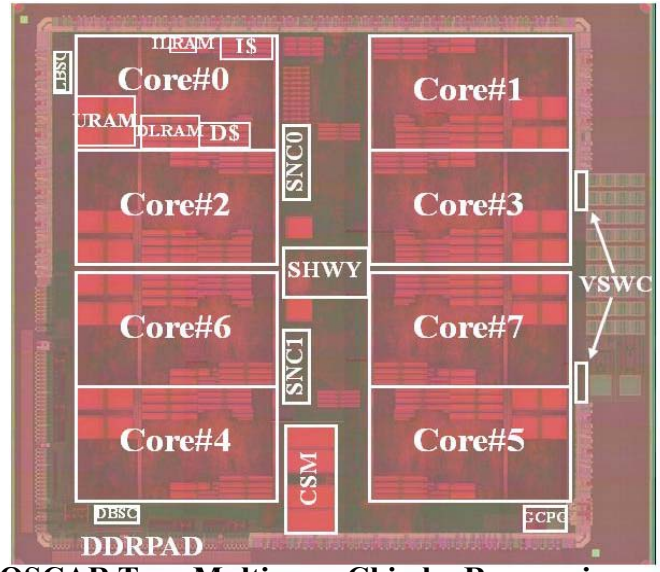
Compiler and API for Low Power High Performance Computing on Multicore and Manycore Processors

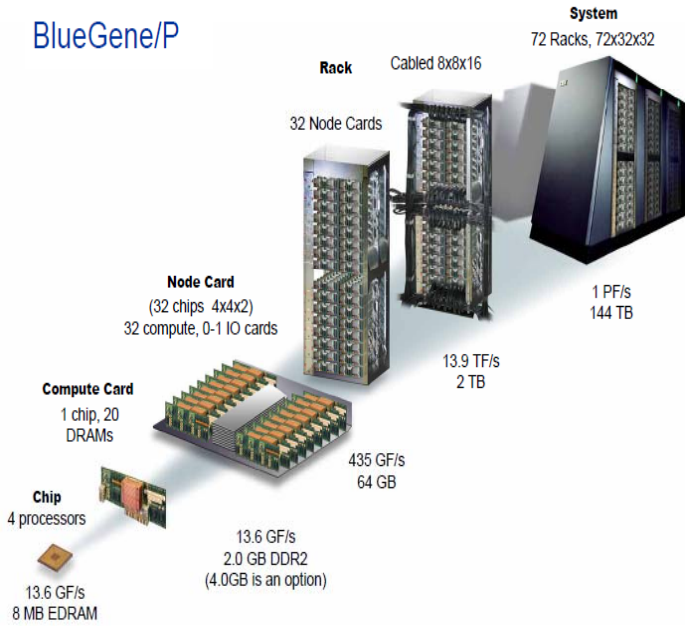
Hironori Kasahara
Professor, Dept. of Computer Science & Engineering
Director, Advanced Multicore Processor Research Institute
Waseda University, Tokyo, Japan
IEEE Computer Society Board of Governors
http://www.kasahara.cs.waseda.ac.jp

UIUC UPCRC Seminar, 16:00-17:00, Oct. 29 , 2009

Multi-core Everywhere



OSCAR Type Multi-core Chip by Renesas in METI/NEDO Multicore for Real-time Consumer Electronics Project (Leader: Prof.Kasahara)



Multi-core from embedded to supercomputers

- **Consumer Electronics (Embedded)**
  - Mobile Phone, Game, Digital TV, Car Navigation, DVD, Camera,
  - IBM/ Sony/ Toshiba Cell, Fuijtsu FR1000,
  - NEC/ARMMPCore&MP211, Panasonic Uniphier,
  - Renesas SH multi-core(4 core RP1, 8 core RP2)
  - Tilera Tile64, SPI Storm-1(16 VLIW cores)
- **PCs, Servers**
  - Intel Quad Xeon, Core 2 Quad, Montvale, Nehalem(8core), 80 core, Larrabee(32core)
  - AMD Quad Core Opteron, Phenom
- **WSs, Deskside & Highend Servers**
  - IBM Power4,5,5+,6 Sun Niagara(SparcT1,T2), Rock
- **Supercomputers**
  - Earth Simulator:40TFLOPS, 2002, 5120 vector proc.
  - IBM Blue Gene/L: 360TFLOPS, 2005, Low power CMP d 128K processor chips, BG/Q :20PFLOPS.2011, BlueWaters: Effective 1PFLOPS, Julty2011,NCSA UIUC

High quality application software, Productivity, Cost performance, Low power consumption are important

Ex, Mobile phones, Games

Compiler cooperated multi-core processors are promising to realize the above futures

METI/NEDO National Project

Multi-core for Real-time Consumer Electronics

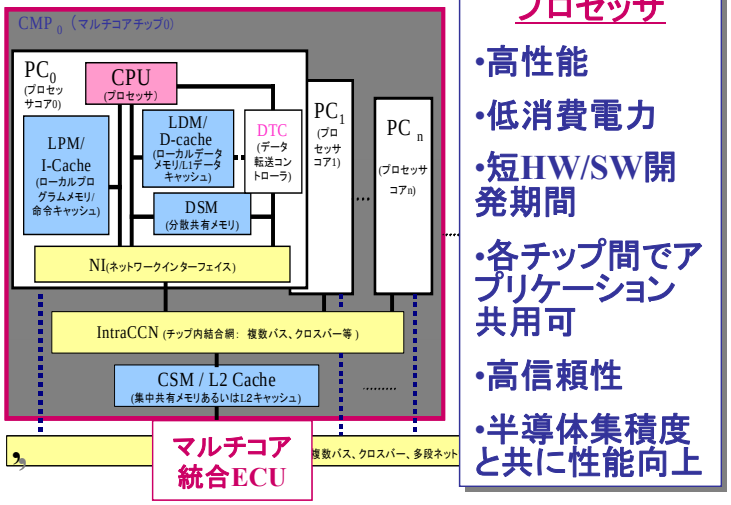
(2005.7～2008.3)**

<Goal> R&D of compiler cooperative multi-core processor technology for consumer electronics like Mobile phones, Games, DVD, Digital TV, Car navigation systems.

<Period> From July 2005 to March 2008

<Features>
- Good cost performance
- Short hardware and software development periods
- Low power consumption
- Scalable performance improvement with the advancement of semiconductor
- Use of the same parallelizing compiler for multi-cores from different vendors using newly developed API

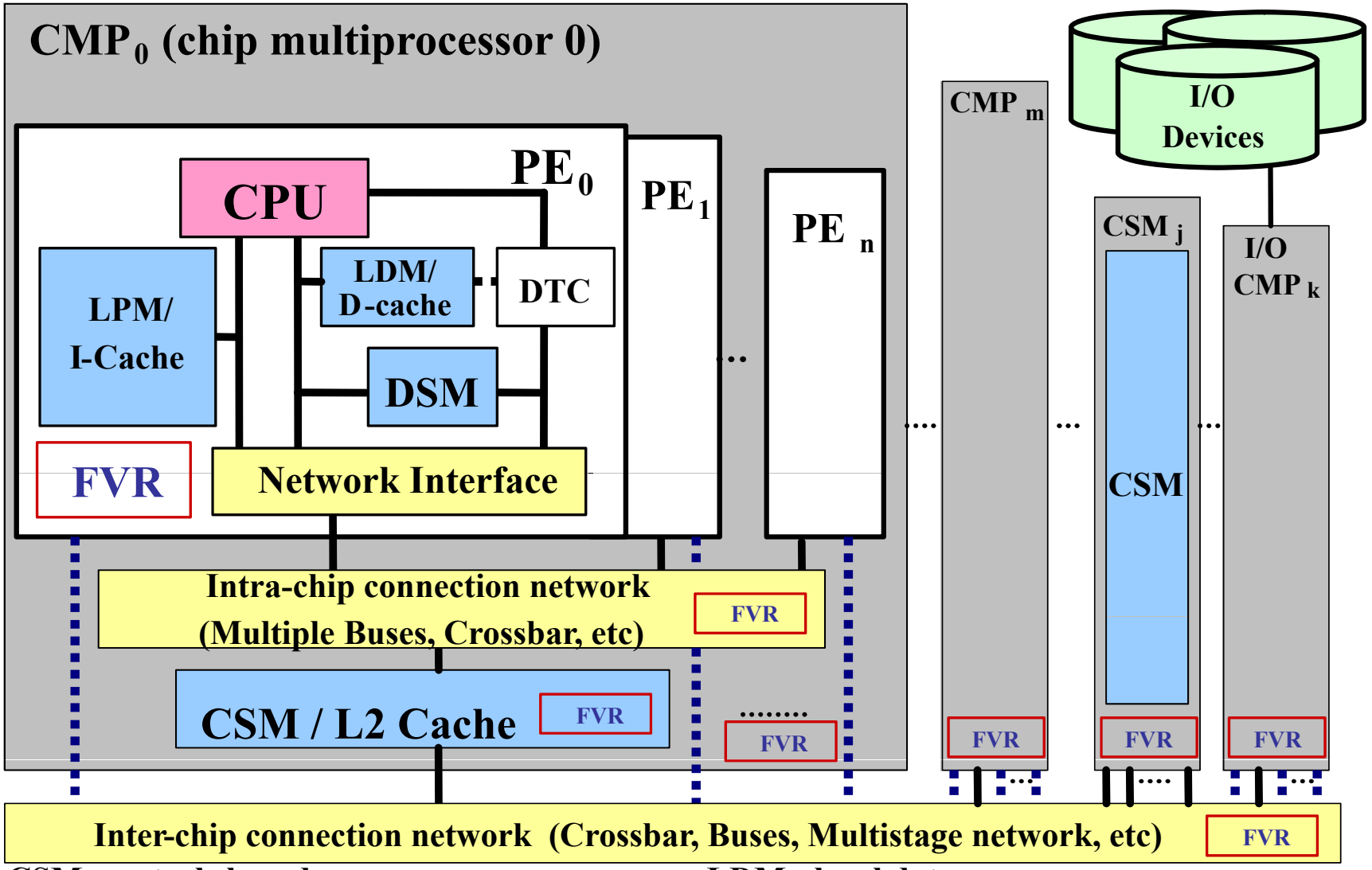
API : Application Programming Interface



開発マルチコアチップは情報家電へ

**Hitachi, Renesas, Fujitsu, Toshiba, Panasonic, NEC

OSCAR Multi-Core Architecture



CSM: central shared mem.

DSM: distributed shared mem.

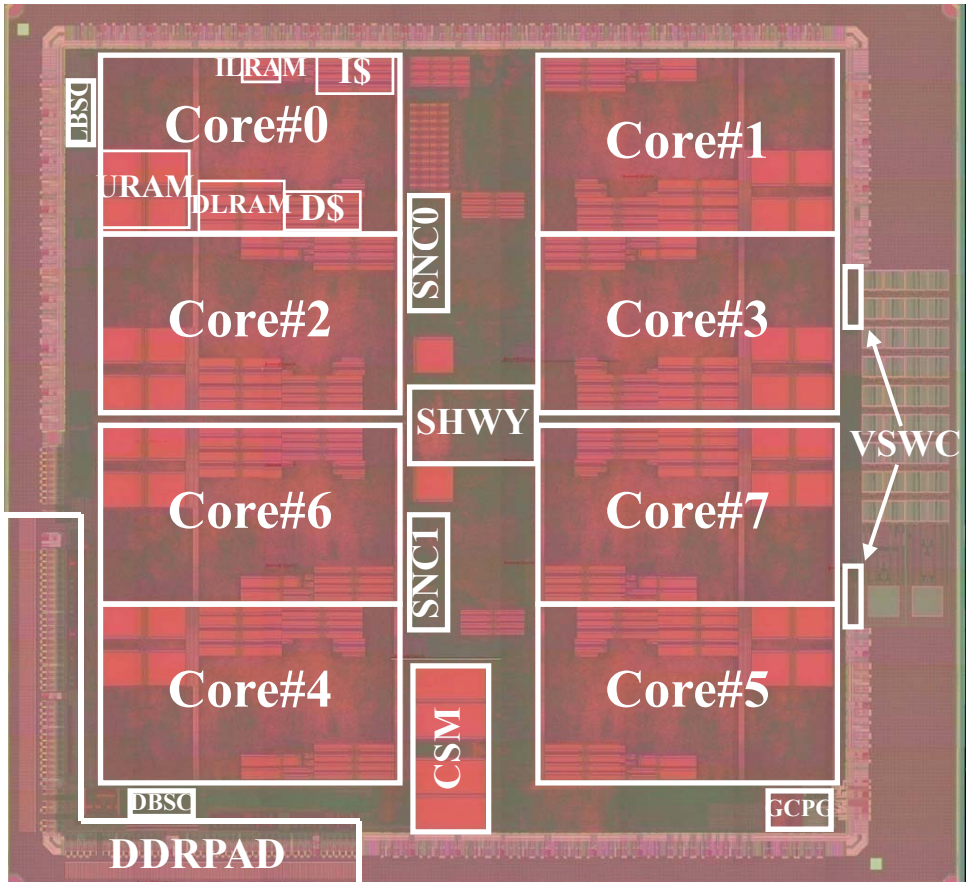
DTC: Data Transfer Controller

LDM : local data mem.

LPM : local program mem.

FVR: frequency / voltage control register

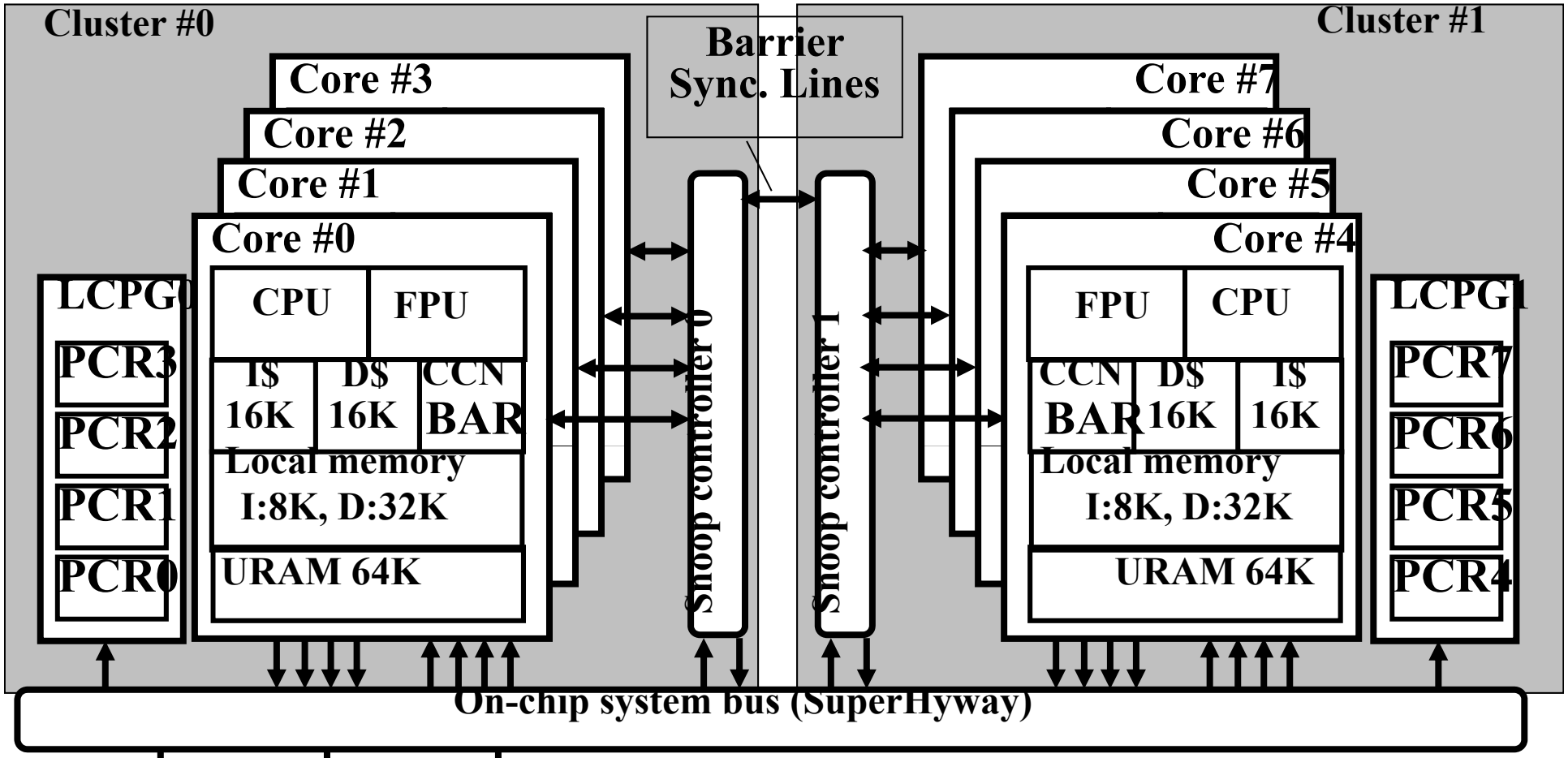
Renesas-Hitachi-Waseda 8 core RP2 Chip Photo and Specifications



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)

IEEE ISSCC08: Paper No. 4.5, M.ITO, … and H. Kasahara, "An 8640 MIPS SoC with Independent Power-off Control of 8 CPUs and 8 RAMs by an Automatic Parallelizing Compiler"

8 Core RP2 Chip Block Diagram



LCPG: Local clock pulse generator
PCR: Power Control Register
CCN/BAR:Cache controller/Barrier Register
URAM: User RAM

Demo of NEDO Multicore for Real Time Consumer Electronics at the Council of Science and Engineering Policy on April 10, 2008

第74回総合科学技術会議【平成20年4月10日】


第74回総合科学技術会議の様子(1)


第74回総合科学技術会議の様子(2)


第74回総合科学技術会議の様子(3)


第74回総合科学技術会議の様子(4)

CSTP Members

Prime Minister:
Mr. Y. FUKUDA

Minister of State for Science, Technology and Innovation Policy:
Mr. F. KISHIDA

Chief Cabinet Secretary:
Mr. N. MACHIMURA

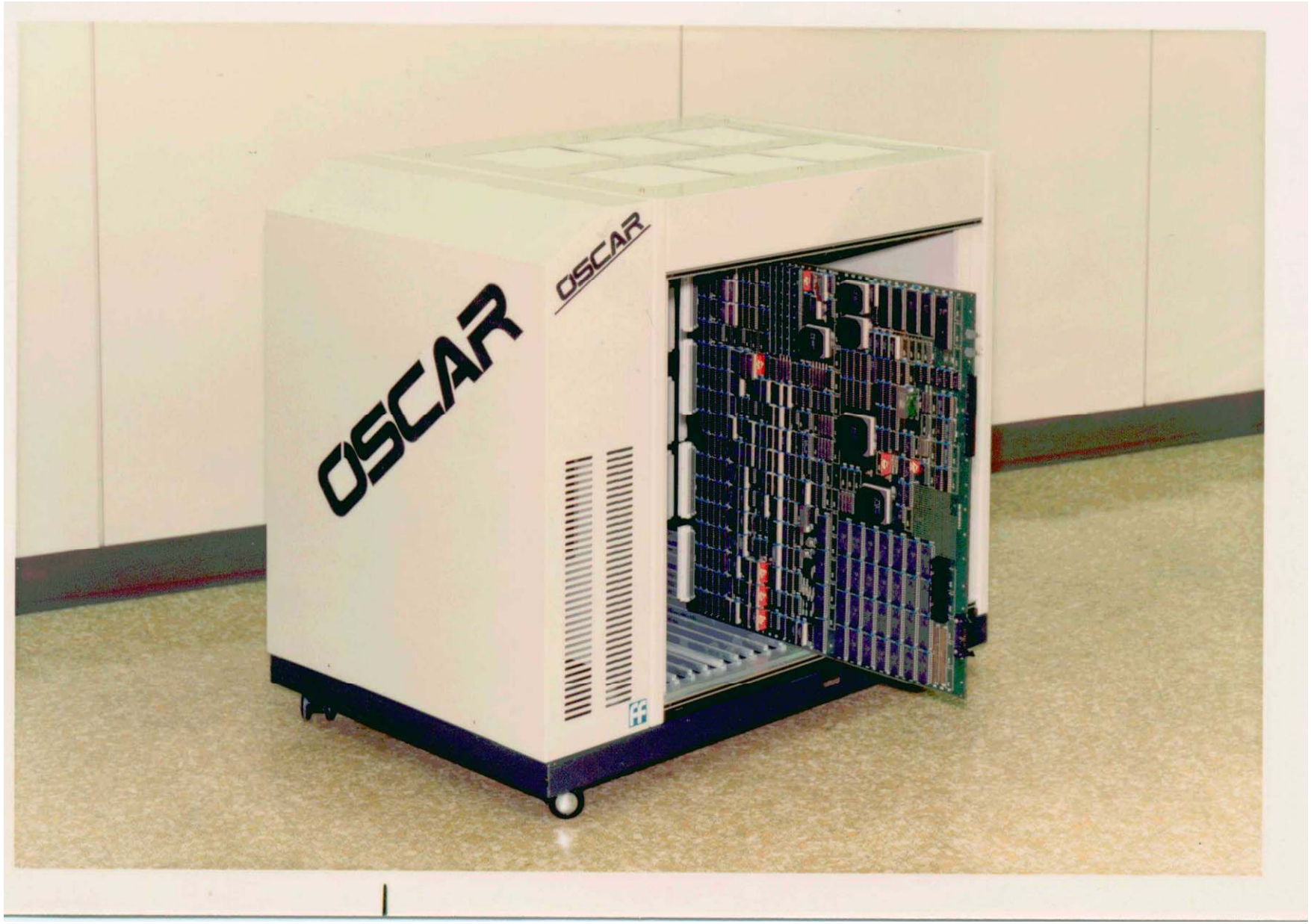
Minister of Internal Affairs and Communications :
Mr. H. MASUDA

Minister of Finance :
Mr. F. NUKAGA

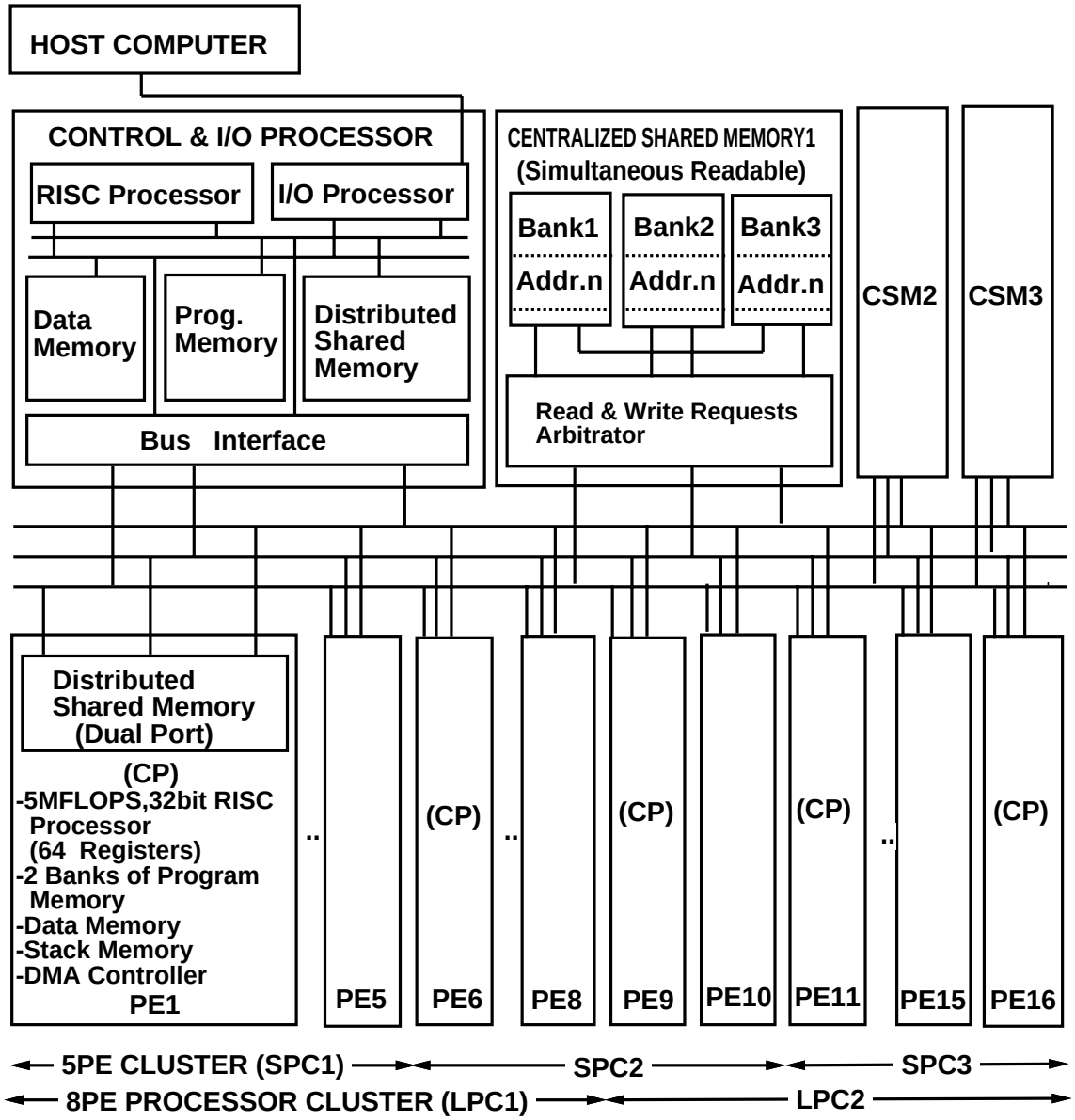
Minister of Education, Culture, Sports, Science and Technology:
Mr. K. TOKAI

Minister of Economy,Trade and Industry:
Mr. A. AMARI

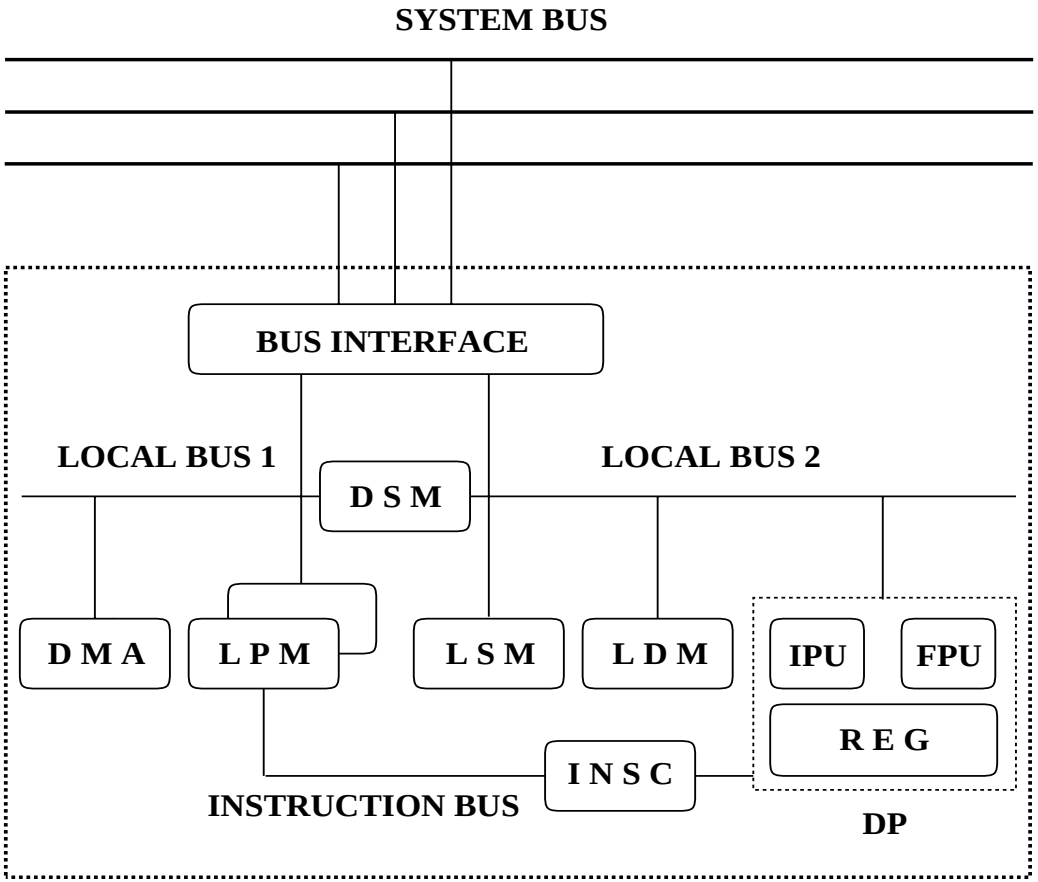
1987 OSCAR(Optimally Scheduled Advanced Multiprocessor)



OSCAR(Optimally Scheduled Advanced Multiprocessor)

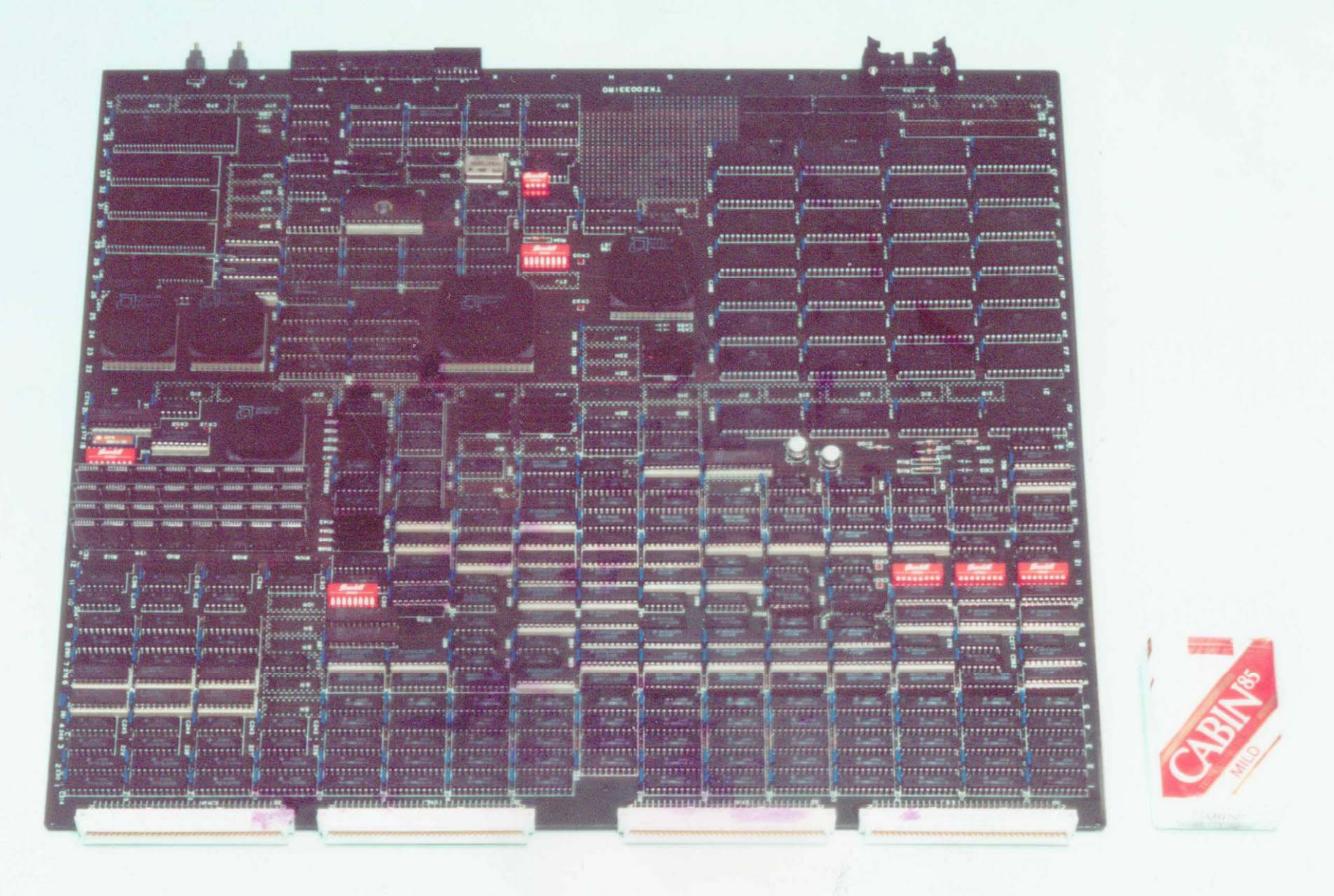


OSCAR PE (Processor Element)

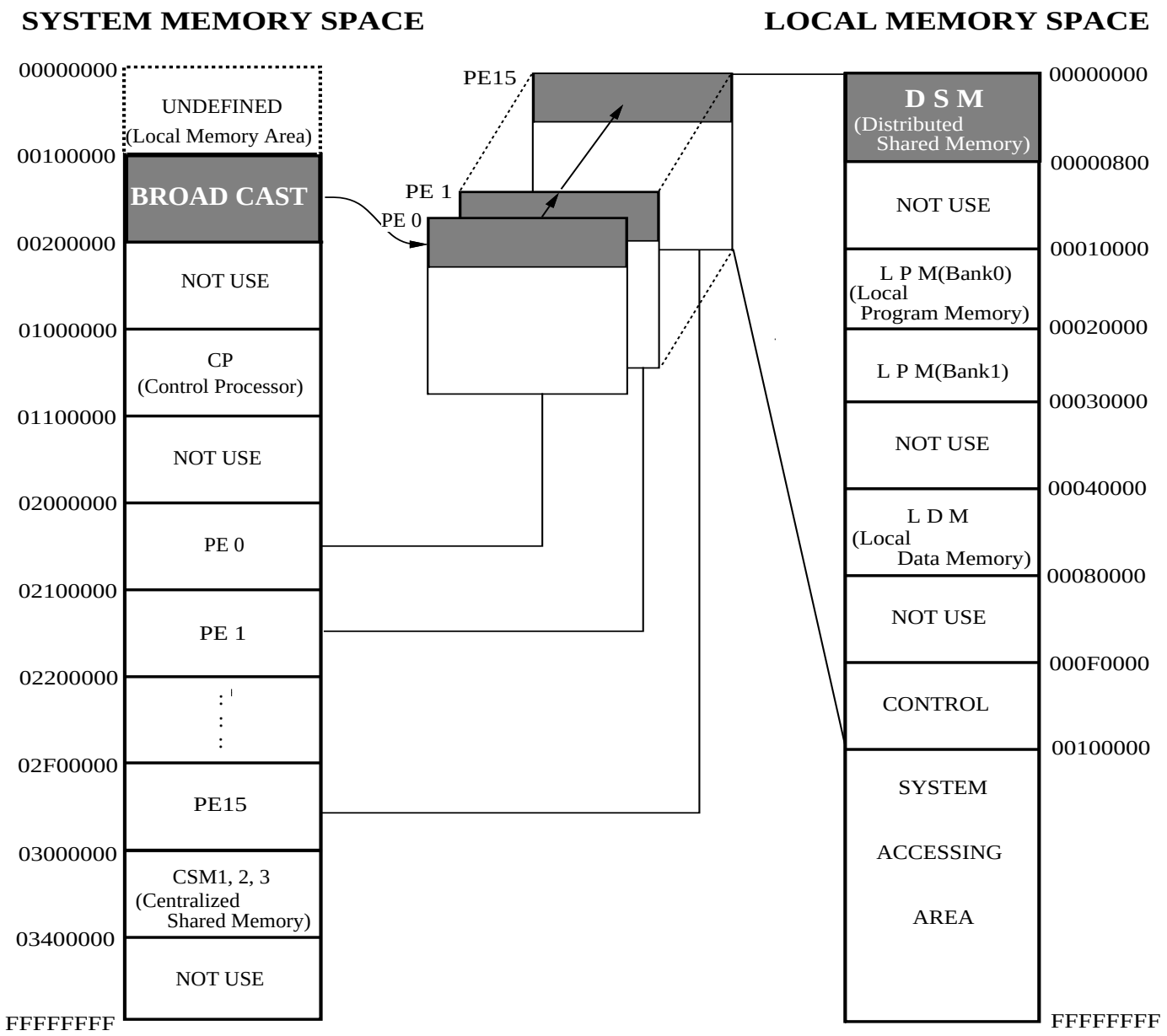


D M A : DMA CONTROLLER
L P M : LOCAL PROGRAM MEMORY (128KW * 2BANK)
I N S C : INSTRUCTION CONTROL UNIT
D S M : DISTRIBUTED SHARED MEMORY (2KW)
L S M : LOCAL STACK MEMORY (4KW)
L D M : LOCAL DATA MEMORY (256KW)
D P : DATA PATH
I P U : INTEGER PROCESSING UNIT
F P U : FLOATING PROCESSING UNIT
R E G : REGISTER FILE (64 REGISTERS)

1987 OSCAR PE Board



OSCAR Memory Space (Global and Local Address Space)



OSCAR Parallelizing Compiler

To improve effective performance, cost-performance and software productivity and reduce power

Multigrain Parallelization

coarse-grain parallelism among loops and subroutines, near fine grain parallelism among statements in addition to loop parallelism

Data Localization

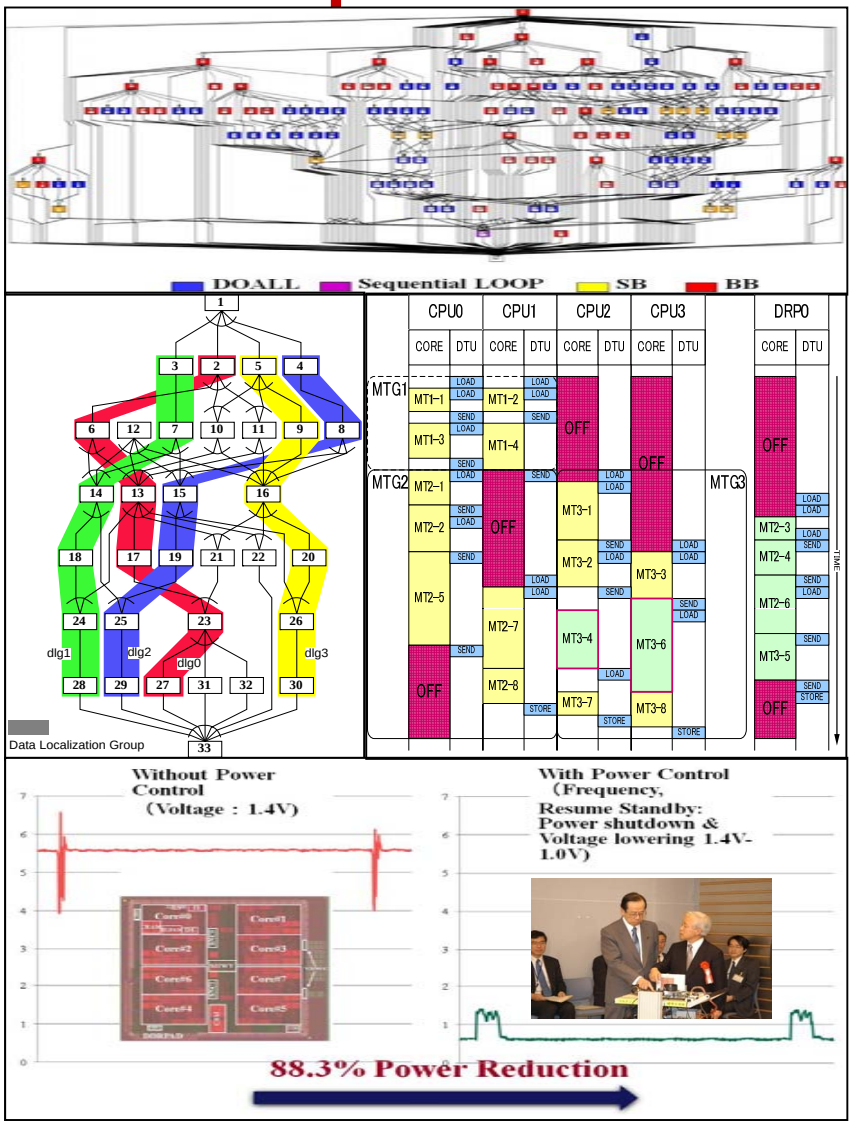
Automatic data management for distributed shared memory, cache and local memory

Data Transfer Overlapping

Data transfer overlapping using Data Transfer Controllers (DMAs)

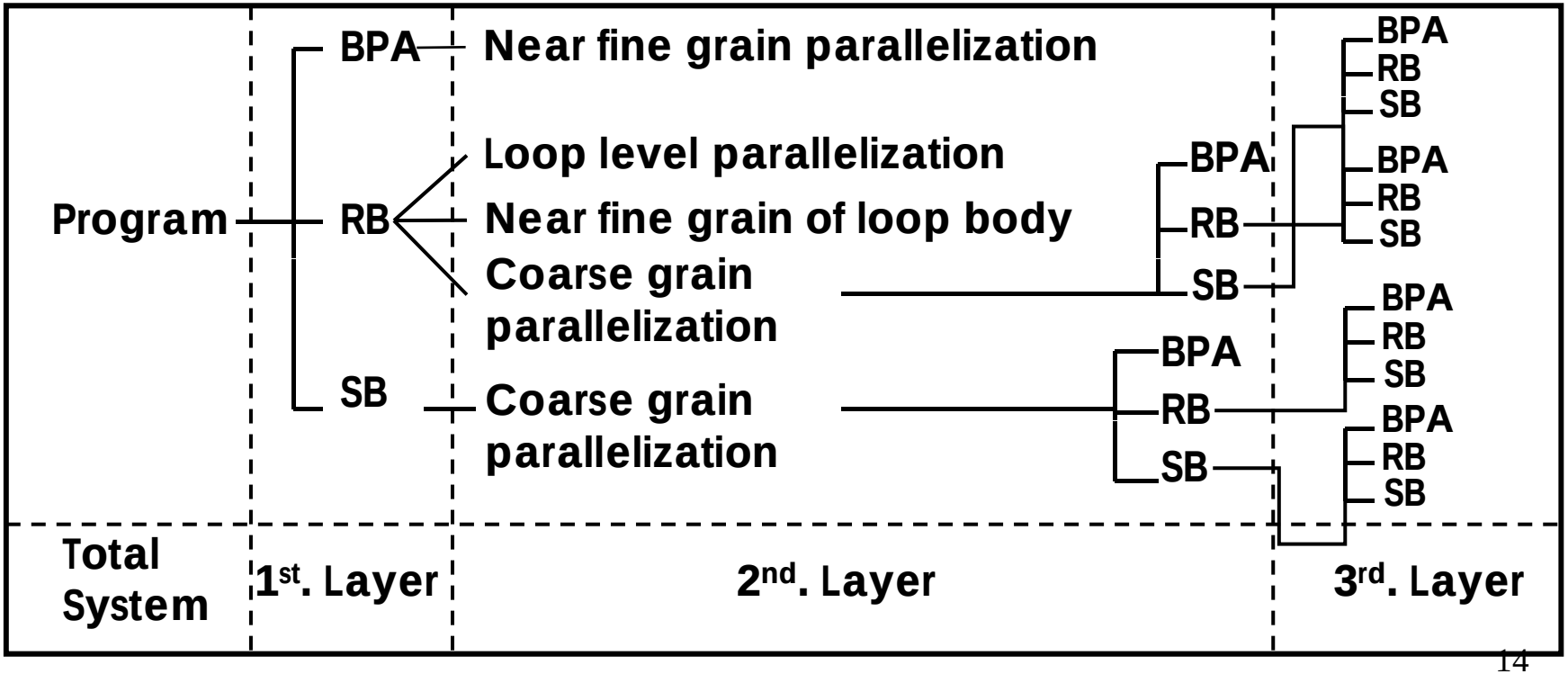
Power Reduction

Reduction of consumed power by compiler control DVFS and Power gating with hardware supports.

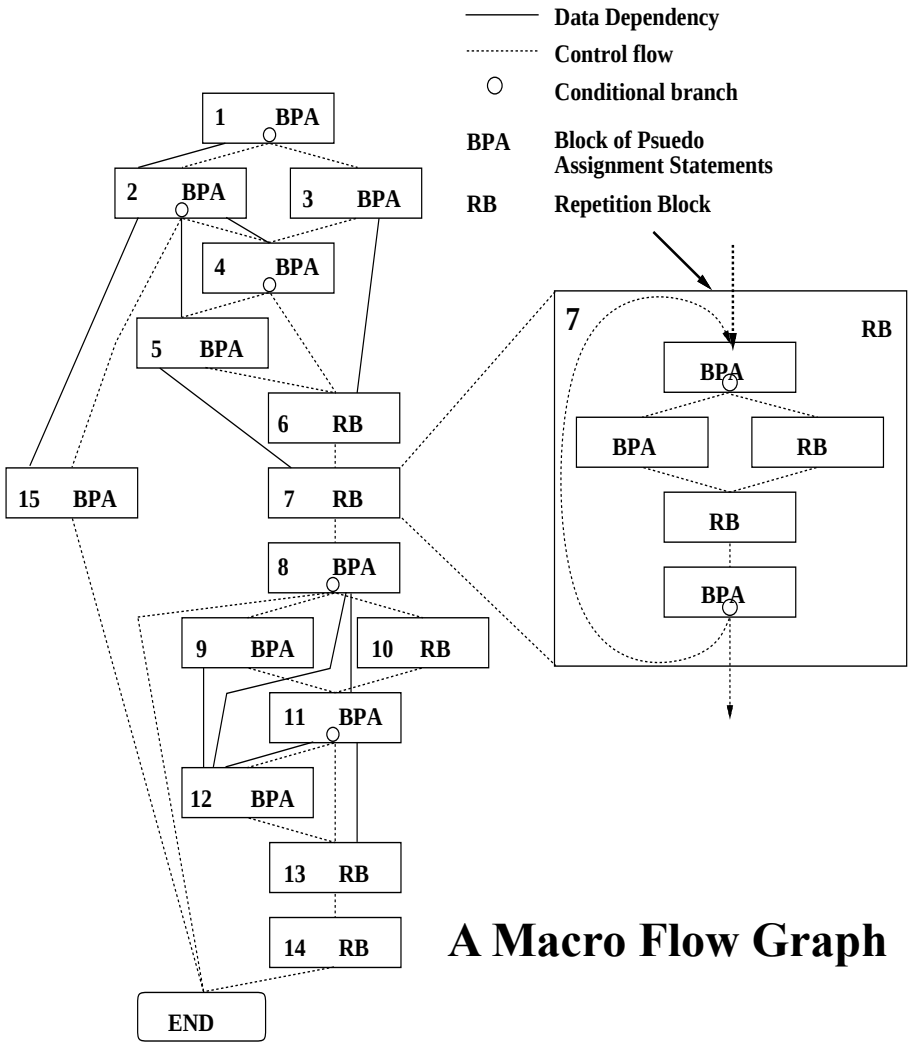


Generation of coarse grain tasks

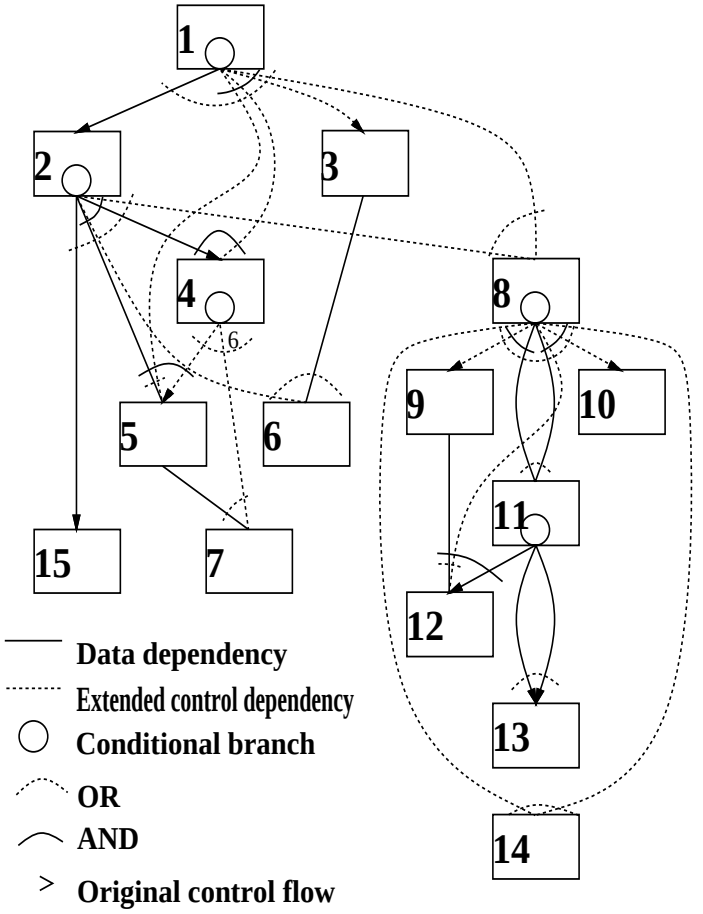
- Macro-tasks (MTs)
  - Block of Pseudo Assignments (BPA): Basic Block (BB)
  - Repetition Block (RB) : natural loop
  - Subroutine Block (SB): subroutine



Earliest Executable Condition Analysis for coarse grain tasks (Macro-tasks)



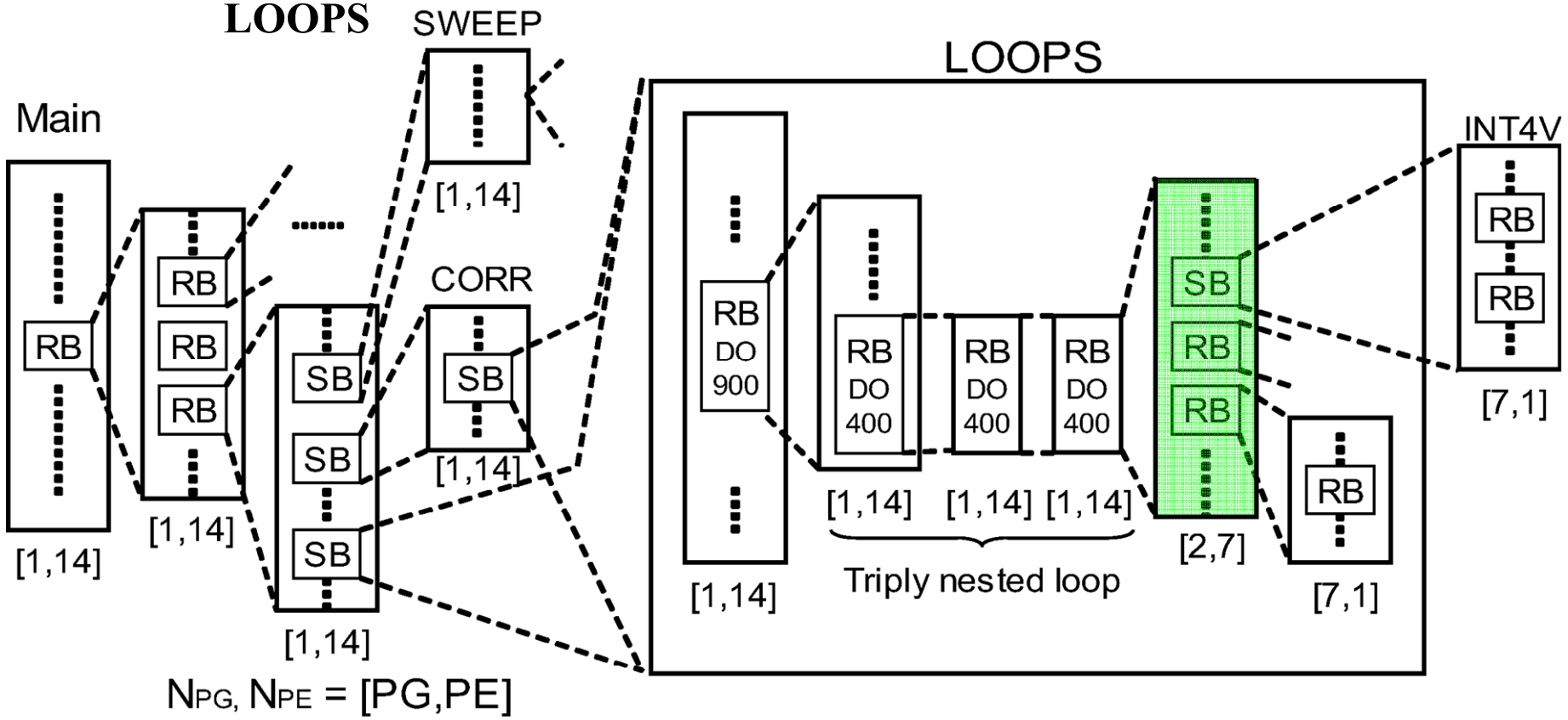
A Macro Flow Graph



A Macro Task Graph

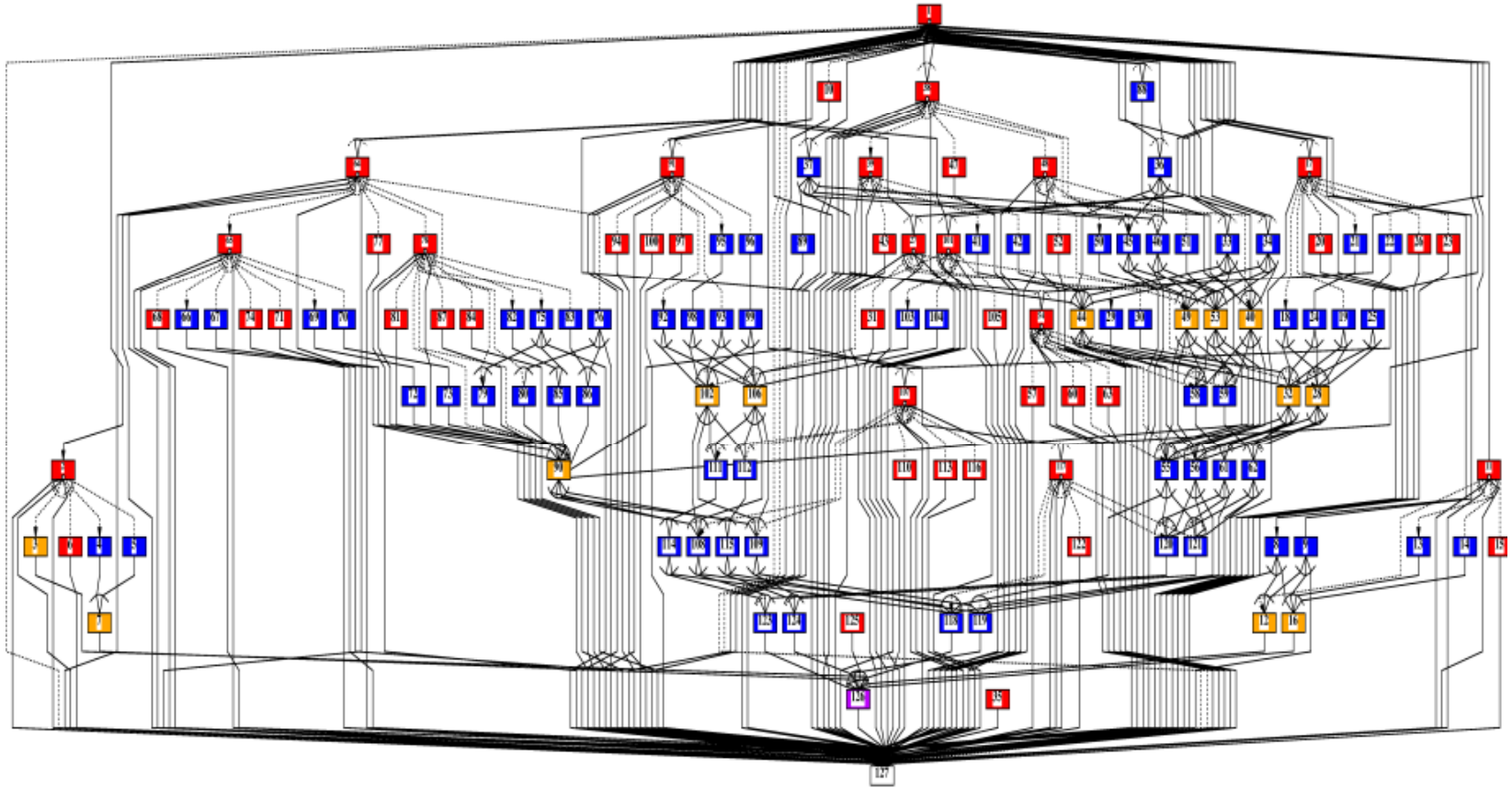
Automatic processor assignment in su2cor

- **Using 14 processors**
  - **Coarse grain parallelization within DO400 of subroutine LOOPS**



MTG of Su2cor-LOOPS-DO400

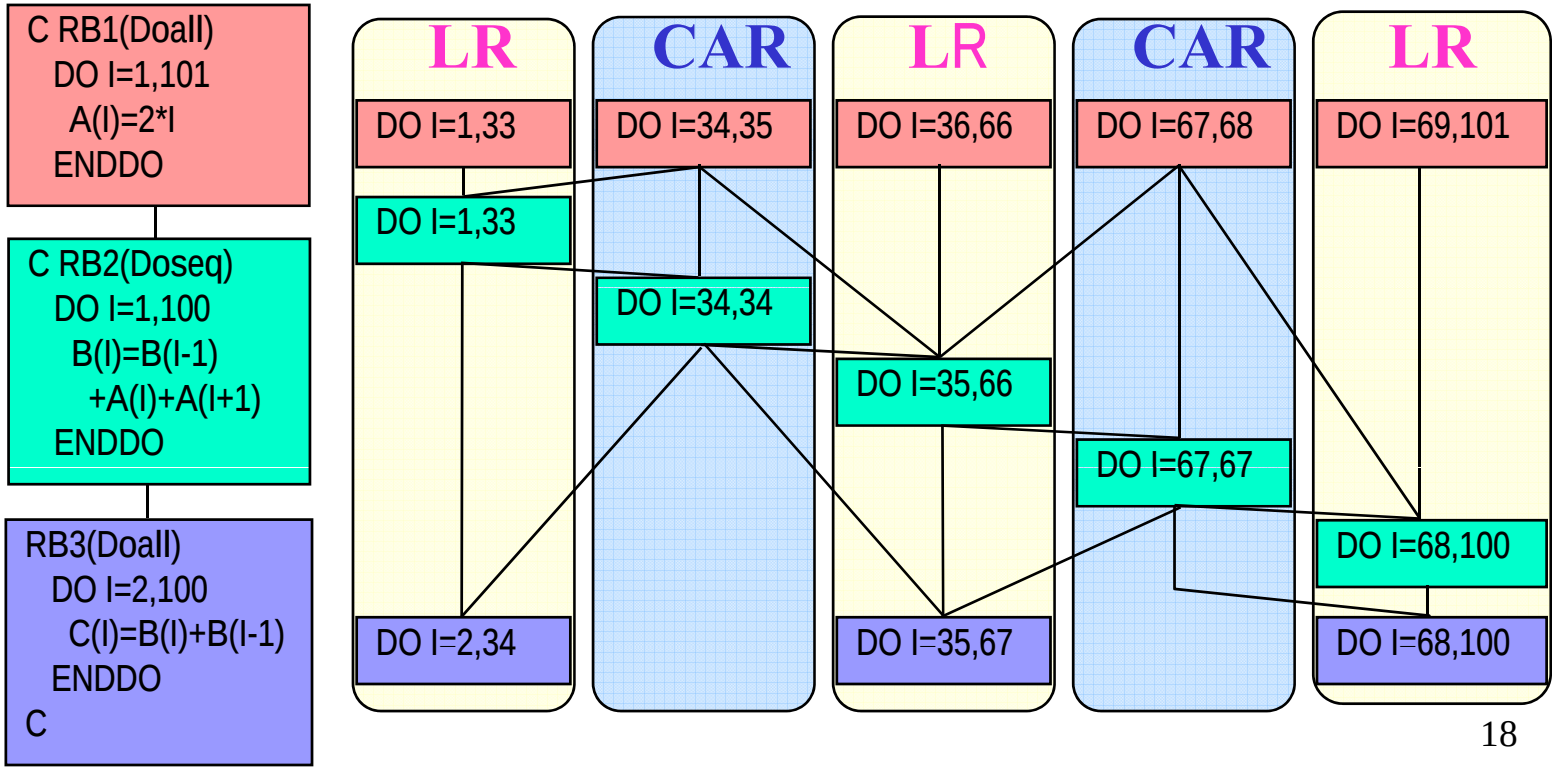
- Coarse grain parallelism PARA_ALD = 4.3



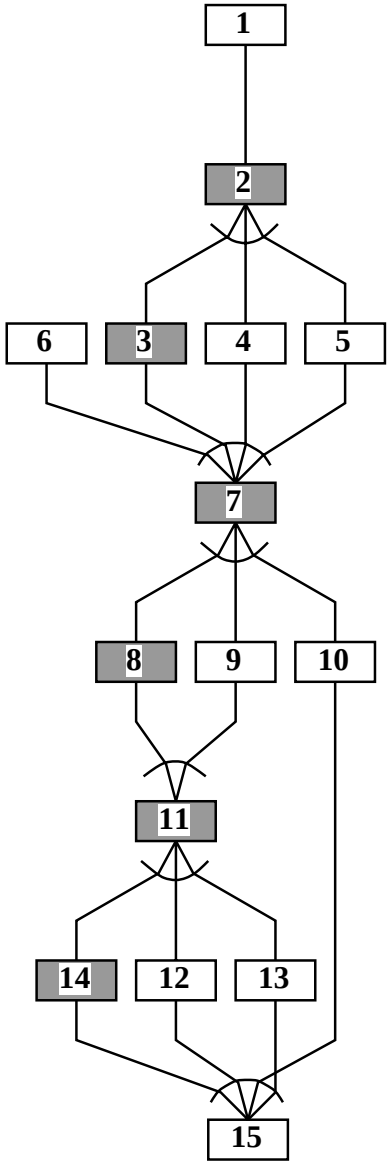
DOALL Sequential LOOP SB BB

Data-Localization
Loop Aligned Decomposition

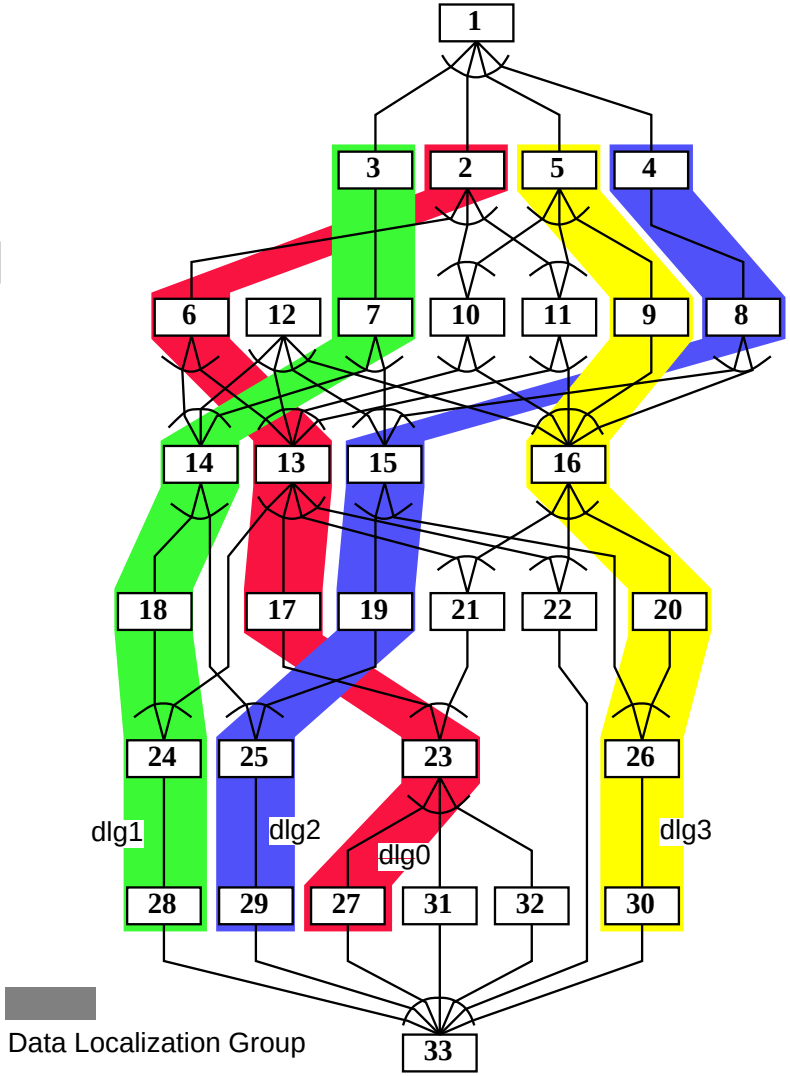
- **Decompose multiple loop (Doall and Seq) into CARs and LRs considering inter-loop data dependence.**
  - **Most data in LR can be passed through LM.**
  - **LR: Localizable Region, CAR: Commonly Accessed Region**



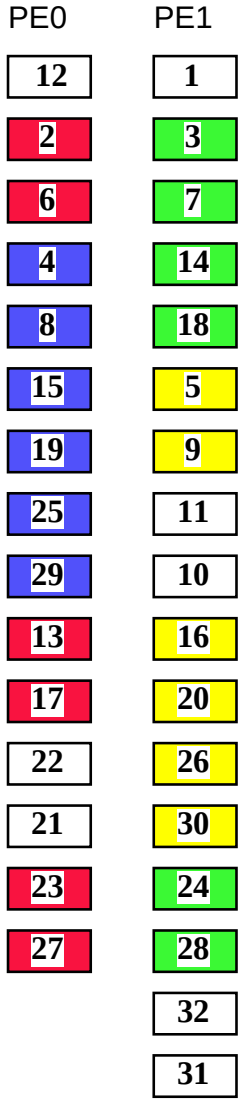
Data Localization



MTG



MTG after Division



A schedule for two processors

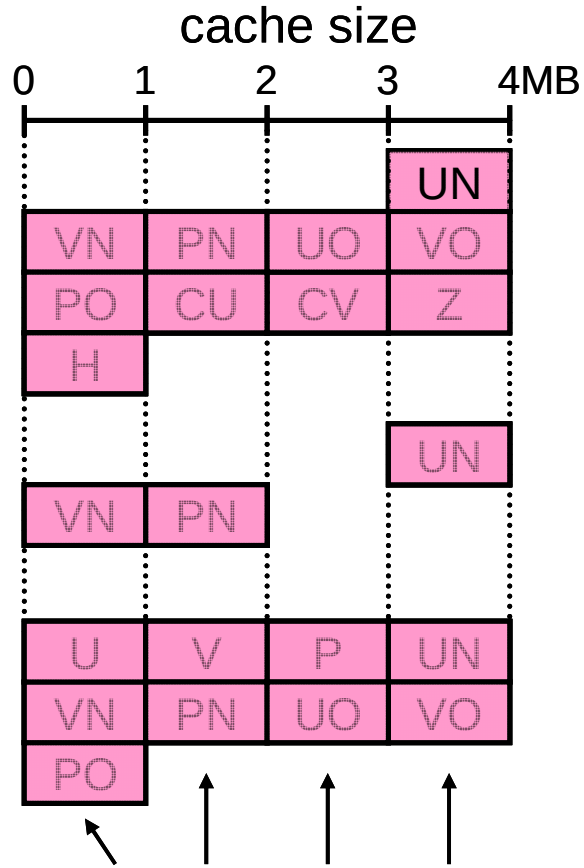
An Example of Data Localization for Spec95 Swim

```
      DO 200 J=1,N
      DO 200 I=1,M
         UNEW(I+1,J) = UOLD(I+1,J)+
     1    TDTS8*(Z(I+1,J+1)+Z(I+1,J))*(CV(I+1,J+1)+CV(I,J+1)+CV(I,J)
     2      +CV(I+1,J))-TDTSDX*(H(I+1,J)-H(I,J))
         VNEW(I,J+1) = VOLD(I,J+1)-TDTS8*(Z(I+1,J+1)+Z(I,J+1))
     1      *(CU(I+1,J+1)+CU(I,J+1)+CU(I,J)+CU(I+1,J))
     2      -TDTSDY*(H(I,J+1)-H(I,J))
         PNEW(I,J) = POLD(I,J)-TDTSDX*(CU(I+1,J)-CU(I,J))
     1      -TDTSDY*(CV(I,J+1)-CV(I,J))
  200 CONTINUE
```

```
      DO 210 J=1,N
         UNEW(1,J) = UNEW(M+1,J)
         VNEW(M+1,J+1) = VNEW(1,J+1)
         PNEW(M+1,J) = PNEW(1,J)
  210 CONTINUE
```

```
      DO 300 J=1,N
      DO 300 I=1,M
         UOLD(I,J) = U(I,J)+ALPHA*(UNEW(I,J)-2.*U(I,J)+UOLD(I,J))
         VOLD(I,J) = V(I,J)+ALPHA*(VNEW(I,J)-2.*V(I,J)+VOLD(I,J))
         POLD(I,J) = P(I,J)+ALPHA*(PNEW(I,J)-2.*P(I,J)+POLD(I,J))
  300  CONTINUE
```

(a) An example of target loop group for data localization



(b) Image of alignment of arrays on cache accessed by target loops

Data Layout for Removing Line Conflict Misses by Array Dimension Padding

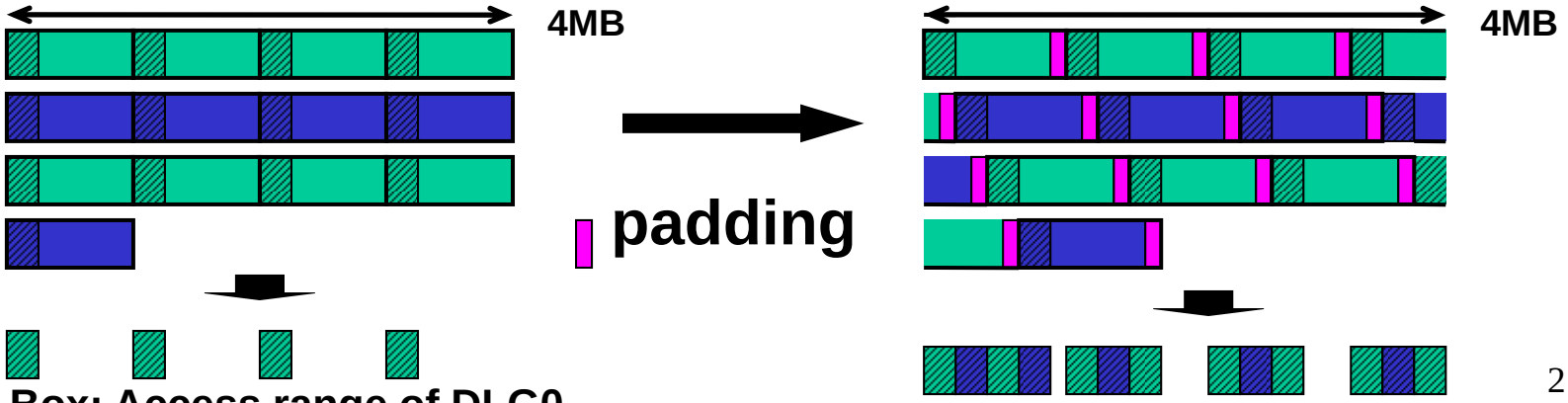
Declaration part of arrays in spec95 swim

before padding

```
PARAMETER (N1=513, N2=513)

COMMON  U(N1,N2), V(N1,N2), P(N1,N2),
*       UNEW(N1,N2), VNEW(N1,N2),
1       PNEW(N1,N2), UOLD(N1,N2),
*       VOLD(N1,N2), POLD(N1,N2),
2       CU(N1,N2), CV(N1,N2),
*       Z(N1,N2), H(N1,N2)
```

after padding

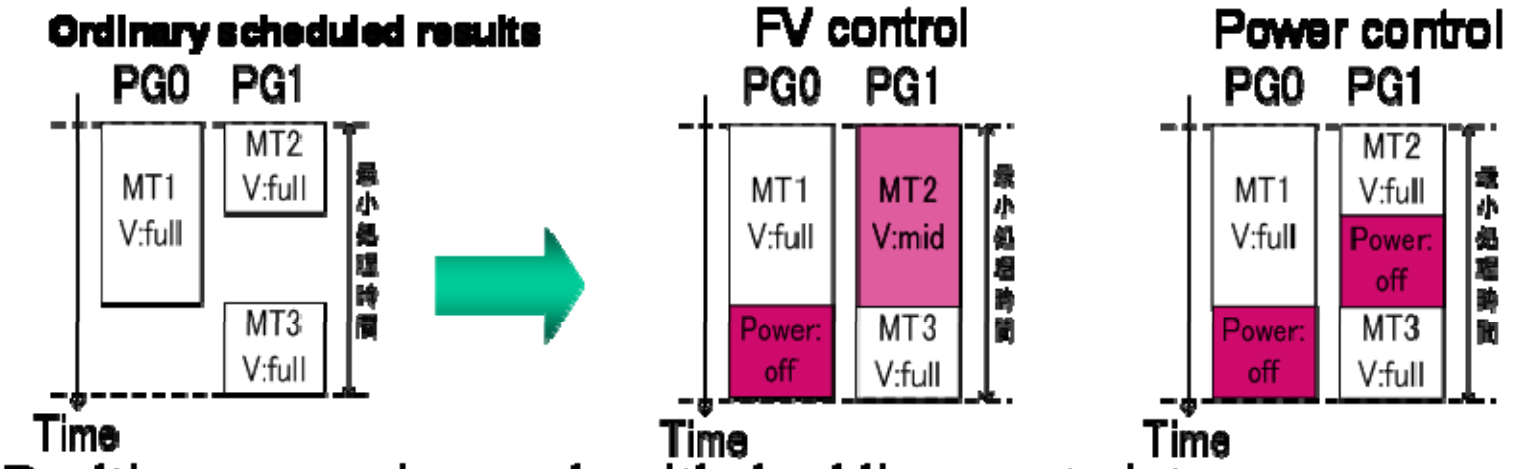
```
PARAMETER (N1=513, N2=544)

COMMON  U(N1,N2), V(N1,N2), P(N1,N2),
*       UNEW(N1,N2), VNEW(N1,N2),
1       PNEW(N1,N2), UOLD(N1,N2),
*       VOLD(N1,N2), POLD(N1,N2),
2       CU(N1,N2), CV(N1,N2),
*       Z(N1,N2), H(N1,N2)
```



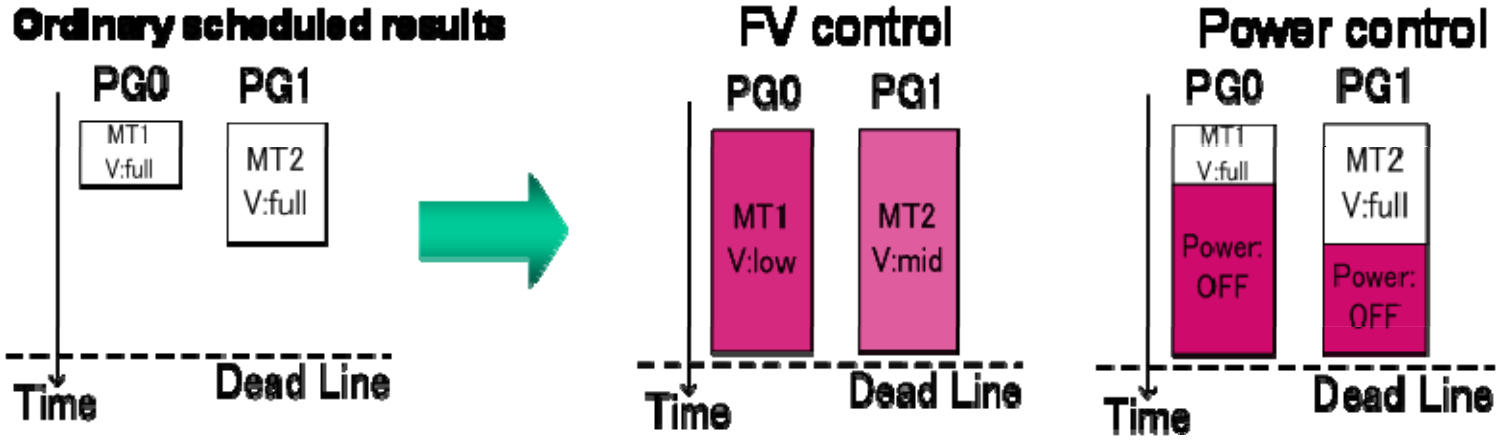
Box: Access range of DLG0

Power Reduction by Power Supply, Clock Frequency and Voltage Control by OSCAR Compiler

- Shortest execution time mode

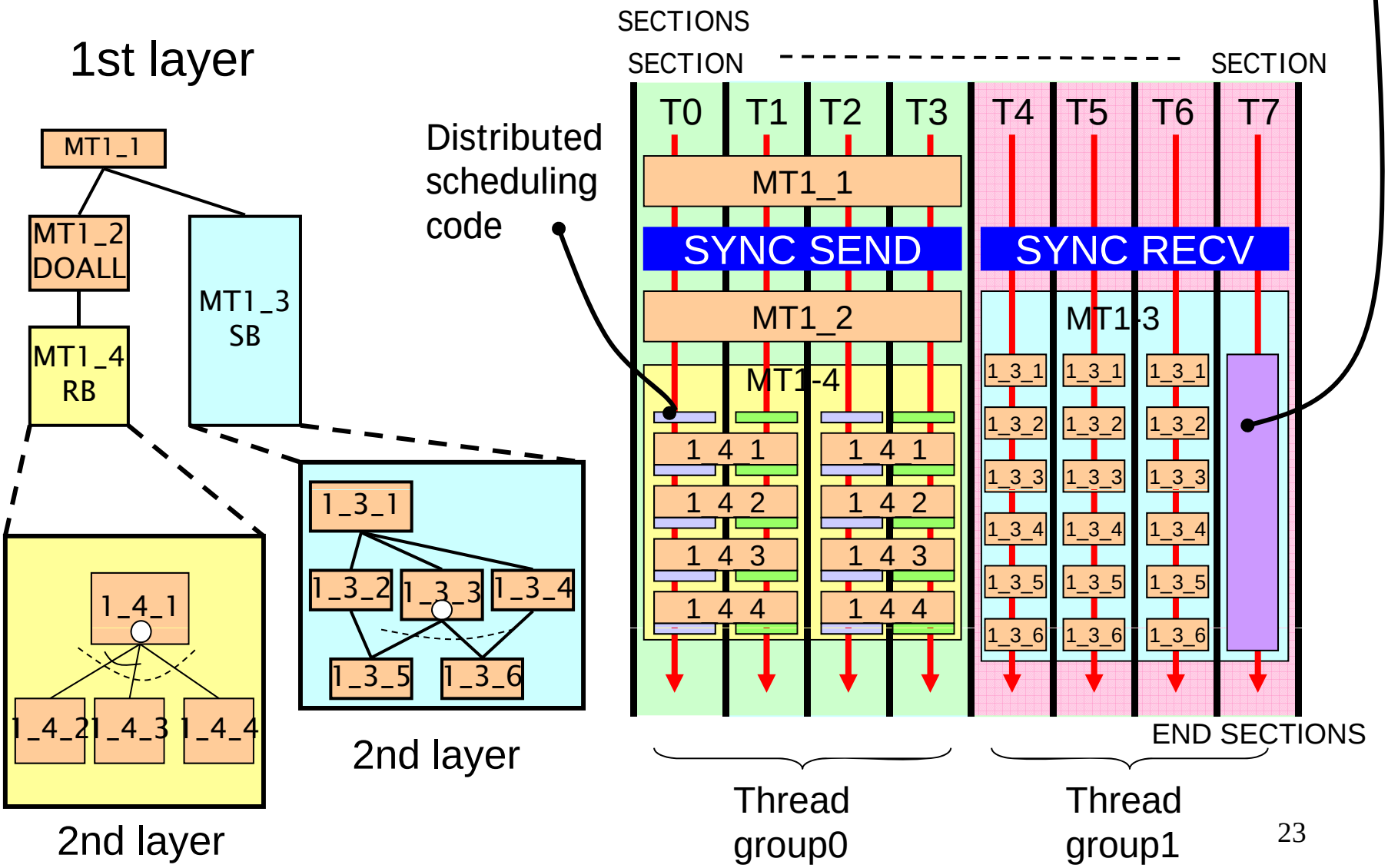


- Realtime processing mode with dead line constraints

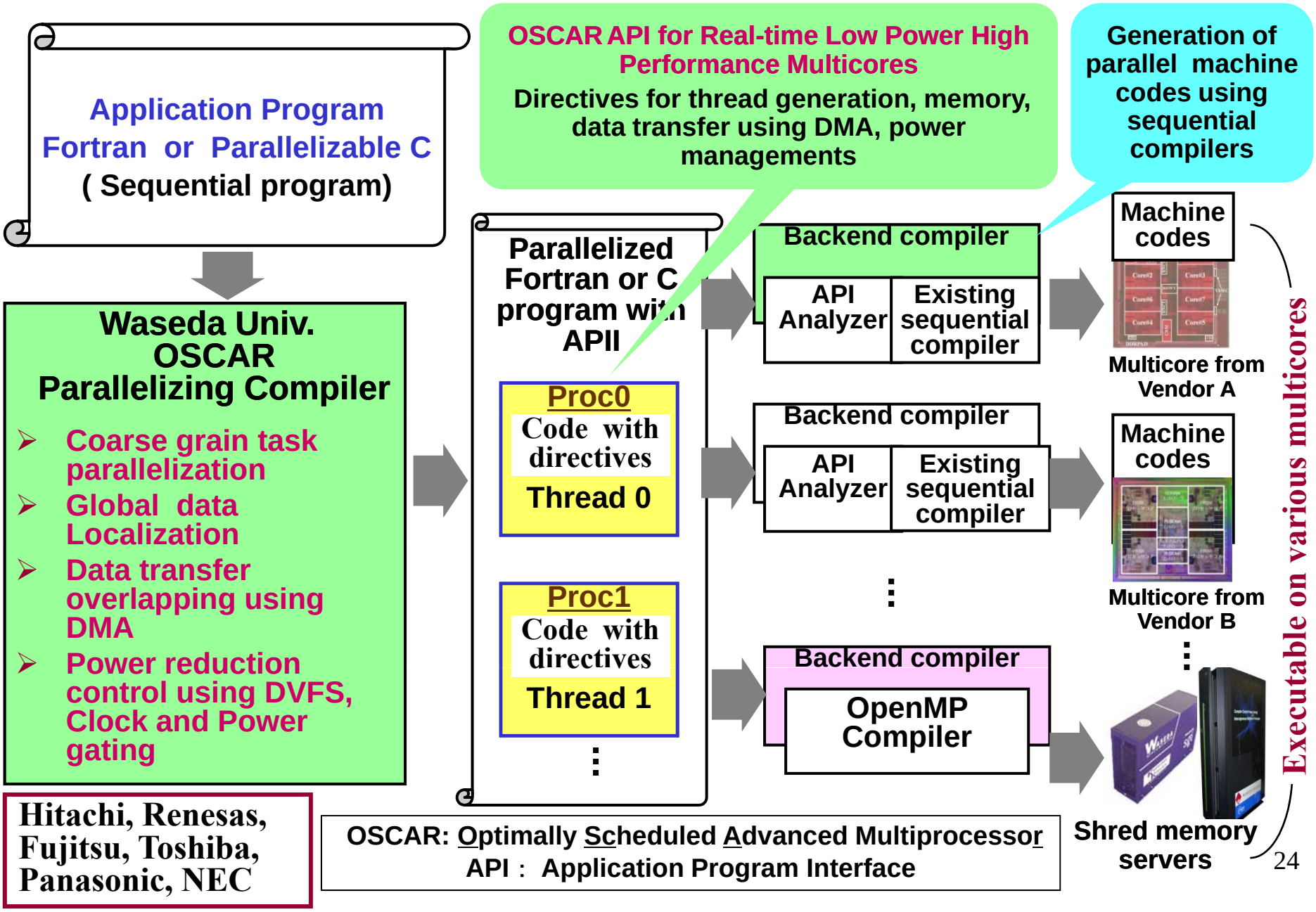


Generated Multigrain Parallelized Code

(The nested coarse grain task parallelization is realized by only OpenMP "section", "Flush" and "Critical" directives.)



Compilation Flow Using OSCAR API



OSCAR API

Now Open! (http://www.kasahara.cs.waseda.ac.jp/)

- **Targeting mainly realtime consumer electronics devices**
  - **embedded computing**
  - **various kinds of memory architecture**
    - **SMP, local memory, distributed shared memory, ...**
- **Developed with Japanese 6 companies**
  - **Fujitsu, Hitachi, NEC, Toshiba, Panasonic, Renesas**
  - **Supported by METI/NEDO**
- **Based on the subset of OpenMP**
  - **very popular parallel processing API**
  - **shared memory programming model**
- **Six Categories**
  - **Parallel Execution (4 directives from OpenMP)**
  - **Memory Mapping (Distributed Shared Memory, Local Memory)**
  - **Data Transfer Overlapping Using DMA Controller**
  - **Power Control (DVFS, Clock Gating, Power Gating)**
  - **Timer for Real Time Control**
  - **Synchronization (Hierarchical Barrier Synchronization)**

Current Application Development Environment for Multicores

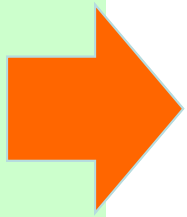
- Parallel API
  - pthread (SMP)
    - Old thread library
  - OpenMP (Shared Memory), MPI (Distributed Memory)
    - for Scientific Applications
  - Co-array Fortran (PGAS), UPC (PGAS)
    - Language extension
  - MCAPI (Distributed Memory)
    - for Embedded Applications
    - Message Passing API
  - NO Good API for Low-Power and Real-time Multicores!
- Parallelizing Compilers for Scientific Applications (Fortran Applications)
  - Several aggressive compilers from academia
    - Polaris, SUIF, CETUS, Pluto,OSCAR, …
    - Source-to-source Compiler
  - Parallelizing Compilers for Low-Power and Real-time Multicores
    - OSCAR

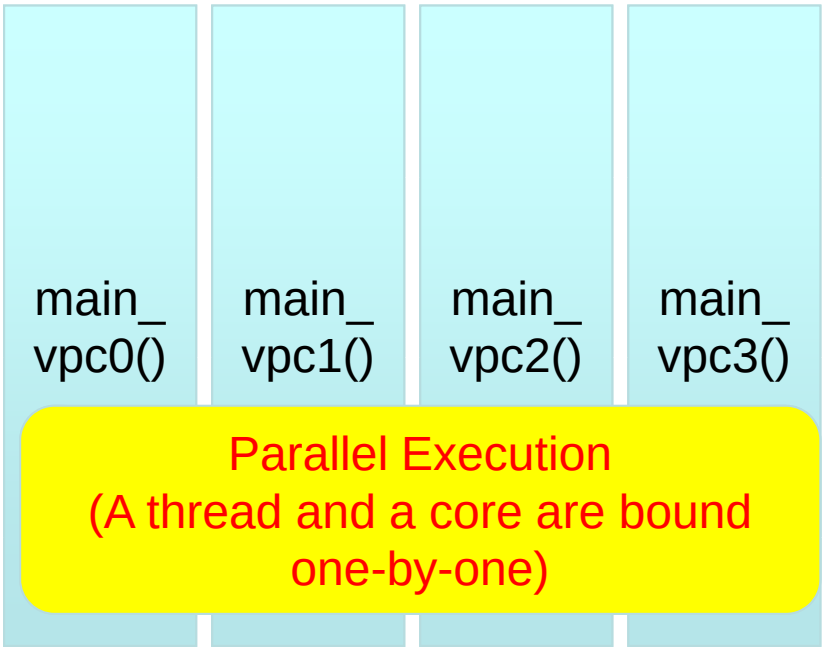
Parallel Execution

- **Start of parallel execution**
  - **#pragma omp parallel sections (C)**
  - **!$omp parallel sections (Fortran)**
- **Specifying critical section**
  - **#pragma omp critical (C)**
  - **!$omp critical (Fortran)**
- **Enforcing an order of the memory operations**
  - **#pragma omp flush (C)**
  - **!$omp flush (Fortran)**
- **These are from OpenMP.**

Thread Execution Model

```
#pragma omp parallel sections
    {
#pragma omp section
        main_vpc0();
#pragma omp section
        main_vpc1();
#pragma omp section
        main_vpc2();
#pragma omp section
        main_vpc3();
    }
```





VPC: Virtual Processor Core

Memory Mapping

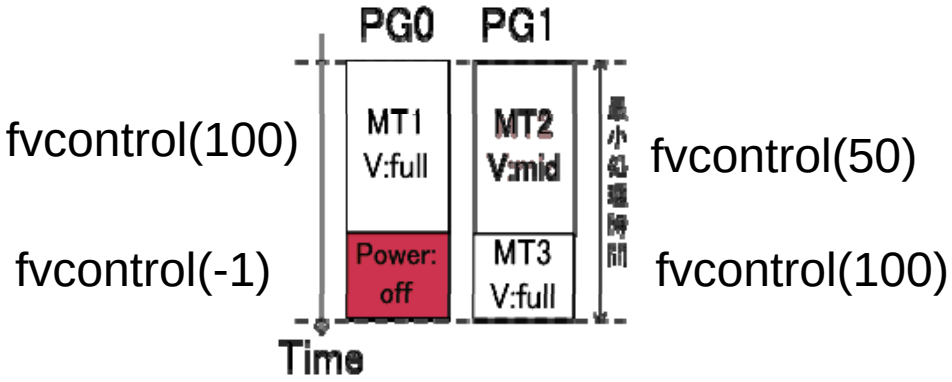
- **Placing variables on an onchip centralized shared memory (onchipCSM)**
  - **#pragma oscar onchipshared (C)**
  - **!$oscar onchipshared (Fortran)**
- **Placing variables on a local data memory (LDM)**
  - **#pragma omp threadprivate (C)**
  - **!$omp threadprivate (Fortran)**
  - **This directive is an extension to OpenMP**
- **Placing variables on a distributed shared memory (DSM)**
  - **#pragma oscar distributedshared (C)**
  - **!$oscar distributedshared (Fortran)**

Data Transfer

- Specifying data transfer lists
  - #pragma oscar dma_transfer (C)
  - !$oscar dma_transfer (Fortran)
  - Containing following parameter directives
- Specifying a contiguous data transfer
  - #pragma oscar dma_contiguous_parameter (C)
  - !$oscar dma_contiguous_parameter (Fortran)
- Specifying a stride data transfer
  - #pragma oscar dma_stride_parameter
  - !$oscar dma_stride_parameter
  - This can be used for scatter/gather data transfer
- Data transfer synchronization
  - #pragma oscsar dma_flag_check
  - !$oscar dma_flag_check

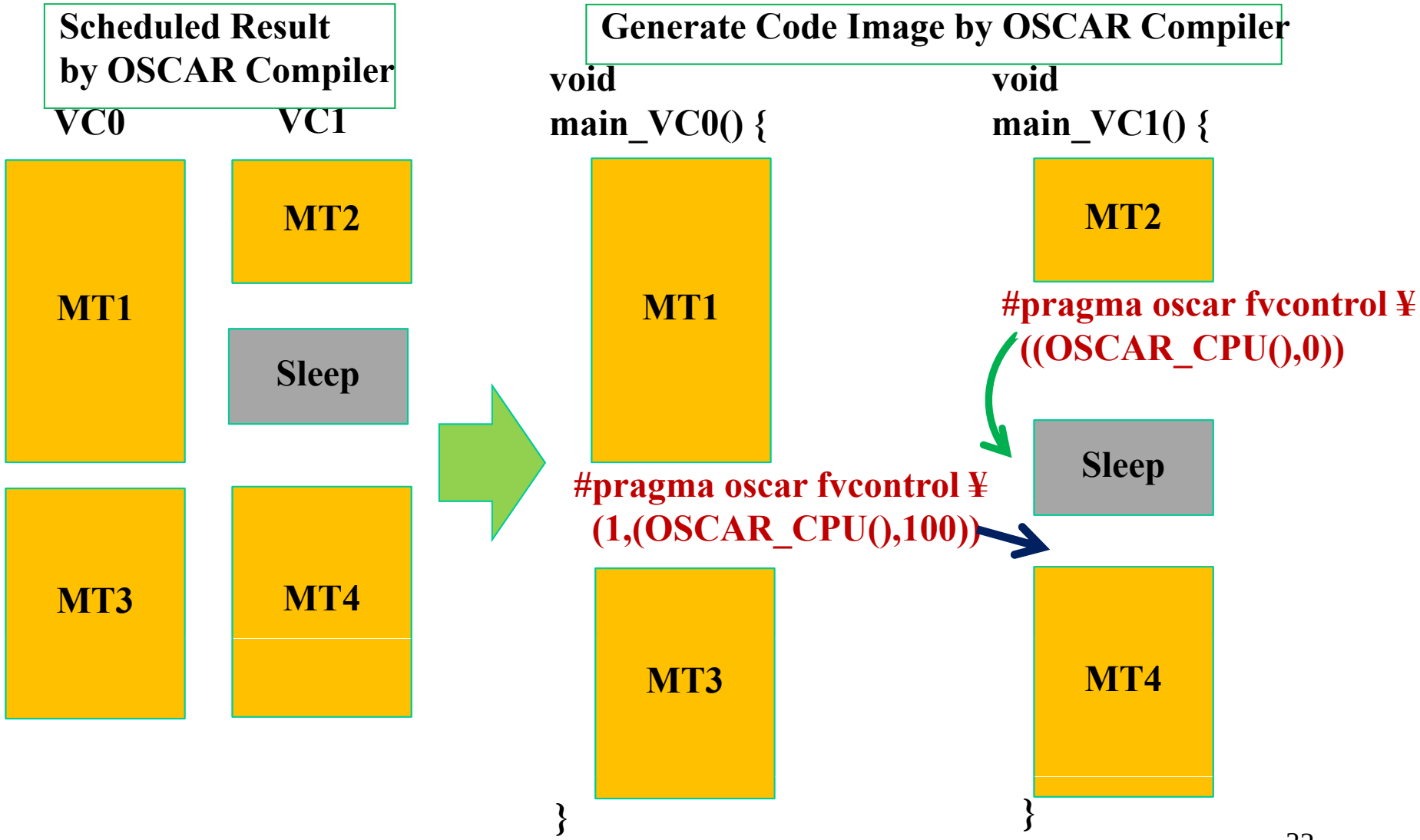
Power Control

- **Making a module into specifying frequency and voltage state**
  - **#pragma oscar fvcontrol (C)**
  - **!$oscar fvcontrol (Fortran)**
  - **state examples**
    - **100: max frequency**
    - **50: half frequency**
    - **0: clock off**
    - **-1: power off**



- **Getting a frequency and voltage state of a module**
  - **#pragma oscar get_fvstatus (C)**
  - **!$oscar get_fvstatus (Fortran)**

Low-Power Optimization with OSCAR API

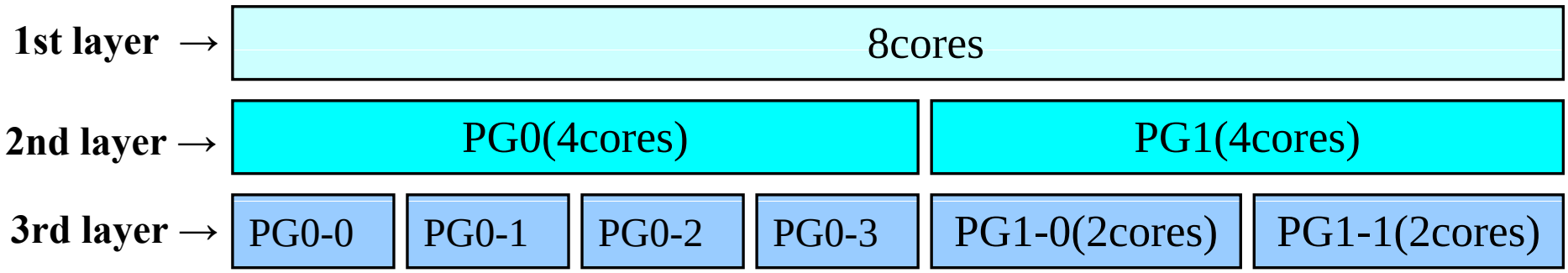


Timer

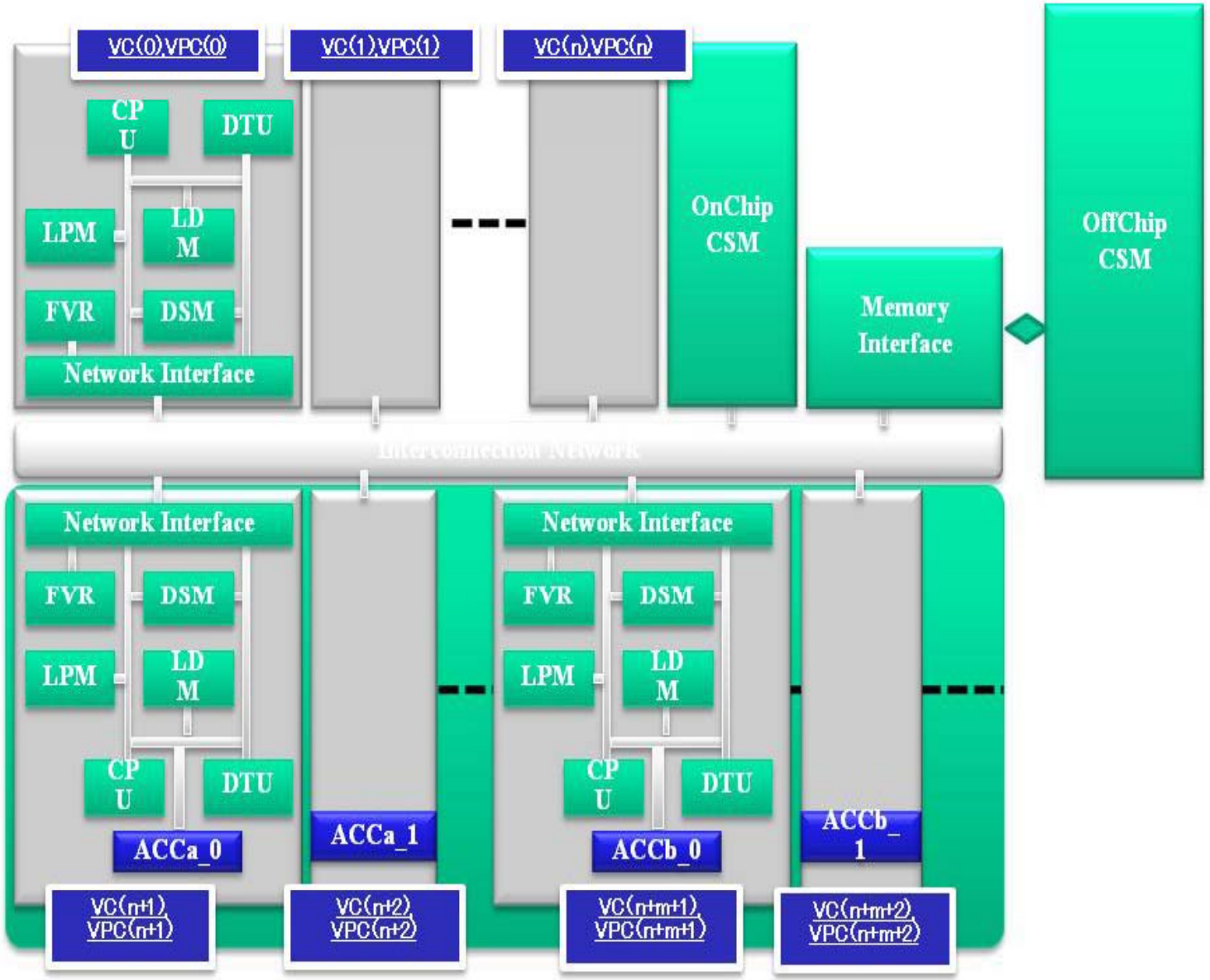
- **Getting an elapsed wall clock time in microseconds**
  - **#pragma oscar get_current_time (C)**
  - **!$oscar get_current_time (Fortran)**
- **For realtime execution**

Hierarchical Barrier Synchronization

- **Specifying a hierarchical group barrier**
  - **#pragma oscar group_barrier (C)**
  - **!$oscar group_barrier (Fortran)**



OSCAR Heterogeneous Multicore



DTU
- Data Transfer Unit

LPM
- Local Program Memory

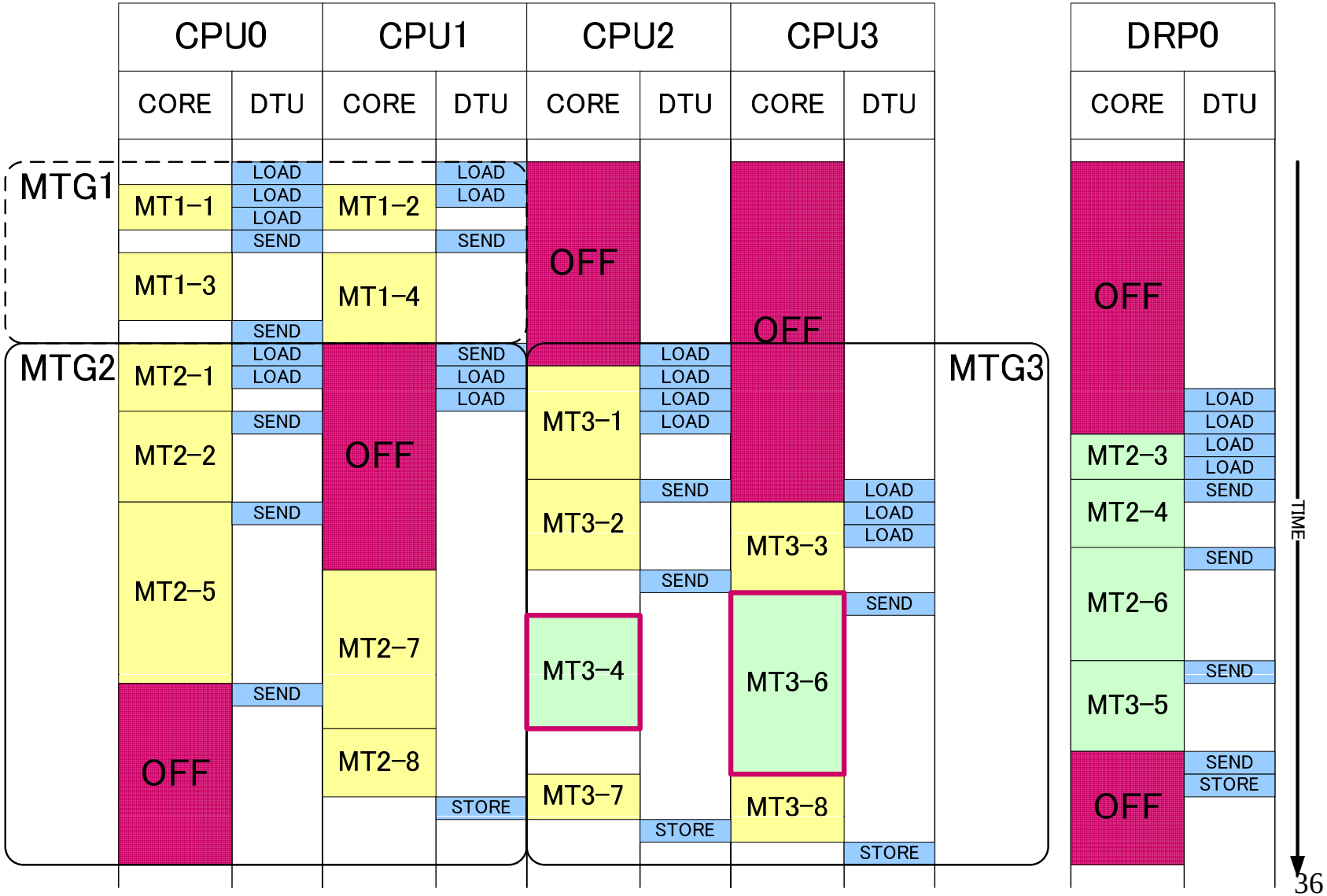
LDM
- Local Data Memory

DSM
- Distributed Shared Memory

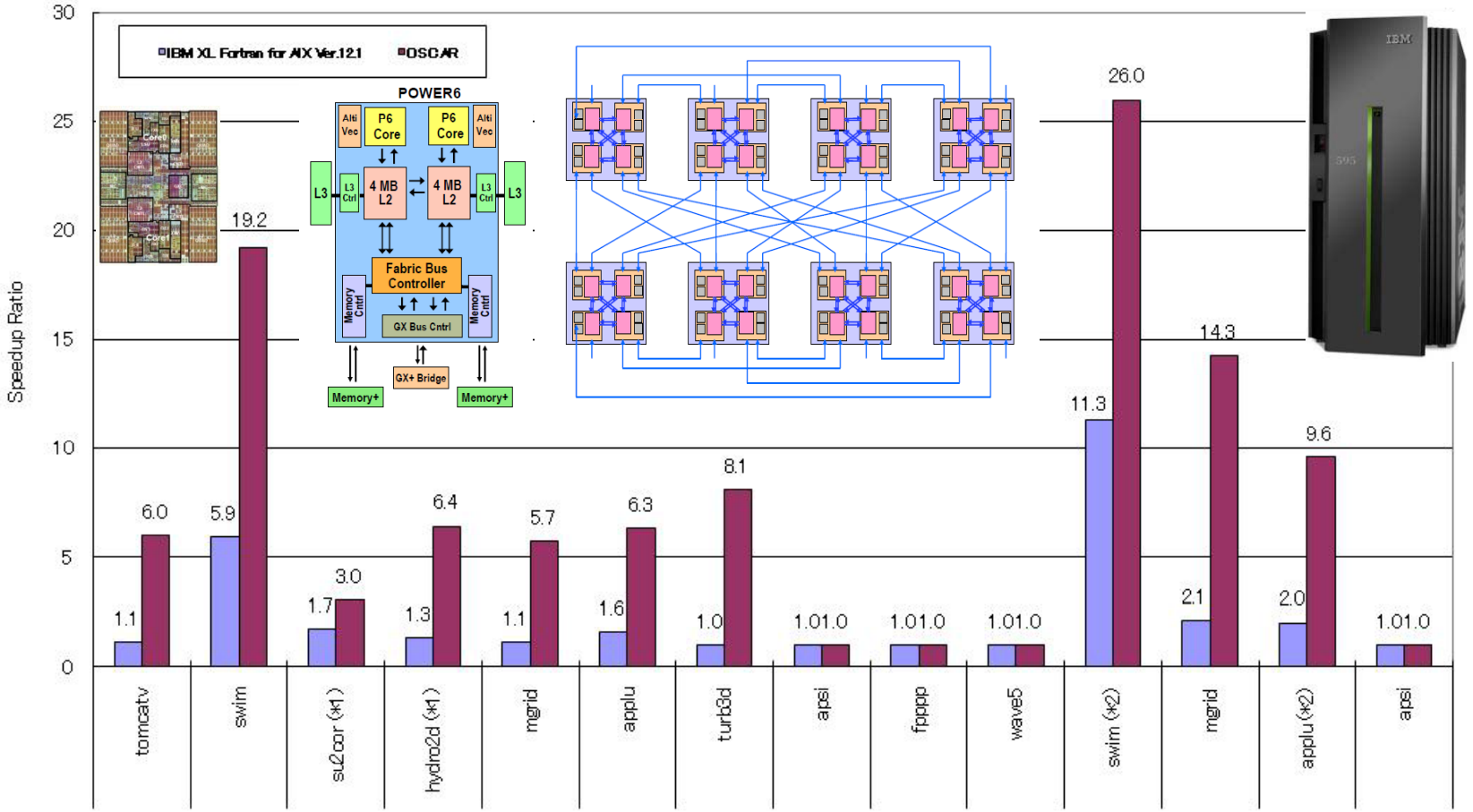
CSM
- Centralized Shared Memory

FVR
- Frequency/Voltage Control Register

An Image of Static Schedule for Heterogeneous Multi-core with Data Transfer Overlapping and Power Control



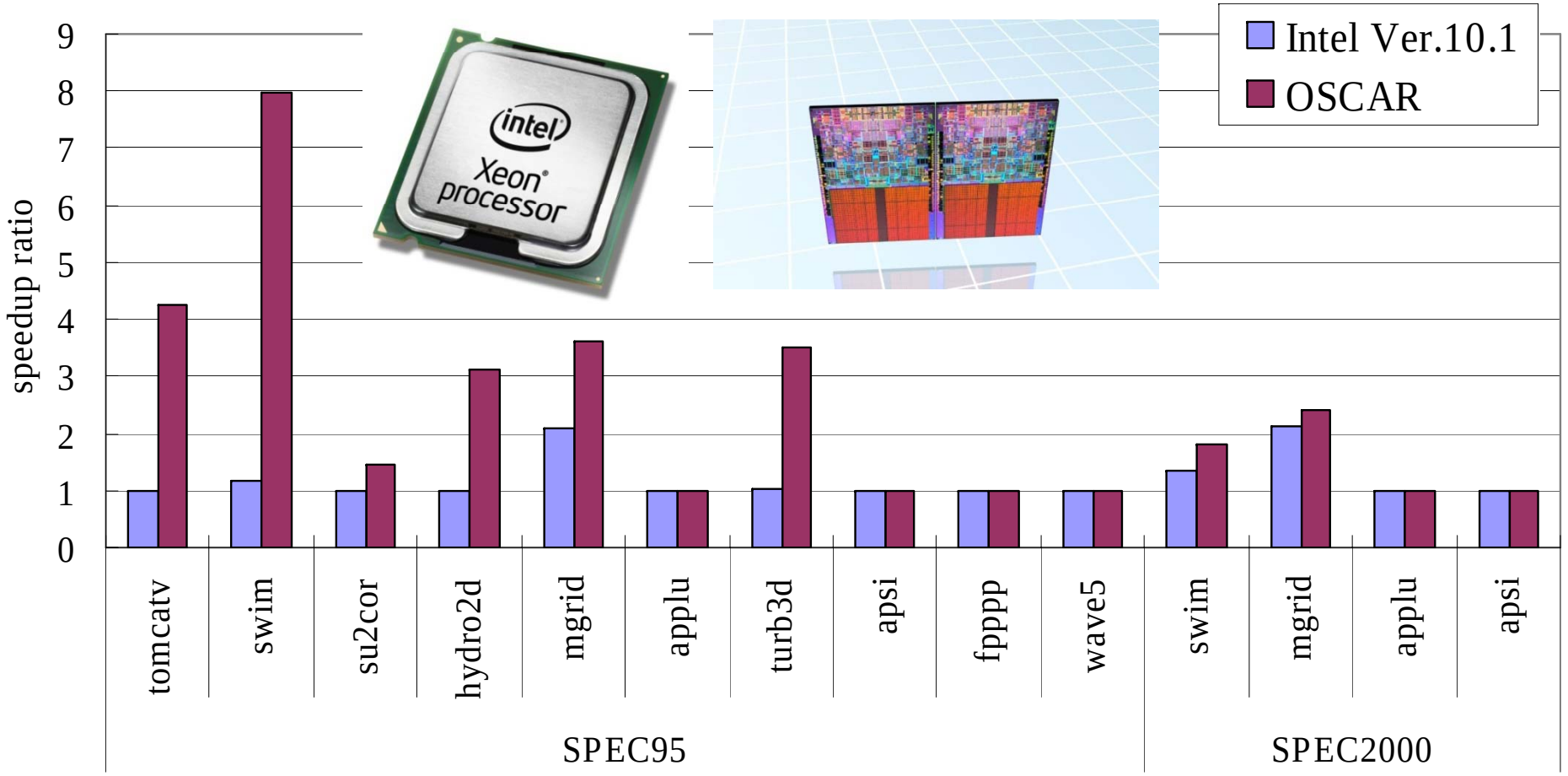
Performance of OSCAR Compiler on IBM p6 595 Power6 (4.2GHz) based 32-core SMP Server



OpenMP codes generated by OSCAR compiler accelerate IBM XL Fortran for AIX Ver.12.1 about 3.3 times on the average

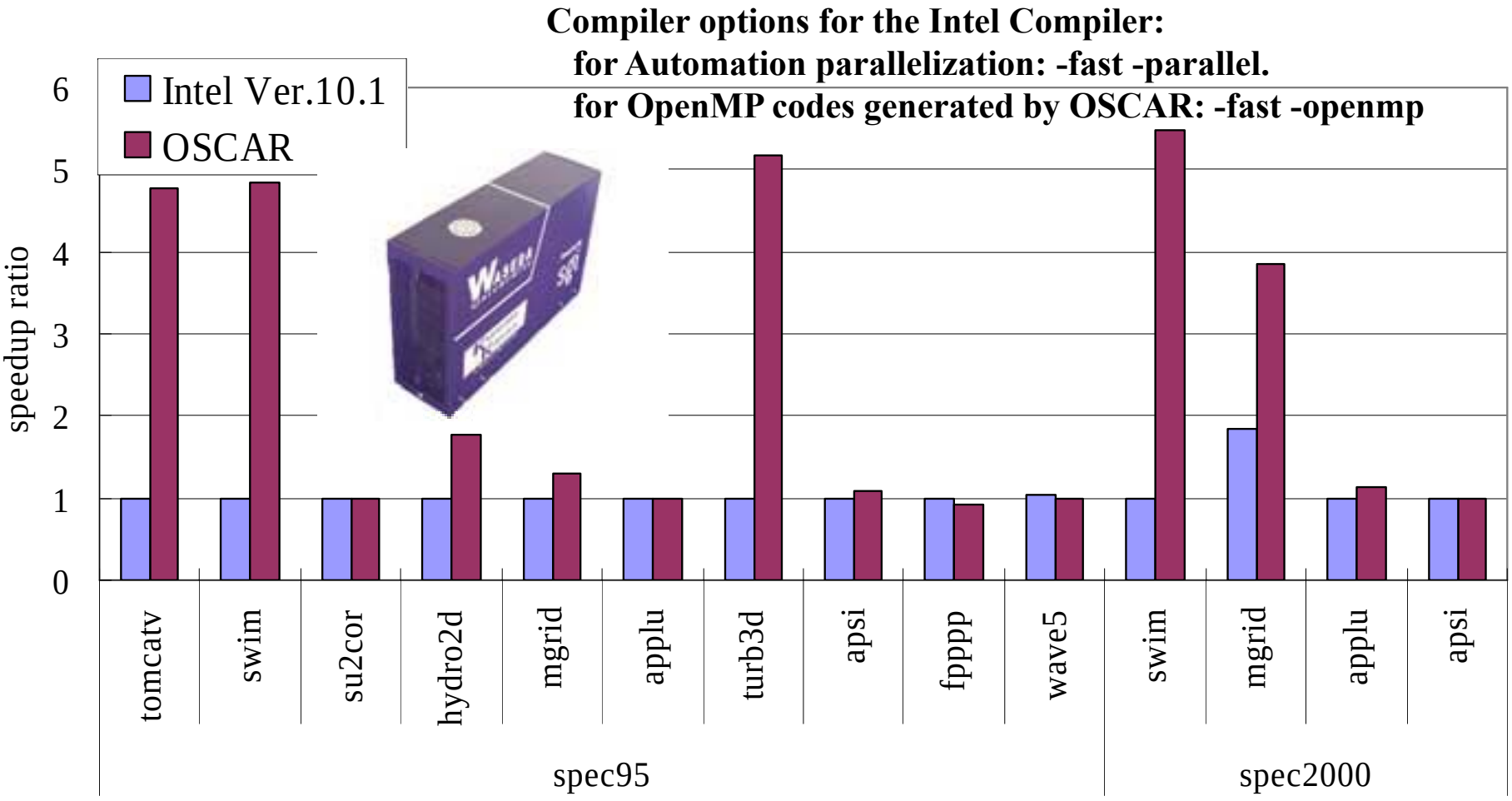
Compile Option:
(*1) Sequential: -O3 –qarch=pwr6, XLF: -O3 –qarch=pwr6 –qsmp=auto, OSCAR: -O3 –qarch=pwr6 –qsmp=noauto
(*2) Sequential: -O5 -q64 –qarch=pwr6, XLF: -O5 –q64 –qarch=pwr6 –qsmp=auto, OSCAR: -O5 –q64 –qarch=pwr6 –qsmp=noauto
(Others) Sequential: -O5 –qarch=pwr6, XLF: -O5 –qarch=pwr6 –qsmp=auto, OSCAR: -O5 –qarch=pwr6 –qsmp=noauto

Performance of OSCAR Compiler Using the Multicore API on Intel Quad-core Xeon



- **OSCAR Compiler gives us 2.1 times speedup on the average against Intel Compiler ver.10.1**

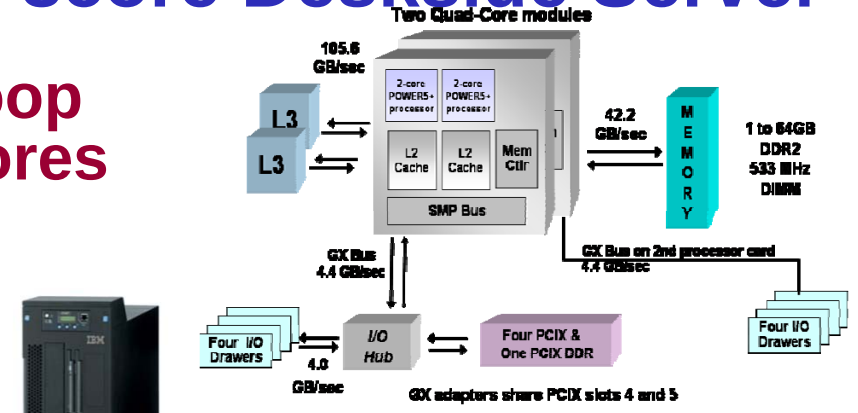
Performance of OSCAR compiler on 16 cores SGI Altix 450 Montvale server



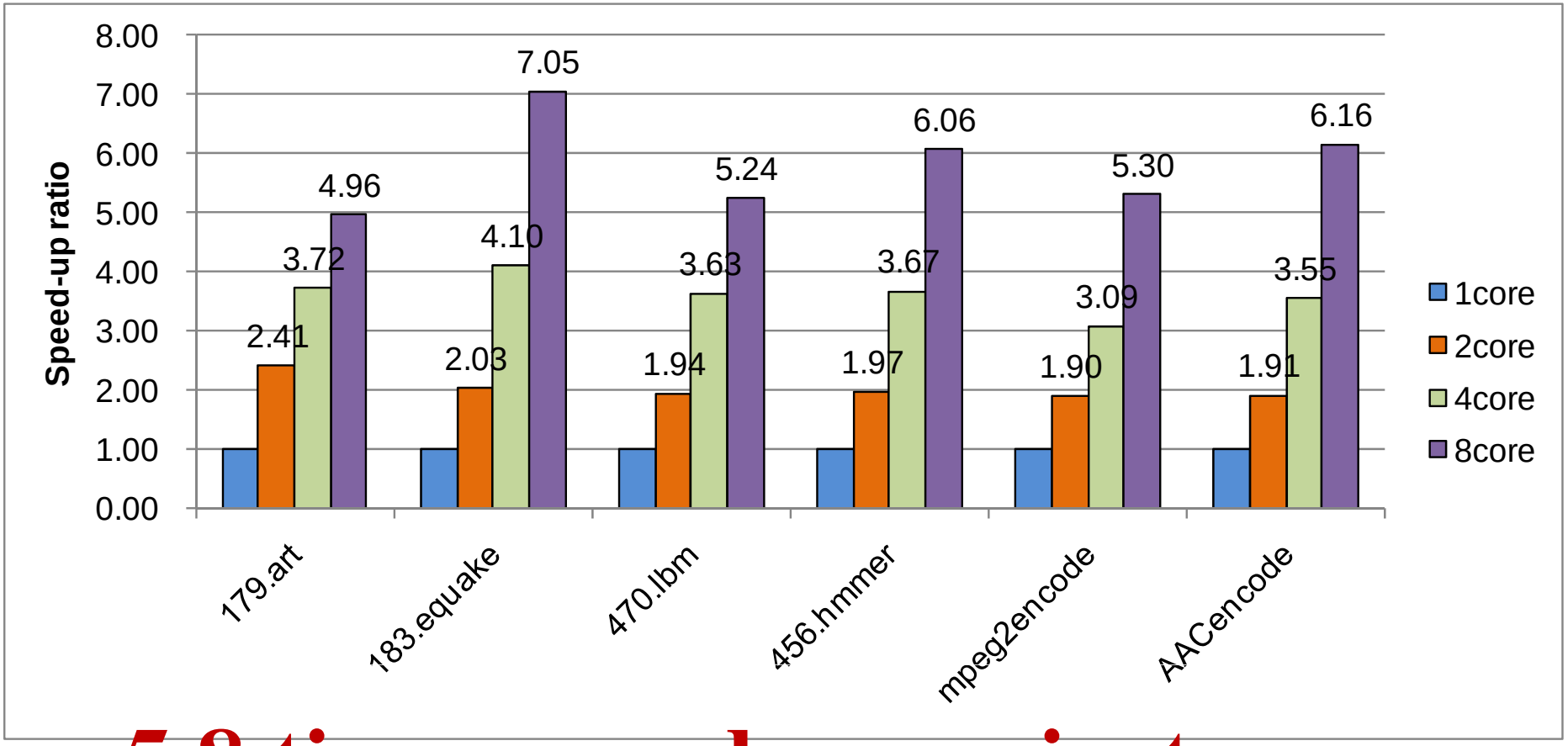
- **OSCAR compiler gave us 2.3 times speedup against Intel Fortran Itanium Compiler revision 10.1**

Performance of OSCAR Compiler for Fortran Programs on a IBM p550q 8core Deskside Server

- 2.7 times speedup against loop parallelizing compiler on 8 cores

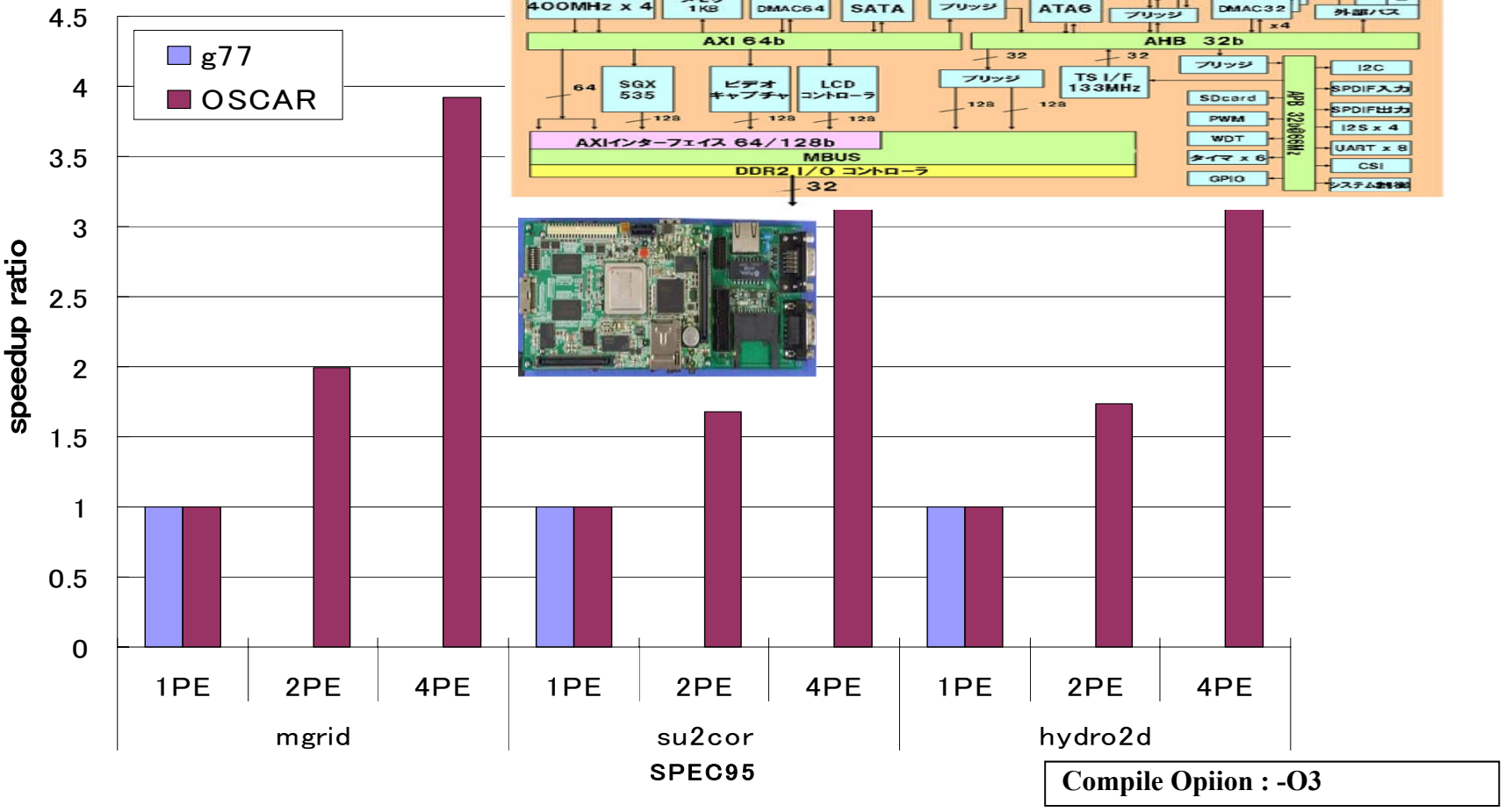


Performance for C programs on IBM p5 550Q



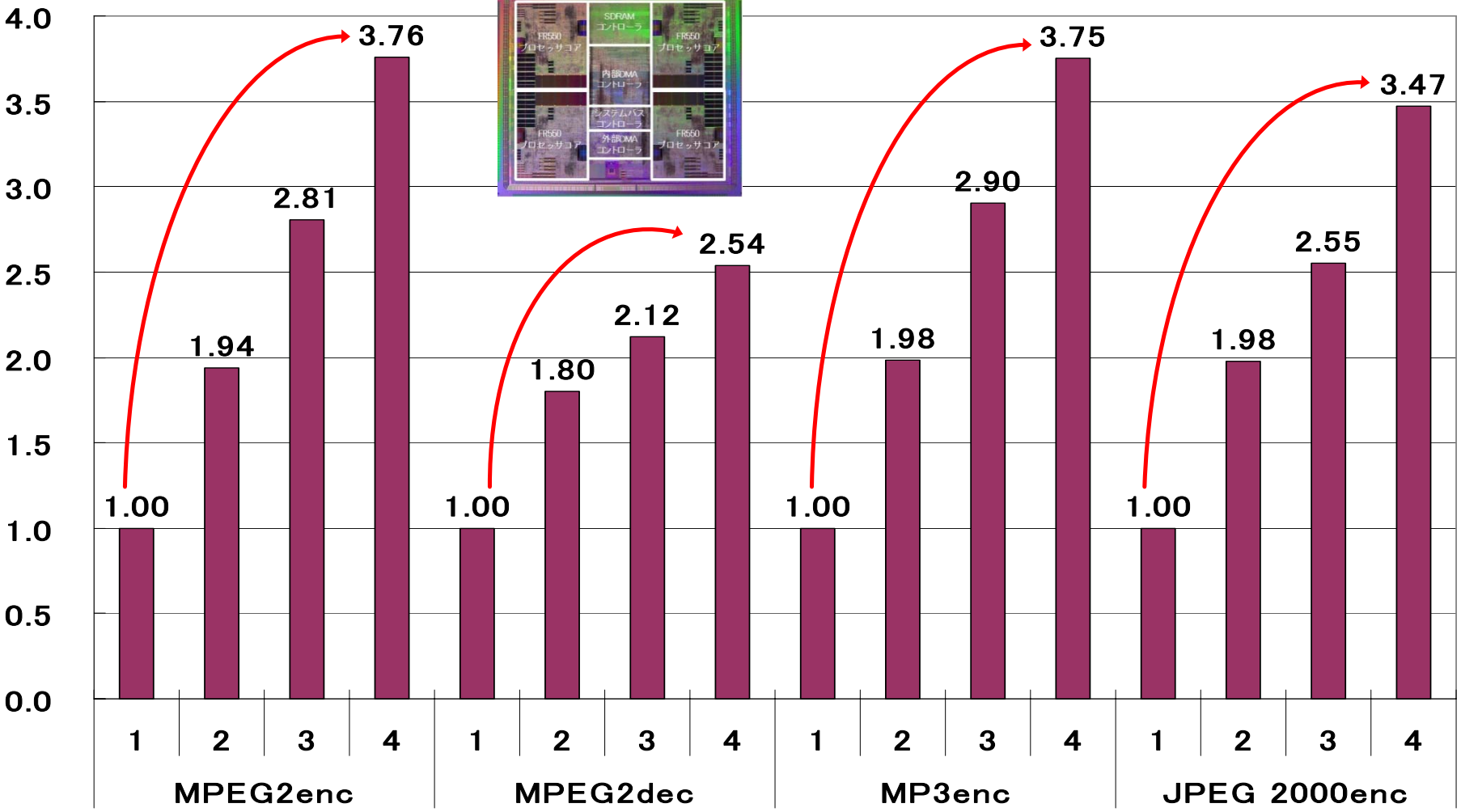
5.8 times speedup against one processor on average

Performance of OSCAR compiler on NEC NaviEngine(ARM-NEC MPcore)



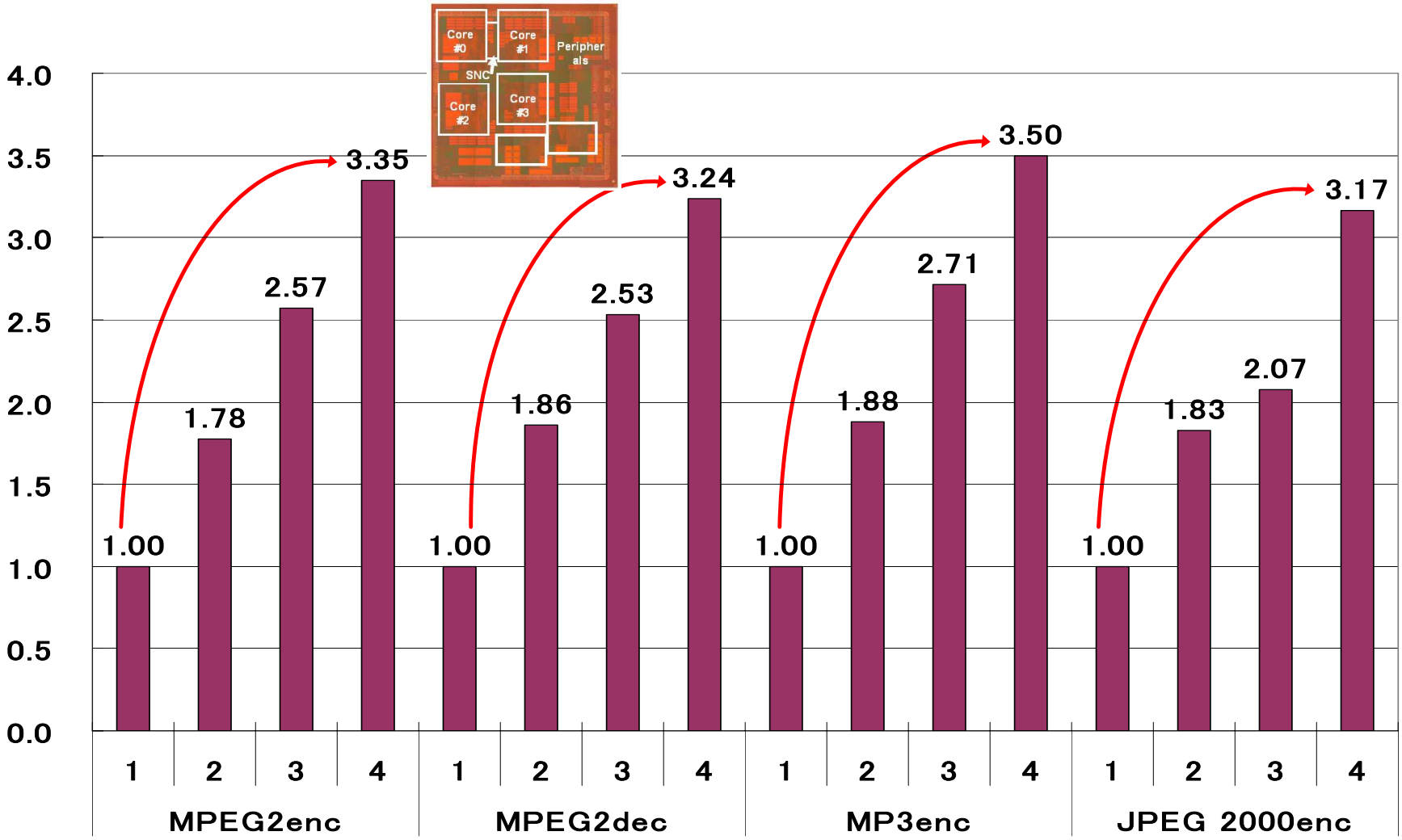
- OSCAR compiler gave us 3.43 times speedup against 1 core on ARM/NEC MPCore with 4 ARM 400MHz cores

Performance of OSCAR Compiler Using the multicore API on Fujitsu FR1000 Multicore



3.38 times speedup on the average for 4 cores against a single core execution

Performance of OSCAR Compiler Using the Developed API on 4 core (SH4A) OSCAR Type Multicore

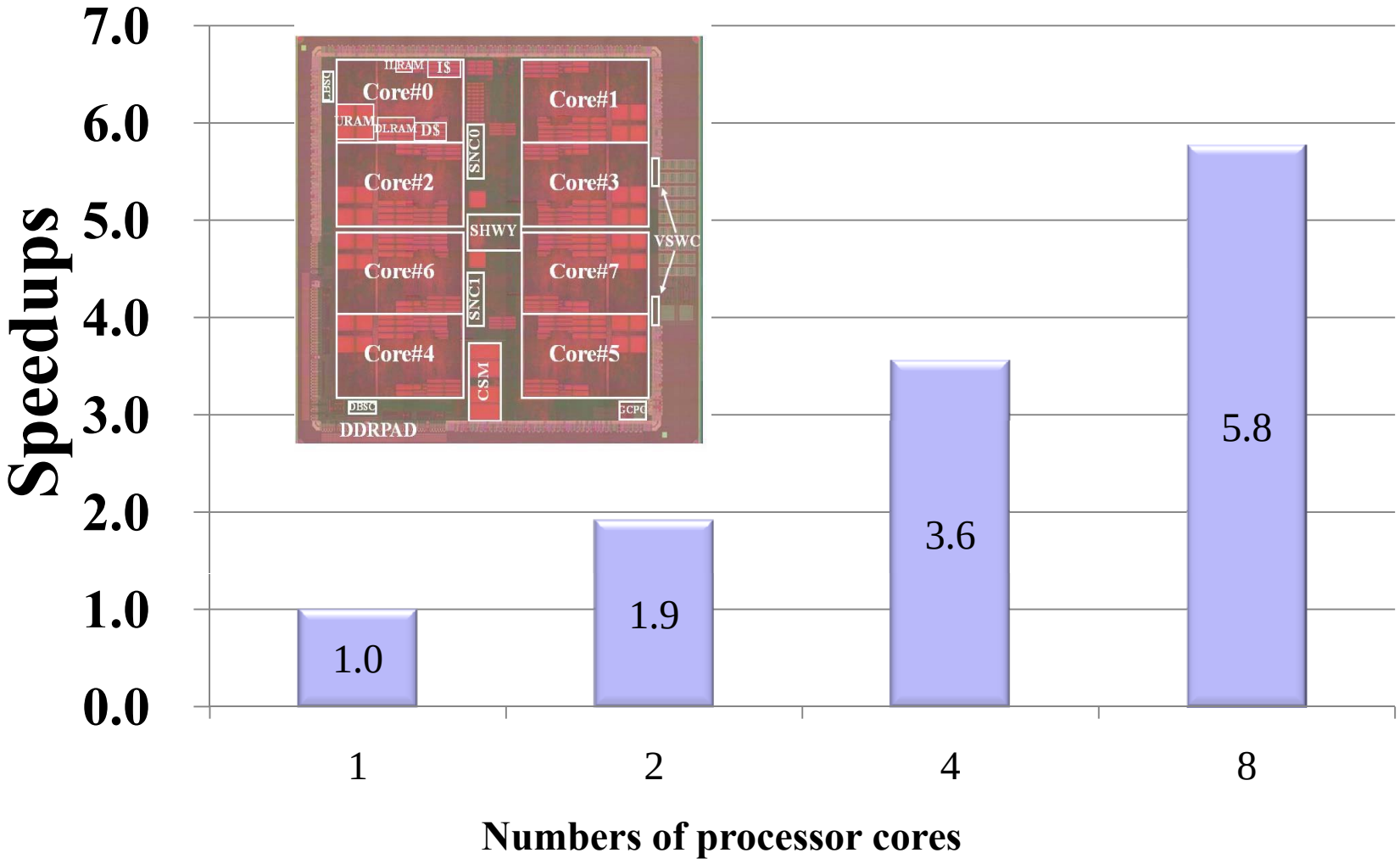


3.31 times speedup on the average for 4cores against 1core

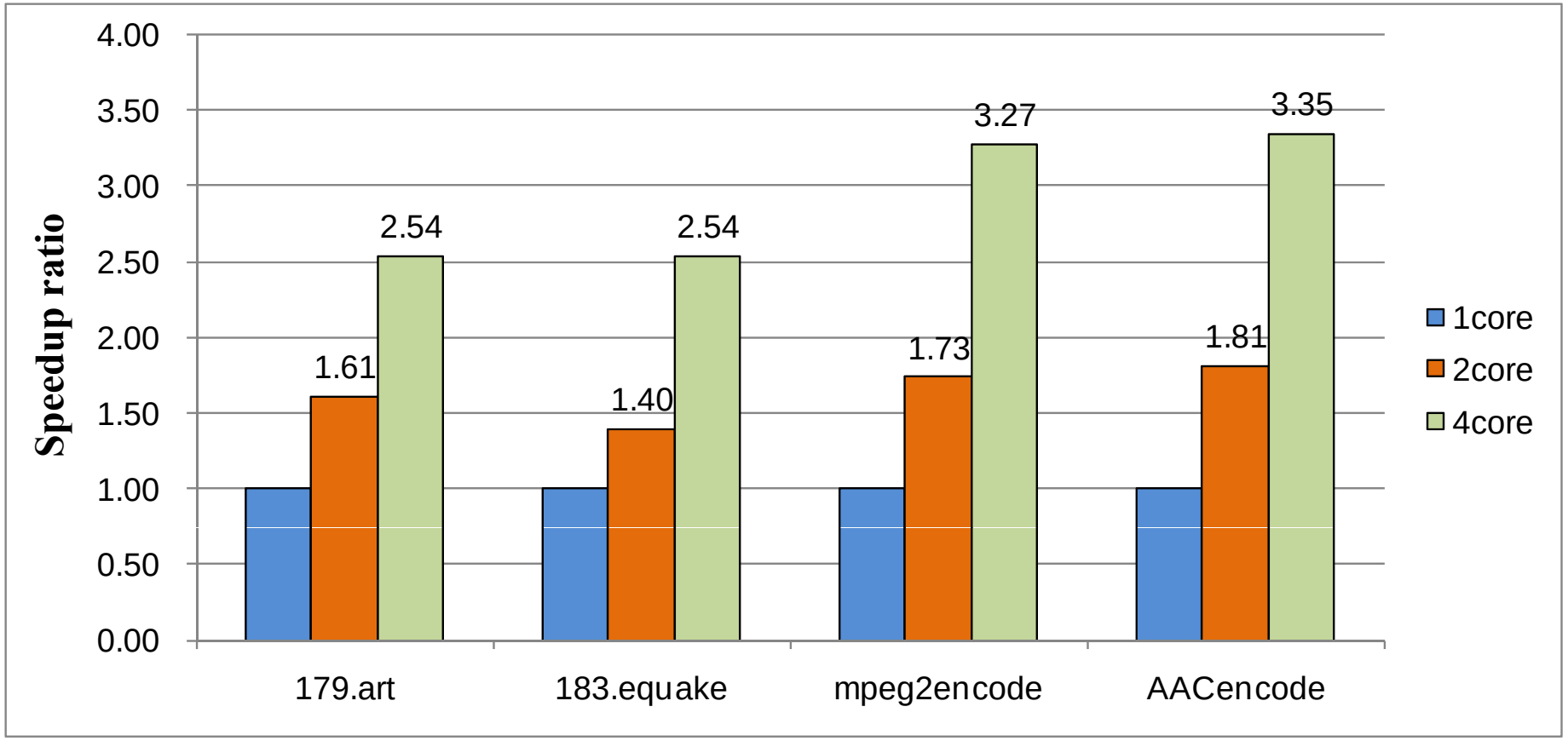
Processing Performance on the Developed Multicore Using Automatic Parallelizing Compiler

Speedup against single core execution for audio AAC encoding

*) Advanced Audio Coding



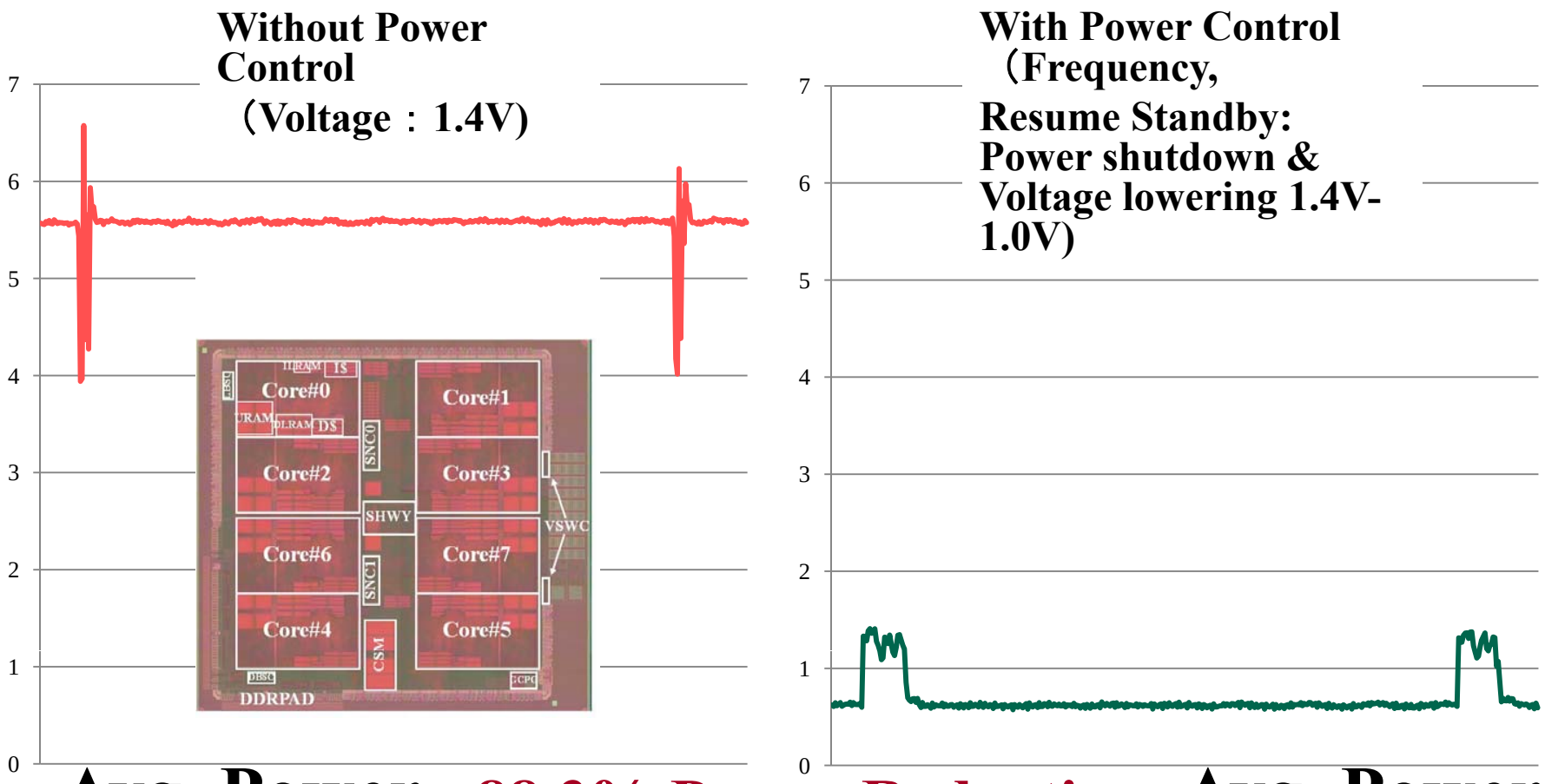
Performance of OSCAR Compiler for C Programs for 4core SMP on RP2



2.9 times speedup on average against one core

Power Reduction by OSCAR Parallelizing Compiler for Secure Audio Encoding

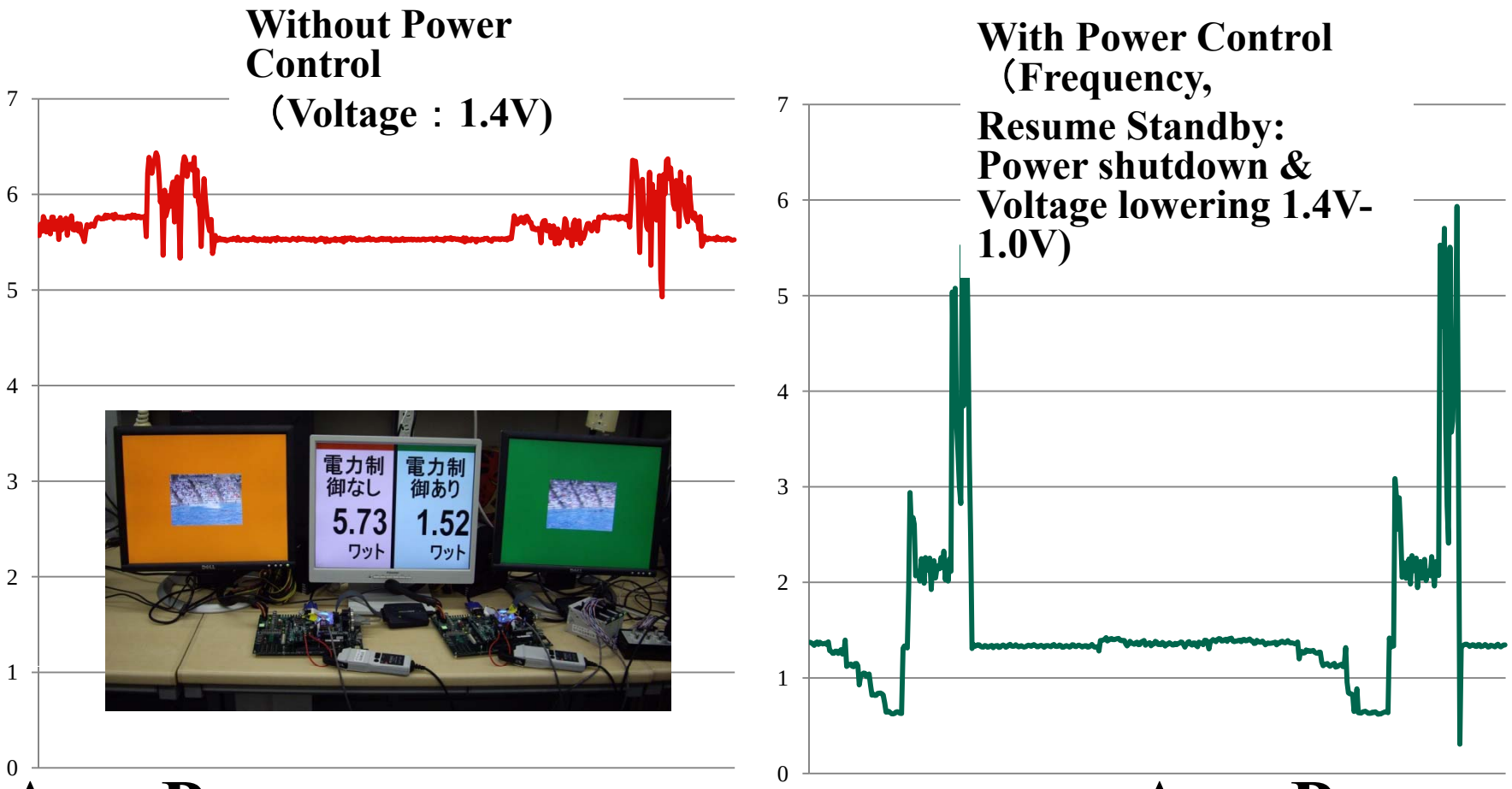
AAC Encoding + AES Encryption with 8 CPU cores



Avg. Power 5.68 [W] → **88.3% Power Reduction** → **Avg. Power 0.67 [W]**

Power Reduction by OSCAR Parallelizing Compiler for MPEG2 Decoding

MPEG2 Decoding with 8 CPU cores



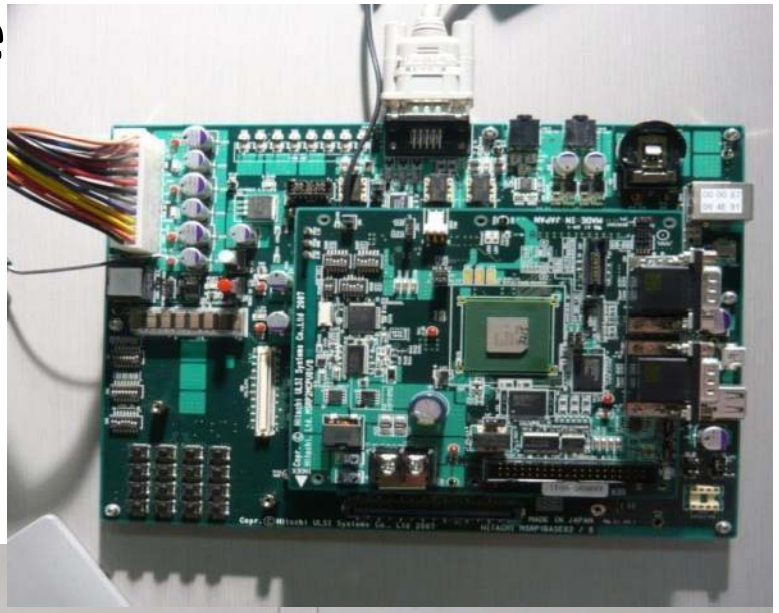
Avg. Power 5.73 [W]

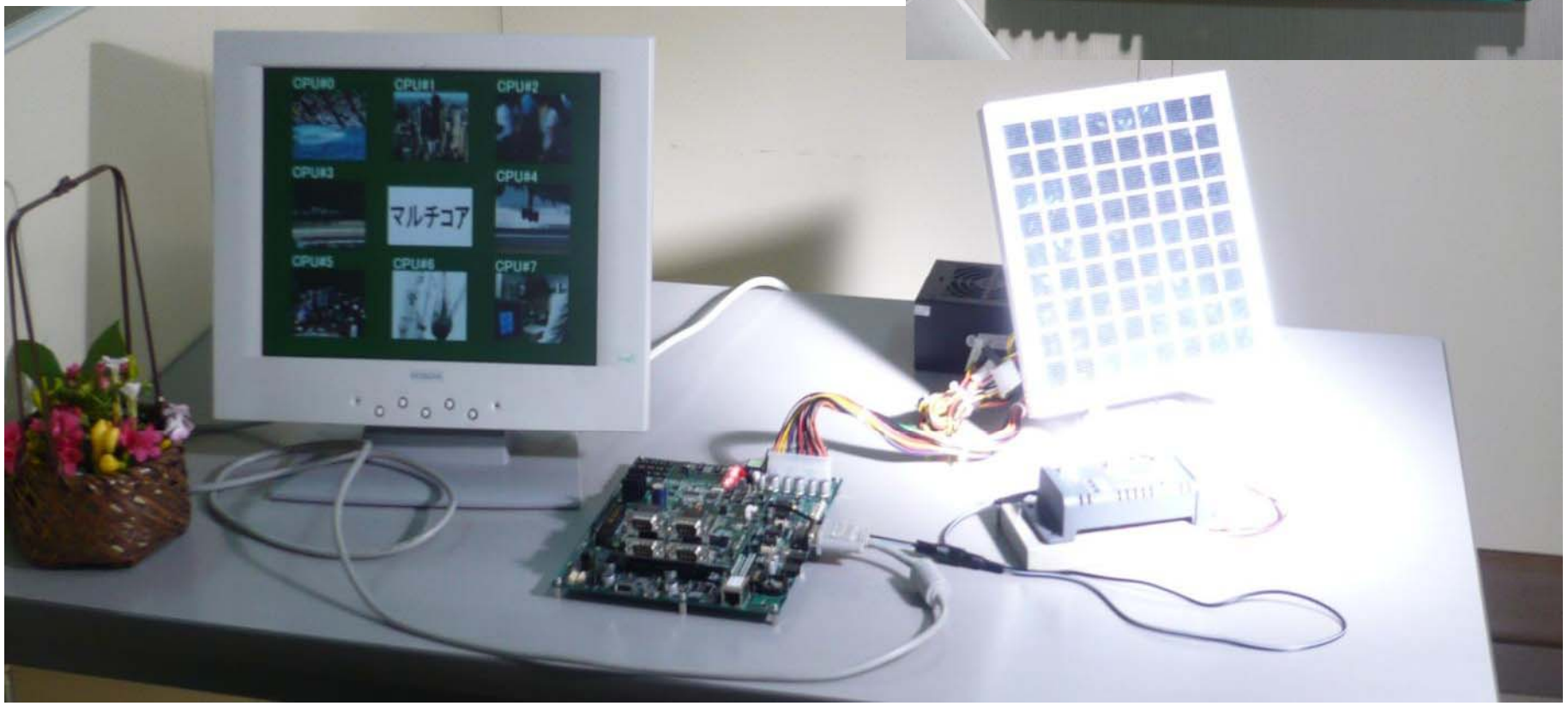
73.5% Power Reduction

Avg. Power 1.52 [W]

Low Power High Performance Multicore Computer with Solar Panel

- Clean Energy Autonomous
- Servers operational in deserts





Green Computing Systems R&D Center

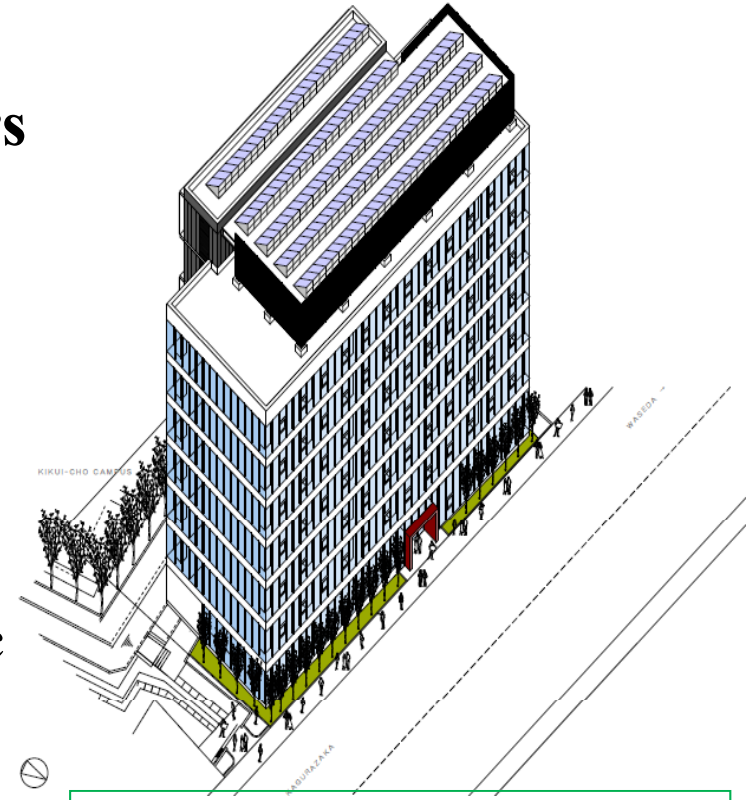
Waseda University

Supported by METI (Mar. 2011 Completion)

＜R & D Target＞

Hardware, Software, Application for Super Low-Power Manycore Processors

- **64-128 cores**
- **Natural air cooling(No cooling fun) Cool, Compact, Clear, Quiet**
- **Possibly operational by Solar Panel**



Beside Subway Waseda Station, Near Waseda Univ. Main Campus

＜Industry, Government, Academia＞

Fujitsu, Hitachi, Renesas, Toshiba, NEC, etc

＜Ripple Effect＞

- **Low CO_2 (Carbon Dioxide) Emissions**
- **Creation Value Added Products**
  - **Consumer Electronics, Automobiles, Servers**

Conclusions

- OSCAR compiler cooperative real-time low power multicore with high effective performance, short software development period will be important in wide range of IT systems from consumer electronics to robotics and automobiles.
- The OSCAR compiler with API allow us boosts up the performance of various multicores and servers and reduce consumed power significantly
  - 3.3 times on IBM p595 SMP server using Power6
  - 2. 1 times on Intel Quad core Xeon
  - 2.3 times on SGI Altix450 using Intel Itanium2 (Montvale)
  - 88% power reduction by the compiler power control on the Renesas-Hitachi-Waseda 8 core (SH4A) multicore RP2 for realtime secure AAC encoding
  - 70% power reduction on the multicore for MPEG2 decoding
- Currently, we are extending API for Hetero and Many core processors.