Speculative Execution

Daniel Sanchez
Computer Science and Artificial Intelligence Laboratory
M.I.T.
Speculative Execution Recipe

1. Proceed ahead despite unresolved dependencies using a prediction for an architectural or micro-architectural value

2. Maintain both old and new values on updates to architectural (and often micro-architectural) state

3. After sure that there was no mis-speculation and there will be no more uses of the old values, discard old values and just use new values

3. In event of mis-speculation, dispose of all new values, restore old values, and re-execute from point before mis-speculation

Why might one use old values?
Value Management Strategies

Greedy (or Eager) Update:
- Update value in place, and
- Provide means to reconstruct old values for recovery
  - often this is a log of old values

Lazy Update:
- Buffer new value, leaving old value in place
- Replace old value only at ‘commit’ time

Why leave an old value in place?
Exception Handling
(In-Order Five-Stage Pipeline)

Strategy for Registers?

Strategy for PC?
Misprediction Recovery

In-order execution machines:
- Guarantee no instruction issued after branch can write-back before branch resolves by keeping values in the pipeline
- Kill all values from all instructions in pipeline behind mispredicted branch

Out-of-order execution?
- Multiple instructions following branch in program order can generate new values before branch resolves
Data-Driven Execution

Renaming table & reg file

Reorder buffer

<table>
<thead>
<tr>
<th>Ins#</th>
<th>use</th>
<th>exec</th>
<th>op</th>
<th>p1</th>
<th>src1</th>
<th>p2</th>
<th>src2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Basic Operation:
Enter op and tag or data (if known) for each sourceReplace tag with data as it becomes availableIssue instruction when all sources are availableSave dest data when operation finishes

Update strategy?
Rollback and Renaming

Convert to lazy by holding data in ROB.

*But how do we find values before they are committed?*

*Search the “dest” field in the reorder buffer*
What is the update policy of rename table?
What events cause mis-speculation?
How can we respond to mis-speculation?
After being cleared, when can instructions be added to ROB?
Take snapshot of register rename table at each predicted branch, recover earlier snapshot if branch mispredicted.
## Map Table Recovery - Snapshots

Speculative value management of microarchitectural state

<table>
<thead>
<tr>
<th></th>
<th>Reg Map</th>
<th>V</th>
<th></th>
<th>Snap Map</th>
<th>V</th>
<th></th>
<th>Snap Map</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0</td>
<td>T20</td>
<td>X</td>
<td></td>
<td>T20</td>
<td>X</td>
<td></td>
<td>T20</td>
<td>X</td>
</tr>
<tr>
<td>R1</td>
<td>T73</td>
<td>X</td>
<td></td>
<td>T73</td>
<td>X</td>
<td></td>
<td>T08</td>
<td></td>
</tr>
<tr>
<td>R2</td>
<td>T45</td>
<td>X</td>
<td></td>
<td>T45</td>
<td>X</td>
<td></td>
<td>T45</td>
<td>X</td>
</tr>
<tr>
<td>R3</td>
<td>T128</td>
<td></td>
<td></td>
<td>T128</td>
<td></td>
<td></td>
<td>T128</td>
<td>X</td>
</tr>
<tr>
<td>R30</td>
<td>T54</td>
<td></td>
<td></td>
<td>T54</td>
<td></td>
<td></td>
<td>T54</td>
<td></td>
</tr>
<tr>
<td>R31</td>
<td>T88</td>
<td>X</td>
<td></td>
<td>T88</td>
<td>X</td>
<td></td>
<td>T88</td>
<td>X</td>
</tr>
</tbody>
</table>

What kind of value management is this?
Branch Predictor Recovery

• 1-Bit Counter Recovery

\[
\begin{array}{c}
0 \\
1 \\
0 \\
1 \\
\end{array}
\]

• 2-Bit Counter Recovery

\[
\begin{array}{c}
00 \\
11 \\
01 \\
10 \\
\end{array}
\]

• Global History Recovery

10101010

• Local History Recovery

\[
\begin{array}{c}
10101010 \\
01010101 \\
\end{array}
\]
O-o-O Execution with ROB
Data-in-ROB design

Basic Operation:
• Enter op and tag or data (if known) for each source
• Replace tag with data as it becomes available
• Issue instruction when all sources are available
• Save dest data when operation finishes
• Commit saved dest data when instruction commits
Unified Physical Register File
(MIPS R10K, Alpha 21264, Pentium 4)

- One regfile for both committed and speculative values (no data in ROB)
- During decode, instruction result allocated new physical register, source regs translated to physical regs through rename table
- Instruction reads data from regfile at start of execute (not in decode)
- Write-back updates reg. busy bits on instructions in ROB (assoc. search)
- Snapshots of rename table taken at every branch to recover mispredicts
- On exception, renaming undone in reverse order of issue (MIPS R10000)
Speculative & Out-of-Order Execution

In-Order

Branch Prediction

Fetch

Decode & Rename

Reorder Buffer

Commit

Update predictors

Out-of-Order

Physical Reg. File

Branch Unit

ALU

MEM

Store Buffer

D$

Execute

In-Order
Lifetime of Physical Registers

- Physical regfile holds committed and speculative values
- Physical registers decoupled from ROB entries (no data in ROB)

a) `ld r1, (r3)`
b) `add r3, r1, #4`
c) `sub r1, r3, r9`
d) `add r3, r1, r7`
e) `ld r6, (r1)`
f) `add r8, r6, r3`
g) `st r8, (r1)`
h) `ld r3, (r11)`

When can we reuse a physical register?

```
ld P1, (Px)
add P2, P1, #4
sub P3, P2, Py
add P4, P3, Pz
ld P5, (P3)
add P6, P5, P4
st P6, (P3)
ld P7, (Pw)
```
Physical Register Management

### Rename Table

<table>
<thead>
<tr>
<th>R0</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P8</td>
<td></td>
<td>P7</td>
<td></td>
<td></td>
<td>P5</td>
<td>P6</td>
</tr>
</tbody>
</table>

### Physical Regs

<table>
<thead>
<tr>
<th>P0</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>P5</th>
<th>P6</th>
<th>P7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&lt;R6&gt;</td>
<td>&lt;R7&gt;</td>
<td>&lt;R3&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>p</td>
<td>p</td>
<td>p</td>
</tr>
</tbody>
</table>

### Free List

<table>
<thead>
<tr>
<th>P0</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### ROB

<table>
<thead>
<tr>
<th>use</th>
<th>ex</th>
<th>op</th>
<th>p1</th>
<th>PR1</th>
<th>p2</th>
<th>PR2</th>
<th>Rd</th>
<th>LPRd</th>
<th>PRd</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- `ld r1, 0(r3)`
- `add r3, r1, #4`
- `sub r6, r7, r6`
- `add r3, r3, r6`
- `ld r6, 0(r1)`

(LPRd requires third read port on Rename Table for each instruction)
### Physical Register Management

#### Rename Table

<table>
<thead>
<tr>
<th>R0</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P6</td>
<td>P8</td>
<td>P7</td>
<td></td>
<td></td>
<td>P5</td>
<td>P6</td>
</tr>
</tbody>
</table>

#### Physical Regs

<table>
<thead>
<tr>
<th>Rename Table</th>
<th>Physical Regs</th>
<th>Free List</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P0</td>
<td>P0</td>
</tr>
<tr>
<td></td>
<td>P1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P2</td>
<td>P1</td>
</tr>
<tr>
<td></td>
<td>P3</td>
<td>P3</td>
</tr>
<tr>
<td></td>
<td>P4</td>
<td>P2</td>
</tr>
<tr>
<td></td>
<td>P5</td>
<td>P0</td>
</tr>
<tr>
<td></td>
<td>&lt;R6&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>P1</td>
</tr>
<tr>
<td></td>
<td>&lt;R7&gt;</td>
<td>P3</td>
</tr>
<tr>
<td></td>
<td>P7</td>
<td>P2</td>
</tr>
<tr>
<td></td>
<td>&lt;R3&gt;</td>
<td>P4</td>
</tr>
<tr>
<td></td>
<td>P8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;R1&gt;</td>
<td>P0</td>
</tr>
<tr>
<td></td>
<td>Pn</td>
<td></td>
</tr>
</tbody>
</table>

#### ROB

<table>
<thead>
<tr>
<th>x</th>
<th>ex</th>
<th>op</th>
<th>p1</th>
<th>PR1</th>
<th>p2</th>
<th>PR2</th>
<th>Rd</th>
<th>LPRd</th>
<th>PRd</th>
</tr>
</thead>
<tbody>
<tr>
<td>ld</td>
<td>ld</td>
<td>p</td>
<td>P7</td>
<td>P7</td>
<td>r1</td>
<td>P8</td>
<td>P0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- ld r1, 0(r3)
- add r3, r1, #4
- sub r6, r7, r6
- add r3, r3, r6
- ld r6, 0(r1)
Physical Register Management

- **Rename Table**
  - R0
  - R1
  - R2
  - R3
  - R4
  - R5
  - R6
  - R7

- **Physical Regs**
  - P0
  - P1
  - P2
  - P3
  - P4
  - P5
  - P6
  - P7
  - P8
  - Pn

- **Free List**
  - P0
  - P1
  - P2
  - P3
  - P4

- **ROB**
  - use
  - ex
  - op
  - p1
  - PR1
  - p2
  - PR2
  - Rd
  - LPRd
  - PRd

<table>
<thead>
<tr>
<th>use</th>
<th>ex</th>
<th>op</th>
<th>p1</th>
<th>PR1</th>
<th>p2</th>
<th>PR2</th>
<th>Rd</th>
<th>LPRd</th>
<th>PRd</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td></td>
<td>ld</td>
<td>p</td>
<td>P7</td>
<td></td>
<td></td>
<td>r1</td>
<td>P8</td>
<td>P0</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>add</td>
<td></td>
<td>P0</td>
<td></td>
<td></td>
<td>r3</td>
<td>P7</td>
<td>P1</td>
</tr>
</tbody>
</table>

- Codesnippets:
  - ld r1, 0(r3)
  - add r3, r1, #4
  - sub r6, r7, r6
  - add r3, r3, r6
  - ld r6, 0(r1)
Physical Register Management

**Rename Table**

<table>
<thead>
<tr>
<th>R0</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P0</td>
<td></td>
<td>P1</td>
<td></td>
<td></td>
<td>P5</td>
<td>P3</td>
</tr>
</tbody>
</table>

**Physical Regs**

<table>
<thead>
<tr>
<th>P0</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>P5</th>
<th>P6</th>
<th>P7</th>
<th>P8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P6</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Free List**

- P0
- P1
- P2
- P3
- P4
- P5
- P6
- P7
- P8

**ROB**

<table>
<thead>
<tr>
<th>use</th>
<th>ex</th>
<th>op</th>
<th>p1</th>
<th>PR1</th>
<th>p2</th>
<th>PR2</th>
<th>Rd</th>
<th>LPRd</th>
<th>PRd</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td></td>
<td>ld</td>
<td>p</td>
<td>P7</td>
<td></td>
<td></td>
<td>r1</td>
<td>P8</td>
<td>P0</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>add</td>
<td></td>
<td>P0</td>
<td></td>
<td></td>
<td>r3</td>
<td>P7</td>
<td>P1</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>sub</td>
<td>p</td>
<td>P6</td>
<td>p</td>
<td>P5</td>
<td>r6</td>
<td>P5</td>
<td>P3</td>
</tr>
</tbody>
</table>

Instructions:
- `ld r1, 0(r3)`
- `add r3, r1, #4`
- `sub r6, r7, r6`
- `add r3, r3, r6`
- `ld r6, 0(r1)`
Physical Register Management

### Rename Table

<table>
<thead>
<tr>
<th>R0</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
<th>R4</th>
<th>R5</th>
<th>R6</th>
<th>R7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P0</td>
<td></td>
<td>P1</td>
<td></td>
<td></td>
<td>P3</td>
<td>P6</td>
</tr>
</tbody>
</table>

### Physical Regs

<table>
<thead>
<tr>
<th>P0</th>
<th>P1</th>
<th>P2</th>
<th>P3</th>
<th>P4</th>
<th>P5</th>
<th>P6</th>
<th>P7</th>
<th>P8</th>
<th>Pn</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>&lt;R6&gt;</td>
<td>&lt;R7&gt;</td>
<td>&lt;R3&gt;</td>
<td>&lt;R1&gt;</td>
<td></td>
</tr>
</tbody>
</table>

### Free List

| P0 | P1 | P2 | P3 | P4 |

### ROB

<table>
<thead>
<tr>
<th>use</th>
<th>ex</th>
<th>op</th>
<th>p1</th>
<th>PR1</th>
<th>p2</th>
<th>PR2</th>
<th>Rd</th>
<th>LPRd</th>
<th>PRd</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td></td>
<td>ld</td>
<td>p</td>
<td>P7</td>
<td></td>
<td></td>
<td>r1</td>
<td>P8</td>
<td>P0</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>add</td>
<td></td>
<td>P0</td>
<td></td>
<td></td>
<td>r3</td>
<td>P7</td>
<td>P1</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>sub</td>
<td>p</td>
<td>P6</td>
<td>p</td>
<td>P5</td>
<td>r6</td>
<td>P5</td>
<td>P3</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>add</td>
<td></td>
<td>P1</td>
<td>P3</td>
<td></td>
<td>r3</td>
<td>P1</td>
<td>P2</td>
</tr>
</tbody>
</table>

- ld r1, 0(r3)
- add r3, r1, #4
- sub r6, r7, r6
- add r3, r3, r6
- ld r6, 0(r1)
Physical Register Management

**Rename Table**
- R0: P0
- R1: P0
- R2: P2
- R3: P2
- R4: 
- R5: P3
- R6: P3, P4
- R7: P6

**Physical Regs**
- P0
- P1
- P2
- P3
- P4
- P5: <R6>
- P6: <R7>
- P7: <R3>
- P8: <R1>
- Pn

**Free List**
- R0
- R1
- R2
- R3
- R4
- R5
- R6
- R7

**ROB**

<table>
<thead>
<tr>
<th>use</th>
<th>ex</th>
<th>op</th>
<th>p1</th>
<th>PR1</th>
<th>p2</th>
<th>PR2</th>
<th>Rd</th>
<th>LPRd</th>
<th>PRd</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td></td>
<td>ld</td>
<td>p</td>
<td>P7</td>
<td></td>
<td></td>
<td>r1</td>
<td>P8</td>
<td>P0</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>add</td>
<td>P0</td>
<td></td>
<td></td>
<td></td>
<td>r3</td>
<td>P7</td>
<td>P1</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>sub</td>
<td>P6</td>
<td>p</td>
<td>P5</td>
<td></td>
<td>r6</td>
<td>P5</td>
<td>P3</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>add</td>
<td>P1</td>
<td>P3</td>
<td>r3</td>
<td></td>
<td>P1</td>
<td>P2</td>
<td></td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>ld</td>
<td>P0</td>
<td></td>
<td>r6</td>
<td>P3</td>
<td>P4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- ld r1, 0(r3)
- add r3, r1, #4
- sub r6, r7, r6
- add r3, r3, r6
- ld r6, 0(r1)
Physical Register Management

Rename Table

<table>
<thead>
<tr>
<th>Physical Regs</th>
<th>Free List</th>
</tr>
</thead>
<tbody>
<tr>
<td>P0 &lt;R1&gt;</td>
<td>p</td>
</tr>
<tr>
<td>P1</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td></td>
</tr>
<tr>
<td>P5 &lt;R6&gt;</td>
<td>p</td>
</tr>
<tr>
<td>P6 &lt;R7&gt;</td>
<td>p</td>
</tr>
<tr>
<td>P7 &lt;R3&gt;</td>
<td>p</td>
</tr>
<tr>
<td>P8 &lt;R1&gt;</td>
<td>p</td>
</tr>
<tr>
<td>Pn</td>
<td></td>
</tr>
</tbody>
</table>

ROB

<table>
<thead>
<tr>
<th>use</th>
<th>ex</th>
<th>op</th>
<th>p1</th>
<th>PR1</th>
<th>p2</th>
<th>PR2</th>
<th>Rd</th>
<th>LPRd</th>
<th>PRd</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>ld</td>
<td>p</td>
<td>P7</td>
<td>p</td>
<td>P2</td>
<td>r1</td>
<td>P8</td>
<td>P0</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>add</td>
<td>P0</td>
<td></td>
<td></td>
<td></td>
<td>r3</td>
<td>P7</td>
<td>P1</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>sub</td>
<td>P6</td>
<td>p</td>
<td>P5</td>
<td>r6</td>
<td>P5</td>
<td>P3</td>
<td></td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>add</td>
<td>P1</td>
<td></td>
<td>P3</td>
<td>r3</td>
<td>P1</td>
<td>P2</td>
<td></td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>ld</td>
<td>P0</td>
<td></td>
<td></td>
<td>r6</td>
<td>P3</td>
<td>P4</td>
<td></td>
</tr>
</tbody>
</table>

Execute & Commit

ld r1, 0(r3)
add r3, r1, #4
sub r6, r7, r6
add r3, r3, r6
ld r6, 0(r1)
Physical Register Management

**Rename Table**

- R0
- R1 -> P0
- R2
- R3 -> P2
- R4
- R5
- R6 -> P4
- R7 -> P6

**Physical Regs**

- P0
  - <R1> -> p
- P1
  - <R3> -> p
- P2
- P3
- P4
- P5
  - <R6> -> p
- P6
  - <R7> -> p
- P7
  - <R3> -> p
- P8
- Pn

**Free List**

- P0
- P1
- P2
- P3
- P4
- P5
- P6
- P7
- P8

**ROB**

<table>
<thead>
<tr>
<th>use</th>
<th>ex</th>
<th>op</th>
<th>p1</th>
<th>PR1</th>
<th>p2</th>
<th>PR2</th>
<th>Rd</th>
<th>LPRd</th>
<th>PRd</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
<td>ld</td>
<td>p</td>
<td>P7</td>
<td>p</td>
<td></td>
<td>r1</td>
<td>P8</td>
<td>P0</td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>add</td>
<td>p</td>
<td>P0</td>
<td></td>
<td></td>
<td>r3</td>
<td>P7</td>
<td>P1</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>sub</td>
<td>p</td>
<td>P6</td>
<td>p</td>
<td>P5</td>
<td>r6</td>
<td>P5</td>
<td>P3</td>
</tr>
<tr>
<td>x</td>
<td></td>
<td>add</td>
<td>p</td>
<td>P1</td>
<td>P3</td>
<td>r3</td>
<td>P1</td>
<td>P2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>p</td>
<td>P0</td>
<td></td>
<td></td>
<td>r6</td>
<td>P3</td>
<td>P4</td>
</tr>
</tbody>
</table>

- ld r1, 0(r3)
- add r3, r1, #4
- sub r6, r7, r6
- add r3, r3, r6
- ld r6, 0(r1)

**Execute & Commit**

- P7
- P0
- P1
- P2
- P3
- P4
- P5
- P6
- P7
- P8

March 12, 2020

MIT 6.823 Spring 2020

L11-23
Reorder Buffer Holds
Active Instruction Window

... (Older instructions)

ld r1, (r3)
add r3, r1, r2
sub r6, r7, r9
add r3, r3, r6
ld r6, (r1)
add r6, r6, r3
st r6, (r1)
ld r6, (r1)

... (Newer instructions)

Commit

Issue

Execute

Decode

Cycle $t$

ld r1, (r3)
add r3, r1, r2
sub r6, r7, r9
add r3, r3, r6
ld r6, (r1)
add r6, r6, r3
st r6, (r1)
ld r6, (r1)

... Cycle $t + 1$

Key: predecode, decoded, issued, executed, committed
### Issue Timing

<table>
<thead>
<tr>
<th>i1</th>
<th>Add R1, R1, #1</th>
<th>Issue&lt;sub&gt;1&lt;/sub&gt;</th>
<th>Execute&lt;sub&gt;1&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>i2</td>
<td>Sub R1, R1, #1</td>
<td></td>
<td>Issue&lt;sub&gt;2&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Execute&lt;sub&gt;2&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

How can we issue earlier?

<table>
<thead>
<tr>
<th>i1</th>
<th>LD R1, (R3)</th>
<th>Issue&lt;sub&gt;1&lt;/sub&gt;</th>
<th>Execute&lt;sub&gt;1&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>i2</td>
<td>Sub R1, R1, #1</td>
<td></td>
<td>Issue&lt;sub&gt;2&lt;/sub&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Execute&lt;sub&gt;2&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

What might make this schedule fail?
Issue Queue with latency prediction

<table>
<thead>
<tr>
<th>Inst#</th>
<th>use</th>
<th>exec</th>
<th>op</th>
<th>p1 lat1 src1</th>
<th>p2 lat2 src2</th>
<th>dest</th>
</tr>
</thead>
<tbody>
<tr>
<td>BEQZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Speculative Instructions

- Fixed latency: latency included in queue entry ('bypassed')
- Predicted latency: latency included in queue entry (speculated)
- Variable latency: wait for completion signal (stall)

Issue Queue (Reorder buffer)
Data-in-ROB vs. Unified RegFile

How does issue speculation differ, e.g., on cache miss?
Superscalar Register Renaming

- During decode, instructions allocated new physical destination register
- Source operands renamed to physical register with newest value
- Execution unit only sees physical register numbers

### Rename Table

<table>
<thead>
<tr>
<th>Op</th>
<th>Src1</th>
<th>Src2</th>
<th>Dest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Op</td>
<td>Src1</td>
<td>Src2</td>
<td>Dest</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Register Free List</th>
</tr>
</thead>
</table>

### Update Mapping

- **Read Addresses**
- **Write Ports**
- **Rename Table**
- **Read Data**
- **Register Free List**

```
Inst 1: Op Dest Src1 Src2
Inst 2: Op Dest Src1 Src2
```

Does this work?
Superscalar Register Renaming

Update Mapping

Must check for RAW hazards between instructions issuing in same cycle. Can be done in parallel with rename lookup.

(MIPS R10K renames 4 serially-RAW-dependent insts/cycle)
Split Issue and Commit Queues

• How large should the ROB be?
  – Think Little’s Law...

• Can split ROB into issue and commit queues

\textit{Issue Queue}

\begin{tabular}{cccccc}
\hline
use & op & p1 & PR1 & p2 & PR2 & tag \\
\hline
\end{tabular}

\textit{Commit Queue}

\begin{tabular}{cccc}
\hline
ex & Rd & LPRd & PRd \\
\hline
\end{tabular}

• Commit queue: Allocate on decode, free on commit
• Issue queue: Allocate on decode, free on dispatch
• Pros: Smaller issue queue \rightarrow simpler dispatch logic
• Cons: More complex mis-speculation recovery
Speculating Both Directions?

An alternative to branch prediction is to execute both directions of a branch *speculatively*

- Resource requirement is proportional to the number of concurrent speculative executions

- Only half the resources engage in useful work when both directions of a branch are executed speculatively

- Branch prediction takes less resources than speculative execution of both paths

*With accurate branch prediction, it is more cost effective to dedicate all resources to the predicted direction*
Thank you!