CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

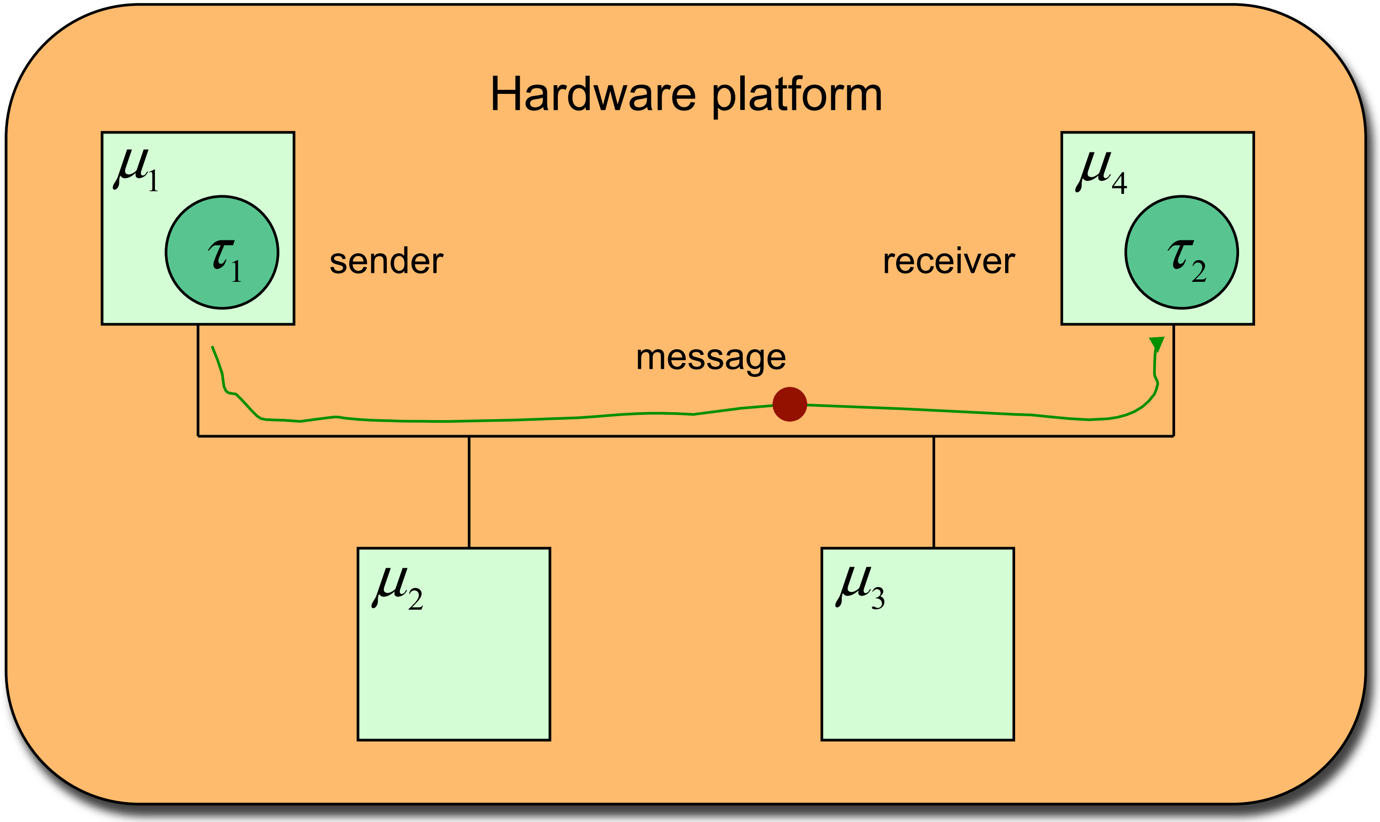
Dependable Real-Time Systems

Lecture #11

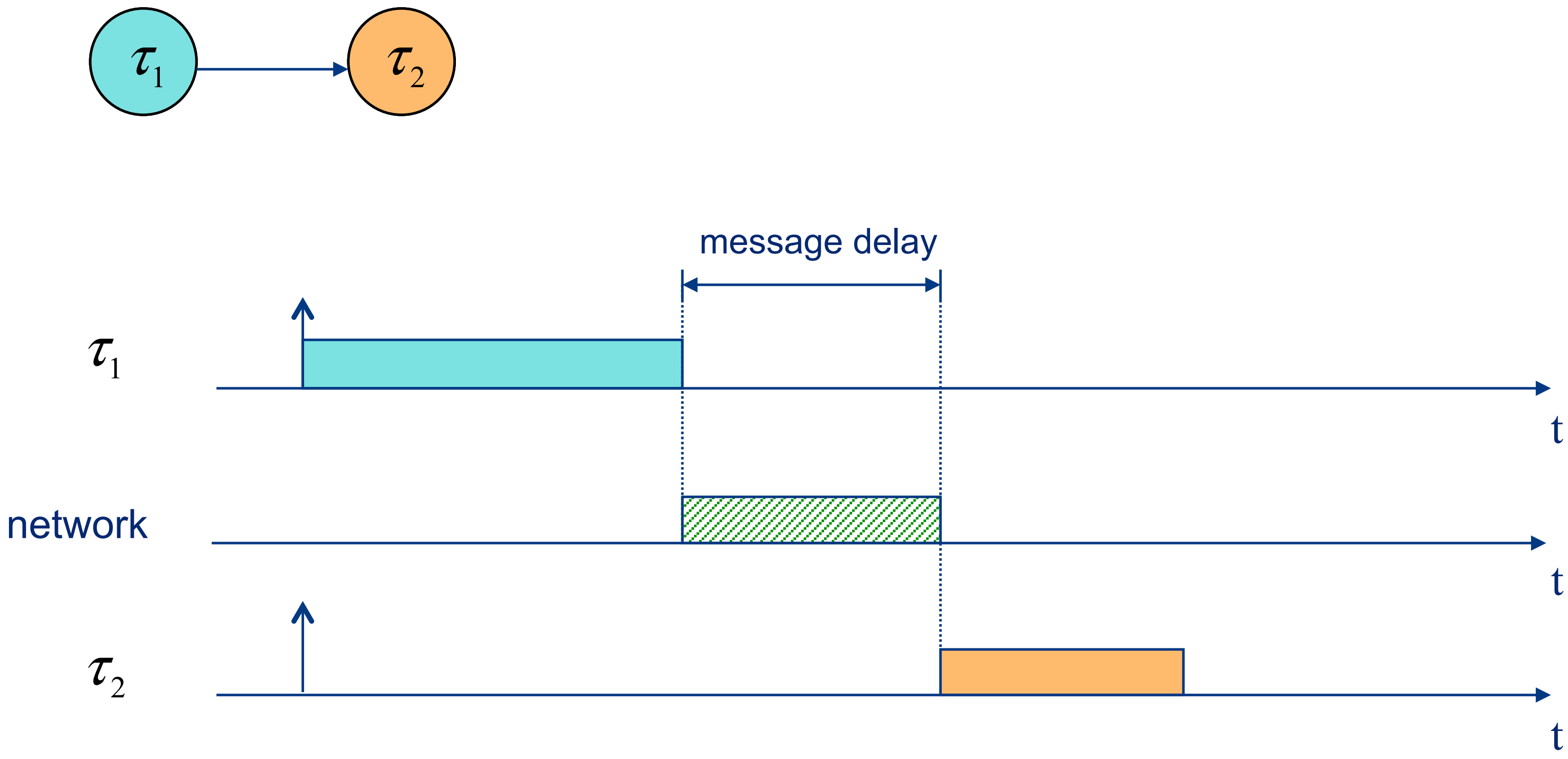
Professor Jan Jonsson

Department of Computer Science and Engineering
Chalmers University of Technology

Network communication



Network communication



Network communication

Message delay:

- Message delays are caused by the following overheads:
  - Formatting (packetizing) the message
  - Queuing the message, while waiting for access to medium
  - Transmitting the message on the medium
  - Notifying the receiver of message arrival
  - Deformatting (depacketizing) the message

Formatting/deformatting overheads are typically included in the execution time of the sending/receiving task.

Network communication

Queuing delay:

- The cause of the queuing delay for a message depends on the actual network used. For example:
  - Waiting for a corresponding time slot (e.g., FlexRay)
  - Waiting for a transmission token (e.g., Token Ring)
  - Waiting for a contention-free transmission (e.g., Ethernet)
  - Waiting for network priority negotiation (e.g., CAN)
  - Waiting for removal from priority queue (e.g., Switched Ethernet)

To be used in a real-time system with hard timing constraints the queuing delay must be bounded.

Network communication

How is a message transfer handled in task scheduling?

- Integrated scheduling:
  - Scheduling of tasks and inter-task communication are regarded as comparable operations.
  - Requires compatible dispatching strategies.
- Separated scheduling:
  - Scheduling of tasks and inter-task communication are performed as separate steps.
  - Allows for different dispatching strategies.

Network communication

Integrated scheduling:

- Suitable for simple homogeneous systems with known assignment of tasks to processors
- Examples:
  - Time-driven task dispatching + TDMA network protocol
  - Static-priority task dispatching + Token Ring network protocol
  - Static-priority task dispatching + CAN protocol

Network communication

Separated scheduling:

- Suitable for heterogeneous systems or when assignment of tasks to processors is not always known in advance
- Motivation:
  - Transmission delay is zero if communicating tasks are assigned to the same processor
  - Number of communication links that a message traverses in a multi-hop network may be a function of the assignment (depends on topology and routing strategy)
  - Different communication links may employ different message dispatching policies

Network communication

How is the message transfer synchronized?

- Asynchronous communication:
  - Sending and reception of messages are performed as independent operations at run-time.
  - Network controller chip administrates message transmission and reception (example: CAN, Ethernet)
- Synchronous communication:
  - Sending and receiving tasks synchronize their network medium access at run-time.
  - Network controller chip makes sure message transmission and reception occurs within a dedicated time slot in a TDMA bus network (example: FlexRay)

Network communication

Jitter in message scheduling

- Queuing jitter:
  - The time lapse from the moment a sending task initiates a message event until the message is queued in the network controller
  - The longest such time lapse should be accounted for in the schedulability analysis of the message delay
- Release jitter:
  - Variations in the message arrival time at the receiving task, due to queuing delays at the sender or in the switches in multi-hop networks
  - The largest such variation should be accounted for in the schedulability analysis of the receiving task

Network communication

Jitter in message scheduling

- Jitter minimization/reduction:
  - Asynchronous communication: Release jitter can be minimized by using offsets at the receiving tasks, or it can be reduced by regulating the message periodicity in the switches in multi-hop networks
  - Synchronous communication: Off-line static scheduling is typically used for matching a message time slot on the network bus with the execution of sending and receiving tasks. The jitter can therefore be kept to a minimum by instructing the off-line scheduling algorithm to use jitter minimization as the scheduling objective

Network communication

How is the message transfer imposed with a deadline?

- As a separate schedulable entity:
  - A suitable deadline-assignment technique must be used to distribute an end-to-end deadline over the sending task, the receiving task, and the message transfer
  - Worst-case message delay must be known beforehand
- As part of the receiving task:
  - No explicit deadline needed for message transfer
  - May impose release jitter on the receiving task

Network communication

How is the message transferred onto the medium?

- Contention-free communication:
  - Senders need not contend for medium access at run-time
  - Examples: TTCAN, FlexRay, Switched Ethernet
- Token-based communication:
  - Each sender using the medium gets one chance to send its messages, based on a predetermined order
  - Examples: Token Ring, FDDI
- Collision-based communication:
  - Senders may have to contend for the medium at run-time
  - Examples: Ethernet, CAN

Network communication

Contention-free communication:

- One or more dedicated time slots for each task/processor
  - Shared communication bus
  - Medium access is divided into communication cycles (normally related to task hyper periods to allow for integrated scheduling)
  - Dedicated time slots provide bounded queuing delays
  - Example: TTCAN ("exclusive mode"), FlexRay ("static segment")
- One sender only for each communication line
  - Point-to-point communication networks with link switches
  - Output and input buffers with deterministic queuing policies in switches provide bounded queuing delays
  - Example: Switched Ethernet

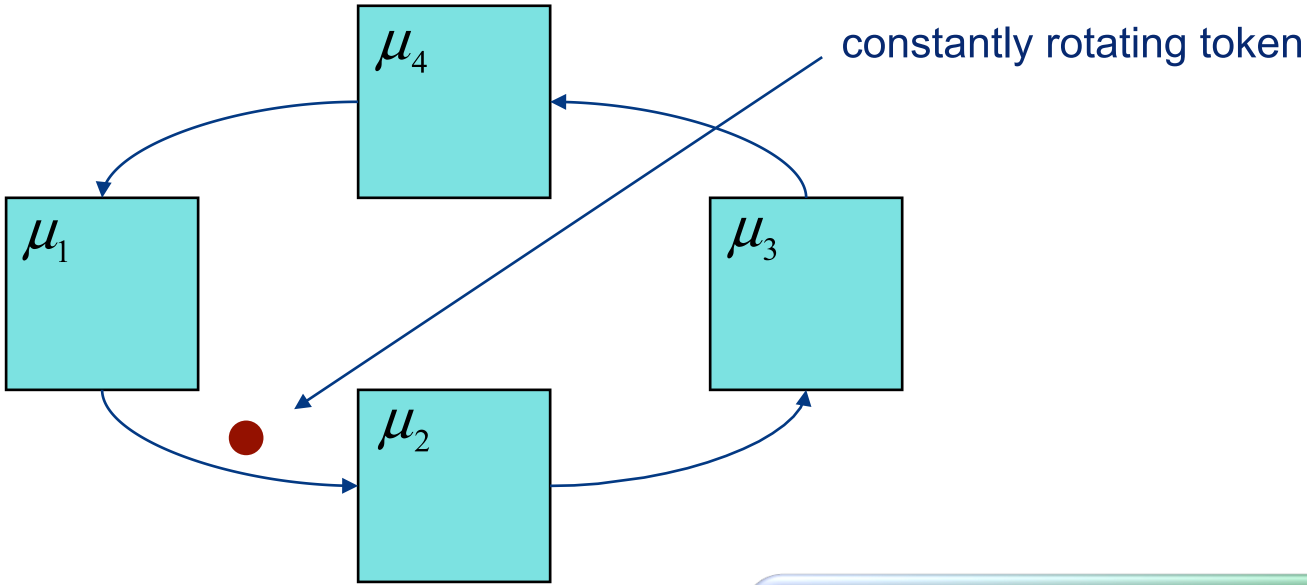
Network communication

Token-based communication:

- Utilize a token for the arbitration of message transmissions on a shared medium
  - The sender is only allowed to transmit its messages when it possesses the token
  - Message priorities or quotas allows for bounded queuing delays
- Examples:
  - Token Bus (IEEE 802.4)
  - Token Ring (IEEE 802.5)
  - FDDI (ANSI X3T9.5)

Token-based communication

Token Ring:

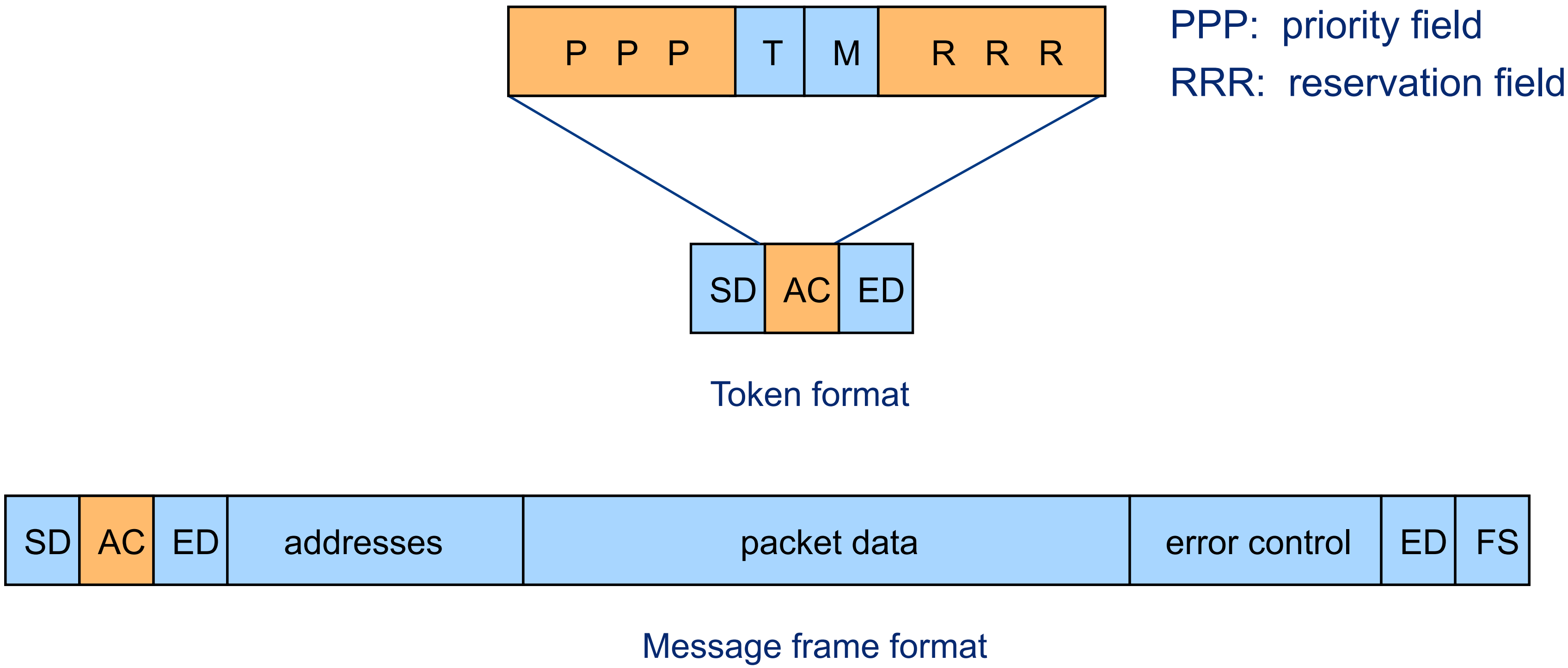


"token walk time": $W_T = (n-1)D_B + L + T_{prop}$

D_B : node delay
L : buffer delay
T_{prop} : ring propagation delay

Token-based communication

Token Ring message frame format:



Token-based communication

Token Ring protocol:

1. Each node examines RRR of a busy token as it passes and inserts the priority of its pending message only if it is greater than the priority currently in RRR.
2. A node does not grab a “free” token unless the priority of its pending message is at least as high as the priority in PPP. Then the token status is changed to “busy”.
3. A transmitting node appends its pending message after the “busy” token and sets RRR appropriately.
4. A transmitting node waits until it receives back the “busy” token before releasing the next “free” token with PPP set to the (possibly) updated RRR.

Token-based communication

Token Ring real-time protocol: (Sathaye & Strosnider, 1994)

The uniprocessor response-time analysis can be adapted to the Token Ring protocol by assuming a non-preemptive dispatching model.

- Caveats:
  - Messages cannot be interrupted during transmission, which means that message scheduling is non-preemptive.
  - Message headers must be included in message size
  - Notion of highest priority might be outdated since the system is distributed.
  - The number of priority bits (3) defined in IEEE 802.5 does not allow for more than 8 priority levels.

Token-based communication

Token Ring real-time protocol: (Sathaye & Strosnider, 1994)

A sufficient and necessary feasibility test:

$$\forall i : R_i = t_{sys} + b_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil e_j \leq D_i$$

t_{sys} : system overhead defined by the system

b_i : blocking time due to ongoing transmissions

e_j : "execution time" consisting of the following time components

- Capture token when node has highest-priority message pending
- Transmit message
- Transmit subsequent free token

Network communication

Collision-based communication:

- Utilize collision-detect mechanism to determine validity of message transmissions on a shared medium
  - The sender tries to send messages independently of other senders' intention to do so
  - Attempts may be done at any time or when some specific network state occurs
- Examples:
  - Ethernet w/ multiple senders (IEEE 802.3)
  - CAN (ISO 11898)

Collision-based communication

Ethernet protocols w/ multiple senders:

- Senders attempt to send a complete message
- If messages collide, all transmissions are aborted
- After collision, re-transmission is made after a random delay

Message queuing delay can in general not be bounded

Therefore, these protocols do not give any guarantees
for meeting imposed message deadlines!

Collision-based communication

Controller Area Network (CAN):

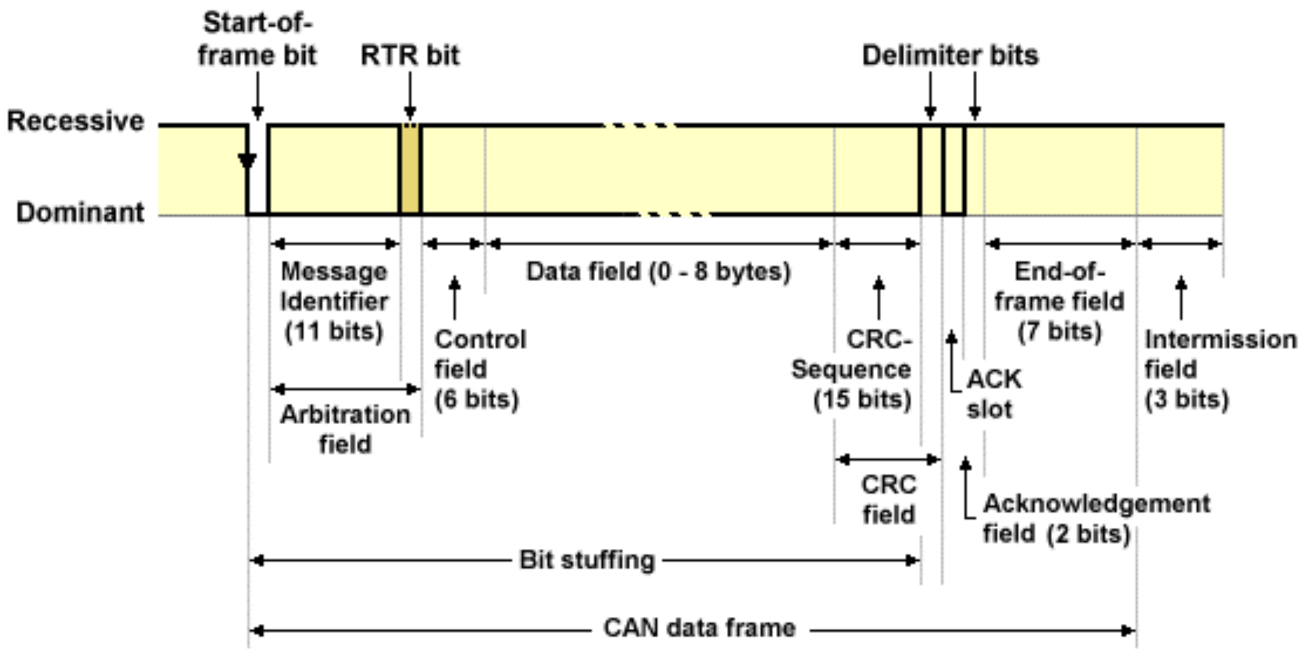
- Senders transmit a message header (with an identifier)
- If messages collide, a hardware-supported protocol is used to determine what sender will be allowed to send the rest of the message; transmissions by other senders are aborted

Message queuing delay can be bounded with appropriate identifier assignment!

Therefore, this protocol makes it possible to meet imposed message deadlines!

Collision-based communication

CAN message frame format: (short format)



Collision-based communication

CAN real-time protocol: (Davis et al, 2007)

The uniprocessor response-time analysis can be adapted to the CAN protocol by assuming a non-preemptive dispatching model.

- Caveats:
  - Messages cannot be interrupted during transmission, which means that message scheduling is non-preemptive.
  - Due to non-preemptive dispatching the busy period for each priority level may not be the same as for the preemptive case.
  - Message headers must be included in message size.
  - The message identifier can be used as message priority, but lower identifier number will mean higher priority.

Collision-based communication

CAN message transmission time: (short format)

Transmission time for message m:

$$C_m = (55 + 10 s_m) \cdot \tau_{\text{bit}}$$

s_m : number of bytes in data field for message m

τ_{bit} : transmission time for a single bit

Collision-based communication

CAN real-time protocol: (Davis et al, 2007)

Length of priority level-m busy period:

$$t_m^{n+1} = B_m + \sum_{k \in hep(m)} \left\lceil \frac{t_m^n + J_k}{T_k} \right\rceil C_k$$

B_m : max blocking time for message m due to lower-priority messages

C_k : transmission time for higher- or equal-priority message k

J_k : queuing jitter for higher- or equal-priority message k

T_k : minimum inter-arrival time (period) for higher- or equal-priority message k

$t_m^0 = C_m$: iterative start value

Collision-based communication

CAN real-time protocol: (Davis et al, 2007)

Number of instances of message m to investigate:

$$Q_m = \left\lceil \frac{t_m + J_m}{T_m} \right\rceil$$

t_m : length of priority level-m busy period

J_m : queuing jitter for message m

T_m : minimum inter-arrival time (period) for message m

Due to non-preemptive dispatching, the priority level-*m* busy period may extend into subsequent periods of message *m*.

Collision-based communication

CAN real-time protocol: (Davis et al, 2007)

Queuing delay for instance q of message m:

$$w_m^{n+1}(q) = B_m + qC_m + \sum_{k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{\text{bit}}}{T_k} \right\rceil C_k$$

B_m : max blocking time for message m due to lower-priority messages

C_k : transmission time for higher-priority message k

J_k : queuing jitter for higher-priority message k

T_k : minimum inter-arrival time (period) for higher-priority message k

q : instance $(0,1,...)$ of message m

$w_m^0(q) = B_m + qC_m$: iterative start value

Collision-based communication

CAN real-time protocol: (Davis et al, 2007)

Response time of instance q of message m:

$$R_m(q) = J_m + w_m(q) - q \cdot T_m + C_m$$

J_m : queuing jitter for message m

C_m : transmission time for message m

T_m : minimum inter-arrival time (period) for message m

q : instance $(0,1,\ldots)$ of message m

$w_m(q)$: queuing delay for instance q of message m

Collision-based communication

CAN real-time protocol: (Davis et al, 2007)

A sufficient and necessary test for message m:

$$\max_{q=0\ldots Q_m-1}(R_m(q)) \leq D_m$$

Q_m : number of instances of message m to investigate

$R_m(q)$: response time for instance q of message m

D_m : deadline for message m

Highlighted article:

Read the paper by Davis et al. (RTS Journal, 2007)

Study particularly how this analysis rectifies a serious flaw in an earlier wide-spread analysis for CAN (Section 3)