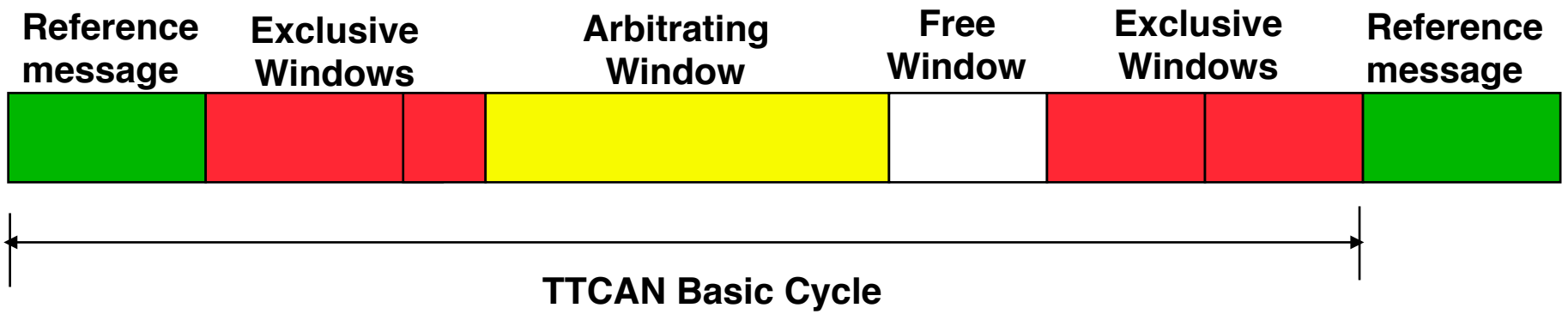
# Time Triggered CAN
# TTCAN

**Time Triggered CAN: TTCAN (Führer, Müller, Dieterle, Hartwich, Hugel, Walther,(Bosch))**

# Basic Cycle and Time Windows




reference message: indicates the start of a cycle,
exclusive window : used for critical periodic state messages,
arbitrating window: used for spontaneous state and event messages,
free window : window for further extensions and gap to the next exclusive window.

**RETRANSMISSIONS ARE GENERALLY NOT ALLOWED IN TTCAN !!**

# Scheduling a Basic cycle on a node

**Node n**

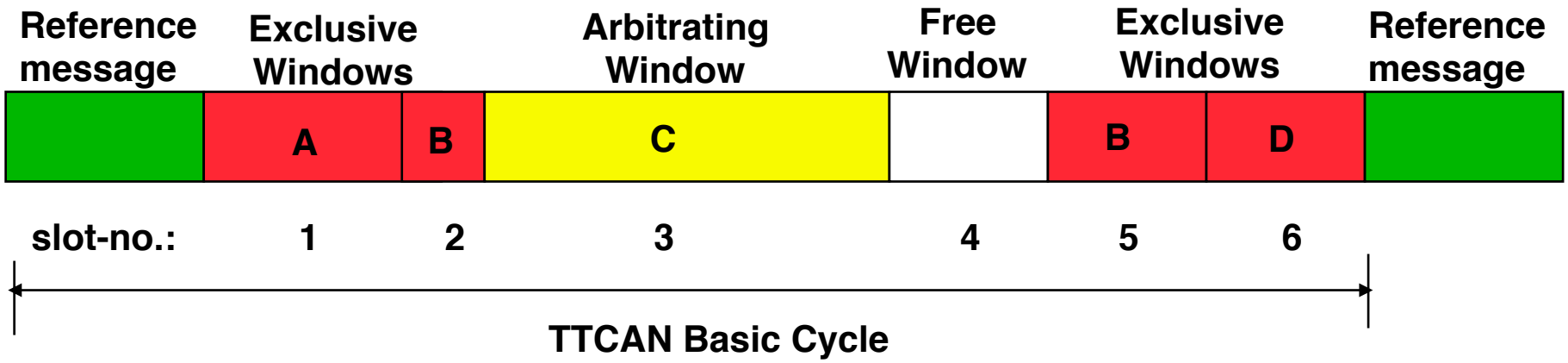
**Send msg B in slot 2 and 5**

**Send msg F in slot 3**

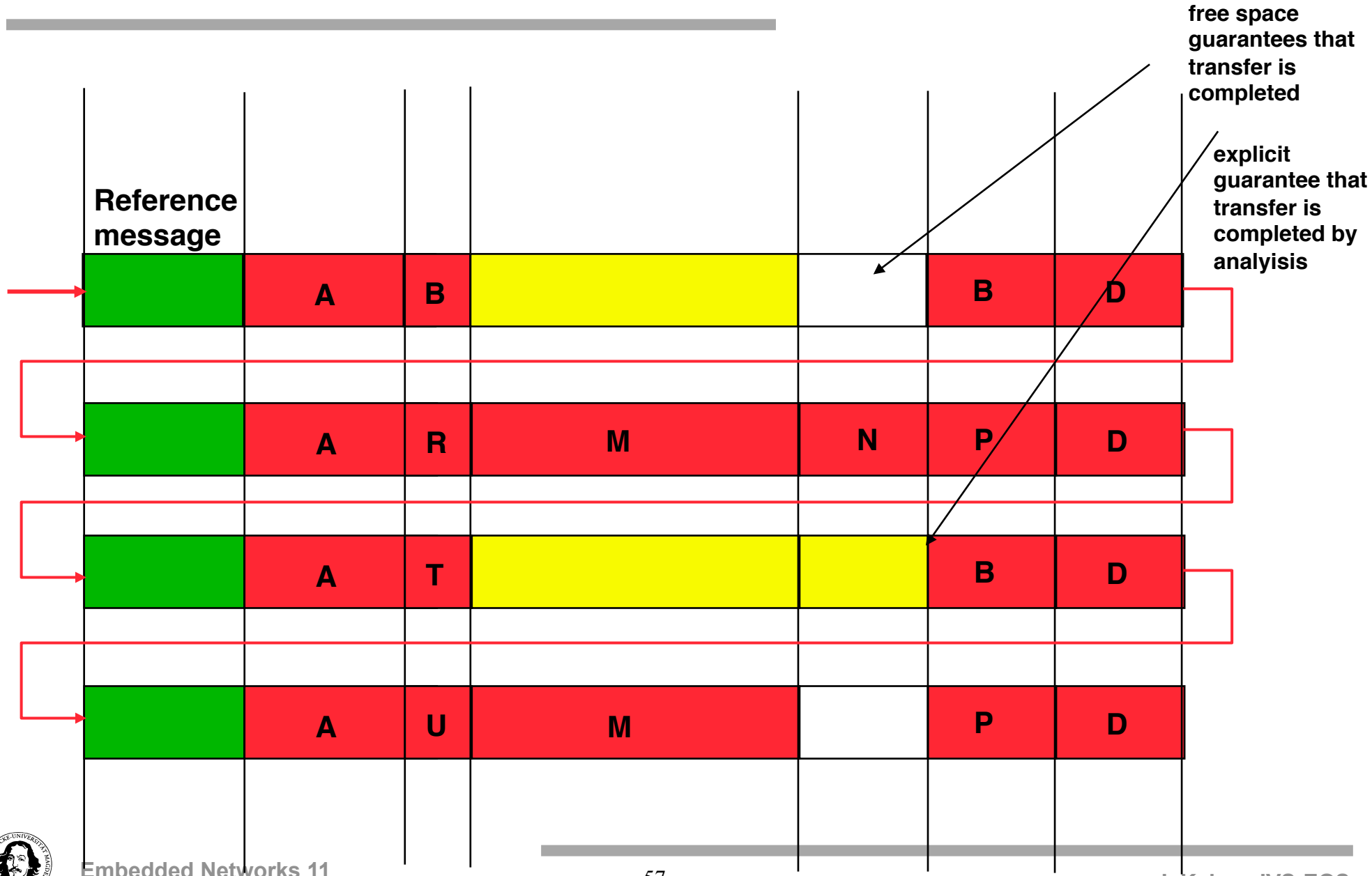
**Receive msg D in slot 6**

**Constraint:**

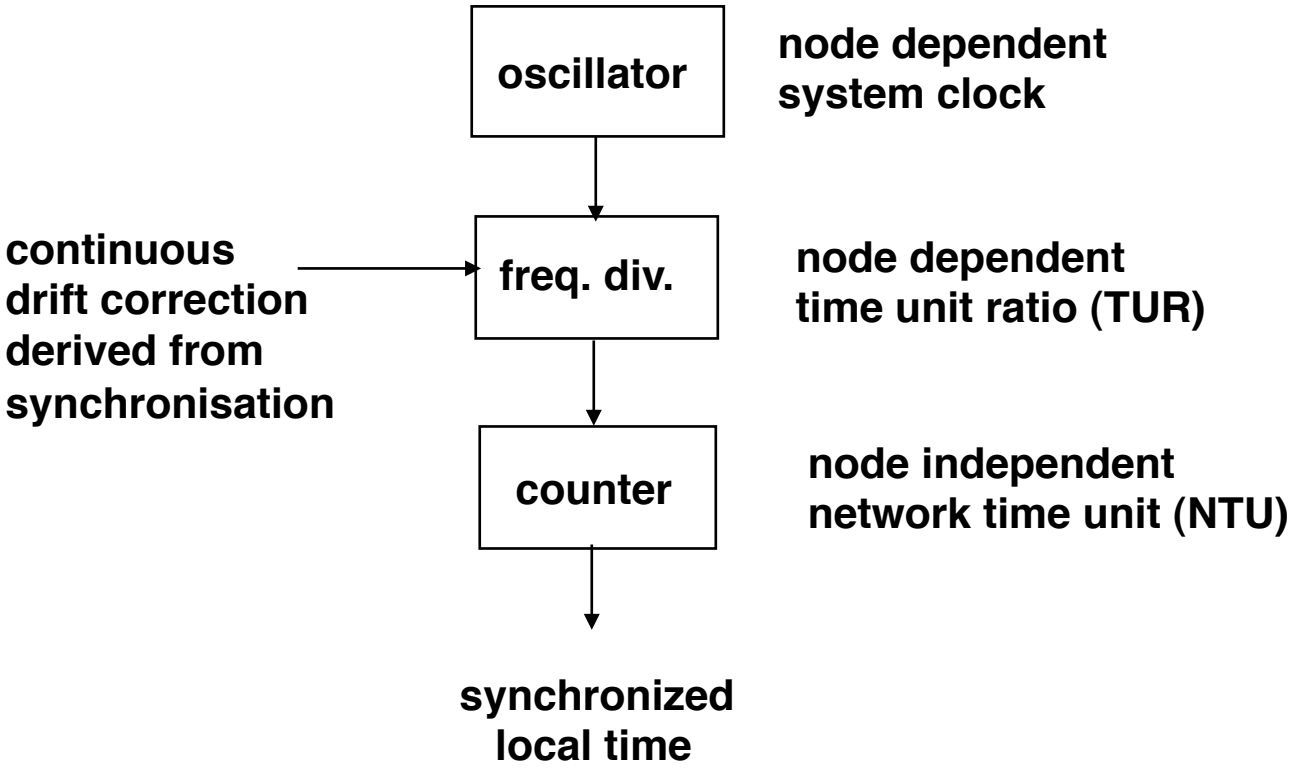
**A message transfer in an arbitrating window must be successfully completed before the start of an exclusive window.**



# Concatenating Basic Cycles to a MATRIX CYCLE



# Time and clock synchronization in TTCAN




Synchronization based on the existence of a Time Master.

All nodes take a snapshot of their local time at the SoF (Start of Frame) bit of the reference message.

Because of dependability reasons, TTCAN supports redundant Time Masters.

Arbitration among Time Masters is based on the priority scheme of CAN.

# Conclusion

**TT-CAN adds predictability to CAN**

**TT-CAN considers periodic message transfer**

**Fault handling differs substantially from Standard CAN**

**Clock synchronization is supported by hardware**

**Hybrid approaches are available in the scientific community**

# Coexistence of time-triggered and event-triggered mechanisms on the CAN-Bus

# ???

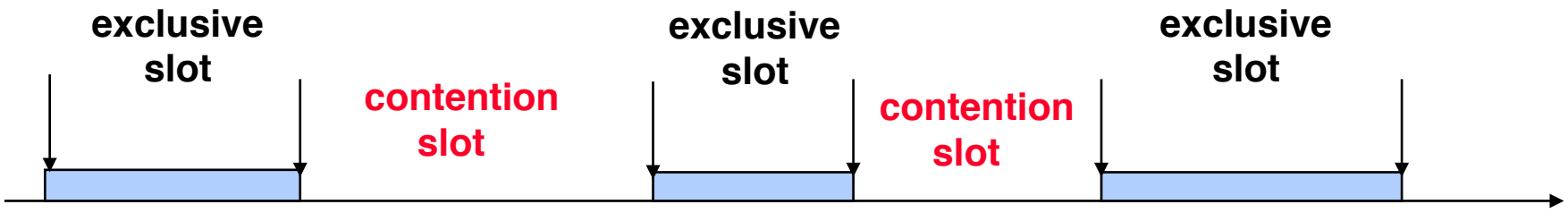
**Is it possible and what are the trade-offs?**

1. **Time Triggered CAN: TTCAN (Führer, Müller, Dieterle, Hartwich, Hugel, Walther,(Bosch))**
2. **Dynamic Priorities (Kaiser, Livani)**

# Integration of TT- and ET- communication

# by dynamic priorities

## Basic Idea: Reserve slots for hard real-time traffic and schedule soft real-time traffic in the remaining slots

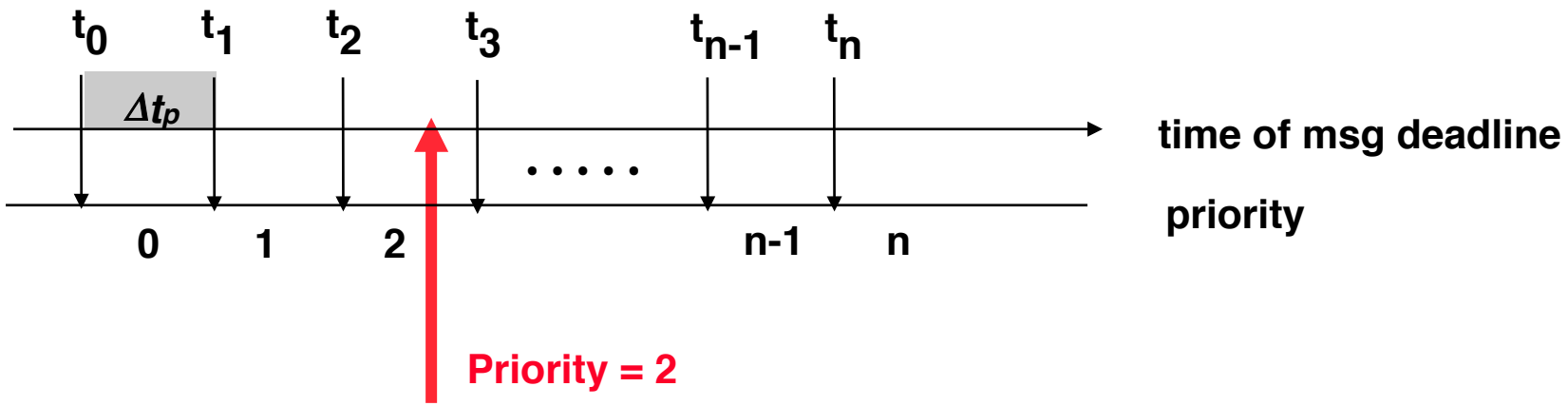



**The priority scheme is used to enforce high priority message transmission in the exclusive slots.**

# What is the advantage over TDMA?

# Mapping Deadlines to Priorities

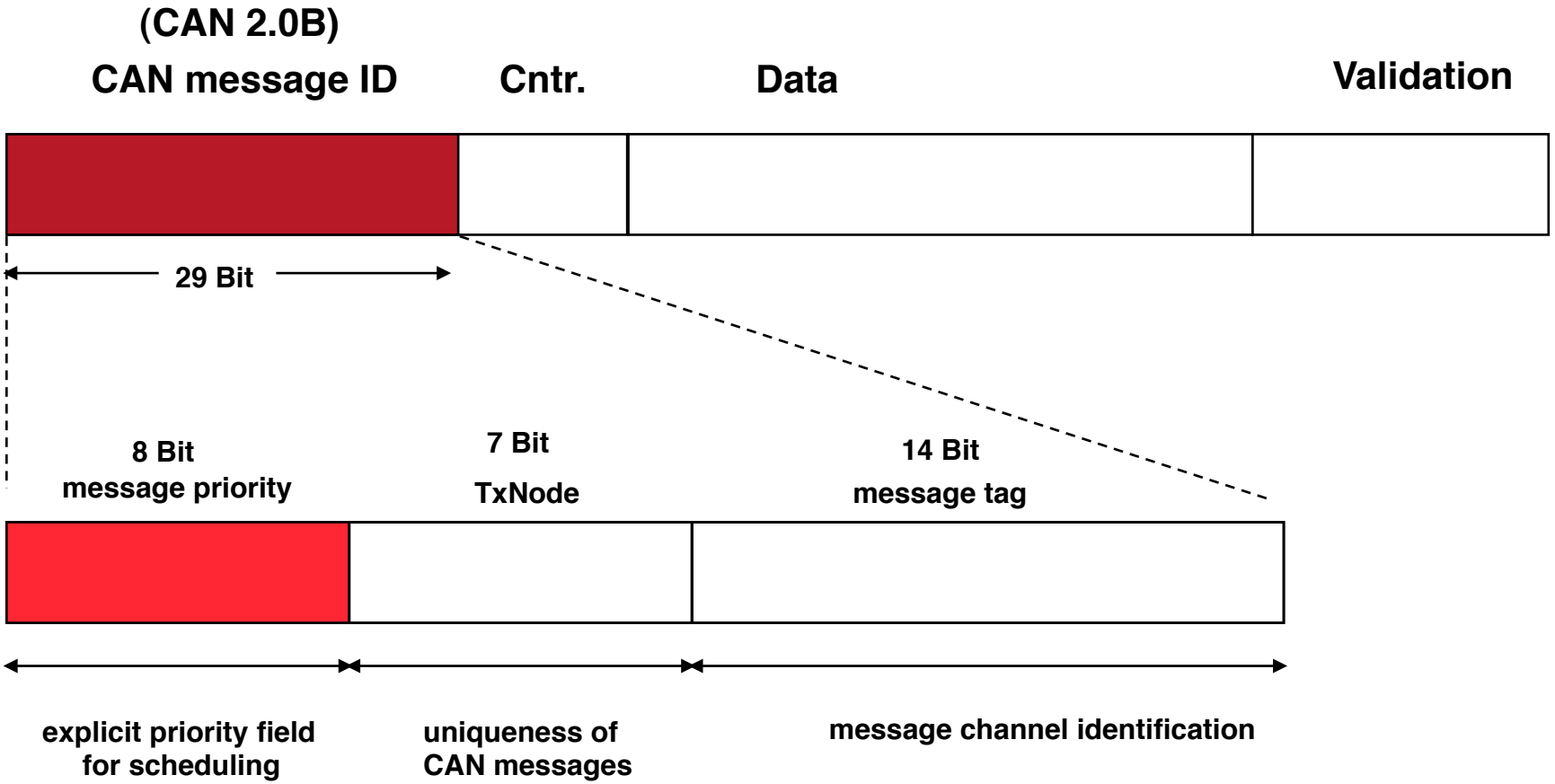
- Messages have deadlines
- Deadlines can be transformed into priorities



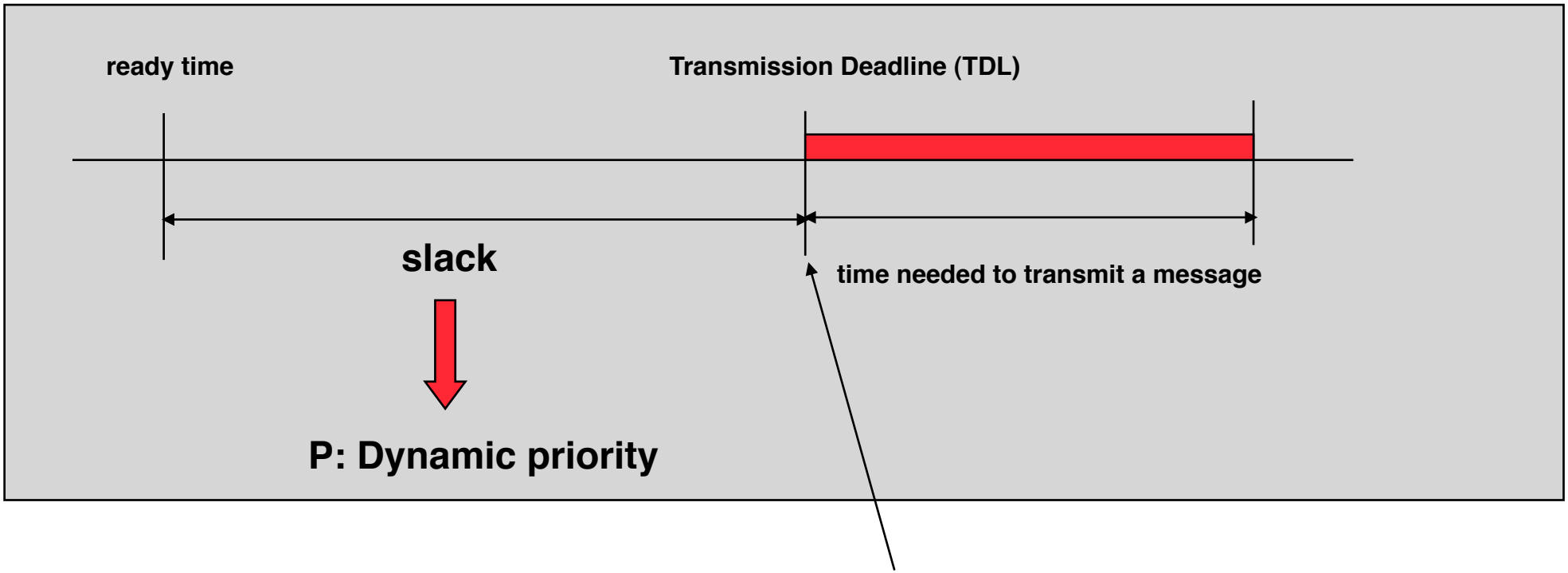

Wanted

a global priority-based message dispatcher

# Structuring the CAN-ID

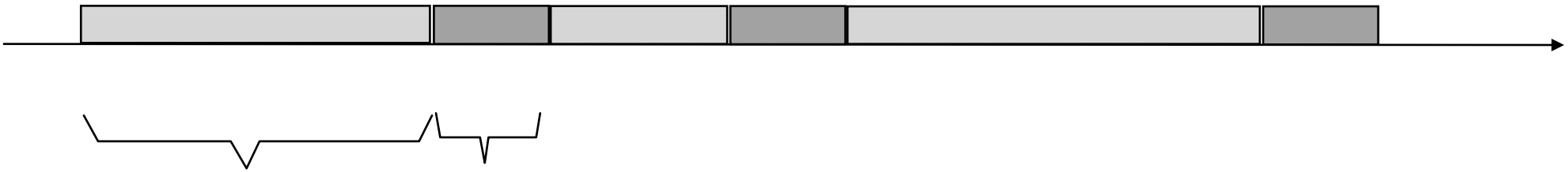


# Scheduling messages with guarantees




**At TDL:** $P_{HRTM} > P_{SRTM} > P_{NRTM}$

# How many HRT-slots can be guaranteed ?



$\Delta C_{max}$ + $\delta_{clock}$

$\Delta C_{max}$ **max. time interval (possibly under failure assumptions), which is neccessary to safely transmit a message to the destination**

**$\Delta C_{max}$ is a worst case assumption under all anticipated load and failure conditions**

$\delta_{clock}$ **max. offset, i.e. the difference between any two local clocks**

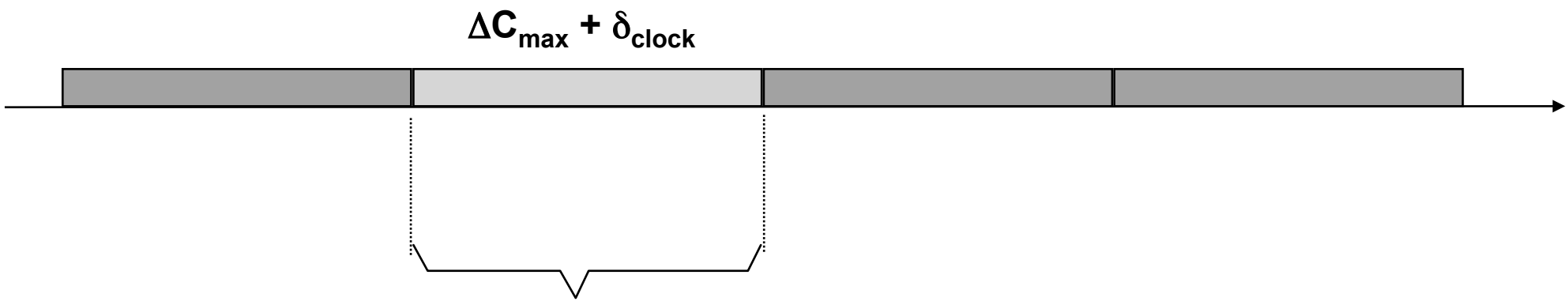
# CAN Inaccessibility Times*

**Data Rate 1 Mbps , Standard Format**

| Scenario | $t_{inacc}$ (µs) | |
|---|---|---|
| Bit Errors | 155.0 | ← worst case single |
| Bit Stuffing Errors | 145.0 | |
| CRC Errors | 148.0 | |
| Form Errors | 154.0 | |
| Ack. Errors | 147.0 | |
| Overload Errors | 40.0 | |
| Reactive Overload Errors | 23.0 | |
| Overload Form Errors | 60.0 | |
| Multiple Consecutive Errors (n=3) | 195.0 | |
| Multiple Successive Errors (n=3) | 465.0 | |
| Transmitter Failure | 2480.0 | ← worst case multiple |
| Receiver Failure | 2325.0 | |

P. Verissimo, J. Ruffino, L. Ming:" How hard is hard real-time communication on field-busses?"

# Utilization of CAN for HRT-messages

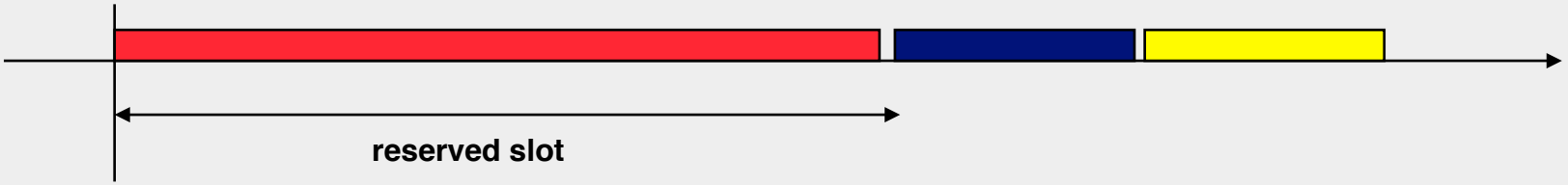



| fault assumption n | m | $\Delta C_{max}$ + 50 µs $\delta_{clock}$ (µs) | HRT messages / sec. # |
|---|---|---|---|
| 0 | 0 | 358 | 2793 |
| 0 | 1 | 532 | 1880 |
| 0 | 3 | 880 | 1136 |
| 1 | 0 | 2988 | 335 |
| 1 | 3 | 3664 | 273 |

# Benefits of the approach

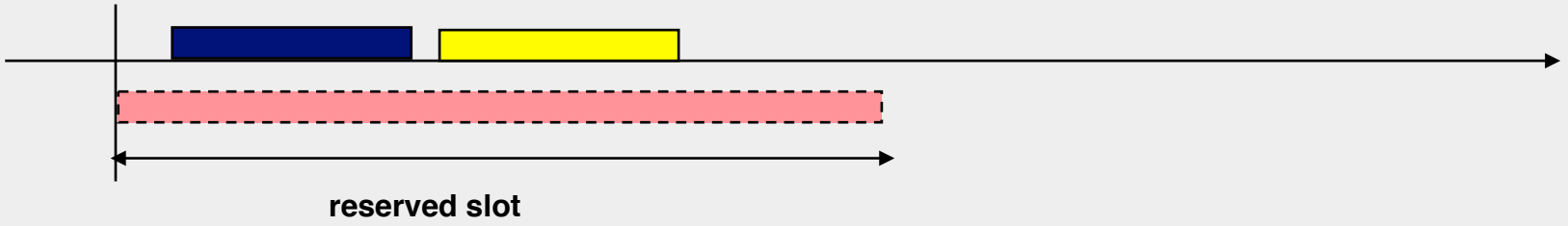
**Media access controlled by global time only (TDMA)**

**All nodes need global time**
**Unused slots remain unused**




**Media access in a system controlled by our priority scheme**

**Only nodes with HRT-msg need global time**
**Unused slots can be used by msg which are ready to be transmitted**



# Cost-Performance Trade-off

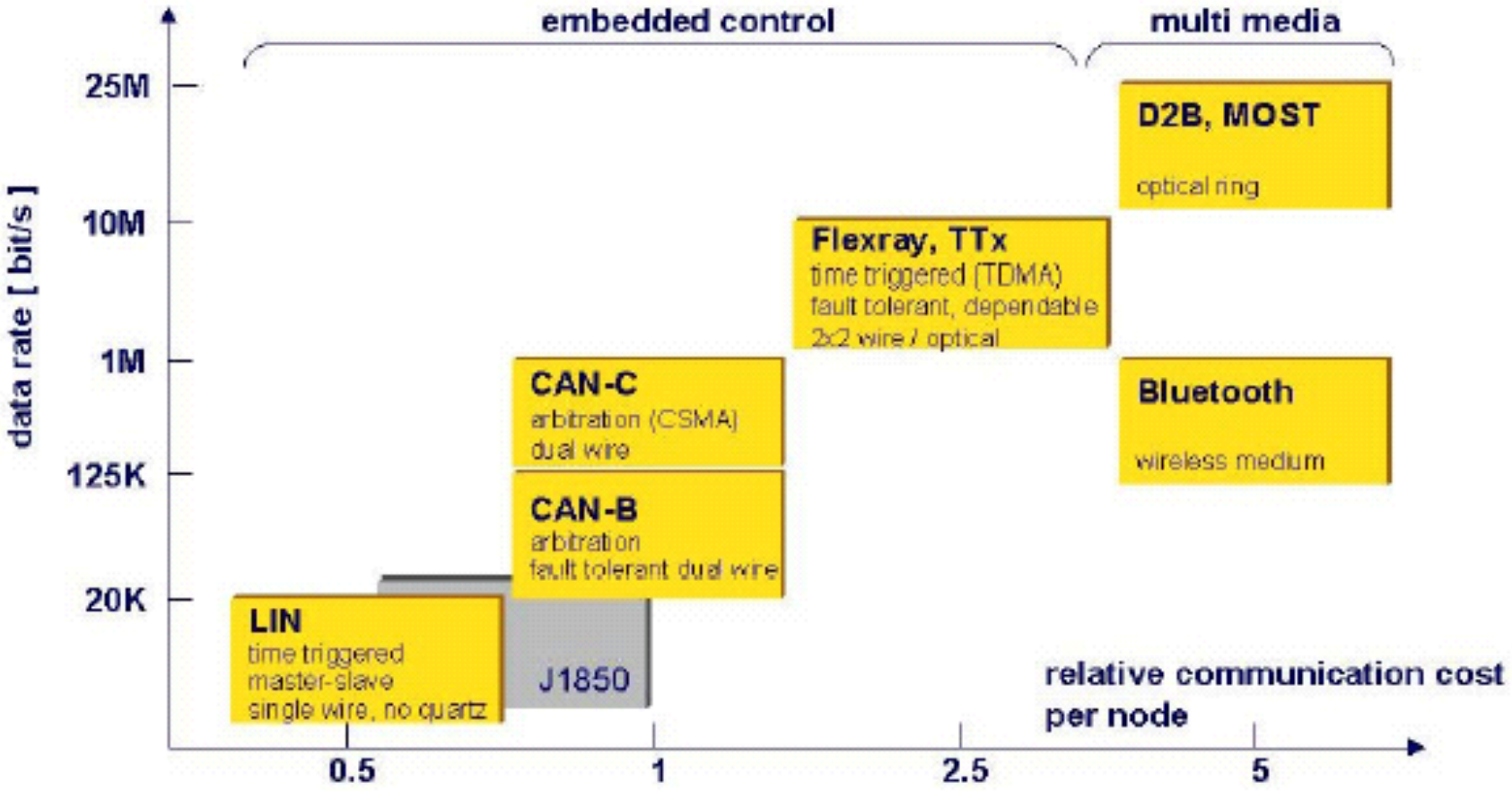
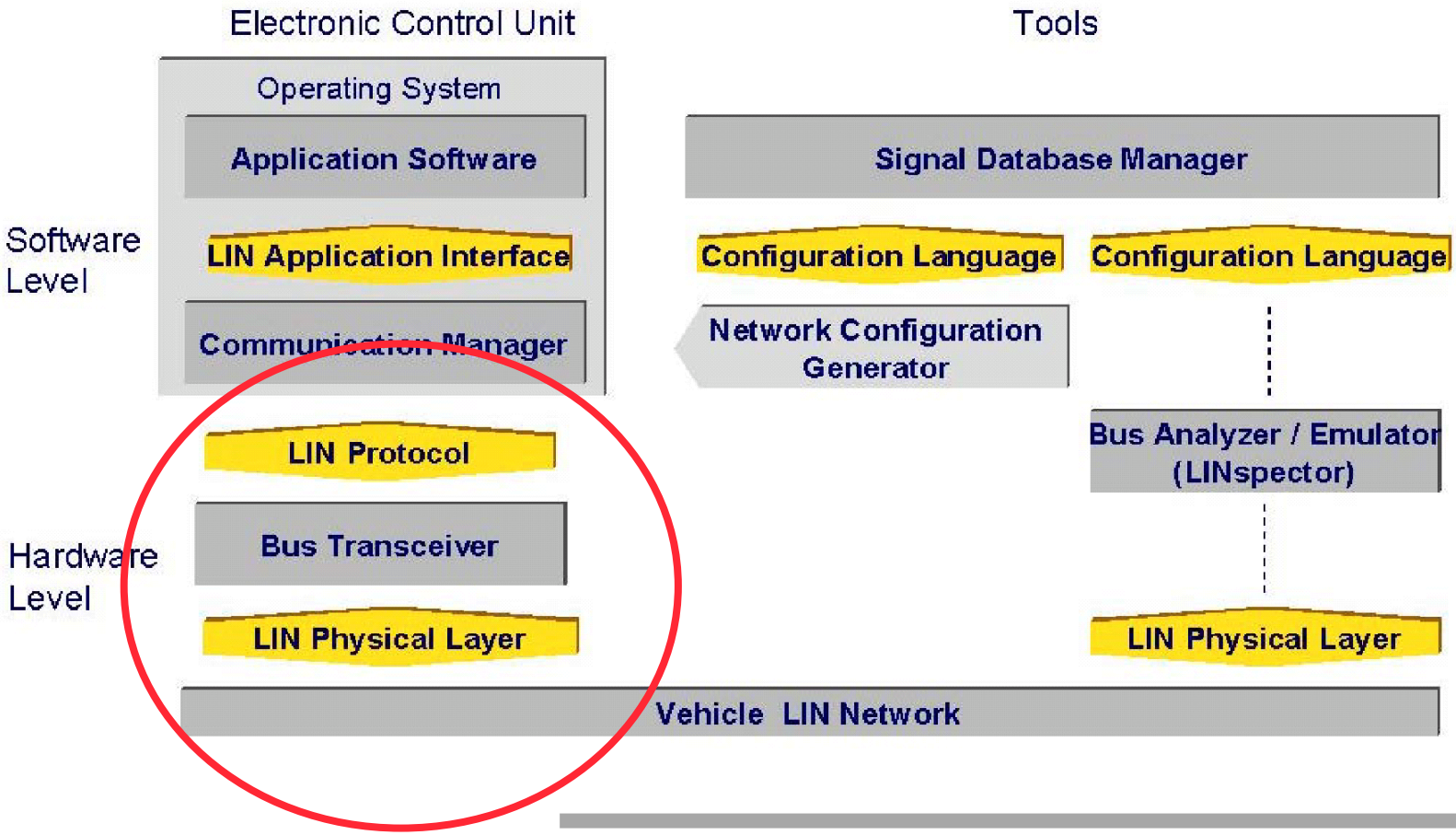



**Figure 1: Major Network Protocols in Vehicles**

# LIN (Local Interconnect Network)

**LIN Specification Package, Revision 1.2, Nov. 17, 2000**



# Properties of LIN

. single-master / multiple-slave concept

. low cost silicon implementation based on common UART/SCI interface hardware, an equivalent in software, or as pure state machine.

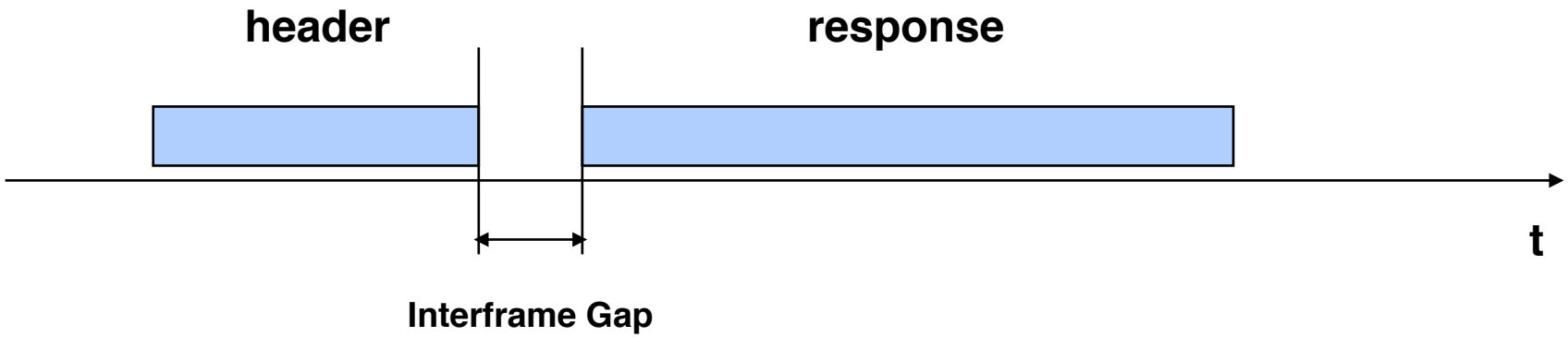
. self synchronization without quartz or ceramics resonator in the slave nodes

. guarantee of latency times for signal transmission

. low cost single-wire implementation

. speed up to 20kbit/s.

# Master-Slave communication in LIN

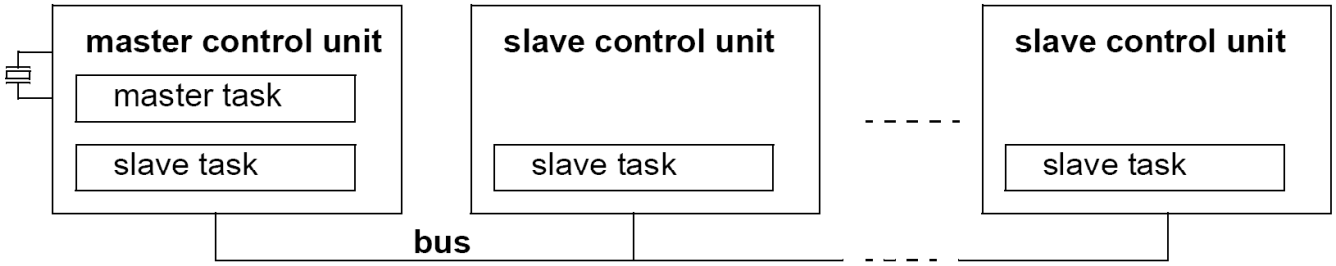



**Header:**
**- serves for the synchronisation of slaves**
**- specifies the sequence and length of the fields in the data frame**

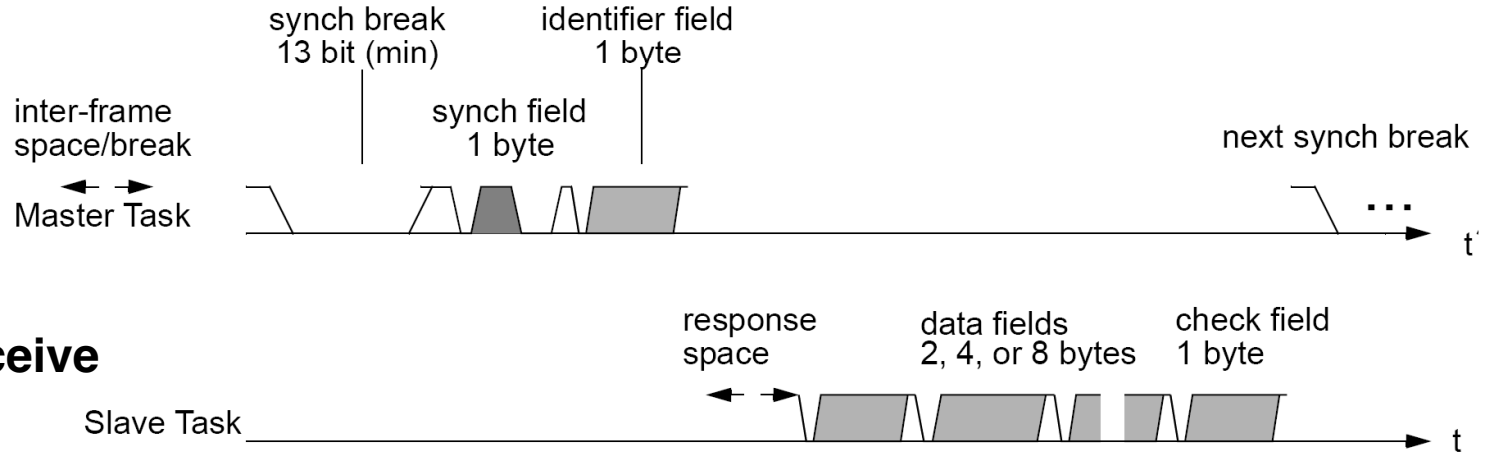
# LIN (Local Interconnect Network)

**LIN Specification Package, Revision 1.2, Nov. 17, 2000**

**time dependent master/slave protocol**




**only 1 slave is allowed to respond!**
**But all slaves receive the response.**



# LIN Specification Package, Revision 1.2, Nov. 17, 2000

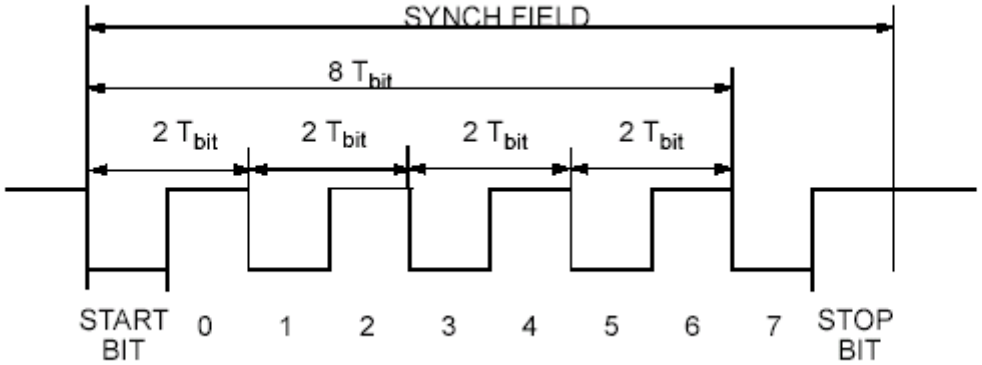


Figure 9.1: SYNCHRONIZATION FIELD

**Synch. Feld 0x55**

| clock tolerance | Name | $\Delta F / F_{Master}$ |
|---|---|---|
| master node | $F_{TOL\_RES\_MASTER}$ | < ±0.5% |
| slave node with quartz or ceramic resonator (without the need to synchronize) | $F_{TOL\_RES\_SLAVE}$ | < ±1.5% |
| slave without resonator, lost synchronization | $F_{TOL\_UNSYNCH}$ | < ±15% |
| slave without resonator, synchronized and for a complete message | $F_{TOL\_SYNCH}$ | < ±2% |

Table 8.1: Oscillator Tolerance

## LIN Specification Package, Revision 1.2, Nov. 17, 2000

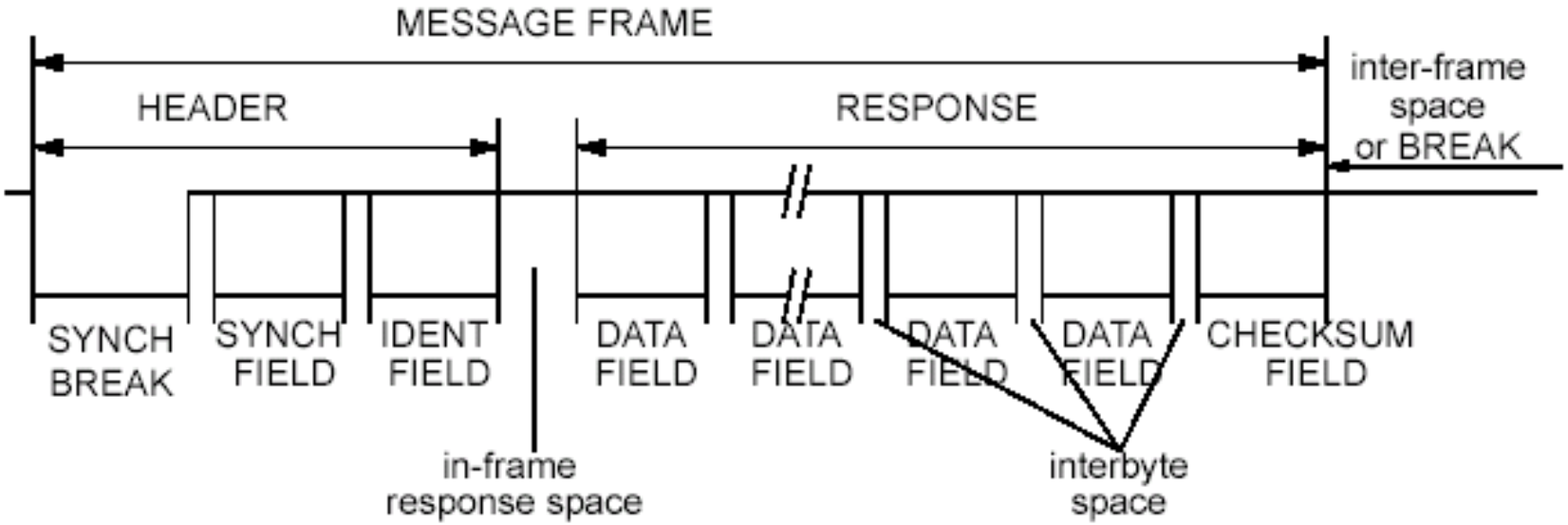



Figure 3.1: LIN MESSAGE FRAME

## LIN Specification Package, Revision 1.2, Nov. 17, 2000

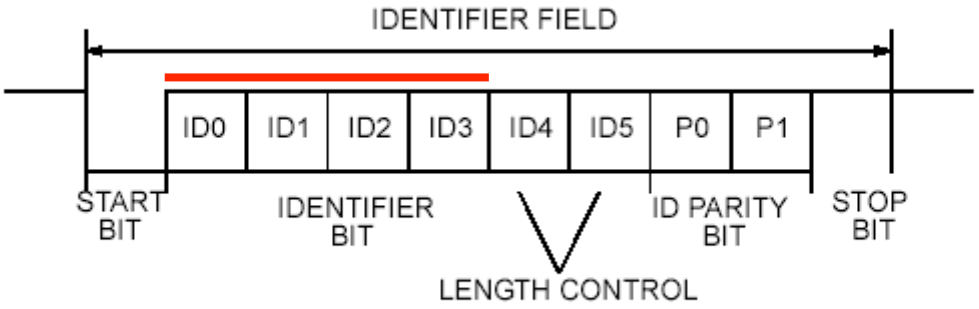



Figure 3.5: IDENTIFIER FIELD

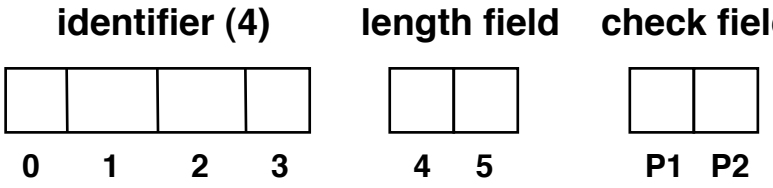
**64 identifiers**

**divided in 4 groups of length: 2,4, and 8 bytes**

**An ID identifies the content of a message, not the sender or receiver !**

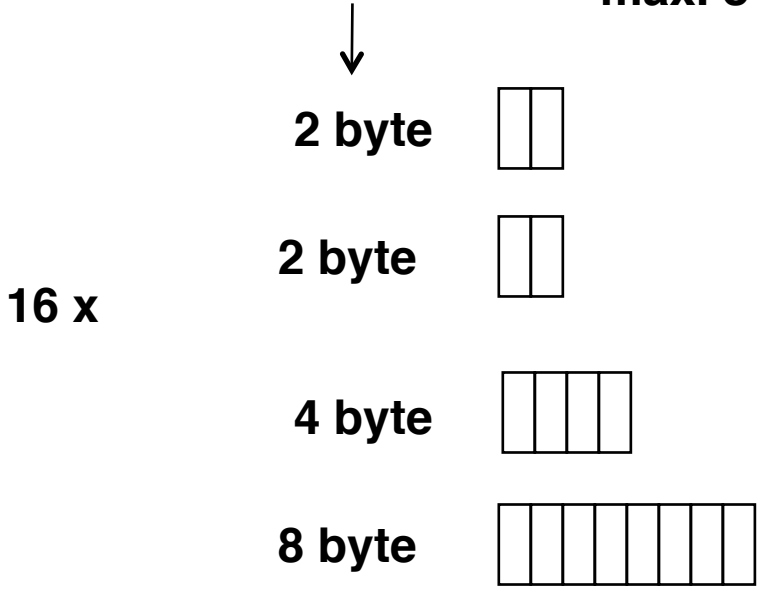
**Slaves can be added or removed without changing any software in the other slaves.**

# LIN frame format




**content-based addressing**

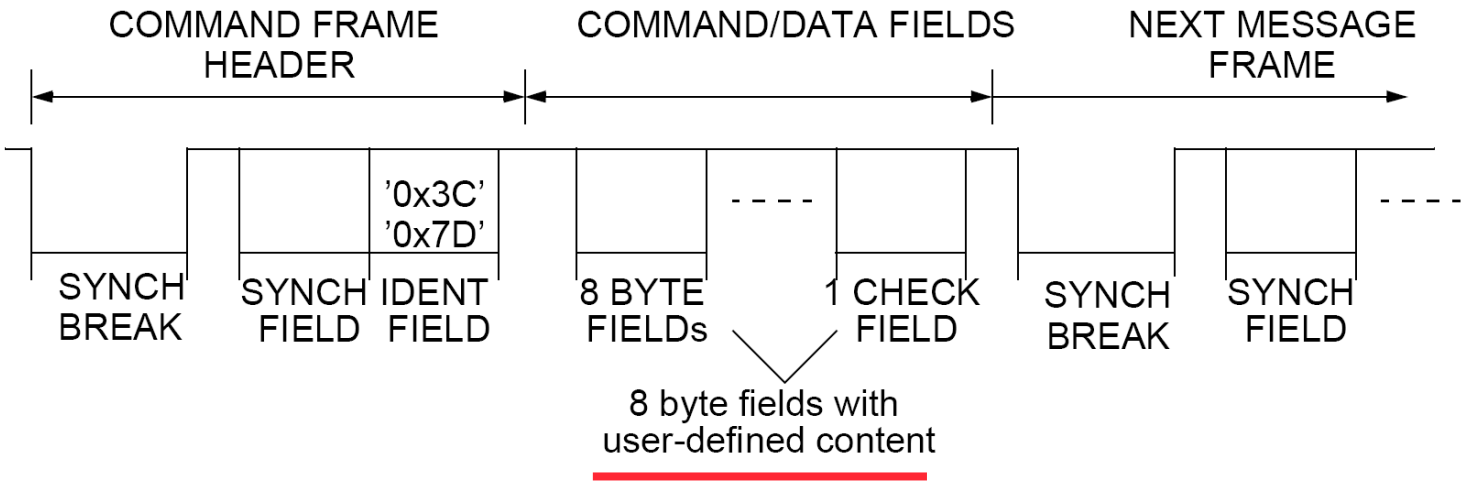
**max. 8 Byte response frame**




| ID5 | ID4 | $N_{DATA}$ (number of data fields) [byte] |
|---|---|---|
| 0 | 0 | 2 |
| 0 | 1 | 2 |
| 1 | 0 | 4 |
| 1 | 1 | 8 |

**reserved IDs: Master request Frame (0x3C), Slave Response Frame (0x3D)**
**Extended Frames (User 0x3E, Reserved 0x3F)**

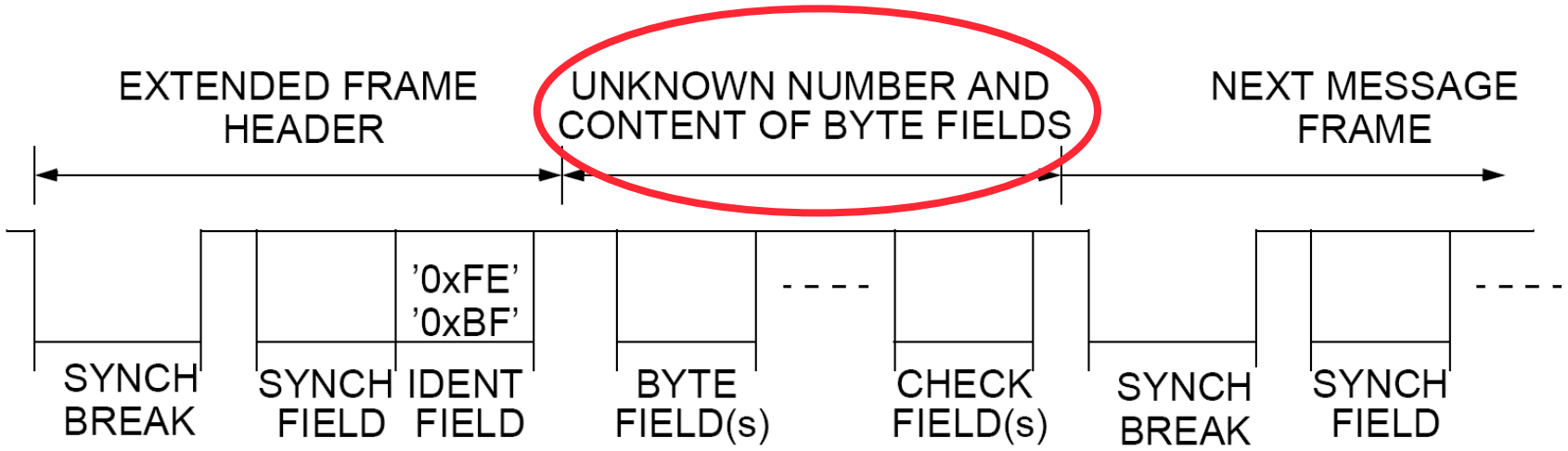
# LIN Master Request Frame




**Download of data to the slave.**
**Request of data from the slave.**

**Multiple 8 byte fields possible!**
**Slave address is part of the command fields.**

## LIN Extended Frame




**slaves, whiche are not addressed (interested resp.) wait until the next SyncBreak!**

# Error detection capabilties of LIN:

**Bit-Error**

**Checksum-Error**

**Identifier-Parity-Error**

**Slave-Not-Responding-Error**

**Inconsistent-Synch-Field-Error**

**No-Bus-Activity**

# Automotive and highly dependable Networks

**TTP/C**
**Byteflight**
**FlexRay**
**Time Triggered CAN (TTCAN)**
**LIN**