Department of Computer Science
National Tsing Hua University

CS 5244: Introduction to Cyber Physical Systems

Unit 21: Networking in Embedded Systems: Control Area Network, FlexRay, IEEE 1588

Instructor: Cheng-Hsin Hsu

Acknowledgement: The instructor thanks Profs. Edward A. Lee & Sanjit A. Seshia at UC Berkeley for sharing their course materials

Today's Cars

- *X-by-wire* vs. conventional mechanical and hydraulic systems
- Basics: power locks/door/window/engine start
- Sensors/Actuators: tire, airbags, powertrain, video, radar, and photoelectrics, etc.
- Control/Safety: ABS, EBD/CBC, EBA/BAS/BA, ASR/TCS/TRC, ESP/DSC/VSC, etc.
- Entertainment System
- Auto-Park
- DARPA's Urban Challenge

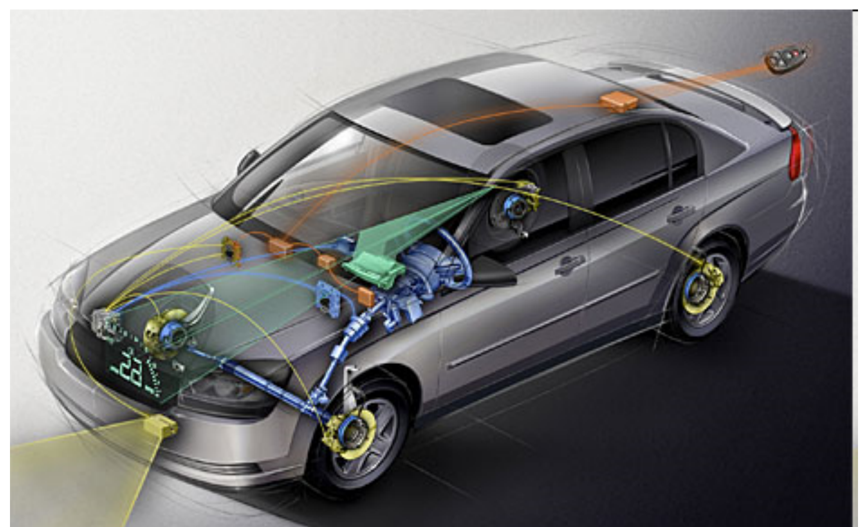


Image: General Motors

Today’s Cars

- Number of Electronic Control Units (ECUs) in a car:
  - Low end: 30 ~ 50 (doors, roof, etc)
  - High end: 70~100
- Lines of code: ~100 million (Future: 200~300 million)
- The radio and navigation system in the current S-class Mercedes-Benz requires over 20 million lines of code alone and that the car contains nearly as many ECUs as the new Airbus A380 (excluding the plane’s in-flight entertainment system).
- Cost of electronics/software: 35% ~ 40% in premium cars (for hybrid it is even higher!)

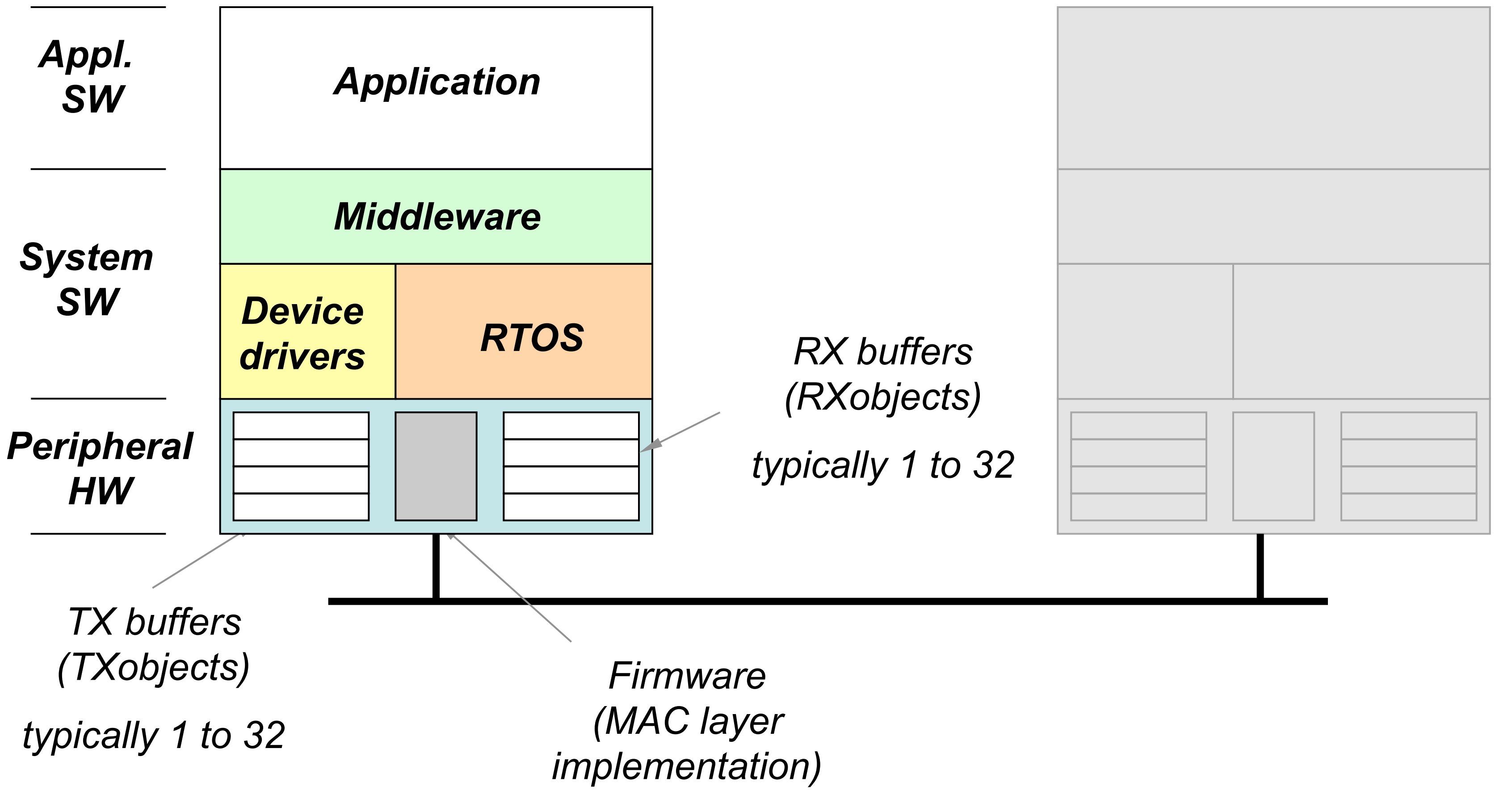
- How can we ensure timely and reliable communication via the “wires”?

[http://www.spectrum.ieee.org/feb09/7649]

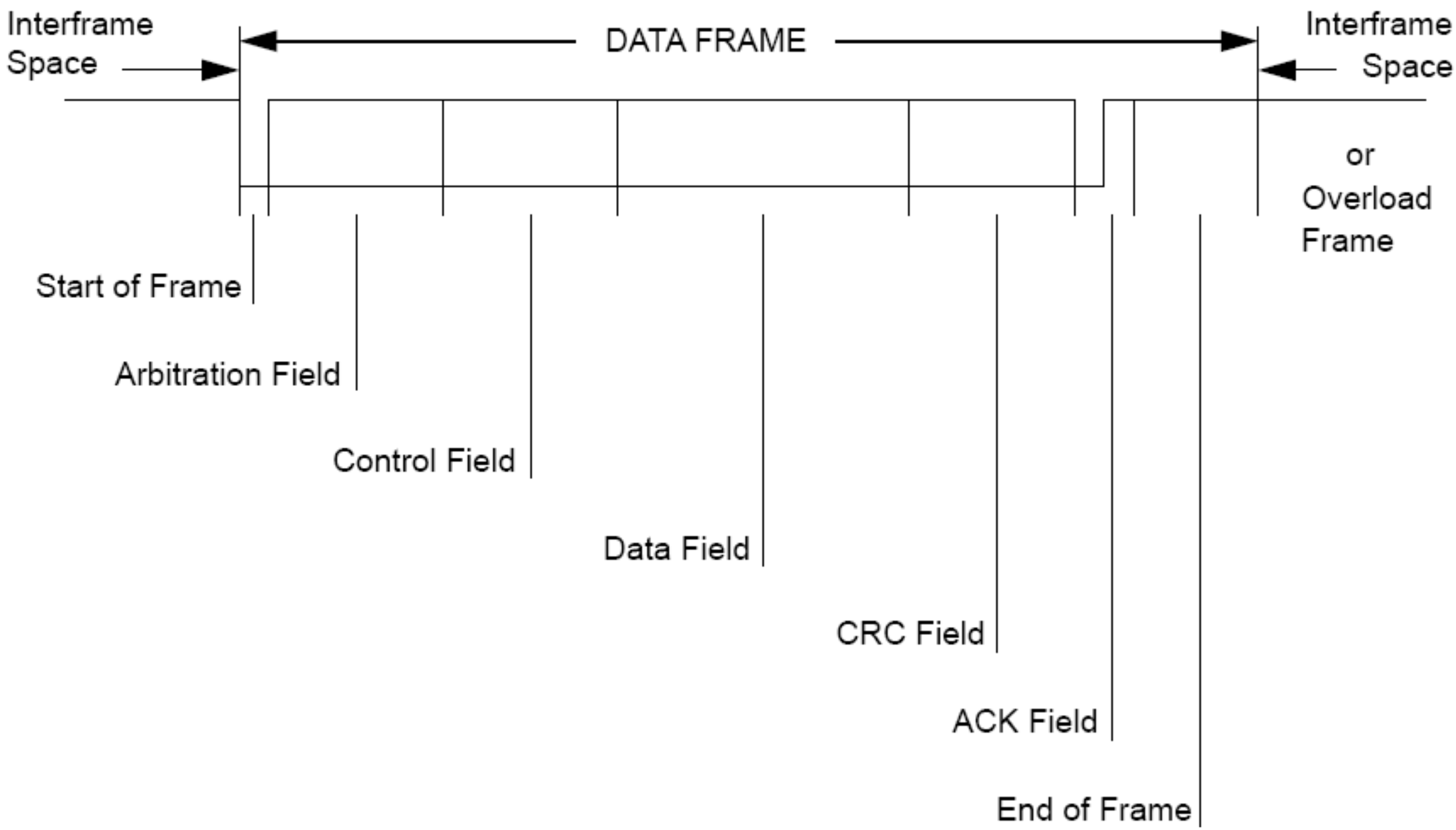
CAN bus

- CAN = Controller Area Network
  - Publicly available communications standard [1] http://www.semiconductors.bosch.de/pdf/can2spec.pdf
- Serial data bus developed by Bosch in the 80s
  - Support for broadcast and multicast comm
  - Low cost
  - Deterministic resolution of the contention
  - Priority-based arbitration
  - Timing analysis for real-time messages
  - Automotive standard but used also in automation, factory control, avionics and medical equipment
  - Simple, 2 differential (copper) wire connection
  - Speed of up to 1Mb/s
  - Error detection and signalling

CAN-based system



CAN bus: Data Frame



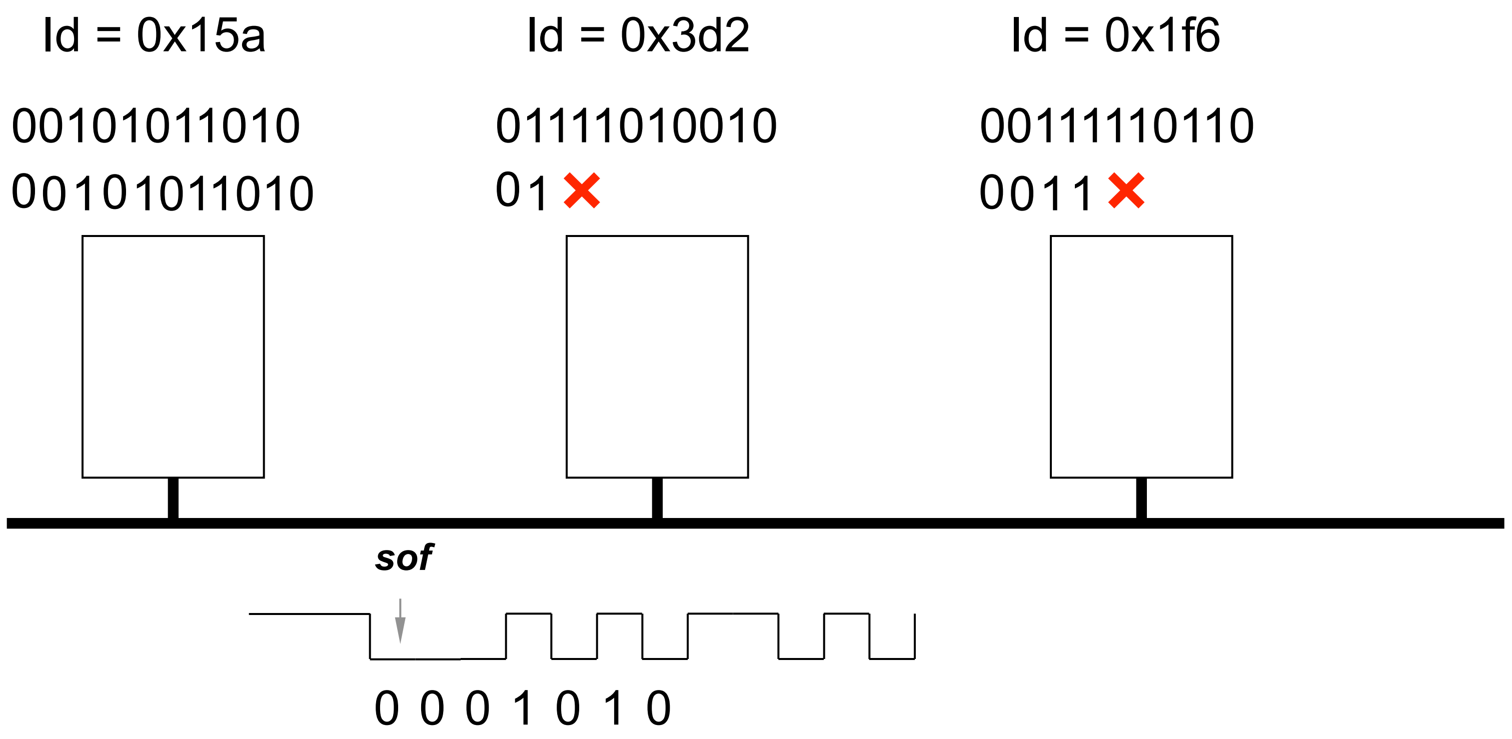
Priority-Based Arbitration

Main points:

All nodes are synchronized on the SOF bit

The bus behaves as a wired-AND

An example …



Priority-Based Arbitration

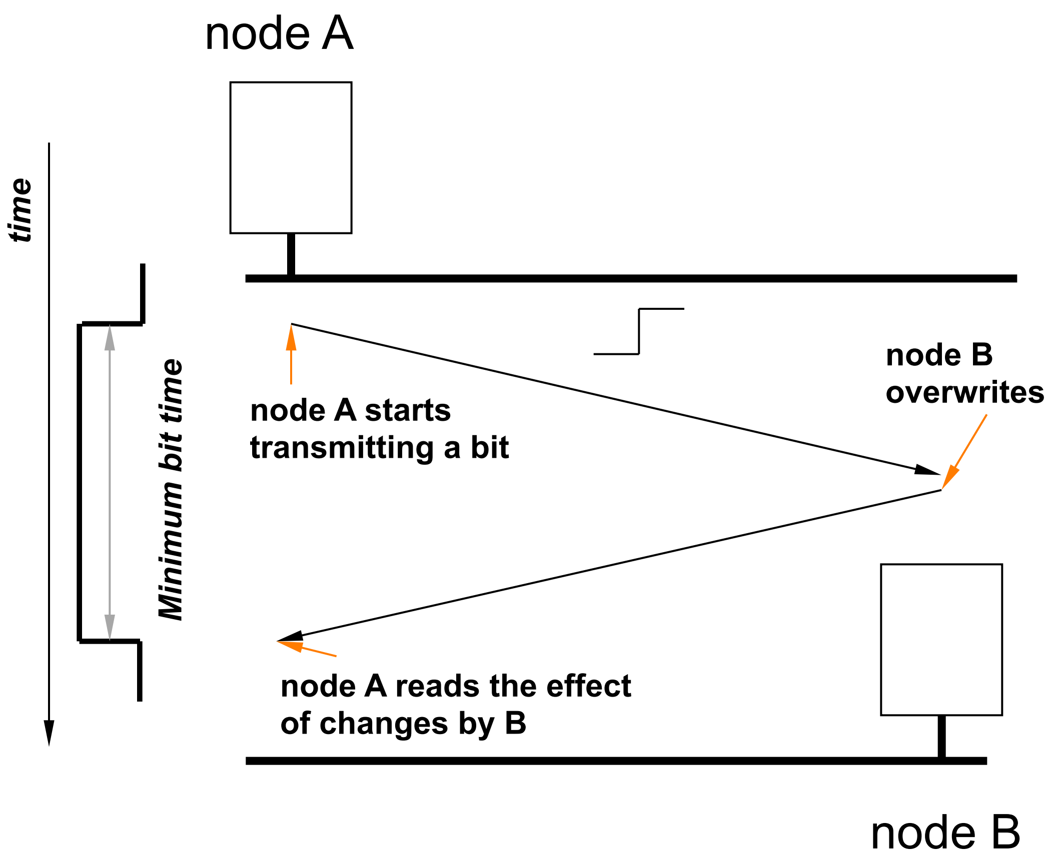
A sender must wait longer than that maximum propagation latency before sending the next bit.

Why?

Priority-Based Arbitration

The type of arbitration implies that the bit time is at least twice the propagation latency on the bus

This defines a relation between the maximum bus length and the transmission speed. The available values are



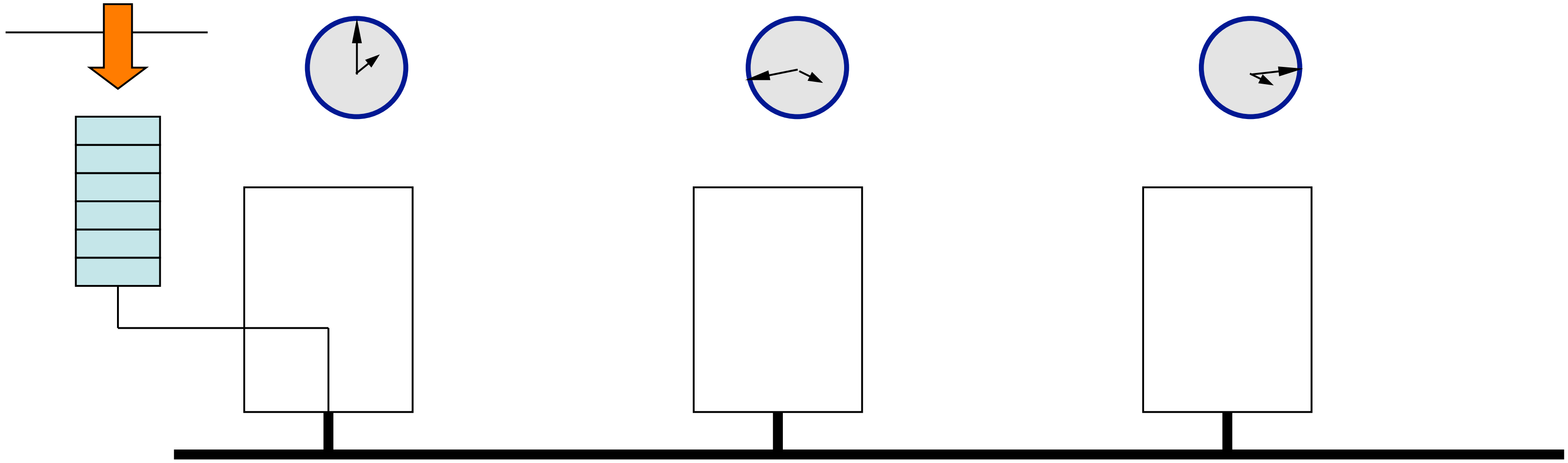
Bit rate	Bus length
1 Mbit/s	25 m
800 kbit/s	50 m
500 kbit/s	**100 m**
250 kbit/s	250 m
125 kbit/s	500 m
50 kbit/s	1000 m
20 kbit/s	2500 m
10 kbit/s	5000 m

Assumptions that Impact Timing

Timing Analysis (and inversions) – Ideal behavior

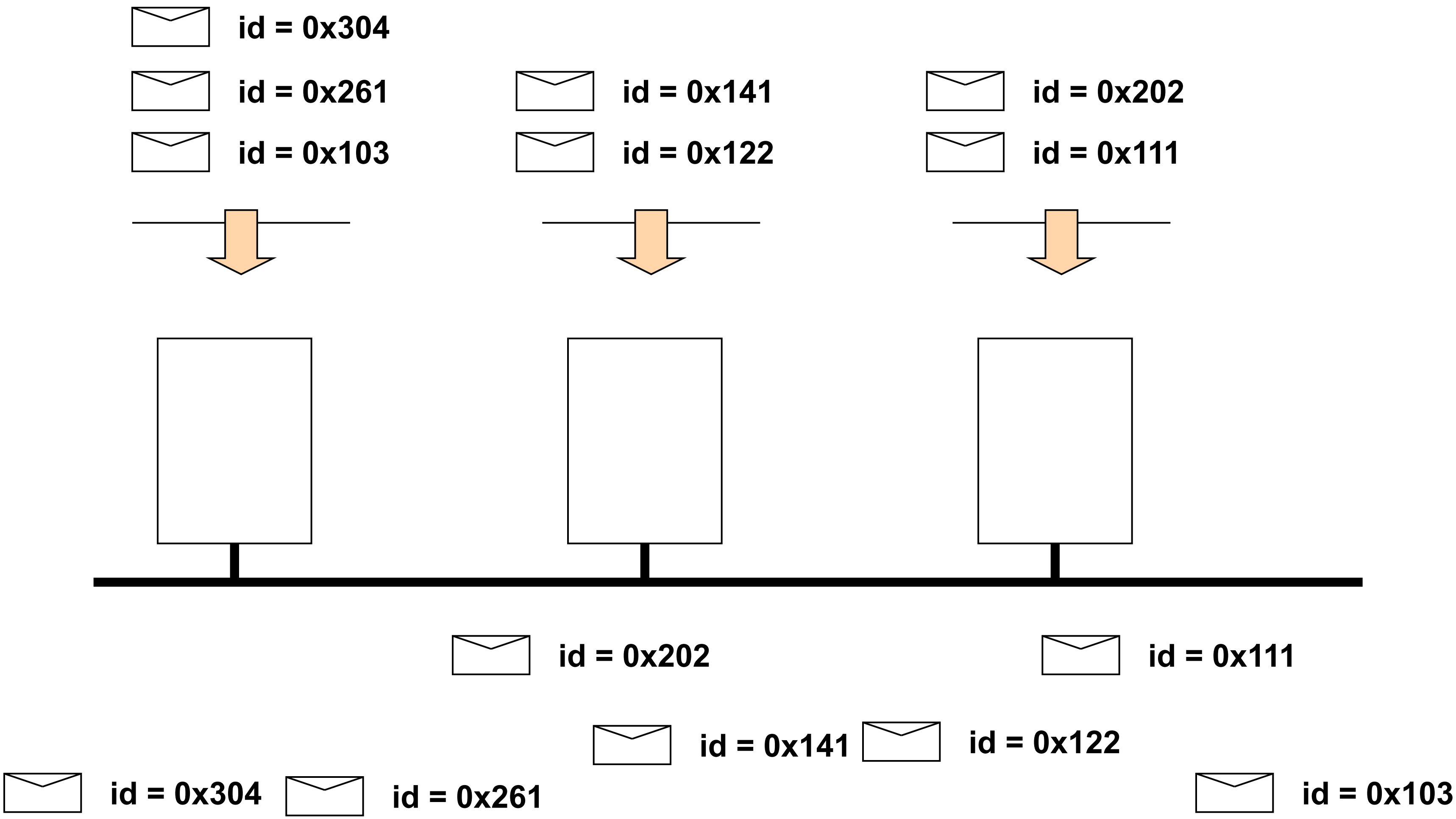
Assumption 3: periodic messages, but no assumption on the message phases

Assumption 1: nodes are not synchronized, but clocks evolve at the same rate

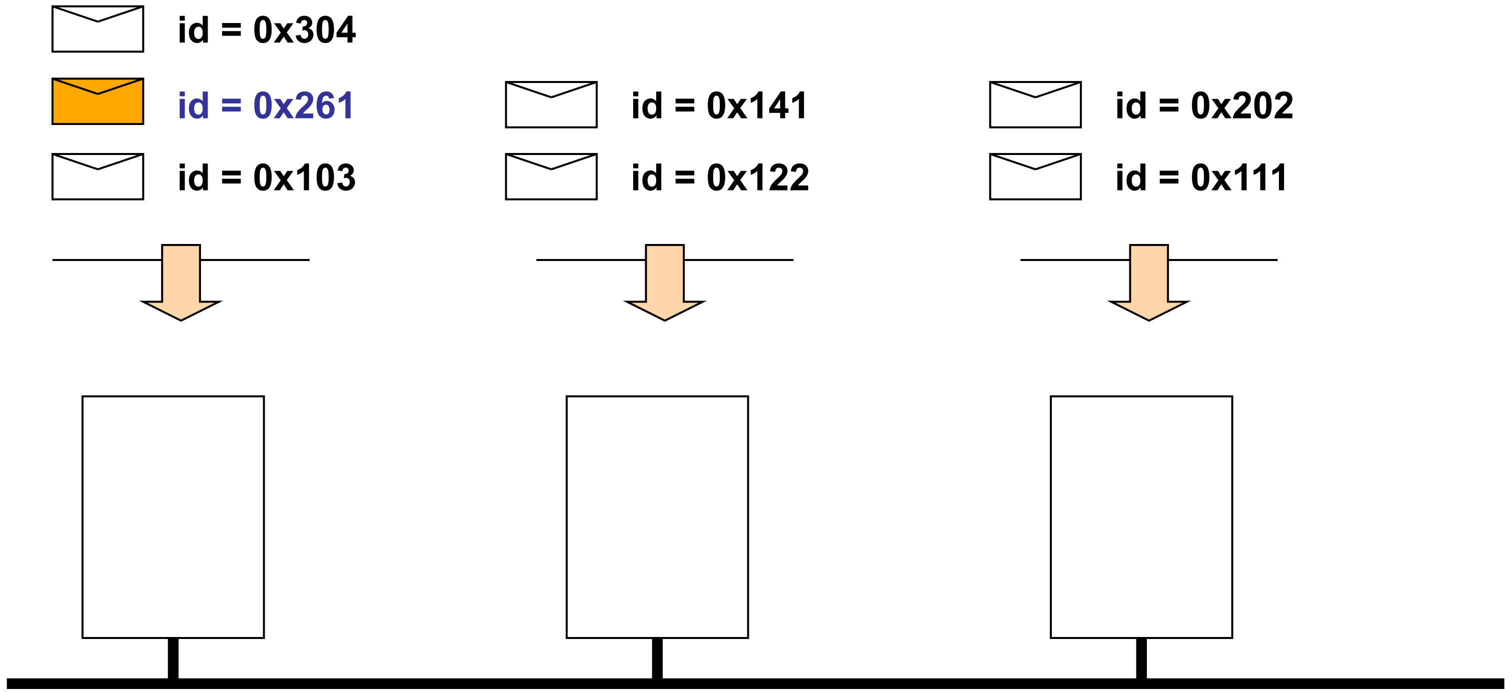


Assumption 2: messages are always transmitted by nodes based on their priority (ID) – ideal priority queue of messages

Timing based on Priorities – Ideal Behavior

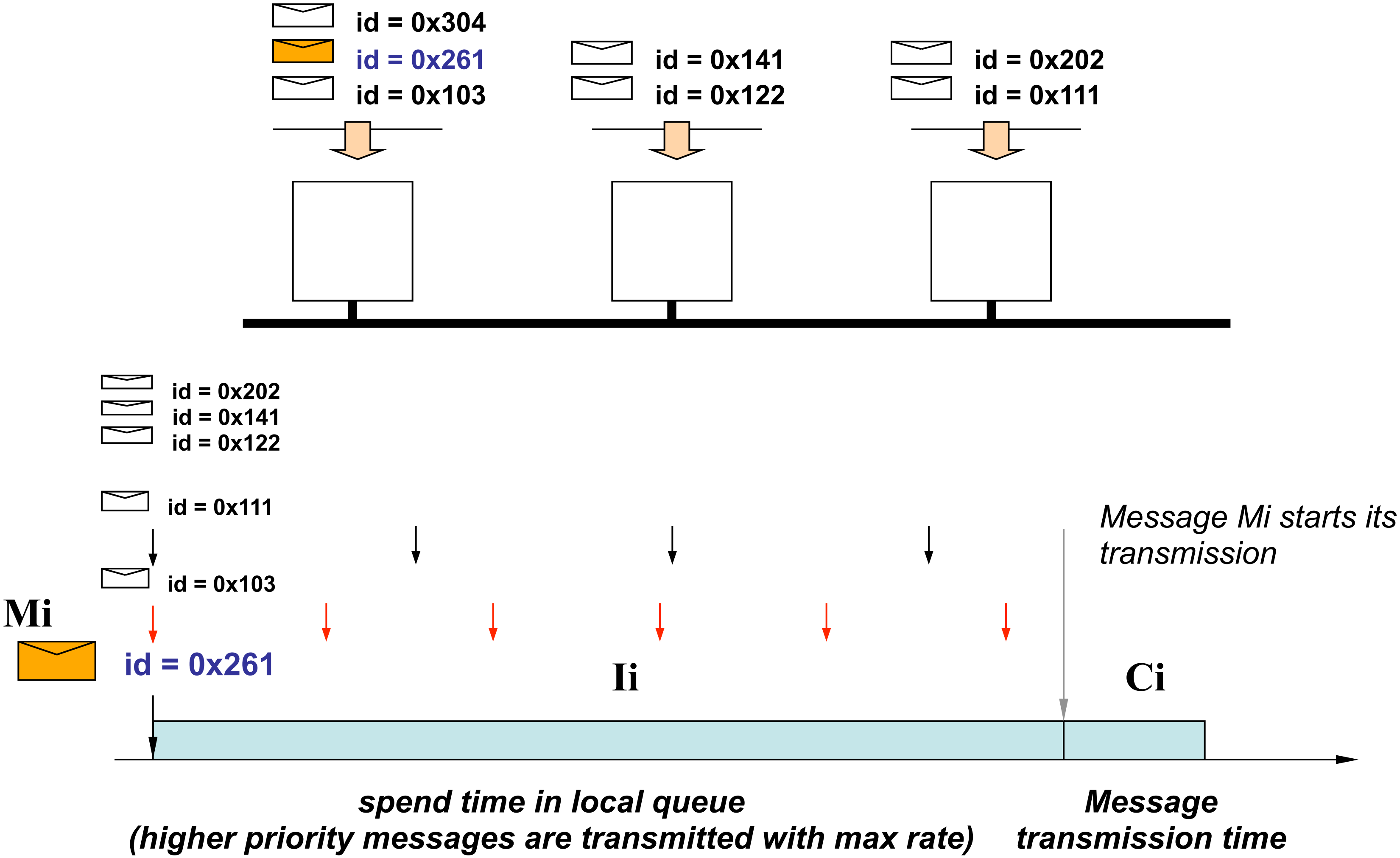


Timing Analysis --- Worst-Case Reponse Time



Critical instant theorem: for a preemptive priority based scheduled resource, the worst case response time of an object occurs when it is released together with all other higher priority objects and they are released with their highest rate

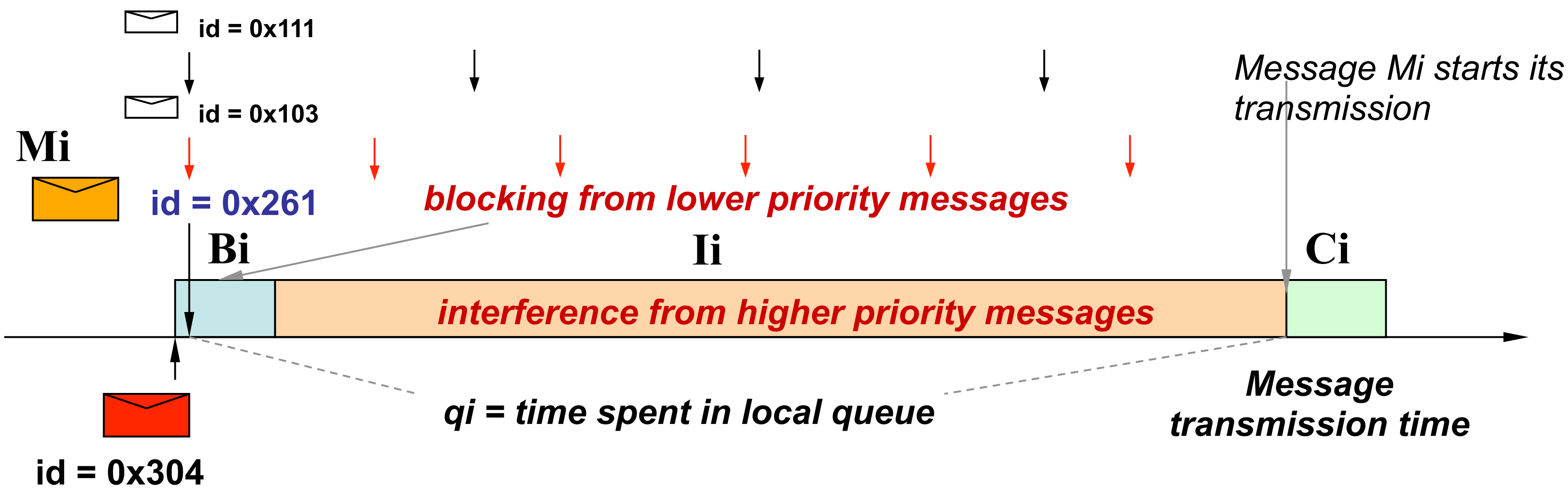
Timing Analysis --- Worst-Case Reponse Time



Timing Analysis under Ideal Conditions

Timing Analysis – worst case latency – Ideal behavior [2]

The transmission of a message cannot be preempted



$$I_i = \sum_{j \in hp(i)} I_{i,j}$$

$$q_i = B_i + I_i$$

$$w_i = q_i + C_i$$

$$I_{i,j} = \left\lceil \frac{q_i}{T_j} \right\rceil C_j$$

$$q_i = B_i + \sum_{j \in hp(i)} \left\lceil \frac{q_i}{T_j} \right\rceil C_j$$

Fixed point formula: solved iteratively by setting qi(0)=0 until the minimum solution is found

CAN bus

In reality, this analysis can give optimistic results!

A number of issues need to be considered …

- …
- Availability of TxObjects at the adapter
- Finite copy time between the queue and the TxObjects

Adapters typically only have a limited number of TXObjects or RxObjects available

CAN bus

A number of issues need to be considered …

– …
– Availability of TxObjects at the adapter

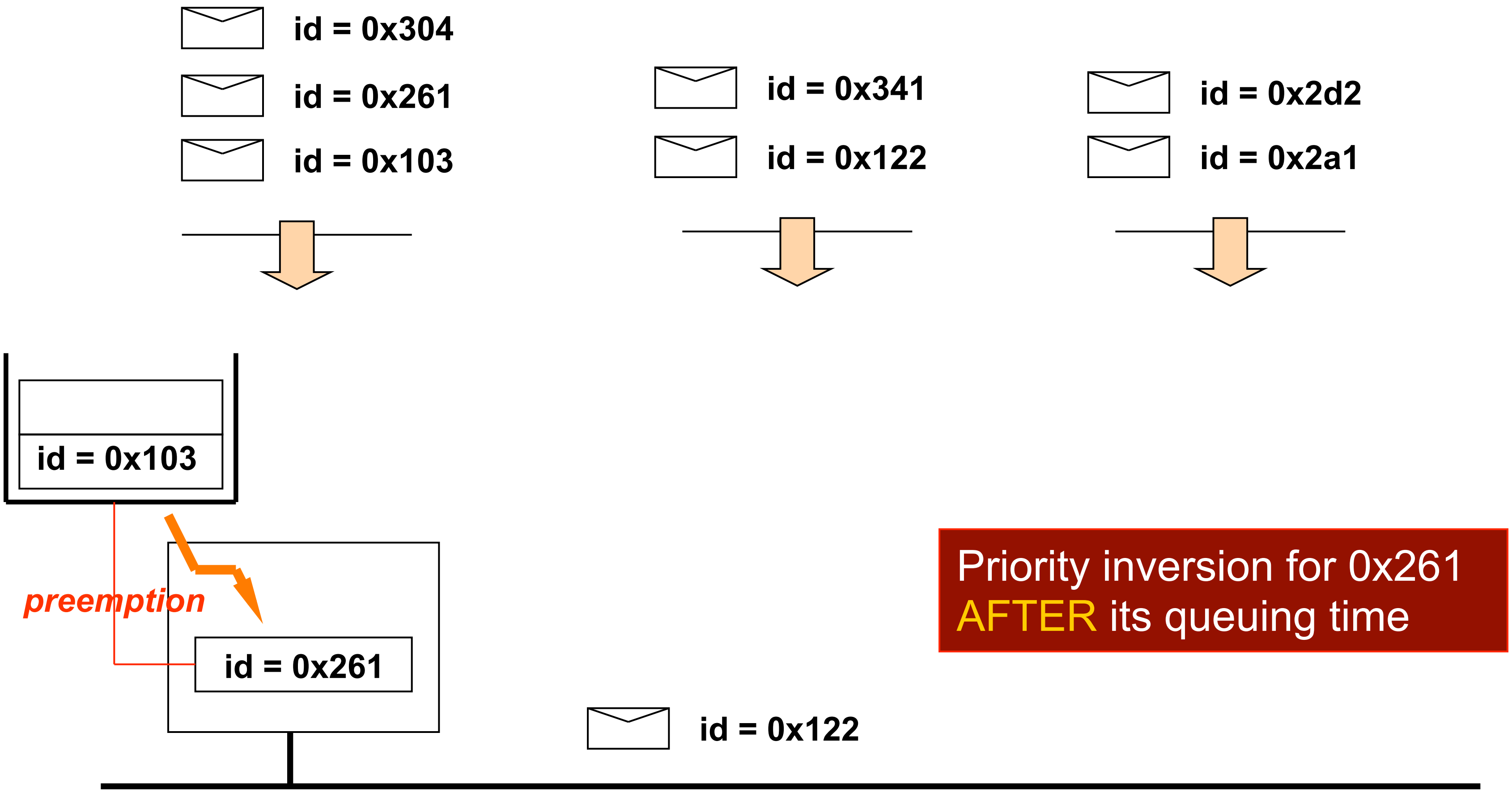
- Let's check the controller specifications!

Model	Type	Buffer Type	Priority and Abort
Microchip MCP2515	Standalone controller	2 RX - 3 TX	lowest message ID, abort signal
ATMEL AT89C51CC03 AT90CAN32/64	8 bit MCU w. CAN controller	15 TX/RX msg. objects	lowest message ID, abort signal
FUJITSU MB90385/90387 90V495	16 bit MCU w. CAN controller	8 TX/RX msg. objects	lowest buffer num. abort signal
FUJITSU 90390	16 bit micro w. CAN controller	16 TX/RX msg. objects	lowest buffer num. abort signal
Intel 87C196 (82527)	16 bit MCU w. CAN controller	14 TX/RX + 1 RX msg. objects	lowest buffer num. abort possible (?)
INFINEON XC161CJ/167 (82C900)	16 bit MCU w. CAN controller	32 TX/RX msg. objects (2 buses)	lowest buffer num., abort possible (?)
PHILIPS 8xC592 (SJA1000)	8 bit MCU w. CAN controller	one TX buffer	abort signal

CAN bus

What happens if only one TxObject is available?

– Assuming preempatbility of TxObject



Violation of Priority-based Queuing

Period (ms)

ECU

Frame ID

Message name

msg1	110	ECU1	10
msg2	120	ECU1	10
msg3	124	ECU1	10
msg4	170	ECU1	500
msg5	308	ECU1	100
msg6	348	ECU1	250
msg7	410	ECU1	100
msg8	510	ECU1	500

1.857316	1	110	Rx	d 8 00 09 BF 00 00 06 00 00
1.857548	1	120	Rx	d 8 03 85 23 83 06 EA 03 85
1.857696	1	170	Rx	d 3 01 00 86
1.858256	1	124	Rx	d 5 00 03 83 03 85
……				
3.877361	1	110	Rx	d 8 00 09 C4 00 00 06 00 00
3.877597	1	120	Rx	d 8 03 83 23 81 06 EA 03 82
3.877819	1	308	Rx	d 7 00 80 2A 00 00 00 AD
3.878309	1	124	Rx	d 5 00 03 81 03 83
……				
4.017366	1	110	Rx	d 8 00 09 C4 00 00 06 00 00
4.017600	1	120	Rx	d 8 03 85 23 80 06 EA 03 81
4.017768	1	348	Rx	d 4 08 48 43 FF
4.018312	1	124	Rx	d 5 00 03 80 03 85

Message 0x170, 0x308, 0x 348 transmitted before 0x124

Possible Effect of Interrupt Service

Message name	Frame ID	ECU	Period (ms)
msg1	150	ECU1	12.5
msg2	151	ECU1	12.5
msg3	320	ECU1	100
msg4	520	ECU1	100

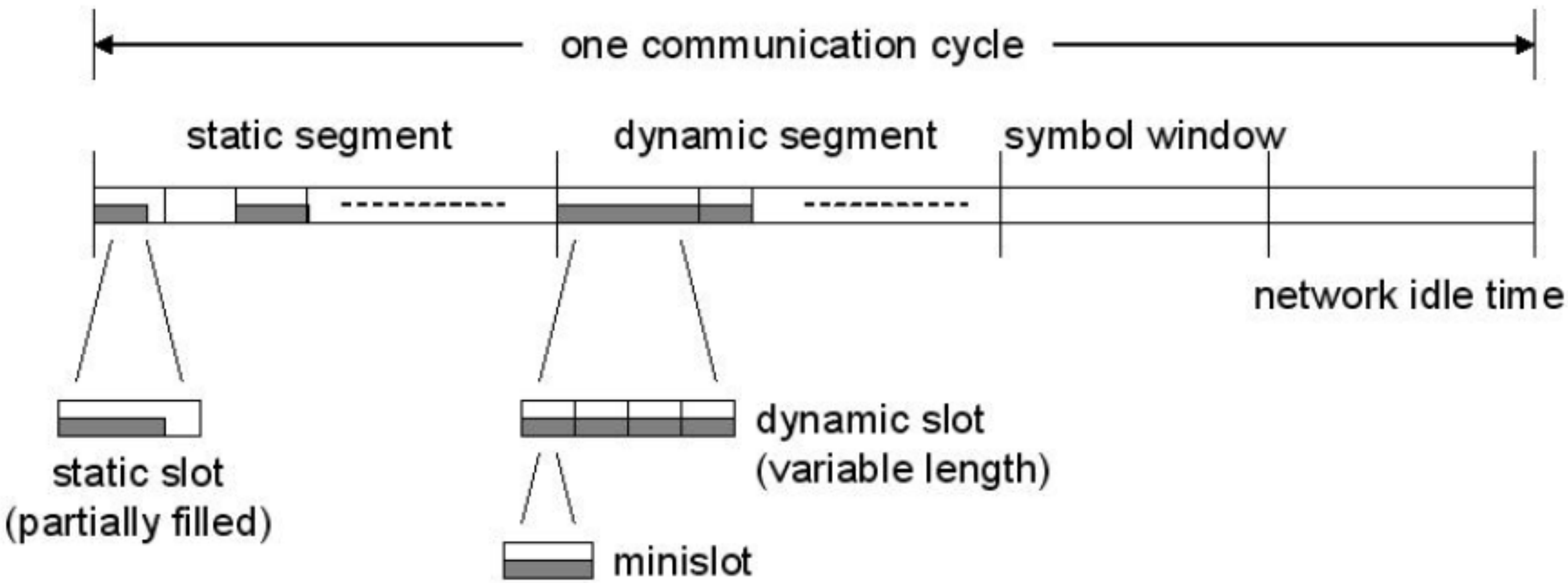
```
0.222236 1  150    Rx   d 8 40 00 09 60 3F FF F6 9F
0.222527 1  380    Rx   d 8 09 42 20 00 70 40 FC BF
0.222766 1  151    Rx   d 8 00 FF 09 22 00 00 0F 3F
……
0.297743 1  150    Rx   d 8 C0 00 09 60 3F FD F6 9D
0.297989 1  410    Rx   d 8 00 00 00 96 2B 00 00 00
0.298229 1  151    Rx   d 8 00 FF 09 25 00 00 0F 3F
……
0.322497 1  150    Rx   d 8 40 00 09 60 3F FF F6 9F
0.322733 1  388    Rx   d 8 21 12 68 19 00 00 DC 80
0.322978 1  151    Rx   d 8 00 FF 09 21 00 00 0F 3F
```

Message 0x380, 0x410, 0x 388 transmitted before 0x151

FlexRay

- Being developed by a consortium of automotive makers and 1-tier suppliers.
- Successor to CAN, higher bit rate, more ECUs, and more reliable
  - FlexRay: max 10 Mbps
  - CAN: max 1 Mbps (but protocol itself has over 40% overhead)
- Allow both *time-triggered* and *event-triggered* communication
- Good clock synchronization (distributive) with built-in fault tolerance

FlexRay – Format of Time Division for Mesg Transmission



FlexRay has a static segment with guaranteed slots for ECUs to transmit (reduce arbitration overhead)

FlexRay Specification v2.1

Bibliography

[1] CAN Specification, Version 2.0. Robert Bosch GmbH. Stuttgard, 1991, http://www.semiconductors.bosch.de/pdf/can2spec.pdf

[2] K. Tindell, H. Hansson, and A. J. Wellings, Analysing real-time communications: Controller area network (can),' Proceedings of the 15th IEEE Real-Time Systems Symposium (RTSS'94), vol. 3, no. 8, pp. 259--263, December 1994.

[3] A. Meschi M. Di Natale M. Spuri Priority Inversion at the Network Adapter when Scheduling Messages with Earliest Deadline Techniques , Euromicro Conference on Real-time systems, L'Aquila, Italy 1996.

[4] R. Davis, A. Burns, R. Bril, and J. Lukkien. Controller area network (can) schedulability analysis: Refuted, revisited and revised. In RTN06, Dresden, Germany, July 2006.

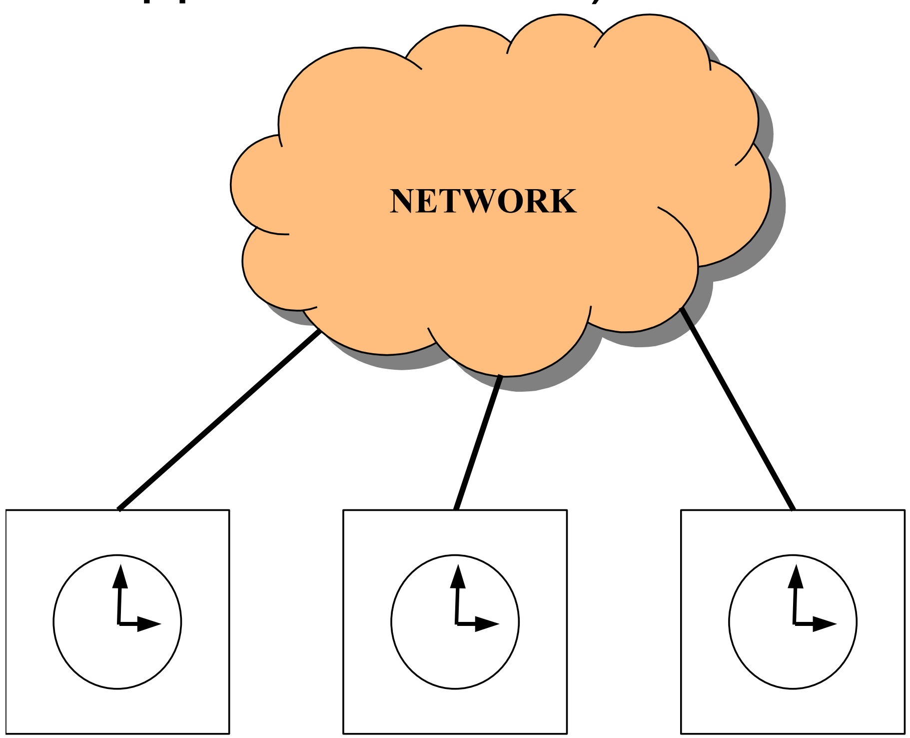
Major time distribution systems used in embedded systems

1. NTP- c <1985, ~10ms
2. GPS- c 1972, operational in 1993,~100ns: (Glonass, Galileo)
3. IRIG-B- c 1960, ~1-10 us
4. IEEE 1588-2008 – c 2002, ~20ns on Ethernet
5. Proprietary or controlled protocols, e.g. FlexRay(c ~2000), TTP(c ~1993), TTE(c ~2005)…

John Eidson

Purpose of IEEE 1588

- IEEE 1588 is a protocol designed to synchronize real-time clocks in the nodes of a distributed system that communicate using a network
  - It does not say how to use these clocks (this is specified by the respective application areas)



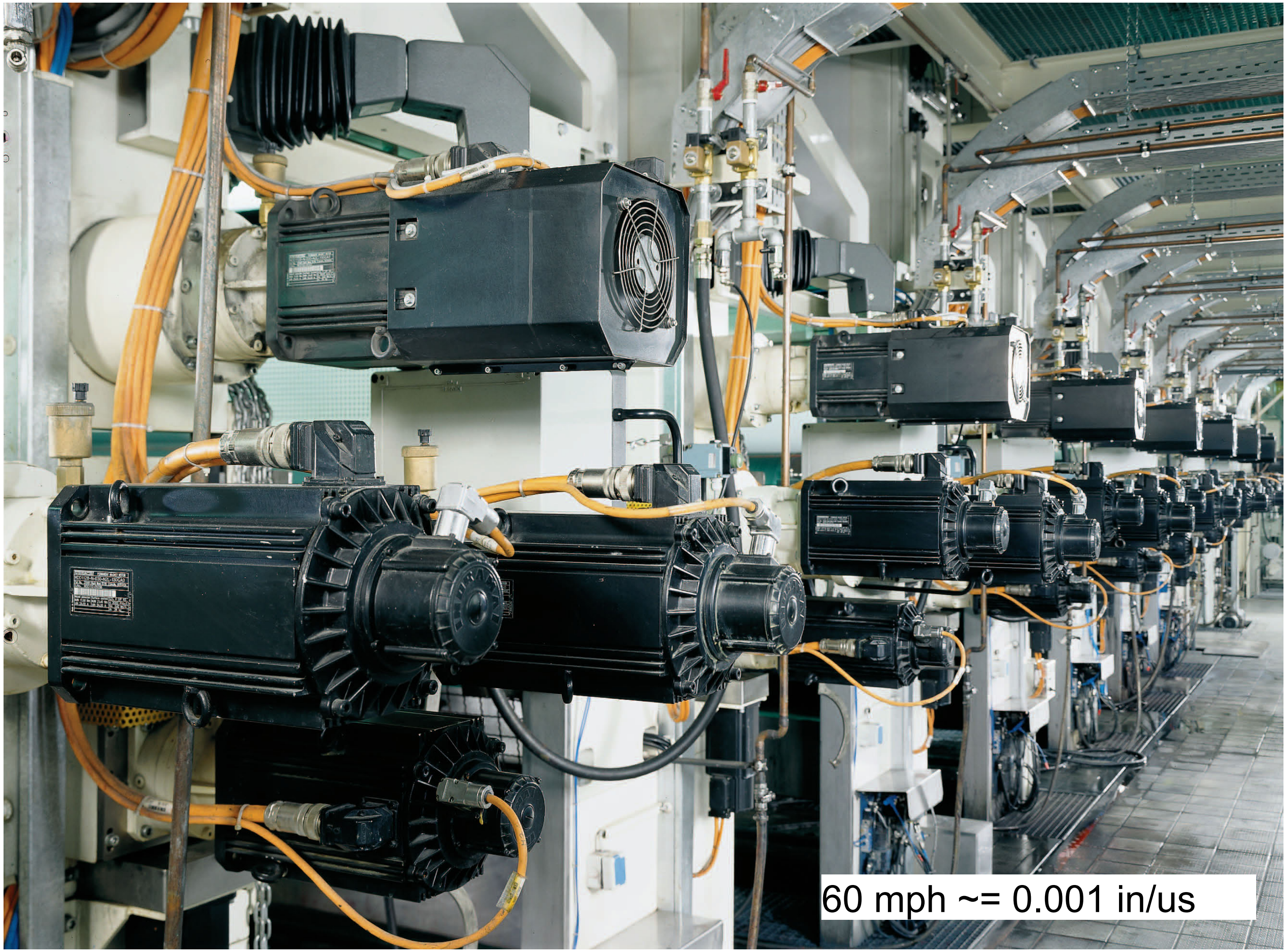
Where is IEEE 1588 being (or likely to be used)?

1. Power generation (>50K nodes in service)
2. Industrial automation (esp. motion control)
3. Telecom (cellular backhaul initially- already field installations)
4. Audio visual systems (as IEEE 802.1AS a specialization of 1588)
5. Military, aerospace, instrumentation (flight qualification, surveillance, data acquisition)
6. Other nascent applications

John Eidson

High speed printing

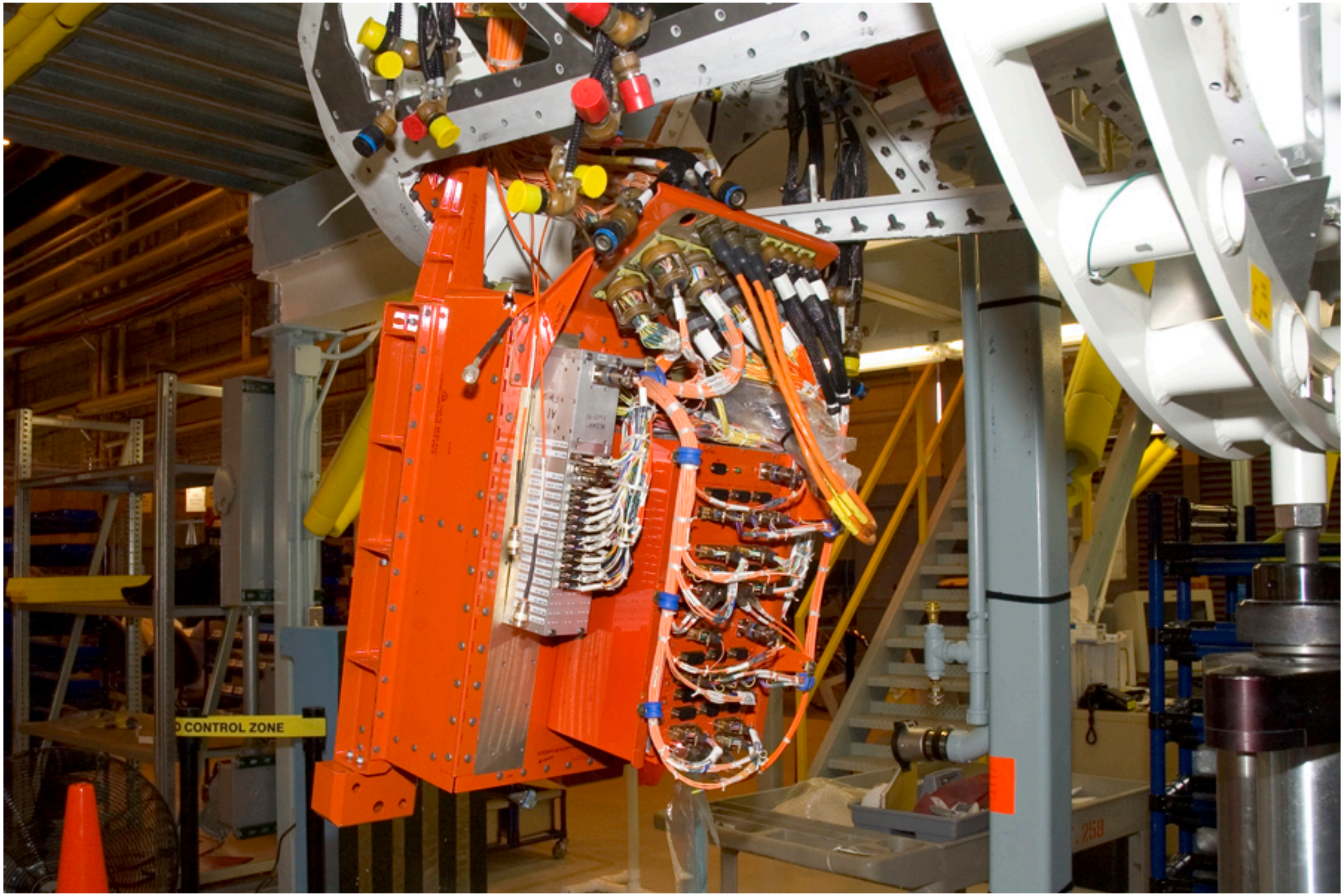
Courtesy of Bosch-Rexroth.



60 mph ~= 0.001 in/us

John Eidson

IEEE 1588 enabled flight test instrumentation in the forward fuselage of a test aircraft. (Data acquisition)



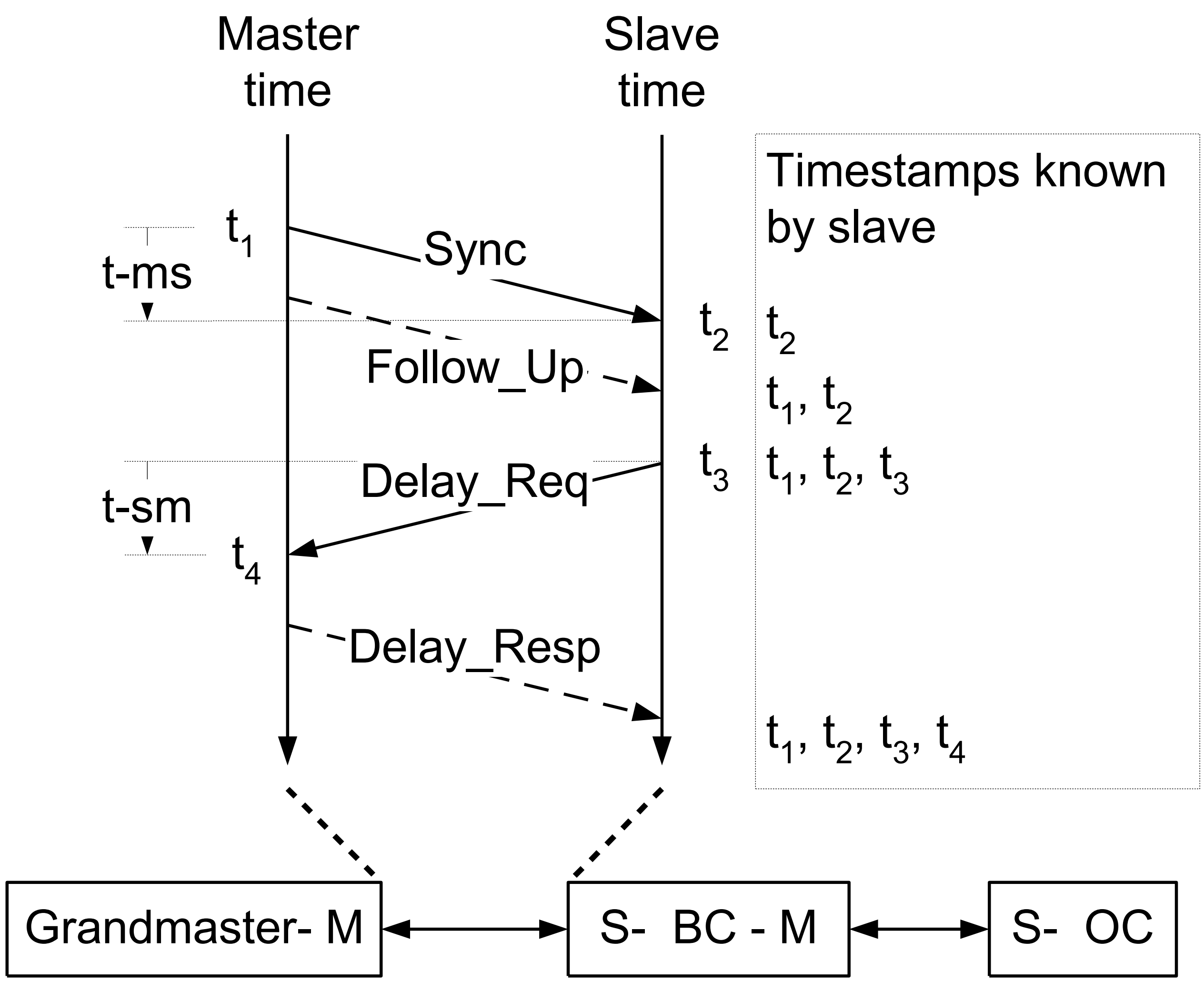
Courtesy of Teletronics

John Eidson

Telecommunications Applications

- **Cellular backhaul is the major telecom application to date**. Metro-Ethernet in field trial. Femtocells beginning.
- Companies involved (partial list):
  – Nokia-Siemens, Brilliant, Semtech, Zarlink, …

Synchronization Basics – Delay Request-Response Mechanism



t-ms: time from master to slave
t-sm: time from slave to master

John Eidson

Synchronization Basics – Delay Request-Response Mechanism - 2

Offset = slave clock – master clock

M-S difference = t-ms = offset + M-S prop delay

S-M difference = t-sm = -offset + S-M prop delay

Under the assumption that the link is symmetric

- M-S prop delay = S-M prop delay
- Offset = [(t-ms) – (t-sm)]/2 = $[(t_2 - t_1) - (t_4 - t_3)]/2$
- Propagation delay = [(t-ms) + (t-sm)]/2

  $= [(t_2 - t_1) + (t_4 - t_3)]/2$

Can rewrite the offset as

- Offset = $t_2 - t_1$ – (propagation delay) = (t-ms) – (propagation delay)

John Eidson

Websites

General IEEE 1588 site: contains product pointers, conference records, general guidance, standards related

http://ieee1588.nist.gov/

ISPCS (International IEEE Symposium on Precision Clock Synchronization) site: Conference on IEEE 1588 and related subjects

http://www.ispcs.org/