Last Time: Multipliers -- Space vs. Time

Five options for multiplier latency ...

1-cycle option is fully spatial.

35-cycle is mini-Lab 2.

2, 4, 5 cycles?

CONFIG IU MUL LATENCY 1:
The multiplier used for UMUL/SMUL instructions can be implemented with 1, 2, 4, 5 or 35 cycles latency. Lower latency gives higher multiplication performance, but increases area and might reduce the maximum clock frequency. The best area/timing/performance compromise is usually 4 or 5. A latency of 5 cycles will use the same multiplier (16x16) as for 4 cycles, but with a pipeline register to improve timing.
Today: Beyond the 5-stage pipeline

- Taxonomy: Introduction to advanced processor techniques.

- Superpipelining: Increasing the number of pipeline stages.

- Superscalar: Issuing several instructions in a single cycle.
5 Stage Pipeline: A point of departure

At best, the 5-stage pipeline executes one instruction per clock, with a clock period determined by the slowest stage.

Application does not need multi-cycle instructions (multiply, divide, etc.)

Seconds Program = Instructions Program Cycles Instruction Seconds Cycle

Perfect caching

Filling all delay slots (branch, load)
Goal: Reduce critical path by adding more pipeline stages.

Example: 8-stage ARM XScale: extra IF, ID, data cache stages.

Difficulties: Added penalties for load delays and branch misses.

Ultimate Limiter: As logic delay goes to 0, FF clk-to-Q and setup.
Superscalar: Multiple issues per cycle Today!

Goal: Improve CPI by issuing several instructions per cycle.

Example: CPU with floating point ALUs: issue 1 FP + 1 integer instruction per cycle.

Difficulties: Load and branch delays affect more instructions.

Ultimate Limiter: Programs may be a poor match to issue rules.
Out of Order: Going around stalls

<table>
<thead>
<tr>
<th>Seconds Program</th>
<th>Instructions Program</th>
<th>Cycles Instruction</th>
<th>Seconds Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD</td>
<td>F2, 34(R2)</td>
<td>1 latency</td>
<td></td>
</tr>
<tr>
<td>LD</td>
<td>F4, 45(R3)</td>
<td>long waiting</td>
<td></td>
</tr>
<tr>
<td>MULTD</td>
<td>F6, F4, F2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>ADDD</td>
<td>F8, F2, F2</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Goal: Issue instructions out of program order

Example: ... so let ADDD go first

Difficulties: Bookkeeping is highly complex. A poor fit for lockstep instruction scheduling.

Ultimate Limiter: The amount of instruction level parallelism present in an application.
Dynamic Scheduling: End lockstep

**Goal:** Enable out-of-order by breaking pipeline in two: fetch and execution.

**Example:** IBM Power 5:

Limiters: Design complexity, instruction level parallelism.
Throughput and multiple threads

Goal: Use multiple CPUs (real and virtual) to improve (1) throughput of machines that run many programs (2) execution time of multi-threaded programs.

Example: Sun Niagara (8 SPARC processors on one chip).

Difficulties: Gaining full advantage requires rewriting applications, OS, libraries.

Ultimate limiter: Amdahl’s law, memory system performance.
Administrivia: No class on Thursday!

- HW 4: due Weds 11/10, 5PM, 283 Soda.
- Friday 11/12: Lab 4 final demo in section.
- Monday 11/15: Lab 4 final report due, 11:59 PM.
- Final project (Lab 5) will be out soon
Mid-Term II Review Session: Sunday, 11/21, 7-9 PM, 306 Soda.

Mid-Term II: Tuesday, 11/23, 5:30 to 8:30 PM, 101 Morgan.

Xilinx field trip date: 11/30. Details on bus transport from Soda Hall soon.

Thanksgiving Holiday!
Superpipelining
Add pipeline stages, reduce clock period

Q. Could adding pipeline stages reduce CPI for an application?
A. Yes, due to these problems:

<table>
<thead>
<tr>
<th>CPI Problem</th>
<th>Possible Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extra branch delays</td>
<td>Branch prediction</td>
</tr>
<tr>
<td>Extra load delays</td>
<td>Optimize code</td>
</tr>
<tr>
<td>Structural hazards</td>
<td>Optimize code, add hardware</td>
</tr>
</tbody>
</table>

ARM XScale
8 stages

CS 152 L20: Advanced Processors I
Hardware limits to superpipelining?

FO4 Delays

Historical limit: about 12

CPU Clock Periods 1985-2005

MIPS 2000
5 stages

Pentium Pro
10 stages

Pentium 4
20 stages

FO4: How many fanout-of-4 inverter delays in the clock period.

Thanks to Francois Labonte, Stanford
Is there an optimal pipeline depth?

Methodology: Simulate standard benchmarks on many different pipeline designs.

Source: “The Optimal Logic Depth Per Pipeline Stage is 6 to 8 FO4 Inverter Delays”. M.S. Hrishikesh et al. ISCA 2002.

In Resources section of class website
Superscalar
Superscalar: A simple example ...

Example: Superscalar MIPS. Fetches 2 instructions at a time. If first integer and second floating point, issue in same cycle

<table>
<thead>
<tr>
<th>Integer instruction</th>
<th>FP instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD F0,0(R1)</td>
<td>ADDD F4,F0,F2</td>
</tr>
<tr>
<td>LD F6,-8(R1)</td>
<td>ADDD F8,F6,F2</td>
</tr>
<tr>
<td>LD F10,-16(R1)</td>
<td>ADDD F12,F10,F2</td>
</tr>
<tr>
<td>LD F14,-24(R1)</td>
<td>ADDD F16,F14,F2</td>
</tr>
<tr>
<td>LD F18,-32(R1)</td>
<td>ADDD F20,F18,F2</td>
</tr>
<tr>
<td>SD 0(R1),F4</td>
<td>ADDD F16,F14,F2</td>
</tr>
<tr>
<td>SD -8(R1),F8</td>
<td>ADDD F20,F18,F2</td>
</tr>
<tr>
<td>SD -16(R1),F12</td>
<td></td>
</tr>
<tr>
<td>SD -24(R1),F16</td>
<td></td>
</tr>
</tbody>
</table>
Superscalar: Visualizing the pipeline

Type                     | Pipe Stages
---|---
Int. instruction         | IF ID EX MEM WB
FP instruction           | IF ID EX MEM WB
Int. instruction         | IF ID EX MEM WB
FP instruction           | IF ID EX MEM WB
Int. instruction         | IF ID EX MEM WB
FP instruction           | IF ID EX MEM WB

Three instructions affected by a single cycle of load delay. Why?
Limitations of “lockstep” superscalar

* Only get 0.5 CPI for a 50/50 mix of float and integer ops with no hazards.
* Extending scheme to general mixes and more instructions is complicated.
* If one accepts building a complicated machine, there are better ways to do it.

Next time: Dynamic Scheduling
Recall: Branch Predictors are Caches

0b0110[...]01001000  BNEZ R1 Loop

Branch Target Buffer (BTB)

28-bit address tag  target address

Branch History Table (BHT)

0b0110[...]0100  PC + 4 + Loop

Note: BHT can be larger than BTB, does not need a tag.

"Taken" Address

"Taken" or "Not Taken"

"Target Address has BHT details"
Conclusion: Superpipelining, Superscalar

The 5 stage pipeline: a starting point for performance enhancements, a building block for multiprocessing.

Superpipelining: Reduce critical path by adding more pipeline stages. Has the potential to increase the CPI.

Superscalar: Multiple instructions at once. Programs must fit the issue rules. Adds complexity.