EECS 452 – Lecture 10
Chip Multiprocessors

Instructor: Gokhan Memik
EECS Dept., Northwestern University
Chip-Multiprocessors

- Place multiple, relatively simple cores on a single chip
  - E.g., 32 RISC cores on a chip
  - E.g., 8 4-way VLIW
  - E.g., 4 3-way superscalar (Intel Core 2 Quad)
  - E.g., 8 8-way SMT (Sun Niagara 2)

- Simple idea, infinite possibilities/issues
Superscalar execution
Superscalar execution

![Diagram showing time and execution resources with highlighted horizontal and vertical waste areas.](image)
CMP execution
I/T-LP Architectures

<table>
<thead>
<tr>
<th></th>
<th>Superscalar</th>
<th>Multiprocessor</th>
<th>Fine-grain MT</th>
<th>SMT</th>
</tr>
</thead>
</table>
Chip-Multiprocessors – Why?

- Wide superscalar is not the way to go
  - Trades fast clock for minor IPC gain
  - Design and verification complexity
- Multiple simple processors (e.g., 2-way or 4-way superscalar)
  - Exploit TLP
  - Moderate ILP plus fast clock
- Have we seen this before?
  - Parallel processors (since the 60’s)
Designing CMPs

- The design-space for chip-multiprocessors is much larger and different than that for Shared-Memory Multiprocessors (SMPs)

- SMPs
  - Take state-of-the-art uni-processor
  - Connect several together with suitable network
    - using defined interfaces
  - Expend hardware to provide cache-coherence
    - with defined interfaces
Examples

- IBM
  - Power4, Power5 — dual-core processors in production for several years
  - Cell processor (in cooperation with Sony)
  - BlueGene chip — dual core
- Sun
  - UltraSPARC IV — dual-core
  - Niagara — 8 SMT cores
- Fujitsu
  - SPARC 64 VI — dual-core
Examples (contd)

- **AMD**
  - Quad-core Opteron

- **Intel**
  - Smithfield — dual-core Pentium4
  - Montecito — dual-core Itanium2
  - Yonah — dual-core mobile Pentium4
  - ...
  - 80-core research chip (behemoth)

- ... and others
Examples (contd)

- ARM MPcore — a quad-core synthesizable IP-core
- Broadcom SILyte — 2-4 core MIPS64 CMP
- Cavium Octeon — 2-16 (!) core MIPS64 CMP
- Freescale PowerPC 8641 — dual-core PowerPC CMP
- PMC Sierra RM11200 — dual-core MIPS64 CMP
- Clearspeed - ~1000 cores
Examples (contd)

- IBM Cell
Cores

- **SPE (synergistic processing elements)**
  - 128-bit SIMD
    - 16 8-bit integers, 8 16-bit integers, 4 32-bit integers, or 4 single precision floating-point numbers in a single clock cycle
  - 256 KB local store: instruction and data cache
  - 3.2 GHz
Alternatives – I (true CMP)

- Standard architecture
- Easily extended from existing core architectures
- Coherent L1 data caches
- Shared L2 cache reduces need for pin bandwidth

Used in:
- Piranha
- Stanford Hydra
- UltraSPARC IV
- Power4/5, BlueGene
- ARM MPcore (modified)
- Broadcom SiByte
- Cavium Octeon
Alternatives – II (SMP on a chip)

- Standard architecture
- Easily extended from existing core architectures
- Coherent L1 data caches
- Coherent L2 caches

Used in:
- AMD dual-core Opteron
- Freescale MPC8641
- PMC-Sierra
- Intel ...
Case for Chip-Multiprocessors

- The Case for a Single-Chip Multiprocessor
  - Kunle Olukotun, et al. [ASPLOS ’96]
- Introduces the notion of chip-multiprocessors
  - Put multiple cores on a single chip
- Excellent evaluation
  - What is really a good way to go
Aggressive Superscalars

- Multiple instruction issue, dynamic scheduling, speculative execution, non-blocking caches,…

- Trend
  - Wider instruction issue
  - Larger amounts of speculative execution

- However,
  - Limited amounts of ILP
  - Fundamental circuit limitations

⇒ Better resource utilization: Multiple processors on a chip
Dynamic Superscalar CPU

- Instruction queue often implemented as multiple instruction queues for different types of instructions

- Stages
  - Fetch
  - Issue
  - Execute
Fetch

- Goal: Present a large window of decoded instructions

- Constraints:
  - 1. Mispredicted branches
  - 2. Instruction misalignment
  - 3. Cache misses
Fetch (contd)

- Mispredicted branches
  - Branch predictor buffers (e.g., 64 kbits)
  - Selective branch predictor
  - Reduce misprediction < 5%

- Instruction misalignment
  - Necessary to align a packet of instructions for decoder
  - If issue width > 4, then with high probability, need fetch across a branch for a single packet of instructions
    - Need fetch from 2 cache lines and merge
    - Scheme: Divide the instruction cache into banks and fetch from multiple banks
Fetch (contd)

- Cache misses
  - High miss rate → limit ability to maintain large window
- Can hide some cache miss latency by executing other instructions already in the window (dynamically scheduled CPUs)

⇒ Overall: Fetch not limit
Issue

- Packet of renamed instructions is inserted into instruction issue queue
  - An instruction is issued when all operands are ready

- Ways of implementing renaming
  - 1. Explicit table to map architectural registers to physical ones → Problem: #ports
  - 2. Combination of reorder buffer/instruction queue
Issue (contd)

- Once instruction in instruction queue
  - Instructions that issue must update their dependences
  - Many comparators (e.g., HP-PA8000: 20% die area)

- Also: large window to find independent instructions
  - Size of instruction issue queue is large
  - Need broadcast of tags → **Wires are slow**

⇒ Overall: Instruction issue queue will limit cycle time
Execution

- **Operand values**: fetched from register file or bypassed from earlier instructions
- **Wide superscalar has problems with**
  - Register file
    - Larger to accommodate more renamed registers
    - Many ports
    - Complexity $\sim \#\text{ports}^2 \sim \text{issue width}^2$
  - Bypass logic
    - Complexity $\sim \#\text{execution units}^2$
    - **Wire delay**
  - Functional units
    - More ports needed to data cache
Single Chip-Multiprocessors

- **Technology push**
  - Benefits of wide issue are limited
  - Decentralized microarchitecture: easier to build several simple fast processors than one complex processor

- **Application pull**
  - Applications exhibit parallelism at different grains
  - < 10 instructions/cycle (INT applications)
  - > 40 instructions/cycle (loops in FP applications)
CMP

- INT applications
  - Use moderate issue processor (e.g., 2-way or 4-way) with very high clock rate

- FP applications
  - Need compiler to parallelize
  - If cannot parallelize or serial section, CMP runs slow
Architecture Comparison

- **SS**: 6-way superscalar (@ 500MHz)
- **CMP**: Four 2-way superscalars (@ 500MHz)

---

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

**Figure 3.** Floorplan for the four-way single-chip multiprocessor.
Results

- ILP only
  - SS is 30% better than CMP
- ILP & fine grain threads
  - SS and CMP comparable
- ILP & coarse grain threads
  - CMP is 1.5–2× better