Maximizing CMP Throughput with Mediocre Cores

John D. Davis, James Laudon†, Kunle Olukotun
Stanford University
†Sun Microsystems, Inc.
CMPs Dominate the Server Space

- What is the right solution for commercial server applications?

Montecito
USIV Dual-core Opteron
Power5

(Complex) Server Spectrum (Simple)
## Understanding Server Applications

<table>
<thead>
<tr>
<th></th>
<th>SpecWeb (web serv)</th>
<th>SpecJBB (java)</th>
<th>TPC-C (OLTP)</th>
<th>TPC-H (DSS)</th>
<th>SAP 3T (ERP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILP</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>TLP</td>
<td>High</td>
<td>High</td>
<td>High</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Inst WS Size</td>
<td>Large</td>
<td>Large</td>
<td>Large</td>
<td>Medium</td>
<td>Large</td>
</tr>
<tr>
<td>Data WS Size</td>
<td>Large</td>
<td>Large</td>
<td>Large</td>
<td>Large</td>
<td>Large</td>
</tr>
</tbody>
</table>

Adapted from *A performance methodology for commercial servers*, Kunkel et. al., IBM J. Res. Dev. Vol. 44. No.6
Rethinking Server Processors

- The quest for ever increasing performance:

<table>
<thead>
<tr>
<th></th>
<th>Monolithic Processor</th>
<th>Chip Multiprocessor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance</td>
<td>ILP</td>
<td>ILP and/or TLP</td>
</tr>
<tr>
<td>Clock Rate</td>
<td>HIGH</td>
<td>MODERATE</td>
</tr>
<tr>
<td>Design Time</td>
<td>LONG</td>
<td>SHORT</td>
</tr>
<tr>
<td>Design Complexity</td>
<td>HIGH</td>
<td>LOW</td>
</tr>
<tr>
<td>Costs</td>
<td>HIGH</td>
<td>LOW</td>
</tr>
<tr>
<td>Bugs</td>
<td>HIGH</td>
<td>LOW</td>
</tr>
</tbody>
</table>
Resulting Architecture

- Low ILP & High TLP →
  - Multithreading: Fine-grain Interleaved or SMT
- High cache miss rate →
  - High memory bandwidth
  - Shared L2 Cache
- Unpredictable control flow →
  - Simple pipeline
  - Simple branch prediction
- Is there a right combination?
  - Chip Multithreaded Multiprocessors (CMTs)
CMT Design Space Exploration

- What Space?
  - Commercial Servers
  - Select the knobs
    - Technology, core and threads
    - L2 Size, L1 parameters
- Area Model
  - Bottom-up CMT model
- Simulation Infrastructure
  - Simics + some special sauce
- Workloads
  - Commercial server benchmarks
- Results
CMT Design Space

- Pruned Simulation Space: ~13,000/benchmark/technology
- Fixed die area: number of cores depends on core and cache configuration
Leveling the Playing Field

- **Area Equivalent Comparison:** 400 mm² die
  - Small CMT: 130 nm
  - Medium CMT: 90 nm
  - Large CMT: 65 nm
- CMT: 75% of die area; I/O, etc. 25% of die area
- L2 Cache: 25%, 40%, 60%, 75% of CMT area
- Remaining core area determines the number of cores for each configuration
Core Area Model

- Linear area model
- Validated against several real designs

September 19, 2005  Maximizing Throughput  Davis © 2005
Scalar Area Model

- Hard partitioned integer pipelines
Superscalar Area Model

- Any thread, any integer pipeline
Comparing 2 Pipelines

- Increasing area with more threads
Comparing 4 Pipelines

- Superscalar cores are expensive

1.84 X

1.75 X
Simulation Infrastructure

• Simics
  – Full-system simulation infrastructure
  – Timing model for CMT (*Special Sauce*)
    • All pipeline stages
    • All threads
    • All cores
    • All memory subsystems
  – Execution driven or *trace driven*
    • Validated trace driven simulation methodology
Commercial Server Benchmarks

- Highly tuned benchmarks with < 1% idle time
- IPC $\propto$ Throughput
- Solaris 9 with 64K base page size
- SPEC JBB2000
- XML Test
- TPC-C
- TPC-W
CMT Configurations

- Completely simulated small and medium-scale CMTs
- Partially simulated large-scale CMT
- Scalar CMTs: $N_p M_t$
  - $N$: Number of scalar pipelines per core
  - $M$: Number of threads per core
- Superscalar CMTs: $N_s M_t$
  - $N$: Issue width of integer superscalar pipeline
  - $M$: Number of threads per core
Core IPC at a Glance

- Average IPC across all cores
- Core performance comparison
- L1 cache size sensitivity

<table>
<thead>
<tr>
<th>Core IPC</th>
<th>L2 Sizes (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>More</td>
<td>1.5 2.5 3.5 4.5</td>
</tr>
<tr>
<td>Less</td>
<td>NpMt</td>
</tr>
</tbody>
</table>
We Are Now Entering Space!
• Medium-scale CMT cores running SPEC JBB2000
• Multithreading improves core IPC
Multithreading or Multiple ALUs?

Medium-scale CMT cores running SPEC JBB2000

- Similar average core performance
- Save core area and gain in aggregate performance
When Does Cache Matter?

Medium-scale CMT cores running SPEC JBB2000

- High thread count per core, small L2
- Trade-off L1 and L2 cache sizes

September 19, 2005       Maximizing Throughput       Davis © 2005
Hitting the Memory Wall?

Medium-scale CMT cores running SPEC JBB2000

- Main memory bottleneck: capacity and conflict misses
- L2 cache: 25 – 40% of CMT area
Mediocre Cores

- Small-scale CMT performance for TPC-C
- C1: 64KB L1 caches, C2: 32 KB L1 caches, both 2p4t cores
- Mediocre cores: Higher Aggregate IPC = Higher Throughput
- Mediocre: adj. ordinary; of moderate … ability, or performance,
Staring into Space

TPC-W, Medium-scale CMT performance

September 19, 2005  Maximizing Throughput  Davis © 2005
Where the CMTs Fall?

Total Cores

Aggregate IPC

Superscalar

1p2t, 1.5 MB cache

3p24t, 4.5 MB cache

Superscalar

TPC-W, Medium-scale

CMT performance
Details on the Scalar CMTs

- High Thread Count, Small L1 & L2, Mediocre Cores
- Medium Thread Count, Large L1 & L2
- Simple Cores with Low Thread Count
Aggregate Performance

- Variety of scalar CMTs outperform superscalar CMTs

TPC-W, Medium-scale CMT performance

September 19, 2005  Maximizing Throughput  Davis © 2005
High Throughput CMT

- Instruction Cache (I$)
- Data Cache (D$)
- Integer Pipeline (IDP)
- Thread

- Core
- Crossbar
- 25% Shared L2 Cache
- DRAM

- Scalar Processor
  - 64KB I$
  - 32KB D$
  - 3p12t
  - OR
  - 4p16t
  - 2p8t
  - 2p16t

September 19, 2005
Maximizing Throughput
Davis © 2005
Conclusions

• Mediocre cores outperform
• Multithreading is essential for high throughput
• Scalar CMTs outperform superscalar CMTs

<table>
<thead>
<tr>
<th>Throughput Increase</th>
</tr>
</thead>
<tbody>
<tr>
<td>MT Scalar outperfroms ST Scalar</td>
</tr>
<tr>
<td>MT Superscalar outperfroms ST Superscalar</td>
</tr>
<tr>
<td>MT Scalar outperfroms MT Superscalar</td>
</tr>
</tbody>
</table>

• ∼4 threads per integer pipeline
• Small L2 Cache, 25-40% of CMT area
• Possible to saturate high bandwidth memory subsystem
Conclusions

• Insensitive to L1 cache set associativity beyond 2-ways
• Possible to saturate high bandwidth memory subsystem
• Conclusions in the paper
  – Configurations were not technology dependent
  – Single Load/Store unit was not a performance bottleneck
  – Single ported primary caches ok with small instruction buffers