Objective

Use a single communication architecture for Onboard Data Handling, multiplexing:
1. Payload data
   Requires high throughput
2. Command and control
   Requires guaranteed latency

Problem

Throughput and latency are conflicting requirements

Solution

Determinism provides guarantees
Scheduling provides determinism

Questions

How can be implemented?
Are there other solutions?
Terms in context

messages / time → throughput

delay → latency

segment

packet

flow

Virtual Channels (VC)

Deterministic → Predictable → No randomness

TDM (schedule)

slot slot slot time
Determinism

SpaceWire is a *deterministic* network
  – There are no random variables in the standard

*BUT the traffic is usually not.*

Predictability or *determinism* of *latency* and *throughput* (metrics) *depends on the traffic injected* to the network.

1. Traffic injected is *deterministic* (known at any time)
   • Precise results can be obtained by simulation.

2. Traffic injected is *stochastic* (random) but the statistics are known
   • Metrics can be obtained by simulation.
     – i.e. Average latency and throughput
   • An upper bound of the worst case latency can be computed

3. Traffic injected is *unknown* but maximum packet length is fixed.
   • An upper bound of the worst case latency can be computed
## Determinism (2)

<table>
<thead>
<tr>
<th>Network</th>
<th>Traffic</th>
<th>System</th>
<th>Maximum latency</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethernet</td>
<td>Deterministic</td>
<td>Stochastic</td>
<td>Stochastic</td>
<td>Bounded</td>
</tr>
<tr>
<td>SpaceWire</td>
<td>Deterministic</td>
<td>Deterministic</td>
<td>Deterministic</td>
<td>Deterministic</td>
</tr>
<tr>
<td><strong>SpaceWire</strong></td>
<td>Stochastic</td>
<td>Stochastic</td>
<td><strong>Bounded</strong> (by message size, topology and # flows)</td>
<td>Bounded</td>
</tr>
<tr>
<td>SpaceWire &amp; virtual networks</td>
<td>Stochastic</td>
<td>Stochastic</td>
<td><strong>Bounded</strong> (by message size, topology and # flows in same VN)</td>
<td>Bounded</td>
</tr>
<tr>
<td>SpaceWire &amp; segmentation</td>
<td>Stochastic</td>
<td>Stochastic</td>
<td><strong>Bounded</strong> (by segment size, topology and # flows)</td>
<td>Bounded</td>
</tr>
<tr>
<td>SpaceWire &amp; schedule</td>
<td>Stochastic</td>
<td>Stochastic</td>
<td>Deterministic</td>
<td>Deterministic</td>
</tr>
</tbody>
</table>
Upper bound computation

Flows
1. Payload Data (4 Kbytes, 204µs)
2. Command reply (128B, 6µs)

Apply worst case

Max Latency flow 4 = \( D(\text{flow 3}) + D(\text{flow 4}) \)

\[
D(\text{flow}, \text{hop}) = T(\text{flow}, \text{hop}) + \sum_{h \in \text{hops}} \sum_{p \in \text{input ports}} \max[D(f, h)] \quad \forall f \text{ using } p \text{ and } (h+1)
\]

Latency without congestion
Latency due to congestion produced by other flows

\[D(\text{flow 3}) = 204 + (204 + 204) + 204 = 816\mu s\]
\[D(\text{flow 4}) = 6 + (204 + 204) + 204 = 618\mu s\]

Max Latency flow 4 = 7*204+6 = 1.434ms
Virtual Networks

Implements virtual channels *mainly only* in routers

- Nodes usually send both data & control
- Nodes require modification to have Virtual Channels

Latency of commands depend on payload data latency

- Maximum latency can be worse than using SpW with segmentation

<table>
<thead>
<tr>
<th>Previous Network</th>
<th>Virtual networks (VCs only in routers)</th>
<th>Segmentation (256B,12µs)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>204*4 + 6 = <strong>822µs</strong></td>
<td>12*7+6 = <strong>90µs</strong></td>
</tr>
</tbody>
</table>

If nodes have Virtual Channels or an extra link, control messages still have to compete between themselves

- Latency is highly improved but still **no guarantees in Throughput**
Segmentation

*Decouples packet length from message length*

- Packet size is constant and equal to segment size

**Benefits**

- **Reduce the upper bound** of the maximum latency.
- Improves average latency for small messages.
- Bounded packet sized is required for TDM
Scheduling

Removes network congestion

Benefits

• Command and control
  – Worst case **latency can be assigned** per each flow
  • By a suitable schedule and topology.
  – High priority flows are not coupled with lower priority
  • i.e. Latency of command and control does not depend on the data payload traffic

• Payload data
  – Minimal **throughput can be assigned** per each flow
Scheduling options

1. Centralized
   - A Master Node initiates all transaction of the network following a schedule

2. Distributed
   - Multiple initiator nodes are **synchronized** and follow a local schedule.
     a) A single initiator can be active at any time
     b) Multiple initiators can execute transactions concurrently if they do not use the same network resources.
     - For each slot, each initiator is allowed to send to a single or multiple specified destinations
### Scheduling options (2)

<table>
<thead>
<tr>
<th>Centralized</th>
<th>Distributed</th>
<th>Transactions during slot i</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mater node</strong>&lt;br&gt;(Mil bus)</td>
<td><strong>Single initiator</strong>&lt;br&gt;(SpW-RT)&lt;br&gt;Multiple initiators&lt;br&gt;Multiple destinations&lt;br&gt;(not possible with E2E flow control)</td>
<td>Lower latency&lt;br&gt;Higher throughput&lt;br&gt;More complexity</td>
</tr>
<tr>
<td></td>
<td><strong>Multiple initiators</strong>&lt;br&gt;<strong>Single destination</strong></td>
<td></td>
</tr>
</tbody>
</table>
End to end (E2E) flow control

Data loss if producer data rate > consumer data rate & consumer buffer gets full

Solutions

1. Source data rate is limited by design to be less than the consumer rate.
   - Best option if consumer rate is unlimited or data is periodic control status

2. Consumer pulls data from the producer
   - Easy option if source buffer unlimited

3. Consumer regulates the data rate of the producer (E2E flow control)
   - May be required between processing units.
## Comparing Features

<table>
<thead>
<tr>
<th>Network</th>
<th>Scheduled</th>
<th>Priorities</th>
<th>Segmentation</th>
<th>E2E</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAN</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Rapid IO</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SpW</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>CAN bus</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>MIL bus</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>FlexRay</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>SpW-RT</td>
<td>Yes/No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>SpW-VN</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
Design flow

Scheduled

Requirements
- Minimum throughput
- Maximum Latency
- Traffic

System architecture
- Topology
- Schedule

No scheduled

System architecture
- Topology
- Traffic

Simulation & analysis
- Change topology, segment size or routing
- Requirements not met

Requirements Verification
- Minimum throughput
- Maximum Latency
- Traffic
No scheduled networks

Upper bounds of the worst case latency can be computed
• Only topology and maximum packet size per flow is required
• Segmentation & virtual channels reduce upper bounds of the latency

Throughput figures can only be obtained by simulation
• Traffic statistics are required
• Variable network throughput could impact buffers utilization and system operation

Scheduled networks

Latency and throughput requirements are assigned at design time
• Maximum latency and minimum throughput per flow is deterministic

Complexity and performance of the network depends on the schedule mechanism implemented
# Conclusions

**How to multiplex data payload & Control?**

<table>
<thead>
<tr>
<th>User case</th>
<th>Proposed solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small network (2 routers)</td>
<td>Use SpW as it is and calculate the latency upper bounds</td>
</tr>
<tr>
<td>Low requirements (&gt;1ms)</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Small network (2 routers)</td>
<td>Use segmentation &amp; perform network simulations</td>
</tr>
<tr>
<td>Strict requirements (&lt;1ms)</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Middle-size network</td>
<td>Uses a <strong>synchronous</strong> network with a suitable schedule scheme</td>
</tr>
<tr>
<td>Strict requirements (including <strong>throughput guarantees</strong>)</td>
<td></td>
</tr>
<tr>
<td>Large network (&gt;5 routers)</td>
<td>Implement <strong>virtual channels</strong> in all nodes and routers (i.e. Virtual Networks with multi-priority nodes)</td>
</tr>
<tr>
<td>or very strict requirements on latency (&lt;50 µsec)</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>

"Make it as simple as possible, but not simpler"

Albert Einstein
Support Slides
Definitions

• **Data integrity** (CRC)
  − Valid data received is guaranteed to be correct
  − Data sent can be lost or be invalid.

• **TDM** (Time Division Multiplexing)
  − The time domain is divided into several recurrent time-slots of fixed length. Time-slots are recurrent in a fixed order and pre-allocated using a schedule.

• **Deterministic** (predictable)
  − System is predictable and produce the same output for a given starting condition, i.e. there is no randomness involved.

• **Throughput** (user data rate)
  − Average rate of successful message delivery over a communication channel
  − The maximum throughput is equal to channel bandwidth if there are no packet losses or protocols overheads.
  − **Channel utilization** is the percentage of the achieved throughput related to the maximum throughput of a channel (or link).
Definitions (2)

• **Latency** (end-to-end delay)
  – Expression of how much time it takes for a packet of data to get from one designated point to another (starting at packet transmission).

• **Flow** (SpW-RT channel)
  – A sequence of packets sent from a particular source to a destination that the source labels as a flow (different flows may have same source and destination)

• **Packet**
  – Formatted unit of data carried by the SpaceWire network layer.

• **Message**
  – User data unit of information
  – **Segment**: Message fragment smaller than the maximum packet size.

• **Node**
  – Device that have a single configuration port with one or multiple SpaceWire interfaces.
  – **Initiator** node: Node that starts a transaction
  – **Target** node: Node that receives a request or message from a initiator
  – **Master** node: Node that arbitrates all communications in the network
SpaceWire characteristics

– Provides high bandwidth and very low minimum packet latency.

– Wormhole switching produce cascade packet congestion if network resources (links) are simultaneous requested by more than one packet.

– Congestion reduce throughput and increase packet latency
Encapsulation

Segments must be encapsulated

– There are two options

• Encapsulate using a new protocol
  – Current devices do not support it by hardware
    » Increases CPU usage

• Encapsulate using an RMAP packet
  – Can benefits from current RMAP hardware implementations
  – A specific mechanism must be provided to notify to the destination that a message has been delivered, including the message size (not equal to the packet size )
    » This is not required if only remote memory R/W is performed.