Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 7 (2011) 146–147

Procedia Computer Science

The European Future Technologies Conference and Exhibition 2011

TERAFLUX: Exploiting Tera-device Computing Challenges

Antoni Portero, Zhibin Yu, Roberto Giorgi

Dipartamento di Ingengneria dell'Informazione, Via Roma 56, Siena 53100, Italy

Abstract

The number of cores per chip keeps increasing in order to improve performance while controlling the power. According to semiconductor roadmaps, future computing systems while reach the scale of 1 Tera devices in a single package. Firstly, such Tera-device systems will expose a large amount of parallelism that cannot be easily and efficiently exploited by current applications and programming models. Secondly, the reliability of Tera-device systems will become a critical issue. Finally, we need to simplify the design of such systems. TERAFLUX aims at providing a framework based on dataflow concepts that could provide a solution for all the three above challenges. We briefly present here our idea on the architectural support for the TERAFLUX execution model.
© Selection and peer-review under responsibility of FET11 conference organizers and published by Elsevier B.V.

Keywords: Tera-device; Many-core; Parallelism; Reliability; Dataflow

1. Overview of the TERAFLUX project

The TERAFLUX project aims to develop new programming models, compiler analysis and optimization technologies, and to build a scalable, reliable architecture based mostly on off the shelf components. Data-flow principles are exploited at all levels to overcome the current limitations. While parallel systems have been around for many years, they were usually programmed and tuned by experts. In the future large scale systems will be widely available and therefore exploiting efficiently the available parallelism will have to be easy enough to be accessible by the common user. We propose to use a model that offers dataflow scheduling of parallel execution threads. Combining multithreading with dataflow allows exploiting the available parallelism without the overheads of the original dataflow techniques. In this paper we focus on the architectural support for a data-driven execution model at level of threads.

2. Distributed Thread Scheduler (DTS)

The Execution model heavily relies on exchanging data threads in a producer-consumer fashion (therefore we refer to this model also as “dataflow execution model” [1,3–5]). In order to support the data driven execution of threads, each core also includes a hardware module that handles the scheduling of threads, the Local Thread Scheduling Unit (L-TSU). In addition to the cores, the nodes also contain a hardware module to coordinate the scheduling of the data driven threads across nodes, the Distributed Thread Scheduling Unit (D-TSU), as well as a hardware module that monitors the performance and faults of the cores, the Distributed Fault Detection Unit (D-FDU). The set of all L-TSU and D-TSU make the DTS. Nodes are connected via an inter-node network, the Network on Chip (NoC).

1877-0509/$ – see front matter © Selection and peer-review under responsibility of FET11 conference organizers and published by Elsevier B.V.
doi:10.1016/j.procs.2011.09.081

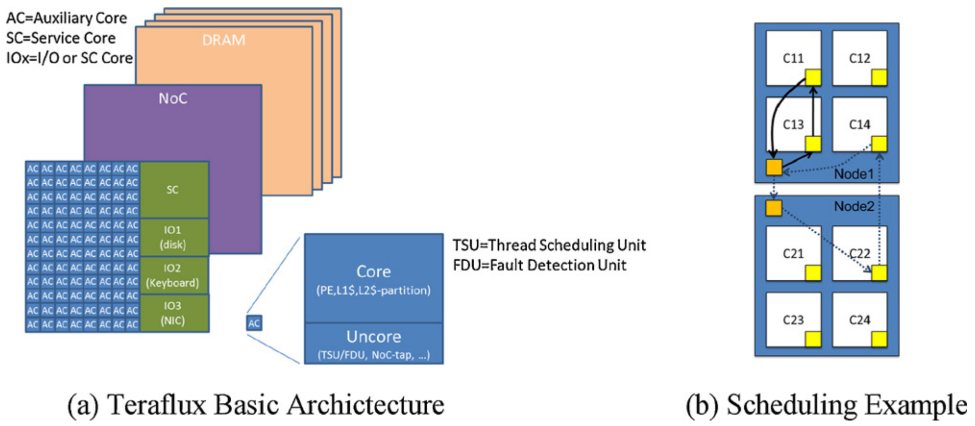

Fig. 1. (a) Teraflux Basic Archictecture. (b) Scheduling Example.

3. Distributed Thread Scheduler (DTS) and Execution Model

In TERAFLUX we identify different types of threads. They differ in the characteristics of their code and functionalities that they may use. The compiler tool will be able to identify the different types of threads. The DF-threads allow us to repeat their execution in case of recovery from faults or even for speculation: it should always be possible to repeat the execution of a DF-thread, eventually discard the results of one of two instances of two speculative executions. Figure 1(b) shows two nodes of four cores. The L-TSU is represented in yellow inside each core C_{jk}. The D-TSU is represented in orange and it is one per node. If the scheduling cannot happen inside a core the requests are forwarded to other cores or nodes (through the D-TSU).

The execution model makes use of special instructions (ISA-extensions) to support the execution of data-flow threads (DF-threads). The project supports the fundamental and complementary concepts of the models developed by the different partners of the projects. In particular the DTA [4] and DDM [1] fine-grain threaded data-flow models as well as the StarSs [6] coarse-grain threaded task data-flow model.

Acknowledgements

We thank Dr. Nikola Puzovic for the many useful discussions and comments and his contribution to the TERAFLUX project. This work was partly funded by the European FP7 project TERAFLUX id. 249013, http://www.teraflux.eu, HiPEAC IST-217068, and IT PRIN 2008 (200855LRP2).

References

[1] C. Kyriacou, P. Evripidou, P. Trancoso, Data-Driven Multithreading Using Conventional Microprocessors, in: Parallel and Distributed Systems, IEEE Transactions on, vol. 17, no. 10, Oct., 2006, pp. 1176–1188.

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

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

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

[6] Pieter Bellens, Josep M. Perez, Rosa M. Badia, Jesus Labarta, CellSS: A Programming Model for the Cell BE Architecture, in: Proceedings of the ACM/IEEE SC Conference, 2006.

Further reading

[2] 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.