ESL_STR FINAL SUMMARY REPORT

Nikolaos Bellas
Department of Computer and Communication Engineering
University of Thessaly, Volos, Greece
Email: nbellas@inf.uth.gr
Web: http://inf-server.inf.uth.gr/~nbellas/

The problem of automatically generating hardware modules from a high level representation of an application has been at the research forefront in the last few years. The challenge is to close the "architectural gap" between the formal specification of a platform system and its architectural synthesis and final implementation. Such a methodology would potentially enable the large pool of software engineers and algorithm IP experts without architectural and hardware expertise to design and implement platform systems, thus dramatically reducing time to market and increase productivity.

ESL_STR project produced an end-to-end CAD tool that utilizes concepts of the stream and data-parallel paradigms to generate synthesizable co-processors targeting a commercial platform SoC FPGA. We used *OpenCL*, an industry supported data-parallel programming standard for writing programs that execute on heterogeneous platforms and accelerators comprising CPUs, GPUs and other forms of accelerators. Our architectural synthesis tool, dubbed *SOpenCL* (Silicon OpenCL), adapts OpenCL into a novel hardware design flow which efficiently maps coarse and fine-grained parallelism of an application onto an FPGA reconfigurable fabric.

SOpenCL objective is to allow a software programmer to develop an OpenCL application once, and deploy it on any platform, without the need for modifications. The tool consists of a two level compilation process: High Level Compilation (HLC) and Low Level Compilation (LLC). The high level compiler processes an OpenCL application and partitions its kernels as appropriate across the available computing platforms (CPU, GPU, and FPGA). The low level compiler processes OpenCL kernels selected to run on FPGA platforms. The task of the LLC is to compile an OpenCL kernel, and generate an equivalent hardware design that fits the target FPGA device and fulfills performance requirements. SOpenCL tool infrastructure also provides runtime environments for each of the target platforms to facilitate their integration and the execution of OpenCL kernels. The target architecture consists of a general purpose processor and an application-specific accelerator connected to the processor through an interconnect bus (Figure 1).

Our experimentation with a variety of OpenCL and C kernel benchmarks also reveals that area, throughput and frequency optimized hardware implementations are attainable using our methodology. The experimental evaluation showed that our methodology generates fast, area efficient and highly structured hardware accelerators even in the case of complex kernels. Our compiler exploits the structure of OpenCL kernels to generate reconfigurable data path and stream units that match the characteristics of the application and the user requirements in terms of throughput and area limitations.

For example, SOpenCL was used to quickly implement complex algorithms such as Low-Density Parity-Check (LDPC) decoders used in modern communication systems. OpenCL was used as a common programming model for developing parallel kernels on multicore CPUs, GPUs and FPGAs without code readjustment or hand tuning across different parallel platforms. In this context SOpenCL allows code portability across different multicore platforms at no extra programming effort or particular need of code hand tuning interventionn

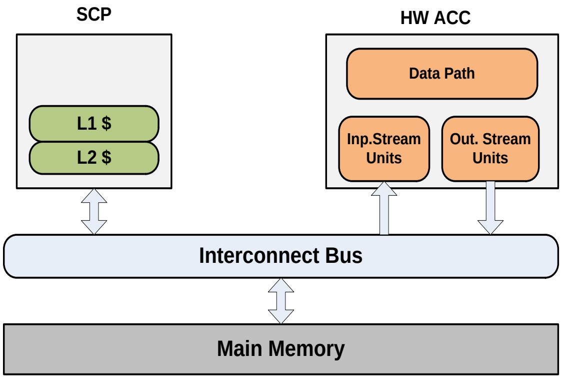

Figure 1. The target System On Chip design. SCP stands for Scalar Processor.

The main target objective of ESL_STR was to investigate the applicability of a single programming representation of an application to be ported both in a heterogeneous platform consisting of fixed architectures, and also as hardware accelerator in a reconfigurable platform. Using OpenCL as a programming abstraction for heterogeneous platforms consisting of both fixed and reconfigurable architectures promises to place novel technologies like FPGAs at the forefront of mainstream and high performance computing.