UNIVERSITY OF MASSACHUSETTS
Dept. of Electrical & Computer Engineering

Computer Architecture
ECE 568

Part 9
Superscalar

Israel Koren

Getting CPI < 1:
Issuing Multiple Instructions (Ops)/Cycle

♦ Vector Processing: Explicit coding of independent loops as
operations on large vectors of numbers
  • Multimedia instructions being added to many processors
♦ Superscalar: varying no. instructions/cycle (1 to 8), scheduled
by HW (Tomasulo)
  • IBM PowerPC, Sun UltraSparc, Intel x86, AMD
♦ (Very) Long Instruction Words (V)LIW:
  Fixed number of instructions (3-6) scheduled by the compiler;
  put ops into wide templates
  • Intel Itanium Architecture-64 (IA-64)
♦ Parallel processing:
  • Intel Core i7: 4-6 cores
♦ Multi-threading

∗ Success of multiple instructions lead to
Instructions Per Clock_cycle (IPC) vs. CPI
**Issuing Multiple Instructions/Cycle**

- **Superscalar MIPS:** 2 instructions, 1 FP & 1 anything (non FP arithmetic op)
  - Fetch 64-bits/clock cycle; Integer/Load on left, FP on right
  - Can only issue 2nd instruction if 1st instruction issues
  - More ports for register files

<table>
<thead>
<tr>
<th>Type</th>
<th>Pipe Stages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Int. instruction</td>
<td>IF ID EX MEM WB</td>
</tr>
<tr>
<td>FP instruction</td>
<td>IF ID EX MEM WB</td>
</tr>
<tr>
<td>Int. instruction</td>
<td>IF ID EX MEM WB</td>
</tr>
<tr>
<td>FP instruction</td>
<td>IF ID EX MEM WB</td>
</tr>
</tbody>
</table>

- 1 cycle load delay affects now 3 instructions in SuperScalar
  - Instruction in right half can't use it, nor instructions in next slot

---

**4-way Alpha 21164 superscalar**

- INT RF: 4 read + 2 write ports
- FP RF: 6 read + 3 write ports
Multiple-Issue Processors: Design Complexity

♦ Issue packet: group of instructions from fetch unit that could potentially issue in 1 clock
  • If instruction causes structural hazard or data hazard either due to earlier instruction in execution or to earlier instruction in issue packet, then instruction should not issue

♦ Performing issue checks in 1 cycle could increase clock cycle time: \( \sim N^2 \) comparisons (e.g., \( N=4 \))
  • Issue stage usually split and pipelined
  • 1st stage decides how many instructions from within this packet can issue, 2nd stage examines hazards among selected instructions and those already been issued

♦ Higher branch penalties \( \Rightarrow \) prediction accuracy very important

Multiple Issue Challenges

♦ Integer/FP split is simple for the HW but CPI of 0.5 achieved only for programs with:
  • Exactly 50% FP operations AND No hazards

♦ Greater difficulty of decode, issue, rename and WB:
  • Even dual-issue \( \Rightarrow \) examine 2 opcodes, 6 register specifiers, decide if 1 or 2 instructions can issue
  • Register file: need 4 reads and 2 writes/cycle (2N and N in general)
  • Rename logic: must be able to rename same register multiple times in one cycle! For instance, consider 4-way issue:
    
    - add \( r_1, r_2, r_3 \)
    - add \( p_{11}, p_4, p_7 \)
    - sub \( r_4, r_1, r_2 \) \( \Rightarrow \) sub \( p_{22}, p_{11}, p_4 \)
    - lw \( r_1, 4(r_4) \) \( \Rightarrow \) lw \( p_{23}, 4(p_{22}) \)
    - add \( r_5, r_1, r_2 \)
    - add \( p_{12}, p_{23}, p_4 \)

Imagine doing this transformation in a single cycle!

• Result buses: Need to complete multiple instructions/cycle
  » So, need multiple buses with associated matching logic at every reservation station.
  » Or, need multiple forwarding paths
Dynamic Scheduling in Superscalar

- Issue two instructions each cycle and keep in-order instruction issue for Tomasulo
  - Assume 1 integer + 1 floating point
  - 1 Tomasulo control for integer (and load), 1 for floating point
- Loads/stores might cause dependency between integer and FP issue:
  - Replace load reservation station with a load queue; operands must be read in the order the Loads are fetched
  - Load queue checks addresses in Store Queue to avoid RAW violation
  - Store queue checks addresses in Load & Store Queues to avoid WAR & WAW

Register renaming, virtual registers versus Reorder Buffers

- Alternative to Reorder Buffer is a larger virtual set of registers and register renaming
- Virtual registers hold both architecturally visible registers + temporary values
  - replace functions of reorder buffer and reservation stations
- Renaming process maps names of architectural registers to registers in virtual register set
- Simplifies instruction commit: mark register as no longer speculative, free register with old value
- Adds 40-80 extra registers: Alpha, certain Pentiums, IBM RS/6000 ...
  - Number of registers limits no. instructions in execution (used until commit)
Is there a limit to IPC?

- If we have unlimited HW is IPC limited?
- True data dependencies (RAW) will always limit IPC
- Instruction Level Parallelism (ILP) is the number of instructions in the program that can be executed in parallel
  - Application dependent
- Study to determine upper limit for ILP
- Assuming ideal machine
  - Register renaming – infinite virtual registers
    - all WAW & WAR hazards are avoided
  - Unlimited number of instructions issued/clock cycle
  - Perfect caches: 1 cycle latency for all instructions
  - Only show the impact of branch prediction

Branch Prediction Impact

<table>
<thead>
<tr>
<th>Program</th>
<th>Perfect</th>
<th>Tournament</th>
<th>BHT (2 bit; 512)</th>
<th>Profile</th>
<th>No prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcc</td>
<td>9</td>
<td>6</td>
<td>2</td>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>espresso</td>
<td>12</td>
<td>6</td>
<td>2</td>
<td>16</td>
<td>7</td>
</tr>
<tr>
<td>li</td>
<td>10</td>
<td>7</td>
<td>2</td>
<td>16</td>
<td>7</td>
</tr>
<tr>
<td>fpupp</td>
<td>49</td>
<td>45</td>
<td>45</td>
<td>58</td>
<td>58</td>
</tr>
<tr>
<td>doducd</td>
<td>15</td>
<td>13</td>
<td>14</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>tomcatv</td>
<td>19</td>
<td>19</td>
<td>19</td>
<td>46</td>
<td>46</td>
</tr>
</tbody>
</table>

IPC

Instruction issues per cycle

FP: 15 - 49

Integer: 9 - 12

Profile

ECE568/Koren Part.9.9 Adapted from UCB and other sources Copyright UCB & Morgan Kaufmann
Examples of superscalars circa 2006

- Intel Pentium 4 Extreme: 3.2 GHz, 170 million devices, 230 mm², 90nm, 94 Watt, 16K L1, 2M L2, dual-core
- AMD Athlon 64 FX: 2.2 GHz, 106 million devices, 193 mm², 130 nm, 89 Watt, 128K L1, 1M L2, support 64-bit OS
- Max issue: 4 instructions (many CPUs)
  - Max rename registers: 128 (Pentium 4)
  - Max BHT: 16Kx2 (Intel Ultra III), 4K BTB (Extreme)
  - Max Window Size: 126 instructions (Pentium 4)
  - Max Pipeline: 22/24 → 31 stages (Intel) vs. 10 → 12/17 (AMD)
- High power density resulted in the move to multi-cores

Dual-core 2007