Protocol Validation System
for On-Board
Communications

Evaluation of the SpW-D draft specification through Simulation and Prototyping

16th SpaceWire Working Group, 21st March 2011
Noordwijk, Netherlands

esa

teletel

Protocol Validation System (PVS) for on-board communications

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

Start of Study: July 2010

End of Study: February 2011

esa

teletel

SpW WG #15 discussions

Efficiency

- Decreased due to the Half Duplex operation of the protocol
- Increased at low link rates for constant Initiator/Target latencies
- Increased for large payload lengths/large Time-Slots

Scheduling

- Draft B proposals: Simple, Concurrent, Multi Slot
- Multi Transaction Slot increases efficiency (full duplex operation), decreases latency for small payload packets and reduces Time Codes “consumption”
- Multi Epoch increases the Time Slots for scheduling. Requires more complex SpW-D Scheduler

- **Segmentation & Reassembly**
  - Large packets transmitted in single slots result in increased latency
  - Segmentation can overcome the latency issue and support large packets
  - Can be implemented at the Initiator side only for non-incremental address RMAP transactions
- **Retry**
  - Use next allocated time slot => Disrupts isochronous communication
  - Use time slots dedicated to Retries => Reduces efficiency

PRESENTATION AGENDA:

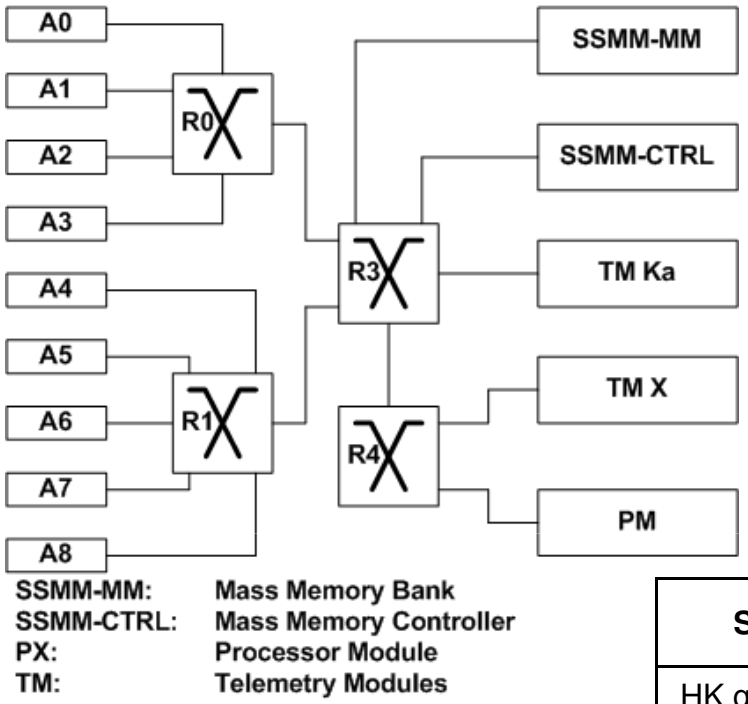
SpW-D Evaluation, simulations on realistic architectures

Proposed SpW-D Evolutions

SpW-D Prototyping, Hardware Implementation Issues

SpW-D Prototyping, Measurements & Evaluation results

Simulated Configurations



Traffic Paths:

- Science data (SC): Ai -> SSMM CTRL -> SSMM-MM
- Housekeeping data (HK): Ai -> PM -> SSMM-CTRL -> SSMM-MM
  PM -> SSMM-CTRL -> SSMM-MM
- C&C data (CC): PM -> SSMM-CTRL
  PM -> Application
- TM data (TM): SSMM-MM -> TM-Ka
  SSMM-MM -> TM-Ka

SCENARIO2: Same overall throughput, much higher packet rate with smaller payloads

Scenario 1	A0	A1	A2	A3	A4	A5	A6	A7	A8
HK generation rate	.16	.2	8	.053	16	1.33	5.33	.32	.16
HK Payload length	20	20	20	20	20	20	20	20	20
SC generation rate	.16	.4	9.67	.026	20	.098	3.2	.007	.002
SC payload length	4K	4K	4K	4K	4K	4K	4K	4K	4K

Scenario 2	A0	A1	A2	A3	A4	A5	A6	A7	A8
SC generation rate	.008	.02	.485	.0013	1	.0049	.16	.0003	.0005
SC payload length	200	200	200	200	200	200	200	200	200

Schedulability Analysis Methodology

No Multi-transaction, no Multi-Epoch scheduling used

1. Determine the Initiator-Target pairs for each path/transaction and type(s) of transaction(s)
2. Determine Link Speeds and Time Slot periods (preferably 122 us or 244 us) which satisfy the Initiator with the maximum throughput requirements
3. For each initiator calculate the Time Slots per epoch required for its traffic profile
   a. Calculate the overall frequency Fs which satisfies all sources traffic profiles for each initiator
   b. If Fs > $F_{TIMESLOT}$ re-iterate TimeSlot period/ Link speed, else determine the number of timeslots required for all transactions and round to the closest higher (or equal) integer value
4. If the sum is greater than 64 repeat the analysis with different Time Slot Period/Link Speed values
5. Allocate the transactions in Time Slots for all Initiator-Target pairs and insert idle Time Slots if common paths are used
6. Map the Initiators processing delays in idle Time Slots for each initiator
7. If the number of Time Slots is more than 64 repeat the analysis with different Link Speed/Time Slot period values

Results Conclusions

RESULTS:

- Scheduling with SpW-D was not possible for Scenario 2
- This is due to the facts that:
  - The science packets generation rate is too high requiring high SpW Link Speeds and short Time Slots
  - Processing delays for science data is constant
  - Small Time Slot periods result in more wasted Time Slots due to the fact that during processing the Initiators are not capable to initiate more transactions

CONCLUSIONS:

- Diversity in packet rate generation results in wasted Time Slots
- Isochronous sources determine the Epoch/Time Slot timings. If the resulting periods are too short, multi-slot timing may be required for sources with larger payloads, consuming more Time Slots
- Traffic profiles shall match the Epoch/Time Slot periods in order not to result in wasted Time Slots
- Application processing delays at the Initiators may waste Time Slots
- A large number of traffic sources imposes challenges in the Schedulability Analysis with SpW-D Draft B specification

The Initiators need more "opportunities" to initiate Transactions. Multi-Epoch and/or Multi-Transaction scheduling can satisfy this requirement

PRESENTATION AGENDA:

SpW-D Evaluation, simulations on realistic architectures

Proposed SpW-D Evolutions

SpW-D Prototyping, Hardware Implementation Issues

SpW-D Prototyping, Measurements & Evaluation results

Requirements, Restrictions and Steering Committeee inputs

C&C

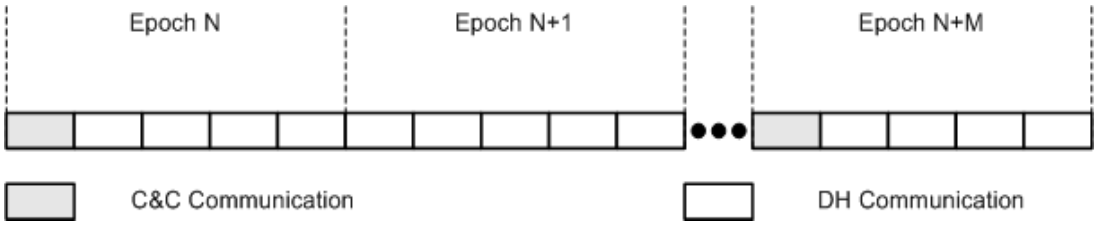
- Support for 1Hz loops for 1pps devices
- Support for 10Hz loops (AOCS 2Hz – 16 Hz are typical cases with 1553)
- 100 Hz (high-end AOCS, pointing, maximum 1553 capability)
- 1KHz for motor control, robotics, fine pointing

- **HTDT**
  - Optimization shall be achieved for HTDT
- **Time Distribution**
  - CUC Time Distribution (see 15th SpW WG meetings)
- **Determinism**
  - To KILL or not to KILL?
- **Integrity**
  - Provided by the SpW parity and RMAP CRC
- **Routers delays**
  - Typical delays for time codes can be up to 10 us

SpW-D Scheduling (1/2)

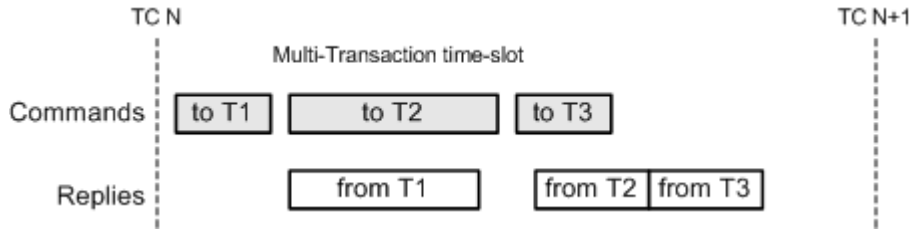
Multi Epoch:

- Supports a wide “dynamic range” of scheduling periods
- Can be implemented with CCSDS CUC Time Distribution
- TimeSlot 0 of certain/all epochs can be allocated for Distribution of CUC frames to SpW-D Initiators
- Each initiator keeps track of the epochs until the reception of the next Time Distribution frame



Multi Transaction slots:

- Can be used for C&C
- Can be combined with Multi-Epoch scheduling
- Open point is the successive Commands transmission or transmission after the reception of the Reply for the previous command



Multi Slot:

- Solves the problem of medium sized blocks (1-16 Kbytes) transfer without segmentation but consumes more Time Slots than the other types of scheduling
- Maximum number of consecutive Time Slots shall be constrained:
  - In order to bound the maximum latency for transmission
  - For implementation issues
    - Verified writes require the presence of a verify buffer at the target of at least the same size as the payload transferred
    - Flight FPGA devices have limited memory resources
  - Long Verified Writes increase the Reply delay prohibitively and reduce the overall efficiency
  - RMAP CRC also needs to be considered, especially for longer packets

Limit	**Verified Writes payload to**	**4K**
	Non Verified Writes payload to	**16K**

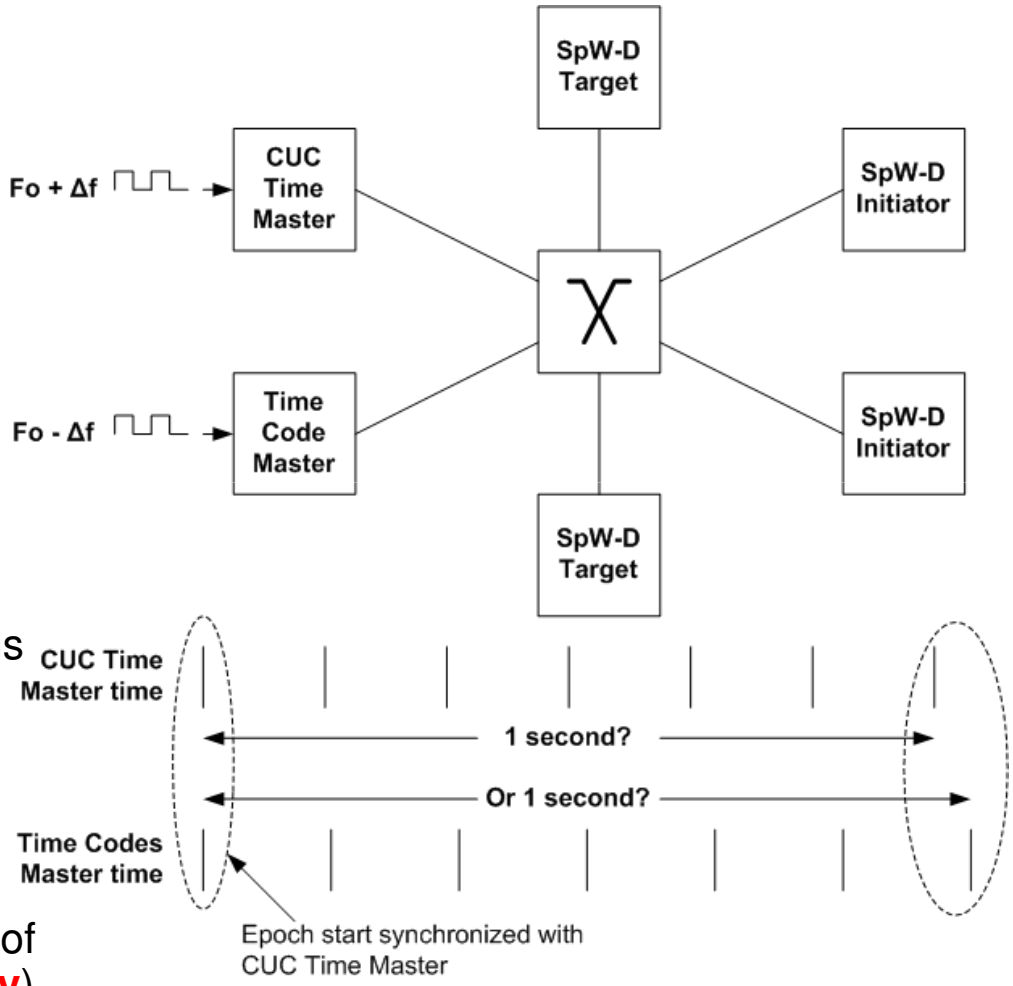
Time Slot Periodicity (1/2)

- Time Distribution shall be supported
  - Epoch period shall be 1s divisor
  - Epoch period shall allow for distribution of time information to many devices within 1s
- The proposal is to have 64 or 128 epochs/s
  - 64 epochs/s => Time Slot periodicity ≈ 244 us
  - 128 epochs/s => Time Slot periodicity ≈ 122 us

Time Slot Duration	Link Speed (Mbps)	Maximum Payload Length	Efficiency
122 microseconds	200	1650	67.62 %
	100	870	71.31 %
	50	432	70.81 %
	10	56	45.90 %
244 microseconds	200	3760	77.04 %
	100	2000	81.96 %
	50	1018	83.44 %
	10	178	79.46 %

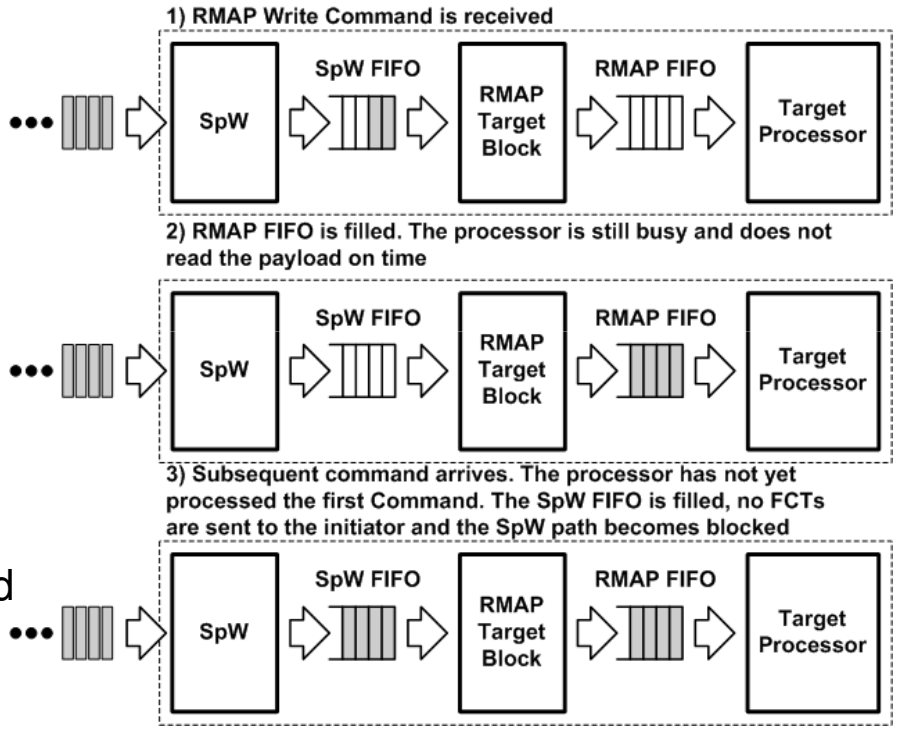
Time Slot Periodicity (2/2)

- Epoch period shall be 1s divisor in order to synchronize the epochs to 1 second, but:
  - **Is the CUC Time Master and the Time Codes Master the same device?**
  - **How is the clock signal distributed within the S/C?**
- The objective can be achieved:
  - If the Time Codes Master and the CUC Time Master is the same device, or
  - If coherent clocking (PLL) is present
- Otherwise:
  - A guard time at the end of certain epochs shall be defined to compensate for asynchronous transmission of CUC frames (**decreased efficiency**)
  - Epoch start shall be aligned. This may cause jitter if alignment is performed at once (**causes jitter**), or implementation of PLL logic at the time master (**complexity**)



RMAP Command Options

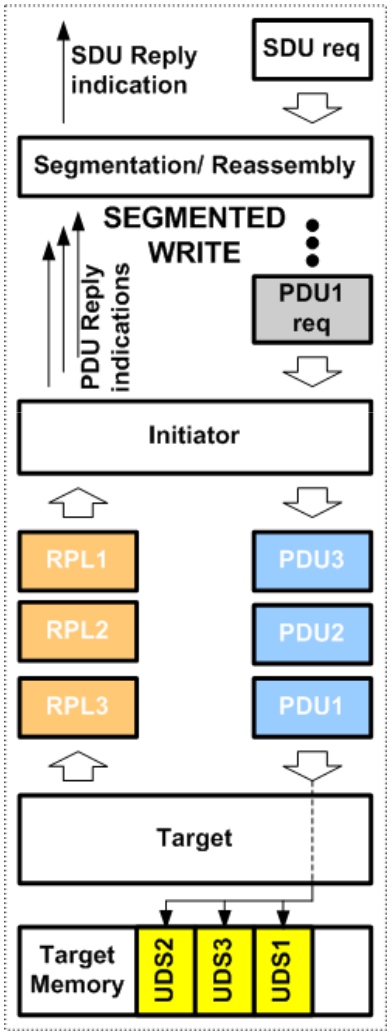
- Non incremental RMAP addressing:
  - Implies the presence of FIFOs at the receiver
  - The FIFO shall have empty space in order to store the RMAP payload
  - If the host application (e.g. processor) does not empty the FIFO before the next RMAP Write is received, the RMAP and SpW FIFOs may become full and no FCTs are sent back
  - **All links from the Initiator to the Target become blocked**
  - In addition, handling of failed segmented transfers presents difficulties
- Incremental RMAP addressing:
  - Shared memory between the RMAP block and the application
  - In case the host system does not process the RMAP payload in time and another RMAP command is received then the payload may be overwritten or corrupted (partial overwrite) but the SpW Path is **NOT blocked**

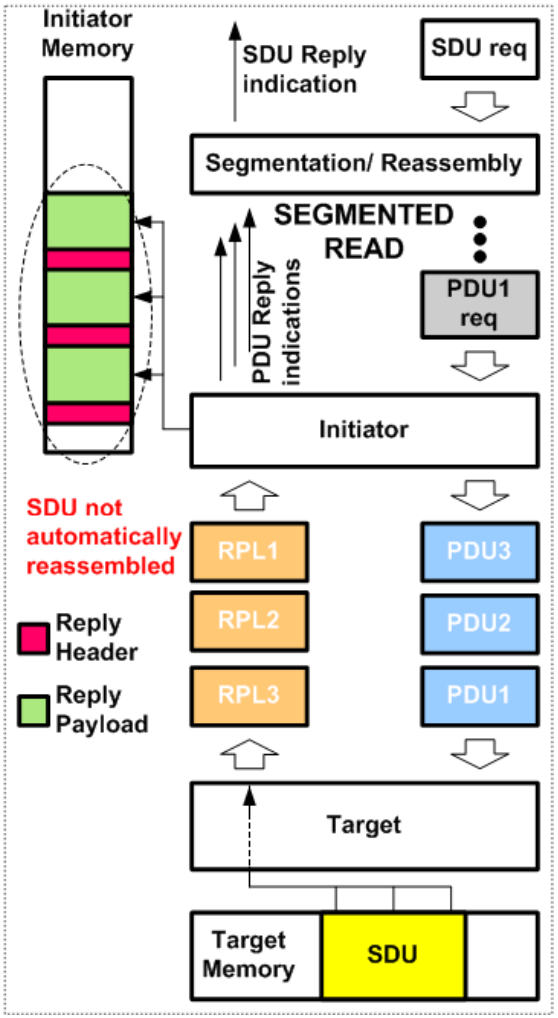


Segmentation

- Implemented as a discrete layer above RMAP
- The applications do not have visibility on Segmentation information
- Implementation Segmentation and Reassembly at the initiator side is possible (non-incremental addressing is not supported)
- A request for a large SDU transfer is segmented to multiple requests to successive Target memory spaces
- For RMAP Writes the SDU is automatically reassembled at the Target memory
- For RMAP Reads the Initiator shall store the Header and Payload of the Reply in separate memory areas in case automatic Reassembly is required
- For RMW Segmentation does not apply. Maximum payload length is 8/4 bytes

Open Point: How is the Target notified that a SDU is available in its memory?





Retry

- Not implemented at SpW-D level
  - Retry will either consume bandwidth or will affect the scheduling. This cannot be tolerated for all kinds of applications
- Retry and Segmentation at the **Target side**:
  - No actions are required
- Retry and Segmentation at the **Initiator side**:
  - Upon a Reply timeout or a Reply indicating error the application shall be immediately notified
  - Subsequent PDUs are aborted
  - At the Initiator side an error is immediately indicated to the application
- Recommended to use a structured TID or indicate SQ of failure to the application:
  - Allows identification of the Target denying the Command
  - May be useful for fault detection
  - May also be useful for Read transfers; no need to re-read the entire SDU.

Other issues

- Channels:
  - The concept of channels offers the following advantages:
    - Priority support
    - Support for non-successive transactions to a single Target
    - Simplifies schedule table construction
  - Not to be included as a “Normative” section necessarily
- Timing fault containment:
  - KILL functionality is proposed for Initiators
  - Transmission abort functionality may be implemented for Initiators in which the transmission has not started after a predefined time after the Time Code. This time depends on whether the Command to be transmitted is Read or Write. Open point how this will be implemented with multi-transaction slots
  - Both of the functions above, does not protect against babbling idiots. This can only be overcome by the Routers.
- Packet Transfer:
  - Where is the protocol information encoded? In PID or in TID?
  - How is a Target notified for the reception of a PTP packet? Use standard address?

PRESENTATION AGENDA:

SpW-D Evaluation, simulations on realistic architectures

Proposed SpW-D Evolutions

SpW-D Prototyping, Hardware Implementation Issues

SpW-D Prototyping, Measurements & Evaluation results

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:

- Channel based scheduling presents may advantages and can is easily extendable to support Multi-Transaction and Multi-Epoch Scheduling

Initiator issues:

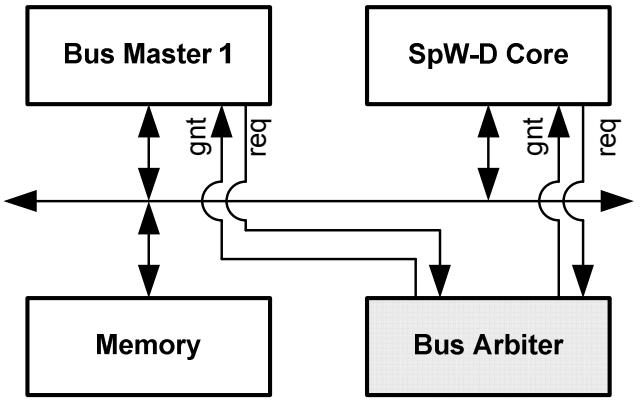
- In order to fulfill the SpW-D timing requirements, the initiator command controller should overlap command fetching from the memory and command transmission. This also ensures a constant "start of transmission"

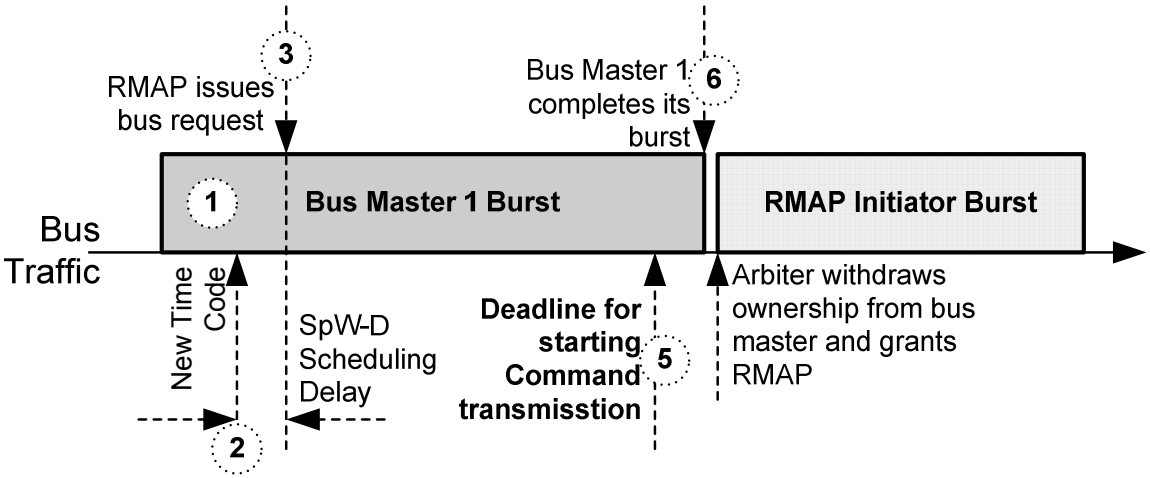
Extensions of an existing RMAP Core for SpW-D operation:

- 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

System on Chip architecture considerations (1/3)

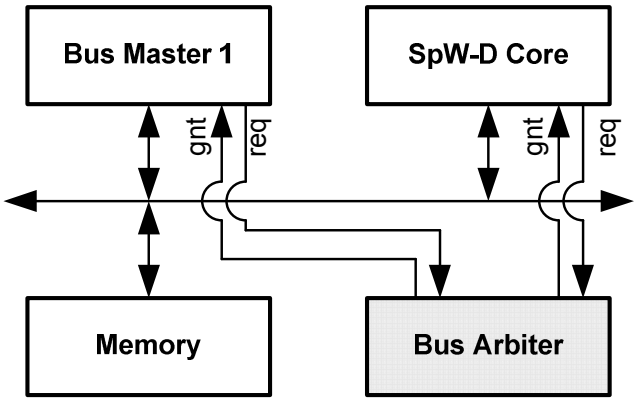
- Bus Master 1 is the system bus owner and performs a transfer to/from the memory.
- Time-Code is received while a long burst is in progress.
- The SpW-D Core requests bus ownership in order to initiate a transaction, but
- IF the bus arbiter does not support interleaved bursts:
- The SpW-D Core does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

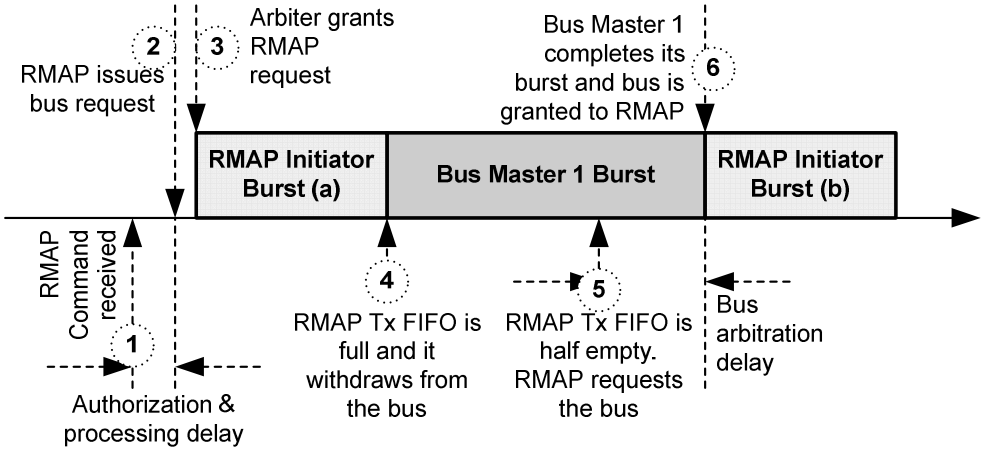




System on Chip architecture considerations (2/3)

- RMAP Read for long payload transfer is received
- Payload is fetched and transmission of first segment starts on time
- Tx FIFO is full and the Core backs off from requesting the bus
- Master 1 starts a burst
- Core Tx FIFO is half empty and requests the bus to fetch the next payload segment
- IF the bus arbiter does not support interleaved bursts:
- Gaps will occur in the transmission of the Reply and the Time Slot boundaries may be violated





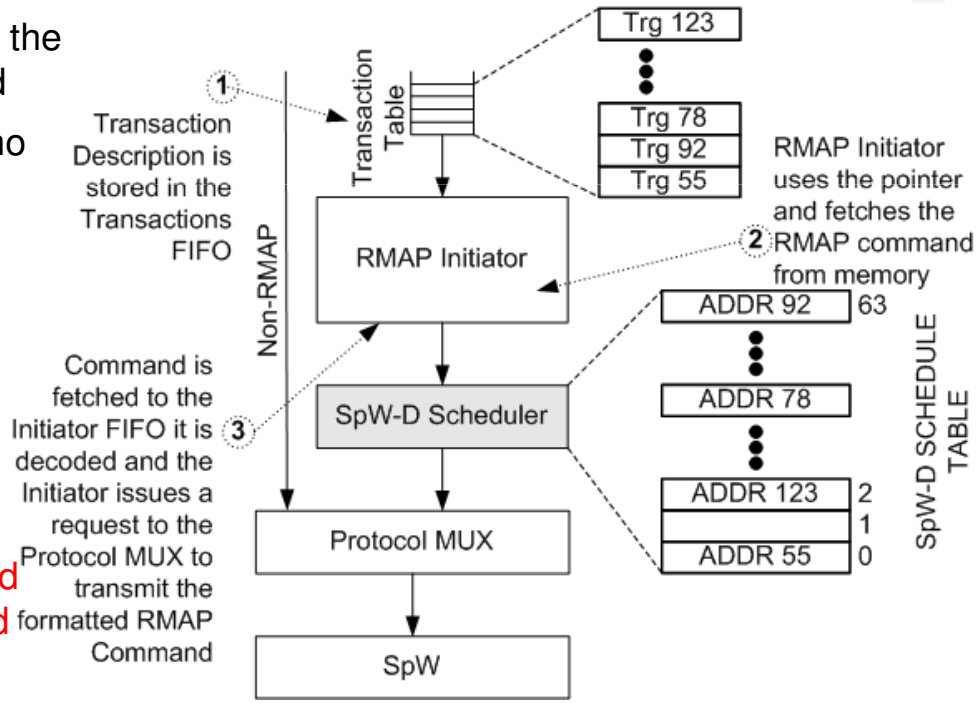
System on Chip architecture considerations (3/3)

CONCLUSION 1:

- 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 (not suitable with designs with multiple Cores)
  - 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 continuous bus ownership
- The bus masters shall support interleaved bursts
- The bus slaves 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(1/2)

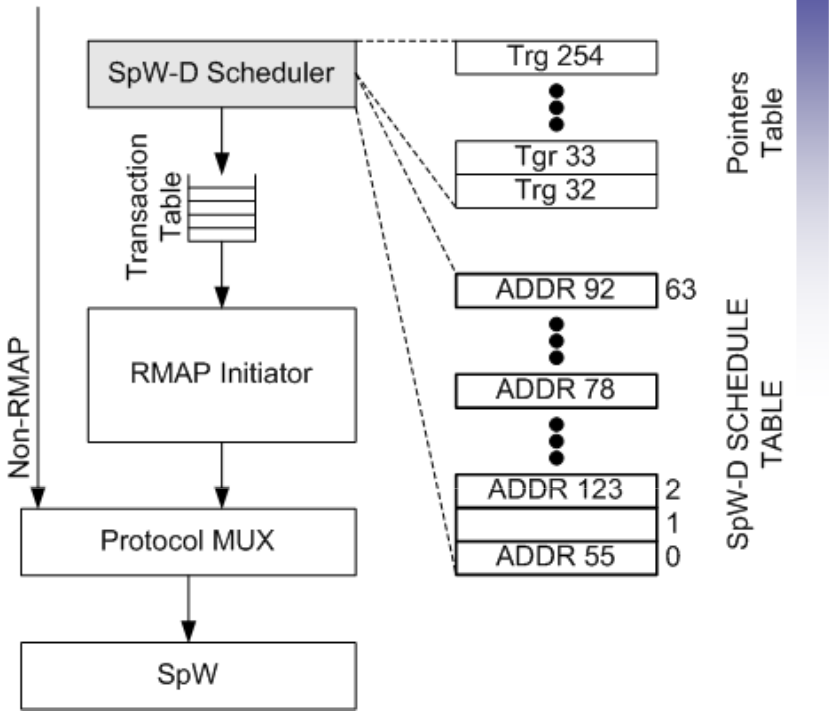
- The RMAP Core contains a transaction table where the host stores the memory addresses of the RMAP commands for transmission
- The RMAP Initiator fetches the pointer, requests access to the memory and fetches the command structure for transmission
- After the command is fetched the initiator requests 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(2/2)

Alternatives:

- 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
- Separate Tx Buffers per destination. **EXPENSIVE**
- 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

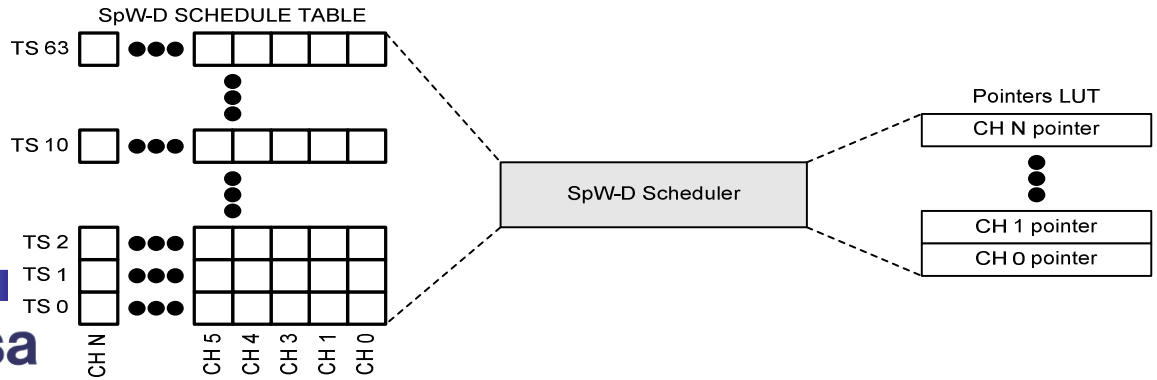


CONCLUSION 2:

Respecting the SOIS Stack may be OK for a SW implementation of SpW-D but does not result in optimum HW implementation.

SpW-D Scheduler considerations

- "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
- "**Channels**" 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

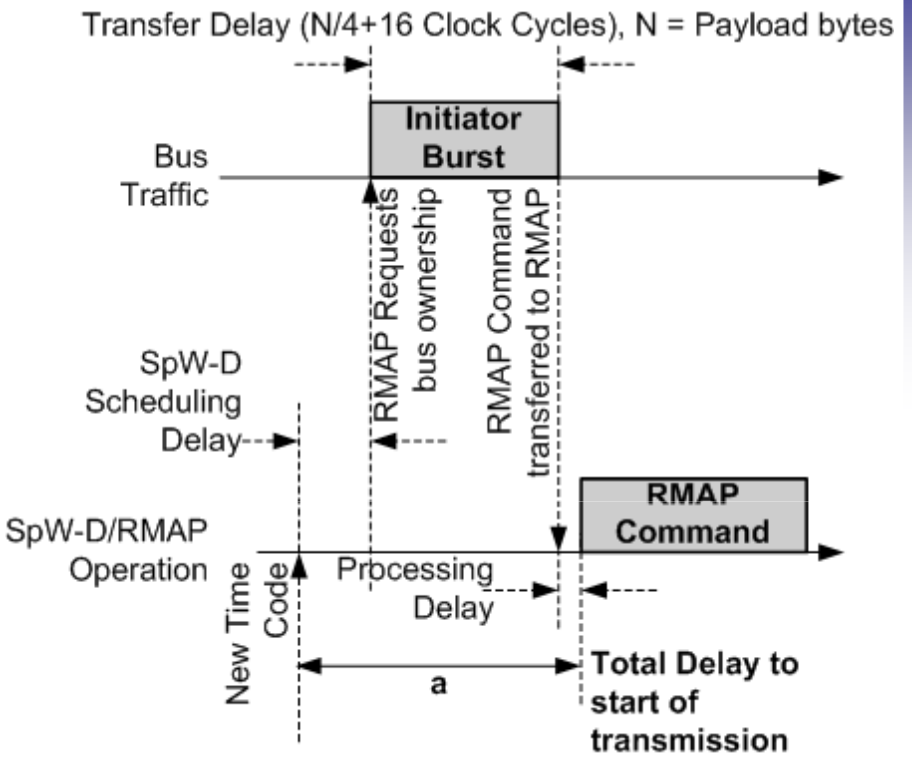


CONCLUSION 3:

Channel based scheduling supports existing and proposed scheduling alternatives and makes efficient use of memory resources.

RMAP Initiator Controller considerations (1/2)

- 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
- Overlapping 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



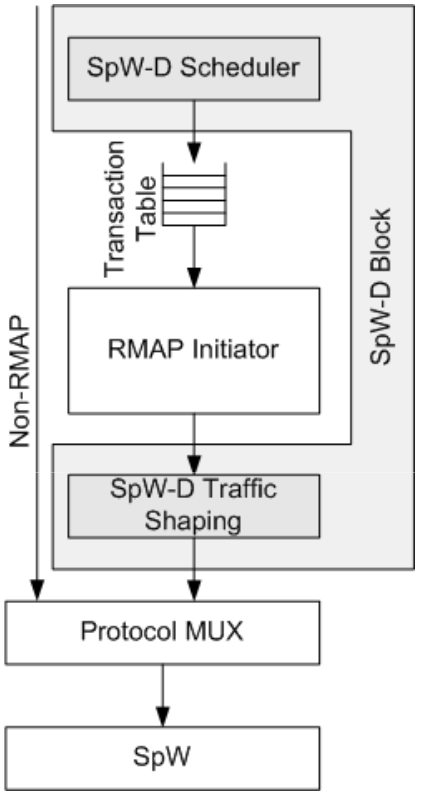
CONCLUSION 4:

The RMAP Initiator should overlap fetching of command from memory and transmission.

RMAP Initiator Controller considerations (2/2)

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



CONCLUSION 5:

Time out block should support two modes; RMAP mode (time, clock cycles etc.) and SpW-D mode (Time Codes).

PRESENTATION AGENDA:

SpW-D Evaluation, simulations on realistic architectures

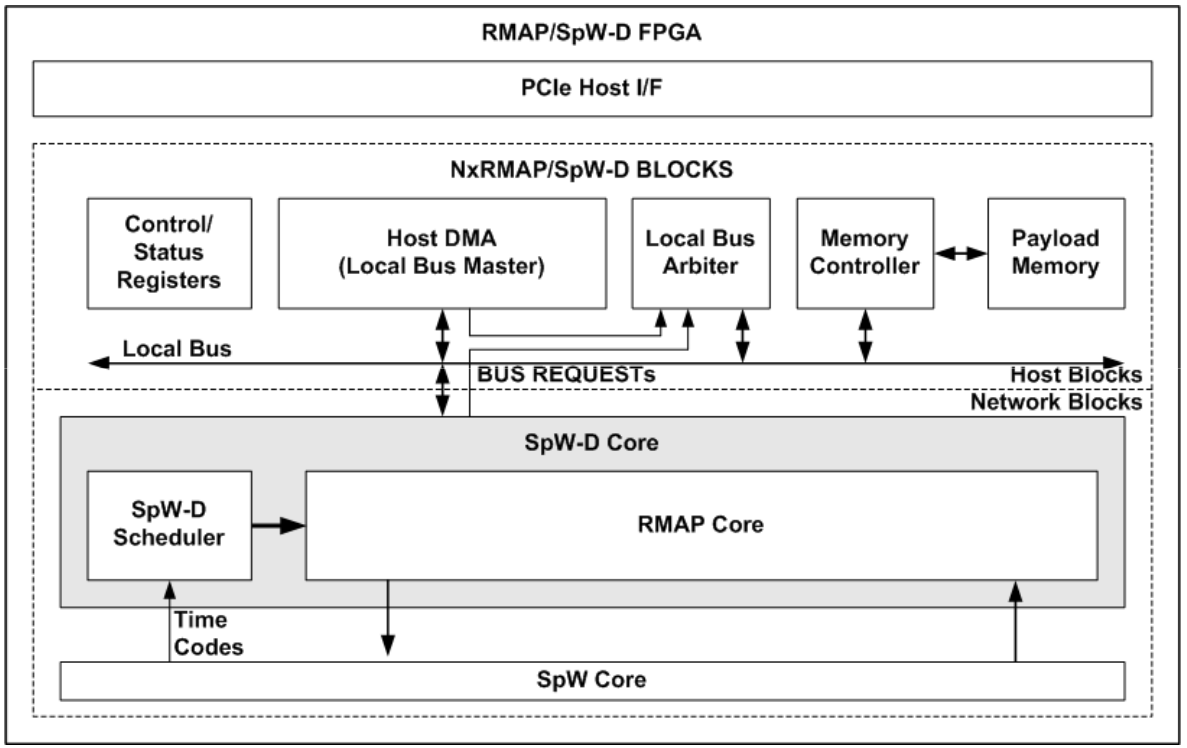
Proposed SpW-D Evolutions

SpW-D Prototyping, Hardware Implementation Issues

SpW-D Prototyping, Measurements & Evaluation results

SpW-D FPGA Architecture

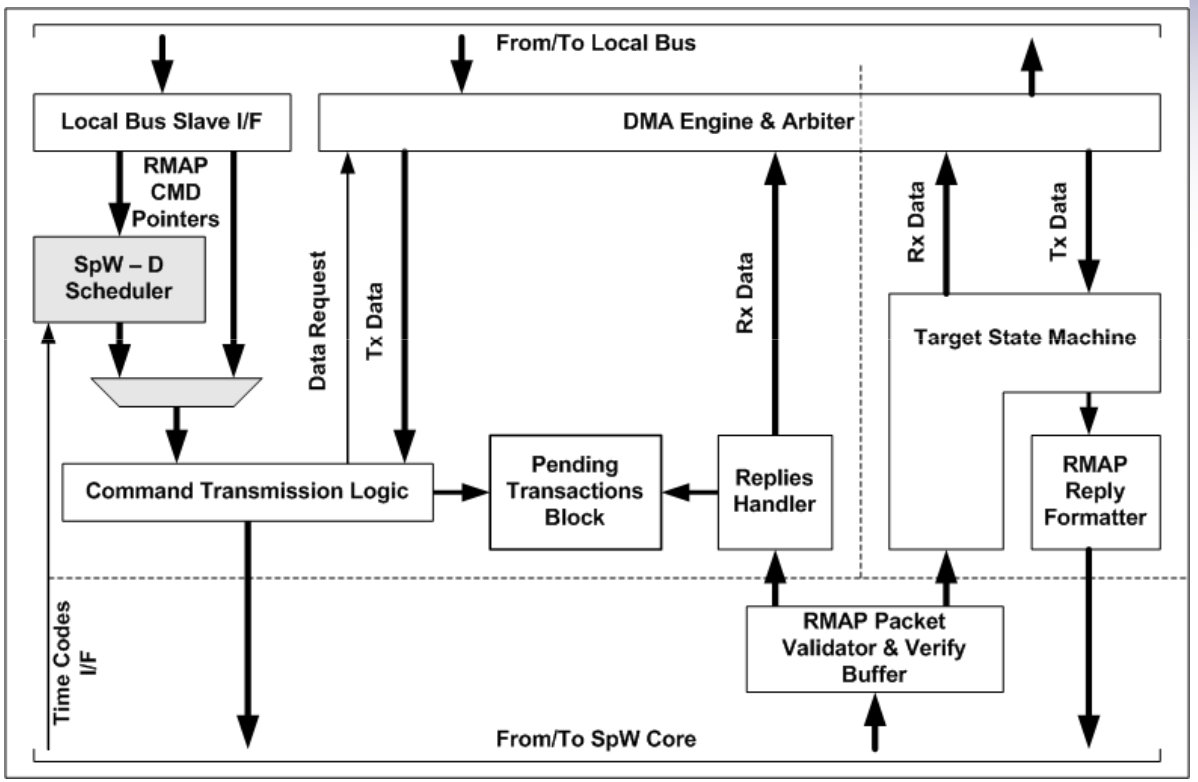
- 4-Lanes PCI Express Host I/F
- DMA Between Host (PC) and on-board memory
- RMAP/SpW-D/SpW Control/Status registers & Statistics
- Local Bus architecture
- Configurable number of RMAP/SpW-D Blocks (up to 4)
- TELETEL RMAP/SpW-D Core
- UoD SpW Core
- 16 KBytes local bus memories
- 125 MHz System clock
- TimeCode generation period 1us to 1s in 1us increments
- Programmable SpW Link Speeds
- Link Speed tested up to 200 Mbps SpW Link Speed



RMAP/SpW-D Core Architecture (1/2)

RMAP Features

- RMAP/SpW-D operation
- Initiator-only/Target-only versions
- READ/WRITE/RMW RMAP Transactions
- RMAP Error handling
- Configurable number of pending RMAP transactions
- Programmable (in time or Time Slots) timeout scan frequency
- Configurable Verfiy Buffer capacity
- Configurable Tx FIFOs capacity
- Registers /Tables relocatable within the system address space
- Configurable Endianess



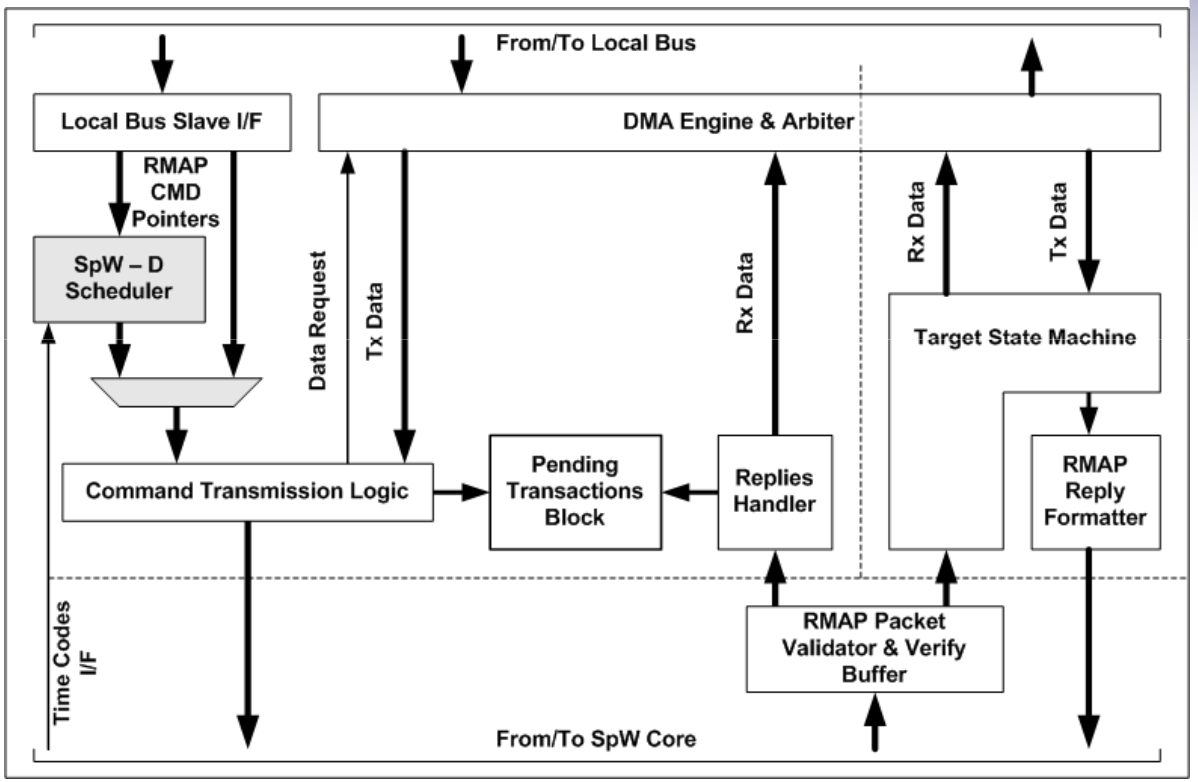
RMAP/SpW-D Core Architecture (2/2)

DMA Features:

- DMA Arbiter with prioritized requests
- Atomic transactions support
- Programmable maximum DMA length per requestor
- Splitted transactions for bursts crossing 1KByte boundary (AMBA restriction)

SpW-D Features:

- Configurable number of supported SpW-D channels
- Channels can be individually configured as “repetitive” for periodic sampling
- Priorities support
- SpW-D Initiator/Target timings emulation



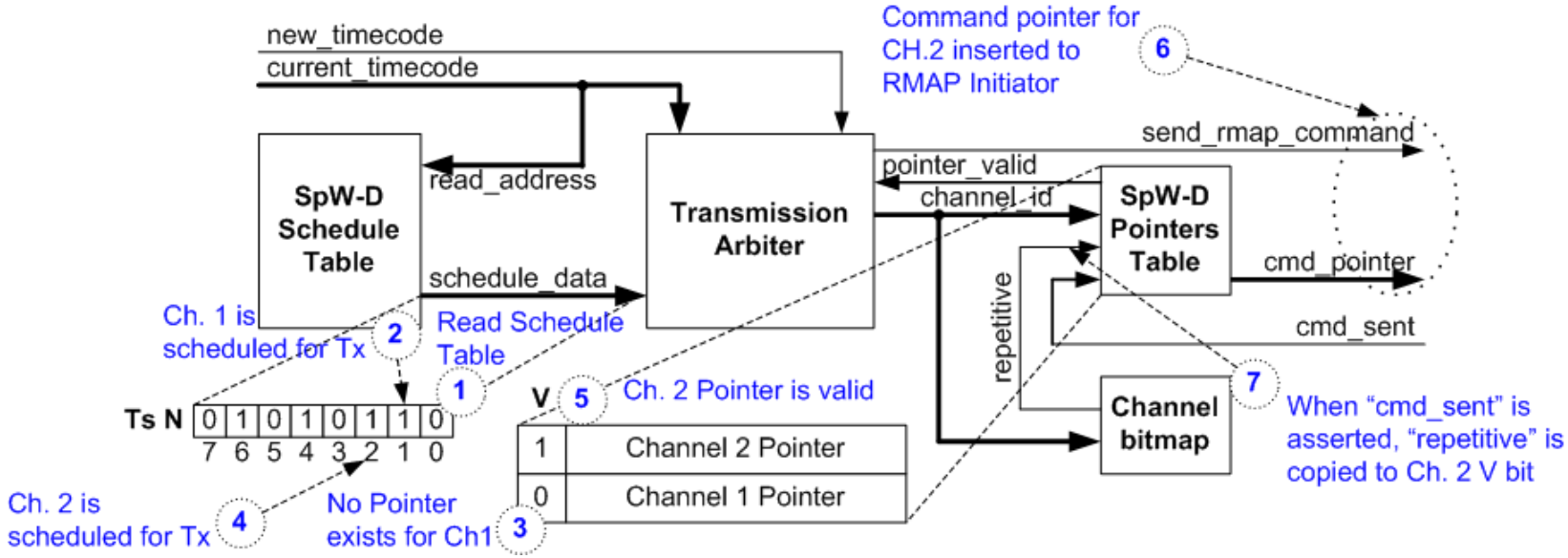
SpW-D Scheduler Architecture

√ "Channel" based scheduler

- Single bit per "channel" in the Schedule Table
- Pointers Table associates the schedule table with addresses of the commands in memory
- "Repetitive" attribute allows periodic Command transmission (sensors polling etc.)
- Supports flows of different priorities, to the same target
- It supports multi-epoch and multi-transaction scheduling
- Supports interleaved transactions to the same Target in the same Time Slot

Functionality:

- When a new Time Code is signaled the respective Schedule Table entry is used for arbitration
- The first channel scheduled for transmission with a valid pointer transmits
- After transmission the pointer is invalidated if the channel is not set in "repetitive" mode



Error-free RMAP functionality Validation

Initiator Functionality Validation with SpW-RTC:

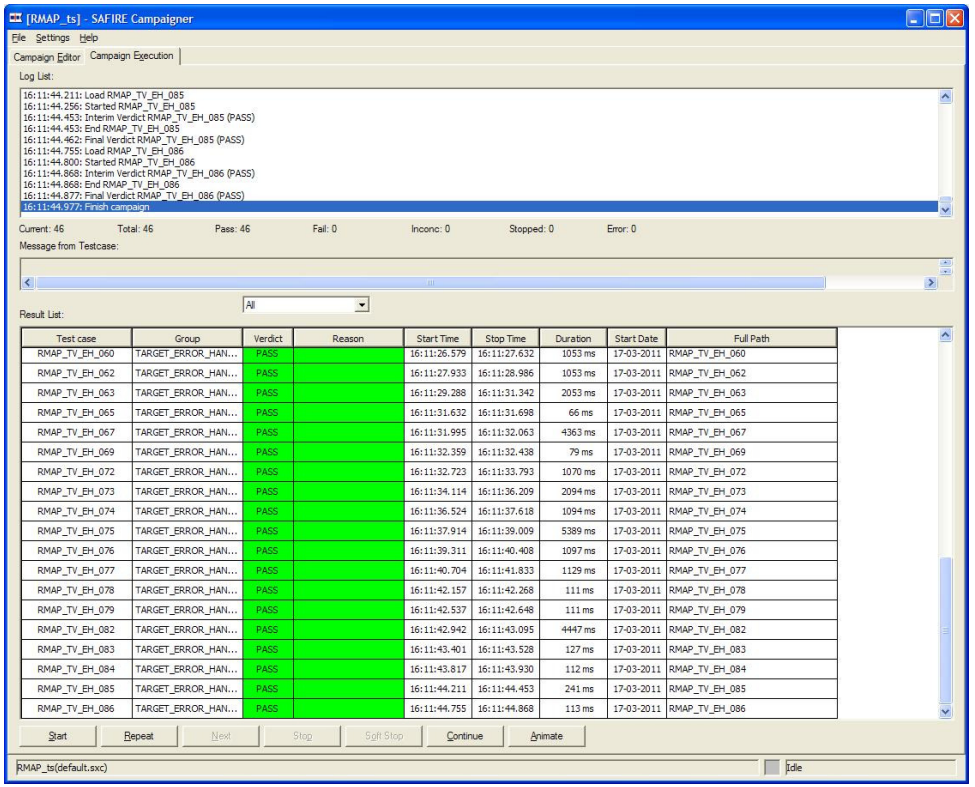
- Validation of Read/Write Commands
- Validation of Verified Write Transactions
- Validation of Incremental Address Transactions

Target Functionality Validation with USB-Brick:

- Validation of Read/Write Commands
- Validation of Verified Write Transactions
- Validation of Incremental Address Transactions

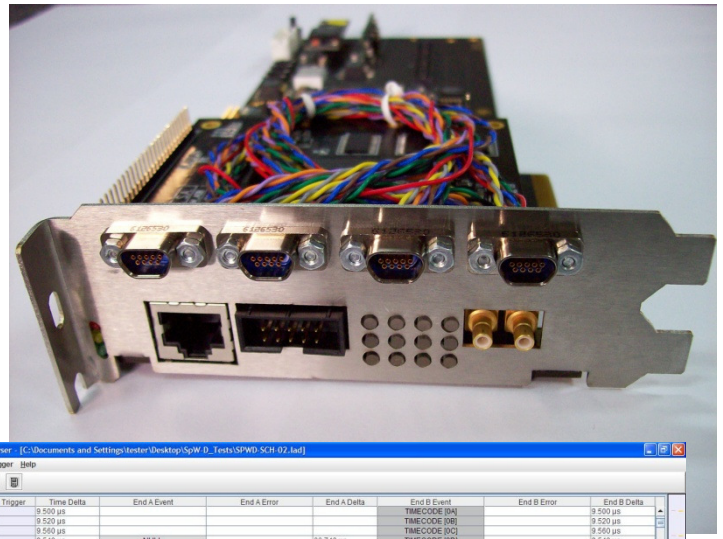
RMAP error handling Validation

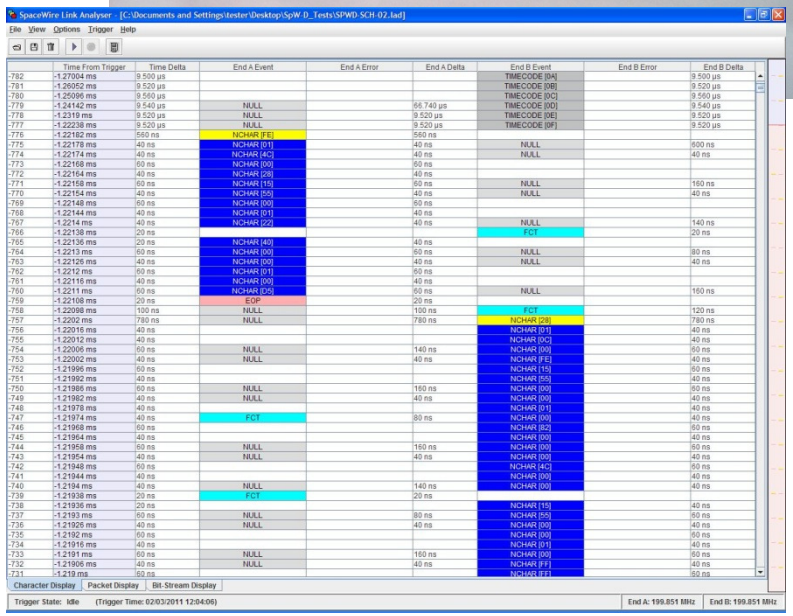
- Development of a behavioral VHDL testbench for error handling verification
  - Target & Initiator tests
  - Various command options (Incrementing/non-incrementing address, Acknowledged/non-acknowledged commands)
- Development of Target Error Handling Validation tests on SAFIRE
  - Header Errors (incomplete HDR, HDR CRC error, Invalid Key, EEP after HDR etc.)
  - Payload errors (Verify Buffer overrun, Payload CRC error, Early EoP etc.)
- Development and execution of Initiator Error Handling Validation tests on SAFIRE
  - Corrupted Replies
  - Reply timeouts



SpW-D validation

- Prototype FPGA:
  - Two SpW-D Cores
  - Eight SpW-D channels per Core
  - Repetitive transmission mode selectable per channel
  - SpW Link speeds 2 – 200 Mbps
- Functional testing against the SpW-RTC
- Stress tests by connecting two SpW-D ports to each other
- SpW links were probed with the Star Dundee Link Analyzer in all tests
- Functional tests:
  - Simple Scheduling, Concurrent, Multi-Slot Scheduling tests
  - Tests with emulated Initiator/Target delays.
- Tests included SpW-D errors:
  - Early Time Codes
  - Late Time Codes





Time Code Issues

- Link speed affects TimeCode propagation
- The TimeCode master marks the TimeSlot t0
- The TimeCode slaves perceive the start of TimeSlot the time instance t0+tp

Link Speed (Mbps)	First character at target (us)	NCHAR transmission Delay (us)	Difference (us)
200	0.6	0.05	0.55
100	0.84	0.1	0.74
50	1.22	0.2	1.02
33	1.6	0.303	1.296
25	2.05	0.4	1.65
20	2.36	0.5	1.86
10	4.3	1	3.3
2	20.5	5	15.5

Initiators connected to low speed Rx links have a different "perception" of the Time Slot boundaries than Initiators connected to high speed links. Problem is worsened if an Initiator is the Time Code Master

CONCLUSION 6: Uniform or near-uniform Link Speeds shall be used throughout the entire network.

Target Reply Latency (1/2)

- Payload size **256 bytes** for all commands
- Read Reply is slightly slower than the Write Reply since data shall be fetched from the System Memory
- Verified Write Reply begins later than the Non-Verified Write Reply since reception and transfer to memory cannot overlap. The payload shall be verified before the transfer in the Verified case.

Link Speed (Mbps)	Read Reply (us)	Non Verified Write Reply (us)	Verified Write Reply (us)
200	0.92	0.84	1.2
100	1.1	1	1.44
50	1.5	1.44	1.9
25	2.18	2.25	2.79
20	2.54	2.6	3.36
10	4.2	5.3	5.3
2	20.6	23	23

<u>CONCLUSION 7:</u> Target Reply latency is not significantly affected by the Command Type when there is no Authorization Delay.

Target Reply Latency (2/2)

- Link Speed is **200 Mbps** for all measurements
- Verified Write Reply begins later than the Non-Verified Write Reply since reception and transfer to memory cannot overlap. The payload shall be verified before the transfer in the Verified case.

Packet Size	Non Verified Write Reply (us)	Verified Write Reply (us)
12	0.62	0.62
32	0.64	0.64
64	0.72	0.72
128	0.84	0.84
256	0.84	1.22
512	0.84	1.96
1024	0.84	3.44
2048	0.84	6.44
4096	0.84	12.42

CONCLUSION 8: Target Reply latency may be excessively long for long Verified Writes. Verified Write payload shall be bounded.

Authorization Delay Impact (1/2)

- Read Command with **256 bytes** payload
- Authorization delay set to 5 us
- Authorization starts after the reception of the Authorization Key field

Link Speed (Mbps)	Reply Latency (us)	Replay Latency Without Authorization Delay (us)
200	5.12	0.92
100	4.8	1.1
50	4.1	1.5
25	2.86	2.18
20	2.6	2.54
10	4.2	4.2
2	21	20.6

Link Speed (Mbps)	Read Reply	Non Verified Write Reply	Verified Write Reply
200	5.12	0.84	1.22

<u>CONCLUSION 9:</u> Authorization Delay impact for Read is significant. It becomes more significant as the Link Speed Increases.

Authorization Delay Impact (2/2)

- Link Speed set at 200 Mbps
- Authorization delay set to 5 us
- Authorization starts after the reception of the Authorization Key field

Packet Size	Read Reply (us)	Non Verified Write Reply (us)	Verified Write Reply (us)
16	4.94	4.14	4.4
32	4.94	3.4	3.9
64	5	1.94	2.9
128	5.1	0.84	0.84
256	5.12	0.84	1.22
512	5.12	0.84	1.96
1024	5.12	0.84	3.44
2048	5.12	0.84	6.44
4096	5.12	0.84	12.42

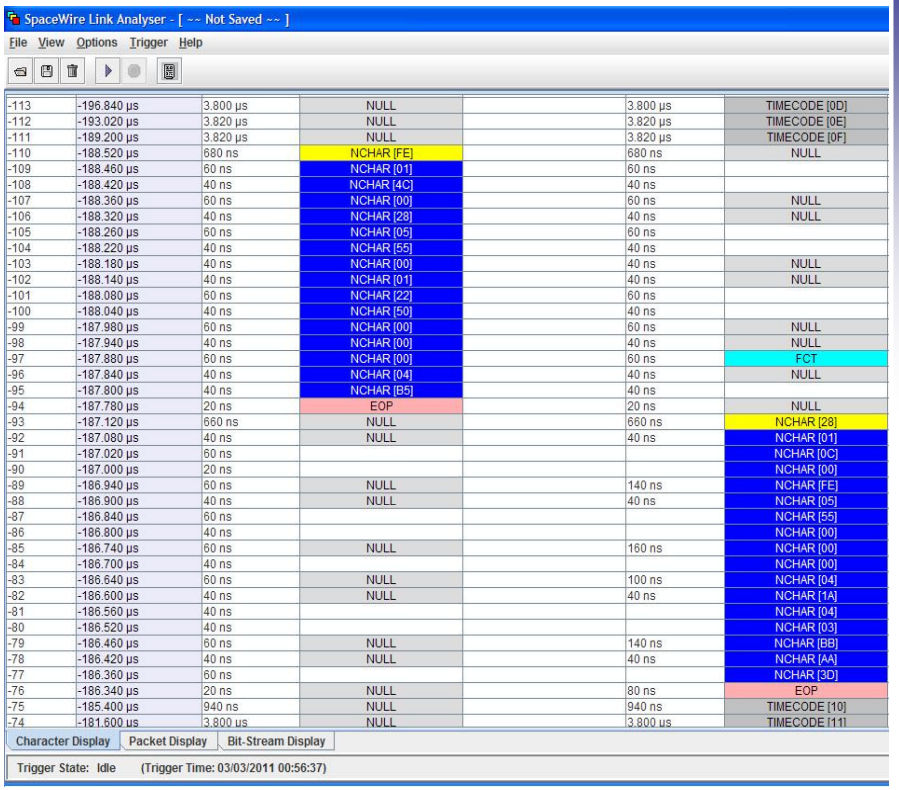
CONCLUSION 10: Authorization Delay impact for Writes is significant for small payloads only.

Evaluation of Draft spec Timings

- Link Speed set at 200 Mbps
- No routers in the paths
- No Start of transmission delay at the Initiator
- No Authorization/Reply Delays at the Target
- No other traffic on Initiator and Target local buses
- 125 MHz System Clock

RESULTS:

- Start of transmission for Channel 0 is 360 ns (when the Initiator generates Time Codes) or (when the Target generates Time Codes) 560 ns after the Time Code
- Start of transmission less than 100 ns later for Channel 7
- No gaps in transmission although transmission starts after header is fetched
- Target Reply latency is less than 1 us
- Transaction with 256 Bytes payload performed in less than 16 us Time Slot
- Verified Write transactions with 4096 Bytes payload in 244 us Time Slot with 34 us unused time
- RMAP Read with 4 Bytes payload in 2.85 us Time Slots



-113	-196.840 µs	3.800 µs	NULL		3.800 µs	TIMECODE [0D]
-112	-193.020 µs	3.820 µs	NULL		3.820 µs	TIMECODE [0E]
-111	-189.200 µs	3.820 µs	NULL		3.820 µs	TIMECODE [0F]
-110	-188.520 µs	680 ns	NCHAR [FE]		680 ns	NULL
-109	-188.460 µs	60 ns	NCHAR [01]		60 ns	
-108	-188.420 µs	40 ns	NCHAR [4C]		40 ns	
-107	-188.360 µs	60 ns	NCHAR [00]		60 ns	NULL
-106	-188.320 µs	40 ns	NCHAR [28]		40 ns	NULL
-105	-188.260 µs	60 ns	NCHAR [05]		60 ns	
-104	-188.220 µs	40 ns	NCHAR [55]		40 ns	
-103	-188.180 µs	40 ns	NCHAR [00]		40 ns	NULL
-102	-188.140 µs	40 ns	NCHAR [01]		40 ns	NULL
-101	-188.080 µs	60 ns	NCHAR [22]		60 ns	
-100	-188.040 µs	40 ns	NCHAR [50]		40 ns	
-99	-187.980 µs	60 ns	NCHAR [00]		60 ns	NULL
-98	-187.940 µs	40 ns	NCHAR [00]		40 ns	NULL
-97	-187.880 µs	60 ns	NCHAR [00]		60 ns	FCT
-96	-187.840 µs	40 ns	NCHAR [04]		40 ns	NULL
-95	-187.800 µs	40 ns	NCHAR [B5]		40 ns	
-94	-187.780 µs	20 ns	EOP		20 ns	NULL
-93	-187.120 µs	660 ns	NULL		660 ns	NCHAR [28]
-92	-187.080 µs	40 ns	NULL		40 ns	NCHAR [01]
-91	-187.020 µs	60 ns				NCHAR [0C]
-90	-187.000 µs	20 ns				NCHAR [00]
-89	-186.940 µs	60 ns	NULL		140 ns	NCHAR [FE]
-88	-186.900 µs	40 ns	NULL		40 ns	NCHAR [05]
-87	-186.840 µs	60 ns				NCHAR [55]
-86	-186.800 µs	40 ns				NCHAR [00]
-85	-186.740 µs	60 ns	NULL		160 ns	NCHAR [00]
-84	-186.700 µs	40 ns				NCHAR [00]
-83	-186.640 µs	60 ns	NULL		100 ns	NCHAR [04]
-82	-186.600 µs	40 ns	NULL		40 ns	NCHAR [1A]
-81	-186.560 µs	40 ns				NCHAR [04]
-80	-186.520 µs	40 ns				NCHAR [03]
-79	-186.460 µs	60 ns	NULL		140 ns	NCHAR [BB]
-78	-186.420 µs	40 ns	NULL		40 ns	NCHAR [AA]
-77	-186.360 µs	60 ns				NCHAR [3D]
-76	-186.340 µs	20 ns	NULL		80 ns	EOP
-75	-185.400 µs	940 ns	NULL		940 ns	TIMECODE [10]
-74	-181.600 µs	3.800 µs	NULL		3.800 µs	TIMECODE [11]

CONCLUSION 11: The timings specified in the draft spec. are realistic provided that the HW design considerations will be followed.

Implementation Metrics

Module	SpW-D Channels	Flip Flops	Schedule Table size	Channel Pointers Table size	Bitmap Table size
Transmission Arbiter	8	19			
	16	27			
	32	40			
SpW-D Scheduler	8	37	64x8	8x33 (8x31)	8x1
	16	45	64x16	16x33 (8x31)	16x1
	32	61	64x32	32x33 (8x31)	32x1

Module	Max. Verified Write Payload	Flip Flops	Verify Buffer Size
RMAP Packet validator	256	243	64x32
	512	247	128x32
	1024	250	256x32
	2048	252	512x32
	4096	256	1024x32

Memory requirements increase linearly (expected)

Less than linear increase for logic

SCALABLE SYSTEM

Conclusions (1/2)

Normative:

- Simple, Concurrent and Multi-slot scheduling cannot support scheduling in complex networks. Multi-transaction and Multi-Epoch "artificially increase" the number of available Time Slots and make scheduling possible in networks with complex topologies/traffic profiles
- Multi-epoch scheduling combined with CCSDS Time Distribution provides synchronized view of Major frames in the network
- CCSDS Time Master and Time Codes master should be a single device in a network
- Time slots with periods over 122 us present high efficiency > 70%
- Same Link Speeds shall be used network-wide to allow for unified perception of Time Slot boundaries
- KILL can increase the level determinism but does not ensure it (babbling idiots problem)
- The Initiators shall start transmission after a predefined time following the Time Code in order:
  - to provide a time window for KILL functionality
  - start transmission timely in case there is traffic on the system bus, or commands with different lengths are transmitted
- Non incremental addressing to be removed since it may cause paths blockage from the Initiator to the Router and makes error handling difficult
- Segmentation can be added as an optional layer above RMAP in order to support Large SDU transfer
- Verified Write payload length shall be bounded for implementation and protocol efficiency reasons

Conclusions (2/2)

Informative:

- Traffic sources should be synchronized with scheduling. This is imperative for isochronous applications
- Authorization delays become significant for Read commands and for Write commands with short lengths
- The Initiator shall store the Header and Payload of the Replies in different memory segments in order to support automated reassembly for SDU Reads
- SpW-D SoCs shall support interleaved bursts, bound the maximum DMA bursts and also bound the time of LOCKed system bus transactions
- Implementing scheduling above RMAP results in optimized HW implementations regarding the number of required memory resources
- Channels based scheduling:
  - Inherently supports priority function
  - facilitates scheduler implementation
  - efficiently supports multi-transaction and multi-epoch scheduling
- Initiator timeout functionality should be implemented in Absolute Time and Time Code modes for RMAP and SpW-D modes respectively


THANK YOU