
A Small, Fast and Low-Power Register File by Bit-Partitioning

Masaaki Kondo, Hiroshi Nakamura

Research Center for Advanced Science and Technology

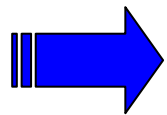
The University of Tokyo

Outline

- ❑ Background
- ❑ Concept of Bit-Partitioned Register File
- ❑ Microarchitectural implementation
- ❑ Evaluation
- ❑ Related work and conclusion

Background (register file)

- ❑ **register file (RF):** the highest level of memory hierarchy closest to the processor core
 - ❑ requires fast access time
 - ❑ power consuming part of the processor
- ❑ recent trend: enlarging the size and the number of ports
 - ❑ due to deeper pipeline, wider issue-width, SMT support etc.
- ❑ **problem:** increasing access time and power consumption of RF
 - ❑ limits the processor performance



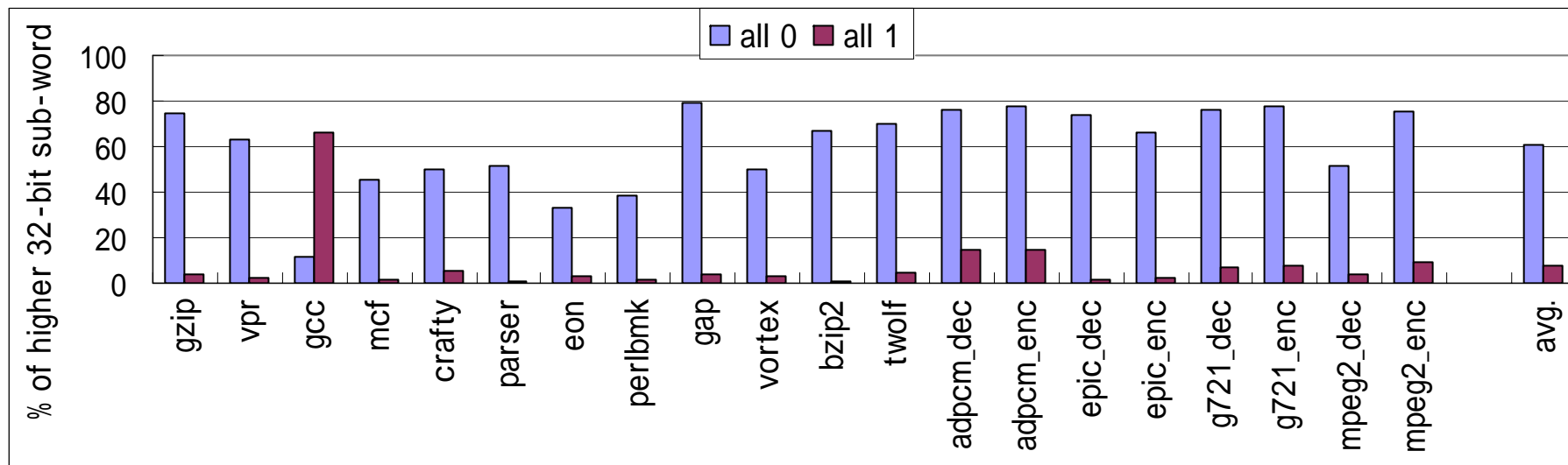
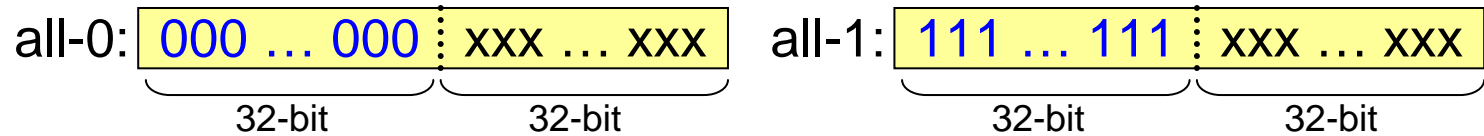
making good use of RF space is indispensable

Background (operand bit-width)

- ❑ 64-bit microprocessor
 - ❑ bit-width of each register entry is 64-bit
- ❑ narrow-width operations are still common
 - ❑ upper-bits of RF entries are ineffectively used
- ❑ **proposal:** making good use of RF by bit-partitioning
 - ❑ **Bit-Partitioned Register File (BPRF)**
 - ❑ to achieve high IPC even in small RF
 - ❑ for shorter access time and lower power consumption

Effective operand bit-width

- SPEC2000int and MediaBench on 64-bit Alpha
- % of operands whose upper 32-bits are all-0 or all-1



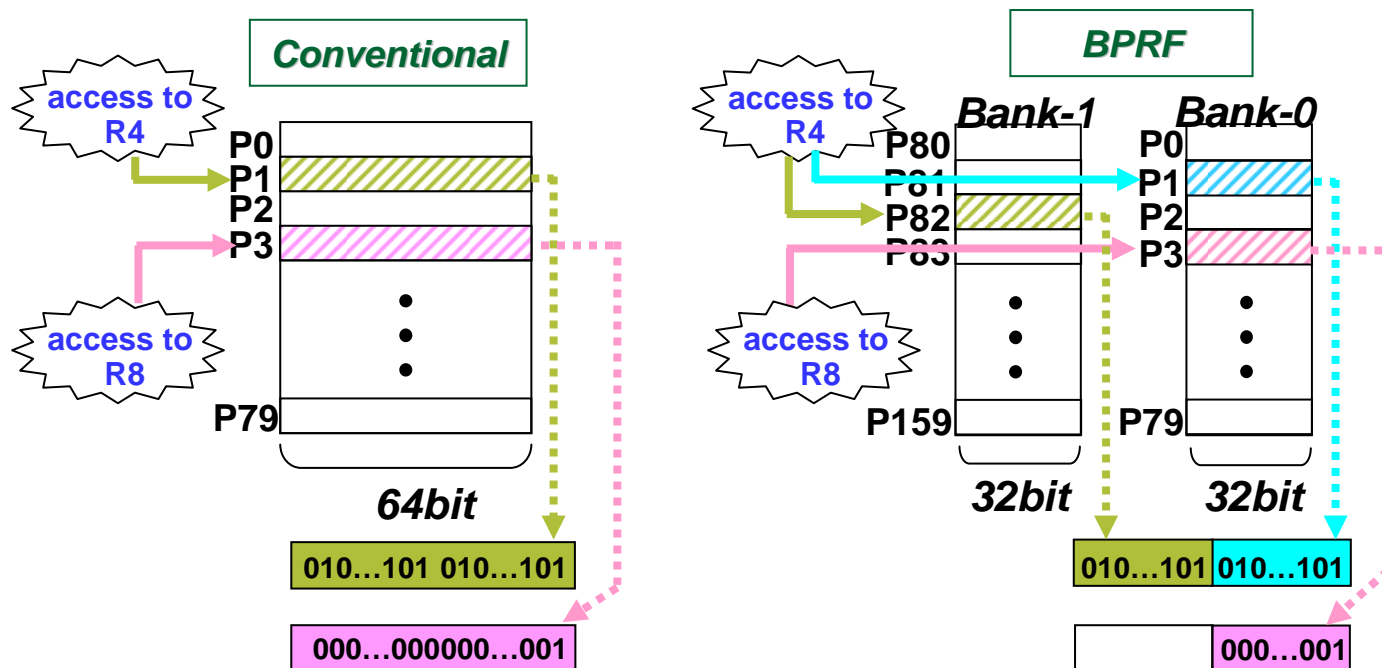
- about 70% of operands are less than or equal to 32-bit width
- all-0 sub-words are dominant
 - focus on the case of all-0 sub-word

Outline

- Background
- **Concept of Bit-Partitioned Register File**
- Microarchitectural implementation
- Evaluation
- Related work and conclusion

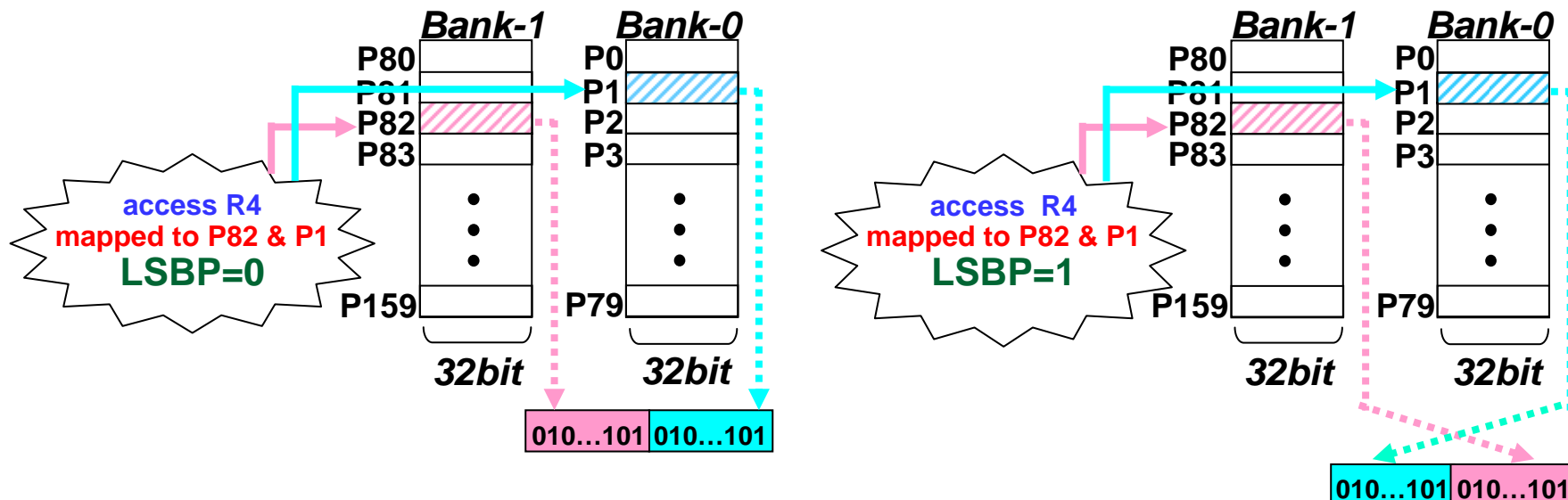
concept of BPRF

- partitioning the RF into multiple sub-word banks
 - not allocate upper bank for operand with small effective bit-width
- example: 80 entries X 64-bit, 2-partitioned
 - if effective bit-width ≤ 32 -bit: assign only one entry
 - if effective bit-width > 32 -bit: assign two entries



Least Significant Bank Pointer

- if lower sub-word are always assigned to Bank-0
 - the entries of Bank-0 tend to be exhausted
- introduce **Least Significant Bank Pointer (LSBP)**
 - indicates the bank which contains the least significant portion of the sub-word
 - bank assignment is changed operand by operand

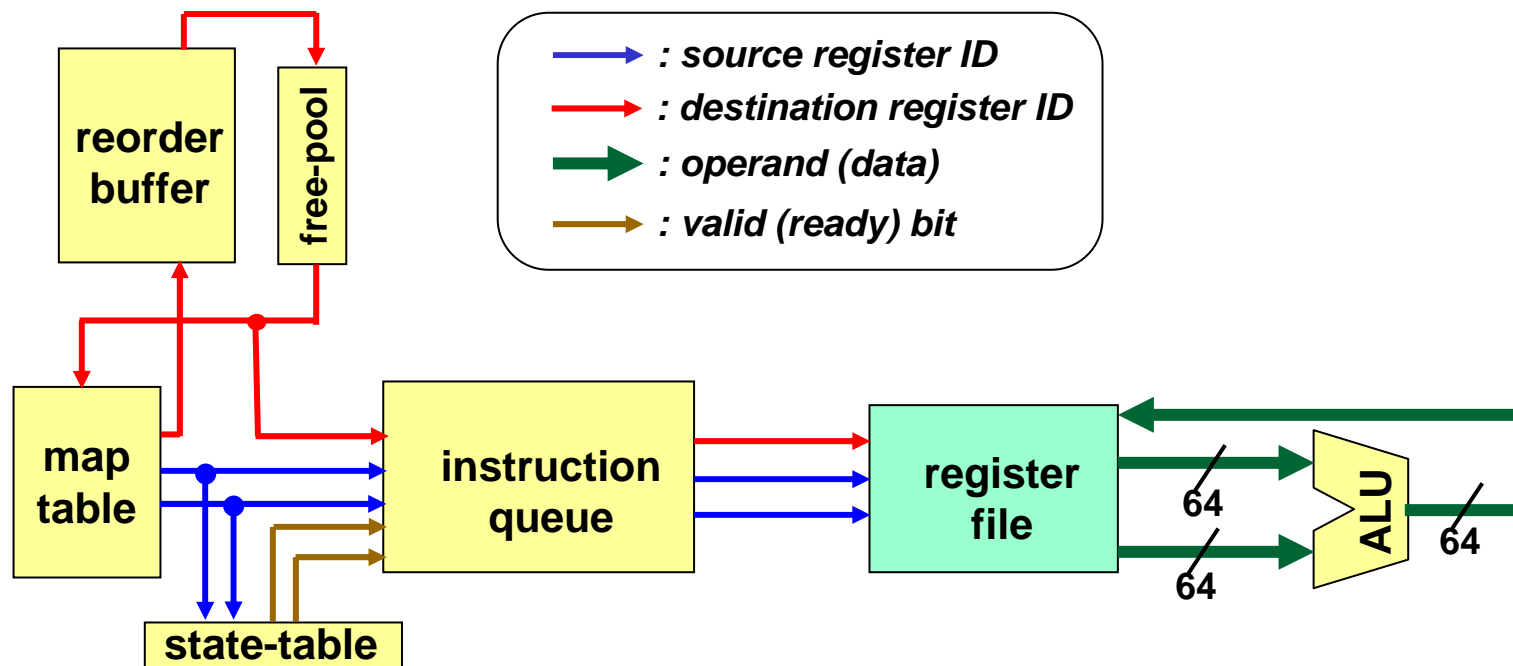
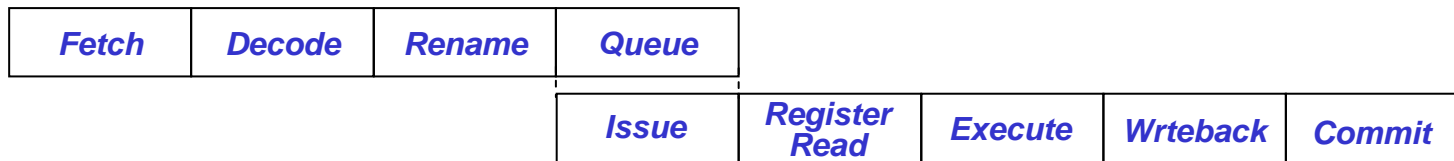


Outline

- Background
- Concept of Bit-Partitioned Register File
- **Microarchitectural implementation**
- Evaluation
- Related work and conclusion

A conventional processor

- pipeline and block diagram of a conventional processor

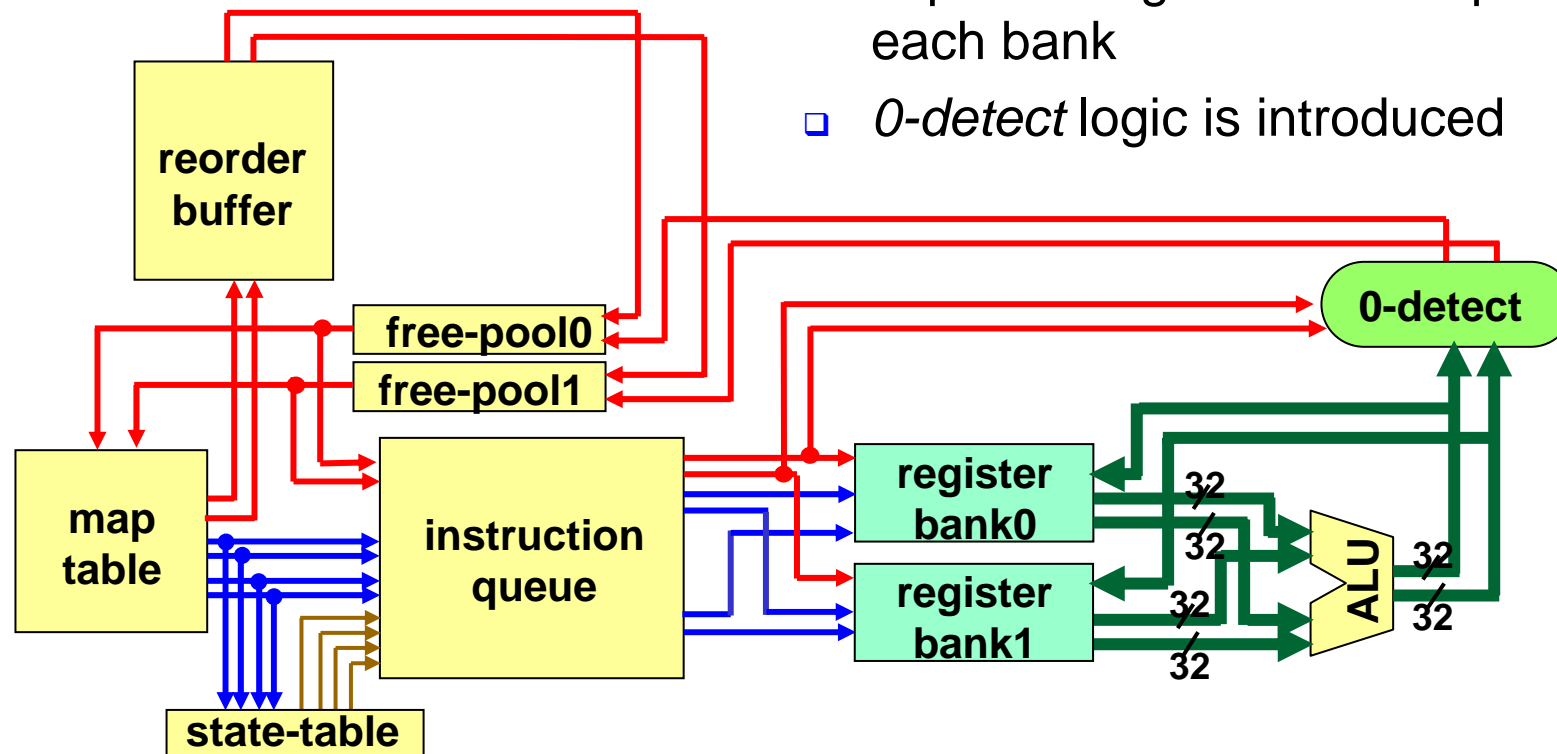


An overview of BPRF implementation

- the case for 2-partitioned BPRF

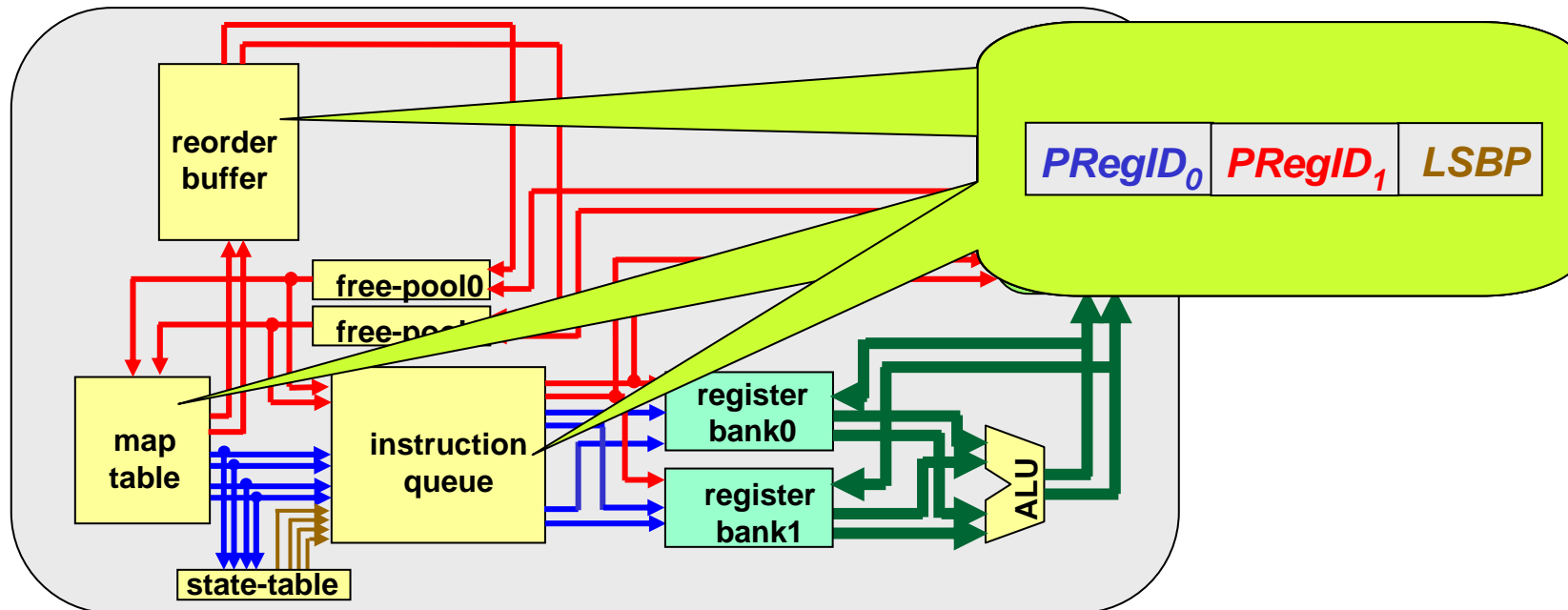
- main differences:

- RF is partitioned into two banks
- separate register ID/data paths for each bank
- *0-detect* logic is introduced



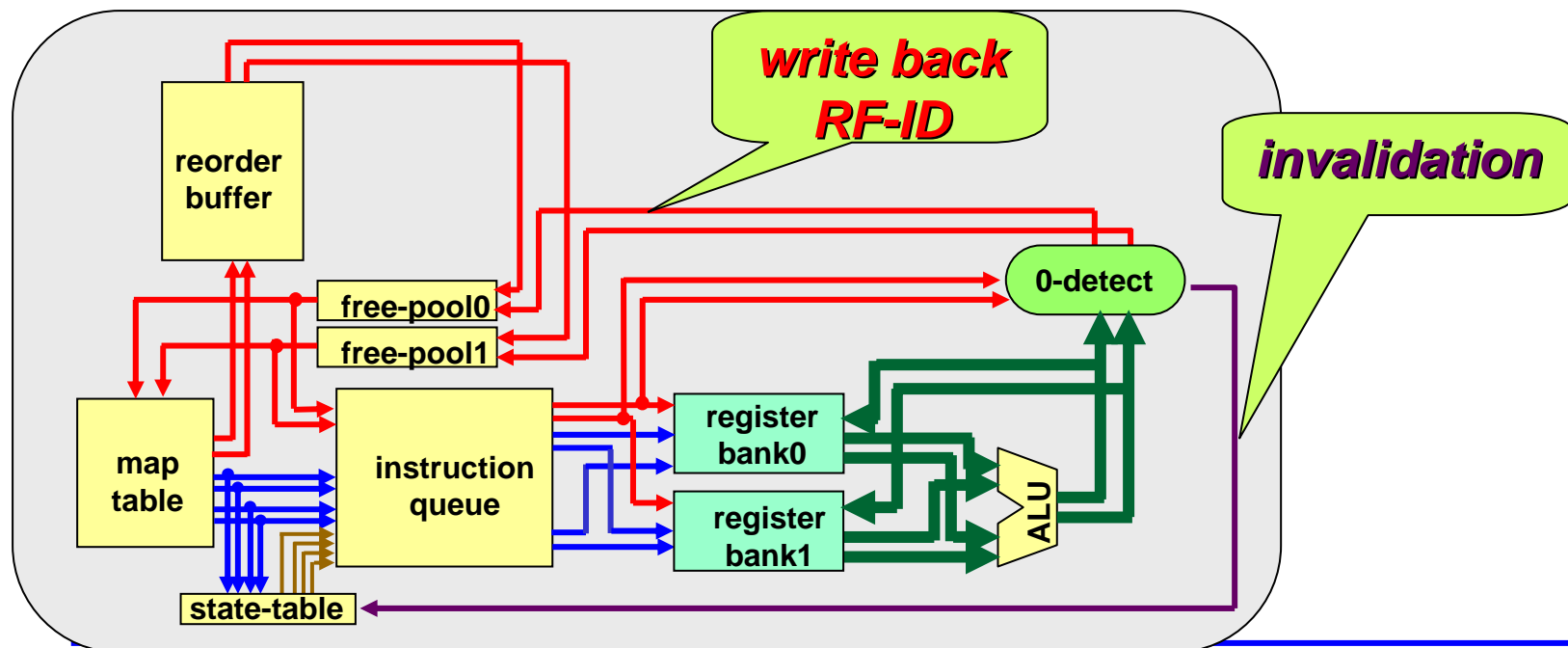
Extension for BPRF

- assigns two entries for each operand in rename stage
- free-pool is divided
- register-ID field in map-table, RB, and IQ is twofold



0-detect logic

- *0-detect*: checks effective bit-width of each ALU result
- when a narrow-width result detected
 - the RF entry is deallocated (*Early Register Deallocation*)
 - writing RF-ID to free-pool and invalidating ID in state-table
 - deallocated entries can be allocated for newly decoded instructions



Early Register Deallocation

- usual deallocation of a register entry
 - when a subsequent instruction that has the same architectural destination register is committed
- **Early Register Deallocation (ERD)**
 - deallocates register entry before the commit phase
 - deallocated entry can be used for another operand
 - increase in free available entries
- in BPRF, all-bits zero entry is deallocated by ERD
 - narrow-width operands are frequent
 - improves RF utilization

Outline

- Background
- Concept of Bit-Partitioned Register File
- Microarchitectural implementation
- Evaluation
- Related work and conclusion

Evaluation

- ❑ evaluation of performance, access time, and power
- ❑ evaluate three cases:
 - ❑ **normal**: conventional processor (64-bit)
 - ❑ **2-bank**: 2-partitioned BPRF (32-bit x 2)
 - ❑ **4-bank**: 4-partitioned BPRF (16-bit x 4)
- ❑ evaluation methodology
 - ❑ SimpleScalar3.0: for performance (IPC) evaluation
 - ❑ modified micro-architecture for simulating BPRF
 - ❑ Wattch: for power/energy evaluation
 - ❑ CACTI-3.2: for register access time evaluation

Compiler and benchmark program

- ❑ compiler
 - ❑ DEC C-compiler (alpha AXP instruction set architecture)
 - ❑ compile option: `-arch ev6 -non_shared -fast -O4`
- ❑ benchmark programs
 - ❑ SPEC2000Int
 - ❑ *ref* input set
 - ❑ fast-forwarded 1 billion and simulated 200-million instructions
 - ❑ MediaBench: adpcm, epic, g721, mpeg2 (encode, decode for each)
 - ❑ adpcm, epic, g721, mpeg2-decode: from beginning to completion
 - ❑ mpeg2-encode: fast-forwarded 1-billion and simulated 200-million instructions

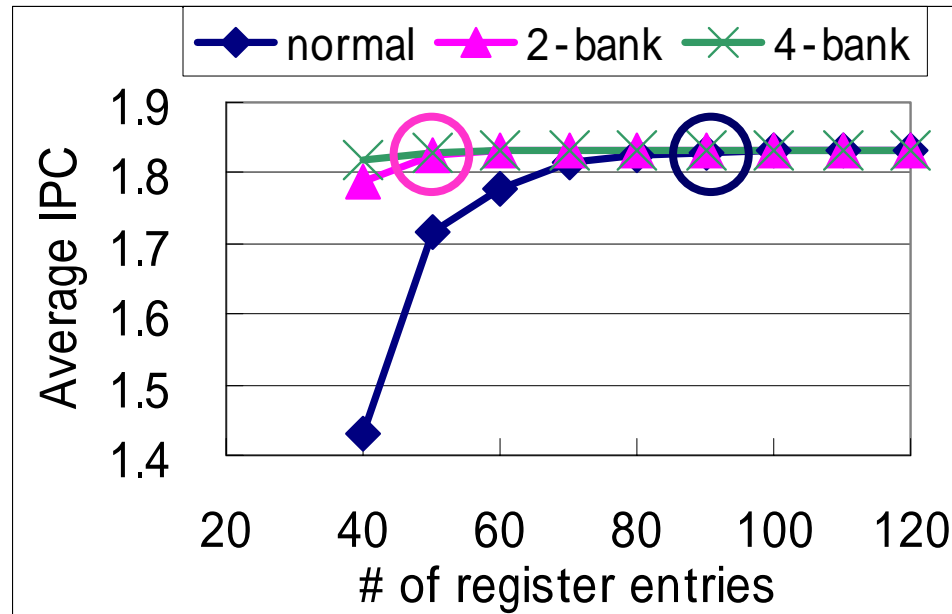
Assumptions on evaluation

- two configurations are evaluated
 - 4-issue
 - 8-issue
- RF size is varied in the evaluation

	4-issue	8-issue
Fetch & Decode width	4	8
Branch prediction	Combined bimodal (4K-entry) gshare (4K-entry), selector(4K-entry)	
BTB	1024 sets, 4-way	
Mis-prediction penalty	4 cycles	
Instruction queue size		
- integer	32	64
- load/store	32	64
- floating-point	32	64
Issue width		
- integer	4	8
- load/store	2	4
- floating-point	2	4
Reorder buffer size	96	192
Commit width	4	8
L1 I-cache	32 KB, 32 B line, 2-way 1-cycle latency	
L1 D-cache	32 KB, 32 B line, 2-way 2-cycle latency	
L2 unified cache	512 KB, 64 B line, 8-way 10-cycle latency	
Memory latency	100 cycles	
Bus width	16 B	
Bus clock	1/4 of processor core	

Average IPC (4-issue configuration)

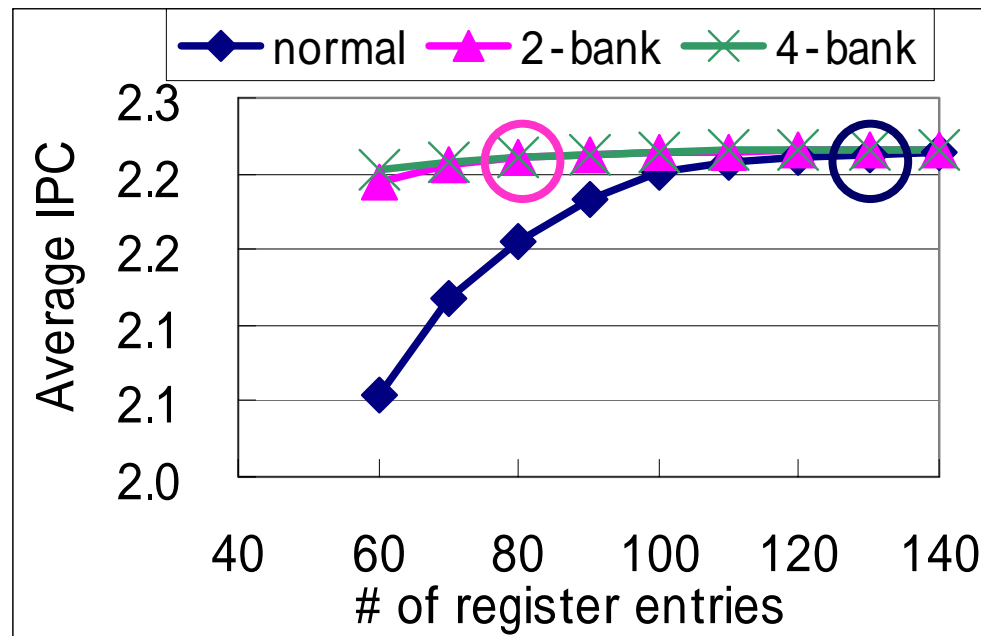
- average IPC (committed Instruction Per Cycle)



- almost no IPC degradation even in small RF size in BPRF
- 50 entries are enough to obtain sustained IPC in 2-bank BPRF
 - 90 entries are required in normal

Average IPC (8-issue configuration)

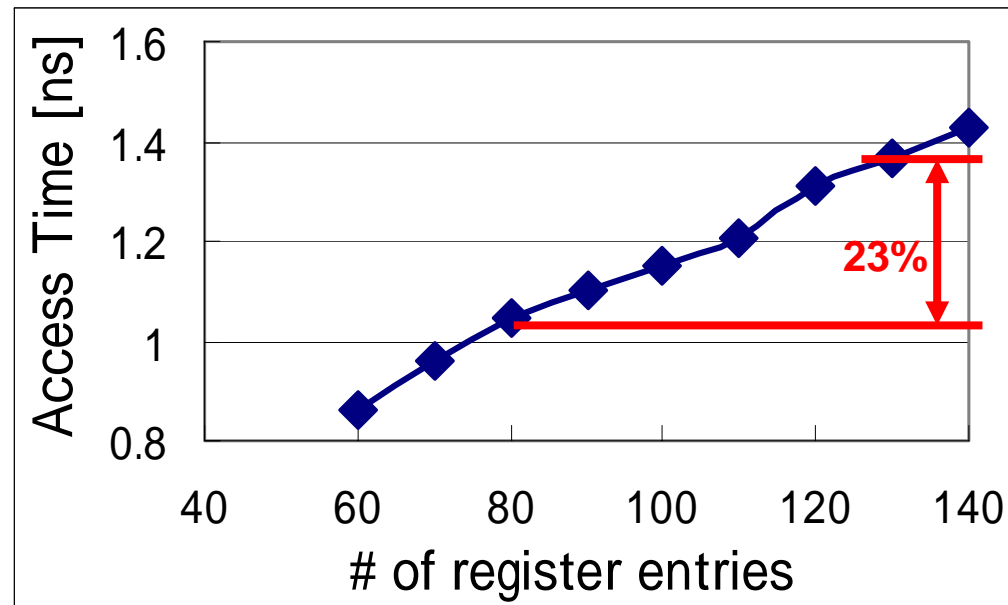
- average IPC (committed Instruction Per Cycle)



- 80 entries are enough to obtain sustained IPC in 2-bank BPRF
 - 130 entries are required in normal
- *high performance even in small RF size in BPRF*

Access time

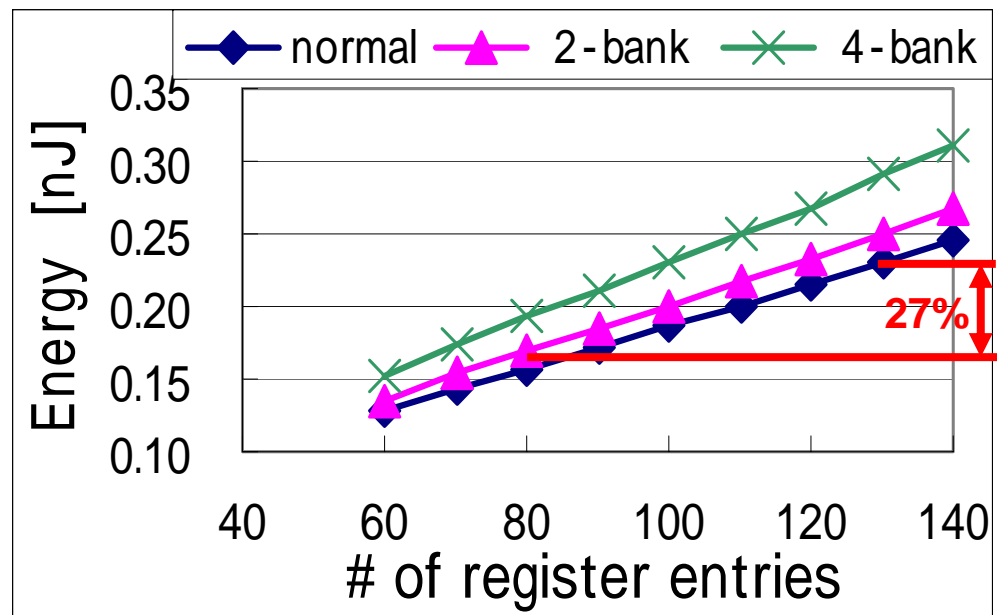
- access time for a RF access
(8-wide configuration, 0.1um process technology)



- access time of **noraml** and **BPRF** is almost the same
- comparison between **2-bank 80 entries** and **normal 130 entries**
 - recap: 80-entry 2bank achieves the same IPC with 130-entry normal
 - 23% reduction of access time in 2-bank BPRF

Energy consumption

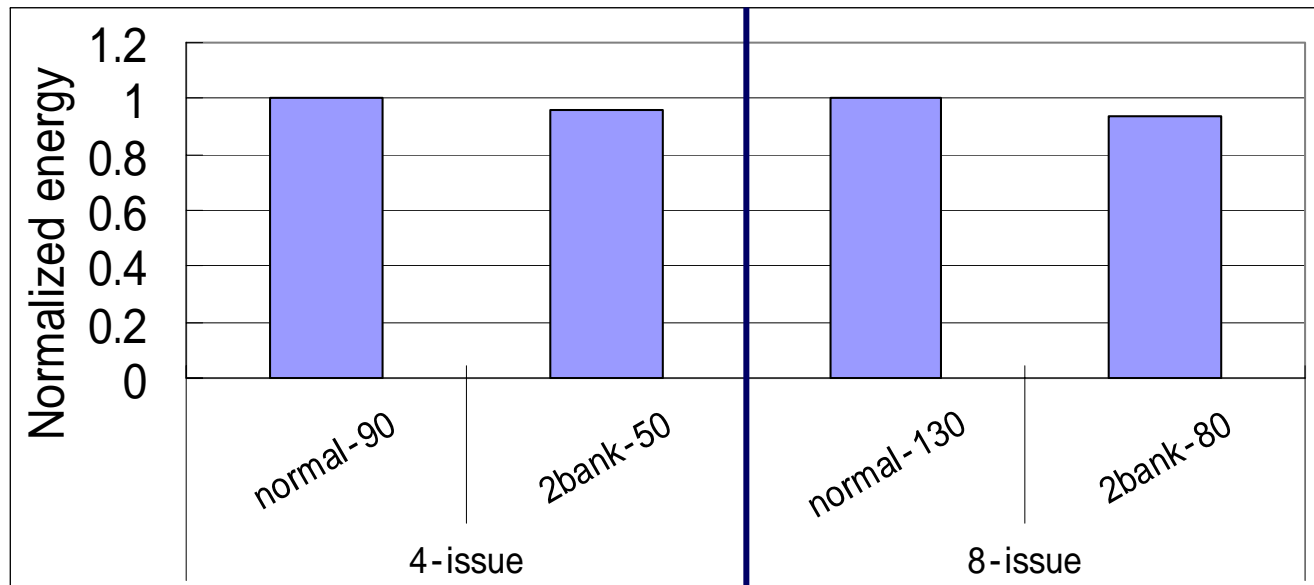
- Energy consumption for a RF access
(for 8-issue configuration, 0.1um process technology)



- comparison between 2-bank 80 entries and normal 130 entries
 - 27% reduction of energy consumption in 2-bank BPRF

Energy including extended part

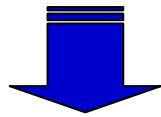
- total energy consumption of processor core (excluding cache energy)



- total processor energy consumption is reduced in 2-bank BPRF

discussion

- higher IPC is achieved even with small RF in BPRF
- BPRF can reduce RF size while keeping performance
 - access time and energy consumption of RF is reduced



can solve the problem of increasing access time and power consumption of large multi-ported register file

- small IPC difference between 2-bank and 4-bank
 - 2-banked configuration is adequate for BPRF
- total chip energy consumption is reduced even including energy consumption of extended part

Outline

- Background
- Concept of Bit-Partitioned Register File
- Microarchitectural implementation
- Evaluation
- Conclusion

Conclusion

- *Bit-Partitioned Register File*
 - high IPC can be achieved even with small register file
- can reduce RF size while keeping performance
 - access time and energy consumption of RF is greatly reduced
- ***potential of solving the problem of increasing access time and power consumption of large multi-ported RF***
- future work
 - applying BPRF concept to a floating-point register file
 - applying BPRF concept to a cache