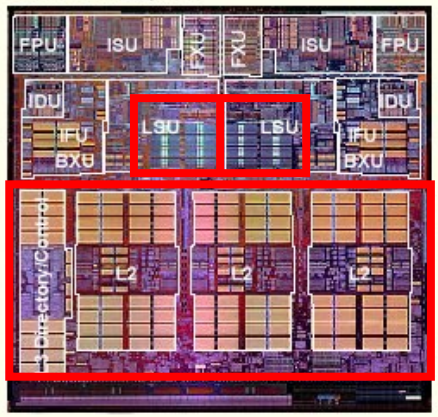
18-742

Lecture 22

Chip Multiprocessors

Spring 2005
Prof. Babak Falsafi
http://www.ece.cmu.edu/~ece742



Slides developed in part by Profs. Falsafi from Hill, Olukotun and Stets of Carnegie Mellon University, Google, Stanford University, and University of Wisconsin.

Readings

Papers and lecture notes only

Reader 7

- **J. M. Tendler, J. S. Dodson, J. S. Fields, Jr., H. Le, and B. Sinharoy, *POWER4 system microarchitecture*, IBM J. of Research and Development, 2002, vol. 46, no. 1.**
- **G. Sohi, S. Breach, T. N. Vijaykumar, *Multiscalar Processors*, ISCA 1995.**

The Parallel Computing Problem

Conventionally:

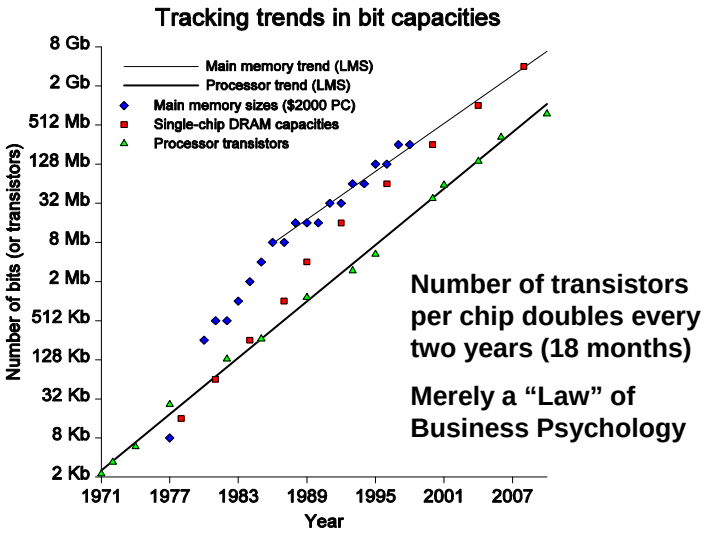
- **Limited to supercomputing**
- **Both research & academic prototypes**

Recently:

- **Server-class computing**
- **Customized (e.g., Google cluster)**
- **General-purpose (e.g., SMPs)**
- **DSMs provide scalability but have not found wide use**
  - **Why?**

But, why not in mainstream computing?

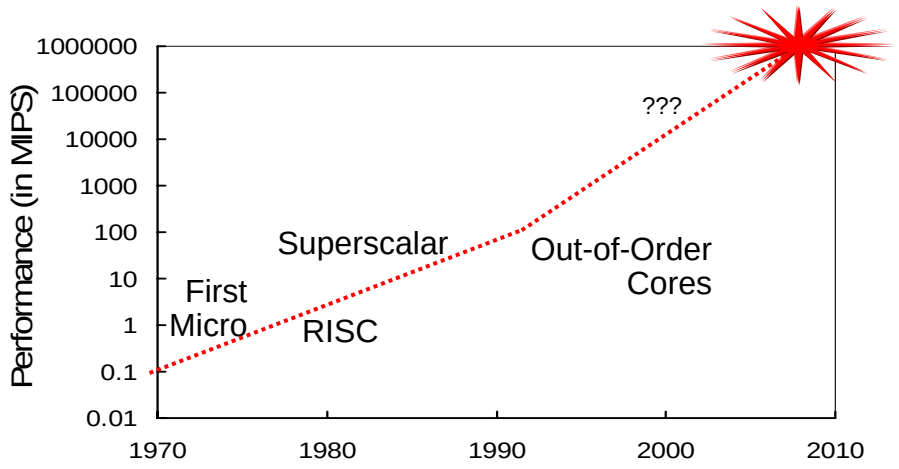
Moore's Law: Scaling the Number of Transistors



Number of transistors per chip doubles every two years (18 months)

Merely a "Law" of Business Psychology

Chip Performance Scalability: History & Expectations



Goal: 1 Tera inst/sec by 2010!

Microprocessors Riding the Moore Curve

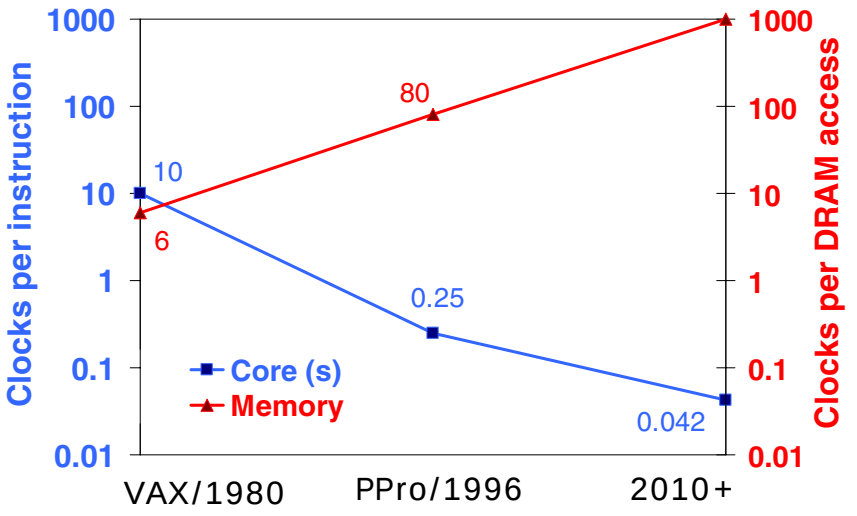
Single thread programming & execution model

- **Killed the prospects of wide spread use of MPs**

But, single-threaded chips have run out of steam!

- **ILP and Memory are the culprits**
- **When ILP or MLP are high**
  - **Scaling core design would be ideal, but not practical**
- **When ILP and MLP are low**
  - **Scaling core design is not cost-effective**

Remember the Memory Wall



Logic/DRAM speed gap continues to increase!

The Current Memory System Approach

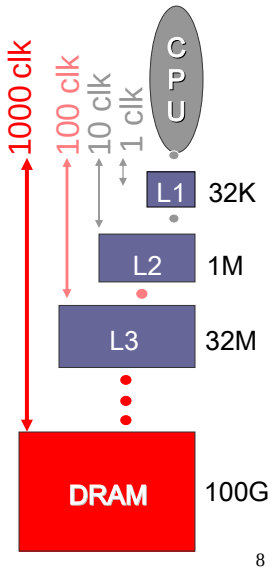
Cache hierarchies

- **Trade off capacity for speed**
- **Exploit "reuse"**
- **Demand fetch/rep. data**

But, in modern servers

- **Only 50% CPU utilization [Ailamaki, VLDB'99]**

Bigger problem for MPs



Future of Chips: Chip Multithreading/Multiprocessing

Chip multithreading/multiprocessing:

- **Chips and cores running multiple threads**

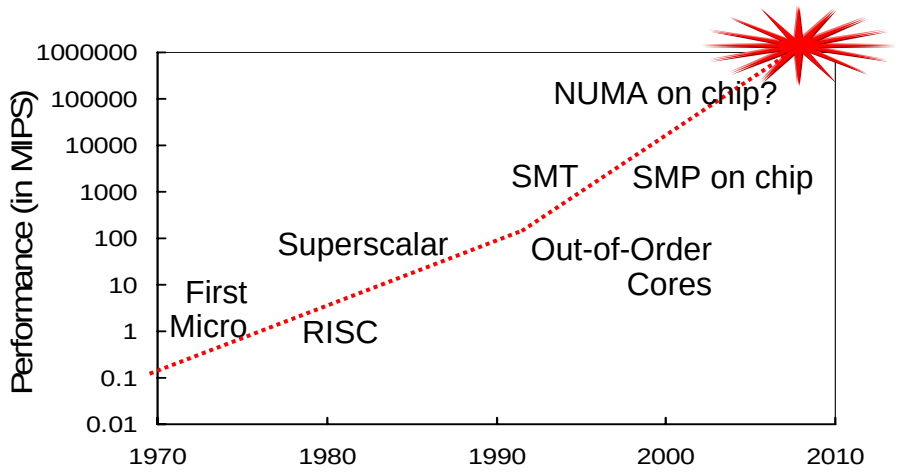
Good news for servers:

- **Servers use MP software**
- **Can port to CMPs/CMTs**
- **Lots of "narrow" cores for higher thread-level MLP**
  - **Why?**

Bad news for mainstream chips:

- **Most other important software is not UP**
- **Need techniques to parallelize these**
  - **By hand (considered difficult)**
  - **By compiler (research in the past 40 years, FORTRAN products)**
  - **By hardware (research in the past 10 years, Sun MAJC)**

Chip Performance Scalability: History & Expectations



Goal: 1 Tera inst/sec by 2010!

Piranha: Performance and Managed Complexity

- **Large-scale DSM based on CMP nodes**
- **CMP architecture**
  - **excellent platform for exploiting thread-level parallelism**
  - **inherently emphasizes replication over monolithic complexity**
- **Design methodology reduces implementation complexity**
  - **novel simulation methodology**
  - **use ASIC physical design process**
- ***Piranha*: *2x performance advantage with team size of approximately 20 people***

Piranha Processing Node

Alpha core:
1-issue, in-order,
500MHz

CPU

Next few slides from

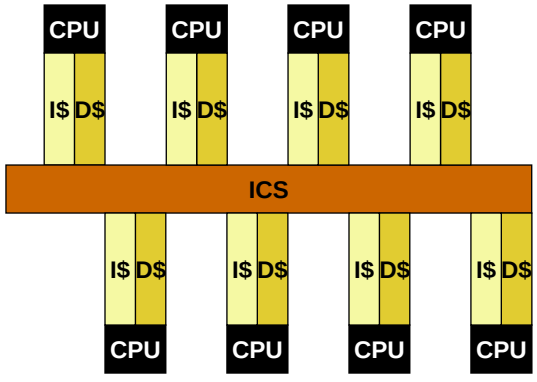
Luiz Barosso's ISCA 2000 presentation of

Piranha: A Scalable Architecture
Based on Single-Chip Multiprocessing

Piranha Processing Node

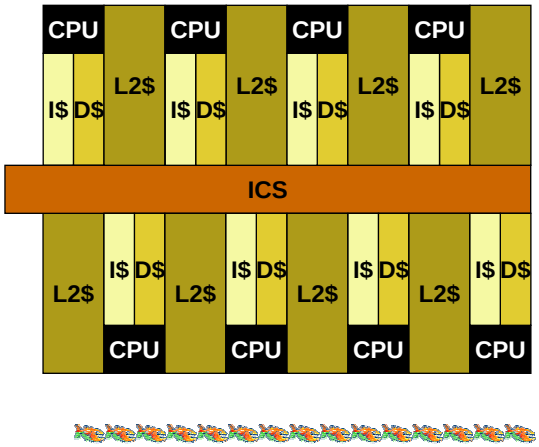
Alpha core:
1-issue, in-order, 500MHz
L1 caches:
I&D, 64KB, 2-way

Piranha Processing Node



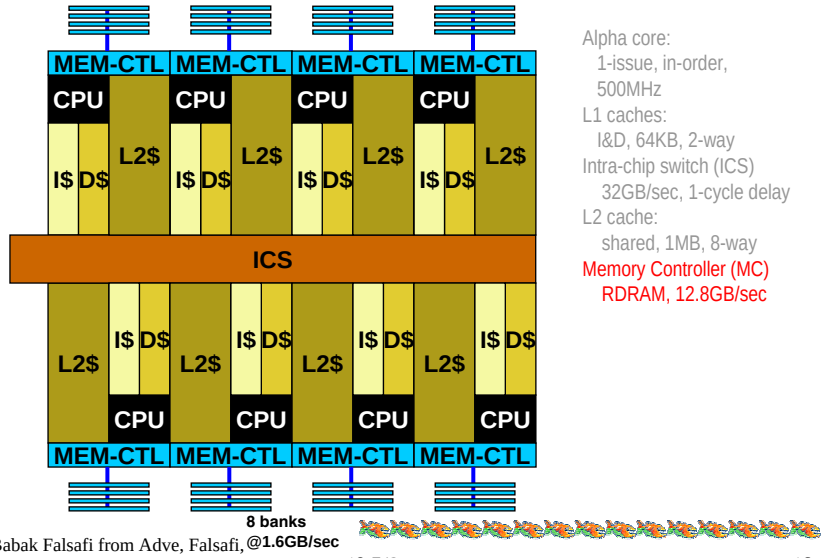
Alpha core:
1-issue, in-order, 500MHz
L1 caches:
I&D, 64KB, 2-way
Intra-chip switch (ICS)
32GB/sec, 1-cycle delay

Piranha Processing Node



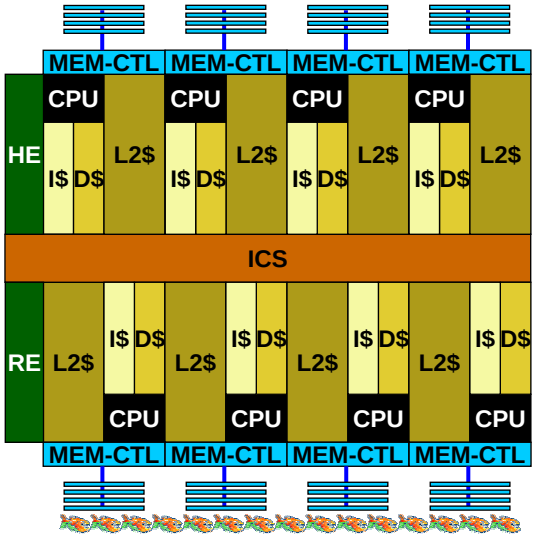
Alpha core:
1-issue, in-order, 500MHz
L1 caches:
I&D, 64KB, 2-way
Intra-chip switch (ICS)
32GB/sec, 1-cycle delay
L2 cache:
shared, 1MB, 8-way

Piranha Processing Node



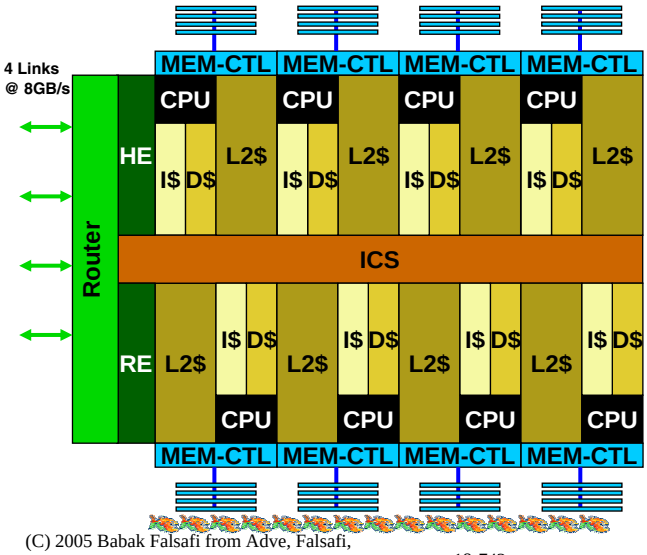
Alpha core:
1-issue, in-order, 500MHz
L1 caches:
I&D, 64KB, 2-way
Intra-chip switch (ICS)
32GB/sec, 1-cycle delay
L2 cache:
shared, 1MB, 8-way
Memory Controller (MC)
RDRAM, 12.8GB/sec

Piranha Processing Node



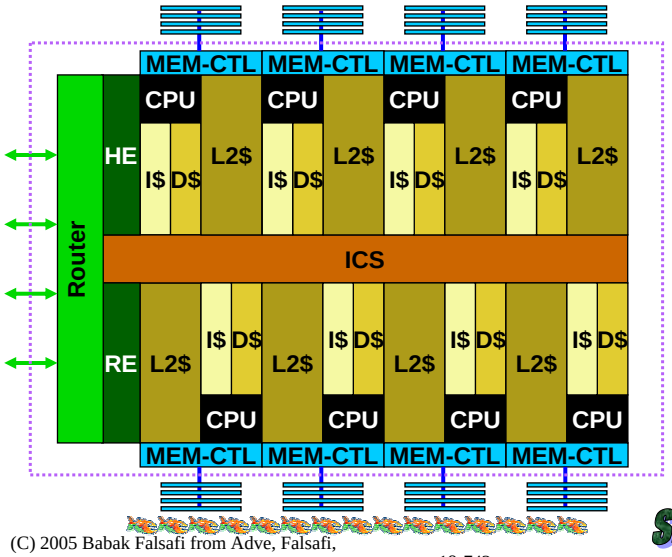
Alpha core:
1-issue, in-order, 500MHz
L1 caches:
I&D, 64KB, 2-way
Intra-chip switch (ICS)
32GB/sec, 1-cycle delay
L2 cache:
shared, 1MB, 8-way
Memory Controller (MC)
RDRAM, 12.8GB/sec
Protocol Engines (HE & RE)
µprog., 1K µinstr., even/odd interleaving

Piranha Processing Node



Alpha core:
1-issue, in-order, 500MHz
L1 caches:
I&D, 64KB, 2-way
Intra-chip switch (ICS)
32GB/sec, 1-cycle delay
L2 cache:
shared, 1MB, 8-way
Memory Controller (MC)
RDRAM, 12.8GB/sec
Protocol Engines (HE & RE):
µprog., 1K µinstr., even/odd interleaving
System Interconnect:
4-port Xbar router
topology independent
32GB/sec total bandwidth

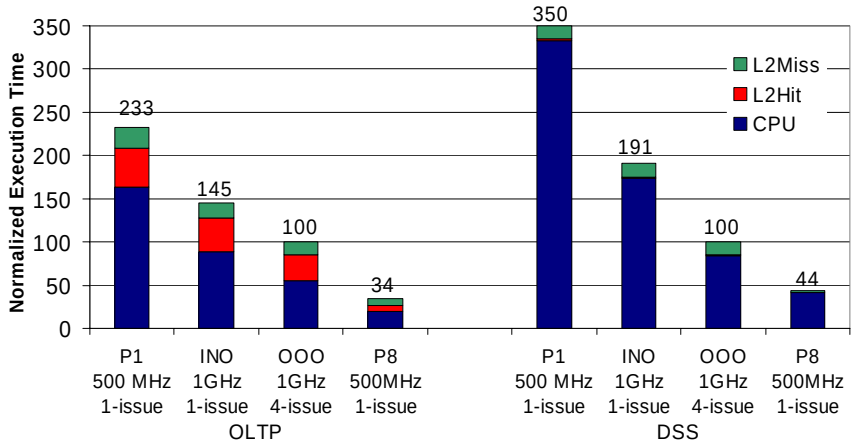
Piranha Processing Node



Alpha core:
1-issue, in-order, 500MHz
L1 caches:
I&D, 64KB, 2-way
Intra-chip switch (ICS)
32GB/sec, 1-cycle delay
L2 cache:
shared, 1MB, 8-way
Memory Controller (MC)
RDRAM, 12.8GB/sec
Protocol Engines (HE & RE):
µprog., 1K µinstr., even/odd interleaving
System Interconnect:
4-port Xbar router
topology independent
32GB/sec total bandwidth

Single Chip

Single-Chip Piranha Performance



- **Piranha's performance margin 3x for OLTP and 2.2x for DSS**
- **Piranha has more outstanding misses ➔ better utilizes memory system**

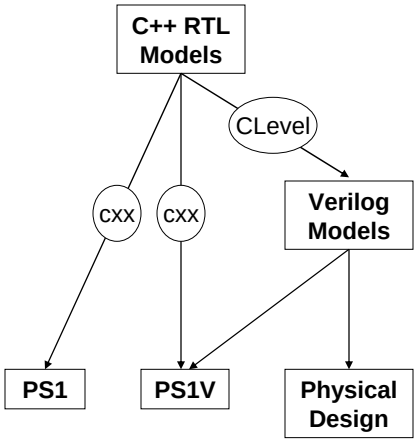
Functional Verification

- **Traditional approach: *isolated sub-unit testing***
  - **often necessitated by Verilog simulation speed**
- **Leads to high infrastructure and management overhead**
  - **need to construct specialized test harnesses**
  - **requires considerable effort to create realistic stimuli**
  - **complicates coordination of testing coverage**
- **Especially high overhead for a tightly integrated design!**
  - **e.g. Piranha memory hierarchy**

Reducing Verification Overhead in Piranha

- **Test fully integrated *memory hierarchy***
  - **implement memory hierarchy in C++, rather than Verilog**
  - **separate simulation of Memory Hierarchy (i.e. System) and Alpha Core**
    - » **allows for more efficient simulation**
- **Heavily leverage pseudo-random, self-checking tests**
- **Approach should not increase overhead in other areas**
  - **e.g. use of C++ should not increase physical design overhead**

System Simulation Methodology



cxx: C++ compiler

CLevel: C++-to-Verilog Translator

C++ RTL Models: Cycle accurate and structural

PS1: Fast (C++) logic simulator

Verilog Models: Machine translated from C++ models

Physical Design: Leverages Verilog-based tools

PS1V: Can "co-simulate" C++ and Verilog module versions and check correspondence

System Simulation Methodology

- ***C++-based simulator*** **enables simulation of full memory hierarchy**
  - **delivers 1000 simulated clocks per second for CMP node**
  - **roughly 50x faster than comparable Verilog simulation**
- **Machine C++-Verilog translation is promising**
  - **enables single code-base**
  - **resulting Verilog synthesizes well**
  - **be careful with C++ module size**

ASIC Physical Design

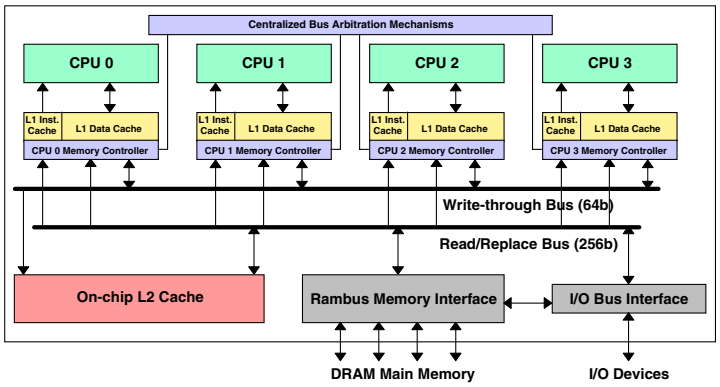
- ***ASIC process*** **automates much of low-level physical layout**
- **Disadvantages compared to full-custom design**
  - **relatively slow logic**
  - **typically larger size and high power dissipation**
  - **possible vendor design restrictions to enable physical validation**
    - » **e.g. no tri-state busses to avoid possibility of electrical shorts**
- **Piranha target: IBM's Cu11 (0.13 micron) process**

Piranha Physical Design

Module	Size mm^2		Timing Status
	Logic	Memory	
Alpha Core	**2.1**	**0.3**	**~400MHz**
L1 Cache	**2.0**	**5.8**	**early re-designs**
L2 Cache	**2.3**	**9.9**	**early re-designs**
Coherence Engines	**1.4**	**1.2**	**early re-designs**
I/O Queues	**1.1**	**0.6**	**~400MHz**
Router	**0.5**	**1.1**	**~400MHz**
Intra-chip Switch	**1.4**	**0**	**early re-designs**
Memory Controller	**4.8**	**0**	**~400MHz**
System Controller	**1.0**	**0**	**early re-designs**
Miscellaneous	**54.6**	**1.0**	**in implementation**
Total	282.6	133.1	---

- Maximum estimated power dissipation at 400MHz: 44W

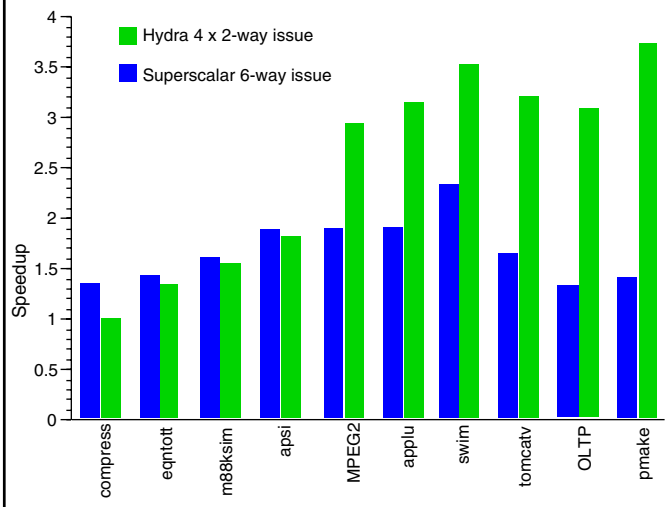
The Base Hydra Design



- Single-chip multiprocessor
- Four processors
- Separate primary caches
- Write-through data caches to maintain coherence

  - Shared 2nd-level cache
  - Low latency interprocessor communication (10 cycles)
  - Separate read and write buses

Hydra vs. Superscalar



- ILP only
  ⇒ SS 30-50% better than single Hydra processor
- ILP & fine thread
  ⇒ SS and Hydra comparable
- ILP & coarse thread
  ⇒ Hydra 1.5–2× better