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 compiler or by HW (Tomasulo)
  • IBM PowerPC, Sun UltraSparc, DEC Alpha, Pentium 4
♦ (Very) Long Instruction Words (V)LIW: fixed number of instructions (4-16) scheduled by the compiler; put ops into wide templates
  • Intel Architecture-64 (IA-64) 64-bit address

♦ Parallel processing:
  • Intel Core 2 Duo/ Quad core

* Anticipated 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 Arith on right
  - Can only issue 2nd instruction if 1st instruction issues
  - More ports for register files

<table>
<thead>
<tr>
<th>Type</th>
<th>PipeStages</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>
<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

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 a data hazard either due to earlier instruction in execution or to earlier instruction in issue packet, then instruction should not issue
  • 0 to N instruction issues per clock cycle, for N-issue

♦ Performing issue checks in 1 cycle could increase clock cycle time: \(O(N^2)\) comparisons
  • => 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 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 and issue:
  - Even dual-issue => examine 2 opcodes, 6 register specifiers, decide if 1 or 2 instructions can issue
  - Register file: need 2N reads and N writes/cycle
  - Rename logic: must be able to rename same register multiple times in one cycle! For instance, consider 4-way issue:
    - \( \text{add } r1, r2, r3 \rightarrow \text{add } p11, p4, p7 \)
    - \( \text{sub } r4, r1, r2 \rightarrow \text{sub } p22, p11, p4 \)
    - \( \text{lw } r1, 4(r4) \rightarrow \text{lw } p23, 4(p22) \)
    - \( \text{add } r5, r1, r2 \rightarrow \text{add } p12, p23, p4 \)
  - 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

- How to issue two instructions 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, Pentium,...
  • Number of registers limits no. instructions in execution (used until commit)

Study: Limits to ILP – View at home (slides 8-12)

Assumptions for ideal/perfect machine:

1. Register renaming – infinite virtual registers
   => all register WAW & WAR hazards are avoided

2. Branch & Jump prediction – perfect; no mispredictions
   => machine with perfect speculation & an unbounded buffer of instructions available

3. Memory-address alias analysis – addresses are known & a store can be moved before a load, provided addresses not equal

Also:
unlimited number of instructions issued/clock cycle;
perfect caches;
1 cycle latency for all instructions (FP *,/);
Upper Limit to ILP: Ideal Machine

- Integer: 18 - 62
- FP: 75 - 150

Programs
- gcc
- espresso
- li
- fppp
- doducd
- tomcat

IPC (Instruction Issues per cycle)
- 54.8
- 62.6
- 17.9
- 75.2
- 118.7
- 150.1

More Realistic HW: Branch Impact

- Change from Infinite window to 2000 and maximum issue of 64 instructions per clock cycle
- Integer: 6 - 16
- FP: 13 - 49

Programs
- gcc
- espresso
- li
- fppp
- doducd
- tomcat

IPC (Instruction Issues per cycle)
- Perfect
- Tournament
- BHT (2 bit; 512)
- Profile
- No prediction
More Realistic HW: Renaming Registers Impact

2000 instr window, 64 instr issue, 8K 2 level Prediction

FP: 11 - 49

IPC

Integer: 9 - 15

Realistic HW: Window Size Impact

Tournament predictor, Issue as many as window, 64 renaming registers

FP: 7 - 47

ECE568/Koren Part 9.11
Adapted from UCB and other sources
Copyright UCB & Morgan Kaufmann
Workstation microprocessors 2006

- Intel and AMD processors:
  - 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)

Dual-cores 2007

- Instruction Fetch and PreDecode
- Instruction Queue
- Decode
- Rename/Alloc
- Retirement Unit (ReOrder Buffer)
- Schedulers
- L1 D-Cache and D-TLB

- Instruction Fetch and PreDecode
- Instruction Queue
- Decode
- Rename/Alloc
- Retirement Unit (ReOrder Buffer)
- Schedulers
- L1 D-Cache and D-TLB