DOI: 10.1109/FPL.2016.7577382

Transparent FPGA Flow

Baptiste Delporte and Anthony Convers and Roberto Rigamonti and Alberto Dassatti
HES-SO — REDS Institute, HEIG-VD — School of Business and Engineering Vaud
CH-1400 Yverdon-les-Bains, Switzerland (*name.surname@heig-vd.ch*)

Heterogeneous computing has recently emerged as a way to circumvent the physical and technological limitations in the design of computing devices. The pressure exerted by the ever-growing demand of increased performances has eventually made the long-awaited dream of having a traditional CPU paired up with an FPGA a reality [1], [2]. FPGAs have proven indeed to be a viable solution for energy efficient high-performance computing [3], [4], [5]. However, while providing the system integrator a very compact and cost effective way to add advanced functionalities to products, this technological evolution has dramatically raised the overall complexity of the system: Exploiting the available capabilities now requires a wide range of competencies which are not always in the background of a company or institution. This requires a considerable effort at development time and often limits the applicability to a reduced set of supported brands/models, while being effective only when predicted usage patterns match the actual ones. High-Level Synthesis (HLS) [6] partially mitigates the above-mentioned problems by removing the language-barrier, but compiling and deploying a bitstream is an extremely long process, and again its development requires establishing a-priori usage patterns that might lead to a suboptimal usage of the available hardware.

In this demo we propose an automated flow that allows the transparent execution of ordinary code on a heterogeneous platform including an FPGA. Our solution requires no change in the code, not even pragma indications to guide the optimization, and dynamically adapts its behaviour to the available data and the workload of the system. Thus, the developer does not need to be aware of the target platform details, nor she has to forecast usage patterns to prevent performance bottlenecks, as the system transparently identifies parallelizable, computationally-intensive code fragments and dispatches them to a data flow overlay architecture built on top of the FPGA. Since the bitstream we use is fixed, and contrary to HLS, we can alter the functionalities offered by the FPGA on-the-fly to adapt them to current usage. Finally, since we operate at the LLVM's Intermediate Representation (IR) level [7], our approach is language-agnostic.

At the heart of our system, depicted by Fig. 1, lies a Just-In-Time (JIT) compiler, coupled with the Linux perf_event performance monitor to automatically detect which code fragments require the largest fraction of resources. Once this region is identified, it is analyzed using Polly [8], a state-of-the-art polyhedral optimizer, to expose parallelization opportunities. The Control Flow Graph and the Data Flow Graph are then extracted and merged, and the overlay pre-synthesized on

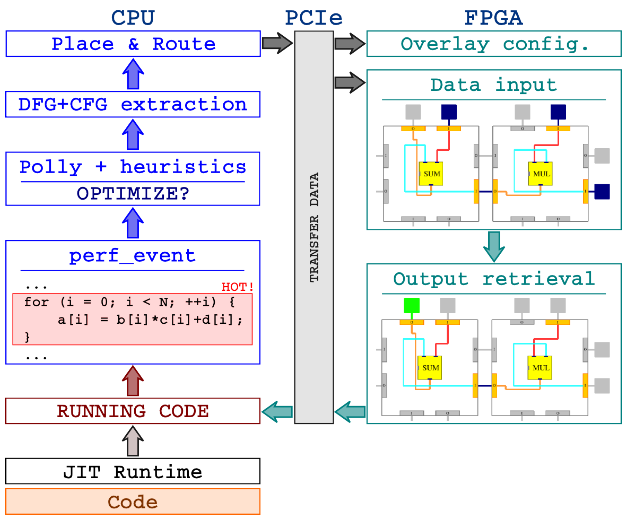

Fig. 1. Schematic representation of the developed system.

the FPGA is reconfigured on-the-fly to execute the new model. We have chosen to adopt the overlay architecture for pipelined execution of data flow graphs presented in [9], and we perform the place&route operation using a custom-made randomized algorithm. Once the overlay is reconfigured — which takes few hundreds of microseconds —, we alter the execution flow of the code as in [10] and we feed the FPGA with the data provided by the application at hand.

To the best of our knowledge, no other approach proposed thus far is capable of achieving these goals.

References

[1] L.H. Crockett et al., *The Zynq Book*. Strathclyde Academic, 2014.
[2] S.R. Alam et al., "Using FPGA Devices to Accelerate Biomolecular Simulations," *Computer*, 2007.
[3] A. Putnam et al., "A Reconfigurable Fabric for Accelerating Large-Scale Datacenter Services," in *ISCA*, 2014.
[4] A. Canis et al., "LegUp: High-level Synthesis for FPGA-based Processor-Accelerator Systems," in *FPGA*, 2011.
[5] R. Chen and V.K. Prasanna, "Accelerating Equi-Join on a CPU-FPGA Heterogeneous Platform," in *FCCM*, 2016.
[6] G. Martin and G. Smith, "High-Level Synthesis: Past, Present, and Future," *IEEE Design Test of Computers*, 2009.
[7] C. Lattner, "LLVM and Clang: Advancing Compiler Technology," in *FOSDEM*, 2011.
[8] T. Grosser and A. Groesslinger and C. Lengauer, "Polly - Performing polyhedral optimizations on a low-level intermediate representation," *Parallel Processing Letters*, 2012.
[9] D. Capalija and T.S. Abdelrahman, "A High-Performance Overlay Architecture for Pipelined Execution of Data Flow Graphs," in *FPL*, 2013.
[10] B. Delporte and R. Rigamonti and A. Dassatti, "Toward Transparent Heterogeneous Systems," in *MULTIPROG-2016*, 2016.