Towards A Formally Verified Network-on-Chip

Tom van den Broek$^1$ Julien Schmaltz$^{1,2}$

$^1$Institute for Computing and Information Sciences
Radboud University Nijmegen
The Netherlands

$^2$School of Computer Science
Open University
The Netherlands

t.vandenbroek@cs.ru.nl & julien.schmaltz@ou.nl

FMCAD ’09
Networks-on-Chips: Hermes

- Implemented as model instance

- Characteristics:
  - XY minimal deterministic routing
  - Wormhole switching
  - Frame structure:
    - Header flit (Route Information)
    - Data flits (Payload)
    - Torn-down flit (Last flit)
Platform-Based Design and Networks-on-Chip

- Platform-Based Design:
  - Re-use of parametric modules (*Intellectual Properties*)
  - High-level of abstraction
  - *Communication-centric*: from buses to networks

- Solves the communication issues
- The components are connected in a communication network

- Advantages
  - Scalable
  - Parallelism
Formal Methods and Networks-on-Chips

- **System Verification:**
  - Proof of each component
  - Proof of their *interconnection*

- **State-of-the-Art:**
  - Model checking or theorem proving of *instances* of systems
  - Often at *hardware* level (RTL)

- **The GeNoC Approach:**
  - A *generic* model for reasoning about NoCs
  - Reduces amount of the user interaction needed to prove properties on NoC instances
**GeNoC approach**

THEOREM
messages reach their expected destination

To be discharged for the given NoC
Instantiated for the given NoC
Contribution

- **Original GeNoC Model**
  - Highly abstract representation of the communications
  - The model has access to the complete precomputed routes the messages will traverse in the network
  - How does the specification level relates to the implementation level?

- **Contribution**
  - A generic implementation model
  - A (generic) specification model
  - A refinement proof between two instances of these models
Method - Specification model

Outline

Contribution and Method

Method - Specification model

Implementation Model

Instance

PO

Theorem

Specification Model

Instantiate

Network Characteristics

Topology Flowcontrol

Datalink Routinglogic

Input
Method - Contribution

Outline
Contribution and Method

van den Broek et al. (RUN & OU ) Towards A Formally Verified NoC November 18, 2009
Structure of the two models

Both models consist of two main parts:

- The NoC characteristics are defined in the **Network model**
  - Topology
  - Router components:
    - Datalink
    - Routing
    - Scheduling
- The **Network interpreter** takes a network model and simulates the network
- Implemented in ACL2
The main interpreter structure

- State
- Messages
- Depart
- Updated state
- Stepnetwork
  - Router
  - UpdateNeighbours
- Delayed

van den Broek et al. (RUN & OU)
Towards A Formally Verified NoC
November 18, 2009
Network interpreter – Implementation Level
Network interpreter – Implementation Level

local ports

north

local ports

east

south

local ports

west

0 0

1 0

M

365 36 1

10 1 1

0 1

1 1

local ports

north

local ports

south

north

local ports

north

south

north

local ports

east

east

west

west

Stepnetwork

Router

UpdateNeighbours

State

Messages

Updated state

delayed

Depart

van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Models

Network interpreter – Implementation Level

local ports

north

local ports

north

local ports

north

local ports

north

local ports

south

local ports

south

local ports

south

local ports

south

0 0 1 0

1 10 1

M

0 1

1 1

0 0

1 0

Depart

Stepnetwork

Router

UpdateNeighbours

State

Messages

Updated state

delayed

ProcessInputs
Network interpreter – Implementation Level

local ports

0 0

1 0

0 1

1 1

local ports

south

north

west east

local ports

south

north

west east

local ports

south

north

west east

local ports

south

north

west east

0 0

1 0

0 1

1 1

depart

stepnetwork

router

update neighbours

state messages

updated state
delayed

stepnetwork

router

update neighbours

routecontrol

van den Broek et al. (RUN & OU)
Towards A Formally Verified NoC
November 18, 2009
Network interpreter – Implementation Level

FlowControl
Network interpreter – Implementation Level

ProcessOutputs
Network interpreter – Implementation Level

local ports

0 0 1 0
1 10 1
M

local ports

south
north
west east
local ports
south
north
west east
local ports
south
north
west east
local ports
south
north
west east

Depart
Stepnetwork
Router
UpdateNeighbours
State Messages
Updated state
delayed

van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Network interpreter – Implementation Level
Network interpreter – Specification Level
Network interpreter – Specification Level

van den Broek et al. (RUN & OU ) Towards A Formally Verified NoC November 18, 2009
Network interpreter – Specification Level

local ports → north
west → 0 1

east → north
local ports

south → local ports
west

0 0 1 0

MENL

1 10 1

R-Depart

Stepnetwork
Spec-Router

UpdateNeighbours

State
Messages

Updated state
delayed

Spec-Router

UpdateNeighbours

van den Broek et al. (RUN & OU ) Towards A Formally Verified NoC November 18, 2009
Network interpreter – Specification Level

local ports

north

local ports

south

local ports

east

0 0

1 1

M NL

0 0

1 0

State

Messages

R-Depart

delayed

Updated state

Spec-Router

Stepnetwork

UpdateNeighbours

van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Network interpreter – Specification Level

local ports

north
east

south

west
east

local ports

north

south

local ports

north

west
east

local ports

south

north

local ports

south

west
east

local ports

south

0 0 1 0
1 10 1
M NL

R-Depart
Stepnetwork
Spec-Router
UpdateNeighbours
State Messages
Updated state
delayed

Stepnetwork
Spec-Router
UpdateNeighbours
van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Network interpreter – Specification Level

local ports

0 0 1 0

0 1

1 1

M NL

0 0

R-Depart

Stepnetwork

Spec-Router

UpdateNeighbours

State Messages

Updated state
delayed

delayed

Updated state

Spec-Router

UpdateNeighbours

van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Network interpreter – Specification Level

local ports

north
east

south
west

east
west

0 1

1 1

0 0

1 0

ML

State
Messages

R-Depart

delayed

Updated state

Spec-Router

Stepnetwork

UpdateNeighbours

van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Network interpreter – Specification Level

local ports

north

local ports

south

east

west

delayed

R-Depart

State

Messages

M L

Updated state

Stepnetwork

Spec-Router

UpdateNeighbours

van den Broek et al. (RUN & OU )

Towards A Formally Verified NoC

November 18, 2009
The implementation model is a refinement of the specification model:

1. Given the same input the models should produce the same output.
2. The messages should traverse the same paths in the network.
The implementation model is a refinement of the specification model:

1. Given the same input, the models should produce the same output.
2. The messages should traverse the same paths in the network.

The Transform relation removes the routes from the network state.
∀ state, transactions :
\[ \text{transform}(\text{GeNoC}_S(state, transactions)) = \text{GeNoC}_I(state, transactions) \]

\text{GeNoC}_I \text{ and } \text{GeNoC}_S \text{ return a tuple of } (\text{arrived}, \text{delayed}, \text{trace}) \text{ so this theorem can be read as:}

1. The transformed arrived messages are equal
2. Delayed messages are equal
3. The transformed simulation trace is the same
Proof - Structure

Main theorem

Pedicates

delayed
arrived
ntkst
ntkmem
accup
Valid-routes

stepnetwork

step-routing

routinglogic-eq-next-hop

Valid-routes

correct-routing
good-switch
good-ntkst
Example theorem - Routinglogic-eq-next-hop

\[ \forall msg : validRoute(msg) \implies computeRoute(cur(msg))(dest(msg)) = getNextHop(msg) \]

This theorem states:
A message with a valid route implies that computing the next step in the route is equal to extracting it from the precomputed route.
## Proof - Statistics

<table>
<thead>
<tr>
<th>Group</th>
<th>number of Theorems</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changed functions</td>
<td>72</td>
</tr>
<tr>
<td>Predicates</td>
<td>140</td>
</tr>
<tr>
<td>Not changed functions</td>
<td>88</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>300</strong></td>
</tr>
</tbody>
</table>

The source code of the proofs and models is available on the web. 

Conclusion - overview

Implementation Model → PO → Instance

refine. Theorem

Specification Model → PO → Instance

Instance → Network Characteristics

Topology, Flowcontrol, Datalink, Routinglogic

Input → PO
Conclusion - contributions

The contributions are:

- First cross-layer verification attempt of a NoC
- A realistic generic implementation model
- Multiple implementation instances of real NoCs
  - Packet, circuit, and wormhole switching
  - XY and Spidergon routing
  - Hermes NoC
  - Octagon NoC
- Instance of a NoC at the specification level
- Refinement proof between two instances
Current and future research directions:

- A generic cross-layer verification method
- Proof between two generic models at two different levels
- More instances of different NoCs
- Integration of deadlock and liveness properties (Verbeek & Schmaltz ACL2 ’09 and DATE ’10)
- Extending the number of layers
  - Towards RTL
  - Layer with “Source” and “Distributed scheduling”
Thank you for listening!
Network Model – Generic Router

- Input Stage
  - Routing Control
  - Flow Control
- Output Stage

Table:

<table>
<thead>
<tr>
<th>Port</th>
<th>Address</th>
<th>Id</th>
<th>Port Name</th>
<th>Direction</th>
</tr>
</thead>
</table>

Data Input

Status Field

Buffer

Data Output

Status Field

Buffer

van den Broek et al. (RUN & OU ) Towards A Formally Verified NoC November 18, 2009
Wormhole switching and XY Routing

Diagram showing the different switching patterns for Wormhole and XY routing.

- 00
- 01
- 10
- 11

Local ports:
- South
- North
- West
- East

Routing examples:
- 0 0 1 0
- 1 1 0 1

Reference:
van den Broek et al. (RUN & OU ) Towards A Formally Verified NoC November 18, 2009
Wormhole switching and XY Routing

van den Broek et al. (RUN & OU ) Towards A Formally Verified NoC November 18, 2009
Appendix

Wormhole switching and XY Routing

van den Broek et al. (RUN & OU )  Towards A Formally Verified NoC  November 18, 2009
Wormhole switching and XY Routing

van den Broek et al. (RUN & OU ) Towards A Formally Verified NoC November 18, 2009
Wormhole switching and XY Routing

local ports

north

west east

local ports

south

north

west east

local ports

south

north

west east

local ports

south

north

west east

local ports

south

north

west east

0 0 1 0

1 10 1

T

booked

booked

booked

D booked

D

H

van den Broek et al. (RUN & OU)
Towards A Formally Verified NoC
November 18, 2009
Wormhole switching and XY Routing

van den Broek et al. (RUN & OU )   Towards A Formally Verified NoC   November 18, 2009
Wormhole switching and XY Routing

local ports

north

west east

south

0 0 1 0

1 10 1

booked

D

H

booked

booked

D

T

booked

van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Wormhole switching and XY Routing

van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Wormhole switching and XY Routing

0 0 1 0
1 1 0 1
booked
d T

van den Broek et al. (RUN & OU) Towards A Formally Verified NoC November 18, 2009
Wormhole switching and XY Routing

van den Broek et al. (RUN & OU)  Towards A Formally Verified NoC  November 18, 2009
Wormhole switching and XY Routing

local ports

0 0 1 0
1 1 0 1

van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Appendix

Circuit Switching

![Circuit Switching Diagram]

van den Broek et al. (RUN & OU)  
Towards A Formally Verified NoC  
November 18, 2009
Circuit Switching
Circuit Switching

van den Broek et al. (RUN & OU) Towards A Formally Verified NoC November 18, 2009
Circuit Switching

![Circuit Switching Diagram]
Circuit Switching

0 0 1 0
1 10 1
m
req
req req
req
req
booked

versus

west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west
east
south
north
west

van den Broek et al. (RUN & OU )
Towards A Formally Verified NoC
November 18, 2009
Circuit Switching

van den Broek et al. (RUN & OU ) Towards A Formally Verified NoC November 18, 2009
Circuit Switching
Appendix

Circuit Switching

van den Broek et al. (RUN & OU ) Towards A Formally Verified NoC November 18, 2009
Circuit Switching

- Local ports: south, north, west, east
- Booked status: m, req, ack
- Network connections:
  - West to north
  - South to north
  - West to south
  - East to south

Diagram showing the state transitions and connections in a circuit switching network.