Intel Nehalem

- Review entire semester by looking at most recent microprocessor from Intel
- Nehalem is code name for microarchitecture at heart of Core i7 and Xeon 5500 series server chips
- First released at end of 2008

Nehalem System Example: Apple Mac Pro Desktop 2009

- Each chip has three DRAM channels attached, each 8 bytes wide at 1.066Gbps (3*8.5GB/s).
- Can have up to two DRAMs on each channel (up to 4GB DDR3).
- "QuickPath" point-to-point system interconnect between CPUs and I/O.
- Up to 25.6 GB/s per link.
- PCI Express connections for Graphics cards and other expansion boards. Up to 8 GB/s per slot.
- Disk drives attached with 3Gb/s serial ATA link.
- Slower peripherals (Ethernet, USB, Firewire, WiFi, Bluetooth, Audio)

Nehalem Design Scalable Via Modularity

In-Order Fetch
In-Order Decode and Register Renaming
Out-of-Order Execution
In-Order Commit
Out-of-Order Completion
2 SMT Threads per Core

Nehalem Die Photo
Front-End Instruction Fetch & Decode

- 128 Entry HTLB (4 way)
- 32 KB L-cache (4 way)
- Instruction Fetch Unit
- 10B Pre-Decide, Fetch Buffer
- 6 instructions
- 16 Entry Instruction Queue
- x86 instruction bits
- µOP is Intel name for internal RISC-like instruction, into which x86 instructions are translated
- Complex Decoder
- Simple Decoder
- Simple Decoder
- Simple Decoder
- 1 pop
- 1 pop
- 1 pop
- 1 pop
- 28 Entry pop LSD Buffer
- Loop Stream Detector (can run short loops out of the buffer)

Branch Prediction

- Part of instruction fetch unit
- Several different types of branch predictor
  - Details not public
- Two-level BTB
  - How many backward branches before loop exit
  - Also predictor for length of microcode loops, e.g., string move
- Return Stack Buffer
  - Holds subroutine targets
  - Renames the stack buffer so that it is repaired after mispredicted returns
  - Separate return stack buffer for each SMT thread

x86 Decoding

- Translate up to 4 x86 instructions into uOPS each cycle
- Only first x86 instruction in group can be complex (maps to 1-4 uOPS), rest must be simple (map to one uOP)
- Even more complex instructions, jump into microcode engine which spits out stream of uOPS

Split x86 in small uOPs, then fuse back into bigger units

Loop Stream Detectors save Power

Out-of-Order Execution Engine

- Renaming happens at uOP level (not original macro-x86 instructions)
SMT effects in OoO Execution Core

- Reorder buffer (remembers program order and exception status for in-order commit) has 128 entries divided statically and equally between both SMT threads
- Reservation stations (instructions waiting for operands for execution) have 36 entries competitively shared by threads

Nehalem Memory Hierarchy Overview

- CPU Core
  - 32KB L1 D$ (8-way)
  - 32KB L1 I$ (8-way)
  - 256KB L2$ (8-way)
  - 8MB Shared L3$ (16-way)
- QuickPath System
- Interconnect

Each direction is 20b@6.4Gb/s
Each DRAM Channel is 64/72b wide at up to 1.33Gb/s

All Sockets can Access all Data

- Local Memory Access ~60ns
- Remote Memory Access ~100ns

Core’s Private Memory System

- Load queue 48 entries
- Store queue 32 entries
- Divided statically between SMT threads
- Up to 16 outstanding misses in flight per core

Cache Hierarchy Latencies

- L1 32KB 8-way, latency 4 cycles
- L2 256KB 8-way, latency <12 cycles
- L3 8MB, 16-way, latency 30-40 cycles
- DRAM, latency ~180-200 cycles
Nehalem Virtual Memory Details

- Implements 48-bit virtual address space, 40-bit physical address space
- Two-level TLB
- I-TLB (L1) has shared 128 entries 4-way associative for 4KB pages, plus 7 dedicated fully-associative entries per SMT thread for large page (2/4MB) entries
- D-TLB (L1) has 64 entries for 4KB pages and 32 entries for 2/4MB pages, both 4-way associative, dynamically shared between SMT threads
- Unified L2 TLB has 512 entries for 4KB pages only, also 4-way associative
- Additional support for system-level virtual machines

Virtualization Support

- TLB entries tagged with virtual machine and address space ID
  - No need to flush on context switches between VMs
- Hardware page table walker can walk guest-physical to host-physical mapping tables
  - Fewer traps to hypervisor

Core Area Breakdown

(Nehalem) Turbo Mode

- Power Gating
  - Zero power for inactive cores
- Turbo Mode
  - In response to workload adds additional performance bins within headroom

CSE 490/590 Administrivia

- Keyboards available for pickup at my office
- Project 2: less than 2 weeks left (Deadline 5/2)
  - Will have demo sessions
- No class on 5/2 (finish the project!)
- Final exam: Thursday 5/5, 11:45pm – 2:45pm
- Project 2 + Final = 55%

Acknowledgements

- These slides heavily contain material developed and copyright by
  - Krste Asanovic (MIT/UCB)
  - David Patterson (UCB)
- And also by:
  - Arvind (MIT)
  - Joel Emer (Intel/MIT)
  - James Hoe (CMU)
  - John Kubiatowicz (UCB)
- MIT material derived from course 6.823
- UCB material derived from course CS252