# Lecture 15: Real-Time Networks and Networked Control Systems

[These slides]

- Background
- Real-Time Networks
- Collapsed OSI Model
- RT-Network Examples
  - CAN
  - TTP
- Networked Control Systems

*These slides are partly based on material from Luis Almeida, Universidade do Porto, Portugal*

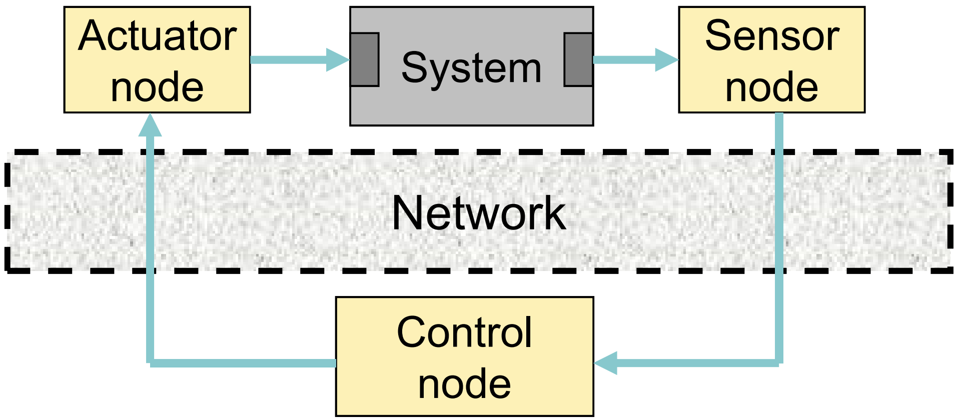
# Background

**Distributed architectures** are common in many applications:

- Industrial automation
- Transportation systems (planes, cars, trucks, trains, …)
- Multimedia systems (surveillance, monitoring, video on demand, …)

In many cases with **critical timeliness** and **safety** requirements

Control loops are often closed over networks (= **networked control**)



# Background

Motivations for distributed architectures:

- Processing closer to the data source/sink
  - “Intelligent” sensors and actuators
- Dependability
  - Error-containment within nodes
- Composability
  - System composition by integrating subsystems
- Scalability
  - Easy addition of new nodes with new or replicated functionality
- Maintainability
  - Modularity and simple node replacement
  - Simplification of the cabling

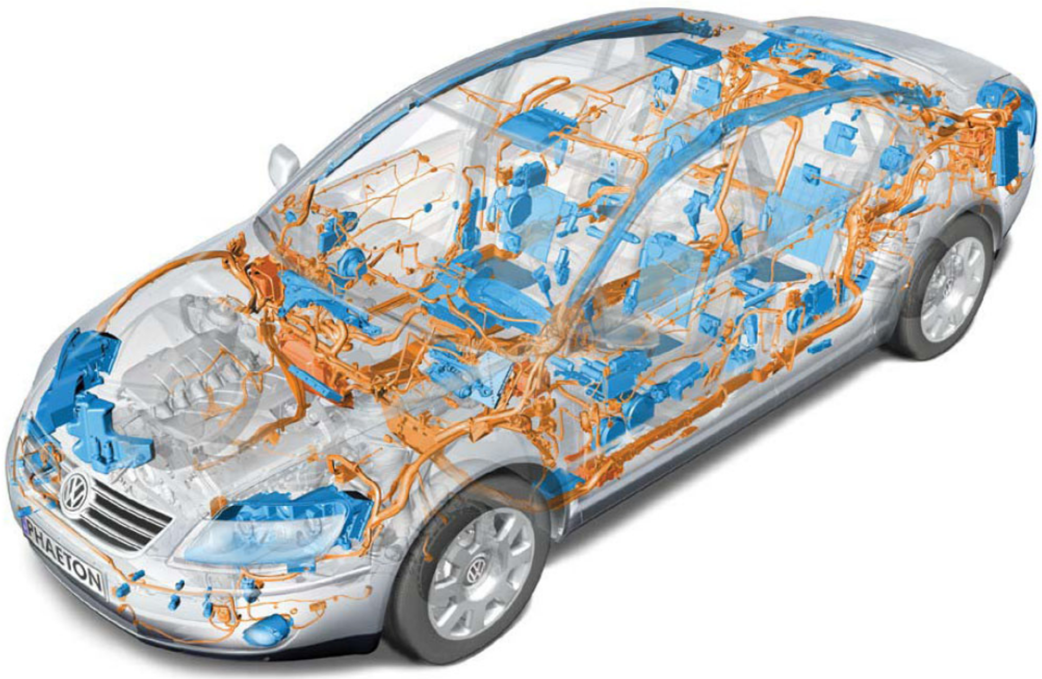
# Background

Different networks with real-time capabilities are aimed at different application domains, e.g.

- ATINC629, SwiftNet, SAFEbus (avionics)
- WorldFIP, TCN (trains)
- CAN, TT-CAN, FlexRay, MOST (cars)
- ProfiBus, WorldFIP, P-Net, DeviceNet (automation)
- Firewire, USB (multimedia, general purpose)

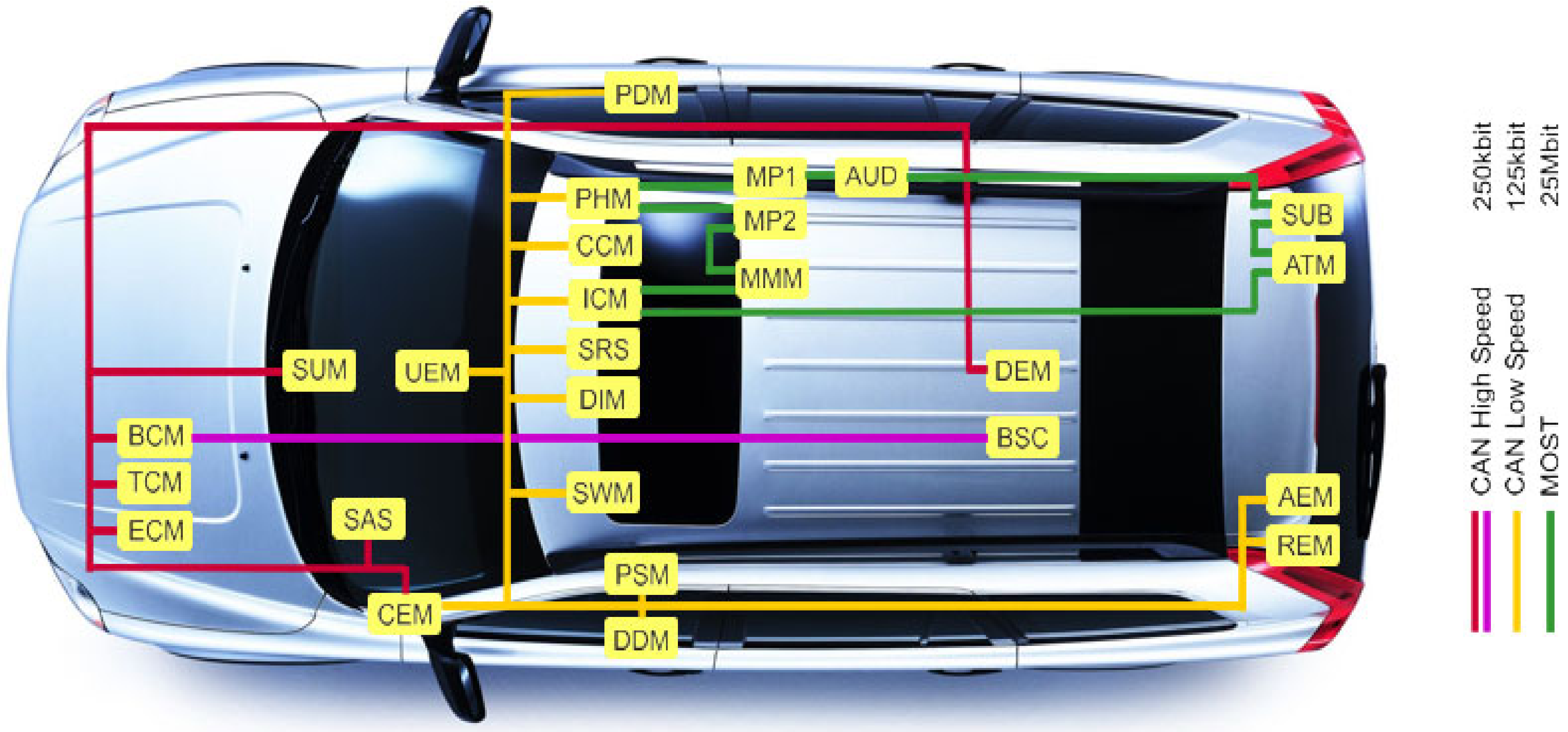
# Example: VW Phaeton

- 11,136 electrical parts
- Total 61 ECUs (Electronic Controller Units = CPUs)
- 35 ECUs connected by 3 CAN buses sharing
  - 2500 signals
  - in 250 CAN messages
- Optical bus for high bandwidth infotainment data




**The VW Phaeton**
Adapted from (Loehold, WFCS2004)

# Example: Volvo XC 90



# Contents

- Background
- **Real-Time Networks**
- Collapsed OSI Model
- RT-Network Examples
  - CAN
  - TTP
- Networked Control Systems

# Requirements

Typical requirements in real-time networks:

- Efficient transmission of **short data** (few bytes)
- **Periodic** transmission (control, monitoring) with **short periods** (ms), **low latency**, and **small jitter**
- Fast transmission (ms) of **aperiodic requests** (alarms, commands, …)
- Transmission of **non-real-time data** (configuration information, log data, …)
- Multicasting as well as unicasting (peer to peer)

# Messages and Packets

- A **message** is a **unit of information** that should be transferred at a given time from a sender to one or more receivers
- Contains both **data** and **control information** that is relevant for the proper transmission of the data (e.g., sender, destination, checksum, …)
- Some networks automatically break large messages into smaller packets (**fragmentation/reassembly**)
- A **packet** is the smallest unit of information that is transmitted

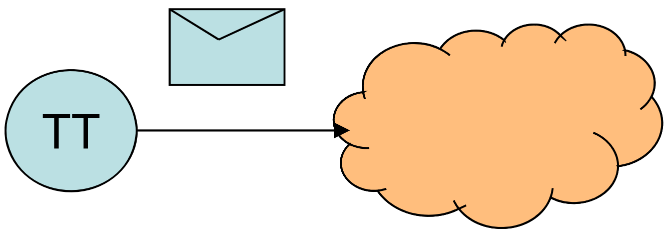
# Real-Time Messages

- A real-time message is associated with a **deadline**
  - Soft or hard
- Real-time messages can have **event** or **state semantics:**
  - **Events** are perceived changes in the system state.
    All events are significant for the state consistency across sender and receiver.
  - **Event messages** must be queued at the receiver and removed upon reading. **Correct order** in delivery must be enforced.
  - **State messages** (containing state data) can be read many times and overwrite the values of the previous message concerning the same real-time entity.

# State vs Event Semantics – Example

## State message

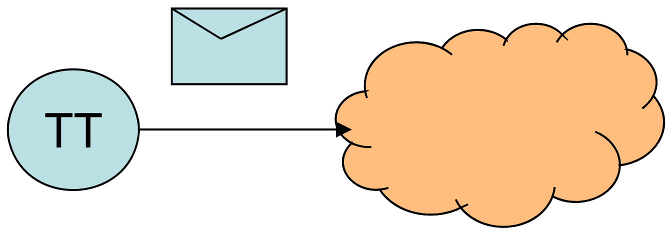
Temperature is 15°C




Temperature
sensor node

## Event message

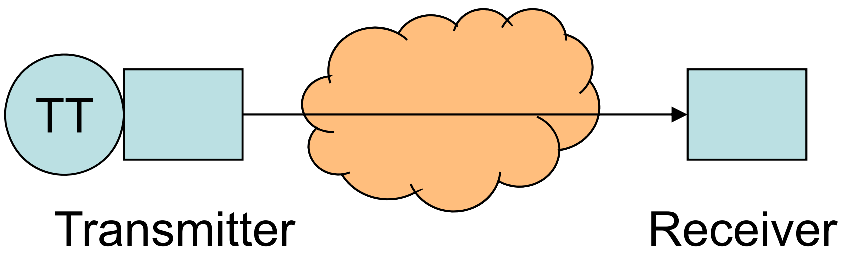
Temperature raised by 2°C

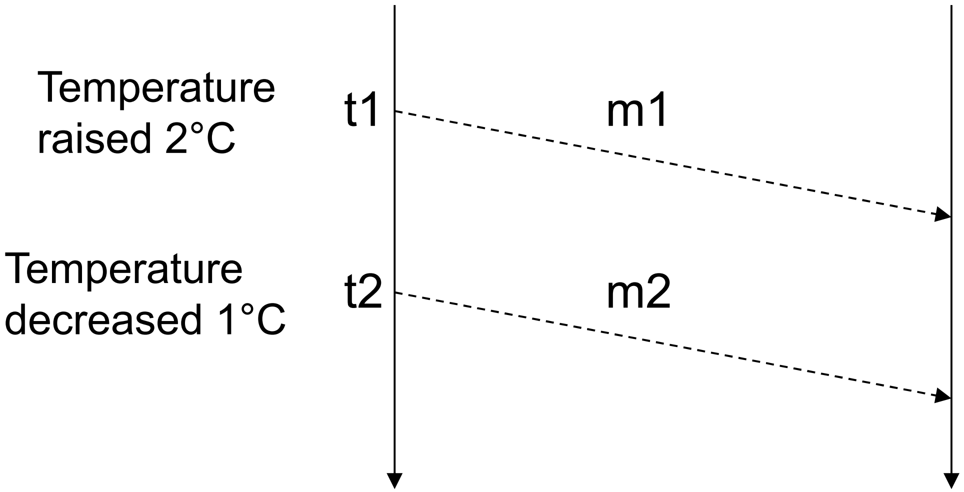



Temperature
sensor node

# Event-Triggered Networks

- Messages are sent asynchronously, upon **events**
- Often, **event semantics** are used for the messages



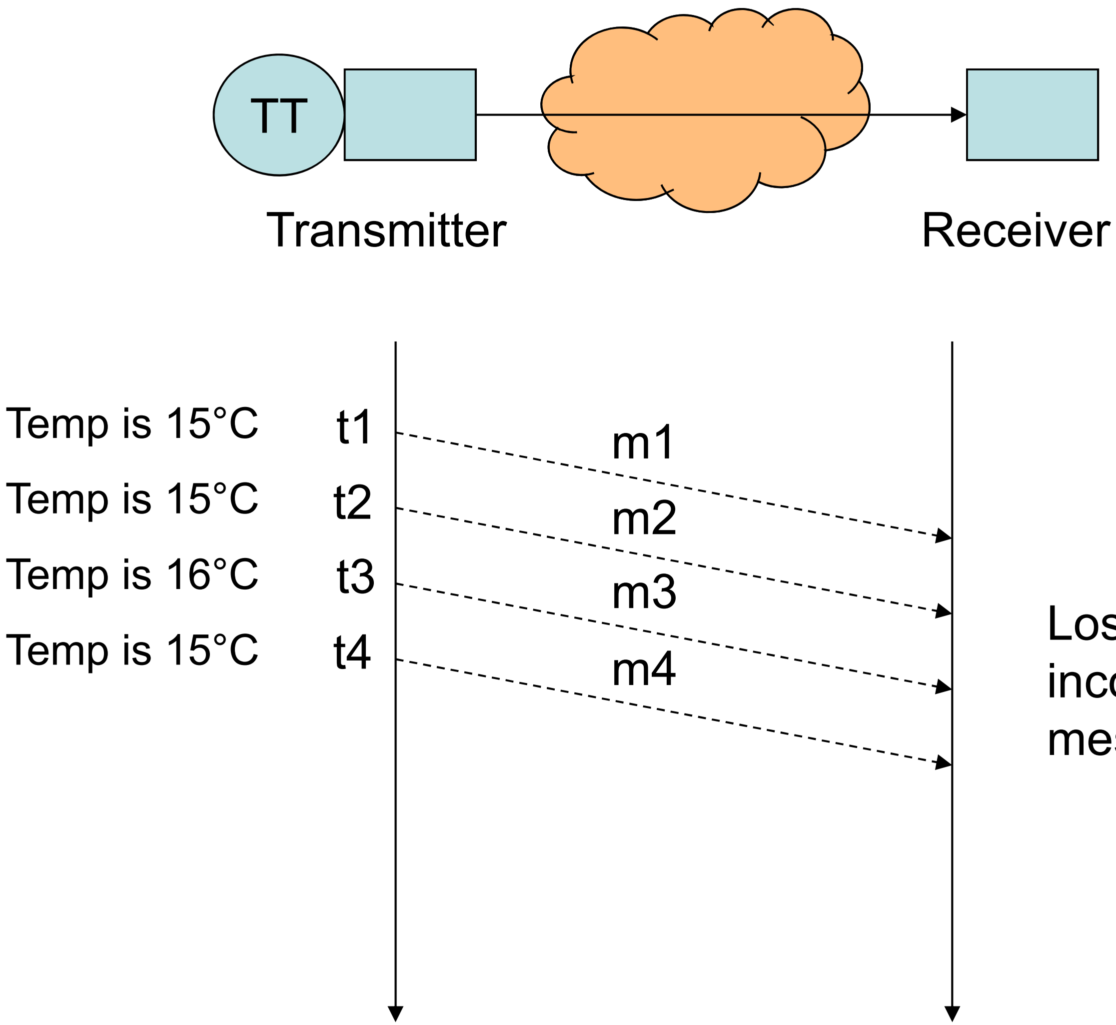




What if m1 is lost?

# Time-Triggered Networks

- In time-triggered networks, there is a notion of **network time**
  - All clocks are globally synchronized
- Messages are sent at **predefined time instants,** typically using **state semantics**
  - Receivers have a **periodic refresh** of the system state
- The network load is predetermined

# Time-Triggered Network




Losing a message gives inconsistency only until next message arrives

# Event vs Time Triggered Networks

**Time-triggered networks:**

- More deterministic
  - All transmission instants are predefined
  - Fault-tolerance mechanisms are easier to design
- Less flexible in reacting to errors
  - Retransmissions often not possible because the schedule is fixed
  - A lost message is not recovered until the next period of the message stream
- Less flexible with respect to changes
  - Everything must be known at design time and very little can be changed dynamically (cp. static cyclic CPU scheduling)

# Event vs Time Triggered Networks

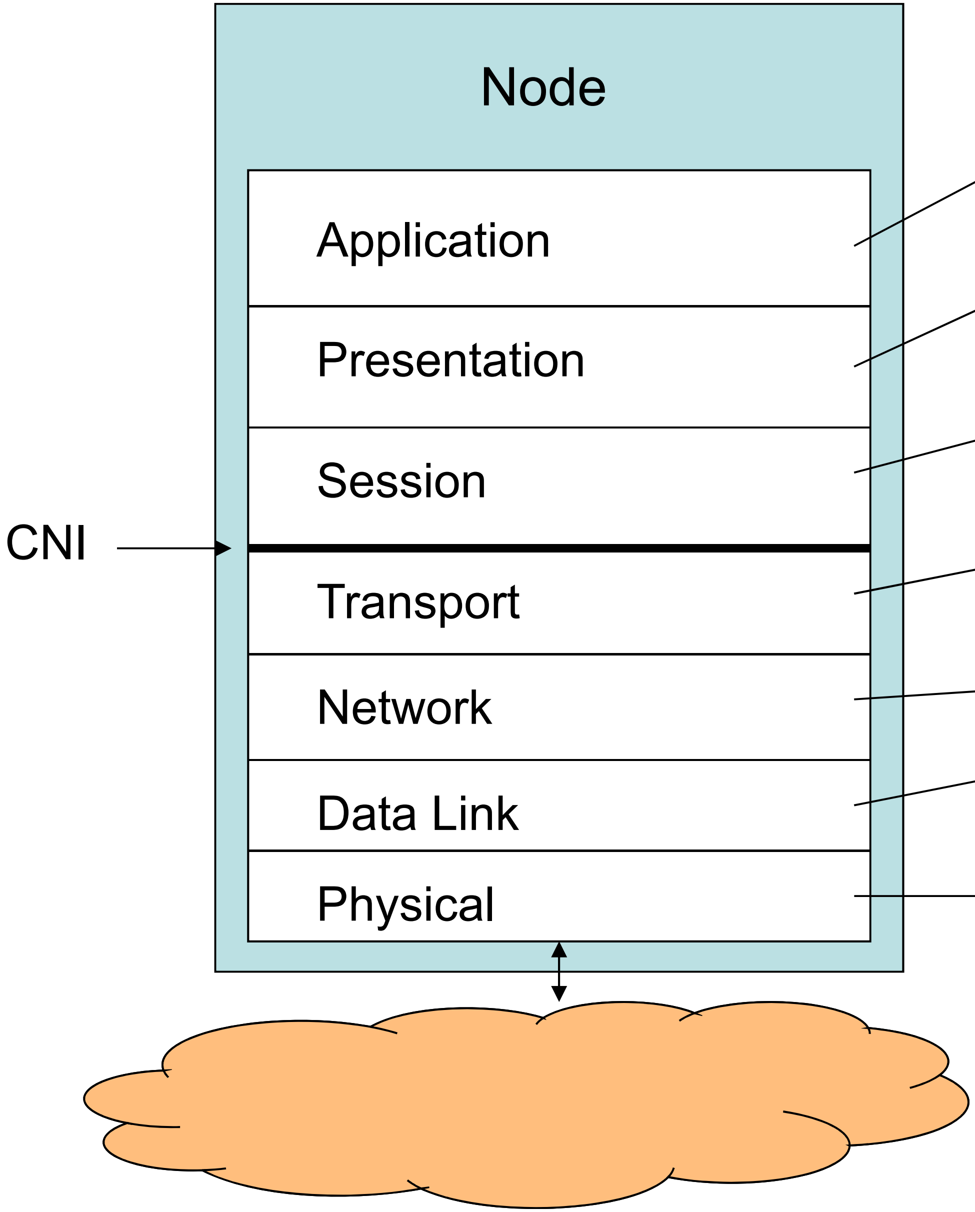
**Event-triggered networks:**

- Lower level of determinism
  - Events can occur at any time (flexible)
  - MAC protocol may be needed
  - Harder to give hard real-time guarantees
- More complex fault-tolerance schemes
- More flexible with respect to transmission errors
  - Retransmissions can be carried out immediately

# Contents

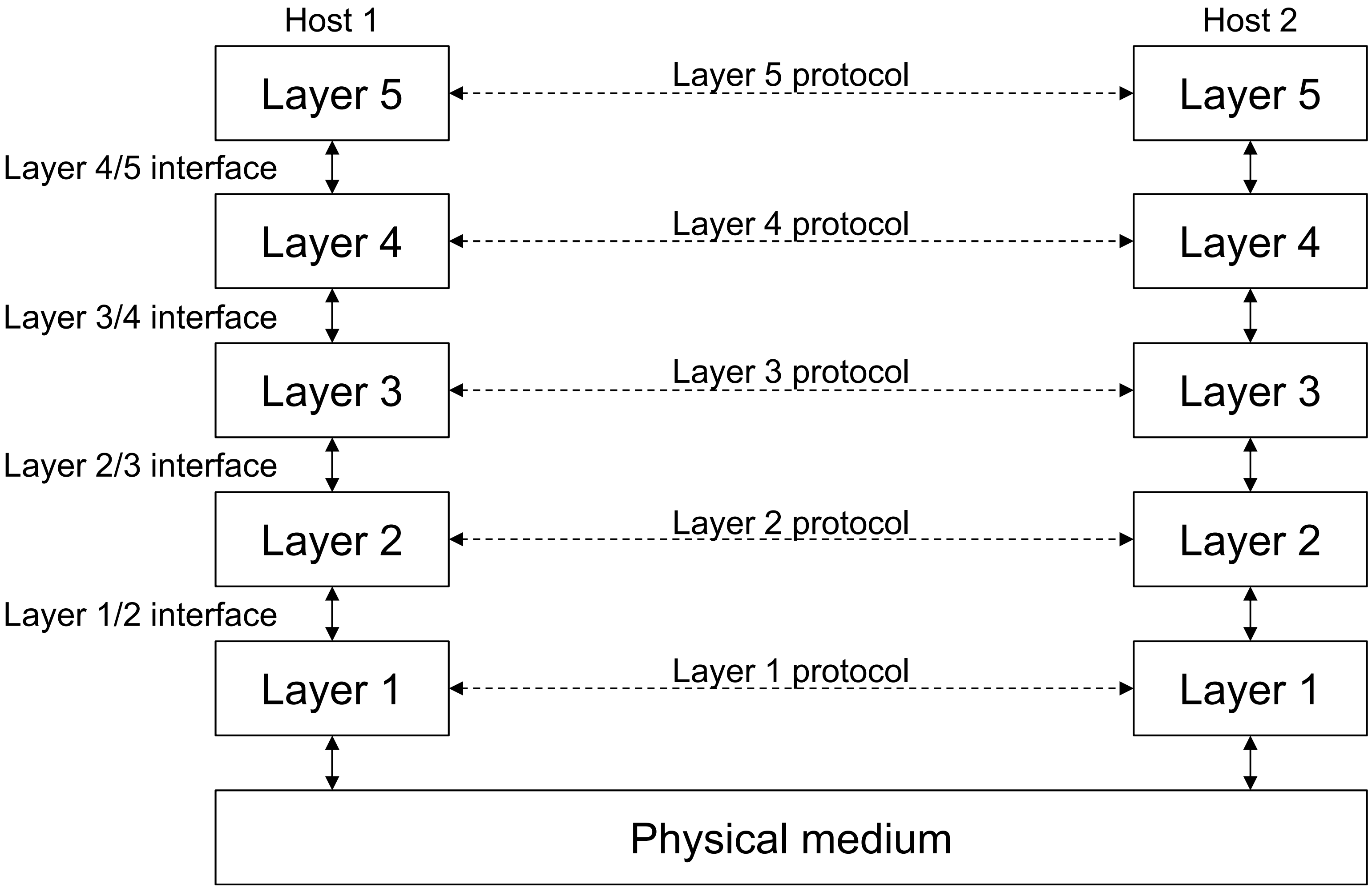
- Background
- Real-Time Networks
- **Collapsed OSI Model**
- RT-Network Examples
  - CAN
  - TTP
- Networked Control Systems

# The OSI Model

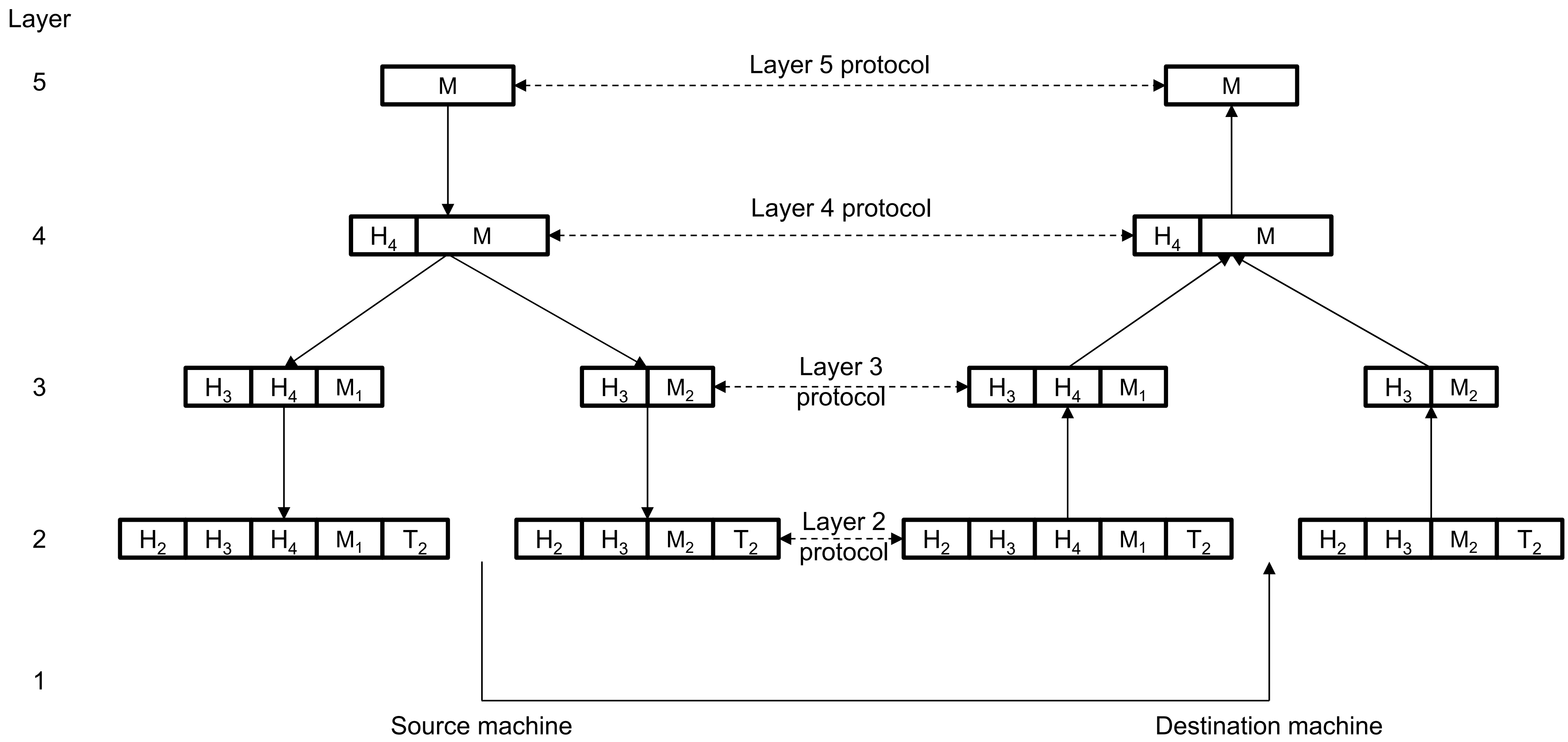



- Common services for a class of apps. (FTP, HTTP, ...)
- Data semantics (conversion, compression, encryption, …)
- Remote actions (name services, directory, billing, ...)
- End-to-end comm. control, e.g., (TCP, UDP)
- Routing, logical addressing
- Physical addressing, medium access control
- Topology, medium, bit-encoding

# Protocol Stack



# Layer Interfaces



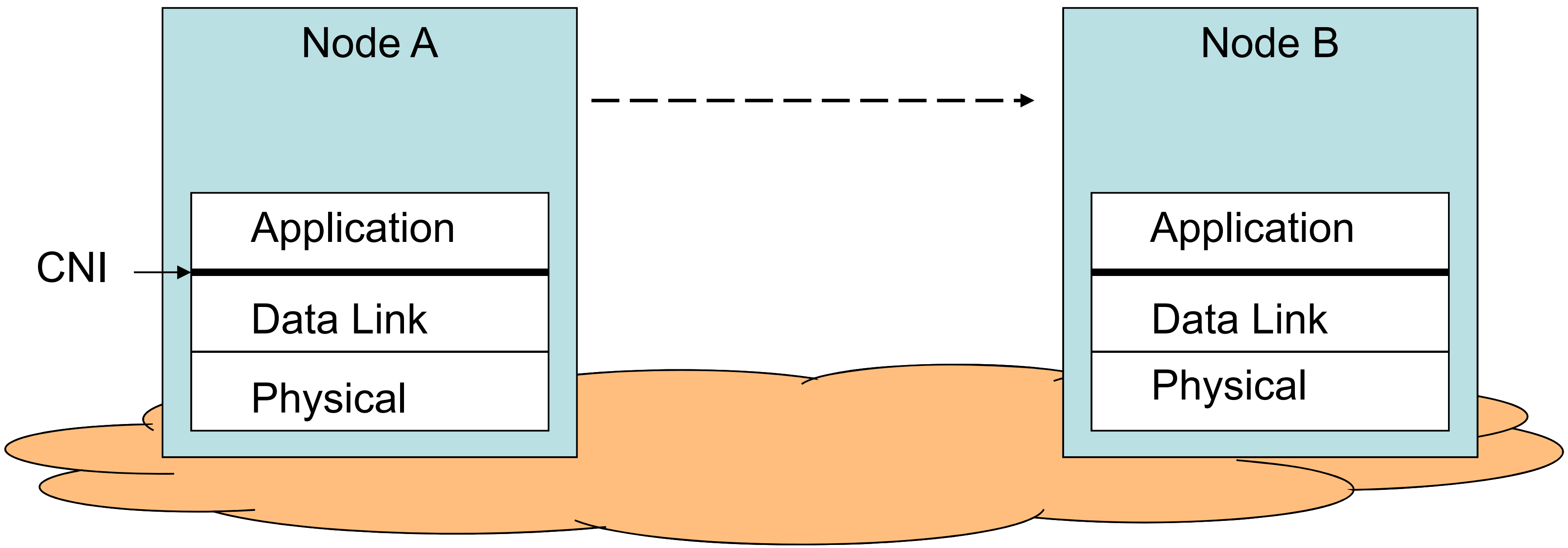

Very large computing and communication **overhead** for short real-time data traffic!

# Real-Time Protocol Stack

- The end-to-end communication delay must be bounded
  - All services at all layers must be time-bounded
  - Requires appropriate time-bounded protocols
- The seven OSI layers impose a considerable overhead
- Many real-time networks
  - are dedicated to a well-defined application (no need for presentation)
  - use a single network domain (no need for routing)
  - use short messages (no need to fragment/reassemble)

# Collapsed OSI Model

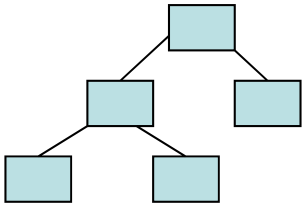
- Application accesses the Data Link directly
- Other layers may be present but are not fully stacked
- In industrial automation these networks are called **fieldbuses**



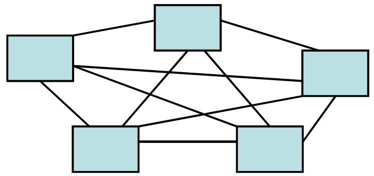
# Physical Layer

- **Topology**
- **Medium**
- Coding of digital information
- …

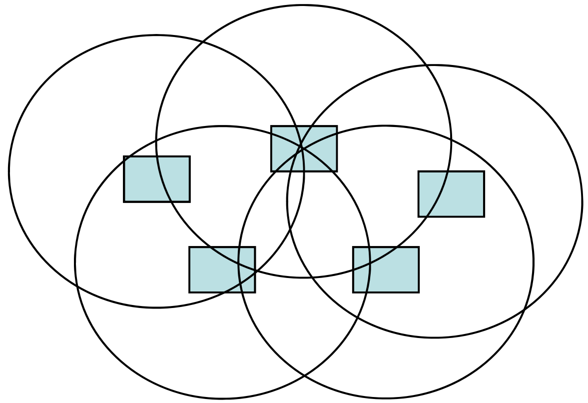
# Physical Layer: Topology



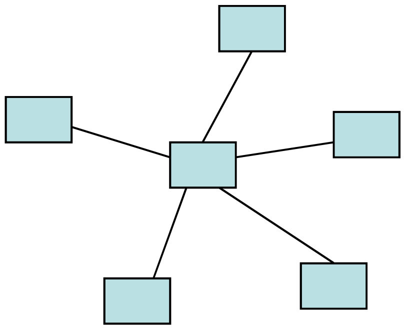
Tree



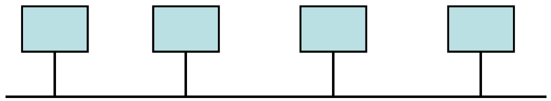
Mesh (wired)



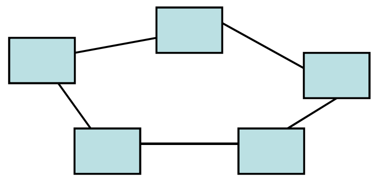
Mesh (wireless)



Star



Bus



Ring

# Physical Layer: Medium

- Copper wiring
  - Cheaper cables and interfaces (+), suffers EMI(-)
- Optical fibres
  - Immune to EMI, wide bandwidth, low attenuation (+), expensive cables and interfaces (-)
- Wireless – Radio Frequency (RF)
  - Mobility, flexibility (+), very susceptible to EMI (-), multi-path fading (-), attenuation (-), open medium (+/-)
- Wireless – Infra-Red light (IR)
  - Mobility, flexibility (+), line-of-sight (-), open medium (+/-)

# Data Link Layer

- **Adressing**
- Logical Link Control (LLC)
  - Transmission error control
- **Medium Access Control (MAC)**
- ...

# Data Link Layer: Addressing

- Direct addressing
  - The sender and receiver(s) are explicitly identified in every transaction, using physical address (e.g., MAC addresses in Ethernet)
- Indirect (source) addressing
  - The message contents are explicitly identified (e.g. temperature of sensor X). Receivers that need the message retrieve it from the network (as in CAN – Controller Area Network)
- Indirect (time-based) addressing
  - The message is identified by the time instant at which it is transmitted (as in TTP – Time Triggered Protocol)

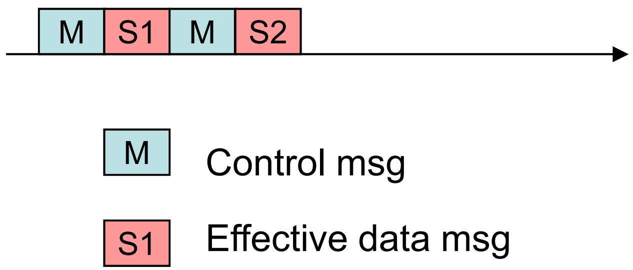
# Data Link Layer: MAC

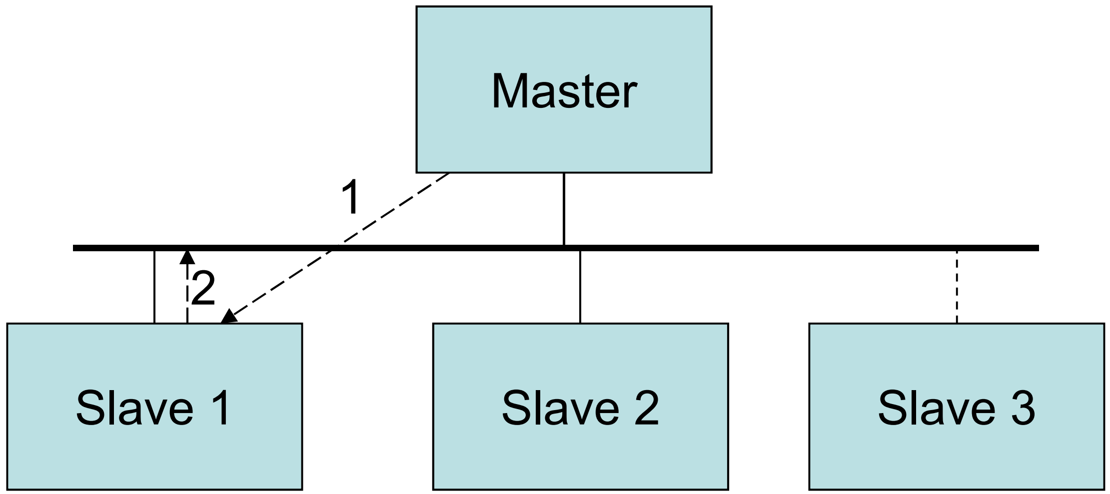
Medium Access Control (MAC)

- Lowest sub-layer of the data link layer
- Determines the order of the network access by contending nodes and, thus, the network access delay
- Is of **paramount importance for the real-time behavior** of networks that use a shared medium

# MAC: Master-Slave

- Access is granted by the Master node
- Nodes synchronized with the master
- Requires one control message per data message

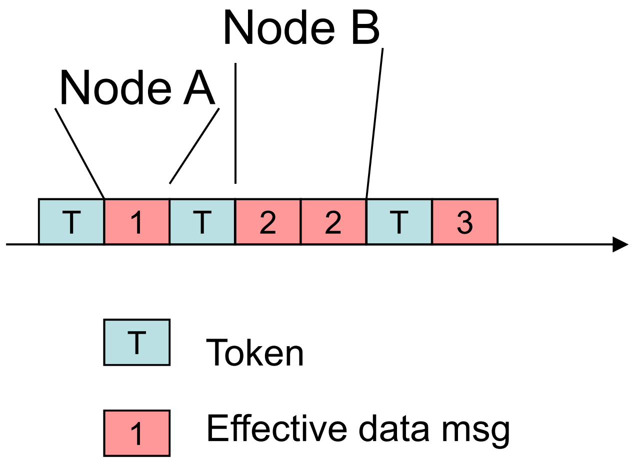


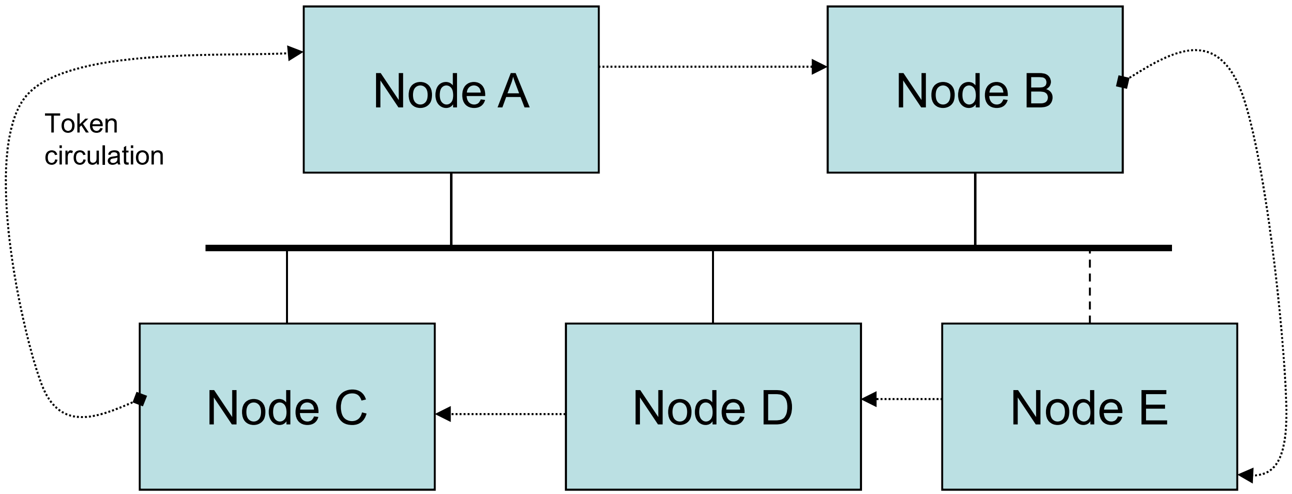




- Ex. WorldFIP, Ethernet Powerlink, Bluetooth (within piconets)

# MAC: Token Passing

- Access is granted by the possession of a token
- Order of access enforced by token circulation
- Real-time operation requires bounded token holding time



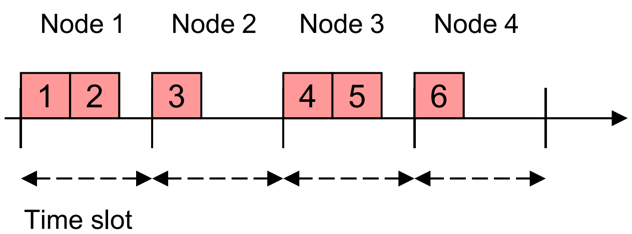



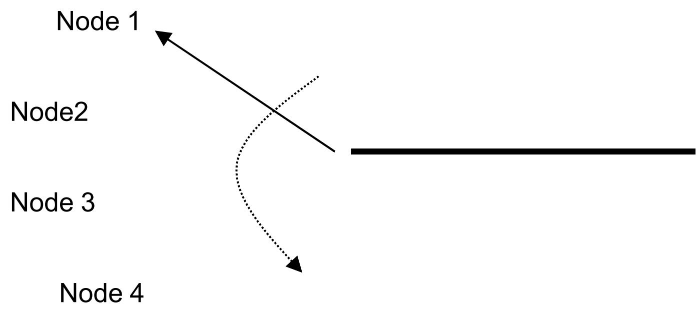

- Ex. FDDI, PROFIBUS

# MAC: TDMA

Time-Division Multiple Access

- Access granted in a dedicated time-slot
- Time slots are pre-defined in a cyclic framework
- Requires global clock syncronization
- High data efficiency, high determinism




- Ex. TTP/C, TT-CAN, PROFINET

# MAC: CSMA

Carrier-Sense Multiple Access:

- Set of protocols based on sensing bus inactivity before transmitting (asynchronous bus access)
- There may be collisions
- Upon collision, the nodes back off and retry later, according to some specific rule

# MAC: CSMA/CD

Carrier-Sense Multiple Access with Collision Detection

- Used in shared Ethernet, WiFi, ZigBee, ...
- Collisions are destructive
- Random, exponentially growing back-off times
- Non-deterministic
- Not suitable for real-time networks. However,
  - An Ethernet physical layer is often used in real-time networks
  - Possible to get real-time performance on an Ethernet network, if access to the medium is scheduled in such a way that collisions are avoided

# MAC: CSMA/BA

Carrier-Sense Multiple Access with Bit-Wise Arbitration

- Also called CSMA/CR (Collision Resolution) or (less accurately) CSMA/CA (Collision Arbitration)
- Bit-wise arbitration with non-destructive collisions
- Upon collision, the highest priority message is unaffected.
- Messages with lower priorities wait until end of transmission and then retry
- Deterministic worst-case behavior
- Ex. CAN

# Contents

# CAN – Controller Area Network

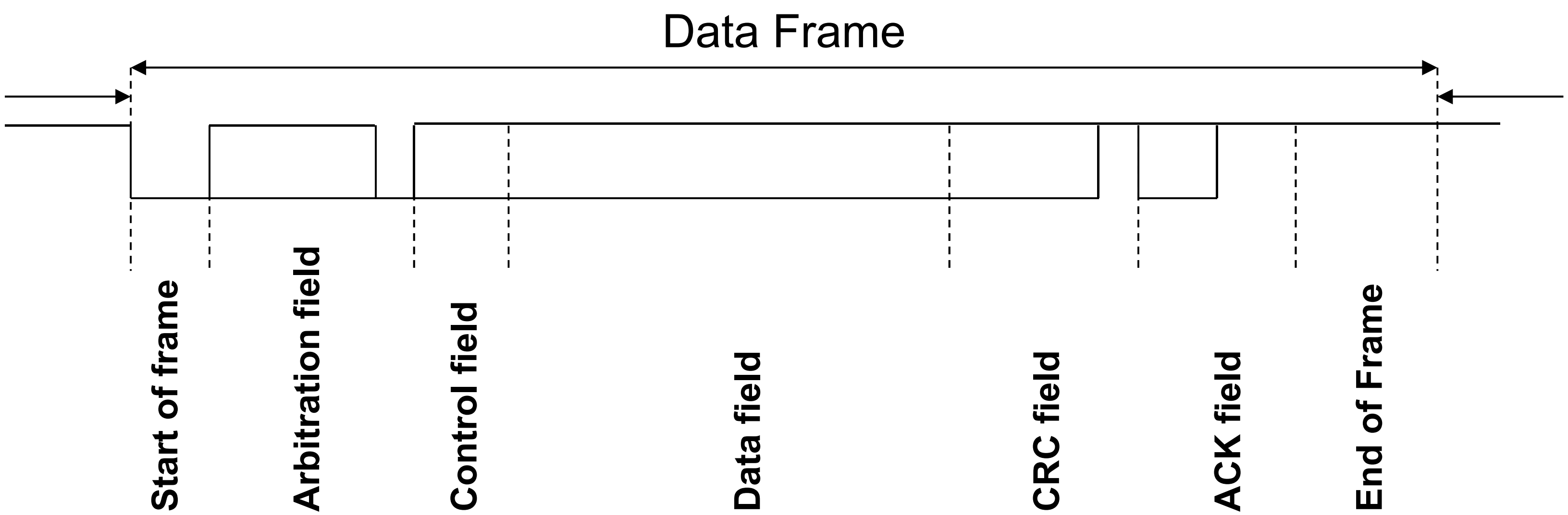
- Created by Bosch for use in the automotive industry
- Used in many European cars today
- Adopted by GM as an in-house standard
- Defines physical and data link layers
- Multi-master, broadcast, serial bus
- Transmision rate from 5 kbit/s to 1 Mbit/s
- Small sized messages: 0–8 bytes
- Relatively high overhead (47 bits + stuff bits)

# CAN

- CAN is like Ethernet …
  - Everybody with a message to send waits until the bus is quiet, then starts transmitting, and if a node detects a collision it backs off and retries later
- … but more deterministic
  - The CAN bus has a special electrical property that allows it to handle collisions better

# CAN Communication

- Messages are called frames
- A frame is tagged by an identifier
  - Indicates the contents of the frame (source addressing)
  - Used in the arbitration for prioritizing frames (frame with lowest identifier is selected to send in case of collision)
- The CAN physical layer behaves as a wired AND, i.e., if any node sends a logical 0, then all nodes receive 0



# CAN Communication

- CAN frames are **bit stuffed**
  - If 5 bits in a row are the same sign then the protocol inserts a bit of the opposite sign
  - Used to ensure enough edges to maintain synchronization
  - Used to distinguish data frames from special error handling frames

# CAN Communication

- All nodes receive all frames
- The handling of the CAN bus communication within a node is done by a special CAN controller (card/chip)
- The CAN controller filters out frames not needed by the node
- Messages that are waiting to be sent are queued in a priority sorted list in the CAN controller

# CAN Arbitration

- Frames start by sending the identifier field's **most significant bit first**
- While sending the identifier the frame is in **arbitration**
  - Other frames may be sent too
  - Need to find the highest priority frame
- If a node sends a 1 (recessive bit) but reads back a 0 (dominant bit) then it gives up and backs off
  - Means that a higher priority frame is being sent
  - Retries sending the frame when the bus is idle again

# CAN Arbitration

| | | |
|---|---|---|
| **Frame 1** | 1344 | 1 |
| **Frame 2** | 1306 | 1 |
| **Frame 3** | 1498 | 1 |
| **Bus** | | 1 |

# CAN Arbitration

| | | |
|---|---|---|
| **Frame 1** | 1344 | 10 |
| **Frame 2** | 1306 | 10 |
| **Frame 3** | 1498 | 10 |
| **Bus** | | 10 |

# CAN Arbitration

| | | |
|---|---|---|
| **Frame 1** | 1344 | 101 |
| **Frame 2** | 1306 | 101 |
| **Frame 3** | 1498 | 101 |
| **Bus** | | 101 |

# CAN Arbitration

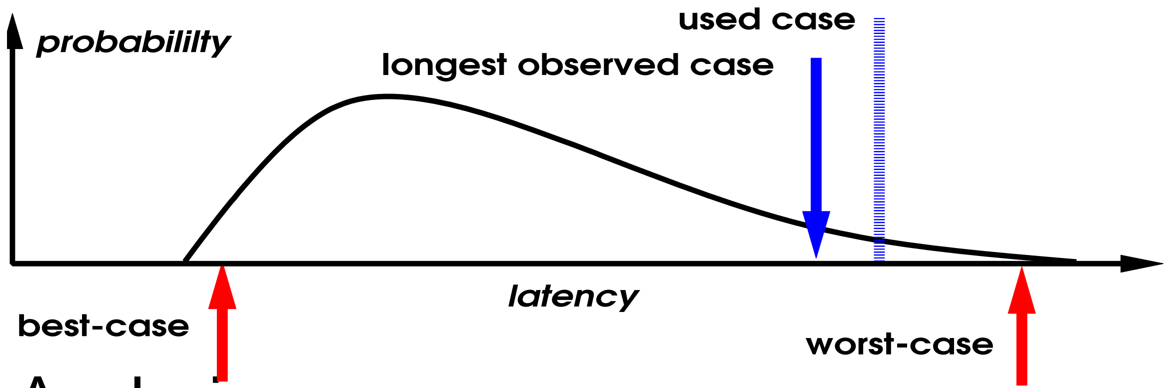
| | | |
|---|---|---|
| **Frame 1** | 1344 | 1010 |
| **Frame 2** | 1306 | 1010 |
| ~~**Frame 3**~~ | ~~1498~~ | ~~1011~~ |
| **Bus** | | 1010 |

# CAN Arbitration

| | | |
|---|---|---|
| ~~**Frame 1**~~ | ~~1344~~ | ~~10101~~ |
| **Frame 2** | 1306 | 10100011010 |
| ~~**Frame 3**~~ | ~~1498~~ | ~~1011~~ |
| **Bus** | | 10100011010 |
| | 1306 | |

# CAN and Hard Real-Time

- Need to bound the worst-case latency (end-to-end delay)
- Option 1: Testing



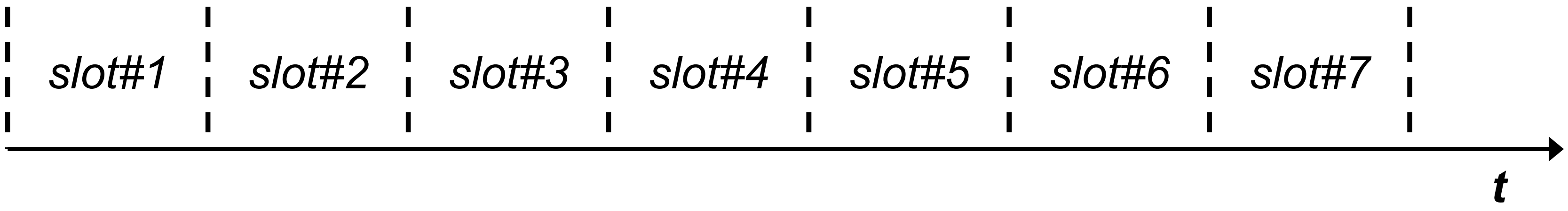

- Option 2: Analysis
  - Bus = shared resource (cp. CPU)
  - Frame = job (invocation of a task)
  - Fixed priority scheduling theory can be applied
    - Blocking factor from transmission of lower-priority frame

# TTP – Time Triggered Protocol

- Not just a network, more of a communication architecture
- Shared broadcast bus, 2–25 Mbit/s
- Popular in car industry for safety-critical applications, e.g., drive-by-wire
- Features
  - Fault tolerance: allow node and network failures without loss of functionality
  - High precision clock synchronization
  - Predictable messages latencies, no jitter

# TTP – Time Triggered Protocol

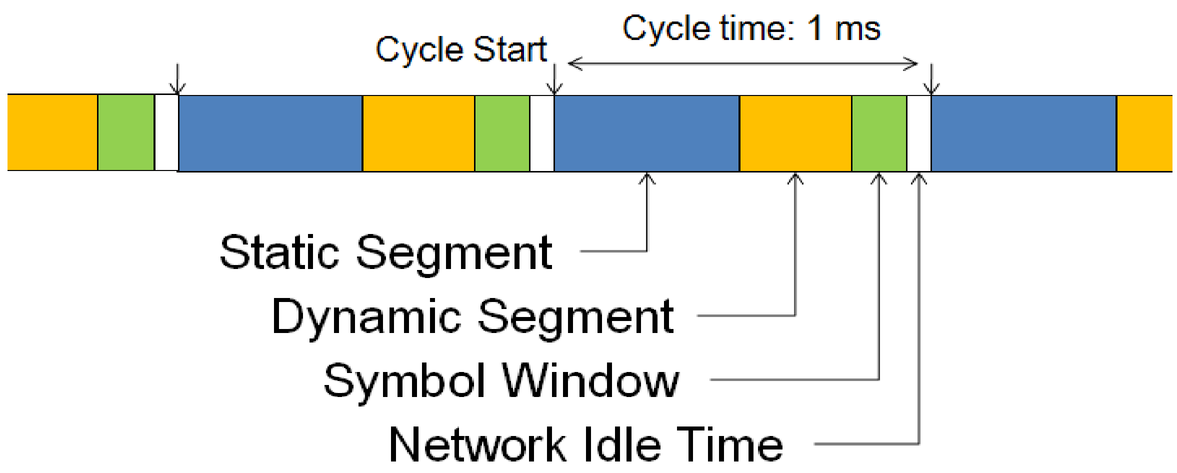
- Mostly periodic messages
- Replicated broadcast communication channels
- Replicated nodes are grouped into FTUs – Fault Tolerant Units
- Access to the network through TDMA (static scheduling)




- More deterministic than CAN

# FlexRay

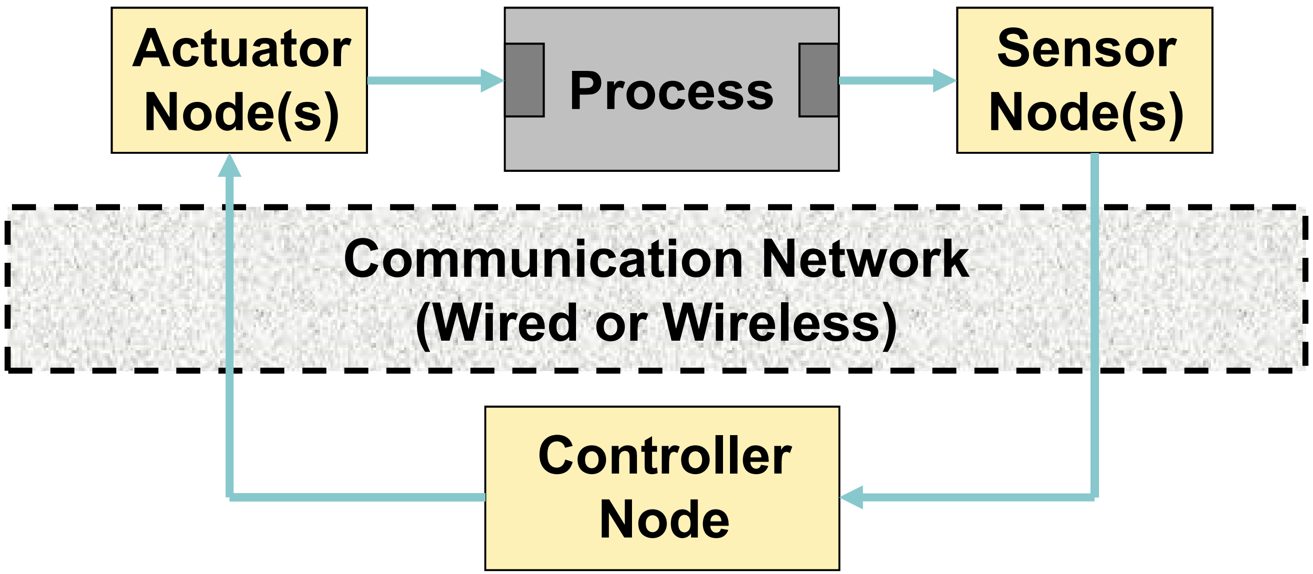
- TTP competitor
- Developed by European car manufacturers
- Combines time-driven and event-driven communication
- Event-driven communication allowed in special slots in the TDMA structure



# Contents

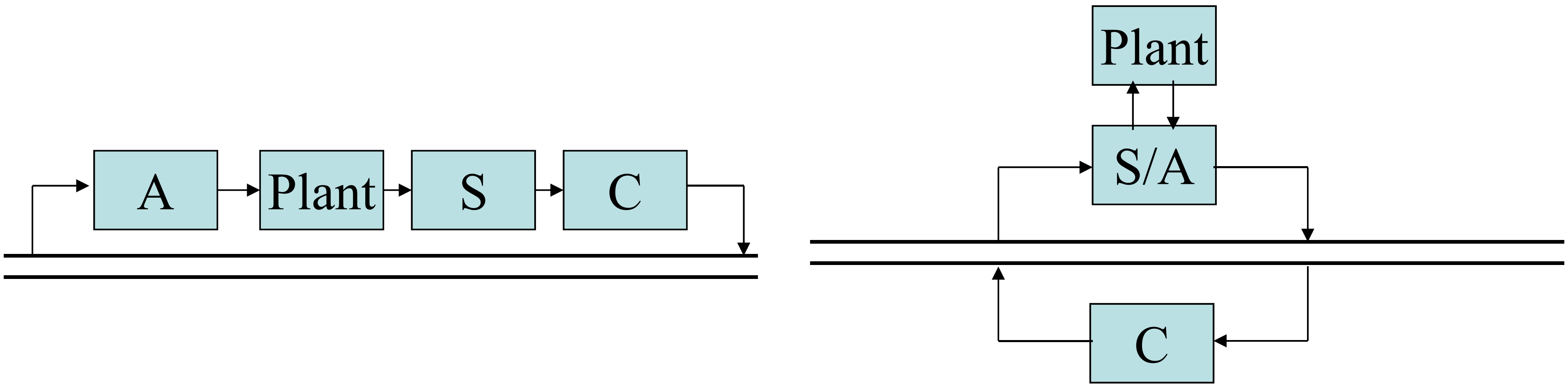
- Background
- Real-Time Networks
- Protocol Stack
- RT-Network Examples
  - CAN
  - TTP
- **Networked Control Systems**

# Networked Control Systems




- Hot research topic in recent years

# Networked SISO Control Structures




- Several more options for MIMO controllers

# Networked Control Loop Timing

- Networked embedded control implies temporal non-determinism
  - network communication
  - real-time scheduling
- Degraded control performance due to
  - sampling jitter
  - control delay and control jitter
  - dropped packets (samples or controls)
- However
  - Most control loops are fairly robust towards temporal non-determinism

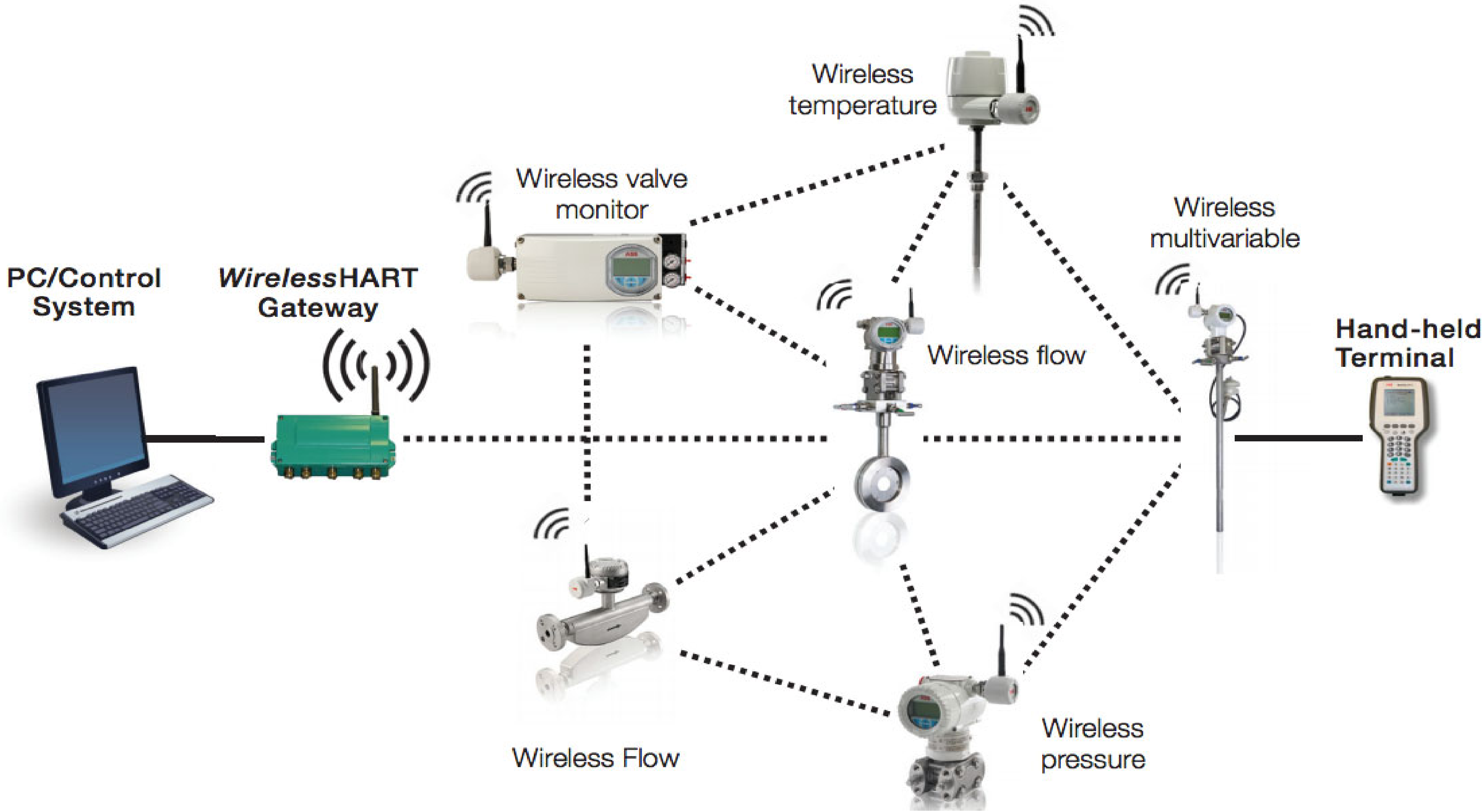
- network interface processing delay
- queuing delay
- transmission delay
- propagation delay
- link layer resend delay
- transport layer ACK delay
- ….

# Wireless Networked Control

Emerging technology. Potential advantages:

- Flexibility
  - Placement: moving parts, mobile units, outdoor, ...
  - Commissioning and maintenance
- Cost
  - Less cables, fewer connectors, less wear and tear
  - Reduced design and installation costs
- New applications

# Example: Wireless Industrial Automation




[ABB product sheet for Wireless HART]

# Example: Car Trains


[Volvo Cars Newsletter: ”Fordonståg en möjlighet på vanliga motorvägar”]

# Networks for Wireless Control

- ISM networks (licence free)
  - IEEE 802.11 (WLAN)
  - IEEE 802.15.1 (Bluetooth)
  - IEEE 802.15.4 (ZigBee)
- Mobile/cellular networks
  - GSM
  - 3G
  - 4G/LTE
  - Device to Device (D2D)

# A Comparison

| Market Name | ZigBee® | --- | Wi-Fi™ | Bluetooth™ |
|---|---|---|---|---|
| Standard | 802.15.4 | GSM/GPRS CDMA/1xRTT | 802.11b | 802.15.1 |
| Application Focus | Monitoring & Control | Wide Area Voice & Data | Web, Email, Video | Cable Replacement |
| System Resources | 4KB - 32KB | 16MB+ | 1MB+ | 250KB+ |
| Battery Life (days) | 100 - 1,000+ | 1-7 | .5 - 5 | 1 - 7 |
| Network Size | Unlimited ($2^{64}$) | 1 | 32 | 7 |
| Maximum Data Rate (KB/s) | 20 - 250 | 64 - 128+ | 11,000+ | 720 |
| Transmission Range (meters) | 1 - 100+ | 1,000+ | 1 - 100 | 1 - 10+ |
| Success Metrics | Reliability, Power, Cost | Reach, Quality | Speed, Flexibility | Cost, Convenience |

[www.zigbee.org]

# Protocols for Wireless Industrial Automation

Two protocols, both based on IEEE 802.15.4:

- Wireless Hart
  - Products since 2008 (ABB, Siemens, ...)
  - International standard (IEC 62591-1), 2010
- ISA 100 Wireless
  - Products since 2009 (Honeywell, GE, ...)
  - ANSI standard 2011
  - International standard (IEC 62734), 2014

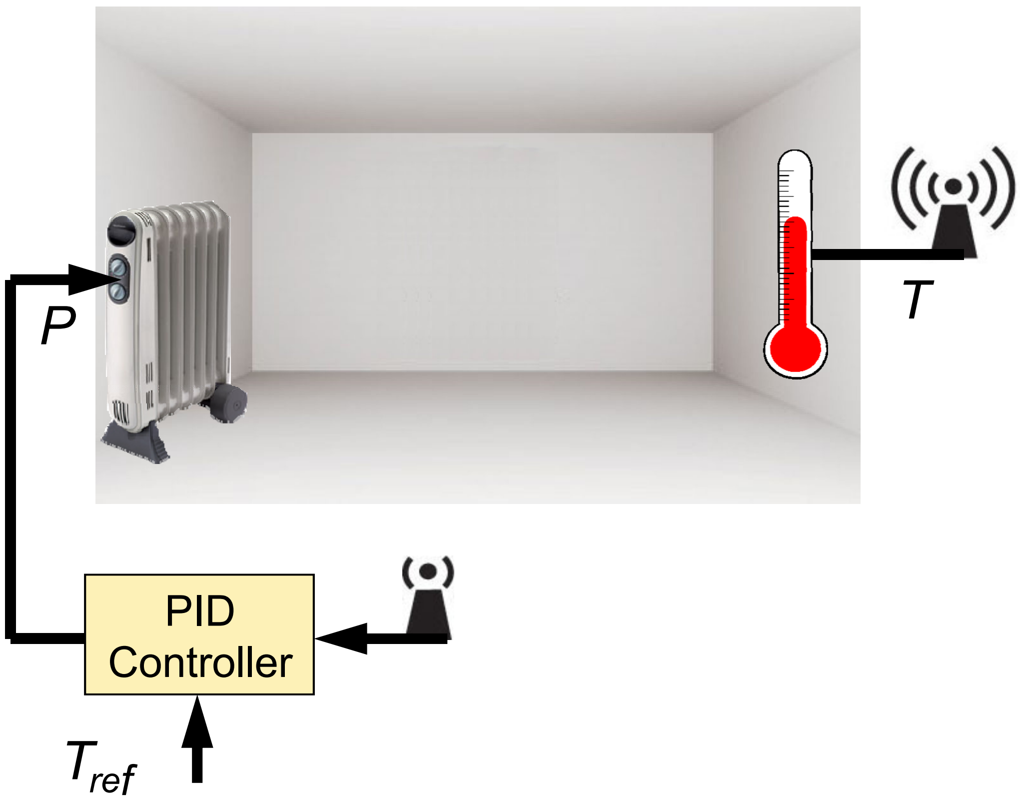
# Challenges in Networked Control

- **Long/variable delays**
- **Dropped packets**
- Safety
  - Must ensure safe error handling and safe shut downs
- Security

Above challenges even greater for wireless control systems

- Radio channel affected by the environment, disturbing nodes
- Routing in multi-hop networks

# Example: Wireless Temperature Control



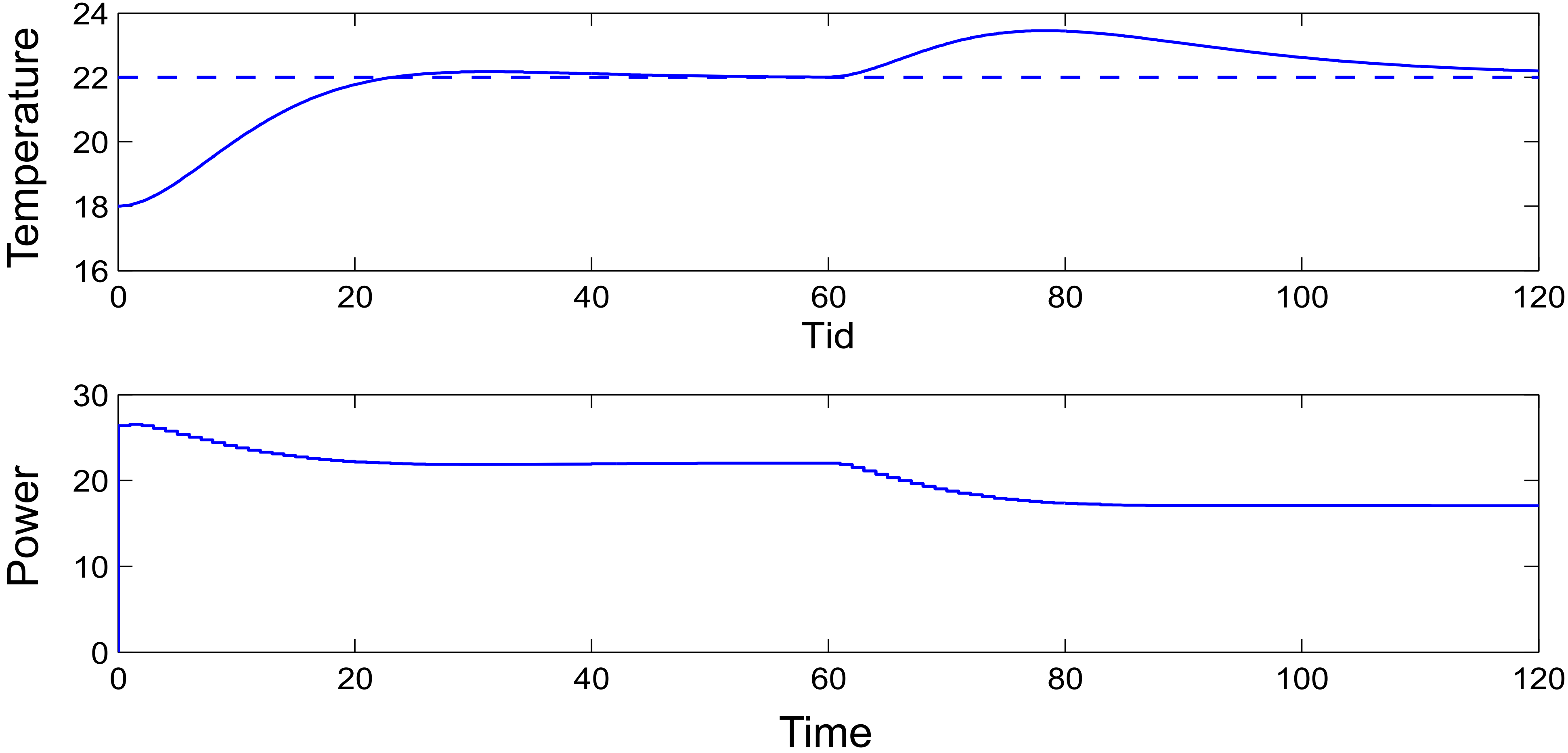

- Process model:

$$G_p(s) = (1 + 10s)^{-2}$$

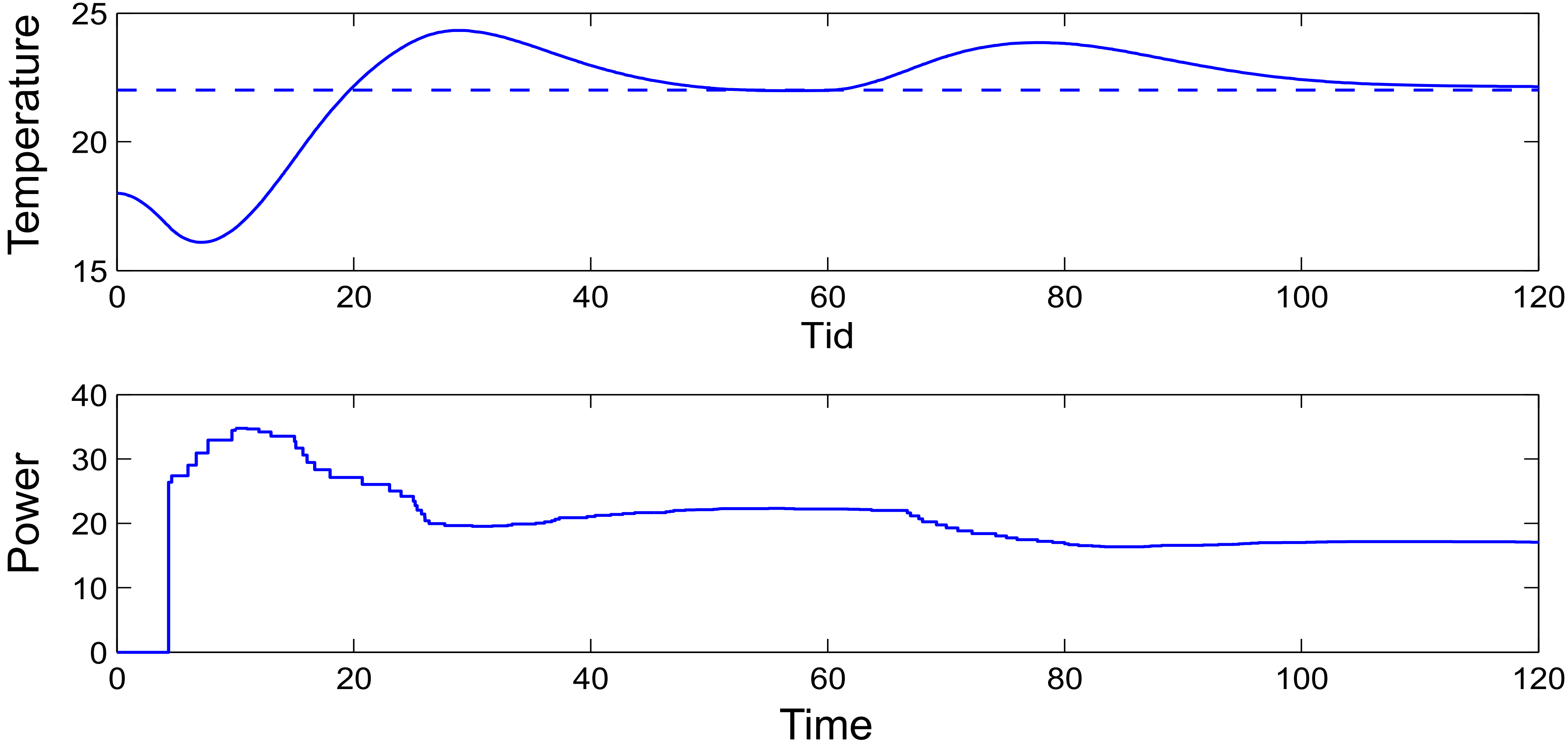
- PID controller, implemented with sampling interval $h$ = 1

➔Simulation

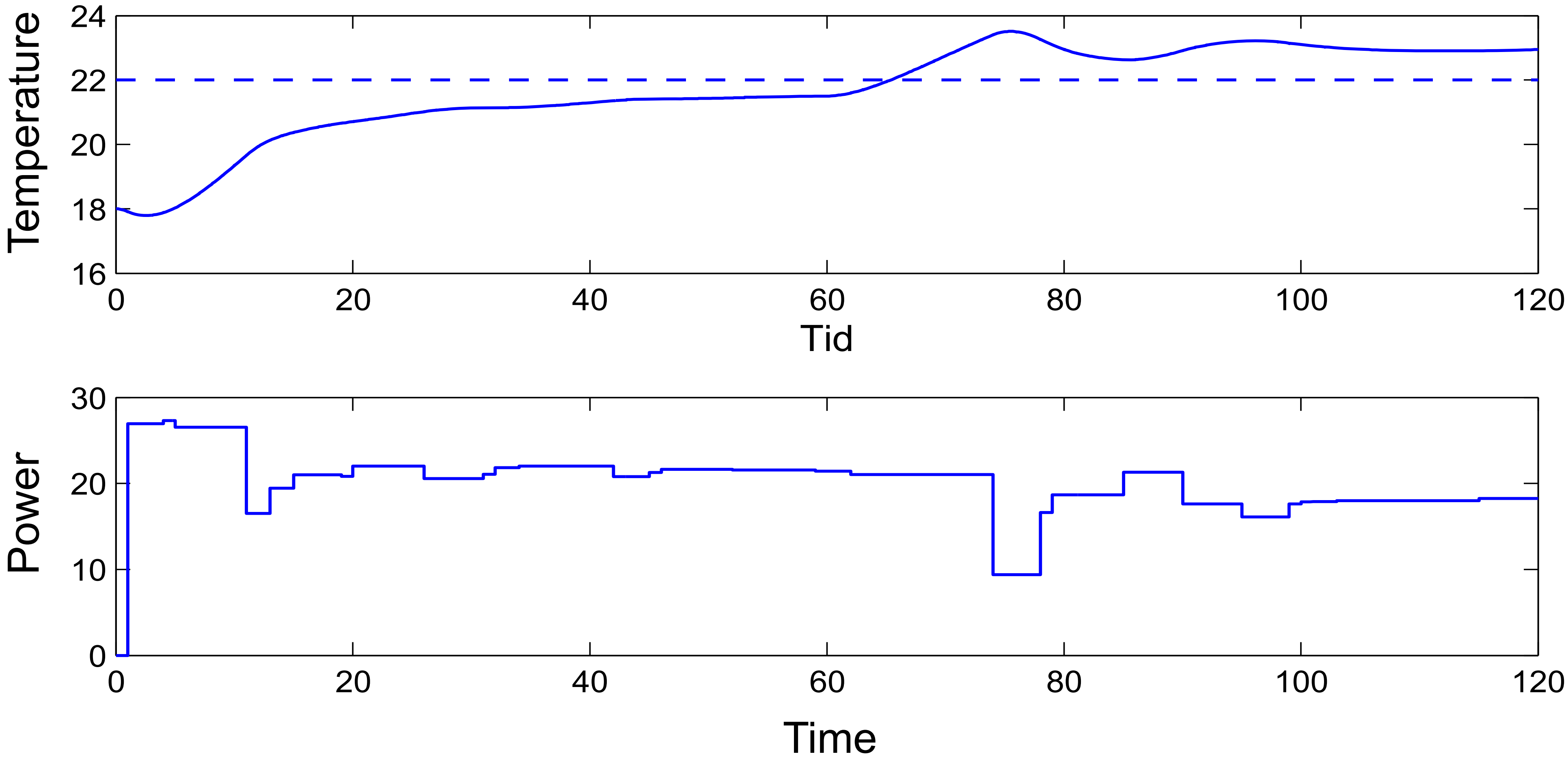
# Simulation: Ideal behavior



# Simulation: Behavior under varying control delay ($\tau$ = [1.0, 7.0] s)



# Simulation: Behavior under dropped packets ($p$ = 70%)

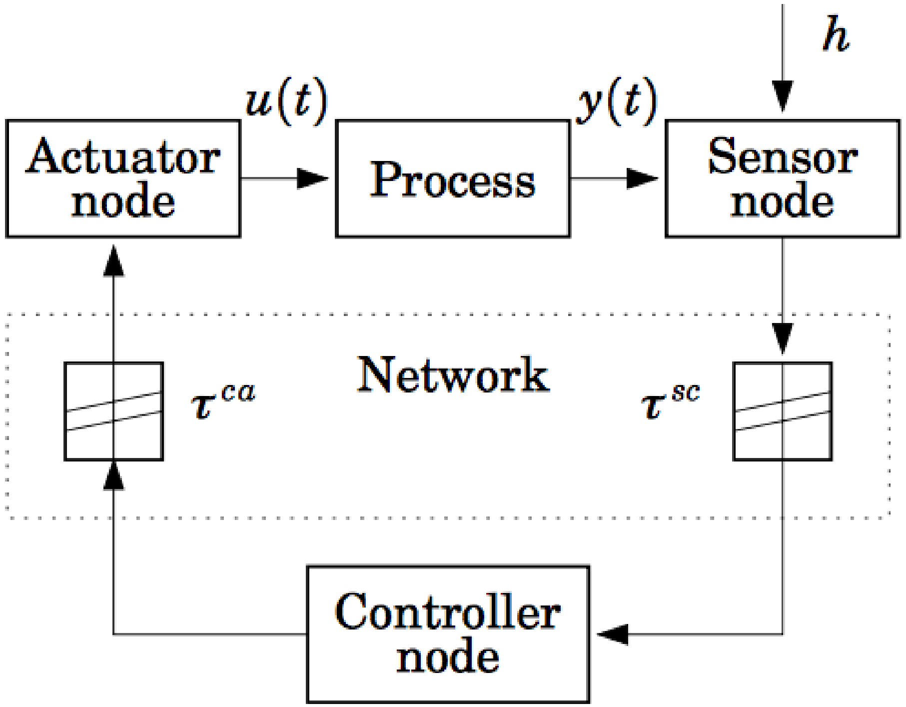


# Some Recent Research Results

- Control with random network delays (J. Nilsson *et al.*)
  - Controller design
- The Jitter Margin (Y. Kao & B. Lincoln)
  - Stability analys
- TrueTime (A. Cervin *et al.*)
  - Simulation toolbox

# Control with Random Network Delays

System model:




- Probability distributions of sensor–controller and controller–actuator delays assumed known
- Time-stamped messages

# Optimal (LQG) Control

- Cost function to be minimized:

$$J_N = \mathrm{E}\ x_N^T Q_N x_N + \mathrm{E} \sum_{k=0}^{N-1} \begin{bmatrix} x_k \\ u_k \end{bmatrix}^T Q \begin{bmatrix} x_k \\ u_k \end{bmatrix}$$

- The optimal state feedback has the form

$$u_k = -L_k(\underline{\tau_k^{sc}}) \begin{bmatrix} x_k \\ u_{k-1} \end{bmatrix}$$

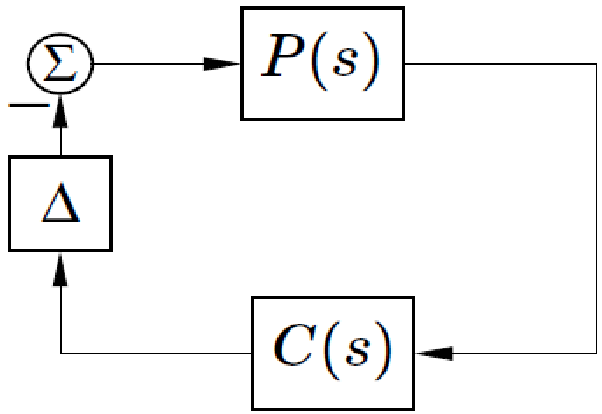
- The optimal observer has the form

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + \overline{K}_k(y_k - C\hat{x}_{k|k-1})$$

$$\hat{x}_{k+1|k} = \Phi\hat{x}_{k|k-1} + \Gamma_0(\underline{\tau_k^{sc}, \tau_k^{ca}})u_k + \Gamma_1(\underline{\tau_k^{sc}, \tau_k^{ca}})u_{k-1} + K_k(y_k - C\hat{x}_{k|k-1})$$

# The Jitter Margin

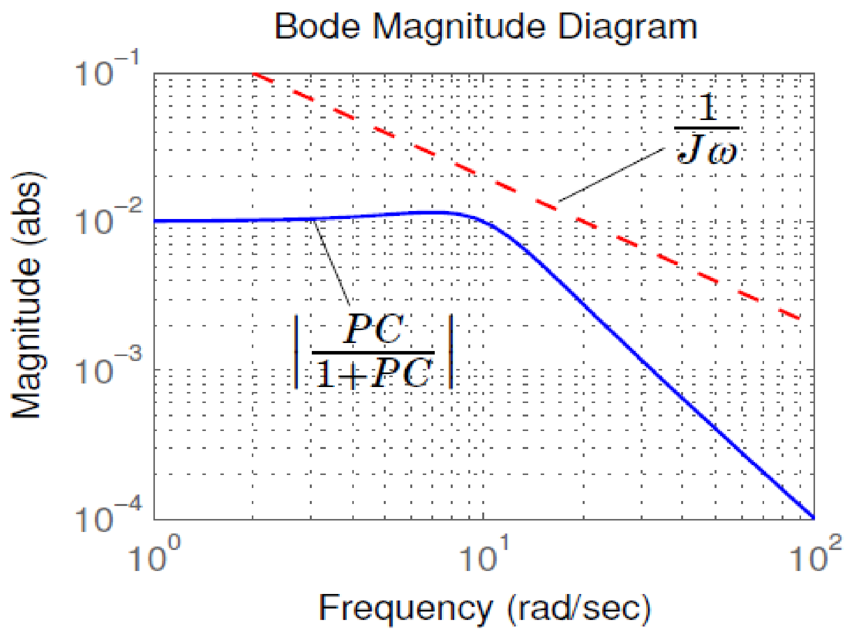
- Generalization of the well known **delay margin**
- How large **variable delay** (**= jitter**) can a feedback loop tolerate and still remain stable?
- System model:



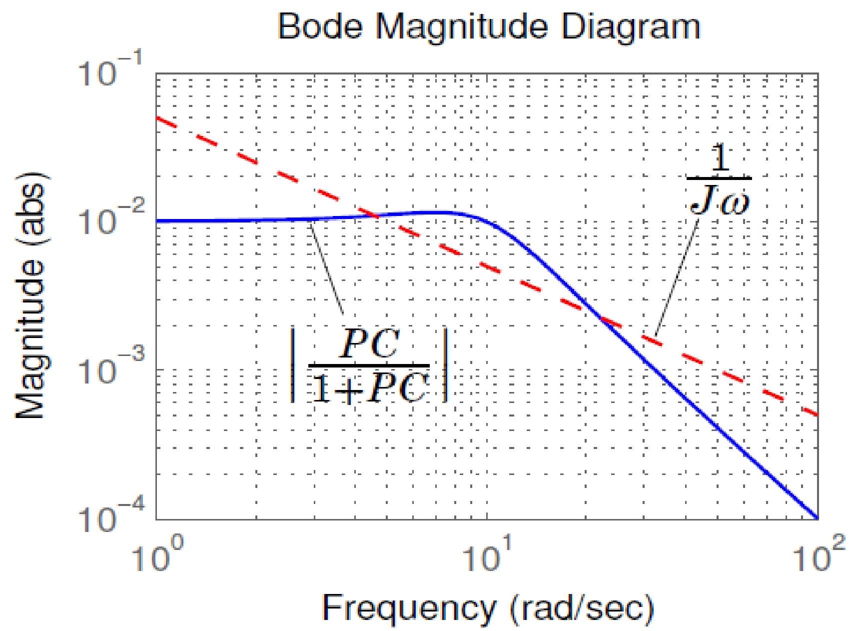

  - Linear process $P(s)$
  - Linear controller $C(s)$
  - Variable delay $\Delta = (0 \ldots J)$
    - Allowed to vary in *any way* within the interval

# The Jitter Margin

- Simple (sufficient) test in Bode magnitude diagram:



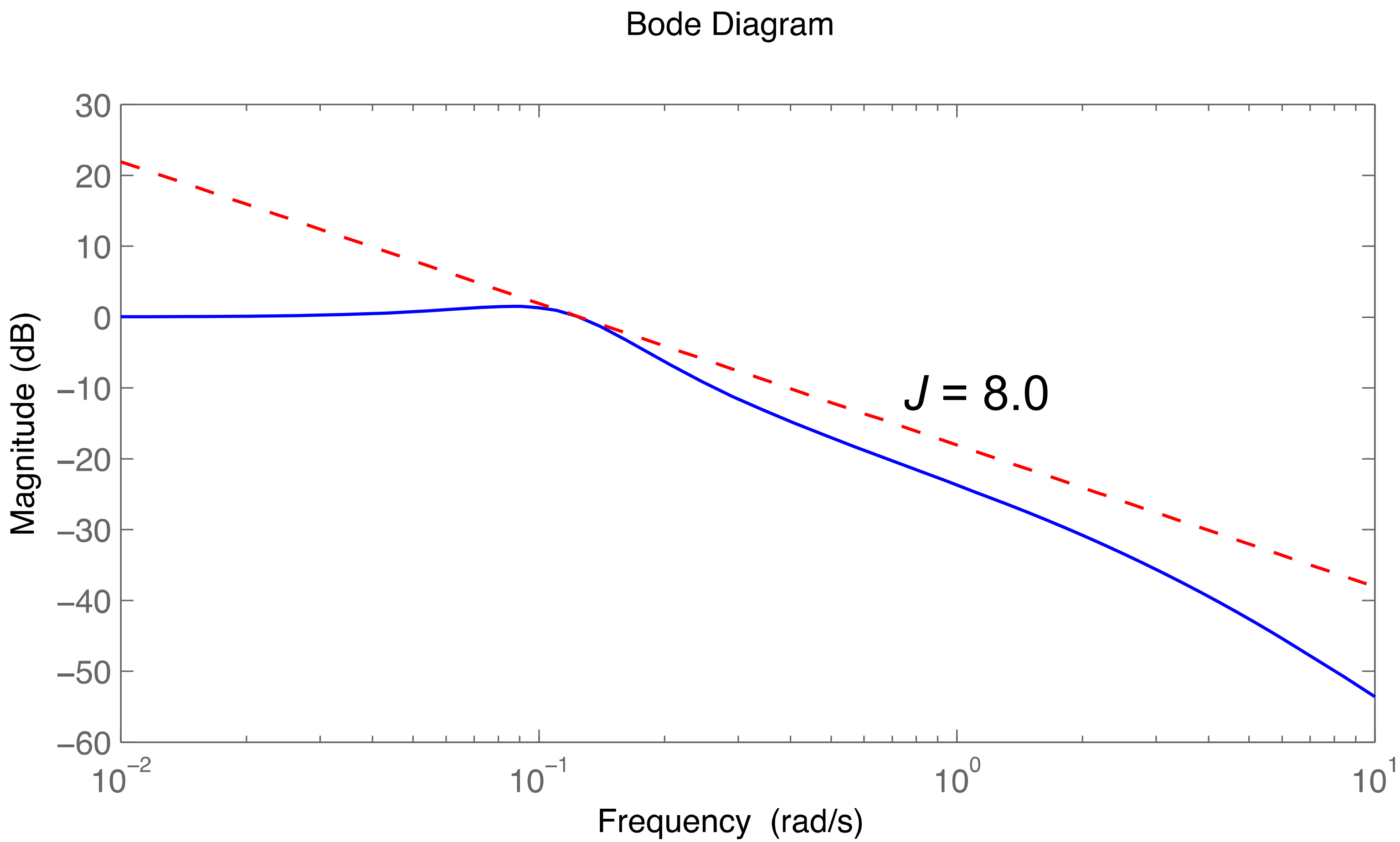

Stable




Could be unstable

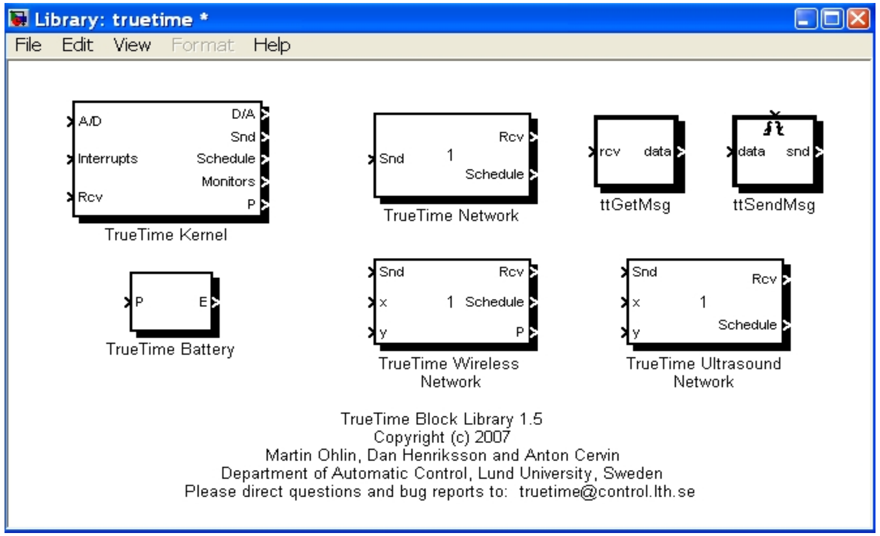
(Similar criteria exist for discrete-time and sampled-data systems)

# Jitter Margin for the Example



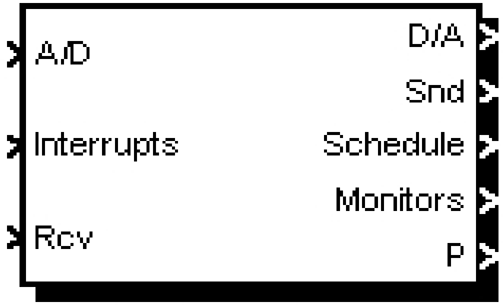
# TrueTime

- A simulator for embedded control systems
- Matlab/Simulink-based
- Co-simulation of
  - Process dynamics
  - CPUs with RTOS
  - Wired and wireless networks
- Developed at Automatic Control LTH since 1999
  - Open source code



# Modeling of CPUs

- Event-based RTOS
- User-defined code (tasks/handlers)
  - C++ or Matlab code
- Advanced scheduling algorithms

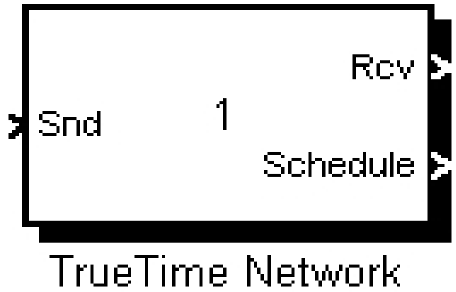


TrueTime Kernel

```
function [exectime,data] = my_ctrl(segment,data)
switch segment,
  case 1,
     data.y = ttAnalogIn(1);
     data.u = calculate_output(data.x,data.y);
     exectime = 0.002;
  case 2,
     ttAnalogOut(1,data.u);
     data.x = update_state(data.x,data.y);
     exectime = 0.004;
   case 3,
     exectime = -1;
end
```

# Modeling of Wired Networks

- Models the lowest layers (PHY, MAC)
- Support for common networks/protocols:
  - TDMA
  - FDMA
  - CSMA/CD (Shared Ethernet)
  - Switched Ethernet
  - CAN
  - Flexray
  - PROFINET IO

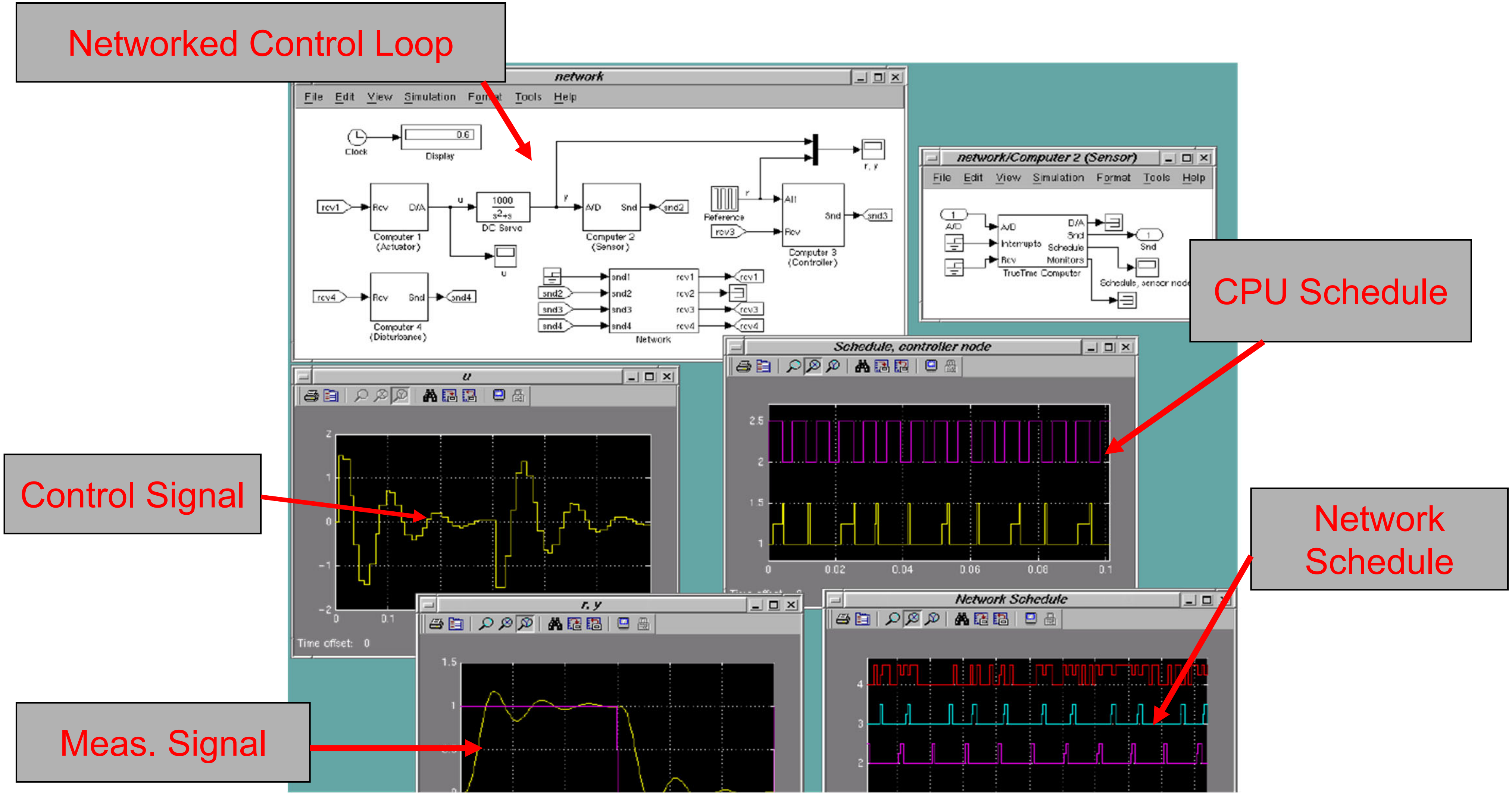


# Modeling of Wireless Networks

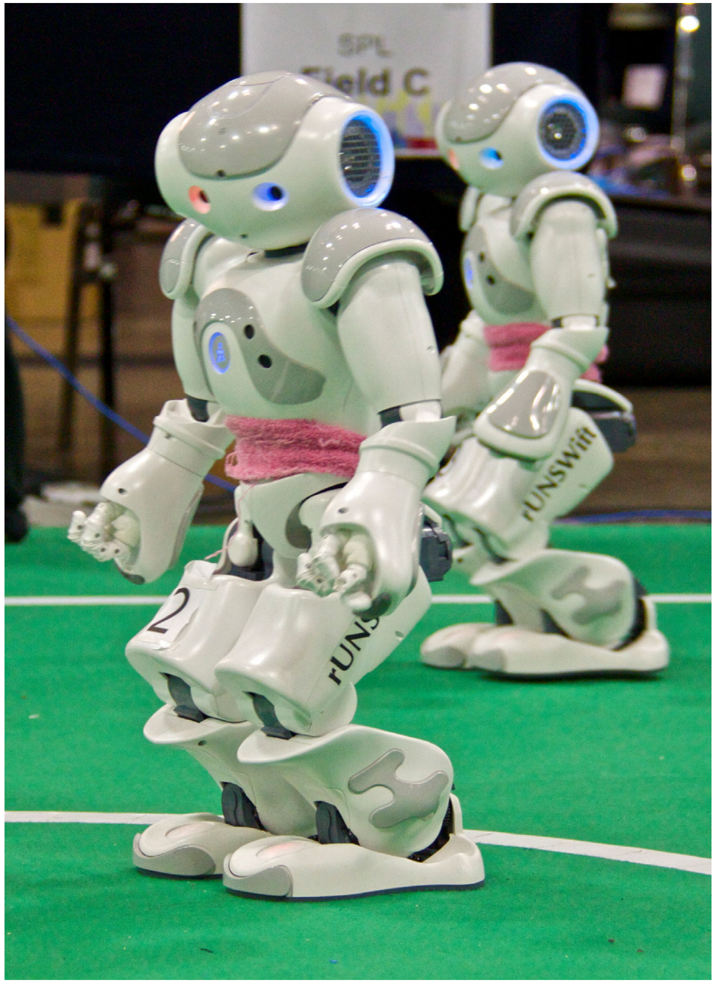
- Two MAC protocols:
  - IEEE 802.11 (WLAN)
  - IEEE 802.15.4 (ZigBee)
    - (Wireless HART implemented by ABB Research)
- *x* and *y* inputs for node positions
- Radio models:
  - Exponential path loss (default)
  - User-defined models for multi-path propagation, fading, etc.

TrueTime Wireless Network

# Example – Networked Control Loop



# TrueTime – Demo



# Conclusions

- Real-time communication increasingly important in several fields
- The traditional OSI model not well suited for real-time networks
- Collapsed OSI model (physical, data link, application) better for real-time networks
- Networked control systems increasingly common
- Potential problems with delays, jitter and lost packets
- Much theory and many tools for networked control exist
  - Some examples from Dept. Automatic Control, LTH:
    - LQG with random delays
    - Jitter margin
    - TrueTime