Software-Oriented Memory-Management Design

Bruce Jacob

Advanced Computer Architecture Lab
University of Michigan

OUTLINE:

• Motivation
• Architecture
• Evaluation
• Problems & Solutions

Motivation

Trends in Microprocessor Design:

• Faster clock speeds
• Larger and cheaper memories
• More on-chip devices
• Increased flexibility
• Increased focus on testability and reliability

One Conclusion:

• Make it SIMPLE
• Do it in SOFTWARE
Implications for Memory Management?

Translation Lookaside Buffer
- Large, fully associative, on critical path

Finite State Machine
- Good performance, limited flexibility

SOFTVM Design

VIRTUAL CACHE HIERARCHY replaces Translation Lookaside Buffer

CACHE-MISS INTERRUPT replaces Finite State Machine

Thesis tests validity of software-oriented design on memory management
**TLB Architecture**

- Virtual Page Number
- Page Offset
- TLB (CAM)
- VPN, P, PFN
- Protection Violation?
- Match?
- DIRECT-MAPPED CACHE
- Cache Block (Data)
- PFNVPN
- TLB (CAM)
- P-Tag
- Protection Violation?
- Match?
- AND
- CACHE DATA

**TLB Miss**

- CPU
- TLB
- CACHE
- NO DATA
- Virtual Address
  - Index
  - Index
  - Offset
  - Page Frame
  - BASE
SOFTVM Architecture

DIRECT-MAPPED CACHE

V-Tag

Match?

Protection Violation?

Pr

AND

CACHE MISS EXCEPTION

CACHE DATA

SOFTVM Cache Miss

CACHE MISS

NO DATA

Virtual Address

Index

Offset

Index

User Table

Page Frame

PHYSICAL ADDR

PHYSICAL ADDR

PHYSICAL & VIRTUAL ADDR

BASE
Expectations

✔ REDUCTION in DIE AREA

✘ SENSITIVE to CACHE ORGANIZATION

✔ MISSES INFREQUENT rel. to TLB

✘ MISSES EXPENSIVE rel. to TLB

✔ INCREASED FLEXIBILITY

Evaluation

Simulated 5 Virtual Memory Designs:

- SOFTVM - No TLB
- Ultrix/MIPS - SW-mgd TLB, part.
- Mach/MIPS - SW-mgd TLB, part.
- BSD/Intel - HW-mgd TLB, no part.
- PA-RISC - SW-mgd TLB, no part.

TLBs: 128-entry I-TLB, 128-entry D-TLB
Fully associative, Random replacement, 16 entries in Protected Partition

Virtual cache hierarchies, Perfect memory
SOFTW ARE ORIENTED MEMO RY MANAGEMENT DESIGN

Bruce Jacob
University of Michigan

SOFTVM VM-simulation

Root-level handler: 20 inst, 1 PTE load

User-level handler: 10 inst, 1 PTE load

ULTRIX VM-simulation

Root-level handler: 20 inst, 1 PTE load

User-level handler: 10 inst, 1 PTE load
MACH VM-simulation

Root-level handler: 500 inst, 1 PTE load
Kernel-level handler: 20 inst, 1 PTE load
User-level handler: 10 inst, 1 PTE load

INTEL VM-simulation

User-level handler: 7 cycles, 2 PTE loads
PARISC VM-simulation

User-level handler: 20 inst, ≥ 1 PTE loads

PTEs are 2x size of other tables

VM Performance: GCC
VM Break-downs: GCC

How Often: GCC
How Much: GCC

- SOFTVM
- MACH
- INTEL
- PARISC

Cycles per Handler Invocation vs. L1 Cache Size - per side (KB)

L1/L2 Linesizes (bytes)
**TLB Sensitivity: GCC**

Mean free path of GCC — ULTRIX page table, 4MB caches

Hardware schemes VERY sensitive to TLB miss-rate ... Software scheme would look 5x better against 64-entry TLBs

**Bottom Line: GCC**

Software scheme: I-side

Hardware schemes: I-side

Software scheme: D-side

Hardware schemes: D-side
Bottom Line (closeup): GCC

Problems & Solutions ...

USE OF EXCEPTION MECHANISM

Exception Problem

CACHE-BOUND PERFORMANCE

Multimedia Problem

LARGE VIRTUAL CACHES

Synonym Problem

LEVEL-2 CACHES

Cost Problem
Exception Problem

Cost of Interrupt = Cost of flushing Reorder Buffer

Potential Gains
Re-Execute Buffer

ROB Entry contains **DEPENDENCY INFO**

On Interrupt, **DO NOT** Flush Reorder Buffer

At Instruction Commit:
- If **DONE**, Commit
- If waiting for **EXECUTE**, Wait
- If waiting for **OPERAND**, Re-Execute

---

### ROB/REB Illustration

<table>
<thead>
<tr>
<th>Reorder Buffer</th>
<th>Re-Ex Buffer</th>
<th>Reorder Buffer</th>
<th>Re-Ex Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inst Dep's</td>
<td></td>
<td>Inst Dep's</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>-</td>
<td>E*</td>
<td>-</td>
</tr>
<tr>
<td>B</td>
<td>-</td>
<td>F</td>
<td>D, E</td>
</tr>
<tr>
<td>C</td>
<td>A, B</td>
<td>G</td>
<td>D</td>
</tr>
<tr>
<td>D</td>
<td>A, D</td>
<td>H</td>
<td>G</td>
</tr>
<tr>
<td>E*</td>
<td>C</td>
<td>x1</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>D, E</td>
<td>x2</td>
<td>x1</td>
</tr>
<tr>
<td>G</td>
<td>D</td>
<td>x3</td>
<td>x2</td>
</tr>
<tr>
<td>H</td>
<td>G</td>
<td>x4</td>
<td>x3</td>
</tr>
</tbody>
</table>

**Detection of Exception Inst E at Head of ROB**

<table>
<thead>
<tr>
<th>Reorder Buffer</th>
<th>Re-Ex Buffer</th>
<th>Reorder Buffer</th>
<th>Re-Ex Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inst Dep's</td>
<td></td>
<td>Inst Dep's</td>
<td></td>
</tr>
<tr>
<td>x1</td>
<td>-</td>
<td>E*</td>
<td>-</td>
</tr>
<tr>
<td>x2</td>
<td>x1</td>
<td>F</td>
<td>D, E</td>
</tr>
<tr>
<td>x3</td>
<td>x2</td>
<td>next: I</td>
<td></td>
</tr>
<tr>
<td>x4</td>
<td>x3</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>x5</td>
<td>x4</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>x6</td>
<td>x5</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>x7</td>
<td>x6</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>x8</td>
<td>x7</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

**Handler at Head of ROB**

**Re-Execution of E & F**
Multimedia Problem
STREAM

MULTIMEDIA HAS
NO TEMPORAL LOCALITY

WORST-CASE SCENARIO:
Take an exception for every cache line

SOLUTIONS:
- Prefetch buffers
- Prefetch into L2 cache
- Provide unmapped regions to user

Unmapped 4MB Superpages

32-bit Effective Address
Seg# (10)  Segment Offset (22)

Segment Cache

44-bit Virtual Address
Segment ID (22)  Segment Offset (22)

To Virtual Caches and physical memory

Alternative (Non-Segmented) Solution:
- Small Superpage TLB (BAT Registers)
**STREAM Performance**

![Graph showing STREAM performance with overhead (CPI) against L1 Cache Size per side (KB) for SOFTVM and ULTRIX. The graph compares L1d-miss, L2d-miss, and L1-miss scenarios.]

**STREAM VM Performance**

![Graph showing STREAM VM performance with overhead (CPI) against L1 Cache Size per side (KB) for SOFTVM and ULTRIX. The graph compares various overhead scenarios such as handler-MEM, rpte-MEM, rpte-L2, and uhandlers.]
... with Superpages

Virtual Cache Synonym Problem
Solutions to Synonym Problem

HARDWARE
- Backpointers

SOFTWARE
- OS/2: Eliminate Aliasing
- SunOS: Aliases map to same cache line
- SASOS: Eliminate Aliasing

HARDWARE/SOFTWARE
- Segmentation ....

Segmentation

32-Bit Virtual Address

SEGMENTATION MECHANISM

Global Virtual Space

PAGE TABLE and TLB

Physical Memory
Segmented Solution

**ALIASING without SYNONYM PROBLEMS**

Cost Problem

**Solution Requires (LARGE) L2 Caches**

What about embedded market?
What about cost-cutter market?

*Implement L2 Caches in DRAM, Call it MAIN MEMORY*

Requirement: ability to pin down critical regions ...
Virtually-Addressed Main Memory

Conclusions

Elimination of MMU is Possible

Cycle Time can DECREASE
Performance can INCREASE

Software-Managed: FLEXIBILITY

Examples of Usefulness:
- Shared-Memory Multiprocessor
- Real-Time Processing
- Architecture Emulation
- Restricted Reconfigurable Computing