


# SpW–D preliminary Protocol Implementation & Analysis

Parameter Identification and Trade–Off

Albert Ferrer–Florit, University of Dundee / TEC–EDP

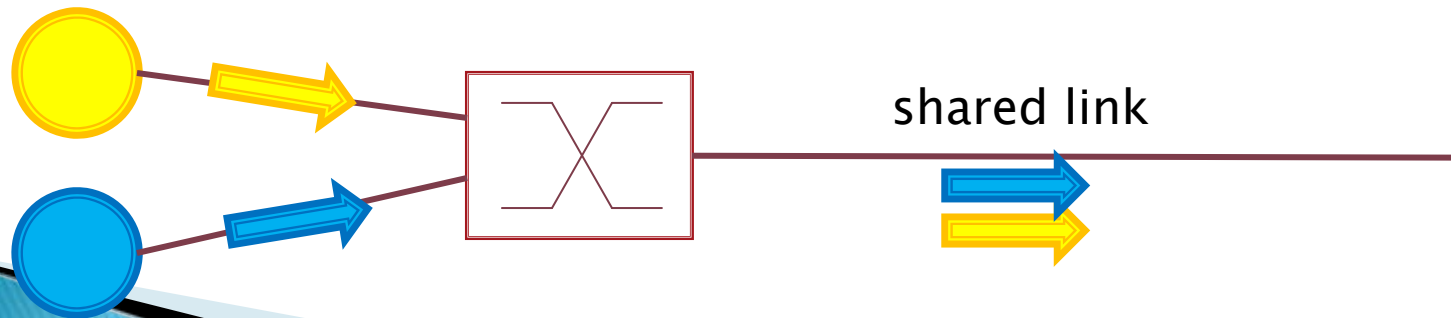
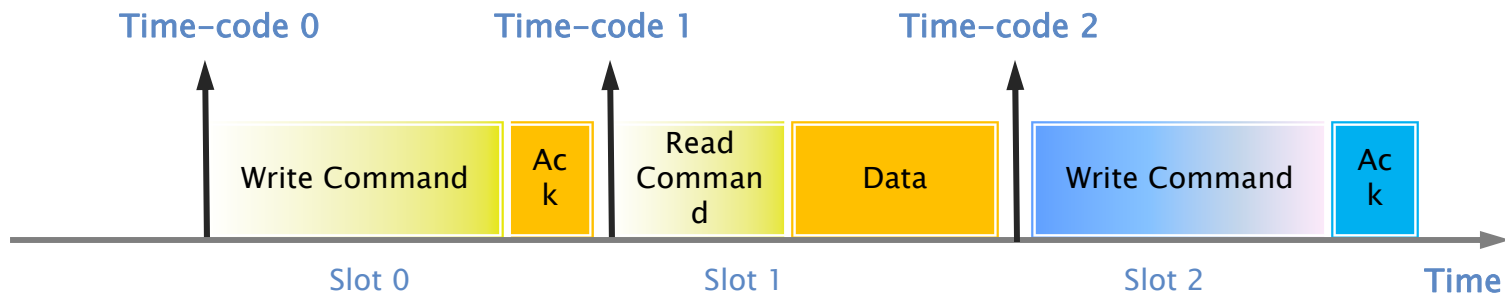


# SpW-D Objectives

- ▶ Provide guaranteed/deterministic latency and throughput, with:
    - High data rate for Payload Data
    - Low latency for Command and Control operations.
  - ▶ Reuse existing SpaceWire devices and protocols
  - ▶ Make it efficient
  - ▶ Make it simple
- 

# SpW-D Overview

- SpW-D provides deterministic packet delivery to SpaceWire networks using time-slots and RMAP transactions
- Time-slots are equally spaced in time, in which a single RMAP transaction can take place.

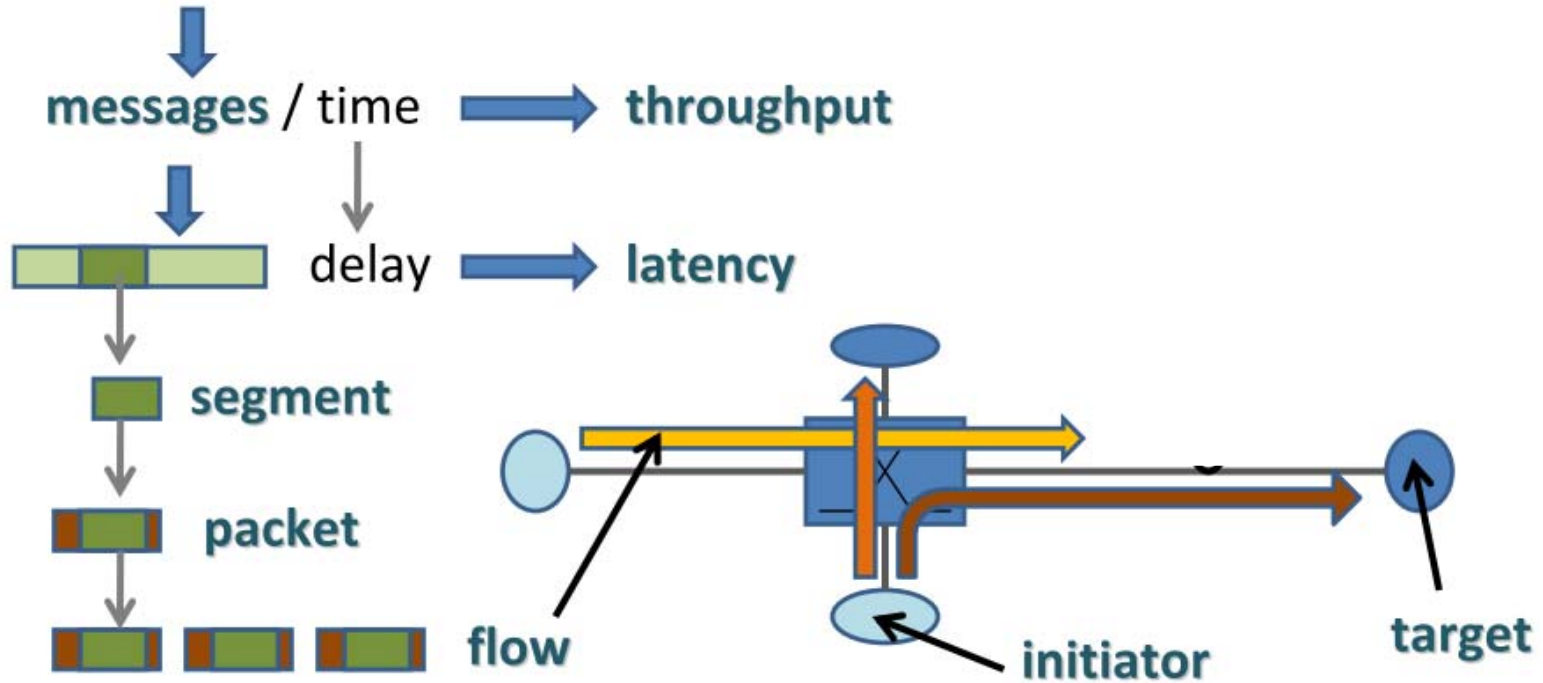


# SpW-D Advantatges

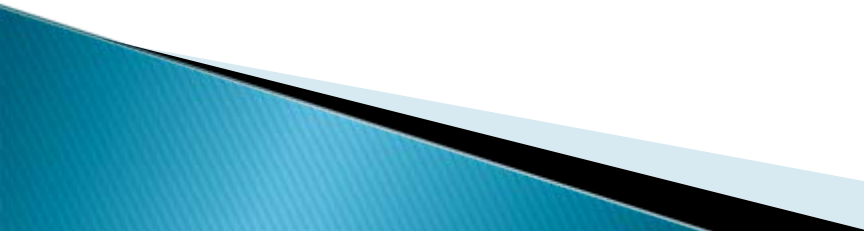
- Why RMAP? An RMAP transaction allows to **read or write** to an address space (**memory or FIFO** ) located in a remote unit using SpaceWire links. It implements **acknowledgments**.
- Why scheduling? Deterministic behaviour provides **guaranteed** maximum **latency** for control messages and **bandwidth** allocation for payload data.

With scheduling... There is NO congestion!

# Terms in context



# SpW–D performance and parameters

- ▶ Performance depends on the following parameters:
    1. Link speed
    2. Slot period
    3. Maximum data length of the RMAP packet
  - ▶ Link utilization determines the efficiency of the protocol and is a function of the previous parameters.
  - ▶ Network latency and processing time of SpW–D devices determine if a set of parameters are valid
- 

# Link speed

- ▶ The higher the link speed the higher the performance of SpW-D
  - The maximum link speed determines the maximum performance of SpW-D
- ▶ So, if we want to set the parameters for maximum performance we should set them based on the maximum link speed (**200 Mbit/s**)
  - **Lower link speeds** can be accommodated **using multi-slot** scheduling (multiple consecutive slots for a single RMAP transaction)
    - One transaction in 2 consecutive slots for 100Mbit/s devices
    - One transaction in 4 consecutive slots for 50Mbit/s devices

# Slot period

- ▶ The slot period must be unique in the network
- ▶ The lower the slot period the lower the latency
  - Target: **less than 150 $\mu$ s** slot period, 5.4ms epoch
- ▶ The minimum slot period is constrained by the protocol header, the network latency and the processing time.  
(15–20 $\mu$ s @ 200Mbit/s)
- ▶ The slot period should be suitable to be use for global timing synchronization
  - ▶ For example, 1 second divided by the slot period could be a power of two:



# Why segmentation?

- ▶ Trade off of the data length field in the RMAP packet:
  - ▶ A small data length increase the protocol overhead due to the protocol header and rmap transaction network and processing delay.
  - ▶ Big data lengths increase the slot period and therefore, the latency.
- ▶ High data rate traffic characteristics:
  - ▶ Use big packets to achieve better processing efficiency
  - ▶ Requires long slot periods and increase the latency
    - High data rate packets of 4Kbyte requires  $>150\mu\text{s}$
  - ▶ It can support small slot periods if we use multi-sloting, but this increase the latency of other messages that use the same link(s).
- ▶ Segmentation is required if we want to send payload packets without increasing the latency of other data flows.
  - Maximum segment size = RMAP maximum data length

# Maximum segment size

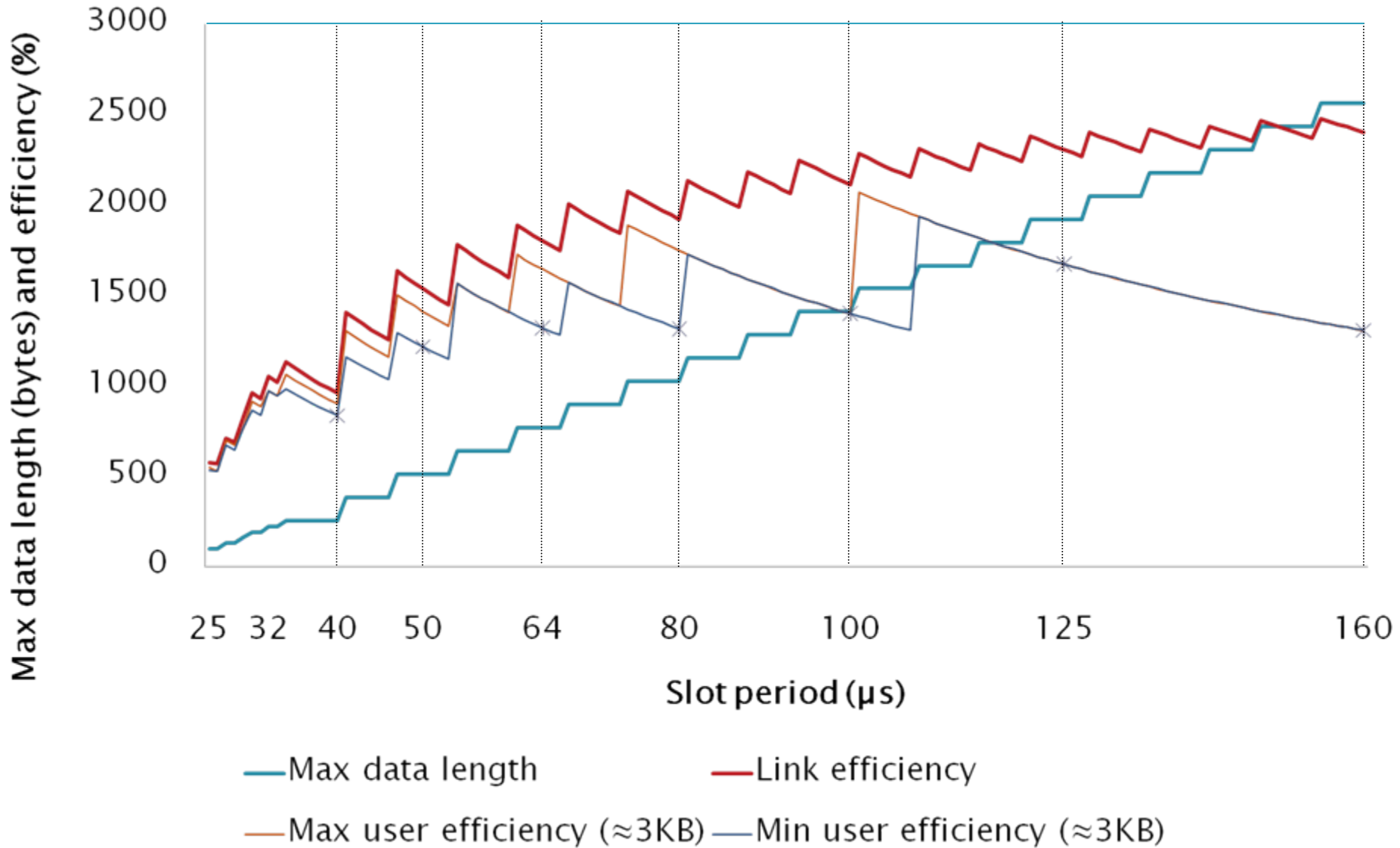
- ▶ A small segment size increase the protocol overhead
  - Due to the protocol header and rmap transaction network and processing delay.
- ▶ A bigger segment size is inefficient if the user packets are small, BUT:
  - Small packets are usually command and control messages, that are not as common as payload data segments (high data rate messages).
  - A bigger segment allows to guarantee that a **command and control is sent in a single segment**.
    - The latency of a control message is then the latency of a single segment
- ▶ If payload message size is a multiple of segment size the efficiency is maximized.

Payload message size is usually a multiple of a power of 2

Segment size should be big enough to accomodate any low latency message  
Segment size should be a multiple of 128 or 256

# Example of a trade off plot

- ▶ Methodology and assumptions
  - **Segment size** is the maximum allowed by the slot period
  - Segment size is a multiple of 128 bytes
  - **Link efficiency** is based on the ratio between SpW-D user data rate and SpW user data rate.
    - Assumes segments are always filled with user data
  - **User data rate efficiency** takes into account the fact that the segments are not always filled with user data
    - Max user data rate efficiency: use the ratio (10%) between command and control packets (100bytes) and data packets (3Kbytes).
    - Min user data rate efficiency: data packets of 3Kbytes+1



Longer slot periods increase the latency of low latency messages but do not always improve the actual data rate of high data rate messages of around 3Kbytes

# Trade off results

Slot period	Max segment size	Link efficiency	Max user data rate efficiency	Min user data rate efficiency
40	256	32%	30%	28%
50	512	51%	47%	41%
61	768	63%	57%	46%
64	768	60%	55%	44%
80	1024	64%	58%	44%
100	1408	70%	46%	46%


The best compromise latency/throughput/timing synchronization/source buffering space, is 61  $\mu$ s slot period with 768 bytes segment size.

The maximum data rate possible is 100Mbps. Higher data rates can be achieved with multi-slots (up to 130Mbit/s), i.e. 3Kbytes require only 3 slots, not 4.

# SpW-D Channel concept within a node

- ▶ A channel (or virtual channel) wraps a single RMAP message configuration with the allocated slots for this message. It also provides the sending status and error reporting.
- ▶ Multiple channels can be active at the same time. This means that they are sending segments of multiple long messages.
  - This increases the global throughput when they are using different slots.
  - If multiple channels are allocated to the same slot, the highest priority one (usually associated to command and control) will be send first.

# SpW-D Channel concept (2)

- ▶ A channel does not send the first or the next segment of a message if the host indicates that there is no data available
    - Data available status can be changed at any time to support sporadic asynchronous messages or data coming from a FIFO interface.
  - ▶ SpW-D checks the channel with the lowest number first, to see if there is data available and if it is allowed to send in the current timeslot.
    - If not selected, it will check the next channel with increasing numbering.
- 

# Network schedule

- ▶ **Define a set of channels** corresponding to each data and control message of the system
- ▶ **Determine** the **bandwidth** required for the data messages and the maximum **latency** required for the control **messages**
- ▶ The **number of slots** allocated per Epoch depends on the bandwidth required
- ▶ The **maximum separation** between slots depend on the latency required.

**Network schedule is not directly related with the scheduling of message at system level!!!**

Only matters bandwidth and latency requirements assigned to each channel used by a message



# SpW-D Channels example

Channel	type	Segments to send per epoch	slots	Destination	Data ready
Ch0	Control	M0-1	0,2,4	A	No
Ch1	Data	M1-1, M1-2	0,2,4	A	Yes
Ch2	Data	M2-1, M2-2	1,3	B	Yes

priority



Slot 0	Slot 1	Slot 2	Slot 3	Slot 4	
M1-1	M2-1	M0-1	M2-2	M1-2	



At this instant Host wants to send control message, sets data ready (Ch0) = yes

# SpW-D Channel Advantages

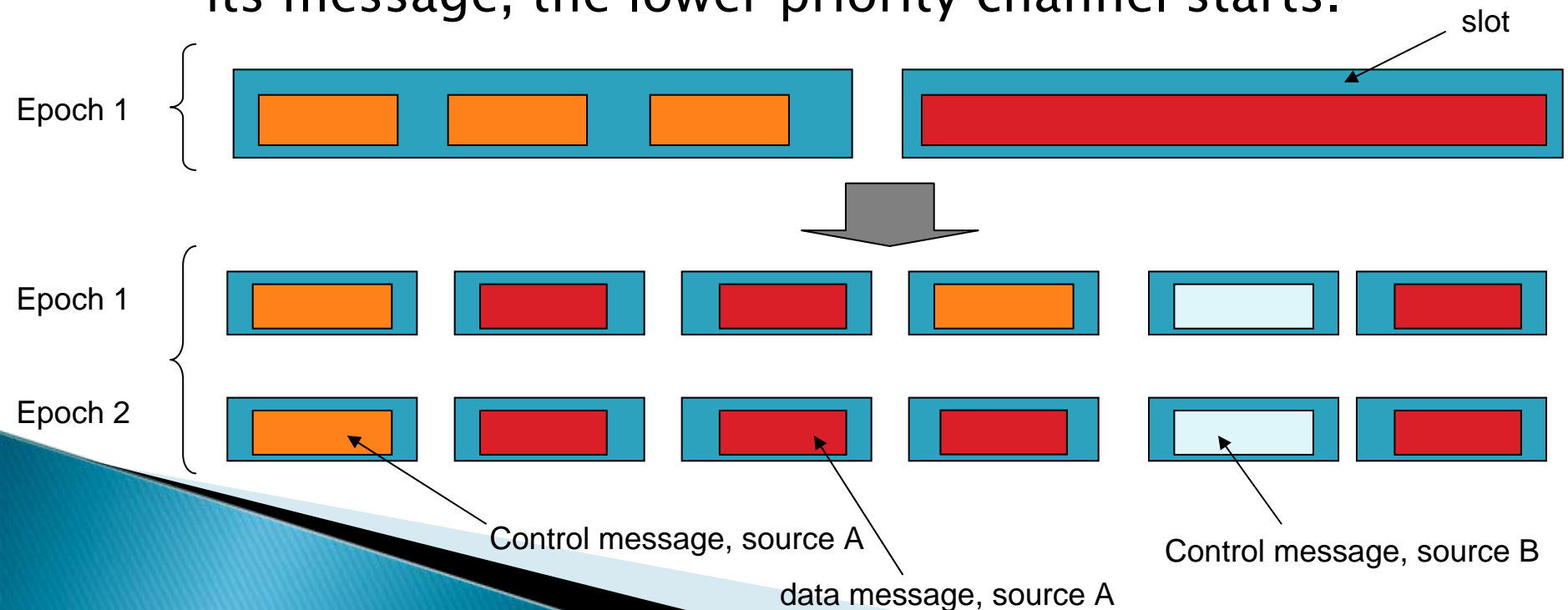
- ▶ Implements a priority scheme that removes the need to allocate slots for sporadic messages.
  - Only the throughput they require must be taken into account.
- ▶ Allows to send segments of messages going to different destinations concurrently, without having to wait for the first one to finish before sending the next message.
  - Slots for the second destination are not wasted while sending to the first destination.
- ▶ It does not add complexity to the basic SpW-D implementation if segmentation is required.

# Asynchronous traffic

- ▶ Waste of network resource in the presence of asynchronous traffic
  - Event-based traffic in **same node** than data traffic
    - Asynchronous traffic is given a channel with high priority sharing slots with data traffic
  - Event-based traffic in **different node** than data traffic
    - Temporally disable data traffic channel
      - Using a local scheduler based on local time (CPU node)
      - Remotely by the network manager (non-CPU node)

# Alternative to multi-transactions

- ▶ Instead of sending multiple commands in the same slot
  - Assign in the **same slot(s) multiple channels**
  - Once the highest priority channel finished sending its message, the lower priority channel starts.



# Duplex SpW-D

- ▶ SpW-D can use SpW full duplex capability with careful scheduling
  - **Maximum two transactions** can share the same link in the same slot
  - The data of the transaction have to be sent in **different directions** from each transaction
    - Write command going in one direction and another write command going in the reverse direction
    - Read command going in one direction and another read command going in the reverse direction
    - Write in one direction and read in the reverse direction is not possible

# Segmentation

- ▶ The targets needs to identify if a RMAP packet is a segment of a message (Big RMAP packet) and if it is the **start or the end segment**.
- ▶ Two bits required (to be implemented in the Transaction ID field of RMAP packet)
  - First/start segment flag
  - Last/end segment flag
- ▶ When single segment both bits are set
- ▶ When middle segment both bits are cleared.
  - If first segment follows a middle segment then the last message received must be considered incomplete (equivalent to EEP)

# Error detection

- ▶ **Time-Code error**: set when a Time-Code is received too early or too late (or it has been lost)
- ▶ Per each channel the following conditions are defined:
  - **TX error**: set when the RMAP command header is invalid or there is a bus error. Disables the corresponding channel.
  - **TX congestion**: set when a RMAP packet is still being send at the beginning of the next slot.
  - **RX error**: set when the RMAP reply is not received or when it has been received with an error code. Disables the corresponding channel.
  - **Late reply**: A reply has been received after the end of the slot but before the deadline set for this channel.

# Late RMAP reply allowed

- ▶ If there is an error in the network, **temporarily network congestion** can affect any channel of the network
  - The routers take time to spill the packet(s) that are using the malfunction link or device. This time could be higher than the duration of a timeslot
  - Channels that are using working links or devices should not be disabled even if they experiment contention
- ▶ It may be better to flag that there has been temporally congestion than to disable the affected channel (**avoid error propagation**)
  - An RMAP reply is considered valid if it has been received after a programmable number of slots.



# Sequence number

- ▶ A sequence number must be used for
  - **Matching a RMAP reply** with the corresponding RMAP command
  - If the increment bit of RMAP is not always set (i.e remote writing to a FIFO interface), the RMAP target needs to know if the current segment received is **out of order**.
    - If multiple channels are used, there must be an independent sequence number per channel
- ▶ The sequence number is stored in the transaction ID field of an RMAP command and it is incremented per segment sent.

# Transaction ID

- ▶ One byte of the transaction ID field is reserved to be used by SpW-D
  - The other byte can be used by the user and it is application dependent.
    - RMAP standard states that the transaction ID field is optionally provided to the destination Host.

- ▶ Formatting of SpW-D transaction ID byte

- **Option 1:** SpW-D does NOT support the non increment address option of the RMAP standard

Start Seg (1 bit)	End Seg (1 bit)	Sequence number (6bits)
----------------------	--------------------	-------------------------

- **Option 2:** SpW-D supports the non increment address option of the RMAP standard

Start Seg (1 bit)	End Seg (1 bit)	Channel number (5bits)	Sequence number (1 bit)
----------------------	--------------------	---------------------------	----------------------------

Send and wait scheme requires a single bit

# Error recovery

- ▶ Retry mechanism
  - Retry must be done in the **next slot allocated to the same channel**.
    - Retry is not performed unless it is indicated by the host or the network manager by clearing the error condition.
- ▶ Automatic enabling a channel when previous channel number got an error.
  - Allows to set a channel that will be used to send a **notification message** to the network manager if another channel fails.
  - Allows to set an **automatic retry** using another path or to another destination.

# Demonstration

