Space
Technology
Centre
University of Dundee

HARDWARE IMPLEMENTATION OF AN RMAP NETWORK SCHEDULER

Albert Ferrer, Steve Parkes (UoD)
Alberto G. Villafranca, Martin Suess (ESTEC)

Outline

- Overview
  - – Motivation, RMAP scheduling, Time-Slot period
- Architecture
  - – Channels
    - Concept, Arbitration, Configuration,
  - – Error handling
    - Detection, Recovery, Report
- Results
  - – Validation & Performance
  - – Implementation cost
- Conclusions

Space
Technology
Centre
University of Dundee

Motivation & Requirements

Data-handling is likely to require

1. Guarantees
   - High **throughput** for Payload Data
   - Low **latency** for Command & Control operations.
2. Error detection
3. Remote R/W memory services
4. Robust implementation using existing qualified components
5. Low cost HW implementation
6. Simple to use and efficient

Scheduling packets with Time-Slots

- No network congestion
- Deterministic delivery
- Throughput allocation
- Guaranteed latency
- Simple with Time-Codes

RMAP protocol

- Standard R/W service
- Acknowledgments
- Data encapsulation

Design supports sporadic congestion

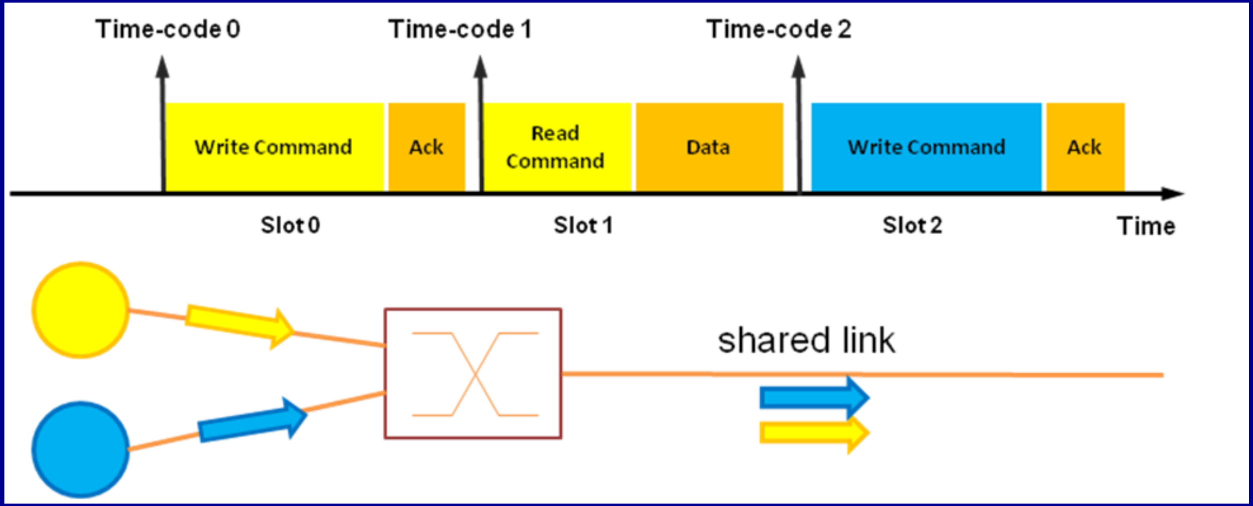
Use RMAP IP cores

Channels

- Segmentation
- Priorities

RMAP scheduling

- RMAP packets are sent at specific moments following a global synchronization using Time-Slots
  - Time-slots are equally spaced in time
  - Two transactions from different sources must not use the same network resources at the same Time-Slot



Time-Slot period

- Longer Time-Slots increase throughput
- Shorter Time-Slots decreases latency
- Trade-off considerations
  - A control packet should fit in a single transaction
  - Slot period should be a power of two division of one second
  - Low protocol overhead
    (delay, processing time, protocol header)
  - Values optimized for the highest speed (200Mbit/s)
- Possible value for a single transaction per slot
  - 61 µs Time-Slot period, 3.9 ms per epoch
  - 768 bytes RMAP data length

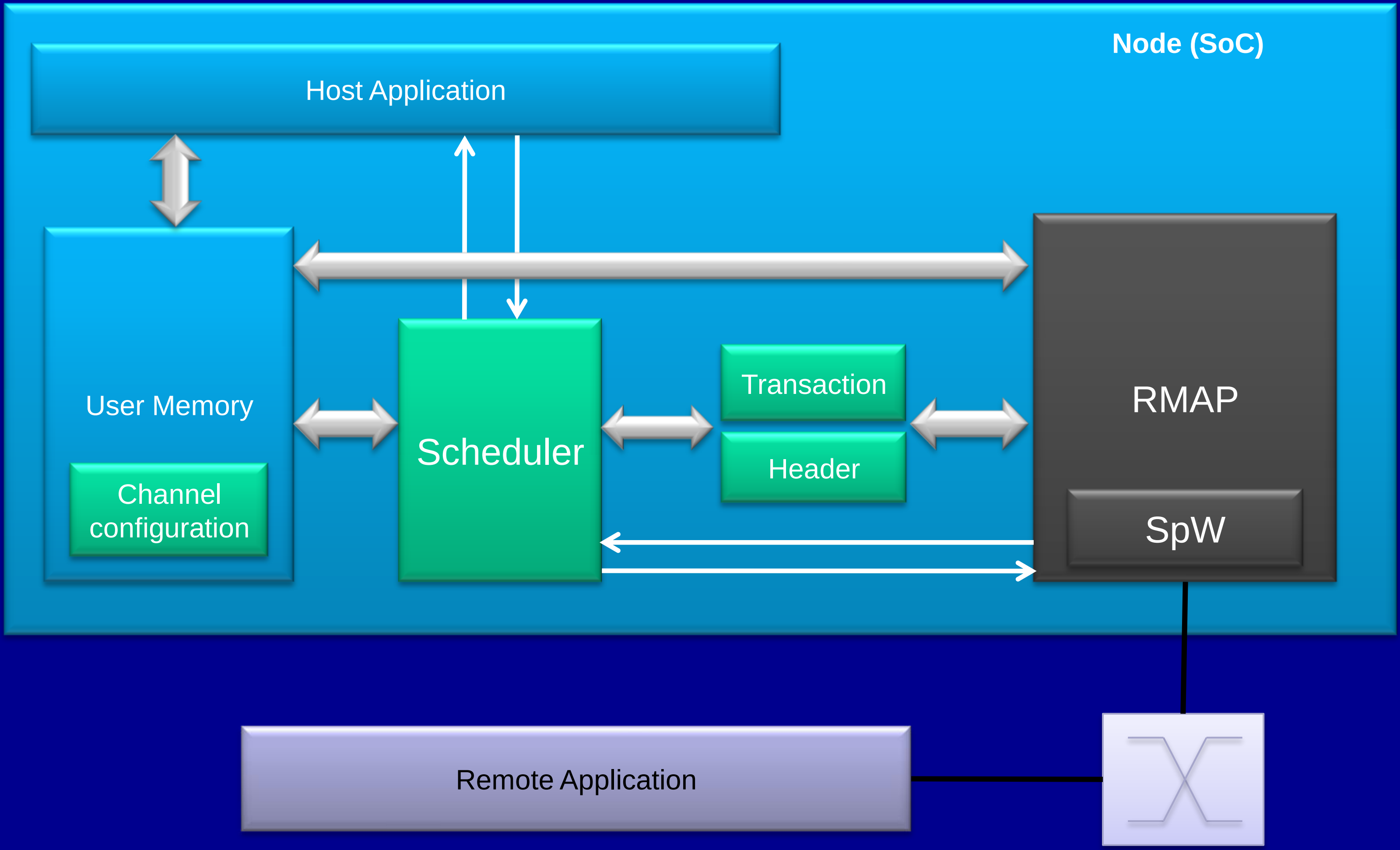
Number of transactions per Time-Slot

- Single RMAP transaction
  - Minimum latency
  - Accurate bandwidth allocation
  - Multi-slotting allows higher throughputs
  - Application do not care about slot allocation, only bandwidth and latency guarantees for each channel.
  - Short slots implies difficult software implementation
  - Inefficient in some command and control scenarios

- Multiple RMAP transactions
  - Multiple control messages can be sent in one slot
  - Increase the throughput
  - Efficient software implementation
  - Application timings can be synchronized with slots.
  - Increase the latency
  - More complex implementation

System Architecture

Space Technology Centre
University of Dundee



Channels

- The RMAP schedule is configured using channels

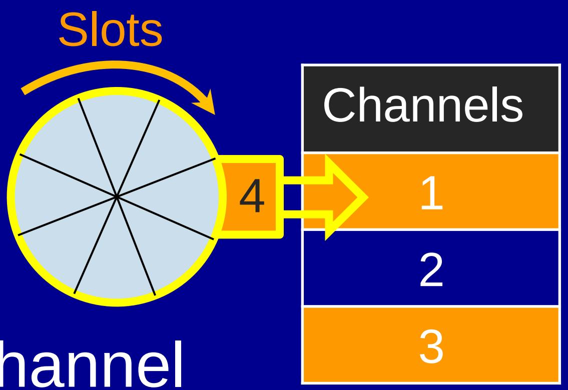
- Each RMAP user message is assigned to a different channel

- A channel provides
  - A segmentation layer
  - Sending status and error reporting
  - Two level arbitration: allocated slots and priority

Channels: Two level arbitration

- With simple scheduling, sporadic messages waste time-slots when they are not active.

- Channels use two level arbitration
  1. Time-Slot scheduling
  2. Channel priority



- Critical sporadic messages
  - Are assigned to a high priority channel
  - Use the same slots that have been already allocated to a long message using a lower priority channel.
  - When the control message must be sent it will be sent in the following allocated slot even if the lower priority channel have not sent all segments of the payload message

Channels: Example

Message	Channel	type	priority	Segments	slots	Path	Data ready
A	Ch0	Control	high	🟨	0,2,4	1,1	**No**
B	Ch1	Data	medium	🟪 🟪	0,2,4	1,2	Yes
C	Ch2	Data	low	🟦 🟦	1,3	2	Yes

Slot 0	Slot 1	Slot 2	Slot 3	Slot 4
□ 🟪	🟦	□ 🟪	🟦	□ 🟪
🟪	🟦	🟨	🟦	🟪

At this instant Host wants to send control message, sets data ready (Ch 0) = yes

Channels: Segmentation

- The targets needs to identify if a RMAP packet is a segment of a message, or if it is the start or the end segment.

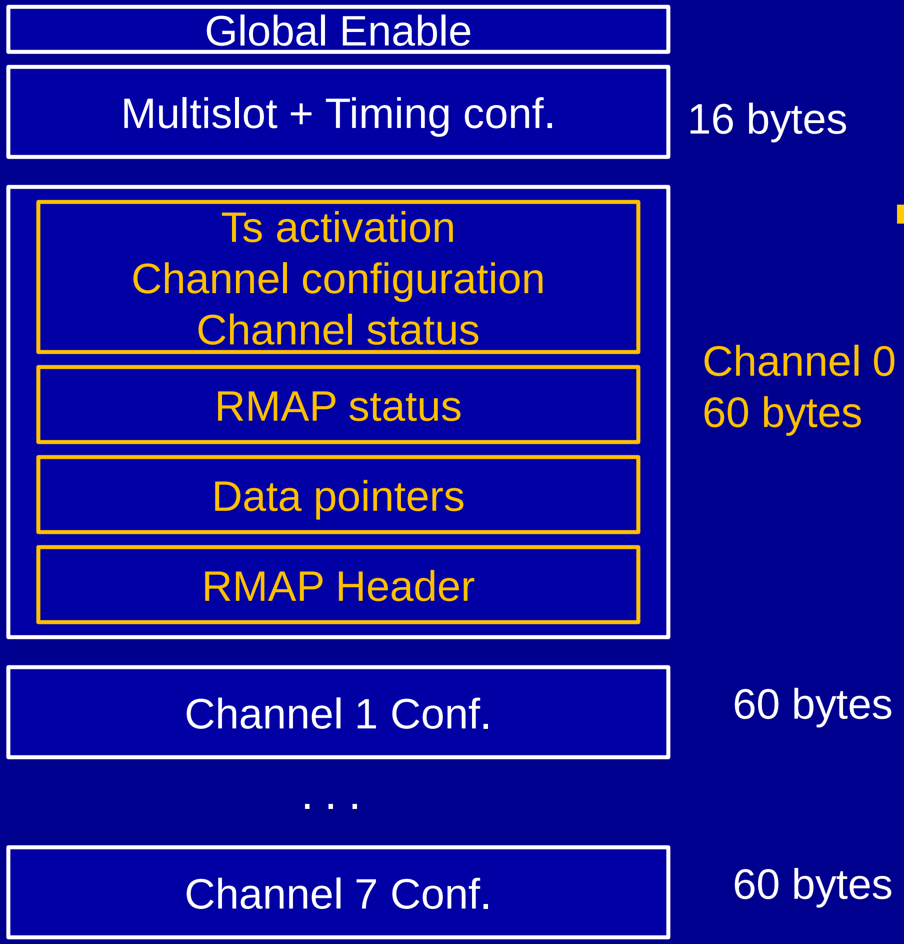
- Two bits required (using the Transaction ID field of RMAP packet)
  - First/start segment flag
  - Last/end segment flag

- When using one segment per message both bits are set
- When is a 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)

RMAP Transaction ID field

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

Channels: Configuration



Global Enable

Multislot + Timing conf.

16 bytes

Ts activation
Channel configuration
Channel status

RMAP status

Data pointers

RMAP Header

Channel 0
60 bytes

Channel 1 Conf.

60 bytes

. . .

Channel 7 Conf.

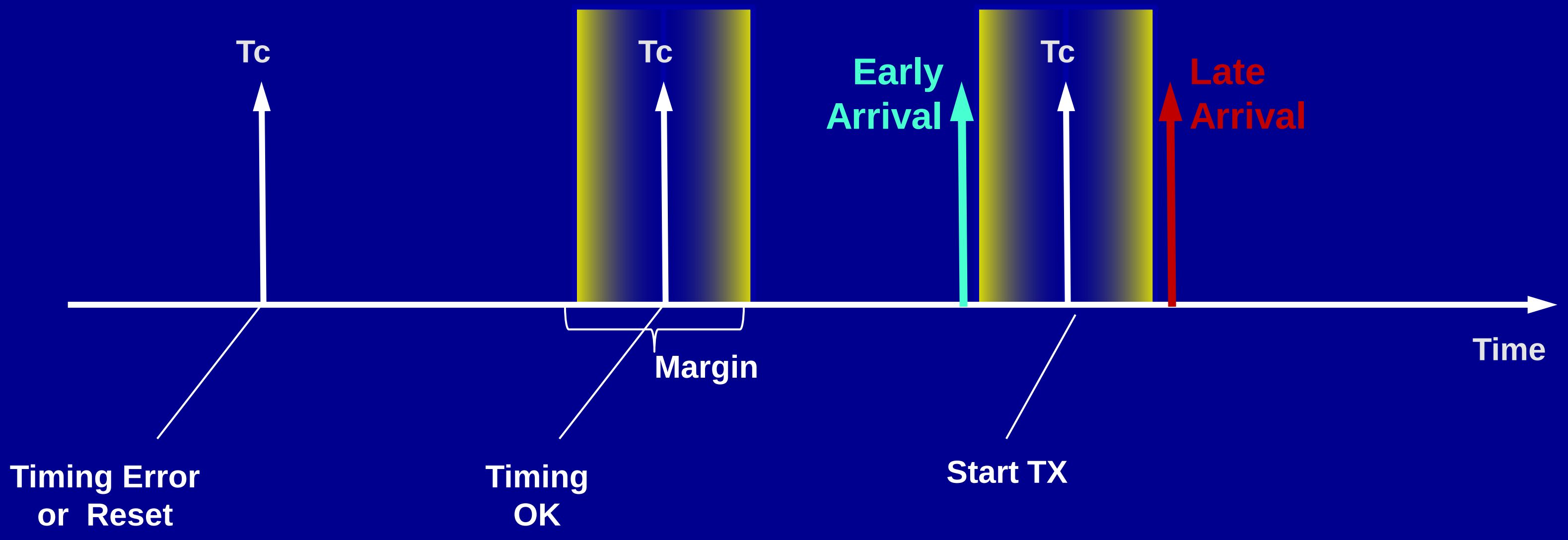
60 bytes

- A channel configuration contains
  - The RMAP header
  - The list of Time-Slots that the message can use
  - The message priority

All configuration can be programmed with a single RMAP packet

Error Handling: Time-Codes

- Time-Code error: set when a Time-Code is received too early or too late (or it has been lost)



Error Handling: Channels

- TX error: set when the RMAP command header is invalid or there is a internal bus error. Disables the corresponding channel.

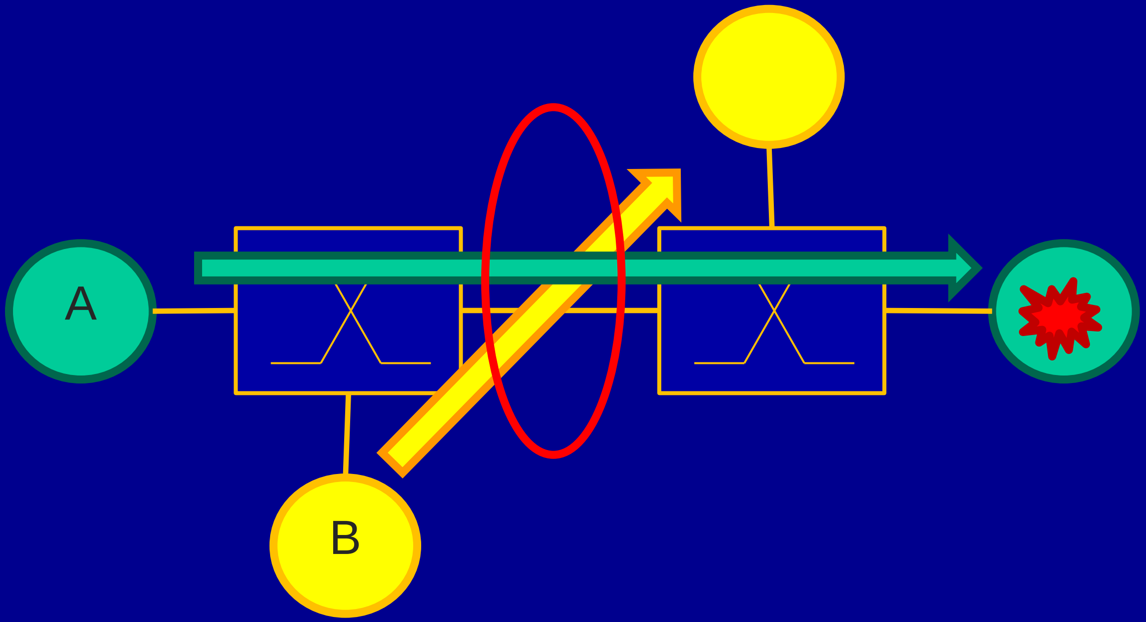
- TX congestion: set when a RMAP packet is still being send at the beginning of the next slot. Indicates there has been an error somewhere in the network. The following slot may be a guard slot in order to remove the network congestion.

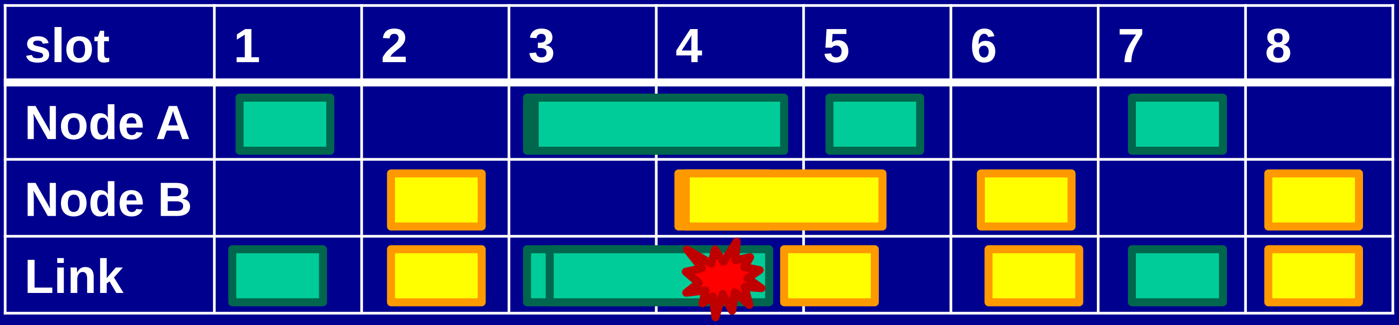
- RX error: set when the RMAP reply is not received or when it has been received with an error code. Disables the corresponding channel. If TX congestion is also set, it indicates that this channel has produced a network error.

- RX late reply: A reply has been received after the end of the slot but before the deadline set for this channel. Indicates there has been an error somewhere in the network.

Space Technology Centre
University of Dundee

Error Handling: Example





slot	1	2	3	4	5	6	7	8
Node A	🟩		🟩		🟩		🟩	
Node B		🟨		🟨		🟨		🟨
Link	🟩	🟨	🟩	✸ 🟨		🟨	🟩	🟨

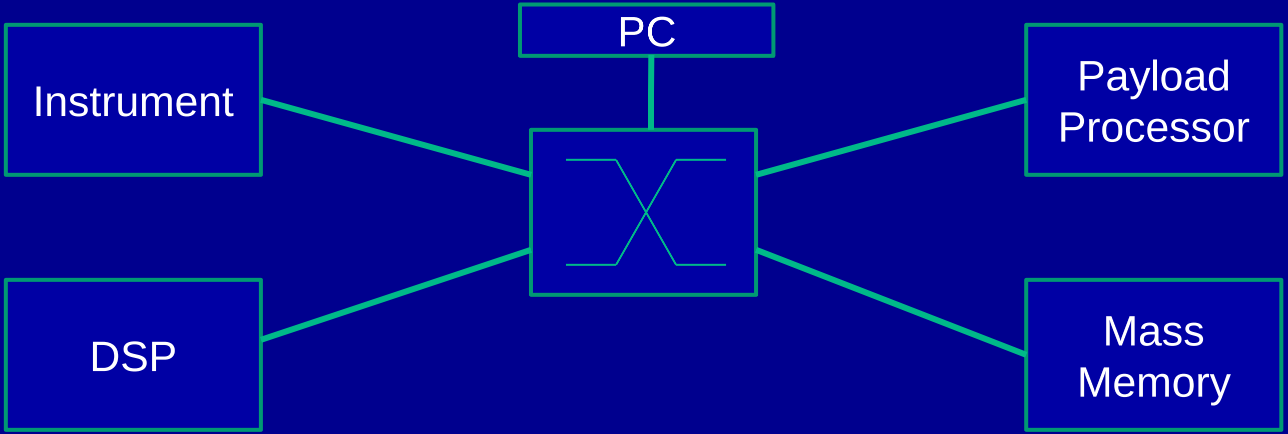
Error Handling: Recovery

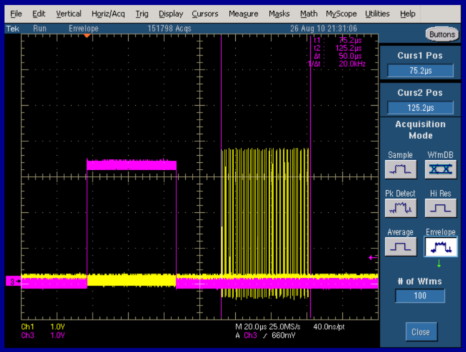
- Retry mechanism
  - – Retry is done in the next slot allocated to the same channel.
    - Retry is not performed unless it is indicated by the host or the network manager by clearing the error condition.
- Automatic enabling a channel when previous channel number got an error.
  - – Allows to set a 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.

Space Technology Centre
University of Dundee

Results: Validation & Performance

- Validated using Virtex II and IV under different scenarios, including
  - Multiple channels, High priority messages
  - Segmentation, Error handling





Tek Run Envelope 151798 Acqs 26 Aug 10 21:31:06

t1 : 75.2µs
t2 : 125.2µs
Δt : 50.0µs
1/Δt : 20.0kHz

Curs1 Pos 75.2µs

Curs2 Pos 125.2µs

Acquisition Mode

# of Wfms 100

Ch1 1.0V
Ch3 1.0V

M 20.0µs 25.0MS/s 40.0ns/pt
A Ch3 ⁄ 660mV

TC arrival → 300 ns → RMAP Triggered → 2.7 µsec → 1st Byte sent

Results: Cost

- The scheduler roughly requires less than a quarter of the RMAP IP core resources.

Virtex IV (LX100)

Logic	RMAP IP core	RMAP Scheduler
Slice Flip-Flops	3002 3%	3868 3%
4 input LUTS	8722 8%	10498 10%
Occupied Slices	5023 10%	6207 12%

Conclusions

- A hardware implementation of an RMAP Network Scheduler has been developed with channels that provide:
  - Segmentation
  - Error handling
  - Priorities
- The solution gives latency and throughput guarantees to SpaceWire networks.
- Designed for existing SpaceWire components
- Only requires 22% of an RMAP IP core.