Parallel Programming Principle and Practice

Lecture 3 — Multicore processors/ multiprocessor with shared memory (SMP)

Outline

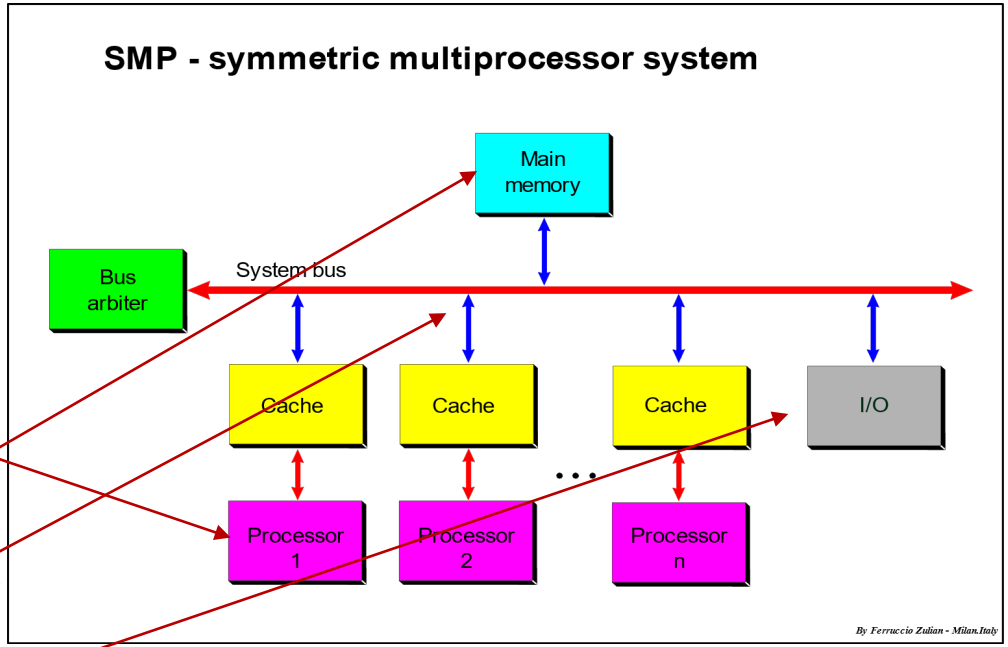
- Generic SMP system architecture
- Memory hierarchy
  - L1 local cache
  - L2 global cache
  - Common shared main memory
  - Cache coherency
- Interconnection network for SMP system
- Real modern multicore / SMP system architecture

Multicore processors/ multiprocessor with shared memory (SMP)

Generic SMP system architecture

Symmetric Multi-core Processor(SMP)

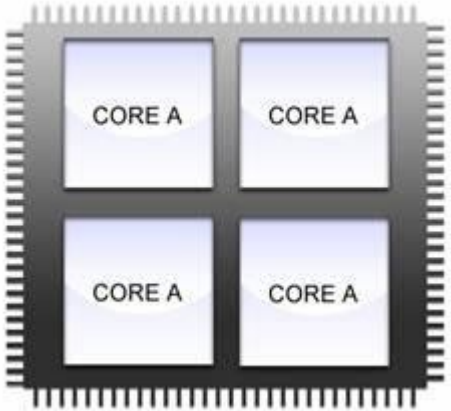
- A symmetric multi-core processor (Symmetric multiprocessing，对称多处理器) is a processor
  - has multiple cores that are all exactly the same: has the same architecture and the same capabilities
  - multiple identical processors are interconnected to a single shared main memory via the same data path
  - Full accessibility to all the I/O devices



multicore/multiprocessor connected to a common memory via interconnection network

Symmetric Multi-core Processor(SMP)

- All cores which exist in a die are exactly identical
  - ✓ **Example: Intel Core 2**
  - ✓ The Core 2 can have either 2 cores on chip ("Core 2 Duo") or 4 cores on chip ("Core 2 Quad").
  - ✓ Each core in the Core 2 chip is symmetrical, and can function independently of one another.
  - ✓ It requires a mixture of scheduling software and hardware to farm tasks out to each core.

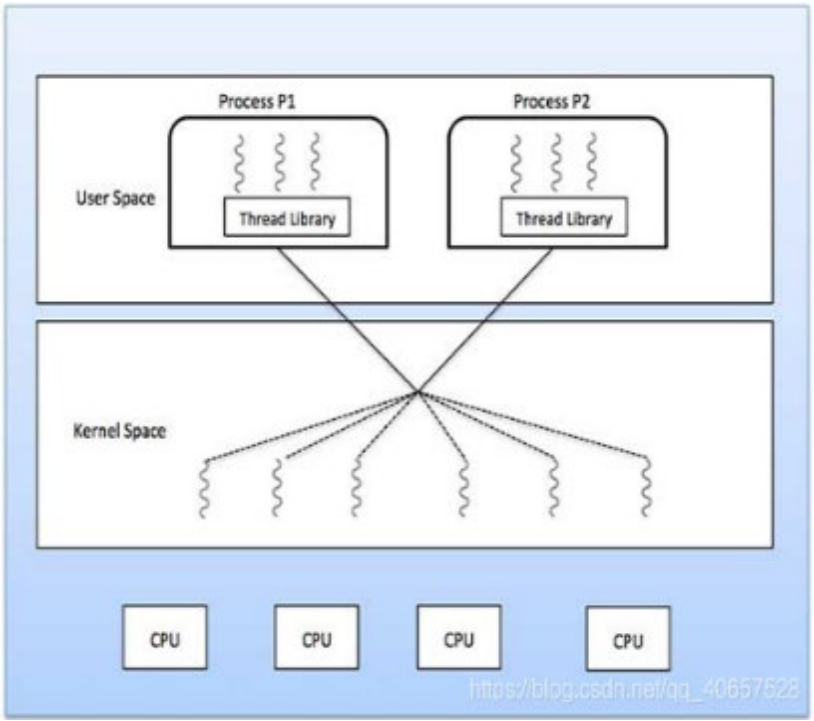


All cores which exist in a die are exactly identical

Symmetric Multi-core Processor(SMP)

- Each core has the same capabilities, so it requires that there is an arbitration unit (仲裁单元) to give each core a specific task
- Software that uses techniques like multithreading (多线程) makes the best use of a multi-core processor like the Intel Core2
- Thread
  - The smallest sequence of programmed instructions
  - A component of a process
  - The multiple threads of a given process may be executed concurrently (via multithreading capabilities), sharing resources such as memory
  - Different processes do not share these resources
  - The threads of a process share its executable code and the values of its dynamically allocated variables and non-thread-local global variables

Symmetric Multi-core Processor(SMP)



- 多线程处理
  - 多线程能够在同一时间执行多于一个线程，进而提升整体处理性能
- 多对多关系线程模型
  - 任意数量N的用户线程到相等或者小于N的内核线程的多路复用
- 在Linux内核启动的最后阶段，系统会创建两个内核线程，一个是init，一个是kthreadd
  - **kthreadd线程是内核的守护线程**，在内核正常工作时，它永远不退出，是一个死循环，它的pid是2

Symmetric Multi-core Processor(SMP)

- Applications
  - Personal Computers

The cpu used by the personal computer is a symmetric multi-core processor

  - Server / Super Computer

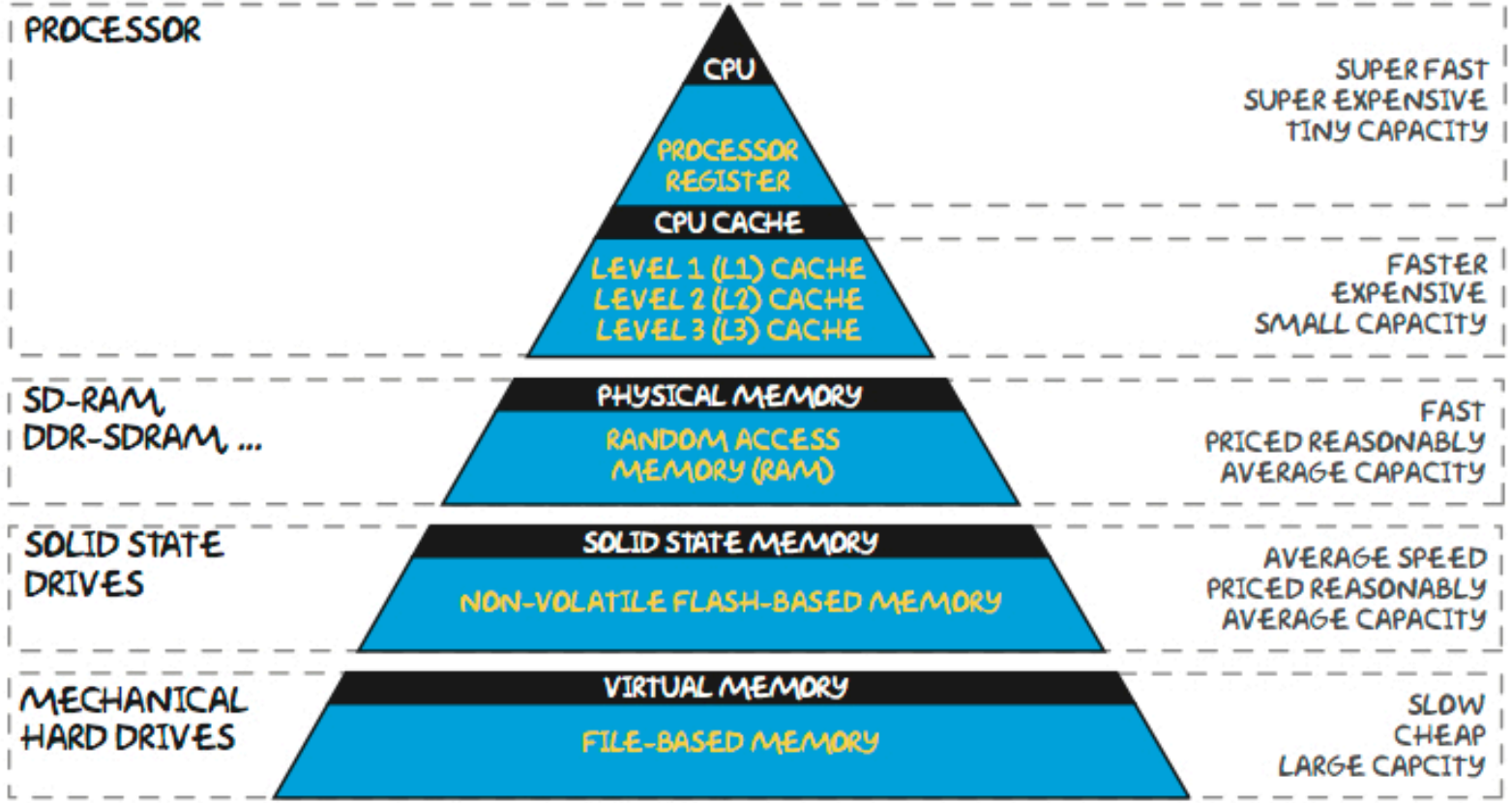
AMD Opteron 64

Servers/Super computers all use symmetric multi-core processors

Multicore processors/ multiprocessor with shared memory (SMP)

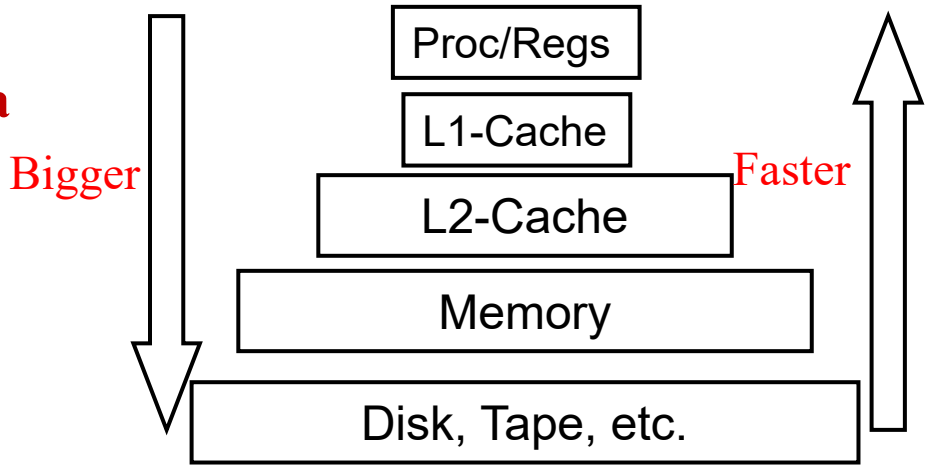
Memory hierarchy

Memory Hierarchy



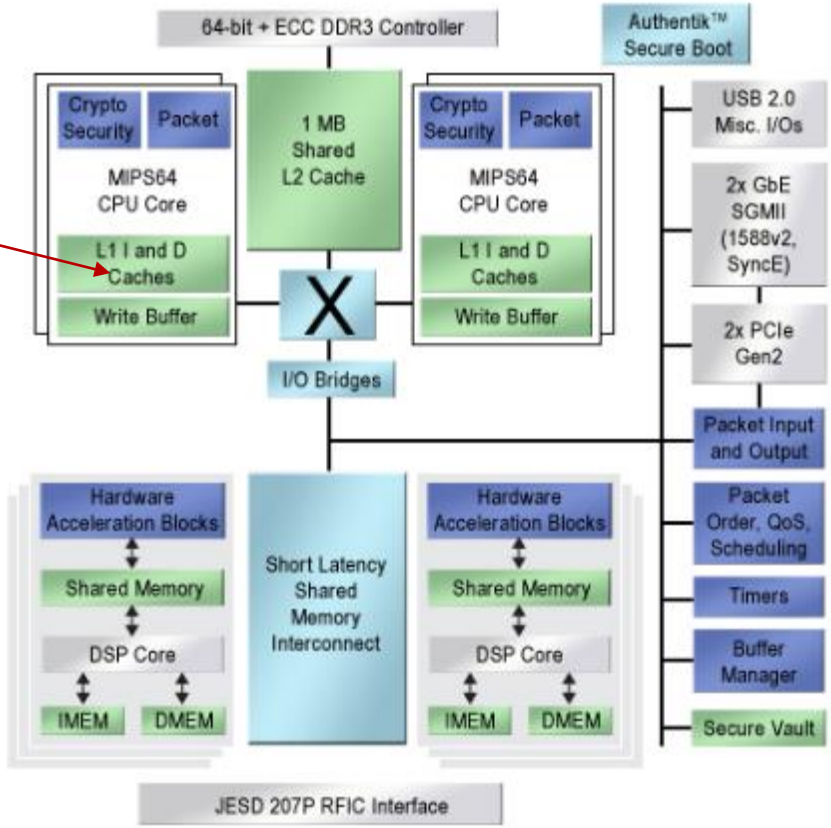
What is a cache?

- Small, fast storage used to improve average access time to slow memory.
- Exploits spacial and temporal locality
- In computer architecture, almost **everything is a cache**!
  - Registers a cache on variables
  - First-level cache a cache on second-level cache
  - Second-level cache a cache on memory
  - Memory a cache on disk (virtual memory)
  - TLB (快表) a cache on page table
  - Branch-prediction a cache on prediction information



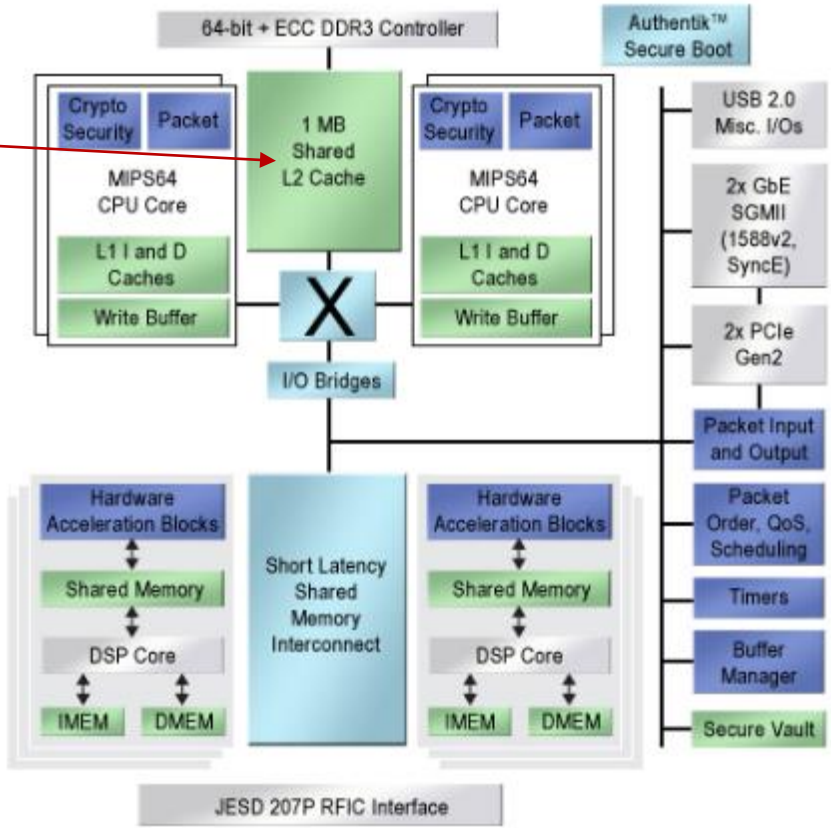
L1 Local Cache

- A level 1 cache (L1 cache, memory cache, primary cache, internal cache, or system cache) that is directly built into the microprocessor
  - the first cache to be accessed and processed when the processor itself performs a computer instruction
  - the fastest cache memory (built within the chip with a zero wait-state interface)
  - most expensive cache among the CPU caches, limited size
  - store data that was accessed by the processor recently, critical files that need to be executed immediately



L2 Global Cache

- A level 2 cache (L2 cache, the secondary cache, an external cache)
  - located outside and separate from the microprocessor chip core, it is found on the same processor chip package
  - the capacity can be increased
  - it is still faster than the main memory
  - served as the bridge for the process and memory performance gap



Cache performance

- Miss-oriented Approach to Memory Access (考虑缺失的开销计算)

$$CPUtime = IC \times \left(CPI_{Execution} + \frac{MemAccess}{Inst} \times MissRate \times MissPenalty \right) \times CycleTime$$

$CPI_{Execution}$ includes ALU and Memory instructions (cache)

Note: memory hit time is included in execution cycles

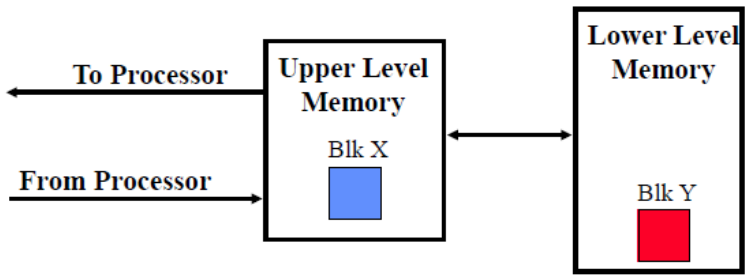
- Separating out Memory component entirely (剥离存取的开销计算)
  - AMAT = Average Memory Access Time
  - CPI_{ALUOps} does not include memory instructions

$$CPUtime = IC \times \left(\frac{AluOps}{Inst} \times CPI_{AluOps} + \frac{MemAccess}{Inst} \times AMAT \right) \times CycleTime$$

$$AMAT = HitTime + MissRate \times MissPenalty$$

Cache performance

- Hit: data appears in some blocks in the upper level (example: Block X)
  - Hit Rate: The fraction of memory access found in the upper level
  - Hit Time: Time to access the upper level which consists of RAM access time + Time to determine hit/miss
- Miss: data needs to be retrieve from a block in the lower level (Block Y)
  - Miss Rate = 1 - (Hit Rate)
  - Miss Penalty: Time to replace a block in the upper level + Time to deliver the block to the processor
- Hit Time << Miss Penalty (500 instructions on 21264!)



L2 Global Cache

- L2 Equations

  $AMAT = Hit\ Time_{L1} + Miss\ Rate_{L1} \times Miss\ Penalty_{L1}$

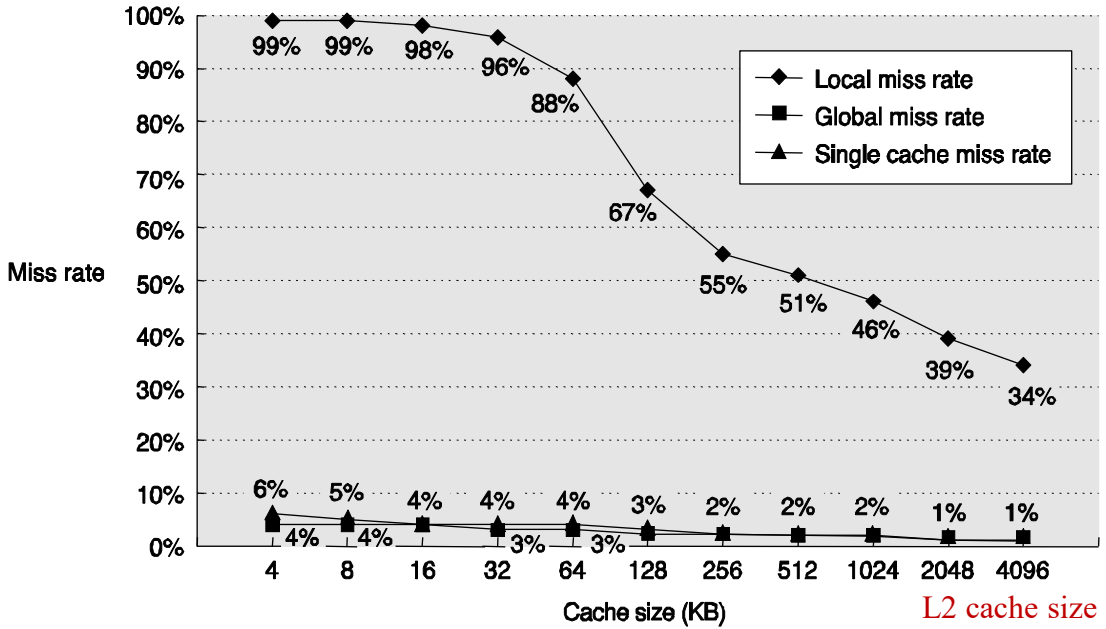
  $Miss\ Penalty_{L1} = Hit\ Time_{L2} + Miss\ Rate_{L2} \times Miss\ Penalty_{L2}$

  $AMAT = Hit\ Time_{L1} + Miss\ Rate_{L1} \times (Hit\ Time_{L2} + Miss\ Rate_{L2} + Miss\ Penalty_{L2})$

- Definitions:
  - *Local miss rate*— misses in this cache divided by the total number of memory accesses *to this cache* ($Miss\ rate_{L2}$)
  - *Global miss rate*—misses in this cache divided by the total number of memory accesses *generated by the CPU* ($Miss\ Rate_{L1} \times Miss\ Rate_{L2}$)
  - Global Miss Rate is what matters

Comparing Local and Global Miss Rates

- L1=32Kb，L2变化
  - 随着L2 cache的增大，local miss rate显著减小
  - 由于L1 cache较大，随着L2 cache容量变大，global miss rate小幅下降
  - Cost 和 A.M.A.T.相关联

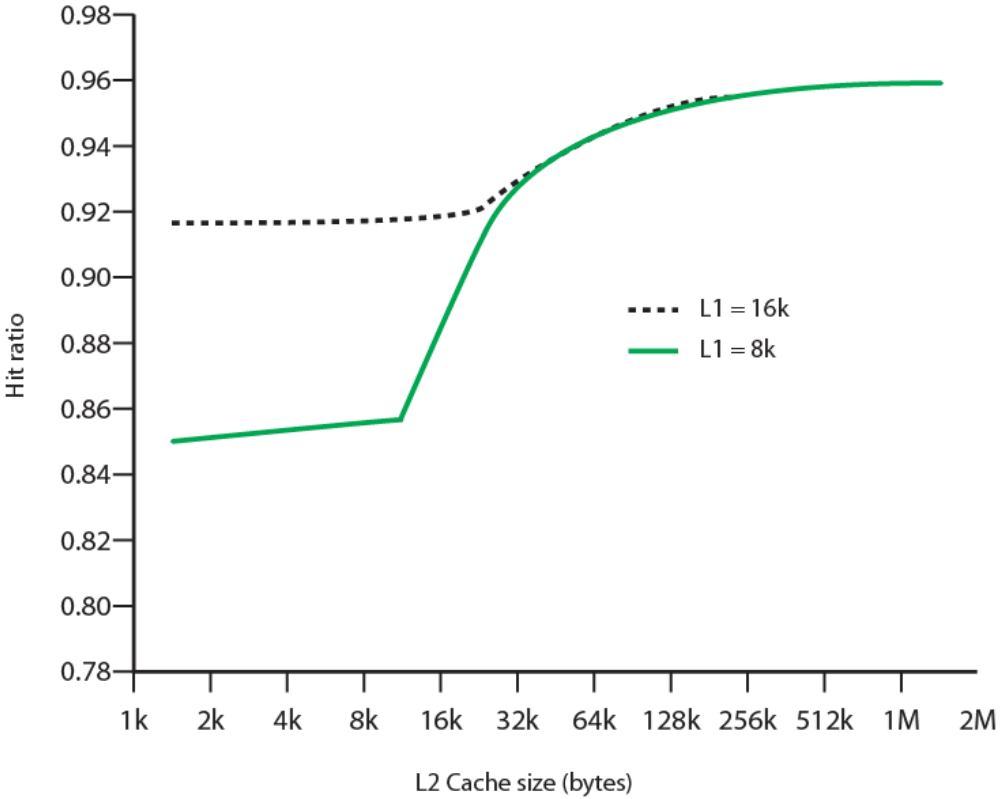


Local miss rate—Miss RateL2
Global miss rate—Miss RateL1 x Miss RateL2

L1 cache size : 32 KByte

Hit Ratio (L1 & L2)

- For 8 kbytes and 16 kbyte L1
- L2 cache size<32kb
  - 其他情况相同情况下，L1 cache的容量大则命中率高
  - 当L1 cache的容量不变时，随着L2 cache的容量变大，命中率也随之提高
- L2 cache size>32kb
  - L1 cache的容量大小对命中没有影响



Reducing Misses: Which apply to L2 Cache?

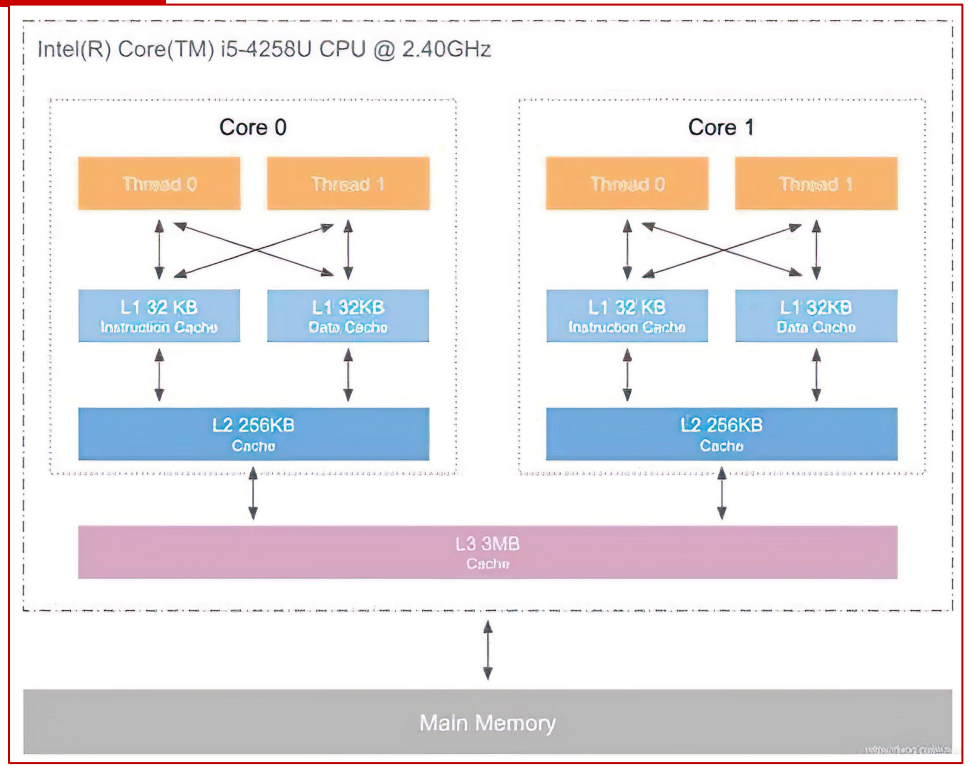
- Reducing Miss Rate
  - *Reduce Misses via Larger Block Size*
  - *Reduce Conflict Misses via Higher Associativity*
  - *Reducing Conflict Misses via Victim Cache*
  - *Reducing Conflict Misses via Pseudo-Associativity*
  - *Reducing Misses by HW Prefetching Instruction, Data*
  - *Reducing Misses by SW Prefetching Data*
  - *Reducing Misses by Compiler Optimizations*

Example(Pentium 4 Cache)

- 80386 – no on chip cache
- 80486 – 8k using 16 byte lines and four way set associative organization
- Pentium (all versions) – two on chip L1 caches
  - Data & instructions
- Pentium III – L3 cache added off chip
- Pentium 4
  - L1 caches
    - ✓ 8k bytes
    - ✓ 64 byte lines
    - ✓ four way set associative
  - L2 cache
    - ✓ Feeding both L1 caches
    - ✓ 256k
    - ✓ 128 byte lines
    - ✓ 8 way set associative
  - L3 cache on chip

Common Shared Main Memory(公共共享主内存)

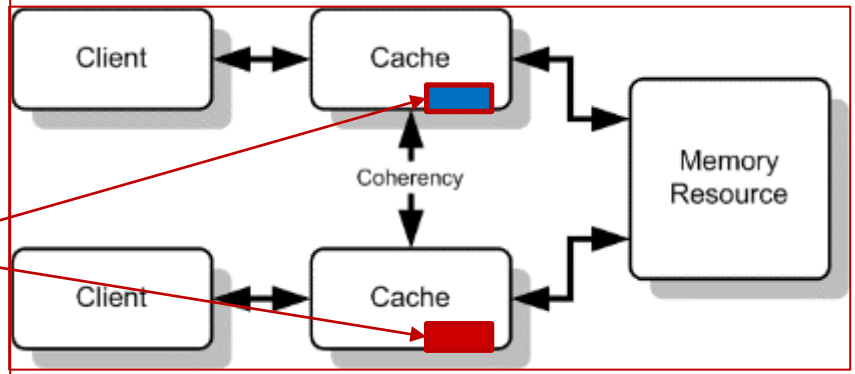
- A memory that may be simultaneously accessed by multiple programs with an intent to
  - provide communication among them or avoid redundant copies.
  - An efficient means of passing data between programs.
  - Depending on context, programs may run on a single processor or on multiple separate processors.
  - Using memory for communication inside a single program, e.g. among its multiple threads, is also referred to as shared memory.



Cache coherency (缓存一致性)

- Uniformity of shared resource data that ends up stored in multiple local caches

- Conflicts:
  - both the clients have a cached copy of a particular memory block from a previous read
  - Suppose the client on the bottom updates/ changes that memory block, the client on the top could be left with an invalid cache of memory without any notification of the change.



- Cache coherence is intended to manage such conflicts by maintaining a coherent view of the data values in multiple caches

Cache coherency

- Cache coherent processors
  - reading processor must get the **most current value**
  - most current value is the **last write**
- Cache coherency **problem**（缓存一致性的问题）
  - **updates** from 1 processor **not known to others**
- Mechanisms for maintaining cache coherency（保持缓存一致性的机制）
  - coherency state **associated with a block of data**
  - bus/interconnect operations on **shared data change the state**
    - ✓ **for other processors** that have the data of the operation resident in their caches

Cache Coherency Protocols

- **Write-invalidate**

  (Sequent Symmetry, SGI Power/Challenge, SPARCCenter 2000)

  - processor obtains **exclusive access for writes** (becomes the "owner") by invalidating data in other processors' caches
  - cache-to-cache transfers
  - good for:
    - ✓ multiple writes to same word or block by one processor
    - ✓ **migratory sharing** from processor to processor
  - a problem is false sharing (虚假共享)
    - ✓ processors read & write to ***different*** words in a shared cache block
    - ✓ cache coherency is maintained on a cache block basis
      - block ownership bounces between processor caches

Cache Coherency Protocols

- **Write-update**

  (SPARCCenter 2000)

  - broadcast each write to actively shared data
  - each processor with a copy snarfs the data
  - good for inter-processor contention

- **Competitive**

  - switches between them

Cache Coherency Protocol Implementations

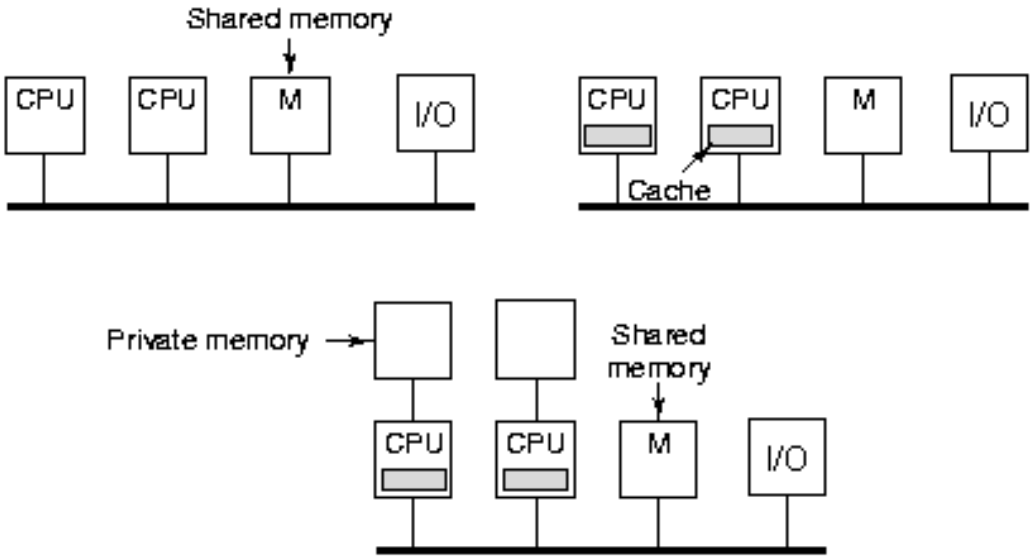
- **Snooping**（侦听）
  - used with low-end, bus-based MPs
    - ✓ few processors
    - ✓ centralized memory（集中存储）
- **Directory-based**（基于目录）
  - used with higher-end MPs
    - ✓ more processors
    - ✓ distributed memory

Multicore processors/ multiprocessor with shared memory (SMP)

Interconnection network for SMP system

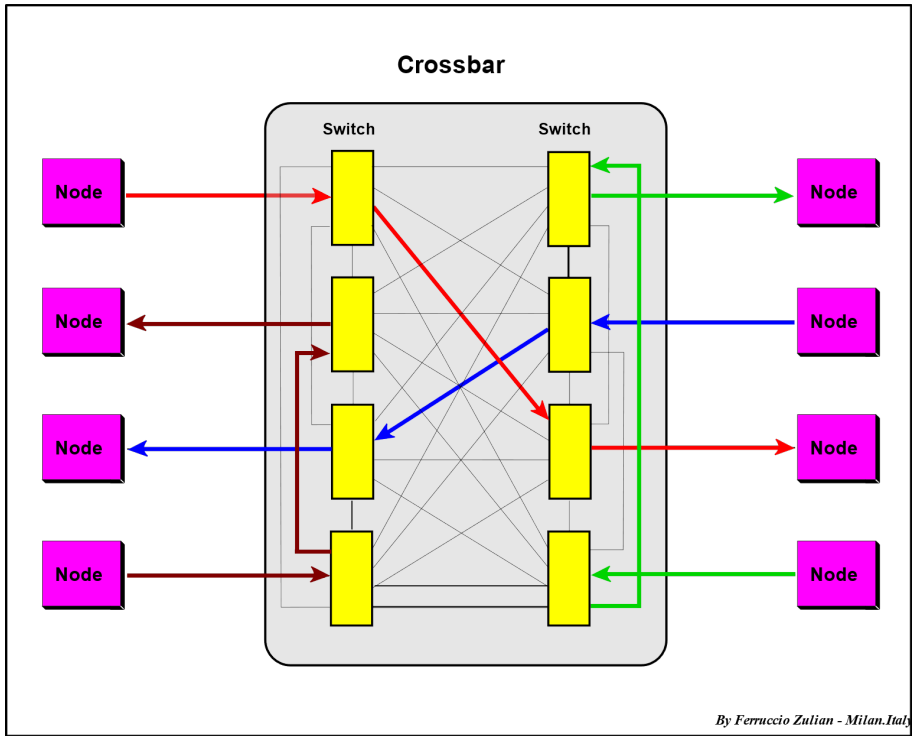
Bus-Based SMP (总线型SMP)

- A group of wires that connects different components of the computer.
  - It is used for transmitting data, control signal and memory address from one component to another.
  - A bus can be 8 bit, 16 bit, 32 bit and 64 bit. A 32 bit bus can transmit 32 bit information at a time.
  - A bus can be internal or external.



Crossbar-Based SMP (增强型SMP，交换式SMP)

- Crossbar networks allow any processor in the system to connect to any other processor or memory unit so that many processors can communicate simultaneously without contention.
  - A new connection can be established at any time as long as the requested input and output ports are free.
  - Crossbar networks are used in the design of high-performance small-scale multiprocessors, in the design of routers for direct networks, and as basic components in the design of large-scale indirect networks.



Multicore processors/ multiprocessor with shared memory (SMP)

Real modern multicore / SMP system architecture

Cluster of SMPs (CLUMPS)

- Four Sun E5000s （SMPs）
  - 8 processors
  - 4 Myricom NICs each (网卡)
- Multiprocessor, Multi-NIC (多网卡), Multi-Protocol （多协议）
- NPACI -> Sun 450s (台式服务器)

Conclusion

- **Multicore processors/ multiprocessor with shared memory (SMP)**
  - Generic SMP system architecture
  - Memory hierarchy
  - Interconnection network for SMP system
  - Real modern multicore / SMP system architecture