Simulation infrastructure for the next kilo-x86-64 Chips

Antoni Portero, Alberto Scionti, Marco Solinas, Ho Nam, Roberto Giorgi

Università degli Studi di Siena- Dipartimento di Ingegneria dell'Informazione Via Roma, 56 -- 53100 SIENA - Italy

ABSTRACT

The enhancement in silicon technology facilitates the integration of a higher number of cores on a single chip. Considering the current CMOS integration technology tendency; in the next future, systems are expected to scale up the number of cores, resulting in architectures composed by thousands of cores (i.e., namely kilo- core architecture). The architecture of these kilo-core systems is still an open issue (i.e., number and type of cores, number of levels in the cache memory hierarchy, usage of specialized accelerators, inter-connections types, etc.). The simulators provides high benefit in finding out architectural designs trade-offs for such next generation systems.

This paper proposes a simulation framework based on the COTSon infrastructure, able to create thousands of virtual x86-64 cores. The framework offers a full-system architectural simulator and a well balanced trade-off between simulation speed and accuracy. Experimental outcomes demonstrates for our framework the possibility correctly simulate a large many-core machine.

KEYWORDS: Performance Analysis and Design Aids; Simulation; Verification; Verification; Worst- case analysis.

1 Introduction

The adoption of architectural simulators has become essential for assuring the correctness of any design. Architectural simulators historically suffered from low simulation speed and accuracy, imposing serious limitations on the ability of predicting correct behaviors of the designed architecture[6, 15], especially in the many-core era. Moreover, the adoption of more scalable yet complex interconnection systems, e.g NoCs [14] that has led to the creation of tools specifically devoted to the accurate timing simulation of these communication infrastructures.

With the aim of providing a tool characterized by a high simulation speed and accuracy for a heterogeneous kilo-core architecture integrating an accurate network-on-chip simulator, this paper proposes a framework based on the COTSon [3] infrastructure. Compared with current state-of-the-art simulation platforms, the our approach offers a complete environment for a many-core full-system simulation, and for its power consumption estimation. In order to guarantee fast simulations, COTSon implements a functional-directed approach, where functional emulation is alternated to a complete timing-based simulation. The result is the

ability of supporting the full stack of applications, middleware and OSs. The modular approach on which COTSon is based, allow us to adopt both the proprietary AMD SimNow [1](available now) emulator and the open source Qemu-based [2](in progress) functional emulator, opening the door to the support of several different micro-architectures. Finally, the integration of the proposed framework with the McPAT tool [4], provides the ability of

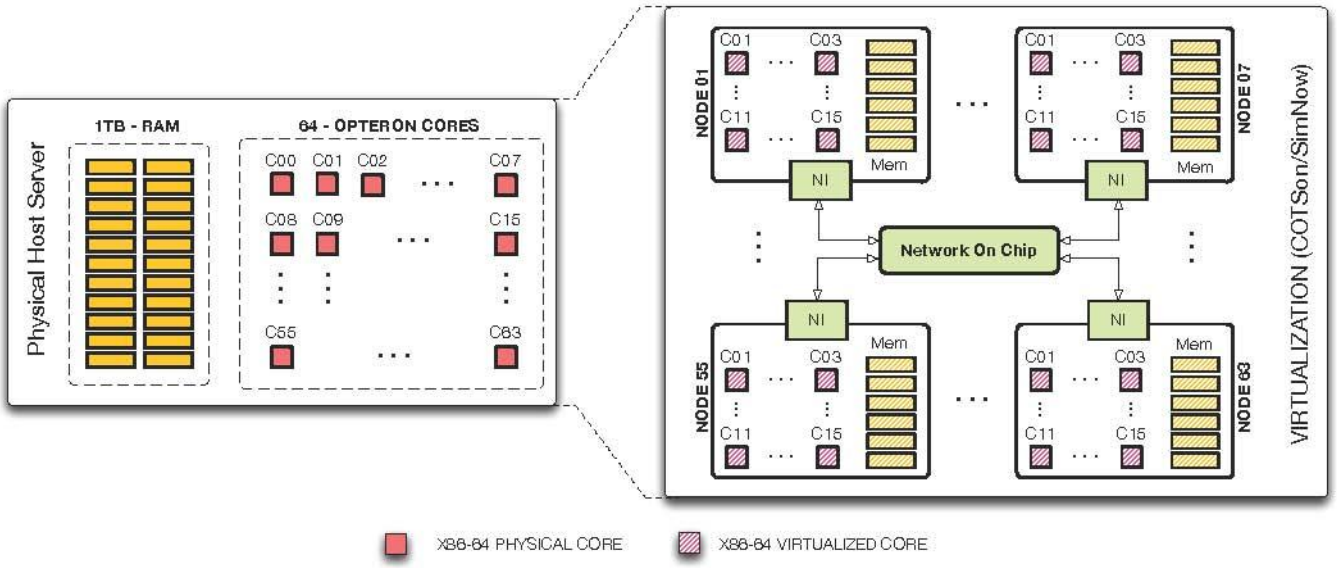

Figure 1. Host system versus Virtual system

2 Simulated platform

In this section, we expose the architecture of the platform able to support the simulation of a very high number of cores. In order to achieve this goal, we need a powerful simulation system. We define the host machine as the computer where we run the simulated virtual processor, and the guest machine as the proper simulated machine. Currently, we use as host machine a DL-Proliant DL585 G7 AMD Opteron™ 6200 Series, in total is equipped with 64 cores coupled to 1TB-DRAM of shared main memory.

There is a trade-off between complexity of the guest machine and the time required by the simulation. Higher complexity in the guest machine (number of simulated cores, memory etc.) produces longer simulations. A good trade-off is to use one host-core for each functional instance (i.e., a functional instance is equivalent to a node in the simulated chip architecture) representing a node. Each node can have till 32 cores but we have experimented that 16 x86-64 cores per node can better scale up in terms of execution time. Since, the simulation of a one thousand core system can be achieved distributing the simulation on more than one host. However since we want to focus on the simulation of a 1K-core system, considering a single host machine is sufficient. In order to correctly simulate a kilo-core architecture, we booted up 64 virtual nodes, each one containing 16 x86-64 cores based on AMD Opteron-L1_JH-F0 (800Mhz) architecture, and 256M DRAM per core. Figure 1 depicts the system host and guest systems.

Each node runs a Linux distribution operating system. On top of this system, we are able to run several benchmarks based on both OpenMP and MPI programming models. One of the main modifications we did, has been the implementation of the support of DF-threads [5, 7, 8, 11, 12, 13] through the ISA extension. DF-threads enable a different execution model based on the availability of data and open the door for many architectural optimizations not possible in current standard off-the-shelf cores.

We can still double the number of virtual nodes from 64 to 128 (one master node and 128 slaves) resulting in a 40% usage of the DRAM memory in the host machine. Figure 2 shows the tendency if we increase the number of virtual nodes. As expected the host main memory consumption and the CPU utilization increase. We can arrive to simulate 220 nodes of 32 cores, 7040 cores in total using the 92% of the main memory and the 93% of the host CPU utilization. This demonstrates the ability of the proposed simulation framework to scale the simulations to 1 kilo-core range and beyond (up to 7 kilo-cores were tested).

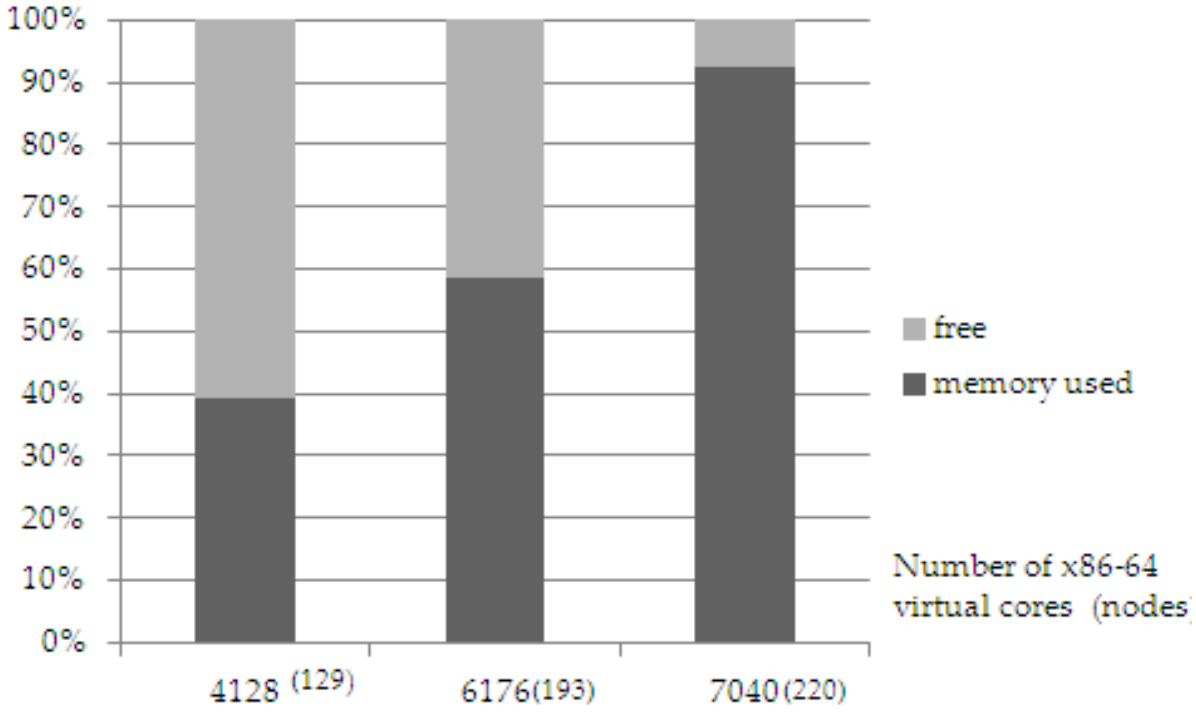

Figure 2: Number of Virtual Cores vs Memory utilization in HP ProLiant DL585 G7 Server (1 TB Memory , 64 x86-64 cores).

Updated information of the project [10]. And An extended explanation of the simulation system is in the references[6,7,9].

Conclusions

The paper presents a simulation framework based on x86-64 instruction set. It has been modified to support ISA extensions(DF-Threads execution)[12] . With the proposed simulation framework we are able to simulate a system composed of more than 7000 x86-64

cores and their corresponding communication infrastructure. The proposed framework serves to find the bottle-necks of the target system, and allow

ACKNOWLEDGEMENTS

This work was partly funded by the European FP7 projects TERAFLUX id. 249013 http://www.teraflux.eu, ERA (Embedded Reconfigurable Architectures) id. 249059 (FP7) http://era-project.eu; HiPEAC IST-217068, and IT PRIN 2008 (200855LRP2).

References

[1] AMD SimNow Simulator 4.6.1 User's Manual, November 2009.

[2] F. Bellard. Qemu, a fast and portable dynamic translator. In Proceedings of the 2005 USENIX Annual Technical Conference, 2005.

[3] E. Argollo et al. Cotson infrastructure for full system simulation. Operating Systems Rev, 43:52–61, 2009.

[4] S. Li and et al. Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In Proceedings of the 42nd Annual International Symposium on Microarchitecture, pages 469–480. IEEE/ACM, December 2009.

[5] Exploiting Dataflow Parallelism in Teradevice Computing. http://www.teraflux.eu, 2010-2014.

[6] Antoni Portero, Alberto Scionti, Zhibin Yu, Paolo Faraboschi, Caroline Concatto, Luigi Carro, Arne Garbade, Sebastian Weis, Theo Ungerer, Roberto Giorgi, Simulating the Future kilo-x86-64 core Processor and their Infrastructure, 45th Annual Simulation Symposium (ANSS), March 2012, Orlando, Florida

[7] Antoni Portero, Zhibin Yu, and Roberto Giorgi. T-star (t*): An x86-64 isa extension to support thread execution on many cores. ACACES Advance Computer Architecture and Compilation for High-Performance and Embedded Systems, 1:277–280, 2011.

[8] Roberto Giorgi, Alberto Scionti, Antoni Portero, Paolo Faraboschi, "Architectural Simulation in the Kilo-core Era, "Architectural Support for Programming Languages and Operating Systems (ASPLOS 2012), poster presentation, London, UK, ACM Association for Computing Machinery

[9] Antoni Portero, Zhibin Yu, Roberto Giorgi, TERAFLUX: Exploiting Tera-device Computing Challanges, TERAFLUX: Exploiting Tera-device Computing Challenges, Procedia Computer Science 7:146-147 (2011)

[10] Roberto Giorgi et al, "Public Report, D7.2– Definition of ISA extensions, custom devices and External COTSon API extensions", FET proactive 1: Concurrent Tera-Device Computing (ICT-2009.8.1) PROJECT NUMBER: 249013

[11] Krishna M. Kavi, Roberto Giorgi, Joseph Arul, "Scheduled Dataflow: Execution Paradigm, Architecture, and Performance Evaluation", IEEE Trans. Computers, Los Alamitos, CA, USA, vol. 50, no. 8, Aug. 2001, pp. 834-846

[12] R. Giorgi, Z. Popovic, N. Puzovic, "DTA-C: A Decoupled multi-Threaded Architecture for CMP Systems", Proc. IEEE SBAC-PAD, Gramado, Brasil, Oct. 2007, pp. 263-270

[13] R. Giorgi, Z. Popovic, N. Puzovic, "Exploiting DMA to enable non-blocking execution in Decoupled Threaded Architecture", Proc. IEEE Int.l Symp. on Parallel and Distributed Processing – MTAAP Multi-Threading Architectures and Applications, Rome, Italy, May 2009, pp. 1-8

[14] A. Portero, R. Pla, J. Carrabina, "SystemC implementation of a NoC",Industrial Technology, 2005. ICIT 2005. IEEE International Conference on. Pages 1132-1135

[15] R. Giorgi, C.A. Prete, G. Prina, L. Ricciardi, "A Hybrid Approach to Trace Generation for Performance Evaluation of Shared-Bus Multiprocessors", IEEE Proc. 22nd EuroMicro Int.l Conf. (EM-96), ISBN:0-8186-7487-3, Prague, Ceck Republic, Sept. 1996, pp. 207-214