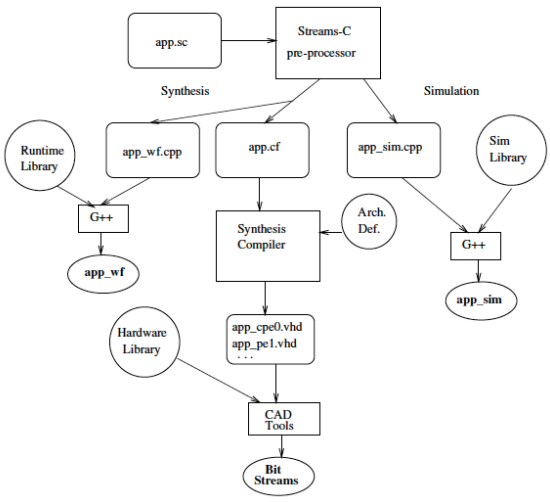
Stream-Oriented FPGA Computing in the Streams-C High Level Language

Maya Gokhale, Jan Stone, Jeff Arnold, Mirek Kalinowski

Year of publication: *2000*
Area: *Languages and Compute Models*



An important step in popularizing the field-programmable custom computing machine approach is to provide design tools that enhance designer productivity by enabling application development using high-level programming languages, rather than hardware description languages such as VHDL or Verilog. However, such productivity enhancement must not sacrifice the performance benefit of the FCCM implementation. This issue was particularly acute in the 1990's when field-programmable gate arrays had a limited amount of resources. If application implementations did not meet timing requirements or did not fit within the target devices, the high-level programming language solution would not be usable.

This paper demonstrates a pioneering effort to meet the challenge of providing design tools for field-programmable custom computing machines that support both designer productivity and design efficiency. There are three reasons for its significance.

First, the proposed approach focuses on stream-oriented computing which has an abundance of data parallelism that can be effectively mapped onto resources of field-programmable gate arrays. This focus allows the tools to cover applications with inherent concurrency without limiting them to a single narrow application domain. Second, the proposed tools support a subset of the C language with annotations to direct a compiler to generate both hardware circuits targeting one or more field-programmable devices, and a multi-threaded control program for a host processor that communicates with the field-programmable devices. The use of annotations has been adopted by various hardware compilers (e.g., the AutoPilot system from AutoESL). Third, the paper addresses the difficulty issue of productivity evaluation. For an image processing application involving contrast enhancement, it reports that designer productivity was improved by 10 times, at the expense of a speed reduction by half and a three times increase in area, when compared with a hand-coded version in VHDL. Just like advances in software compilers, the efficiency gap between implementations from hardware compilers and from hand-coded designs would increasingly narrow.

It is a tribute to this paper that many of its ideas continue to live on in the commercial Impulse-C compiler, which has been advertised for a range of applications from scientific computing to secure communications.

Wayne Luk

DOI: http://dx.doi.org/10.1109/FPGA.2000.903392