CSE251: System Programming

11. Memory Hierarchy

Seongil Wi

Department of Computer Science and Engineering

HW2: Bomb Defuse Game

- Due: Apr. 9 (Today), 11:59PM

- Your goal is to defuse your bomb
  - You will earn points for each phase you defuse, with a maximum total of 145 points
  - Each time the bomb explodes, you will lose 3 points (Only to the first 20 explosions)

- You should submit your report via BlackBoard

HW3: Symbol Resolution

- Due: Apr. 24, 11:59 PM

- Main task: Your job is to use your knowledge of C, ELF, and linkers to recreate `hw3_a.c` and `hw3_b.c` so that their symbol tables match those shown in the hw3 handout

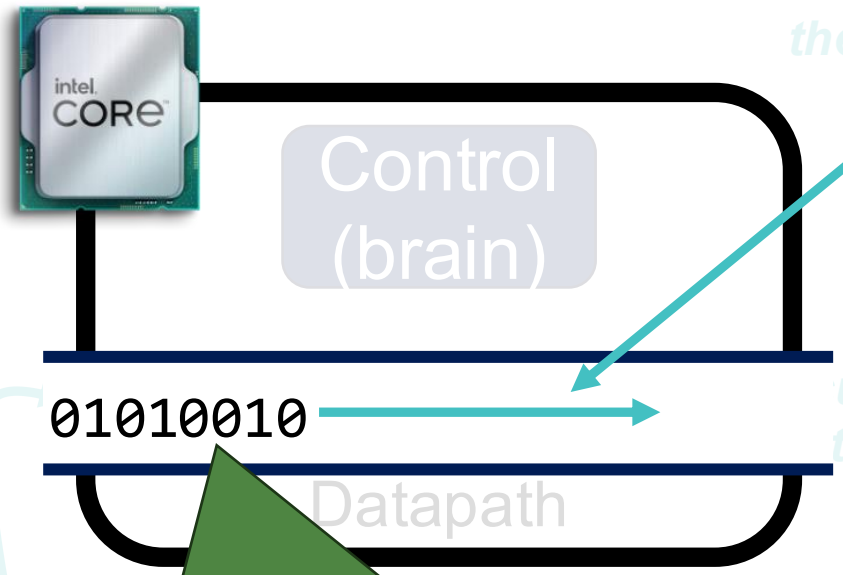
- You should submit a zip via BlackBoard

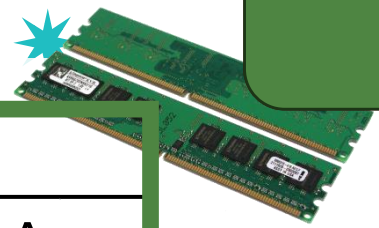
No Class on April 16 ☺

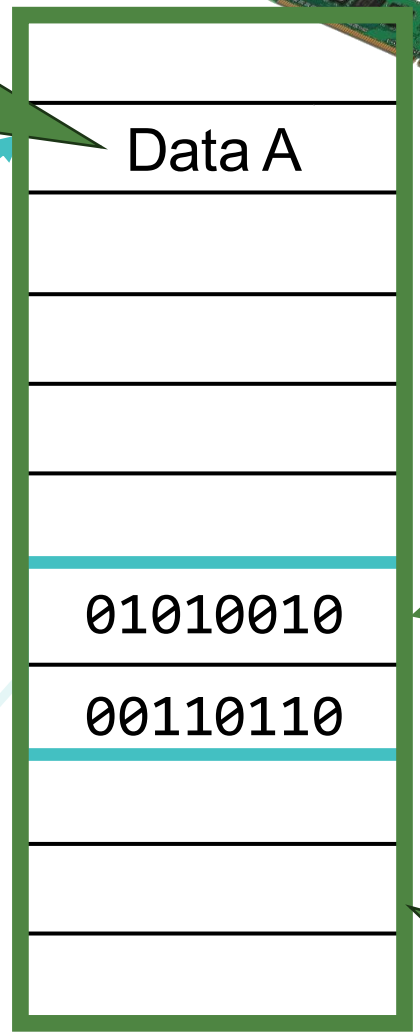
- There will be no class on April 16 to accommodate students' exam period and to provide time for working on HW3

Where Are We?

How data (bit, byte, int, …) is represented in memory?







Main memory

Executable and Linkable Format (ELF), Linking Process

1. Load the Program A

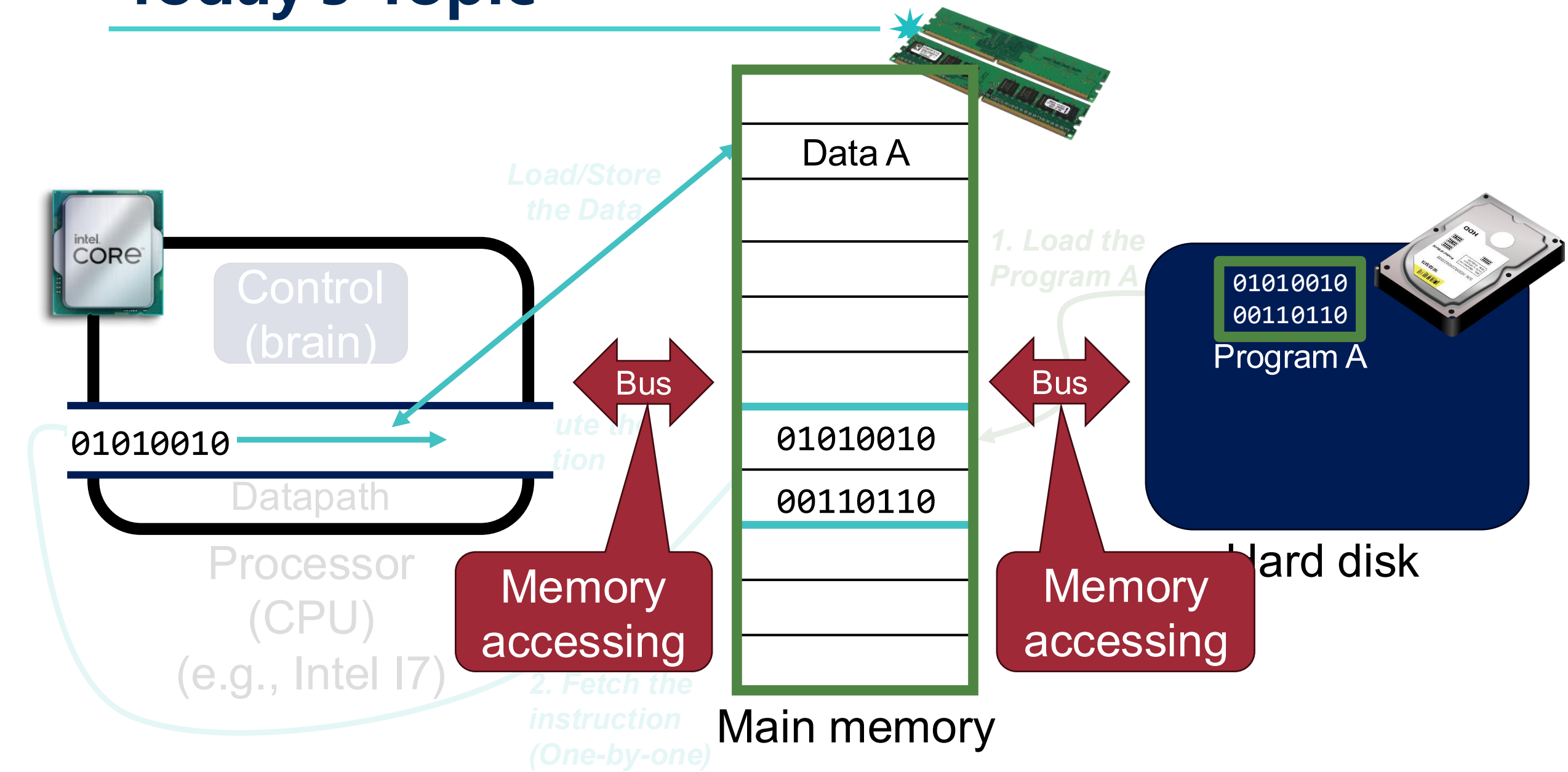
Hard disk

Loading process via loader (kernel)

How is assembly code interpreted, and how does it interact with memory?

Stack, heap, code, data regions

Today's Topic



Today's Topic

Memory

Accessing memory is one of the biggest performance bottlenecks ☹

Performance Bottleneck

Performance bottleneck ☹

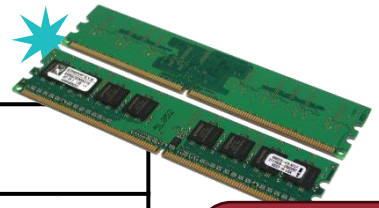
Control (brain)

01010010

Datapath

Processor (CPU) (e.g., Intel I7)

Bus



Data A

01010010

00110110

Main memory

Bus

Performance bottleneck ☹

01010010
00110110
Program A

11001110
01000110
Program B

01101101
11010111
Program C

Hard disk

Storage Inside in CPU

Performance bottleneck ☹

Performance bottleneck ☹

Control (brain)

Register 1

Register 2

01010010

Datapath

(CPU)

(e.g., Intel I7)

Bus

Bus

Data A

01010010

00110110

Main memory

01010010
00110110

Program A

11001110

Program C

01101101
11010111

Fast but small

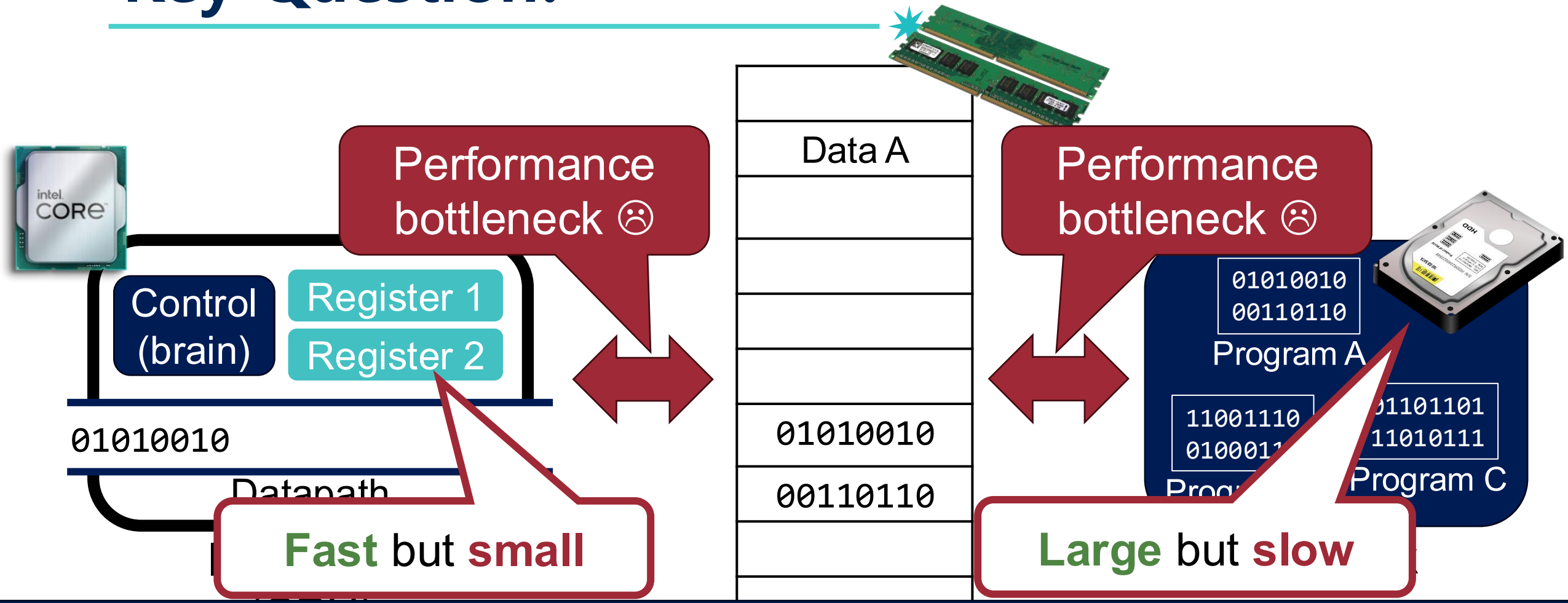
Large but slow

Storage: Smaller is Faster!

The faster memories are *more expensive per bit* than the slower memories and thus are smaller!

Memory	Typical access time	$ per GB (in 2020)
Static RAM (SRAM)	0.5ns – 2.5ns	$2,000 – $5,000
Dynamic RAM (DRAM)	50ns – 70ns	$20 – $75
Magnetic Disk	5ms – 20ms	$0.20 – $2

Key Question!



There is a *conflict* having large and fast memory ☹

How can we achieve *large and fast memory*?

Memory Hierarchy

Solution: Memory Hierarchy

A structure that uses multiple levels of memories

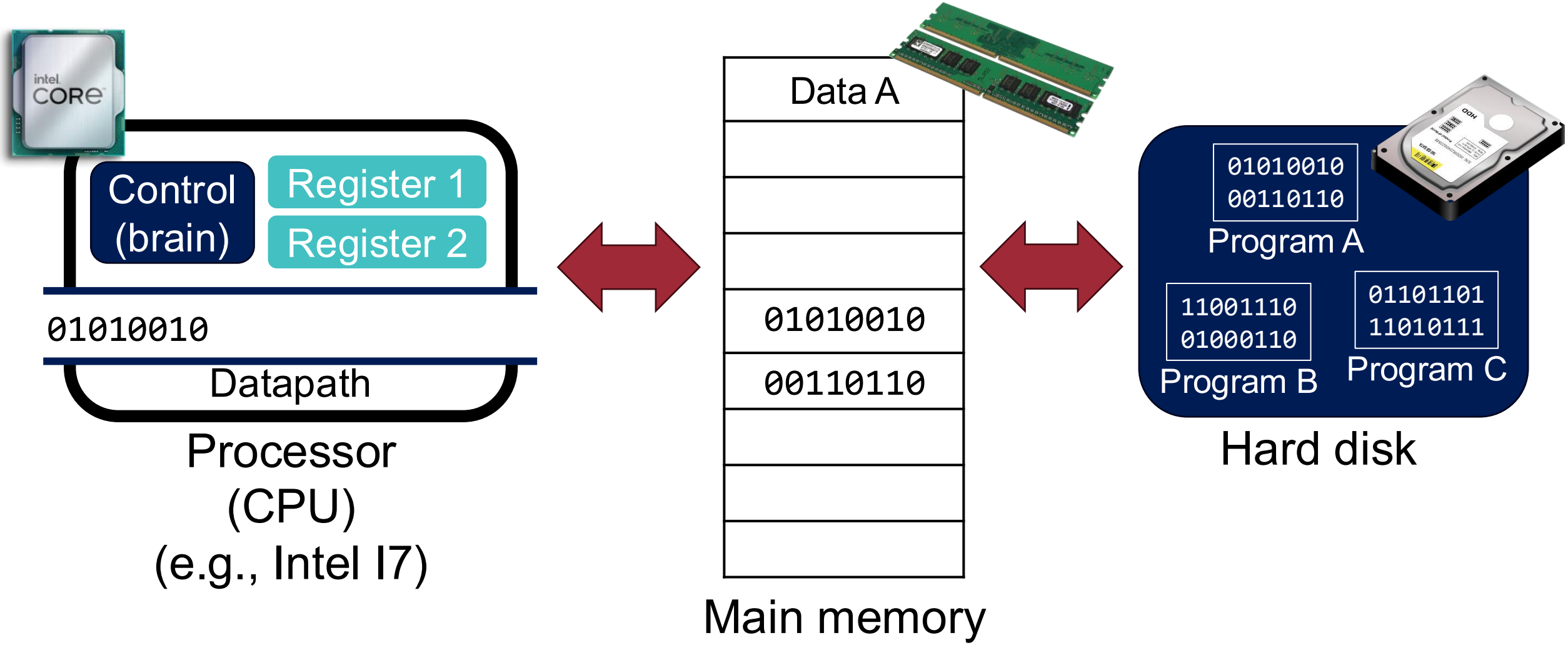
Exploiting Hierarchy Example: Book Storage

Exploiting Hierarchy Example: Book Storage

How about Level 3?

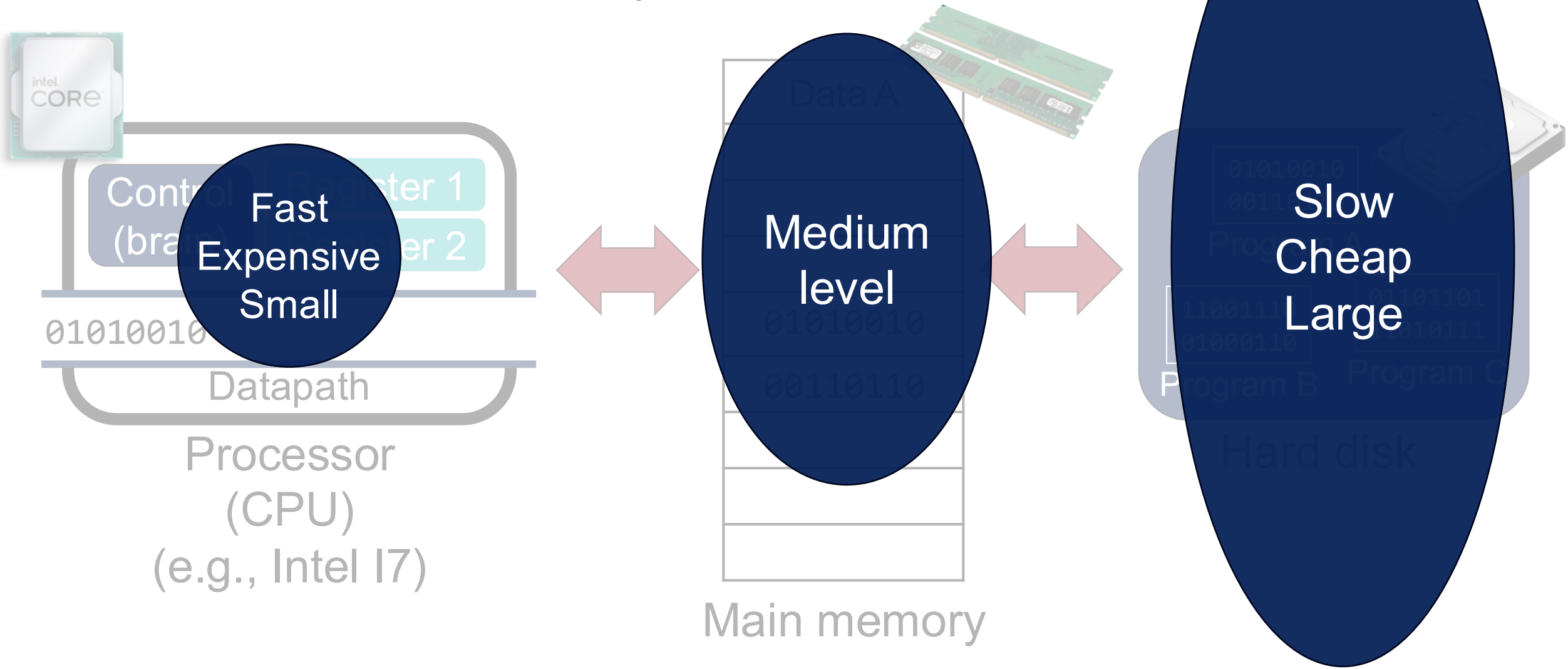
Solution: Memory Hierarchy

A structure that uses multiple levels of memories



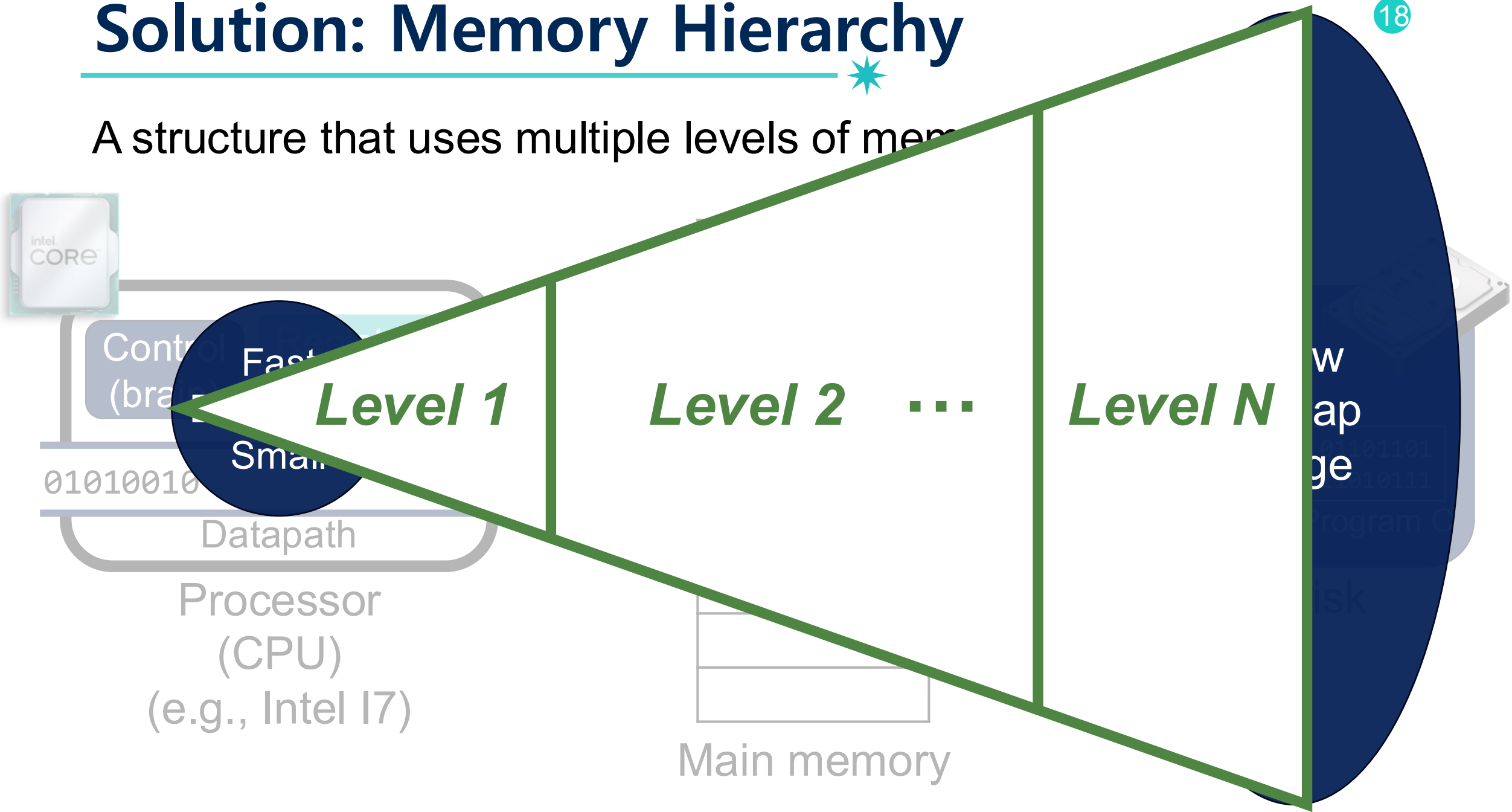
Solution: Memory Hierarchy

A structure that uses multiple levels of memories



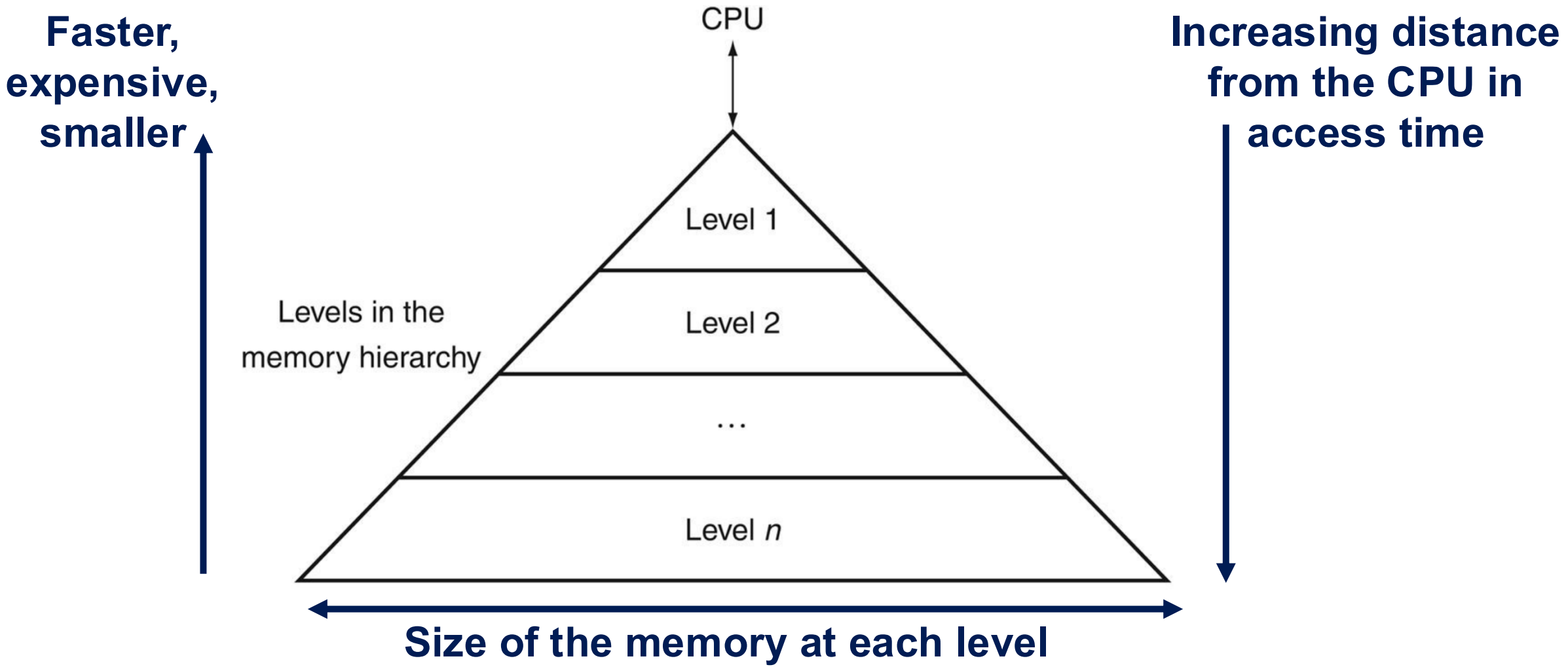
Solution: Memory Hierarchy

A structure that uses multiple levels of mem



Solution: Memory Hierarchy

A structure that uses multiple levels of memories



Memory Hierarchy Details

Register 1
Register 2
On-chip cache (L1 cache)

Processor (CPU) (e.g., Intel I7)

SRAM (L2 cache)

DRAM (Main memory)

Main memory

Secondary storage (Hard disk)

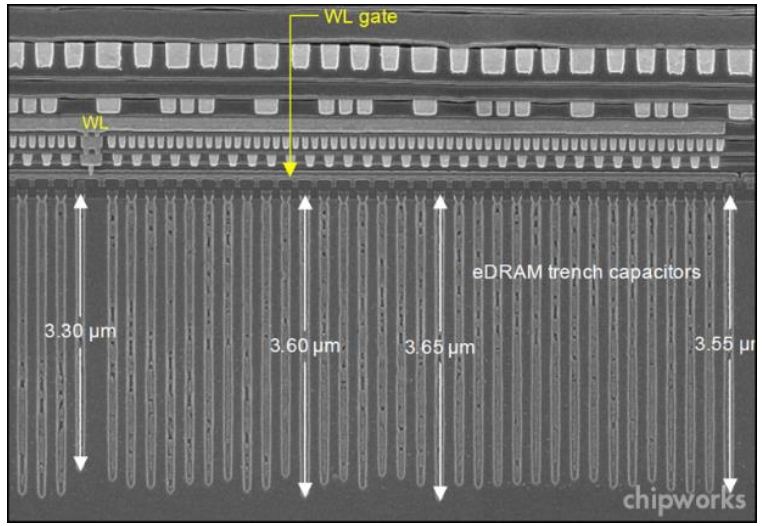
FYI: Random-Access Memory (RAM)

- Key features
  - RAM is traditionally packaged as a chip
    - or embedded as part of processor chip
  - Basic storage unit is normally a cell (one bit per cell)
  - Multiple RAM chips form a memory

- RAM comes in two varieties:
  - SRAM (Static RAM)
  - DRAM (Dynamic RAM)

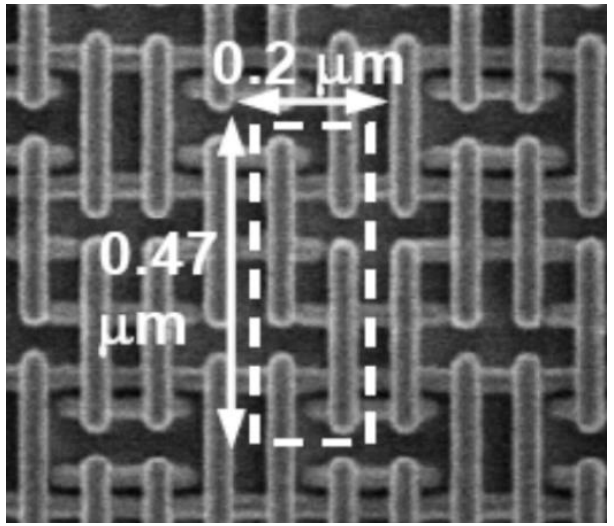
FYI: RAM Technologies

- DRAM



- 1 Transistor + 1 capacitor / bit
  - Capacitor oriented vertically
- Must refresh state periodically

- SRAM



- 6 transistors bit
- Holds state indefinitely

FYI: SRAM vs DRAM Summary

	Transistors per bit	Access time	Needs refresh?	Needs *EDC?	Cost	Applications
SRAM	6 or 8	1x	No	Maybe	100x	Cache memories
DRAM	1	10x	Yes	Yes	1x	Main memories, frame buffers

*EDC: Error detection and correction

- Trends
  - SRAM scales with semiconductor technology
    - Reaching its limits
  - DRAM scaling limited by need for minimum capacitance
    - Aspect ratio limits how deep can make capacitor
    - Also reaching its limits

Memory Hierarchy Details

Register 1
Register 2
On-chip cache (L1 cache)

Processor (CPU) (e.g., Intel I7)

SRAM (L2 cache)

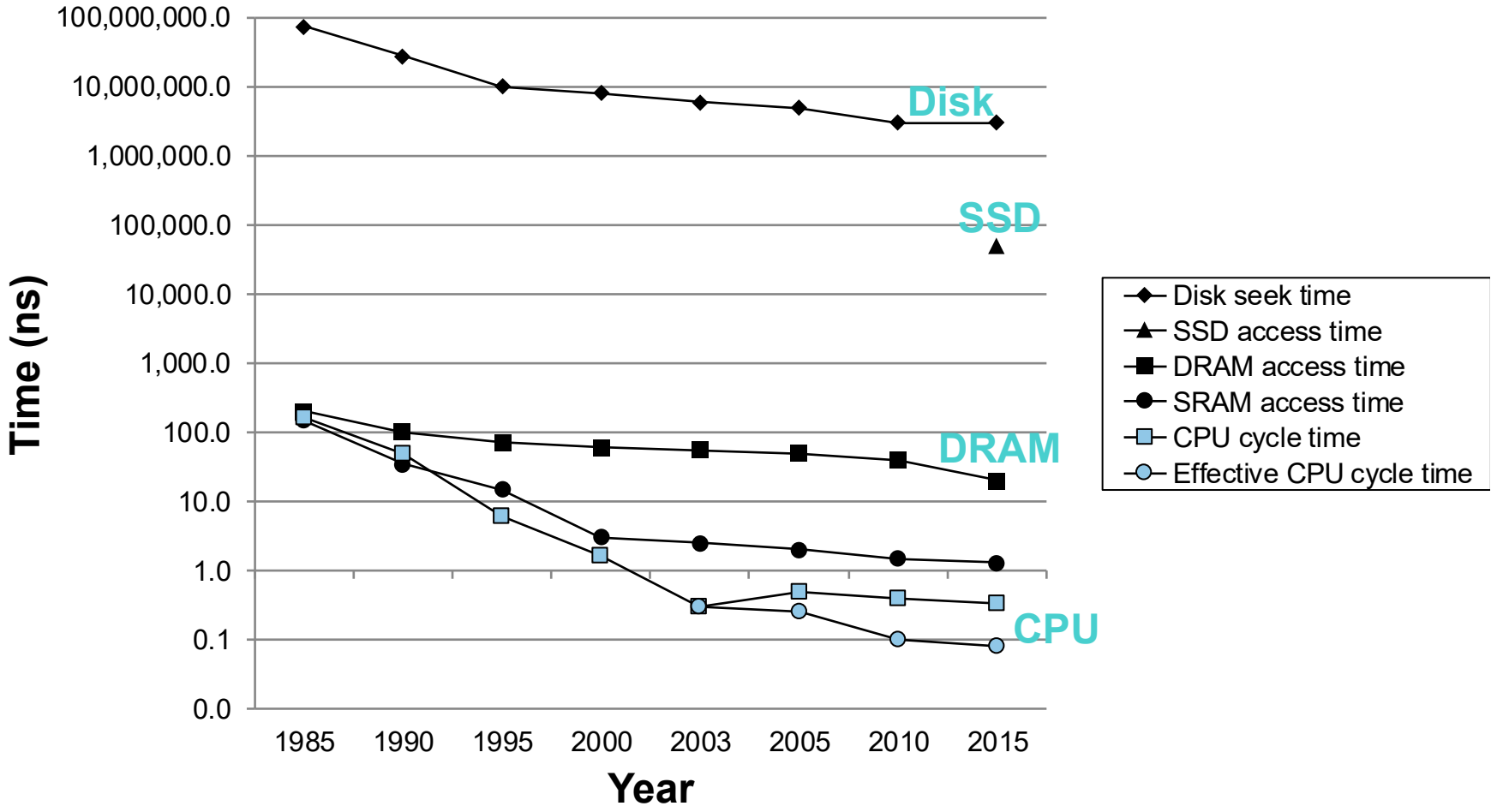
DRAM (Main memory)

Main memory

Secondary storage (Hard disk)

The CPU-Memory Gap

- The gap *widens* between DRAM, disk, and CPU speeds.



Why does the Memory Hierarchy Work?

Because of the principle of ***locality!***

Important Principle: Locality (지역성)

The tendency to access the same set of memory locations *repetitively* over a short period of time

1. **Temporal locality** (locality in time)

2. **Spatial locality** (locality in space)

Important Principle: Locality (지역성)

The tendency to access the same set of memory locations *repetitively* over a short period of time

1. **Temporal locality** (locality in time)
   - If an item is referenced, *the same item* will tend to be referenced again soon

2. **Spatial locality** (locality in space)

Important Principle: Locality (지역성)

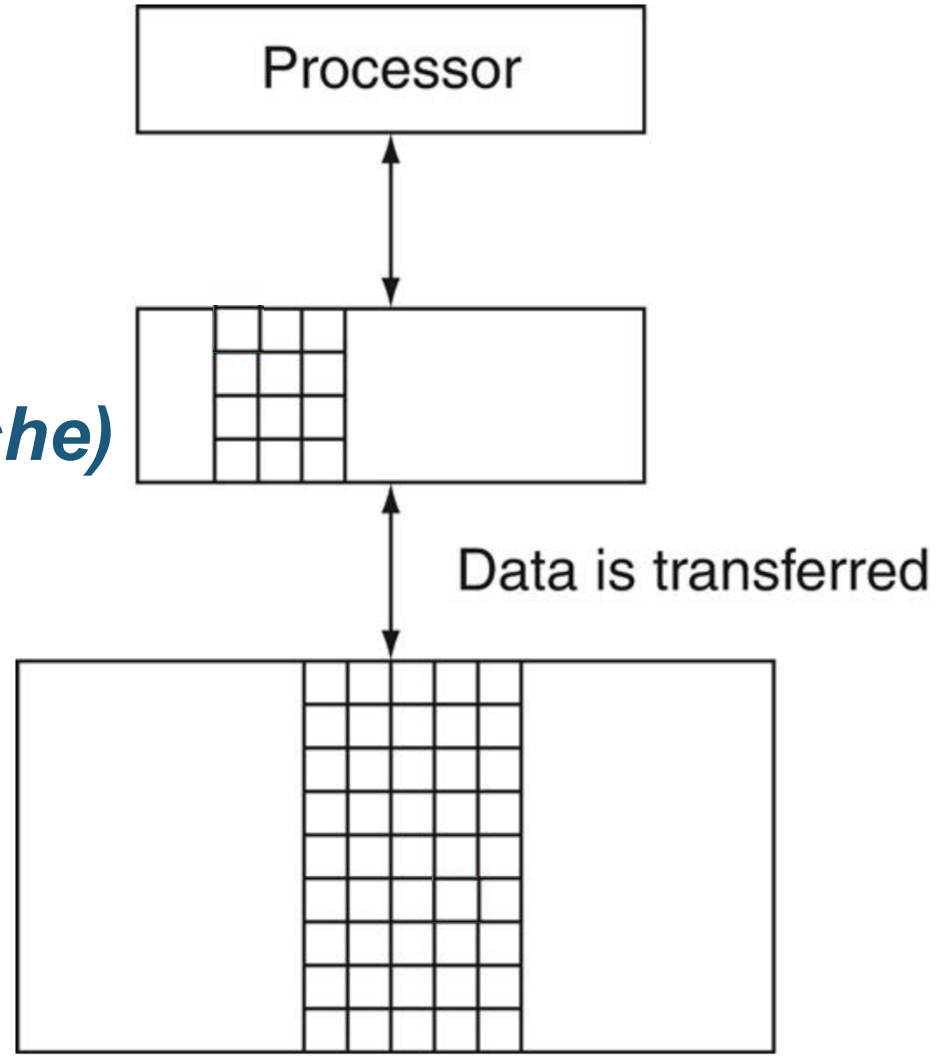
The tendency to access the same set of memory locations *repetitively* over a short period of time

1. **Temporal locality** (locality in time)
   - If an item is referenced, *the same item* will tend to be referenced again soon

2. **Spatial locality** (locality in space)
   - If an item is referenced, *nearby items* will tend to be referenced soon

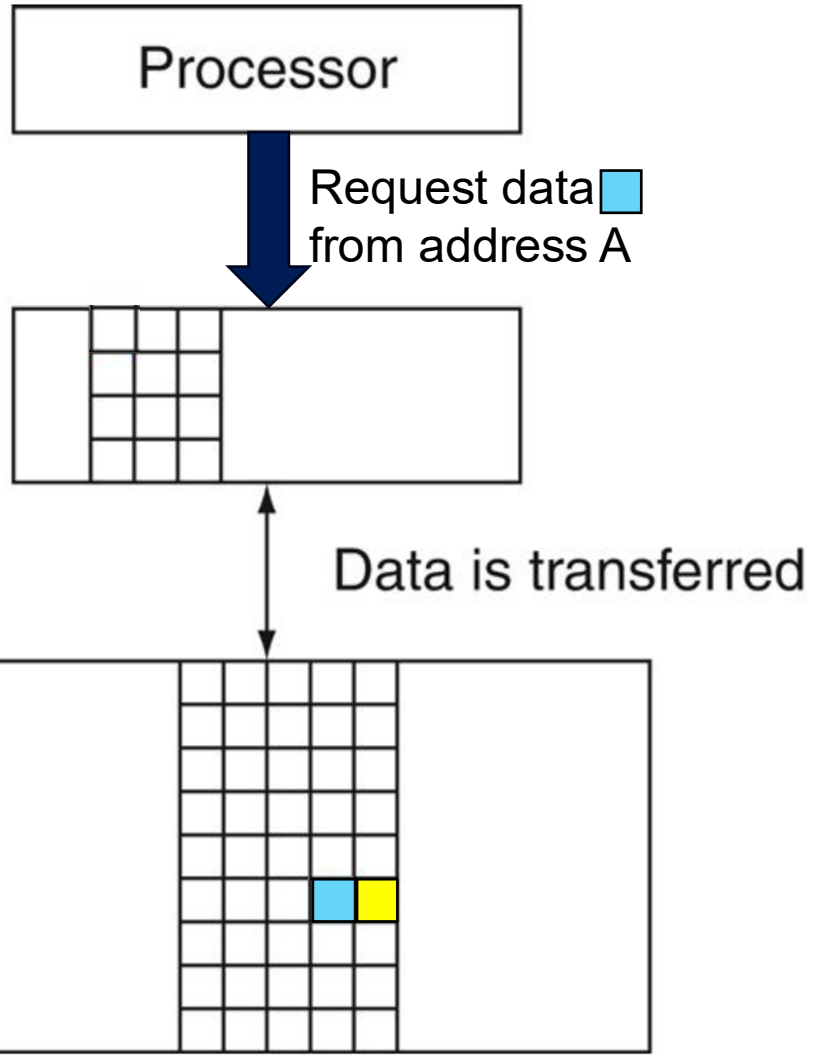
Locality Example: 1st Access

- Assumption: two levels (*upper*, *lower*)
  - Each pair of levels in the memory hierarchy can be thought of as having an *upper* and *lower level*



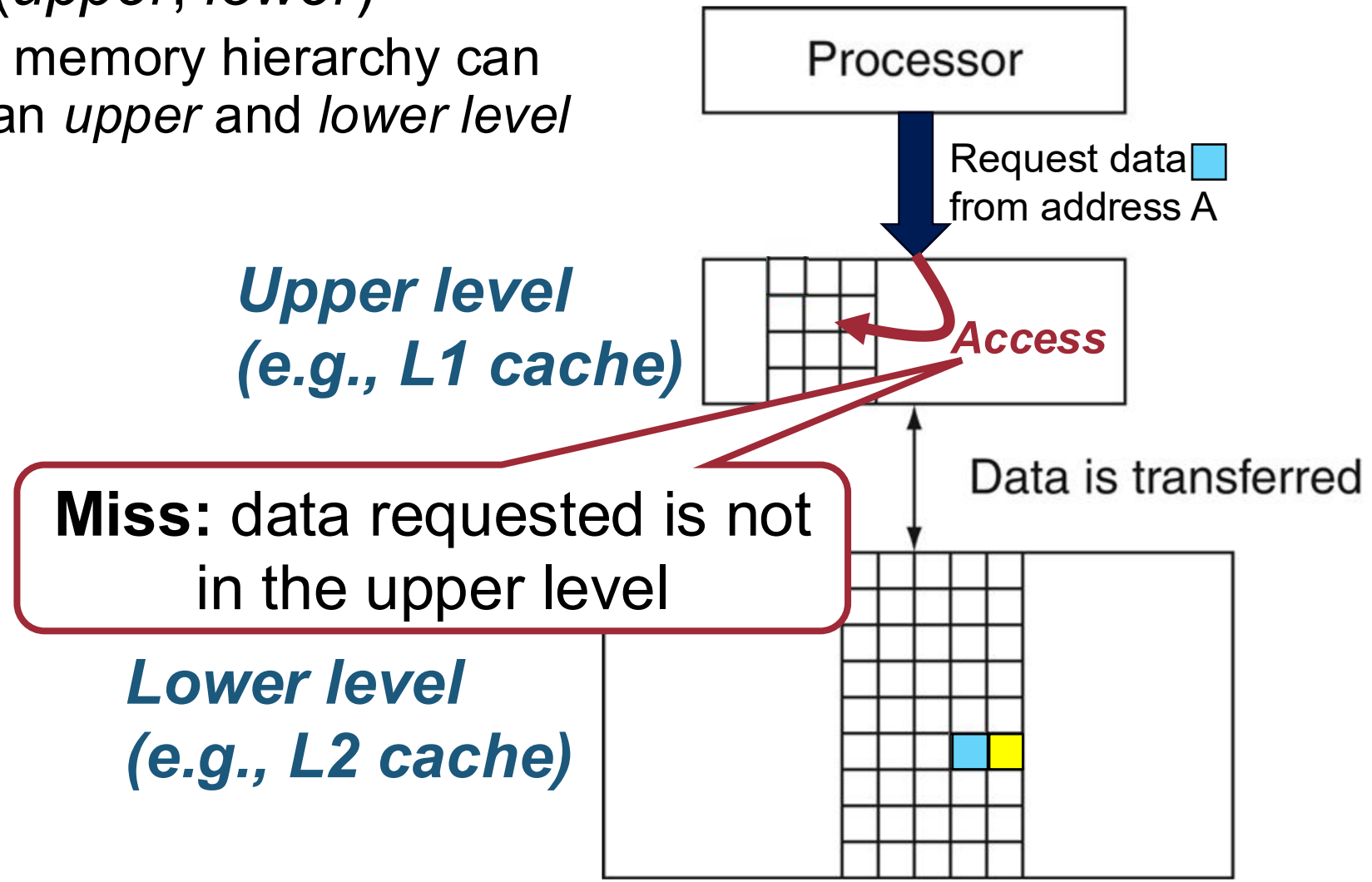
Locality Example: 1st Access

- Assumption: two levels (*upper*, *lower*)
  - Each pair of levels in the memory hierarchy can be thought of as having an *upper* and *lower level*



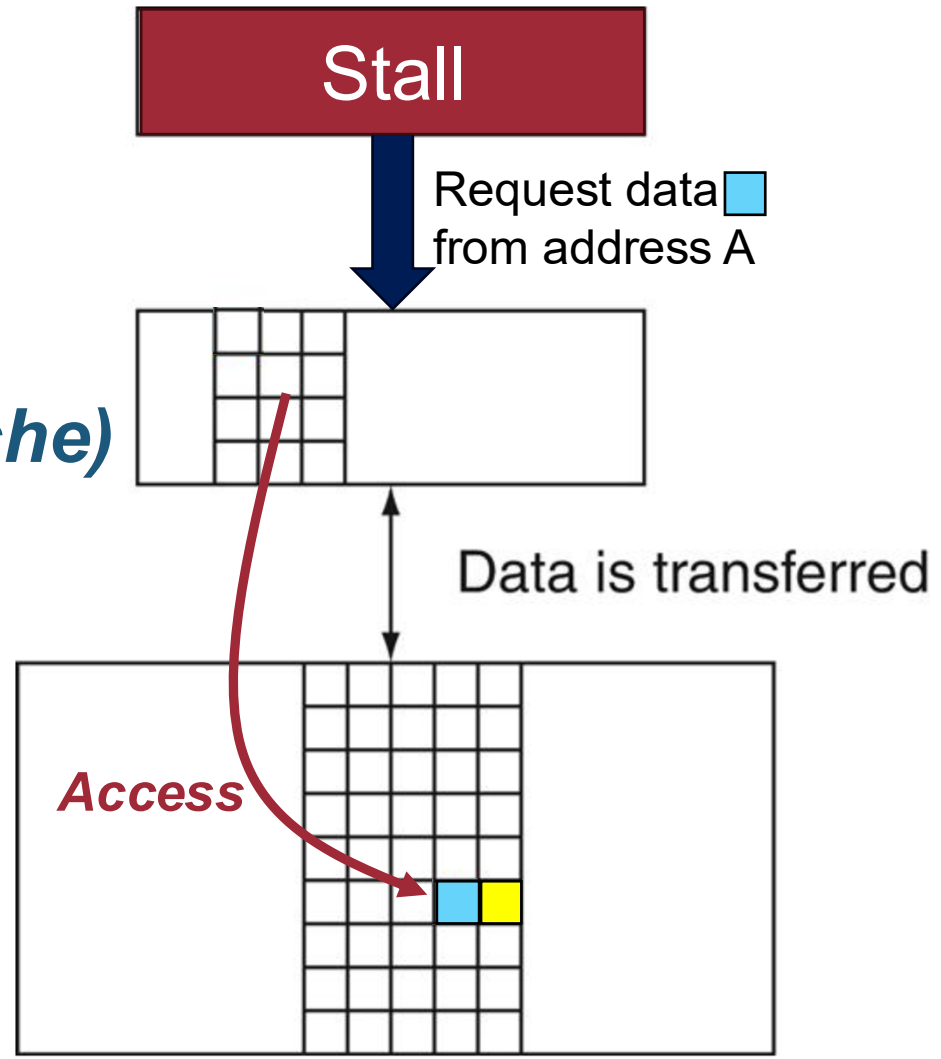
Locality Example: 1st Access

- Assumption: two levels (*upper*, *lower*)
  - Each pair of levels in the memory hierarchy can be thought of as having an *upper* and *lower level*



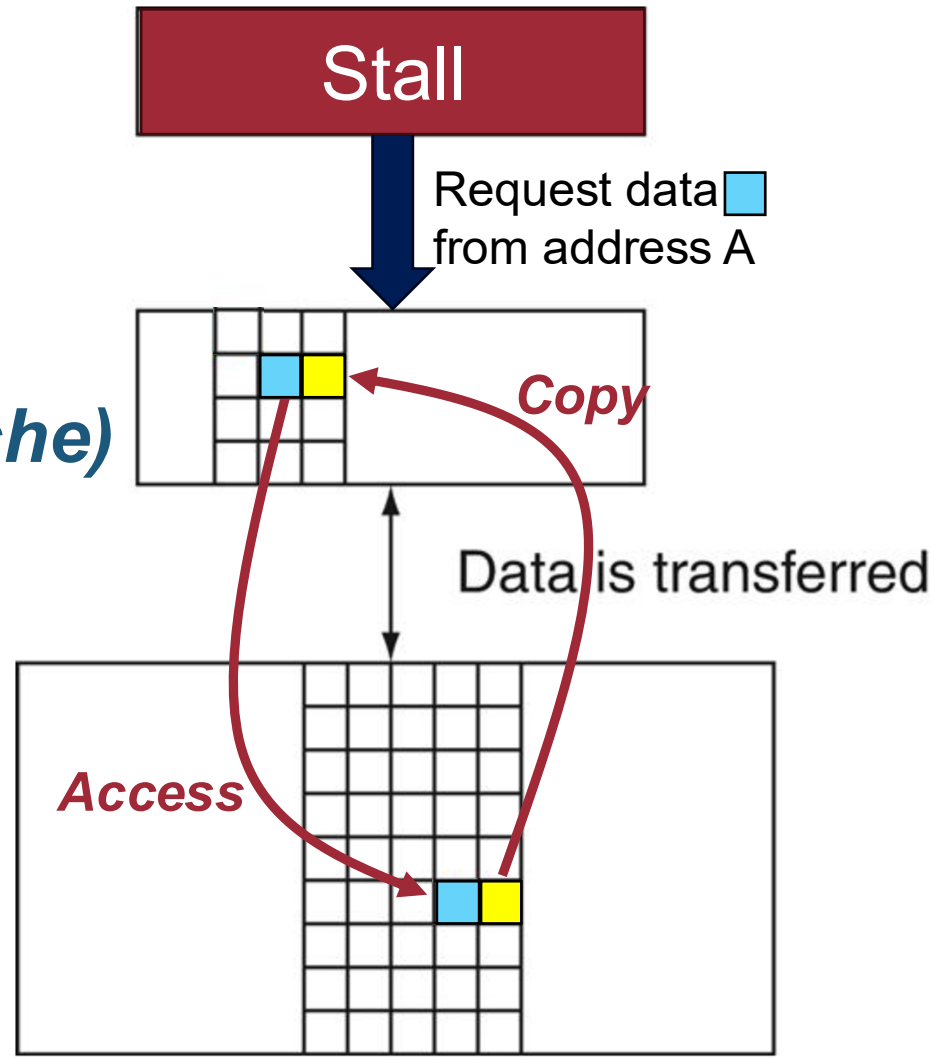
Locality Example: 1st Access

- Assumption: two levels (*upper*, *lower*)
  - Each pair of levels in the memory hierarchy can be thought of as having an *upper* and *lower level*



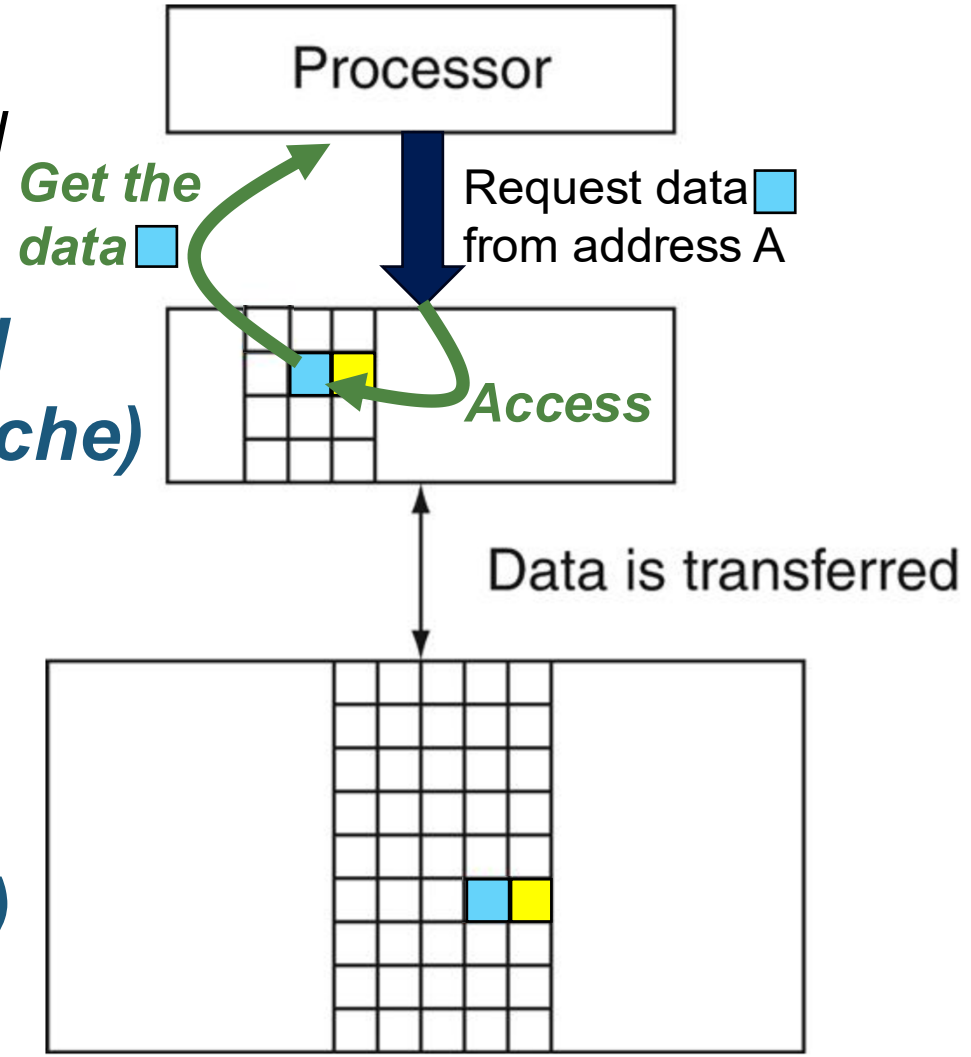
Locality Example: 1st Access

- Assumption: two levels (*upper*, *lower*)
  - Each pair of levels in the memory hierarchy can be thought of as having an *upper* and *lower level*



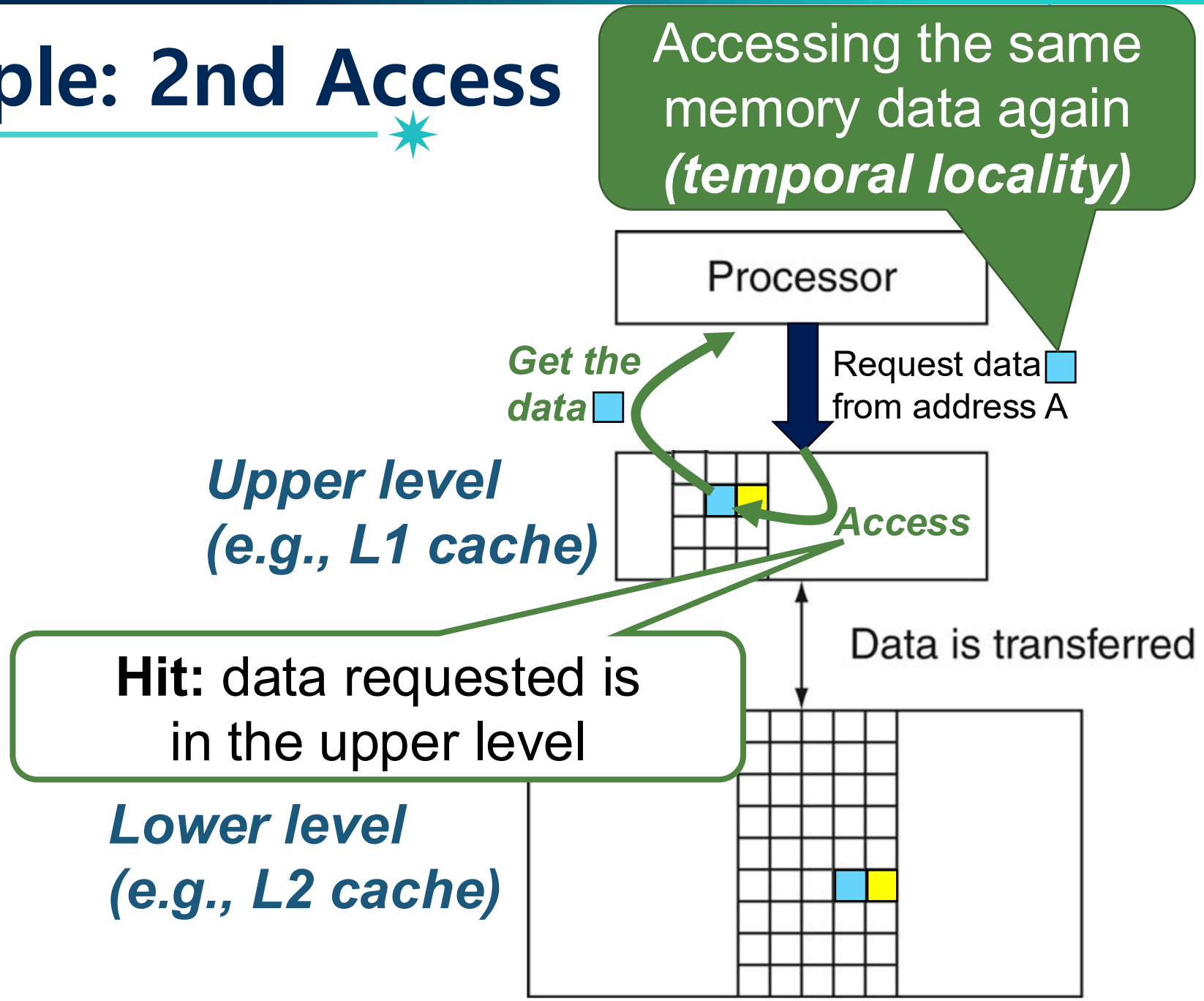
Locality Example: 1st Access

- Assumption: two levels (*upper*, *lower*)
  - Each pair of levels in the memory hierarchy can be thought of as having an *upper* and *lower level*

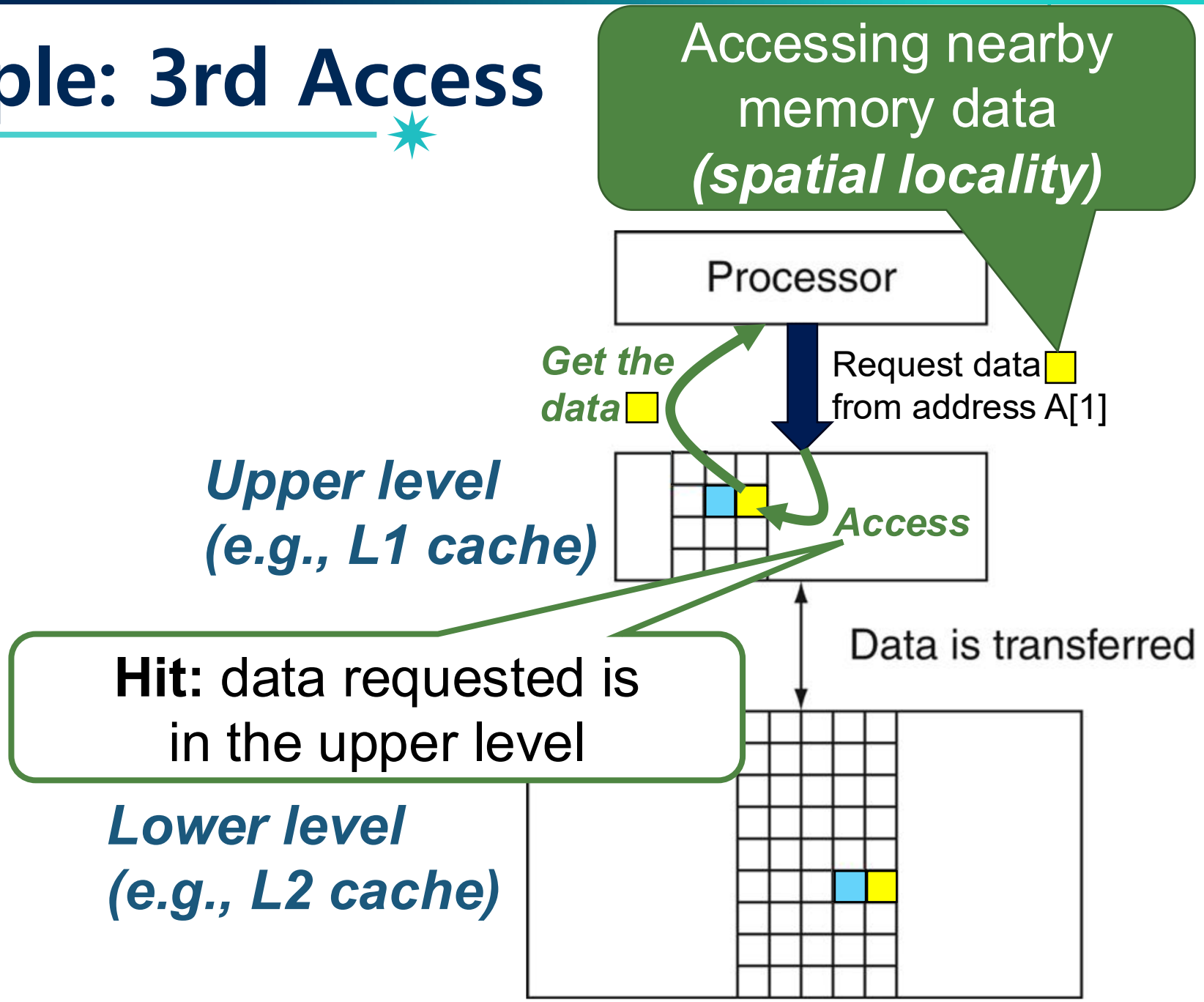


When a miss occurs, performance degrades significantly ☹

Locality Example: 2nd Access



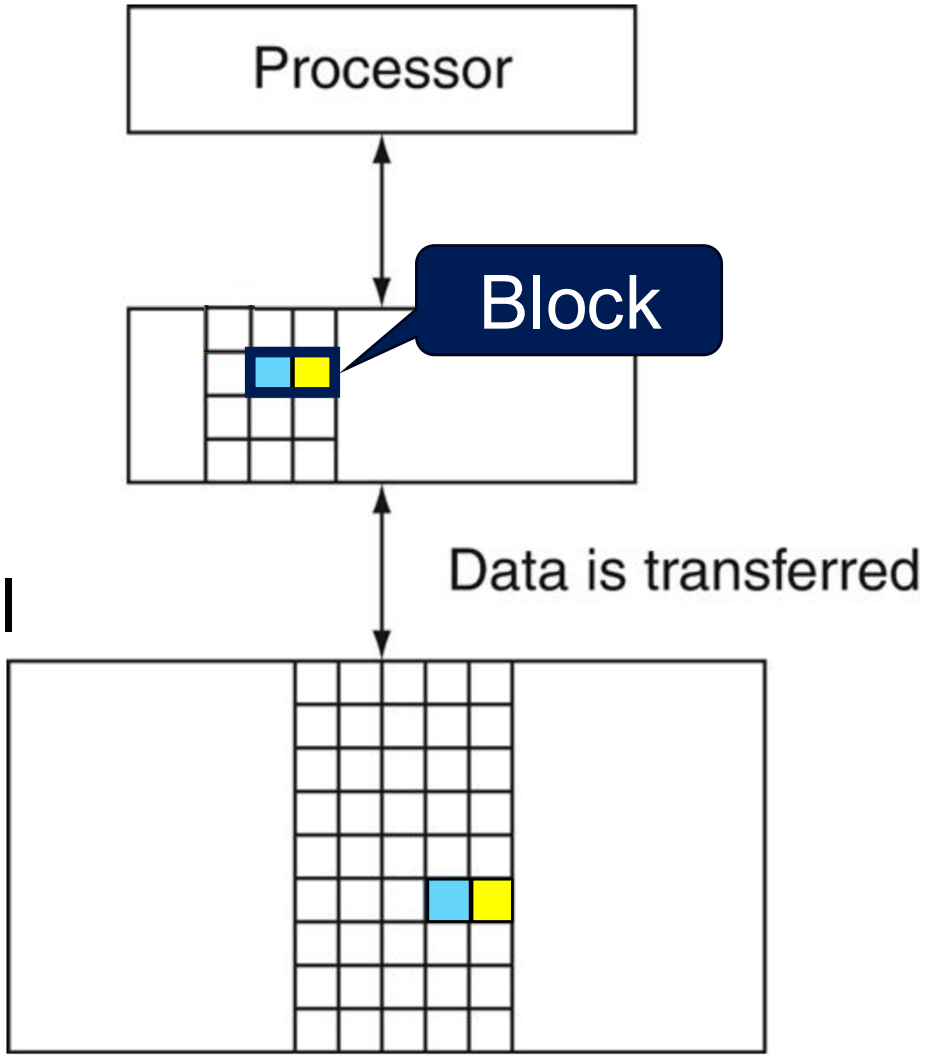
Locality Example: 3rd Access



When a miss occurs, performance degrades significantly ☹
However, once a miss happens, many hits can be expected due to ***locality***, leading to performance improvement!

Terms

- Block (a.k.a., line): unit of copying
  - Several words in cache memory
- **Hit**: data requested is in the upper level
- **Miss**: data requested is not in the upper level
  - Block copied from lower level

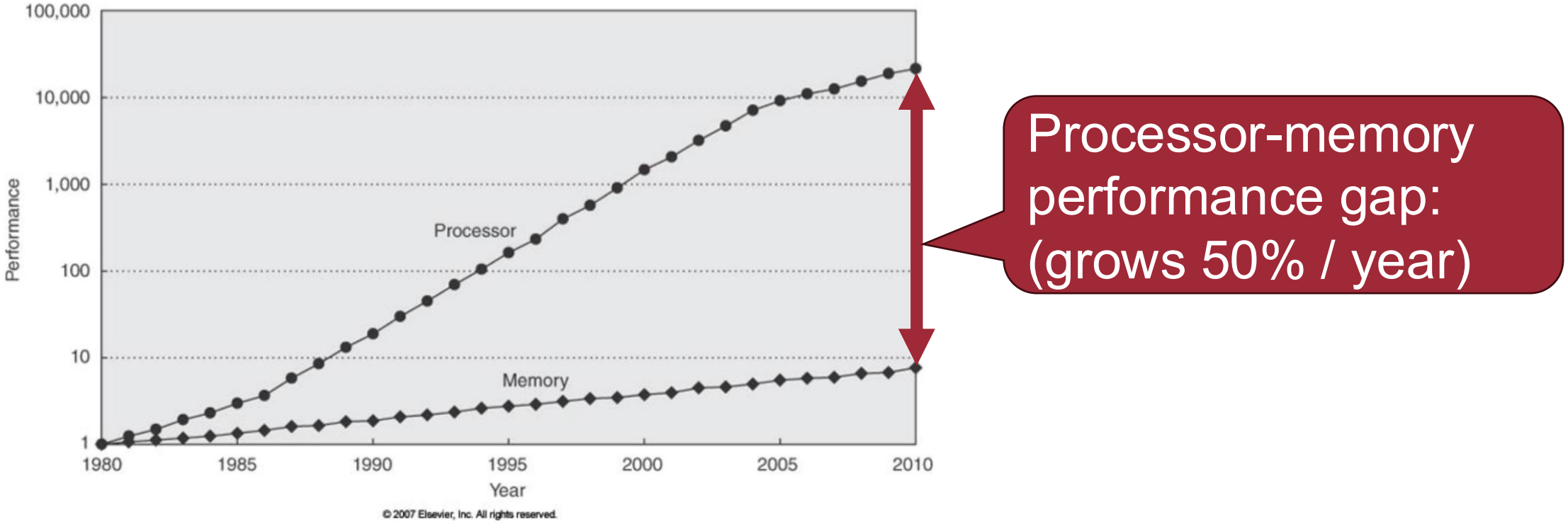


Memory Hierarchy

- Taking advantage of ***locality***
  - Store everything on disk
  - Copy recently accessed (and nearby) items from disk to smaller DRAM memory
  - Copy more recently accessed (and nearby) items from DRAM to smaller SRAM memory

Why Memory Hierarchy Crucial?

- Processor speed improvement: 60%/ year (2x/1.5 year)
- Memory latency improvement: 9% / year (2x/10 years)



- Still it takes around 100+ CPU cycles for DRAM access
  - Hierarchy (계층구조) is the key!

Question?