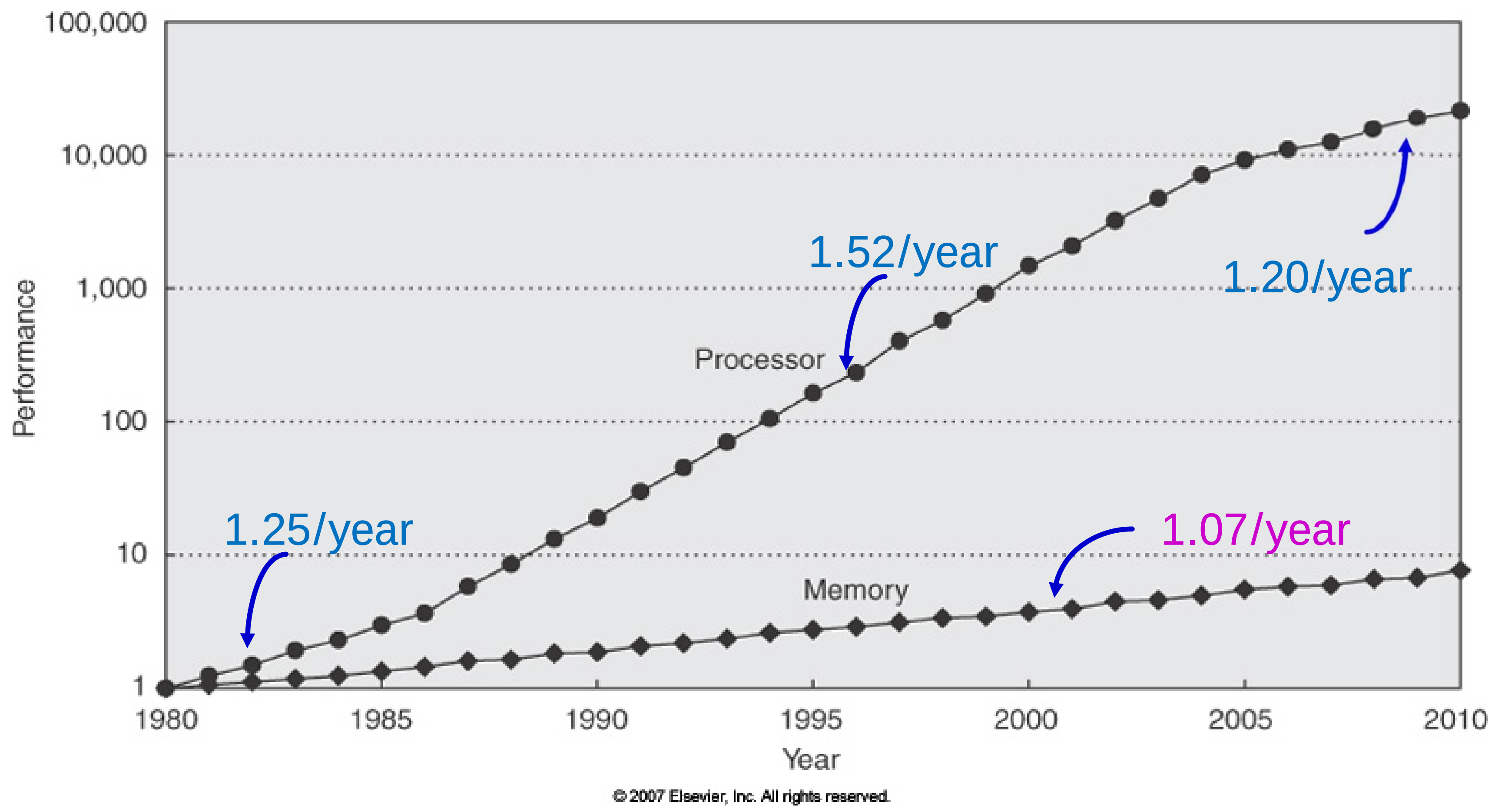
Lecture 20
Memory Hierarchy

School of Computer Science and Engineering
Soongsil University

5. Large and Fast: Exploiting Memory Hierarchy

5.1 Introduction

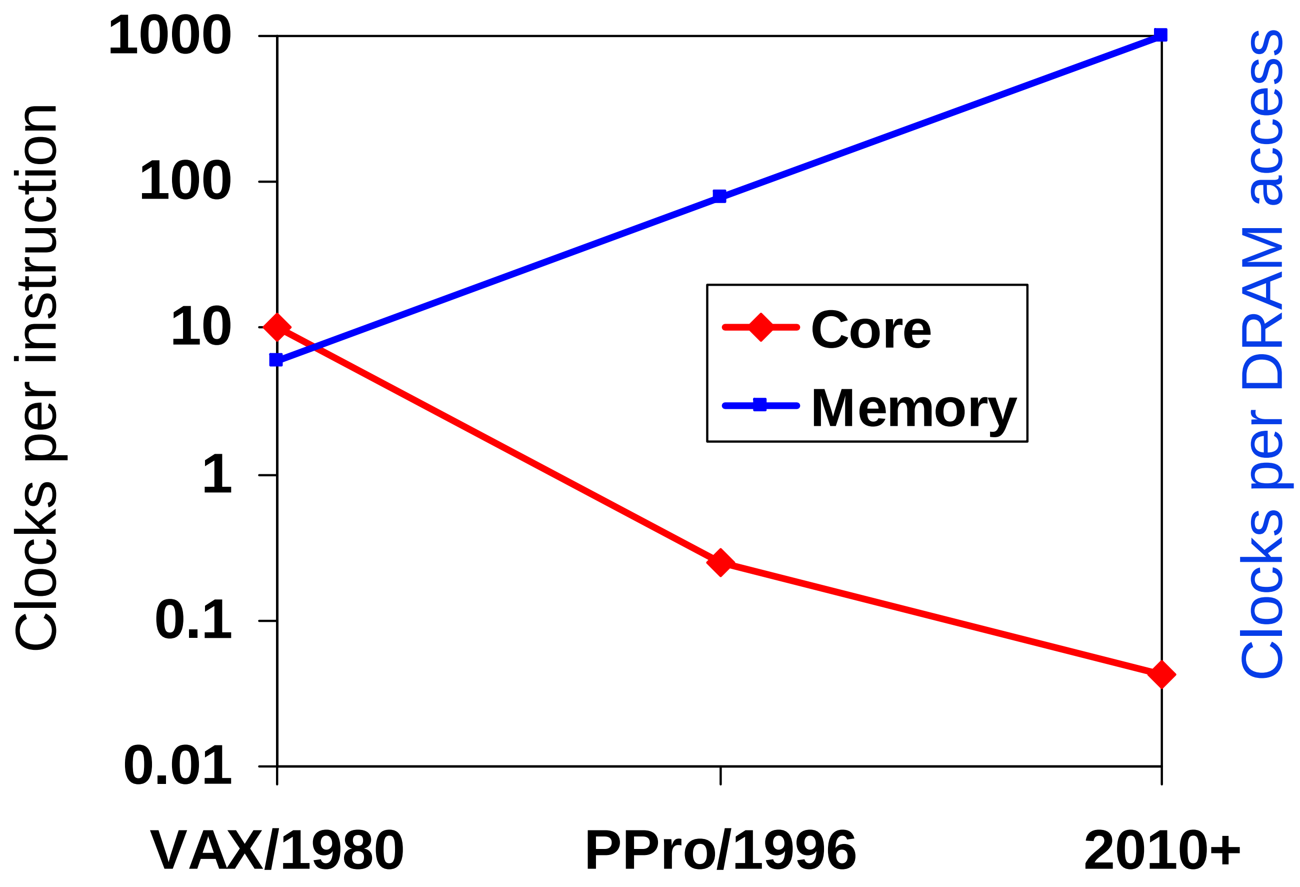
- **Processor-Memory Performance Gap**



Apple II (1977): CPU…1000ns, Memory … 400ns

The “Memory Wall”

- Processor vs. DRAM speed disparity continues to grow
- Good memory hierarchy (cache) design is increasingly important to overall performance



Memory Hierarchy

- **Engineering problem**
  - Maximum performance with minimum cost
- **Fact**
  - Fast memory → expensive → small
  - Slow memory → inexpensive → large
- **Goal**
  - To create the illusion of a large memory that we can access as fast as a very small memory
- **Multiple levels of memory**
  - Faster memory: Close to the processor
  - Slower and less expensive memory: Below that
- **Principle of locality**
  - Programs access a small proportion of their address space at any time

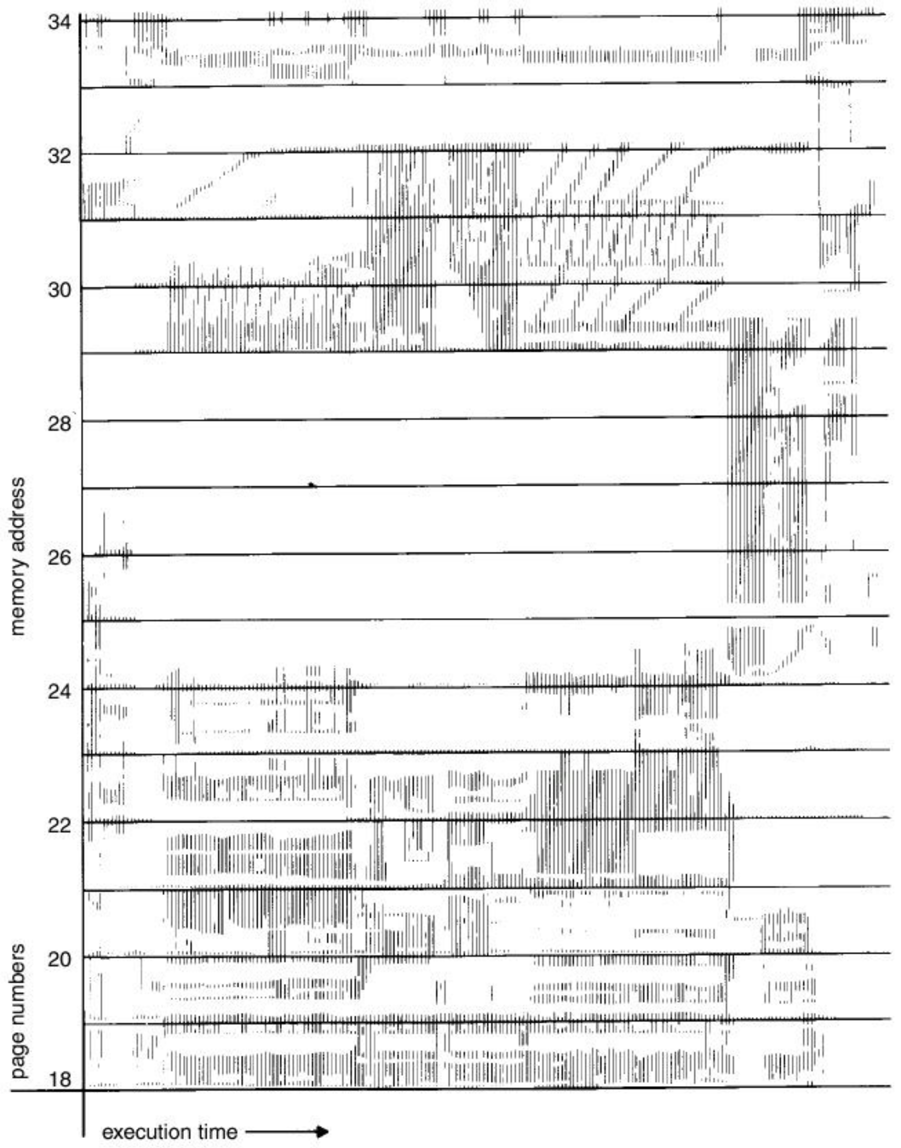
Principle of Locality

1. **Temporal locality**
   - Locality in time
   - Items accessed recently are likely to be accessed again soon

   [ex] loops
2. **Spatial locality**
   - Locality in space
   - Items near those accessed recently are likely to be accessed soon

   [ex] sequential execution

   array accesses



Basic Structure of a Memory Hierarchy

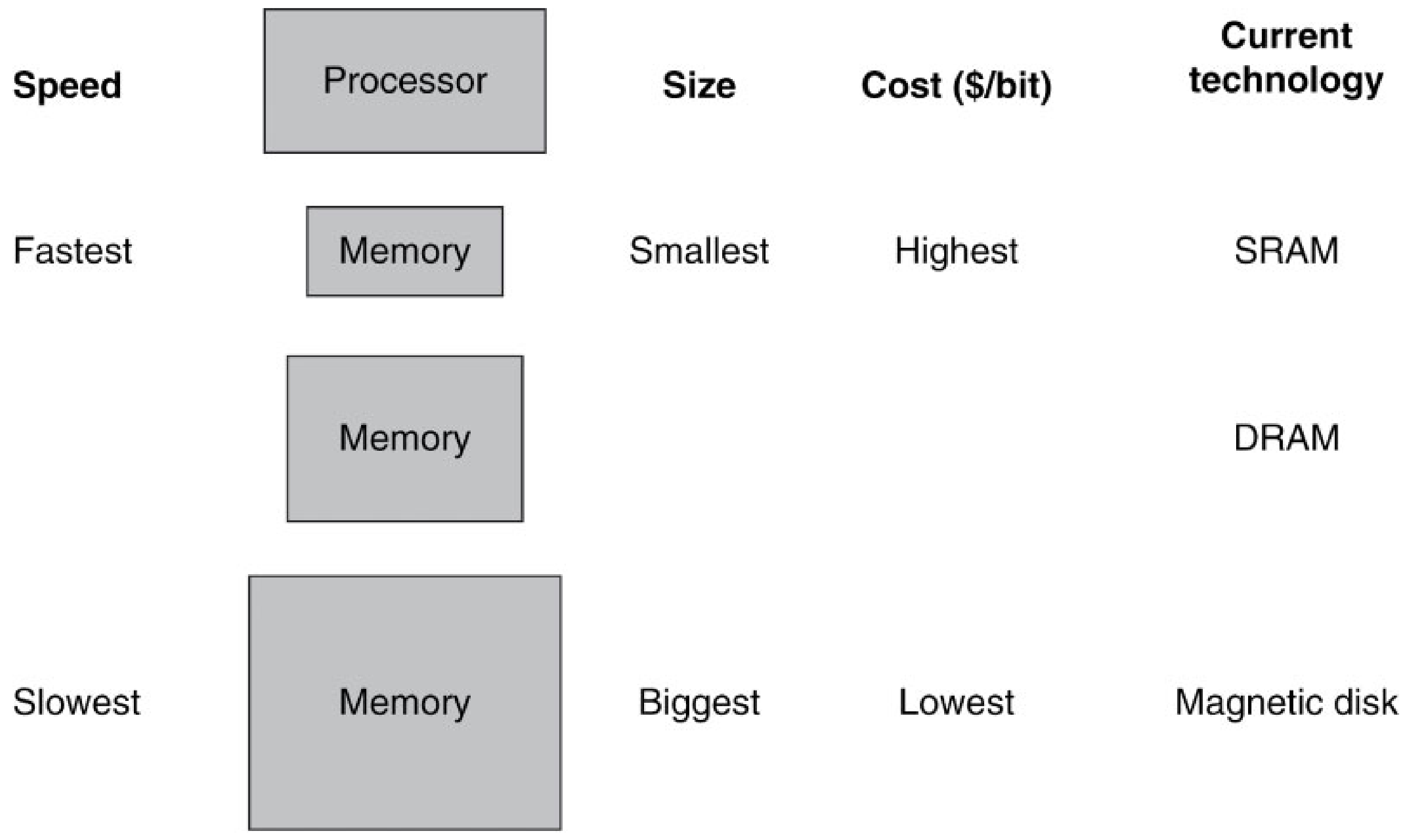
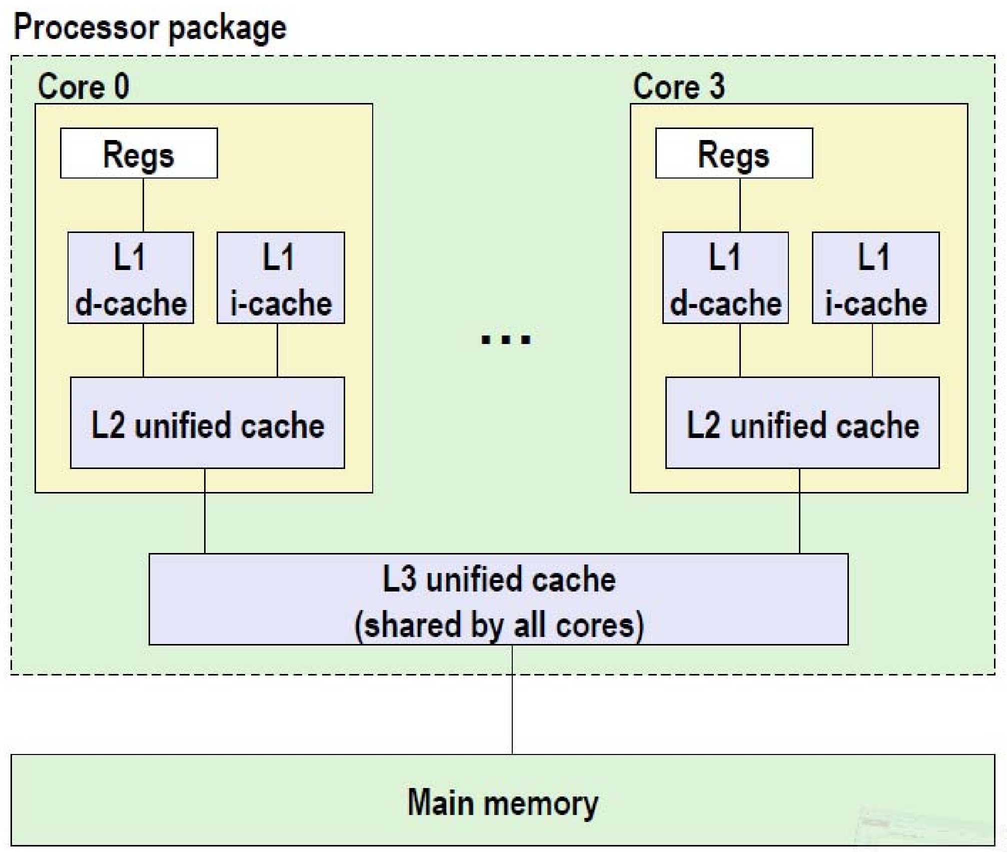


Figure 5.1

Memory Hierarchy in a Multicore Chip

Intel Core i7 Cache Hierarchy



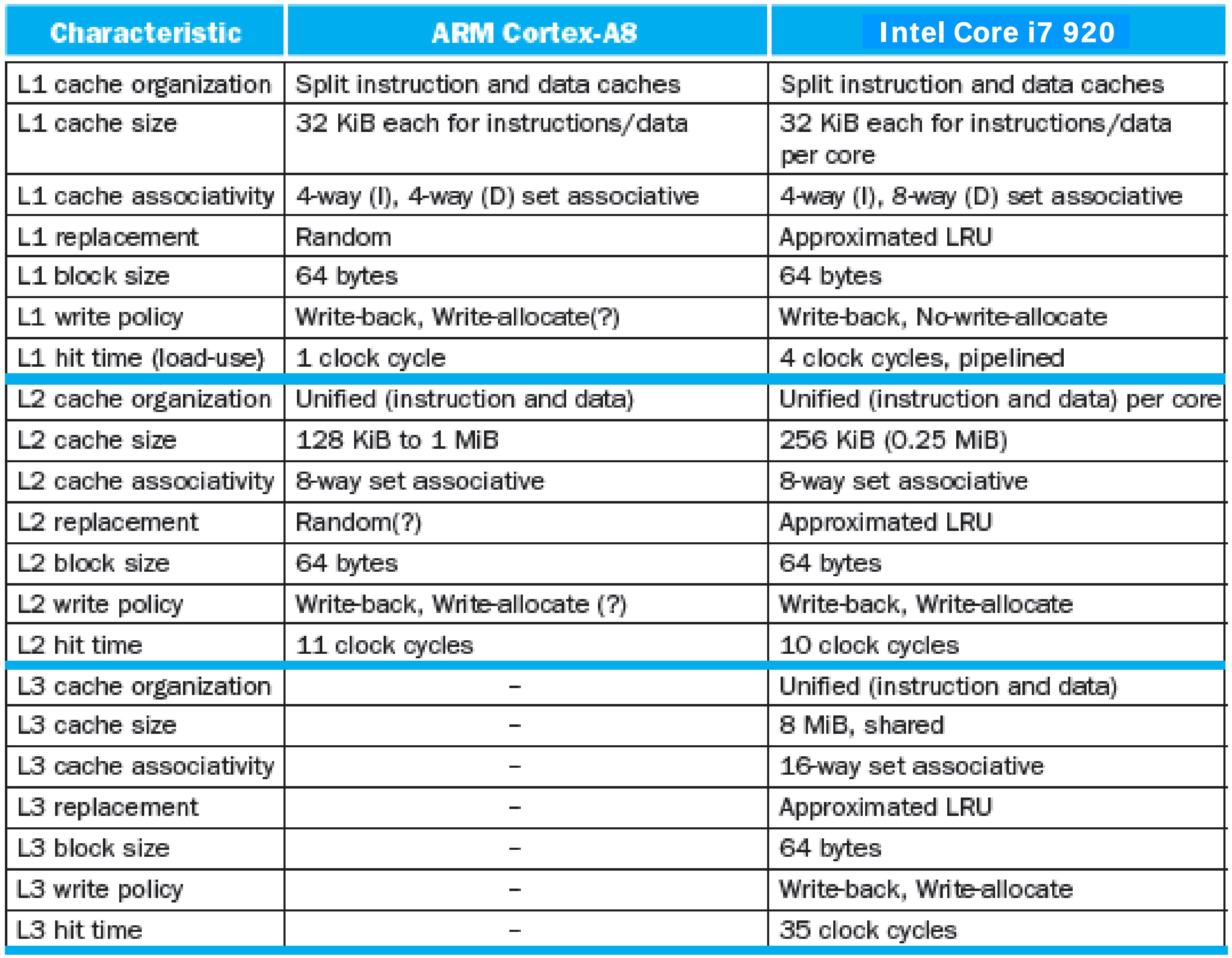
L1 i-cache and d-cache:
32 KB, 8-way,
Access: 4 cycles

L2 unified cache:
256 KB, 8-way,
Access: 11 cycles

L3 unified cache:
8 MB, 16-way,
Access: 30-40 cycles

Block size: 64 bytes for all caches.

ARM Cortex-A8 and Intel Core i7 920



Characteristic	ARM Cortex-A8	Intel Core i7 920
L1 cache organization	Split instruction and data caches	Split instruction and data caches
L1 cache size	32 KiB each for instructions/data	32 KiB each for instructions/data per core
L1 cache associativity	4-way (I), 4-way (D) set associative	4-way (I), 8-way (D) set associative
L1 replacement	Random	Approximated LRU
L1 block size	64 bytes	64 bytes
L1 write policy	Write-back, Write-allocate(?)	Write-back, No-write-allocate
L1 hit time (load-use)	1 clock cycle	4 clock cycles, pipelined
L2 cache organization	Unified (instruction and data)	Unified (instruction and data) per core
L2 cache size	128 KiB to 1 MiB	256 KiB (0.25 MiB)
L2 cache associativity	8-way set associative	8-way set associative
L2 replacement	Random(?)	Approximated LRU
L2 block size	64 bytes	64 bytes
L2 write policy	Write-back, Write-allocate (?)	Write-back, Write-allocate
L2 hit time	11 clock cycles	10 clock cycles
L3 cache organization	–	Unified (instruction and data)
L3 cache size	–	8 MiB, shared
L3 cache associativity	–	16-way set associative
L3 replacement	–	Approximated LRU
L3 block size	–	64 bytes
L3 write policy	–	Write-back, Write-allocate
L3 hit time	–	35 clock cycles

Figure 5.44

Hit and Miss

- **Block (= line)**
  - Minimum unit of information transfer between the two levels
- **Hit**
  - When the data requested by the processor is in the upper level
- **Miss**
  - When the data is not found in the upper level
  - Then access the lower level
- **Hit rate (=hit ratio)**
  - The fraction of memory accesses found in the upper level
- **Miss rate**
  - 1 - hit rate

Performance of the Memory Hierarchy

- **Hit time**
  - Time to access the upper level of the memory hierarchy
  - Time to access the block + Time to determine hit/miss
- **Miss penalty**
  - Time to access the block in the lower level

    + Time to transmit that block to the level that experienced the miss

    + Time to insert the block in that level

    + Time to pass the block to the requestor
- **Average memory access time (AMAT)**
  - hit time + miss rate x miss penalty

The BIG Picture

- **Temporal locality and memory hierarchy**
  - Keeping more recently accessed data items closer to the processor
- **Spatial locality and memory hierarchy**
  - Moving blocks consisting of multiple contiguous words to upper level
- **Memory hierarchy with high hit rate**
  - Access time: close to that of the highest level
  - Size: equal to that of the lowest level
- **Multi-level inclusion property**
  - Level(i) $\subset$ Level(i+1)

5.2 Memory Technologies

Memory technology	Typical access time	$ per GiB in 2012
SRAM semiconductor memory	0.5–2.5 ns	$500–$1000
DRAM semiconductor memory	50–70 ns	$10–$20
Flash semiconductor memory	5,000–50,000 ns	$0.75–$1.00
Magnetic disk	5,000,000–20,000,000 ns	$0.05–$0.10

- **SRAM (Static random access memory)**
  - Fixed access time to any datum, though the read and write access times may differ
  - No need to refresh, and so the access time is very close to the cycle time
  - Six to eight transistors per bit
- **DRAM (Dynamic random access memory)**
  - Value kept in a cell is stored as a charge in a capacitor … 1 transistor/bit
  - Dynamic: needs to be “refreshed” regularly (~ every 8 ms)
    - Consumes 1% to 2% of the active cycles of the DRAM

5.3 The Basics of Caches

- **Definition of Cache**
  - The level of memory hierarchy between CPU and main memory
  - Any storage managed to take advantage of locality of access
- **The first paper … Wilkes[1965]**
  - “Slave memories and dynamic storage allocation”
- **The first implementation**
  - At the University of Cambridge by Scarrott
- **The first commercial machine with a cache**
  - IBM 360/85, late 1960s
- **The first usage of the term “cache”**
  - Conti, Gibson and Pitkowsky
  - a paper in IBM Systems Journal in 1968

Simple Cache Scenario

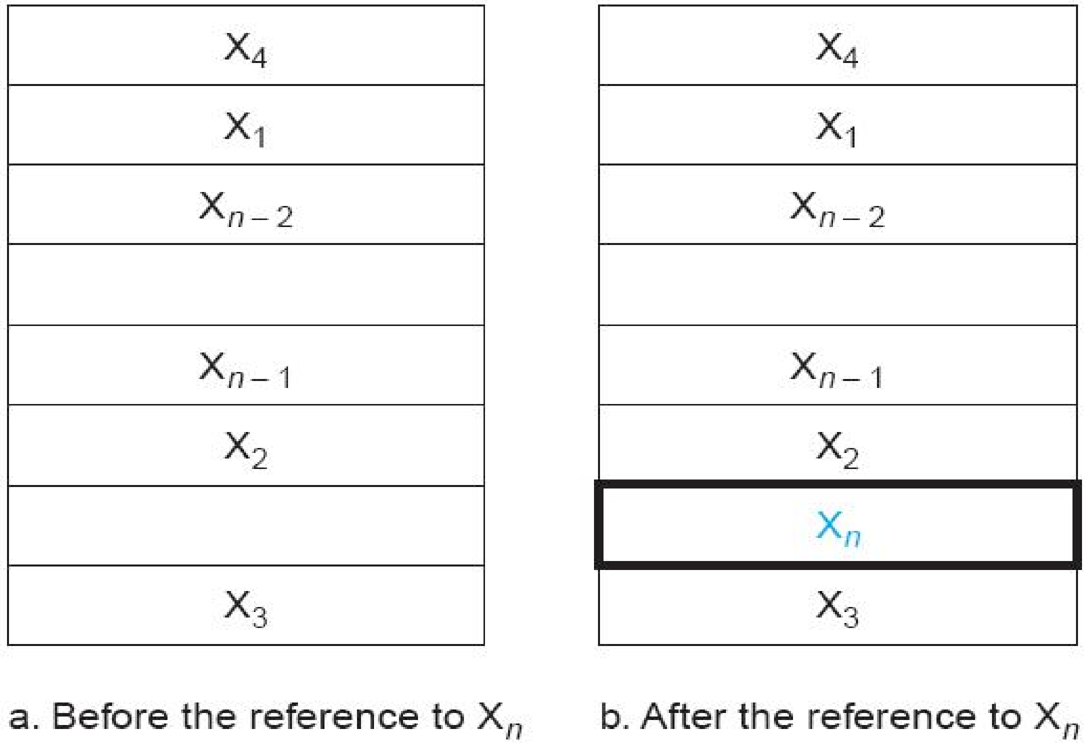


Figure 5.7

- **2 questions**
  - Q1: How do we know if a data item is in the cache?
  - Q2: If it is, how do we find it?