MOST (Modeling of SpaceWire Traffic):
MOST v2.2 presentation

WE LOOK AFTER THE EARTH BEAT

07/01/2014

Ref.:

OPEN

ThalesAlenia Space
A Thales / Finmeccanica Company

83230918-DOC-TAS-EN-001

1

MOST short introduction

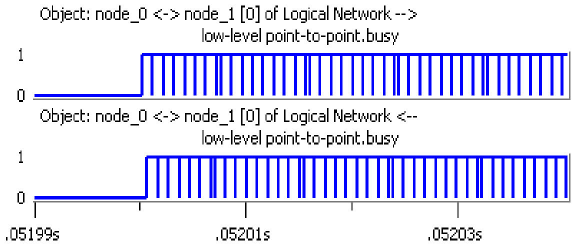
MOST short introduction (1/5)

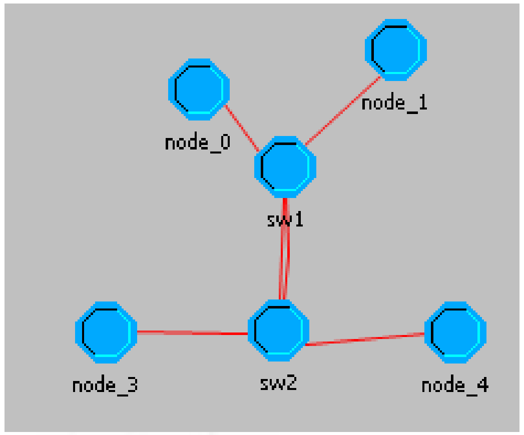
- Current steps of development:
  - Validated simulator delivery and installation in ESA facilities in Q4 2011
  - TAS: simulation activities & continuous development
  - ESA: simulation activities & development of specific needs
  - Now: merging of both developments, adding new features, optimize the software & new simulation activities to come
- A first release of MOST v2.2 has been presented in SpW WG20 (April 2013), this second release corresponds to the latest status of this development cycle. **Both releases are operational and have been used in TAS to conduct mission analysis**
  - **This new development running on OPNET Modeler ® 17.5 targets a release to the SpaceWire community in 2013/2014 with MOST v2.2**

- People currently involved in MOST:
  - ESA:
    - David Jameux
  - TAS-F Cannes:
    - Brice Dellandrea
    - Philippe Fourtier
    - Baptiste Gouin
- People involved in initial phase:
  - 4Links:
    - Barry Cook
    - Paul Walker
  - Scisys:
    - Peter Mendham
    - Stuart Fowell

MOST short introduction (2/5)

- MOST simulator is dedicated to the following users:
  - **System engineers** who have to design network topology and to perform validation tests
  - **Developers** who would need to test new component features or protocol





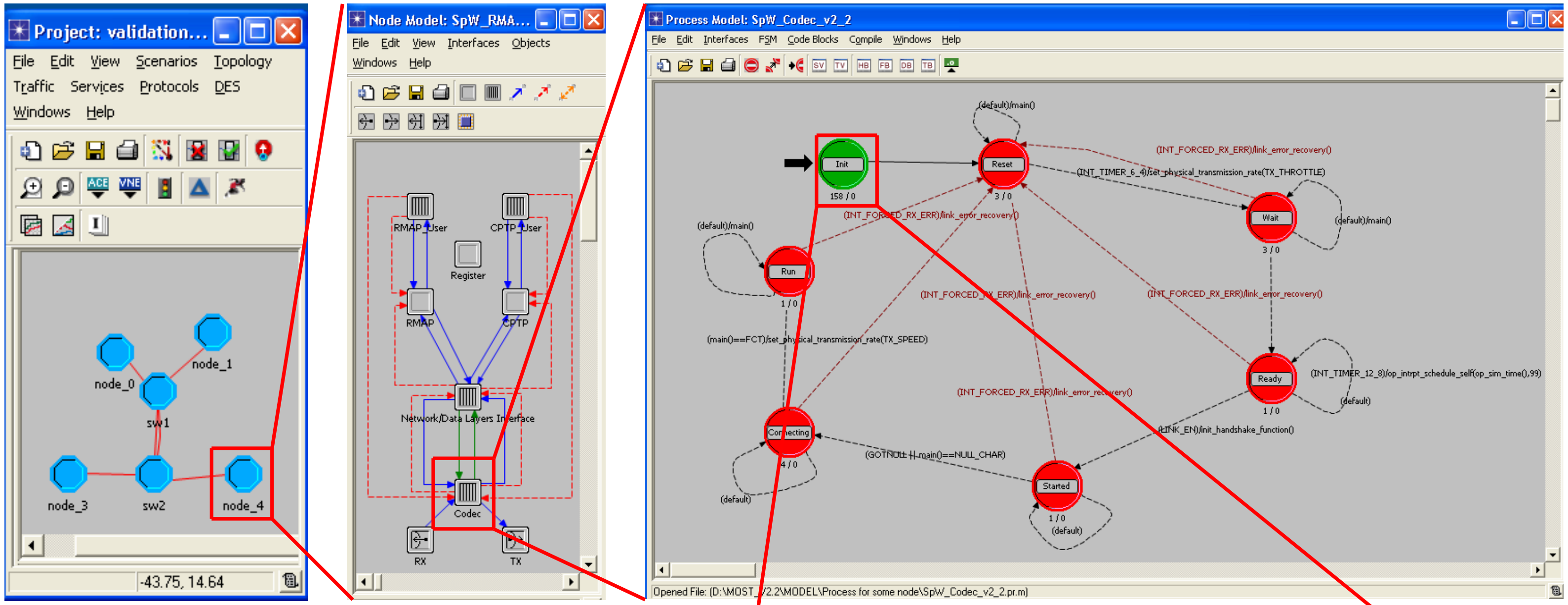
- MOST can be used during all phases of a project:
  - During early steps of projects, MOST mainly plays a role in the following design activities :
    - Phase A and before : performs evaluations, starting from a preliminary specification of network and nodes
    - Phase B : consolidate design by enhancing and completing nodes models behavior in terms of data provider and consumer
  - During development steps of a project, MOST participates to :
    - Phase C, D : design, validation and investigation
  - During maintenance step of a project, MOST takes part to :
    - Phase E : investigations, support to very specific operations

MOST short introduction (3/5)

- <u>Why do we need a model for SpaceWire traffic?</u>
  - The SpaceWire links are bidirectional,
  - The link speed is set by the transmitter, the receiver adapts! A SpW link can have different speeds per direction (ex.: Node_0 to Node_1 at 200MHz and Node_1 to Node_0 at 100MHz),
  - The flow control management is performed by the receivers through the sending of flow control tokens: this creates a dependency on the bidirectional link between sending direction & receiving direction,
  - SpaceWire can carry infinite data streams but preferably packets which might be of various sizes,
  - There are no “bus master” scheduling the communications: any endpoint can send data anytime to anyone on the network,
  - Switches perform wormhole routing which may lead to arbitration conflicts and congestion,
  - Transmission speed, latency and jitter are variable according to sources, destinations and paths (switches),
  - Each node has its own limited buffering for both emission and reception and useful data absorption capabilities,
  - SpaceWire FDIR implementation is not standard and vary from components to components.

→ There are many ways to use SpaceWire networks, managing a full network and size the reception & emission buffer, link speeds to avoid loss of data might be tricky !

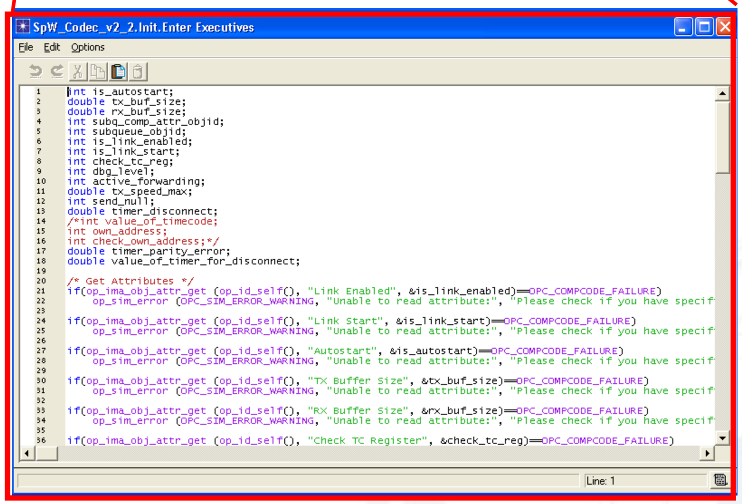
MOST short introduction (4/5)



Project or Network domain

Node domain

Process domain



OPNET Source domain

- MOST is built on 4 different domains:
  - Network domain (connexion between components),
  - Node domain (**building blocks**/layer decomposition),
  - Process domain (machine states)
  - C code in each automate's states

OPEN

ThalesAlenia Space
A Thales / Finmeccanica Company

MOST short introduction (5/5)

- What is the actual scope of MOST, is it covering all of the SpaceWire standards?
  - MOST is a traffic analysis tool, its implementation starts at the character formatting level:
    - No simulation of the physical interface! For instance, common modes are not simulated
    - Consequently, wrong character formatting, PLL de-synchronization, wrong electrical states, BER insertion are not simulated in the current version of MOST
    - These features can be emulated at MOST level by triggering an automatic disconnect or generating a transient BER alarm (simulation of the effect is implemented)
    - According to this logic, neither parity bits nor CRC are computed in the current version of MOST for simulation speed optimization (no simulation of the physical cause)

→ Starting from the Character level, MOST implements all the currently existing SpaceWire protocol stack up to the User applications managing packets exchange over SpaceWire

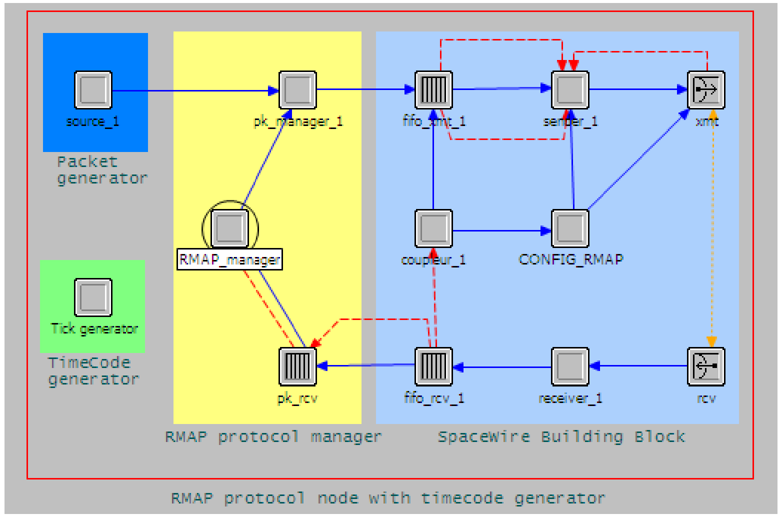
2

MOST v2.2 development targets

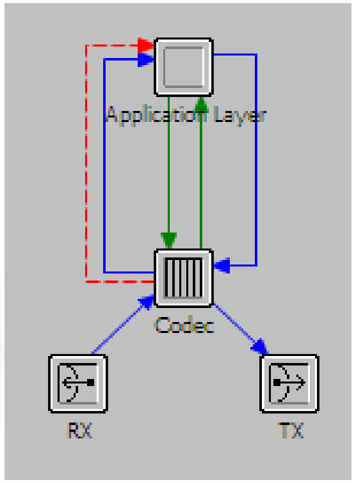
- ECSS-E-ST-50-12C: physical up to network layers
- ECSS-E-ST-50-51C: upper-layer protocols management
- ECSS-E-ST-50-52C: RMAP protocol handling
- ECSS-E-ST-50-53C: CPTP protocol handling
- Generic components
- Implementation of the advanced model of the 10X router

MOST v2.2: SpW layering simplification

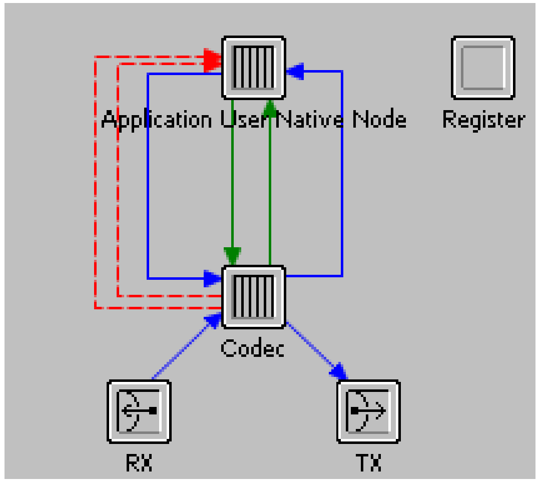
MOST v1.4 (Dec 2011):

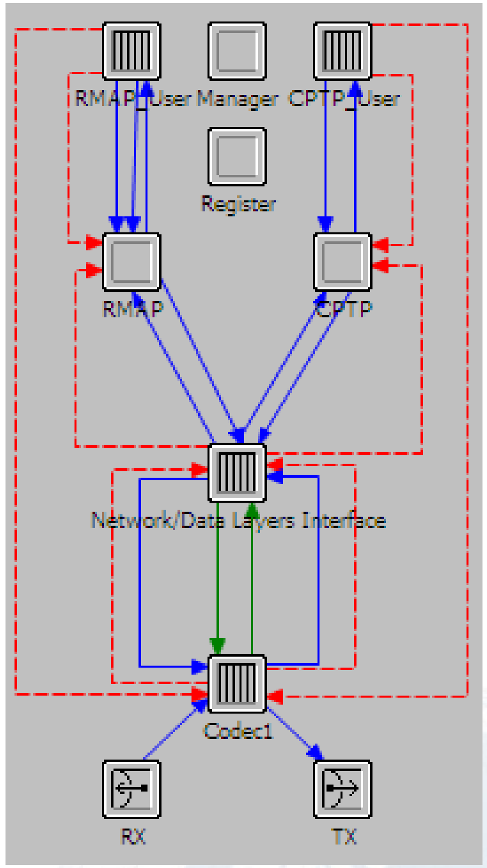


MOST v2.1 (Prototype/June 2012):



MOST v2.2 (Native Node and CPTP& RMAP Node):





Each layer provides an independent set of configurable parameters & statistics

OPEN

Ref.:

MOST v2.2: Performance & easiness of use

- MOST v1.4 (Q4 2011 version) was fairly complete:
  - Components:
    - Generic CPTP & RMAP Node
    - SPW 10X Router
    - SMCS116SPW & SMCS332SPW
    - Remote Terminal Controller
    - From MTG simulation: Generic Buffer coupling node
    - From MTG simulation: Generic Virtual Channel Multiplexer
  - Links: SpaceWire link
- MOST v2.2 (Q2 2013 & Q4 2013) implements:
  - Generic Native Node (1 CODEC + Application)
  - Generic CPTP & RMAP Node (1 CODEC + PID + CPTP + RMAP + Applications)
  - Generic 32-ports Switch (31 CODECs + PID + RMAP + Application)
  - An advanced model of the 10X router ***(new in release 2)***

→ ***Easier to use and adapt to user's needs and runs much faster***

3

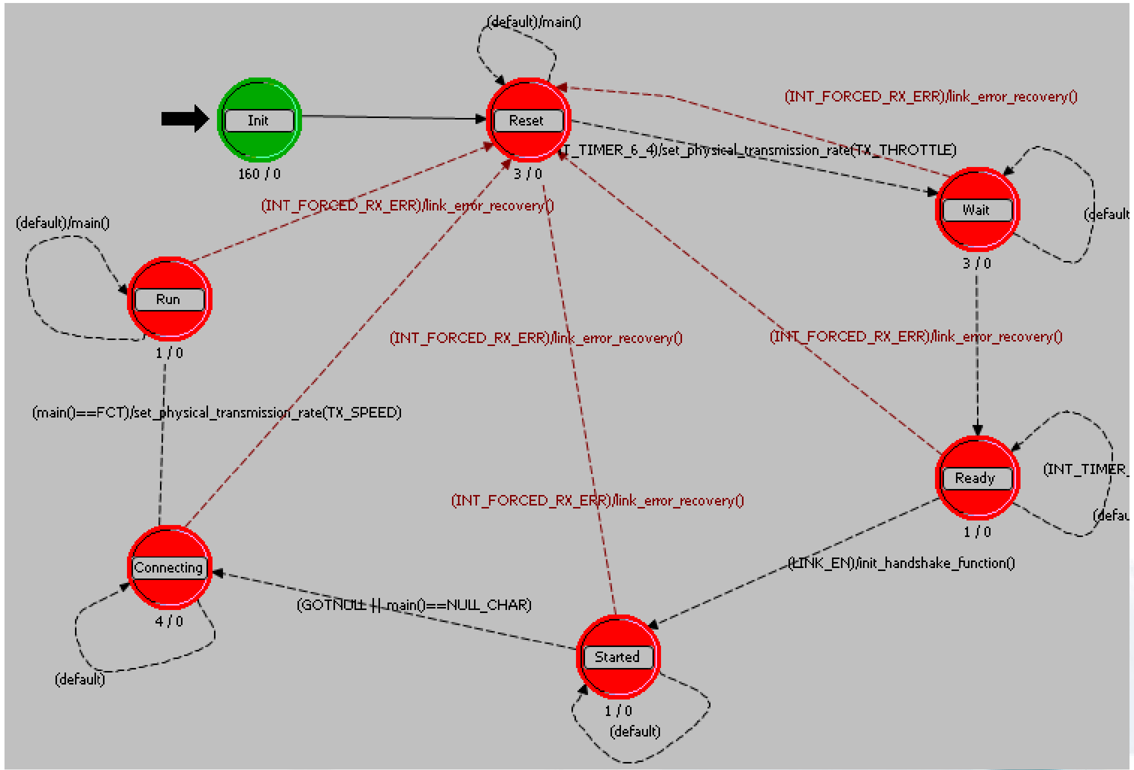
MOST v2.2 components:

- *All based on the same CODEC building block!*

MOST v2.2: A single common building block: SpW CODEC

- The same Building Block is implemented in all components: **the SpW CODEC**
  - It is implementing ECSS-E-ST-50-12C: Network, Link, Signal & Physical layers

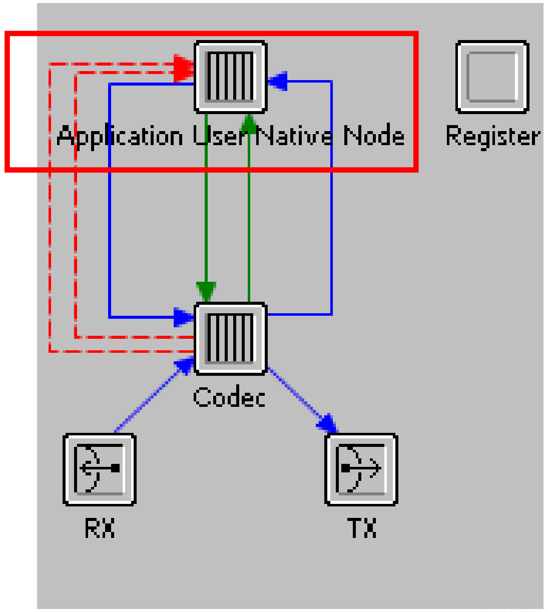
- Configurable through OPNET user interface:
  - Link Enabled status (= NOT [Link disabled])
  - Autostart status,
  - Link Start status,
  - TX Data Rate,
  - RX Buffer Size,
  - Show NULL Messages,
  - Timer Disconnect,
  - Timer Parity Error,
  - Delay For Disconnection After Parity Error,
  - Initial Timecode Register Value,
  - Debug Level (console messages).



OPEN

MOST v2.2 : Native Node presentation

- Native Node is quite basic:
  - CODEC + Application
- Protocol handling: ECSS-E-ST-50-12C
  - Packets (without PID!!)
    - Packet Emission & Reception buffers in Application layer
  - Time-codes with a local register
- The application is fairly generic and configurable:
  - Timecode Master status,
  - Timecode Inter-Arrival Time,
  - Time Code Start & Stop Time,
  - Debug Level (console messages),
  - Packet Type,
  - Cargo Size,
  - SpW Packet Inter-Arrival Time,
  - SpW Destination Address,
  - Packet Generator Start & Stop Time,
  - SpW Packet Deadline.



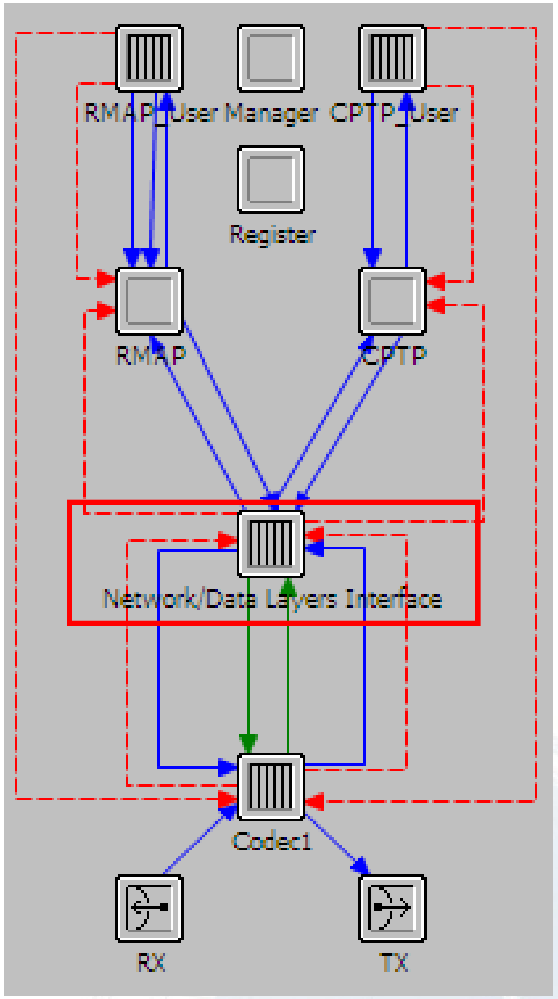
- **! The application is developed in C-Code and can be easily modified !**

ThalesAlenia Space
A Thales / Finmeccanica Company

MOST v2.2: CPTP & RMAP Node presentation (1/2)

- CPTP & RMAP Node is more advanced:
  - CODEC + PID + RMAP + CPTP + Applications (1 per CPTP or RMAP protocol)

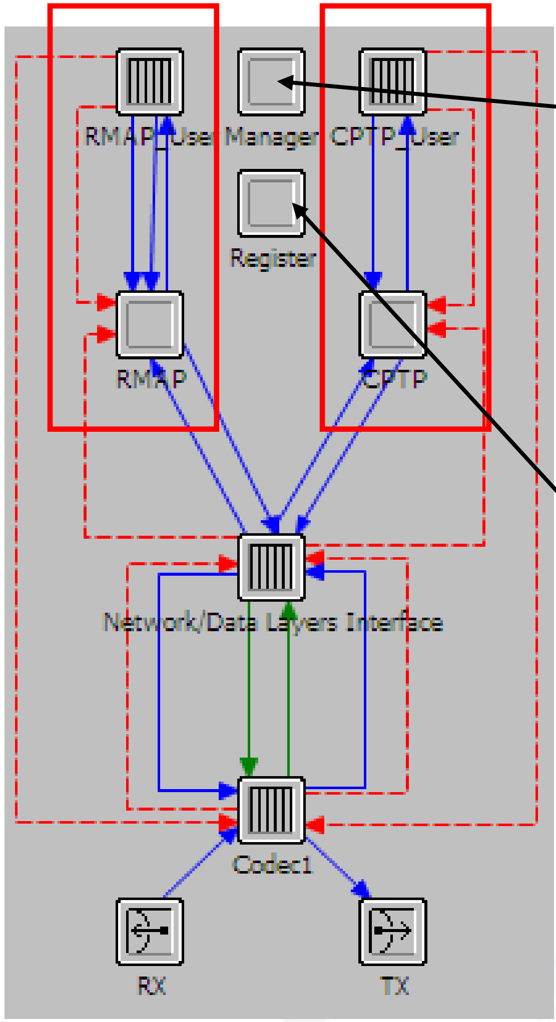
- Protocol handling:
  - ECSS-E-ST-50-12C (in CODEC)
  - ECSS-E-ST-50-51C (in NDLI)
  - ECSS-E-ST-50-52C (RMAP)
  - ECSS-E-ST-50-53C (CPTP)



- NDLI switches the packets between RMAP & CPTP and handles Time-codes:
  - NDLI Timecode Master status,
  - NDLI Timecode Interarrival Time,
  - NDLI Time Code Start / Stop Time,
  - NDLI Debug Level (console messages),
  - NDLI Emission Buffer Size,
  - NDLI Local address,
  - NDLI Local address Check.

MOST v2.2: CPTP & RMAP Node presentation (2/2)

- As for the Native Node application, the CPTP application is generic and configurable:
  - CPTP Packet Type,
  - CPTP Packet Deadline,
  - CPTP Destination Address,
  - CPTP Reception Buffer Size,
  - CPTP Debug level (console messages),
  - CPTP Packet EEP Status,
  - CPTP Elephant Message Size,
  - CPTP Elephant Message Destination Address,
  - CPTP Elephant Message Start Time.
- The RMAP application is similar:
  - RMAP Packet Interarrival Time,
  - RMAP Packet Type,
  - RMAP Command Value,
  - RMAP Start / Stop Time,
  - RMAP Packet Deadline,
  - RMAP Debug Level (console messages),
  - RMAP Key,
  - RMAP Reception Buffer Size,
  - RMAP Reply Delay,
  - RMAP Local Address,
  - RMAP Reply Packet Type.



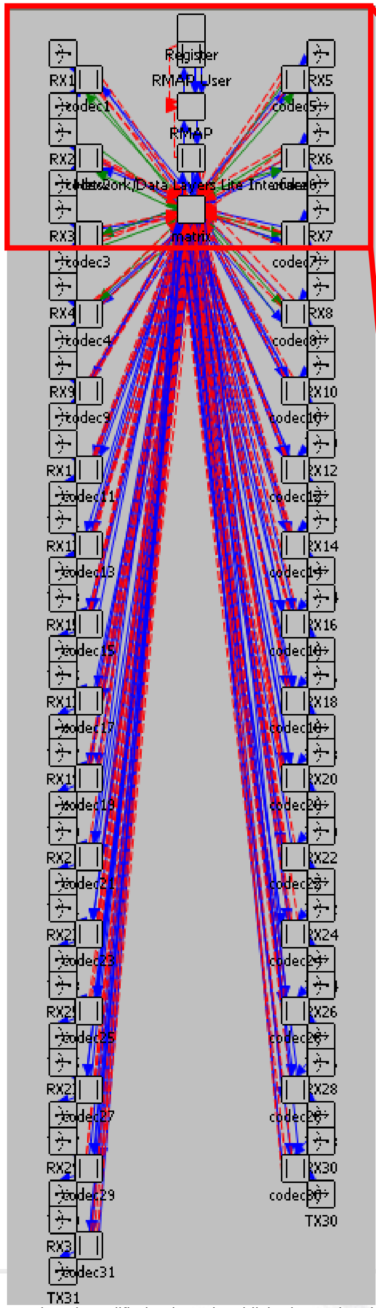
- One « manager » process has been added for management of codec errors (internal interrupts)
- The « register » can be read & modified by RMAP command and has direct effect on simulation

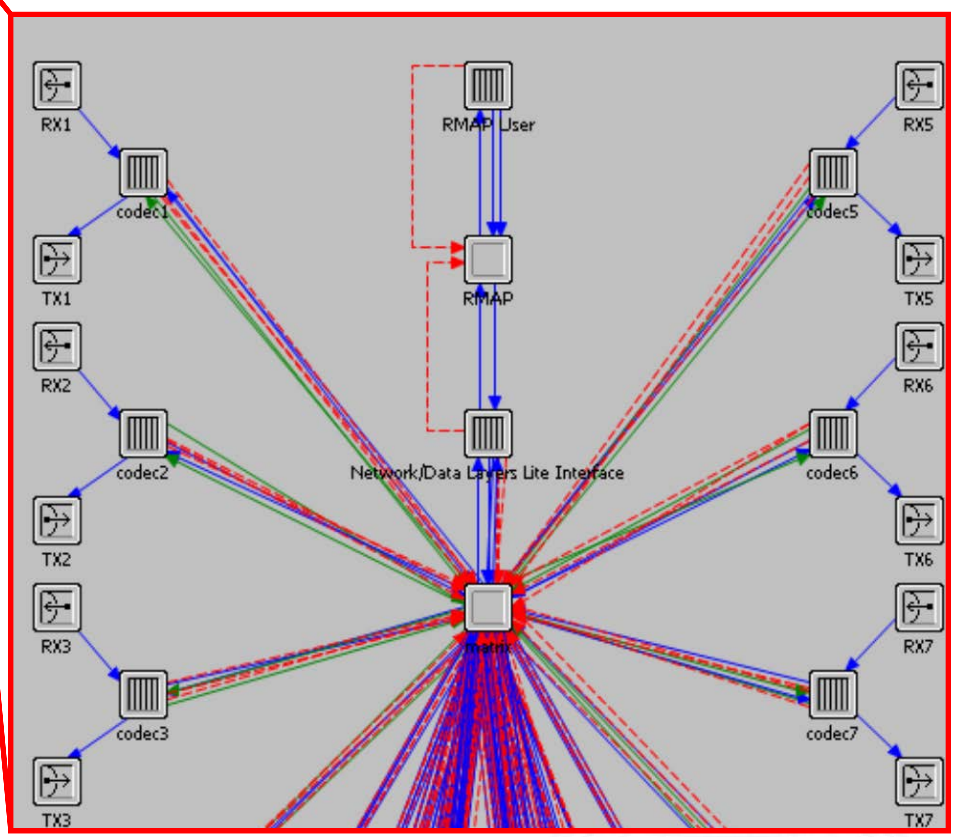
OPEN

ThalesAlenia Space
A Thales / Finmeccanica Company

MOST v2.2: Generic Switch

- 32 port generic switch:
  - 31 CODECs
  - 1 configuration port
- Routing table can be Dynamically configured by RMAP commands
- Generic registers mapping based on 10X router logic
- Also configurable:
  - Watchdog Timer status,
  - Timeout value,
  - Switching Table,
  - Router Latency,
  - Debug Level.

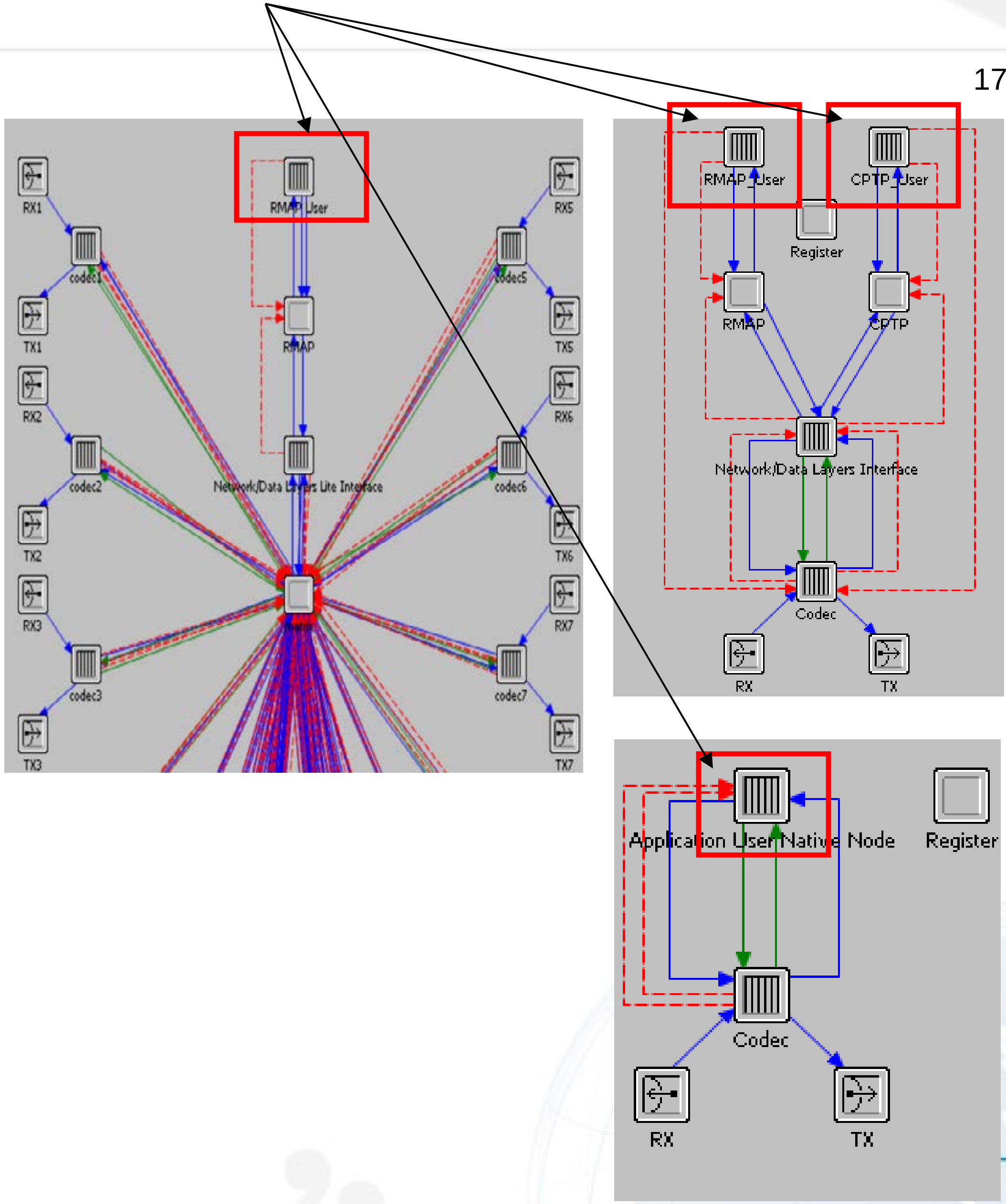




ThalesAlenia Space
A Thales / Finmeccanica Company

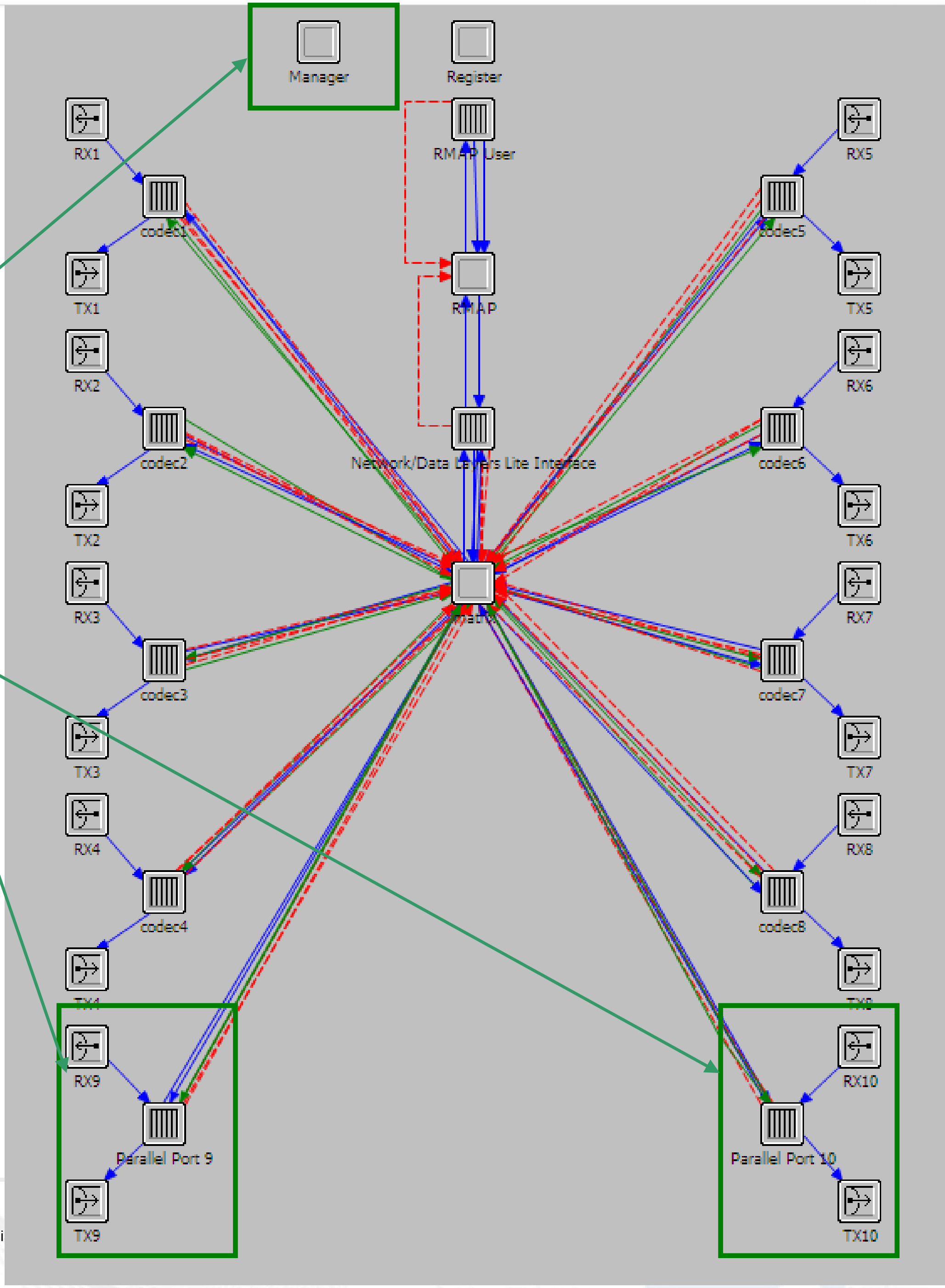
MOST v2.2: Application Layers

- All Nodes and Switches have at least one embedded application:
  - Native Node:
    - App User Native Node
  - CPTP & RMAP Node:
    - RMAP User
    - CPTP User
  - Generic Switch & 10X Switch:
    - RMAP User
- These applications are developed in C-Code to handle basic SpW features:
  - CPTP Packets sending/receiving
  - RMAP initiators
  - RMAP targets
- Other C-Code can be added in apps:
  - PUS management
  - CUC time transfer
  - TC handling triggering TM generation
  - Etc… *(Virtual Channel Multiplexing, Mass memory, couplers, FDIR management, …)*



MOST v2.2: Advanced 10X simulation

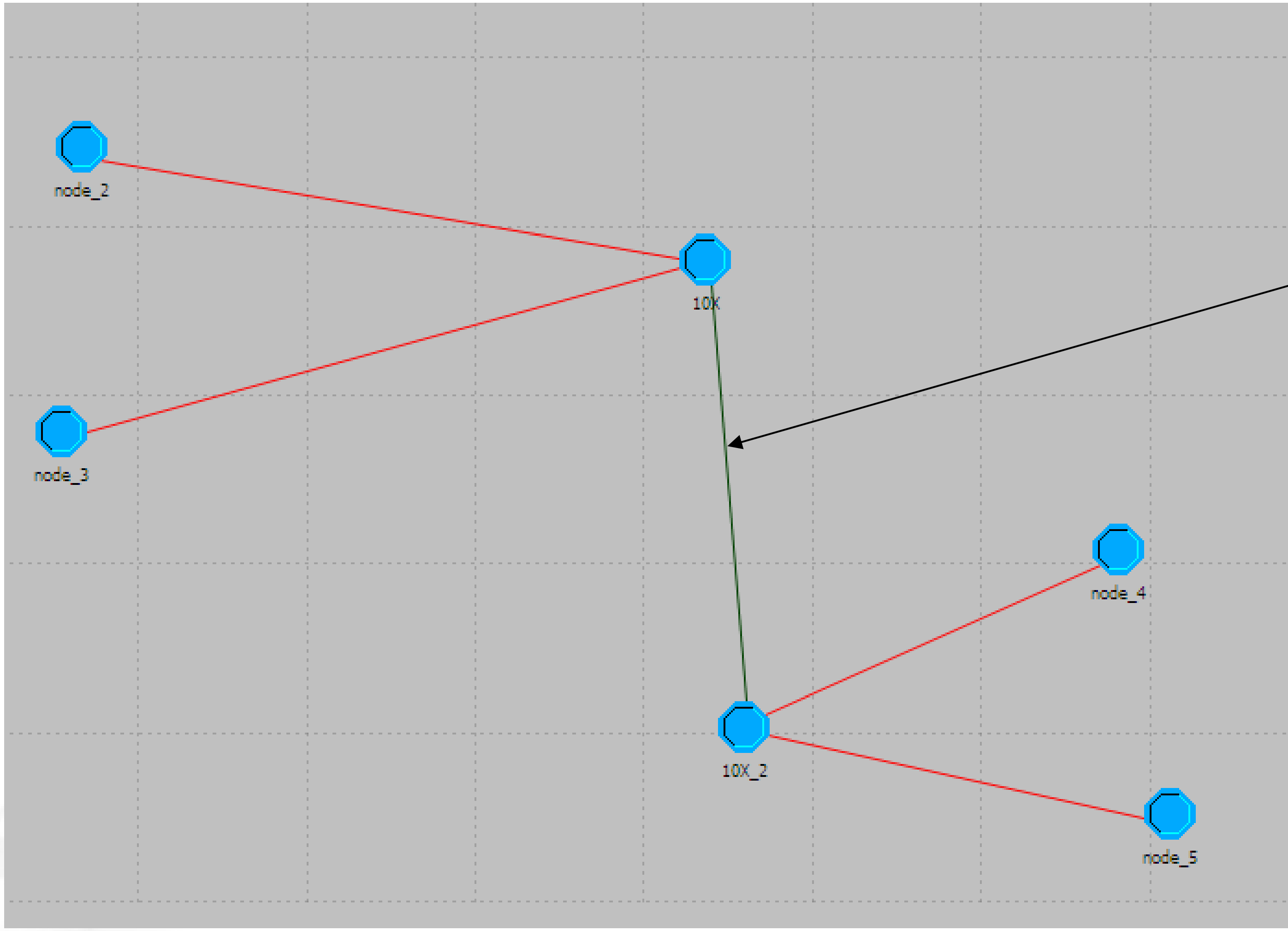
- MOST v2.2 10X router is tunable by RMAP commands with matching configuration registers:
  - Ports can be enabled/disabled/restarted during the simulation
  - Link speed can be changed by RMAP command
  - Error registers are updated on error occurrence, can be read and re-initiated
- The parallel ports have been developed with their specific error management and signaling system:
  - Management of data transfer at 240Mbps (character transfer at 30MHz scheduled on system clock)
  - Management of flow control links:
    - EXT_IN_FULL
    - EXT_IN_WRITE
    - EXT_OUT_EMPTY
    - EXT_OUT_READ



OPEN

MOST v2.2: parallel link management

- This scheme has been used for testing the new 10X functionalities

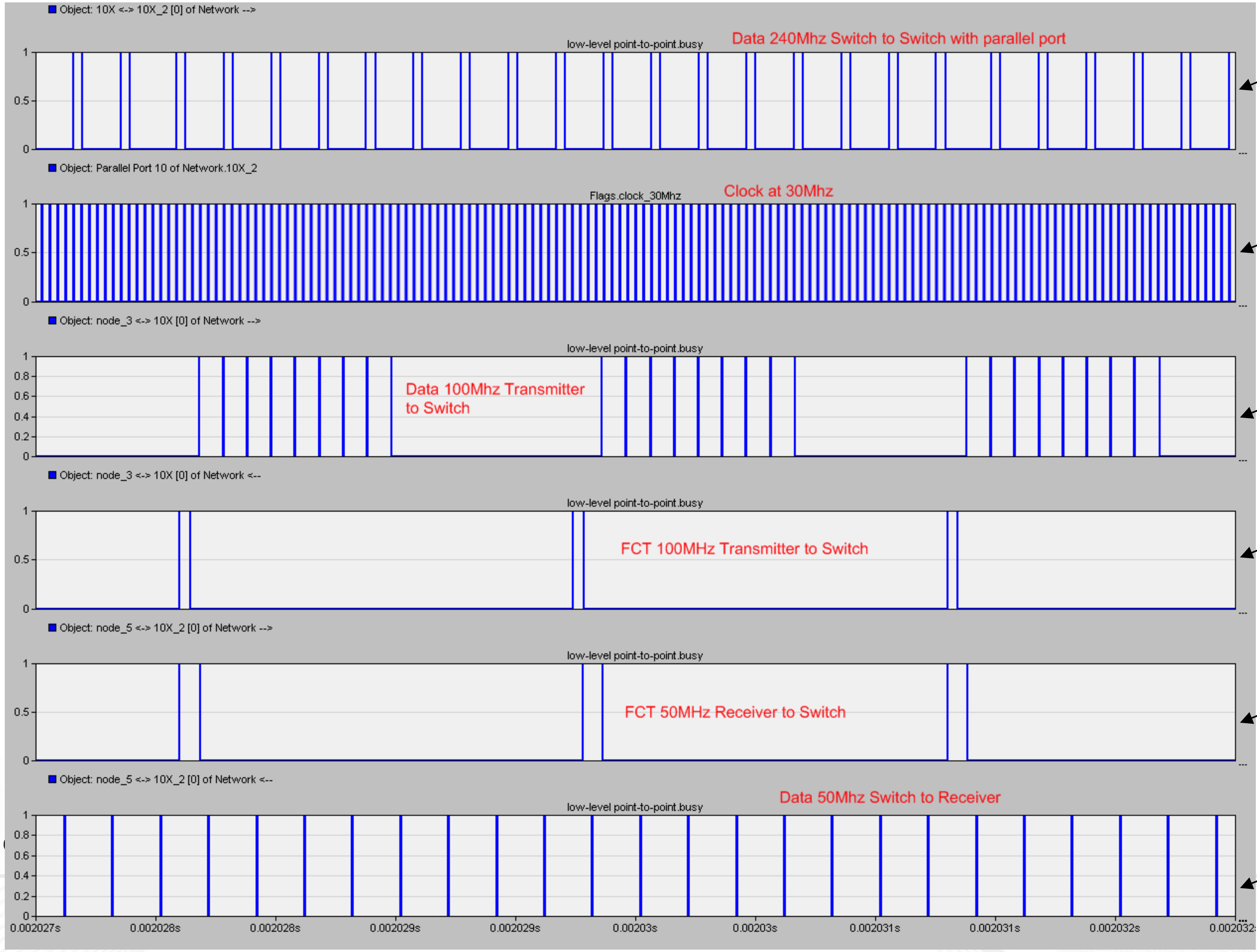


- Parallel link

OPEN

MOST v2.2: parallel link management

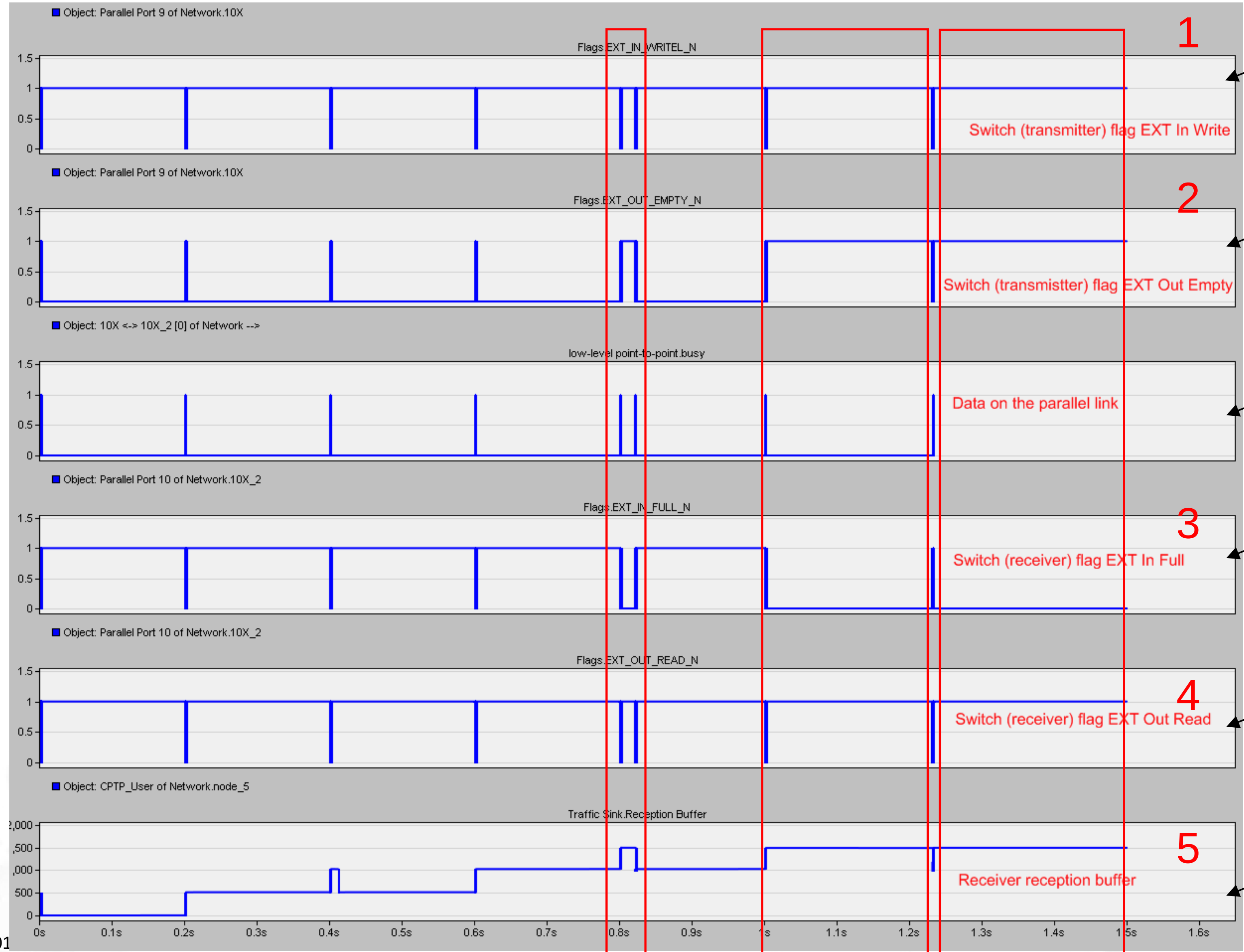
- Parallel ports: management of flow control, here with congestion in the switch by link with a lower data rate (50MHz output vs 100MHz input):



- Char exchange on Parallel link
- System Clock (30MHz)
- N-Char transfer to switch
- FCT transfer from switch to send° node
- FCT transfer from rec° node to switch
- N-Char transfer from switch

MOST v2.2: parallel link management

- Parallel ports: management of flow control, here with congestions on the link:



- EXT_IN_WRITE
- EXT_OUT_EMPTY
- Char transfer to // link
- EXT_IN_FULL
- EXT_OUT_READ
- Dest° node reception buffer

- Saturations

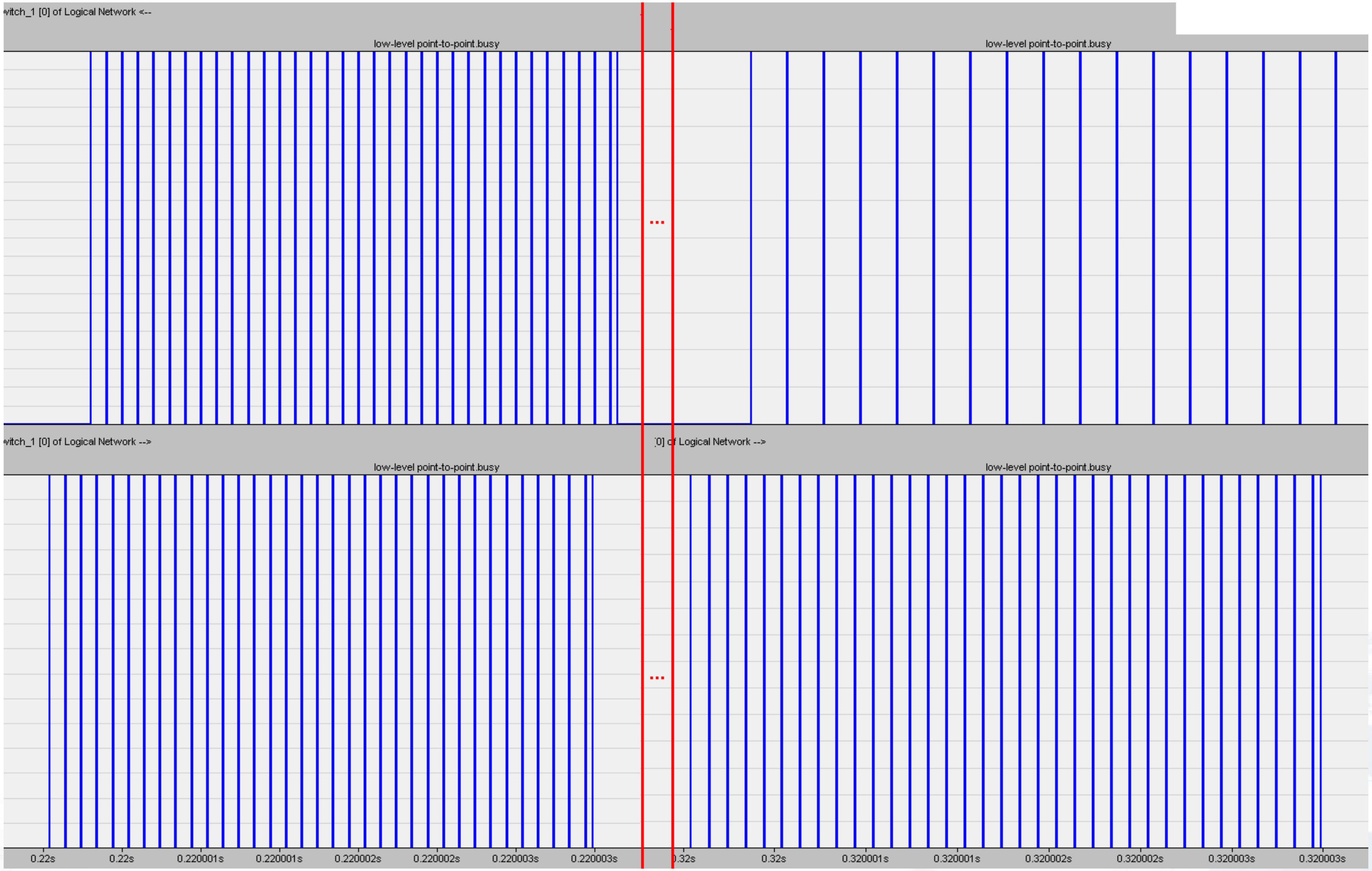
OPEN

MOST v2.2: Registers management

- All the 10X registers have been simulated and most have direct effect with the simulation:
  - Port Control/Status Registers (0-31): status of the ports & control on their state
  - Group Adaptive Routing Table Registers (32-255): configuration of logical addresses
  - Router Control/Status Registers (256-265)
    - Network discovery register
    - Router identity register
    - Router control register
    - Error active register
    - Time-code register
    - Device Manufacturer and Chip ID register
    - General Purpose register
    - Time-Code Enable register
    - Clock Control register
    - Destination Key Register

MOST v2.2: Registers management

- Example of commanding the Clock Control register (change of TXDIV parameter):



- After resetting the codec during the simulation, its rate has been halved

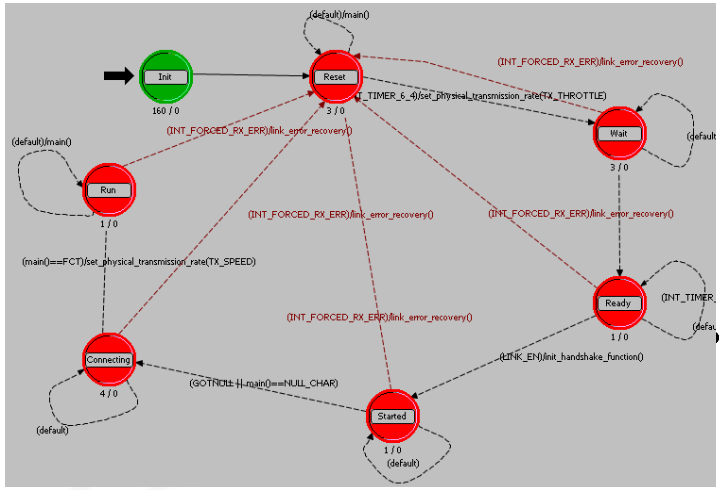
OPEN

4

Some of the MOST v2.2 capabilities

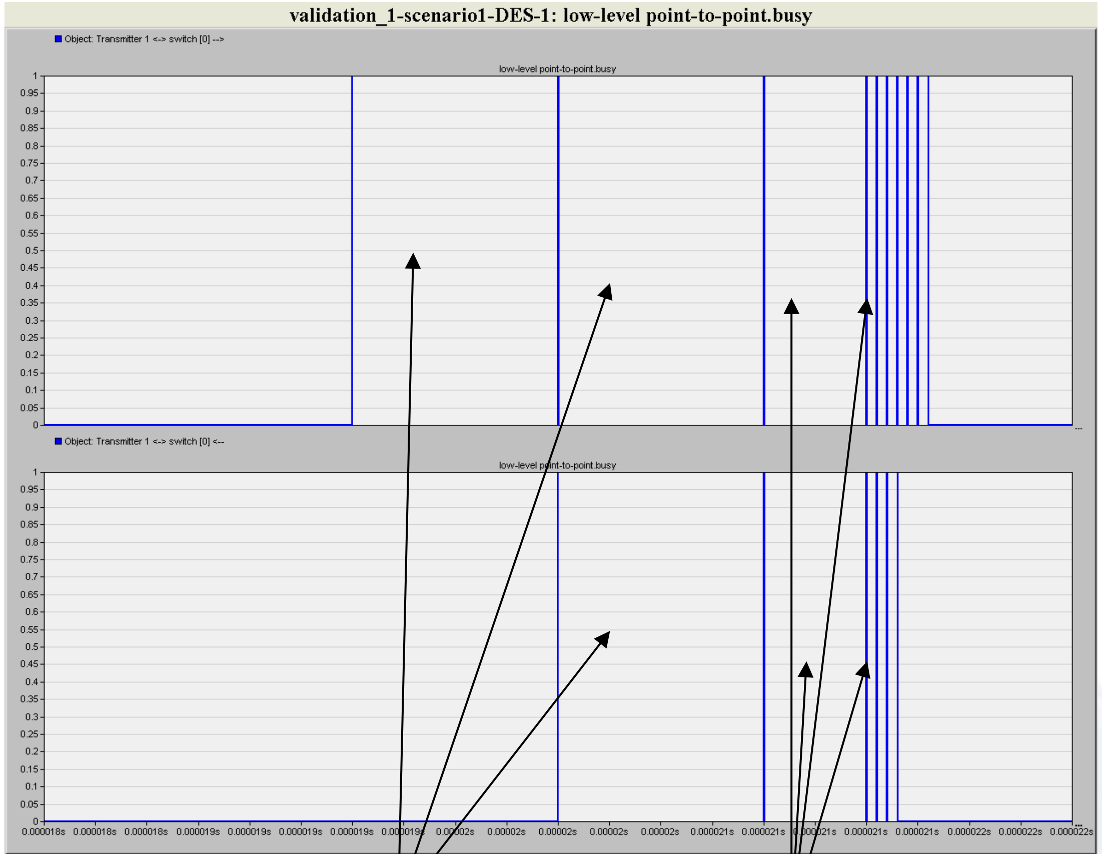
MOST v2.2: Links initialisation

- You can provoke link disconnection during the simulation to analyze the impacts on the network (loss of packets, reconfiguration procedures, etc…), enable or disable a link in the switches
- You can test any combination of init status with:
  - LinkStart
  - AutoStart
  - Link Disabled



- *LinkDisabled OFF*
- *LinkStart ON*
- *AutoStart OFF*

- *LinkDisabled OFF*
- *LinkStart OFF*
- *AutoStart ON*



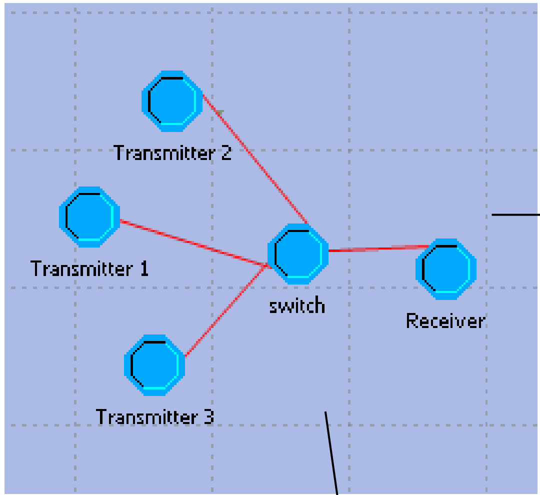
•NULL

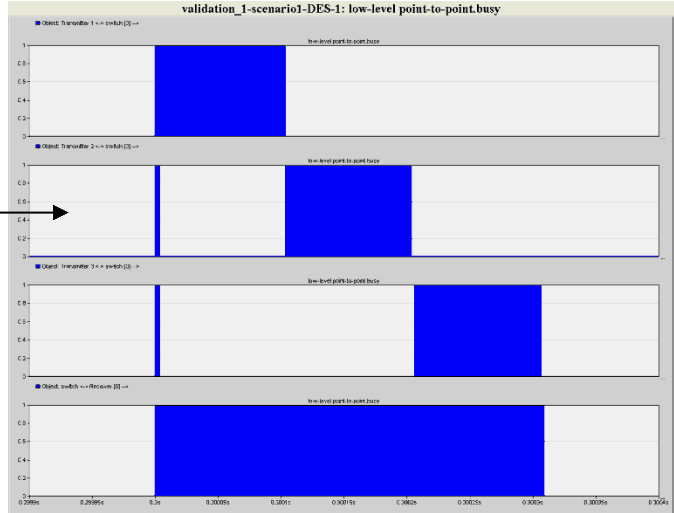
•FCT

- **Now these statuses can be updated by RMAP, for instance to test network init/re-init sequences**

MOST v2.2: Detailed traffic analysis

- You can finely tune packet generation sequence, length, type and destination, allowing to see the effects of congestions on your network and on the emission buffers of the nodes:



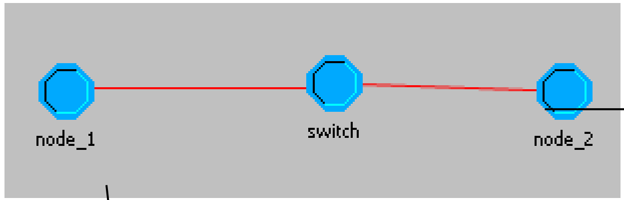


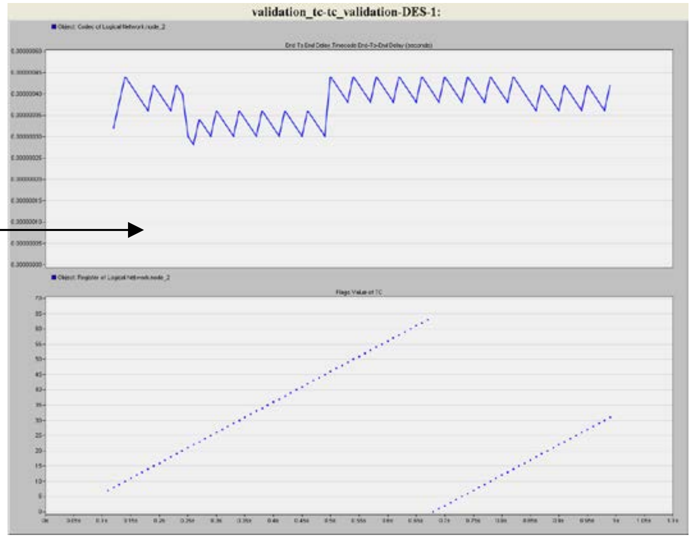
- Sender 1
- Sender 2
- Sender 3
- Receiver input

- Evolution of Node packet output buffer content

MOST v2.2: Time Code propagation measurement

- You can finely tune time code generation sequence and if a node is Time-Master or not
- Multiple “time masters” are allowed in MOST:





- Time-code propagation jitter
- Time-code value

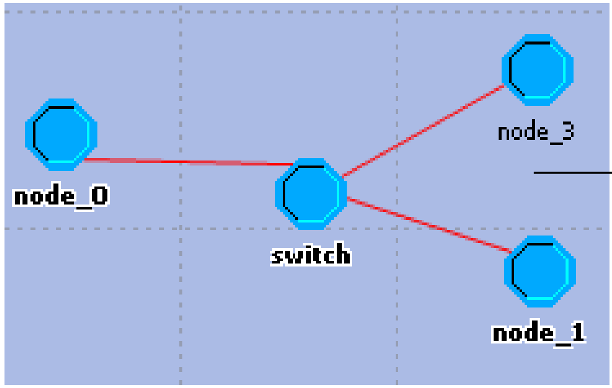
- Time code propagation busy states

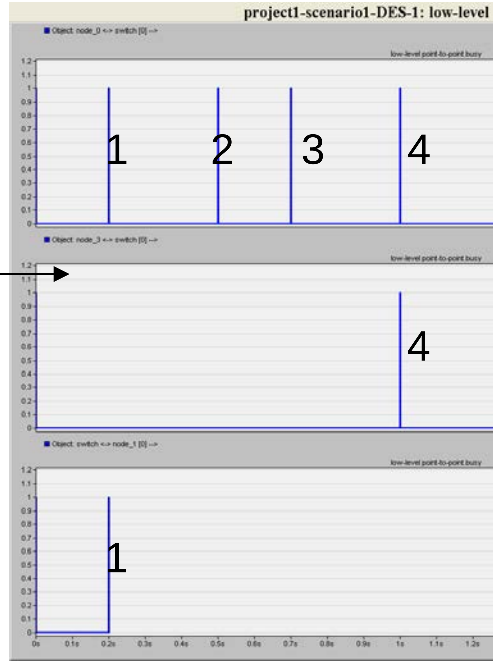
OPEN

ThalesAlenia
A Thales / Finmeccanica Company Space

MOST v2.2: Dynamic Switch reconfiguration

- Any Node can send a RMAP message to a Switch with enabled RMAP configuration port to reconfigure its switching table during the simulation:





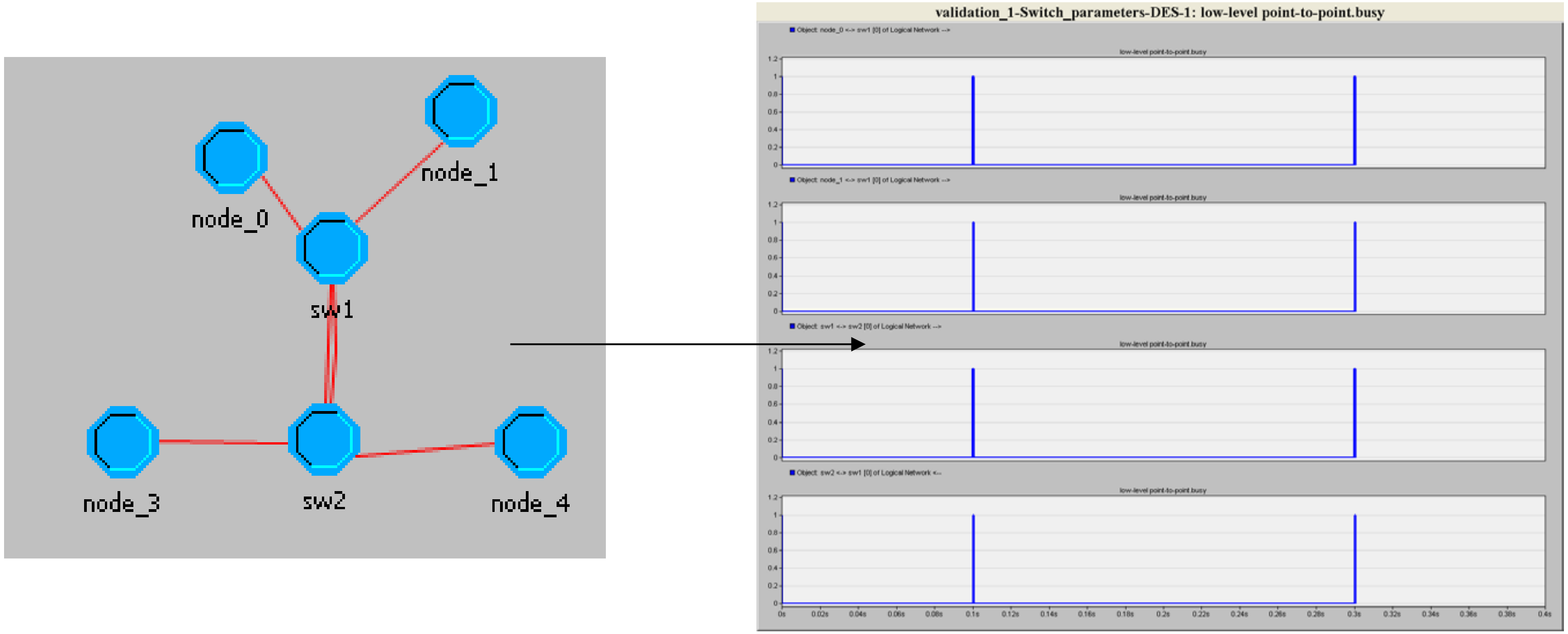
- Node 0 →
- Node 3 ←
- Node 1 ←

- Seq1 is a CPTP packet sent from Node_0 to a logical address associated with Node_1
- Seq2 is a RMAP switching reconfiguration packet changing the allocation of the logical address
- Seq3 is a FCT sequence sent in response of the reception of the RMAP reply packet from the Switch acknowledging the reconfiguration request
- Seq4 is a CPTP packet sent to the same logical address as Seq1. The CPTP packet goes to node_3.

OPEN

ThalesAlenia Space
A Thales / Finmeccanica Company

- Illustration of Group Adaptative Routing:

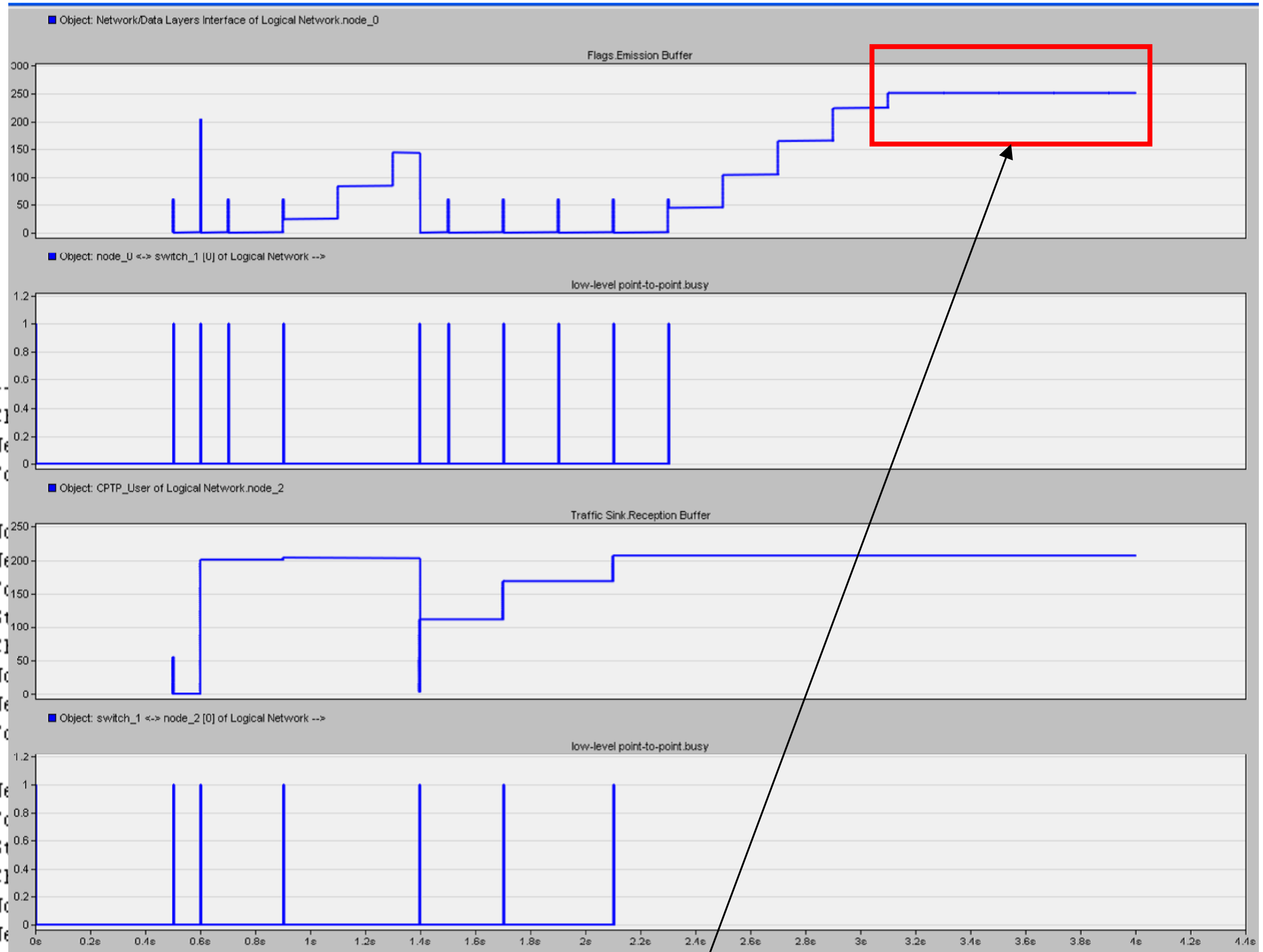


- This shows that both links between the Switches are used simultaneously to switch packets:
  - From Node_0 to Node_3
  - From Node_1 to Node_4

OPEN

MOST v2.2: Emission & reception buffers sizing

- Buffer occupation is an exportable statistic, and packet loss due to buffer saturation is printed in the OPNET console:

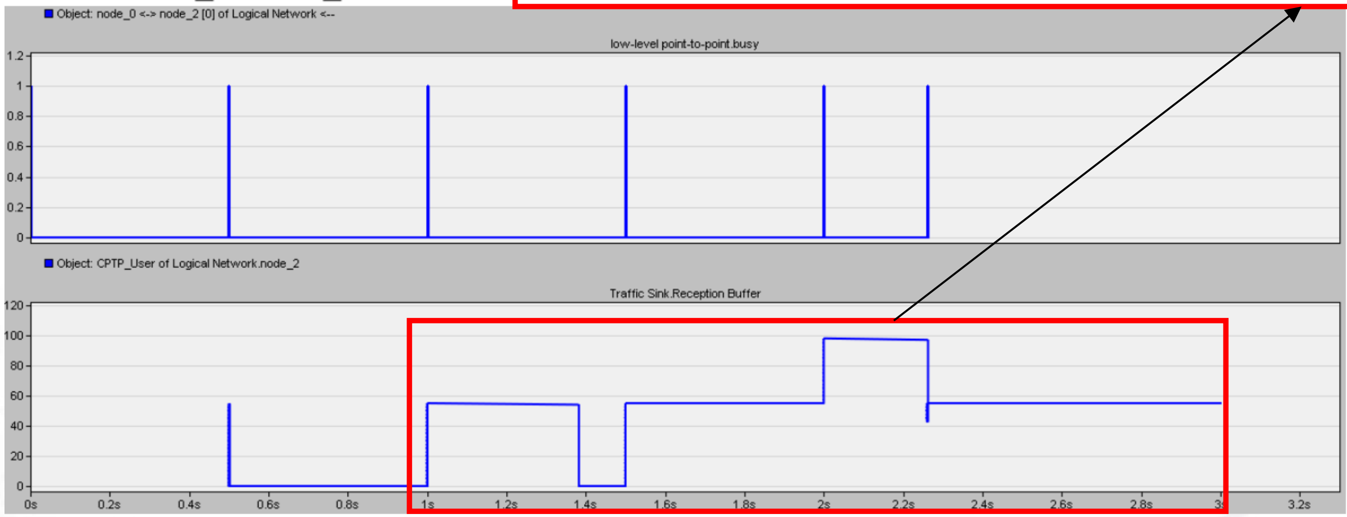



```
[1.396006140000]  [node_2:CPTP_User]          ---------------
[1.396011340000]  [switch_1:matrix] [END]  (1 | 2)      Cl
[1.396011340000]  [switch_1:matrix] [NEW]  (1 |  )      Ne
[1.396011340000]  [switch_1:matrix] [NEW]  (1 |  )      Fo
[1.396011340000]  [switch_1:matrix] [NEW]  (1 |  )
[1.396011440000]  [switch_1:matrix] [ARB]  (  | 2)      No
[1.396608940000]  [switch_1:matrix] [NEW]  (1 |  )      Ne
[1.396608940000]  [switch_1:matrix] [NEW]  (1 |  )      Fo
[1.396608940000]  [switch_1:matrix] [-->]  (1 | 2)      St
[1.396614800000]  [switch_1:matrix] [END]  (1 | 2)      Cl
[1.396614900000]  [switch_1:matrix] [ARB]  (  | 2)      No
[1.500000160000]  [switch_1:matrix] [NEW]  (1 |  )      Ne
[1.500000160000]  [switch_1:matrix] [NEW]  (1 |  )      Fo
[1.500000160000]  [switch_1:matrix] [NEW]  (1 |  )
[1.700000180000]  [switch_1:matrix] [NEW]  (1 |  )      Ne
[1.700000180000]  [switch_1:matrix] [NEW]  (1 |  )      Fo
[1.700000180000]  [switch_1:matrix] [-->]  (1 | 2)      St
[1.700006020000]  [switch_1:matrix] [END]  (1 | 2)      Cl
[1.700006120000]  [switch_1:matrix] [ARB]  (  | 2)      No
[1.900000120000]  [switch_1:matrix] [NEW]  (1 |  )      Ne
[1.900000120000]  [switch_1:matrix] [NEW]  (1 |  )      Found potential TX ports
[1.900000120000]  [switch_1:matrix] [NEW]  (1 |  )       (2) No valid output ports. Mark port as spilling port
[2.100000140000]  [switch_1:matrix] [NEW]  (1 |  )      New char arrived with address: 150
[2.100000140000]  [switch_1:matrix] [NEW]  (1 |  )      Found potential TX ports
[2.100000140000]  [switch_1:matrix] [-->]  (1 | 2)      Start Transfer
[3.100000000000]  [node_0:Network/Data Layers Interface]    Emission buffer is saturated, CPTP packet is discarded, EEP is inserted in buffer
[3.300000000000]  [node_0:Network/Data Layers Interface]    Emission buffer is saturated, CPTP packet is discarded
[3.500000000000]  [node_0:Network/Data Layers Interface]    Emission buffer is saturated, CPTP packet is discarded
```

MOST v2.2: Packets end-to-end delay check

- A communication deadline can be set per type of packet (number between 0 & 99). MOST checks this value and can trigger alarms:

```
[0.500005840000]  [node_2:CPTP_User]    ------------------------------------------------------------------
[0.500005840000]  [node_2:CPTP_User]    Delay: 0.000006 | Deadline: 0.000010 | SpW-Type: 5 | PK Size: 55 (bits: 436)
[0.500005840000]  [node_2:CPTP_User]    Normal packet with EOP
[0.500005840000]  [node_2:CPTP_User]    Packet within deadline | rounded time: 0.000006
[0.500005840000]  [node_2:CPTP_User]    ------------------------------------------------------------------
[1.380005840000]  [node_2:CPTP_User]    ------------------------------------------------------------------
[1.380005840000]  [node_2:CPTP_User]    Delay: 0.380006 | Deadline: 0.000010 | SpW-Type: 5 | PK Size: 55 (bits: 436)
[1.380005840000]  [node_2:CPTP_User]    Normal packet with EOP
[1.380005840000]  [node_2:CPTP_User]    Packet missed deadline | rounded time: 0.380006
[1.380005840000]  [node_2:CPTP_User]    ------------------------------------------------------------------
[2.260005840000]  [node_2:CPTP_User]    ------------------------------------------------------------------
[2.260005840000]  [node_2:CPTP_User]    Delay: 0.760006 | Deadline: 0.000010 | SpW-Type: 5 | PK Size: 55 (bits: 436)
[2.260005840000]  [node_2:CPTP_User]    Normal packet with EOP
[2.260005840000]  [node_2:CPTP_User]    Packet missed deadline | rounded time: 0.760006
[2.260005840000]  [node_2:CPTP_User]    ------------------------------------------------------------------
```

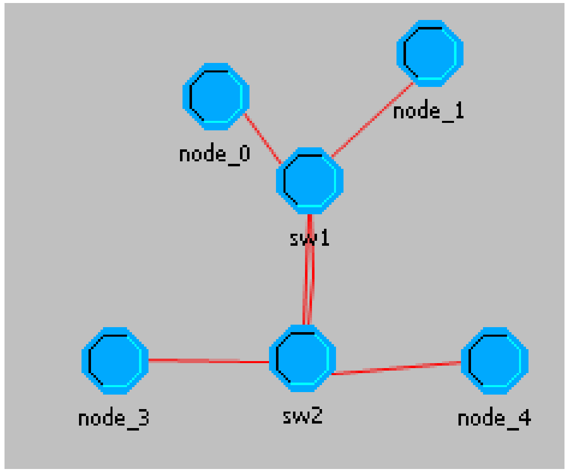


OPEN

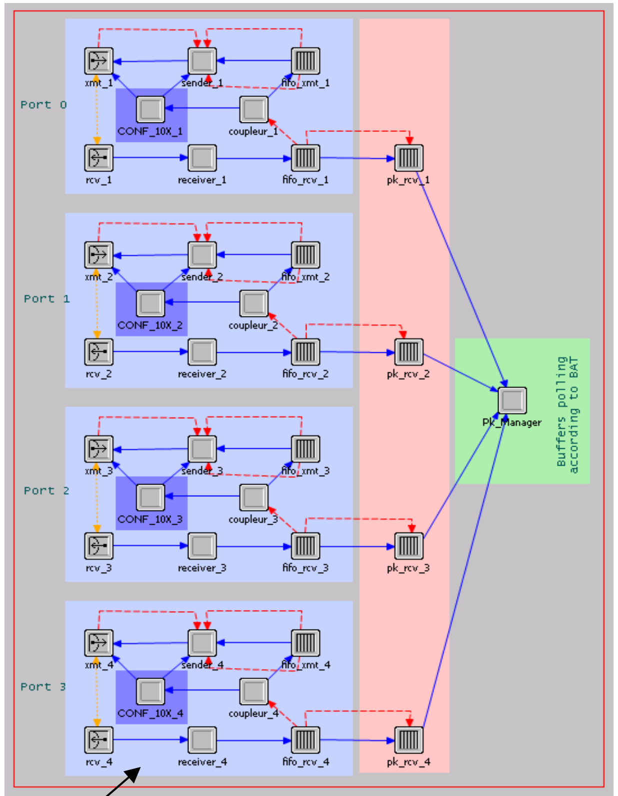
ThalesAlenia Space
A Thales / Finmeccanica Company

MOST v2.2: New components design

- Anyone can design new components or change their behavior:
  - ***(Level 1):*** Either use standard components (currently available: Native Node, CPTP & RMAP Node, 32-port dynamically configurable Switch & advanced 10X model)



  - ***(Level 2):*** Optionally: modify C-Code in User Applications to add services
    - As illustration, taking into account the reception of a TC to generate 3 TM packets takes several minutes to implement
  - ***(Level 3):*** Optionally: modify the Nodes shaping to add new layers / additional CODECs, etc…
    - That requires to change Nodes configuration in OPNET: it takes several hours (illustration: VCM machine in MOST v1.4)



  - **Example: Advance 10x model took ~1 month to develop & validate with MOST v2.2**

ThalesAlenia Space
A Thales / Finmeccanica Company

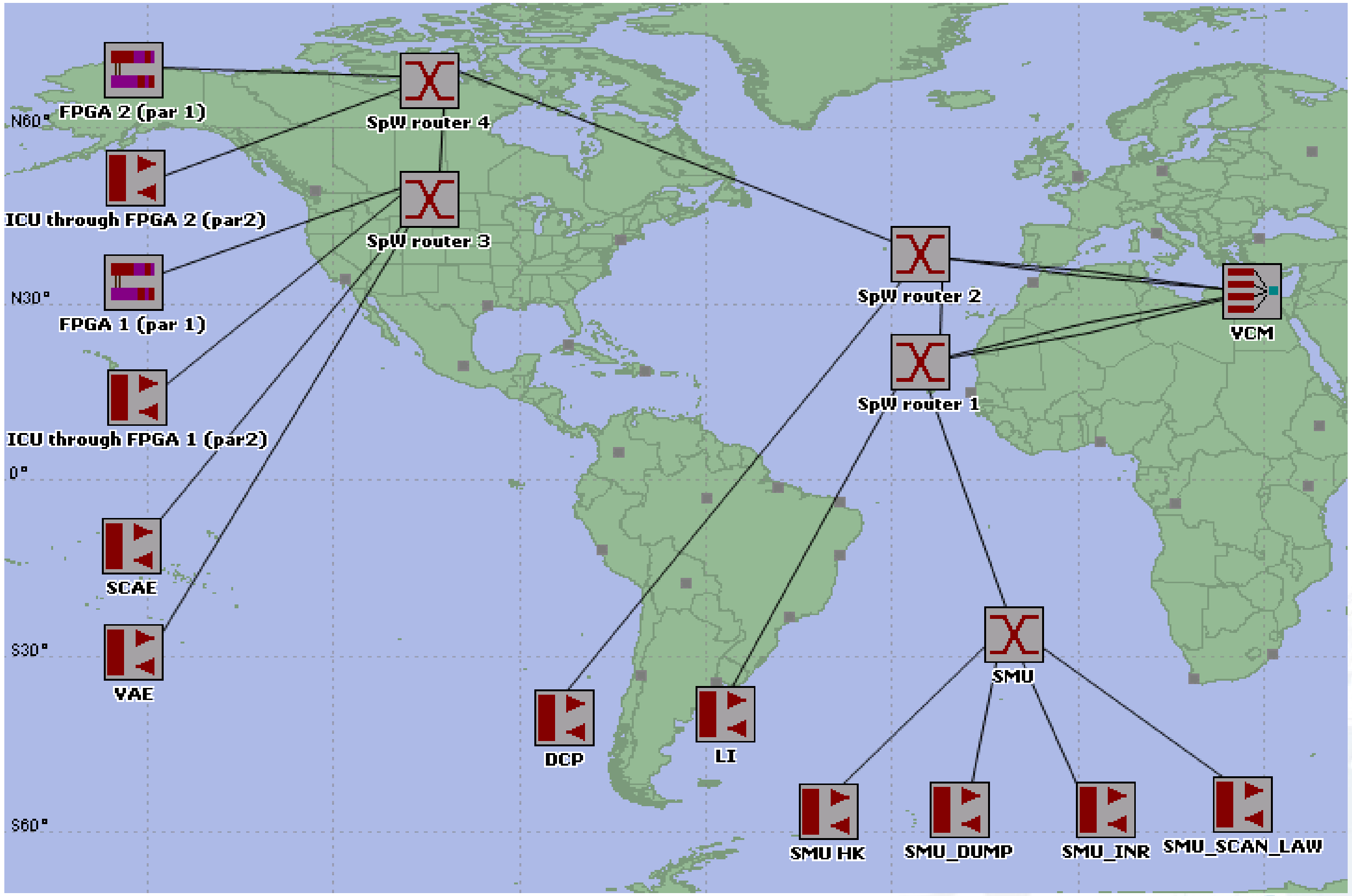
5

Example of a complex network analysis: MTG satellite

(extract from the SPW WG19 presentation)

MTG SpaceWire Network – MTG-I simulation overview

- MOST simulation of the MTG-I Topology:
  - **3 instruments, 1 SMU, 1 DDU incl. routing function & VC Multiplexing**
  - DDU is the Data Distribution Unit performing payload data multiplexing and conditionning (virtual channel multiplexing, CCSDS framing, encryption, coding)



OPEN

ThalesAlenia Space
A Thales / Finmeccanica Company

MTG SpaceWire Network – MTG-I simulation overview

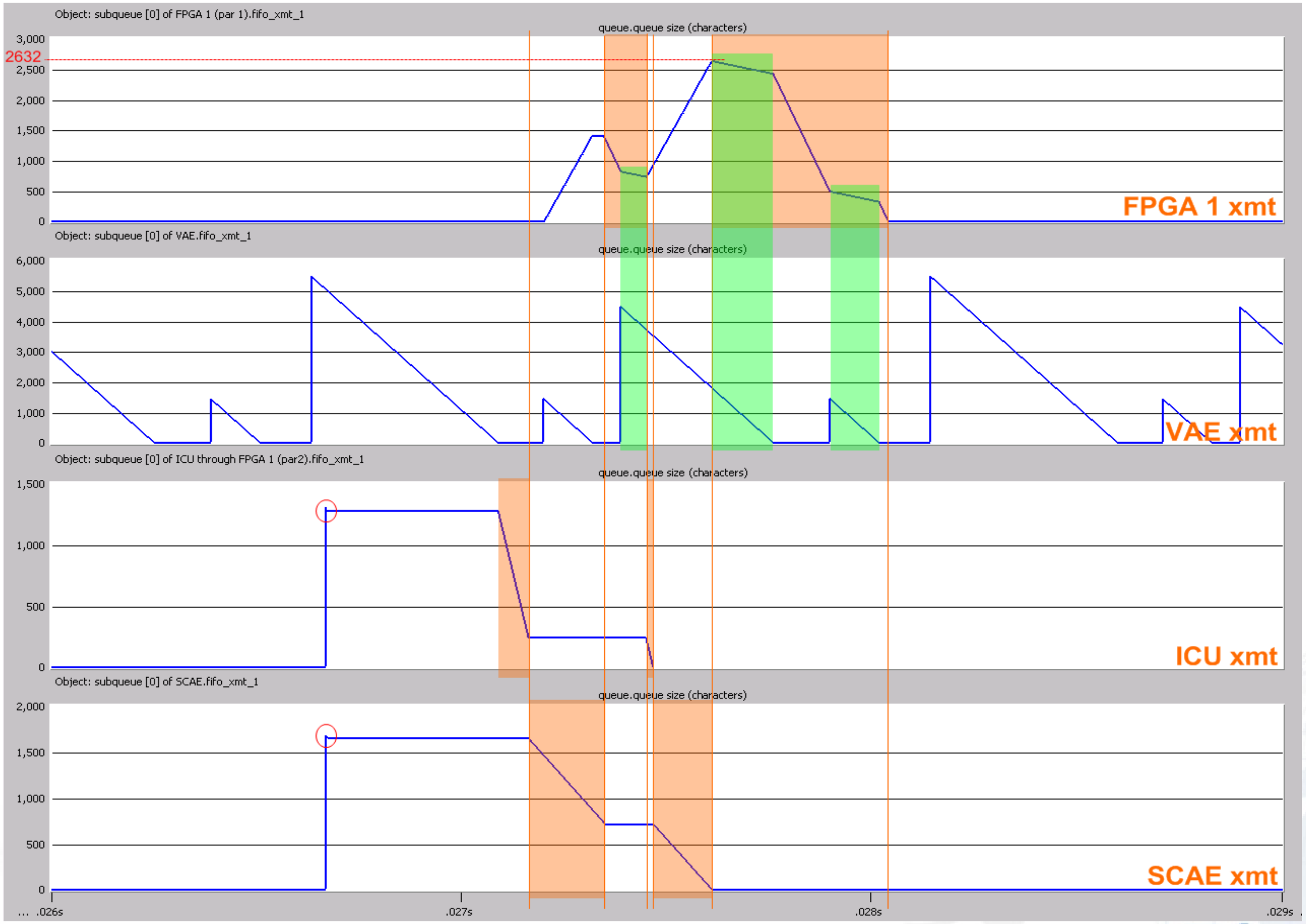
- Other MTG Constraints
  - Transmittion of packets over the network according to SpW PTP
  - Compliance to ECSS-E-ST-50-12C, ECSS-E-ST-50-51C, ECSS-E-ST-50-52C, ECSS-E-ST-50-53C
  - **Network shall support maximum spacewire interruption or delay up to 2ms**
  - Some instrument transmitting nodes **have no buffer** and rely on **coupling nodes** with buffer capability (either character forwarding or packet forwarding)
  - MOST shall analyse:
    - All SpaceWire buffers occupancy and variation
    - DDU and Virtual Channels behavior
    - Margins of each SpaceWire link according to the specified link-rate
    - Latency between packet generation and reception on the correct VCA shall be measured
    - Effect of 2ms failure of the DDU (Data Distribution Unit)
    - Impact of character forwarding instead of packet forwarding

→ 3 study cases: network with packet forward, character forward, and worst-case with 2ms traffic stall

ThalesAlenia Space
A Thales / Finmeccanica Company

FCI Coupling Buffer occupation and variation

- The maximal FPGA 1 buffer occupation in worst-case is **2632** characters

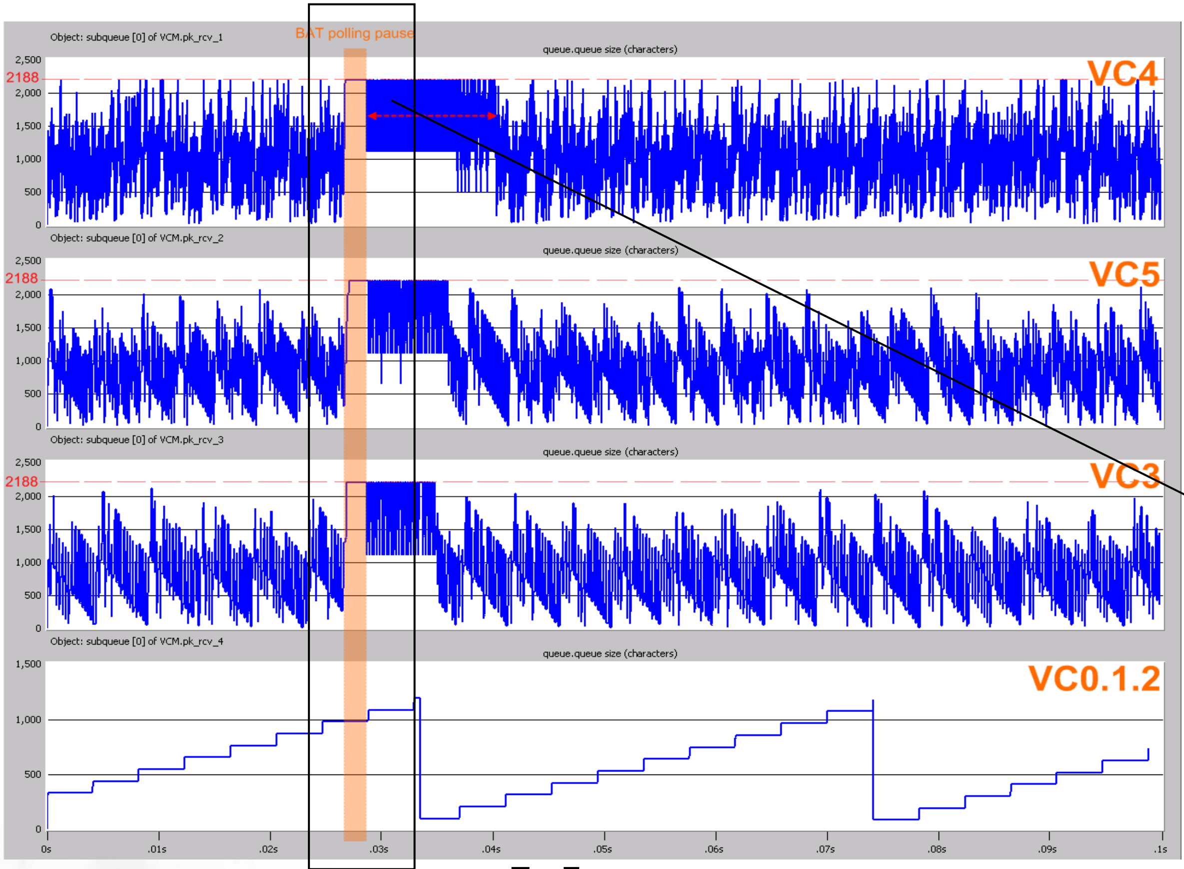


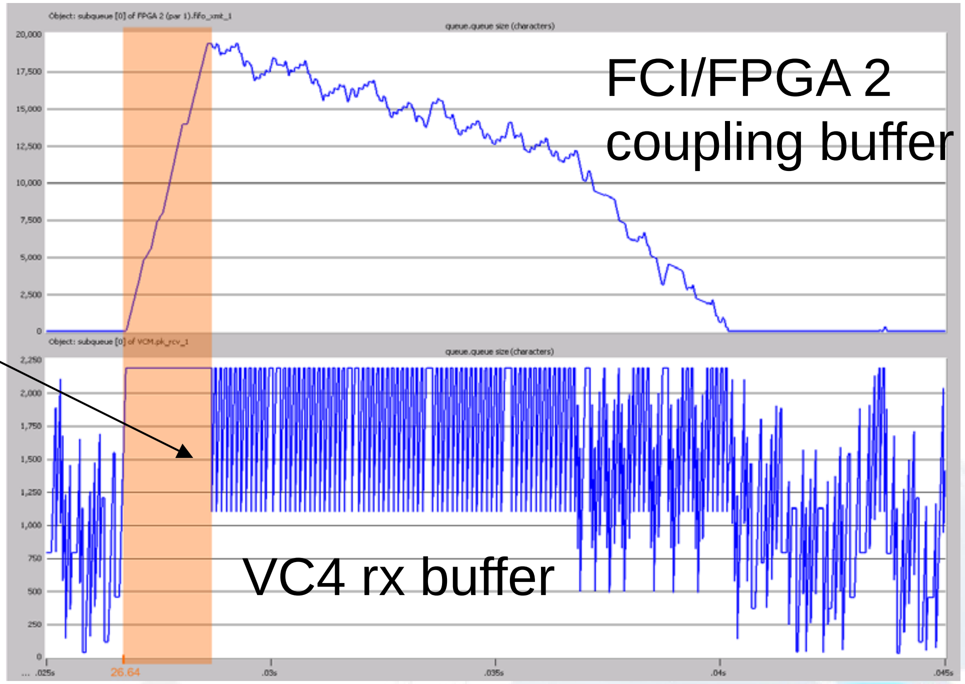
OPEN

ThalesAlenia Space
A Thales / Finmeccanica Company

VCM Buffer occupation and variation

- During the 2ms VCM stall, saturation is reached for all VC except VC0,1,2.
- Saturation of VC4 from 27ms to recovery at ~40ms creates an increase of the FCI emission buffer occupation as visible on the second graph





Worst-case zoom

OPEN

ThalesAlenia Space
A Thales / Finmeccanica Company

Links usage & occupation

- The higher link usage is 70.58% inside the FCI (character transfer time versus link capability), whereas maximal link/port occupation is 77.94%.

	Link usage	Link occupation *(incl. congestion delays)*		
	T_{char} / $T_{simulation}$	Average usage (Mean + T FCT/2)	Accuracy (+/- T FCT/2)	**Upper bound (Mean + T FCT)**
R3 -> R4 *(1)*	61.16 %	71.34 %	0.18 %	71.53 %
R4 -> R3 *(2)*	10.37 %	22.04 %	1.52 %	23.56 %
R4 -> R2 *(3)*	61.14 %	75.09 %	0.18 %	75.28 %
R2 -> R4 *(4)*	10.37 %	22.04 %	1.52 %	23.56 %
R1 -> R2	7.33 %	20.52 %	0 %	20.52 %
R2 -> R1	0.37 %	0.37 %	-	0.37 %
FPGA2 -> R4 *(5)*	63.81 %	76.42 %	1.52 %	77.94 %
R4 -> FPGA 2 *(6)*	63.81 %	72.66 %	1.52 %	74.18 %
VAE -> R3 *(7)*	70.58 %	70.6 %	0 %	70.6 %
R3 -> VAE	3.53 %	3.53 %	-	3.53 %
ICU -> R3	0.096 %	0.8 %	0 %	0.8 %
R3 -> ICU	0.005 %	0.005%	-	0.005 %
SCAE -> R3	1.7 %	2.17 %	0.51 %	2.69 %
R3 -> SCAE	20.55 %	20.53 %	0.02 %	20.55 %
FPGA1 -> R3 *(8)*	63.5 %	72.44 %	1.51 %	73.95 %
R3 -> FPGA1 *(9)*	63.5 %	72.07 %	1.51 %	73.58 %
DCP -> R2	45.93 %	50.72 %	0 %	50.72 %
R2 -> DCP	2.3 %	2.3 %	-	2.3 %
LI -> R1	37.51 %	42.33 %	0 %	42.33 %
R1 -> LI	1.88 %	1.88 %	-	1.88 %
SMU -> R1	21.09 %	21.1 %	0 %	21.1 %
R1 -> SMU	1.06%	1.06 %	-	1.06 %

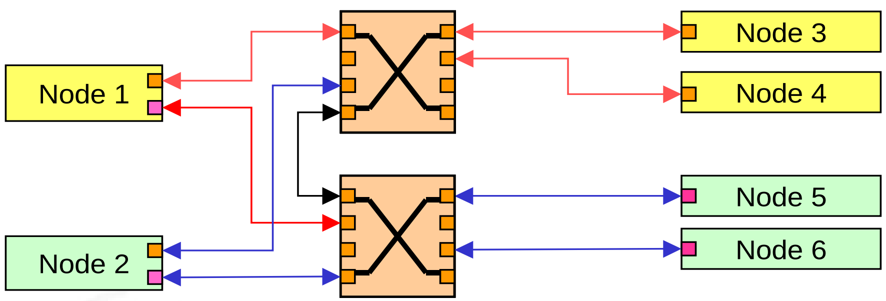
6

MOST validation with real hardware

HW cross validation on mock-up

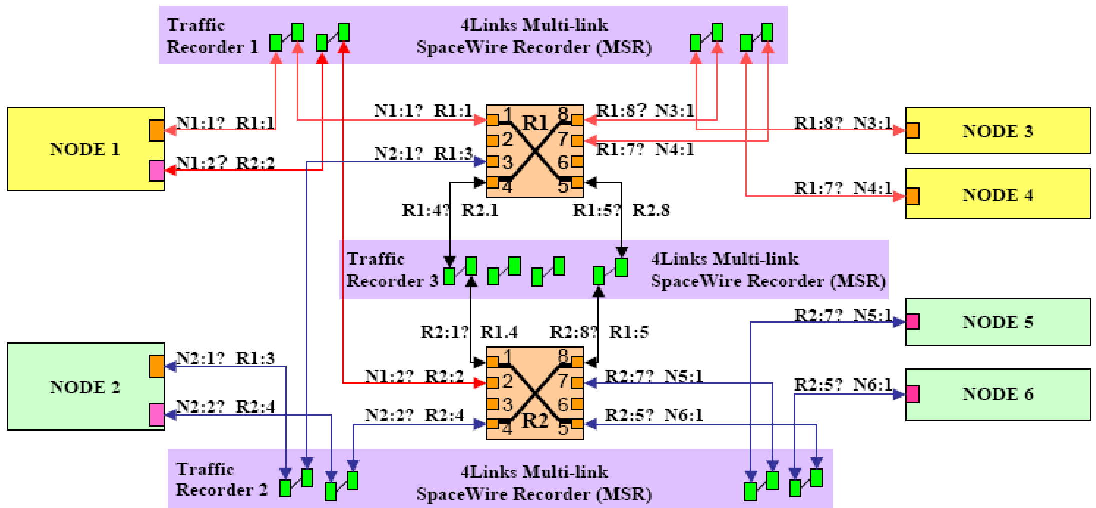
- Test on mock-up with 2 routers and 6 nodes (Gaisler Leon board or Dundee bricks)
- 4Links support to analyze traffic on HW network
- Same topology was used in MOST and HW mock-up. Comparison of results

- **MOST**



- vs

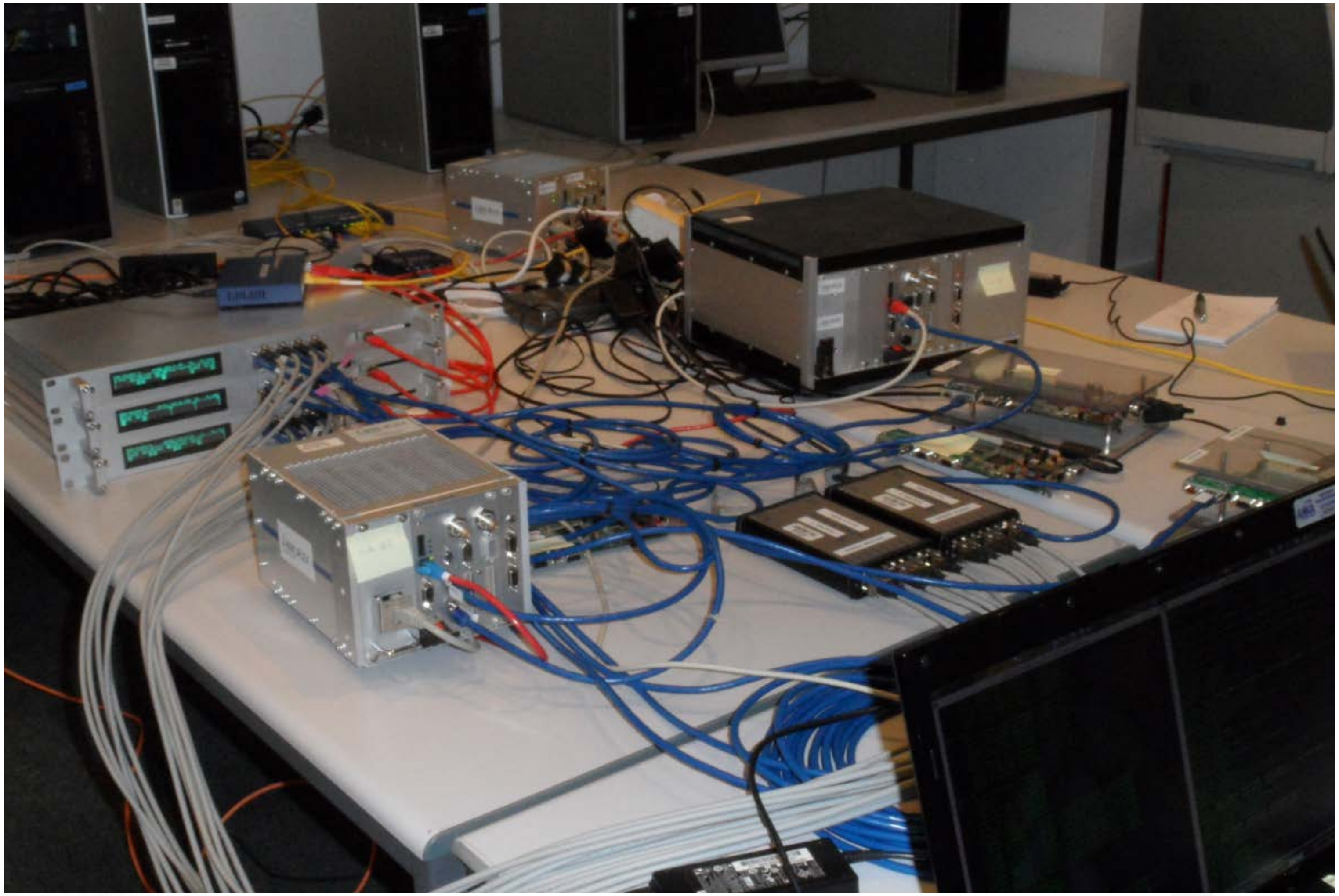
- **HW mock-up**



2012 May 15th

OPEN

ThalesAlenia Space
A Thales / Finmeccanica Company

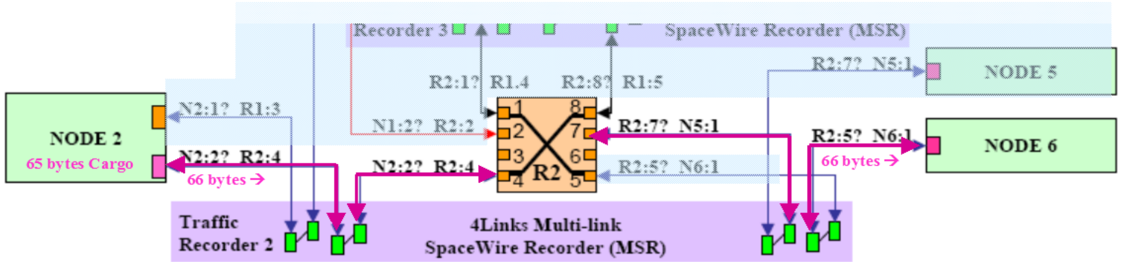


- List of devices provided by the SME 4Links:

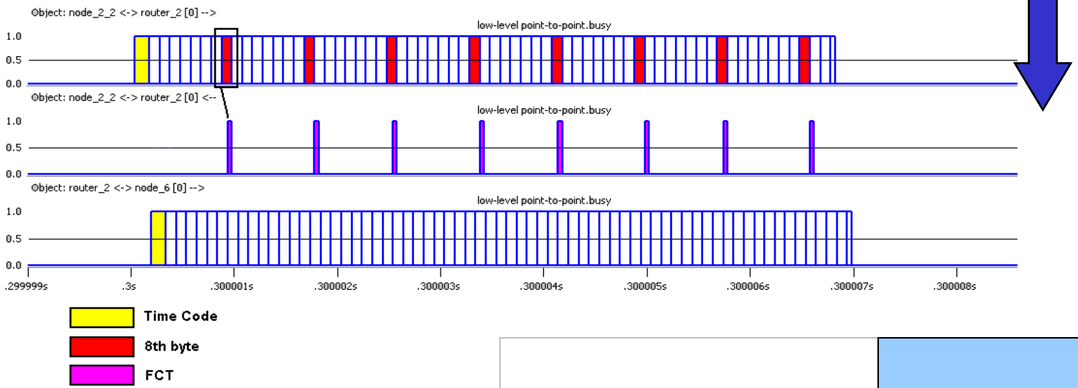
- Dundee routers
- Gaisler’s Leon boards
- 4Links SpW recorders
- SpW cables

Example of a test performed

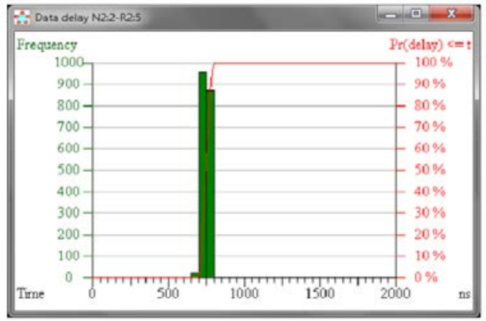
Logical Addressing & Time Code



•MOST



•HW



	MOST	HW
Time-Code duration	140ns	140ns
Transfer duration	6790ns	7920ns (Node to Router) 8810ns (Router to Node)
FCT duration	40ns	40ns
Router latency	150ns	760ns (Packet) 560 (TCs)
FCT generation	Every 8 rx bytes	Every 8 rx bytes

MOST v2.2: Package

- MOST is a SpaceWire library to OPNET Modeler v17.5 ®
- The following documentation has been written to support MOST users:
  - MOST User Manual, containing the following information:
    - How to install OPNET Modeler ® and the compilation environment
    - How to install MOST
    - How to setup your first simulation
    - How to use the MOST v2.2 embedded library
    - How to update the library, create new components and new functions
  - MOST Test & Validation report, containing the following information:
    - Generic tests performed on the codec (common building block)
    - Generic tests performed on the SpaceWire Links
    - Generic tests performed on the SpaceWire Parallel Links
    - Specific tests performed on the RMAP_CPTP Node
    - Specific tests performed on the Native Node
    - Specific tests performed on the 10X SpaceWire Switch
    - Specific tests performed on the 32 ports Switch

Conclusion

- MOST has already proved itself very useful in TAS providing the capability to test SpaceWire networks and help sizing the main parameters (link speed, buffers, …), identify bottlenecks, critical components of the network, etc…
- MOST concept provides a progressive tool, built with independent SpW building blocks which can be exchanged to test new SpaceWire technology or even SpW standard evolutions, without waiting for HW development,
- This new version is much more easy to use and runs faster than MOST v1.4: “real-time” depending on exchanged data volumes and selection of OPNET monitorings

- An advanced model of the 10X router has been implemented to demonstrate the capabilities of MOST

- **MOST v2.2 is made to evolve in parallel of the SpaceWire community progresses and as support tool for this community.**

Thanks for your attention !