Computer Architecture
计算机体系结构

Lecture 9. CMP and Multicore System
第九讲、片上多处理器与多核系统

Chao Li, PhD.
李超 博士

SJTU-SE346, Spring 2019

Review

- Classification of parallel architectures
- Shared-memory system
- Cache coherency problem
- Snooping protocol
- A simple write-through invalidation protocol
- 3-state MSI protocol

Exercises

- What would happen if a cache has the block in modified state and it observes a BusRd transaction on the bus?

- How would register allocation affect semantics in a parallel program (showing below) running on a multiprocessor?

```
/* Assume the initial value of A and flag is 0 */
 P1                P2
 A = 1;            while (flag == 0); /*spin idly */
 flag = 1;         print A;
```

Outlines

- Thread-Level Parallelism
- Introduction to Multicore
- Design Space Exploration
- From Multicore to Manycore

Thread-Level Parallelism

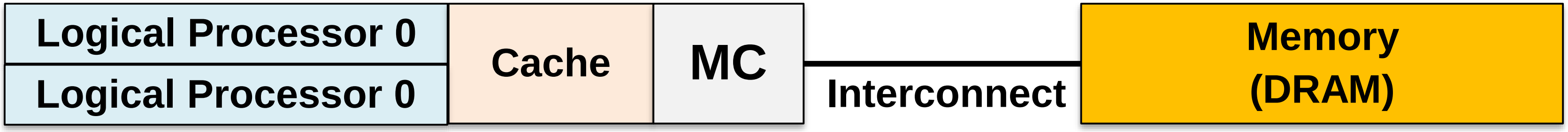
- Thread: a basic unit of processor utilization
  - Has all the states necessary to allow it to execute
  - Thread in the same process share code and data
- The term thread here is often used in a casual way:
  - May be a subpart of a parallel program (the real “thread”)
  - May be an independent program (a heavyweight process)
- Thread-Level Parallelism
  - The use of multiple thread of execution that are inherently parallel
  - The grain size is the amount of computation within each thread

The Need for Multithreading

- Long **memory stalls** are unlikely to be hidden by available ILP – the utilization of FU drops dramatically

- Modern processors typically use **hardware multithreading** to keep the otherwise idle on-chip resources busy

- **Multithreaded Processor**: can execute multiple instruction streams from multiple threads in parallel
  - Duplicates processor resources, such as registers and PC
  - Addition of logic to the pipeline to switch between threads



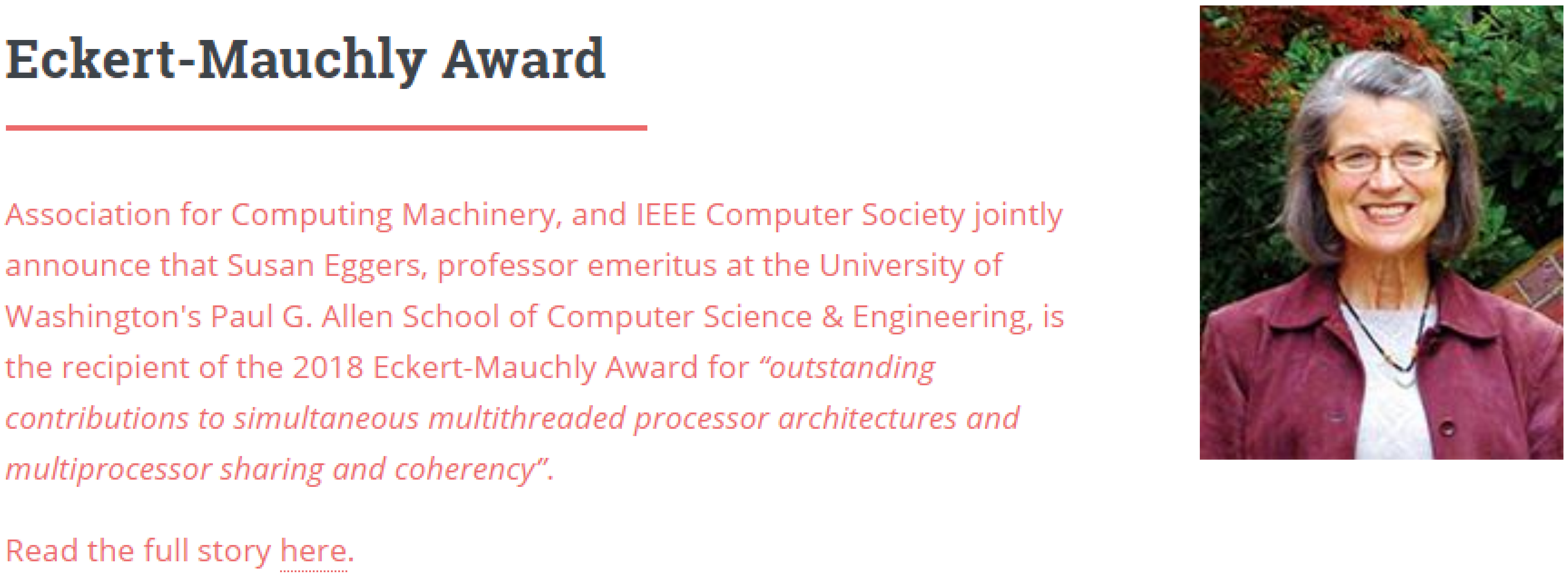
Style of Multithreading

- Fine-grained (interleaved) multithreading
  - Switches between threads on each clock cycle
  - Slows down the execution of an individual thread

- Coarse-grained multithreading
  - Long latency stalls trigger the thread switch
  - Short-latency events cannot be hidden

- Simultaneous multithreading (**SMT**) 同步多线程
  - Instructions may be issued from multiple threads during the same cycle (e.g., Intel’s Hyper-Threading)

SMT and Eckert-Mauchly Award



Eckert-Mauchly Award

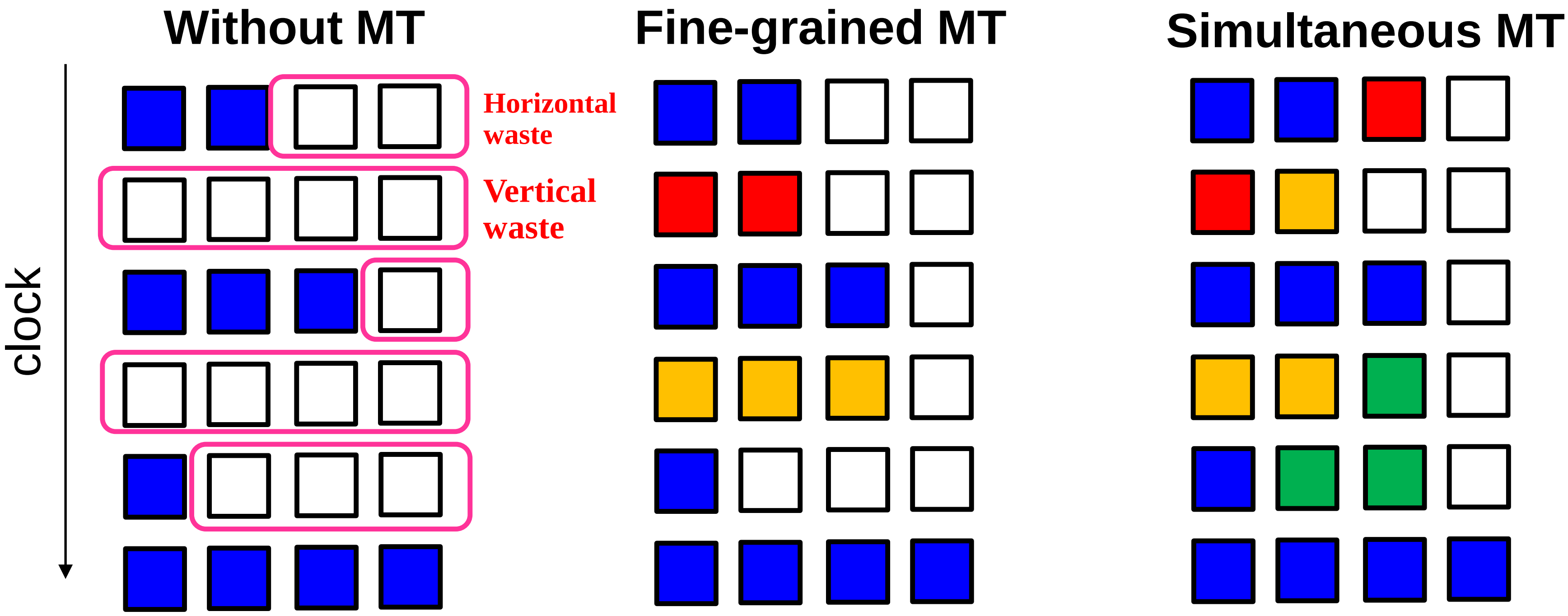
Association for Computing Machinery, and IEEE Computer Society jointly announce that Susan Eggers, professor emeritus at the University of Washington's Paul G. Allen School of Computer Science & Engineering, is the recipient of the 2018 Eckert-Mauchly Award for *"outstanding contributions to simultaneous multithreaded processor architectures and multiprocessor sharing and coherency"*.

Read the full story here.

Widely recognized as one of the leading computer architects in the field, Eggers will be the first woman to receive the Eckert-Mauchly Award in its 39-year history. She is also atypical among engineers in that she received a BA degree in Economics in 1965 and worked in related fields for 18 years before deciding to switch careers and pursue research in computer engineering. In 1983 she joined the graduate program in the Department of Electrical Engineering and Computer Sciences at the University of California, Berkeley and began working toward a PhD. She completed her PhD in 1989, starting her faculty career as assistant professor at the University of Washington at the age of 47.

Impacts of SMT on Utilization

- Multithreaded processor improves hardware utilization in different dimensions



Thread 1 Thread 2 Thread 3 Thread 4

Impacts of SMT on Utilization

- SMT can exploit the parallelism of independent programs or the parallelism in a single program

- The scheduling and mixing of instructions is done by hardware (no compiler support)

- Requires fetching from multiple program counters and accessing multiple complex register sets

Chip Multi-Processor (CMP)

- Multiprocessors have been around a long time
  - Just not on a single chip (e.g., supercomputers, mainframes)
- CMP is a special type of multiprocessor
  - Multiple cores fit on a single processor socket
- Also known as multicore processors
- In general, multicore processors are:
  - Shared-memory multiprocessor
  - Multiple Instruction Multiple Data (MIMD)

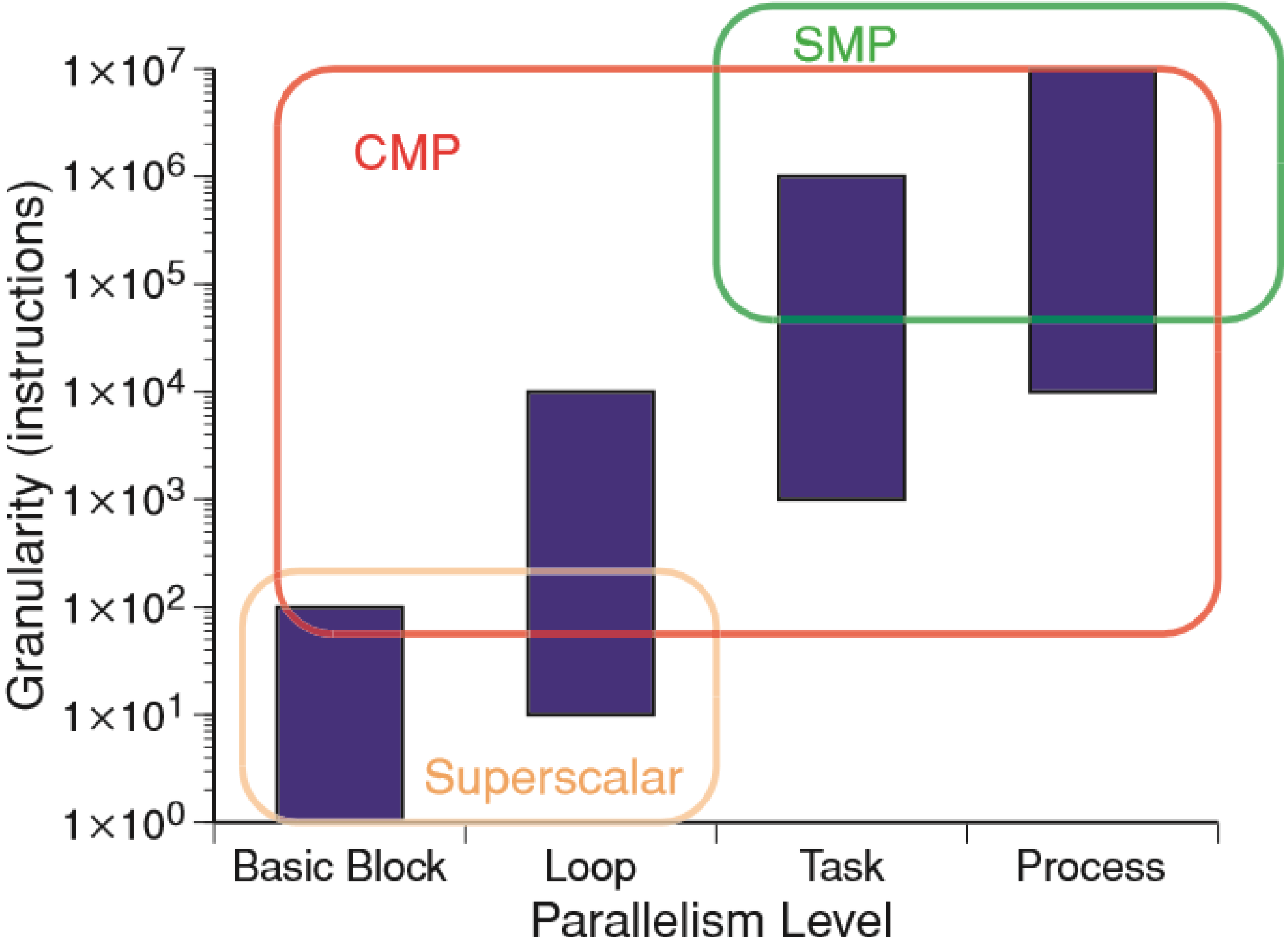
CMP vs. SMT

- CMP: chip multi-processing
  - Multiple physical cores that have unique resources
  - L1 cache, TLB, PC, GPR are unique; L2 cache may be shared

- SMT: simultaneous multithreading
  - Multiple threads that share all the processor resources
  - Caches and TLBs are shared, PC and GPR unique

- HT: hyper threading
  - Intel's SMT technology

Parallelism Comparison



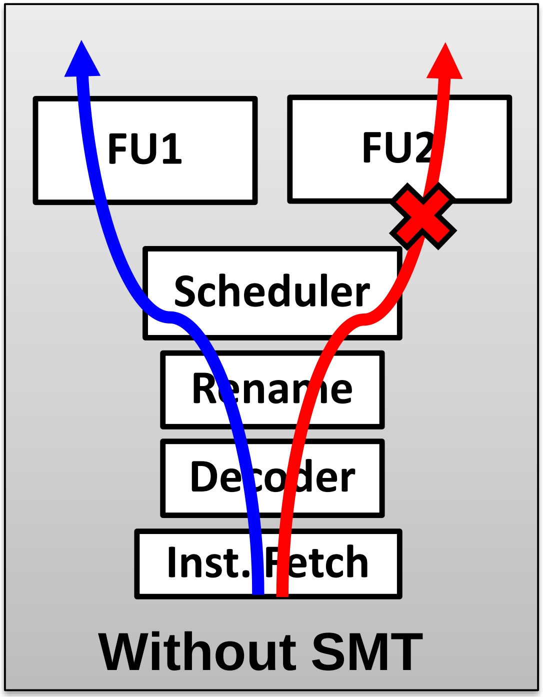
A summary of the various "ranges" of parallelism that different processor architectures may attempt to exploit.

Outlines

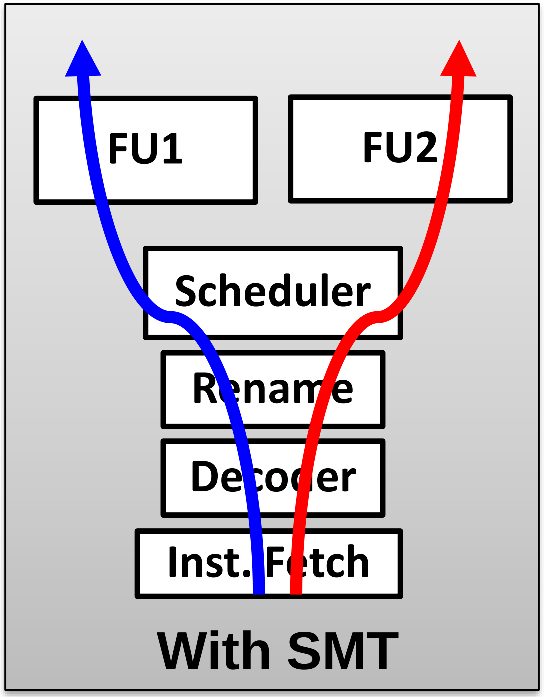
- Thread-Level Parallelism
- Introduction to Multicore
- Design Space Exploration
- From Multicore to Manycore

Why Multicore?

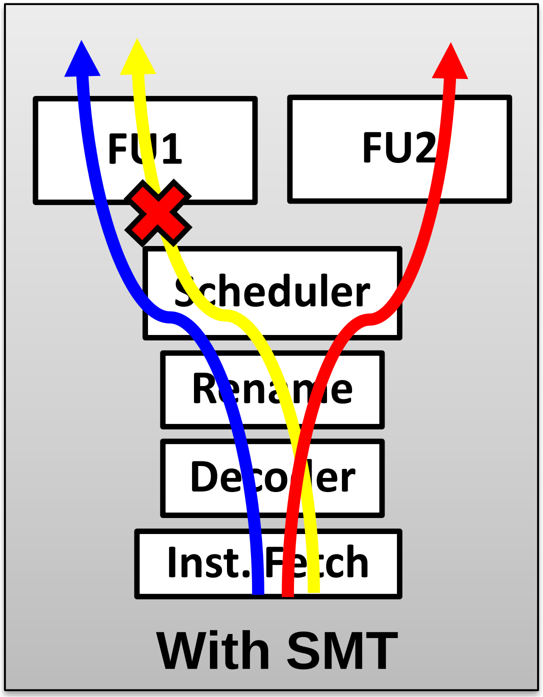
- Single-core superscalar processors cannot fully exploit TLP



Only a single thread can run at any given time



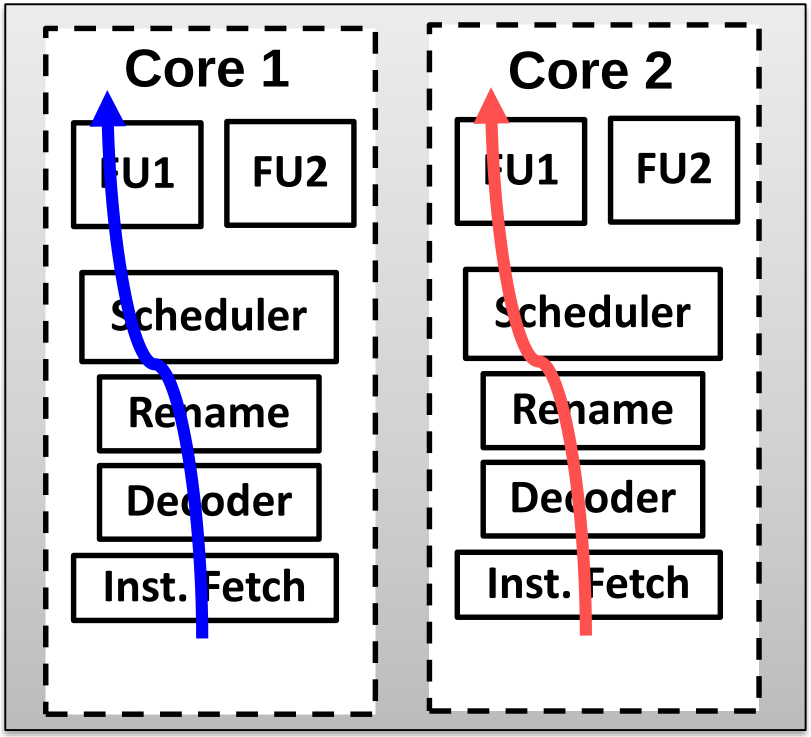
Both threads can run concurrently



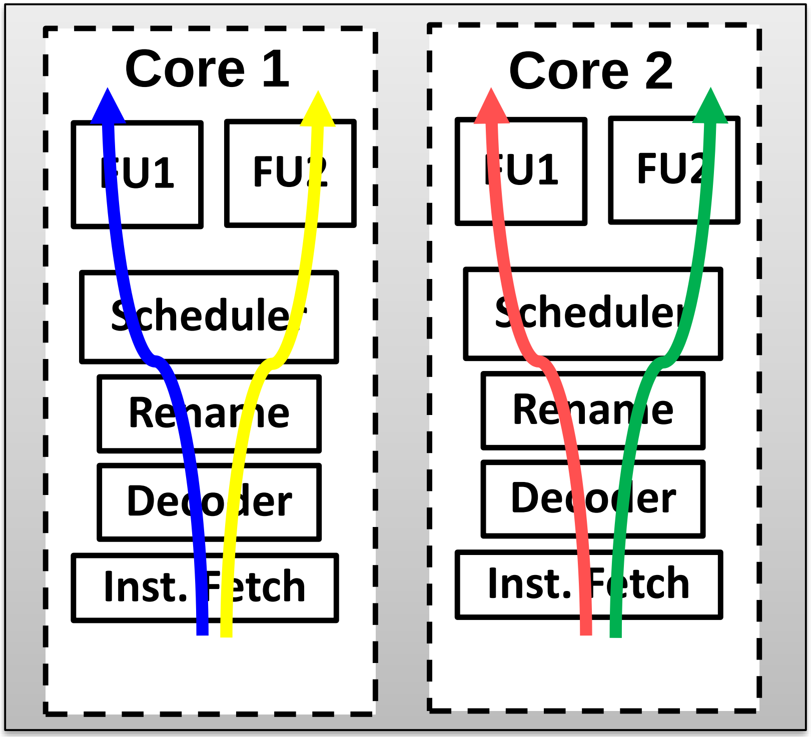
Can’t simultaneously use the same FU on a single core

Why Multicore? (Cont’d)

- Multi-core architectures explicitly exploits TLP



Threads can run on separate cores



SMT Dual-core: 4 threads can run concurrently

Why Multicore? (Cont'd)

- Single core SMT:
  - Mostly still only exploits instruction-level parallelism
  - To execute the tasks faster we must increase the clock frequency
  - Drastically increases power consumption and heat dissipation
  - increasingly time-consuming design, difficult verification

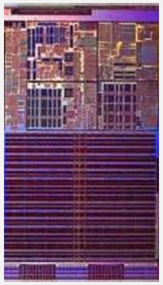
- Multicore solution:
  - Great with thread-level parallelism
  - Integrating two or more cores on the same chip
  - Utilizing cores running at an efficient frequency level
  - Uses proven processor designs with lower manufacturing cost

Power Efficiency of Multicore

A 15% Reduction in Voltage

Frequency Reduction	Power Reduction	Performance Reduction
15%	45%	10%

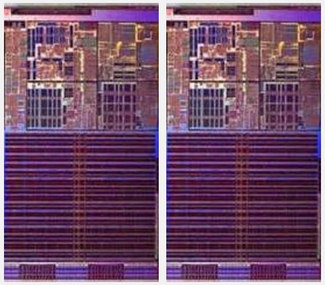
Single Core



Area	= 1
Voltage	= 1
Freq.	= 1
Power	= 1
Perf.	= 1

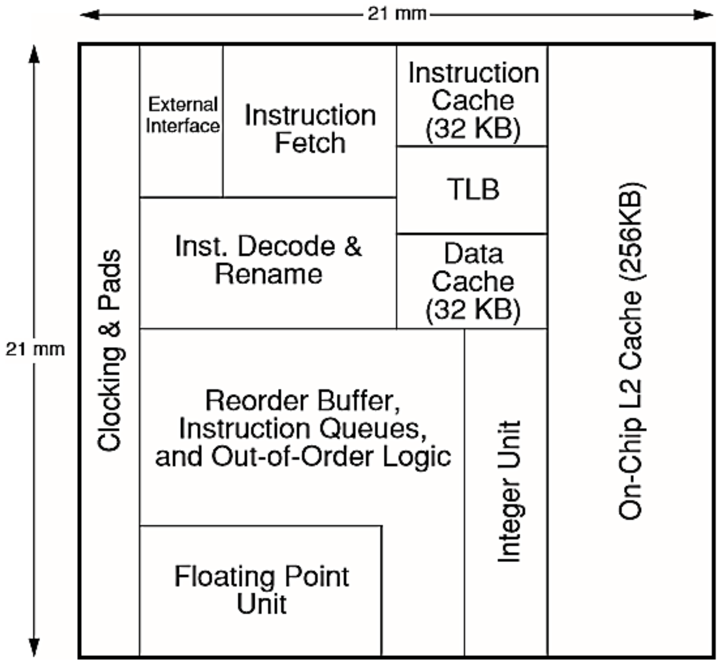
V.S.

Dual Core

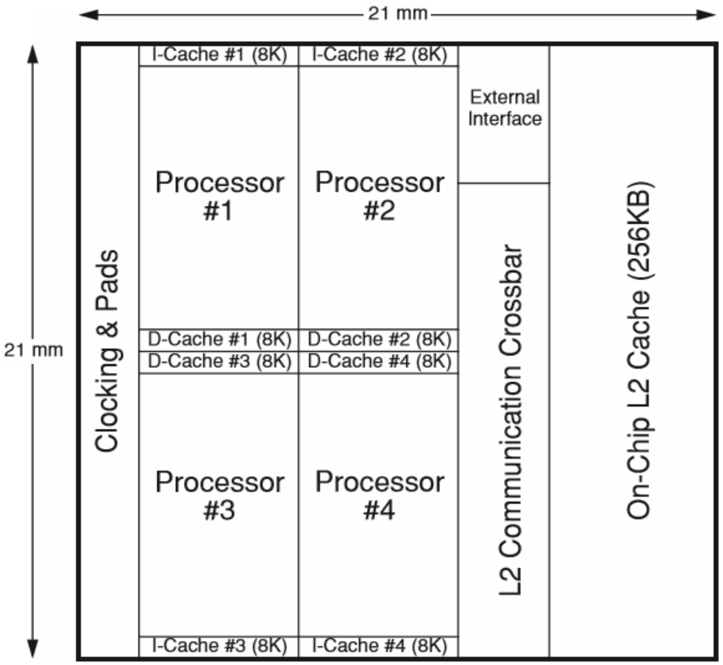


Area	= 2
Voltage	= 0.85
Freq.	= 0.85
Power	= 1.1
Perf.	~ 1.8

Case Study: Large Superscalar vs. CMP



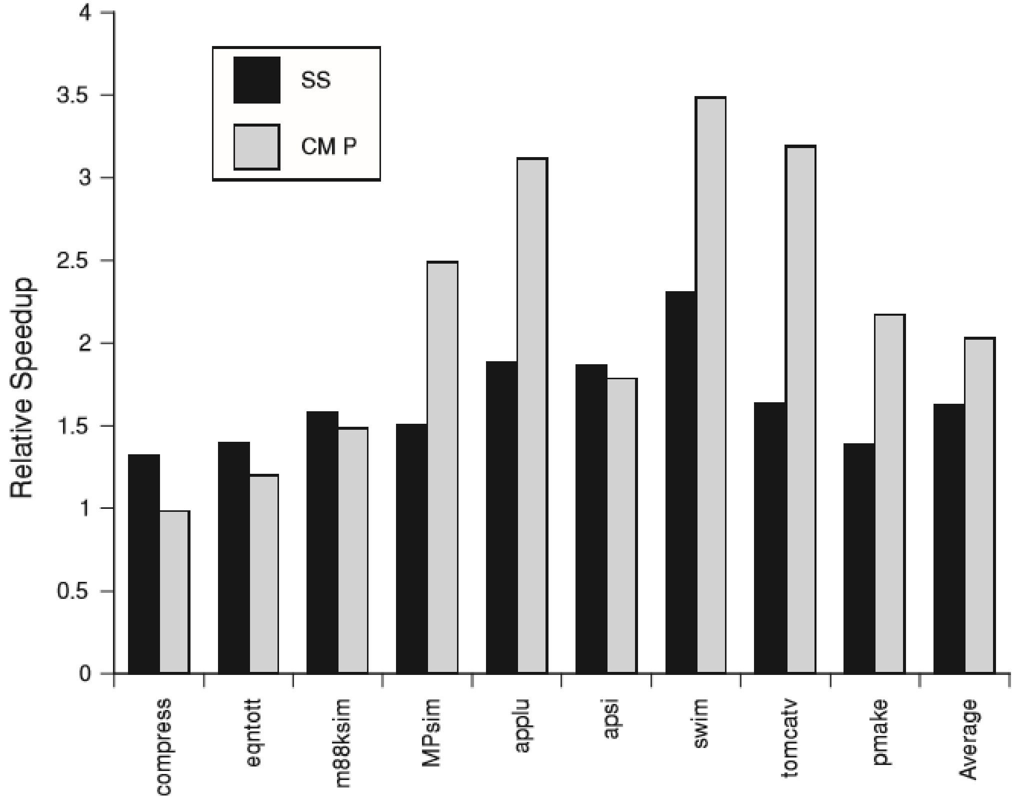
6-issue superscalar processor



4×2-way chip-multiprocessor

- Assumption: Identical processing technology, same die area, same off-chip resources, same clock frequency

Case Study: Large Superscalar vs. CMP (Cont'd)

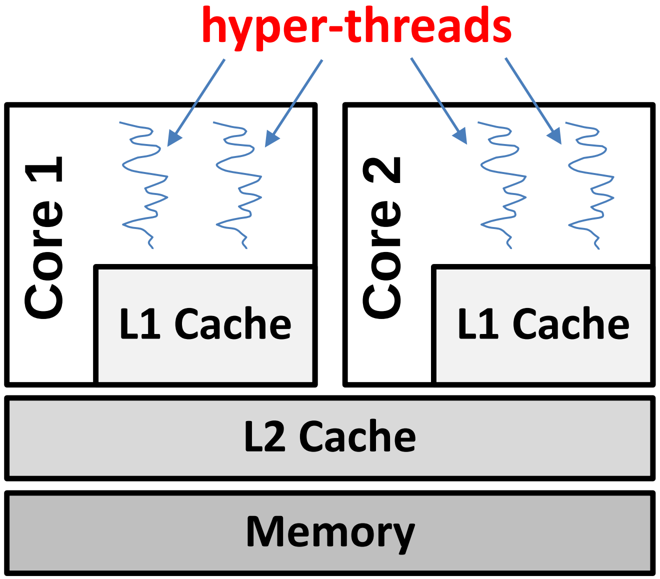


Performance comparison of SS and CMP (relative to a single 2-issue processor running alone)

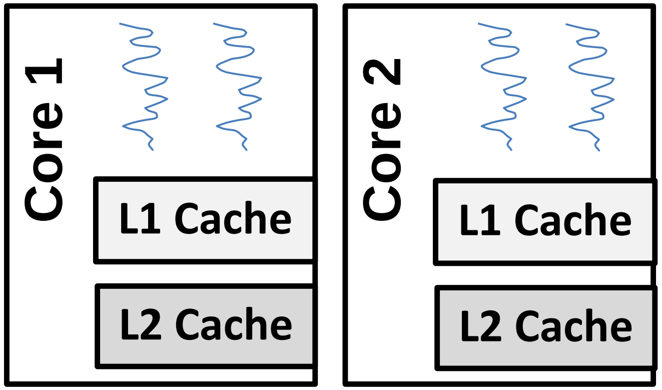
- Large Superscalar:
  - Extract ILP from a single thread
  - Requires minor programmer effort
- CMP:
  - Some ILP + Some fine-grained TLP
  - Requires more programmer effort
  - Favors large amount of parallelism

Typical Multicore Cache Organization

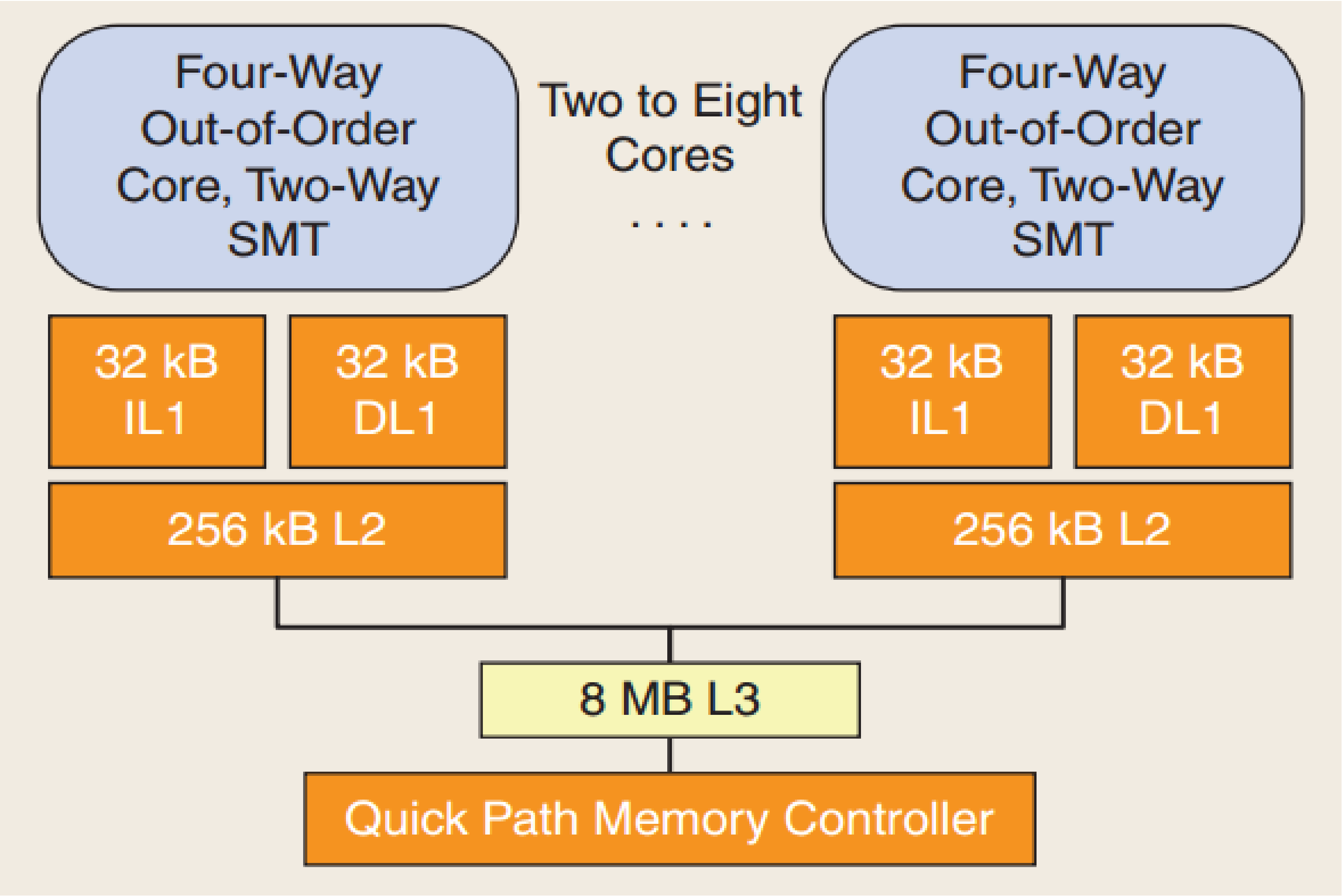
- Private L1 + Shared L2
- Example:
  - Dual-core Xeon Processors



- Both L1 and L2 are private
- Example:
  - AMD Athlon, Intel Pentium D

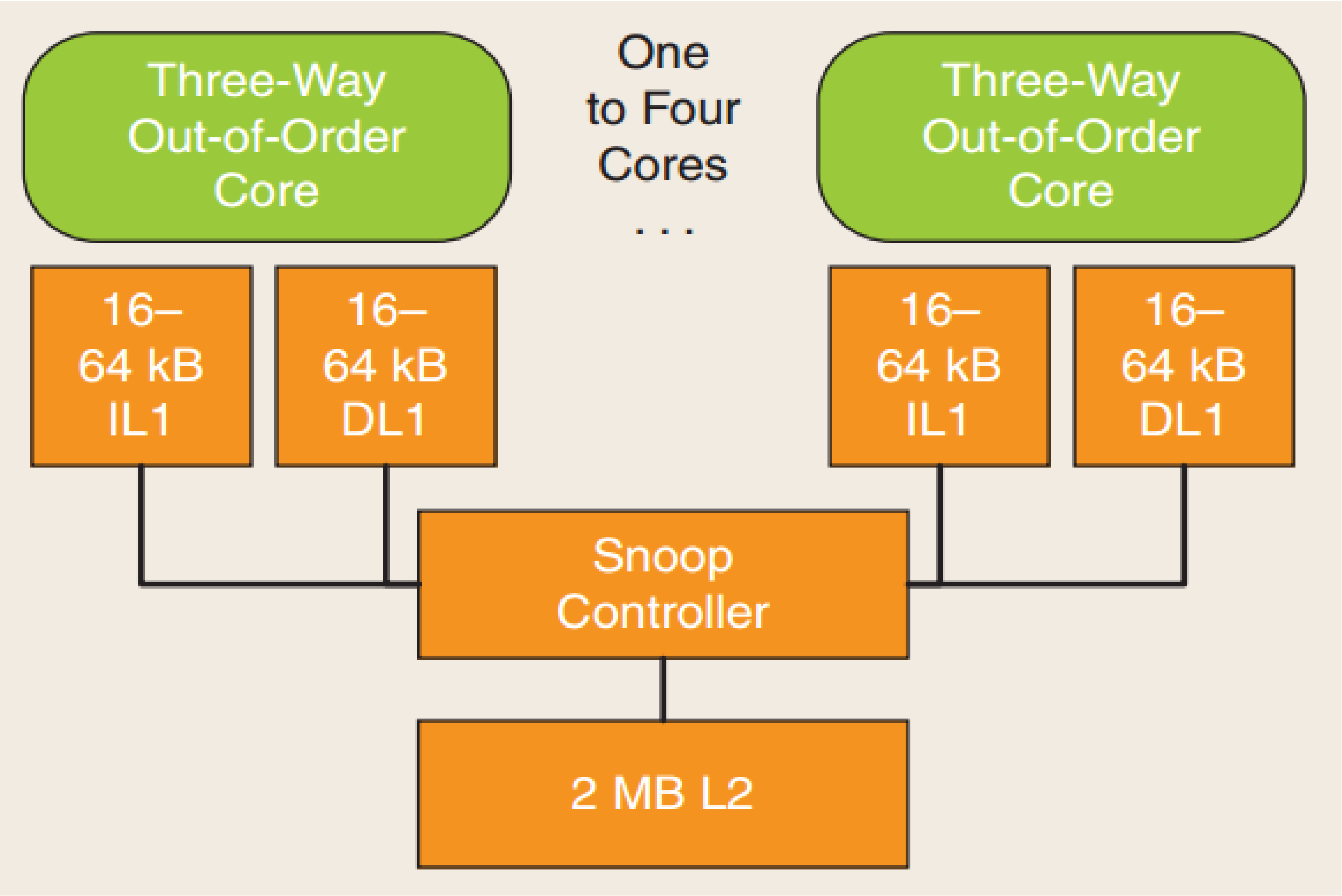


Multicore Example


Four-Way Out-of-Order Core, Two-Way SMT
32 kB IL1
32 kB DL1
256 kB L2
Two to Eight Cores
. . . . .
Four-Way Out-of-Order Core, Two-Way SMT
32 kB IL1
32 kB DL1
256 kB L2
8 MB L3
Quick Path Memory Controller

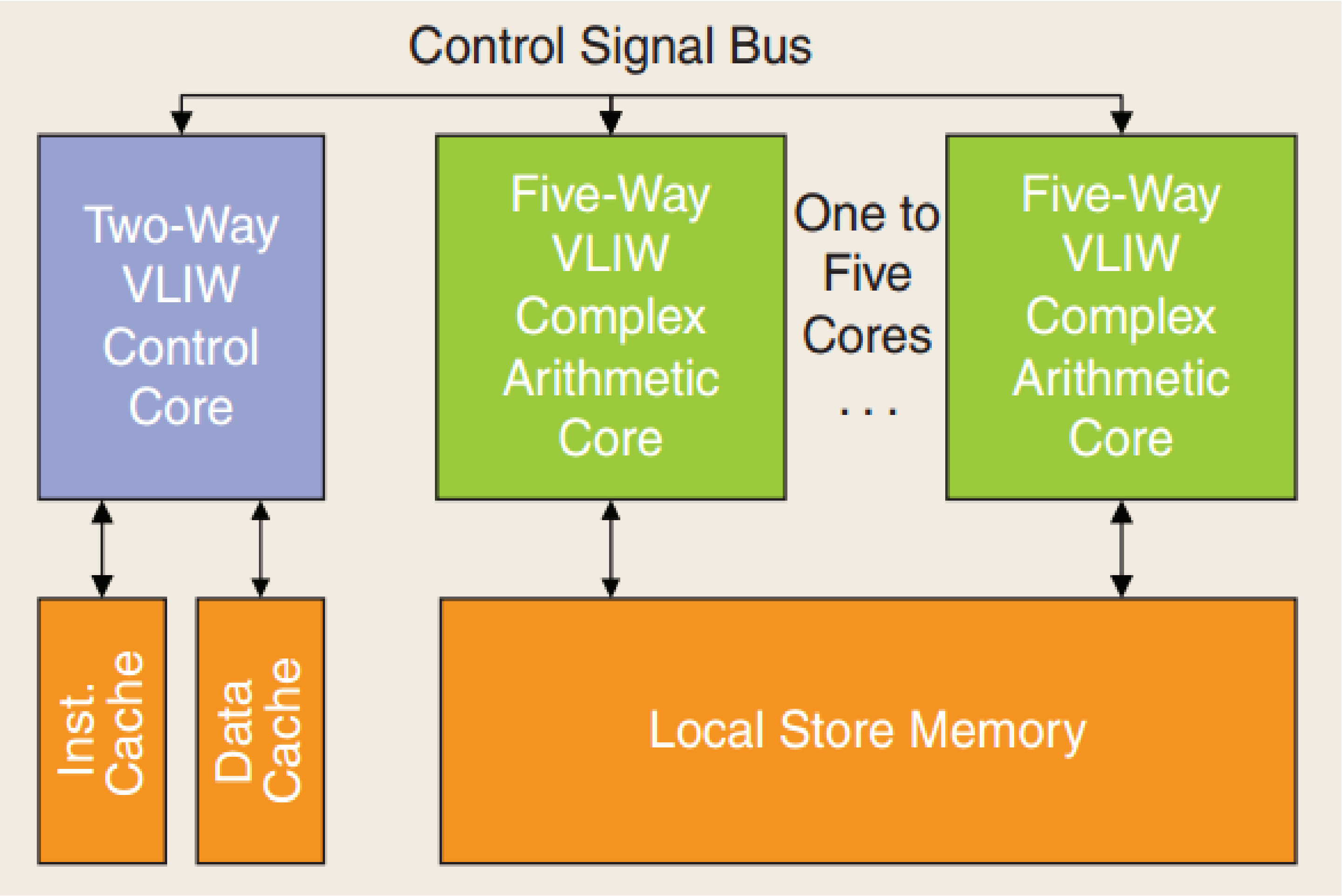
Intel Core i7 block diagram

Multicore Example



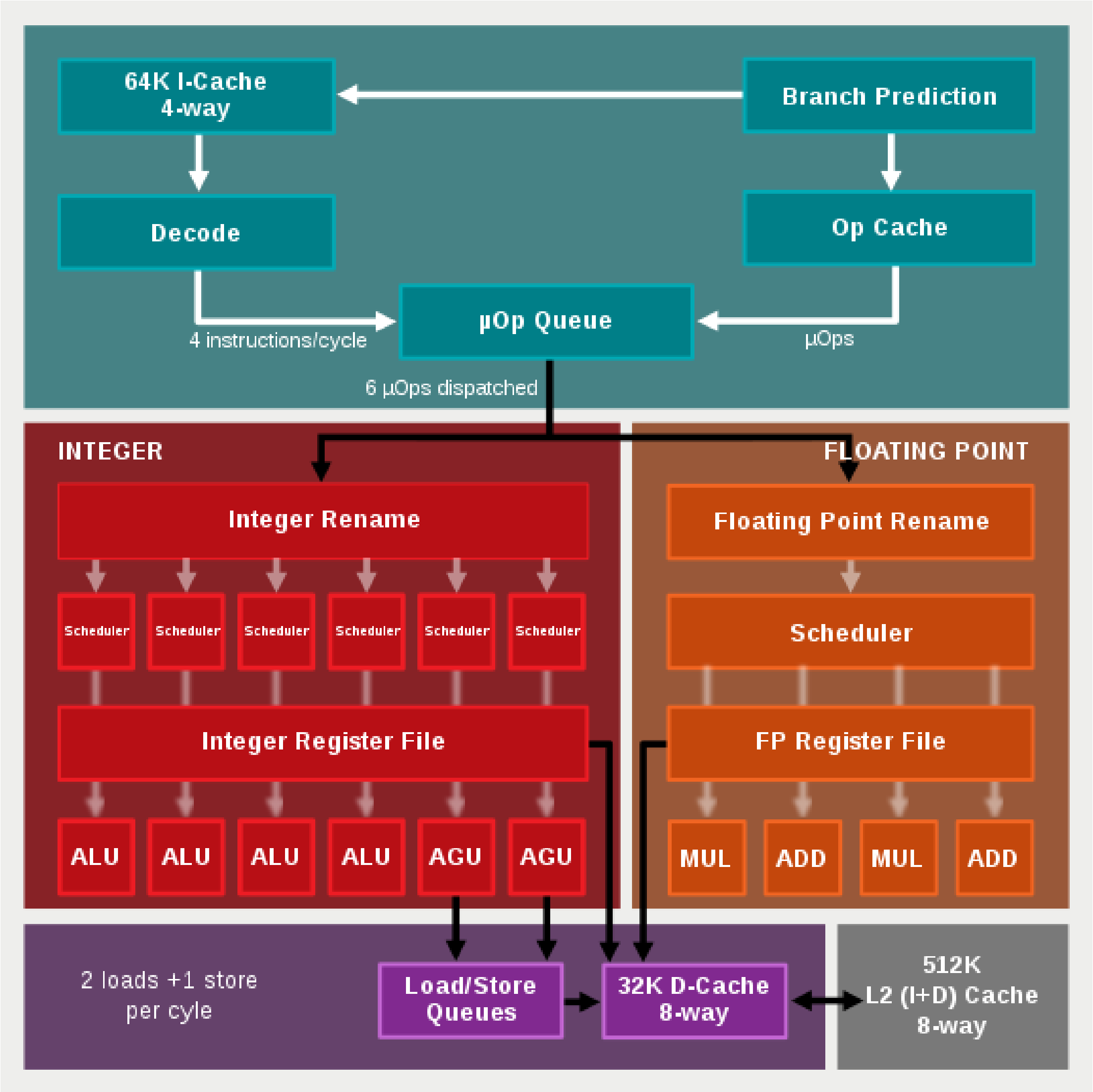
ARM Cortex-A9 block diagram

Multicore Example


Control Signal Bus
Two-Way VLIW Control Core
Five-Way VLIW Complex Arithmetic Core
One to Five Cores . . .
Five-Way VLIW Complex Arithmetic Core
Inst. Cache
Data Cache
Local Store Memory

Silicon Hive HiveFlex CSP2x00 block diagram

Multicore Example



AMD's High-Performance Zen CPU

General Server/Mobile/Embedded Multicores

	ISA	Micro arch	# of Core	Coherence	Interconnect
AMD Phenom	X86	3-way OOO	4	Directory	P2P
Intel Core i7	X86	4-way OOO	2~8	Broadcast	P2P
Sun Niagara	SPARC	2-Way In-order	8	Directory	Crossbar
Intel Atom	X86	2-way In-order	1~2	Broadcast	Bus
ARM Cortex A9	ARM	3-way OOO	1~4	Broadcast	Bus
XMOS XS1-G4	XCORE	1-way in-order	4	None	Crossbar

The microarchitecture of the above cores is traditional and based on a powerful conventional uniprocessor.

High-Performance Multicore/Manycore

	ISA	Micro arch	# of Core	Coherence	Interconnect
AMD Radeon	N/A	5-way VLIW	160	None	N/A
NVIDIA G200	N/A	1-way In-order	240	None	N/A
Intel Larrabee	X86	2-Way In-order	Up to 48	Broadcast	Bidirectional Ring
IBM Cell	POWER	2-way In-order	8 SPUs	None	Bidirectional Ring
Microsoft Xenon	POWER	2-way In-order	3	Broadcast	Crossbar

The above cores are more specialized and are targeted to high-performance computing

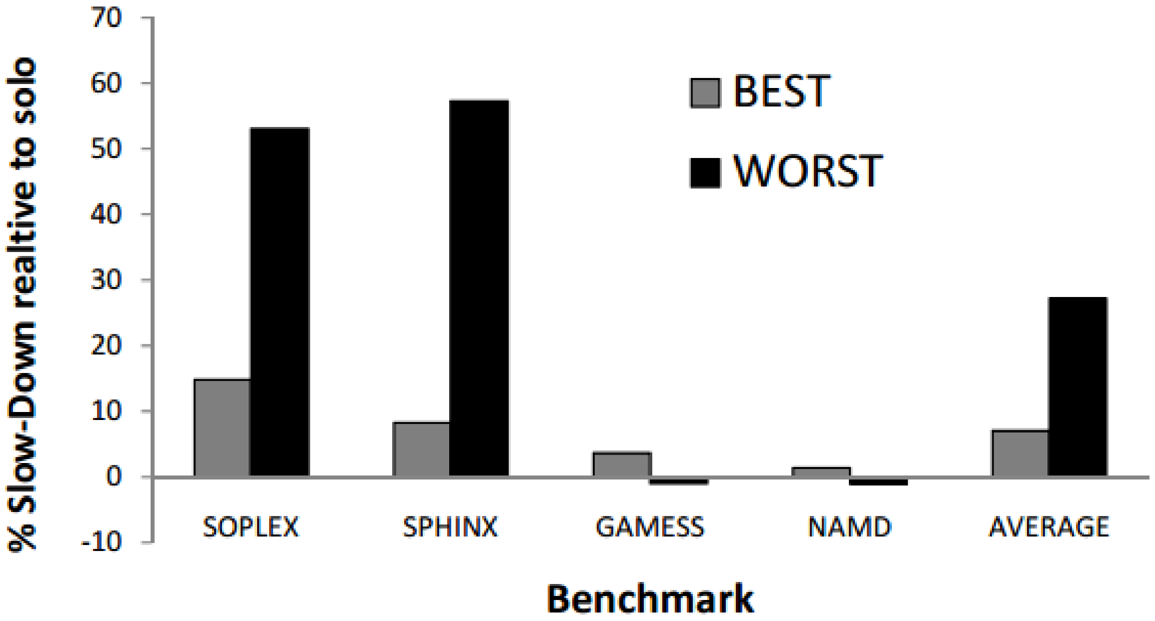
Architectural Challenges of Multicore

- Shared resource management
- ILP and TLP tradeoffs and balance
- Grain size vs. number of cores
- On-/Off- chip bandwidth requirements
- Latencies (execution, cache, memory) reduction
- Multiple domains in terms of power management
- Partitioning resources between threads/cores
- Memory coherence/consistency
- On-die interconnects
- ……

Multicore Resource Contention

A core is not an independent processor but rather a part of a larger on-chip system sharing resources with other cores

threads contend for LLC, MC, bus, prefetching hardware...



The performance degradation relative to running solo for two different schedules of SPEC CPU2006 applications on an Intel Xeon X3565 quad-core processor (two cores share an LLC).

Outlines

- Thread-Level Parallelism
- Introduction to Multicore
- Design Space Exploration
- From Multicore to Manycore

Design Space: Choosing Cores

How to mix different processor core?

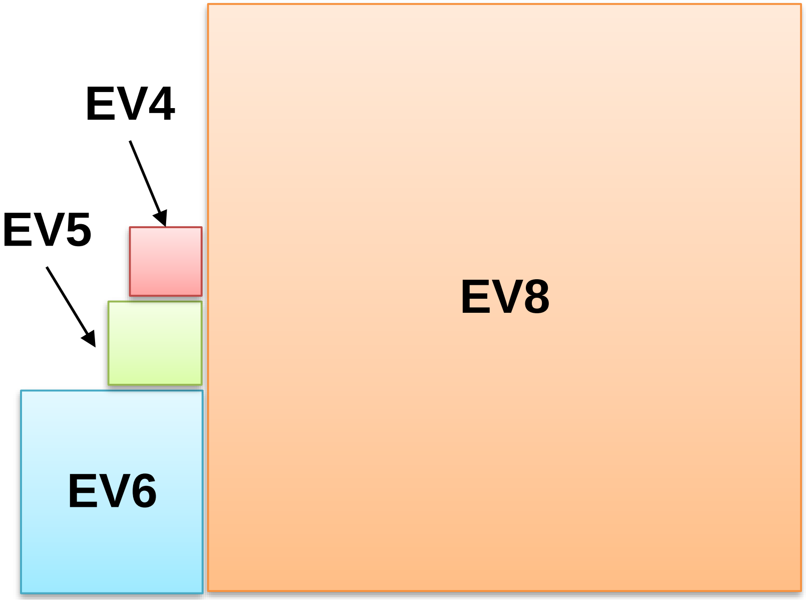
- A small number of complex, heavyweight processor core
  - Emphasizes low thread latency over core area
- A larger number of simple, lightweight processor core
  - Emphasizes core area over thread latency

- General processor cores
  - Small, power-efficient cores
  - Large, high-performance cores
- Specialized cores
  - Accelerators for a particular class of tasks

Design Space: Choosing Cores (Cont'd)

Power and relative performance of Alpha processor cores

Core	Peak Power	Avg. Power	Norm. Perf
EV4	5.0	3.7	1.00
EV5	9.8	6.9	1.30
EV6	17.8	10.7	1.87
EV8	92.88	46.4	2.14



Relative sizes of the Alpha cores

Alpha processors were also identified by EV numbers:

- EV4: early CMOS microprocessor
- EV5: with secondary cache on chip
- EV6: supports out-of-order execution
- EV8: includes simultaneous multithreading

Heterogeneous Chip Multiprocessors

- Also knowns as **asymmetric chip multiprocessors**
  - Match each application to the core best suited to meet its performance demands
  - Provide better area-efficient coverage of the whole spectrum of workload demands

- Single-ISA Heterogeneous CMP
  - CMPs comprising a heterogeneous set of processor cores all of which can execute the same ISA

- The heterogeneity comes from:
  - Raw execution bandwidth (superscalar width), cache sizes, and other characteristics (e.g., in-order vs. out-of-order).

Heterogeneous-ISA Chip Multiprocessor

- It is beneficial to choose a diverse set of ISAs
  - ARM's Thumb, Intel's x86-64, and DEC's Alpha.
- To reap the benefits of the heterogeneity
  - Applications should be able to migrate freely between the cores
- Challenges: runtime state is kept in ISA-specific form
  - Migration involves expensive program state transformation
- Typical migration process involves:
  - Process scheduling: reschedule the process on another core
  - Page table manipulation: change page table mappings
  - Binary translation: translation until a specific point
  - State transformation: transform program states for execution

Classification of Heterogeneous Multicore

	Same Cores	Different Cores
Different ISA	??	Heterogeneous-ISA Chip Multiprocessors
Same ISA	Homogeneous Multi-/Many- Core	Cores of Different Capabilities

Outlines

- Thread-Level Parallelism
- Introduction to Multicore
- Design Space Exploration
- From Multicore to Manycore

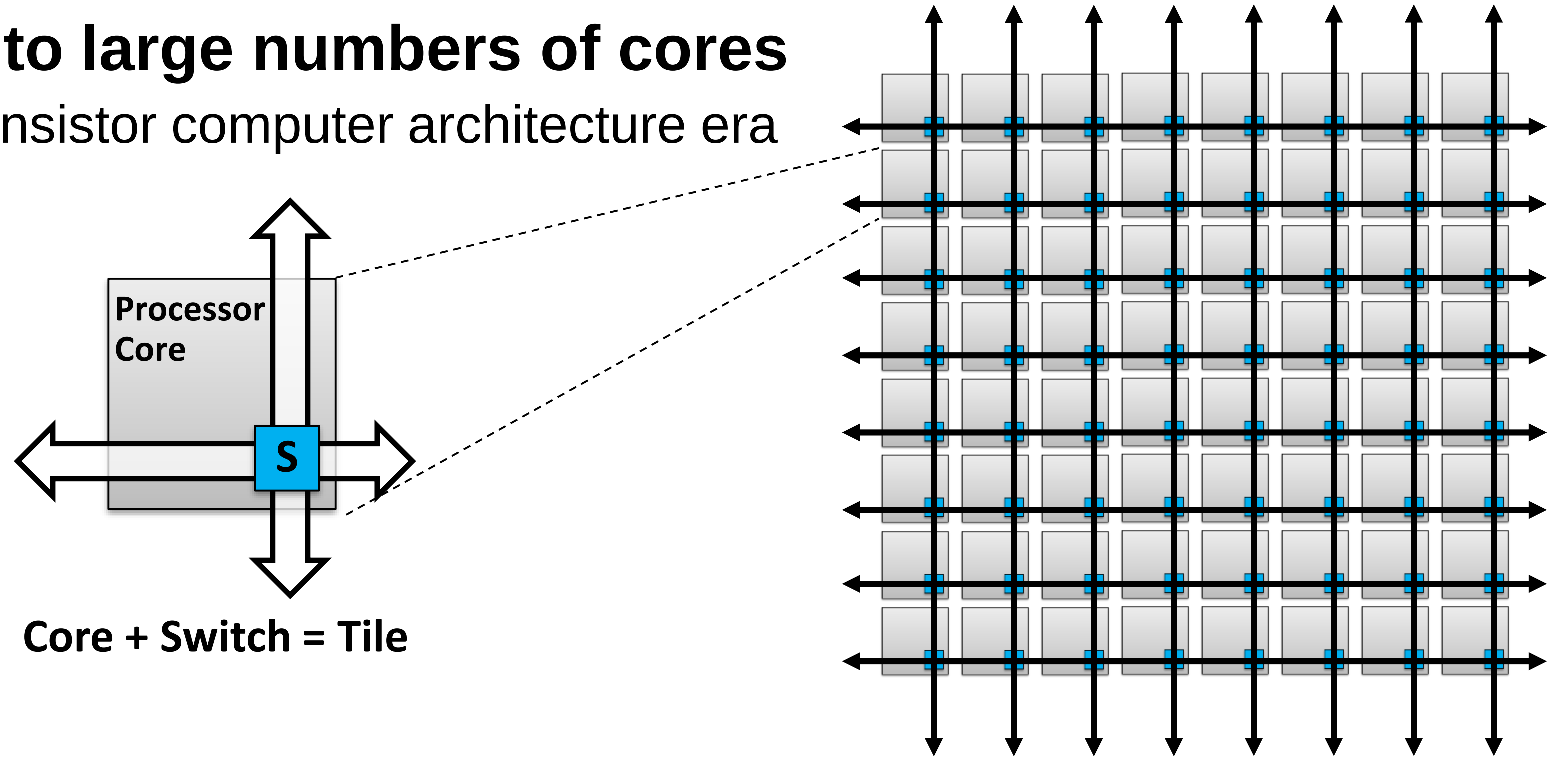
Multicore vs Manycore

- Manycore:
  - Large number of cores
  - Optimized for higher degree of parallelism
  - Higher throughput
  - Lower single thread performance
- Multicore:
  - Small number of cores
  - Optimized for both parallel and serial codes
  - Utilizing OOO, deeper pipelines, etc.
  - More emphasis on high single thread performance

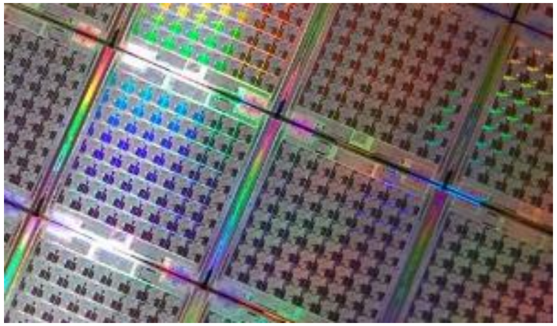
Tilera’s Tile Architecture

Scales to large numbers of cores

billion-transistor computer architecture era



- Repeated Tile Approach:
  - Compute + router
  - Modular, scalable, coherent
  - Short design cycle



Intel Many Integrated Core (MIC) Architecture

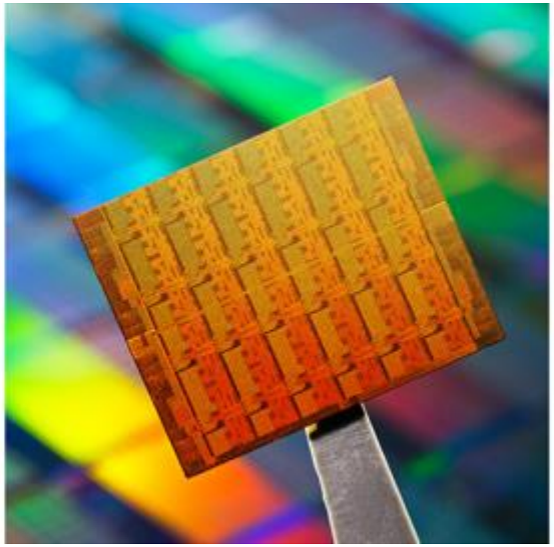
- Intel Xeon Phi Coprocessor:
  - A many-core processor with up to 61 single in-order cores
  - Each of the cores supports four hyper threads
  - The L2 caches between the cores are fully coherent
  - Runs an OS inside, which may take up a processor core
- It is a co-processing element providing optimal power performance efficiency to the overall system

In a PCIe card form factor

Intel's Single-Chip Cloud Computer (SCC)

- Integrate a cloud of computers on chip
  - Basically, it is a many-core architecture
  - In a sense, the SCC is a microcosm of cloud datacenter

- SCC does not have hardware support for cache coherence
  - Uses message passing as its primary programming paradigm



Discussion: Multicore and “Dark Silicon”

- Ref: http://www.darksilicon.org/

Summary

- Thread, Multithreading, SMT
- CMP and multicore
- Benefits of multicore
- Multicore system architecture
- Heterogeneous multicore system
- Heterogeneous-ISA CMP
- Multicore and manycore
- Design challenges

References

- 课本内容：J. Hennessy, D. Patterson. Computer Architecture, Fifth Edition: A Quantitative Approach.
  - Chapters: 3.12, 5.2

- 其它参考：K. Olukotun et al., 《Chip Multiprocessor Architecture: Techniques to Improve Throughput and Latency》, Synthesis Lectures on Computer Architecture.
  - Chapters: 1, 2.1

- 其它阅读：
  - R. Kumar et al. “Heterogeneous Chip Multiprocessors”. *Computer*, 2005. IEEE
  - G. Blake et al. “A Survey of Multicore Processors”, *Signal Processing Magazine*, 2009. IEEE

Exercises

- Can we keep adding cores to a CPU? Why?

- What are the opportunities of multicore design optimization?