PEPPHER: Performance Portability and Programmability for Heterogeneous Many-core Architectures

Sabri Pllana
(project coordinator)
University of Vienna
Project Consortium

Universities
- University of Vienna (coord.), Austria
- Chalmers University, Sweden
- Karlsruhe Institute of Technology, Germany
- Linköping University, Sweden
- Vienna University of Technology, Austria

Research center
- INRIA, France

Companies
- Intel, Germany
- Codeplay Software Ltd., UK
- Movidius Ltd. Ireland
Addressed Issue(s) and the PEPPHER Approach
PEPPHER addresses heterogeneous systems
- Single-node (instance: CPU and GPU/MIC*)
- Single-chip (instance: APU, Cell BE)

We do not propose a new programming model/language
- Different programming models/languages may be suitable for different core types

Aim: enable combination of existing programming models/languages

* Intel® Many Integrated Core Architecture. Intel is a trademark of Intel Corporation in the U.S. and/or other countries. (www.intel.com)
Aim: Application Runs on Any Hardware

Application types:
Embedded, General-purpose, HPC

Single Application Description

performance-portable execution
on various multi/many-core architectures

Sabri Pilana (University of Vienna)
PEPPHER Holistic Approach: Five Layers

Application types: Embedded, General-purpose, HPC

1. component-based software development
2. portable compilation techniques
3. adaptable algorithms and data structures
4. flexible run-time system
5. hardware support mechanisms for portability

HW 1  HW 2  HW 3  HW 4  HW 5  HW 6  HW 7  ...  HW N
Overview
Applications

PEPPHER targets applications from various domains
- from small kernels to larger programs
- Applications
  - KIT: Suffix array construction
  - UNIVIE: Bzip2, OpenCV
  - Codeplay: Bullet (games physics simulation)
  - Movidius: Computational photography
  - Intel: GROMACS
- Kernels
  - INRIA: FFT
  - INRIA: MAGMA/PLASMA (QR)
  - INRIA: RODINIA (CFD solver)
  - KIT: STL (sort, find, random_shuffle)
1: Component-based SW Development

- **Component-based SW Development**
  - **PEPPHER component**
    - implements a specific functionality declared in an interface
    - annotated software module
    - rich meta-data (e.g. perf. model)
    - performance-aware
  - **Implementation variants**
    - target execution platform
    - algorithm
    - programming model/language
    - ...

**Applications**
- Embedded, General Purpose, HPC

**PEPPHER Components**
- C/C++, OpenMP, CUDA, OpenCL, Offload, TBB

**Performance Models**

---

**Program**

Statement_0
Statement_1

#pragma ...
Function_c

Statement_n

**Performance-critical parts of program (“hot spots”)**
- are candidates for PEPPHER components

**Abstract Interface - Functional Definition**

**Concrete implementation for Interface**

**Component Variant**
- Variant Meta-Data
  - execution target platform
  - performance model for aspects
  - invocation interface
  - parameter assertions

**Component Interface**
- Interface Meta-Data
  - abstract function interface
  - supported performance aspects

**Concrete implementation for Interface**

**Component Variant**
- Variant Meta-Data
  - execution target platform
  - performance model for aspects

---

Sabri Pllana (University of Vienna)
2: Portable Compilation Techniques

- Source to source transformation
  - target the run-time system
  - generate/preselect component implementation variants

- Offload C++
  - compiler and run-time system for offloading parts of C++ applications to run on accelerator cores

- OffloadCL
  - generates OpenCL code for host and OpenCL device from annotated Offload C++ source code

Offload™ is a trademark of Codeplay Software Ltd (www.codeplay.com)

Sabri Pilana (University of Vienna)
3: Algorithms and Data Structures

- Adaptable algorithms and data structures
  - expert-written libraries
  - provide component implementation variants

- Algorithmic toolbox
  - generate different implementation variants based on compile-time architecture-dependent tuning parameters

- Synchronization library
  - lock-free templated data structures for CPU and GPU
4: Flexible Run-time System

- PEPPHER run-time system
  - component meta-data supports scheduling decisions
  - dynamic scheduling of tasks on a pool of heterogeneous cores
  - provides a Virtual Shared Memory subsystem
  - provides performance feedback

- Open scheduling platform
  - scheduling algorithm = plug-in

- Tasks
  - data input & output
  - dependencies with other tasks
  - multiple implementations (GPU, CPU)

Applications
Embedded, General Purpose, HPC

PEPPHER Components
C/C++, OpenMP, CUDA, OpenCL, Offload, TBB

Performance Models
Autotuned Algorithms, Data Structures

Transformation & Composition
PEPPHER Task Graph

PEPPHER Run-time
Data Management (Virtual Shared Memory,..)
Schedulers (HEFT, Work-stealing,..)
Drivers (CPU, MIC, CUDA, OpenCL, Cell,..)

CPU & GPU/MIC
APU
5: Hardware Support Mechanisms

- **PeppherSim**
  - simulates existing or conceptual architectures
  - provides an OpenCL interface
  - supports temporal and energy metrics
  - enables investigation of new synchronization primitives

- Integration of run-time system with PeppherSim
  - generation of temporal and energy-consumption performance models

- **PePU (work in progress)**
  - PEPPHER Processing Unit
  - a demonstration hardware platform
  - comprises multiple Movidius SABRE SoCs with an FPGA
Putting It All Together

- **Applications**: C/C++ source code with annotated components
- **Component implementation variants for various hardware, input characteristics, and optimization criteria**
- **Variants may be parallelized in the most suitable framework, or supplied by “expert” programmers as part of libraries**
- **Transformation and compilation techniques support the variant generation/preselection**
- **Intermediate representation**: component task graph with explicit data dependencies
- **PEPPHER run-time selects dynamically variants and schedules on the available resources**
- **Hardware mechanisms for synchronization and performance monitoring**
Expected Impact and Beyond PEPPHER
Expected Impact

- **Strengthen the European excellence in heterogeneous multi-core systems**
  - high-level software development, compilation technologies, algorithms and data structures, run-time systems, hardware support for programmability and portability

- **Industrial use of results**
  - OffloadCL by Codeplay
  - PeppherSim and PePU by Movidius
  - potential take-up of interesting PEPPHER technology by Intel

- **Academic use of results for research-driven teaching**
  - deliver state-of-the-art knowledge from this domain to students

---

**First achievements**

- PEPPHER source-code transformation system
- Tuned sorting algorithms for multi-core and GPU with world-leading performance
- OffloadCL compiler generates OpenCL code from annotated Offload C++ source code
- StarPU-based run-time system supports various schedulers, target devices, power-based optimization
- PeppherSim simulator supports temporal and energy metrics
Address fundamental parallel programming issues
- funding scheme: FET Open, ERC grants, ...

Investigate resource-aware parallel programming techniques
- energy-awareness
- architectural support for resource-efficient parallel programming

Develop intelligent software development environments
- programming environment supports proactively the programmer
- automation & autonomy
- Pllana et al. LNCS 5415, pp. 137–147, Springer 2009
Acknowledgments

- European Commission (ec.europa.eu)
- HiPEAC Network of Excellence (www.hipeac.net)
- PEPPHER Consortium (www.peppher.eu)