The Pipeline Design of Modern Processors

Hung-Wei Tseng
Recap: Pipelining
Recap: Pipelining

After this point, we are completing an instruction each cycle!
Local predictor — every branch instruction has its own state
- 2-bit — each state is described using 2 bits
- Change the state based on **actual** outcome
- If we guess right — no penalty
- If we guess wrong — flush (clear pipeline registers) for mis-predicted instructions that are currently in IF and ID stages and reset the PC

Recap: 2-bit/Bimodal local predictor

<table>
<thead>
<tr>
<th>branch PC</th>
<th>target PC</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x400048</td>
<td>0x400032</td>
<td>10</td>
</tr>
<tr>
<td>0x400080</td>
<td>0x400068</td>
<td>11</td>
</tr>
<tr>
<td>0x401080</td>
<td>0x401100</td>
<td>00</td>
</tr>
<tr>
<td>0x4000F8</td>
<td>0x400100</td>
<td>01</td>
</tr>
</tbody>
</table>

Predict Taken
Recap: Global history (GH) predictor

Branch Target Buffer

\[
\begin{align*}
\text{branch PC} & \quad \text{target PC} \\
0x400048 & \quad 0x400032 \\
0x400080 & \quad 0x400068 \\
0x401080 & \quad 0x401100 \\
0x4000F8 & \quad 0x400100 \\
\end{align*}
\]

Global History Register

\[0100\] \rightarrow (NT, T, NT, NT)

Predict Taken

States associated with history
Outline

• Super Scalar
• Out-of-order execution
How many pairs of instructions in the following MIPS instructions will result in data hazards/stalls in a basic 5-stage MIPS pipeline with “full” data forwarding?

lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10, 8
bne  $10,$5, LOOP

A. 0
B. 1
C. 2
D. 3
E. 4
The effect of code optimization

• By reordering which pair of the following instruction stream can we eliminate all stalls without affecting the correctness of the code?

   1. lw  $6, 0($10)
   2. add $7,$6,$12
   3. sw  $7, 0($10)
   4. addi $10,$10, 8
   5. bne $10,$5, LOOP

A. (1) & (2)
B. (2) & (3)
C. (3) & (4)
D. (4) & (5)
E. None of the pairs can be reordered
Dynamic instruction scheduling/
Out-of-order (OoO) execution
Tips of drawing a pipeline diagram

• Each instruction has to go through all 5 pipeline stages: IF, ID, EXE, MEM, WB in order — only valid if it’s single-issue, RISC-V 5-stage pipeline

• An instruction can enter the next pipeline stage in the next cycle if
  • No other instruction is occupying the next stage
  • This instruction has completed its own work in the current stage
  • The next stage has all its inputs ready

• Fetch a new instruction only if
  • We know the next PC to fetch
  • We can predict the next PC
  • Flush an instruction if the branch resolution says it’s mis-predicted.
What do you need to execute an instruction?

- Whenever the instruction is decoded — put decoded instruction somewhere
- Whenever the inputs are ready — all data dependencies are resolved
- Whenever the target functional unit is available
Scheduling instructions: based on data dependencies

- Draw the data dependency graph, put an arrow if an instruction depends on the other.
  1. `lw $6,0($10)`
  2. `add $7, $6,$12`
  3. `sw $7,0($10)`
  4. `addi $10,$10, 8`
  5. `bne $10, $5, LOOP`
  6. `lw $6,0($10)`
  7. `add $7, $6,$12`
  8. `sw $7,0($10)`
  9. `addi $10,$10, 8`
  10. `bne $10, $5, LOOP`

- **In theory**, instructions without dependencies can be executed in parallel or out-of-order.
- Instructions with dependencies can never be reordered.
We are still limited by **false dependencies**

They are not “true” dependencies because they don’t have an arrow in data dependency graph

- **WAR (Write After Read):** a later instruction overwrites the source of an earlier one
  - 4 and 1 4 and 3, 6 and 2, 7 and 3, 9 and 5, 9 and 6, 9 and 8
- **WAW (Write After Write):** a later instruction overwrites the output of an earlier one
  - 6 and 1, 7 and 2
Out-of-order execution

• Any sequence of instructions has set of RAW, WAW, and WAR hazards that constrain its execution.
• Can we design a processor that extracts as much parallelism as possible, while still respecting these dependences?
Register renaming

Register renaming

• Provide a set of “physical registers” and a mapping table mapping “architectural registers” to “physical registers”
• Allocate a physical register for a new output
• Stages
  • Dispatch (D) — allocate a “physical” for the output of a decoded instruction
  • Issue (I) — collect pending values/branch outcome from common data bus
  • Execute (INT, AQ/AQ/MEM, M1/M2/M3, BR) — send the instruction to its corresponding pipeline if no structural hazards
  • Write Back (WB) — broadcast the result through CDB
Overview of a processor supporting register renaming

Fetch/decode instruction

Renaming logic

Instruction Queue

Unresolved Branch

Register mapping table

Physical Registers

Address Resolution

Integer ALU

Floating-Point Adder

Floating-Point Mul/Div

Branch

Load Queue

Store Queue

Memory

Address

Value

Addr.

Dest.

Reg.

Addr.

Data
Register renaming in motion

1. lw $6,0($10)
2. add $7,$6,$12
3. sw $7,0($10)
4. addi $10,$10,8
5. bne $10,$5,LOOP
6. lw $6,0($10)
7. add $7,$6,$12
8. sw $7,0($10)
9. addi $10,$10,8
10. bne $10,$5,LOOP

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw P1, 0($10)</td>
<td>$5</td>
<td>0</td>
<td>1</td>
<td>P6</td>
<td>P1</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td></td>
<td>$6</td>
<td></td>
<td></td>
<td>P1</td>
<td>P2</td>
<td></td>
<td>P1</td>
</tr>
<tr>
<td></td>
<td>$7</td>
<td></td>
<td></td>
<td>P1</td>
<td>P3</td>
<td></td>
<td>P1</td>
</tr>
<tr>
<td></td>
<td>$10</td>
<td></td>
<td></td>
<td>P1</td>
<td>P4</td>
<td></td>
<td>P1</td>
</tr>
<tr>
<td></td>
<td>$12</td>
<td></td>
<td></td>
<td>P1</td>
<td>P5</td>
<td></td>
<td>P1</td>
</tr>
</tbody>
</table>

Diagram:
Register renaming in motion

```
lw   $6,0($10)  R I
add  $7,$6,$12  R
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP
lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP
```

Renamed instruction

<table>
<thead>
<tr>
<th>Renamed instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw      P1, 0($10)</td>
</tr>
<tr>
<td>add     P2, P1, $12</td>
</tr>
<tr>
<td>P1, 0($10)</td>
</tr>
<tr>
<td>P2, P1, $12</td>
</tr>
<tr>
<td>$10, $5, LOOP</td>
</tr>
</tbody>
</table>

Physical Register

```
<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>I</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Valid Value In use

```
P1 | 0 | 1 | P6
P2 | 0 | 1 | P7
P3 |    |   | P8
P4 |    |   | P9
P5 |    |   | P10
```
Register renaming in motion

1. lw  $6,0($10)
2. add $7,$6,$12
3. sw  $7,0($10)
4. addi $10,$10,8
5. bne $10,$5,LOOP
6. lw  $6,0($10)
7. add $7,$6,$12
8. sw  $7,0($10)
9. addi $10,$10,8
10. bne $10,$5,LOOP

Renamed instruction
1. lw  P1, 0($10)
2. add P2, P1, $12
3. sw  P2, 0($10)

Physical Register

<table>
<thead>
<tr>
<th>$5</th>
<th>$6</th>
<th>$7</th>
<th>$10</th>
<th>$12</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>P1</td>
<td>P2</td>
<td>P1</td>
<td>P1</td>
</tr>
</tbody>
</table>

Valid | Value | In use | Valid | Value | In use
P1   | 0     | 1     | P6   |       |       
P2   | 0     | 1     | P7   |       |       
P3   |       |       | P8   |       |       
P4   |       |       | P9   |       |       
P5   |       |       | P10  |       |       

27
Register renaming in motion

% lw $6,0($10)
% add $7,$6,$12
% sw $7,0($10)
% addi $10,$10,8
% bne $10,$5,LOOP
% lw $6,0($10)
% add $7,$6,$12
% sw $7,0($10)
% addi $10,$10,8
% bne $10,$5,LOOP

Renamed instruction

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw P1, 0($10)</td>
<td>$5</td>
<td>0</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td>add P2, P1, $12</td>
<td>$6</td>
<td>0</td>
<td>1</td>
<td>P7</td>
</tr>
<tr>
<td>sw P2, 0($10)</td>
<td>$7</td>
<td>0</td>
<td>1</td>
<td>P8</td>
</tr>
<tr>
<td>addi P3, $10, 8</td>
<td>$10</td>
<td>0</td>
<td>1</td>
<td>P9</td>
</tr>
<tr>
<td></td>
<td>$12</td>
<td></td>
<td></td>
<td>P10</td>
</tr>
</tbody>
</table>
Register renaming in motion

1. \texttt{lw} $6, 0($10)
2. \texttt{add} $7, 6, 12
3. \texttt{sw} $7, 0($10)
4. \texttt{addi} $10, 10, 8
5. \texttt{bne} $10, 5, LOOP
6. \texttt{lw} $6, 0($10)
7. \texttt{add} $7, 6, 12
8. \texttt{sw} $7, 0($10)
9. \texttt{addi} $10, 10, 8
10. \texttt{bne} $10, 5, LOOP

\begin{verbatim}
Renamed instruction
1  lw    P1, 0($10)
2  add   P2, P1, $12
3  sw    P2, 0($10)
4  addi  P3, $10, 8
5  bne   P3, $5, LOOP

<table>
<thead>
<tr>
<th>Renamed instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw      P1, 0($10)</td>
</tr>
<tr>
<td>add     P2, P1, $12</td>
</tr>
<tr>
<td>sw      P2, 0($10)</td>
</tr>
<tr>
<td>addi    P3, $10, 8</td>
</tr>
<tr>
<td>bne     P3, $5, LOOP</td>
</tr>
</tbody>
</table>

Physical Register

<table>
<thead>
<tr>
<th>Physical Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5</td>
</tr>
<tr>
<td>$6</td>
</tr>
<tr>
<td>$7</td>
</tr>
<tr>
<td>$10</td>
</tr>
<tr>
<td>$12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\end{verbatim}
Register renaming in motion

 lw  $6,0($10)  R  I  AR  LSQ  MEM  WB
 add $7,$6,$12   R  I  I  I  I
 sw  $7,0($10)   R  I  I  I  I
 addi $10,$10,8  R  I  INT  I  I
 bne $10,$5,LOOP  R  I  I  I  I
 lw  $6,0($10)  R  I  I  I  I
 add $7,$6,$12   R  I  I  I  I
 sw  $7,0($10)   R  I  I  I  I
 addi $10,$10,8  R  I  I  I  I
 bne $10,$5,LOOP  R  I  I  I  I

Renamed instruction
1 lw  P1, 0($10)
2 add P2, P1, $12
3 sw  P2, P1, $10
4 addi P3, $10, 8
5 bne P3, P5, LOOP
6 lw  P4, P3
7
8
9
10

Physical Register

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$6</td>
<td>P1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$7</td>
<td>P2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$10</td>
<td>P3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$12</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Valid  Value  In use  Valid  Value  In use

P1  1  1  P6
P2  0  1  P7
P3  0  1  P8
P4  0  1  P9
P5  P10
Register renaming in motion

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 lw P1, 0($10)</td>
<td>$5</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 add P2, P1, $12</td>
<td>$6</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 sw P2, 0($10)</td>
<td>$7</td>
<td>P1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 addi P3, $10, 8</td>
<td>$10</td>
<td>P5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 bne P3, $5, LOOP</td>
<td>$12</td>
<td>P3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 lw P4, 0(P3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 add P5, P1, $12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Register renaming in motion

- lw $6,0($10)
- add $7,$6,$12
- sw $7,0($10)
- addi $10,$10,8
- bne $10,$5,LOOP
- lw $6,0($10)
- add $7,$6,$12
- sw $7,0($10)
- addi $10,$10,8
- bne $10,$5,LOOP

Renamed instruction:
1. lw P1, 0($10)
2. add P2, P1, $12
3. sw P2, 0($10)
4. addi P3, P10, 8
5. bne P3, P5, LOOP
6. lw P4, P3
7. add P5, P1, $12
8. sw P5, P3
9. lw P4, P3
10. bne P3, P5, LOOP

Physical Register:
- $5
- $6
- $7
- $10
- $12

Valid Value In use
P1 1 1 P6
P2 1 1 P7
P3 1 1 P8
P4 0 1 P9
P5 0 1 P10
Register renaming in motion

```
  lw   $6, 0($10)
  add  $7, $6, $12
  sw   $7, 0($10)
  addi $10, $10, 8
  bne  $10, $5, LOOP
  lw   $6, 0($10)
  add  $7, $6, $12
  sw   $7, 0($10)
  addi $10, $10, 8
  bne  $10, $5, LOOP
```

```
   Physical Register

<table>
<thead>
<tr>
<th>R</th>
<th>I</th>
<th>AR</th>
<th>LSQ</th>
<th>MEM</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>INT</td>
</tr>
<tr>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>R</td>
<td>I</td>
<td>INT</td>
<td>WB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>I</td>
<td>I</td>
<td>BR</td>
<td>WB</td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>I</td>
<td>I</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

  Renamed instruction

1. lw    P1, 0($10)
2. add   P2, P1, $12
3. sw    P2, 0($10)
4. addi  P3, $10, 8
5. bne   P3, 5, LOOP
6. lw    P4, 0(P3)
7. add   P5, P1, $12
8. sw    P5, 0(P3)
9. addi  P6, P3, 8
10. bne   P5, $5, LOOP

  Valid  Value  In use  Valid  Value  In use
P1    1       1       P6    0       1
P2    1       1       P7    1       1
P3    1       1       P8    1       1
P4    0       1       P9    1       1
P5    0       1       P10  1       1
```
Register renaming in motion

lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP
lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP

Renamed instruction
1 lw   P1, 0($10)
2 add  P2, P1, $12
3 sw   P2, 0($10)
4 addi P2, $10, 8
5 bne  P3, $5, LOOP
6 lw   P4, 0($10)
7 add  P5, P1, $12
8 sw   P5, 0($10)
9 addi P6, P3, 8
10 bne P6, 0($10)
Register renaming in motion

```
① lw $6,0($10)
② add $7,$6,$12
③ sw $7,0($10)
④ addi $10,$10,8
⑤ bne $10,$5,LOOP
⑥ lw $6,0($10)
⑦ add $7,$6,$12
⑧ sw $7,0($10)
⑨ addi $10,$10,8
⑩ bne $10,$5,LOOP
```

### Renamed instruction

1. lw $6,0($10) → lw P1, 0($10)
2. add $7,$6,$12 → add P2, P1, $12
3. sw $7,0($10) → sw P2, 0($10)
4. addi $10,$10,8 → addi P3, $10, 8
5. bne $10,$5,LOOP → bne P3, 0($10)
6. lw $6,0($10) → lw P4, 0($10)
7. add $7,$6,$12 → add P5, P1, $12
8. sw $7,0($10) → sw P5, 0($10)
9. addi $10,$10,8 → addi P6, P3, 8
10. bne $10,$5,LOOP → bne P6, 0($10)

### Physical Register

<table>
<thead>
<tr>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5 P1</td>
<td>1</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td>$6 P1</td>
<td>1</td>
<td>1</td>
<td>P7</td>
</tr>
<tr>
<td>$7 P5</td>
<td>1</td>
<td>1</td>
<td>P8</td>
</tr>
<tr>
<td>$10 P3</td>
<td>1</td>
<td>1</td>
<td>P9</td>
</tr>
<tr>
<td>$12 P5</td>
<td>0</td>
<td>1</td>
<td>P10</td>
</tr>
</tbody>
</table>

### Valid Value In use

<table>
<thead>
<tr>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>P7</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>P8</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>P9</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>P10</td>
</tr>
</tbody>
</table>
Register renaming in motion

1. lw $6,0($10)
2. add $7,$6,$12
3. sw $7,0($10)
4. addi $10,$10,8
5. bne $10,$5,LOOP
6. lw $6,0($10)
7. add $7,$6,$12
8. sw $7,0($10)
9. addi $10,$10,8
10. bne $10,$5,LOOP

Renamed instruction

1. lw   P1, 0($10)
2. add   P2, P1, $12
3. sw   P2, 0($10)
4. addi  P3, $10, 8
5. bne  P3, $5, LOOP
6. lw   P4, 0(P3)
7. add   P5, P1, $12
8. sw   P5, 0(P3)
9. addi  P6, P3, 8
10. bne  P6, 0($10)

Physical Register

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 lw P1, 0($10)</td>
<td>$5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1 lw P1, 0($10)</td>
<td>$5</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2 add P2, P1, $12</td>
<td>$6</td>
<td>P1</td>
<td></td>
<td></td>
<td>2 add P2, P1, $12</td>
<td>$6</td>
<td>P1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 sw P2, 0($10)</td>
<td>$7</td>
<td>P5</td>
<td></td>
<td></td>
<td>3 sw P2, 0($10)</td>
<td>$7</td>
<td>P5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 addi P3, $10, 8</td>
<td>$10</td>
<td>P3</td>
<td></td>
<td></td>
<td>4 addi P3, $10, 8</td>
<td>$10</td>
<td>P3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 bne P3, $5, LOOP</td>
<td>$12</td>
<td></td>
<td></td>
<td></td>
<td>5 bne P3, $5, LOOP</td>
<td>$12</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Valid | Value | In use | Valid | Value | In use

P1 1 1 P6 1 1
P2 1 1 P7
P3 1 1 P8
P4 1 1 P9
P5 0 1 P10
Register renaming in motion

1. lw $6,0($10)
2. add $7,$6,$12
3. sw $7,0($10)
4. addi $10,$10,8
5. bne $10,$5,LOOP
6. lw $6,0($10)
7. add $7,$6,$12
8. sw $7,0($10)
9. addi $10,$10,8
10. bne $10,$5,LOOP

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw P1, 0($10)</td>
<td>$5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>P6</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>add P2, P1, $12</td>
<td>$6</td>
<td>P1</td>
<td></td>
<td></td>
<td>P2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>sw P2, 0($10)</td>
<td>$7</td>
<td>P5</td>
<td></td>
<td></td>
<td>P3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>addi P3, $10, 8</td>
<td>$10</td>
<td>P3</td>
<td></td>
<td></td>
<td>P4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>bne P3, $5, LOOP</td>
<td>$12</td>
<td></td>
<td>1</td>
<td></td>
<td>P5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>lw P4, 0(P3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P6</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>add P5, P1, $12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw P5, 0(P3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>addi P6, P3, 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bne P6, 0($10)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Register renaming in motion

```
 lw  $6,0($10)
 add $7,$6,$12
 sw  $7,0($10)
 addi $10,$10,8
 bne $10,$5,LOOP
 lw  $6,0($10)
 add $7,$6,$12
 sw  $7,0($10)
 addi $10,$10,8
 bne $10,$5,LOOP
```

Renamed instruction

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw P1, 0($10)</td>
<td>$5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>P1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>add P2, P1, $12</td>
<td>$6</td>
<td>1</td>
<td>1</td>
<td></td>
<td>P2</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>sw P2, 0($10)</td>
<td>$7</td>
<td>1</td>
<td>1</td>
<td></td>
<td>P5</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>addi P3, $10, 8</td>
<td>$10</td>
<td>1</td>
<td>1</td>
<td></td>
<td>P3</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>bne P9, 0, LOOP</td>
<td>$12</td>
<td>1</td>
<td>1</td>
<td></td>
<td>P5</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Register renaming in motion

lw $6,0($10)
add $7,$6,$12
sw $7,0($10)
addi $10,$10,8
bne $10,$5,LOOP
lw $6,0($10)
add $7,$6,$12
sw $7,0($10)
addi $10,$10,8
bne $10,$5,LOOP

Renamed instruction
1 lw P1, 0($10)
2 add P2, P1, $12
3 sw P2, 0($10)
4 addi P3, $10, 8
5 bne P5, P3, LOOP
6 lw P4, 0(P3)
7 add P5, P1, $12
8 sw P5, 0(P3)
9 addi P6, P3, 0
10 bne P6, 0($10)
Register renaming in motion

Takes 12 cycles to issue all instructions
Through data flow graph analysis

INT — 2 cycles for depending instruction to start
MEM — 4 cycles for the depending instruction to start
MUL/DIV — 4 cycles for the depending instruction to start
BR — 2 cycles to resolve

lw $6,0($10)
add $7,$6,$12
sw $7,0($10)
addi $10,$10,8
bne $10,$5,LOOP
lw $6,0($10)
add $7,$6,$12
sw $7,0($10)
addi $10,$10,8
bne $10,$5,LOOP
Register renaming in motion

1. `lw $10, 8($10)`
2. `addi $7, $7, 1`
3. `bne $10, $0, LOOP`
4. `lw $10, 8($10)`
5. `addi $7, $7, 1`
6. `bne $10, $0, LOOP`
7. `lw $10, 8($10)`
8. `addi $7, $7, 1`
9. `bne $10, $0, LOOP`

### Renamed instruction

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>lw</td>
<td>P1, 0($10)</td>
</tr>
</tbody>
</table>

### Physical Register

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$6</td>
<td></td>
<td>P1</td>
</tr>
<tr>
<td>$7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$12</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Valid | Value | In use | Valid | Value | In use |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0</td>
<td>1</td>
<td>P6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td></td>
<td></td>
<td>P7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td></td>
<td></td>
<td>P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td></td>
<td></td>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td></td>
<td></td>
<td>P10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Register renaming in motion

1. `lw $10, 8($10)`
2. `addi $7, $7, 1`
3. `bne $10, $0, LOOP`
4. `lw $10, 8($10)`
5. `addi $7, $7, 1`
6. `bne $10, $0, LOOP`
7. `lw $10, 8($10)`
8. `addi $7, $7, 1`
9. `bne $10, $0, LOOP`

### Renamed instruction

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>lw</code></td>
<td>$10, 0($10)</td>
</tr>
<tr>
<td>2</td>
<td><code>add</code></td>
<td>$2, $7, 1</td>
</tr>
</tbody>
</table>

### Physical Register

<table>
<thead>
<tr>
<th></th>
<th>Physical Register</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$5</td>
<td>$6</td>
<td>$7</td>
</tr>
<tr>
<td></td>
<td>$10</td>
<td></td>
<td>$10</td>
</tr>
<tr>
<td></td>
<td>$12</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td>P2</td>
<td>0</td>
<td>1</td>
<td>P7</td>
</tr>
<tr>
<td>P3</td>
<td></td>
<td></td>
<td>P8</td>
</tr>
<tr>
<td>P4</td>
<td></td>
<td></td>
<td>P9</td>
</tr>
<tr>
<td>P5</td>
<td></td>
<td></td>
<td>P10</td>
</tr>
</tbody>
</table>

### Table

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<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>
</tr>
<tr>
<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>
</tr>
<tr>
<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>
</tr>
<tr>
<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>
</tr>
</tbody>
</table>
Register renaming in motion

```
lw   $10, 8($10)   R I AR
addi $7, $7, 1       R I
bne $10, $0, LOOP
lw   $10, 8($10)   R
addi $7, $7, 1       R I
bne $10, $0, LOOP
lw   $10, 8($10)   R
addi $7, $7, 1       R I
bne $10, $0, LOOP
```

### Renamed instruction

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>lw P1, 0($10)</td>
</tr>
<tr>
<td>2</td>
<td>add P2, $7, 1</td>
</tr>
<tr>
<td>3</td>
<td>bne P1, $0, LOOP</td>
</tr>
</tbody>
</table>

### Physical Register

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>lw P1, 0($10)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>add P2, $7, 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>bne P1, $0, LOOP</td>
<td>0</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```
Register renaming in motion

1. `lw $10, 8($10)`
2. `add $7, $7, 1`
3. `bne $10, $0, LOOP`
4. `lw $10, 8($10)`
5. `add $7, $7, 1`
6. `bne $10, $0, LOOP`
7. `lw $10, 8($10)`
8. `add $7, $7, 1`
9. `bne $10, $0, LOOP`

Renamed instruction

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>lw P1, 0($10)</code></td>
<td>$5</td>
<td>0</td>
<td>1</td>
<td>P1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td><code>add P2, $7, 1</code></td>
<td>$6</td>
<td>0</td>
<td>1</td>
<td>P2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td><code>bne P1, $0, LOOP</code></td>
<td>$7</td>
<td>P2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td><code>lw P3, $0(P1)</code></td>
<td>$10</td>
<td>P3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>$12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Physical Register

<table>
<thead>
<tr>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td>P2</td>
<td>0</td>
<td>1</td>
<td>P7</td>
</tr>
<tr>
<td>P3</td>
<td>0</td>
<td>1</td>
<td>P8</td>
</tr>
<tr>
<td>P4</td>
<td></td>
<td></td>
<td>P9</td>
</tr>
<tr>
<td>P5</td>
<td></td>
<td></td>
<td>P10</td>
</tr>
</tbody>
</table>

Valid: 1 = Available; 0 = In use
Register renaming in motion

1. `lw  $10, 8($10)`
2. `addi $7, $7, 1`
3. `bne $10, $0, LOOP`
4. `lw  $10, 8($10)`
5. `addi $7, $7, 1`
6. `bne $10, $0, LOOP`
7. `lw  $10, 8($10)`
8. `addi $7, $7, 1`
9. `bne $10, $0, LOOP`

Renamed instruction

1. `lw    P1, 0($10)`
2. `add   P2, $7, 1`
3. `bne   P1, $0, LOOP`
4. `lw    P3, 0(P1)`
5. `add   P4, P2, 1`

Physical Register

<table>
<thead>
<tr>
<th>Instruction</th>
<th>R</th>
<th>I</th>
<th>AR</th>
<th>LSQ</th>
<th>MEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R</td>
<td>I</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>R</td>
<td>I</td>
<td></td>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>I</td>
<td></td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>I</td>
<td></td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>I</td>
<td></td>
<td>I</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>P3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P5</td>
<td>P4</td>
<td>1</td>
</tr>
<tr>
<td>P6</td>
<td>P3</td>
<td>1</td>
</tr>
<tr>
<td>P7</td>
<td>P5</td>
<td>1</td>
</tr>
<tr>
<td>P8</td>
<td>P5</td>
<td>1</td>
</tr>
<tr>
<td>P9</td>
<td>P5</td>
<td>1</td>
</tr>
<tr>
<td>P10</td>
<td>P5</td>
<td>1</td>
</tr>
</tbody>
</table>
Register renaming in motion

Renamed instruction:
1. lw P1, 0($10)
2. add P2, $7, 1
3. bne P1, $0, LOOP
4. lw P3, 0(P1)
5. add P4, P2, 1
6. bne P3, $0, LOOP
7.  
8.  
9.  
10.  

Physical Register:

<table>
<thead>
<tr>
<th>Physical Register</th>
<th>Renamed instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5</td>
<td>lw P1, 0($10)</td>
</tr>
<tr>
<td>$6</td>
<td>add P2, $7, 1</td>
</tr>
<tr>
<td>$7</td>
<td>bne P1, $0, LOOP</td>
</tr>
<tr>
<td>$10</td>
<td>lw P3, 0(P1)</td>
</tr>
<tr>
<td>$12</td>
<td>add P4, P2, 1</td>
</tr>
<tr>
<td>$15</td>
<td>bne P3, $0, LOOP</td>
</tr>
</tbody>
</table>

Valid | Value | In use | Valid | Value | In use
P1 | 1 | 1 | P6 | 1 | 1
P2 | 1 | 1 | P7 | 1 | 1
P3 | 0 | 1 | P8 | 0 | 1
P4 | 0 | 1 | P9 | 0 | 1
P5 | 1 | 1 | P10 | 1 | 1

Physical Register:

<table>
<thead>
<tr>
<th>Physical Register</th>
<th>Renamed instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5</td>
<td>lw P1, 0($10)</td>
</tr>
<tr>
<td>$6</td>
<td>add P2, $7, 1</td>
</tr>
<tr>
<td>$7</td>
<td>bne P1, $0, LOOP</td>
</tr>
<tr>
<td>$10</td>
<td>lw P3, 0(P1)</td>
</tr>
<tr>
<td>$12</td>
<td>add P4, P2, 1</td>
</tr>
<tr>
<td>$15</td>
<td>bne P3, $0, LOOP</td>
</tr>
</tbody>
</table>

Valid | Value | In use | Valid | Value | In use
P1 | 1 | 1 | P6 | 1 | 1
P2 | 1 | 1 | P7 | 1 | 1
P3 | 0 | 1 | P8 | 0 | 1
P4 | 0 | 1 | P9 | 0 | 1
P5 | 1 | 1 | P10 | 1 | 1
Register renaming in motion

1. \( \text{lw} \ P1, 0(P10) \)
2. \( \text{add} \ P2, S7, 1 \)
3. \( \text{bne} \ P1, 0, \text{LOOP} \)
4. \( \text{lw} \ P3, 0(P1) \)
5. \( \text{add} \ P4, P2, 1 \)
6. \( \text{bne} \ P3, 0, \text{LOOP} \)
7. \( \text{lw} \ P5, 0(P3) \)

Renamed instruction

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid Value In use</th>
<th>Physical Register</th>
<th>Valid Value In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( \text{lw} \ P1, 0(P10) )</td>
<td>$5</td>
<td>P1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>( \text{add} \ P2, S7, 1 )</td>
<td>$6</td>
<td>P2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>( \text{bne} \ P1, 0, \text{LOOP} )</td>
<td>$7</td>
<td>P3</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>( \text{lw} \ P3, 0(P1) )</td>
<td>$10</td>
<td>P4</td>
<td>P4</td>
</tr>
<tr>
<td>5</td>
<td>( \text{add} \ P4, P2, 1 )</td>
<td>$12</td>
<td>P5</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>( \text{bne} \ P3, 0, \text{LOOP} )</td>
<td></td>
<td>P5</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>( \text{lw} \ P5, 0(P3) )</td>
<td></td>
<td>P5</td>
<td>1</td>
</tr>
</tbody>
</table>

\( R \)
Register renaming in motion

1. lw $10, 8($10)
2. addi $7, $7, 1
3. bne $10, $0, LOOP
4. lw $10, 8($10)
5. addi $7, $7, 1
6. bne $10, $0, LOOP
7. lw $10, 8($10)
8. addi $7, $7, 1
9. bne $10, $0, LOOP
10. lw $10, 8($10)

Physical Register

- $5
- $6
- $7
- $10
- $12

Renamed instruction

1. lw P1, 0($10)
2. add P2, $7, 1
3. bne P1, $0, LOOP
4. lw P3, 0(P1)
5. add P4, P2, 1
6. bne P3, $0, LOOP
7. lw P5, 0(P3)
8. add P6, P4, 1
9. 
10. 

Valid | Value | In use
---|---|---
P1 | 1 | 1
P2 | 1 | 1
P3 | 0 | 1
P4 | 0 | 1
P5 | 0 | 1
P6 | | |
P7 | | |
P8 | | |
P9 | | |
P10 | | |
Register renaming in motion

1. lw $10, 8($10)
2. addi $7, $7, 1
3. bne $10, $0, LOOP
4. lw $10, 8($10)
5. addi $7, $7, 1
6. bne $10, $0, LOOP
7. lw $10, 8($10)
8. addi $7, $7, 1
9. bne $10, $0, LOOP

Renamed instruction
1. lw P1, 0($10)
2. add P2, $7, 1
3. bne P1, $0, LOOP
4. lw P3, 0(P1)
5. add P4, P2, 1
6. bne P3, $0, LOOP
7. lw P5, 0(P3)
8. add P6, P4, 1
9. bne P5, $0, LOOP

Physical Register

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 lw P1, 0($10)</td>
<td>$5</td>
</tr>
<tr>
<td>2 add P2, $7, 1</td>
<td>$6</td>
</tr>
<tr>
<td>3 bne P1, $0, LOOP</td>
<td>$7</td>
</tr>
<tr>
<td>4 lw P3, 0(P1)</td>
<td>$10</td>
</tr>
<tr>
<td>5 add P4, P2, 1</td>
<td>$12</td>
</tr>
</tbody>
</table>

Valid | Value | In use | Valid | Value | In use
---|-------|--------|-------|-------|--------
P1 | 1     | 1      | P6 | 0     | 1      
P2 | 1     | 1      | P7 |       |        
P3 | 0     | 1      | P8 |       |        
P4 | 0     | 1      | P9 |       |        
P5 | 0     | 1      | P10|       |        

Register renaming in motion

1. lw $10, 8($10)
2. addi $7, $7, 1
3. bne $10, $0, LOOP
4. lw $10, 8($10)
5. addi $7, $7, 1
6. bne $10, $0, LOOP
7. lw $10, 8($10)
8. addi $7, $7, 1
9. bne $10, $0, LOOP
10. lw $10, 8($10)

Renamed instruction

1 lw P1, 0($10)
2 add P2, $7, 1
3 bne P1, $0, LOOP
4 lw P3, 0(P1)
5 add P4, P2, 1
6 bne P3, $0, LOOP
7 lw P5, 0(P3)
8 add P6, P4, 1
9 bne P5, $0, LOOP

Physical Register

<table>
<thead>
<tr>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>1</td>
<td>1</td>
<td>P6</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P2</td>
<td>1</td>
<td>1</td>
<td>P7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>0</td>
<td>1</td>
<td>P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>0</td>
<td>1</td>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td>0</td>
<td>1</td>
<td>P10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Register renaming in motion

1. lw    $10, 8($10)
2. addi  $7, $7, 1
3. bne   $10, $0, LOOP
4. lw    $10, 8($10)
5. addi  $7, $7, 1
6. bne   $10, $0, LOOP
7. lw    $10, 8($10)
8. addi  $7, $7, 1
9. bne   $10, $0, LOOP
Super Scalar
Superscalar

• Since we have more functional units now, we should fetch/decode more instructions each cycle so that we can have more instructions to issue!

• Super-scalar: fetch/decode/issue more than one instruction each cycle
  • Fetch width: how many instructions can the processor fetch/decode each cycle
  • Issue width: how many instructions can the processor issue each cycle
Overview of a processor supporting register renaming

What if we widen the pipeline to fetch/issue two instructions at the same time?
2-issue RR processor in motion

1. lw  $6,0($10)
2. add $7,$$6,$$12
3. sw $7,0($10)
4. addi $10,$$10,8
5. bne $10,$$5,LOOP
6. lw  $6,0($10)
7. add $7,$$6,$$12
8. sw $7,0($10)
9. addi $10,$$10,8
10. bne $10,$$5,LOOP
2-issue RR processor in motion

1. lw $6, 0($10)
2. add $7, $6, $12
3. sw $7, 0($10)
4. addi $10, $10, 8
5. bne $10, $5, LOOP
6. lw $6, 0($10)
7. add $7, $6, $12
8. sw $7, 0($10)
9. addi $10, $10, 8
10. bne $10, $5, LOOP

Renamed instruction

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw P1, 0($10)</td>
<td>$5</td>
<td>0</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td>add P2, P1, $12</td>
<td>$6</td>
<td>0</td>
<td>1</td>
<td>P7</td>
</tr>
<tr>
<td>sw P2, 0($10)</td>
<td>$7</td>
<td>0</td>
<td>1</td>
<td>P8</td>
</tr>
<tr>
<td>addi P3, $10, 8</td>
<td>$10</td>
<td>0</td>
<td>1</td>
<td>P9</td>
</tr>
<tr>
<td></td>
<td>$12</td>
<td>0</td>
<td>1</td>
<td>P10</td>
</tr>
</tbody>
</table>
2-issue RR processor in motion

- lw $6, 0($10)
- add $7, $6, $12
- sw $7, 0($10)
- addi $10, $10, 8
- bne $10, $5, LOOP
- lw $6, 0($10)
- add $7, $6, $12
- sw $7, 0($10)
- addi $10, $10, 8
- bne $10, $5, LOOP

Renamed instruction
1. lw P1, 0($10)
2. add P2, P1, $12
3. sw P2, 0($10)
4. addi P3, $10, 8
5. bne P3, $5, LOOP
6. lw P4, 0(P3)
7. sw P4, 0(P5)
8. addi P5, $5, LOOP
9. bne P5, $3, LOOP
10. lw P5, 0($10)

Physical Register
- $5  P1
- $6  P1
- $7  P2
- $10 P3
- $12 P4

Valid Value In use
- P1  0  1  P6
- P2  0  1  P7
- P3  0  1  P8
- P4  0  1  P9
- P5  0  1  P10
2-issue RR processor in motion

1. `lw $6, 0($10)`
2. `add $7, $6, $12`
3. `sw $7, 0($10)`
4. `addi $10, $10, 8`
5. `bne $10, $5, LOOP`
6. `lw $6, 0($10)`
7. `add $7, $6, $12`
8. `sw $7, 0($10)`
9. `addi $10, $10, 8`
10. `bne $10, $5, LOOP`

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 lw P1, 0($10)</td>
<td>$5</td>
</tr>
<tr>
<td>2 add P2, P1, $12</td>
<td>$6</td>
</tr>
<tr>
<td>3 sw P2, 0($10)</td>
<td>$7</td>
</tr>
<tr>
<td>4 addi P3, $10, 8</td>
<td>$10</td>
</tr>
<tr>
<td>5 bne P3, $5, LOOP</td>
<td>$12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P5</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2-issue RR processor in motion

lw  $6,0($10)
add $7,$6,$12
sw  $7,0($10)
add $10,$10,8
bne $10,$5,LOOP
lw  $6,0($10)
add $7,$6,$12
sw  $7,0($10)
add $10,$10,8
bne $10,$5,LOOP

Renamed instruction

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw P1, 0($10)</td>
<td>$5</td>
<td>0</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td>add P2, P1, $12</td>
<td>$6</td>
<td>0</td>
<td>1</td>
<td>P7</td>
</tr>
<tr>
<td>sw P2, 0($10)</td>
<td>$7</td>
<td>1</td>
<td>1</td>
<td>P8</td>
</tr>
<tr>
<td>addi P3, $10, 8</td>
<td>$10</td>
<td>0</td>
<td>1</td>
<td>P9</td>
</tr>
<tr>
<td>bne P3, $5, LOOP</td>
<td>$12</td>
<td>0</td>
<td>1</td>
<td>P10</td>
</tr>
</tbody>
</table>
2-issue RR processor in motion

Renamed instruction

1 lw P1, 0($10)
2 add P2, P1, $12
3 sw P2, 0($10)
4 addi P3, $10, 8
5 bne P3, $5, LOOP
6 lw P4, 0(P3)
7 add P5, P1, $12
8 sw P5, 0(P3)
9 addi P6, P3, 8
10 bne P6, 0($10)

Physical Register

- $5
- $6 P1
- $7 P5
- $10 P5
- $12

Valid Value In use
P1 1 1 P6
P2 0 1 P7
P3 1 1 P8
P4 0 1 P9
P5 0 1 P10

Valid Value In use

2-issue RR processor in motion

1. lw   $6,0($10)
2. add  $7,$6,$12
3. sw   $7,0($10)
4. addi $10,$10,8
5. bne  $10,$5,LOOP
6. lw   $6,0($10)
7. add  $7,$6,$12
8. sw   $7,0($10)
9. addi $10,$10,8
10. bne $10,$5,LOOP
2-issue RR processor in motion

- `lw $6,0($10)`
- `add $7,$6,$12`
- `sw $7,0($10)`
- `addi $10,$10,8`
- `bne $10,$5,LOOP`
- `lw $6,0($10)`
- `add $7,$6,$12`
- `sw $7,0($10)`
- `addi $10,$10,8`
- `bne $10,$5,LOOP`

Renamed instruction:
1. `lw P1, 0($10)`
2. `add P2, P1, $12`
3. `sw P2, 0($10)`
4. `addi P3, $10, 8`
5. `bne P9, $5, LOOP`
6. `lw P4, 0(P3)`
7. `add P5, P1, $12`
8. `sw P5, 0(P3)`
9. `addi P6, P3, 8`
10. `bne P6, 0($10)`

Physical Register:
- $5
- $6
- $7
- $10
- $12

Valid | Value | In use | Valid | Value | In use
--- | --- | --- | --- | --- | ---

P1 | 1 | 1 | P6
P2 | 1 | 1 | P7
P3 | 1 | 1 | P8
P4 | 0 | 1 | P9
P5 | 0 | 1 | P10
2-issue RR processor in motion

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw P1, 0($10)</td>
<td>$5 P1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add P2, P1, $12</td>
<td>$6 P1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw P2, 0($10)</td>
<td>$7 P5</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>addi P3, $10, 8</td>
<td>$10 P5</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bne P5, $5, LOOP</td>
<td>$12 P3</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Physical Register:
- $5
- $6
- $7
- $10
- $12
2-issue RR processor in motion

1. \text{lw} $6,0($10)
2. \text{add} $7,$6,$12
3. \text{sw} $7,0($10)
4. \text{addi} $10,$10,8
5. \text{bne} $10,$5,LOOP
6. \text{lw} $6,0($10)
7. \text{add} $7,$6,$12
8. \text{sw} $7,0($10)
9. \text{addi} $10,$10,8
10. \text{bne} $10,$5,LOOP

Renamed instruction

1. \text{lw} P1, 0($10)
2. \text{add} P2, P1, $12
3. \text{sw} P2, 0($10)
4. \text{addi} P3, $10, 8
5. \text{bne} P3, $5, LOOP
6. \text{lw} P4, 0(P3)
7. \text{add} P5, P1, $12
8. \text{sw} P5, 0(P3)
9. \text{addi} P6, P3, 0
10. \text{bne} P6, 0($10)

Physical Register

<table>
<thead>
<tr>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5</td>
<td>P5</td>
<td></td>
<td>P1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$6</td>
<td>P1</td>
<td>1</td>
<td>P2</td>
<td>1</td>
<td>P6</td>
</tr>
<tr>
<td>$7</td>
<td>P5</td>
<td>1</td>
<td>P3</td>
<td>1</td>
<td>P7</td>
</tr>
<tr>
<td>$10</td>
<td>P3</td>
<td>1</td>
<td>P4</td>
<td>1</td>
<td>P8</td>
</tr>
<tr>
<td>$12</td>
<td>P6</td>
<td>0</td>
<td>P5</td>
<td>0</td>
<td>P9</td>
</tr>
</tbody>
</table>

Valid Value In use Valid Value In use

P1 1 1 P6
P2 1 1 P7
P3 1 1 P8
P4 1 1 P9
P5 0 1 P10
2-issue RR processor in motion

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw P1, 0($10)</td>
<td>$5</td>
</tr>
<tr>
<td>add P2, P1, $12</td>
<td>$6</td>
</tr>
<tr>
<td>sw P2, 0($10)</td>
<td>$7</td>
</tr>
<tr>
<td>addi P3, $10, 8</td>
<td>$10</td>
</tr>
<tr>
<td>bne P9, $5, LOOP</td>
<td>$12</td>
</tr>
<tr>
<td>lw P4, 0(P3)</td>
<td></td>
</tr>
<tr>
<td>add P5, P1, $12</td>
<td></td>
</tr>
<tr>
<td>sw P5, 0(P3)</td>
<td></td>
</tr>
<tr>
<td>addi P6, P3, 0</td>
<td></td>
</tr>
<tr>
<td>bne P6, 0($10)</td>
<td></td>
</tr>
</tbody>
</table>

```
① lw   $6,0($10)
② add  $7,$6,$12
③ sw   $7,0($10)
④ addi $10,$10,8
⑤ bne  $10,$5,LOOP
⑥ lw   $6,0($10)
⑦ add  $7,$6,$12
⑧ sw   $7,0($10)
⑨ addi $10,$10,8
⑩ bne  $10,$5,LOOP
```
2-issue RR processor in motion

- `lw  $6,0($10)`
- `add  $7,$6,$12`
- `sw  $7,0($10)`
- `addi $10,$10,8`
- `bne $10,$5,LOOP`
- `lw  $6,0($10)`
- `add  $7,$6,$12`
- `sw  $7,0($10)`
- `addi $10,$10,8`
- `bne $10,$5,LOOP`

### Renamed instruction table

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>lw  P1, 0($10)   </code></td>
<td>$5</td>
</tr>
<tr>
<td><code>add  P2, P1, $12  </code></td>
<td>P1</td>
</tr>
<tr>
<td><code>sw  P2, 0($10)   </code></td>
<td>$6</td>
</tr>
<tr>
<td><code>addi P3, $10, 8   </code></td>
<td>P1</td>
</tr>
<tr>
<td><code>bne P3, $5, LOOP  </code></td>
<td>$7</td>
</tr>
<tr>
<td><code>lw  P4, 0(P3)  </code></td>
<td>P5</td>
</tr>
<tr>
<td><code>add  P5, P1, $12 </code></td>
<td>P5</td>
</tr>
<tr>
<td><code>sw  P5, 0(P3)  </code></td>
<td>$10</td>
</tr>
<tr>
<td><code>addi P6, P3, 0   </code></td>
<td>P3</td>
</tr>
<tr>
<td><code>bne P6, 0($10)  </code></td>
<td>$12</td>
</tr>
</tbody>
</table>

### Physical Register Table

<table>
<thead>
<tr>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>1</td>
<td>1</td>
<td>P6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>1</td>
<td>1</td>
<td>P7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>1</td>
<td>1</td>
<td>P8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>1</td>
<td>1</td>
<td>P9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td>1</td>
<td>1</td>
<td>P10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2-issue RR processor in motion

- LW $6,0($10)
- Add $7, $6, $12
- SW $7,0($10)
- Addi $10, $10, 8
- BNE $10, $5, LOOP
- LW $6,0($10)
- Add $7, $6, $12
- SW $7,0($10)
- Addi $10, $10, 8
- BNE $10, $5, LOOP

Takes 10 cycles to issue all instructions.
2-issue RR processor in motion

1. lw $6,0($10)
2. add $7,$6,$12
3. sw $7,0($10)
4. addi $10,$10,8
5. bne $10,$5,LOOP
6. lw $6,0($10)
7. add $7,$6,$12
8. sw $7,0($10)
9. addi $10,$10,8
10. bne $10,$5,LOOP

Renamed instruction:
1. lw P1, 0($10)
2. add P2, P1, $12
3. sw P2, 0($10)
4. addi P3, $10, 8
5. bne P4, 0($P3)
6. lw P4, 0($P3)
7. add P5, P1, $12
8. sw P5, 0($P3)
9. addi P6, P3, 0
10. bne P6, 0($10)
2-issue RR processor in motion

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw  $6,0($10)</td>
<td>$5</td>
<td>P1</td>
<td>1</td>
<td>1</td>
<td>P6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>add $7,$6,$12</td>
<td>$6</td>
<td>P1</td>
<td>1</td>
<td>1</td>
<td>P7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw  $7,0($10)</td>
<td>$7</td>
<td>P5</td>
<td>1</td>
<td>1</td>
<td>P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>addi $10,$10,8</td>
<td>$10</td>
<td>P5</td>
<td>1</td>
<td>1</td>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bne $10,$5,LOOP</td>
<td>$12</td>
<td>P3</td>
<td>1</td>
<td>1</td>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw  $6,0($10)</td>
<td>$5</td>
<td>P1</td>
<td>1</td>
<td>1</td>
<td>P6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>add $7,$6,$12</td>
<td>$6</td>
<td>P1</td>
<td>1</td>
<td>1</td>
<td>P7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sw  $7,0($10)</td>
<td>$7</td>
<td>P5</td>
<td>1</td>
<td>1</td>
<td>P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>addi $10,$10,8</td>
<td>$10</td>
<td>P5</td>
<td>1</td>
<td>1</td>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bne $10,$5,LOOP</td>
<td>$12</td>
<td>P3</td>
<td>1</td>
<td>1</td>
<td>P9</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Physical Register:

- **P1**: Valid Value In use
- **P2**: Valid Value In use
- **P3**: Valid Value In use
- **P4**: Valid Value In use
- **P5**: Valid Value In use
- **P6**: Valid Value In use

Diagram:
- **R**: Read
- **I**: Issue
- **AR**: Address Resolve
- **AQ**: Address Resolve
- **MEM**: Memory
- **WB**: Write Back
- **BR**: Branch
- **INT**: Interrupt
- **WB**: Write Back

Instructions:
- lw  $6,0($10)
- add $7,$6,$12
- sw  $7,0($10)
- addi $10,$10,8
- bne $10,$5,LOOP
- lw  $6,0($10)
- add $7,$6,$12
- sw  $7,0($10)
- addi $10,$10,8
- bne $10,$5,LOOP
2-issue RR processor in motion

```
1 lw $6,0($10)
2 add $7,$6,$12
3 sw $7,0($10)
4 addi $10,$10,8
5 bne $10,$5,LOOP
6 lw $6,0($10)
7 add $7,$6,$12
8 sw $7,0($10)
9 addi $10,$10,8
10 bne $10,$5,LOOP
```

What if exception occurs here?
Speculative Execution

- Any execution of an instruction before any prior instruction finishes is considered as **speculative execution**
- Because it’s speculative, we need to preserve the capability to restore to the states before it’s executed
  - Branch mis-prediction
  - Exceptions
Reorder Buffer (ROB)
Reorder buffer/Commit stage

- Reorder buffer — a buffer keep track of the program order of instructions
  - Can be combined with IQ or physical registers — make either as a circular queue
- Commit stage — should the outcome of an instruction be realized
  - An instruction can only leave the pipeline if all its previous are committed
  - If any prior instruction failed to commit, the instruction should yield its ROB entry, restore all its architectural changes
Renamed instruction

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>lw</td>
<td>P1, 0($10)</td>
</tr>
<tr>
<td>2</td>
<td>add</td>
<td>P2, P1, $12</td>
</tr>
</tbody>
</table>

Physical Register

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$6</td>
<td>P1</td>
</tr>
<tr>
<td>2</td>
<td>$7</td>
<td>P2</td>
</tr>
<tr>
<td>3</td>
<td>$10</td>
<td>P3</td>
</tr>
<tr>
<td>4</td>
<td>$12</td>
<td>P4</td>
</tr>
</tbody>
</table>

Valid Value In use

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2-issue RR processor in motion

- lw   $6,0($10)  
- add  $7,$6,$12 
- sw   $7,0($10)  
- addi $10,$10,8 
- bne  $10,$5,LOOP 
- lw   $6,0($10)  
- add  $7,$6,$12 
- sw   $7,0($10)  
- addi $10,$10,8 
- bne  $10,$5,LOOP
2-issue RR processor in motion

Renamed instruction
1 lw P1, 0($10)
2 add P2, P1, $12
3 sw P2, 0($10)
4 addi P3, $10, 8

Physical Register

$5
$6 P1
$7 P2
$10 P3
$12

Valid Value In use
P1 0 1 P6
P2 0 1 P7
P3 0 1 P8
P4
P5
P9
P10

Valid Value In use

head

tail
2-issue RR processor in motion

- lw $6,0($10)
- add $7,$6,$12
- sw $7,0($10)
- addi $10,$10,8
- bne $10,$5,LOOP
- lw $6,0($10)
- add $7,$6,$12
- sw $7,0($10)
- addi $10,$10,8
- bne $10,$5,LOOP

Renamed instruction:
1. lw P1, 0($10)
2. add P2, P1, $12
3. sw P2, 0($10)
4. addi P3, $10, 8
5. bne P3, $5, LOOP
6. lw P4, 0(P3)
7. 
8. 
9. 
10.

Physical Register:
- $5
- $6
- $7
- $10
- $12

Valid Value In use
P1 0 1 P6
P2 0 1 P7
P3 0 1 P8
P4 0 1 P9
P5 0 P10

Valid Value In use
head

tail
2-issue RR processor in motion

1. `lw $6,0($10)`
2. `add $7,$6,$12`
3. `sw $7,0($10)`
4. `addi $10,$10,8`
5. `bne $10,$5,LOOP`
6. `lw $6,0($10)`
7. `add $7,$6,$12`
8. `sw $7,0($10)`
9. `addi $10,$10,8`
10. `bne $10,$5,LOOP`

### Renamed instruction

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>lw P1, 0($10)</code></td>
</tr>
<tr>
<td>2</td>
<td><code>add P2, P1, $12</code></td>
</tr>
<tr>
<td>3</td>
<td><code>sw P2, 0($10)</code></td>
</tr>
<tr>
<td>4</td>
<td><code>addi P3, $10, 8</code></td>
</tr>
<tr>
<td>5</td>
<td><code>bne P3, $5, LOOP</code></td>
</tr>
<tr>
<td>6</td>
<td><code>lw P4, 0(P3)</code></td>
</tr>
<tr>
<td>7</td>
<td><code>add P5, P1, $12</code></td>
</tr>
<tr>
<td>8</td>
<td><code>sw P5, 0(P3)</code></td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

### Physical Register

<table>
<thead>
<tr>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P5</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>P6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P10</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Valid Value

- **Valid**: Indicates whether the register is valid.
- **Value**: Value of the register.
- **In use**: Indicates whether the register is in use.

**Head**: `lw P1, 0($10)`

**Tail**: `sw P5, 0(P3)`
2-issue RR processor in motion

lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP
lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP

Renamed instruction

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>lw</td>
</tr>
<tr>
<td>2</td>
<td>add</td>
</tr>
<tr>
<td>3</td>
<td>sw</td>
</tr>
<tr>
<td>4</td>
<td>addi</td>
</tr>
<tr>
<td>5</td>
<td>bne</td>
</tr>
<tr>
<td>6</td>
<td>lw</td>
</tr>
<tr>
<td>7</td>
<td>add</td>
</tr>
<tr>
<td>8</td>
<td>sw</td>
</tr>
<tr>
<td>9</td>
<td>addi</td>
</tr>
<tr>
<td>10</td>
<td>bne</td>
</tr>
</tbody>
</table>

Physical Register

<table>
<thead>
<tr>
<th></th>
<th>R</th>
<th>I</th>
<th>AR</th>
<th>AQ</th>
<th>MEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>2</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>4</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>INT</td>
<td>WB</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>7</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>8</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>9</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
<tr>
<td>10</td>
<td>R</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
</tr>
</tbody>
</table>

Valid  Value  In use
P1  0  1  P6
P2  0  1  P7
P3  1  1  P8
P4  0  1  P9
P5  0  1  P10

Valid  Value  In use
P1  0  1  P6
P2  0  1  P7
P3  1  1  P8
P4  0  1  P9
P5  0  1  P10

head

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>lw</td>
</tr>
<tr>
<td>2</td>
<td>add</td>
</tr>
<tr>
<td>3</td>
<td>sw</td>
</tr>
<tr>
<td>4</td>
<td>addi</td>
</tr>
<tr>
<td>5</td>
<td>bne</td>
</tr>
<tr>
<td>6</td>
<td>lw</td>
</tr>
<tr>
<td>7</td>
<td>add</td>
</tr>
<tr>
<td>8</td>
<td>sw</td>
</tr>
<tr>
<td>9</td>
<td>addi</td>
</tr>
<tr>
<td>10</td>
<td>bne</td>
</tr>
</tbody>
</table>
2-issue RR processor in motion

lw $6,0($10)
add $7,$6,$12
sw $7,0($10)
addi $10,$10,8
bne $10,$5,LOOP
lw $6,0($10)
add $7,$6,$12
sw $7,0($10)
addi $10,$10,8
bne $10,$5,LOOP
2-issue RR processor in motion

- lw $6, 0($10)
- add $7, $6, $12
- sw $7, 0($10)
- addi $10, $10, 8
- bne $10, $5, LOOP
- lw $6, 0($10)
- add $7, $6, $12
- sw $7, 0($10)
- addi $10, $10, 8
- bne $10, $5, LOOP

Renamed instruction

1. lw P1, 0($10)
2. add P2, P1, $12
3. sw P2, 0($10)
4. addi P3, $10, 8
5. bne P3, $5, LOOP
6. lw P4, 0($P3)
7. add P5, P1, $12
8. sw P5, 0($P3)
9. addi P6, P3, 8
10. bne P6, 0($10)

Physical Register

- $5
- $6
- $7
- $10
- $12

Valid | Value | In use
-----|-------|-------
P1 | 1 | 1
P2 | 0 | 1
P3 | 1 | 1
P4 | 0 | 1
P5 | 0 | 1

head

tail
## 2-issue RR processor in motion

### Renamed instruction
1. `lw P1, 0($10)`  
2. `add P2, P1, $12`  
3. `sw P2, 0($10)`  
4. `addi P3, $10, 8`  
5. `bne P3, $5, LOOP`  
6. `lw P4, 0($3)`  
7. `add P5, P1, $12`  
8. `sw P5, 0($3)`  
9. `addi P6, P3, 8`  
10. `bne P6, 0($10)`

### Physical Register
- **$5**
- **$6**
- **$7**
- **$10**
- **$12**

### Valid Value In use
- **P1**
- **P2**
- **P3**
- **P4**
- **P5**

### In motion
- **wb**
- **br**
- **aq**
- **int**
- **mem**
- **wb**
- **br**
- **aq**
- **int**
- **mem**

### Valid Value In use

---

**head**

**tail**
2-issue RR processor in motion

lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP
lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP

Renamed instruction

1 lw    P1, 0($10)
2 add   P2, P1, $12
3 sw    P2, 0($10)
4 addi  P3, $10, 8
5 bne   P3, $5, LOOP
6 lw    P4, 0(P3)
7 add   P5, P1, $12
8 sw    P5, 0(P3)
9 addi  P6, P3, 8
10 bne  P6, 0($10)
2-issue RR processor in motion

lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP
lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP

Renamed instruction
1 lw    P1, 0($10)
2 add   P2, P1, $12
3 sw    P2, 0($10)
4 addi  P3, $10, 8
5 bne   P3, $5, LOOP
6 lw    P4, 0(P3)
7 add   P5, P1, $12
8 sw    P5, 0(P3)
9 addi  P6, P3, 8
10 bne  P6, 0($10)
2-issue RR processor in motion

1. lw   $6,0($10)
2. add $7,$6,$12
3. sw   $7,0($10)
4. addi $10,$10,8
5. bne $10,$5,LOOP
6. lw   $6,0($10)
7. add $7,$6,$12
8. sw   $7,0($10)
9. addi $10,$10,8
10. bne $10,$5,LOOP
2-issue RR processor in motion

úmeros de instrucción:

1. lw $6, 0($10)
2. add $7, $6, $12
3. sw $7, 0($10)
4. addi $10, $10, 8
5. bne $10, $5, LOOP
6. lw $6, 0($10)
7. add $7, $6, $12
8. sw $7, 0($10)
9. addi $10, $10, 8
10. bne $10, $5, LOOP

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$5</td>
</tr>
<tr>
<td>2</td>
<td>$6</td>
</tr>
<tr>
<td>3</td>
<td>$7</td>
</tr>
<tr>
<td>4</td>
<td>$10</td>
</tr>
<tr>
<td>5</td>
<td>$12</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>9</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Physical Register</th>
<th>Valid</th>
<th>Value</th>
<th>In use</th>
</tr>
</thead>
<tbody>
<tr>
<td>$6</td>
<td>P1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$7</td>
<td>P5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$10</td>
<td>P3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>$12</td>
<td>P5</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Head: $5
Tail: $12

Diagram:

- R: Register
- I: Instruction
- AR: Address Register
- AQ: Address Queue
- MEM: Memory
- WB: Write Back
- C: Control

Valid Value In use
P1 1 1 P6
P2 1 1 P7
P3 1 1 P8
P4 1 1 P9
P5 1 1 P10
2-issue RR processor in motion

- lw $6,0($10)
- add $7,$6,$12
- sw $7,0($10)
- addi $10,$10,8
- bne $10,$5,LOOP
- lw $6,0($10)
- add $7,$6,$12
- sw $7,0($10)
- addi $10,$10,8
- bne $10,$5,LOOP

Renamed instruction

1. lw P1, 0($10)
2. add P2, P1, $12
3. sw P2, 0($10)
4. addi P3, $10, 8
5. bne P5, $5, LOOP
6. lw P4, 0($3)
7. add P5, P1, $12
8. sw P5, 0($3)
9. addi P6, P3, 8
10. bne P6, 0($10)
2-issue RR processor in motion

1. lw   $6,0($10)
2. add  $7,$6,$12
3. sw   $7,0($10)
4. addi $10,$10,8
5. bne  $10,$5,LOOP
6. lw   $6,0($10)
7. add  $7,$6,$12
8. sw   $7,0($10)
9. addi $10,$10,8
10. bne $10,$5,LOOP
2-issue RR processor in motion

lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP
lw   $6,0($10)
add  $7,$6,$12
sw   $7,0($10)
addi $10,$10,8
bne  $10,$5,LOOP

Renamed instruction

<table>
<thead>
<tr>
<th>Renamed instruction</th>
<th>Physical Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 lw P1, 0($10)</td>
<td>$5 P1 P6</td>
</tr>
<tr>
<td>2 add P2, P1, $12</td>
<td>$6 P1 P7</td>
</tr>
<tr>
<td>3 sw P2, 0($10)</td>
<td>$7 P1 P8</td>
</tr>
<tr>
<td>4 addi P3, $10, 8</td>
<td>$10 P5 P9</td>
</tr>
<tr>
<td>5 bne P3, $5, LOOP</td>
<td>$12 P3 P10</td>
</tr>
<tr>
<td>6 lw P4, 0(P3)</td>
<td>$5 P1 P6</td>
</tr>
<tr>
<td>7 add P5, P1, $12</td>
<td>$6 P1 P7</td>
</tr>
<tr>
<td>8 sw P5, 0(P3)</td>
<td>$7 P1 P8</td>
</tr>
<tr>
<td>9 addi P6, P3, 8</td>
<td>$10 P5 P9</td>
</tr>
<tr>
<td>10 bne P6, 0($10)</td>
<td>1 1 P10</td>
</tr>
</tbody>
</table>
Putting it all together
Pipeline SuperScalar/OoO/ROB

Front-end
- Instruction Fetch
- Instruction Decode
- Register renaming logic

Issue/Schedule
- ALU
- MUL/DIV
- Address Resolution
- Address Queue

Back-end
- ROB
- FP1
- FP2
- MUL/DIV
- MEM

Branch predictor
The pipelines of Modern Processors
Intel Pentium 4
The Skylake microarchitecture builds on the successes of the Haswell and Broadwell microarchitectures. The basic pipeline functionality of the Skylake microarchitecture is depicted in Figure 2-1.

The Skylake microarchitecture offers the following enhancements:

- Larger internal buffers to enable deeper OOO execution and higher cache bandwidth.
- Improved front end throughput.
- Improved branch predictor.
- Improved divider throughput and latency.
- Lower power consumption.
- Improved SMT performance with Hyper-Threading Technology.
- Balanced floating-point ADD, MUL, FMA throughput and latency.

The microarchitecture supports flexible integration of multiple processor cores with a shared uncore subsystem consisting of a number of components including a ring interconnect to multiple slices of L3 (an off-die L4 is optional), processor graphics, integrated memory controller, interconnect fabrics, etc. A four-core configuration can be supported similar to the arrangement shown in Figure 2-3.
ZEN MICROARCHITECTURE

- Fetch Four x86 instructions
- Op Cache instructions
- 4 Integer units
  - Large rename space – 168 Registers
  - 192 instructions in flight/8 wide retire
- 2 Load/Store units
  - 72 Out-of-Order Loads supported
- 2 Floating Point units × 128 FMACs
  - built as 4 pipes, 2 Fadd, 2 Fmul
- I-Cache 64K, 4-way
- D-Cache 32K, 8-way
- L2 Cache 512K, 8-way
- Large shared L3 cache
- 2 threads per core
Announcements

• CAPE/Survey
  • Screenshot of your CAPE
  • Fill the survey
  • Count as a full-credit assignment and we’re dropping your lowest two assignments now.

• Regarding final exam —
  • 9/4 8am—6pm — any consecutive, non-stop 3-hour slot you pick
  • Open books, open notes, but it’s going to be twice longer than the midterm
  • Lockdown browser (considering)
  • No zoom, no response to piazza posts for fairness