ParaFPGA15: Exploring threads and trends in programmable hardware

Erik H. D'Hollander[a,1] , Dirk Stroobandt[a] and Abdellah Touhafi[b]
[a] *ELIS department, Ghent University, Belgium*
[b] *ETRO department, Vrije Universiteit Brussel, Belgium*

Abstract. The symposium ParaFPGA focuses on parallel techniques using FPGAs as accelerator in high performance computing. The green computing aspects of low power consumption at high performance were somewhat tempered by long design cycles and hard programmability issues. However, in recent years FPGAs have become new contenders as versatile compute accelerators because of a growing market interest, extended application domains and maturing high-level synthesis tools. The keynote paper highlights the historical and modern approaches to high-level FPGA programming and the contributions cover applications such as NP-complete satisfiability problems and convex hull image processing as well as performance evaluation, partial reconfiguration and systematic design exploration.

Keywords. FPGA performance, high-level synthesis, OpenCL, functional programming, satisfiability problem, image processing

Introduction

After hyping expectations on the role of FPGAs in high performance and low power computing, data centers, internet of things, streaming applications, accelerators and extensions, there is a general consensus that a lot of work needs to be done. FPGAs in SoC and datacenters are on the downward slide in Gartner's hype cycle but the same company hints that a significant research effort is underway to optimize the productivity and acceptance in promising application domains [1]. At the same time, FPGAs evolve in speed and complexity to programmable processing systems with an integrated logic fabric and specialized IP cores. In addition, the major vendors have gone long strides to accelerate the design cycle with high level synthesis tools and standardized languages such as OpenCL. This makes the FPGA arena a prosperous playfield for research and innovation. The fifth edition of ParaFPGA, Parallel Computing with FPGAs, follows this trend with papers covering NP-hard problems, image processing, high-level synthesis and heterogeneous computing.

1. Contributions

The keynote paper "*FPGAs as Components in Heterogeneous High-Performance Computing Systems: Raising the Abstraction Leve*l" [2] discusses the historical

[1] Corresponding Author. E-mail: erik.dhollander@elis.ugent.be

initiatives and trends in high level synthesis over the three ages of FPGAs. In order to maintain correctness and productivity, HLS languages should have an affine mapping onto the hardware components. The author advocates a functional language paradigm to express parallel and pipelining operations in an implicit and effective way. While OpenCL is a viable alternative because of its cross platform compatibility, the compute model behind this language foregoes the inherent coarse-grain pipelining capabilities of FPGAs [3].

FPGAs are well fit for 7 non-numerical problems defined in the 13 dwarfs prototype paradigms for parallel computing — a dwarf is an algorithmic method that captures a pattern of computation and communication [4]. Two symposium papers deal with the NP-complete Boolean satisfiability (SAT) problem of the branch and bound dwarf. The first paper "*FPGA Acceleration of SAT Preprocessor*" [5] implements a new branch and bound technique, "unhiding", to reduce the search space and to simplify the Boolean SAT-formula. To this end a binary implication graph (BIG) is generated from the dependencies between the literals. The literals visited during the graph traversal receive several "time stamps" to identify their ordering with respect to several characteristics. Using time stamps it is possible to find tautologies between the clauses. "Unhiding" means detecting and removing these tautologies. The implementation of the unhiding-algorithm on a Kintex FPGA is discussed. Ample parallelism is available in huge formulas with $O(10^6)$ clauses and variables, however due to marginal benefits, the parallelism deployed is limited to 16.

In the second paper on satisfiability, "*Leveraging FPGA clusters for SAT computations*" [6], a cluster of 200 FPGAs is used to find van der Waerden numbers. A van der Waerden number $W(K,L)$ is the smallest integer n such that if the positive consecutive integers $\{1,2,...,n\}$ are partitioned into K classes, then at least one class contains an arithmetic progression of length L, i.e. sequence $\{a, a + d, a + 2d, ..., a + (L - 1)d\}$ for some integers a and d. Earlier development was done on a Beowulf cluster [7]. In order to speed up the computation, a Beowulf cluster is replaced by a cluster of FPGA boards, containing 4 FPGAs, each capable to solve 2 computation tasks. Using dynamic task assignment on 400 solvers, new van der Waerden numbers have been identified in a time frame from 6 to 9 months. The paper describes the hardware and software setup of this application.

Image processing tends to be suited for GPU acceleration. Nevertheless, for applications with a streamlined data access pattern and irregular computations, FPGAs may be an interesting alternative. A case in point is the paper "*High-Speed Calculation of Convex Hull in 2D Images Using FPGA"* [8] where a fast bounding box calculation algorithm is described for 640x480 as well as 1920x1080 monochrome images. Andrew's monotone chain algorithm [9] operates on a stream of sorted input pixels, by repeatedly evaluating the convexity using he incoming points and incrementally building the convex hull. In the FPGA, a "trimming step" calculator eliminates redundant pixels. Next, left- and right half-hull are calculated in parallel and merged into the complete convex hull. The algorithm is I/O bound for small images and compute bound for large images. In the GPU implementation the trimming and partial convex hull calculations do not overlap. As a consequence the FPGA implementation yields a speedup boost of up to 23 times with respect to the GPU implementation. The validity of the technique is shown in a surface-mounted device detection application.

Accelerators are commonplace in today's HPC environment. Still, providing a common programming language to efficiently use architecturally different accelerators such as FPGAs and GPUs is quite a challenge. In "*Workload Distribution and*

Balancing in FPGAs and CPUs with OpenCL and TBB" [10] different accelerator topologies are used to compute the components of a sliding window object detection algorithm (filter, histogram, classifier). Interestingly, all accelerators are programmed with the same OpenCL source code. In the application test case, the accelerators show comparable performance, but the FPGAs have substantially lower power consumption. The results of this experiment demonstrate that one standard parallel programming language may be able to close the semantic gap between accelerators, hindering unbridled use of heterogeneous computing.

The time to load FPGA designs remains expensive. This can be alleviated by partial reconfiguration, which allows sharing computing resources between related tasks. Still a framework is needed to operate dynamic task switching from the connected CPU. The paper "*A Run-Time System for Partially Reconfigurable FPGAs: The case of STMicroelectronics SPEAr board"*[11] describes the experience gained with a development board connecting a Virtex-5 FPGA accelerator daughter board and an ARM Cortex A9 dual-core processor. A run-time system manager is presented which schedules software and hardware tasks, including dynamic reconfiguration of the FPGA. The benefits and possible improvements of the architecture are explored with a hardware Ray Tracer application running on the accelerator in parallel with a software edge detection application running on the processor.

In contrast with accelerators having a fixed architecture, the performance of an FPGA largely depends on the way the hardware is used. This entails exploring the reconfigurable design landscape to maximize the performance and minimize the resource usage. In the paper "*Exploring Automatically Generated Platforms in High Performance FPGAs*" [12] a number of heuristics are presented to guide the selection of memories, bus structures and interfaces. These guidelines are then applied to an image processing and a Monte Carlo simulation application. Experimental results show that employing these rules allows to systematically improve the design.

2. Acknowledgement

The organizers of ParaFPGA15 thank the members of the program committee who provided timely and elaborated reviews helping to maintain the scope and quality of this symposium.

References

[1] M. Reitz, "Competitive Landscape: FPGA Vendors Closing in on ASICs, ASSPs, the IoT and Security as Chip Costs Rise, 2014," Gartner, G00263185, Oct. 2014.

[2] W. Vanderbauwhede, "FPGAs as Components in Heterogeneous High-Performance Computing Systems: Raising the Abstraction Level," in *Proceedings of the conference Parallel Computing 2015, Edinburgh, Series Advances in Parallel Computing, Vol. 27, IOS Press*, 2015.

[3] H.-S. Kim, M. Ahn, J. A. Stratton, and W. W. Hwu, "Design evaluation of OpenCL compiler framework for Coarse-Grained Reconfigurable Arrays," in *Field-Programmable Technology (FPT), 2012 International Conference on*, 2012, pp. 313–320.

[4] K. Asanovic, R. Bodik, B. C. Catanzaro, J. J. Gebis, P. Husbands, K. Keutzer, D. A. Patterson, W. L. Plishker, J. Shalf, S. W. Williams, and others, "The landscape of parallel computing research: A view from Berkeley," Technical Report UCB/EECS-2006-183, EECS Department, University of California, Berkeley, 2006.

[5] M. Suzuki and T. Maruyama, "FPGA Acceleration of SAT Preprocessor," in *Proceedings of the conference Parallel Computing 2015, Edinburgh, Series Advances in Parallel Computing, Vol. 27, IOS Press*.

[6] M. Kouril, "Leveraging FPGA clusters for SAT computations," in *Proceedings of the conference Parallel Computing 2015, Edinburgh, Series Advances in Parallel Computing, Vol. 27, IOS Press*, 2015.

[7] M. Kouril, "A Backtracking Framework for Beowulf Clusters with an Extension to Multi-cluster Computation and Sat Benchmark Problem Implementation," University of Cincinnati, Cincinnati, OH, USA, 2006.

[8] K. Kanazawa, K. Kemmotsu, Y. Mori, N. Aibe, and M. Yasuanga, "High-Speed Calculation of Convex Hull in 2D Images Using FPGA," in *Proceedings of the conference Parallel Computing 2015, Edinburgh, Series Advances in Parallel Computing, Vol. 27, IOS Press*, 2015.

[9] A. M. Andrew, "Another efficient algorithm for convex hulls in two dimensions," *Inf. Process. Lett.*, vol. 9, no. 5, pp. 216–219, 1979.

[10] R. Asenjo, A. Navarro, A. Rodriguez, and J. Nunez-Yanez, "Workload distribution and balancing in FPGAs and CPUs with OpenCL and TBB," in *Proceedings of the conference Parallel Computing 2015, Edinburgh, Series Advances in Parallel Computing, Vol. 27, IOS Press*, 2015.

[11] G. Charitopoulos, D. Pnevmatikos, M. D. Santambrogio, K. Papadimitriou, and D. Pau, "A Run-Time System for Partially Reconfigurable FPGAs: The case of STMicroelectronics SPEAr board," in *Proceedings of the conference Parallel Computing 2015, Edinburgh, Series Advances in Parallel Computing, Vol. 27, IOS Press*, 2015.

[12] P. Skrimponis, G. Zindros, I. Parnassos, M. Owaida, N. Bellas, and P. Ienne, "Exploring Automatically Generated Platforms in High Performance FPGAs," in *Proceedings of the conference Parallel Computing 2015, Edinburgh, Series Advances in Parallel Computing, Vol. 27, IOS Press*, 2015.