Prof. Hironori Kasahara

Department of Electrical, Electronics and Computer Engineering

Waseda University

3-4-1 Ohkubo Shinjuku-ku, Tokyo, 169-8555, Japan

E-mail: kasahara@oscar.elec.waseda.ac.jp

URL: http://www.kasahara.elec.waseda.ac.jp

Hironori Kasahara

＜Personal History＞

B.S. (1980,Waseda), M.S.(1982,Waseda), Ph.D.(1985,EE, Waseda). Res.Assoc. (1983,Waseda), Assist. Prof. (1986.Waseda), Assoc. Prof.(1988,Waseda), Prof.(1997-,Dept. EECE, Waseda).

Visiting Scholar (1985.Univ.California at Berkeley), Visiting Research Scholar(1989-1990. Center for Supercomputing R&D, Univ.of Illinois at Urbana-Champaign)

IFAC World Congress Young Author Prize (1987), IPSJ Sakai Memorial Special Award (1997)

＜Activities for Societies＞

IPSJ: Sig. Computer Architecture(Chair), Trans of IPSJ Editorial Board (HG Chair), Journal of IPSJ Editorial Board (HWG Chair), Representative Member.

IEEJ: Sig. Parallel Processing(Chair), IT committee(Secretary)

ACM: Int'l conf. on Supercomputing (Vice Program Chair&PC)

IEEE: Int'l Symp.on Parallel and Distributed Processing Program (PC), PARELEC

Other: PC of many other conferences on Supercomputing and Parallel Processing, ISHPC, ICPP, "Scientific Programming"(Editorial Advisory Board), etc.

<Activities for Governments>

MITI: Millennium Project "Advanced Parallelizing Compiler"(Project Leader)

IT Policy Proposal Forum (Architecture/HPC WG Chair)

Supercompiler committee (WG Chair), Petaflops machine committee, HECC committee,

STA: Earth Simulator project evaluation committee, CCSE 1st class invited researcher,

JAERI research accomplishment evaluation committee

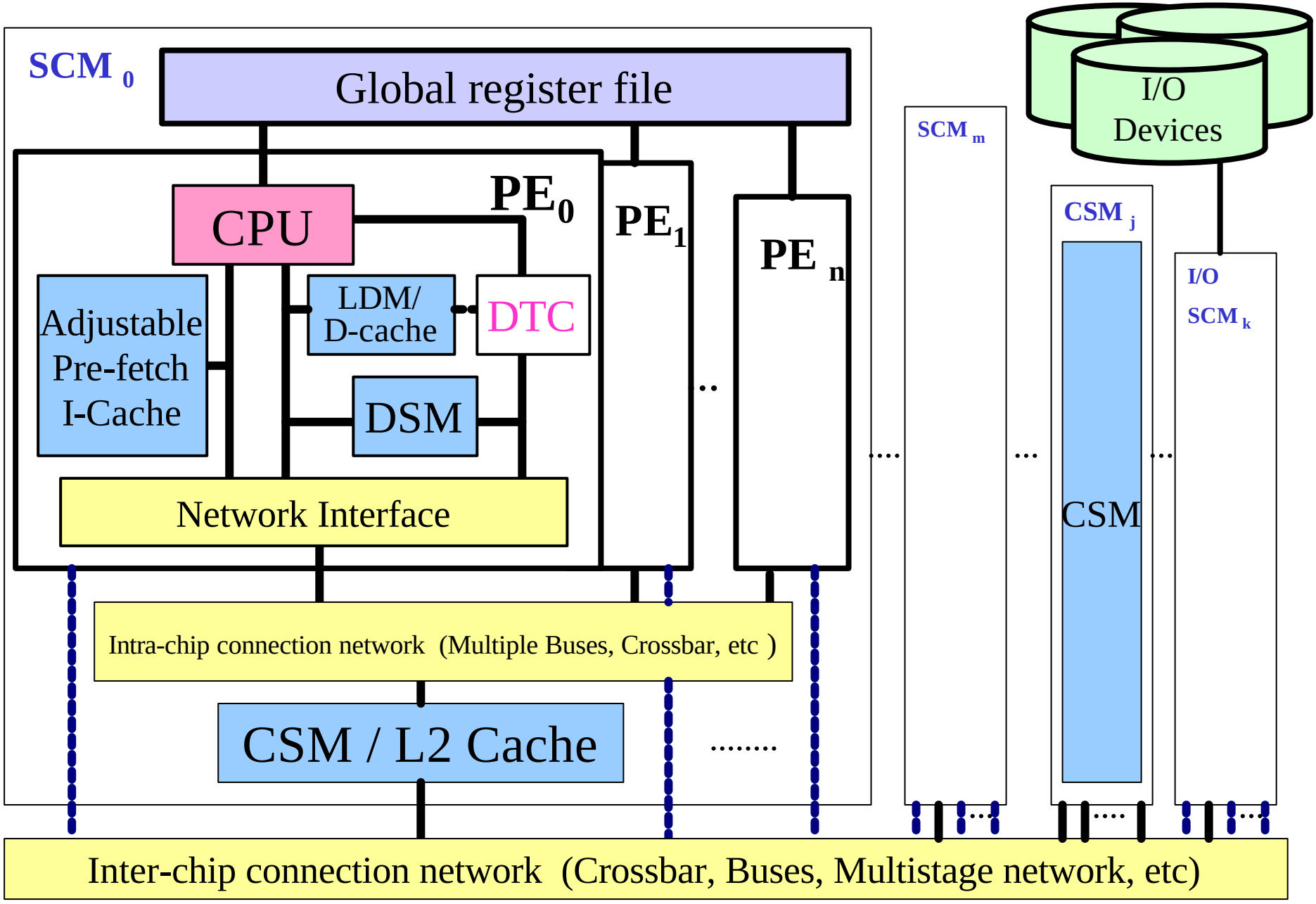
<Others>

Tokyo Electric Power Company Academic Evaluation committee, Japanese Representative for Conference on EU-Japan Co-operation in Education, Science and Technology: Round Table on Science and Technology, United Nations University.

Architecture

- **Architecture for Supercomputers, Microprocessors, Embedded processors for Games and Mobile Phones**
  - **Multigrain automatic parallelizing Compiler cooperative multiprocessor architecture, which gives us scalable performance improvement with progress of semiconductor technology and ease of use.**
  - **Single Chip Multiprocessor(SCM)**
    - **SCM with OSCAR type memory architecture to realize multigrain parallel processing inside a chip**
  - **High Performance Computing**
    - **HPC multiprocessor connecting SCM chips with OSCAR type memory architecture to improve effective performance, cost-performance and ease of use.**

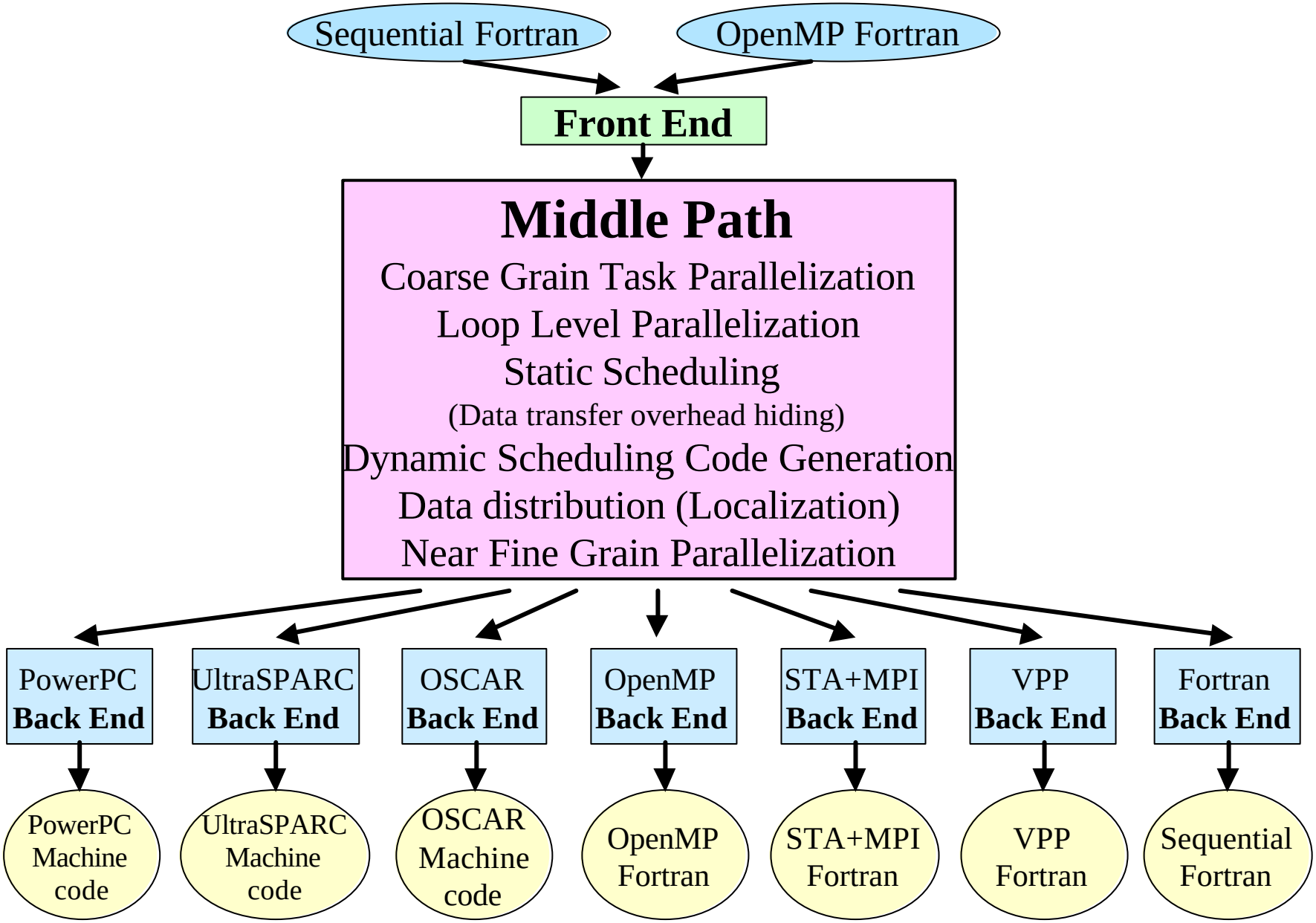
SCM Architecture for Multigrain Parallel Processing



Compilers

- **Automatic parallelizing technology of sequential programs to improve effective performance, cost-performance and usability**
  - **Multigrain Parallelization**
    - **Innovative parallel processing technology to automatically exploit parallelism from the whole program by use of coarse-grain parallelism among loops and subroutines, near fine grain parallelism among statements in addition to loop parallelism**
  - **Data Localization**
    - **Automatic data distribution for distributed shared memory, distributed cache and distributed memory on multiprocessor systems.**
  - **Data Preload and Poststore**
    - **Data transfer overhead hiding by overlap of task execution and inter processor data transfer and by instruction and data prefetching**

OSCAR Multigrain Fortran Compiler



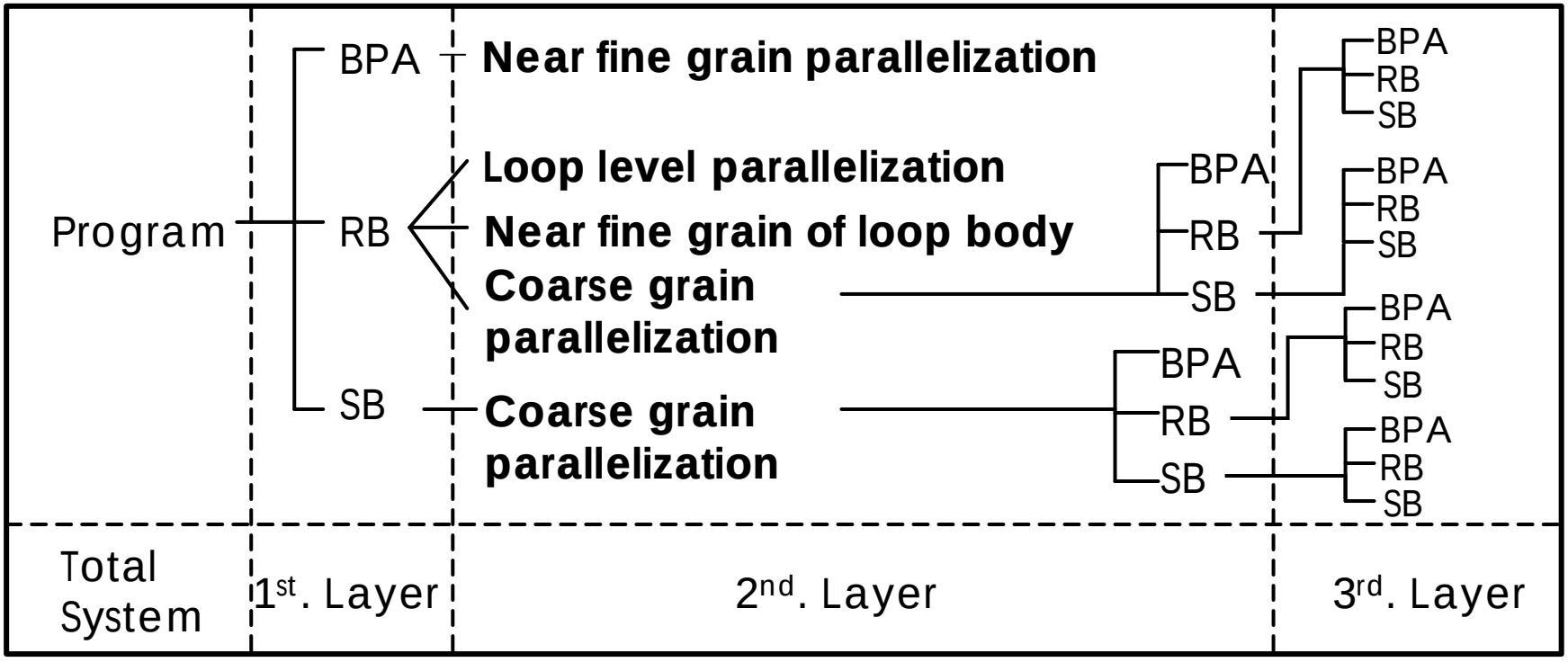
Multigrain Parallel Processing in OSCAR parallelizing compiler

- **Coarse grain task parallelism**
  - Among subroutines, loops & basic blocks
  - Statically or dynamically schedule to processors or PCs (Processor Clusters) at compile or run time
- **Loop parallelism**
  - Among loop iterations
  - Schedule to processors or PCs
- **(Near) Fine grain parallelism**
  - Among statements in a basic block
  - Statically schedule to processors in a PC
- **Their hierarchical combination**

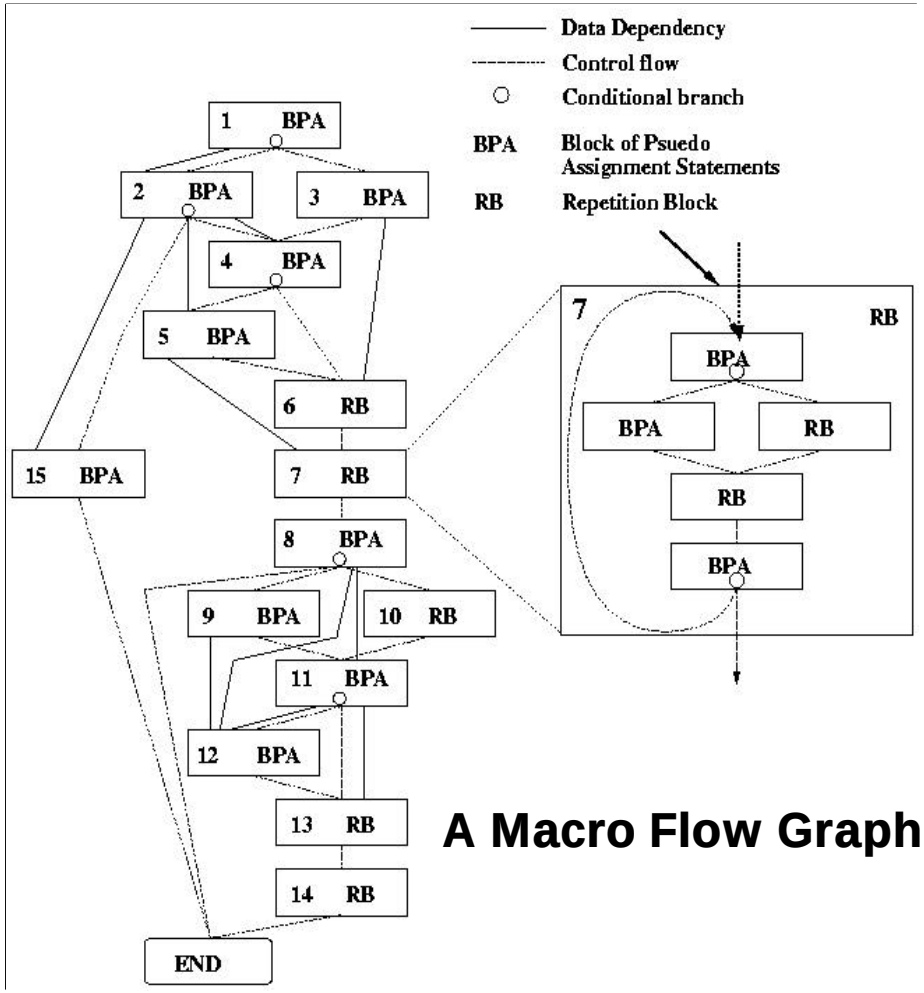
Generation of Coarse Grain Tasks

■Macro-tasks (MTs)

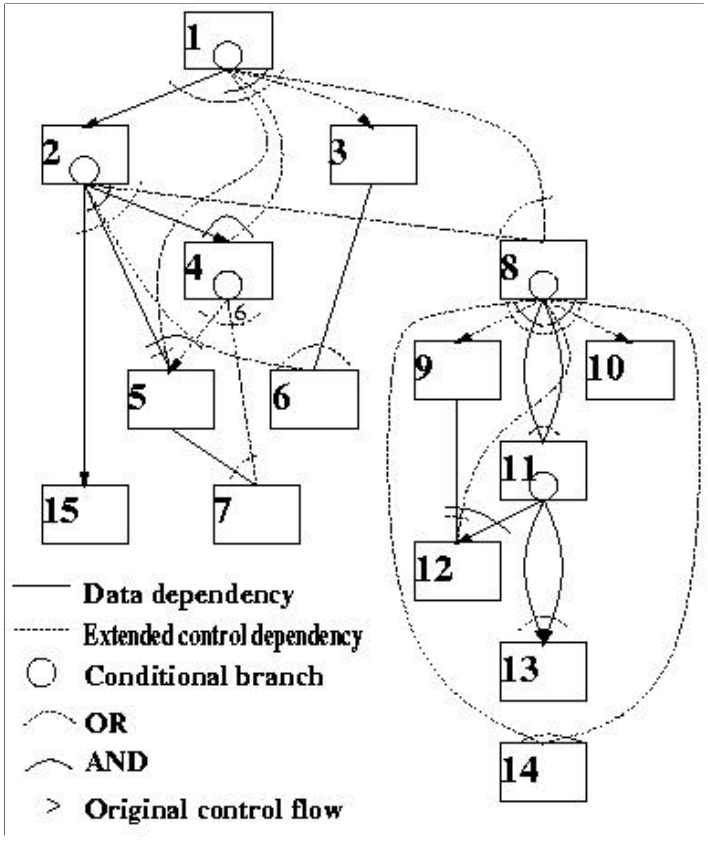
- ➢ Block of Pseudo Assignments (**BPA**): Basic Block (BB)
- ➢ Repetition Block (**RB**) : outermost natural loop
- ➢ Subroutine Block (**SB**): subroutine



Earliest Executable Condition Analysis for Coarse Grain Tasks (Macro-tasks)



A Macro Flow Graph



A Macro Task Graph

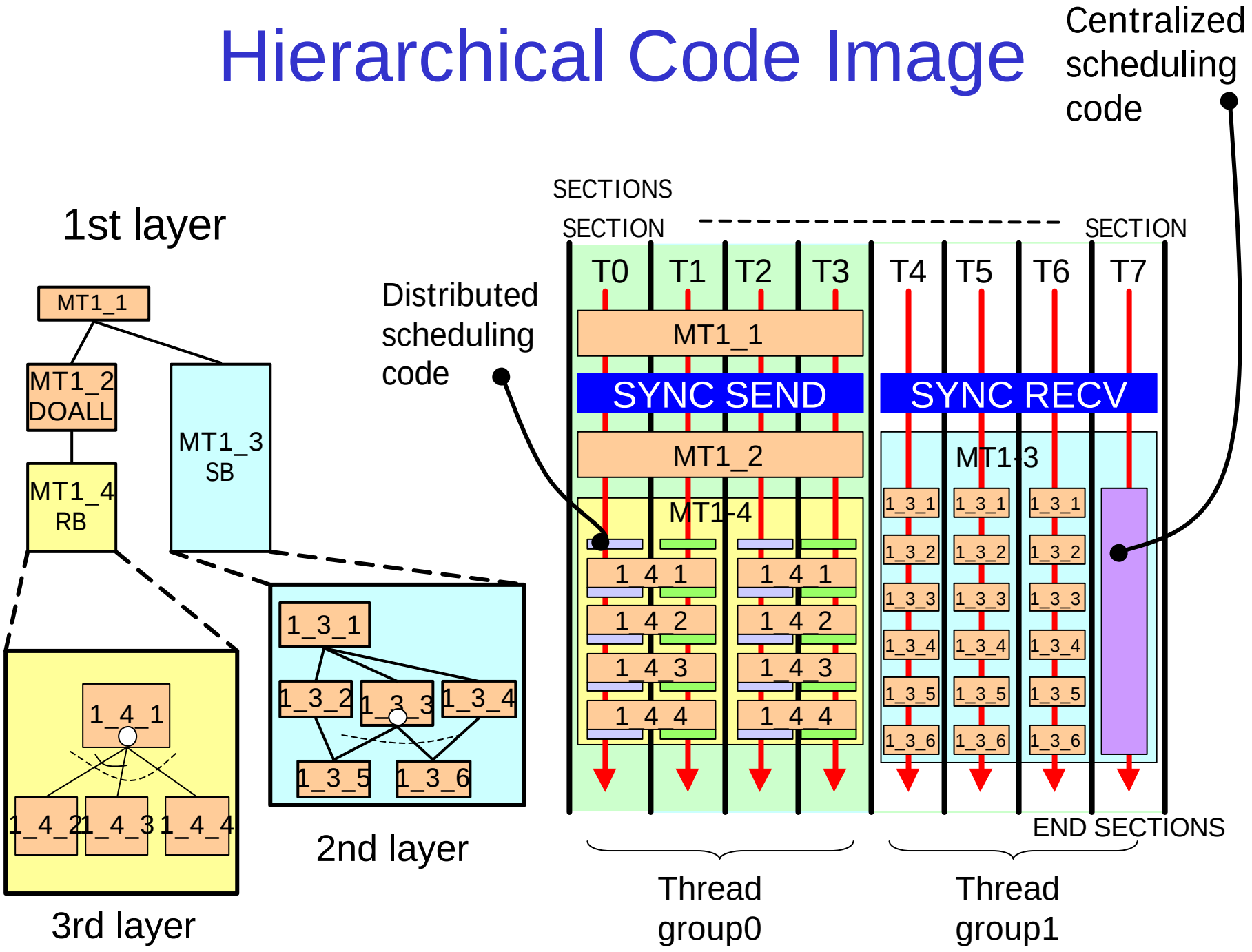
Code Generation Using OpenMP

- Compiler generates a parallelized program using OpenMP API
- One time single level thread generation
  - Threads are forked only once at the beginning of a program by OpenMP “PARALLEL SECTIONS” directive
  - Forked threads join only once at the end of program
- Compiler generates codes for each threads using static or dynamic scheduling schemes
- Extension of OpenMP for hierarchical processing is not required

Scheduling of Coarse Grain Tasks on MTG

- **Static Scheduling**
  - At compilation time, macro-tasks are assigned to processors statically.
  - Minimization of scheduling overhead and data transfer overhead
- **Centralized/Distributed Dynamic Scheduling**
  - Macro-tasks are assigned dynamically by the scheduling code exclusively generated for the MTG by compiler
  - Cope with runtime uncertainty like conditional branches
- **A suitable scheduling scheme is chosen for each layer of Macro-Task-Graph**

Hierarchical Code Image



Near-fine Grain Parallel Processing

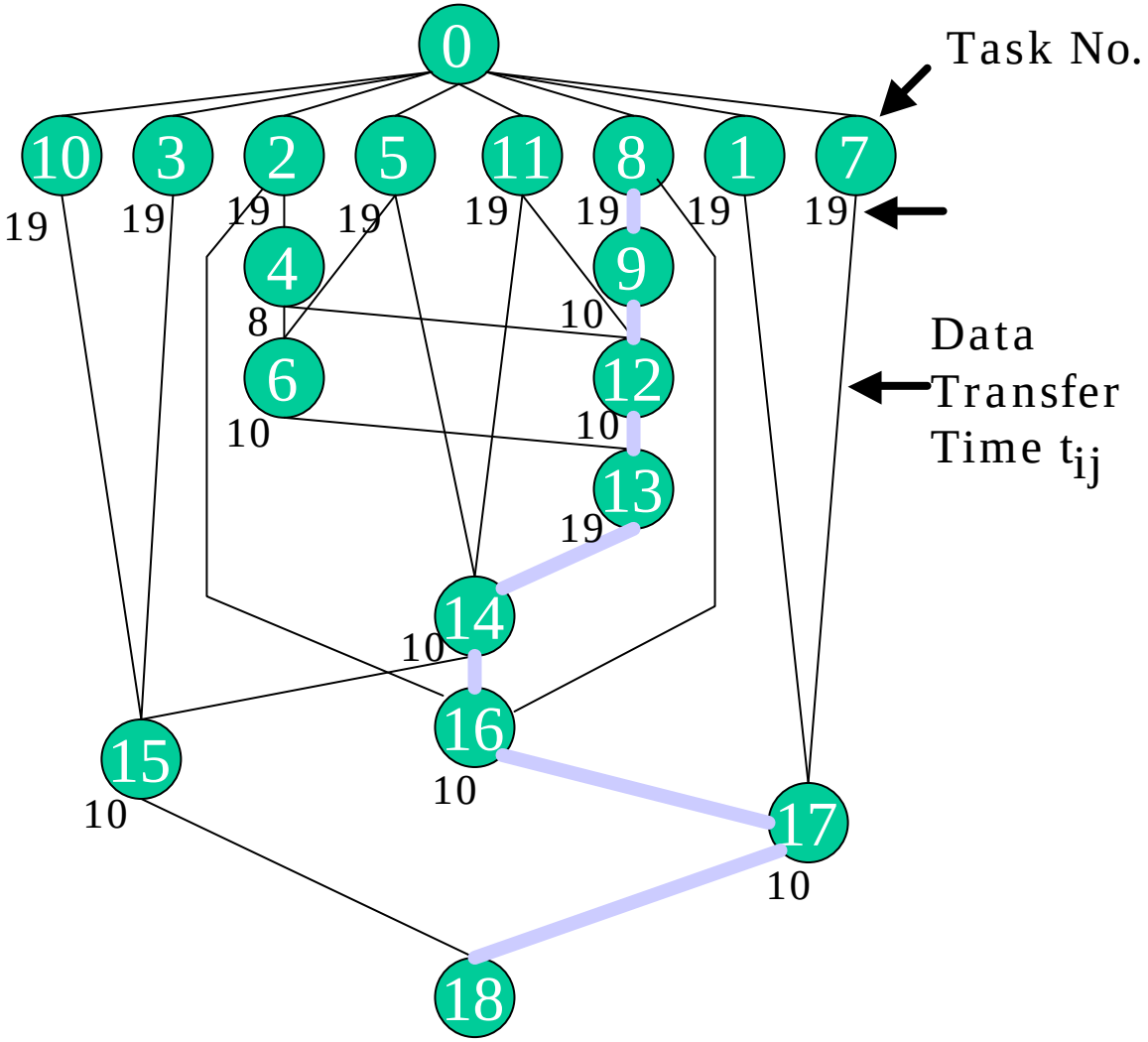
- Decompose a Basic Block into statement level near-fine grain tasks
- Analyze data dependencies among the tasks
- Generate a task graph
- Schedule tasks to processors using four heuristic scheduling algorithms and chose the best schedule
  CP/DT/MISF, DT/CP, CP/ETF/MISF, ETF/CP
- Generate optimized parallel machine code
  Elimination of redundant synchronization

Example of Near Fine Grain Tasks

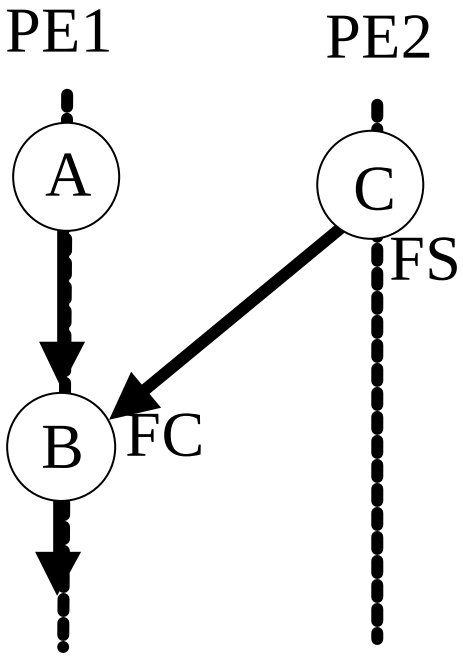
<<LU Decomposition>>
1) u12 = a12/l11
2) u24 = a24/l22
3) u34 = a34/l33
4) l54 = -l52 * u24
5) u45 = a45/l44
6) l55 = u55 - l54 * u45

<<Forward Substitution>>
7) y1 = b1 / l11
8) y2 = b2 / l22
9) b5 = b5 - l52 * y2
10) y3 = b3 / l33
11) y4 = b4 / l44
12) b5 = b5 - l54 * y4
13) y5 = b5 / l55

<<Backward Substitution>>
14) x4 = y4 - u45 * y5
15) x3 = y3 - u34 * x4
16) x2 = y2 - u24 * x4
17) x1 = y1 - u12 * x2



Generated parallel machine code for near fine grain parallel processing




```
        PE1
; ---- Task A ----
; Task Body

  FADD R23, R19, R21
; ---- Task B ----
; Flag Check : Task C
L18:
  LDR R28, [R14, 0]
  CMP R28, R0
  JNE  L18
; Data Receive
  LDR R29, [R14, 1]
; Task Body
  FMLT R24, R23, R29
```

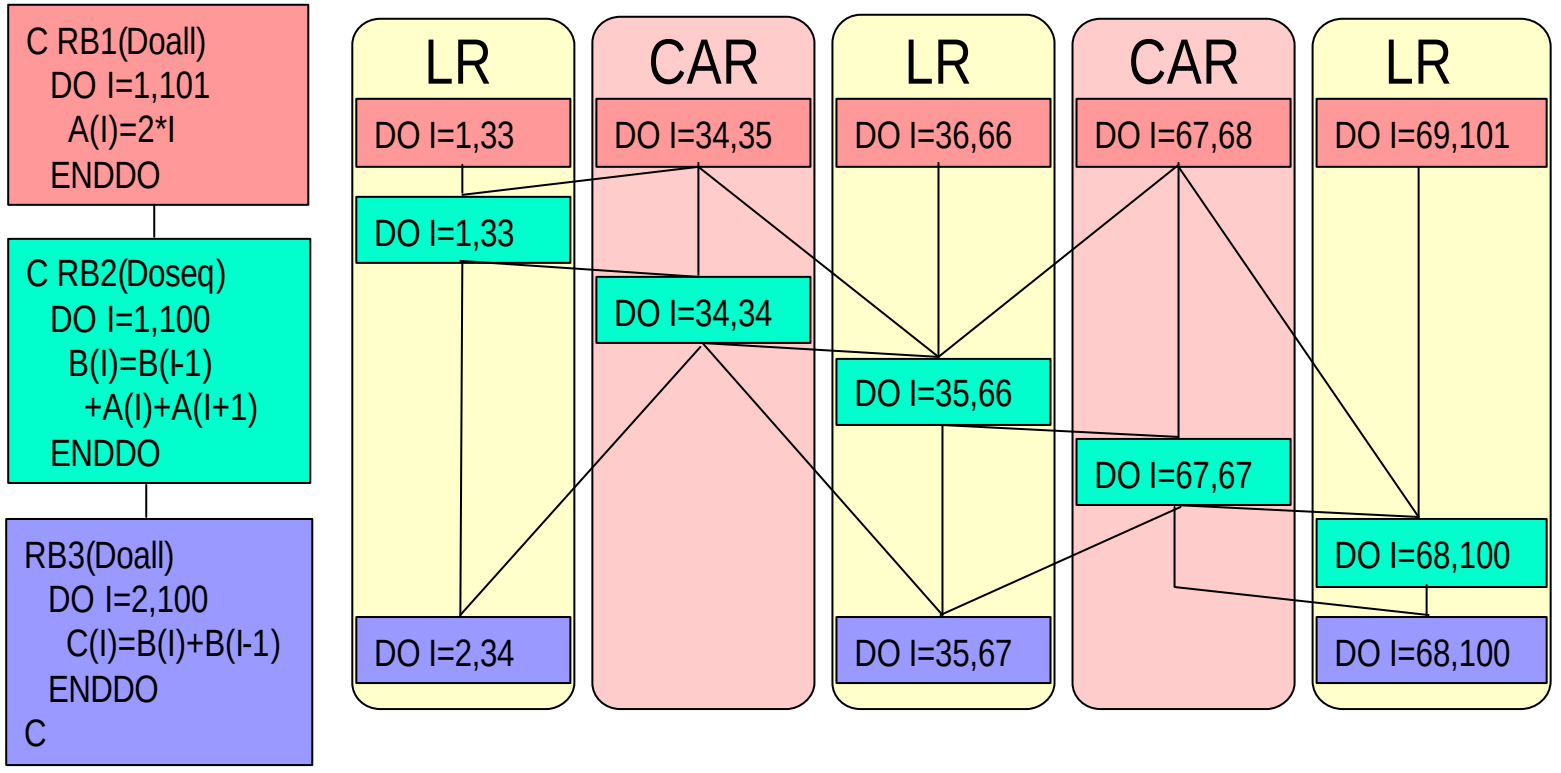
```
        PE2
; ---- Task C ----
; Task Body

  FMLT R27, R28, R29
  FSUB R29, R19, R27
; Data Transfer
  STR [R14, 1], R29
; Flag Set
  STR [R14, 0], R0
```

Data-Localization

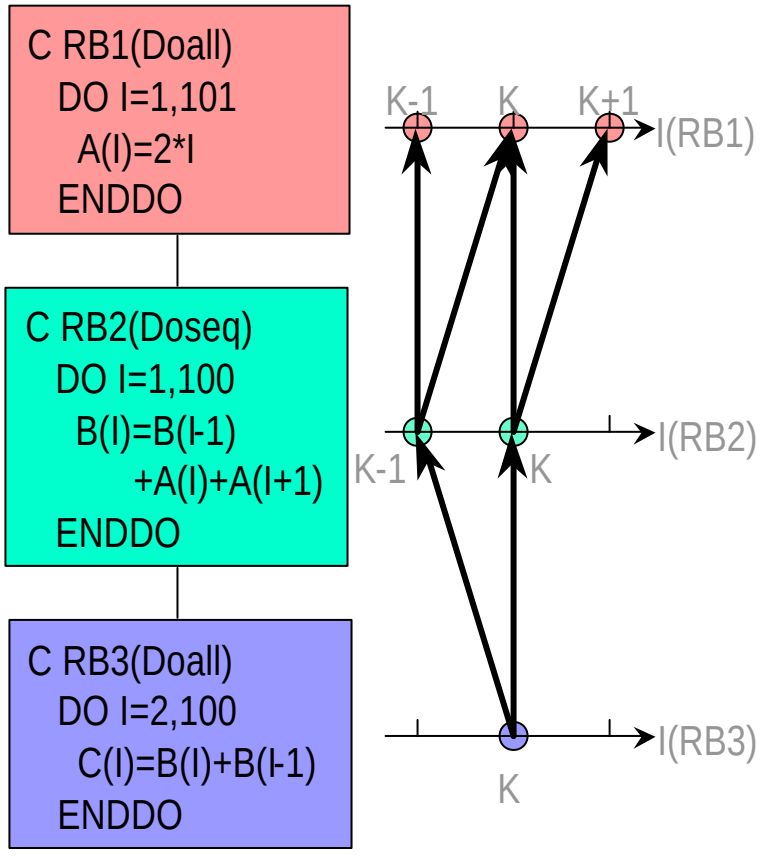
Loop Aligned Decomposition

- 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



Inter-loop data dependence analysis in TLG

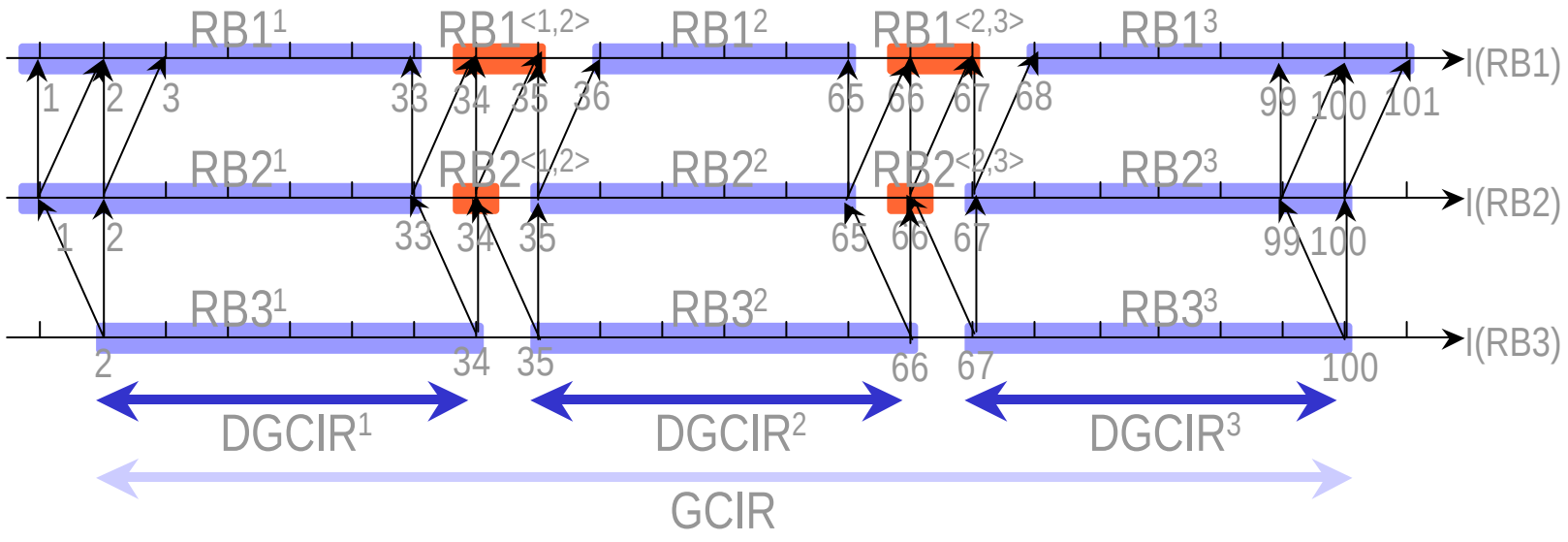
- Define exit-RB in TLG as Standard-Loop
- Find iterations on which a iteration of Standard-Loop is data dependent
  - e.g. K_{th} of RB3 is data-dep on $K\text{-}1_{th}$,K_{th} of RB2, on $K\text{-}1_{th}$,K_{th},$K\text{+}1_{th}$ of RB1



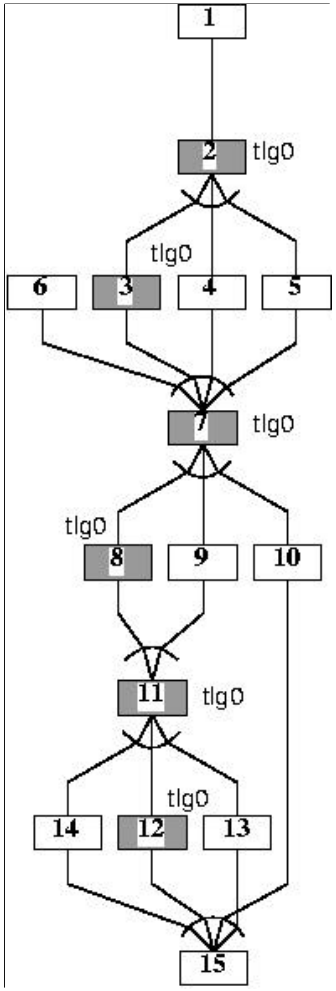
Example of TLG

Decomposition of RBs in TLG

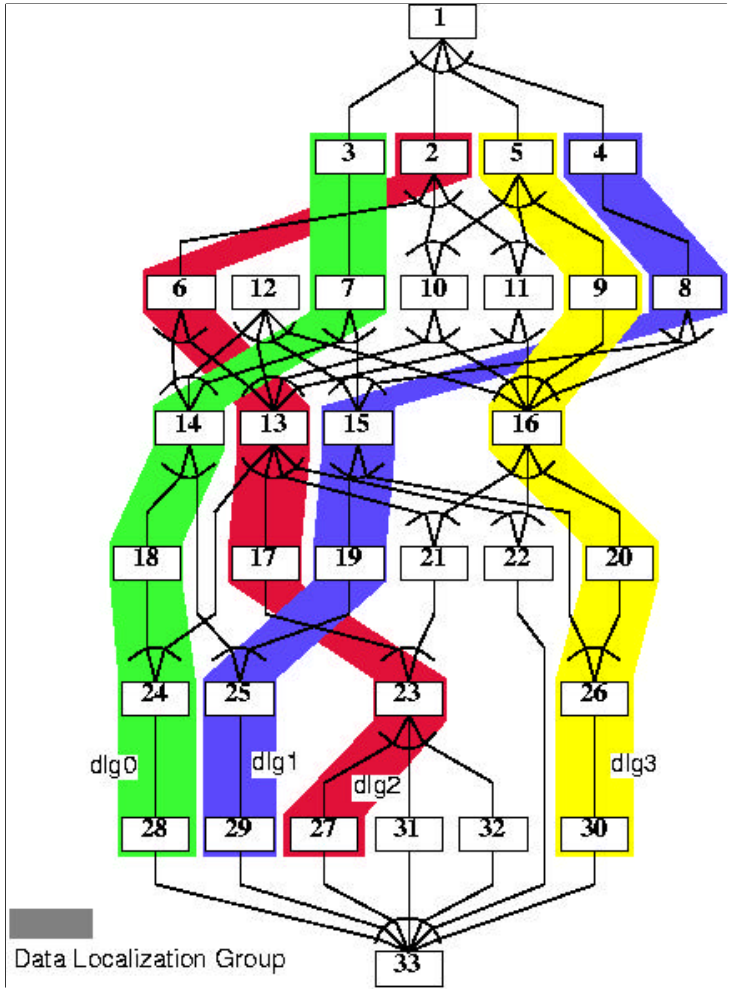
- Decompose GCIR into $DGCIR^p(1 \leqq p \leqq n)$
  - n: (multiple) num of PCs, DGCIR: Decomposed GCIR
- Generate CAR on which $DGCIR^p$&$DGCIR^{p+1}$ are data-dep.
- Generate LR on which $DGCIR^p$ is data-dep.



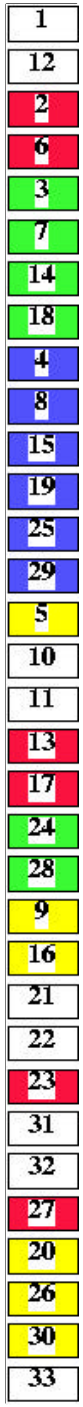
Data Localization Task Scheduling



Before Decomposition



After decomposition (into 4 groups)



←

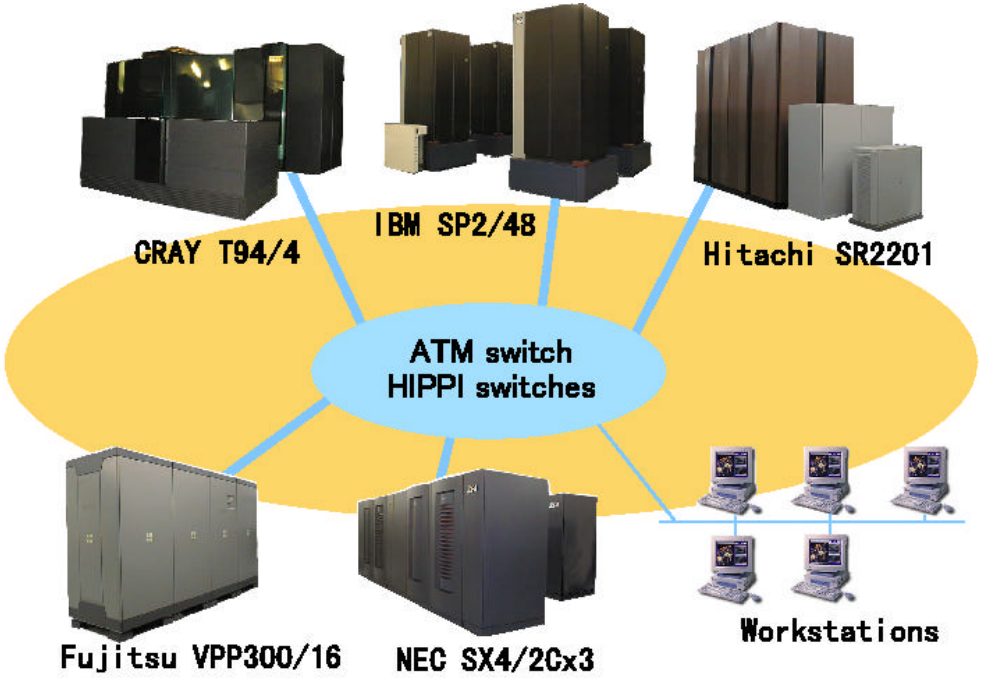
A task schedule onto a single processor

Scheduling

- **Development of practical algorithms for a NP hard intractable combinational optimization problem to assign tasks to processors**
  - **Practical minimum execution time multiprocessor scheduling algorithms**
    - **Practical optimization sequential algorithm DF/HIS and parallel algorithmPDF/HIS which can solve huge problems with thousand of tasks**
    - **Heuristic algorithms CP/DT/MISF, DT/CP, ETF/CP and so on**
  - **Performance Evaluation of Scheduling Algorithms**
    - **Benchmarking : Proposal of Standard Task graph Set**

      http://www.kasahara.elec.waseda.ac.jp/schedule/

Meta-scheduling



CCSE Supercomputer Cluster

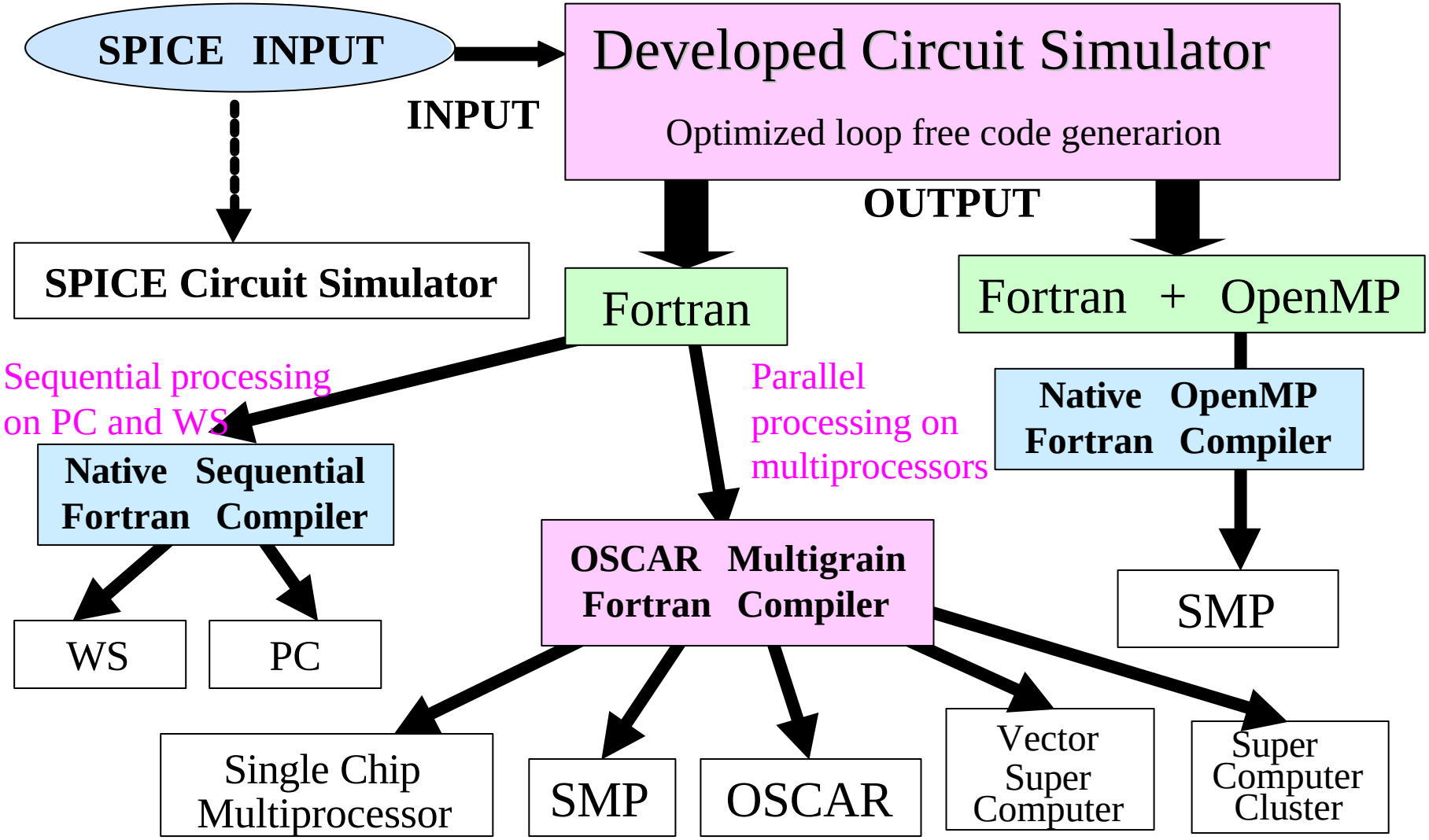
- **Global computing(GRID)**
- **Cluster computing**

Advanced automatic load distribution scheme based on OSCAR multigrain compiler for a heterogeneous network computing system

Applications

- **High speed parallel processing of scientific computation and performance evaluation technology of parallel hardware and software.**
  - **Electronic Circuit Simulation**
    - **Parallel processing of VLSI Circuit Simulation on PC/WS or multiprocessor systems**
  - **Power System Analysis**
    - **Parallel processing of load flow and transient stability analysis**
  - **Performance Evaluation Technology of Parallelising Compilers and SCM**
    - **Development of a fair evaluation method of automatic parallelizing compilers and single chip multiprocessors**

Parallel Processing of VLSI Electronic Circuit Simulation



Related Projects

- **National Project: METI Millennium Project IT21**

  (NEDO) Advanced Parallelizing Compiler

  Members : Waseda Univ., Fujitsu, Hitachi, AIST, Univ. of Electro-Communications, Toho Univ, Tokyo Institute of Technology, JIPDEC

- **STARC Project**

  (Semiconductor Technology Academic Research Center supported by 12 Domestic IT companies)

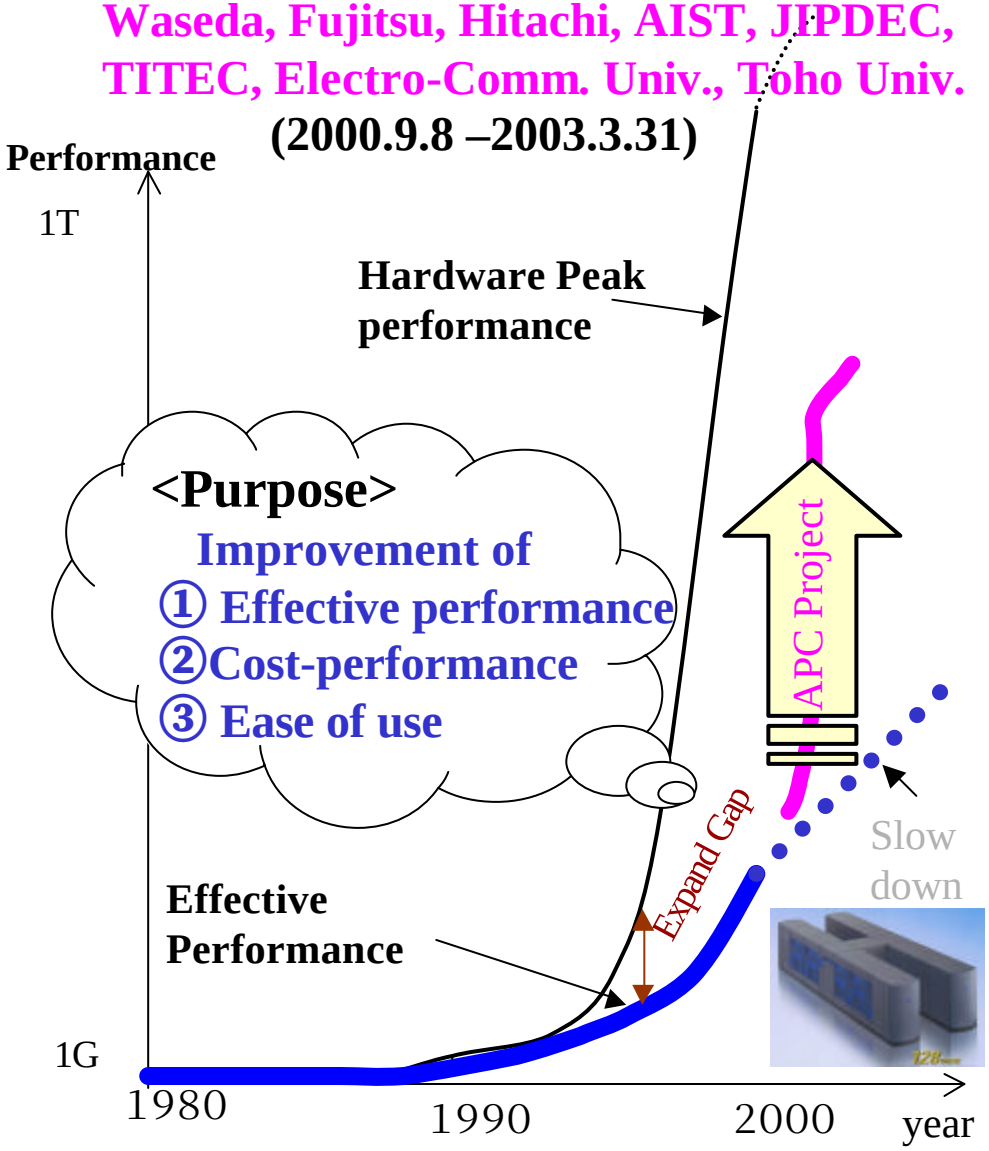
  Automatic Parallelizing cooperative Single Chip Multiprocessor Architecture

  Members : Fujitsu, Sony, Toshiba, Panasonic

Advanced Parallelizing Compiler Technology Project

Waseda, Fujitsu, Hitachi, AIST, JIPDEC, TITEC, Electro-Comm. Univ., Toho Univ.

(2000.9.8 –2003.3.31)



Theoretical maximum performance vs. Effective performance of HPC

Background and Problems

①Adoption of parallel processing as a core technology on PC to HPC

② Increase of importance of software on IT

③ Need for improvement of cost-performance and usability

Contents of Research and Development

① R & D of advanced parallelizing compiler
Multigrain, Data localization, Overhead hiding

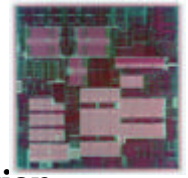
② R & D of Performance evaluation technology for parallelizing compilers

Goal: Double the effective performance

Ripple Effect

① Development of competitive next generation PC and HPC

② Putting the innovative automatic parallelizing compiler technology to practical use



③ Development and market acquisition of future single-chip multiprocessors

④ Boosting R&D in the following many fields:
IT, Bio-tech., Device, Earth environment, Next-generation VLSI design, Financial engineering, Weather forecast, New clean energy, Space development, Automobile, Electric Commerce, etc