Superscalar processor
Instruction Scheduling and Reordering

Tasks of superscalar processing
- Parallel Decoding and issue
- Parallel instruction execution
- Preserving the sequential consistency of instruction execution and exception processing

Operand Fetch Policies
- Issue bound fetch
- Dispatch bound fetch

Issue bound operand fetch (with single register file)

Dispatch bound operand fetch (with single register file)

Three Important Concepts of Superscalar
- Renaming
  - Remove WAR and WAW data dependency
- Out of Order Scheduling
  - Score boarding and Tomasulo approach
- Re-Ordering
  - Maintaining instruction order of execution
Part I
Register Renaming

How renaming works?
(in context of combined reg file)

Who does renaming?

- Compiler
  - Done statically
  - Limited by registers visible to compiler
- Hardware
  - Done dynamically
  - Limited by registers available to hardware

Score-board

Introduced with CDC6600
Components of RS in Score board type Sceduling

- **Op**—Operation to perform in the unit
- **Qj, Qk**
  - From which FU is it will get Operand
- **Rj, Rk**
  - Ready Status of Source Registers for the Operation
  - If Both Ri and Rj is Yes, then Operation can be scheduled
- **Busy**—Indicates reservation station or FU is busy

CDC 6600: simple example (Ref: Flynn book 7.6.5)

1. Issue I to DIV (R1), (R2) -> DIV, Begin DIV
2. Issue I2 to MUL (Dependency) to RS TAG [DIV] (from R3) goes to MUL TAG [MUL] is placed in [R4] read score board
3. Decode I3 to ADD (Dependency) to RS TAG [ADD] for [R4] read score board TAG [ADD] for [R7] read score board

Precedence handle Hazards
But Stall for all RAW, WAR and WAW

INSTRUCTION ISSUE READ OP EX COMPL WRITERES
LF F6, 34(R2) √ √√√
LF F2, 45(R3) √
MUL F0,F2,F4 √
SUB F8,F6,F2 √
DIV F10,F0,F6 √
ADD F6,F8,F2

INSTRUCTION ISSUE READ OP EX COMPL WRITERES
LF F6, 34(R2) √ √√√
LF F2, 45(R3) √
MUL F0,F2,F4 √
SUB F8,F6,F2 √
DIV F10,F0,F6 √
ADD F6,F8,F2

<table>
<thead>
<tr>
<th>FU No</th>
<th>F0</th>
<th>F2</th>
<th>F4</th>
<th>F6</th>
<th>F8</th>
<th>F10</th>
<th>F12</th>
<th>F14</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>MUL1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MUL2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ADD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>DIV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

INSTRUCTION ISSUE READ OP EX COMPL WRITERES
LF F6, 34(R2) √ √√√
LF F2, 45(R3) √
MUL F0,F2,F4 √
SUB F8,F6,F2 √
DIV F10,F0,F6 √
ADD F6,F8,F2

INSTRUCTION ISSUE READ OP EX COMPL WRITERES
LF F6, 34(R2) √ √√√
LF F2, 45(R3) √
MUL F0,F2,F4 √
SUB F8,F6,F2 √
DIV F10,F0,F6 √
ADD F6,F8,F2

<table>
<thead>
<tr>
<th>FU No</th>
<th>F0</th>
<th>F2</th>
<th>F4</th>
<th>F6</th>
<th>F8</th>
<th>F10</th>
<th>F12</th>
<th>F14</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>INT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>MUL1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MUL2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ADD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>DIV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Precedence status

No NAME BUSY OP F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 F13 F14
1 INT
2 MUL1 Y LF
3 MUL2 Y MUL
4 ADD Y SUB
5 DIV Y DIV

Precedence status

No NAME BUSY OP F1 F2 F3 F4 F5 F6 F7 F8 F9 F10 F11 F12 F13 F14
1 INT Y LF
2 MUL1 Y MUL
3 MUL2 N
4 ADD Y SUB
5 DIV Y DIV
<table>
<thead>
<tr>
<th>Instruction</th>
<th>Issue</th>
<th>Read OP</th>
<th>EX Compl</th>
<th>Writeres</th>
</tr>
</thead>
<tbody>
<tr>
<td>LF F6, 34(R2)</td>
<td>√</td>
<td></td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>LF F2, 45(R3)</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mul F0,F2,F4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sub F8,F6,F2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Div F10,F0,F6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Functional Units**

- FU No 2
- FU No 1
- FU No 4
- FU No 5
- FU No 3

**Instruction status**

- No NAME BUSY
- OP Fi Fj Fk Qj Qk Rj Rk
- Name
- BUSY

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Issue</th>
<th>Read OP</th>
<th>EX Compl</th>
<th>Writeres</th>
</tr>
</thead>
<tbody>
<tr>
<td>LF F6, 34(R2)</td>
<td>√</td>
<td></td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>LF F2, 45(R3)</td>
<td>√</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mul F0,F2,F4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sub F8,F6,F2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Div F10,F0,F6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Functional Units**

- FU No 2
- FU No 1
- FU No 4
- FU No 5
- FU No 3

**Instruction status**

- No NAME BUSY
- OP Fi Fj Fk Qj Qk Rj Rk
- Name
- BUSY
### Instruction Status

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Issue</th>
<th>Read OP</th>
<th>EX Compl</th>
<th>Writeres</th>
</tr>
</thead>
<tbody>
<tr>
<td>LF, F6, 34(R2)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>LF, F2, 45(R3)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Mul, F0, F2, F4</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Sub, F8, F6, F2</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Div, F0, F6, F6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Add, F6, F8, F2</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

### Functional Units

<table>
<thead>
<tr>
<th>Functional Units</th>
<th>FU No</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>F0, F2, F4, F6, F8, F10, F12, F14</td>
<td>2</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

### Tomasulo's Scheduling

- **Out of Order Scheduling**
- No renaming required, can be done in Hardware by using special tagging
- Developed in IBM but extensively used by Intel/AMD
- **Tomasulo**: Eckert-Mauchly Award in 1997
- All modern processors use this method
  - Pentium Pro, Core 2 Duo, Core i3/i5/i7, AMD Opteron/Phenom

---

A Sahu
Online Demos of Tamasulo’s Scheduling
• It is so important to Read in ACA Course.
• Many Demo’s available online
• Ref:
  – http://www.ecs.umass.edu/ece/koren/architecture/Tomasulo/AppletTomasulo.html
  – http://www.dcs.ed.ac.uk/home/hase/webhase/demo/tomasulo.html
  – http://www.ecs.umass.edu/ece/koren/architecture/Tomasulo1/tomasulo.htm

Three Stages of Tomasulo Algorithm
1. Issue — get instruction from FP Op Queue
   • If reservation station free (no structural hazard), control issues instr & sends operands (renames registers).
2. Execution — operate on operands (EX)
   • When both operands ready then execute;
   • if not ready, watch Common Data Bus for result
3. Write result — finish execution (WB)
   • Write on Common Data Bus to all awaiting units;
   • mark reservation station available
   • Issue: build dependence for new inst
   • Writeback: Wakeup dependent instructions

Issue Stage and Renaming Table
• Renames its two source registers (source renaming)
• Assigns it to a free RS
• Updates Renaming table (dest renaming)
• Also decodes the inst and read register values in parallel

Execute Stage
• Only “ready” instructions can join the competition
• There is a select logic to select instructions for FU execution
  – Some policy may be used, e.g. age based
• Non-ready instructions can be “waken up” during writeback of its parent inst

Writeback and Common Data Bus
• Normal data bus
  – data + destination (“go to” bus)
• Common data bus
  – data + source (“come from” bus)
  – 64 bits of data + 4 bits of source index (tag)
  – Does the broadcast to every instruction in the fly
• How it do
  – Child instructions do tag matching and update their ready bits and value fields (if the tag matches theirs)

Tomasulo’s Organization

Adapted from UCB CS252 S98 Copyright 1998 UCB
Illustration 2 (Ref Hennessy Book Sec 3.4)

IBM 360/91 - Tomasulo’s scheme

- Issue bound fetch
- FUs: LOAD, STORE, 3 x ADD/SUB, 2 x MUL/DIV
- Group RS’s with 1 slot per FU
- 1 RF
- In order issue, out of order execution

Components of RS in Tomasulo’s Approach

- **Op**—Operation to perform in the unit
- **Vj, Vk** (Value of Source operands)
  - Store buffers have V field, result to be stored
- **Qj, Qk**
  - Qk—Reservation stations producing source registers (value to be written)
  - Store buffers only have Qj for RS producing result
- **Busy**—Indicates reservation station or FU is busy
Part III

Instructions Reordering
(RO Buffer)