A Mixed Verification Strategy Tailored for Networks on Chip

G.Tsiligiannis, L.Pierre

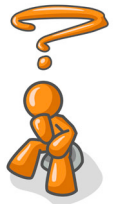
TIMA Laboratory, Grenoble, France

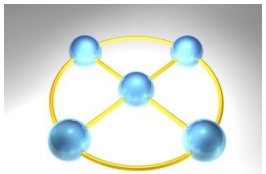
Introduction

- Questions:
  - Is it worth defining a specific *verification methodology* for NoCs?
  - How should it look like?

- Answer as a proposal:
  - Two-level approach:
    - Verification of coarse-grained features at the *algorithmic level*
    - Verification of fine-grained properties at the *implementation level* (VHDL RTL description)
  - For this latter goal → taxonomy of properties

Introduction

- « Verification » 
  - Correctness of communications, considering NoC features:
    - Network architectural characteristics (topology)
    - Routing algorithm
    - Switching technique
    - Access control
    - Synchronous or asynchronous transmission protocol
    - Buffered or bufferless transmission
    - ...



Introduction

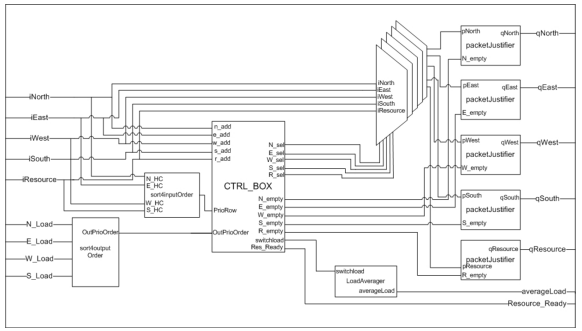
- Nowadays, few verification-oriented results
  - [Salaün et al, ASYNC'2007]: model checking approach, CHP specification translated into LOTOS description, some properties proven for FAUST (using CADP)
  - [Yean-Ru et al, ICGCS'2010]: verification of properties for a part of a wormhole XY-routing NoC router, using State Graph Manipulator
  - [Chenard et al, workshop at DATE'2007]: Assertion-Based Verification for a hierarchical ring network, debugging infrastructure
  - [Goossens et al, NoCs'2007]: also uses a monitor-based solution, and proposes a debug architecture

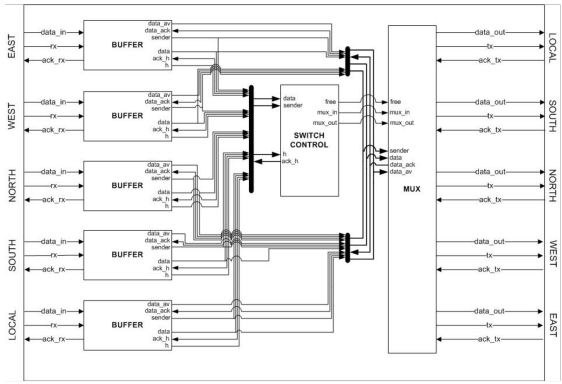
Proposal

- Mixed verification strategy
  - Verification of high-level (algorithmic) properties
    - Network description at a high level of abstraction, focuses on functional features and ignores implementation details
    - *Algorithmic* specification + formal verification (description in a functional language and verification by *theorem proving*)
  - Complemented by verification of low-level (implementation) properties
    - Network description at a low level of abstraction: VHDL RTL source code
    - Specification by *logic and temporal properties* + semi-formal verification (*Assertion-Based Verification*)

Applied to 2 state-of-the-art NoCs

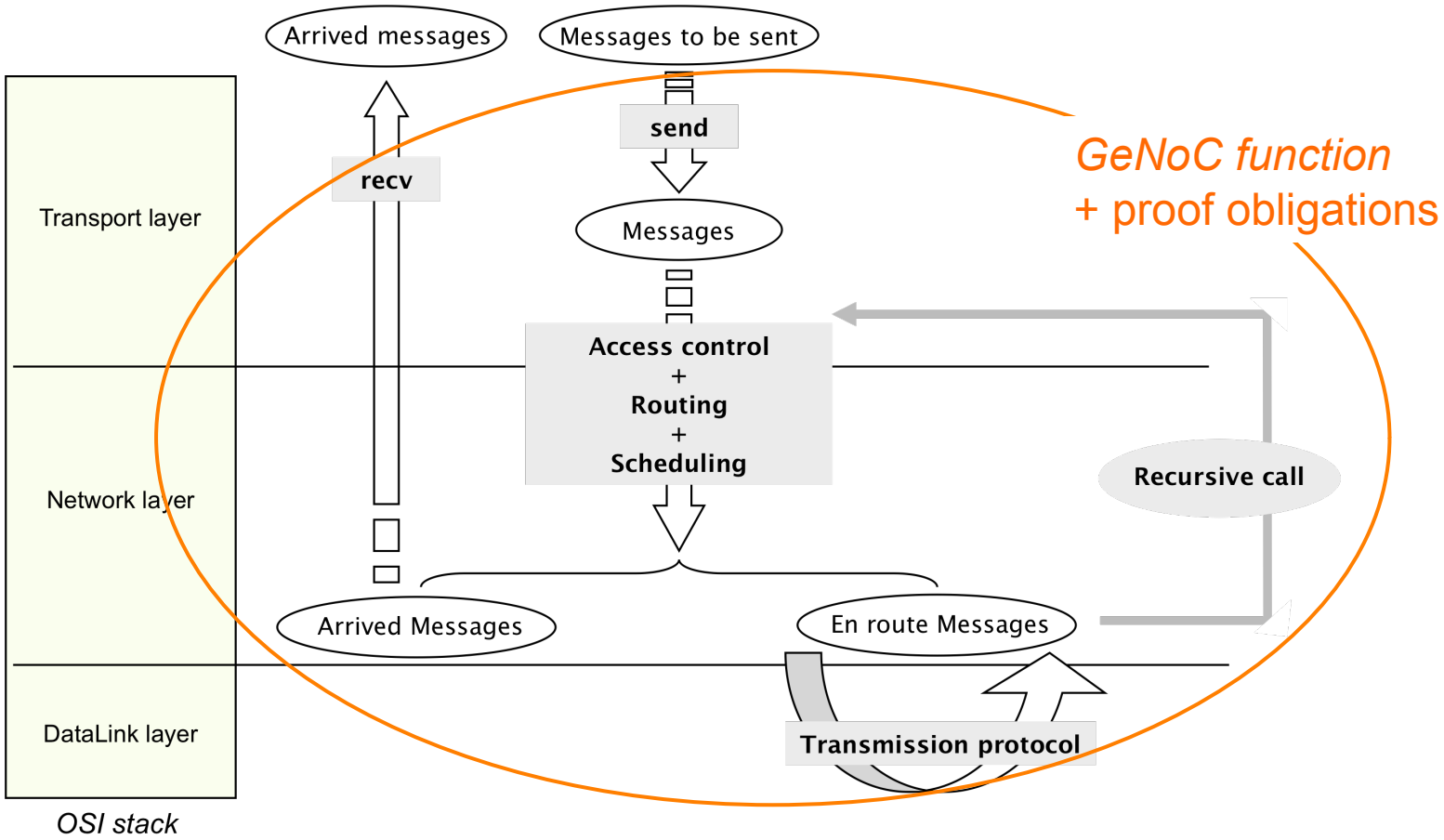
- Nostrum (http://www.ict.kth.se/nostrum/)
  - 2D-mesh topology
  - Hot potato routing
  - Packet switching
  - Synchronous
- Hermes (PUCRS, Brazil)
  - 2D-mesh topology
  - Configurable routing (here minimal negative first routing)
  - Wormhole switching
  - Asynchronous (handshake protocol)





At the algorithmic level

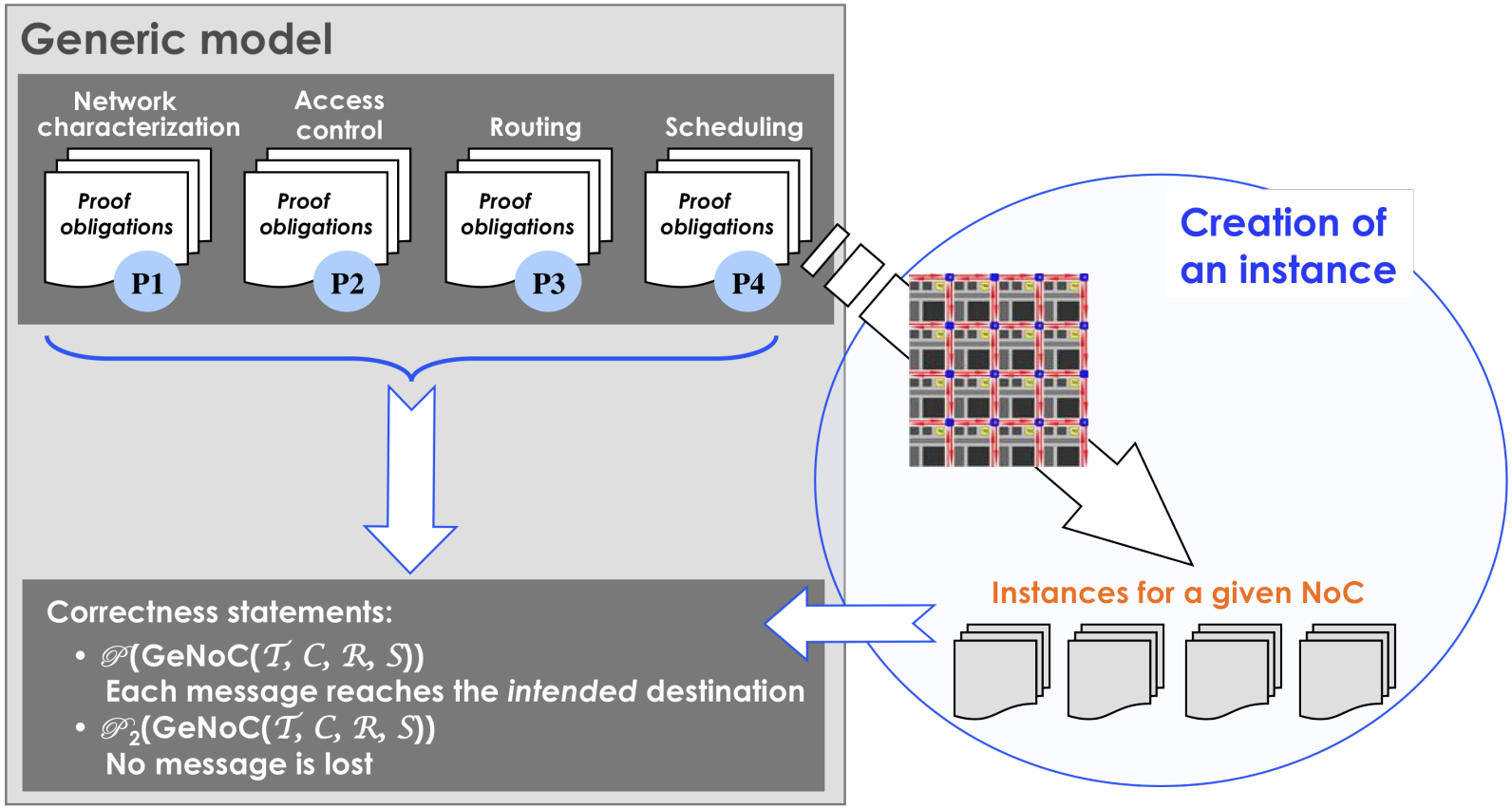
- Generic modeling and verification technique (*)



(*) *D. Borrione, A. Helmy, L. Pierre, J. Schmaltz: "A formal approach to the verification of networks on chip", EURASIP J. Embedded Systems, 2009.*

At the algorithmic level

- Generic modeling and verification technique (*)



(*) *D. Borrione, A. Helmy, L. Pierre, J. Schmaltz: "A formal approach to the verification of networks on chip", EURASIP J. Embedded Systems, 2009.*

At the implementation level

- Assertion-Based Verification
  - *Assertion*: statement about the intended behaviour or a requirement of the design
    - Temporal logics: CTL, LTL,…
    - Specification languages: SVA (IEEE Std 1800), PSL (IEEE Std 1850)
  - *Assertion-Based Verification*: does the design obey these temporal assertions?
    - Static analysis (model checking)
    - Dynamic verification (during simulation)

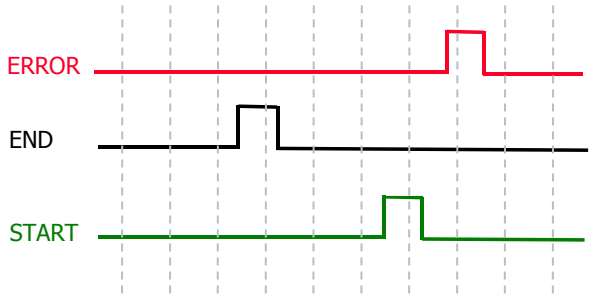
At the implementation level

- Assertion-Based Verification
  - *Assertion*: statement about the intended behaviour or a requirement of the design
    - Temporal logics: CTL, LTL,…
    - Specification languages: SVA (IEEE Std 1800), PSL (IEEE Std 1850)
  - *Assertion-Based Verification*: does the design obey these temporal assertions?
    - Static analysis (model checking)
    - Dynamic verification (during simulation)

ASSERTION-BASED VERIFICATION

- Verification of fine-grained properties on the *signals* of the design
- Examples:
  - Temporal operators

default clock = (posedge clk);
always (END ->
next (START before ERROR))

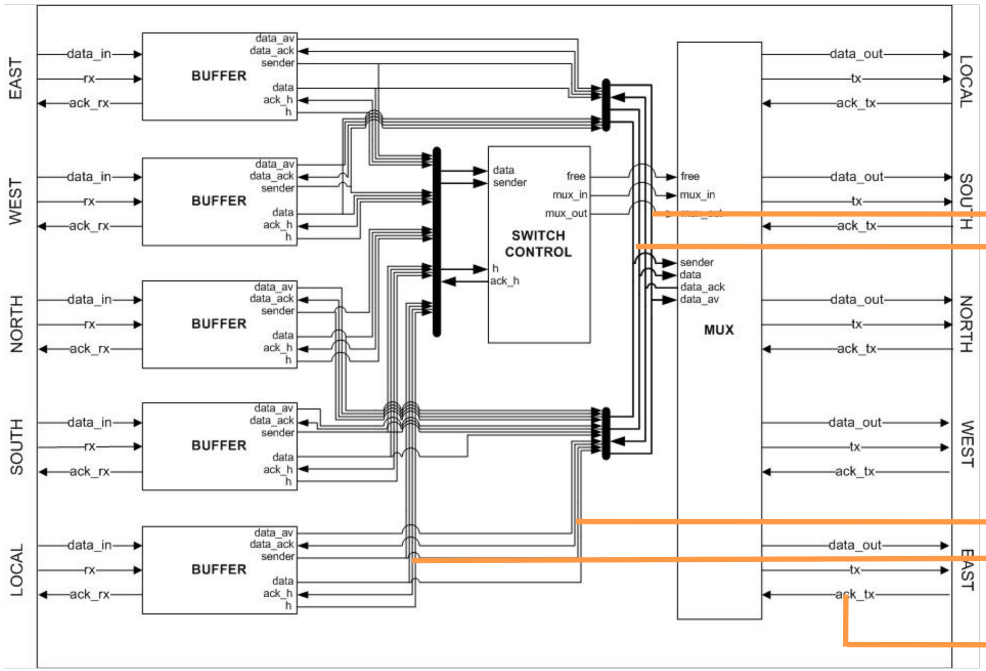


  - Extended regular expressions

default clock = (posedge clk);
*always ({ X and not Y; X and Y } |-> { not Ctrl [*8] ; Ctrl })*

ASSERTION-BASED VERIFICATION

- Hardware monitors from PSL assertions for router properties



valid

valid

Classification of properties

- General-purpose assertions for routers

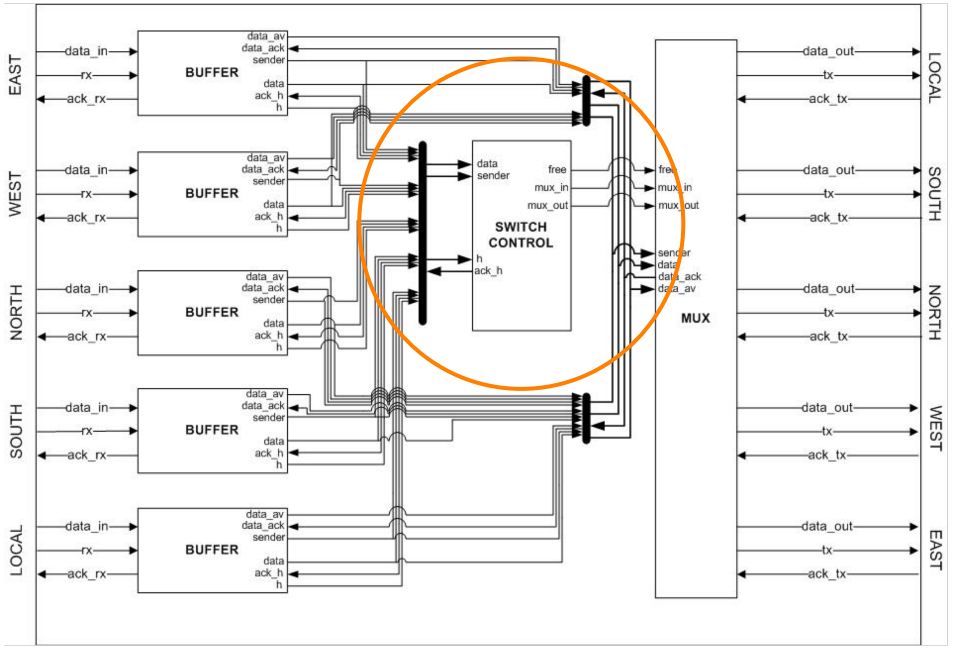
Property	Routing	QoS	Synchronous/ asynchronous	Buffered/ bufferless
No packet loss			X	+ switching technique
No packet duplication			X	X
Correct delivery upon arrival	Deterministic/ fully adaptive			
Routing decision integrity				
Satisfaction of QoS		Guaranteed/best effort		
Packet progression				

CLASSIFICATION OF PROPERTIES

- Example: No packet loss
  - Inside the router
    - Case of buffered communications
      - Wormhole: *the allocated resources will remain allocated to the same packet flow until the last flit is transmitted*
      - *...*
    - Case of bufferless communications
      - *If a packet enters the router, it will be ready to leave the router at the same cycle*
      - *A packet will not be dropped if the requesting destination is available*
  - Between two routers
    - Case of synchronous communications
    - Case of asynchronous communications

CLASSIFICATION OF PROPERTIES

- Example: No packet loss
  - Inside the router
    - Hermes (PUCRS, Brazil)
      - *The allocated resources will remain allocated to the same packet flow until the last flit is transmitted*



Classification of properties

- Example: No packet loss
  - Inside the router
    - Hermes (PUCRS, Brazil)
      - *The allocated resources will remain allocated to the same packet flow until the last flit is transmitted*

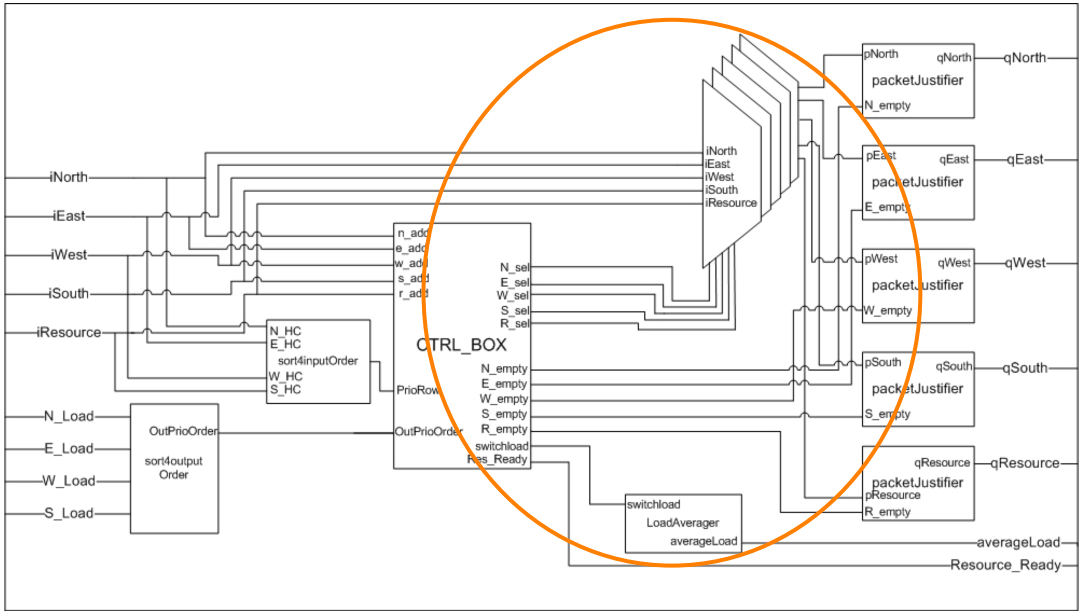
```
forall i in {0 to 4}:
always ((free(CONV_INTEGER(source(i)))='0'
        and sender(i)='1') ->
            (free(CONV_INTEGER(source(i)))='0'
             and sender(i)='1')
            until! sender(i)='0' );
```

Relation output port / input port

End of transmission

Classification of properties

- Example: No packet loss
  - Inside the router
    - Nostrum (http://www.ict.kth.se/nostrum/)
      - *If a packet enters the router, it will be ready to leave the router at the same cycle*



Classification of properties

- Example: No packet loss
  - Inside the router
    - Nostrum (http://www.ict.kth.se/nostrum/)
      - *If a packet enters the router, it will be ready to leave the router at the same cycle*

```
forall i in {0 to 4}:
always (empty_temp(i)='1' ->
        (N_select(i)='1' and N_empty='0') or
        (S_select(i)='1' and S_empty='0') or
        (E_select(i)='1' and E_empty='0') or
        (W_select(i)='1' and W_empty='0') or
        (R_select(i)='1' and R_empty='0'));
```

Packet at the input

Routed to one output

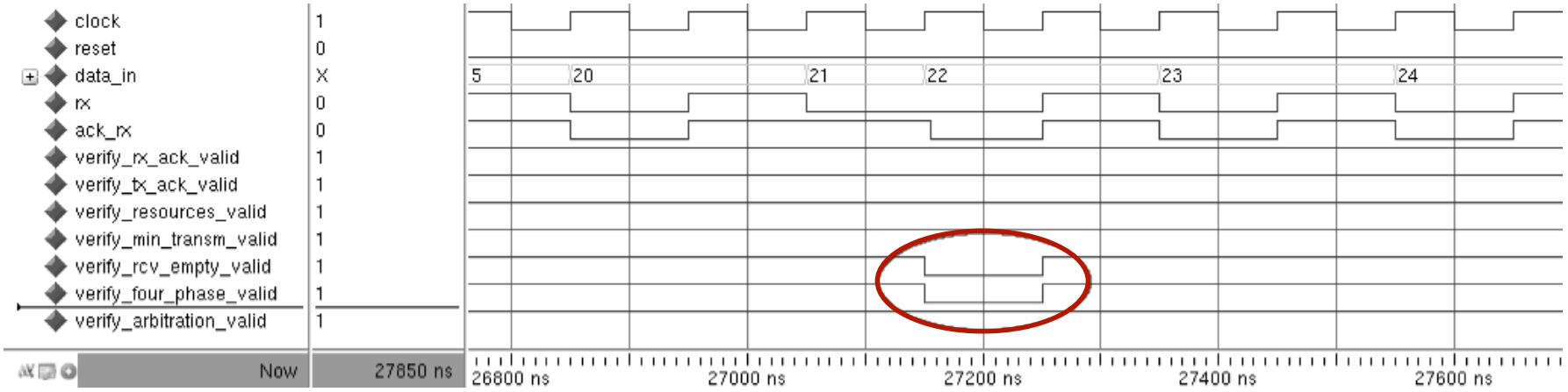
Experimental results

- Instrumentation of NoCs with PSL properties
  - Nostrum: 39 assertions
  - Hermes: 30 assertions
- Properties automatically transformed into synthesizable verification components (*)
  - Can be used within the simulation or FPGA prototyping procedures: for debug during NoC design
  - Or can be used as embedded verification components (ASIC/FPGA synthesis): for the online verification of safety requirements

(*) TIMA « HORUS » technology (Y.Oddos, K.Morin-Allory, D.Borrione: "Assertion-Based Design with Horus", Proc. MEMOCODE'2008) integrated into Dolphin EDA tools: http://www.dolphin.fr/medal/sled/segment/sled_sdg.php

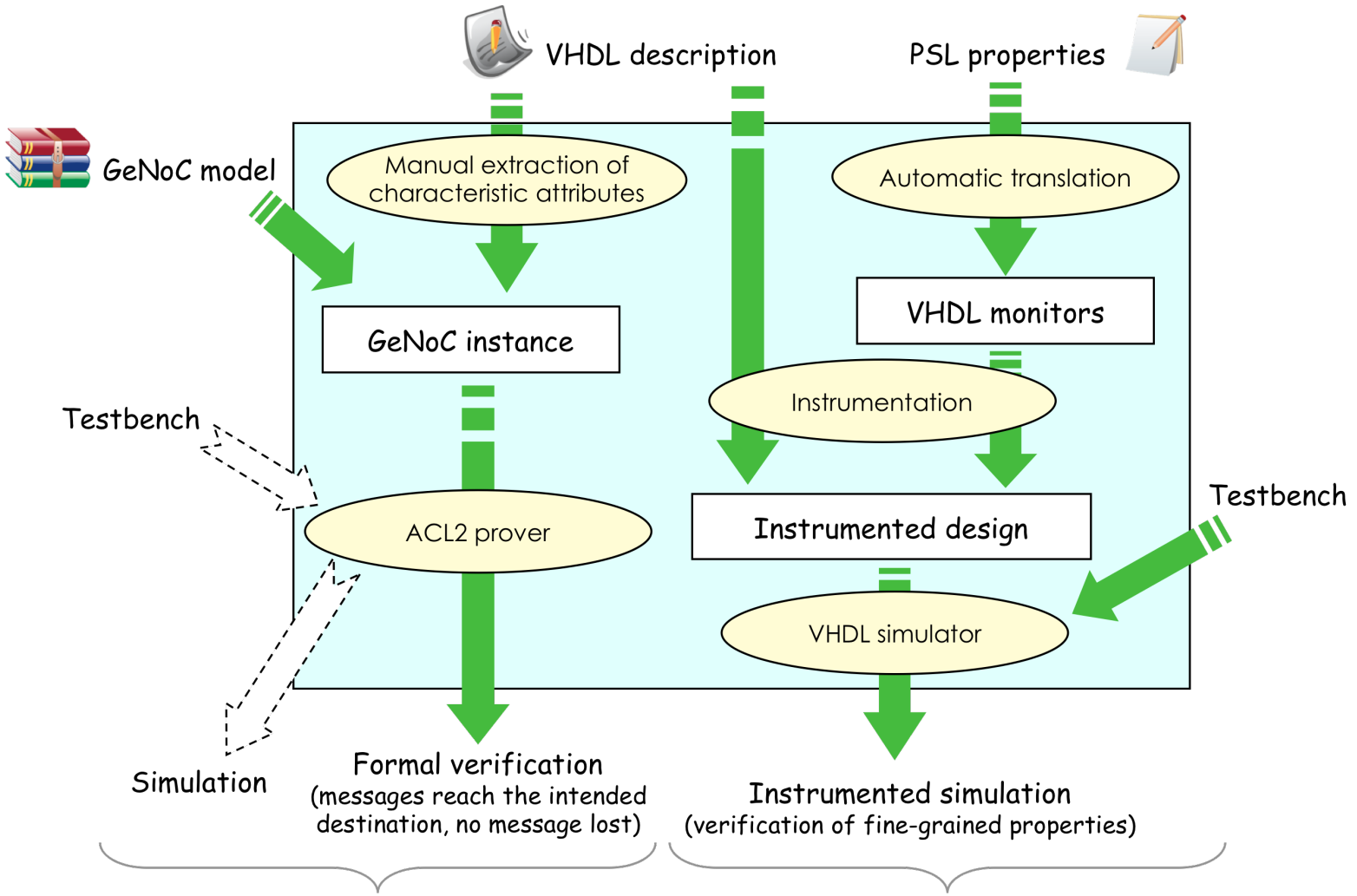
Experimental Results

- Simulation (with fault injection)



- FPGA synthesis (Xilinx Virtex 5)
  - Example: 4 x 4 Hermes with 7 monitors on the 6 central routers
    - Without monitors: 21173 LUT, 64 MHz
    - With monitors: 22581 LUT (+ 6.65%), 48.25 MHz (- 24.6%)

Conclusion – Overall approach



CONCLUSION

- Multi-level verification solution
  - Experimented on:
    - Nostrum: 199 ACL2 theorems (proof obligations and auxiliary theorems), 39 PSL properties
    - Hermes: 272 ACL2 theorems, 30 PSL properties
- Future work:
  - Embedded verification components

    → need specific synthesis optimisations, and specific facilities for collecting relevant diagnosis information
  - Runtime Assertion-Based Verification at the system level (NoC infrastructure in a SoC, SystemC TLM)

Thanks for your attention…