Introduction to Part 2

Current approaches to embedded compiler development are no longer sustainable. With each generation of embedded architecture, the development time increases and the performance improvement achieved decreases. As high-performance embedded systems move from application-specific integrated circuits to programmable multi-core parallel systems, this problem will become critical. This section of the volume investigates emerging alternative approaches to compiler construction exploring adaption, parallelization, speculation and automatic tuning.

The first paper by Nicholas Nethercote, Doug Burger and Kathryn S. McKinley investigates the idea of convergent compilation where optimization is refined on each specific compilation. This modification is based on a static evaluation of optimization effectiveness. When applied to loop unrolling on the TRIPS architecture, it automatically selects the best versions found by hand-tuning.

The second paper, by Harald Devos, Kristof Beyls, Mark Christiaens, Jan Van Campenhout, Erik H. D'Hollander, and Dirk Stroobandt, studies loop transformations, based on a polyhedral model, to better exploit FPGA. Important loops are detected in applications then refactored to finally issue VHDL code.

The third paper by Ghaffari Fakhreddine, Auguin Michel, Abid Mohammed, and Benjemaa Maher addresses the issue of partitioning code at run-time for soft real-time applications and an FPGA-based target. This method is particularly significant when the execution flow is dependent on the content of input data.

The fourth paper, by Shane Ryoo, Sain-Zee Ueng, Christopher I. Rodrigues, Robert E. Kidd, Matthew I. Frank, and Wen-mei W. Hwu, explores the impact that different static analysis techniques have, both in isolation and in conjunction, when used to automatically parallelize modern media applications. The paper then presents one case study using an MPEG-4 encoder.

Finally, the fifth paper, by Guilin Chen and Mahmut Kandemir, addresses the issue of minimizing the number of off-chip accesses in a chip-multiprocessor by ensuring that computations are organized such that adjacent cores that share data can share recently fetched values. The work is concerned with array transformations of stencil computations which are common in embedded applications.

Mike O'Boyle
University of Edinburgh

François Bodin
IRISA

Marcelo Cintra
University of Edinburgh

Guest editors
Transactions on HiPEAC

P. Stenström (Ed.): Transactions on HiPEAC I, LNCS 4050, p. 139, 2007.