OSCAR Compiler and OSCAR API for Heterogeneous Computing

Keiji Kimura, Waseda University
What is OSCAR Compiler?

- **OSCAR Automatically Parallelizing Compiler**
  - Multigrain Parallel Processing
  - Data Locality Optimization
  - Data Transfer Optimization
  - Low-Power Optimization
  - Good for Embedded Applications as well as Scientific Applications

  - Parallel API for Low-power and Real-time Multicores
    - Developed by CATS, DENSO, e-SOL, Fujitsu, Fujitsu Laboratory, GAIO Technology, Hitachi, MITSUBISHI Electric, NEC, Olympus, Panasonic, Renesas Electronics, Renesas Solutions, Toshiba, Toho University, Nagoya University and Waseda University in METI/NEDO Project
  - Supporting Local Memory in addition to Shared Memory
    - Local Memory (Scratch Pad Memory), Distributed Shared Memory, On-chip/Off-chip Centralized Shared Memory
  - Supporting power control mechanism
    - DVFS, Clock Gating, Power Gating
  - **Supporting accelerators** and many-cores
    - from Version 2.0
  - Using as an interface between OSCAR Compiler and various Multicores
Main Optimizations by OSCAR Compiler

**Multigrain Parallel Processing**
- Hierarchical and Global Parallelization
  - Coarse grain task parallel
  - Loop iteration parallel
  - Statement level parallel

**Data Locality Optimization**
- Task (or loop) decomposition considering cache size or local memory size
- Task scheduling considering data affinity

**Low power optimization**
- Power scheduling with DVFS and power gating by software

Task level or statement level parallelization
Overview of OSCAR API v2.0

- Targeting mainly real-time Embedded Computing
  - Various kinds of memory architecture
    - SMP, local memory, distributed shared memory, non-coherent cache ...
  - Power control mechanisms
  - Accelerators
- Based on the subset of OpenMP
  - Very popular parallel processing API
  - Shared memory programming model
  - Supporting both of C and Fortran
- Eight Categories
  - Parallel Execution
  - Memory Mapping
  - Data Transfer
  - Power Control
  - Timer
  - Synchronization
  - **Accelerator**
  - Cache Control
Application Development Environment with OSCAR Compiler, OSCAR API

OSCAR Compiler
- Multigrain parallel processing
- Data locality optimization
- Data Transfer optimization
- Low power optimization

Written in Fortran or Parallelizable C

Parallelized Fortran or C code with OSCAR API 2.0
Consumer Electronics Multicore: RP2 (ISSCC 2008)

<table>
<thead>
<tr>
<th>Process Technology</th>
<th>90nm, 8-layer, triple-Vth, CMOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chip Size</td>
<td>104.8mm²</td>
</tr>
<tr>
<td></td>
<td>(10.61mm x 9.88mm)</td>
</tr>
<tr>
<td>CPU Core Size</td>
<td>6.6mm²</td>
</tr>
<tr>
<td></td>
<td>(3.36mm x 1.96mm)</td>
</tr>
<tr>
<td>Supply Voltage</td>
<td>1.0V–1.4V (internal), 1.8/3.3V (I/O)</td>
</tr>
<tr>
<td>Power Domains</td>
<td>17 (8 CPUs, 8 URAMs, common)</td>
</tr>
</tbody>
</table>

Shonan Meeting/Keiji Kimura 18/09/03
Scalability Evaluation on RP2

2.9 times speedup on average
Low-Power Optimization and OSCAR API on MPEG2 decoder

Without Power Control (Voltage: 1.4V)

With Power Control (Frequency, Resume Standby: Power shutdown & Voltage lowering 1.4V-1.0V)

Avg. Power
5.41 [W]

76.0% Power Reduction

Avg. Power
1.30 [W]
 Compile flow for Heterogeneous Multicores

Source program

Compiler for ACCa ➔ Source with Hint directives ➔ Compiler for ACCb ➔ Source with Hint directives ➔ Compiler for ACCz ➔ Source with Hint directives

Chip Configuration Information

OSCAR Compiler

for CPU and each ACCs

for CPUs (C or Fortran)

for ACCa_0 functions

for ACCz_0 functions

Compiler for ACCa

Compiler for ACCz

for CPU and each ACCs

for CPUs (C or Fortran)

for ACCa_0 functions

for ACCz_0 functions

Compiler for ACCa

Compiler for ACCz

Heterogeneous directives

• #pragma oscar_hint
  • accelerator_task

Homogeneous directives

• #pragma oscar
  • accelerator_task_entry

Inserting hint directives for OSCAR Compiler

Chip Configuration Information

API Translator ➔ CPU code ➔ Backend Compiler ➔ Linker ➔ Executable object

Accelerator Library

Shonan Meeting/Keiji Kimura

18/09/03
Compile flow for Heterogeneous Multicores

- Source program
- Compiler for ACCa
- Source with Hint directives
- Compiler for ACCb
- Source with Hint directives
- Compiler for ACCz
- Source with Hint directives

Chip Configuration Information

- OScAR Compiler
- for CPU and each ACCs
  - for CPUs (C or Fortran)
    - Homogeneous directives
  - for ACCa_0 functions
  - for ACCz_0 functions

- Compiler for ACCa
- ACCa_0 control code
- ACCa_0 object

- Compiler for ACCz
- ACCz_0 control code
- ACCz_0 object

- API Translator
- CPU code
- Backend Compiler
- Linker

Accelerator Library

Executable object

Inserting hint directives for OScAR Compiler

Hint directives
- #pragma oscar_hint accelerator_task
- #pragma oscar_comment

Heterogeneous directives
- #pragma oscar accelerator_task_entry

Shonan Meeting/Keiji Kimura
18/09/03
Optimizing compiler flow for Heterogeneous Multicores

- Source program
- Compiler for ACCa
- Source with Hint directives
- Compiler for ACCb
- Source with Hint directives
- Compiler for ACCz
- Source with Hint directives
- Chip Configuration Information
- OSCAR Compiler
- for CPU and each ACCs
  - for CPUs (C or Fortran)
  - for ACCa_0 functions
  - for ACCz_0 functions
- Compiler for ACCa
- Compiler for ACCz
- ACCa_0 control code
- ACCa_0 object
- ACCz_0 control code
- ACCz_0 object
- API Translator
- CPU code
- Backend Compiler
- Linker
- Executable object
- Inserting hint directives for OSCAR Compiler
- Hint directives
  - #pragma oscar_hint accelerator_task
  - #pragma oscar_comment
- Heterogeneous directives
  - #pragma oscar accelerator_task_entry

Shonan Meeting/Keiji Kimura 18/09/03
Low-power consumer electronics heterogeneous multicore “RP-X”
(ISSCC 2009)

Cluster #0

SH-4A

SHwy#0 (Address=40, Data=128)

DBSC #0

DMAC #0

VPU5

SHPB

SHwy#2 (Address=32, Data=64)

PCI exp

SATA

SPU2

LBSC

SHwy#1 (Address=40, Data=128)

FE #0-3

DMAC #1

DBSC #1

MX2 #0-1

SNC

Y. Yuyama, et al., "A 45nm 37.3GOPS/W Heterogeneous Multi-Core SoC”,
ISSCC2010

developed by Renesas, Hitachi, Titech, and Waseda

Cluster #1

SH-4A

Cluster #1

SHwy#0 (Address=40, Data=128)

DBSC #0

DMAC #0

VPU5

SHPB

SHwy#2 (Address=32, Data=64)

PCI exp

SATA

SPU2

LBSC

SHwy#1 (Address=40, Data=128)

FE #0-3

DMAC #1

DBSC #1

MX2 #0-1

SNC

Y. Yuyama, et al., "A 45nm 37.3GOPS/W Heterogeneous Multi-Core SoC”,
ISSCC2010

developed by Renesas, Hitachi, Titech, and Waseda
Scalability of Optical Flow (Demo)

<table>
<thead>
<tr>
<th></th>
<th>Homogeneous</th>
<th>Heterogeneous</th>
</tr>
</thead>
<tbody>
<tr>
<td>1SH</td>
<td>3.5 [fps]</td>
<td>32.65</td>
</tr>
<tr>
<td>2SH</td>
<td>2.29</td>
<td>26.71</td>
</tr>
<tr>
<td>4SH</td>
<td>3.09</td>
<td></td>
</tr>
<tr>
<td>8SH</td>
<td>5.4</td>
<td></td>
</tr>
<tr>
<td>2SH+1FE</td>
<td>18.85</td>
<td></td>
</tr>
<tr>
<td>4SH+2FE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8SH+4FE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Speedups against a single SH processor
Low power optimization on Optical Flow (Demo)

Without Power Reduction

With Power Reduction by OSCAR Compiler

70% of power reduction

Average: 1.76[W] → 0.54[W]

1 cycle: 33[ms] → 30[fps]

Shonan Meeting/Keiji Kimura 18/09/03
OSCAR with Vector Accelerator
Compile flow for OSCAR + Vector

1. Source Code
2. OSCAR Compiler (Heterogeneous Parallelization)
3. Backend Compiler for Host CPU core (gcc, llvm, ...)
4. C + OSCAR API
5. Backend Compiler for Vector Core (LLVM w/ our extension)
6. C + Vector Intrinsics
7. Linker
8. Executable Binary for Target Heterogeneous Chip
Conclusions

- **OSCAR Compiler**
  - Automatically parallelizing C and Fortran Programs
    - Multigrain Parallelization
    - Memory Optimization
    - Low Power Optimization
  - Targeting on Heterogeneous Multicores as well as Homogeneous Multicores
  - OSCAR API as an Interface between OSCAR Compiler and Homogeneous/Heterogeneous Multicores

- **Our ongoing project**
  - Development of New Accelerators to Attain More Performance with Low Power Consumption
  - Of course, with new Compilation Technologies