Time-Triggered Techniques for Quality of Service over SpaceWire

Preliminary Results of WP 200 and WP 300

Tenth SpaceWire Working Group Meeting, ESA/ESTEC

February 20th/21st 2008

Wilfried Steiner
Wilfried.Steiner@tttech.com
TTTech Computertechnik AG

Timeframe: October 2007 – June 2008

Initiator: Philippe Armbruster and David Jameux

ESTEC Contract Number: 21050/07/NL/LvH

R&D programme: TRP (Technology Research Programme)

Budget: 150kEUR

Subcontractor: Vienna University of Technology (20% of Budget)

Workpackages:

- **Requirements & QoS Classes**
- **Real-Time Communication over SpaceWire**
- **Mixed RT & Non- RT Com. Over SpaceWire**
- **Survey of Time-Triggered Protocols**
- **Comparison to SpaceWire**

Overview

Concept of Time-Triggered Service Classes

Clock Synchronization and related Service Classes

- SpaceWire Time-Codes applicability as basic building block

From “Isochronous” Communication to Time-Triggered Communication

- Review of “Isochronous” Communication [Parkes2003]
- Establishing and Maintaining Synchronization

From Time-Triggered Communication to Mixed Time-Triggered Event-Triggered Communication

- The problem of dataflow integration and which components have to solve it.

Fault-Masking Issues and Discussion on Requirements

- Failure Scenarios to be considered
- Standard Fault-Masking and Recovery Techniques

Conclusions and Go-Forward Plan

Concept of Time-Triggered Service Classes

Clock Synchronization and related Service Classes

- SpaceWire Time-Codes applicability as basic building block

From "Isochronous" Communication to Time-Triggered Communication

- Review of "Isochronous" Communication [Parkes2003]
- Establishing and Maintaining Synchronization

From Time-Triggered Communication to Mixed Time-Triggered Event-Triggered Communication

- The problem of dataflow integration and which components have to solve it.

Fault-Masking Issues and Discussion on Requirements

- Failure Scenarios to be considered
- Standard Fault-Masking and Recovery Techniques

Conclusions and Go-Forward Plan

Concept of Time-Triggered Service Classes

The number of time-triggered protocols is increasing.

Examples of time-triggered protocols are

- TTP/C: used in A380, Boeing 787, …
- FlexRay: used in BMW, will be used in Audi A8, …will become the dominant time-triggered standard in the automotive industry
- TT-CAN: low-cost fieldbus
- SAFEbus: used in Boeing 777
- SPIDER (ROBUS): developed by NASA
- Time-Triggered Ethernet: under development by TTTech
- Siemens ProfiNet IRT: used in industrial automation
- Ethernet Powerlink: used in industrial automation

While the time-triggered protocols differ significantly in the algorithms they implement to realize time-triggered communication, there is a common set of problems that has to be solved.

We call this common set of problems the Time-Triggered Service Classes.

Concept of Time-Triggered Service Classes ii

Scheduled Dispatch Service Class: This class specifies methods for time-triggered dispatch of messages according an off-line specified schedule table. This includes the representation of the schedule in the components, e.g. how the schedule is stored in local memory.

Clock Synchronization Service Class: This service class represents services that ensure that the local clocks of the components in the communication infrastructure are synchronized to each other once synchronization is established.

Startup Service Class: The startup service class covers methods and services to initially synchronize the components in the communication infrastructure. This can either be a coldstart procedure or an integration/reintegration procedure.

Concept of Time-Triggered Service Classes iii

External Synchronization Service Class: This service class specifies methods that allow the communication infrastructure to synchronize to an external time source, e.g. GPS.

Clique Detection and Resolution Service Class: This service class defines measures that detect clique scenarios. These are unintended scenarios where disjoint sets of components are synchronized within the subset but not over subset boundaries. Clique Resolution services define methods that re-establish synchronization when cliques have been formed.

Configuration and Maintenance Service Class: This service class defines services on how a communication infrastructure can be configured and maintained. Such services include for example configuration download procedures.

Concept of Time-Triggered Service Classes iv

Membership Service Class: Membership services are low-level diagnosis services that continually monitor the system's health state. In particular such services could reflect which end systems are present in the systems and which are not – for example because of transient/permanent failures.

Gateway Service Class: These services define measures on how message classes with different characteristics can be integrated such that all those message classes can use the same physical medium. In particular the integration of event-triggered and time-triggered messages classes is of interest in this service class.

Legacy Service Class: Existing protocols have interoperability requirements. This service class aims to identify these requirements and provide glue functionality to allow interoperability.

Integrity Service Class: This service class defines services that enhance the integrity of the communication infrastructure. In particular we are interested in two types of integrity measures: a guardian measure that would be either central or local, or end-to-end arguments, such as sequence numbers.

Availability Service Class: This service class defines services that enhance the availability of a communication infrastructure. Such services include redundancy management of communication channels and redundancy management in case of fault-tolerant computation entities such as TMR configurations.

Security Service Class: This service class defines services that enhance the security of a communication infrastructure.

Overview

TTTech

Concept of Time-Triggered Service Classes

Clock Synchronization and related Service Classes

- SpaceWire Time-Codes applicability as basic building block

From "Isochronous" Communication to Time-Triggered Communication

- Review of "Isochronous" Communication [Parkes2003]
- Establishing and Maintaining Synchronization

From Time-Triggered Communication to Mixed Time-Triggered Event-Triggered Communication

- The problem of dataflow integration and which components have to solve it.

Fault-Masking Issues and Discussion on Requirements

- Failure Scenarios to be considered
- Standard Fault-Masking and Recovery Techniques

Conclusions and Go-Forward Plan

Clock Synchronization and related Service Classes

Clock Synchronization Service Class: ***This service class represents services that ensure that the local clocks of the components in the communication infrastructure are synchronized to each other once synchronization is established.***

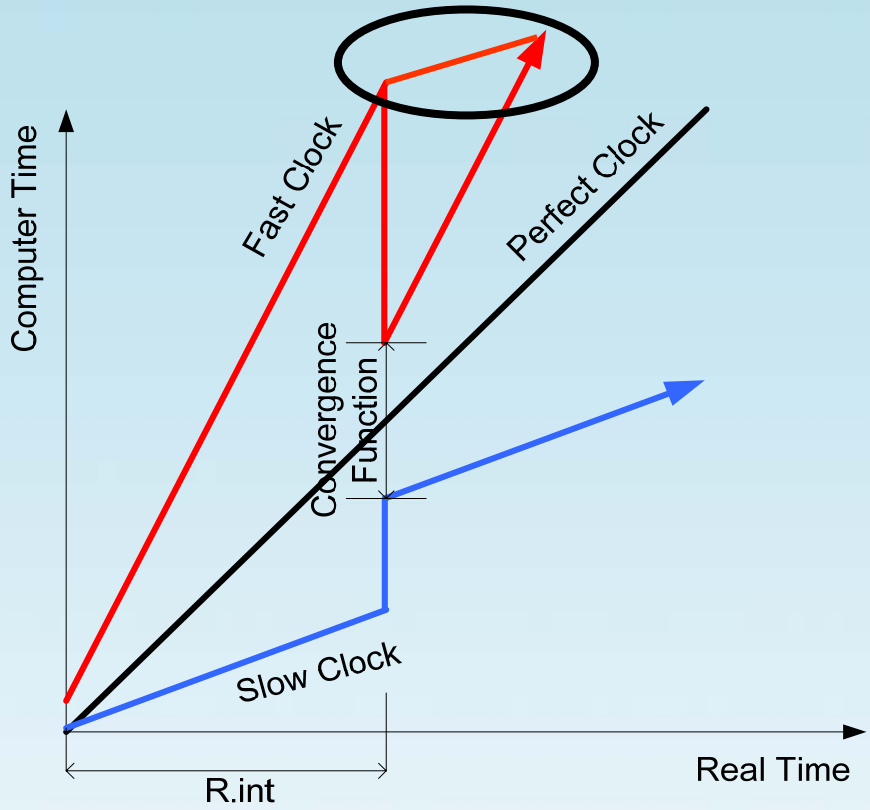
Clocks have to be resynchronized.

R.int is the resynchronization interval.

Convergence Function is the offset of the clocks after clock correction.

Drift Offset = 2 * Drift Rate * R.int

Chronoscopic Correction is also possible



What does that mean in reality:

		Drift Rate (ppm)		Drift Offset (sec)
Frequency (Hz)	R.int (sec)	100		
1000	0.001		0.000000100	0.000000200
500	0.002		0.000000200	0.000000400
200	0.005		0.000000500	0.000001000
100	0.010		0.000001000	0.000002000
50	0.020		0.000002000	0.000004000

Clock Synchronization and related Service Classes iii

Clock Synchronization in SpaceWire

Time can be distributed using a Control Code

- Time-Code: [P 1 1 1 1 0 T0 T1 T2 T3 T4 T5 T6 T7]

Time-Codes have highest priority.

Transmission jitter is bounded by

- the length of one control code (8 Bit)
- the length of one data code (10 Bit) → to be discussed

According to SpW Specification under 7.7

- (d) *"Only one node in a SpaceWire Network should have an active TICK_IN signal."*
- (e) *"All other nodes should keep the active TICK_IN signal not asserted."*

→ SpaceWire specifies Master-Slave Clock Synchronization via Time-Codes.

Pros and Cons of Master-Slave Clock Synchronization Algorithms

Pros:

- Maximum Distance of local clocks in the system is given by:
  - Precision = Latency Jitter + Drift Offset
    = Latency Jitter + 2 * Drift Rate * R.int
  - Latency Jitter is discussed in 8.12 (p) Note 2
  - ST.jitter = 10 N/R
    - N is the number of Links traversed
    - R is the average link operating rate
- Algorithm is simple
- Low overhead in specification, implementation, testing, certification.

Cons:

- No fault-masking, if the master fails:
  - no time is generated at all or
  - a malicious timeline is generated or
  - an interrupted timeline is generated.

If fault masking is not required, Master-Slave is a good solution.

If fault masking is a requirement, Master-Slave has to be enhanced via distributed algorithms.

More Remarks on the Time-Codes:

- SpW Spec. 7.3 (d): *“Six bits of time information shall be held in the least significant six bits of the Time-Code (T0 – T5) and the two most significant bits (T6, T7) shall contain control flags that are distributed isochronously with the Time-Code.”*
- 6 bits = 64 different Time-Codes
  - A single time-code is the minimum needed for time-triggered communication (details later).
- Sheynin and Gorbachev propose to split the Time-Code into
  - Time-Code
  - Interrupt-Code
  - Interrupt-Acknowledgment Code

Clock Synchronization and related Service Classes vi

Startup Service Class: The startup service class covers methods and services to initially synchronize the components in the communication infrastructure. This can either be a coldstart procedure or an integration/reintegration procedure.

Startup becomes important when moving to distributed algorithms.

- Not to rely on the correct functionality of a single component.

The startup algorithm establishes an initial synchronization.

The startup algorithm is sometimes also referred to as the actual Protocol State Machine of a time-triggered protocol.

- For example in FlexRay and in TTP.

→ The Startup Algorithm would be one major part of an upcoming TT over SpW if fault masking is required.

Overview

Concept of Time-Triggered Service Classes

Clock Synchronization and related Service Classes

- SpaceWire Time-Codes applicability as basic building block

From “Isochronous” Communication to Time-Triggered Communication

- Review of “Isochronous” Communication [Parkes2003]
- Establishing and Maintaining Synchronization

From Time-Triggered Communication to Mixed Time-Triggered Event-Triggered Communication

- The problem of dataflow integration and which components have to solve it.

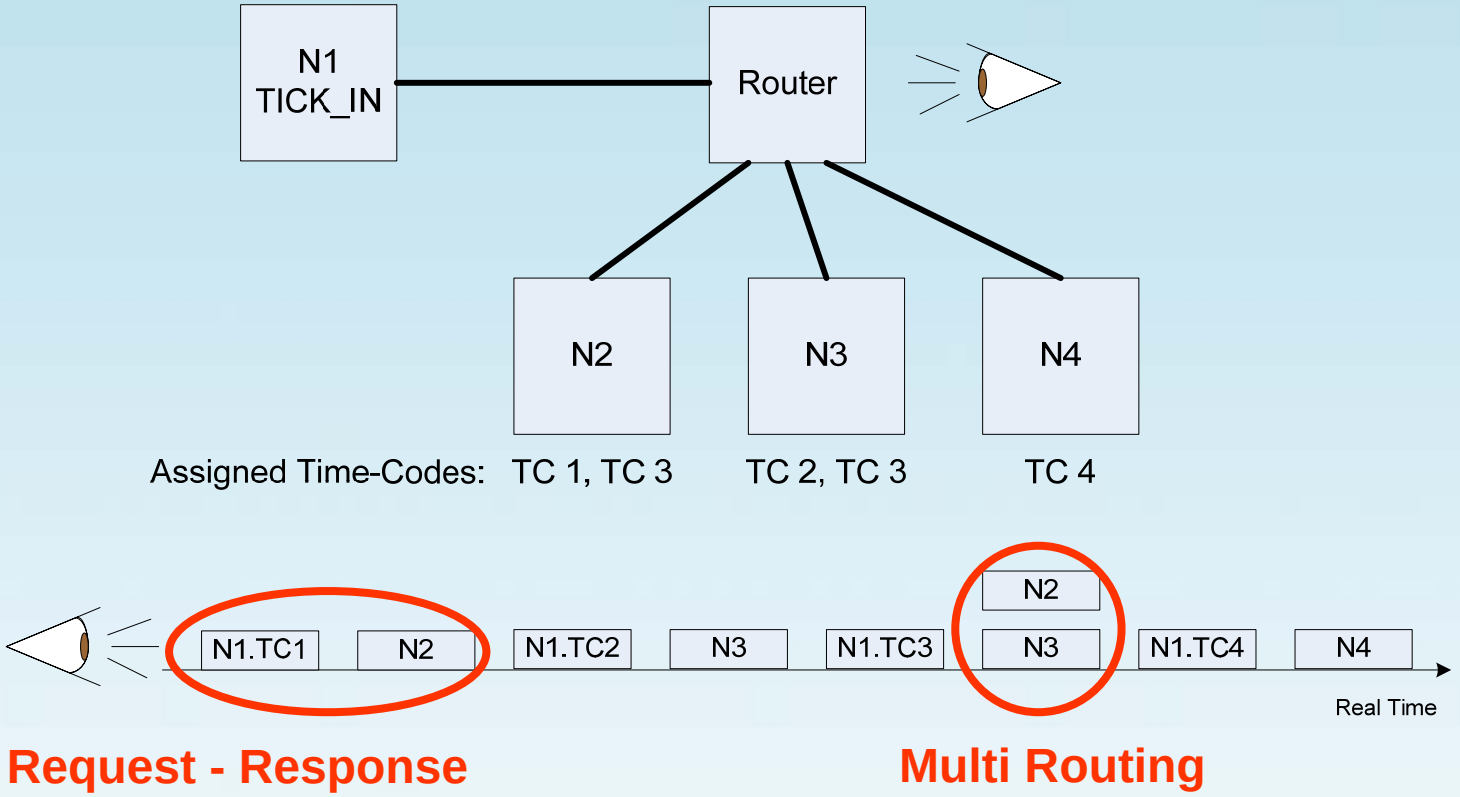
Fault-Masking Issues and Discussion on Requirements

- Failure Scenarios to be considered
- Standard Fault-Masking and Recovery Techniques

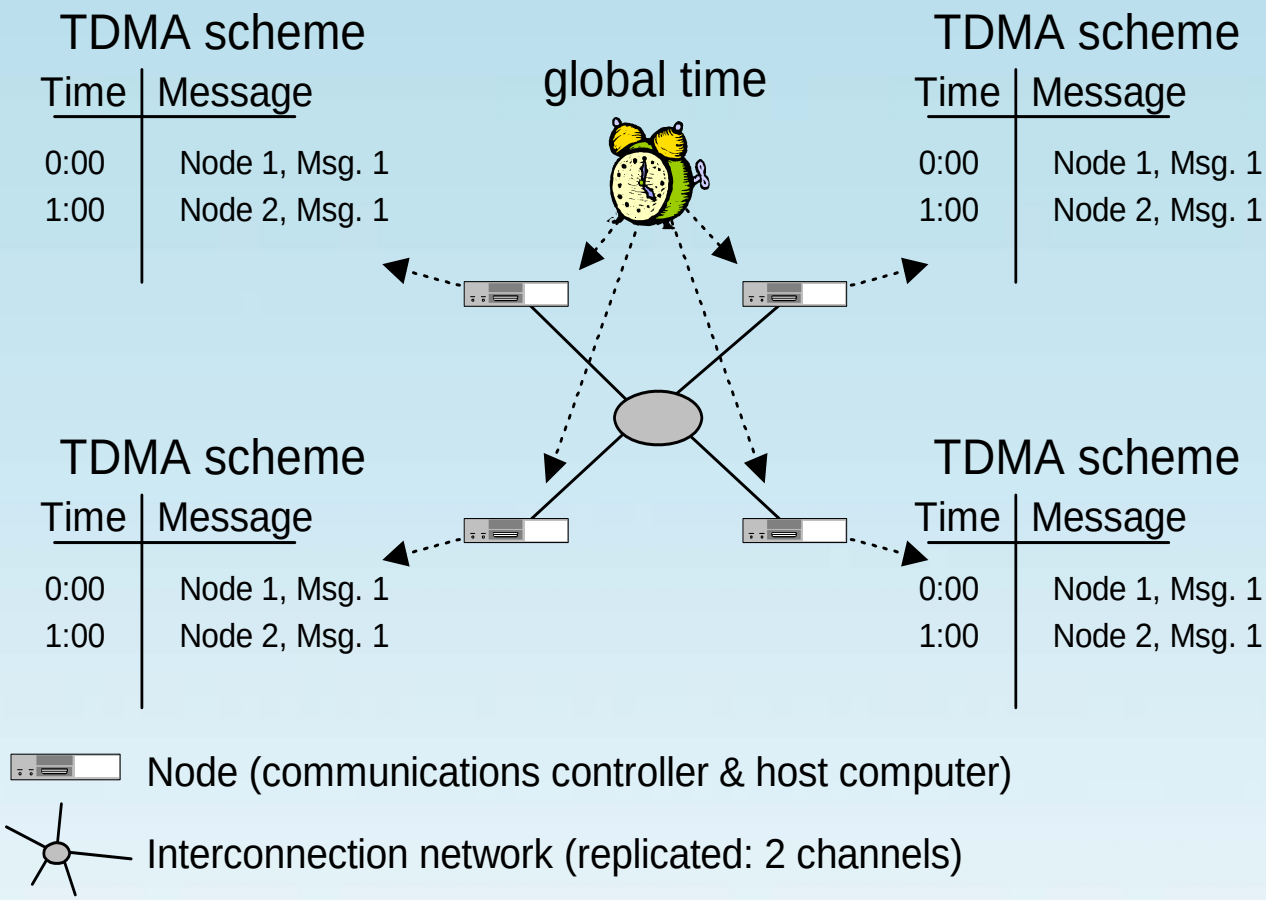
Conclusions and Go-Forward Plan

Isochronous Communication [Parkes2003]

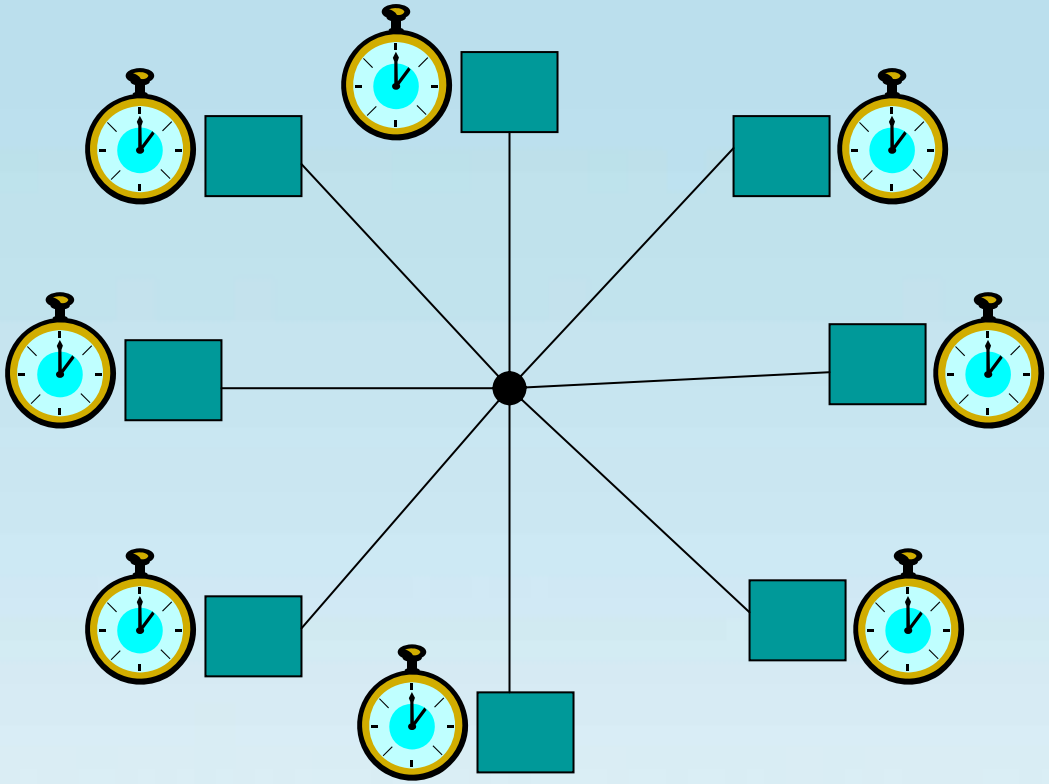
[Parkes2003] "Every time a time-code is received one node gets the chance to send out an isochronous packet."



Time-Triggered Communication



Time-Triggered Communication ii



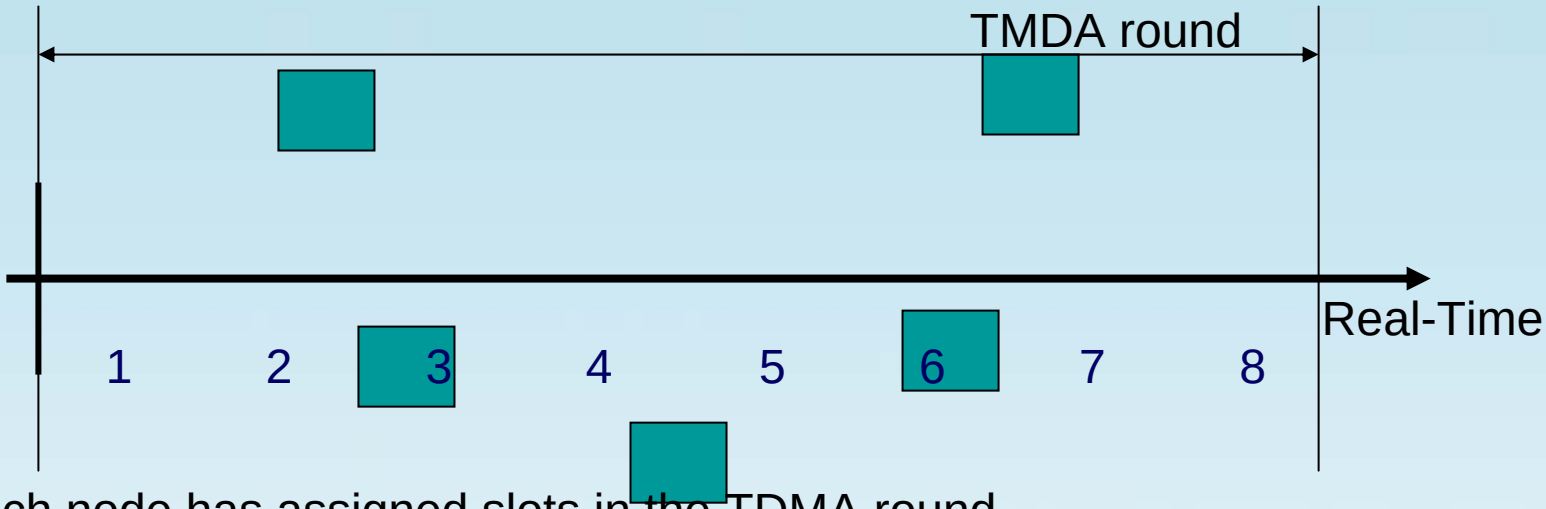
- A system is depicted that consists of eight nodes that are connected in Star Topology.

- Each node maintains a local clock and the local clocks are synchronized.

- Basic Concept:

- Time is split-up into (not necessarily equal) slots depending on message length

- Slots are grouped into TDMA rounds



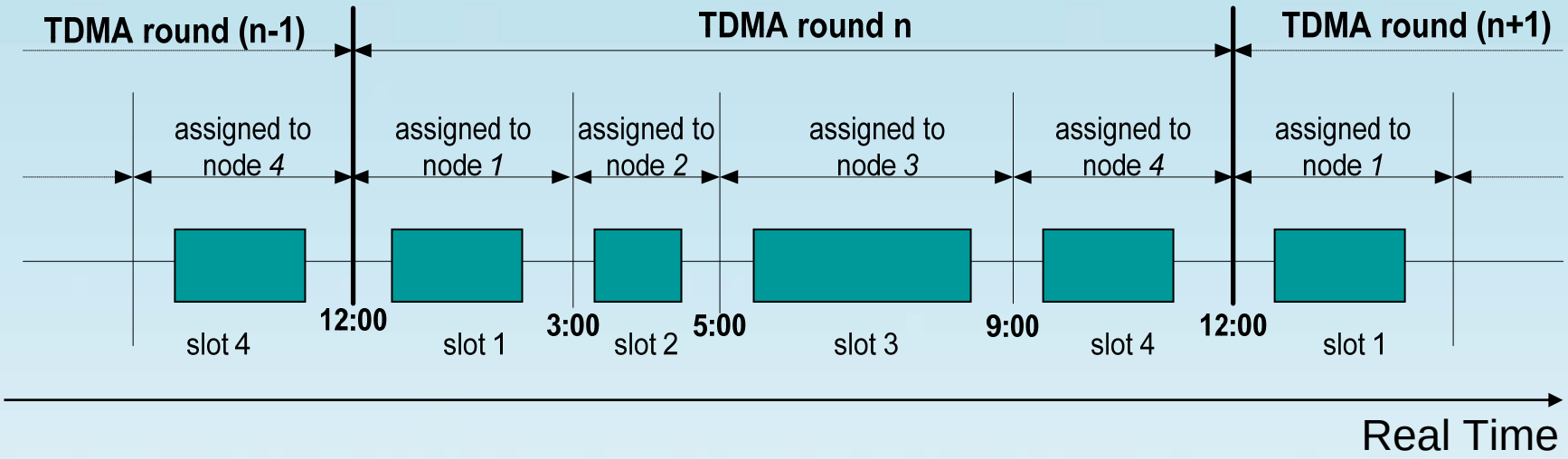
- Each node has assigned slots in the TDMA round

- This assignment is identical for every TDMA round

- Actual Transmission Phase < Sending Slot

Time-Triggered Communication iv

- Example of a TDMA schedule with four nodes:



- A minimum distance between transmission phases is necessary, if messages shall not overlap.

How is time-triggered different to isochronous:

- Global time base is established first.
- All nodes (and optionally routers) are synchronized to each other.
- Trigger to send a message is derived from the local time in a node that reflects the global time base.
- Protocol messages are used to establish and maintain the global time.

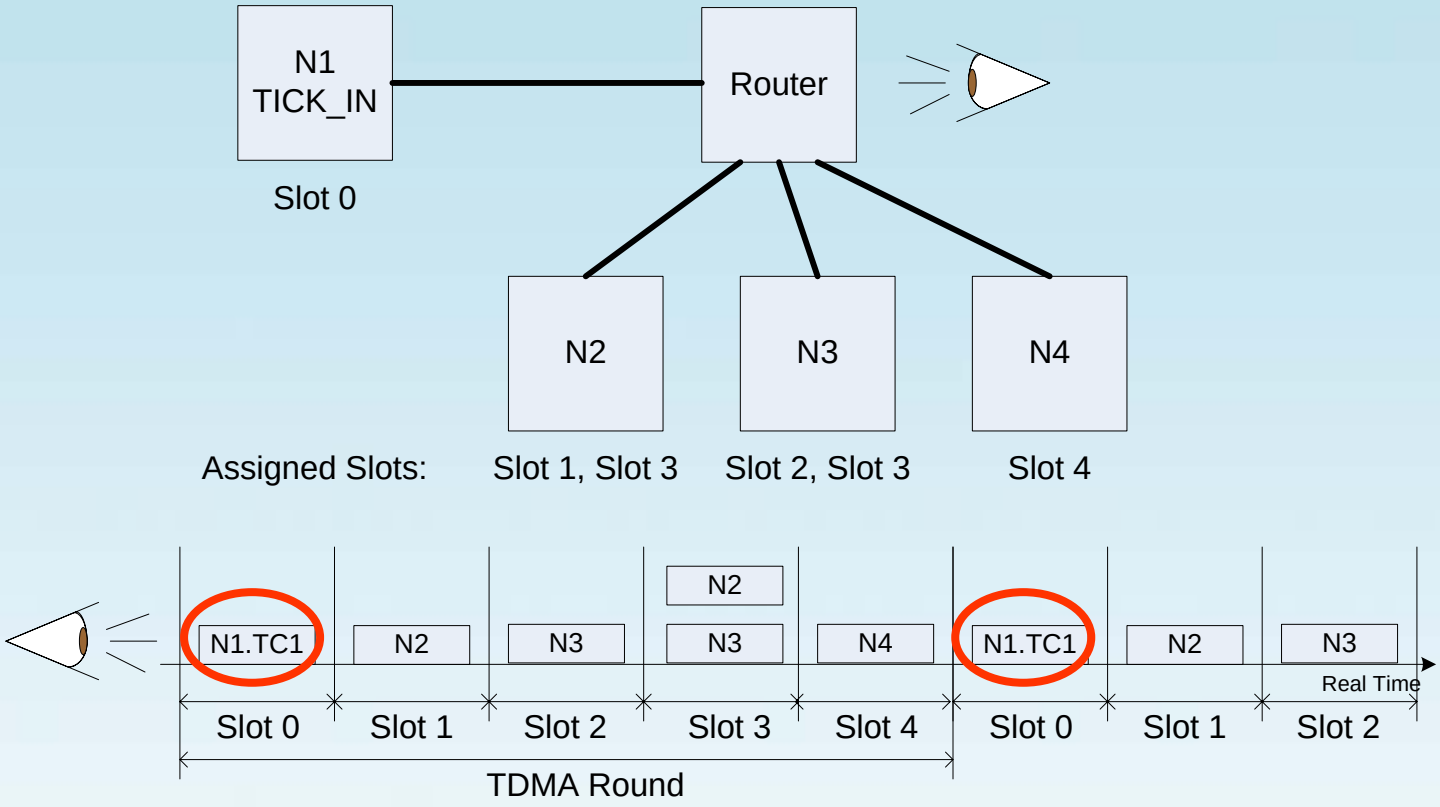
Time-Triggered Communication over SpaceWire

- Can Use Time-Codes for Clock Synchronization.
- Time-Codes may reflect only integration points in the schedule, rather than reflect individual slots.
- Minimum number of Time-Codes needed for time-triggered communication is one.

In this example we take minimal advantage of the Time-Code concept.

- Only one Time-code is used to establish and maintain a global time base.

Different Time-Codes can be used to let nodes/routers integrate inside a TDMA round.



Time-Code Signals TDMA Round Start

Overview

Concept of Time-Triggered Service Classes

Clock Synchronization and related Service Classes

- SpaceWire Time-Codes applicability as basic building block

From "Isochronous" Communication to Time-Triggered Communication

- Review of "Isochronous" Communication [Parkes2003]
- Establishing and Maintaining Synchronization

From Time-Triggered Communication to Mixed Time-Triggered Event-Triggered Communication

- The problem of dataflow integration and which components have to solve it.

Fault-Masking Issues and Discussion on Requirements

- Failure Scenarios to be considered
- Standard Fault-Masking and Recovery Techniques

Conclusions and Go-Forward Plan

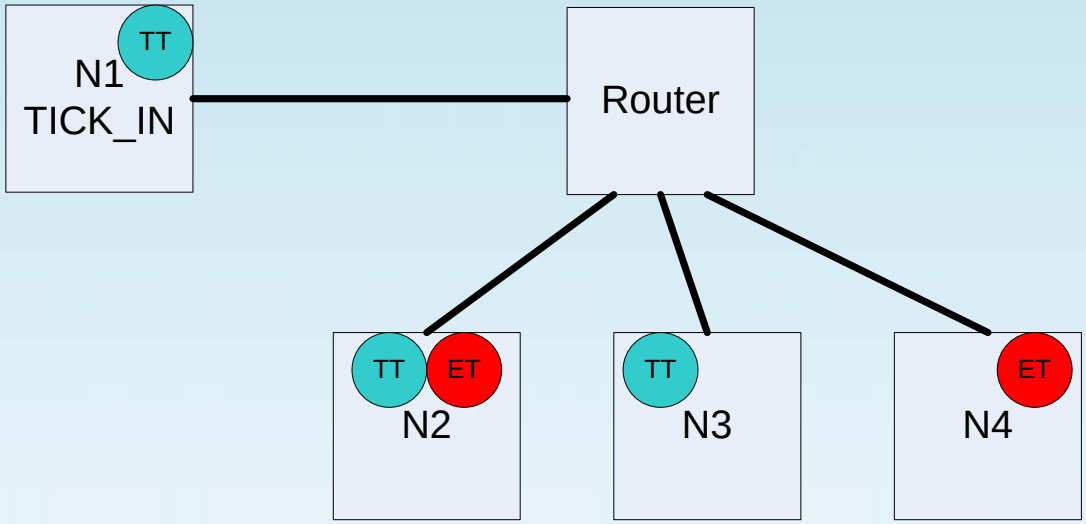
Mixed ET-TT Communication

Which components use time-triggered dataflows?

Is it a problem to force each node to be “global-time aware”?

What are the real temporal requirements on event-triggered messages?

Are there different event-triggered and different time-triggered dataflows?



Mixed ET-TT Communication ii

Option A – All nodes are synchronized.

- Option A.1: Nodes tunnel their event-triggered messages in time-triggered messages
  - e.g. CAN over TTP
- Option A.2: The schedule specifies a “long slot”
  - e.g. Dynamic segment in FlexRay

Option B – Router is doing the dataflow integration

- Option B.1: Event-triggered messages get lower priority to time-triggered messages.
- Option B.2.: Clairvoyant and dynamic integration of event-triggered messages into a time-triggered schedule
  - e.g. Dataflow integration in Time-Triggered Ethernet

→ **No claim for completeness for the listed options.**

Overview

Concept of Time-Triggered Service Classes

Clock Synchronization and related Service Classes

- SpaceWire Time-Codes applicability as basic building block

From "Isochronous" Communication to Time-Triggered Communication

- Review of "Isochronous" Communication [Parkes2003]
- Establishing and Maintaining Synchronization

From Time-Triggered Communication to Mixed Time-Triggered Event-Triggered Communication

- The problem of dataflow integration and which components have to solve it.

Fault-Masking Issues and Discussion on Requirements

- Failure Scenarios to be considered
- Standard Fault-Masking and Recovery Techniques

Conclusions and Go-Forward Plan

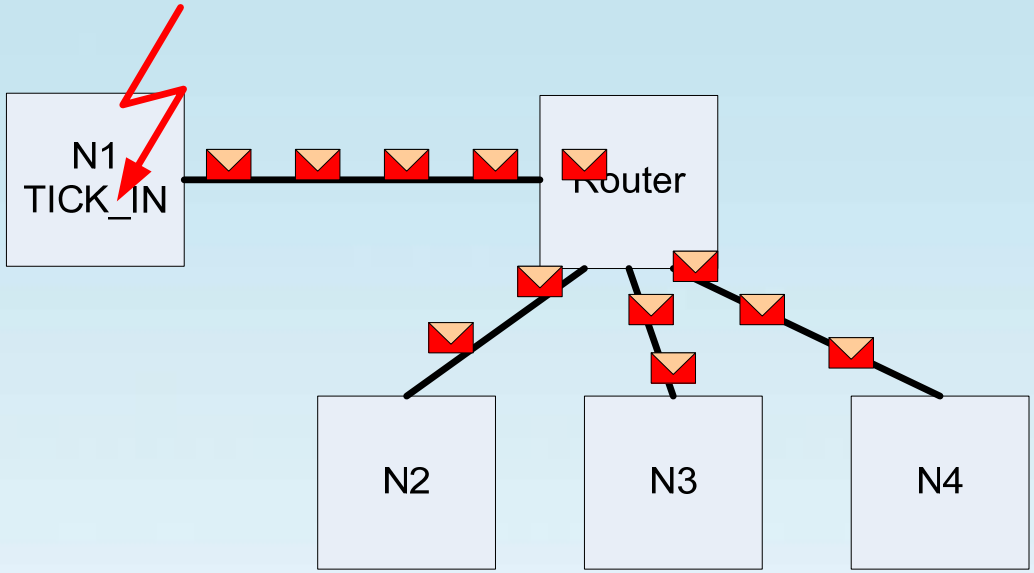
Failure Scenarios to be considered

Babbling Idiot Failure

A node sends arbitrary packets with an arbitrary frequency on the network.

Scenario: TICK_IN Node is the babbler and sends random Time-Codes.

- SpaceWire has already some end-to-end checks built in, but may not be sufficient.



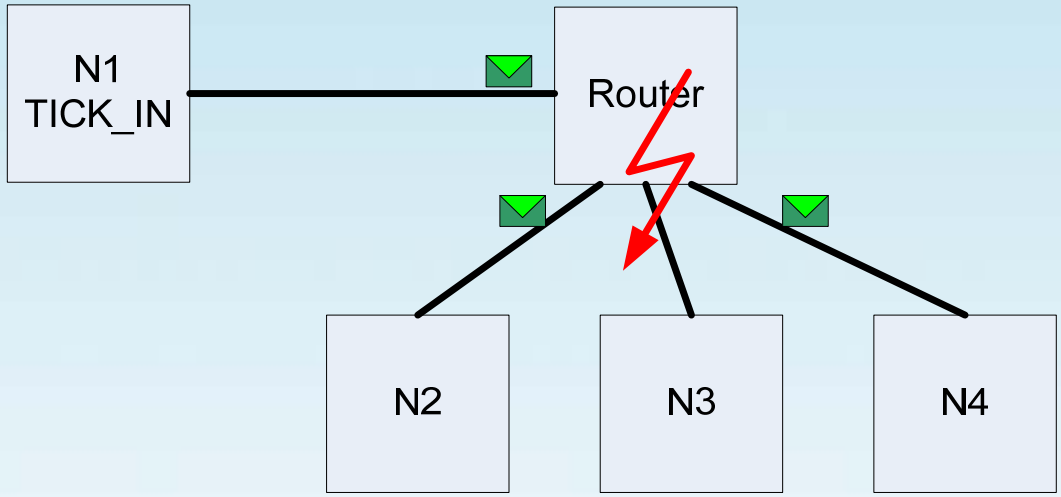
→ What happens if another node, say N3 starts sending malicious Time-Codes ?

Failure Scenarios to be considered ii

Inconsistent and/or delayed relay

The router fails to relay the message in a correct way.

Scenario: The Time-Codes are relayed correctly to a subset of nodes, while another subset receives the time-code delayed.



→ What happens if the router starts sending malicious Time-Codes ?

Fault-Masking Techniques

- Triple Modular Redundancy (TMR):
  - Replication of nodes / channels
  - A message will only be considered correct if two replicas are in agreement.
- Local Guardian Mechanisms / Self-Checking Component Design:
  - Aims to restrict failure modes to benign via Chip IP design.
- Central Guardian Mechanism
  - Switch/Router controls correctness of messages sent by nodes.

Recovery Techniques

- Self
  - -stabilization, -healing, -reconfiguration Techniques
- e.g. Clique detection and resolution

→ A combination of techniques may also be used.

Overview

- Concept of Time-Triggered Service Classes
- Clock Synchronization and related Service Classes
  - SpaceWire Time-Codes applicability as basic building block
- From “Isochronous” Communication to Time-Triggered Communication
  - Review of “Isochronous” Communication [Parkes2003]
  - Establishing and Maintaining Synchronization
- From Time-Triggered Communication to Mixed Time-Triggered Event-Triggered Communication
  - The problem of dataflow integration and which components have to solve it.
- Fault-Masking Issues and Discussion on Requirements
  - Failure Scenarios to be considered
  - Standard Fault-Masking and Recovery Techniques
- **Conclusions and Go-Forward Plan**

Conclusions and Go-Forward Plan

Time-Triggered Communication over SpaceWire is possible.

- A straight forward way is in using the Time-Code Concept.

Mixed Time-Triggered Event-Triggered Communication over SpaceWire is possible.

- For example by configuring a “long slot” for event-triggered communication.
- Size of this “long slot” may impact the precision in the system.

Crucial point is fault masking.

- If non-benign failure modes have to be tolerated, then distributed algorithms have to be used.
- These algorithms can run as a layer on top of SpaceWire.
- Algorithms would either have to be developed from scratch or re-used from existing standards (or standards that are currently under development).

→ Achieving fault masking is not trivial and should not be underestimated.

Thank You!

Any Questions?