NoCs Simulation Framework for OMNeT++

Yaniv Ben-Itzhak[1] Eitan Zahavi[1] Israel Cidon[2] Avinoam Kolodny[2]
Electrical Engineering Department
Technion – Israel Institute of Technology
Haifa, Israel
[1]{yanivbi,ezahavi}@tx.technion.ac.il [2]{cidon, kolodny}@ee.technion.ac.il

ABSTRACT

As chip density keeps doubling every process generation, the use of Network-on-Chip becomes the prevalent architecture of SoC, MPSoC and large scale CMP designs. To that end, diverse NoC solutions are developed by the industry and the research community in order to meet heterogeneous on-chip communication requirements. Consequently, there is a growing need to rely on a simulation tools in order to explore, evaluate and optimize these new NoC architectures and topologies. The simulation platform is based on OMNeT++. It provides an open-source, modular, scalable, extendible and fully parameterizable framework for modeling NoC. In this demo we describe the structure of this framework.

Categories and Subject Descriptors

I.6.0 [**Computing Methodologies**]: Simulation and Modeling – *General.*

General Terms

Measurement, Performance, Experimentation.

Keywords

Networks-on-Chip, NoC Simulator.

1. INTRODUCTION

Network-on-Chip has emerged as a new on-chip communication approach. It offers better scalability, throughput, overall latency and area compared to bus-based on-chip interconnect. However, the NoC design space is very large and high dimensional. It includes the optimization of topology, traffic switching technique, routing mechanism, congestion control methodologies, link capacities, number of buffers and virtual channels per link, etc… Furthermore, the NoC research area is still at its infancy and consequently new architectures, techniques and ideas are being proposed, developed and evaluated. Simulators are essential tools in evaluating the performance of different NoC designs and new proposals. Therefore, in order to be able to cover the existing and future NoC diversities, NoC simulators should be modular, scalable, extendible and fully parameterizable.

Table 1. NoC Simulators Comparison

	Frame-work	Availability	Parallel-ism	Topologies	Open-Source
SICOSYS [7]	C++	+	-	Limited	+
Noxim [2]	SystemC	+	-	Mesh	+
NNSE [6]	SystemC	+	-	Mesh/Torus	+
Nirgam [4]	SystemC	+	-	All	+
gpNoCsim [3]	Java	+	-	All	+
[1]	OMNeT++	-	+	All	-
DARSIM [5]	C++	+	+	All	+
Our Simulator	OMNeT++	+	+	All	+

In this demo we introduce an OMNeT++ based modular and open-source NoC simulator. Several previous NoC simulators have been presented; however, none of which satisfies all the aforementioned requirements needed to cover the design and research space. Table 1 presents a comparison between previous NoC simulators and our proposed simulator. The simulator is based on OMNeT++ [8], which is an extensible, modular, open-source component-based C++ simulation library and framework, primarily aimed at building network simulators.

OMNeT++ provides several important advantages to the simulation framework. It offers easy traceability and debug-utilities to reduce debug-time, and built-in parallelism support to reduce simulation run-time. Moreover, it supports flexible and efficient topology definition using NED, the OMNeT++ topology description language. NED has a simple syntax. Yet, it is very powerful for defining arbitrary topologies. Furthermore, OMNeT++ offers free license for academic usage. All these advantages give it the potential to become a highly beneficial and extensible open source simulation platform for the service of the NoC research community.

Currently, it supports three router types: synchronous, asynchronous and virtual output queue (VoQ).

2. SIMULATOR STRUCTURE

The simulator architecture is optimized for extendibility and comparative architecture evaluation. This requirement is translated to several major features: the use of module interfaces, message classes, modules directory, support for arbitrary topologies and module selection as a simulation parameter. In this section we focus on describing the modules, the traffic configuration and the statistics collection.

An OMNeT++ module represents a hardware or a software entity that is capable of receiving messages (from itself or other modules) and implements a self-contained logic. Modules are

NOCS'11, May 1–4, 2011, Pittsburgh, PA, USA.
Copyright 2011 ACM 978-1-4503-0720-8…$10.00.

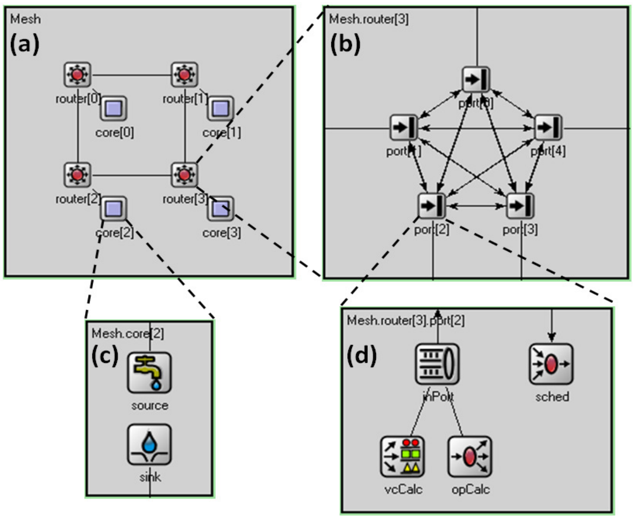

Figure 1. A 4x4 mesh NoC: (a) a complete mesh built from routers and cores; (b) a hierarchical router of 5 ports; (c) a core structure; (d) a single port structure.

declared by specifying their attributes and their "ports" – where messages can arrive or leave. A module-interface is a skeleton that has no real module implementation behind it. Modules may declare their adherence to such an interface (meaning the set of ports and properties) and thus may be used in place of the skeletons in a dynamic manner.

Generally, a NoC is built from two main modules: Routers and network interfaces (NI) as shown in Figure 1(a) (the NIs are named "core[*]" and routers "router[*]"). Internally the NI contains sources and sinks (see Figure 1(c)). Routers are either flat or hierarchically built as a collection of connected ports. Flat router has no sub-modules. Hierarchical routers consist of ports as depicted in Figure 1(b). The ports include scheduler (sched), input-buffer (inPort), VC-allocator (vcCalc) and output-port selector (opCalc), as shown in Figure 1(d). The scheduler arbitrates the VC to win the output of the port which implements the switch allocation pipe-stage. The input port (inPort) stores the incoming flits in a set of buffers. The VC-allocator decides about the output VC for each packet. The output port selector implements the routing decision. The switch is built by the cross connections of port (Port_Ifc) outputs (driven by the input-buffer and named "sw_in") to the other ports inputs (driving the scheduler and named "sw_out").

The traffic is configured by setting the destination and packet-arrival time parameters for each source. The destination can be either deterministic or randomly distributed. The distributions are set using the built-in OMNeT++ random number generation functions (e.g. intuniform). The packet inter-arrival times can be either distributed (e.g. exponential, constant) or driven by a trace file. The trace file includes the specific times where a packet should be generated. Hence, it can cover a variety of traffic patterns and schemes.

The simulator provides a rich set of statistical measurements collected by the sink, the source and the in-port modules. The sinks collects throughput and different latency statistics at the flit and package levels. The source collects several source queue indicators in order to identify the point of NoC saturation. The in-port collects VC acquisition latencies (i.e. latency to acquire an out-port VC) and transfer latencies.

3. NUMERICAL RESULTS

We present comparison between three routers implemented by the simulator (i.e. synchronous, asynchronous and virtual output queue (VoQ)) for a uniform traffic pattern. Figure 2 presents the average throughput and end-to-end latency for a uniform traffic pattern over 4x4 NoC with two VCs.

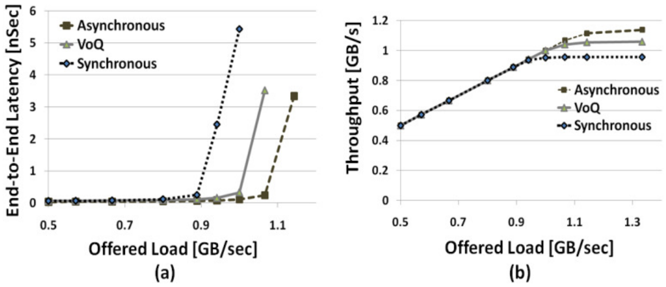

Figure 2. Uniform traffic pattern. (a) Average end-to-end latency versus offered load; (b) Average throughput versus offered load.

4. SUMMARY

A modular NoC simulator based on OMNeT++ framework has been presented. Several advantages of the simulator compared to previous NoC simulators were discussed. It was demonstrated that it offers an open-source, scalable, extendible and fully parameterizable framework for modeling NoC.

5. REFERNCES

[1] R. Al-Badi, M. Al-Riyami, and N. Alzeidi. A parameterized NoC simulator using OMNet++. In *Ultra Modern Telecommunications & Workshops, 2009. ICUMT'09. International Conference on*, pages 1–7. IEEE, 2009.

[2] F. Fazzino, M. Palesi, and D. Patti. Noxim: Network-on-chip simulator, 2008.

[3] H. Hossain, M. Ahmed, A. Al-Nayeem, T. Islam, and M. Akbar. Gpnocsim-A General Purpose Simulator for Network-On-Chip. In *Information and Communication Technology, 2007. ICICT'07. International Conference on*, pages 254–257. IEEE, 2007.

[4] L. Jain, B. Al-Hashimi, M. Gaur, V. Laxmi, and A. Narayanan. NIRGAM: a simulator for NoC interconnect routing and application modeling. In *Workshop on Diagnostic Services in Network-on-Chips, Design, Automation and Test in Europe Conference (DATE'07)*, pages 16–20, 2007.

[5] M. Lis, K. Shim, M. Cho, P. Ren, O. Khan, and S. Devadas. DARSIM: a parallel cycle-level NoC simulator. In *6th Annual Workshop on Modeling, Benchmarking and Simulation*. Citeseer, 2010.

[6] Z. Lu, R. Thid, M. Millberg, E. Nilsson, and A. Jantsch. NNSE: Nostrum network-on-chip simulation environment. In *Swedish System-on-Chip Conference (SSoCC'03)*, pages 1–4. Citeseer, 2005.

[7] V. Puente, J. Gregorio, and R. Beivide. SICOSYS: an integrated framework for studying interconnection network performance in multiprocessor systems. In *Parallel, Distributed and Network-based Processing, 2002. Proceedings. 10th Euromicro Workshop on*, pages 15–22. IEEE, 2002.

[8] A. Varga et al. The OMNeT++ discrete event simulation system. In *Proceedings of the European Simulation Multiconference (ESM'2001)*, pages 319–324, 2001.