ECE/CS 757: Advanced Computer Architecture II

Instructor:Mikko H Lipasti

Spring 2009
University of Wisconsin-Madison

Lecture notes based on slides created by John Shen, Mark Hill, David Wood, Guri Sohi, and Jim Smith, Natalie Enright Jerger, and probably others

Multicore Processors

- Readings:
  - CMP design space exploration (thermal vs. power)
  - Heterogenous CMP
  - Hill's Amdahl's law
  - Piranha
  - Multicore CPUs for the masses
  - Victim replication

Objective

- Use available transistors efficiently
  - Provide better perf, perf/cost, perf/watt
- Effectively share expensive resources
  - Socket/pins:
    - DRAM interface
    - Coherence interface
    - I/O interface
- On-chip area/power
  - Mem controller
  - Cache
  - FPU? (Conjoined cores, e.g. Niagara)

High-Level Design Issues

1. Where to connect cores?
   - Time to market:
     - at off-chip bus (Pentium D)
     - at coherence interconnect (Opteron)
   - Requires substantial (re)design:
     - at L2 (Power 4, Core Duo, Core 2 Duo)
     - at L3 (Opteron, Itanium)

High-Level Design Issues

2. Share caches?
   - yes: all designs that connect at L2 or L3
   - no: all designs that don't
3. Coherence?
   - Private caches? Reuse existing MP/socket coherence
     - Optimize for on-chip sharing? [Zhang reading]
   - Shared caches?
     - Need new coherence protocol for on-chip caches
     - Often write-through L1 with back-invalidates for other caches (mini-directory)

High-Level Design Issues

4. How to connect?
   - Off-chip bus? Time-to-market hack, not scalable
   - Existing pt-to-pt coherence interconnect (hypertransport)
   - Shared L2/L3:
     - Crossbar, up to 3-4 cores (8 weak cores in Niagara)
     - 1D "dancehall" organization
   - On-chip bus? Not scalable (8 weak cores in Piranha)
   - Interconnection network
     - scalable, but high overhead
     - E.g. 2D tiled organization, mesh interconnect

Shared vs. Private Caches

- Bandwidth issues
  - Data: if shared then banked/interleaved
  - Tags: snoop b/w into L2, L1 if not inclusive
- Misses: per core vs. per chip
  - Shared: cold/capacity/conflict/comm
  - Private: cold/capacity/conflict/comm

7

Shared vs. Private Caches

- Access latency: fixed vs. NUCA (interconnect)
  - Classic UMA (dancehall) vs. NUMA
- Complexity due to bandwidth:
  - Arbitration
  - Concurrency/interaction
- Coherent vs. non-coherent shared cache
  - LLC can be "memory cache" below "coherence"

8

Multicore Coherence

- All private caches:
  - reuse existing protocol, if scalable enough
- Some shared cache
  - New LL shared cache is non-coherent (easy)
    - Use existing protocol to find blocks in private L2/L1
    - Serialize L3 access; use as memory cache
  - New shared LLC is coherent (harder)
    - Complexity of multilevel protocols is underappreciated
    - Could flatten (treat as peers) but:
      - Lose opportunity
      - May not be possible (due to inclusion, WB/WT handling)
    - Combinatorial explosion due to multiple protocols interacting

9

Multicore Coherence

- Shared L2 is coherent via writethru L1
  - Still need sharing list to forward invalidates/writes (or broadcast)
  - Ordering of WT stores and conflicting loads, coherence messages not trivial
- Shared L2 with writeback L1
  - Combinatorial explosion of multiple protocols

10

Multicore Interconnects

- Bus/crossbar - dismiss as short-term solutions?
- Point-to-point links, many possible topographies
  - 2D (suitable for planar realization)
    - Ring
    - Mesh
    - 2D torus
  - 3D - may become more interesting with 3D packaging (chip stacks)
    - Hypercube
    - 3D Mesh
    - 3D torus

11

On-Chip Bus/Crossbar

- Used widely (Power4/5/6, Piranha, Niagara, etc.)
  - Assumed not scalable
  - Is this really true, given on-chip characteristics?
  - May scale "far enough" : watch out for arguments at the limit
- Simple, straightforward, nice ordering properties
  - Wiring is a nightmare (for crossbar)
  - Bus bandwidth is weak (even multiple busses)
  - Compare piranha 8-lane bus (32GB/s) to Power4 crossbar (100+GB/s)
  - Workload: commercial vs. scientific

12

On-Chip Ring

- Point-to-point ring interconnect
  - Simple, easy
  - Nice ordering properties (unidirectional)
  - Every request a broadcast (all nodes can snoop)
  - Scales poorly: O(n) latency, fixed bandwidth
- Optical ring (nanophotonic)
  - HP Labs Corona project
  - Latency is arguably O(sqrt(n))
    - Covert switching – broadcast not easy any more
  - Still fixed bandwidth (but lots of it)

13

On-Chip Mesh

- Widely assumed in academic literature
- Tilera, Intel 80-core prototype
- Not symmetric, so have to watch out for load imbalance on inner nodes/links
  - 2D torus: wraparound links to create symmetry
    - Not obviously planar
    - Can be laid out in 2D but longer wires, more intersecting links
- Latency, bandwidth scale well
- Lots of existing literature

14

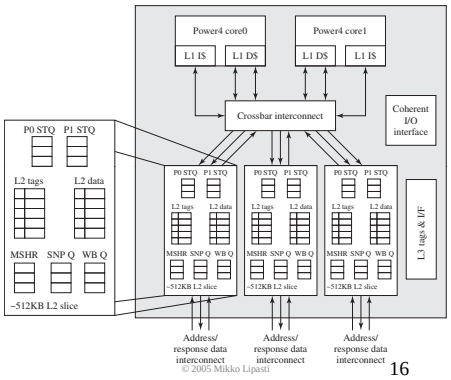
CMP Examples

- Chip Multiprocessors (CMP)
- Becoming very popular

Processor	Cores/ chip	Multi-threaded?	Resources shared
IBM Power 4	2	No	L2/L3, system interface
IBM Power 5	2	Yes (2T)	Core, L2/L3, system interface
Sun Ultrasparc	2	No	System interface
Sun Niagara	8	Yes (4T)	Everything
Intel Pentium D	2	Yes (2T)	Core, nothing else
AMD Opteron	2	No	System interface (socket)

15

IBM Power4: Example CMP



16

Multithreading vs. Multicore

MT Approach	Resources shared between threads	Context Switch Mechanism
None	Everything	Explicit operating system context switch
Fine-grained	Everything but register file and control logic/state	Switch every cycle
Coarse-grained	Everything but I-fetch buffers, register file and con trol logic/state	Switch on pipeline stall
SMT	Everything but instruction fetch buffers, return address stack, architected register file, control logic/state, reorder buffer, store queue, etc.	All contexts concurrently active; no switching
CMT	Various core components (e.g. FPU), secondary cache, system interconnect	All contexts concurrently active; no switching
CMP	Secondary cache, system interconnect	All contexts concurrently active; no switching

- Many approaches for executing multiple threads on a single die
  - Mix-and-match: IBM Power5 CMP+SMT

17

Multicore Summary

- Objective: resource sharing
  - Where to connect
  - Cache sharing
  - Coherence
  - How to connect
- Readings

18