ACCELERATING EMULATION FOR RAPID DESIGN SPACE EXPLORATION OF ON-CHIP NETWORKS

by

GUMMIDIPUDI KRISHNAIAH

Department of Computer Science and Engineering

Submitted
in fulfillment of the requirements of the degree of
Doctor of Philosophy

to the

Indian Institute of Technology Delhi
January 2013

Certificate

This is to certify that the thesis titled **"Accelerating Emulation for Rapid Design Space Exploration of On-chip Networks"** being submitted by **Gummidipudi Krishnaiah** for the award of **Doctor of Philosophy** in **Computer Science & Engg.** is a record of bona fide work carried out by him under our guidance and supervision at the **Department of Computer Science & Engineering, Indian Institute of Technology Delhi**. The work presented in this thesis has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma.

Preeti Ranjan Panda

Professor
Dept. of Computer Science & Engg.
Indian Institute of Technology Delhi.

Anshul Kumar

Professor
Dept. of Computer Science & Engg.
Indian Institute of Technology Delhi.

Ackowledgements

It gives me immense pleasure to acknowledge several people who have contributed towards my dissertation.

First and foremost, I thank my advisors Prof. Preeti Ranjan Panda and Prof. Anshul Kumar for giving me an opportunity to work with them and for guiding me through out my research work as well as courses. I thank Prof. Kolin Paul, Prof. M. Balakrishnan and Prof. G.S. Visweswaran for their valuable feedback.

I would like to thank Intel India, Lucent and Microsoft for sponsoring my research work, providing monthly stipend and supporting my conference travels. I thank Ashok Jagannathan and Sreenivas Subramoney for mentoring me during my internship at Intel.

I thank B.V.N. Silpa for all the brainstorming sessions and keeping me motivated throughout my research work. I would also like to thank all my fellow students – Aryabartta Sahu, Anant Vishnoi, Neeraj Goel, Lava Bhargava, Sharat Varma, Namita Sharma and Arun Parakh for all the help, discussions and great company. I thank Vandana Ahluwalia and Som Dutt Sharma for providing access to the required compute resources in the lab.

Lastly, I am grateful to my parents for all their support.

G. Krishnaiah

Abstract

With Moore's law still continuing to hold good, today, billions of transistors are fabricated on a single die. To utilize these transistors, large number of solutions or features are being integrated on a single chip (eg. System-on-Chip). The increasing CPU frequency has led to an increase in memory latency *memory-wall* and an exponential increase in power consumption *power-wall*. Also, the diminishing returns from further exploitation of instruction-level parallelism has shifted the focus towards thread-level parallelism resulting in multi-core or chip multiprocessor architectures [1]. Network-on-Chip is a popular choice for designing many-core chips and is being increasingly used in the commercial processors for scalability and efficiency reasons [2, 3, 4, 5]. The development of these SoCs and CMPs goes through several phases such as, high level exploration, micro-architectural development, synthesis and fabrication. Increasing design complexities and constraints on time-to-market have put enormous stress on the entire design cycle of these products. With large number of modules being integrated on a chip, the interconnection network plays a vital role in determining its overall power and performance. Designing a custom NoC for the processor involves exploring huge number of design options (such as topology, routing, flow-control, micro-architecture and quality of service) which is a very time consuming process. Speeding-up exploration will lead to faster time to market and allow architects or designers to explore larger design space within a given period of time, thus helping them to find more efficient architectural solutions.

Traditionally software simulators are a popular choice for high level exploration due to their flexibility and relatively low development costs. Lately, due to their inadequate speeds, FPGAs are increasingly being used for early exploration. In this thesis, we address the problem of improving the speed of NoC simulations through hardware accelerated solutions. FPGAs can be easily programmed to emulate any target hardware. The need for hardware acceleration of Network-on-Chip (NoC) simulation has been motivated by their growing complexity and large design space. NoC simulation can exploit the inherent concurrency in hardware by using FPGA based emulators. For trace-driven NoC emulators, we enhance this raw hardware speedup by exploiting the traffic characteristics for a more efficient utilization of hardware. The main contributions from this thesis are:

- We study the effect of abstraction level on the both simulation speed and synthesis efficiency and propose a few guidelines to alleviate the limitations of synthesis tools while maintaining the flexibility in the model.

- We analyze the NoC traffic pattern based on traces from real applications and propose a simple emulation speedup technique to exploit the NoC traffic characteristics.

- We propose to relax lock-step synchronization during FPGA emulation for parallel simulation of temporally-decoupled network transactions. We devise an algorithm to maintain the correctness in measured statistics while the lock-step synchronization is relaxed.
- We present a case study to show the effectiveness of our emulation methodology and speedup techniques through NoC design space exploration using the SPLASH benchmark suite.

The software tool-chain for FPGA design flow is highly optimized for RTL input; however developing detailed RTL models for exploration defeats the purpose. High level synthesis (HLS) tools allow generating detailed RTL code from high level C++ or SystemC models. However, the efficiency of the generated RTL is typically low compared to hand coded versions, and hence, the emulation speed of the generated RTL may be much lower than expected. We compare the effect of different modeling abstractions on the simulation and emulation speed. Based on our studies using commercial HLS tools, we show that techniques such as scalarization and pipelining enhance the synthesis efficiency while quickly generating RTL code from high level models. For NoC synthesis, our modeling guidelines have improved the emulation speed by more than $6\times$ while synthesizing from high-level SystemC models.

The speedup provided through FPGA emulation mainly depends on three factors – the quality of the high level model, the availability of hardware resources on the FPGA fabric and the efficiency of synthesis and mapping tools. We propose to enhance the emulation speed by exploiting the characteristics of the emulated application. We study the NoC traffic characteristics using real benchmarks and observe that it is generally bursty with several intervals of zero activity on the network. On an average, for about 50% of the simulation cycles, the network remains idle. To exploit this nature of network traffic we propose a technique, *FastFwd*, to skip the idle period — which does not contribute to NoC power-performance statistics — and enhance the emulation speed. We have devised a hardware controller that dynamically monitors the activity on the network and skips the evaluation of idle period. In a way, we are proposing an event-driven methodology for NoC emulation to skip idle periods. Our synthesis and emulation results show that we achieve close to $1.8\times$ speedup compared to conventional emulation. We further develop a distributed/hierarchical controller to exploit the temporal nature of network transactions in a course-grained manner; this technique further enhances the speedup by 10%.

Our studies indicate that when NoC is emulated in lock-step synchronized manner it leads to inefficient utilization of FPGA resources. For example, two different network transactions which are temporally decoupled and confined to two different regions in an NoC can be simulated in parallel if the lock-step restriction is relaxed during emulation. Thus, the simulation of events can be completed quickly, improving the effective simulation speed. A large percentage of network transactions fall into this category and hence our proposed technique, *OTO-Sim* (Out-of-Time-Order simulation), can potentially provide considerable speedup. However, the

out-of-time-order simulation of transactions may lead to incorrect statistics defeating the whole purpose of simulation. In order to preserve the correctness, we develop a technique that uses the temporal order among the events. We store this information along with the trace after a static-preprocessing stage. This, along with the network status, is dynamically processed by the controller to maintain the correctness of measured statistics while simulating temporally decoupled events in parallel. Considering the overhead and complexity of the controller, this technique achieves an average speedup of more than $3\times$ compared to conventional emulation.

Lastly we show the effectiveness of our techniques by exploring a few options within the large design space of NoC using SPLASH-2 benchmarks. Our experiments of studying the effect of router micro-architecture (varying virtual channels, buffer sizes), network topologies and link bandwidth on power-performance-cost metrics such as, total power dissipated, latency, throughput and area of the design identifies suitable configuration for various design constraints (power, performance per watt etc). The FPGA emulation would save several weeks of time when compared to software simulation on host and our techniques would further reduce the exploration time by $3\times$.

Contents