Fast and Accurate Transaction-Level Model of a Wormhole Network-on-Chip with Priority Preemptive Virtual Channel Arbitration

Leandro Soares Indrusiak, Osmar Marchi dos Santos

Real Time Systems Group (RTS)
Computer Science Department
University of York – United Kingdom

lsi@cs.york.ac.uk
http://www.cs.york.ac.uk/~lsi
Outline

- Motivation and related work
- Wormhole NoC
- Priority-preemptive virtual channels
- Transaction-level modelling
- Experimental results
- Conclusions
Motivation

Context
- Embedded Multicores
- Tasks mapped over PEs interconnected with a wormhole NoC

Problem
- Estimate latency of inter-task communication over NoCs
- Cycle-accurate simulation can take hours/days to simulate a few seconds of target time
Related work

- Transaction-level modelling (TLM) can reduce simulation time by abstracting away details of the interconnect architecture
  - trade-off between simulation speed and accuracy
- TLM models for simple processing and communication architectures can provide orders of magnitude of simulation speed-up with acceptable accuracy
  - situation is not quite the same for complex multicores based on NoCs

<table>
<thead>
<tr>
<th>NoC TLM approach</th>
<th>authors</th>
<th>speed-up</th>
<th>accuracy with regard to cycle-accurate</th>
</tr>
</thead>
<tbody>
<tr>
<td>multiple simulation time references</td>
<td>Hosseinabady et al</td>
<td>1.3x</td>
<td>100%</td>
</tr>
<tr>
<td>packet payload abstraction</td>
<td>Ost et al</td>
<td>2-6x</td>
<td>up to 95%</td>
</tr>
<tr>
<td>multiple simulation time references, neglects NoC internal contention</td>
<td>Viaud et al</td>
<td>50x</td>
<td>99.999%</td>
</tr>
</tbody>
</table>
Wormhole Networks-on-Chip

- Common switching technique for NoCs due to low buffer overhead
  - routers do not need to buffer a complete packet

- Packet is forwarded from one router to the next as soon as its header flit has arrived
  - packet can be temporarily stored in several routers simultaneously

- If there is no contention, basic packet latency is deterministic

- Major Drawback
  - network contention results in latency variability
Wormhole Networks-on-Chip
Wormhole Networks-on-Chip

- Packet Header
- Packet Data
Wormhole Networks-on-Chip

Packet Header
Packet Data

PE
Wormhole Networks-on-Chip

- Packet Header
- Packet Data

DATE 2011 - Grenoble
Wormhole Networks-on-Chip

- Packet Header
- Packet Data

DATE 2011 - Grenoble
Wormhole Networks-on-Chip

Packet is blocked

Packet Header
Packet Data

PE

R

R

R

R

R

PE
Wormhole Networks-on-Chip

Packet Header
Packet Data

R

PE

PE
Wormhole Networks-on-Chip

Packet Header
Packet Data
PE
Wormhole Networks-on-Chip

- Packet Header
- Packet Data

new packet released
Wormhole Networks-on-Chip

- Packet Header
- Packet Data

PE

DATE 2011 - Grenoble
Wormhole Networks-on-Chip

Packet Header
Packet Data

PE
Wormhole Networks-on-Chip

- Packet Header
- Packet Data

DATE 2011 - Grenoble
Wormhole Networks-on-Chip

- Packet Header
- Packet Data

PE
Wormhole Networks-on-Chip

- TLM models of wormhole NoCs tend to have very low accuracy for inter-task communication latency
  - blocking behaviour depends of fine-grained events which are not accurately captured at transaction level

- Alternative: wormhole NoCs with arbitration and switching techniques that can be accurately modelled at transaction level
  - not significantly affected by events at the clock cycle granularity

our choice:
  - virtual channels with priority preemptive arbitration
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data

PE
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data

high priority packet released
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data

PE

PE
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data

PE

PE
Priority preemptive virtual channels

Packet Header
Packet Data

wormhole NoC with priority preemptive virtual channels

first packet is preempted
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data
PE
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header

Packet Data

PE

PE
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data

PE

PE
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data
PE
PE
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data

PE

PE
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data

PE

PE
Priority preemptive virtual channels

wormhole NoC with priority preemptive virtual channels

Packet Header
Packet Data
Priority preemptive virtual channels

- Not a new idea

- Supported by existing NoC architectures such as QNoC (Bolotin et al 2004) and HERMES (Vieira de Mello et al 2005)

- Existing static analysis technique (Shi and Burns 2008) can determine worst case latency for such architectures if application follows the sporadic task model

- Simulation is still needed to determine latency figures in the general case
Transaction level modelling

- Proposed approach: an approximately timed transaction-level model of a wormhole NoC with priority preemptive virtual channels
  - PEs are transaction initiators and targets
  - transactions represent packets
  - the whole NoC is modelled as an interconnect component
Transaction level modelling

- Proposed TLM uses interference analysis to calculate latency of each communication flow

\[ \text{pri(t1)} > \text{pri(t2)} > \text{pri(t3)} > \text{pri(t4)} \]
Transaction level modelling

- Interference analysis

\[ \text{pri}(t_1) > \text{pri}(t_2) > \text{pri}(t_3) > \text{pri}(t_4) \]
Transaction level modelling

- Interference analysis
Transaction level modelling

- Interference analysis
Transaction level modelling

- Interference analysis
Transaction level modelling

- Interference analysis
Transaction level modelling

- Interference analysis
Transaction level modelling

- Simulation speed-up due to reduction of simulation events
  - system is simulated only when transactions initiate or terminate
  - at those points, graph is updated and next event is scheduled (earliest termination)
  - if new transactions are initiated in the meantime, update the graph and recalculate the next event if needed

  - simulation time increases with the number of flows and with the interference between them
  - simulation time is independent of packet size
Experimental results

- Synthetic applications
- 4x4 Mesh
- Scaling the number of flows

<table>
<thead>
<tr>
<th>application flows</th>
<th>simulation time (s) cycle-accurate</th>
<th>TLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>17105.42</td>
<td>5.47</td>
</tr>
<tr>
<td>40</td>
<td>31490.13</td>
<td>19.17</td>
</tr>
<tr>
<td>60</td>
<td>55561.87</td>
<td>45.78</td>
</tr>
<tr>
<td>80</td>
<td>70231.76</td>
<td>83.06</td>
</tr>
<tr>
<td>100</td>
<td>86626.07</td>
<td>105.28</td>
</tr>
</tbody>
</table>

![Graph showing simulation time versus application flows for cycle-accurate and TLM methods.]
Experimental results

- Synthetic applications
- 4x4 Mesh
- Scaling the number of flows

<table>
<thead>
<tr>
<th>application flows</th>
<th>simulation time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>cycle-accurate</td>
</tr>
<tr>
<td>20</td>
<td>17105.42</td>
</tr>
<tr>
<td>40</td>
<td>31490.13</td>
</tr>
<tr>
<td>60</td>
<td>55561.87</td>
</tr>
<tr>
<td>80</td>
<td>70231.76</td>
</tr>
<tr>
<td>100</td>
<td>86626.07</td>
</tr>
</tbody>
</table>
Experimental results

- Autonomous vehicle testbench – 33 tasks, 38 traffic flows
- 4x4 Mesh
- Scaling application target time

<table>
<thead>
<tr>
<th>application time (s)</th>
<th>simulation time (s)</th>
<th>cycle-accurate</th>
<th>TLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1465.18</td>
<td>7.67</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2895.69</td>
<td>7.89</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5978.32</td>
<td>7.98</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>11806.91</td>
<td>8.28</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>29927.24</td>
<td>9.77</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>1465.18</td>
<td>7.67</td>
<td></td>
</tr>
<tr>
<td>800</td>
<td>29927.24</td>
<td>9.77</td>
<td></td>
</tr>
<tr>
<td>1400</td>
<td>169.06</td>
<td>7.98</td>
<td></td>
</tr>
</tbody>
</table>

![Graph showing simulation time vs. application time](image)
Experimental results

- Autonomous vehicle testbench – 33 tasks, 38 traffic flows
- 4x4 Mesh
- Scaling application target time

<table>
<thead>
<tr>
<th>application time (s)</th>
<th>cycle-accurate simulation time (s)</th>
<th>TLM simulation time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1465.18</td>
<td>7.67</td>
</tr>
<tr>
<td>2</td>
<td>2895.69</td>
<td>7.89</td>
</tr>
<tr>
<td>4</td>
<td>5978.32</td>
<td>7.98</td>
</tr>
<tr>
<td>8</td>
<td>11806.91</td>
<td>8.28</td>
</tr>
<tr>
<td>20</td>
<td>29927.24</td>
<td>9.77</td>
</tr>
<tr>
<td>200</td>
<td>26.55</td>
<td>26.55</td>
</tr>
<tr>
<td>800</td>
<td>86.29</td>
<td>86.29</td>
</tr>
<tr>
<td>1400</td>
<td>169.06</td>
<td>169.06</td>
</tr>
</tbody>
</table>
Experimental results

- Comparing with a cycle-accurate model, the proposed transaction-level model produced worst-case latency figures with a maximum percent difference of 6.25%.
- Best case and average case latency had a percent difference below 1%.
Experimental results

- Comparing with a cycle-accurate model, the proposed transaction-level model produced worst-case latency figures with a maximum percent difference of 6.25%.
- Best case and average case latency had a percent difference below 1%.
Experimental results

- Comparing with a cycle-accurate model, the proposed transaction-level model produced worst-case latency figures with a maximum percent difference of 6.25%.
- Best case and average case latency had a percent difference below 1%.
Experimental results

- Comparing with a cycle-accurate model, the proposed transaction-level model produced worst-case latency figures with a maximum percent difference of 6.25%.
- Best case and average case latency had a percent difference below 1%.
Experimental results

- Comparing with a cycle-accurate model, the proposed transaction-level model produced worst-case latency figures with a maximum percent difference of 6.25%.
- Best case and average case latency had a percent difference below 1%.
Experimental results

- Comparing with a cycle-accurate model, the proposed transaction-level model produced worst-case latency figures with a maximum percent difference of 6.25%.
- Best case and average case latency had a percent difference below 1%.
Conclusions

- Identified a particular NoC architecture that can be accurately modelled at transaction level

- Experiments with the transaction-level model show three orders of magnitude simulation speed-up with less than 10% accuracy loss with regard to a cycle-accurate counterpart

- Valuable tool to support design space exploration of embedded multicore systems
  - NoC topology, routing, link width, clock frequency
  - task mapping and priority assignment
Fast and Accurate Transaction-Level Model of a Wormhole Network-on-Chip with Priority Preemptive Virtual Channel Arbitration

Leandro Soares Indrusiak, Osmar Marchi dos Santos

Real Time Systems Group (RTS)
Computer Science Department
University of York – United Kingdom

lsi@cs.york.ac.uk
http://www.cs.york.ac.uk/~lsi