Architecture

Traditional Chip Multiprocessor (CMP)

CMP + Speculation
Architecture MDT

- Round Robin
- No flush dirty data

MDT

0x124

line addr
tag
w0 w1 w2 w3

L0 L1 L2 L3 S0 S1 S2 S3

Multibank
Transition

spec → non-spec → commit → spec

no displacement of L1 Dirty lines; else stall

write-through displacement OK

synch: flush all Dirty data to L2

In-order commit
Extra Bits per L1 line

For now: per-word bits

<table>
<thead>
<tr>
<th>Flush (F)</th>
<th>Invalid (I)</th>
<th>Dirty (D)</th>
<th>Safe Read (SR)</th>
<th>Safe Write (SW)</th>
</tr>
</thead>
</table>

SR/SW: Allow ld/st w/o informing MDT

F: Allow L1 lines to remain valid across thread initiation

D: Data not consistent w/ L2

/* may be diff in different L1's*/

I: No data
## Example

<table>
<thead>
<tr>
<th>Valid</th>
<th>Addr Tag</th>
<th>LD bits (W0)</th>
<th>St bits (W0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0x1234</td>
<td>0010</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0x4321</td>
<td>0011</td>
<td>0100</td>
</tr>
</tbody>
</table>

**Note:**
- LD bits (W0): 0010
- St bits (W0): 0100
Load: MDT operation

if (spec)
    /* access MDT to find who's got data and to record the L */
if (noentry)
    allocate one

Li = 1

Who has data: closest predecessor (including myself) \( w/S_i = 1 \)

/* use mask bits */

Forward request to right L1  /* maybe non-spec */
/* if none, send it to L2 */
write and \( SW \neq 0 \)

predecessors \( \rightarrow \) successors

\( F=1 \) \( \rightarrow \) may squash / invalidate
Store: MDT Operation

/* access MDT to squash successors with premature loads, send invalidations, and record the ST */

if (noentry found && spec), allocate

if (spec), $S_i = 1$

Evaluate the Squash Logic: "has any successor done an unsafe ld without any intervening thread performing a ST"?

e.g. if O non-spec: $L_1 + \overline{S}_1 (L_2 + \overline{S}_2 L_3)$

/* use mask bits */

if (true) complemented
Send signal to all predecessors: F = 1

/* because of data from prev i.e. */

if (L = 0) that redefines the word (S = 1)

invalidate up to (not including) the successor

else at same time: invalidate the entry from their caches

1. Squash all successors starting w/ the one w/ L = 1

if (true)

Store (cont.)
Overflow

- The MDT may overflow
- If so, thread trying to add entry \( \rightarrow \) stalls
  (must be spec)
- When non-spec commits: clear LS \( \Rightarrow \) may free up
- Small MDT OK
Details on MDT

- when a thread commits or gets squashed, its L,S are cleared.

- MDT may be incorporated in L2

- Like a directory: keeps track of who's sharing the data

- Overall, small size (see later)
Hardware Support → Enhanced Scoreboard

Hardware for Register Communication

Synchronizing Scoreboard (SS)
- Local bits: V, P, X
- Replicated Global bits: F, S
  - consistency easily maintained
  - F: when a thread is initiated
  - S: when value is written on the bus

Register Transfer Bus (SS bus)
- Transfer register values between processors
- Simple Broadcast bus
  - limited bandwidth: 1 word/cycle
  - only 1 read & 1 write port / processor
  - latency: 1-3 cycles + contention
Hardware Support

Hardware for Register Communication

<table>
<thead>
<tr>
<th>Status Mask</th>
<th>Thread Mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-Speculative</td>
<td>0001</td>
</tr>
<tr>
<td>Speculative Successor 1</td>
<td>0011</td>
</tr>
<tr>
<td>Speculative Successor 2</td>
<td>0111</td>
</tr>
<tr>
<td>Speculative Successor 3</td>
<td>1111</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Synchronizing Scoreboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register</td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

GLOBAL (replicated)
LOCAL

Equations:

1. Thread startup:
   \[ V = V - \bigcup F_{\text{pred}} \]
2. thread 0 is non-speculative; thread 3 accesses r3
   \[ V_3 + \overline{S}_2(F_2 + \overline{S}_1(F_1 + \overline{S}_0)) \]
3. thread 1 is non-speculative; thread 0 accesses r3
   \[ V_0 + \overline{S}_3(F_3 + \overline{S}_2(F_2 + \overline{S}_1)) \]
4. thread 1 is non-speculative; thread 3 accesses r3
   \[ V_3 + \overline{S}_2(F_2 + \overline{S}_1) \]
Register Communication

• **Producer-initiated:**
  
  Thread clears S bit, puts req on S bus

  Successors check own V
  
  F of threads between producer and itself
  
  if all 0
  
  \( \Rightarrow \) load req
  
  \( \Rightarrow V=1 \)

• **Consumer initiated:**
  
  Check own V bit and F,S of predecessors
  
  Available? e.g. \[ V_3 + S_2 (F_2 + S_1 (F_1 + S_0)) \]
  
  If so, read and set \( V=1 \)

  else stall
Software Support

- Annotation of the Binary

Executable

Identify Basic Blocks
Generate CFG & perform
live variable & reaching
definition analysis

Identify loops using dominator
and back-edge information
Annotate thread initiation
and termination points

Get looplive registers (i.e. registers live at loop
entry/exits & also redefined in loop)
Identify looplive reaching definitions at all exits
Identify safe definitions & release points for all
looplive reaching definitions
Identify induction variables & move induction
updates dominating all exits closer to entry point
Annotate safe/release points

Annotated
Executable

looplive: r3

(entry)

..=r3

(unsafe)r3=...

(safe)r3=...

(release r3)

(entry)

Binary Annotation Process
Safe definitions & release points

05/07/98

Final Examination
evaluation

- 4x4-way chip MP vs. 12-way/16-way superscalar
  - Complexity of 4x4-way compared to a conventional superscalar
    - Area: comparable to a 12-way conventional superscalar
    - Timing: for 0.18μ: 8-way is 25% slow; higher for 12-/16-way [Palacharla] → for more

<table>
<thead>
<tr>
<th>compress</th>
<th>mpeg</th>
<th>ijpeg</th>
<th>eqntott</th>
<th>li</th>
<th>swim</th>
<th>tomatc</th>
<th>hydro2d</th>
<th>su2cor</th>
<th>mgrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.35</td>
<td>3</td>
<td>3.87</td>
<td>3.51</td>
<td>3.61</td>
<td>3.63</td>
<td>4.77</td>
<td>4.93</td>
<td>4.72</td>
<td>4.73</td>
</tr>
<tr>
<td>4.41</td>
<td>2.44</td>
<td>2.45</td>
<td>1.83</td>
<td>2.33</td>
<td>2.38</td>
<td>2.12</td>
<td>2.15</td>
<td>2.6</td>
<td>2.68</td>
</tr>
<tr>
<td>2.6</td>
<td>2.65</td>
<td>2.65</td>
<td>3.92</td>
<td>3.92</td>
<td>3.92</td>
<td>5.16</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

05/07/98
Final Examination