The case for a Single-Chip multiprocessor

German E. Alfaro
During the 80's and 90's advances in integrated circuit technology allowed increased circuit density and higher clock rates in microprocessors. Computer architects relied on techniques like: multiple instruction issue, dynamic scheduling, speculative execution and non-blocking caches to gain performance.
Superscalar Trends: (limits?)

- Dynamic Scheduling-> uses hardware to track register dependencies between instructions, and instruction is executed possible out of order as soon as all of its dependencies are satisfied.
- Scoreboard-> register checking in hardware.
- Register renaming-> Improve the efficiency of dynamic scheduling using hardware structures called reservation station.
Fetch Phase

The goal of Fetch phase is to present the rest of the CPU with a large and accurate window of decoded instructions.

Fetch phase constraints:

- Miss predicted branches
- Instruction misalignment
- Cache Misses

Branch prediction crucial but not enough!
Issue phase

- A packet of renamed instructions is inserted on to the instruction issue queue, an instruction is executed once all operands are ready.

- Two ways to implement renaming:
  - Mapping table
  - Reorder buffer

- These increment die-size and wiring causing long delays, limiting the cycle time of the processor.

- Increment in die size and wiring also = POWER

“Issue logic is one of the most complex parts of superscalar processors, one of the largest consumers of energy, and one of the main sites of power density.”

Single-Chip Multiprocessor

Motivation

- Avoid delay of the complex issue queue and multi-port register files in Superscalar processors.
- Exploit parallelism of Floating-Point programs.
- Recent advances in compilers make a Multiprocessor an efficient and flexible way to exploit parallelism.
- CMP individual processors are simple and achieve very high clock rates, will work well on integer programs.
- Low latency interprocessor connection.

“Our measurements suggest that the level of aggressive out-of-order, speculative execution present in modern processors is already beyond the point of diminishing performance returns for such Programs”

“We believe that the potential for CMP systems is even greater. CMP designs, such as Hydra and Piranha, seem especially promising…”

According to Wall Study in Parallel Applications:

- Applications fall in two cases:
  
  - **Moderate-low parallelism** \(< 10\) Instruction per cycle (mostly integers)
    - Works best on processors moderately scalar (2 issue) very high clock rate
  
  - **Large amount of parallelism** \(>40\) Instructions per cycle (mostly FP & loop-level parallelism)
    - Benefits most by: superscalar, VLIW, Vector processing.

The Application Parallelism landscape

- Instruction
- Basic block
  - Small group of instructions terminated by a branch.
- Loop iterations
  - A loop with independent data
- Task
  - Large independent tasks from a single application.
- Processes
  - Independent OS processes from different applications.
CMP implementations

- Execution of multiple processes in parallel under control of Operating system CMP aware.
- Multiple threads in parallel from a single application.
- Acceleration of execution of sequential applications using automatic parallelization technology in scientific applications.
4x2 CMP VS 6-way scalar

- Same die size and same process technologies
- Same clock frequency (500 MHz)
- Extremely different Microarchitecture

Figure 2. Floorplan for the six-issue dynamic superscalar microprocessor.

Figure 3. Floorplan for the four-way single-chip multiprocessor.
<table>
<thead>
<tr>
<th># of CPUs</th>
<th>1</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Degree superscalar</td>
<td>6</td>
<td>$4 \times 2$</td>
</tr>
<tr>
<td># of architectural registers</td>
<td>32int/32fp</td>
<td>$4 \times 32$int/32fp</td>
</tr>
<tr>
<td># of physical registers</td>
<td>160int/160fp</td>
<td>$4 \times 40$int/40fp</td>
</tr>
<tr>
<td># of integer functional units</td>
<td>3</td>
<td>$4 \times 1$</td>
</tr>
<tr>
<td># of floating pt. functional units</td>
<td>3</td>
<td>$4 \times 1$</td>
</tr>
<tr>
<td># of load/store ports</td>
<td>8 (one per bank)</td>
<td>$4 \times 1$</td>
</tr>
<tr>
<td>BTB size</td>
<td>2048 entries</td>
<td>$4 \times 512$ entries</td>
</tr>
<tr>
<td>Return stack size</td>
<td>32 entries</td>
<td>$4 \times 8$ entries</td>
</tr>
<tr>
<td>Instruction issue queue size</td>
<td>128 entries</td>
<td>$4 \times 8$ entries</td>
</tr>
<tr>
<td>I cache</td>
<td>32 KB, 2-way S. A.</td>
<td>$4 \times 8$ KB, 2-way S. A.</td>
</tr>
<tr>
<td>D cache</td>
<td>32 KB, 2-way S. A.</td>
<td>$4 \times 8$ KB, 2-way S. A.</td>
</tr>
<tr>
<td>L1 hit time</td>
<td>2 cycles</td>
<td>1 cycle</td>
</tr>
<tr>
<td>L1 cache interleaving</td>
<td>8 banks</td>
<td>N/A</td>
</tr>
<tr>
<td>Unified L2 cache</td>
<td>256 KB, 2-way S. A.</td>
<td>256 KB, 2-way S. A.</td>
</tr>
<tr>
<td>L2 hit time/L1 penalty</td>
<td>4 cycles</td>
<td>5 cycles</td>
</tr>
<tr>
<td>Memory latency/L2 penalty</td>
<td>50 cycles</td>
<td>50 cycles</td>
</tr>
</tbody>
</table>

Same cache total size
IPC Breakdown comparison

4x2 CMP

6-Way Scalar
Performance comparison

- With compress an application with parallelism CMP achieves 75% of SS with 3 of 4 processors idle.

- Both architectures exploit fine-grain parallelism in different ways. SS relies on dynamic extraction ILP, CMP takes moderate advantage of ILP and exploit fine-grained thread-level parallelism.
Impact

Profr. Olukotun author of this paper created Niagara, acquired by SUN in 2002.

Hottest technologies of the decade according to IEEE Spectrum
Conclusions

- In applications with large amount of parallelism CMP takes advantage of coarse-grained parallelism in addition to fine grained parallelism and ILP.
- CMP easier to implement and will reach higher clock rate.
- CMP is 10% better than SS in fine grain thread-level parallelism at the same clock rate.
- Applications with large grained thread-level parallelism and multiprogramming loads the CMP perform 50-100% better.
- In applications that cannot be parallelized the SS microarchitecture performs 30% better than one processor of CMP
Questions?

Thanks