Analysis of Modeling Styles on Network-on-Chip Simulation

Lasse Lehtonen, Erno Salminen, Timo D. Hämäläinen
Tampere University of Technology Department of Computer Systems,
P.O.Box 553, FIN-33101 Tampere, Finland
lasse.lehtonen@tut.fi

***Abstract*—This paper analyses the effects of Network-on-Chip (NoC) models written in SystemC on simulation speed. Two Register Transfer Level (RTL) models and Approximately Timed (AT) and Loosely Timed (LT) Transaction Level (TL) models are compared against reference RTL VHDL 2D mesh model. Three different mesh sizes are evaluated using a commercial simulator and OSCI SystemC reference kernel. Studied AT model achieved 13-40x speedup with modest 10% estimation error.**

I. Introduction

Today's System-on-Chips (SoCs) consists of many Intellectual Property (IP) components, such as Processing Elements (PEs), memories and Network-on-Chip (NoC) components executing complex applications.

As SoCs keep growing larger the design space grows even more rapidly. Larger design space implies longer simulations and more simulation runs for Design Space Exploration (DSE) to find near optimum scheduling, configurations and mappings for IP blocks and the NoC connecting them.

Simulation performance is one of the key issues in verification and DSE. Fast and accurate estimates are required early in products design process to provide information of important factors, such as performance, area and power consumption [3].

Simulation performance is a sum of many different factors, such as the choise of modeling language, simulator tool, abstraction level and the quality of the description.

One widely used method to improve simulation performance is to raise the abstraction level. Register Transfer Level (RTL) models have to describe every signal and their behaviour between clock edges to be synthesizeable which causes significant overhead when such a precision is not needed. Transaction Level (TL) modeling abstracts detailed signaling from communication events using abstract function calls to annotate only the significant signaling events.

Transaction Level Models (TLMs) can be modelled using different abstraction levels. Untimed models offer best speedup but lack timing annotation totally. On the other hand cycle accurate models annotate time every clock cycle but have considerably smaller speedup in general.

This paper presents a study regarding modeling styles for NoC models consentrating on the raise of abstraction level. Results show that the implemented Approximately Timed (AT) TLM 2.0 model offers a notable speedup (ranging from 13x to 40x) with modest latency estimation error (under 10% on average).

II. Related Work

Simulation speed of hardware models can be accelerated in many ways. There are multiple languages to choose from for describing hardware models, most used being VHDL, Verilog, SystemVerilog and SystemC. According to [1] the choise of language and tools have significant impact on simulation speed. Most commercial simulators were found to be optimized for one language only and having usually more than 1.8x difference in simulation times for tool's slower languages. In one extreme case a 10x difference in simulation time was measured between two commercial simulators for the same model.

Thorough evaluation of VHDL, Verilog, SystemVerilog and SystemC implemented with different abstraction levels and data types was performed in [1] using three commercially available simulators and gcc compiler. Verilog was found to be 2x faster than VHDL and SystemC 10x slower on average for RTL models. SystemC TLM was 2.6x slower than SystemVerilog TLM Programmer's View (PV) model. SystemVerilog was fastest to simulate in all abstraction levels on average.

Speedup for simulation times can be gained through model optimization, transformation and reduction [5]. Model optimization is a straightforward process of removing unnecessary or replacing inefficient code constructs with more efficient ones for the used simulator without modifying model's structure. Model transformation means changing language constructs into functionally equivalent ones offering better performance in simulation time. Model reduction comprises the process of removing all unrelevant parts of the model for a specific simulation run.

In [5] simulation performance was reported to improve significantly by altering the source code. Simulation speedup measured compilation times included was usually in range from 1.1x to 10x depending on the used technique. Although these measurements were done about ten years ago, measurements in [1] implies that commercial simulators don't perform all transformation optimizations automatically or effectively. Hierarchically modelled designs were found in [1] to be on average 1.5x slower than their flat counterparts and different value types affecting the simulation time with up to 4x factor.

TLM modeling techniques have been studied widely. For example Proteo NoC architecture [10] used VHDL to construct

978-1-4244-8973-2/10/$26.00 ©2010 IEEE

higher level model achieving an order of magnitude speedup for a 16 node NoC.

Shirner et al. [9] created two TLM models for AMBA bus and compared them against synthesizeable Bus Functional Model version. More abstract TLM model was four orders of magnitude faster with error up to 45% and the more accurate TLM reached two order of magnitude speedup with an error of 35% in worst cases.

Another research to speed up shared bus architecture explorations in [7] used abstraction level between TLM and Bus Cycle Accurate (BCA) to create accurate models offering a 1.55x speedup. OSCI (Open SystemC Initiative) TLM 2.0 based methodology to create BCA shared bus protocol models that retain the same level of accuracy as RTL models was introduced in [11]. Their BCA model was measured to be between one and two orders of magnitude faster than RTL model.

III. Analysis Environment

Analysis is performed for 2-dimensional mesh topology (Figure 1). It uses 6 word fifos on links, wormhole switching and XY routing with fixed priority arbitration. This paper focuses on using SystemC in three different abstraction levels to gain better performance in simulation speed compared to the reference synthesizeable RTL VHDL model.

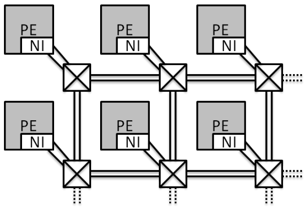

Figure 1: Mesh 2D NoC topology

A. Network-on-Chip Models

2D Mesh was implemented in synthesizeable RTL VHDL and SystemC. Two SystemC RTL models were implemented using VHDL coding style following the same structure as the reference VHDL model. First RTL version used 4-state logic and second 2-state logic. In addition, two SystemC TLM versions were created using OSCI TLM 2.0 [6] function calls with generic payload.

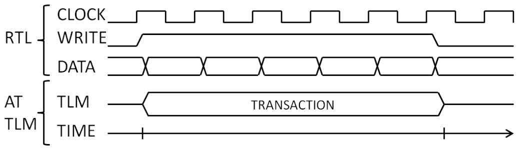

Figure 2: Conseptual difference between RTL and TLM.

First model (called TLM AT) was written in Approximately Timed (AT) coding style using two timing points to annotate the start and end of transactions (Figure 2). Two SystemC processes and queues were used to model network interfaces. Router components instantiate one process and a priority queue per incoming link. Model was calibrated for 4 word packets and the error in average latencies was under 10%.

Second, Loosely Timed (LT) TLM (TLM LT) model was made using same network interface models but abstracting all routers using only one SystemC process. This implementation was left nearly untimed as the main interest was in simulation performance. A method to estimate latencies in one process for wormhole switched mesh topology was described in [4]. Estimation error depended linearly on the network utilization. They measured 45% deviation in average latencies for saturation load but on lower loads the estimation error was in more acceptable ranges for DSE.

B. Transaction Generator

Transaction Generator (TG) [8], a freely available SystemC traffic generator for NoC benchmarking and DSE was used to generate traffic for NoC models. TG uses abstract application and platform models to mimic traffic patterns captured from real data flow applications (Figure 4). It's implemented using SystemC events and waits.

Network traffic created by TG is modelled in Transaction Level using a simple custom interface for RTL NoC models and OSCI TLM 2.0 function calls with generic payload for TLM models.

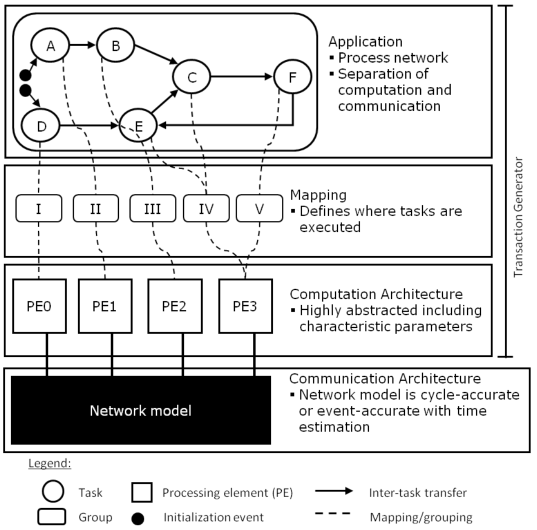
A
B
C
F
D
E
Application
▪ Process network
▪ Separation of computation and communication
I
II
III
IV
V
Mapping
▪ Defines where tasks are executed
PE0
PE1
PE2
PE3
Computation Architecture
▪ Highly abstracted including characteristic parameters
Transaction Generator
Network model
Communication Architecture
▪ Network model is cycle-accurate or event-accurate with time estimation
Legend:
Task
Group
Processing element (PE)
Initialization event
Inter-task transfer
Mapping/grouping

Figure 4: Transaction Generator uses abstract application and processing element models to mimic the computation of real SoCs to create accurate traffic

C. Measurement

All measurements were executed on a modern workstation with E8400 dual core processor, 3 GB RAM and 32-bit Windows XP. Mixed-language simulation was run with a commercial simulator. SystemC-only simulations were run with both the commercial simulator and OSCI SystemC reference kernel compiled with GNU gcc in Cygwin. Optimization level -O3 was used with gcc and similar optimization were enabled for the commercial simulator. No signal trace was gathered

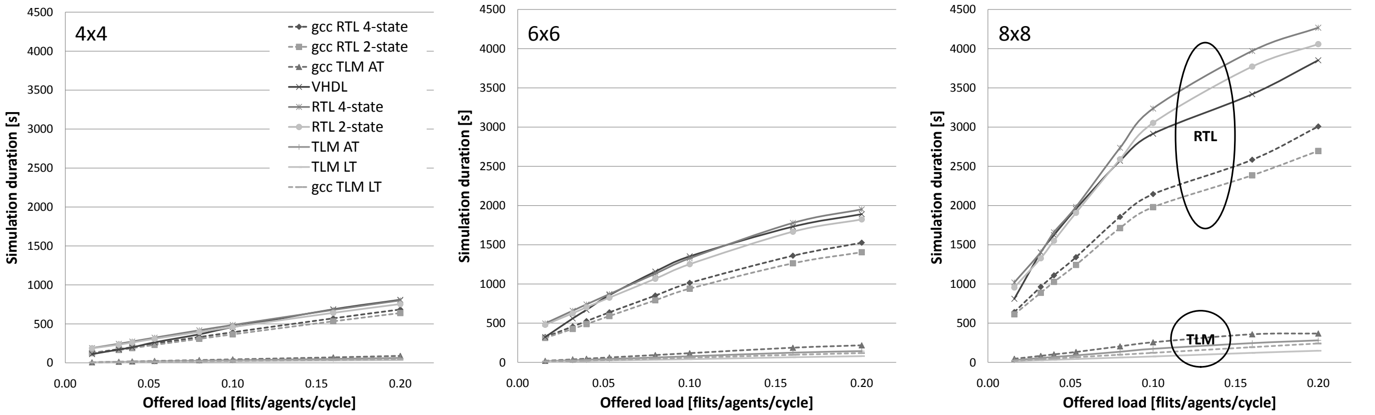

Figure 3: Simulation times for 4x4, 6x6 and 8x8 meshes (4 word payload). Time increases with the amount of data sent. The difference between RTL and TLMs is clearly seen. Gcc compiled OSCI reference kernel simulations are marked with gcc prefix and dashed lines.

and commercial simulator was run on console mode during simulations.

Frequency of 50 MHz was used for NoC and processing elements and simulations were run for 100 ms. A deterministic all-to-all traffic pattern was used generating the load.

Measured simulation times are for whole program so the speedups shown are not only for the NoC models. Measurements were performed more than twice to exclude variations in workload affecting results. Table I lists the parameters that were varied on different load conditions.

Table I: Summary of measured parameters

Simulator	NoC Size	Tx Size	NoC Model
Commercial	4x4, 6x6, 8x8	4, 20, 40	3 RTL, 2 TLM
OSCI	4x4, 6x6, 8x8	4, 20, 40	2 RTL, 2 TLM

IV. Results

A. Size of the NoC

NoC sizes of 16, 36 and 64 routers were measured. Size of the NoC affects nearly linearly the simulation time. With the used traffic pattern the simulation speed for RTL models decreases more than the increase in the number of mesh nodes would imply. This is because in all-to-all traffic pattern on bigger mesh means on average more hops to target and thus more cycles to simulate for the same offered load. 6x6 mesh simulated 2.6x and 8x8 5.4x slower than 4x4 mesh on average as shown in figure 3.

TLM models scaled better. On average 6x6 mesh TLM models were 2.2x and 8x8 mesh models 4.6x slower than 4x4 mesh models.

B. Size of the Transfer

Transfer legths depends on the application and the speedup grows with the transfer length for both the RTL and TLM models. By using 20 word transfers SystemC RTL simulation was approximately 2x faster on same load levels than with 4 word payload. VHDL model was 2.5x faster. SystemC RTL was 2.3x faster when the size was increased to 40 words and VHDL 3.2x faster.

Due to the fact that TLM model processes activate only when transaction begins and ends a great increase in simulation speed was achieved with bigger transfer sizes.

Simulations with gcc compiled OSCI reference SystemC kernel were on average 4.5x faster for 20 word transfers and 3.5x faster for the commercial simulator when compared to 4 word transfers on same load conditions. Increasing size to 40 words brought 8.5x speedup for gcc and 6.5x speedup for the commercial simulator.

When compared to 6x6 VHDL RTL model with 40 word transfers the AT model offered on 30x-35x speedup and the one process model 55x-60x speedup (figure 5) on average. With 20 word transfers same speedups were 22x-28x and 40x-45x.

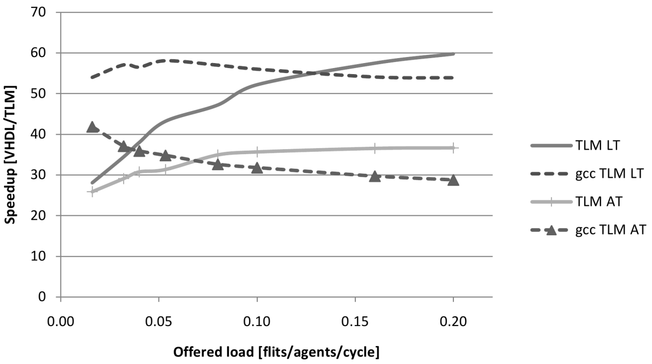

Figure 5: TLM Speedups compared to RTL VHDL (40 word payload, 6x6 mesh)

C. SystemC Logic Levels

Difference in simulation times between 4-state and 2-state SystemC logic values was small. Gcc simulations were approximately 7% faster using 2-state values on all measurements except cases with very low utilization. With commercial simulator the difference was even smaller ranging from 5% to 2% depending on network utilization.

D. SystemC Models

For RTL models simulated with the commercial simulator only small difference was between VHDL and SystemC performance. Largest measured difference in times was 16%. SystemC gcc version using 4-state values simulated 10-30% and 2-state version 13-50% faster than VHDL depending on the network size and utilization.

Figure 6 shows TLM speedups over RTL VHDL model measured with the commercial simulator. Speedup was significantly larger for bigger meshes on low utilization scenarios. When mesh utilization got closer to saturation the difference in speedup between mesh sizes decreased. Approximately timed model was 13x-15x and loosely timed model 20x-30x faster on near saturation load. With gcc compiled version TLM speedups followed the same trend but were on average 1.5x slower.

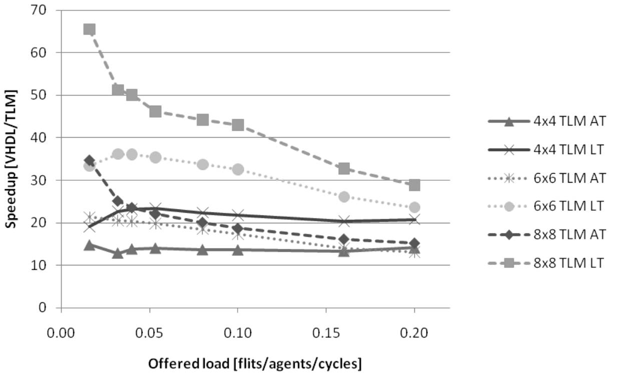

Figure 6: TLM Speedups compared to RTL VHDL (4 word payload)

E. SystemC Implementations

SystemC simulation kernel's and TLM 2.0 library implementation naturally affects simulation speed. Both the OSCI reference kernel and the commercial one used in these simulations were single threaded. Alternatives such as the parallel SystemC kernel presented in [2] could be used to speed up SystemC simulations.

Gcc compiled SystemC RTL was approximately 1.5x faster than the commercial simulator with 4 word packet sizes. Commercial tool on the other hand had faster implementation of OSCI TLM 2.0 library and executed nearly 1.5x faster than gcc compiled version. On larger packet sizes the difference between used simulators was smaller as seen in Figure 5.

V. Conclusions

A simulation performance of SystemC 2D mesh NoC modelled in three abstraction levels was compared against the reference sythesizeable RTL VHDL model.

Table II summarises the results under near saturation load. RTL results are for gcc and TLM for the commercial simulator.

The commercial simulator used was slower for RTL level but faster for TLM. Gcc SystemC on RT level was measured to be 1.5x faster than the VHDL model.

TLM models offered a significant increase in simulation performance. For Approximately Timed (AT) model speedup ranged from 13x to 40x. Error on estimated latencies proved to be sufficient for DSE on most simulation scenarios analysed.

The more abstract Loosely Timed TLM model was approximately 2x faster than AT model to simulate but if implemented like in [4] the significant increase in latency estimation errors brings doubts to the useability of this approach especially under heavier loads.

Based on the study we recommend using AT coding style over LT as the estimation error was small and LT didn't offer significant speedup compared to AT.

Table II: Summary of SystemC speedups over VHDL in near saturation load conditions

Scenario	4x4 Mesh	6x6 Mesh	8x8 Mesh
4 word, RTL	1.1	1.4	1.5
20 word, RTL	1.2	1.4	1.6
40 words, RTL	1.1	1.3	1.5
4 word, AT	13	13	14
20 word, AT	20	28	28
40 words, AT	25	37	40
4 word, LT	23	23	26
20 word, LT	30	50	61
40 words, LT	33	60	81

References

[1] W. Ecker, V. Esen, L. Schonberg, T. Steininger, M. Velten, and M. Hull, "Impact of Description Language, Abstraction Layer, and Value Representation on Simulation Performance," *Design, Automation Test in Europe Conference Exhibition, 2007. DATE '07.*, pp. 1–6, apr. 2007.

[2] P. Ezudheen, P. Chandran, J. Chandra, B. Simon, and D. Ravi, "Parallelizing SystemC Kernel for Fast Hardware Simulation on SMP Machines," *Principles of Advanced and Distributed Simulation, 2009. PADS '09*, pp. 80–87, jun. 2009.

[3] M. Gries, "Methods for Evaluating and Covering the Design Space during Early Design Development," *Integration, the VLSI Journal*, vol. 38, no. 2, pp. 131–183, 2004.

[4] A. Kohler and M. Radetzki, "A SystemC TLM2 model of communication in wormhole switched Networks-On-Chip," *Forum on Specification Design Languages, 2009. FDL 2009*, pp. 1–4, sep. 2009.

[5] A. Morawiec and J. Mermet, "Techniques for Improving the HDL Simulation Performance," 1999.

[6] Open SystemC Initiative OSCI, "SystemC Documentation." [Online]. Available: http://www.systemc.org

[7] S. Pasricha, M. Ben-romdhane, and N. Dutt, "High Level Design Space Exploration of Shared Bus Communication Architectures," *CECS Technical Report*, 2004.

[8] E. Salminen, C. Grecu, T. Hämälainen, and A. Ivanov, "Application modelling and hardware description for network-on-chip benchmarking," *Computers Digital Techniques, IET*, vol. 3, no. 5, pp. 539–550, sept. 2009.

[9] G. Schirner and R. Domer, "Quantitative Analysis of Transaction Level Models for the AMBA Bus," *Design, Automation and Test in Europe, 2006. DATE '06.*, vol. 1, pp. 1–6, mar. 2006.

[10] D. Siguenza-Tortosa and J. Nurmi, "VHDL-based simulation environment for Proteo NoC," *High-Level Design Validation and Test Workshop, 2002.*, pp. 1–6, oct. 2002.

[11] H. van Moll, H. Corporaal, V. Reyes, and M. Boonen, "Fast and accurate protocol specific bus modeling using TLM 2.0," *Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09.*, pp. 316–319, apr. 2009.