Lecture 20: Core Design

• Today: Innovations for ILP, TLP, power

• ISCA workshops

• Sign up for class presentations
ILP Limits  Wall 1993
Techniques for High ILP

• Better branch prediction and fetch (trace cache) → cascading branch predictors?
• More physical registers, ROB, issue queue, LSQ → two-level regfile/IQ?
• Higher issue width → clustering?
• Lower average cache hierarchy access time
• Memory dependence prediction
• Latency tolerance techniques: ILP, MLP, prefetch, runahead, multi-threading
2Bc-gskew Branch Predictor

Address → BIM

Address+History → G0

Address+History → G1

Address+History → Meta

BIM → Vote

G0 → Vote

G1 → Vote

Meta → Vote

Vote → Pred

44 KB; 2-cycle access; used in the Alpha 21464
Rules

• On a correct prediction
  ➢ if all agree, no update
  ➢ if they disagree, strengthen correct preds and chooser

• On a misprediction
  ➢ update chooser and recompute the prediction
    ▪ on a correct prediction, strengthen correct preds
    ▪ on a misprediction, update all preds
Impact of Mem-Dep Prediction

- In the perfect model, loads only wait for conflicting stores; in naïve model, loads issue speculatively and must be squashed if a dependence is later discovered.

From Chrysos and Emer, ISCA’98
When the oldest instruction is a cache miss, behave like it causes a context-switch:

- checkpoint the committed registers, rename table, return address stack, and branch history register
- assume a bogus value and start a new thread
- this thread cannot modify program state, but can prefetch
Memory Bottlenecks

- 128-entry window, real L2 $\rightarrow$ 0.77 IPC
- 128-entry window, perfect L2 $\rightarrow$ 1.69
- 2048-entry window, real L2 $\rightarrow$ 1.15
- 2048-entry window, perfect L2 $\rightarrow$ 2.02
- 128-entry window, real L2, runahead $\rightarrow$ 0.94
SMT Pipeline Structure

SMT maximizes utilization of shared execution engine
SMT Fetch Policy

• Fetch policy has a major impact on throughput: depends on cache/bpred miss rates, dependences, etc.

• Commonly used policy: ICOUNT: every thread has an equal share of resources
  ▪ faster threads will fetch more often: improves throughput
  ▪ slow threads with dependences will not hoard resources
  ▪ low probability of fetching wrong-path instructions
  ▪ higher fairness
Area Effect of Multi-Threaded Systems

- The curve is linear for a while
- Multi-threading adds a 5-8% area overhead per thread (primary caches are included in the baseline)

From Davis et al., PACT 2005
Figure 4: SPEC JBB average core IPC range (maximum to minimum) for medium-scale CMTs. The secondary cache size range is 1.5MB, 2.5MB, 3.5MB, and 4.5MB from left to right for each core.
# Maximal Aggregate IPCs

## Table 3: Maximum AIPC for medium-scale CMTs for SPEC JBB, TPC-C, TPC-W, and XML Test.

<table>
<thead>
<tr>
<th>Core Config</th>
<th>SPEC JBB 2000</th>
<th>TPC-C</th>
<th>TPC-W</th>
<th>XML Test</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>L1</td>
<td>L2</td>
<td>Cores</td>
<td>AIPC</td>
</tr>
<tr>
<td>1p2t</td>
<td>16/32</td>
<td>1.5/12</td>
<td>20</td>
<td>9.8</td>
</tr>
<tr>
<td>1p4t</td>
<td>16/32</td>
<td>1.5/12</td>
<td>17</td>
<td>13.2</td>
</tr>
<tr>
<td>1p8t</td>
<td>16/32</td>
<td>2.5/10</td>
<td>12</td>
<td>11.7</td>
</tr>
<tr>
<td>2p2t</td>
<td>16/32</td>
<td>1.5/12</td>
<td>16</td>
<td>8.6</td>
</tr>
<tr>
<td>2p4t</td>
<td>32/32</td>
<td>1.5/12</td>
<td>14</td>
<td>12.9</td>
</tr>
<tr>
<td>2p8t</td>
<td>16/32</td>
<td>1.5/12</td>
<td>12</td>
<td>16.5</td>
</tr>
<tr>
<td>2p16t</td>
<td>32/64</td>
<td>2.5/10</td>
<td>7</td>
<td>13.3</td>
</tr>
<tr>
<td>3p3t</td>
<td>32/32</td>
<td>1.5/12</td>
<td>13</td>
<td>10.3</td>
</tr>
<tr>
<td>3p6t</td>
<td>32/32</td>
<td>1.5/12</td>
<td>11</td>
<td>14.4</td>
</tr>
<tr>
<td>3p12t</td>
<td>32/64</td>
<td>1.5/12</td>
<td>9</td>
<td><strong>17.3</strong></td>
</tr>
<tr>
<td>3p24t</td>
<td>32/64</td>
<td>2.5/10</td>
<td>5</td>
<td>13.6</td>
</tr>
<tr>
<td>4p8t</td>
<td>32/32</td>
<td>1.5/12</td>
<td>9</td>
<td>14.9</td>
</tr>
<tr>
<td>4p16t</td>
<td>32/64</td>
<td>1.5/12</td>
<td>7</td>
<td>16.8</td>
</tr>
<tr>
<td>2s2t</td>
<td>64/64</td>
<td>1.5/12</td>
<td>11</td>
<td>4.4</td>
</tr>
<tr>
<td>2s4t</td>
<td>64/64</td>
<td>1.5/12</td>
<td>10</td>
<td>7.0</td>
</tr>
<tr>
<td>2s8t</td>
<td>64/64</td>
<td>1.5/12</td>
<td>9</td>
<td>10.5</td>
</tr>
<tr>
<td>4s1t</td>
<td>64/64</td>
<td>1.5/12</td>
<td>7</td>
<td>12.1</td>
</tr>
<tr>
<td>4s2t</td>
<td>64/64</td>
<td>1.5/12</td>
<td>7</td>
<td>2.9</td>
</tr>
<tr>
<td>4s4t</td>
<td>64/64</td>
<td>1.5/12</td>
<td>6</td>
<td>4.5</td>
</tr>
<tr>
<td>4s8t</td>
<td>64/64</td>
<td>1.5/12</td>
<td>5</td>
<td>6.6</td>
</tr>
</tbody>
</table>

Note: The L1 refers to the primary data/instruction cache size. The L2 cache configuration size (MB)/set associativity (SA) are provided along with the total number of cores for that CMT configuration.
Power/Energy Basics

- Energy = Power x time

- Power = Dynamic power + Leakage power

- Dynamic Power = $\alpha C V^2 f$
  - $\alpha$ switching activity factor
  - $C$ capacitances being charged
  - $V$ voltage swing
  - $f$ processor frequency
Guidelines

• Dynamic frequency scaling (DFS) can impact power, but has little impact on energy

• Optimizing a single structure for power/energy is good for overall energy only if execution time is not increased

• A good metric for comparison: $ED^2$ (because DVFS is an alternative way to play with the E-D trade-off)

• Clock gating is commonly used to reduce dynamic energy, DFS is very cheap (few cycles), DVFS and power gating are more expensive (micro-seconds or tens of cycles, fewer margins, higher error rates)
Criticality Metrics

- Criticality has many applications: performance and power; usually, more useful for power optimizations

- **QOLD** – instructions that are the oldest in the issue queue are considered critical
  - can be extended to oldest-N
  - does not need a predictor
  - young instructions are possibly on mispredicted paths
  - young instruction latencies can be tolerated
  - older instructions are possibly holding up the window
  - older instructions have more dependents in the pipeline than younger instructions
Other Criticality Metrics

- QOLDDEP: Producing instructions for oldest in q
- ALOLD: Oldest instr in ROB
- FREED-N: Instr completion frees up at least N dependent instrs
- Wake-Up: Instr completion triggers a chain of wake-up operations
- Instruction types: cache misses, branch mpreds, and instructions that feed them
Title

• Bullet