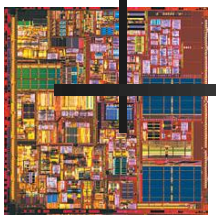
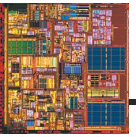
# Hyperthreading



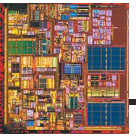
# Why multi-core/hyperthreading ?

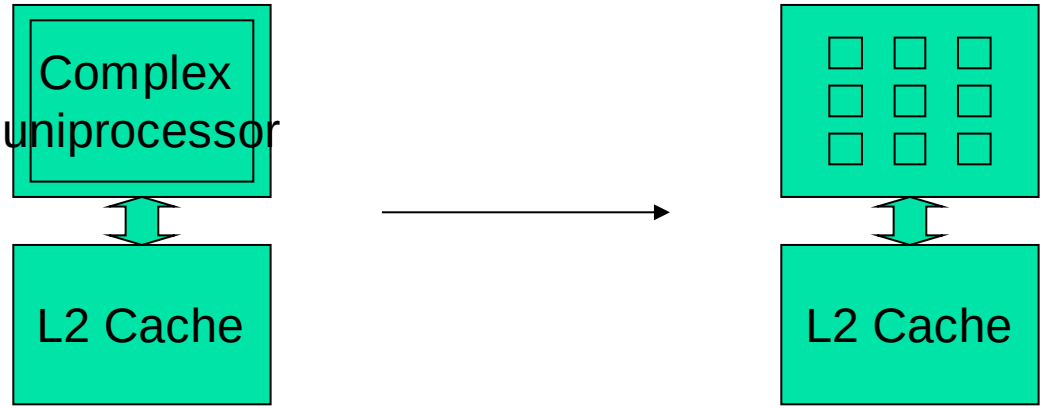


- Difficult to make single-core clock frequencies even higher
- Deeply pipelined circuits:
  - heat problems
  - speed of light problems
  - difficult design and verification
  - large design teams necessary
  - server farms need expensive air-conditioning

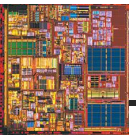

- Many new applications are multithreaded
- General trend in computer architecture (shift towards more parallelism)

# Multi-core Computing



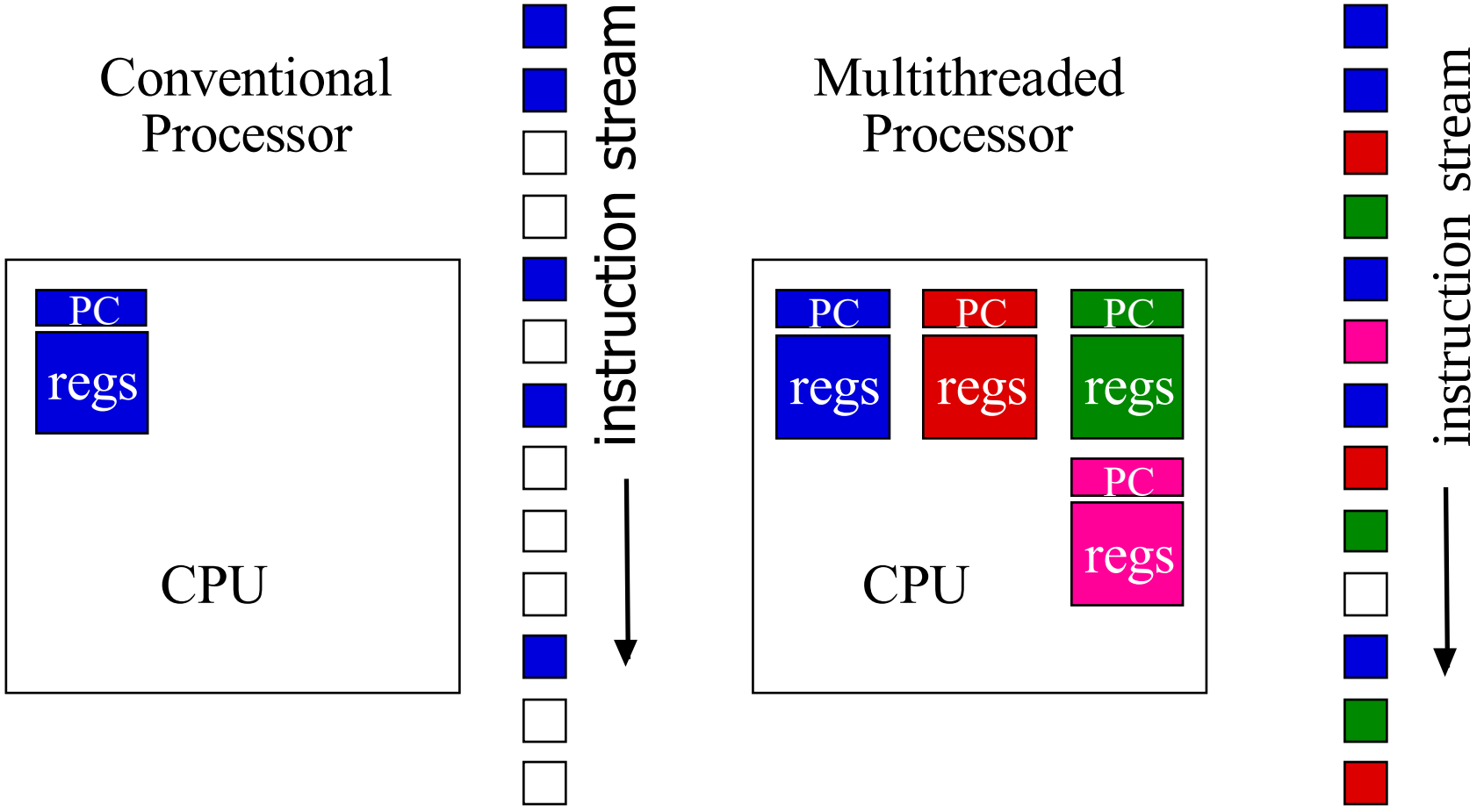


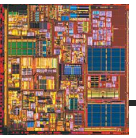
# Motivation for Hardware multithreading ("hyperthreading")



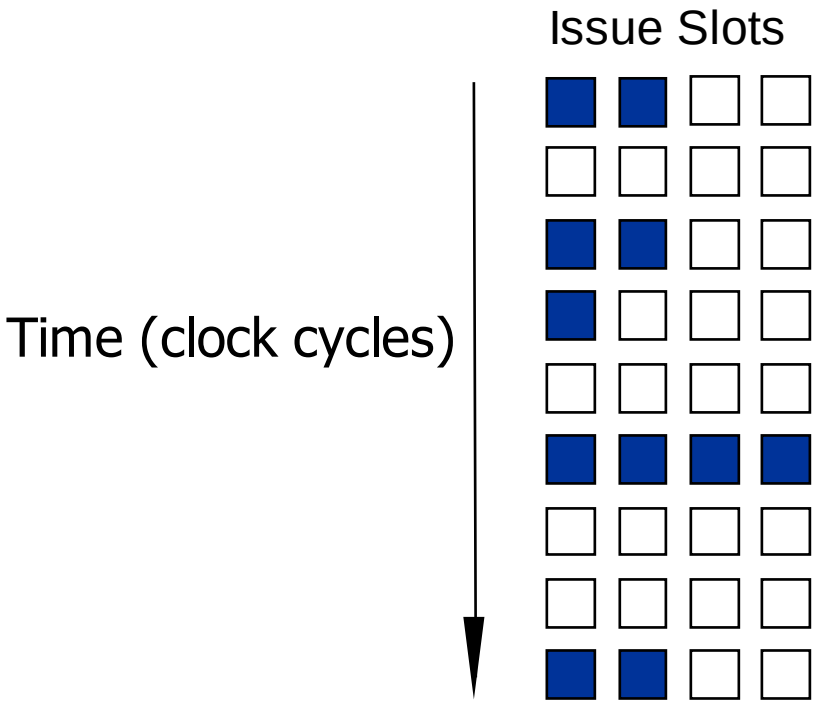
- Modern processors fail to utilize execution resources well.
- There is no single culprit.
- Attacking the problems one at a time (e.g., *specific* latency-tolerance solutions) always has limited effectiveness.
- However, a *general latency-tolerance solution* which can hide all sources of latency can have a large impact on performance.

# Hardware Multithreading

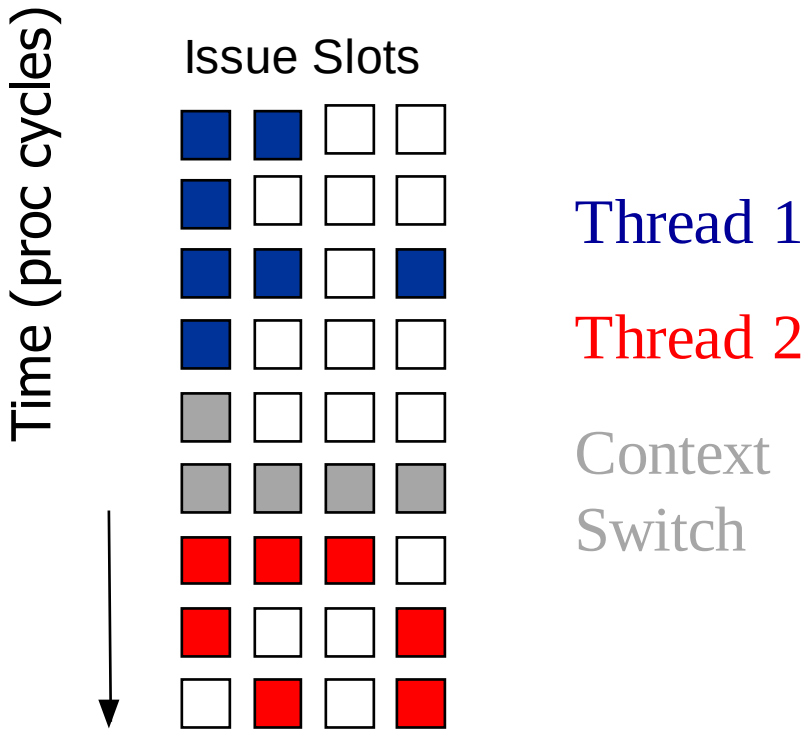




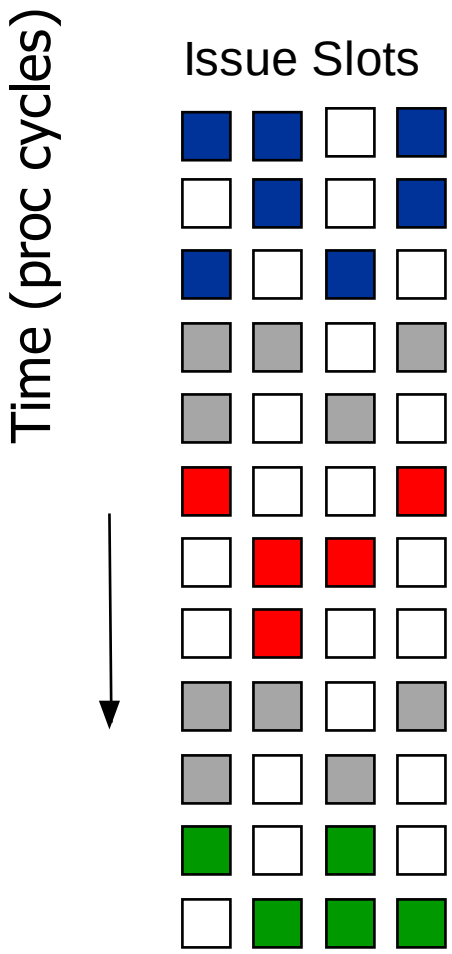
# Superscalar Execution

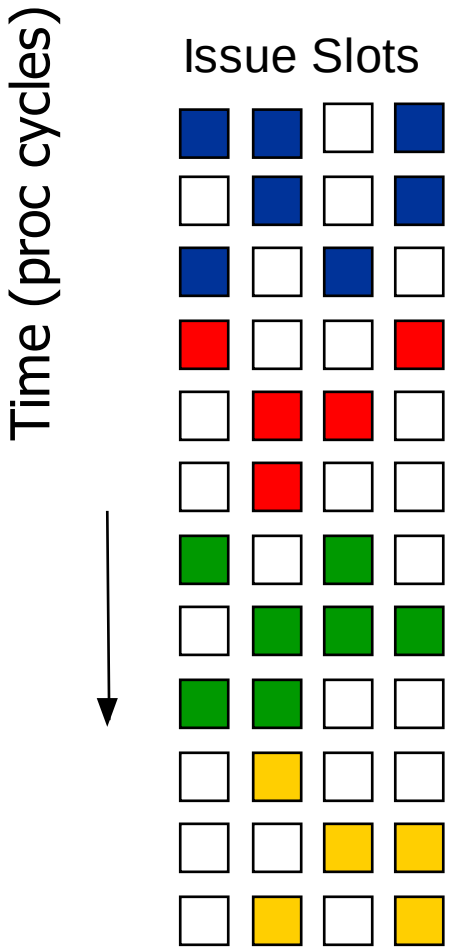


# Multithreading on Superscalar



# Superscalar Execution with Coarse-Grained Multithreading




Thread 1

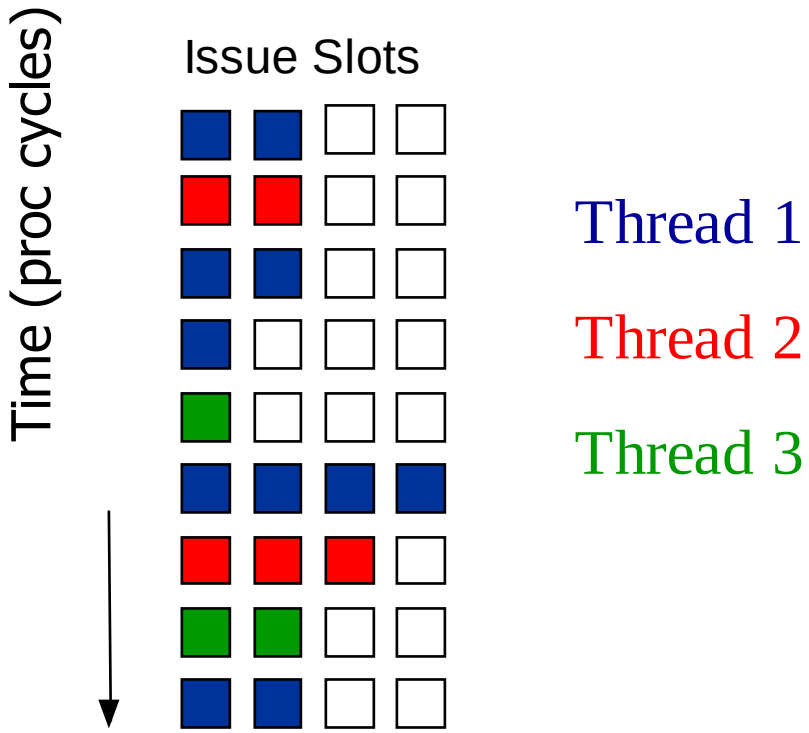
Thread 2

Thread 3

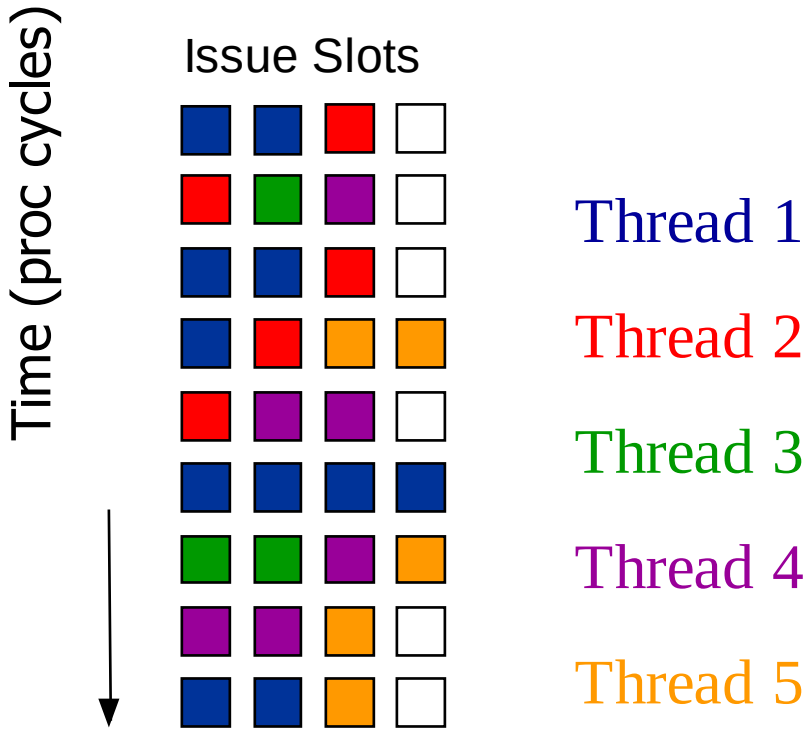
Thread 4

Context Switch

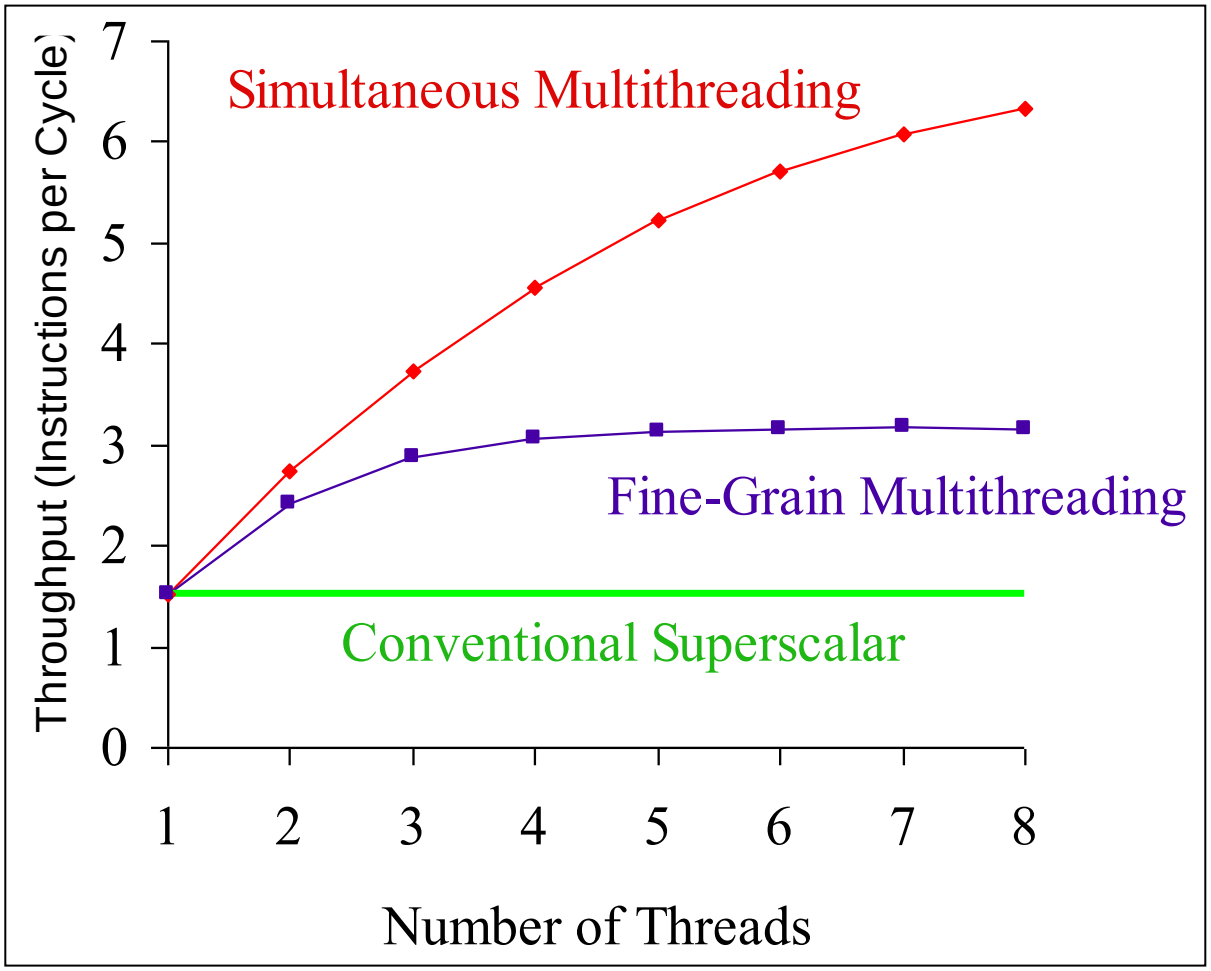
# Superscalar Execution with Fine-Grain Multithreading

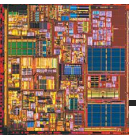


# Simultaneous Multithreading



# The Potential for SMT





# Goals

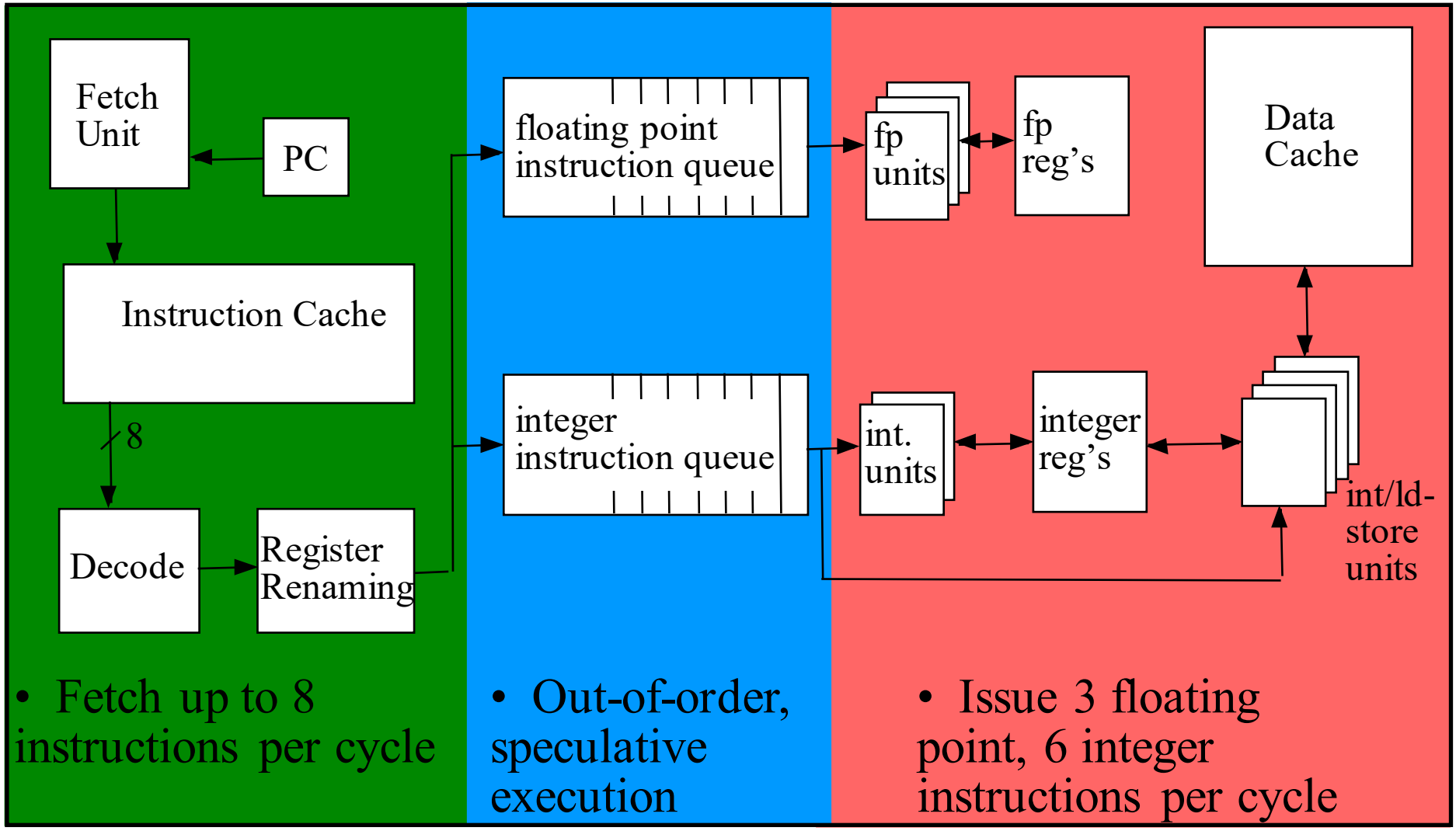
Three primary goals for this SMT:

1. Minimize the architectural impact on conventional superscalar design.

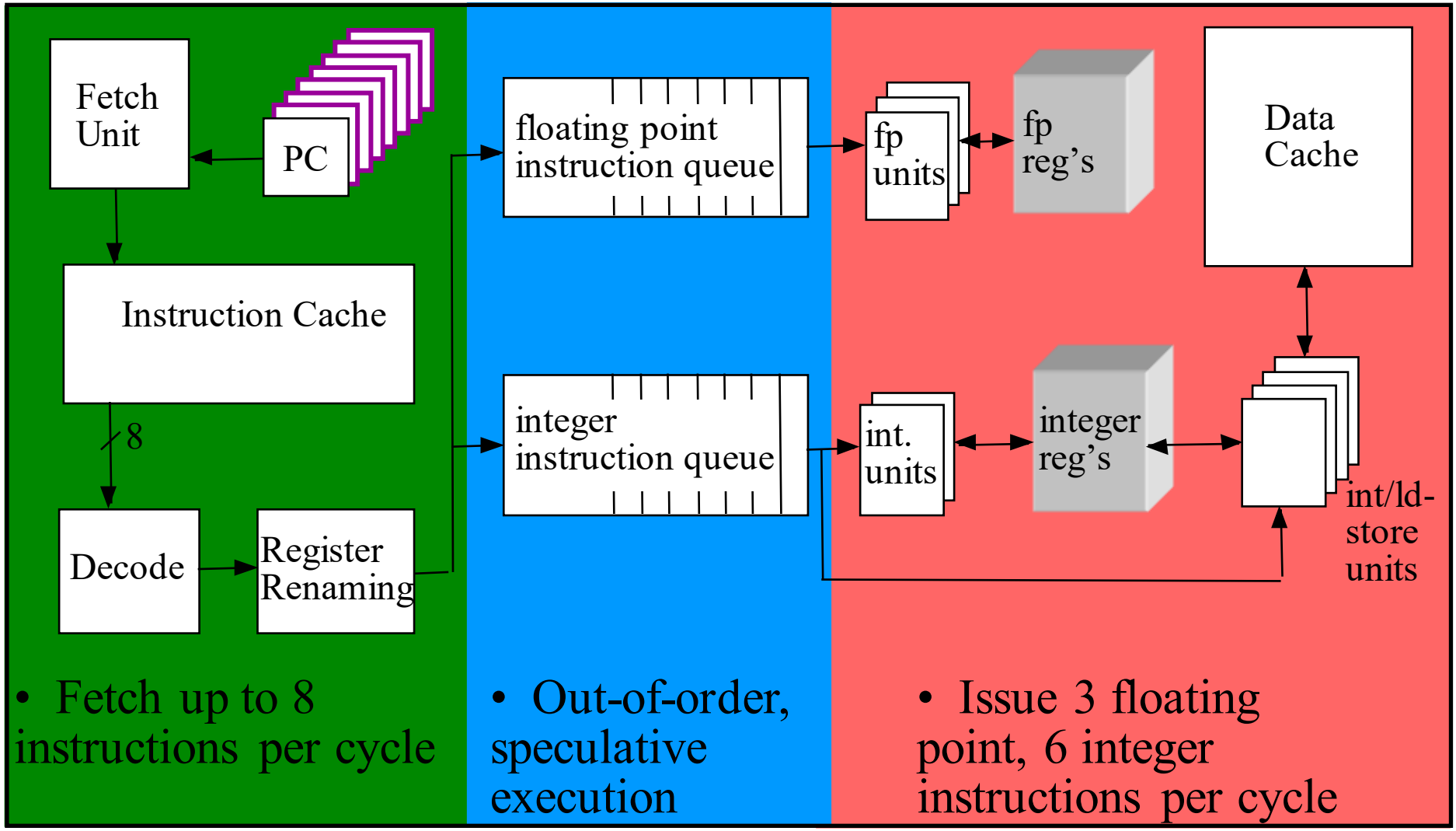
2. Minimize the performance impact on a single thread.

3. Achieve significant throughput gains with many threads.

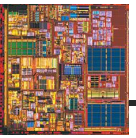
# A Conventional Superscalar Architecture



# An SMT Architecture

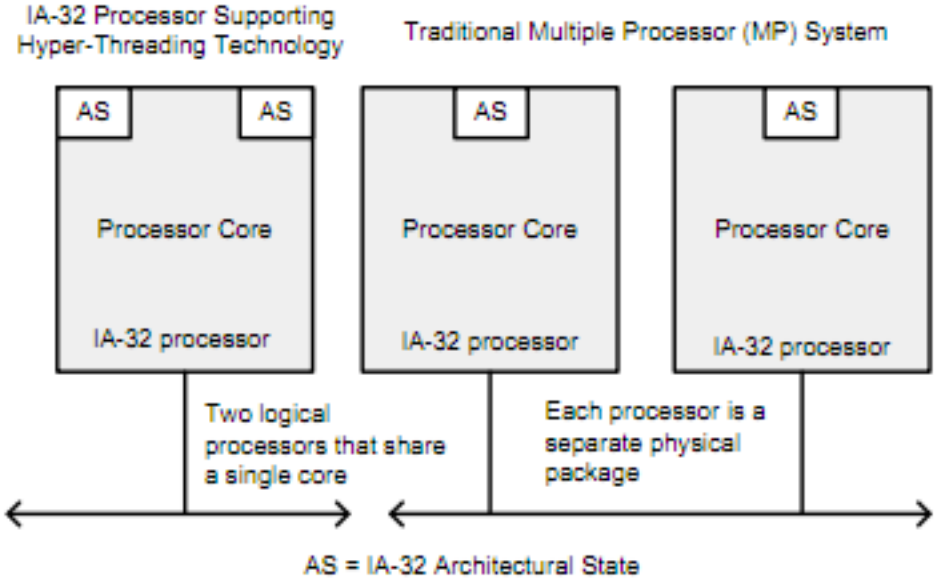


# Real-World SMT



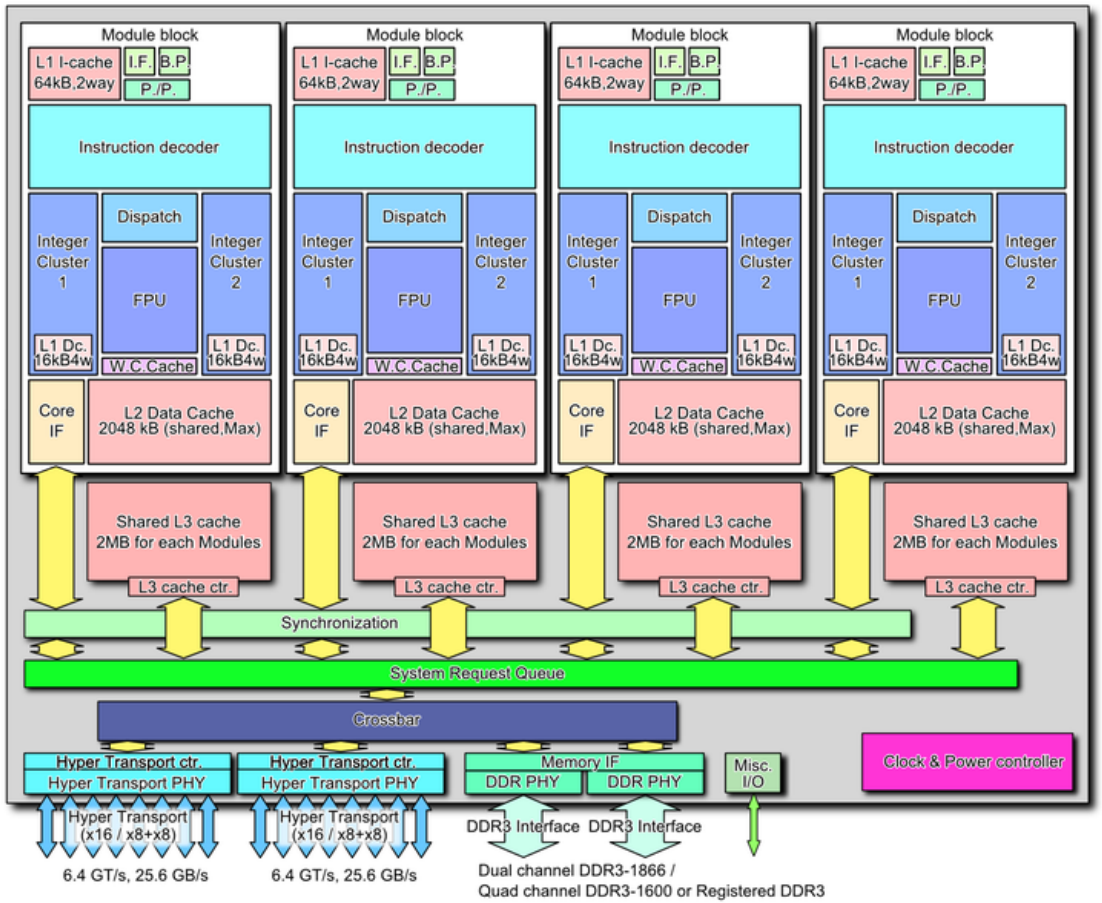
Intel – Hyperthreading

- IBM whitepaper: 20-50% performance benefit

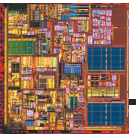


# Real-World SMT (2)

- AMD - "It's all about the cores!"
  - "Our cores are real." – January, 2010
    - "Hyperthreading is stupid. So is Intel." - paraphrase
  - October 12, 2011: Bulldozer: 4 "modules", 8 threads



# More SMT



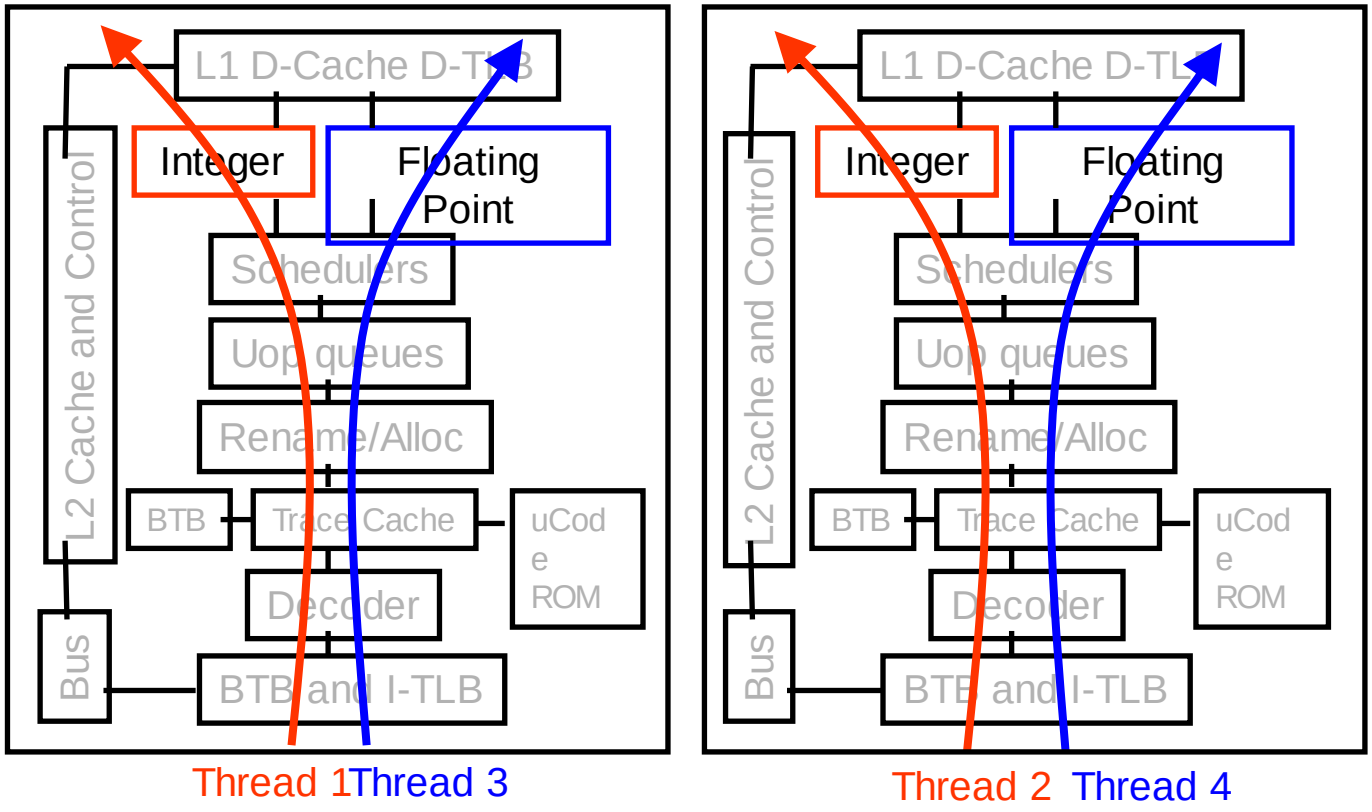
- Network Processors
- CMT processors (Oracle)
- Many Intel processors, etc.

# Combining Multi-core and SMT

- Cores can be SMT-enabled (or not)
- different combinations:
  - single-core, non-SMT: standard uniprocessor
  - single-core, w/ SMT
  - multi-core, non-SMT
  - multi-core, w/ SMT
- Number of SMT threads
  - 2, 4, or sometimes 8 simultaneous threads
- Intel calls them “hyper-threads”

# SMT Dual-core

- All four threads can run concurrently



# Comparison: Multi-core vs. SMT

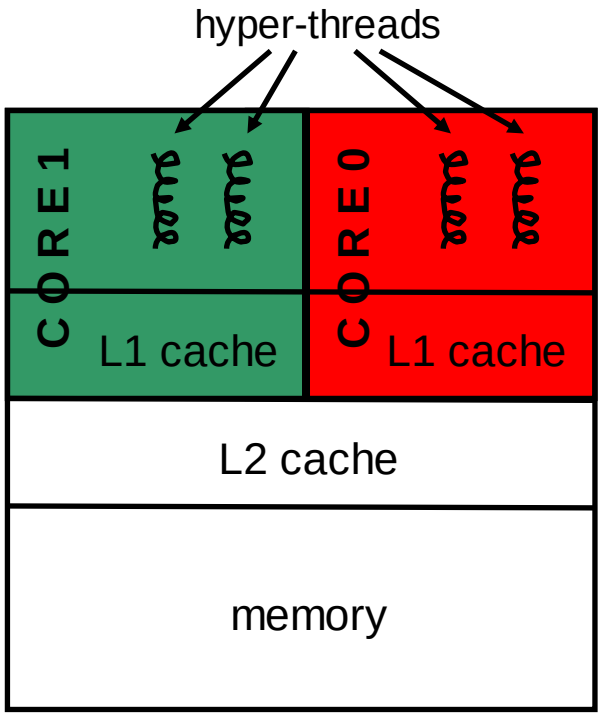
- Multi-core:
  - Since there are several cores, each is smaller and not as powerful (but also easier to design and manufacture)
  - However, great with thread-level parallelism
- SMT
  - Can have one large and fast superscalar core
  - Great performance on a single thread
  - Mostly still only exploits instruction-level parallelism

# Memory Hierarchy

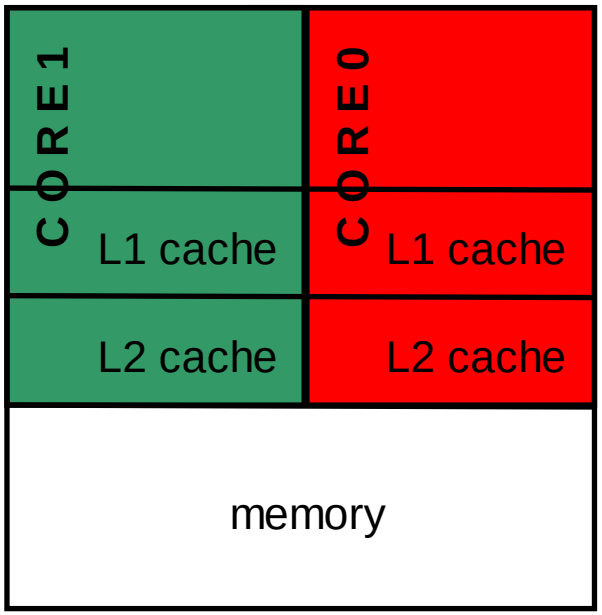
- If simultaneous multithreading only:
  - all caches shared
- Multi-core chips:
  - L1 caches private
  - L2 caches private in some architectures and shared in others
- Memory is always shared

# Dual-core Intel Xeon Processors

- Each core
  - hyper-threaded
- Private L1 caches
- Shared L2 caches

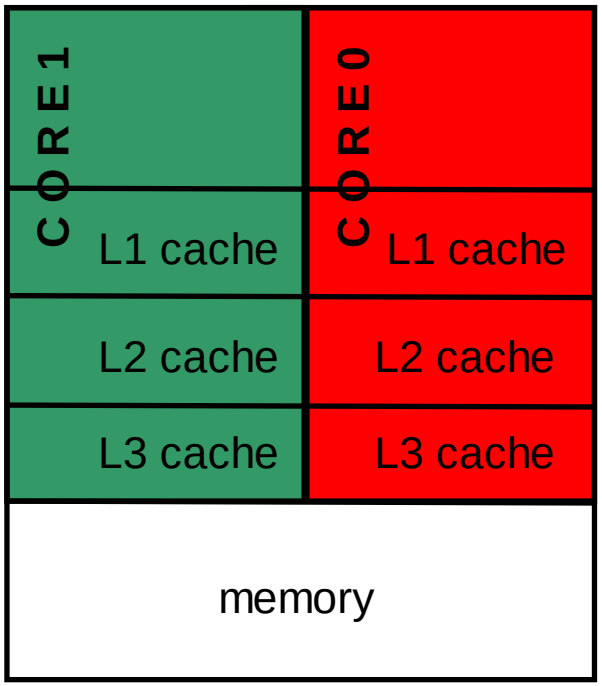


# Designs w/ private L2 caches




both l1 and l2 are private

examples: amd opteron,
amd athlon, intel pentium d




a design with l3 caches

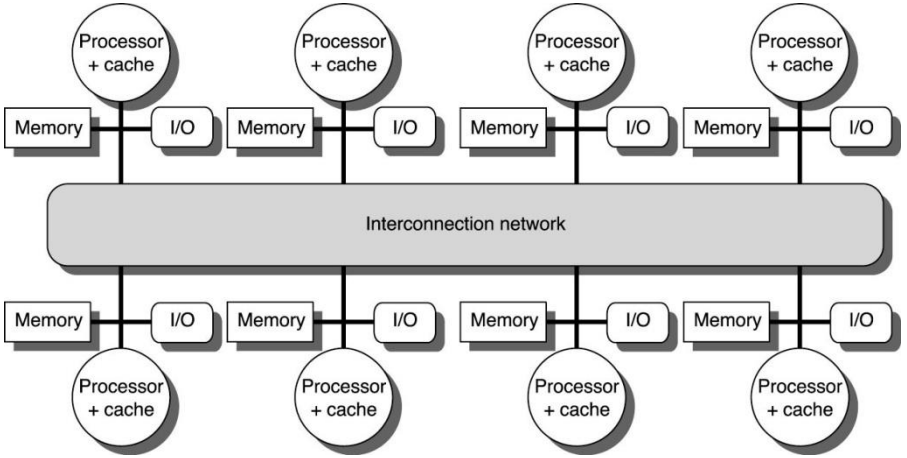
example: intel itanium 2

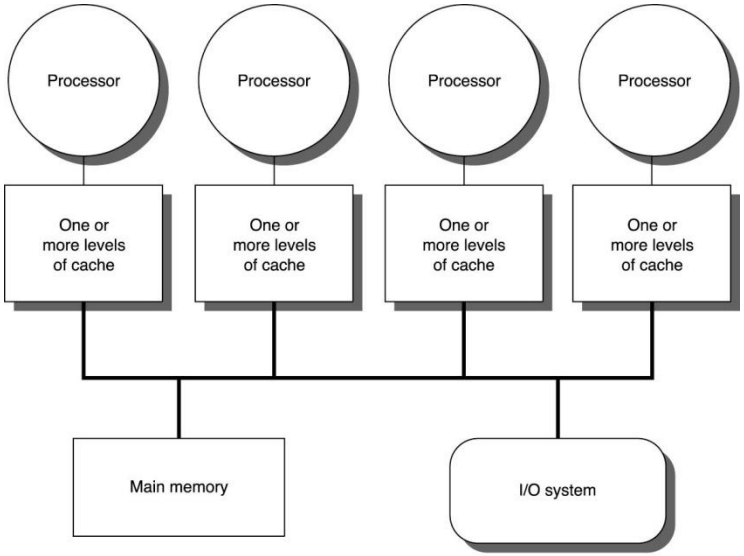
# Private vs shared caches

- Advantages of private:
  - They are closer to core, so faster access
  - Reduces contention
- Advantages of shared:
  - Threads on different cores can share the same cache data
  - More cache space available if a single (or a few) high-performance thread runs on the system

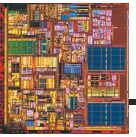
# Interconnection Network

- Bus
- Network
- pros/cons?



# Programming Model



- Shared Memory -- every processor can name every address location
- Message Passing -- each processor can name only it's local memory. Communication is through explicit messages (multicomputer).
- pros/cons?

- *find the max of 100,000 integers on 10 processors.*