Reconfigurable Issue Logic for Microprocessor Power/Performance Throttling

Dave Maze
Edwin Olson
Andrew Menard

Observation

- Complex issue logic for out-of-order, speculative machines consumes a significant amount of power.
- Performance increase by complex issue logic is small.
Power/Performance Throttling

- Many applications have significant peak processing performance requirements but low average performance requirements.
- Very low power microprocessors can’t deliver best of breed performance. Fastest uPs consume way too much power.
- Examples: handheld device which might be in standby, waiting for user input, MP3 player mode, or MPEG4 video playback

What about Voltage/Frequency Scaling

- Reducing voltage (factor of $\alpha$ decreases performance by factor of $\alpha$ but decreases power consumption by $\alpha^2$.)
- Voltage scaling runs out of steam as $V_{dd}$’s approach a few Vt. Need other mechanisms for throttling power/performance.
Why is Issue Logic so power hungry?

- In issue queue, every instruction is checked *every cycle* to see if it can be dispatched. This involves broadcasts of data on long bitlines.
- In 21264, queues compaction accounts for additional energy.
- Alpha 21264 consumes 18-46% of total energy in issue logic [Gowan] [Gupta].

Our Three Approaches

- Add separate simple core to complex uP and switch between them
  - Mode switches slow. Only 2 performance points.
- Only use a subset of the issue window
  - Probably not as low-power. Provides continuum of performance points. Mode switches easy.
- Bypass issue logic completely
  - Must flush issue window. Only 2 performance points.
Preliminary Results

<table>
<thead>
<tr>
<th></th>
<th>1x4 IO</th>
<th>2x4 IO</th>
<th>4x16 IO</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPC</td>
<td>0.57</td>
<td>0.64</td>
<td>1.39</td>
</tr>
<tr>
<td>Issue Power</td>
<td>1.75</td>
<td>2.06</td>
<td>6.12</td>
</tr>
<tr>
<td>Total Power</td>
<td>6.83</td>
<td>7.1</td>
<td>14.7</td>
</tr>
<tr>
<td>Issue Power %</td>
<td>25.6</td>
<td>29</td>
<td>41.6</td>
</tr>
<tr>
<td>IPC/Power</td>
<td>0.083</td>
<td>0.09</td>
<td>0.095</td>
</tr>
<tr>
<td>IPC/Issue Power</td>
<td>0.33</td>
<td>0.31</td>
<td>0.23</td>
</tr>
</tbody>
</table>

- Issue Width X Window Size X In/Out of Order
- Using identical technologies
- 1 issue is very poorly modeled; IPC is probably too low and power is almost certainly too high.
- SpecInt95 li benchmark

Issue window throttling

<table>
<thead>
<tr>
<th></th>
<th>4x32</th>
<th>4x16</th>
<th>4x8</th>
<th>4x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPC</td>
<td>1.41</td>
<td>1.39</td>
<td>1.23</td>
<td>0.89</td>
</tr>
<tr>
<td>Issue Power</td>
<td>7.47</td>
<td>6.12</td>
<td>4.75</td>
<td>3.14</td>
</tr>
<tr>
<td>Total Power</td>
<td>16.83</td>
<td>14.71</td>
<td>12.4</td>
<td>9.28</td>
</tr>
<tr>
<td>Issue Power %</td>
<td>44.4</td>
<td>41.6</td>
<td>38.3</td>
<td>33.8</td>
</tr>
<tr>
<td>IPC/Power</td>
<td>0.084</td>
<td>0.094</td>
<td>0.099</td>
<td>0.096</td>
</tr>
<tr>
<td>IPC/Issue Power</td>
<td>0.032</td>
<td>0.033</td>
<td>0.032</td>
<td>0.026</td>
</tr>
</tbody>
</table>

Issue width x window size
Tools

- Using Wattch (based on SimpleScalar) for high-level architectural modeling. Wattch gives us IPC and power data.
- Unable to measure critical path differences with Wattch. Open to suggestions... ??

Plan

- Build better models of in-order processors to provide fairer comparison.
  - Custom tool?
- Try to get timing information (?)
- For checkpoint 2, paper mostly done except for some final data results.