Protocol Validation System
for On-Board
Communications

Analysis & Assessment of the SpW-D Draft specification B

15th SpaceWire Working Group, 18th October 2010
Noordwijk, Netherlands

esa

teletel

Protocol Validation System (PVS) for on-board communications – CCN3

- Analysis and evaluation of SpW-D draft specification
- Functional requirements for RMAP and SpW-D validation
- Implementation and validation of a RMAP IP Core
- SpW-D protocol implementation & validation

Start of Study: July 2010

End of Study: January 2011

esa

teletel

PRESENTATION AGENDA:

SpW-D Simulation & results

SpW-D scheduling analysis & proposals

Proposals for Segmentation & Retry mechanisms

Other issues

SpW-D Simulation tool (1/2)

- SpW-D Simulations in MATLAB
  - Simulation parameters configuration: SpW link speed, Time-slot interval, SpW router forwarding delay (Tb), initiator command start delay (Ta), RMAP authorization delay (Td), target memory arbitration latency, target memory throughput, RMAP reply delay (Tg), Time-slot safe-margin
  - Topology configuration: Paths from Initiators, Targets, number of SpW Routers
  - Scheduling Tables configuration
  - Traffic sources configuration (per time-slot)
- Computation of latency, data rate, efficiency, utilisation, per time-slot, per initiator, per epoch and total
- Simulation of SpW flow control not supported
- Link data rate assumed equal to 8/10 SpW link speed

Assumptions:

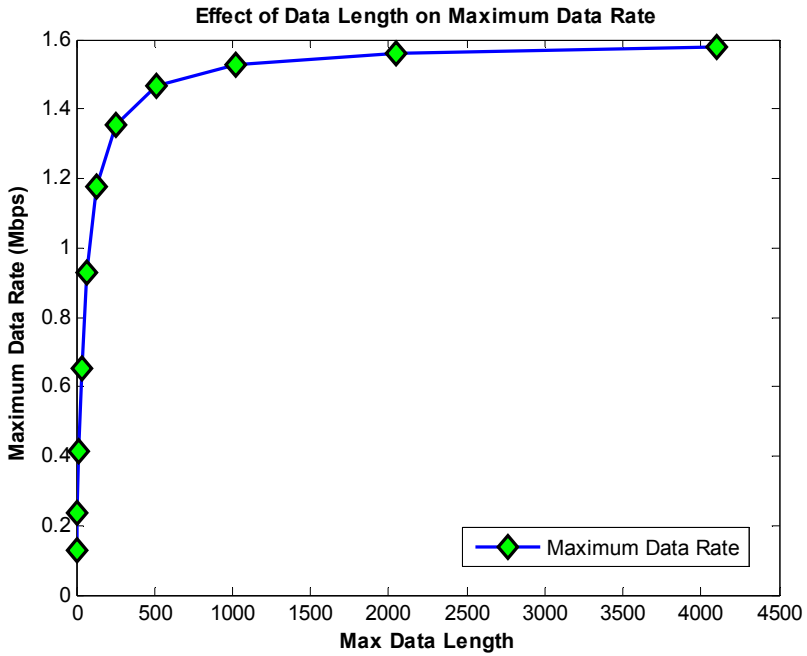
- Initiator and Target delays set to 5µs (as in SpW-D draft B)
- Two differences from SpW-D draft B calculations:
  1. Memory delay is the sum of Arbitration delay (constant 3µs) and memory transfer delay = Size/Throughput (configurable, default value 1Gbps)
     - *As a result the memory delay of 256bytes payload is 3+2.048 ~ 5µs (as in SpW-Draft B)*
  2. A safety margin of 2µs is added at the end of all time-slot durations

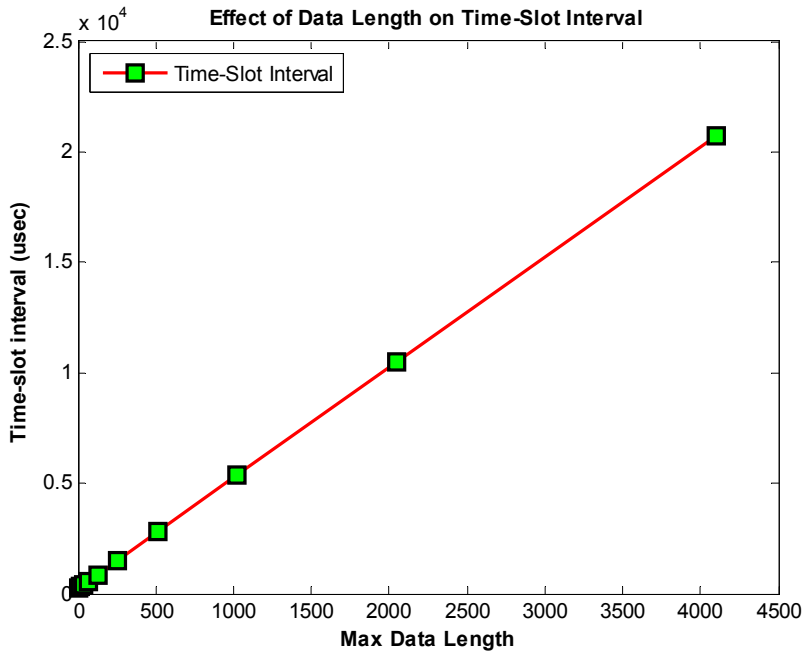
Configuration:

- The presented results are for a fully utilised link with Simple Schedule or Concurrent Schedule with no conflicts
- Time-slot duration is calculated to support up to 4 SpW routers
- Traffic sources: constant bit rate, maximum payload length in all time-slots, 32 read and read write transactions per epoch

=> Data Rate is the maximum

Minimum Time-slot interval and maximum data-rate vs packet length for 2Mbps SpW link speed

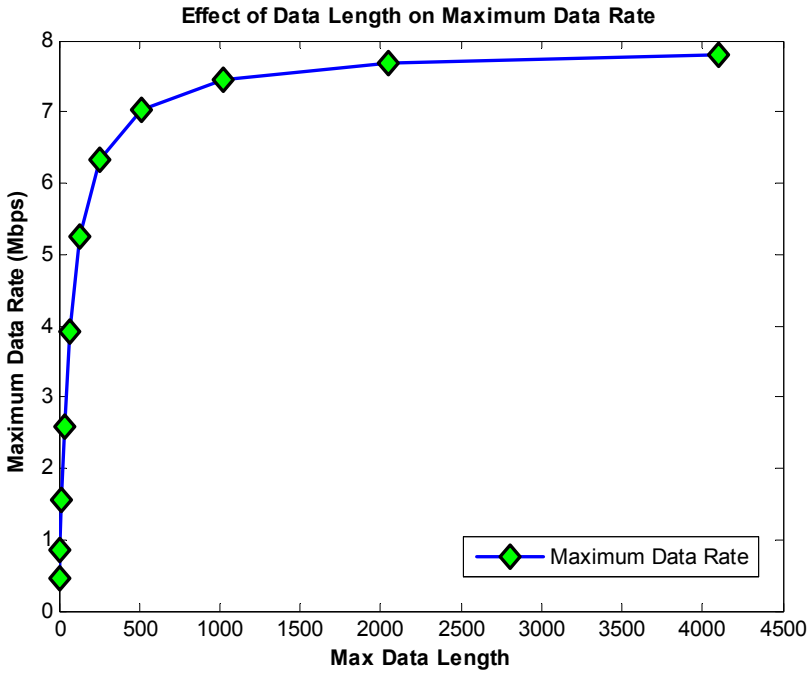


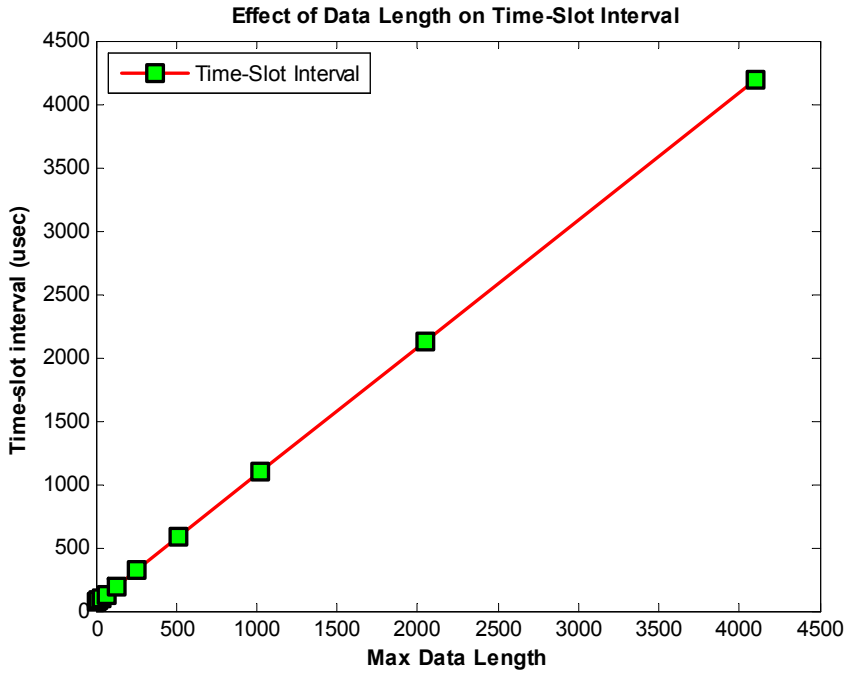


Packet Length (Bytes)	Time-slot (µs)	Data Rate (Mbps)
256	1512	1.35
4096	**20743**	**1.58**

Good efficiency
Data Rate not affected by HW delays

Minimum Time-slot interval and maximum data-rate vs packet length for 10Mbps SpW link speed

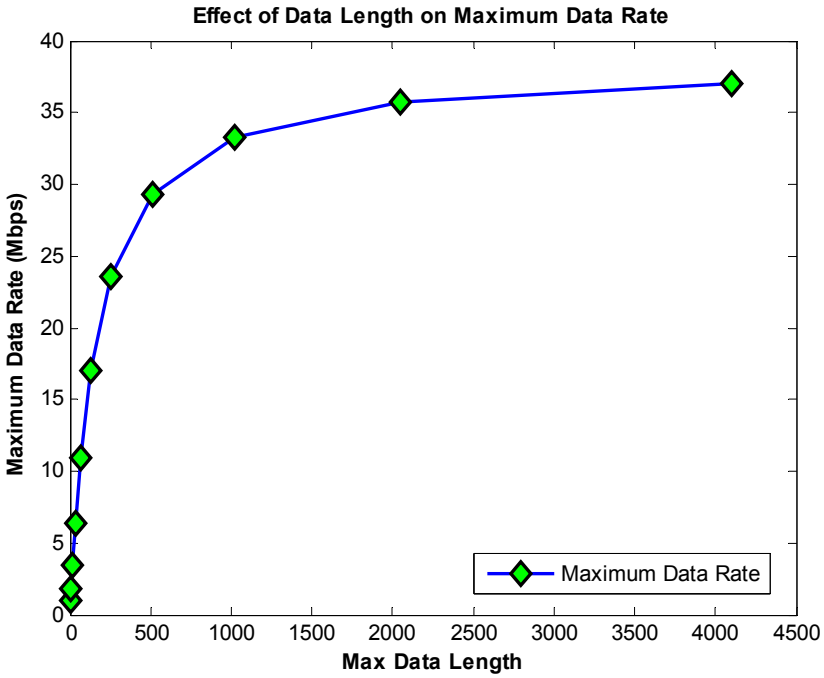


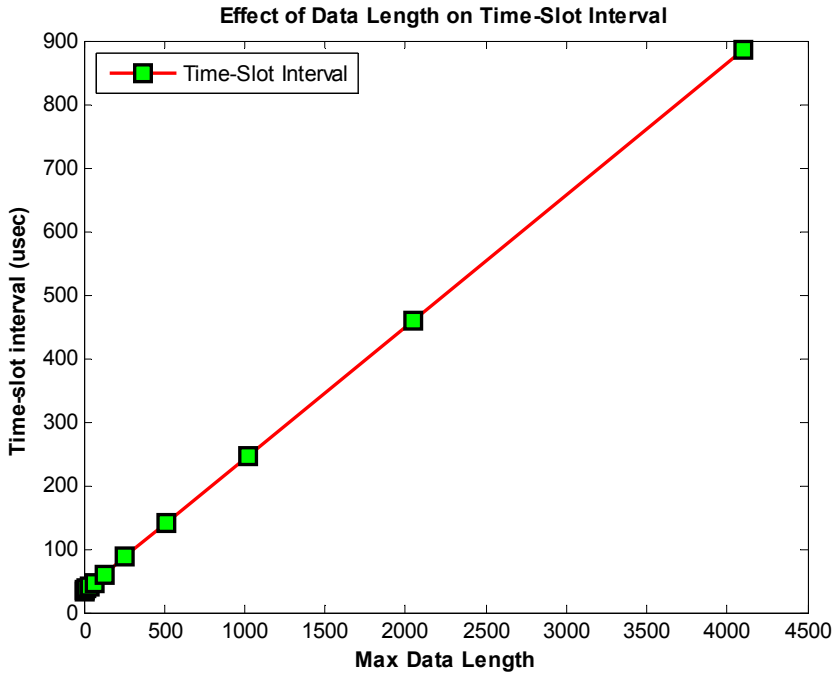


Packet Length (Bytes)	Time-slot (µs)	Data Rate (Mbps)
256	324	6.32
4096	**4195**	**7.81**

Good efficiency
Data Rate not affected by HW delays

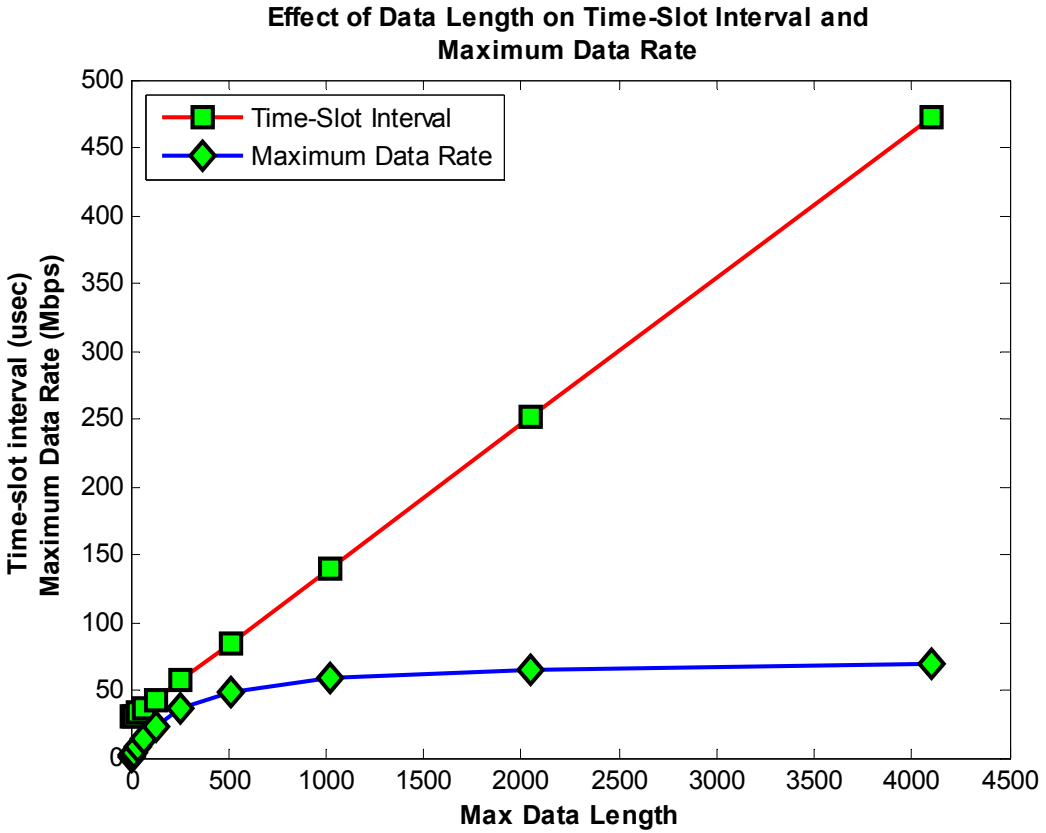
Minimum Time-slot interval and maximum data-rate vs packet length for 50Mbps SpW link speed





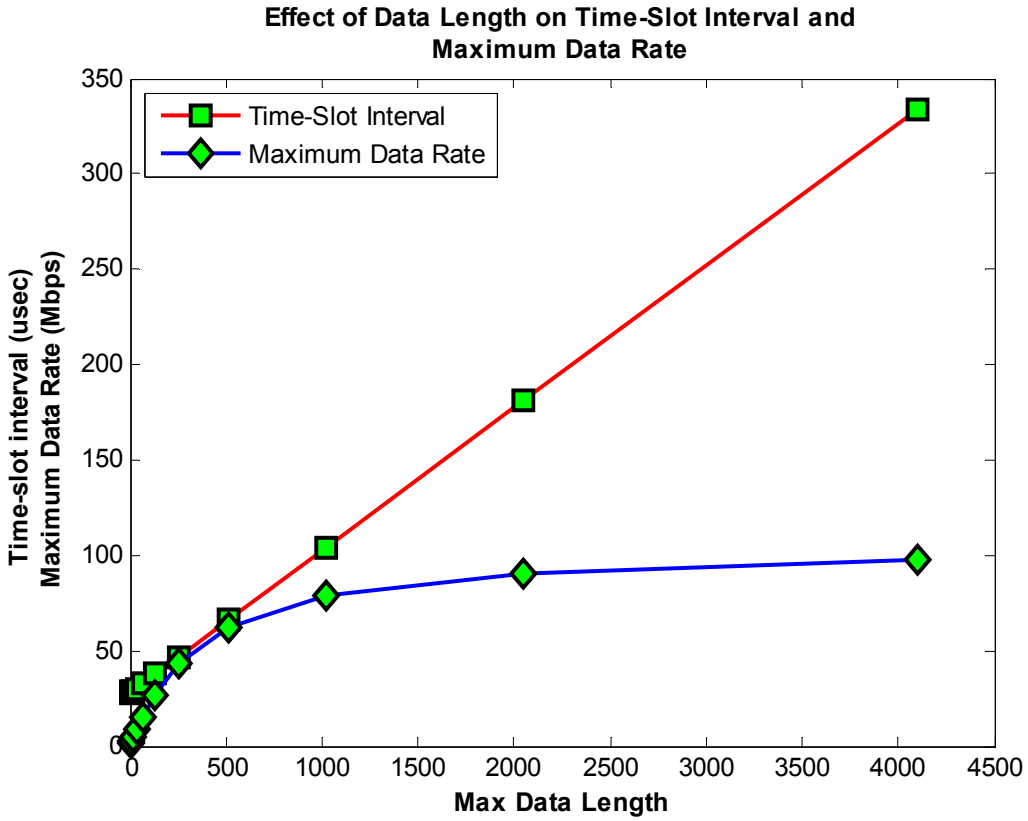
Packet Length (Bytes)	Time-slot (µs)	Data Rate (Mbps)
256	87	23.54
4096	885	37.03

Minimum Time-slot interval and maximum data-rate vs packet length for 100Mbps SpW link speed



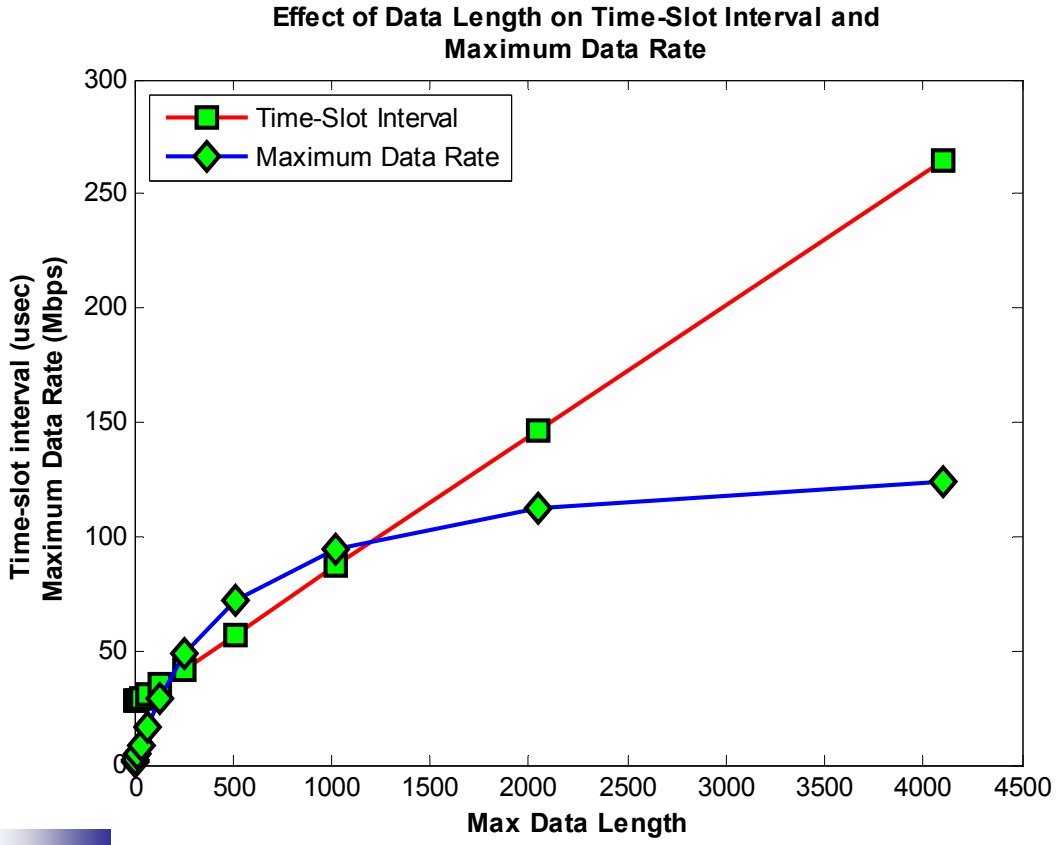
Packet Length (Bytes)	Time-slot (µs)	Data Rate (Mbps)
256	57	35.93
4096	472	69.42

Minimum Time-slot interval and maximum data-rate vs packet length for 150Mbps SpW link speed



Packet Length (Bytes)	Time-slot (µs)	Data Rate (Mbps)
256	47	43.57
4096	334	98.11

Minimum Time-slot interval and maximum data-rate vs packet length for 200Mbps SpW link speed

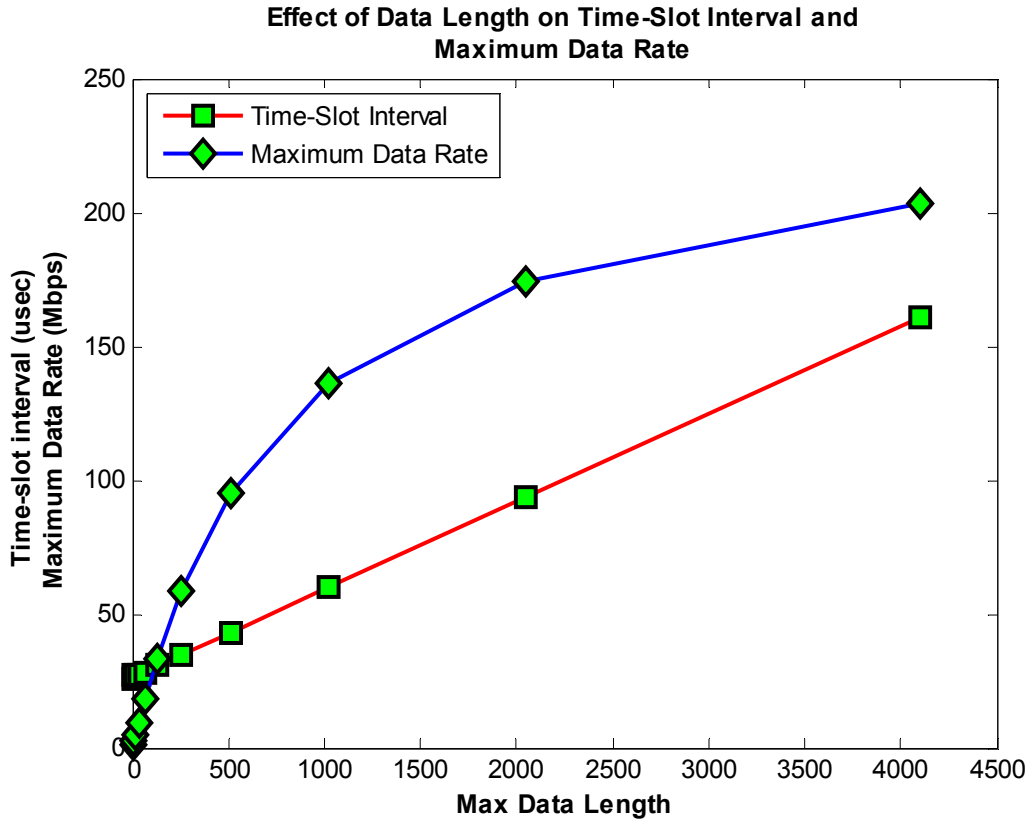


In SpW-D draft B:
for 132 Bytes average payload length, data rate is 30Mbps

Results are similar

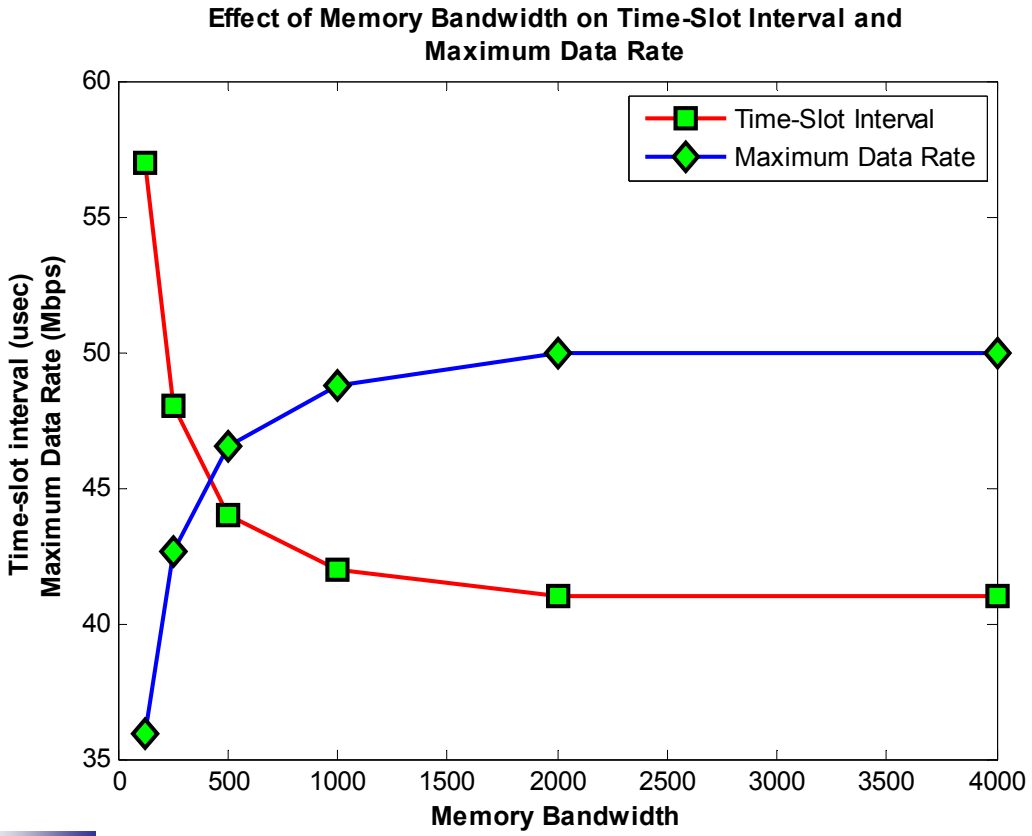
Packet Length (Bytes)	Time-slot (µs)	Data Rate (Mbps)
128	35	29.26
256	42	48.76
4096	265	123.65

Minimum Time-slot interval and maximum data-rate vs packet length for 400Mbps SpW link speed



Packet Length (Bytes)0	Time-slot (µs)	Data Rate (Mbps)
256	35	58.51
4096	161	203.53

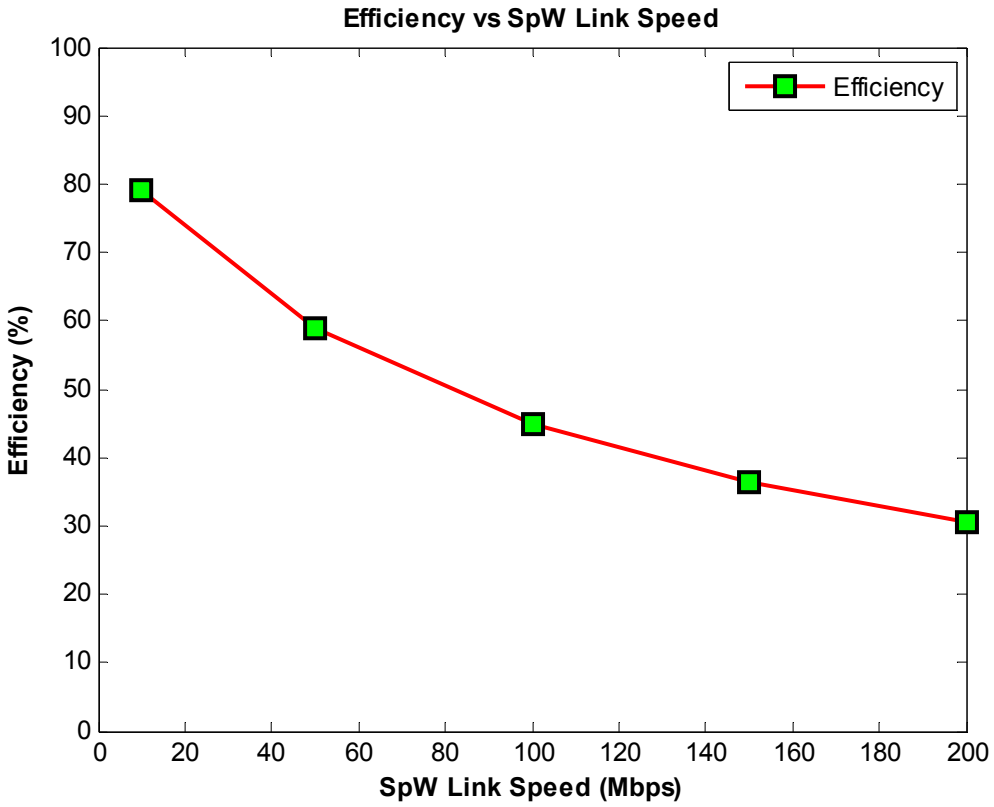
Minimum Time-slot interval and maximum data-rate vs RMAP target Memory Throughput for 200Mbps SpW link speed and 256 Bytes maximum data length



Data rate is decreased by slow memories at high SpW link speeds

Memory BW (Mbps)	Time-slot (µs)	Data Rate (Mbps)
125	57	35.93
250	48	42.67
1000	42	48.76

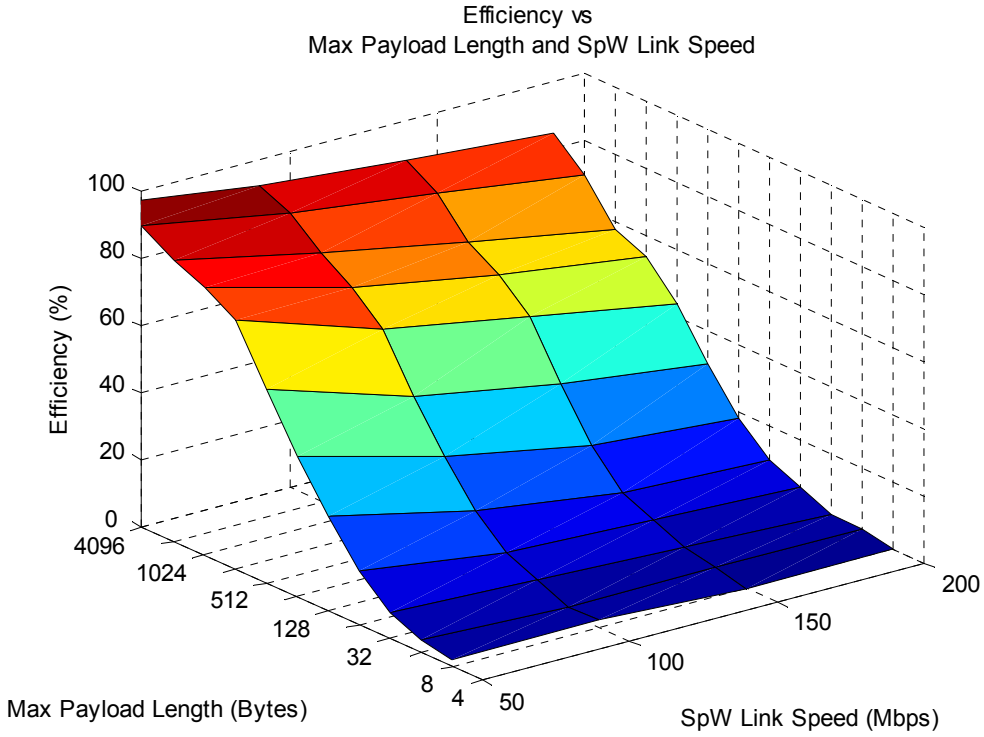
Efficiency vs SpW link speed for 256 Bytes maximum data length

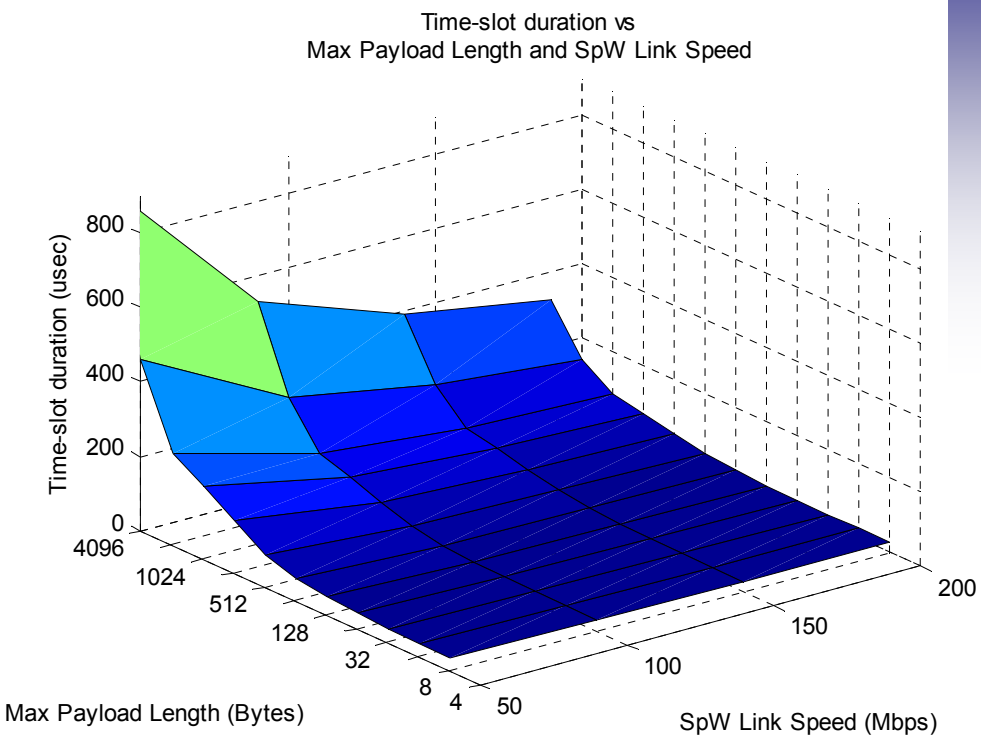


SpW link speed (Mbps)	Efficiency (%)
10	79.00
50	58.86
100	44.92
150	36.30
200	**30.48**

At 200Mbps SpW link speed,
Data Rate is 48.76Mbps => Efficiency is 30%!
(compared to single-direction SpW link speed, 160Mbps)

Efficiency and Time-slot duration for different Max Payloads and SpW Link Speeds





Efficiency increases as Max Payload length increases

Efficiency decreases as SpW link speed increases

Time-slot increases as Max Payload length increases

Time-slot increases as SpW link speed decreases

Efficiency and Time-slot duration

Max Payload (Bytes)	10Mbps SpW link		50Mbps SpW link		100Mbps SpW link		150Mbps SpW link		200Mbps SpW link	
	Time-slot (µs)	Efficiency (%)	Time-slot (µs)	Efficiency (%)	Time-slot (µs)	Efficiency (%)	Time-slot (µs)	Efficiency (%)	Time-slot (µs)	Efficiency (%)
4	70	6	34	2	30	1	28	1	28	1
8	74	11	35	5	30	3	29	2	28	1
16	82	20	37	9	31	5	29	4	28	3
32	99	32	40	16	33	10	30	7	29	6
64	131	49	47	27	36	18	33	13	31	10
128	195	66	60	43	43	30	38	22	35	18
256	324	79	87	59	57	45	47	36	42	30
512	582	88	140	73	85	60	66	52	57	45
768	840	91	193	80	112	69	85	60	72	53
1024	1098	93	246	83	140	73	104	66	87	59
2048	2131	96	459	89	251	82	181	75	146	70
4096	4195	98	885	93	472	87	334	82	265	77

Time-slot < 100µs

Time-slot < 100µs and Efficiency > 60%

Potential candidate values for max payload length: 256, 512, 768, 1024 Bytes

At low SpW link speeds Time-slot duration is greatly increased

Evaluation with different traffic profiles

Efficiency (%) for 200Mbps SpW link					
Time-slot duration set for:	Traffic Profile 32 Bytes	Traffic Profile 256 Bytes	Traffic Profile 1024 Bytes	Traffic Profile 4096 Bytes	AVERAGE
Max. Payload 256 Bytes	4	30	78	69	45
Max. Payload 512 Bytes	3	22	57	67	37
Max. Payload 768 Bytes	2	18	45	71	34
Max. Payload 1024 Bytes	2	15	59	77	38

Efficiency (%) for 150Mbps SpW link					
Time-slot duration set for:	Traffic Profile 32 Bytes	Traffic Profile 256 Bytes	Traffic Profile 1024 Bytes	Traffic Profile 4096 Bytes	AVERAGE
Max. Payload 256 Bytes	5	36	48	82	43
Max. Payload 512 Bytes	3	26	66	78	43
Max. Payload 768 Bytes	3	20	51	80	39
Max. Payload 1024 Bytes	2	16	66	66	37

Efficiency (%) for 100Mbps SpW link					
Time-slot duration set for:	Traffic Profile 32 Bytes	Traffic Profile 256 Bytes	Traffic Profile 1024 Bytes	Traffic Profile 4096 Bytes	AVERAGE
Max. Payload 256 Bytes	6	45	59	79	47
Max. Payload 512 Bytes	4	30	77	75	46
Max. Payload 768 Bytes	3	23	58	80	41
Max. Payload 1024 Bytes	2	18	73	73	42

Efficiency (%) for 50Mbps SpW link					
Time-slot duration set for:	Traffic Profile 32 Bytes	Traffic Profile 256 Bytes	Traffic Profile 1024 Bytes	Traffic Profile 4096 Bytes	AVERAGE
Max. Payload 256 Bytes	7	59	77	88	58
Max. Payload 512 Bytes	5	37	93	82	54
Max. Payload 768 Bytes	3	27	68	93	48
Max. Payload 1024 Bytes	3	21	83	83	47

Multi-slot scheduling is used
Segmentation is not used

4 Traffic profiles provided with 32, 256, 1024 and 4096 bytes payload

Multi-slot scheduling used when required (yellow cells) Segment is NOT used for large packets

Efficiency (%) estimated for different Time-slot intervals (based on Max Payloads of 256, 512, 768, 1024) and different SpW link speeds (50, 100, 150, 200 Mbps)

Max payload length of 256 bytes has the best average efficiency for the provided traffic profiles

Conclusions on Simulation Results

1. There is no “one-size-fits-all” solution for the Max Payload length or Time-slot duration
2. Possible values for Max Payload can be 256, 512, 768 or 1024 bytes which seem to present good results for a variety of network traffic profiles
3. SpW-D is characterized by low efficiency compared to SpW link speed due to the following:
   - SpW-D scheduling scheme
   - Impact of SpW-D protocol functional model and HW delays on efficiency in high SpW link speeds

PRESENTATION AGENDA:

SpW-D Simulation & results

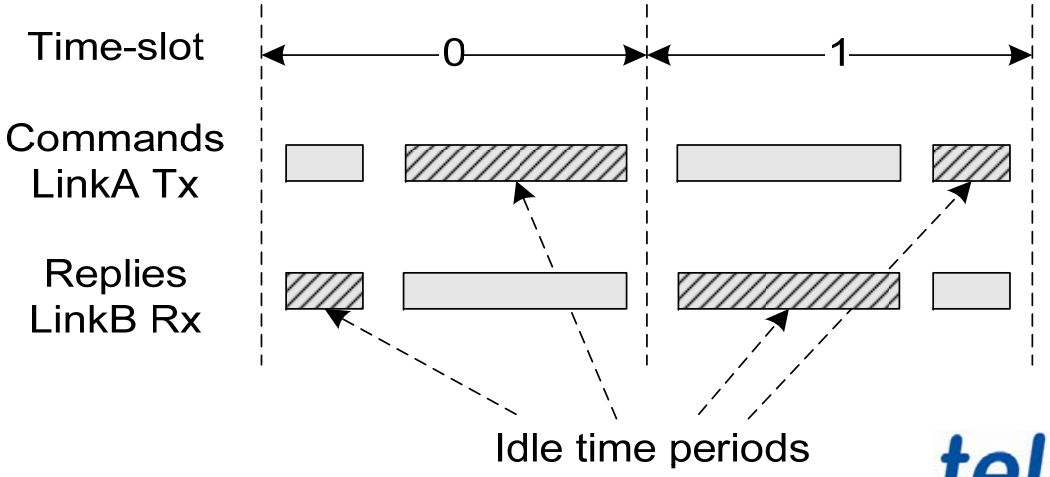
SpW-D scheduling analysis & proposals

Proposals for Segmentation & Retry mechanisms

Other issues

Bandwidth utilization

- Waste of network resources in the presence of asynchronous traffic
  - caused when initiators have no traffic to transmit
- Fixed size of the maximum payload length and consequently the time-slot duration must be carefully selected based on communication requirements
  - If too small it will cause overhead (due to RMAP headers) & reduce efficiency
  - If too large network bandwidth is wasted by initiators that transmit/receive small payloads
- SpaceWire-D does not utilize the full duplex SpW link and path capability
  - Idle time-periods reduce data rate

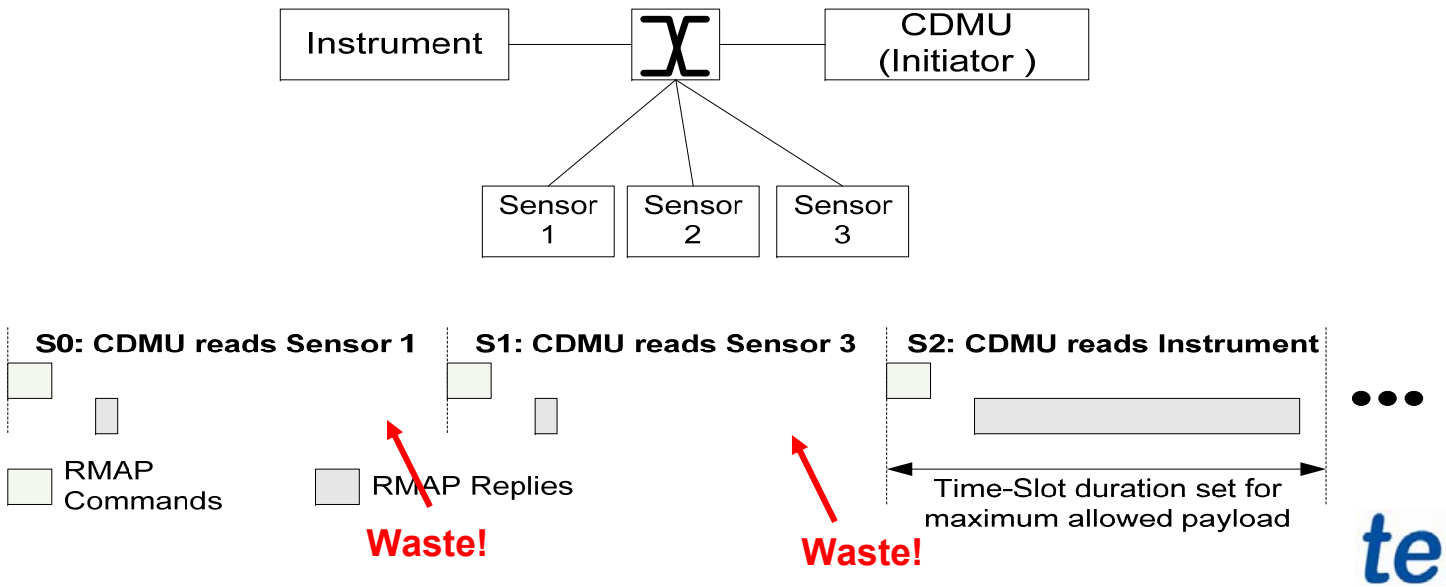


Large data transfers in DH

- Large data transfers require a large number of time-slots to complete
- For example, 1 Mbyte data transfer with a maximum PDU payload length of 256 bytes, requires 4096 time-slots or 64 fully utilized epochs
- Pre-allocating a large number of time-slots may waste network resources when transfers are not periodic
- Pre-allocating a small number of time-slots increases the time required for the completion of the large data transfer

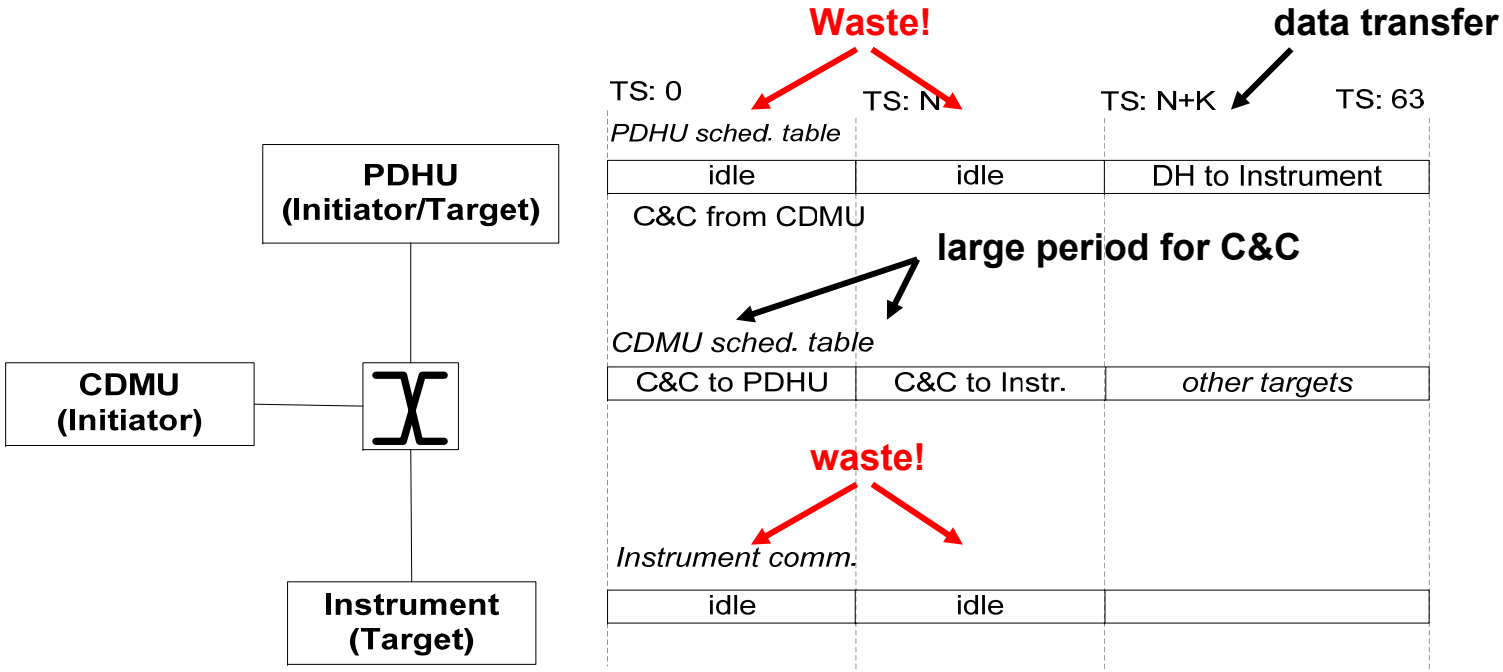
Scheduling C&C

- If control loops need to perform reads and writes from/to different sensors actuators with small latency between transactions, this requires allocation of consecutive time-slots
  - Consumes a large number of time-slots in the Schedule table
- Large number of sensors and actuators or large payloads result in a large number of time-slots for the completion of the transactions required for control loop
  - Reduces the number of time-slots for DH even more
- If the payload of C&C RMAP transactions is relatively small compared to the maximum RMAP payload length then there will be an idle period during time-slot duration
  - Bandwidth is wasted



Scheduling C&C and DH in a common network

- Control loops in command and control communications (e.g. AOCS) need to be performed at large periods (e.g. >20 ms) compared to epoch interval (e.g. ~2.5msec, Time-slot ~40µs for 256B at 200Mbps SpW link)
- May result in waste of bandwidth when scheduling in a common network for C&C and DH, and there are common paths in C&C and DH communications

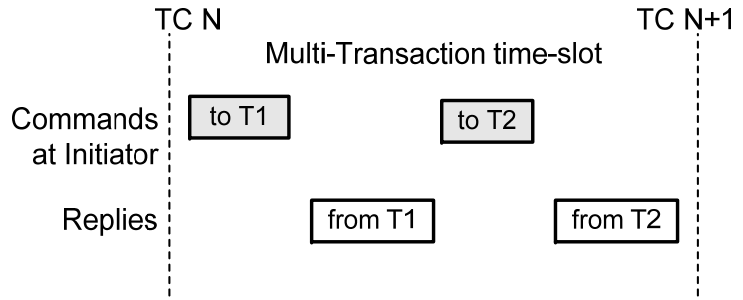


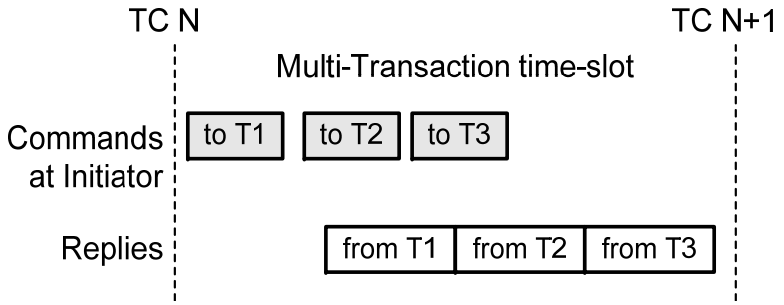
SpW-D Scheduling(A): Multi-Transaction scheduling (1/2)

- **More than one transaction <u>in the same</u> time-slot**

Two alternative implementations:

- Initiator issues next command after receiving the previous reply
  - easy to implement
  - Allows transactions to the same target
  - No conflicts occur in the network but does not increase the throughput significantly since network operates in half-duplex mode
- Initiator may issue a next command before receiving the previous reply
  - More transactions can fit in one time-slot
  - Increases the overall throughput, since the network can operate in full-duplex
  - More difficult to implement
  - Targets must be different





SpW-D Scheduling(A): Multi-Transaction scheduling (2/2)

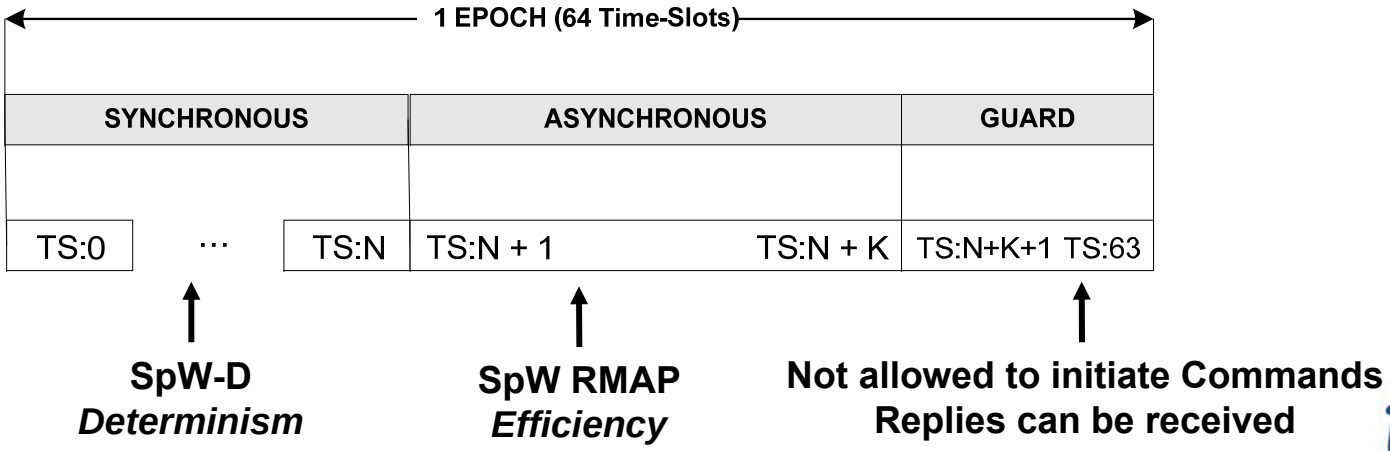
Conclusions:

- Throughput increases in the presence of small-payload transactions
- Latency decreased for small-payload transactions
- Saves time-codes used for small payload transactions
- Slight modifications to the SpW-D Scheduler required
- Memory resources required for the implementation of the SpW-D scheduler increase if different number of transactions shall be supported per multi-transaction time-slot

SpW-D Scheduling(B):Use of dedicated time-slots for asynchronous traffic (1/3)

Epoch is divided to different segments:

- Synchronous: The network operates as specified in the SpW-D draft B specification
- Asynchronous: The network operates as a SpW network with RMAP packets of limited length
  - Time-Codes are transmitted
  - The initiators use the schedule table to determine to which targets they are allowed to send commands,
  - Conflicts are allowed
  - Time-slot boundaries can be violated
- Guard: Segment in which initiation of transactions is not allowed. Required only in order to prevent next Synchronous Segment violation, by commands that have already been initiated in the Asynchronous Segment



SpW-D Scheduling(B):Use of dedicated time-slots for asynchronous traffic (2/3)

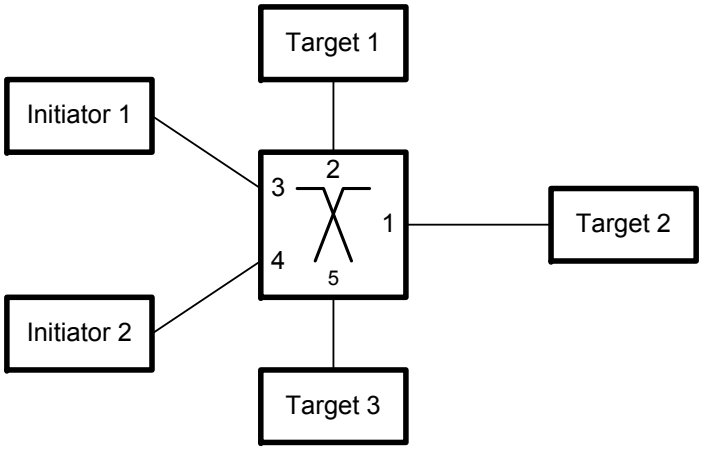
if N initiators may traverse a common path during the last Asynchronous Segment time-slot,

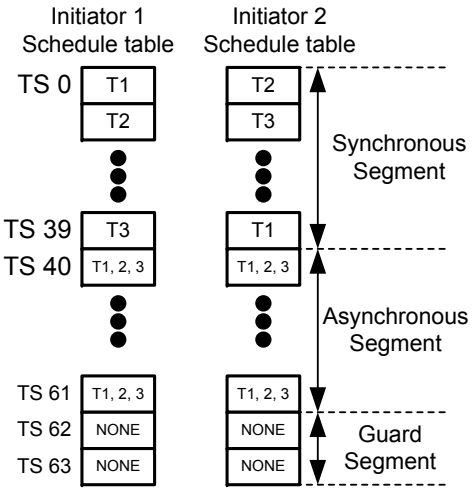
=> the Guard Segment shall be set to N time-slots

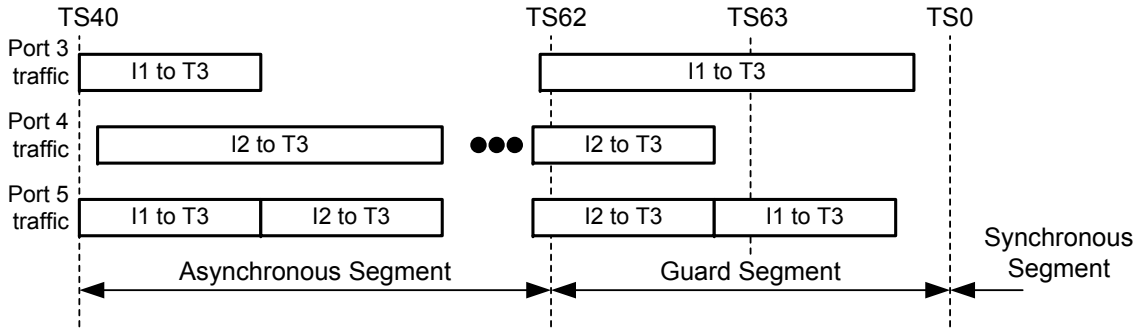
Asynchronous and Guard Segments are network configurable

If Asynchronous = 64 & Guard = 0, the network operates in SpW-RMAP mode

If Asynchronous = 0 & Guard = 0, the network operates in SpW-D mode







SpW-D Scheduling(B): Use of dedicated time-slots for asynchronous traffic (3/3)

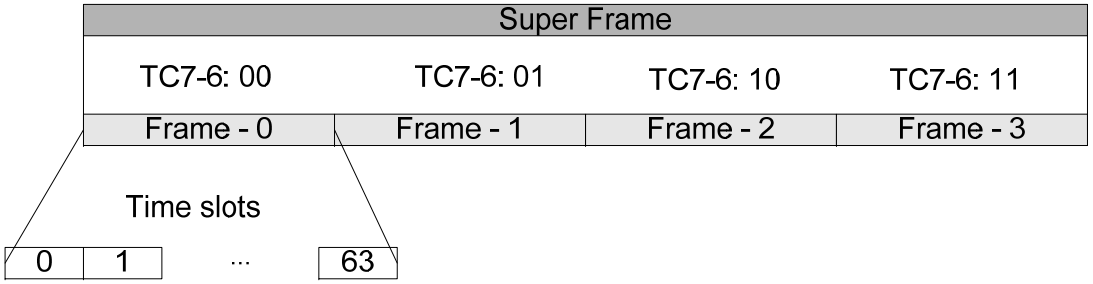
Conclusions:

- Waste of bandwidth is reduced in the presence of high asynchronous traffic
- Network behaves as a hybrid of SpW-D and standard SpW RMAP network (in different time-periods) combining benefits from both, determinism from SpW-D and improved performance and efficiency from standard SpW - RMAP
- Slight modifications to the SpW-D Scheduler required
- Guard segment may reduce network efficiency since some time-slot(s) are wasted
- If the guard and asynchronous segments use a large number of time-slots then the number of available time-slots for synchronous traffic is reduced

SpW-D Scheduling(C):Use of different schedule tables in different epochs (1/3)

- Utilization of unused time-codes
- Each initiator maintains more than one Schedule tables
- SpW-D Scheduler based on TC control bits schedules transactions from the respective table based on 8-bit time-code value
- Same as Super-Frame (composed of Frames) and Frame (composed by Time-Slots) in TDMA networks
- Different usage alternatives are possible, scheduling becomes very flexible

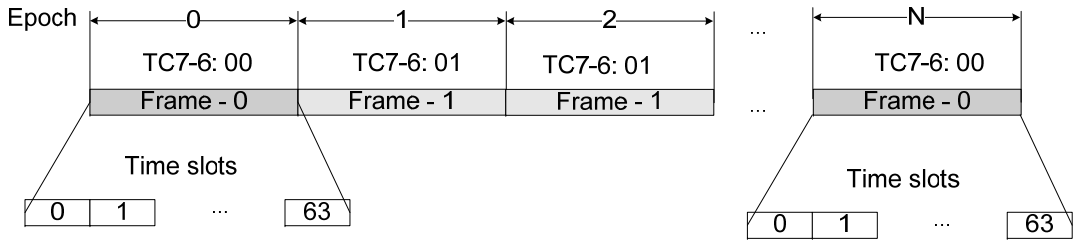
SpW-D Scheduling(C):Use of different schedule tables in different epochs (2/3)



Usage 1:

Frames repeated => 256 Time-Slots

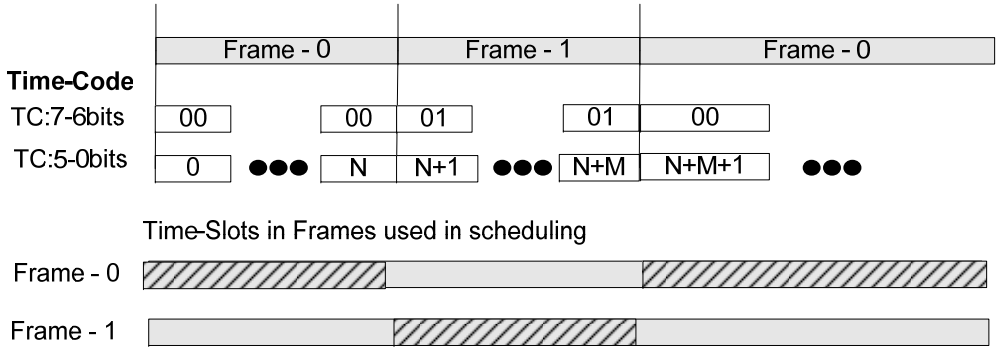
Non-compatible with possible SpW evolutions, mitigates /does not solve the problem of time-slots availability



Usage 2:

Frame 0 repeated every N epochs

e.g. for scheduling C&C communications



Usage 3:

[T5-T0] always increases and rolls over from 63 to 0

[T7-T6] pattern periodically changes

Schedule table has different entries for different frames

Open Point: Is this correctly propagated by existing SpW Routers?

SpW-D Scheduling(C): Use of different schedule tables in different epochs (3/3)

Conclusions:

- Better scheduling capability may improve network efficiency
- Increased number of available time-slots providing flexibility in scheduling
- Possibility to schedule isochronous RMAP transactions with a higher periodicity than the epoch interval
- Increases the SpW-D Scheduler required memory resources
- Time-master modification is required
- Open issue regarding the compatibility of such solution with existing routers
- Reserves available broadcast SpW characters (e.g. may be required by SpW evolutions)
- Jitter may be imposed in data handling applications due to the repetition of the schedule table for command and control communications

Applicability of scheduling schemes

	Simple Schedule	Concurent Schedule	Multi-Slot Schedule	Multi-Transaction Schedule	Concurrent scheduling with different scheduling tables in different epochs	Use of dedicated time-slots for asynchronous traffic
Simple topology or small number of initiators						
CC or DH with small payloads	x					
DH with large payloads	x		x			
DH with high asychronous traffic	x		x			
Common network for CC and DH, DH with small payload	x					
Common network for CC and DH, DH with large payload	x		x	x		
Common network for CC and DH, DH with high asychronous traffic	x		x	x		
More complex topologies or large number of initiators						
CC or DH with small payloads		x				
DH with large payloads		x	x			
DH with high asychronous traffic		x	x			x
Common network for CC and DH, DH with small payload		x				
Common network for CC and DH, DH with large payload		x	x	x		
Common network for CC and DH, DH with high asychronous traffic		x	x	x		x
Large number of nodes with many transactions between nodes						
CC or DH with small payloads					x	
DH with large payloads			x		x	
DH with high asychronous traffic			x		x	x
Common network for CC and DH, DH with small payload					x	
Common network for CC and DH, DH with large payload			x	x	x	
Common network for CC and DH, DH with high asychronous traffic			x	x	x	x

PRESENTATION AGENDA:

SpW-D Simulation & results

SpW-D scheduling analysis & proposals

Proposals for Segmentation & Retry mechanisms

Other issues

Open issues:

- How can the target distinguish between RMAP payload containing a SDU segment and a standard RMAP payload?
- How can the target know the exact size of the SDU in order to perform reassembly?
- How can the target identify that the payload of the RMAP write transaction is the start, middle or end segment of a SDU?
- How shall the target react is case of out of order delivery of a PDU? (assuming that such case can be caused by the Retry mechanism)

Segmentation is not yet specified in SpaceWire-D draft B spec.

SpW-D Segmentation & Reassembly

- Different PID is required for RMAP and segmented SpW-D in order for a target to support both protocols

ALTERNATIVES:

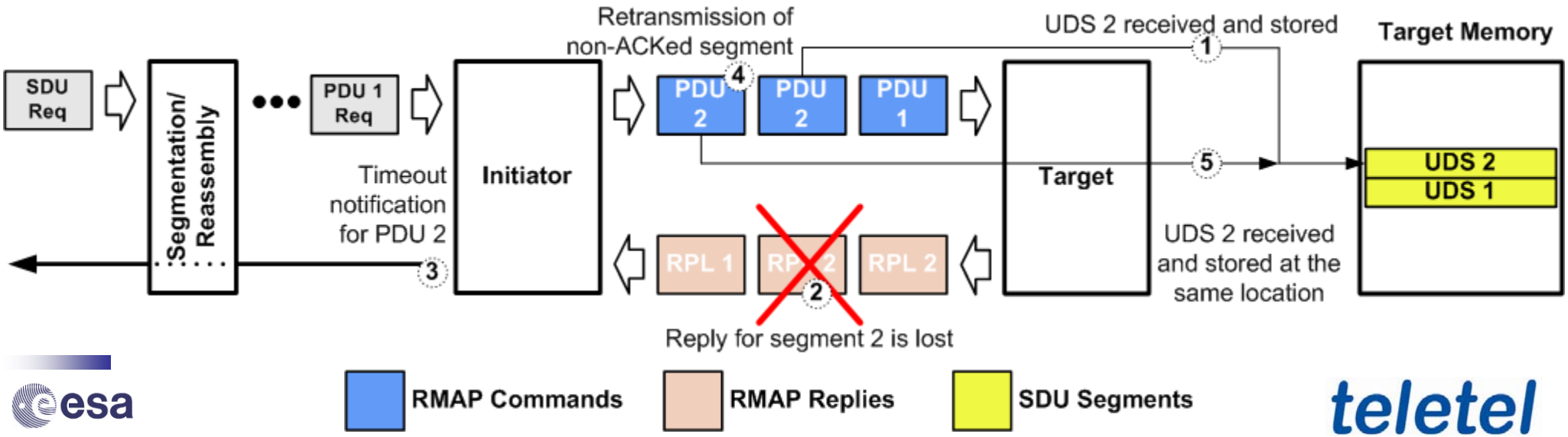
- Segmentation & Reassembly at the initiator side only:
  - Large SDU transfer request at the initiator
  - Segmentation chops the SDU request and issues multiple requests to the RMAP initiator
  - The target receives the commands, executes them and returns the replies
  - If a reply is missing it is the responsibility of the initiator to take any further actions (e.g. retry)
- Target performs reassembly:
  - The initiator operates as in the first case
  - The target receives segmentation information (e.g. Sequence Number) and can determine when a PDU is received out of order
  - The target rejects an out of order segment and does not transmit reply
- Target performs segmentation:
  - The initiator receives a request for to read a large SDU from the target
  - It passes a single RMAP read command to the target for the entire SDU
  - The target returns multiple reply packets containing segments of the SDU

Non-compatible with RMAP functional model; multiple replies for one command

SpW-D Segmentation & Reassembly(A): Implemented at Initiator only (1/2)

Incrementing Address RMAP Write transaction

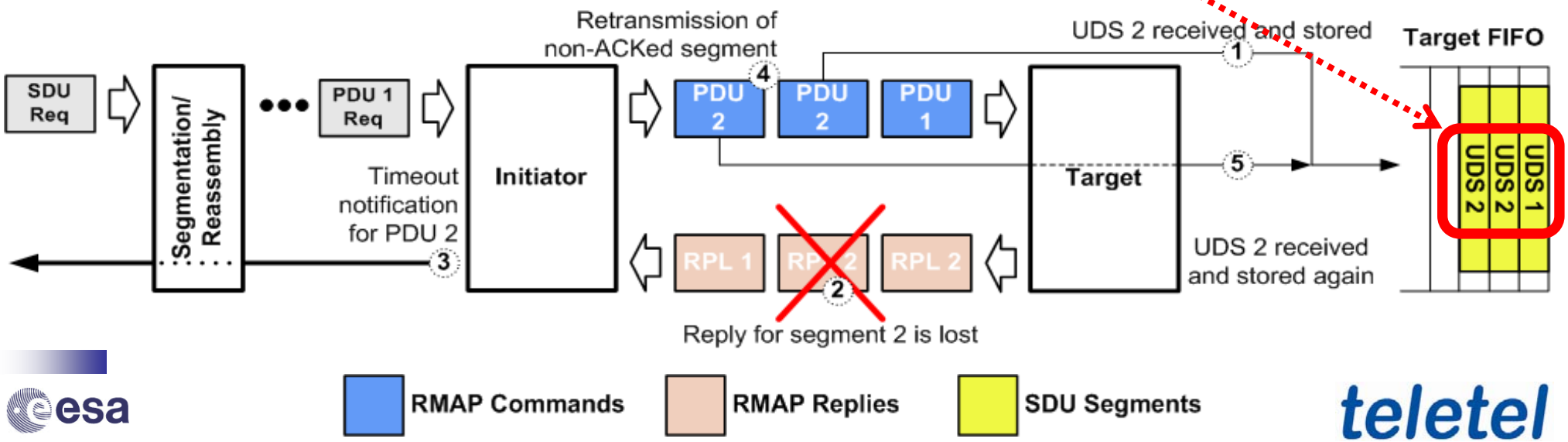
- Segmentation splits the host request to multiple RMAP write requests and:
  - Successive segments have successive Sequence Numbers (SQ)
  - Each segment is marked as start/middle/end/unsegmented
  - Successive segments contain successive EXTENDED ADDRESS and ADDRESS fields
- The initiator can detect non-acknowledged segments by the SQ in the RMAP reply
- If a reply is missing the initiator retries using the same EXTENDED ADDRESS and ADDRESS fields
- SDU is correctly reassembled for both RMAP write and read



SpW-D Segmentation & Reassembly(A): Implemented at Initiator only (2/2)

Non-Incrementing Address RMAP Write transaction

- Segmentation splits the host request to multiple RMAP write requests and:
  - Successive segments have successive Sequence Numbers (SQ)
  - Each segment is marked as start/middle/end/unsegmented
  - Successive segments contain identical EXTENDED ADDRESS and ADDRESS fields
- The initiator can detect non-acknowledged segments by the SQ in the RMAP reply
- If a reply is missing the initiator retries the non-ACKed PDU
- SDU is corrupted in case of retry for both RMAP write and read



SpW-D Segmentation & Reassembly(B): Reassembly at both the Initiator and the target

Incrementing Address RMAP Write transaction

- Initiator operates in the same way as in the previous approach:
- If a reply is missing the initiator performs retry
- The target, upon the reception of a command it checks the SQ and accepts or rejects the packet
  - If a packet does not contain the expected SQ it is rejected and no reply is sent
  - If a packet contains the correct SQ it is accepted, the command is executed and reply is sent
- Except for SQ the target also performs Packet Type checks:
  - If a “middle segment” or “last segment” command is received as the first segment of a segmented SDU it is rejected
  - If a “first segment” or “unsegmented” command is received as a continuation segment of a segmented SDU it is rejected

- Ensures correct SDU reassembly with both Incrementing and non-Incrementing address transactions
- May require RMAP target modifications

SpW-D Segmentation & Reassembly: Mapping of control information (1/2)

How are the SQ and Packet type fields conveyed?

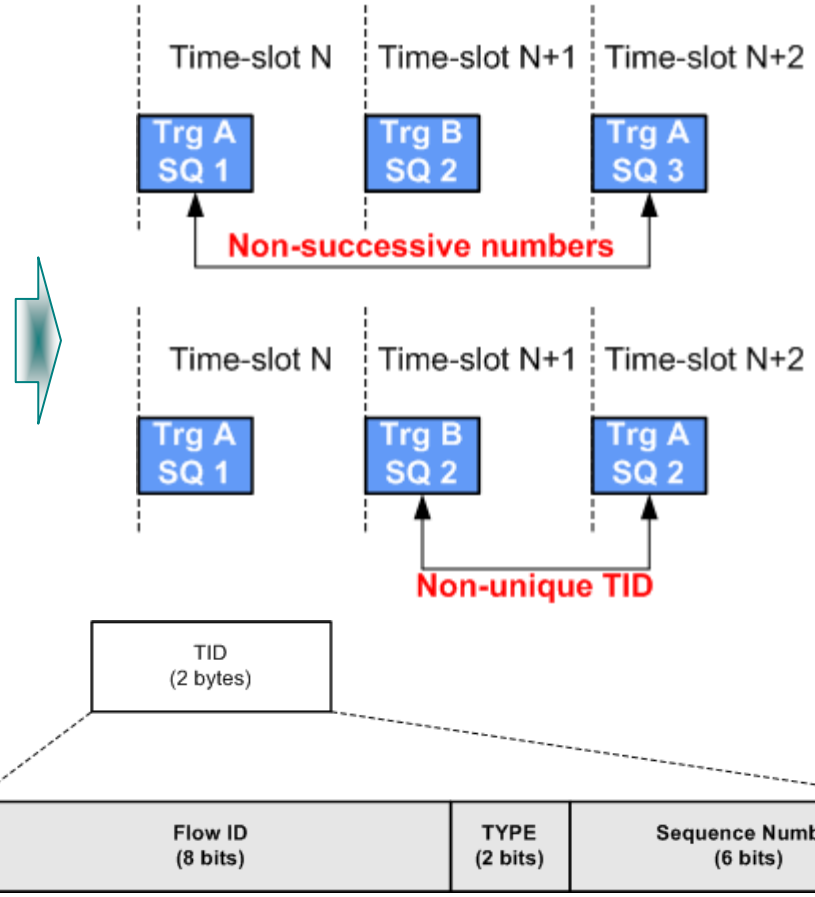
- Do not modify the RMAP header. Use the payload instead:
  - More flexible but,
  - payload is not present Read Commands,
  - the approach is not compatible with existing RMAP Cores
- Use the RMAP Header :
  - Segmentation information will exist in all RMAP packets, but
  - header fields serve a purpose in the RMAP protocol
  - only a few bits in the RMAP header fields are unused
  - since the field contains SQ information, the selected field shall be one that is retained in the Initiator until the reply is received
  - at the target side header info is not visible to layers above RMAP => target modification, or possible implementation through the Authorization logic I/F
- SOLUTION: Use the, 2-bytes, Transaction Identifier (TID) RMAP Header field
  - Upon the transmission of a command the TID is stored in the Initiator
  - It is kept until either a time-out occurs or a reply with identical TID is received
  - If SQ is contained in TID, then upon time-outs the Initiator will pass SQ information of the non-ACKed PDU to the application
  - Since SQ is contained in TID, then upon reception of a reply the Initiator receives the SQ of the ACKed RMAP command

SpW-D Segmentation & Reassembly: Mapping of control information (2/2)

TID field structure

- Flat TID:
  - Contains a 2-bits Packet Type field
  - Contains a 14-bits SQ field
  - Problems with SQ sequence for a single target, or
  - Problems with identical TIDs in replies from different targets

- Structured TID:
  - Contains a 2-bits packet type field
  - TID contains Flow ID of 8 (or less) bits Flow ID can be associated to a target
  - Contains a 6-bits SQ field
  - Compatible with ECSS-E-ST-50-52C RMAP Service Interface
  - FlowID or SQ field may be further reduced to support redundancy information

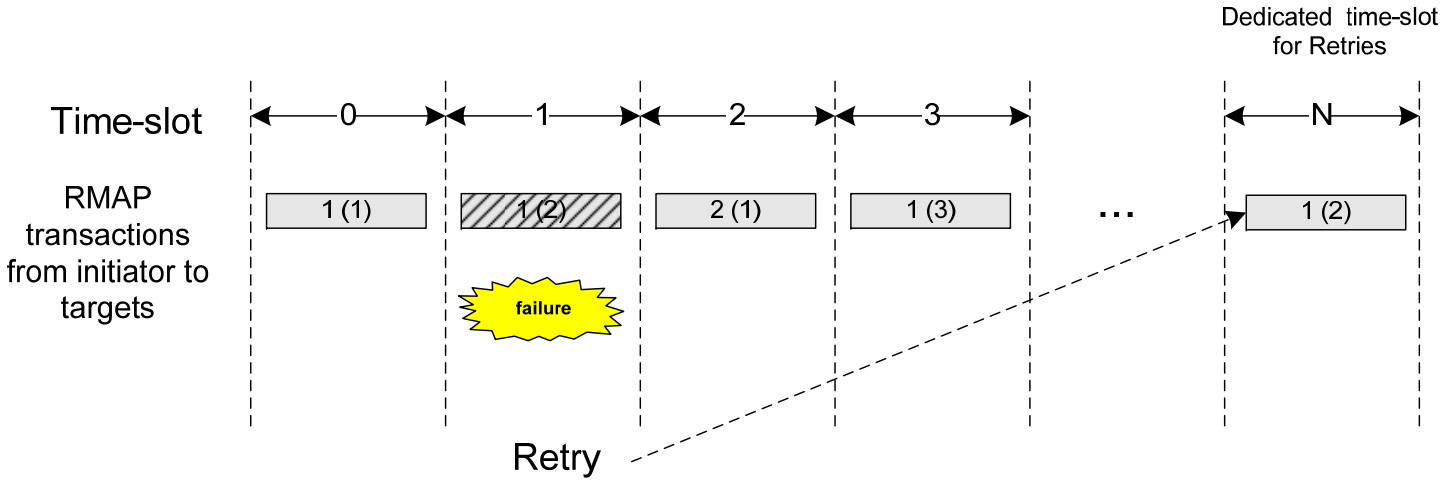


Open issues:

- What is the maximum latency in the retransmission of a failed transaction?
- Is there a possibility for out of order reception of PDUs if segmentation is used?
- Is network efficiency affected?
- Is the retry mechanism compatible with all the scheduling techniques?
- Does the implementation of the retry mechanism introduce significant problems?
- **A major issue for the implementation of the Retry mechanism is to define the time-slot(s) in which a failed RMAP transaction is allowed to be re-transmitted**

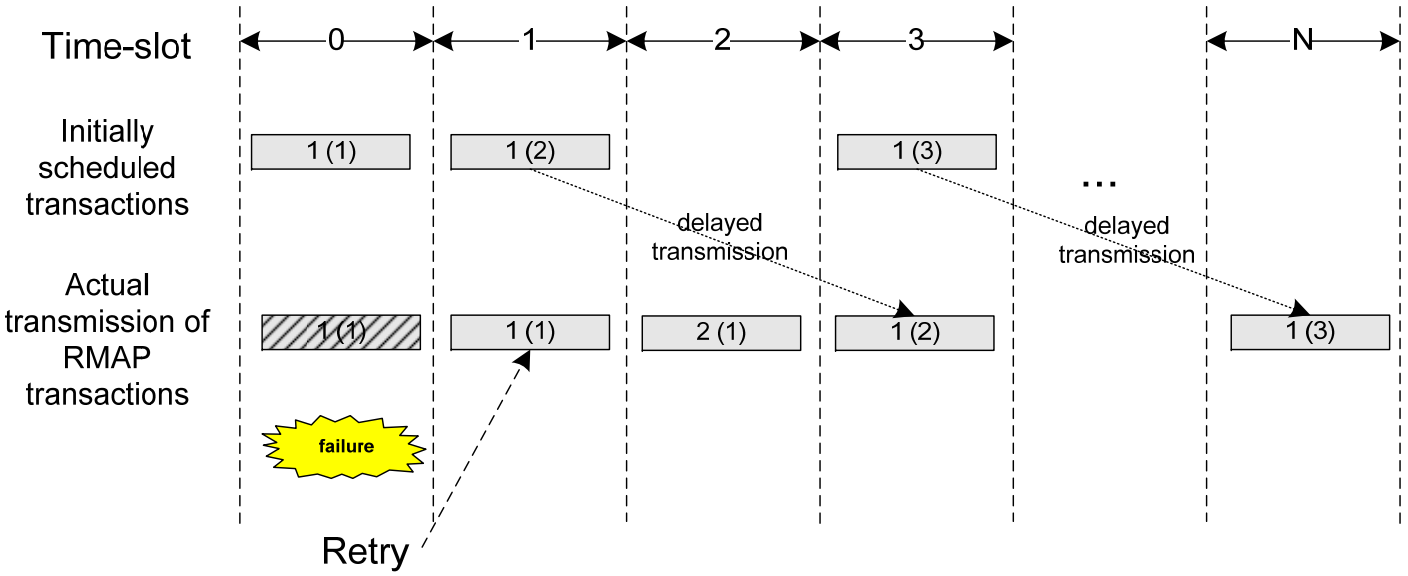
Retry is not yet specified in SpaceWire-D draft B spec.

SpW-D Retry(A): Use of dedicated time-slot(s)



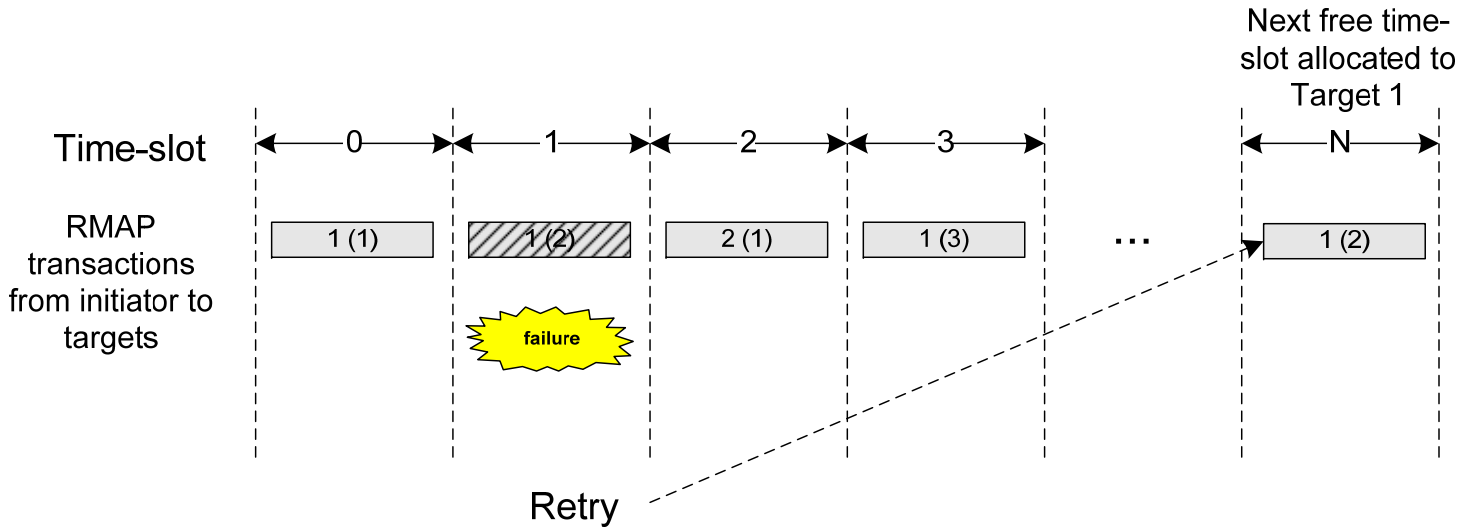
- Allocate a number of time-slot(s) dedicated for Retries
- Send Retry in the dedicated time-slot(s)
  - Bounded delay
  - No disruption of scheduled transactions
  - Waste on network bandwidth
  - Required time-slots may increase depending on number of initiators
  - Out of order delivery

SpW-D Retry(B): Use next time-slot for the same Target



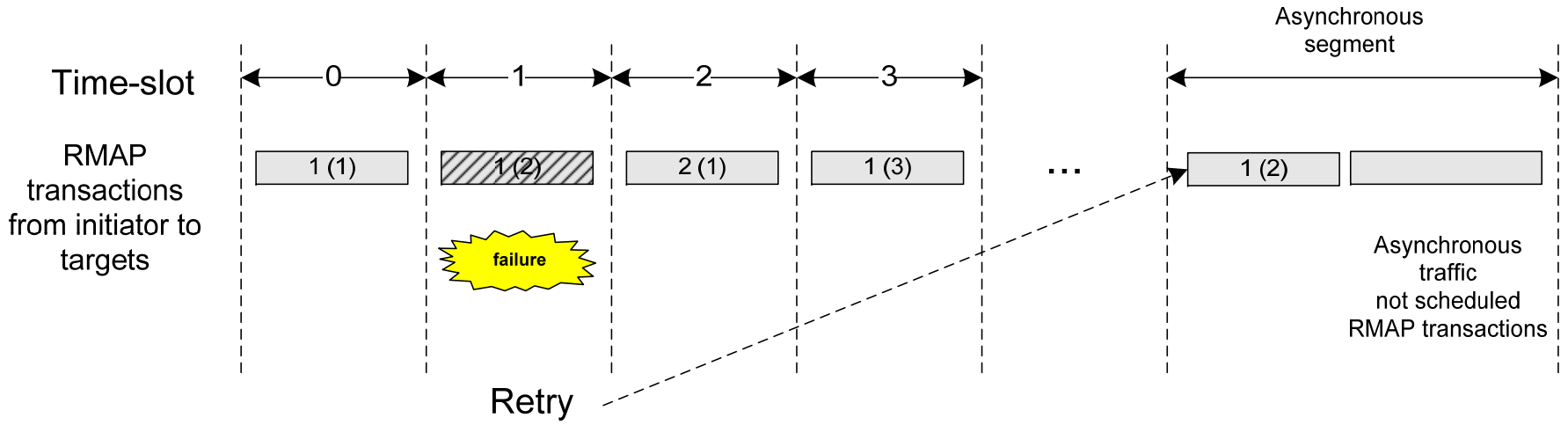
- Send Retry in the next time-slot for this Target
- With highest priority than any scheduled transaction (insert at head of Tx FIFO)
  - Bounded delay
  - No out of order delivery
  - Disruption of scheduled transactions (inserts Latency)

SpW-D Retry(C): Use next free time-slot for the same Target



- Send Retry in a free time-slot for the same Target
- If there are no free time-slots retry will never be performed at all, and Retry queue will overflow
- So, if there is no free time-slot, the Retry must be sent in a next time-slot for this Target, with lowest priority than any scheduled transaction (inserted at the tail of Tx FIFO)
  - No disruption of scheduled transactions
  - Highly variable delay
  - Out of order delivery

SpW-D Retry(D): Use asynchronous segment



- Send Retry in the asynchronous segment with highest priority than any asynchronous transactions
  - Requires Asynchronous Segment
  - Bounded delay
  - No disruption of scheduled transactions (synchronous traffic)
  - Out of order delivery
  - Initiators shall extend scheduling to ensure that they wait for an interval for possible retries from other initiators before transmitting asynchronous traffic
  - Asynchronous & Guard segment duration shall be set equal to the number of initiators even if less time-slots are required for Asynchronous traffic

Evaluation of SpW-D alternative retry mechanisms

	Out of order delivery	Latency in Retransmission	Waste of Resources	Other
Retransmission in dedicated time-slots(s)	Yes	From 1 to 63 time-slots	Yes	Latency increases to more than one epoch for multiple failures
Retransmission in the next free time-slot for this target	Yes	Highly Variable Depends on schedule table configuration and traffic profile	No	
Retransmission in the next time-slot for this target	No	Depends on schedule table configuration	No	Inserts latency to the next transaction(s)
Retransmission in dedicated time-slots(s) for asynchronous traffic	Yes	From 1 to 63 time-slots, depends on current time-slot	Possible	Requires an asynchronous segment in the network

Retransmission in the next time-slot for this target is the more favourable alternative

PRESENTATION AGENDA:

SpW-D Simulation & results

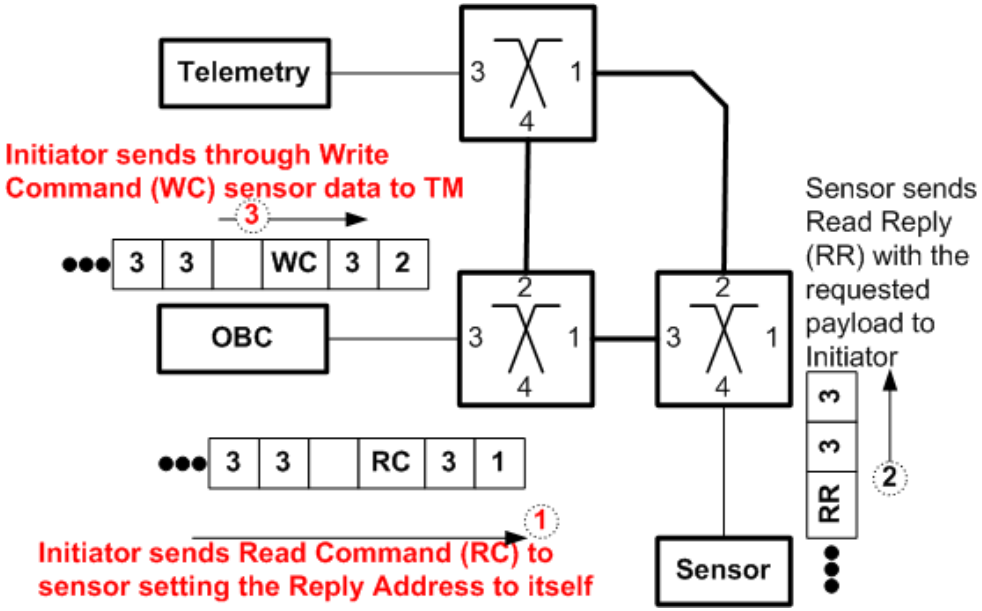
SpW-D scheduling analysis & proposals

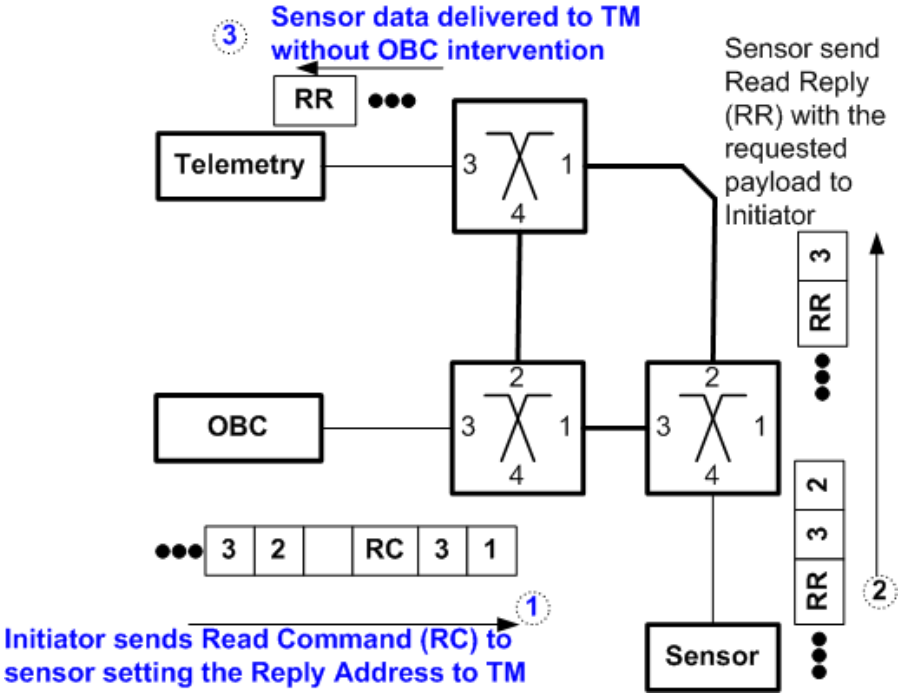
Proposals for Segmentation & Retry mechanisms

Other issues

Other Issues (1/2)

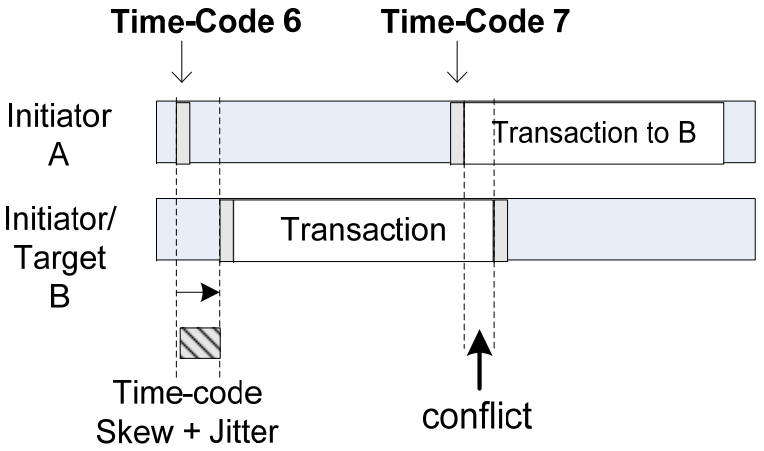
- Overhead caused to transfer data from a target to an initiator through another initiator
- Using acknowledged transactions and setting the Reply Address Field and to the final recipient reduces consumed BW
- Open points:
  - How shall the reply time-out on Initiator 1 (command sender) be handled?
  - How does initiator 2 (reply receiver) handles a reply for which no command has been sent?

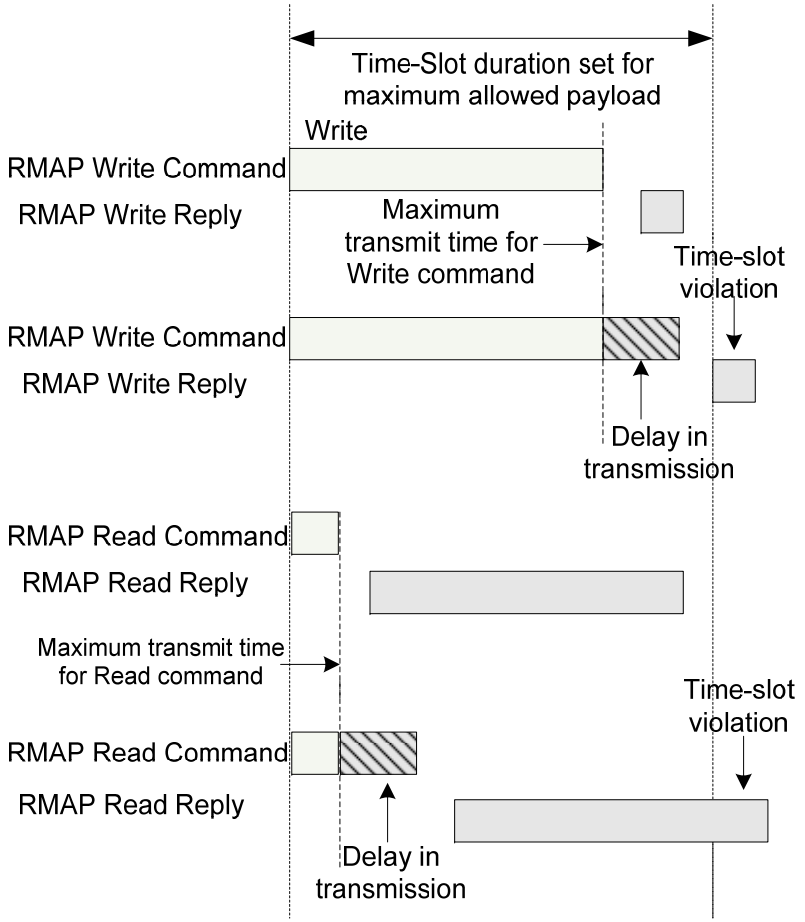




Other Issues (2/2)

- Initiator/target response to possible time-slot violation issues is not specified
  - Transmit time exceeds the maximum
  - Receive time exceeds the maximum
- Overhead Time-code distribution issues are not analysed
  - behaviour when time-codes are lost, time-code watchdog timer expires
  - time-code skew & jitter in networks with many routers and low speed SpW links





SpW-D HW implementation issues

System Architecture issues:

In a shared memory architecture in which the SpW-D is a local bus master:

- The bus arbiter shall: support interleaved bursts and priorities, bound the master's continuous burst time
- The bus specification shall support "unspecified length" transactions in order to resume interrupted transfers
- The system designer shall ensure that atomic transactions have bounded duration

SpW-D Scheduler issues:

- Prefetching commands in order to obtain SpW-D timings requires more memory resources and presents difficulties with Retry and with the proposed scheduling schemes

Initiator issues:

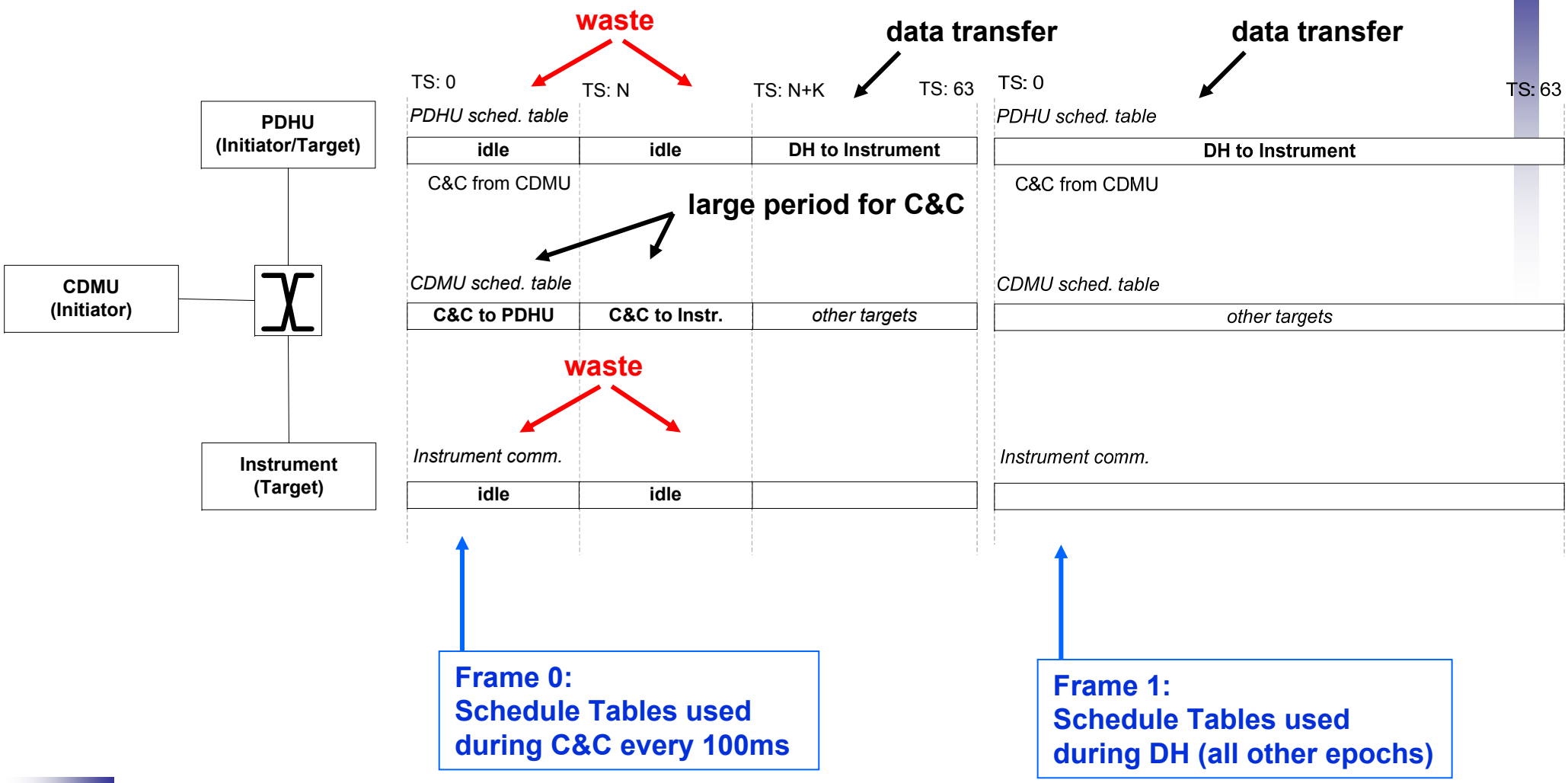
- In order to fulfill the SpW-D timing requirements, the initiator command controller should overlap command fetching from the memory and command transmission. This also ensures a constant "start of transmission"

Extensions of an existing RMAP Core for SpW-D operation:

- RMAP commands at the initiator are processed in FIFO order, **BUT**, commands are not issued by the application(s) synchronously to the underlying network schedule. Therefore, commands shall not be issued to the RMAP but to the SpW-D scheduler. The SpW-D scheduler will forward them to the RMAP core according to the network schedule
- Timeouts in existing RMAP cores are programmed by the application in time (e.g. clock ticks) and not in time-slots; this is not suitable for SpW-D since time-codes have jitter. In addition, the application shall know the exact transmission time of the command in order to program the timeout value accurately

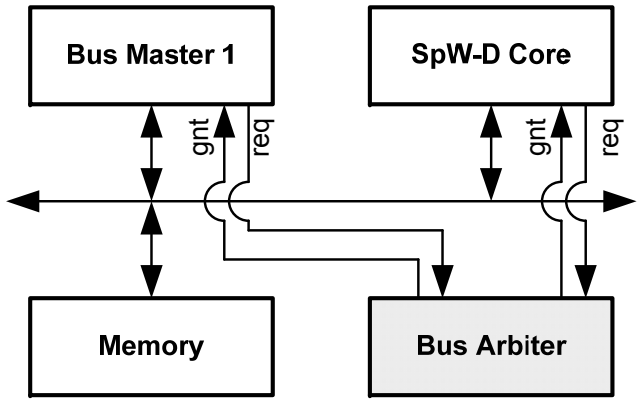

THANK YOU

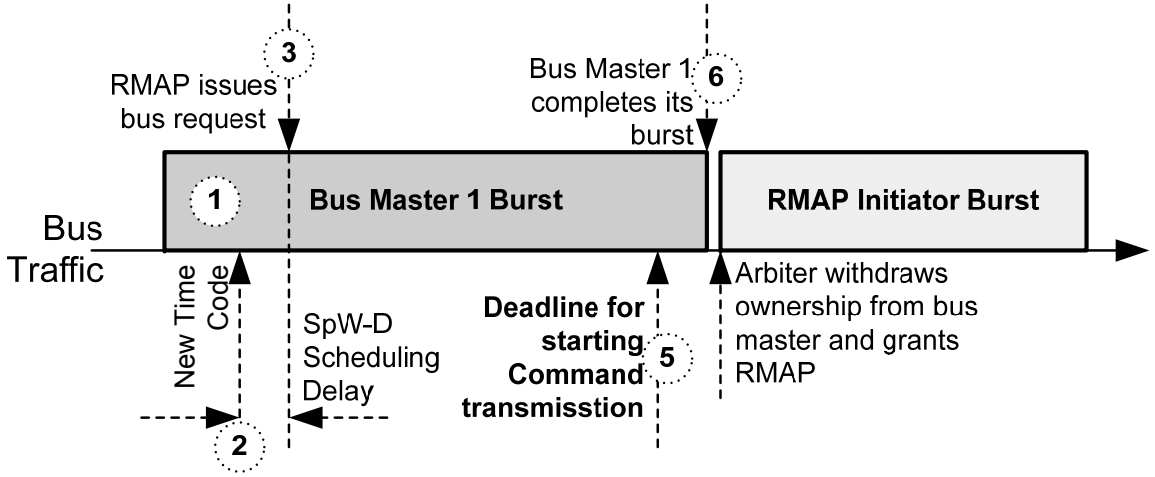
SpW-D Scheduling(C):Use of different schedule tables in different epochs



System Architecture considerations (1/2)

- Bus Master 1 is the system bus owner and performs a transfer to/from the memory
- Time-Code is received while the Master 1 burst is still in progress
- The SpW-D Core requests bus ownership in order to initiate a transaction, but
- IF the bus arbiter does not support interleaved bursts:
- The SpW-D Core does not initiate transmission on time and the time-slot boundaries will be violated. The same problem may happen with the target response



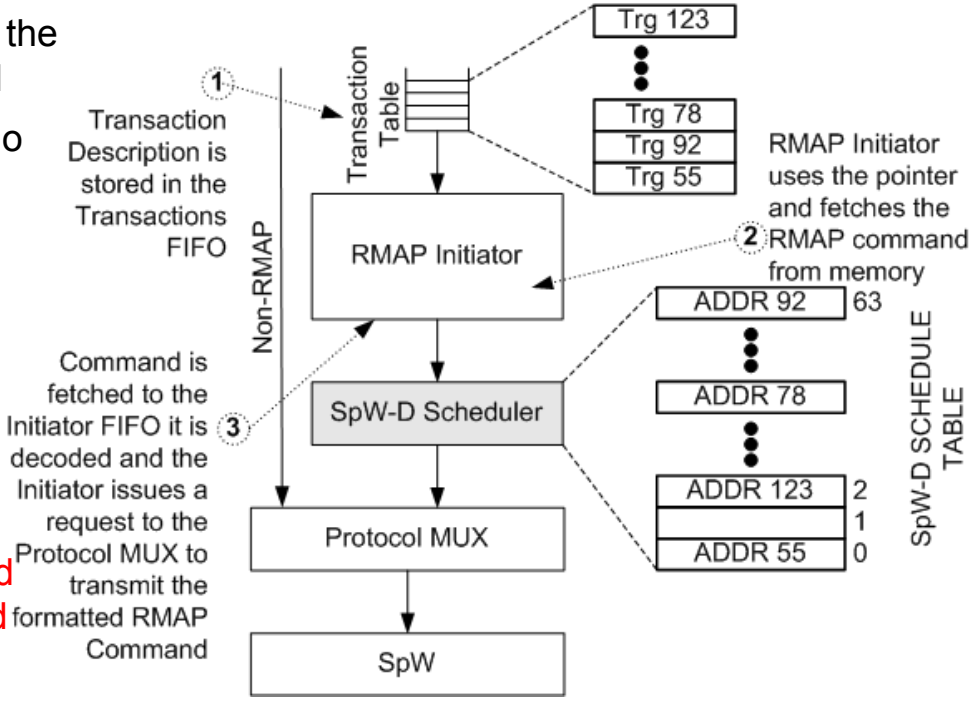


System Architecture considerations (2/2)

- The bus arbiter shall support interleaved bursts
- The bus arbiter shall support either:
  - Static priorities and the SpW-D shall be connected to the highest priority
  - Different priority levels so that a bus master can assert a “priority” signal in case it is about to miss its deadline
  - Bounding the time of burst by a single master
- The bus masters shall support interleaved bursts
- The bus specification shall support “unspecified length transactions in order to resume interrupted transfers
- The system designer shall ensure that atomic transactions have bounded duration

SpW-D Core Architecture considerations(1/3)

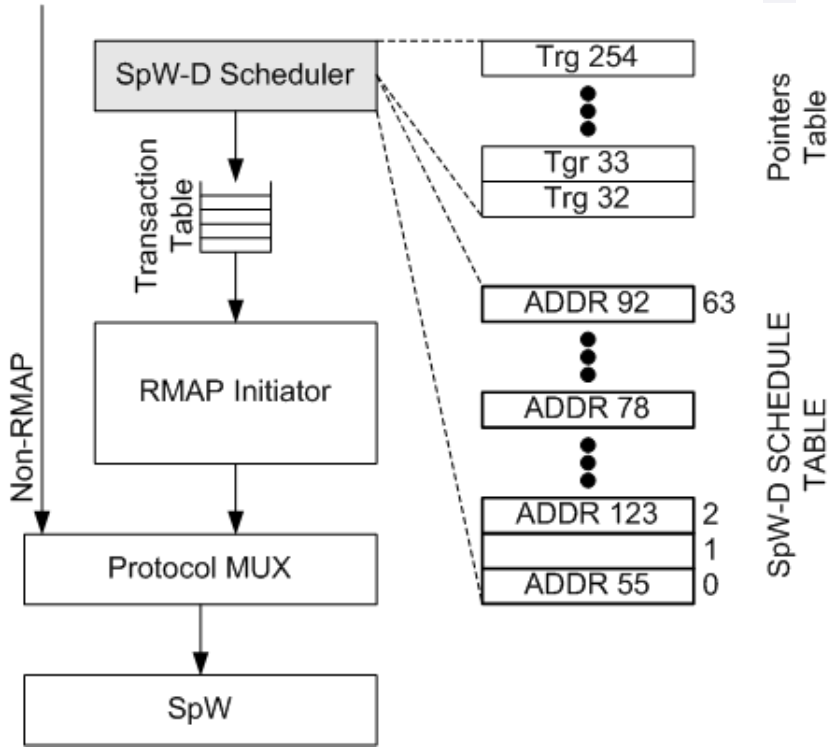
- The RMAP Core contains a transaction table where the host stores the memory addresses of the RMAP commands for transmission
- The RMAP Initiator fetches the pointer, requests access to the memory and fetches the command structure for transmission
- After the command is fetched the initiator requests from the Protocol MUX to transmit its formatted RMAP command
- This approach is ok for RMAP operation since there is no scheduling
- However, with SpW-D the following problem exists:
  - Pointers download by the host SW cannot be synchronized with the network schedule
  - Different SW tasks download pointers asynchronously
  - Commands will be issued with delay
- In the example, commands to targets 55 & 92 are issued with no delay, command to 78 with one epoch delay and command to 123 with two epochs delay



SpW-D Core Architecture considerations(2/3)

Solutions:

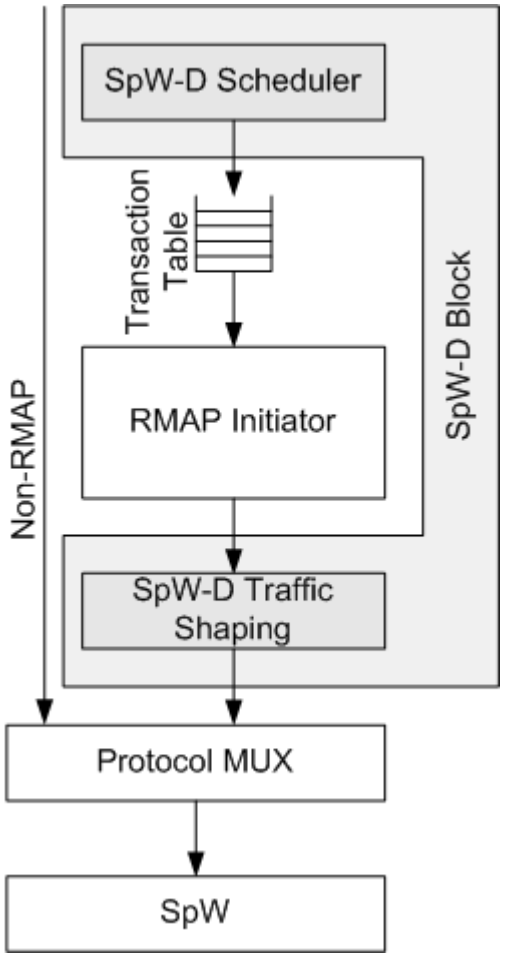
- Modification of an existing RMAP Core:
  - SpW-D scheduler will pass Target address and
  - The RMAP Core will issue the command to the respective target
  - Searches in HW are costly
  - Non-scalable; search delay increases as the number of supported targets increases
  - RMAP Core modification is not always possible nor is it desirable
- Placement of the SpW-D scheduler above RMAP:
  - The SpW-D intercepts write accesses to the Transactions Table and stores them into its own table
  - Schedule table same as with the previous approach
  - SpW-D scheduler forwards commands to the RMAP Core according to the schedule
  - SpW-D scheduler pointers table shall be NxT, where N is the number of simultaneously supported commands for a target and T the number of supported targets



SpW-D Core Architecture considerations(3/3)

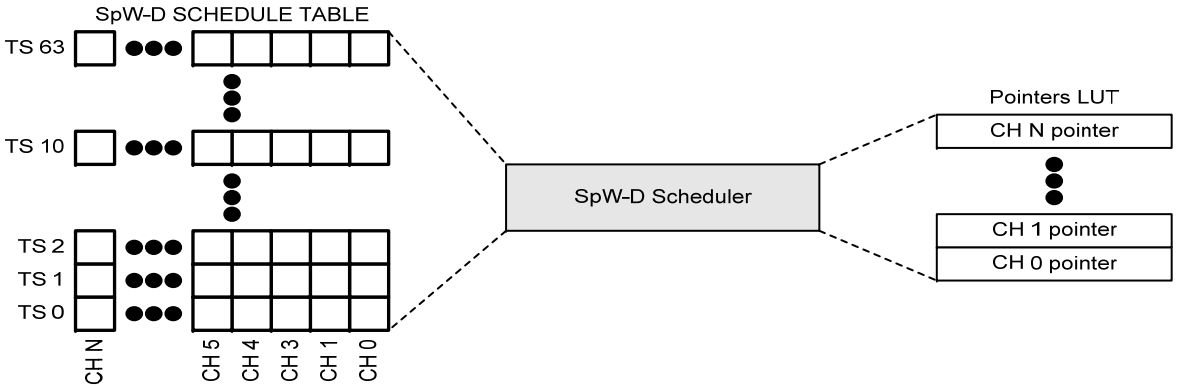
Time-outs handling issues with existing RMAP Cores:

- Existing RMAP Cores implement time-outs in clock cycles and not in time-slots
- Not suitable for SpW-D operation since:
  - The time-out shall be set equal to the time difference from the transmission of the command until the arrival of the next time-code
  - The exact time instance the Command has been is transmitted shall be known to the application
  - Not always possible since it relates to system bus arbitration delays, transfer delays etc.
  - Time-code has jitter and time-outs cannot follow slightly early/late time-codes
- The problem can be mitigated (not solved) by extending the SpW-D block in order to schedule initiator transmission at microsecond level
  - It will ensure that RMAP Commands are transmitted after a constant (programmable) time after the time-code
  - It shall be ensured that the selected RMAP Core starts counting the time-out from the time the first NCHAR is transmitted in order to support common time-out for commands of different lengths



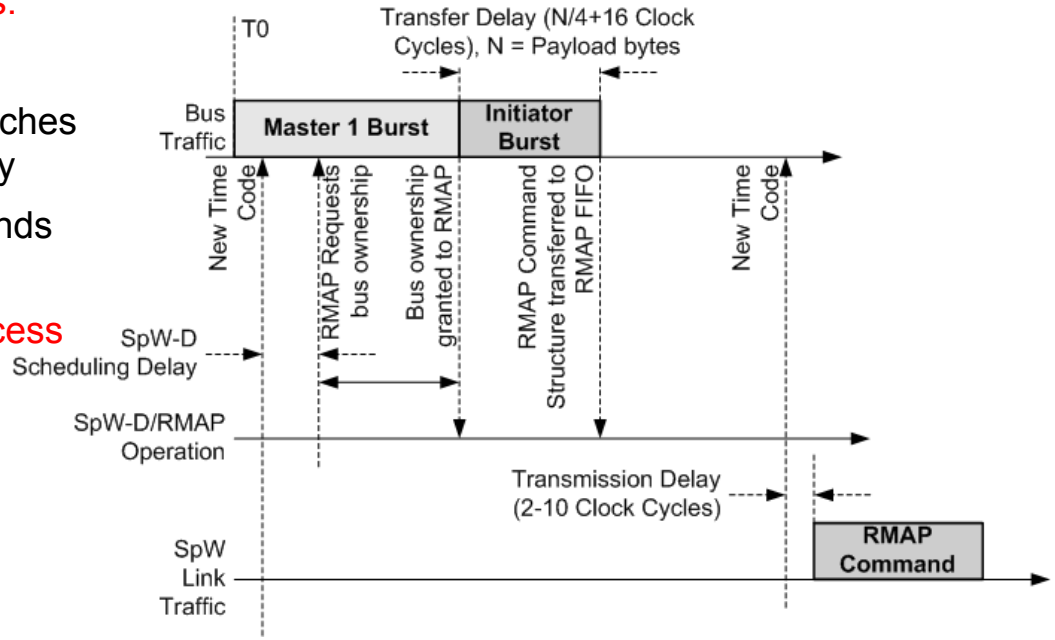
SpW-D Scheduler considerations – Schedule table structure

- "Target addresses" based Schedule Table
  - Increases the size of the schedule table prohibitively for a large number of supported targets (8-bits used per target)
  - Leaves unused entries in the schedule table if non-contiguous Target Address space is used
  - Does not allow the transmission of different flows to a single target
- "Channel" based scheduler
  - A schedule table containing a single bit per "channel" is used
  - A LUT associates the schedule table with addresses of the commands in memory
  - Logic at application is slightly more complex, but
  - Decreases the size of the schedule table
  - Allows the transmission of different pieces of information, with different priorities, to the same target
  - It is possible to send commands to different targets in different epochs without modifying the schedule table



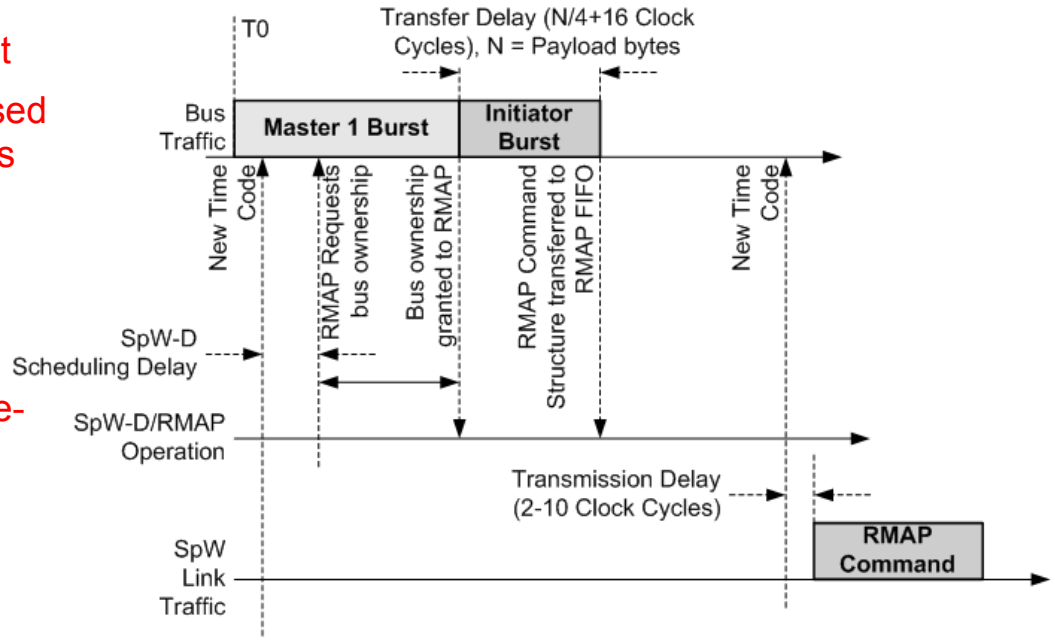
SpW-D Scheduler considerations – Commands prefetching(1/2)

- During time-slot N, the command that shall be transmitted in time-slot N+1 is fetched by the Initiator
  - The time interval between time-code reception and transmission is decreased dramatically
  - Time-slot duration is decreased and consequently overall performance is increased
- Technique is non applicable to existing RMAP Cores:
  - RMAP Initiators operate as bus masters
  - With pre-fetching the SpW-D scheduler pre-fetches the command and stores it into its own memory
  - When the time-slot arrives the SpW-D commands the RMAP initiator to fetch a command, but
  - The RMAP will request bus ownership, not access to the SpW-D memory
  - Therefore modification of the RMAP core is required



SpW-D Scheduler considerations – Commands prefetching(2/2)

- Retries are much more complex and either:
  - Cancel the advantage of prefetching; during time-slot N the Command sent in the previous time-slot shall be transmitted. However the next command is at the initiator's transmission buffer already
  - Retry cannot be performed in the next time-slot
  - Require RMAP Core modifications and increased memory resourcesin order to keep the previous and current time-slot commands
- Complicates the support of alternative SpW-D scheduling schemes:
  - Multiple transactions per time-slot increases memory resources requirements in order to pre-fetch multiple commands
  - Supporting different scheduling in different epochs is complex since each initiator shall be aware of the time-masters schedule



RMAP Initiator Controller considerations

- RMAP does not have timing constraints
- The state machine in a RMAP initiator may either:
  - Fetch the RMAP command segments and then transmit
  - Overlap command fetching and RMAP transmission
- Fetching command segments and then transmitting
  - Is simpler since it can be implemented with a single state machine, but
  - Increases the total delay to the transmission of the RMAP command and may exceed the timings in a SpW-D network
  - Introduces jitter when commands with different sizes are transmitted
- Pipelining command fetching and transmission:
  - Is more complex, but
  - It can guarantee that the SpW-D timings will be respected
  - Has a constant “start of transmission” time