Architecture and Compilation techniques for CELL

Abstract

The recently announced CELL processor, which targets multimedia and gaming applications incorporates several interesting architectural features to better support such highly parallel, compute intensive codes. CELL delivers high performance computing capabilities for such compute intensive codes by exploiting chip multiprocessing to address inherent power and memory latency issues. This first generation CELL processor provides an unparalleled level of compute performance on a single chip with a collection of heterogeneous processors, including a POWER processor with two levels of cache and eight attached streaming processors with their own local memories and globally consistent DMA engines. In addition to processor-level parallelism, each processing element has Single Instruction Multiple Data (SIMD) units that can each process from 2 double floats up to 16 chars per cycle.

The tutorial material will be presented in two parts. In the architecture section we will describe the CELL Architecture and provide an in-depth discussion of the various design motivations and tradeoffs. In addition to the CELL system architecture, we will also describe the novel SIMD-based architecture of the streaming processor unit, its motivations, and its integration in the CELL system. The second part of the tutorial will deal with the compilation challenges involved in generating high performance code from this novel architecture. This will include issues such as how to generate good scalar code in a SIMD engine where significant hardware function has been offloaded to software, how to exploit the multiple levels of parallelism, and how the compiler can provide a single system abstraction of the underlying heterogeneous processors with attached local memories. The discussion will be in the context of a functioning prototype compiler for the Cell architecture.

Intended Audience

This tutorial is intended for those with a background in Computer Architecture and Compiler Writing. Some knowledge of parallelization techniques would also be useful.

The Speakers

- Kevin O'Brien has spent the last 24 years at IBM working in the field of compilation and architecture. Initially, at the IBM Toronto Lab, he was the architect of the TOBEY optimizing backend (used in IBM's xlc, xlf, and xlC (C++) product compilers). Subsequently, Kevin has spent 17 years at IBM Research, where his research interests have included Multithreaded Architecture, Smalltalk, Java, continuous optimization, binary translation and optimization, parallelization and vectorization (including SIMDization), for several processors, most recently the Cell processor. He has also contributed to the architectures of Power, PowerPC and Cell. Currently, Kevin is investigating memory related optimizations for the Cell processor.
- Alexandre Eichenberger is a compiler researcher at IBM TJ Watson Research Center and has been involved in the CELL compiler project since its inception. During the initial port of the IBM XL production compiler to the SPEs (the 8 SIMD-centric attached processors in the CELL architecture), Dr. Eichenberger addressed SPE-specific scheduling and bundling issues, including compiler techniques to prevent instruction fetch starvation. Dr. Eichenberger currently contributes to the automatic generation of SIMD code targeting the SIMD units found in the CELL (SPE/VMX), Power (VMX), and BlueGene/L (double-precision floating-point) architectures, focusing on data alignment

related issues. Prior research interests include instruction-level parallelism, predicated execution, profiling techniques, and software pipelining.

- Kathryn O'Brien has worked at IBM for 23 years, 17 of them as a researcher at IBM TJ Watson Research Center, where she has been involved in several static and dynamic compiler projects. She was involved in the initial IBM XL Fortran compiler, and the early vectorization and parallelization efforts in the XL compilers. Most recently she has played a key role in prototyping compilers for the CELL architecture, where her specific interests are in both scalar compilation techniques and in compiler exploitation of the multiple levels of parallelism through a single source compiler.
- Peng Wu is a compiler researcher at IBM TJ Watson Research Center. She is a primary contributor to the simdization optimization in the CELL compiler. Currently Dr. Wu leads the effort to build a general simdization infrastructure that targets a variety of SIMD units found in the CELL, Power, and BlueGene/L. In the early prototyping of the CELL compiler, Dr. Wu has worked on various backend scalar optimizations and played a key role in defining SPE intrinsic.
- Michael Gschwind is a computer architect at the IBM TJ Watson Research Center. Dr. Gschwind was one of the initiators of the CELL project at IBM TJ Watson Research Center, a leading contributor to the initial CELL system architecture definition and one of the lead architects of the novel SIMD-centric SPU architecture used in CELL. During the definition of the SPU architecture, Dr. Gschwind also developed the first SPU prototype compiler. Prior to the inception of the CELL project, Dr. Gschwind contributed to the development of the DAISY tree-based VLIW core and was an architect for the BOA high-frequency VLIW design, based on advanced dynamic compilation technology to achieve compatible PowerPC implementations. Dr. Gschwind is an IBM Master inventor and holds patents on dynamic compilation, VLIW architecture, media processing technology, and computer microarchitecture.