2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE 10 YEAR ANNIVERSARY

Grid-based Mapping and Analysis of a GoogLeNet CNN using MapGL Editor

Claudio Raccomandato, Politecnico di Torino, Turin, Italy (*claudio.raccomandato@studenti.polito.it*)

Emad M. Arasteh and Rainer Dömer, CECS, University of California Irvine, Irvine, USA ({*emalekza,doemer*}*@uci.edu*)

***Abstract*—The Grid of Processing Cells (GPC) has been proposed as a scalable many-core architecture, modeled using SystemC TLM-2.0 methodology. This work introduces a graphical CAD software called Map Grid-based Layouts (MapGL) to facilitate the design process of GPC-based applications, automatically generate their SystemC models, and perform analyses on memory usage and speed. Using MapGL, we map a GoogLeNet Convolutional Neural Network (CNN) to a suitable GPC and improve it with a new modular Memory Access Resources and Interfaces (MARI) library for better communication between processing cells and lower resource usage.**

Keywords—System modeling; SystemC-TLM2.0; CAD IDE

I. Introduction

Over the last two decades, computer systems focus shifted from raising the clock frequency toward increasing the number of processors [1]. Shared main memory can delay many-core processors for thousands of cycles due to bus contention despite sophisticated multi-level cache hierarchies [2]. As an alternative scalable computer organization, tiled network-on-chip architectures have been proposed with separate local memories, such as the Grid of Processing Cells (GPC) [3], where processor-memory pairs are arranged on-chip in a two-dimensional array with only local interconnect.

Without the assumption of one shared memory, software must be explicitly partitioned among cells. To mitigate this programming problem, this paper presents a graphical CAD tool called *Map Grid-based Layouts (MapGL)* that allows the user to interactively map an application to a GPC platform, automatically generate the SystemC model, and evaluate it for performance and resource usage.

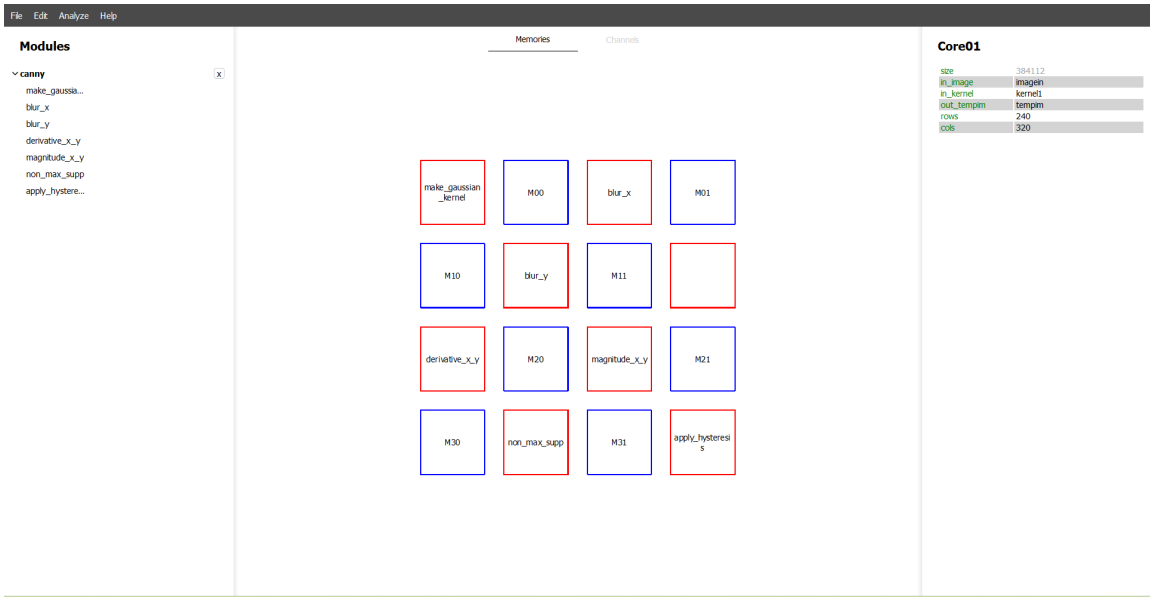

Figure 1. MapGL, GPC mapping of a canny edge detector algorithm.

II. Related Work

A large body of research addresses the partition and mapping problem to many-core network-on-chip (NoC) platforms. In [4], Yang et al. proposed a multiple applications mapping method on the many-core NoC that finds a region on the NoC for each application and then maps tasks into each regions. In [5], Murali et al. proposed a methodology to map different use-cases into the NoC architecture, satisfying the performance constraints of each individual use-case. While these works focus on many-core architecture mapping, our work is a holistic application mapping approach on the Grid of Processing Cells.

In [6], Bruch et al. proposed a graphical user interface computer-aided design (CAD) tool which allows the user to evaluate the performance of NoCs systems using traffic generators in SystemC simulations. While the proposed BrownPepper simulator [6] allows to design and profile RTL and transaction-level models on a system-on-chip 2D-mesh architecture, it does not allow the user to interact, visualize and map an application.

III. Map Grid-Based Layouts: MapGL

We present MapGL, a CAD software that lets the user design a custom GPC, automatically generate the SystemC model, and analyze it. A MapGL mapping model is reusable and portable, saved as a single JSON file. Figure 1 shows MapGL's main window with a canny edge detector example opened.

Inside the MapGL editor, every design is defined using *Modules* and *Channels*. A module describes the behavior of a core using C/C++, while a channel allows two or more cores to communicate. The GPC structure forces each core to communicate by reading and writing data into the shared memories. Our MARI library simplifies interactions and optimizes memory usage by providing a set of software-based channels that the user can use to transfer data directly between cores. After a simulation, MARI can generate a trace with all the channels accesses. This last feature is used by MapGL to enable a more in-depth performance evaluation.

From the provided mapping, MapGL generates a SystemC model that can be used to perform static and dynamic analyses. The detailed results of the profiling are then exported as text file reports.

Figure 2. GoogLeNet CNN on GPC, preliminary mapping.

IV. Experiments and Results

To demonstrate the scalability and usability of the GPC architecture, three SystemC TLM-2.0 models of the GoogLeNet image classification CNN [7] have been mapped and profiled using MapGL combined with MARI. The first model, called *preliminary*, represents the first attempt to map the application. The other two models use an *improved* mapping; one tends to increase the application's speed, called *high-speed*, and the other optimizes memory usage, called *low-memory*. The GoogLeNet network is composed of 142 layers, and each of them was manually mapped to one of the GPC cores using MapGL.

The *preliminary* model, shown in Figure 2, used additional modules to serialize and deserialize data for communication purposes, which increased the required number of cores to 195, of which 26 were not even used in the 15 by 13 grid. The *improved* mapping removed the extra modules thanks to the clever implementation of multi-channels in the MARI library, which reduced the required number of cores to 150, of which just 7 were not used inside the 15 by 10 grid.

MapGL profiling highlighted long delays and high memory usage in the first layers of the network in all the models. The *improved* mapping reduces by almost one-fourth the grid size compared to the *preliminary*. This result caused the two following models to become faster and require fewer memories. Overall the *high-speed* model performs better than the *preliminary* model, while the low-memory model represents a valuable alternative to reduce memory consumption.

V. Conclusion

This paper presented the MapGL editor to map and evaluate the performances of three GPC-based GoogLeNet CNN models, which exploited the scalability of the GPC architecture using a grid of up to 195 cores. All the SystemC models were interactively mapped and automatically generated using MapGL. The integrated analysis tools were then used to evaluate the models' memories usage and speed. The results showed the superiority of the high-speed model over the preliminary model, indicating the low-memory model as a valuable trade-off to reduce memory usage.

References

[1] L. Azriel, A. Mendelson, and U. Weiser, "Peripheral memory: a technique for fighting memory bandwidth bottleneck," IEEE Computer Architecture Letters, vol. 14, no. 1, pp. 54–57, 2015.

[2] G. Liu, T. Schmidt, A. Dingankar, D. Kirkpatrick, and R. D¨omer, "Optimizing Thread-to-Core Mapping on Manycore Platforms with Distributed Tag Directories," in Proceedings of the Asia and South Pacific Design Automation Conference (ASPDAC), Jan. 2015.

[3] R. D¨omer, "A Grid of Processing Cells (GPC) with Local Memories," Center for Embedded and Cyber-physical Systems, University of California, Irvine, Tech. Rep. CECS-TR-22-01, Apr. 2022.

[4] B. Yang, L. Guang, T. C. Xu, A. W. Yin, T. S ¨antti, and J. Plosila, "Multi-application multi-step mapping method for many-core network-on-chips," in NORCHIP 2010, 2010, pp. 1–6.

[5] S. Murali, M. Coenen, A. Radulescu, K. Goossens, and G. De Micheli, "Mapping and configuration methods for multi-use-case networks on chips," in Asia and South Pacific Conference on Design Automation, 2006., 2006, pp. 6 pp.–.

[6] J. V. Bruch, M. R. Pizzoni, and C. A. Zeferino, "Brownpepper: A systemc-based simulator for performance evaluation of networks-on-chip," in 2009 17th IFIP International Conference on Very Large Scale Integration (VLSI-SoC), 2009, pp. 223–226

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1–9.