Three Types of CAN Modules :

- 2.0A :

  Considers 29 bit ID as an error

- 2.0B Passive:

  Ignores 29 bit ID.

- 2.0B Active :

  Handles both 11 and 29 bit ID.

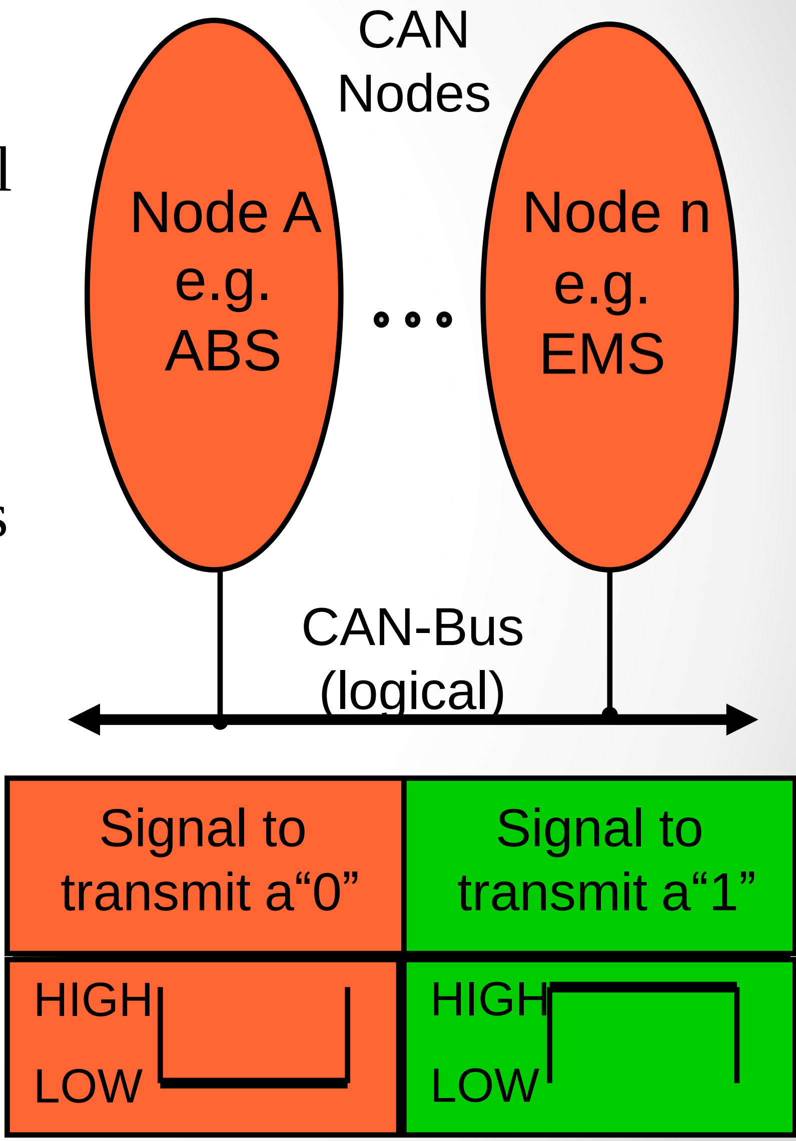
Care when mixing protocol versions !

	Frame with 11 bit ID	Frame with 29 bit ID
V2.0B Active CAN	Tx/Rx OK	Tx/Rx OK
V2.0B Passive CAN	Tx/Rx OK	Tolerated
V2.0A CAN	Tx/Rx OK	Bus ERROR

CAN

CAN properties :

- Multi Master Concept
- Number of nodes not limited by protocol
- No node addressing, Message identifier specifies contents & priority
- Easy connection/ disconnection of nodes
- Broadcast capability
- Sophisticated error-detection and error handling mechanisms
- Non-Return-to-Zero bit coding



CAN Nodes

Node A e.g. ABS

…

Node n e.g. EMS

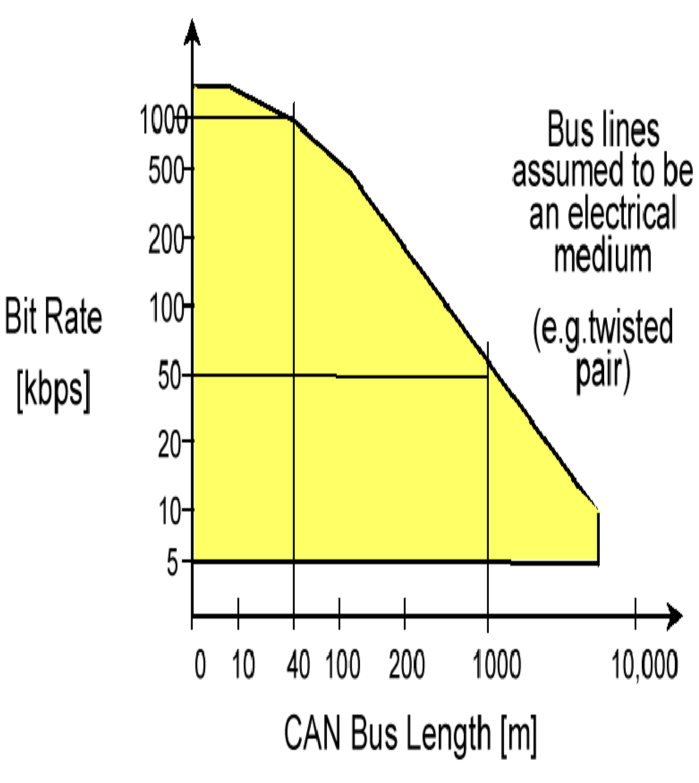
CAN-Bus (logical)

Signal to transmit a“0”	Signal to transmit a“1”
HIGH LOW	HIGH LOW

CAN

CAN properties :

- High data transfer rate of 1000 kbps at a maximum bus length of 40 meters
- Twisted wire pair is the bus medium used for CAN.
- Multiple Access/Collision Detection with Non-Destructive Arbitration.



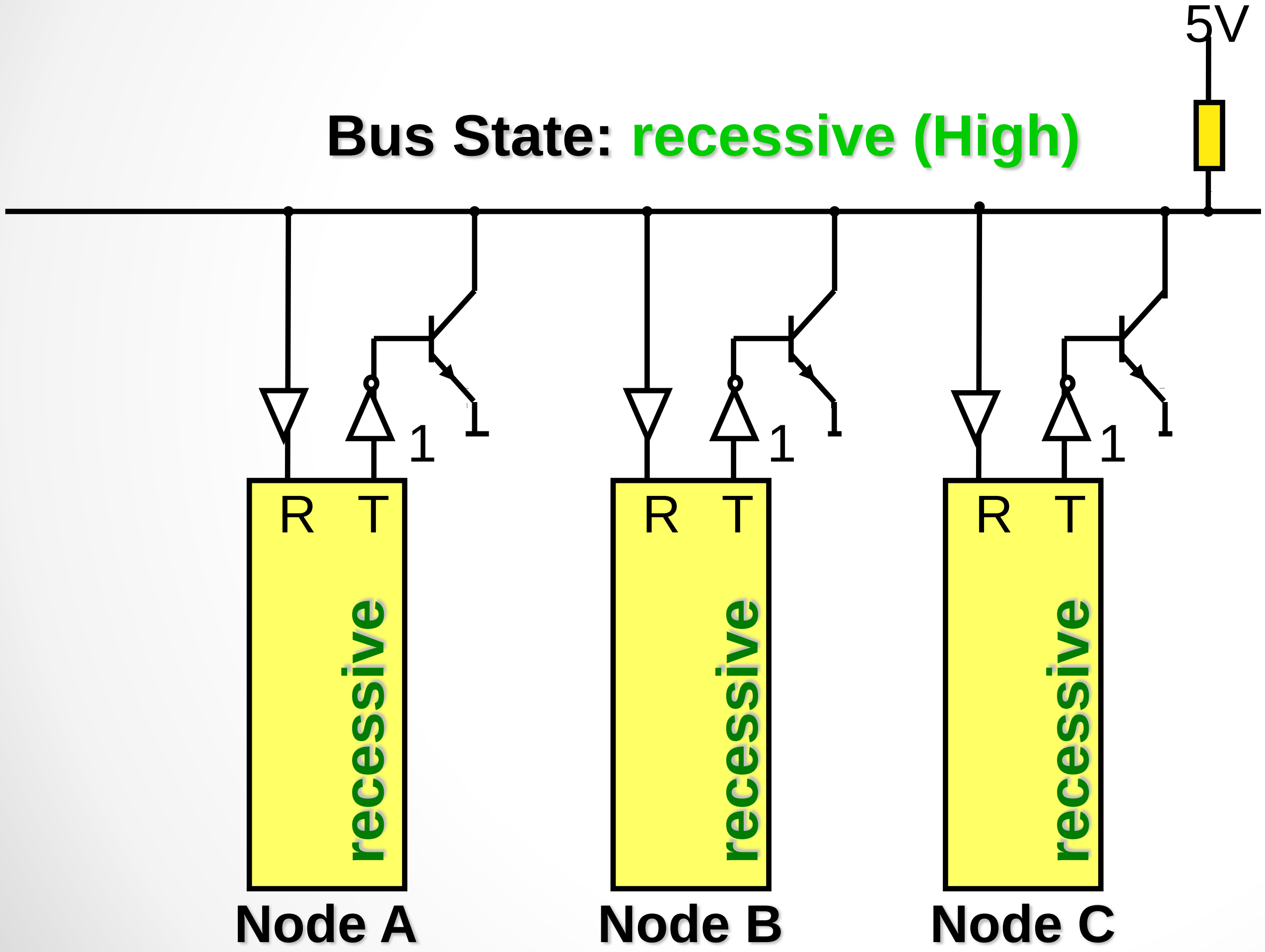
CAN

CAN properties :

- The bus can have one of two complementary logical:
  - Dominant(‘0’)
  - Recessive(‘1’).
- During simultaneous transmission of ’dominant’ and ’recessive ’bits, the resulting bus value will be ’dominant’.

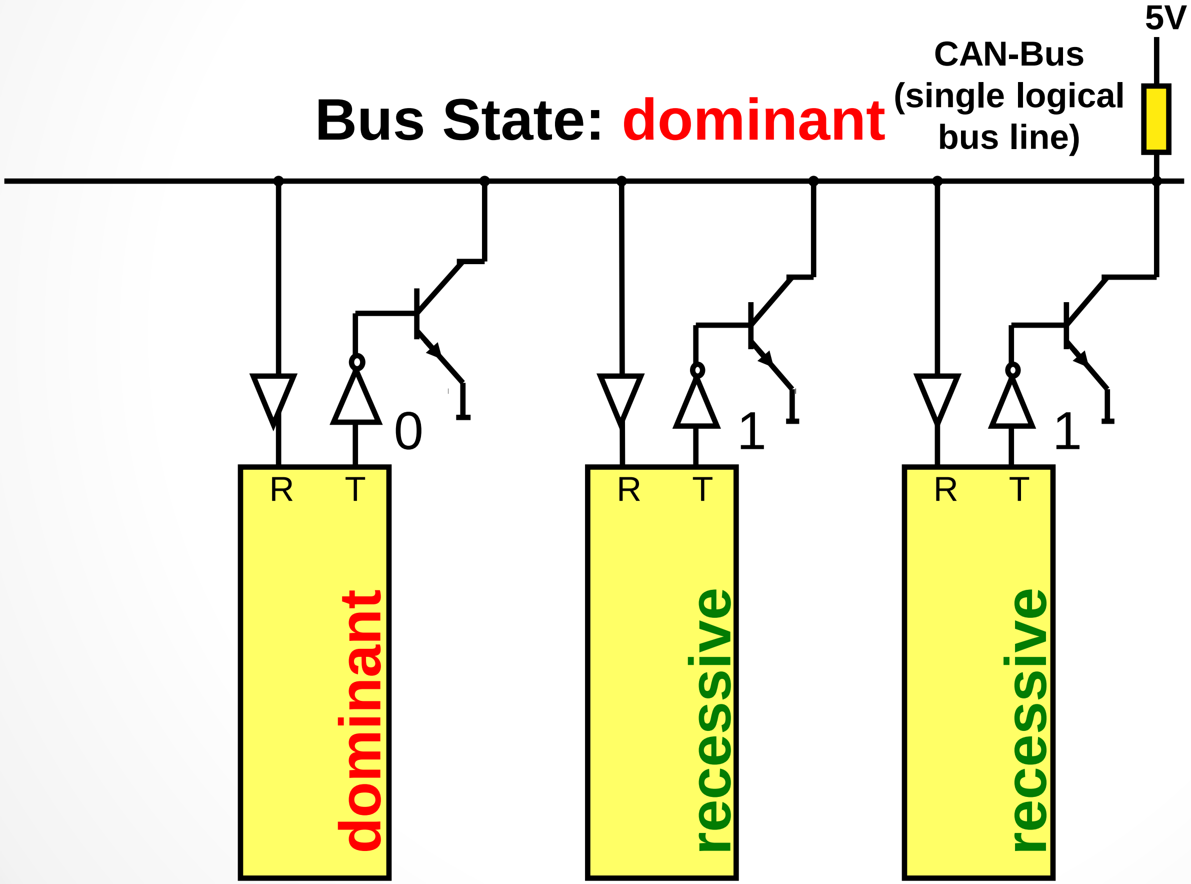
A	B	C	BUS
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

CAN properties :


Bus State: recessive (High)

A	B	C	BUS
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

CAN properties :



Bus State: dominant

A	B	C	BUS
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

CAN properties :

Bus Arbitration :

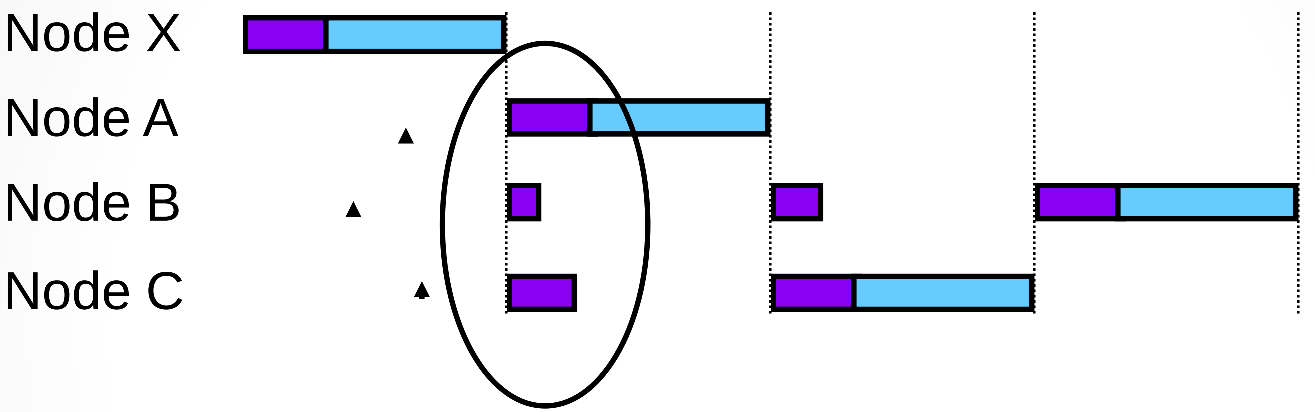
- CAN protocol handles bus accesses according to :

  "Carrier Sense Multiple Access with Arbitration on Message Priority"

  (CSMA/CD w/AMP).
- It avoids collisions of transmit messages simultaneously
- Arbitration priories the massages using the identifier .
- It isn't permitted for different nodes to send messages with the same identifier as arbitration could fail leading to collisions and errors.

CAN properties :

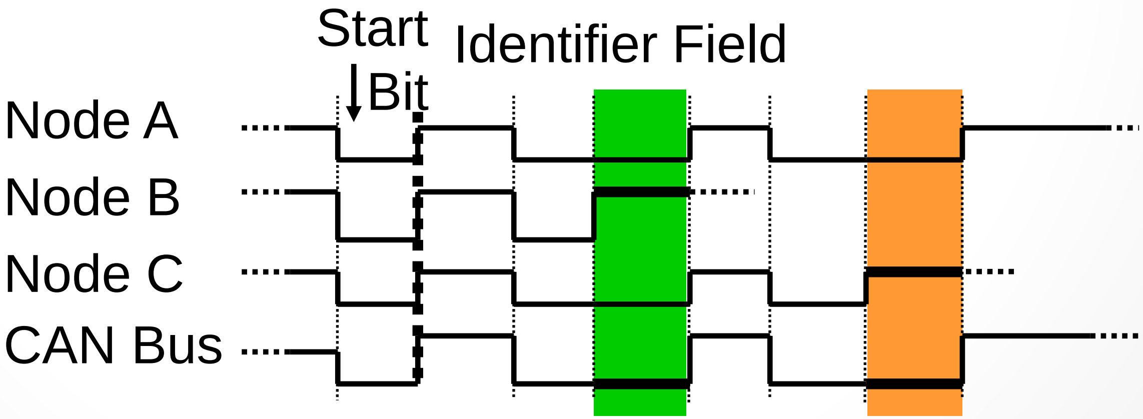
Bus Arbitration :



🟪 Arbitration phase

🟦 Rest of Frame

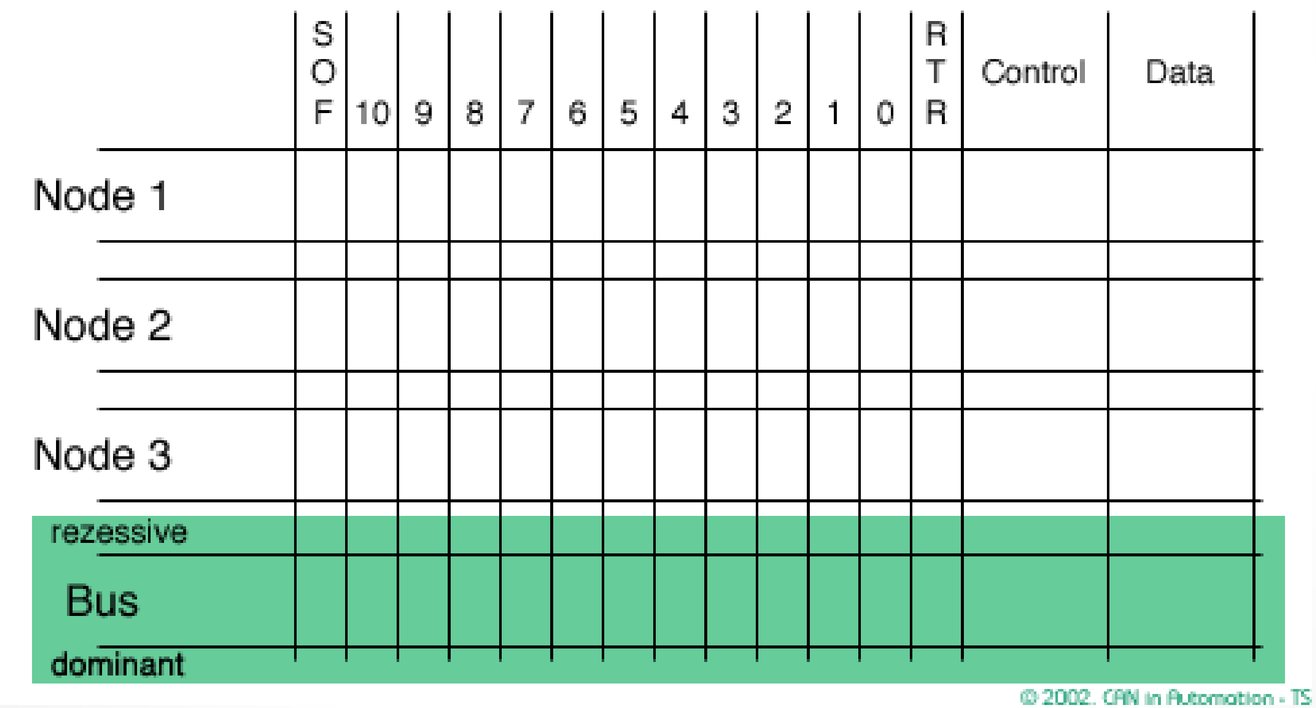
▲ Transmit Request



Node B loses Arbitration Node C loses Arbitration

CAN properties :

Bus Arbitration :



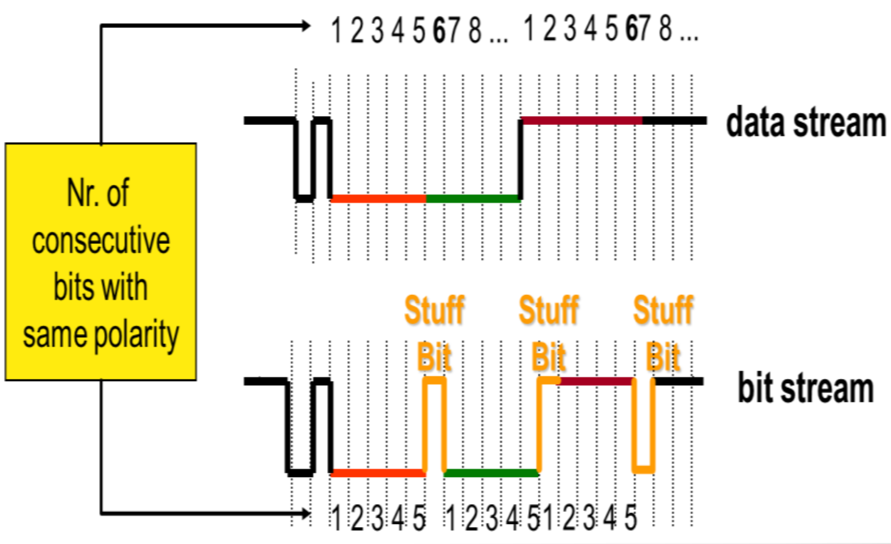
	SOF	10	9	8	7	6	5	4	3	2	1	0	RTR	Control	Data
Node 1															
Node 2															
Node 3															
rezessive Bus dominant															

© 2002, CAN in Automation - TS

CAN properties :

Bit Stuffing :

- NRZ if there are no edges, receivers lose track of bits.
- Periodic edges allow receiver to resynchronize to sender clock
- Bit stuffing is used to ensure synchronization of all bus nodes.



CAN properties :

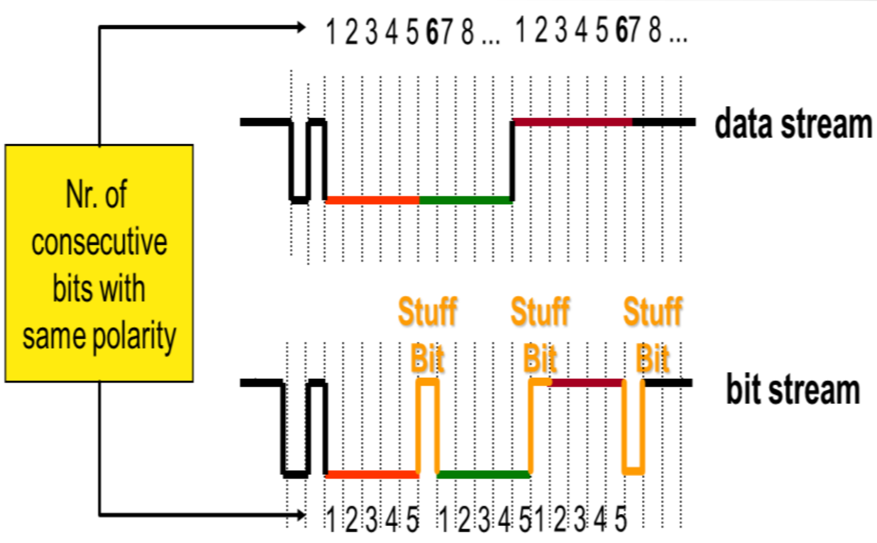
Bit Stuffing :

- When five consecutive bits of the same polarity transmitted:

  Transmitter insert one additional bit of the opposite polarity.

- The receiver checks the number of bits with the same polarity

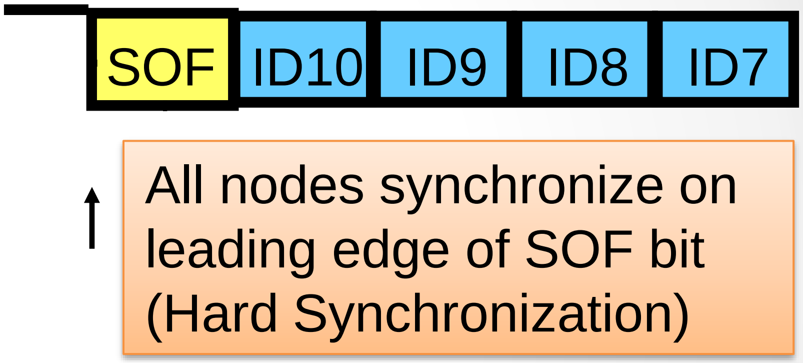
  Removes the stuff bits again from the bit stream. "destuffing".



CAN properties :

Synchronization:

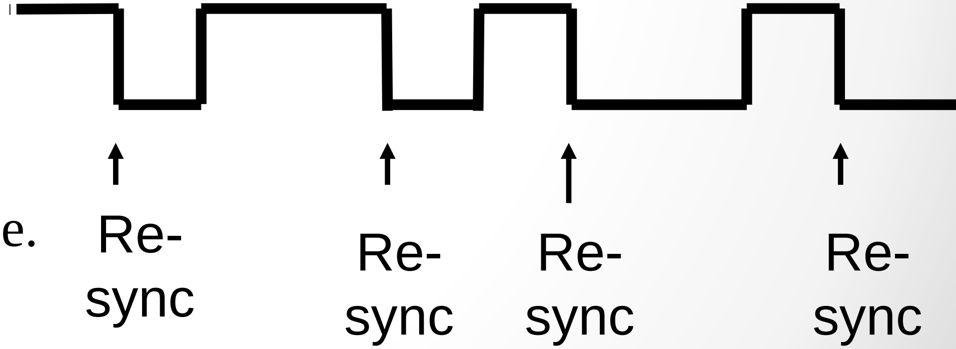
- Hard Synchronization:
  - CAN synchronize the message transfer for each node with the leading edge of SOF bit.



SOF | ID10 | ID9 | ID8 | ID7

All nodes synchronize on leading edge of SOF bit (Hard Synchronization)

- Re-synchronize
  - With each recessive to dominant edge.



Re-sync

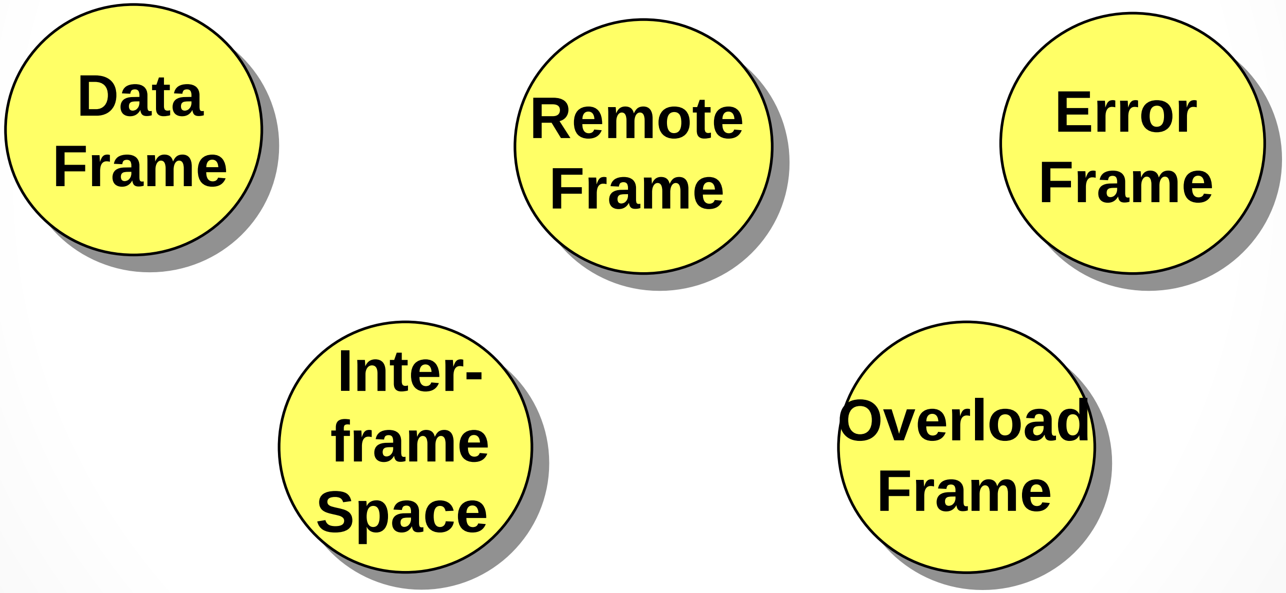
Re-sync

Re-sync

Re-sync

CAN Frames

Standard CAN Frames



Data Frame

Remote Frame

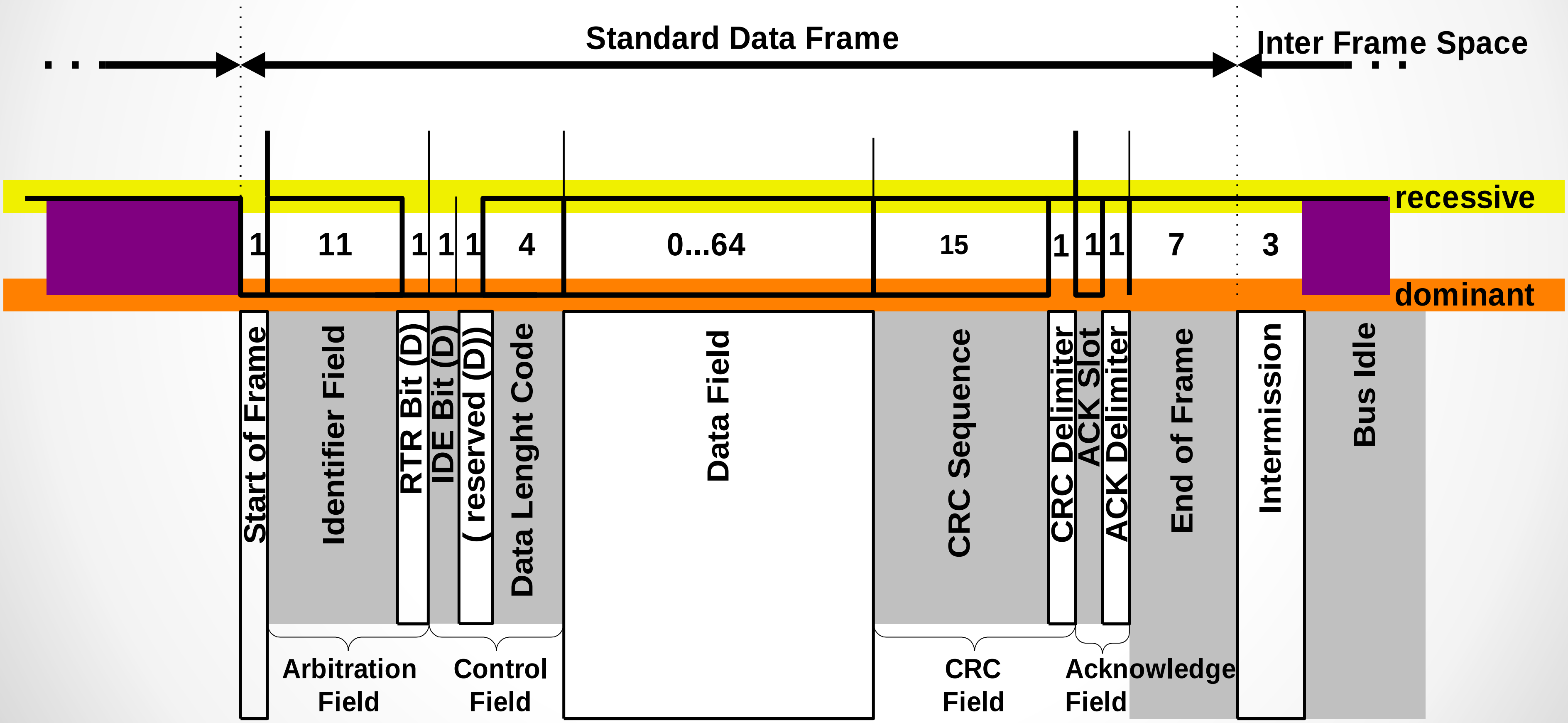
Error Frame

Inter-frame Space

Overload Frame

CAN Frames

Standard Data Frame



RTR: Remote Transmission Request

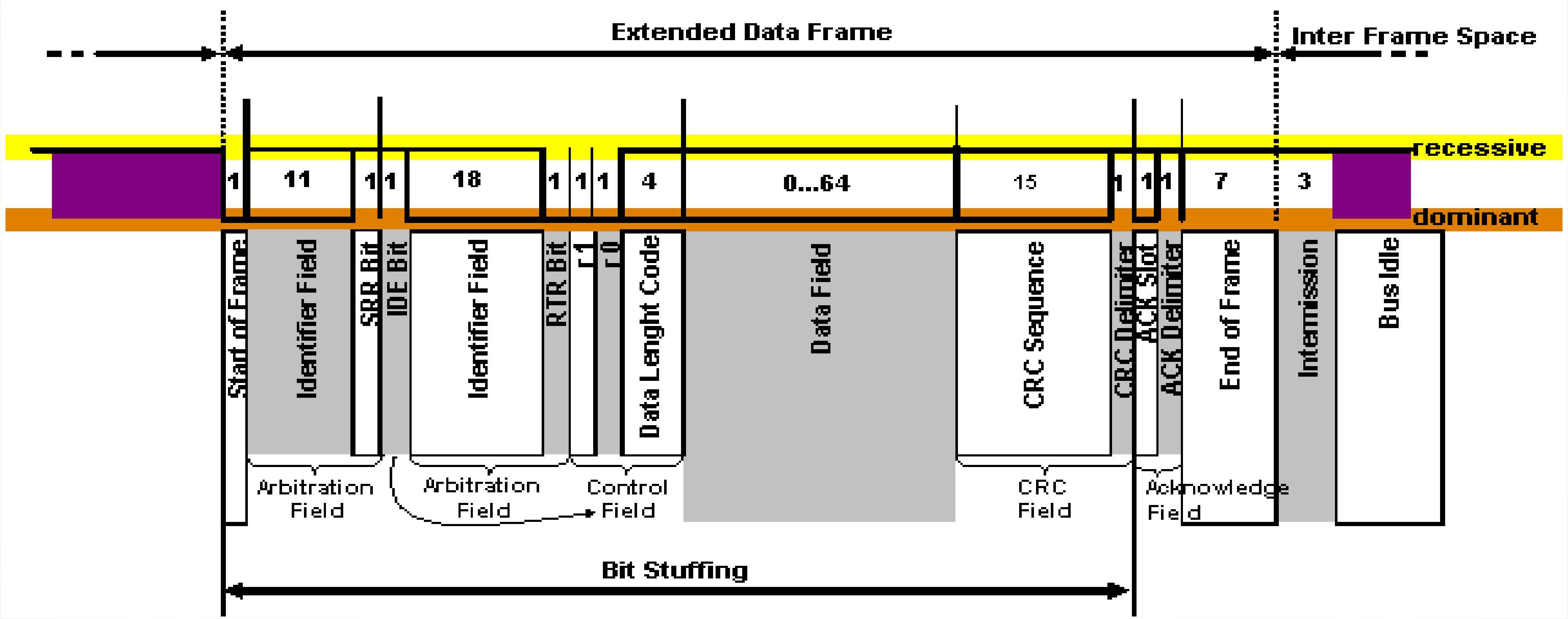
IDE :Identifier Extension

CRC :Cyclic Redundancy

CAN

CAN Frames

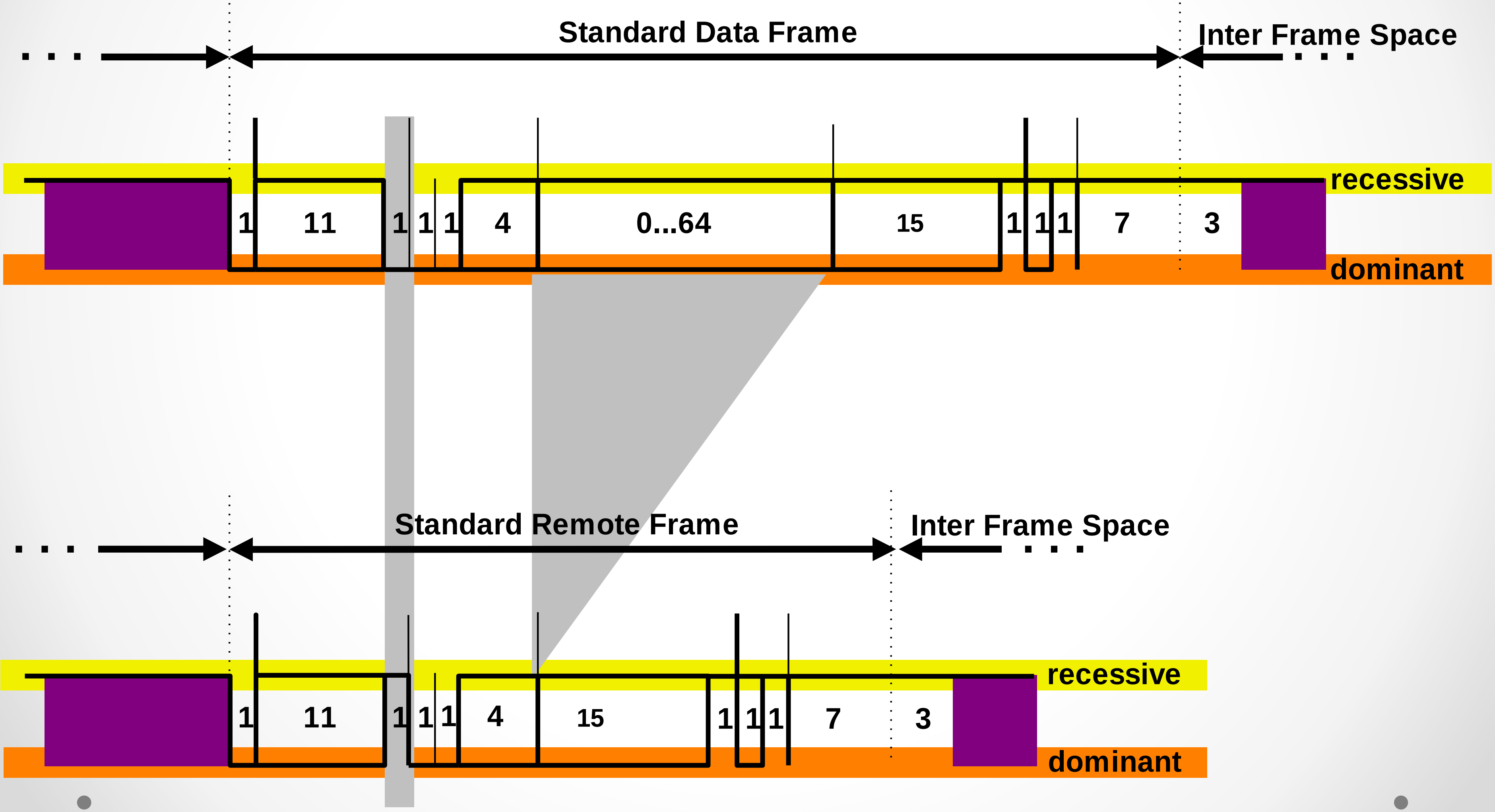
Standard Data Frame



- SRR (Substitute Remote Request) :replace for the RTR .
- r0 and r1: reserve for future applications.

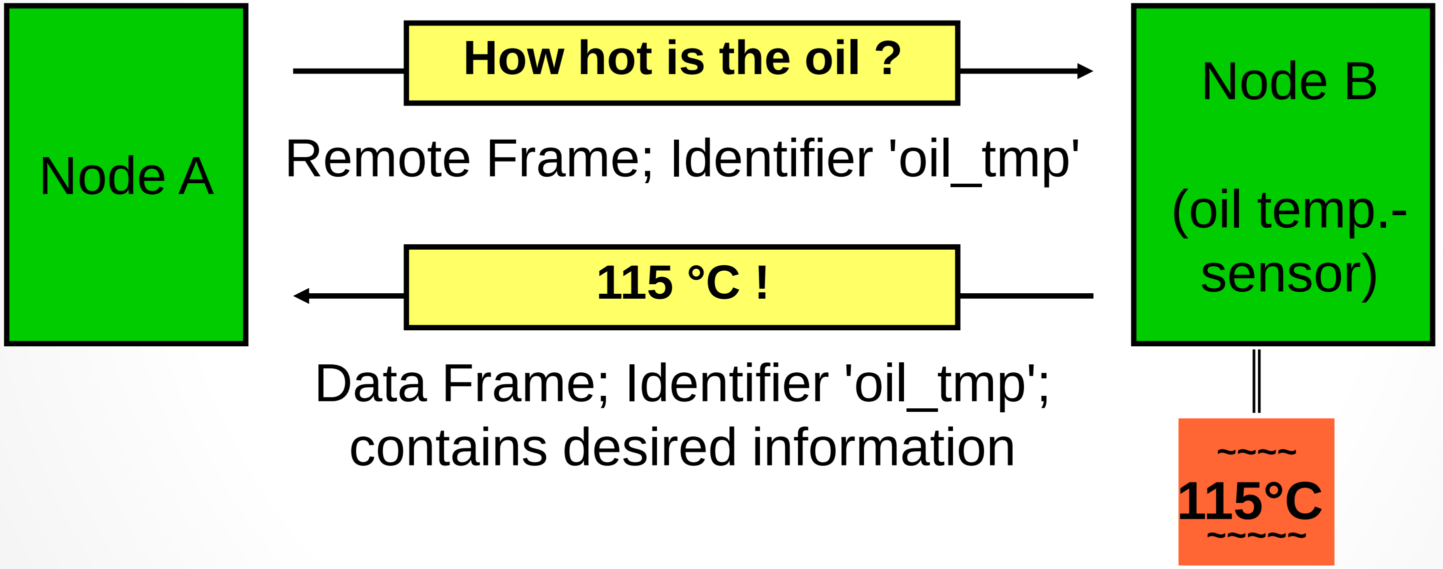
CAN Frames

Remote Frame



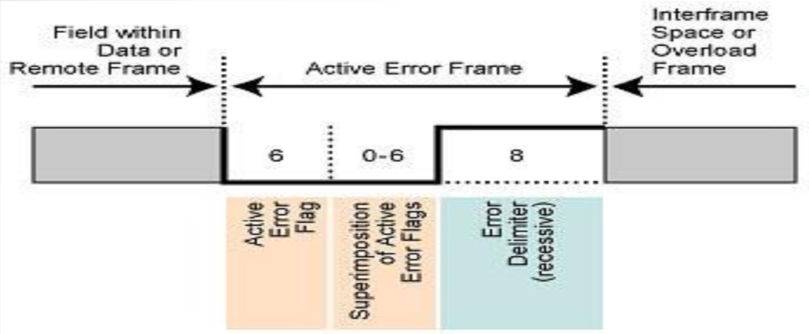
CAN Frames

Data and Remote Frame scenario

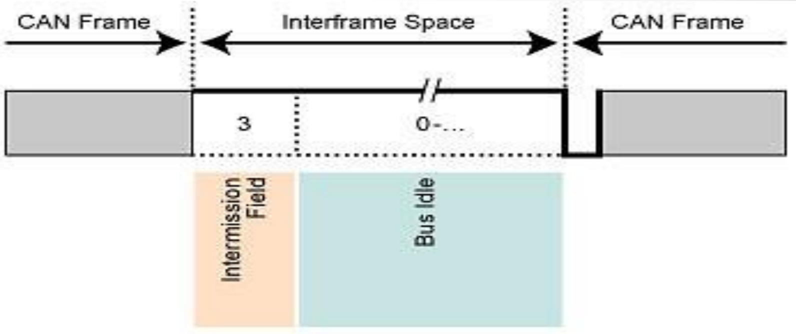

Node A
How hot is the oil ?
Remote Frame; Identifier 'oil_tmp'
Node B
(oil temp.-sensor)
115 °C !
Data Frame; Identifier 'oil_tmp';
contains desired information
~~~~~
115°C
~~~~~~

CAN Frames

Error Frame

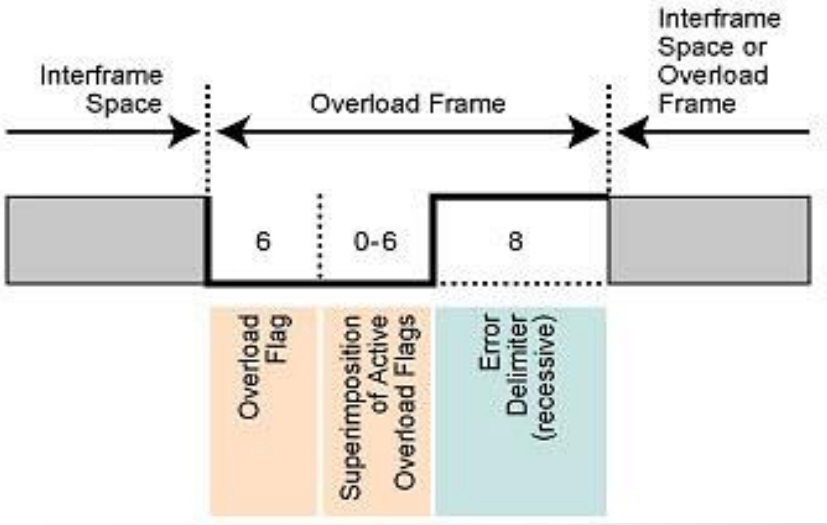


Inter-Frame space



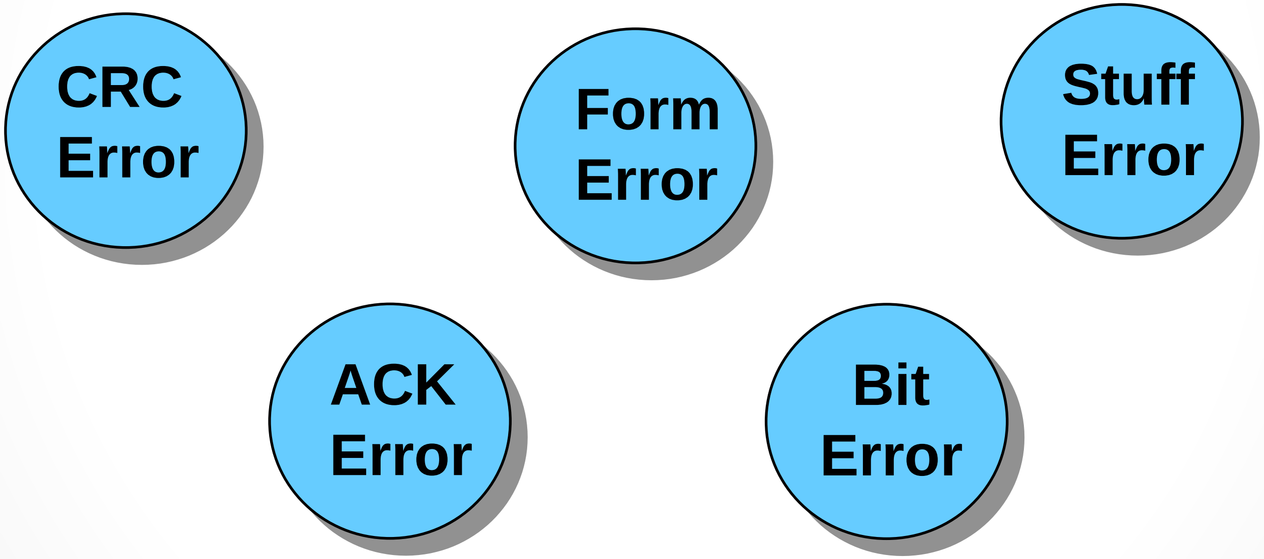
Overload Frame

- Generated due to internal conditions of a receiver, which requires a delay of the next DATA or REMOTE frame.



CAN Error Detection:

- There are 5 error detection mechanisms:



CRC Error

Form Error

Stuff Error

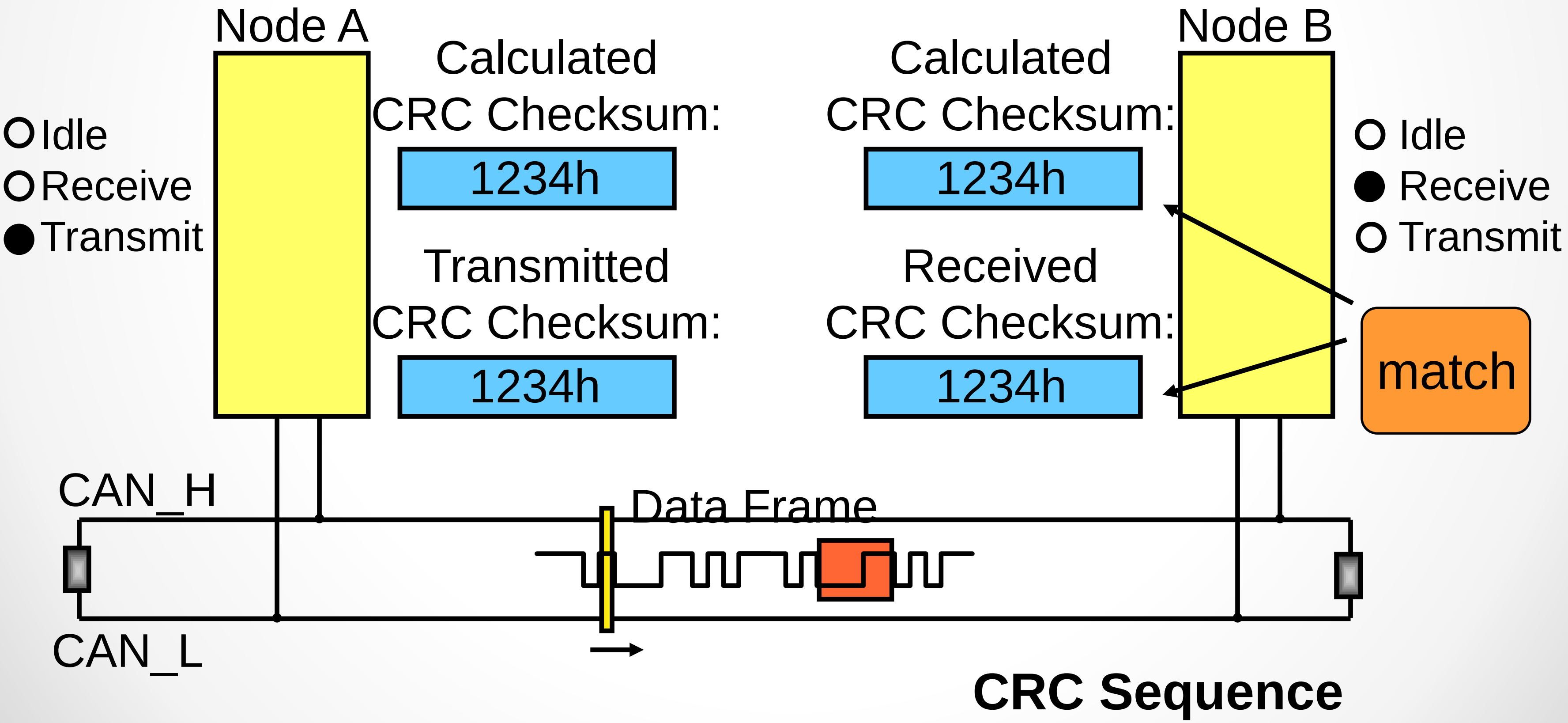
ACK Error

Bit Error

CAN Error Detection:

CRC

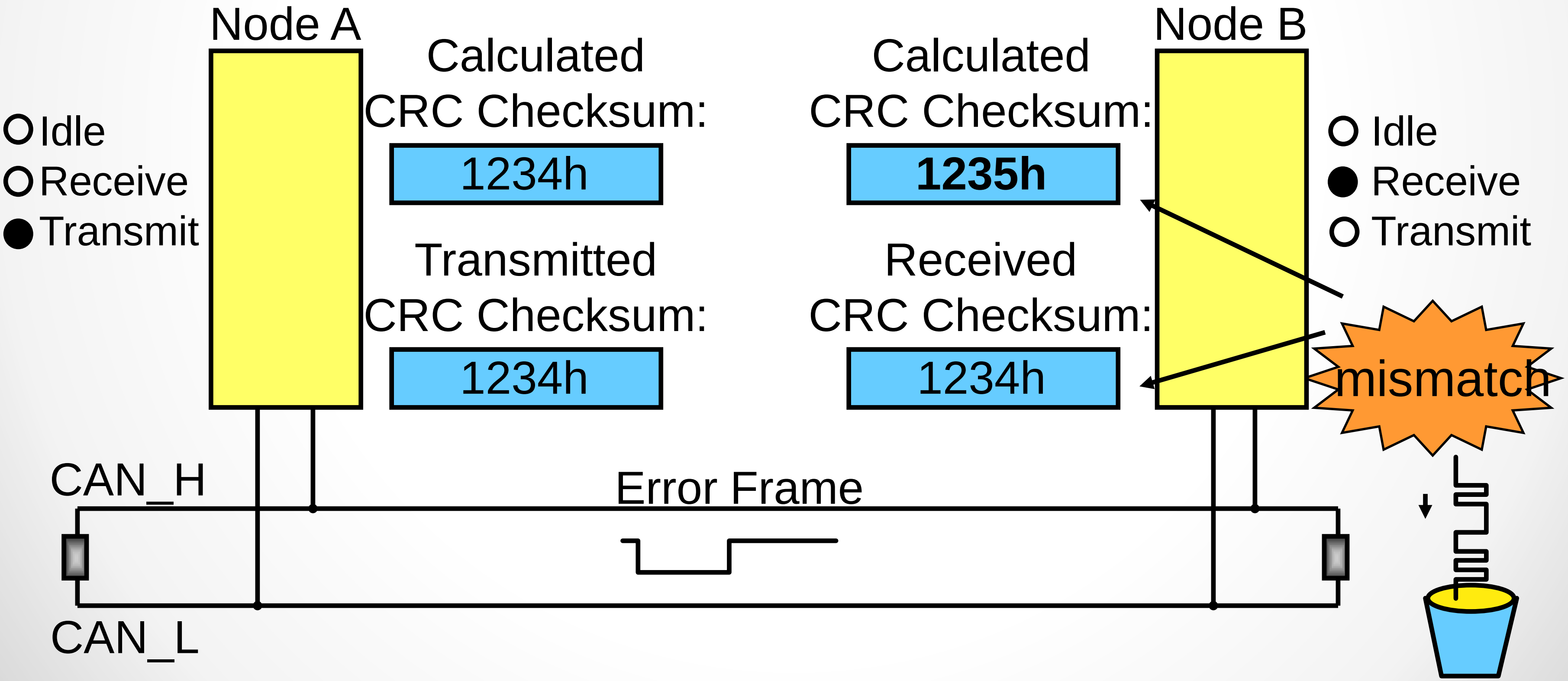
- Calculated and received CRC checksum must match.



CAN Error Detection:

CRC

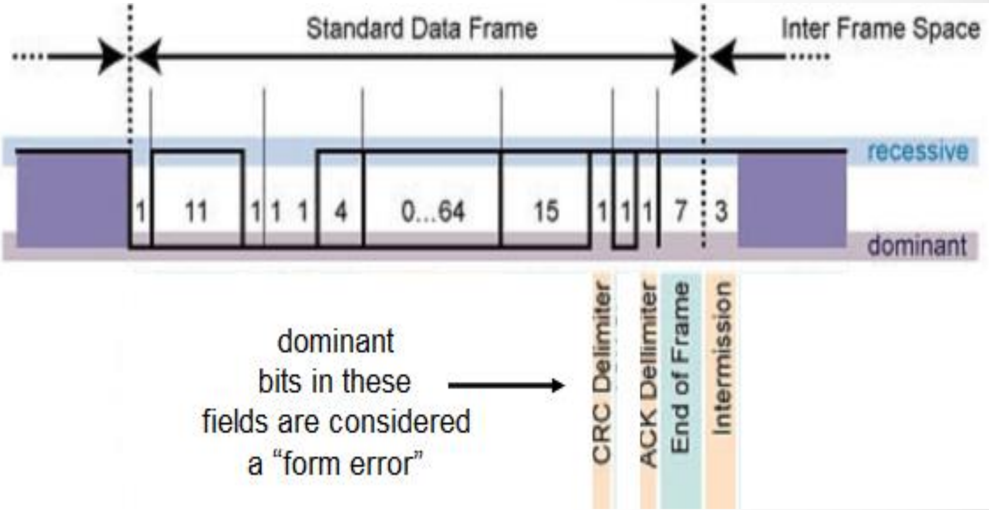
- Otherwise Frame was not received correctly (CRC Error)



CAN Error Detection:

Frame Check

- No Dominant Bits allowed in:
  - CRC Delimiter
  - ACK Delimiter
  - End of Frame
  - Inter-frame space
- Otherwise Form Error is generated

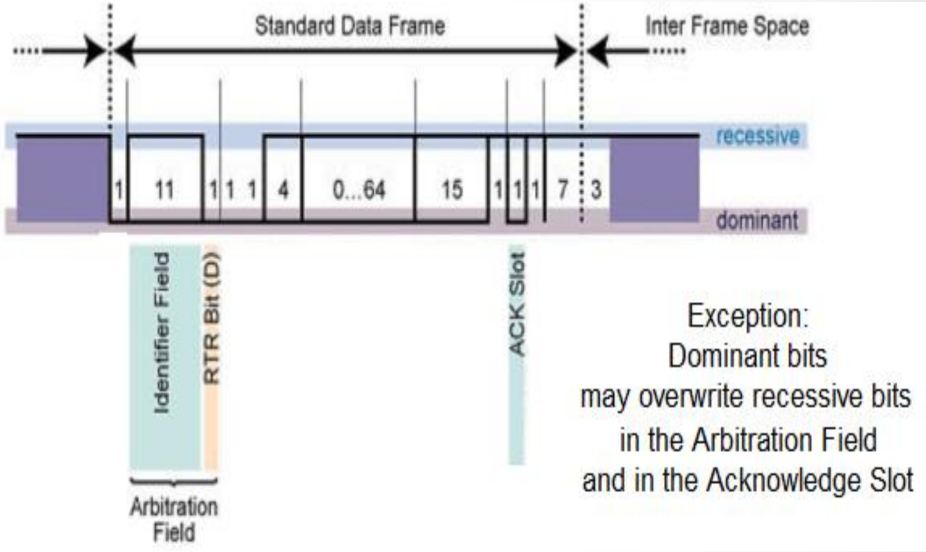


CAN

CAN Error Detection:

Bit Monitoring

- Each Node transmits a bit must be able to read back it correctly.
- Dominant bits are allowed to overwrite recessive bits only in the arbitration phase and ACK slot.
- Otherwise, a bit error should be issued.

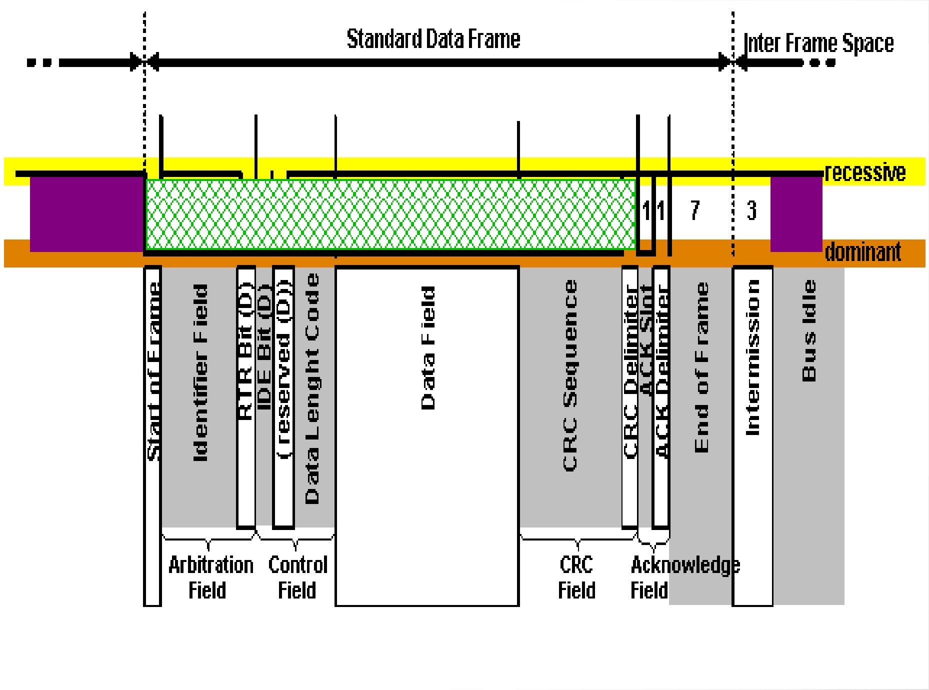


Exception:
Dominant bits
may overwrite recessive bits
in the Arbitration Field
and in the Acknowledge Slot

CAN Error Detection:

Bit Stuffing

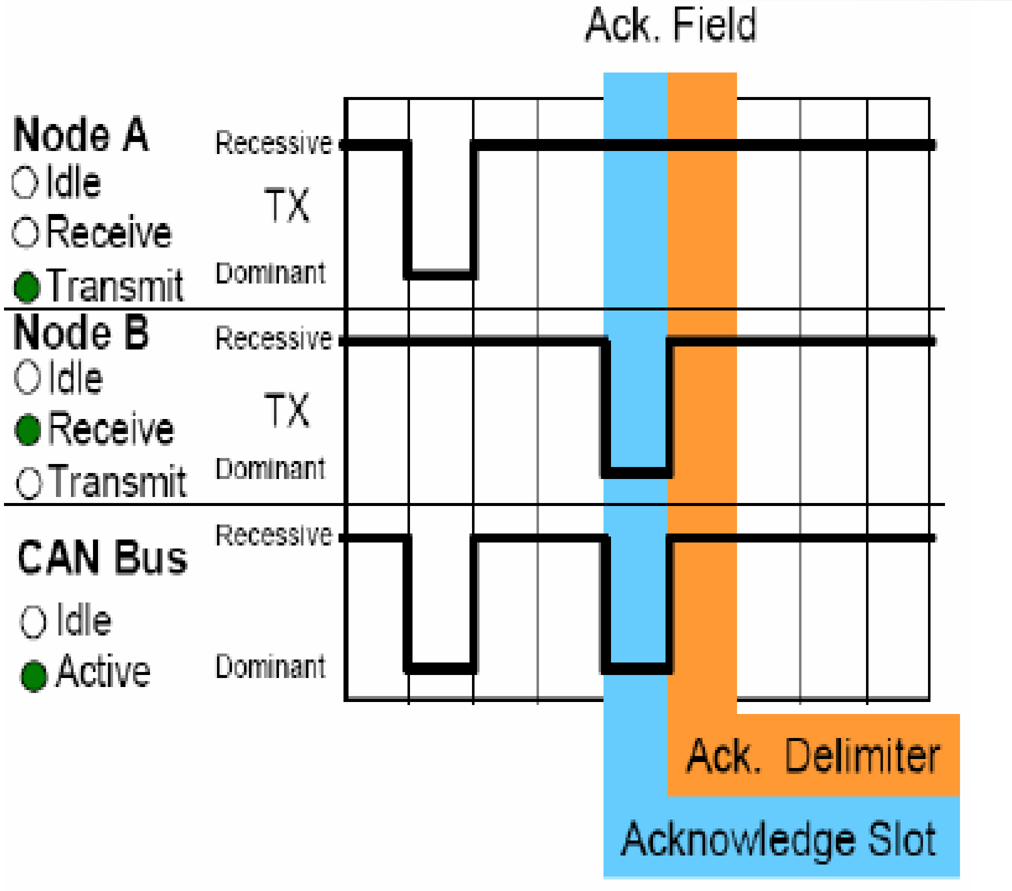
- 6 consecutive bits with same polarity are not allowed between Start Of Frame and CRC Delimiter
- Otherwise Bit Stuffing Error.



CAN Error Detection:

Acknowledgment (ACK)

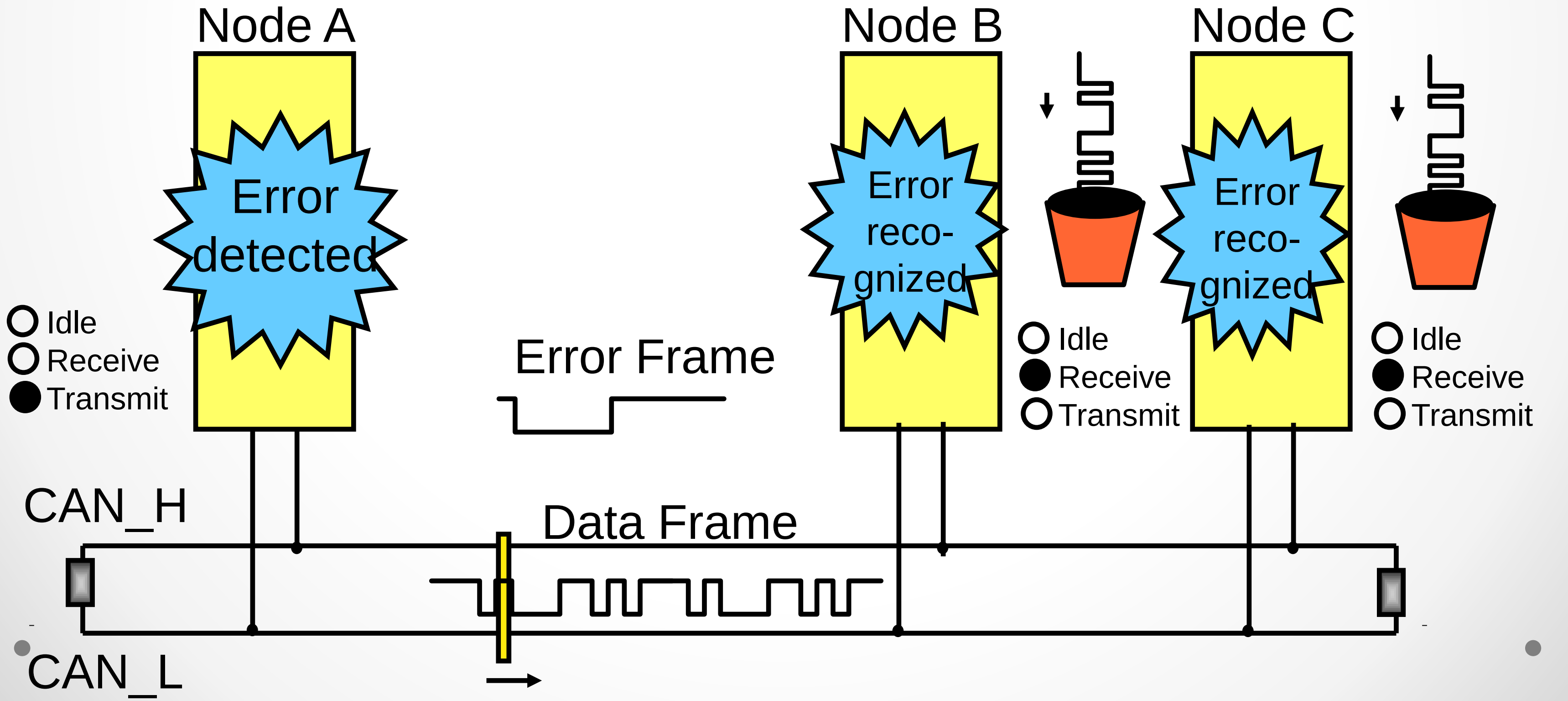
- A frame must be acknowledged by at least one other node.
- Otherwise ACK Error.



CAN

CAN Error Handling:

- Detected errors are made public to all other nodes via Error Frames.
- The transmission of the erroneous message is aborted and the frame is repeated as soon as possible.



CAN :

- http://www.engineersgarage.com/article/what-is-controller-area-network?page=3
- http://www.nxp.com/news/press-releases/2013/12/nxp-releases-smallest-most-efficient-can-system-basis-chips-for-in-vehicle-networks.html#!