OSCAR Compiler and OSCAR API for Heterogeneous Computing

Keiji Kimura, Waseda University

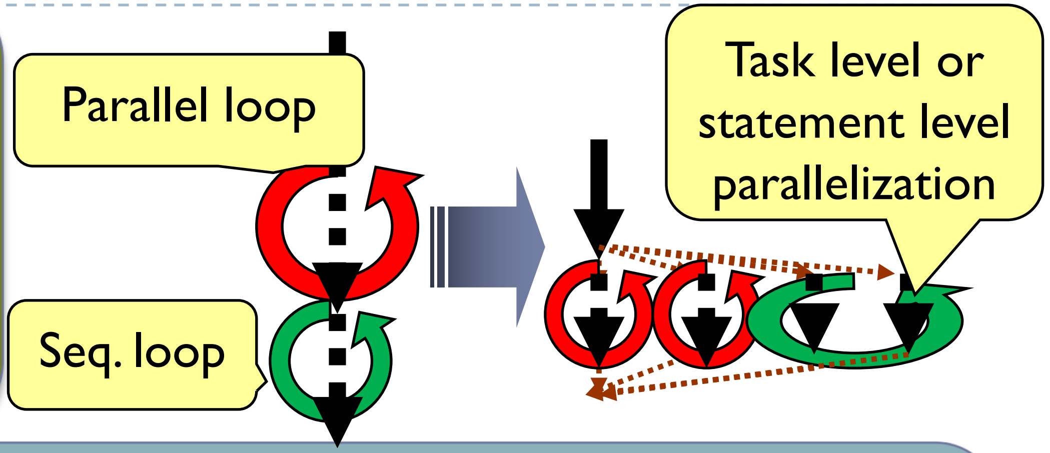
What is OSCAR Compiler?

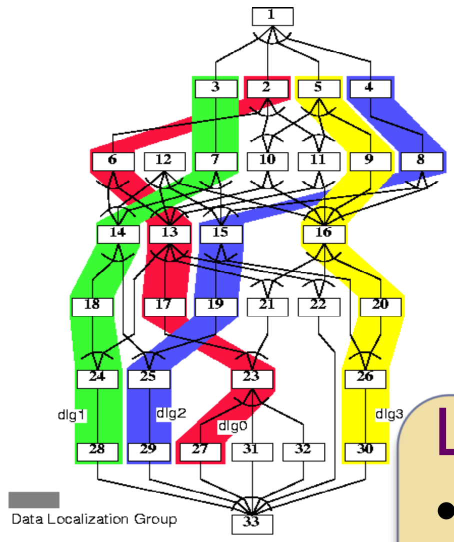
- OSCAR Automatically Parallelizing Compiler
  - Multigrain Parallel Processing
  - Data Locality Optimization
  - Data Transfer Optimization
  - Low-Power Optimization
  - Good for Embedded Applications as well as Scientific Applications
- OSCAR API (http://www.kasahara.cs.waseda.ac.jp/)
  - Parallel 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



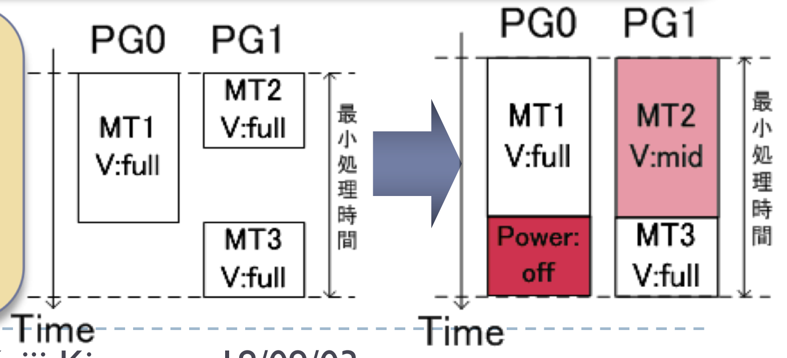


Data Locality Optimization

- Task (or loop) decomposition considering cache size or local memory size
- Task scheduling considering data affinity

Low power optimization

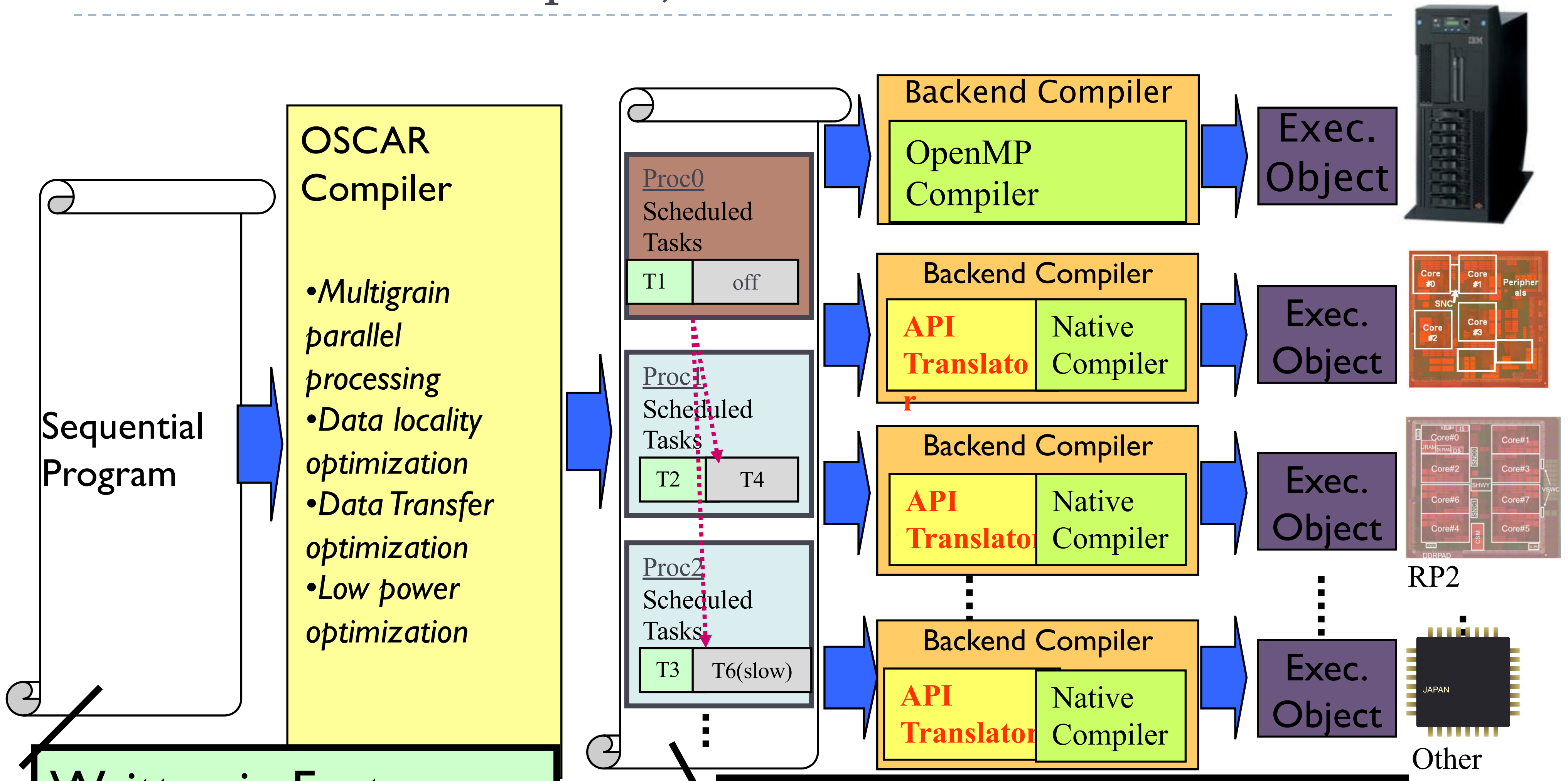
- Power scheduling with DVFS and power gating by software



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

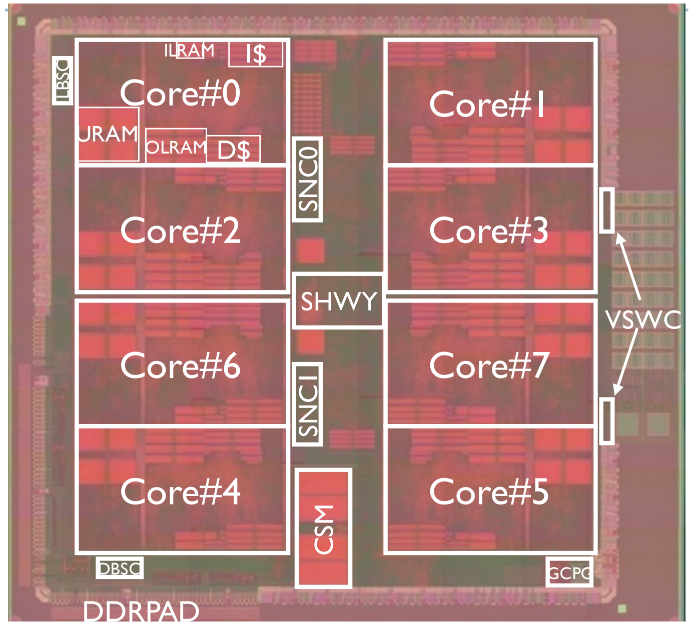
Application Development Environment with OSCAR Compiler, OSCAR API

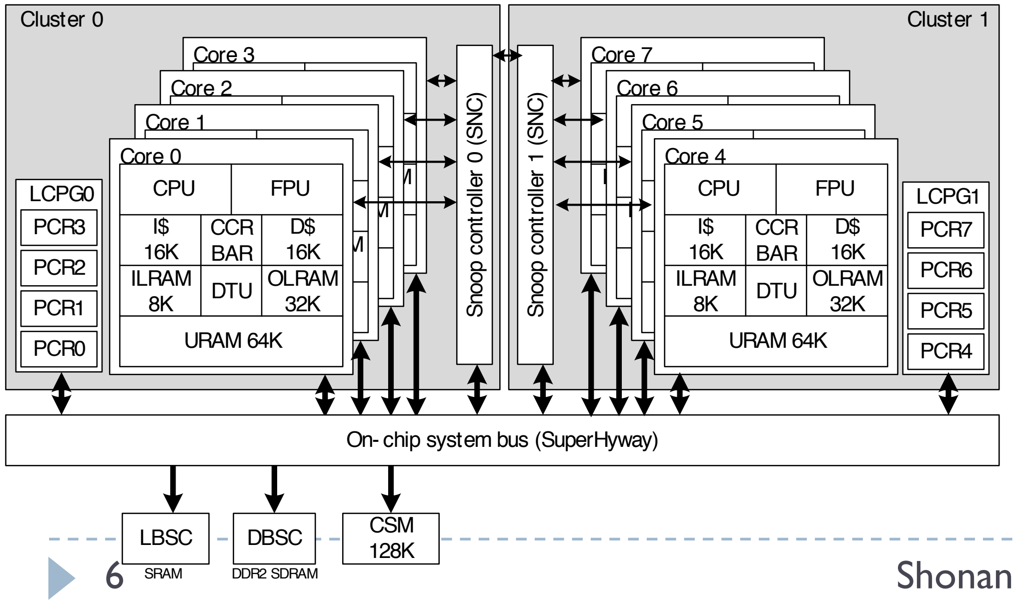


Written in Fortran or Parallelizable C

Parallelized Fortran or C code with OSCAR API 2.0

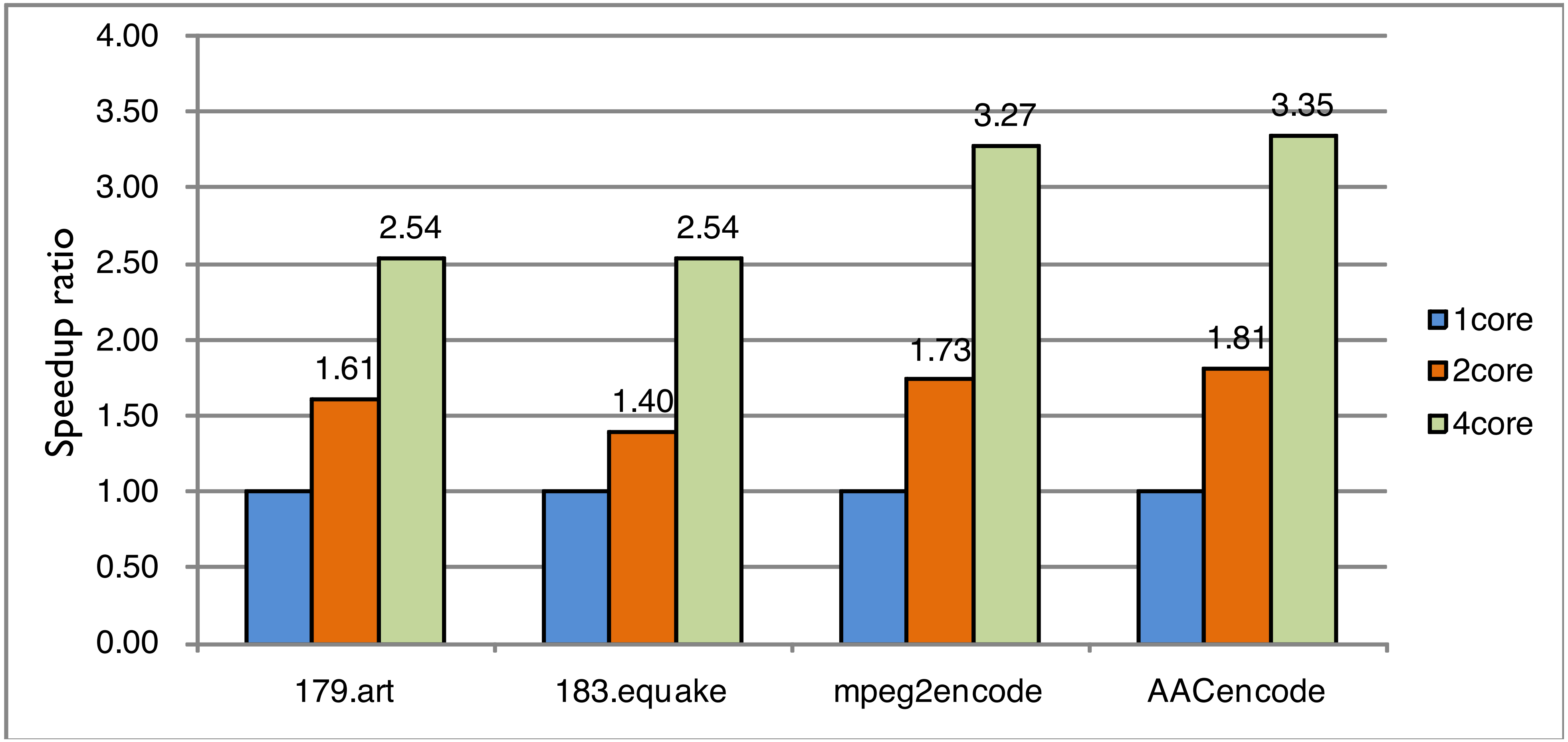
Consumer Electronics Multicore: RP2 (ISSCC 2008)





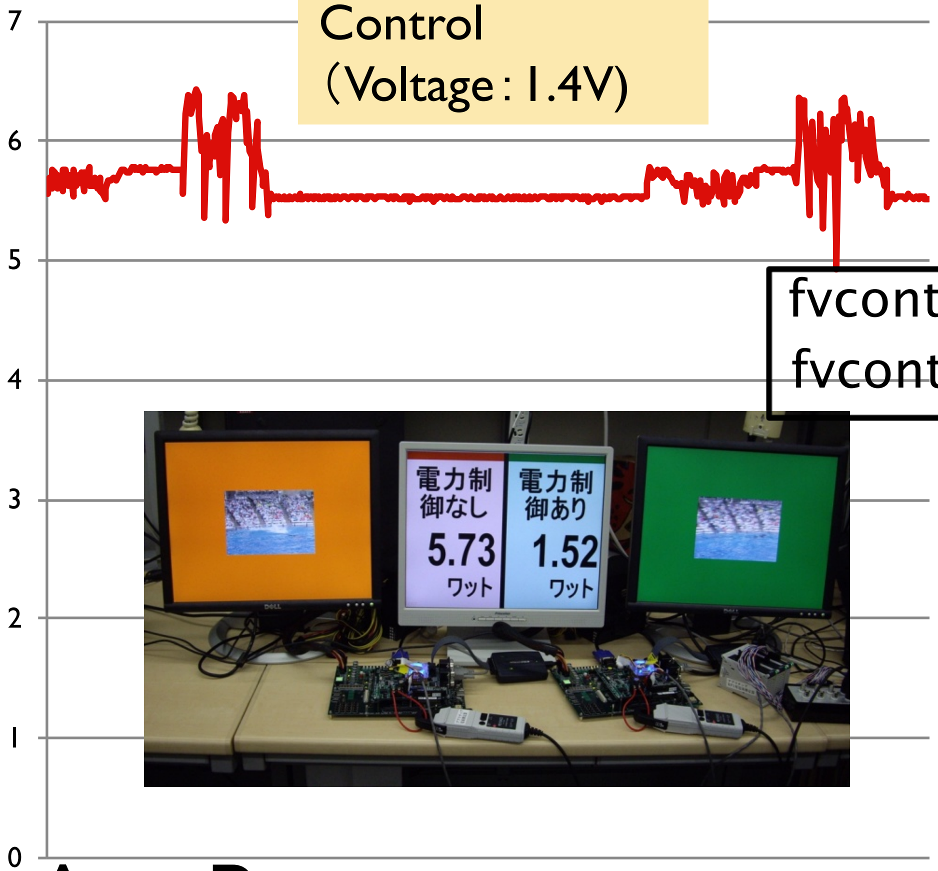
Process Technology	90nm, 8-layer, triple-Vth, CMOS
Chip Size	104.8mm^2 (10.61mm x 9.88mm)
CPU Core Size	6.6mm^2 (3.36mm x 1.96mm)
Supply Voltage	1.0V–1.4V (internal), 1.8/3.3V (I/O)
Power Domains	17 (8 CPUs, 8 URAMs, common)

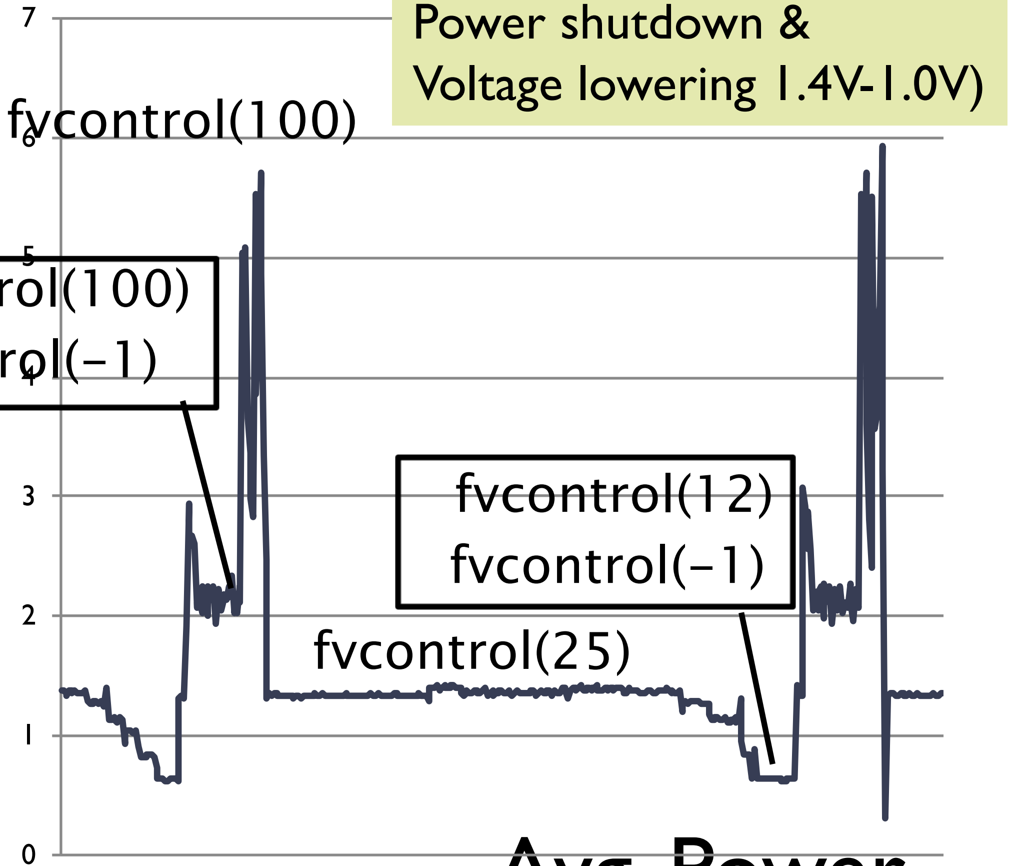
Scalability Evaluation on RP2



2.9 times speedup on average

Low-Power Optimization and OSCAR API on MPEG2 decoder



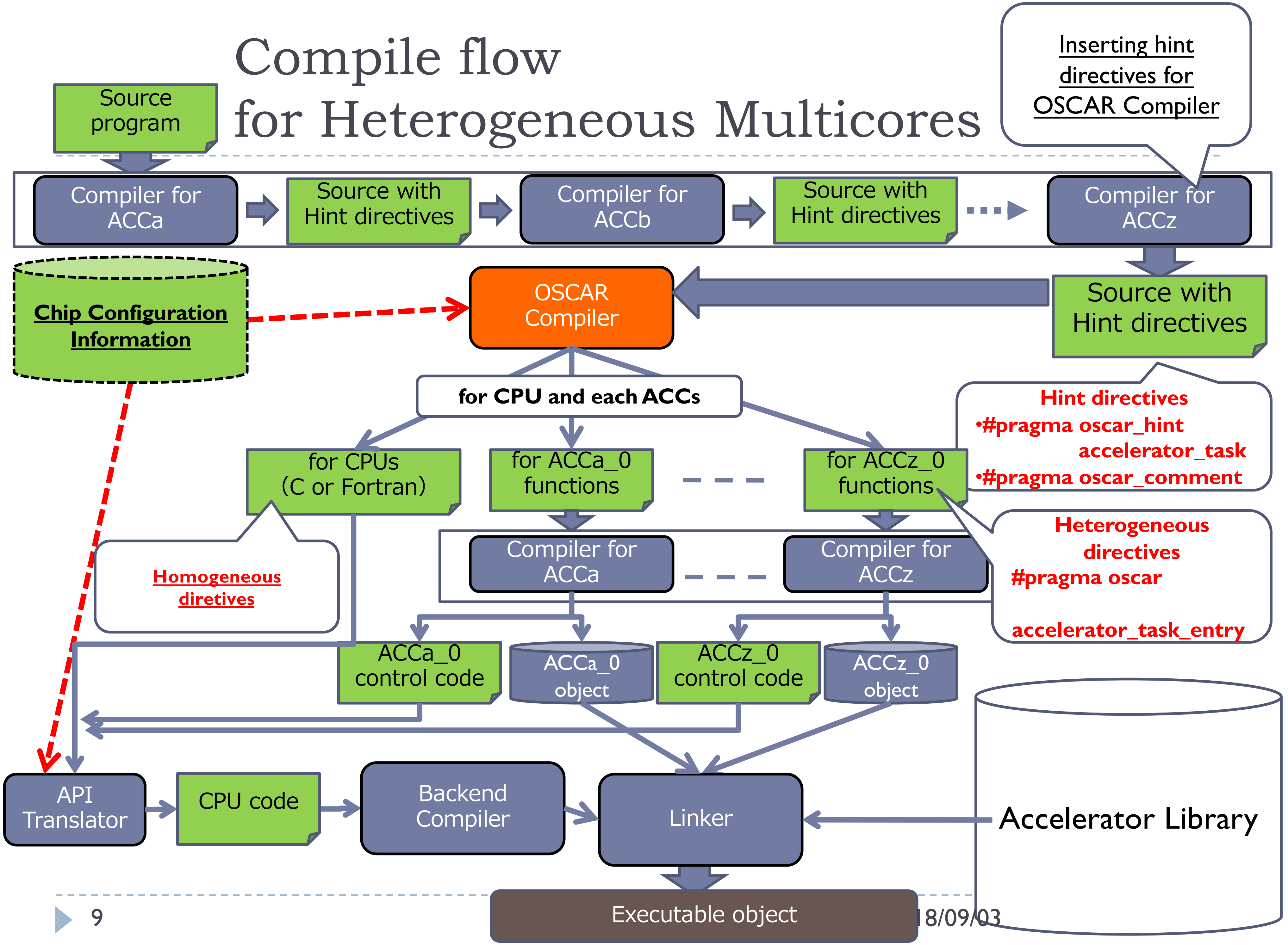


Avg. Power 5.41 [W]

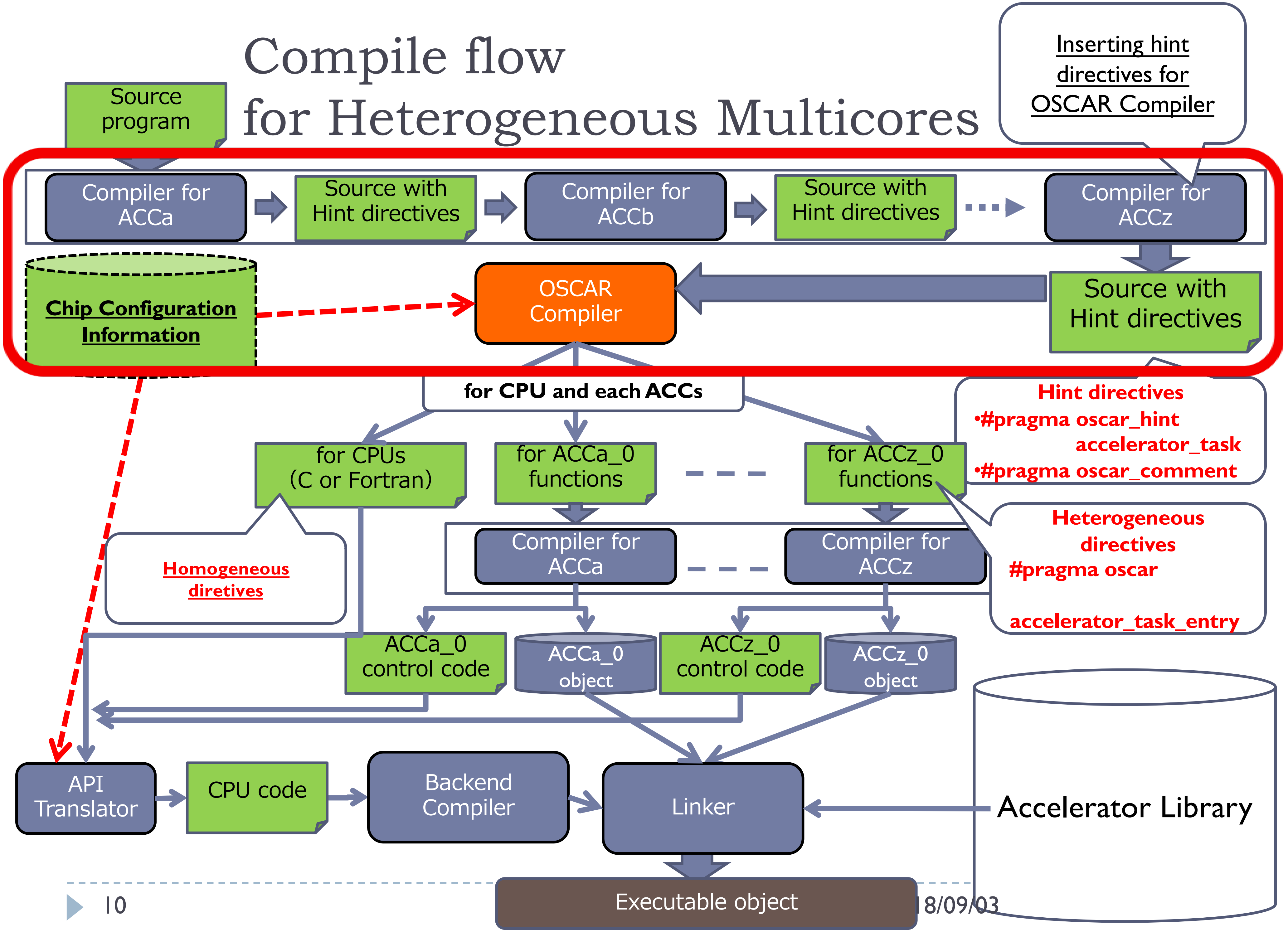
76.0% Power Reduction

Avg. Power 1.30 [W]

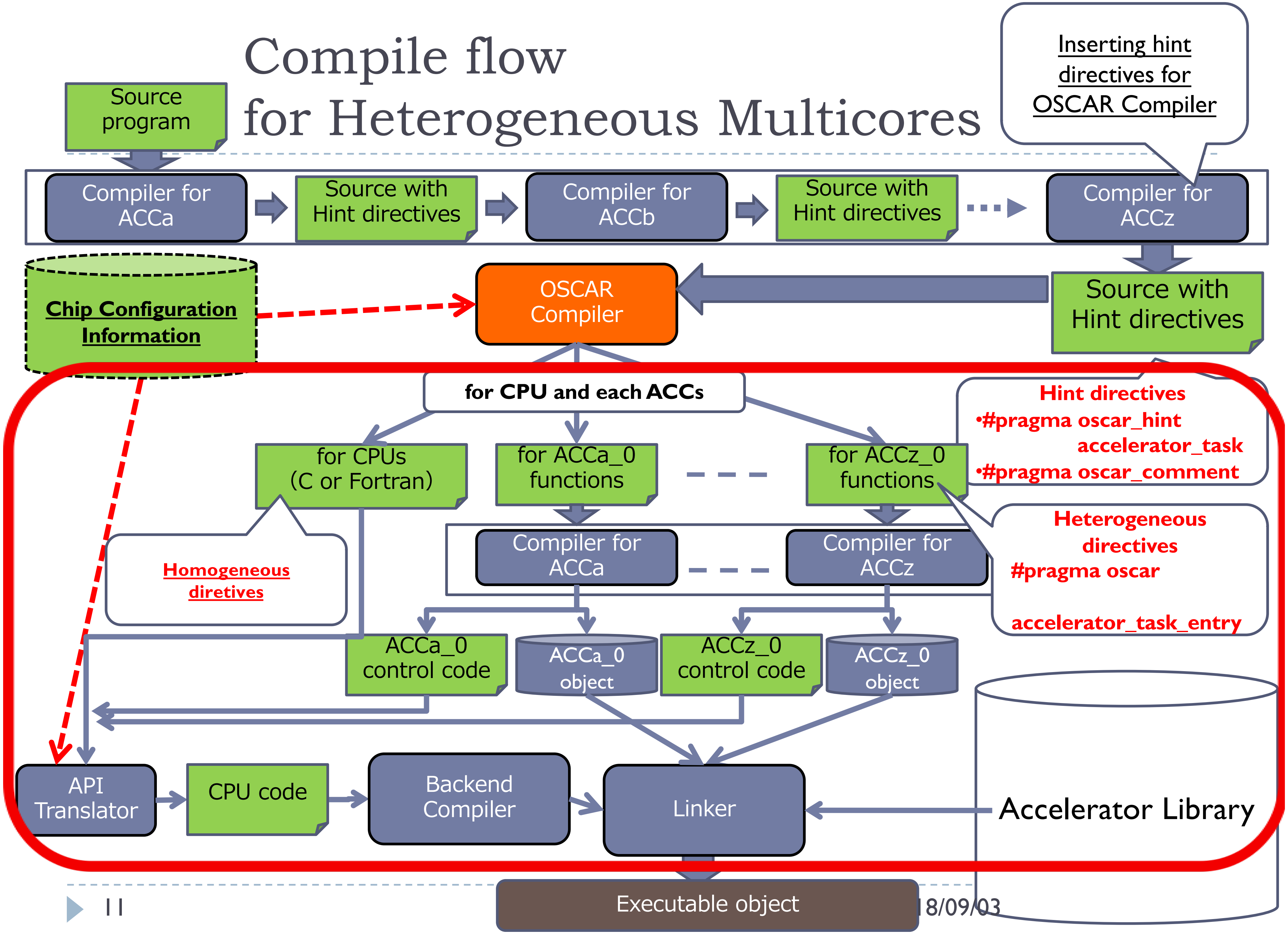
Compile flow for Heterogeneous Multicores



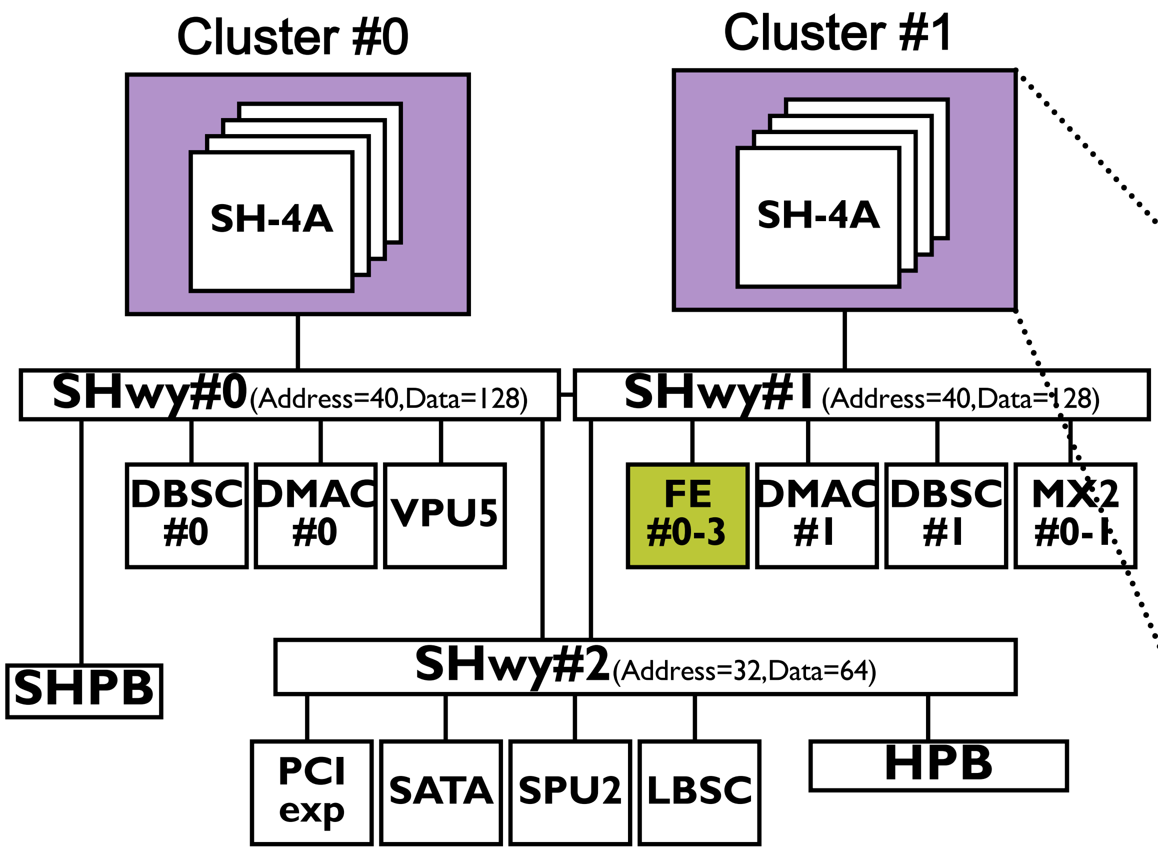
Compile flow for Heterogeneous Multicores

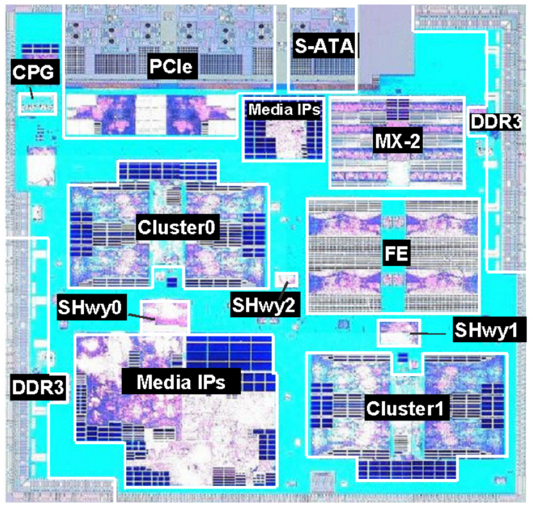


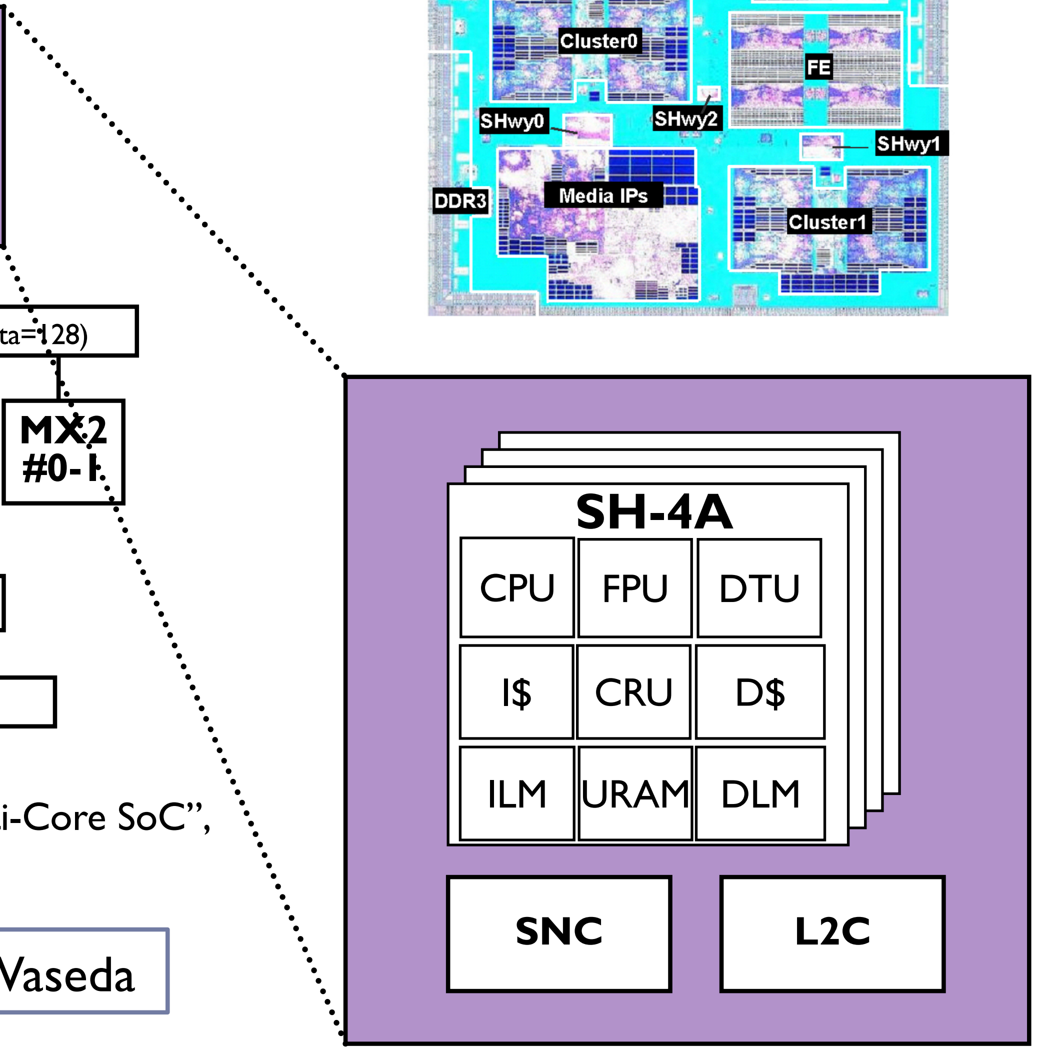
Compile flow for Heterogeneous Multicores



Low-power consumer electronics heterogeneous multicore "RP-X" (ISSCC 2009)



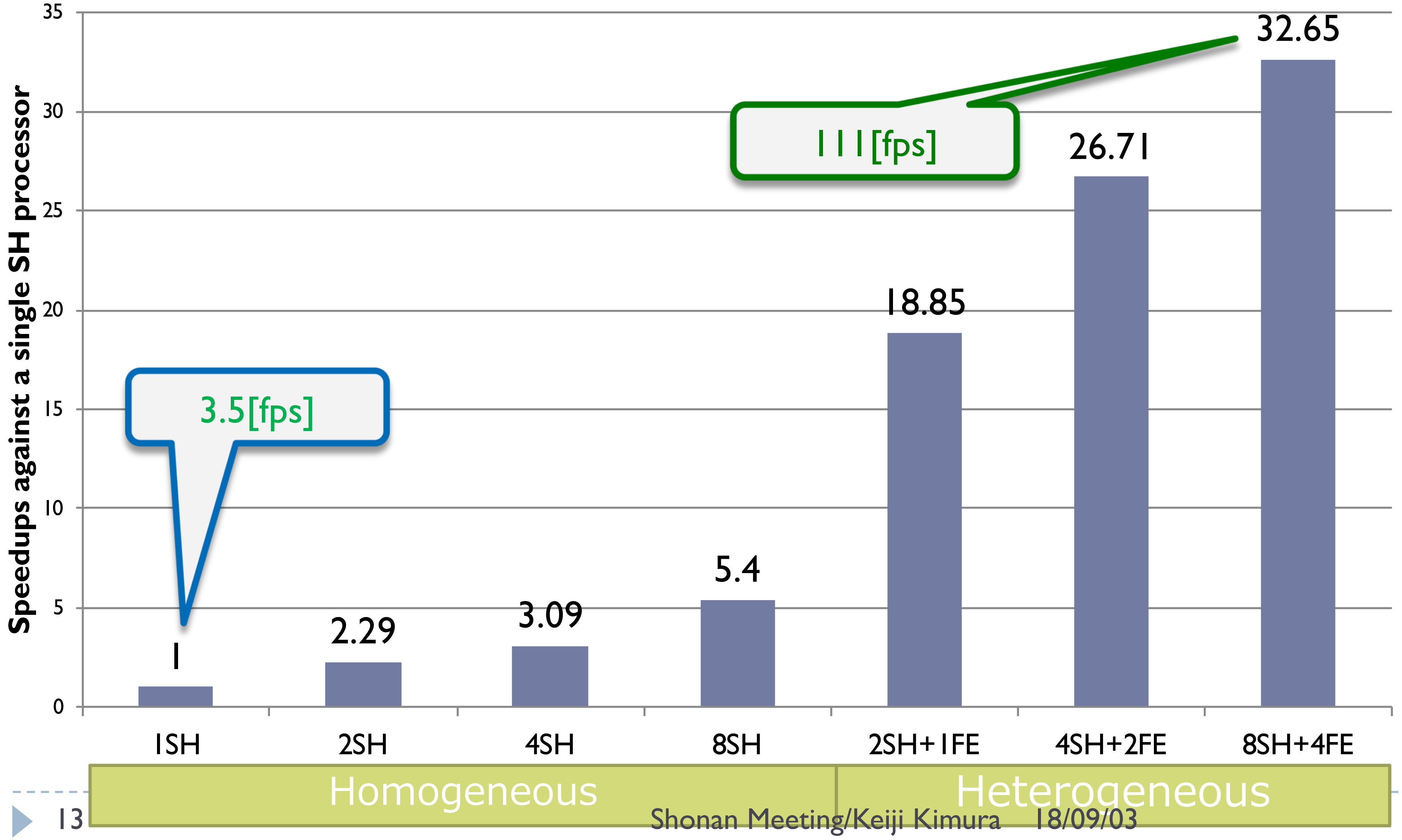




Y.Yuyama, et al., "A 45nm 37.3GOPS/W Heterogeneous Multi-Core SoC", ISSCC2010

developed by Renesas, Hitachi, Titech, and Waseda

Scalability of Optical Flow (Demo)



Low power optimization on Optical Flow (Demo)

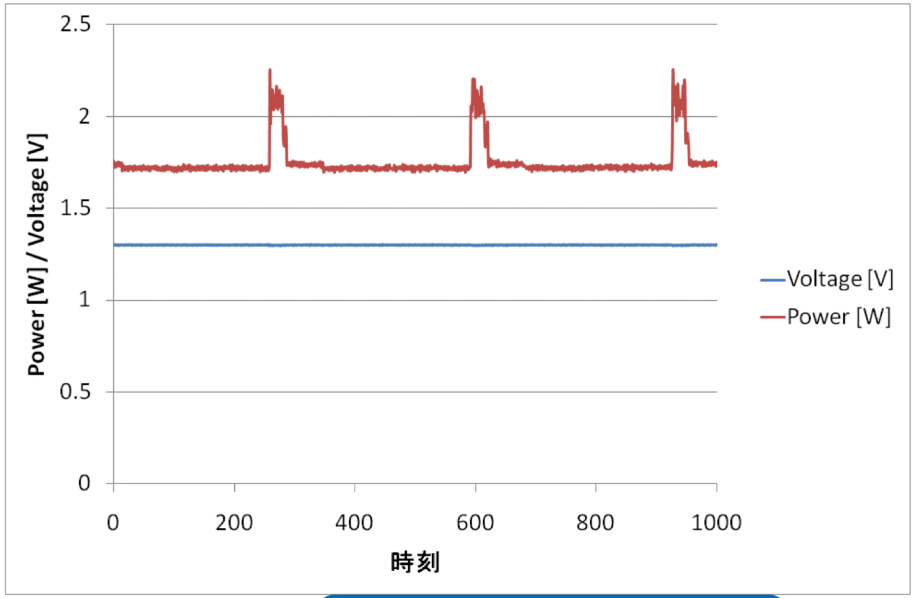
Without Power Reduction

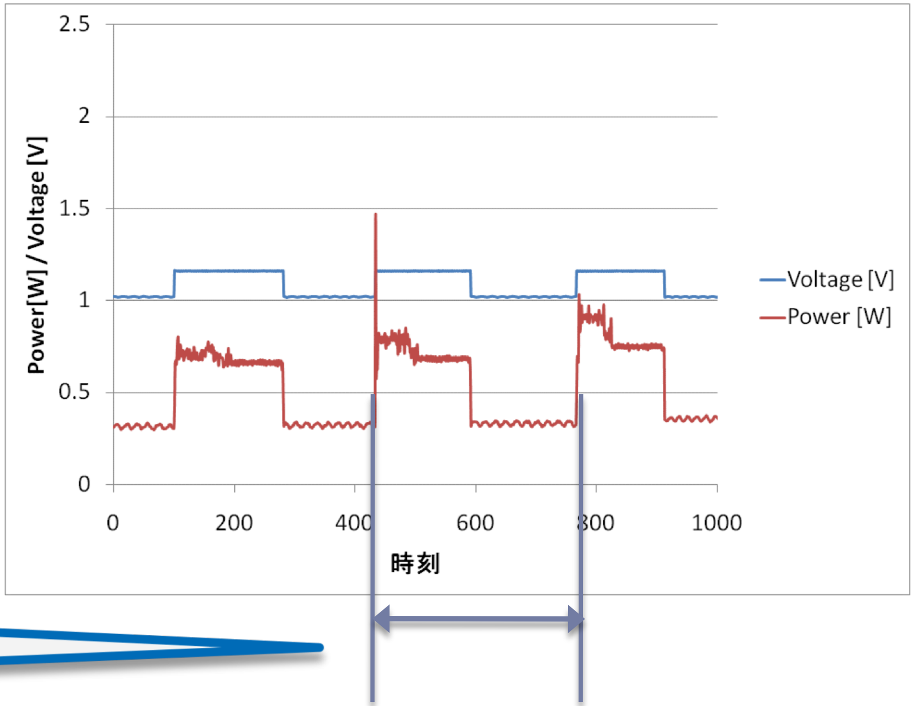
With Power Reduction by OSCAR Compiler

70% of power reduction

Average:1.76[W]

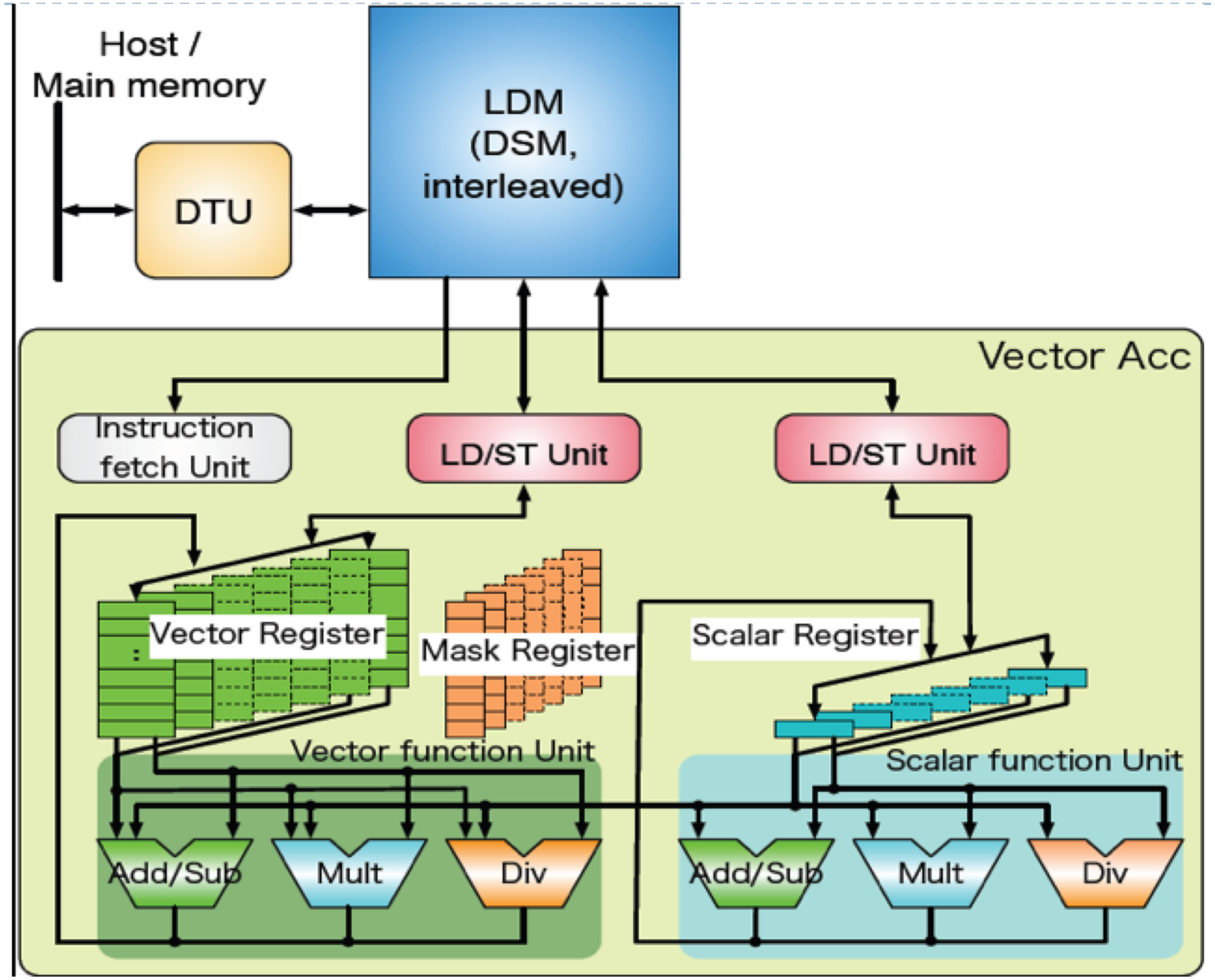
Average:0.54[W]



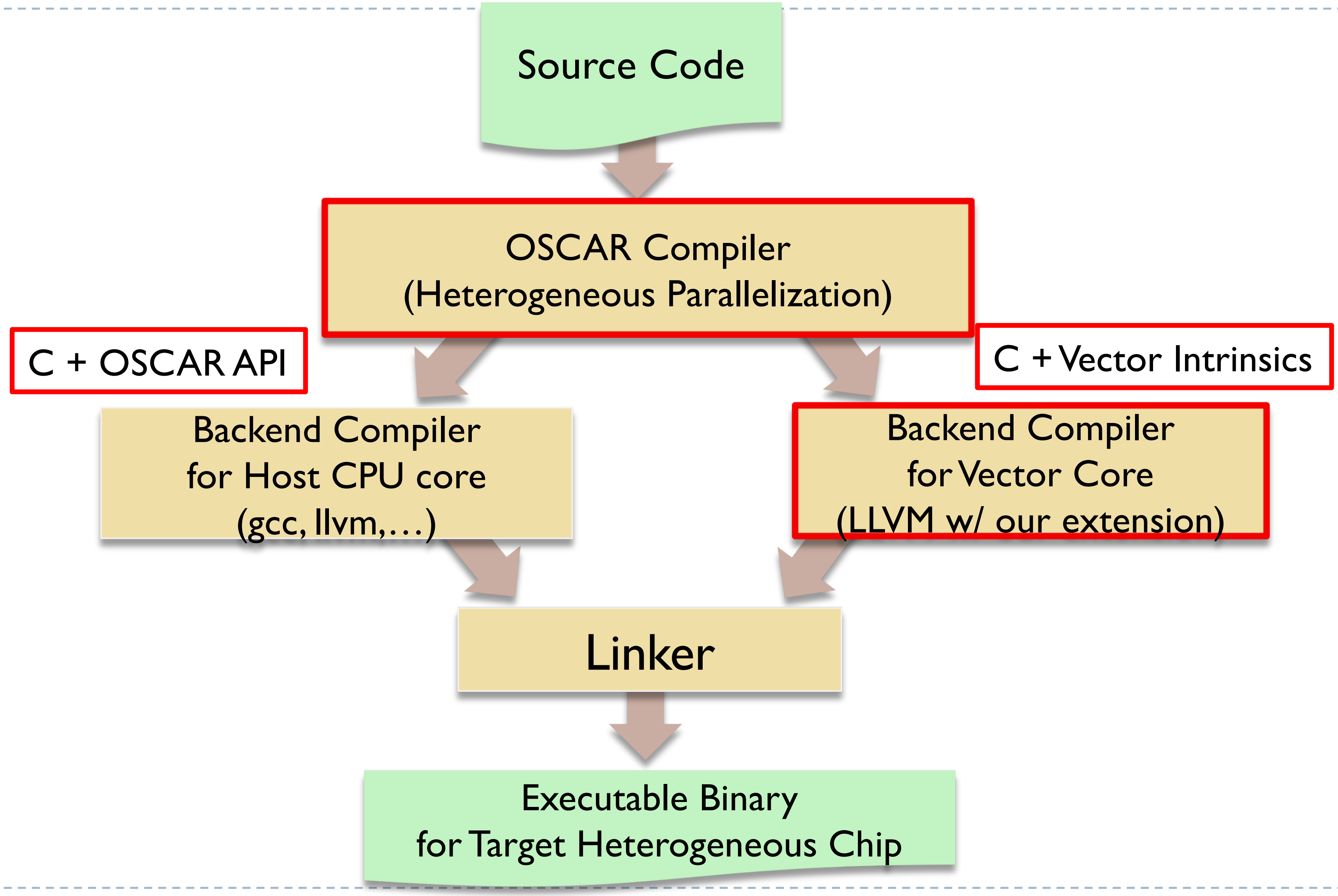


1cycle : 33[ms]
→30[fps]

OSCAR with Vector Accelerator



Compile flow for OSCAR + Vector



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