Real-Time Communication 2

Communication Protocols

slide credits: H. Kopetz, P. Puschner

Overview

Medium Access Protocols

- CSMA/CD
- CSMA/CA
- Token Bus
- Minislotting
- Central Master
- TDMA

Maximum Protocol Execution Time (d_{max})

At the transport level, d_{max} depends on

- Protocol stack at sender (including error handling)
- Message scheduling strategy at sender
- Medium access protocol
- Transmission time
- Protocol stack at the receiver
- Task scheduling at the receiver

In general purpose operating systems, the execution path for the transport of a single message may comprise tens of thousands of instructions.

Medium Access – CSMA/CD

CSMA: Carrier Sense Multiple Access

CD: with Collision Detection

- Communication controller that wishes to send, senses the bus for traffic; it starts sending if it detects no carrier signal
- Collision: different nodes start sending at the same time
- Transmitter listens to signal to detect collisions
- Collision: jam signal; re-send after random time interval; max. *k* (e.g., 10) attempts to send

Example: Ethernet (bus topology, shared medium)

Medium Access – CSMA/CA

CA: with Collision Avoidance

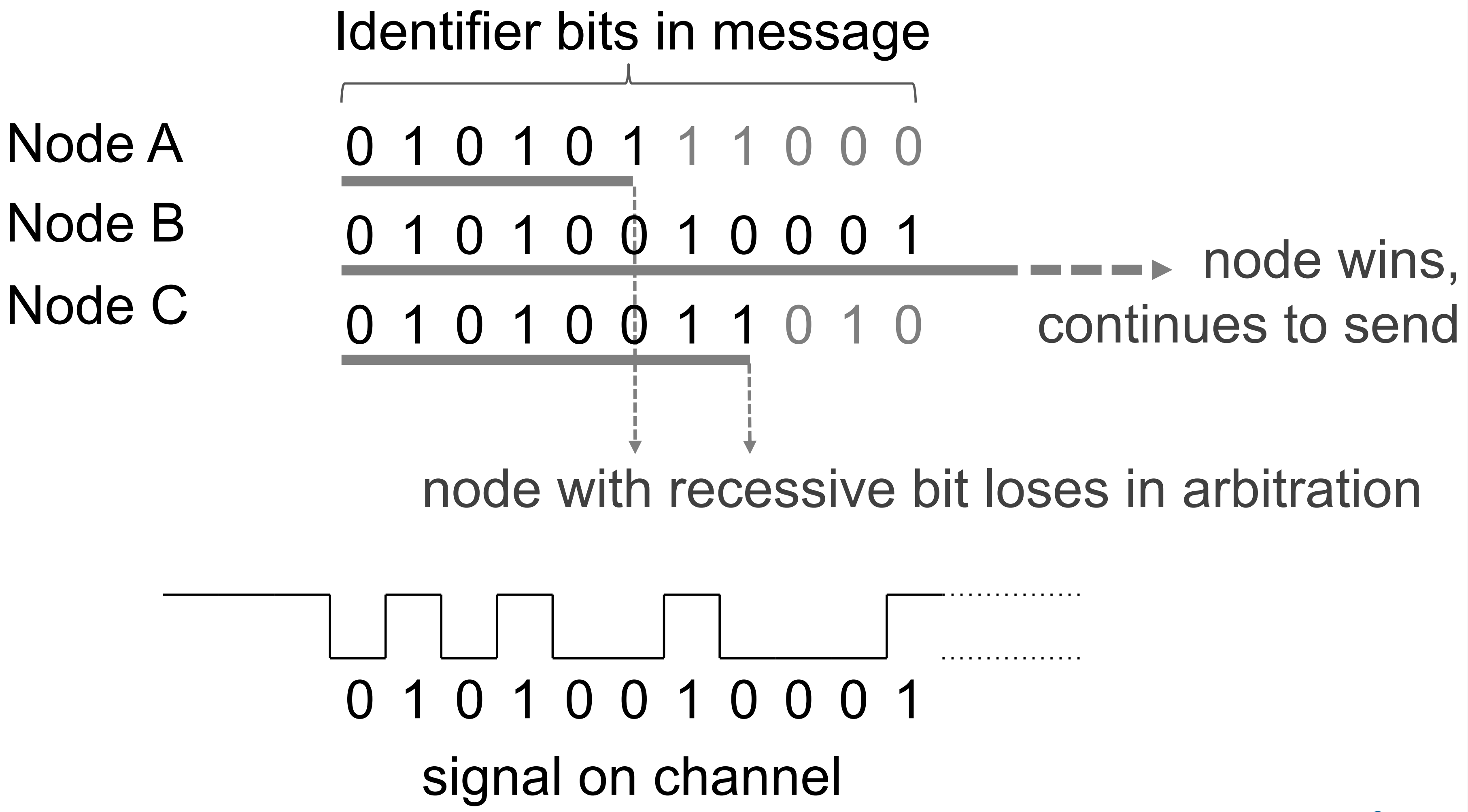
- Typical mechanism for CA: bit arbitration;
  Messages start with identifier ≈ priority for bit arbitration
- There are two states on the communication channel
    - dominant (e.g., bit = 0)
    - recessive (e.g., bit = 1)

  If two stations start to transmit at the same time

  ⇨ the station with a dominant bit in its arbitration field wins

  ⇨ the station with a recessive bit has to give in.

Bit Arbitration in CSMA/CA



CSMA/CA Timing Parameters

Arbitration: every bit has to stabilize before arbitration

⇨ Propagation delay of channel d_{prop} << length of a bitcell

Example:

Bus length: 40 m, d_{prop}: 200 nsec

⇨ length of a bitcell: 1 µsec = 5 propagation delays!

CAN – Control Area Network

Arbitration: CSMA/CA

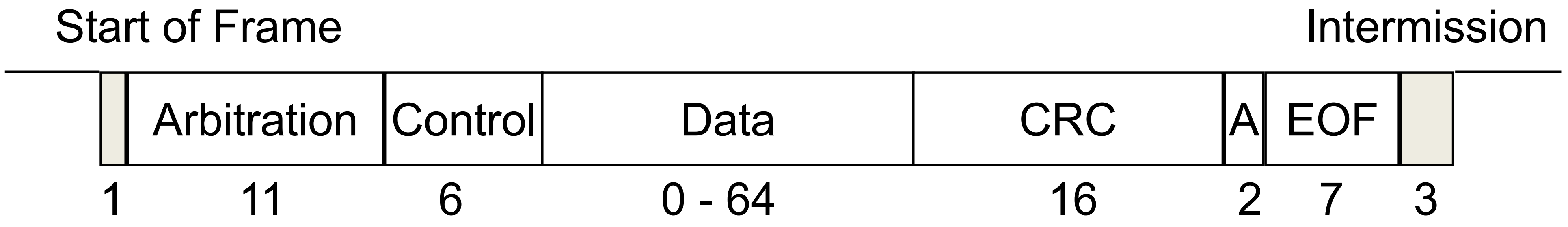
Communication speed: 1 Mbit/second

Channel length: about 40m

Standard Format: 2032 Identifiers, 11 bit arbitration field

Extended Format: > 10^8 Identifiers, 32 bit arbitration field

Frame format



Medium Access – Token Bus/Ring

- Token: special control message to transfer the right to transmit
- Only the token holder is allowed to transmit
- Senders form a physical or logical ring

- Central timing parameters
  - Token Hold Time: longest time a node is allowed to hold the token
  - Token Rotation Time: longest time for a full rotation of the token

- Token Loss constitutes a serious problem in HRT context
  - Token recovery: node creates new token after a random timeout – may lead to collision and retry

Medium Access – Minislotting

- Time is partitioned into a sequence of minislots
- Duration of minislot > maximum propagation delay
- Each node is assigned a unique number of minislots that must elaps with silence on the channel, before it can start to send

Example: ARINC 629

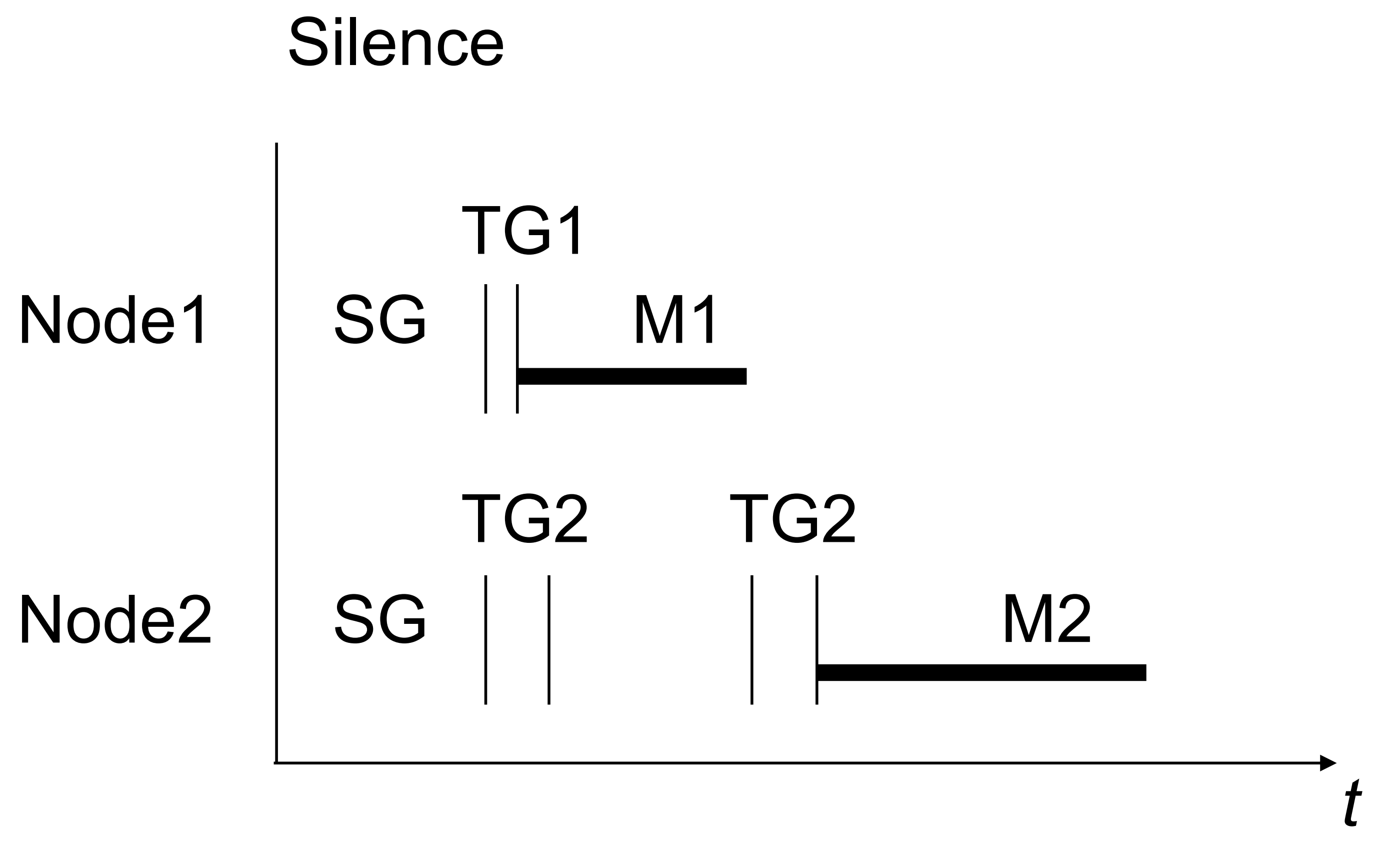
ARINC 629

ARINC 629 is a mini-slotting protocol that is used in the aerospace community.

Medium access is controlled by the intervals:

TG:	Terminal Gap, different for every node, longer than the propagation delay of the channel, determines send order (node-specific number of minislots)
SG:	Synchronization Gap, longer than longest TG; starts a new communication round

ARINC 629 – Timing Diagram



Medium Access – Central Master

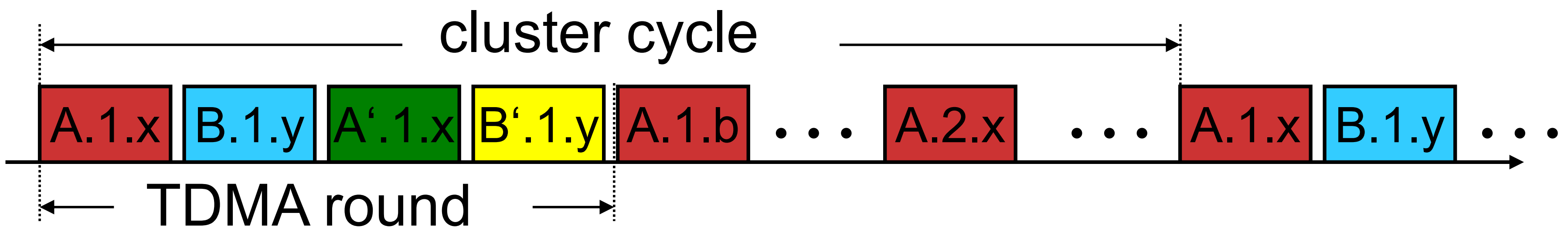
- A central master controls the access to the channel
- In case the master fails, another node must take over the role of the master
- Central master is called bus arbitrator

- Master periodically broadcasts variable names
- Node producing the variable then broadcasts its value
- If time remains, nodes may also send sporadic data after being polled by the master

Example: FIP

Medium Access – TDMA

- TDMA: Time Division Multiple Access
- Requires a (fault-tolerant) global time base
- Time is statically divided into time slots; a static cyclic message schedule assigns each slot to one node that may send
  - TDMA round contains one slot per node
  - Cluster cycle: global sequence of TDMA rounds that is repeatedly executed



The Time-Triggered Protocol, TTP

Integrated time-triggered protocol for real-time systems that provides the following services:

- clock synchronization
- temporal encapsulation
- composability for system integration
- predictable transmission for all messages (TDMA schedule)
- membership service
- support for mode changes
- fault-tolerance support

TTP: TTP/C for safety-critical applications
(TTP/A: master-slave protocol for field buses)

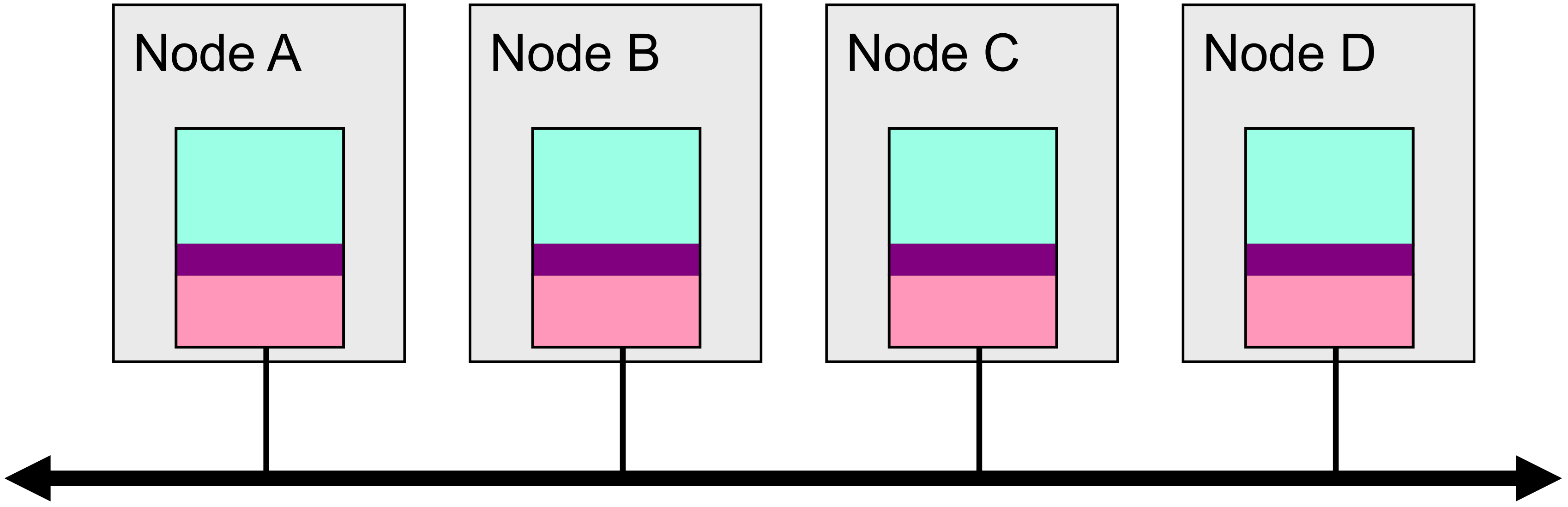
TTP Basics

TTP Node: one electronic control unit (ECU)

TTP Cluster: nodes connected by TTP channel

Bus topology with broadcast communication

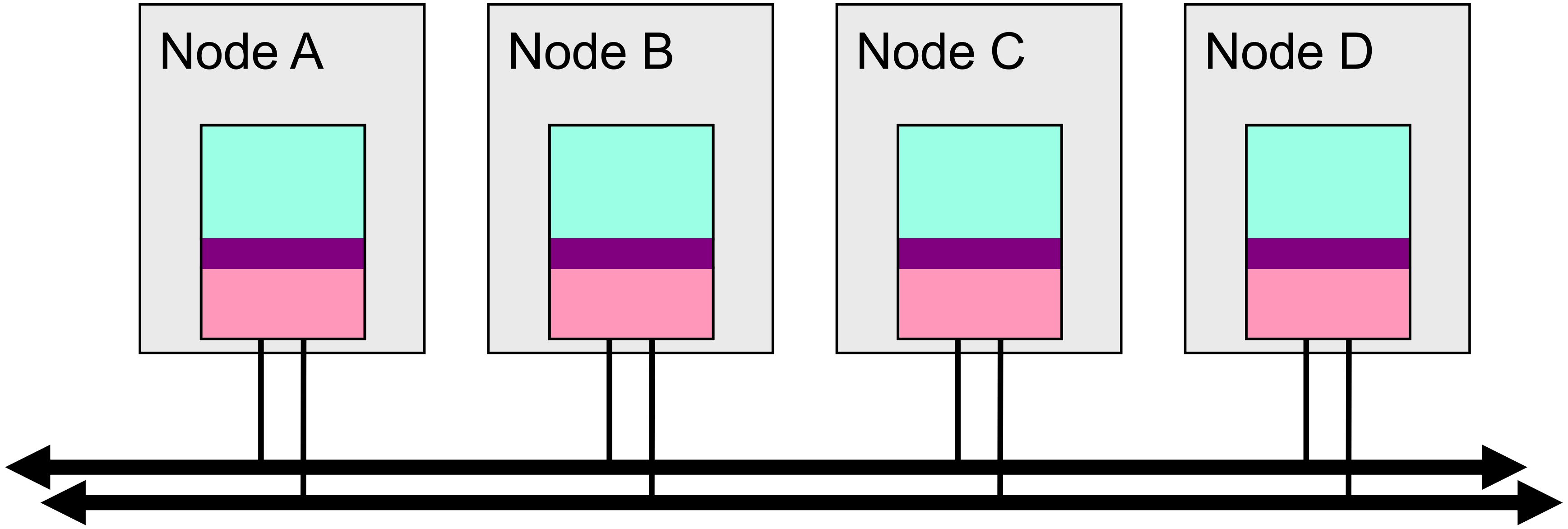
- Publisher writes to bus (e.g., sensor values)
- Subscriber: reads values from bus, reacts (e.g., actuator output))



Redundant Channels

Single fault hypothesis: communication system must tolerate one fault at a time

➪ Redundant communication channels make bus fault tolerant

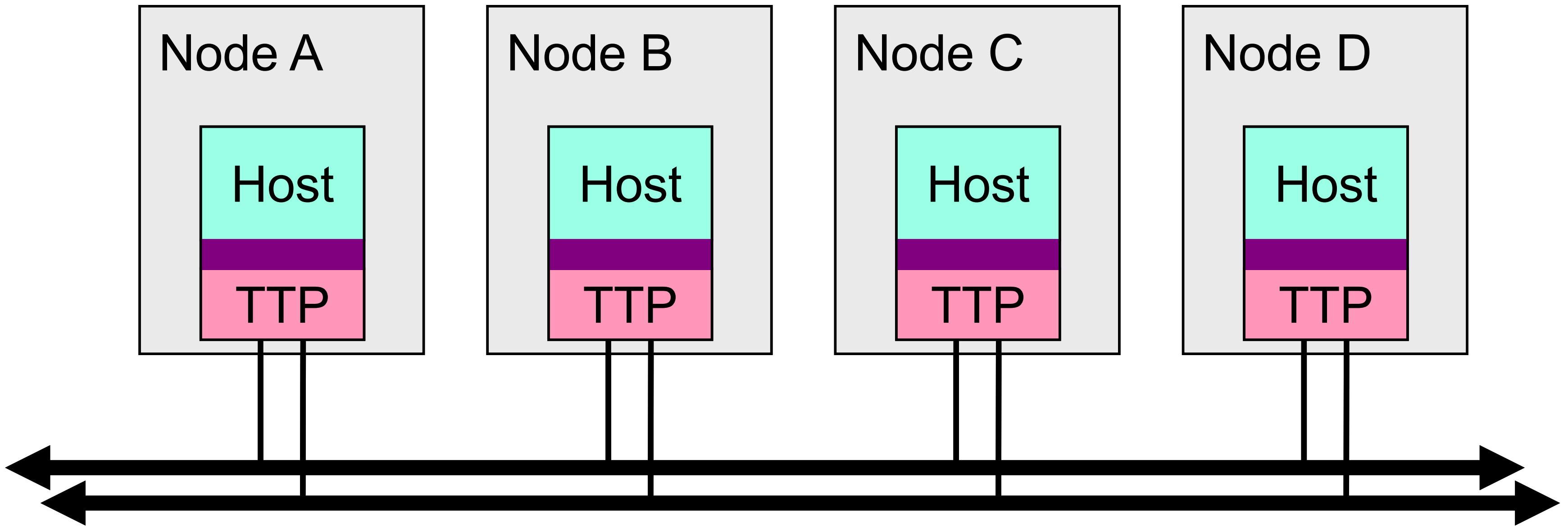


Separation Host – TTP Controller

Single fault hypothesis: a single fault in a node must be tolerated

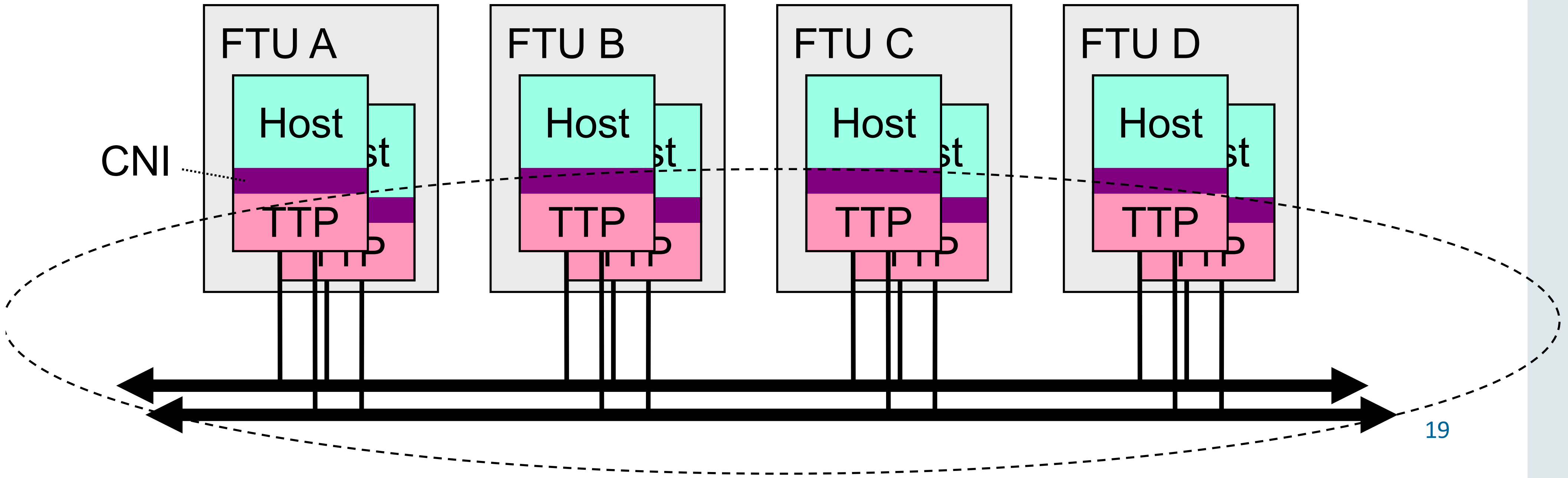
⇨ Nodes are partitioned into two independent parts

- Host computer: executes the application code
- TTP controller: provides TTP services



Fault-Tolerant Architecture

- The communication network, consisting of the bus interconnect and the TTP controllers is duplicated to tolerate network faults
- Components are duplicated to tolerate faulty hosts



Communication Network Interface – CNI

CNI: data-sharing interface between host and TTP Controller

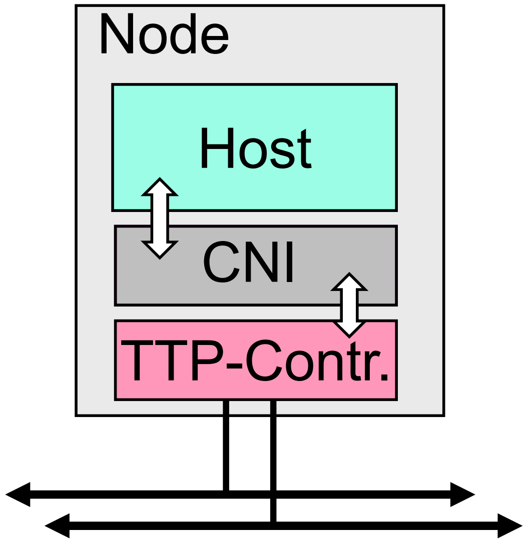
- Contains incoming/outgoing data; network status info., bits to control the TTP controller
- Acts as temporal firewall for control signals

Host

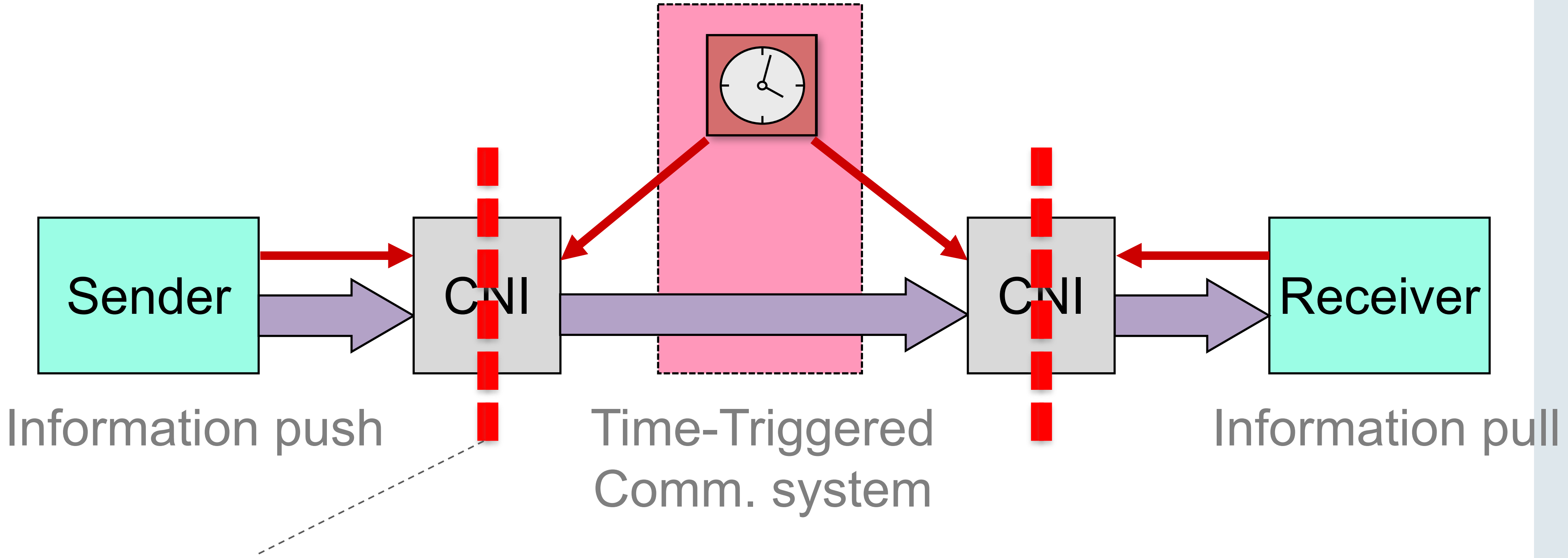
- writes data for sending to CNI
- writes control data to CNI
- reads received data/status info from CNI

Controller

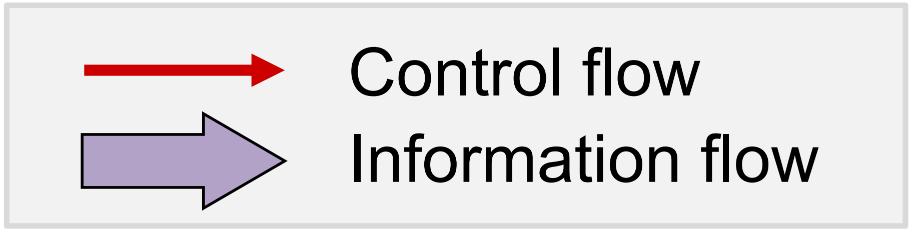
- writes received message data to CNI
- sets status
- sends messages composed from
  CNI data over the network



Handling Control Conflicts in TT Comm.

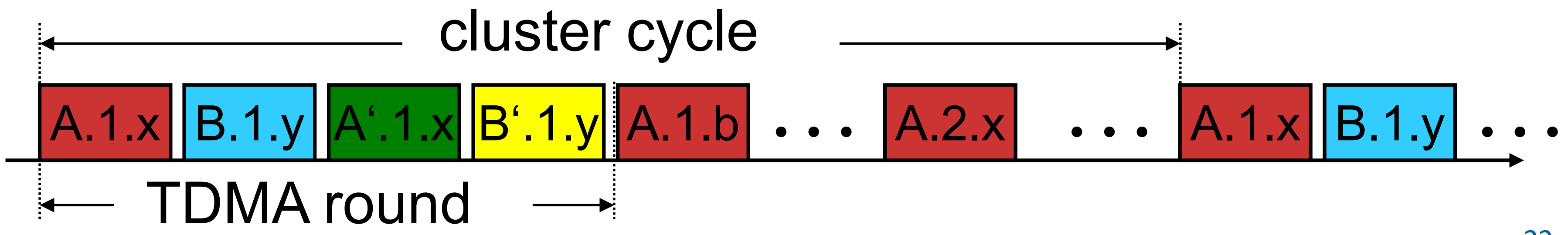


Control conflicts: masking vs. avoidance ➪ see OS comp. services



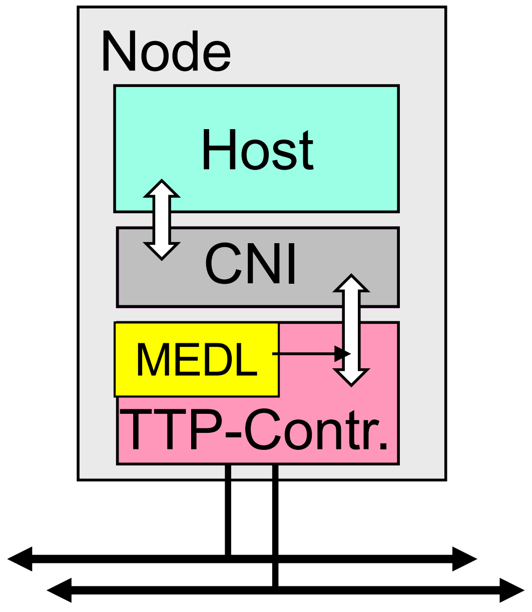
Time-Triggered Communication

- TDMA – Time Division Multiple Access to communication medium ⇨ global time base
- Components send regularly in pre-defined time slots
- Messages have different periods – components send different messages in successive TDMA rounds
- Implicit naming / addressing



Message Schedule

- The message schedule is stored in the Message Descriptor List (MEDL)
- MEDL is application dependent
- MEDL is stored in every TTP controller (flash memory initialized at startup)
- Controller sends messages according to MEDL
- Controller works completely independent from host, no waiting



TTP – Principle of Operation

- TTP generates a global time-base
- Error detection is at the receiver, based on the a-priori known receive time of messages
- Acknowledgement implicit by membership
- State agreement between sender and receiver is enforced
- Every message header contains 3 mode change bits that allow the specification of up to seven successor modes

Use of A-Priori Knowledge

A-priori knowledge about the behavior permits:

- Error Detection: Message send/receive times are globally known (*Life sign for membership*).
- Message Identification: The point in time of message transmission identifies a message (*Reduction of message size*)
- Flow control: It is known a priori how many messages will arrive in a peak-load scenario (*Resource planning*).

For event-triggered asynchronous architectures, there exists an impossibility result: *'It is impossible to distinguish a slow node from a failed node!'* This makes the solution to the membership problem very difficult.

Fail-Silent Nodes

Error Containment and Fault Management are simplified and accelerated, if the nodes of a distributed system exhibit "clean" failure modes.

If a node sends either correct messages (in the value and time domain) or detectably incorrect messages in the value domain, then the failure mode is clean, i.e., the node is fail-silent.

TTP is based on a two level approach:

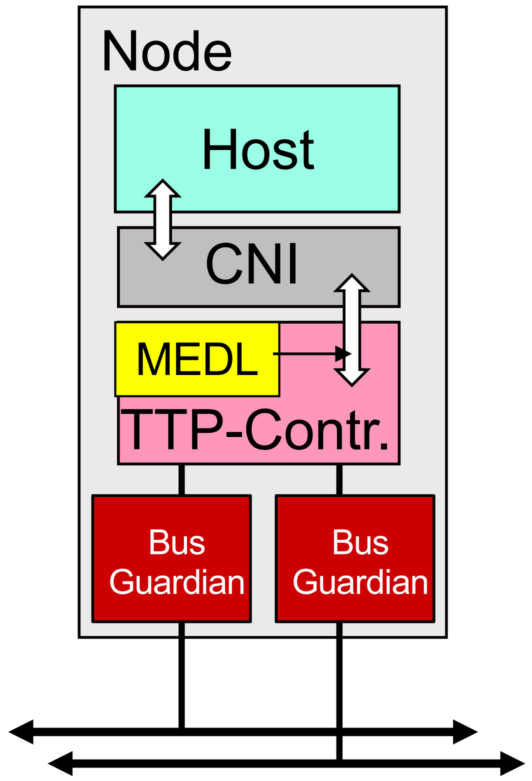
- Architecture level: fault management is based on the assumption that all nodes are fail-silent
- Node level: mechanisms are provided that increase the error detection coverage to justify the fail-silent assumption.

Bus Guardian

- Babbling Idiot: erroneous controller sends at arbitrary times, i.e., outside its assigned time slot
- Babbling idiot violates fault hypothesis (node + comm. medium affected)

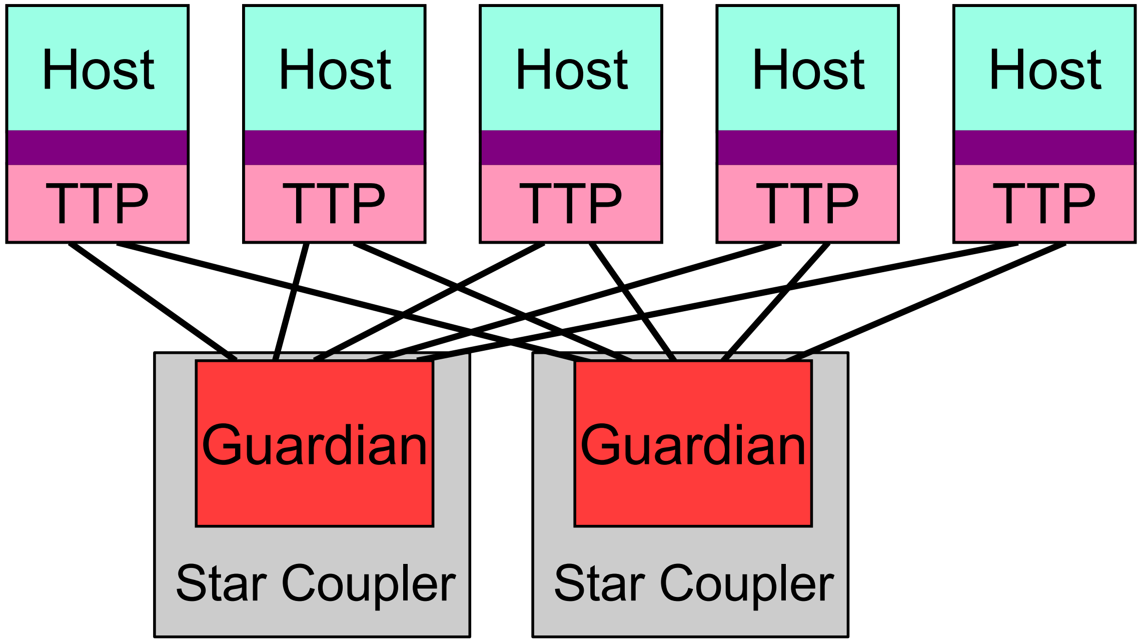
Bus guardian: independent controller for bus access (gate keeping function)

- Knows when node is allowed to send
- Opens gate to bus for sending only during the designated sending slot



TTP with Star Topology

- TTP can be laid out in a star topology
- Bus guardians are only needed in star-coupler switches
- Star coupler deals with slightly-off-specification (SOS) errors
  - SOS errors: inputs that some nodes classify as correct, others as erroneous (e.g., bus-signal voltage at tolerance limit)
  - Unambiguous interpretation and propagation of messages



Continuous State Agreement

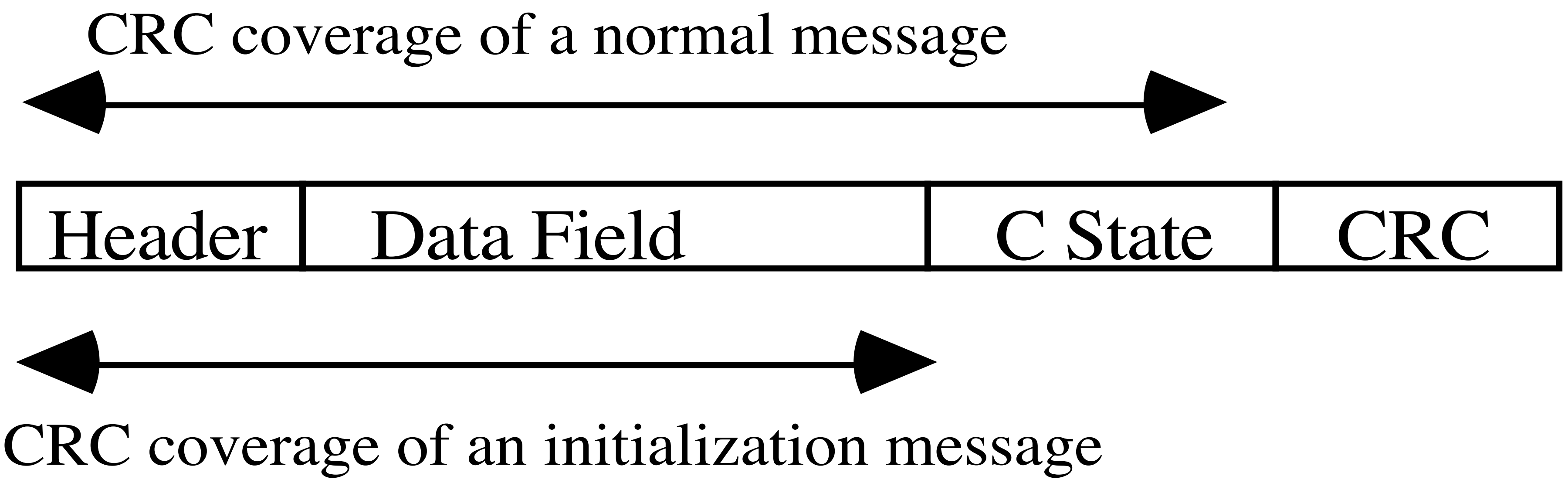
The internal state of a TTP controller (C-state) is formed by

- Time
- Operational Mode (MEDL position), and
- Membership

The Protocol will only work properly, if sender and receiver contain the same state.

Therefore TTP contains mechanisms to guarantee continuous state agreement (extended CRC checksum) and to avoid clique formation (counts of positive and negative CRC checks).

CRC Calculation in TTP


CRC coverage of a normal message
Header
Data Field
C State
CRC
CRC coverage of an initialization message

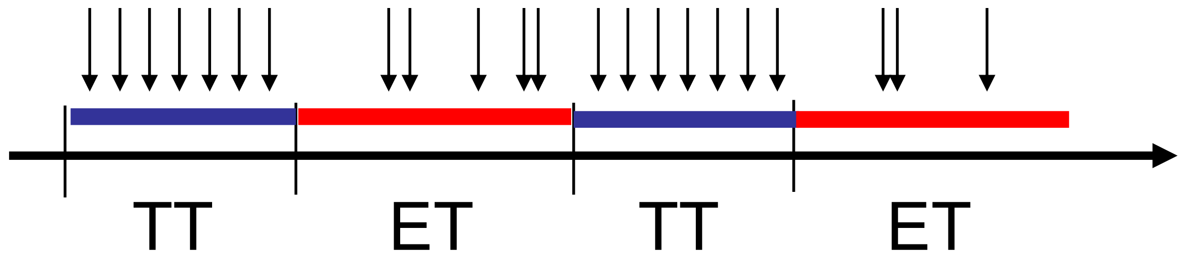
C -State: Time, MEDL Position, and Membership Information at sender respectively at receiver

Clock Synchronization in TTP

- The expected arrival time of a message is known a priori
- The actual arrival time of a message is measured by the controller.
- The difference between the expected and the actual arrival time is an indication for the deviation between the clock of the sender and the clock of the receiver.
- These differences are used by the FTA clock synchronization algorithm to periodically adjust the clock of each node.
- No extra message, no special field within the message needed for FTA clock synchronization.

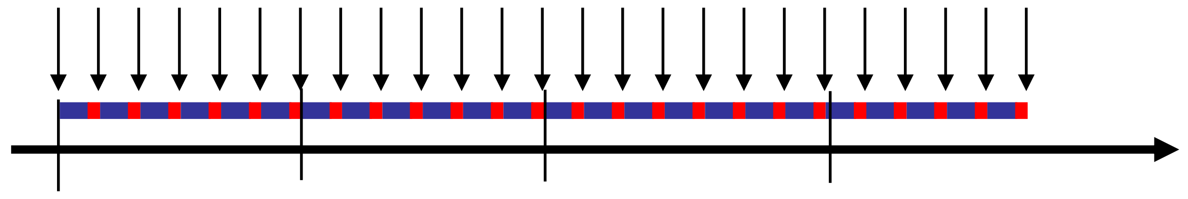
Integrating TT and ET Messages

- Alternating time windows for TT and ET communication
- Two different communication protocols (TT, ET)
- Loss of temporal composability



Integrating TT and ET Messages (2)

- Layered protocol: ET services on top of TT protocol
- Single TT communication protocol
- Loss of global bandwidth as TT messages that transport ET contents are assigned to a-priory defined sending nodes



TTEthernet

TT-Ethernet: fully compatible with existing Ethernet systems in hardware and software:

- Message format in full conformance with Ethernet standard
- Standard Ethernet traffic must be supported in all configurations
- Existing Ethernet controller hardware must support TT Ethernet traffic.
- IEEE 1588 standard for global-time representation is supported

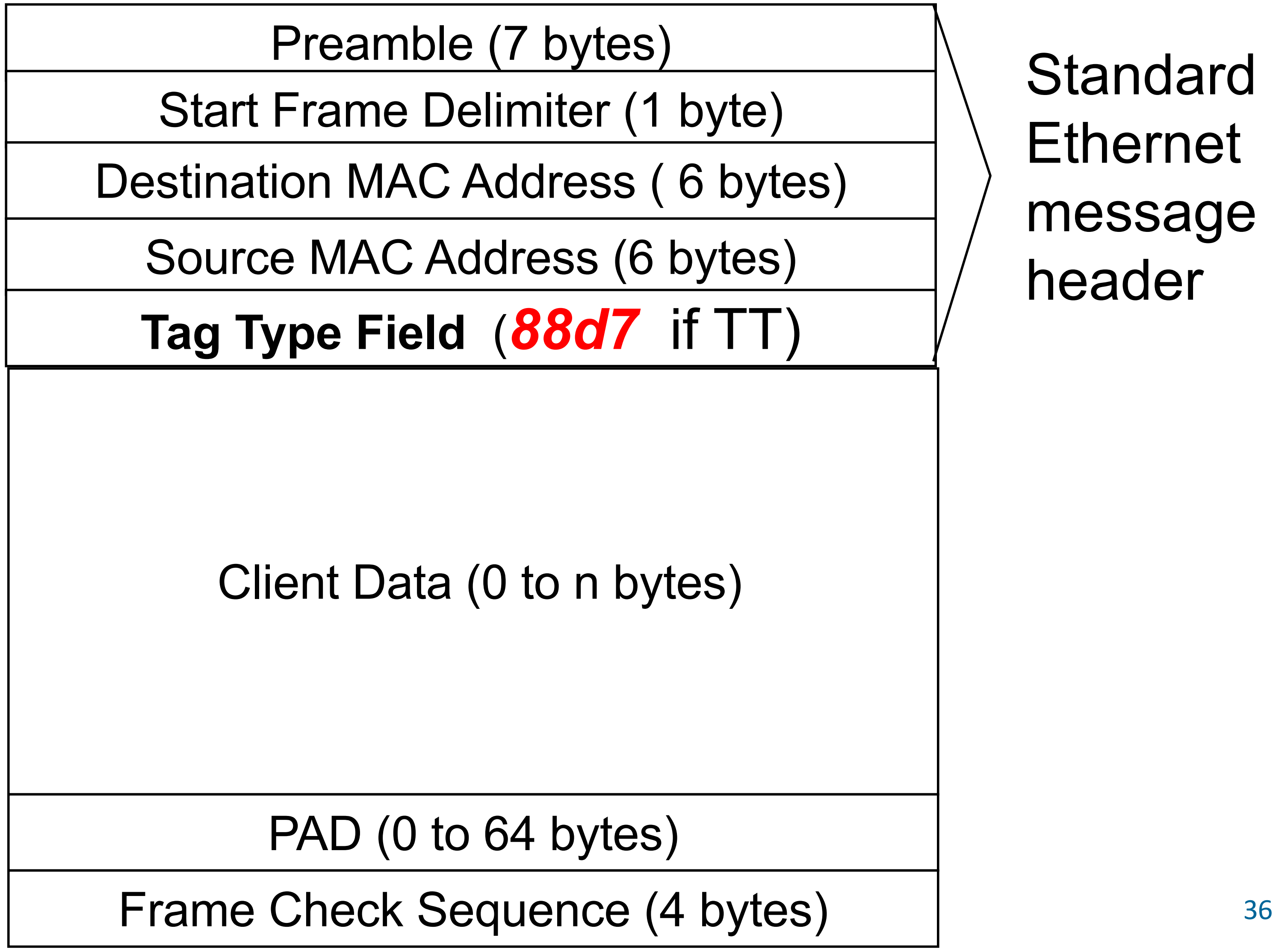
TTEthernet

Provides a *uniform* communication system for all types of distributed non-real-time and real-time applications, from

- very simple uncritical data acquisition tasks, to
- multimedia systems and *up to*
- *safety-critical control applications* (*fly-by-wire*, *drive-by wire*).

It should be possible to upgrade an application from standard TT-Ethernet to a safety-critical configuration with minimal changes to the application software.

TT and ET Message Formats


Preamble (7 bytes)
Start Frame Delimiter (1 byte)
Destination MAC Address (6 bytes)
Source MAC Address (6 bytes)
Tag Type Field (***88d7*** if TT)
Client Data (0 to n bytes)
PAD (0 to 64 bytes)
Frame Check Sequence (4 bytes)
Standard Ethernet message header

Two Categories of Messages

ET-Messages:

- Standard Ethernet Messages
- Open World Assumption
- No Guarantee of Timeliness and No Determinism

TT-Messages:

- Scheduled Time-Triggered Messages
- Closed World Assumption
- Guaranteed *a priori* known latency
- Determinism

Conflict Resolution in TTEthernet

Assumes use of Switched Ethernet

⇨ *TTEthernet switch* acts as a message arbitrator:

- TT versus ET: TT message wins, ET message is delayed.
- TT versus TT: Failure! TT messages are assumed to be scheduled without conflicts (closed world system)
- ET versus ET: One message has to wait until the other is finished (standard Ethernet policy)

There is no guarantee of timeliness and determinism for ET messages!

Points to Remember

Different real-time protocols

Different levels of RT QoS

- Prioritization
- Situation / load dependence
- Limited guarantees
- Encapsulation?

Only TT protocols provide HRTS services

- Timing guarantees & temporal encapsulation
- Dependability support