Re-form: FPGA-powered true codesign flow for high-performance computing in the post-Moore era

Franck Cappello, Kazutomo Yoshii, Hal Finkel, Jason Cong

Abstract—Multicore scaling will end soon because of practical power limits. Dark silicon is becoming a major issue even more than the end of Moore's law. In the post-Moore era, the energy efficiency of computing will be a major concern. FPGAs could be a key to maximizing the energy efficiency. In this paper we address severe challenges in the adoption of FPGA in HPC and describe "Re-form," an FPGA-powered codesign flow.

1 Introduction

The performance progress of microprocessors has been driven by Moore's law, doubling the number of transistors every 18 to 24 months [2]. In the first three decades, every technology generation with doubled transistor density made the transistor switching 40 % faster and improved energy efficiency 65 %, an effect known as MOSFET scaling or Dennard's law [9]. With 100 nm or smaller feature sizes, however, the static power or the leakage current became too large to ignore [17], requiring the frequency scaling to stop. After the end of Dennard's law, multicore design became mainstream in order to exploit transistor density and support more parallelism. Unfortunately multicore scaling will also end soon primarily because of practical power limits [25]. In fact, dark silicon [12] and the utilization wall [25] are becoming a major concern.

The International Technology Roadmap for Semiconductors forecasts sizes of 7 nm in 2020 and 5 nm in 2023, so there is still another decade before reaching the limit of CMOS technology. However, moving toward a new manufacturing process requires significant investments (e.g., multi billion U.S. dollars), and further parametric variations and leakage current may override the performance benefits from advanced technology. Thus, the end of the transistor size scaling may come earlier.

To overcome this scaling issue, researchers are intensively investigating new structures, new materials, new switching technologies and new manufacturing technologies, including tunneling FETs [20], spintronics [3], carbon nanotubes [24], nanoscale vacuum tubes [16], Josephson junctions [15], and single-atom transistors [14]. Emerging 3D integration of CMOS [26] is one of the most promising solutions to extend the Moore's law, and it will also improve energy efficiency; but it poses several technical challenges such as cost, design complexity, and dynamic thermal variability. Unlike the transition from bipolar to CMOS, however, no technology breakthrough that offers exponential growth is likely to become ready for deployment in the foreseeable post-Moore era.

Another direction is to develop specialized architectures such as Anton [10] and Anton II [23], which were developed by D. E. Shaw for molecular dynamics. A common limitation of these systems, however, is that only few applications will benefit from the special hardware. Clearly, specific architectures require higher nonrecurring engineering costs (NRE).

- *Franck Cappello, Kazutomo Yoshii, Hal Finkel are with Argonne National Laboratory.*
- *Jason Cong is with the University of California, Los Angeles.*

PMES Workshop, Salt Lake City, 14 Nov 2016. `http://j.mp/pmes2016`

Modern FPGA platforms with thousands of hardened digital signal processing (DSP) or floating-point blocks are becoming attractive alternatives because of lower overall NRE compared with specialized architectures. Indeed the adoption rate in other information technology domains has clearly accelerated in the past two years, with a number of significant events. One was the public acknowledgment of the use of FPGAs in datacenters by some of the largest Internet service providers, such as Microsoft and Baidu, for a number of latency-sensitive applications, such as search [22] and speech recognition [21]. Another significant event was Intel's acquisition of Altera, the second largest FPGA company worldwide. These developments indicate that FPGA-based customizable computing is going from advanced research projects into mainstream computing technologies. However, the use of FPGA in scientific computing has been limited for multiple reasons.

In this paper we present our gap analysis of the adoption of FPGA technology in high-performance computing, and we briefly describe "Re-form," an FPGA-powered true codesign flow, which is at an early stage of development.

2 Gap analysis

As of this wiring, multicore or GPU based systems dominate in the Top 500 supercomputer list. Reconfigurable computing and FPGAs in particular have not been adopted broadly by the scientific computing community for five main reasons: capability limits, cost, compilation time, programmability, and performance. Concerning capabilities, FPGAs did not feature enough resources (logic cells and DSPs) to compete with CPUs and GPUs of the same generation on floating-point performance. The cost of high-end FPGAs compared with CPUs and GPUs was detrimental. The compilation of a complex C program with high-level synthesis could take tens of hours, drastically impacting productivity and making performance optimization and debugging difficult. Until recently no parallel programming model existed for programming FPGAs for scientific applications. Moreover, compared with CPUs with similar capabilities, reaching high performance on scientific applications with FPGAs required much more programmer time and wider skills (the programmer needs to know hardware description languages).

The two first reasons (capability limits and cost) are rapidly fading as high-end FPGAs SoCs are integrating significantly more resources and are becoming adopted in extreme-scale data centers. The third reason (compilation time) is related to the place and route step that is proprietary in the FPGA tool

chain and represents a significant issue. Programmability and performance are the primary factors still blocking adoption. These are the two major problems that we discuss below.

Programmability: Scientific programmers of large HPC applications cannot code applications or even kernels at the hardware level (with a hardware description language). Typical applications are written in C, C++ or Fortran, exploiting distributed-memory parallelism wtih MPI and node-level shared-memory parallelism with OpenMP. Moreover, a large portion of HPC codes are being enhanced, or will be enhanced, to use OpenMP4 parallelization and target-offloading directives. However, no production-quality compiler is capable of compiling OpenMP4 codes for FPGAs. FPGA high-level synthesis (HLS) production tools [7], [27] compile ANSI C and C++ (some tools accept other languages), and recent HLS tools can compile codes with OpenCL directives [8]. Other tools translate OpenACC- and OpenMP-like codes into OpenCL codes that HLS tools can compile. The most advanced production-quality OpenMP-like compiler for FPGA, Merlin from Falcon Computing Solutions [6], [13], offers only a limited subset of OpenMP-like constructs. The most advanced OpenMP-like academic compiler (OmpSs [11] from Barcelona Supercomputing Center) does not provide the same level of performance optimizations compared with Merlin. Thus, a significant gap remains between what the programmers need (OpenMP) and what HLS compilers offer (OpenCL and OpenMP like).

Performance: In theory, programmers can reuse their existing OpenCL codes, developed for GPUs, and generate an FPGA design that runs the codes without requiring any hardware skills. In practice, however, this is not the case, as demonstrated by the paper on Gzip compression [1]. The authors used multiple optimizations at the C code level, including rewriting loop cores to optimize the hardware produced by the OpenCL compiler and the rest of the tool chain. This effort required not only deep knowledge of circuit design and FPGA hardware but also a profound understanding of how the OpenCL compiler and tool chain transforms a given code into an FPGA hardware configuration. Thus, an important gap remains between the optimizations that scientific programmers are used to and the optimizations that FPGA requires.

3 Approach

Figure 1 outlines "Re-form," our true codesign flow. Our objective is to provide a compiler frontend that accepts OpenMP4-based codes, applies automatic source-level optimizations, and generates an intermediate representation (e.g., SPIR-V, OpenCL) for underlying HLS tools.

Programmability: We leverage many pre-existing FPGA software technologies and components. Because no OpenMP4 compiler yet exists for FPGA, we will first explore and develop a path for the compilation of OpenMP4 applications, reusing the open-source LLVM/Clang compiler [19] infrastructure and vendor OpenCL/C-based high-level-synthesis (HLS) tools. The parsing, semantic analysis, basic code generation, and runtime system for OpenMP4 have been implemented in the LLVM/Clang compiler framework by contributors to the LLVM project. The output of Clang consumed by the optimizer when processing source code with OpenMP4 directives can be logically divided into two pieces: (1) host (i.e., CPU-targeted) code, which calls the OpenMP runtime library to transfer data between the host and the accelerator and to run functions on the accelerator; and (2) target (i.e., accelerator-targeted) code, which is to run on the accelerator. The OpenMP4 directives provide information to the compiler both about the parallelism in the algorithms and about the data required to run those algorithms on an accelerator. That data is consolidated and transformed by Clang into calls to the OpenMP4 runtime library on the host side. This library is hardware agnostic and supports plug-ins for different kinds of accelerator hardware.

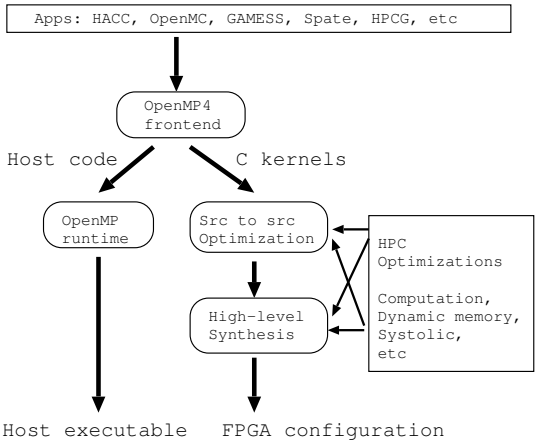

Fig. 1. "Re-form" codesign flow

On a related node, longer FPGA compilation times negatively impacts productively and programmability. Addressing compile-time issues without significantly degrading performance will be challenging. There are some techniques with the potential to help: Creating regular grids of smaller individually-routed elements has been shown to significantly decrease overall place-and-route time [4]. Overlay architectures, running at nearly peak speed, have been demonstrated [5], [18], and targeting a family of such pre-place-and-routed architectures with more-traditional compiler technology may be able to restrict the compile-time problem to cases where tuned synthesis inputs are deemed worthwhile.

Performance: We will explore and develop optimizations at the software and hardware level to improve the performance/watt of the generated FPGA configurations following a cyclical codesign approach: we will design optimizations for software (OpenMP4 parallelization, OpenMP4 compiler, HLS) and hardware (computing and memory interface structures) for HPC motifs, in a coordinated way.

Especially pertaining to off-chip memory, we will employ a library approach to optimize expensive off-chip memory accesses and implement memory interfaces as predefined hardware libraries that can be selected via "extended pragmas," focusing on three memory interface techniques: per-data structure memory hierarchy, irregular access optimization, and data reduction.

4 Conclusion

FPGAs are gaining the spotlight as a computing resource; modern FPGAs include thousands of hard DSPs or floating-point units. In the preparatory stages, we addressed the technology gaps in adopting FPGA technology for HPC. Our goal is to design and implement "Re-form," an FPGA-powered true codesign flow that significantly improves the energy efficiency of the post-Moore era supercomputers.

Acknowledgments

This material is based upon work supported by the U.S. Department of Energy Office of Science, under contract DE-AC02-06CH11357.

REFERENCES

[1] M. S. Abdelfattah, A. Hagiescu, and D. Singh, "Gzip on a chip: High performance lossless data compression on FPGAs using OpenCL," in *Proceedings of the International Workshop on OpenCL 2013 & 2014*, ser. IWOCL '14. New York, NY, USA: ACM, 2014, pp. 4:1–4:9. [Online]. Available: http://doi.acm.org/10.1145/2664666.2664670

[2] D. C. Brock and G. E. Moore, *Understanding Moore's law: four decades of innovation*. Chemical Heritage Foundation, 2006.

[3] M. Cahay, "Spin transistors: Closer to an all-electric device," *Nature nanotechnology*, vol. 10, no. 1, pp. 21–22, 2015.

[4] D. Capalija and T. S. Abdelrahman, "Tile-based bottom-up compilation of custom mesh-of-functional-units fpga overlays," in *2014 24th International Conference on Field Programmable Logic and Applications (FPL)*, Sept 2014, pp. 1–8.

[5] H. Y. Cheah, S. A. Fahmy, and D. L. Maskell, "idea: A dsp block based fpga soft processor," in *Field-Programmable Technology (FPT), 2012 International Conference on*, Dec 2012, pp. 151–158.

[6] J. Cong, M. Huang *et al.*, "Source-to-source optimization for HLS," in *FPGAs for Software Programmers*. Springer, 2016, pp. 137–163.

[7] J. Cong, B. Liu *et al.*, "High-level synthesis for FPGAs: From prototyping to deployment," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 4, pp. 473–491, 2011.

[8] T. S. Czajkowski, U. Aydonat *et al.*, "From OpenCL to high-performance hardware on FPGAs," in *22nd International Conference on Field Programmable Logic and Applications (FPL)*. IEEE, 2012, pp. 531–534.

[9] R. H. Dennard, F. H. Gaensslen *et al.*, "Design of ion-implanted MOSFET's with very small physical dimensions," *IEEE Journal of Solid-State Circuits*, vol. 9, no. 5, pp. 256–268, 1974.

[10] R. O. Dror, C. Young, and D. E. Shaw, "Anton, a special-purpose molecular simulation machine," in *Encyclopedia of Parallel Computing*. Springer, 2011, pp. 60–71.

[11] A. Duran, E. Ayguadé *et al.*, "OmpSs: a proposal for programming heterogeneous multi-core architectures," *Parallel Processing Letters*, vol. 21, no. 02, pp. 173–193, 2011.

[12] H. Esmaeilzadeh, E. Blem *et al.*, "Dark silicon and the end of multicore scaling," in *Computer Architecture (ISCA), 2011 38th Annual International Symposium on*. IEEE, 2011, pp. 365–376.

[13] Falcon Computing Solutions, http://www.falcon-computing.com/index.php/solutions/merlin-compiler/.

[14] M. Fuechsle, J. A. Miwa *et al.*, "A single-atom transistor," *Nature Nanotechnology*, vol. 7, no. 4, pp. 242–246, 2012.

[15] F. Giazotto, "Superconducting transistors: A boost for quantum computing," *Nature Physics*, vol. 11, no. 7, pp. 527–528, 2015.

[16] J. W. Han and M. Meyyappan, "Nanoscale vacuum channel transistor," in *14th IEEE International Conference on Nanotechnology*, Aug 2014, pp. 172–175.

[17] N. S. Kim, T. Austin *et al.*, "Leakage current: Moore's law meets static power," *computer*, vol. 36, no. 12, pp. 68–75, 2003.

[18] C. E. LaForest and J. G. Steffan, "Octavo: An fpga-centric processor family," in *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, ser. FPGA '12. New York, NY, USA: ACM, 2012, pp. 219–228. [Online]. Available: http://doi.acm.org/10.1145/2145694.2145731

[19] C. Lattner, "LLVM and Clang: Next generation compiler technology," in *The BSD Conference*, 2008, pp. 1–2.

[20] H. Lu and A. Seabaugh, "Tunnel field-effect transistors: State-of-the-art," *IEEE Journal of the Electron Devices Society*, vol. 2, no. 4, pp. 44–49, July 2014.

[21] J. Ouyang, S. Lin *et al.*, "SDA: Software-defined accelerator for large-scale DNN systems," in *Proceedings of the HotChips26, Cupertino, CA*, 2014.

[22] A. Putnam, A. M. Caulfield *et al.*, "A reconfigurable fabric for accelerating large-scale datacenter services," in *2014 ACM/IEEE 41st International Symposium on Computer Architecture (ISCA)*, June 2014, pp. 13–24.

[23] D. E. Shaw, J. Grossman *et al.*, "Anton 2: Raising the bar for performance and programmability in a special-purpose molecular dynamics supercomputer," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Press, 2014, pp. 41–53.

[24] M. M. Shulaker, G. Hills *et al.*, "Carbon nanotube computer," *Nature*, vol. 501, no. 7468, pp. 526–530, 2013.

[25] G. Venkatesh, J. Sampson *et al.*, "Conservation cores: reducing the energy of mature computations," in *ACM SIGARCH Computer Architecture News*, vol. 38, no. 1. ACM, 2010, pp. 205–218.

[26] Y. Xie, J. Cong, and S. S. Sapatnekar, *Three-dimensional integrated circuit design*. Springer, 2010.

[27] Z. Zhang, Y. Fan *et al.*, "AutoPilot: A platform-based ESL synthesis system," in *High-Level Synthesis*. Springer, 2008, pp. 99–112.

GOVERNMENT LICENSE