Time-multiplexed Execution on the Dynamically Reconfigurable Processor
-A Performance/Cost Evaluation-

Yohei Hasegawa, Shohei Abe,

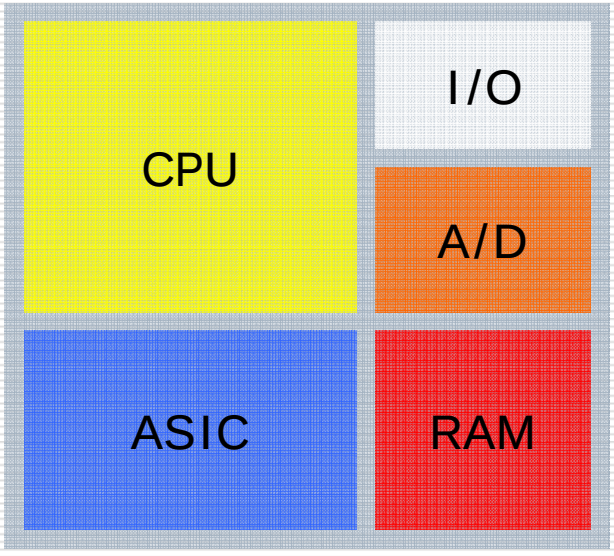
Katsuaki Deguchi, Masayasu Suzuki,

and Hideharu Amano

Graduate School of Science and Technology, Keio University, Japan

drp@am.ics.keio.ac.jp

Problem of System-on-a-Chip (SoC)

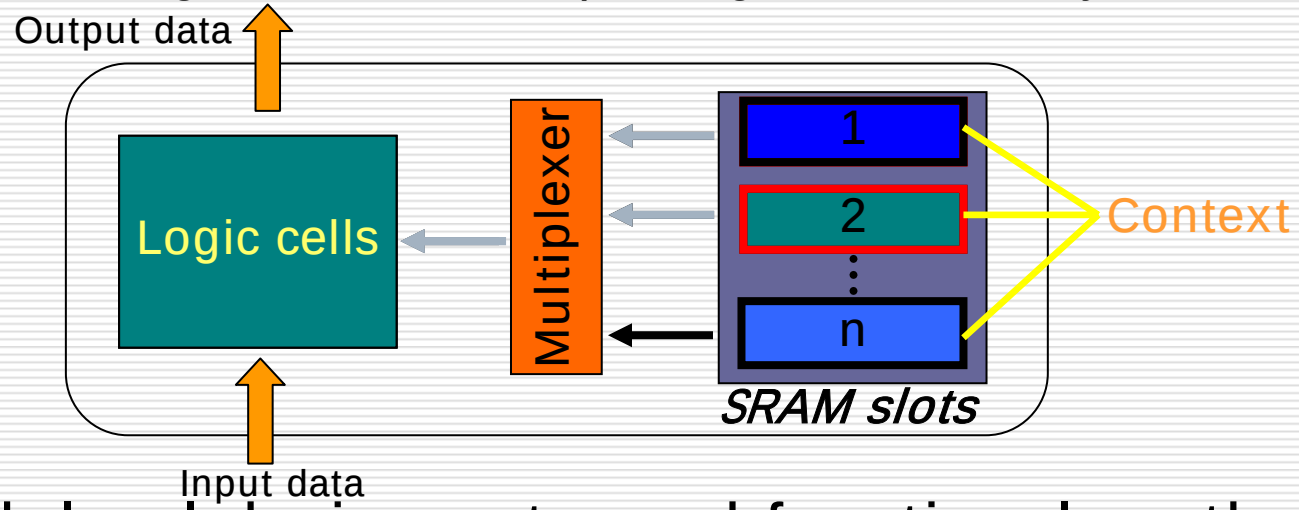


System-on-a-Chip

- LSI chips for various applications
  - Cellular Phones
  - Network Controllers
  - Mobile Terminals
- The performance is depending on Application Specific Hardware
- Various new techniques must be implemented (JPEG2000, AES, Turbo code...)
- Design cost for Application Specific Hardware becomes great!
- **Powerful but flexible structure is required!**

Dynamically Reconfigurable Processor

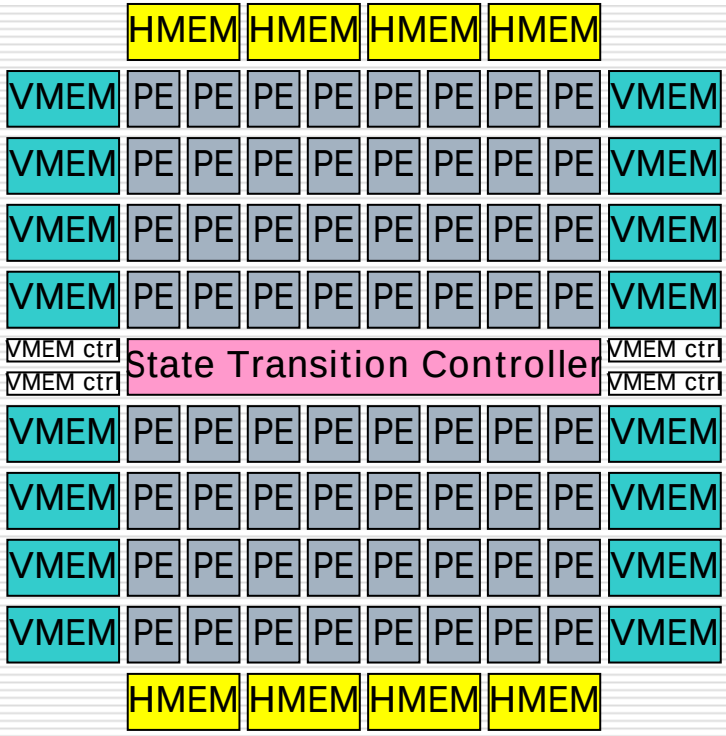
- Coarse grain cell architecture
- Dynamic reconfiguration
  - The hardware context is dynamically changed often in one clock = **Multi-Context Functionality**
  - Reducing the cost and improving area efficiency



- High level design entry and functional synthesis

NEC's DRP -Architecture Overview-

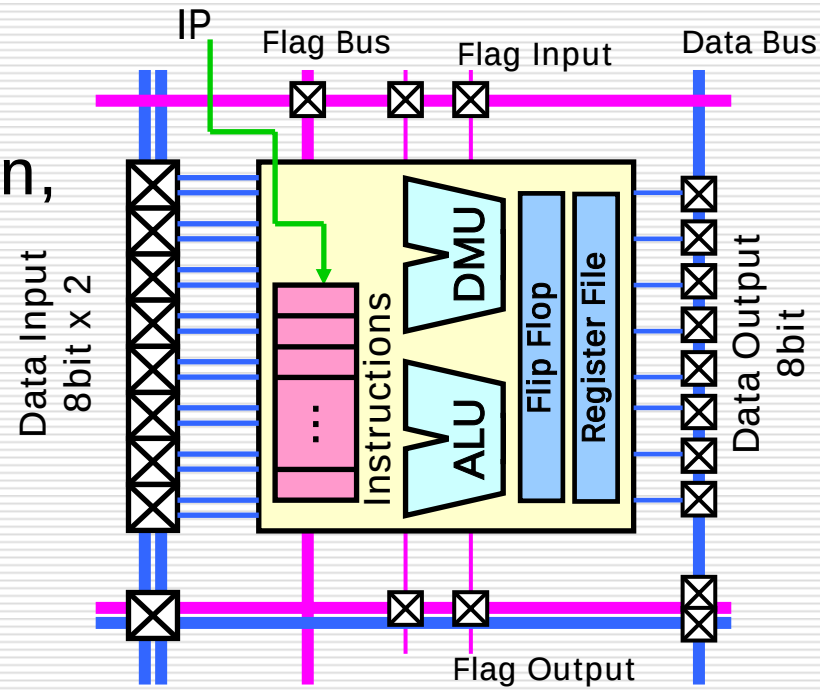
- Multi-Context Device
- Array of byte-oriented Processing Elements(PEs)
- Fully programmable inter-PE wiring resources
- A simple sequencer: State Transtion Controller (STC)
- Array of configurable data memories (VMEM, HMEM)



Tile = DRP Core

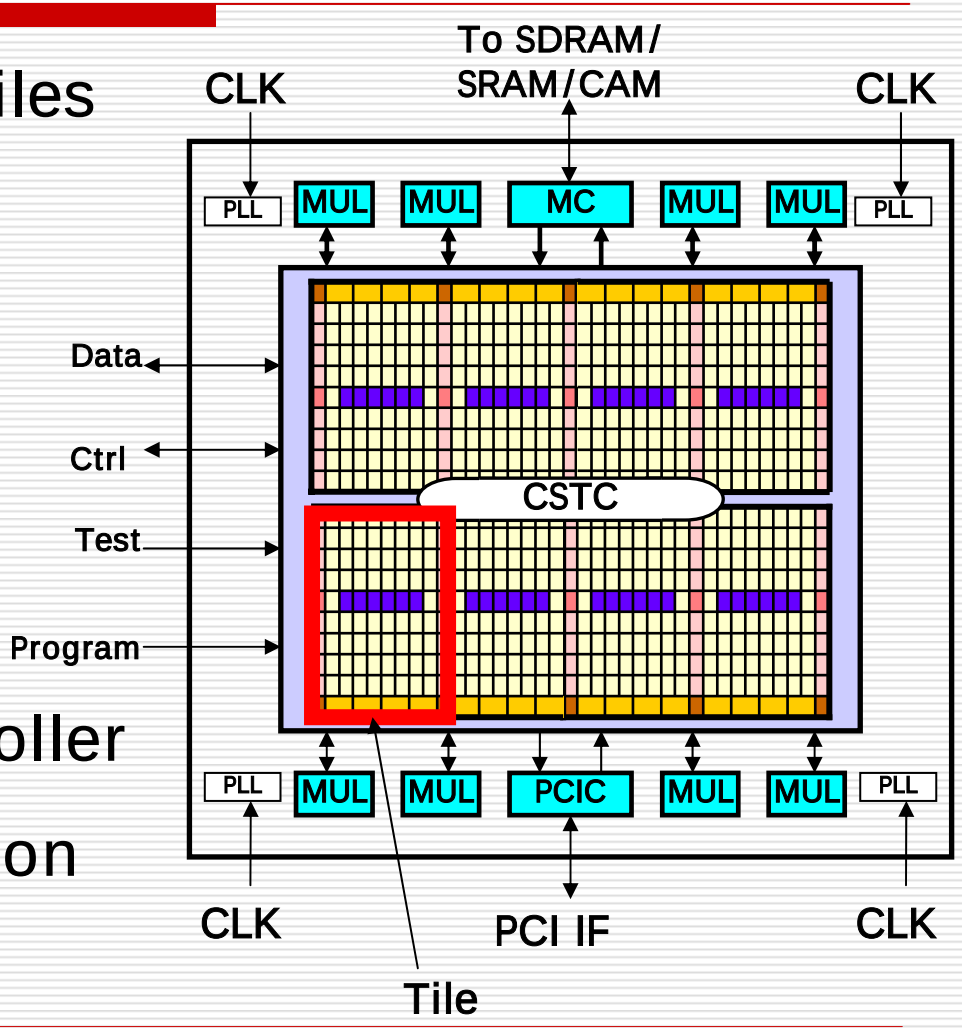
Processing Element (PE)

- ALU: arithmetic/logic operation
- DMU: byte select, shift, mask, constant generation, etc.
- Byte Flip-Flop/Register File
- An instruction dictates ALU/DMU operation and PE interconnection
- Instruction Pointer is provided from STC

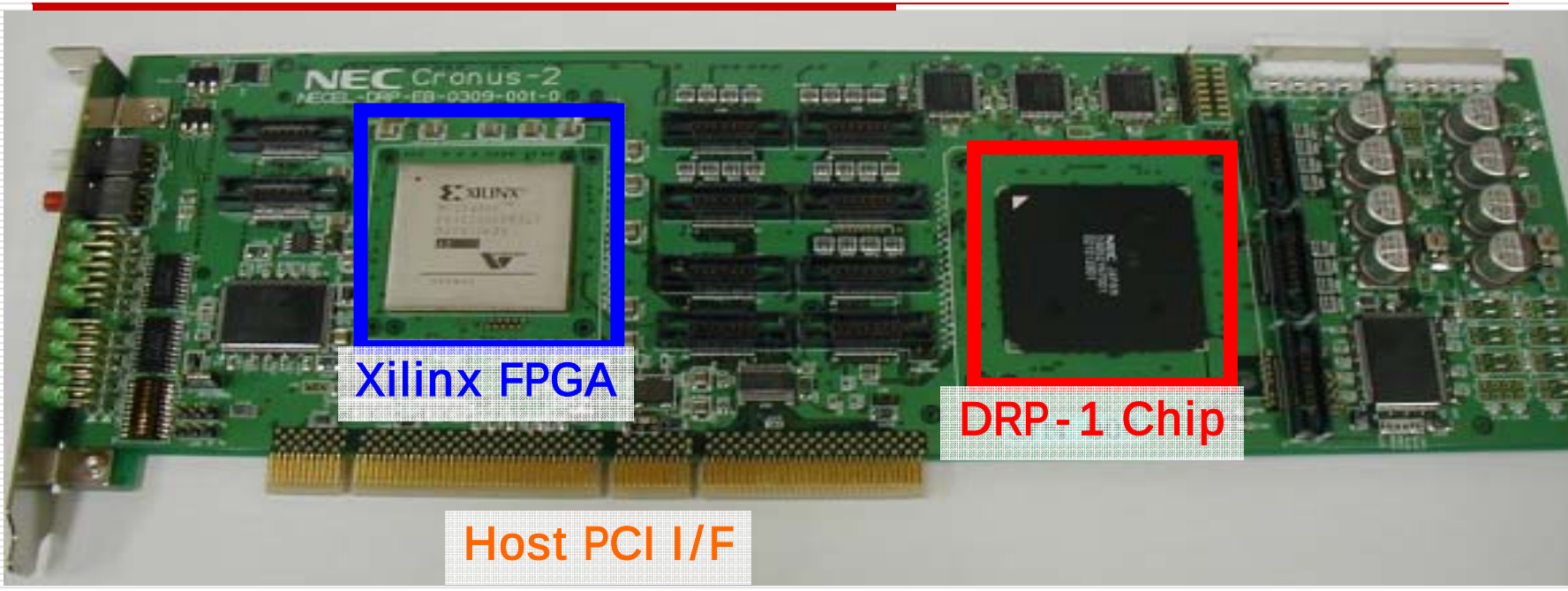


DRP-1: A Prototype Chip

- DRP Core with 4x2 Tiles (512PEs)
- VMEM x 80 (160Kbit)
- HMEM x 32 (2Mbit)
- 32bit Multiplier x 8
- PCI Interface
- External SRAM Controller
- Central State Transition Controller (CSTC)



DRP-1 Evaluation Board



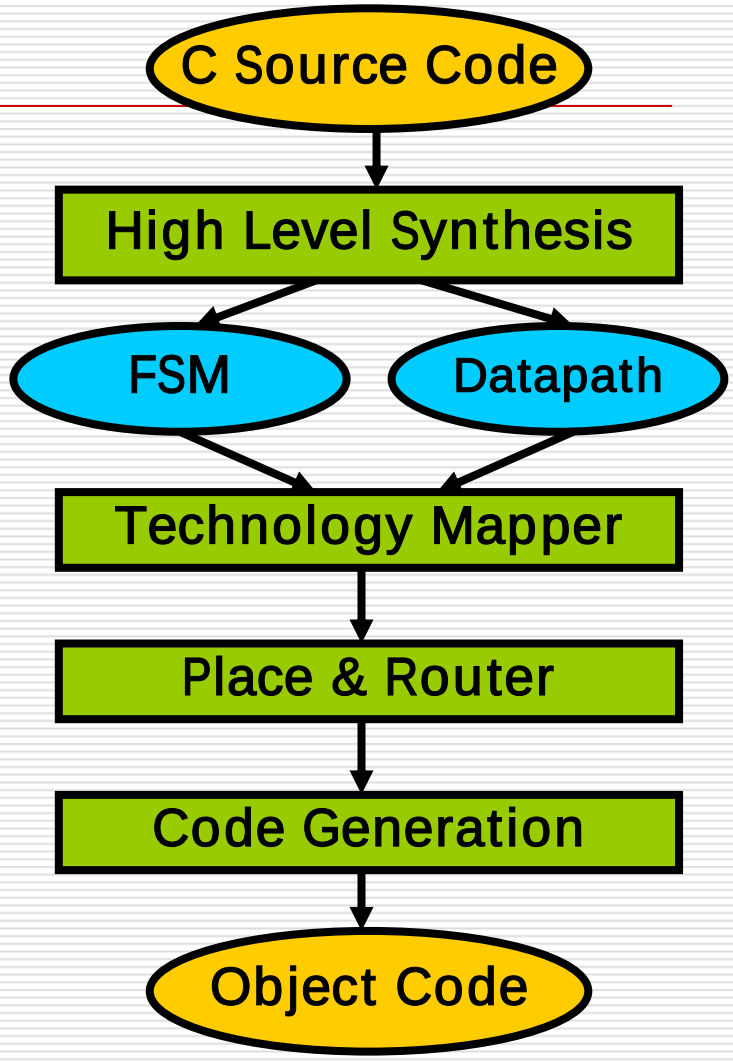
- One DRP-1 Core
- I/O Controller (Xilinx FPGA)
- External SRAM (4MByte)
- PCI Connection to Host PC via 64bit, 33MHz PCI Bus
- Local PCI Interface is used to configure DRP-1

DRP Compiler

- Compiling C source code into DRP object code

  ➔ Behavaioral Description Language (BDL)

- **High level synthesis**: generates finite state machines (FSMs) and associated datapath planes
  - The ASIC behavioral design tool: Cyber is modified and used.
- **Mapper**: maps FSMs and datapath plane to STC and PEs respectively
- **Place & Router**: physically locates the PEs and memories and mutually connects them



BDL Code Example

```
mem(0:16) d0[8], d1[8], d2[8], d3[8], d4[8], d5[8], d6[8], d7[8];
void row() {
  ter(0:16) SUMT0, SUMT1, SUMT2, SUMT3;
  reg(0:16) SUB0, SUB1, SUB2, SUB3;
  ter(0:16) z0, z1, z2, z3, z4, z5, z6, z7;
  reg(0:8)  i=0;
  $
  for(; i < 8; i++) {
    d0[i], d1[i], d2[i], d3[i], d4[i], d5[i], d6[i], d7[i];
    $
    SUMT0 = d0[i] + d7[i];  SUB0 = d0[i] - d7[i];
    SUMT1 = d1[i] + d6[i];  SUB1 = d1[i] - d6[i];
            . . . . .
    z0 = A * SUMT0 + A * SUMT1 + A * SUMT2 + A * SUMT3;
    z2 = B * SUMT0 + C * SUMT1 – C * SUMT2 – B * SUMT3;
            . . . . .
    $
    z1 = D * SUB0 + E * SUB1 * F * SUB2 + G * SUB3;
    z3 = E * SUB0 – G * SUB1 – D * SUB2 – F * SUB3;
            . . . . .
    $
}
```

16bit memory:
Allocated to VMEM

Terminals & Registers

Delimiter for the state/context

Memory Access for
giving an address

Terminals must be used In
the assigned state/context

Registers can be used in
the next states/contexts

Design Examples and Evaluation

Application	Number of Context	Max PEs	Frequency [MHz]	Throughput On DRP-1	Throughput On DSP
FFT	14	59	33	2,802 FFT/s	383 FFT/s
DCT	16	440	15MHz	173.5Mbit/s	66.0Mbit/s
IMDCT	16	222	36	329Mbit/s	267Mbit/s
DWT	15	210	13.2		
AES-CBC	6	129	56.9	364.0Mbit/s	16.9Mbit/s

- Target DSP is Texas Instruments TMS320C6713
  - 225MHz VLIW Floating Point DSP
  - 4KB data cache and instruction cache
  - 256KB L2 cache
  - Compiled with Code Composer StudioTM 2.20.05

Time-Multiplexed Execution

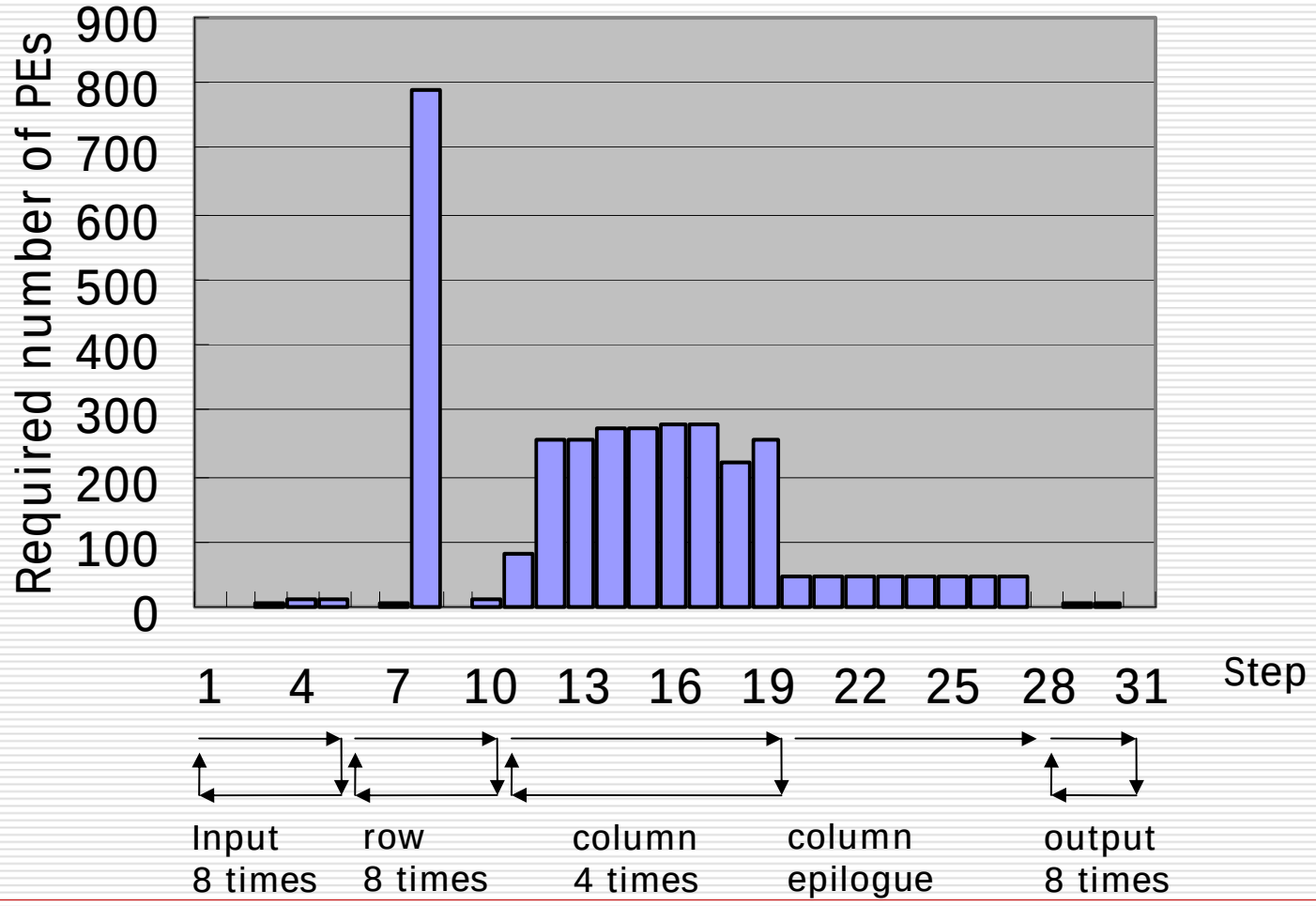
- A single task can be executed with multiple contexts, and can improve the area efficiency

However

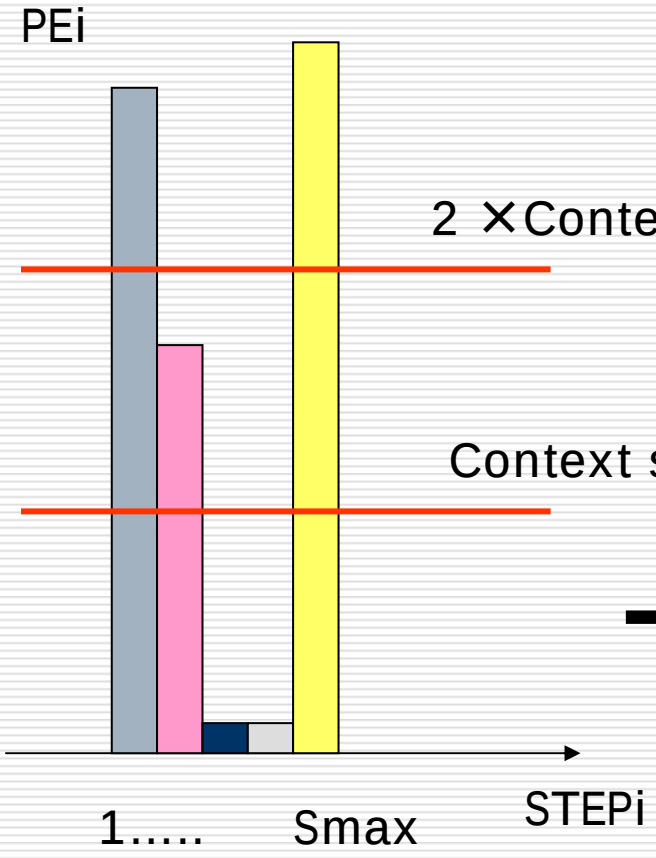
- No quantitative evaluation about the impact of performance and cost of time-multiplexed execution
  - ➔ We evaluate on some real stream applications on DRP-1

- Serial step in an algorithm:
  - Data is read out from distributed memory modules and/or registers
  - Required processing is done with multiple PEs
  - The results are stored into the distributed memory modules and/or registers
- **Parallelism Diagram** = Required number of PEs in each serial step

Parallelism on each step DCT in JPEG

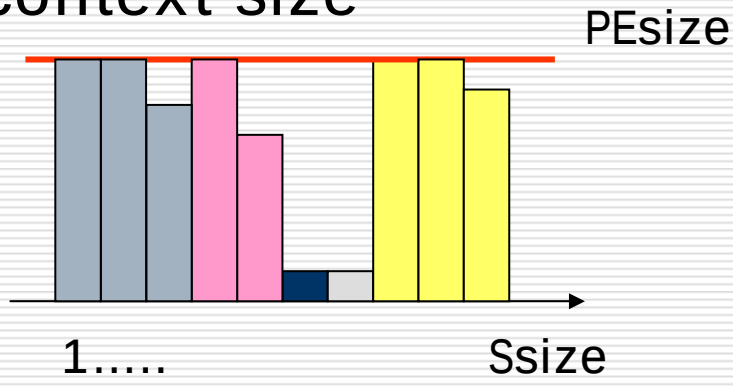


Context division



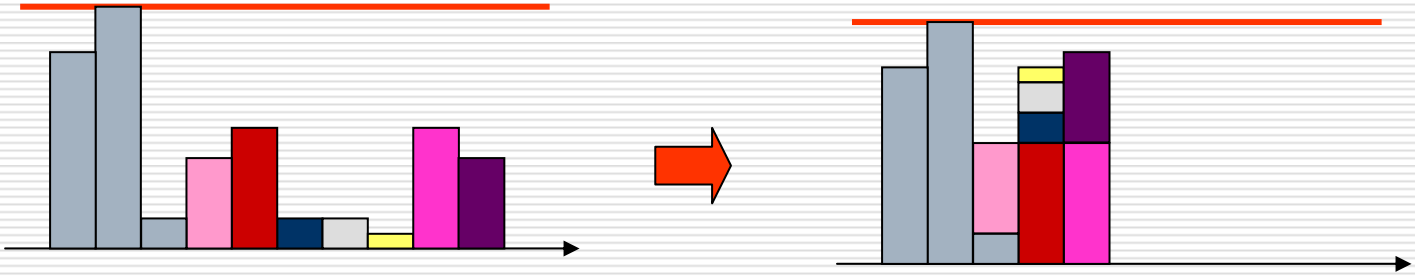
With the time-multiplexed execution,
the step is increased, but
not so much, since the required PEs are various in each step.

Time multiplexed execution with a certain context size



Analysis of Cost

- Multiple Step Allocation
  - Multiple steps are assigned in one context.
  - Frequency of context switch is reduced.
  - More PEs are used in a context.



- Required Number of Context
  - The number of context decreases if each context size increases.
  - Loss of the context division and integration becomes large because of imbalanced size and number of context

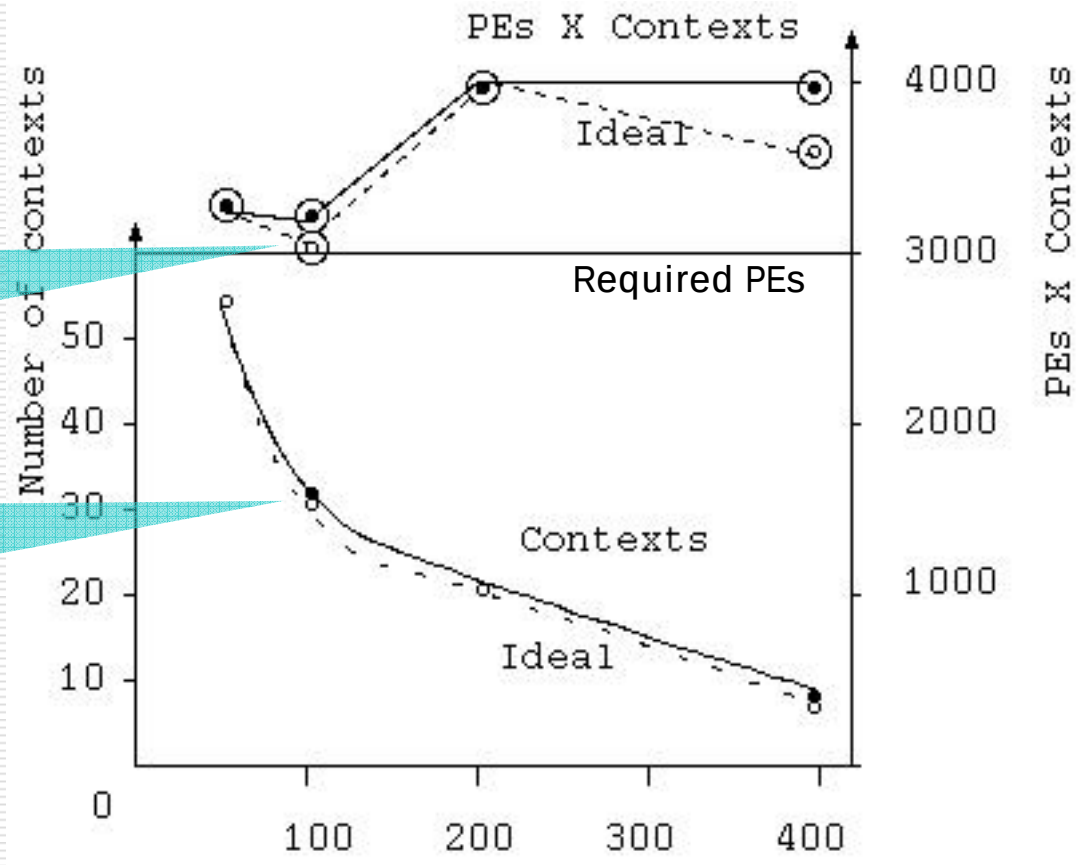
Analysis of Performance

- Applications:
  - DCT for JPEG, IMDCT for MP3, AES-CBC, DWT for JPEG2000
- Evaluation Summary:
  - The number of steps increases **20%-40%** when the context size becomes **1/2**.
  - The critical path is reduced **2%-5%** when the context size becomes **1/2**.
  - The performance/cost is improved **4.5-14** times with time-multiplexing execution.

Context Size vs the Number of Context

The overhead caused by context division is minimized

As the context size increases, the number of contexts decreases



Required Contexts vs. Context Size (DCT)

Summary

- Dynamically Reconfigurable Processor was introduced
  - Coarse Grain Structure → High Performance
  - Dynamic Reconfiguration → High Area Efficiency
- C-Level Programming Environment
- We evaluated the impact of time-multiplexed execution for some real applications on NEC's DRP-1

Demonstration: DRP Compile Tools

- **Musketeer**: An Integrated Development Environment for DRP