

Memory Hierarchy (1): Inside out of our memory hierarchy

Hung-Wei Tseng



Disney·PIXAR INSIDE OUT

GET DISNEY+

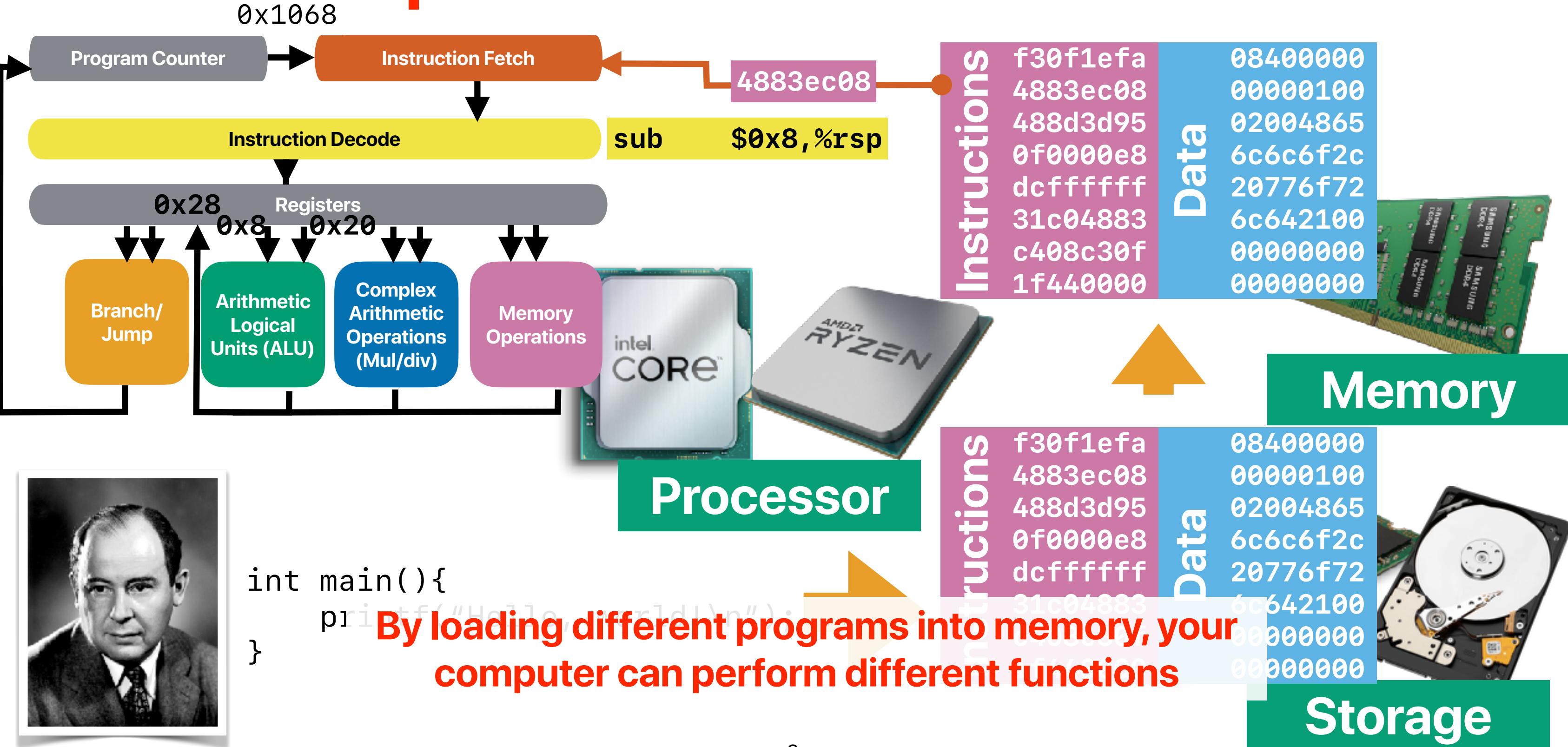
► TRAILER

PG 2015 • 1h 35m • Coming of age, Family, Animation

When 11-year-old Riley moves to a new city, her Emotions team up to help her through the transition. Joy, Fear, Anger, Disgust and Sadness work together, but when Joy and Sadness get lost, they must journey through unfamiliar places to get back home.



Recap: von Neumann architecture



Recap: Amdahl's Law

- Definition of "Speedup of Y over X" or say Y is n times faster than X:

$$speedup_{Y_over_X} = n = \frac{Execution\ Time_X}{Execution\ Time_Y}$$

- Amdahl's Law — $Speedup_{enhanced}(f, s) = \frac{1}{(1-f) + \frac{f}{s}}$

- Corollary 1 — each optimization has an upper bound $Speedup_{max}(f, \infty) = \frac{1}{(1-f)}$

- Corollary 2 — make the common case (the most time consuming case) fast!

$$Speedup_{max}(f_1, \infty) = \frac{1}{(1-f_1)}$$

$$Speedup_{max}(f_2, \infty) = \frac{1}{(1-f_2)}$$

$$Speedup_{max}(f_3, \infty) = \frac{1}{(1-f_3)}$$

$$Speedup_{max}(f_4, \infty) = \frac{1}{(1-f_4)}$$

- Corollary 3 — Optimization has a moving target
 - Corollary 4 — Exploiting more parallelism from a program is the key to performance gain in modern architectures

$$Speedup_{parallel}(f_{parallelizable}, \infty) = \frac{1}{(1-f_{parallelizable})}$$

- Corollary 5 — Single-core performance still matters

$$Speedup_{parallel}(f_{parallelizable}, \infty) = \frac{1}{(1-f_{parallelizable})}$$

- Corollary 6 — Don't hurt the non-common case too much

$$Speedup_{enhanced}(f, s, r) = \frac{1}{(1-f) + perf(r) + \frac{f}{s}}$$

What percentage of “English words” from the dictionary do you think someone has to know for their everyday life?

federalreserve.gov/releases/z1/dataviz/dfa/di...

crn.ccm/news/security/18821726/microsofts-ceo-80-20-rule-applies-to-bugs-not-just-features

Board of Governors of the Federal Reserve System

Sections

Wealth by wealth percentile group

The chart displays the total wealth in Trillions of Dollars over time, broken down into five percentile groups. The legend indicates:

- Top 0.1%: \$20.66 T
- 99-99.9%: \$25.52 T
- 90-99%: \$55.44 T
- 50-90%: \$46.28 T
- Bottom 50%: \$3.78 T

Total wealth: \$151.68 T

Percentile Group	Value (Trillions of Dollars)
Top 0.1%	\$20.66 T
99-99.9%	\$25.52 T
90-99%	\$55.44 T
50-90%	\$46.28 T
Bottom 50%	\$3.78 T
Total	\$151.68 T

Microsoft's CEO: 80-20 Rule Applies To Bugs, Not Just Features

BY PAULA ROONEY ►
OCTOBER 03, 2002, 03:56 PM EDT

Microsoft

One common adage in the IT industry is that 80 percent of all end users generally use only 20 percent of a software application's features.

In recent months, Microsoft has learned that 80 percent of the errors and crashes in Windows and Office are caused by 20 percent of the entire pool of bugs detected, and that more than 50 percent of the headaches derive from a mere 1 percent of all flawed code.

In an e-mail update sent out broadly to enterprise customers on Oct. 2, Microsoft CEO Steve Ballmer highlighted initial progress being made on the company's Trustworthy Computing initiative, an effort to be concluded by the end of next January to improve its reputation in reliability and security arenas. For one thing, there will be faster bug-fixing as a result of an error-reporting facility embedded in Office and Windows. And that error-reporting tool will be part of the forthcoming Windows Server 2003.

80% of users use only 20% of features

Top 10% own 67% of the wealth in the U.S.

https://en.wikipedia.org/wiki/Pareto_principle

 AI Overview[Learn more](#) 

Native English speakers use around 1,200–3,000 words in daily life, which account for about 80% of their communication. These words include common verbs like "eat," "sleep," "work," "talk," and "walk," as well as pronouns like "I," "you," "he," and "she," and basic nouns like "house," "car," "food," and "water". The most commonly used words in English are "the," "be," and "to". 

 Eton Institute · LinkedIn · 1y 

How many words are in the English language (95/5 RULE)?

How Many Words Do Native Speakers Use In Daily Life? Native speakers of...

 GeeksforGeeks 

Daily Used English Words: List of 100+ Most Common Words

May 6, 2024 — Words used in daily life often include common verbs like...

Show more 

With 2,500 to 3,000 words, you can understand 90% of everyday English conversations, English newspaper and magazine articles, and English used in the workplace.



ef.edu

<http://www.ef.edu/english-vocabulary/1000-3000-w...> 

3000 most common words in English | EF English Charts

 AI Overview

The number of words in an English dictionary depends on the dictionary, whether it includes obsolete words, combinations, and phrases:

Oxford English Dictionary

The second edition of this 20-volume dictionary includes 171,476 words in current use, 47,156 obsolete words, and around 9,500 derivative words. It also includes 169,000 combinations and derivatives, and over 600,000 word forms. The dictionary is updated annually to include new and new meanings for existing words.

Webster's Third New International Dictionary

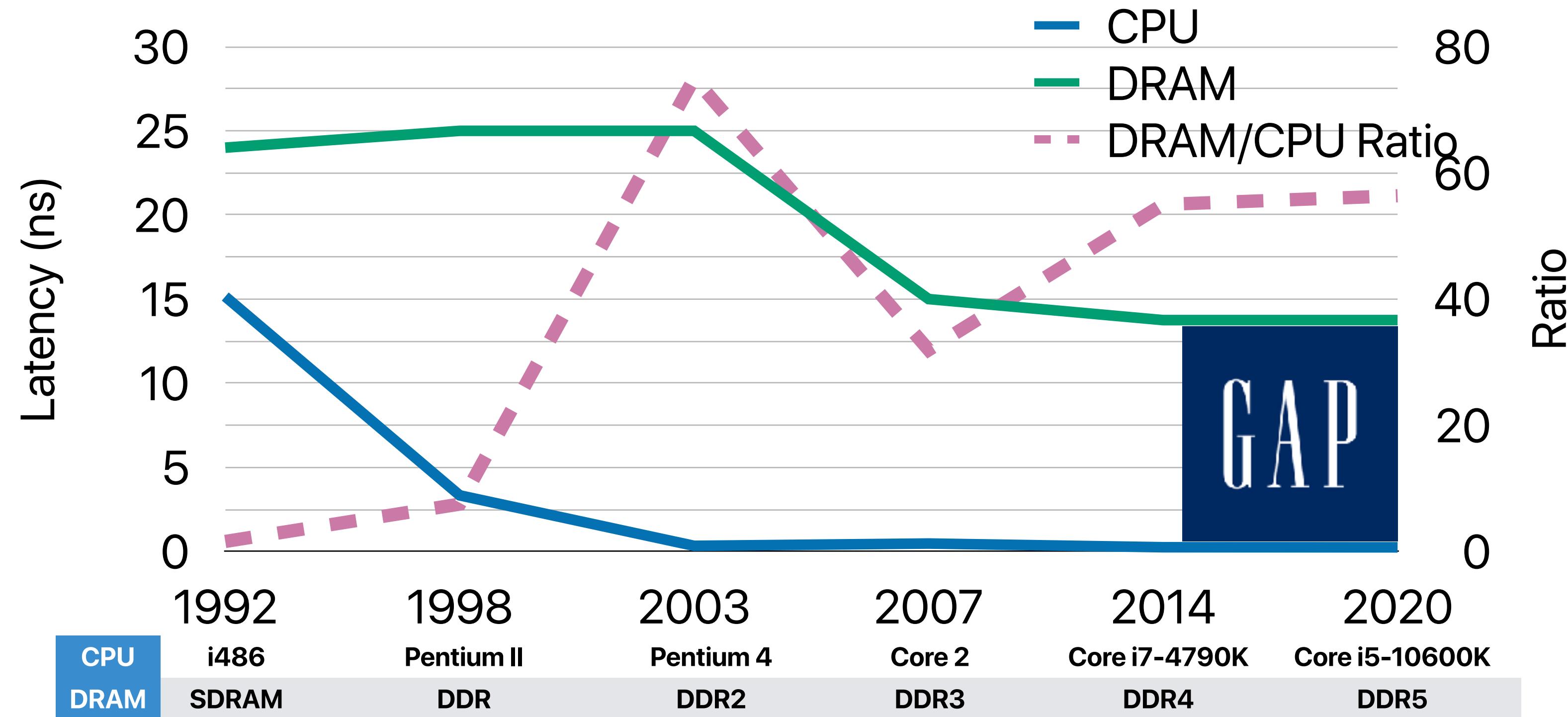
This dictionary, along with its 1993 Addenda Section, includes around 470,000 entries. 

You only need to know 2% English words
to understand 90% of conversations

Outline

- The memory-wall problem
- The “predictable” code behavior
- Designing a cache that captures the predictability

The “latency” gap between CPU and DRAM





The impact of “slow” memory

- Assume that we have a processor running @ 4 GHz and a program with 20% of load/store instructions. If the instruction has no memory access and the processor already fetches the instruction, the CPI is just 1. Now, consider we have DDR5. The program is well-optimized so precharge is never necessary — the memory access latency is 13.75 ns. What's the average CPI (pick the closest one)?

- A. 5
- B. 10
- C. 30
- D. 50
- E. 70

The impact of “slow” memory

- Assume that we have a processor running @ 4 GHz and a program with 20% of load/store instructions. If the instruction has no memory access and the processor already fetches the instruction, the CPI is just 1. Now, consider we have DDR5. The program is well-optimized so precharge is never necessary — the memory access latency is 13.75 ns. What's the average CPI (pick the closest one)?

A. 5

$$CPU \text{ cycle time} = \frac{1}{4 \times 10^9} = 0.25 \text{ ns}$$

B. 10

$$Each \text{ DRAM access} = \frac{13.75}{0.25} = 55 \text{ cycles}$$

C. 30

$$CPI_{average} = 1 + 100\% \times 55 + 20\% \times 55 = 67 \text{ cycles}$$

D. 50

Don't forget, instructions are also from “memory”

E. 70

$$\frac{66}{67} = 98.5 \% \text{ of time, we're dealing with memory accesses!}$$

20% is under-estimating ...

Instruction class	MIPS examples	HLL correspondence	Frequency	
			Integer	Ft. pt.
Arithmetic	add, sub, addi	Operations in assignment statements	16%	48%
Data transfer	lw, sw, lb, lbu, lh, lhu, sb, lui	References to data structures, such as arrays	35%	36%
Logical	and, or, nor, andi, ori, sll, srl	Operations in assignment statements	12%	4%
Conditional branch	beq, bne,slt, slti, sltiu	If statements and loops	34%	8%
Jump	j, jr, jal	Procedure calls, returns, and case/switch statements	2%	0%

FIGURE 2.48 MIPS instruction classes, examples, correspondence to high-level program language constructs, and percentage of MIPS instructions executed by category for the average integer and floating point SPEC CPU2006 benchmarks.

Figure 3.24 in Chapter 3 shows average percentage of the individual MIPS instructions executed.

Recap: Speedup and Amdahl's Law?

- Definition of “Speedup of Y over X” or say Y is n times faster than X: $speedup_{Y_over_X} = n = \frac{Execution\ Time_X}{Execution\ Time_Y}$

- Amdahl's Law — $Speedup_{enhanced}(f, s) = \frac{1}{(1-f) + \frac{f}{s}}$ $Speedup_{max}(f, \infty) = \frac{1}{(1-f)}$
- Corollary 1 — each optimization has an upper bound

- Corollary 2 — make the common case (the most time consuming case) fast!

- Corollary 3: Optimization has a moving target

- Corollary 4: Exploiting more parallelism from a program is the key to performance gain in modern architectures

$$Speedup_{parallel}(f_{parallelizable}, \infty) = \frac{1}{(1 - f_{parallelizable})}$$

- Corollary 5: Single-core performance still matters

$$Speedup_{parallel}(f_{parallelizable}, \infty) = \frac{1}{(1 - f_{parallelizable})}$$

Take-aways: inside out our memory hierarchy

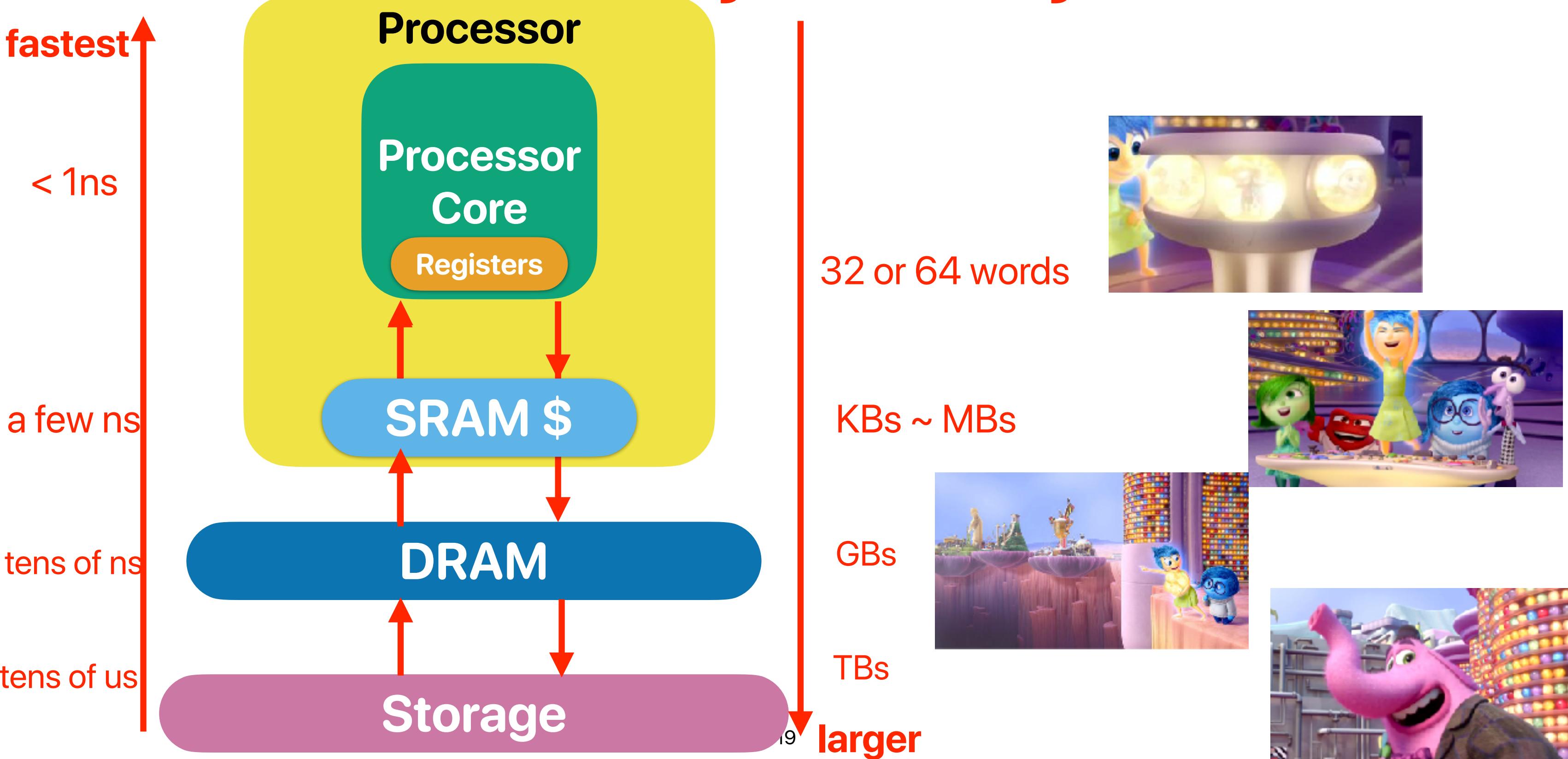
- Memory access time is the most critical performance problem
 - One memory operation is as expensive as 50 arithmetic operations
 - Processor has to fetch instructions from memory
 - We have an average of 33% of data memory access instructions!

Alternatives?

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

Fast, but expensive \$\$\$

Memory Hierarchy





How can “memory hierarchy” help in performance?

- Assume that we have a processor running @ 4 GHz and a program with 20% of load/store instructions. If the instruction has no memory access, the CPI is just 1. Now, in addition to we DDR5, whose latency 13.75 ns, we also got an SRAM cache with latency of just at 0.5 ns and can capture **90%** of the desired data/instructions. what's the average CPI (pick the closest one)?

- A. 5
- B. 10
- C. 30
- D. 50
- E. 70



How can “memory hierarchy” help in performance?

- Assume that we have a processor running @ 4 GHz and a program with 20% of load/store instructions. If the instruction has no memory access, the CPI is just 1. Now, in addition to we DDR5, whose latency 13.75 ns, we also got an SRAM cache with latency of just at 0.5 ns and can capture **90%** of the desired data/instructions. what's the average CPI (pick the closest one)?

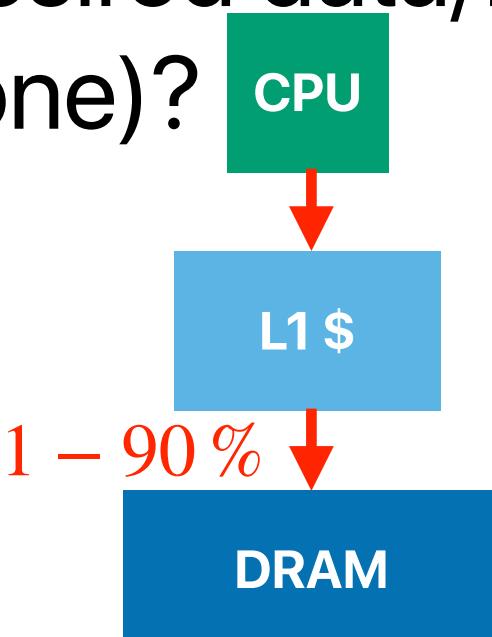
A. 5

B. 10

C. 30

D. 50

E. 70



$$CPU \text{ cycle time} = \frac{1}{4 \times 10^9} = 0.25\text{ns}$$

$$\text{Each \$ access} = \frac{0.5}{0.25} = 2 \text{ cycles}$$

$$\text{Each DRAM access} = \frac{13.75}{0.25} = 55 \text{ cycles}$$

$$CPI_{average} = 1 + 100\% \times [2 + (1 - 90\%) \times 55] + 20\% \times [2 + (1 - 90\%) \times 55] = 10 \text{ cycles}$$

L1? L2? L3?

CPU-Z - ID : vfljg

CPU Mainboard Memory SPD Graphics Bench About

Processor

Name	AMD Ryzen 7 7700X		
Code Name	Raphael	Max TDP	105 W
Package	Socket AM5 (LGA1718)		
Technology	5 nm	Core Voltage	1.288 V

Specification

AMD Ryzen 7 7700X 8-Core Processor			
Family	F	Model	1
Ext. Family	19	Ext. Model	61
Stepping	2	Revision	RPL-B2

Instructions

MMX(+), SSE, SSE2, SSE3, SSSE3, SSE4.1, SSE4.2, SSE4A x86-64, AMD-V, AES, AVX, AVX2, AVX512, FMA3, SHA			
--	--	--	--

Clocks (Core #0)

Core Speed	5188.99 MHz
Multiplier	x 52.0 (4 - 55.5)
Bus Speed	99.79 MHz
Rated FSB	

Cache

L1 Data	8 x 32 KB
L1 Inst.	8 x 32 KB
Level 2	8 x 1024 KB
Level 3	32 MBytes

Selection Socket #1 Cores 8 Threads 16

CPU-Z Ver. 2.09.0.x64 Tools Validate Close

CPU-Z - ID : pk15b

CPU Mainboard Memory SPD Graphics Bench About

Processor

Name	Intel Core i7 14700K		
Code Name	Raptor Lake	Max TDP	125 W
Package	Socket 1700 LGA		
Technology	10 nm	Core Voltage	1.412 V

Specification

Intel(R) Core(TM) i7-14700K			
Family	6	Model	7
Ext. Family	6	Ext. Model	B7
Stepping	1	Revision	B0

Instructions

MMX, SSE, SSE2, SSE3, SSSE3, SSE4.1, SSE4.2, EM64T VT-x, AES, AVX, AVX2, FMA3, SHA			
--	--	--	--

Clocks (Core #0)

Core Speed	5287.07 MHz
Multiplier	x 53.0 (8 - 55)
Bus Speed	99.76 MHz
Rated FSB	

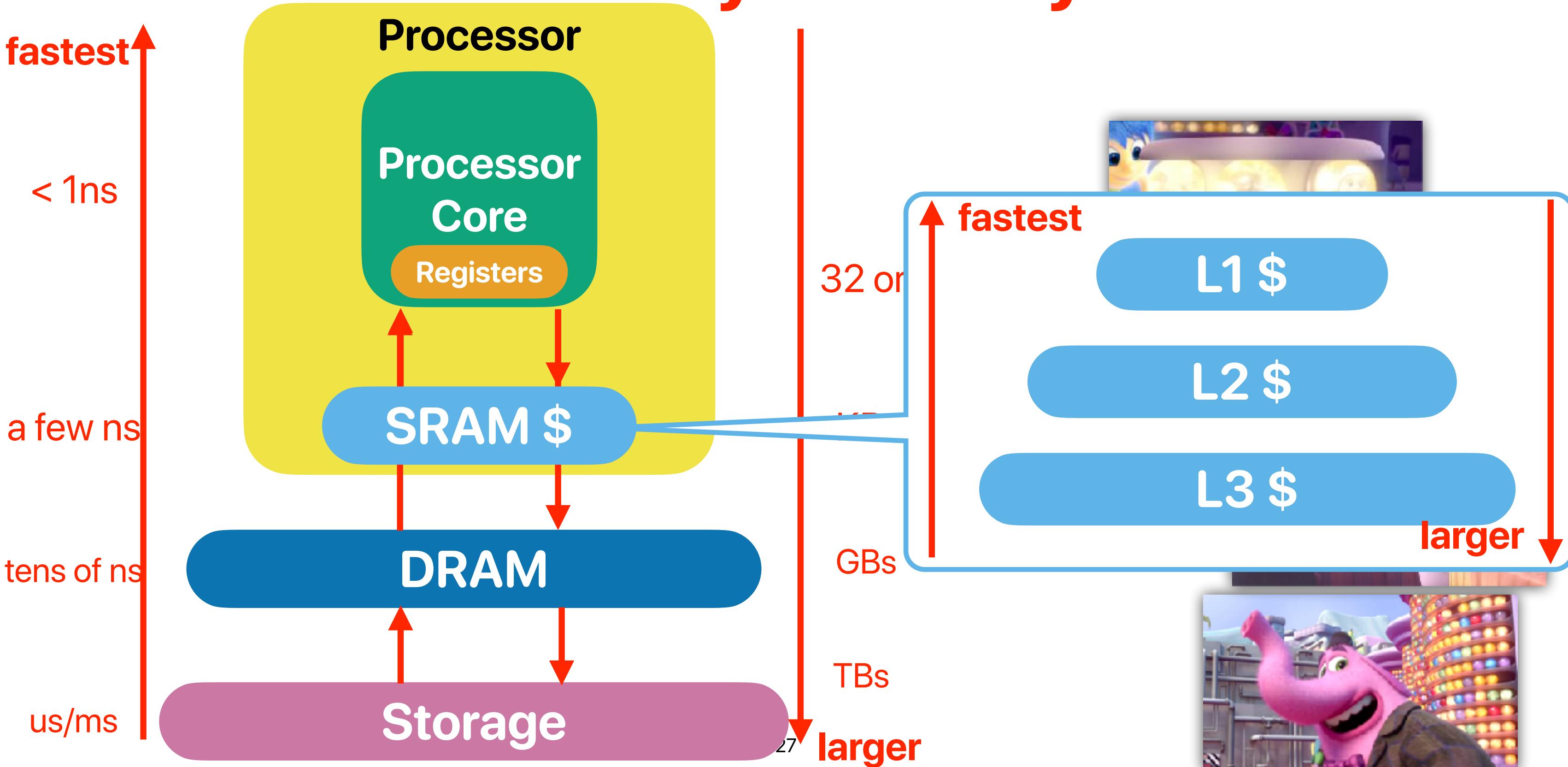
Cache

L1 Data	8 x 48 KB + 12 x 32 KB
L1 Inst.	8 x 32 KB + 12 x 64 KB
Level 2	8 x 2 MB + 3 x 4 MB
Level 3	33 MBytes

Selection Socket #1 Cores 8 + 12 Threads 28

CPU-Z Ver. 2.08.0.x64 Tools Validate Close

Memory Hierarchy





How can a deeper memory hierarchy help in performance?

- Assume that we have a processor running @ 4 GHz and a program with 20% of load/store instructions. If the instruction has no memory access, the CPI is just 1. Now, in addition to we DDR5, whose latency 13.75 ns, we also got a 2-level SRAM caches with
 - it's 1st-level one at latency of 0.5ns and can capture 90% of the desired data/instructions.
 - the 2nd-level at latency of 5 ns and can capture 60% of the desired data/instructions

What's the average CPI (pick the closest one)?

- A. 5
- B. 10
- C. 30
- D. 50
- E. 70

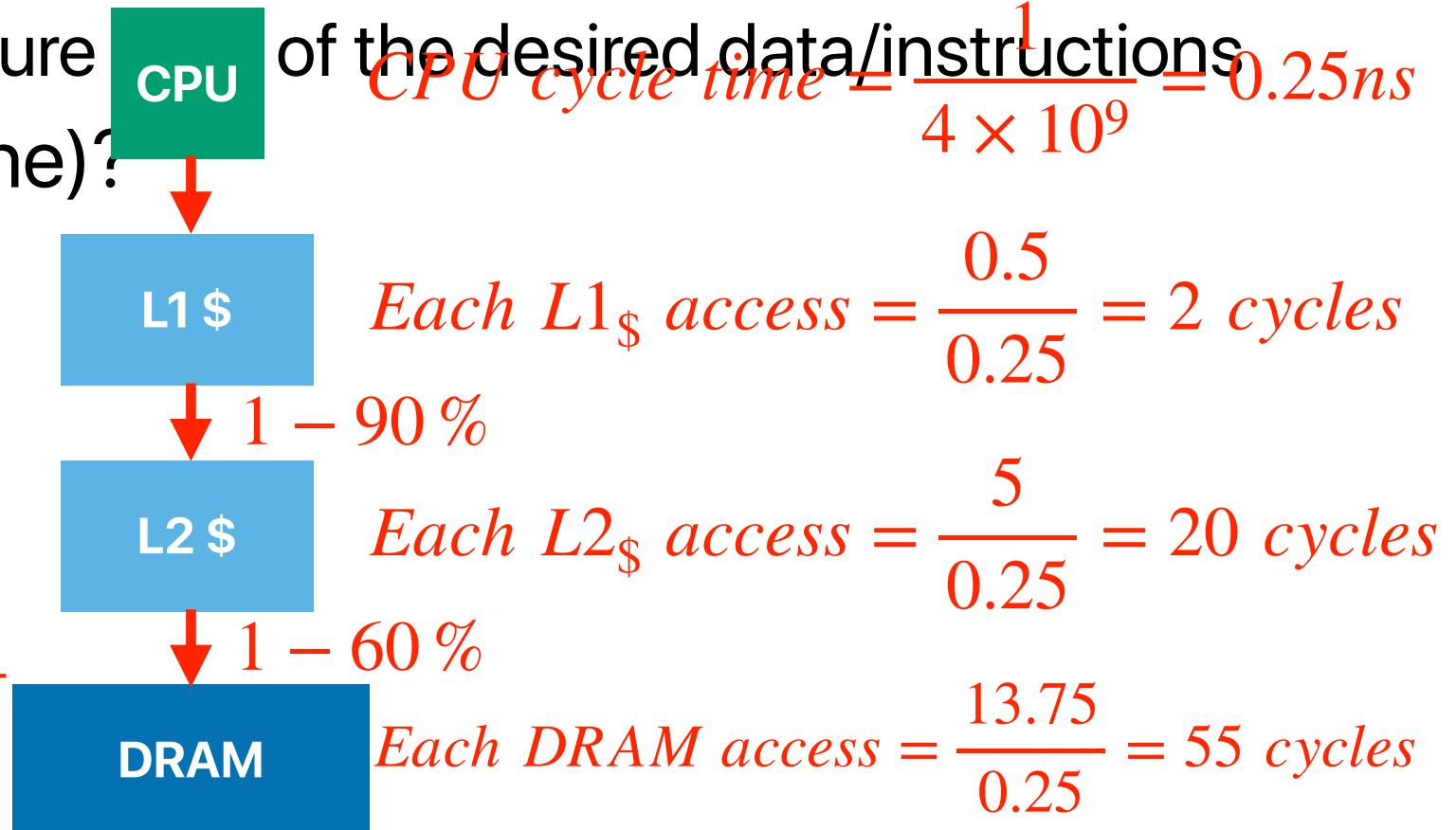


How can a deeper memory hierarchy help in performance?

- Assume that we have a processor running @ 4 GHz and a program with 20% of load/store instructions. If the instruction has no memory access, the CPI is just 1. Now, in addition to the DDR5, whose latency is 13.75 ns, we also got 2 levels SRAM caches where
 - the 1st-level one at latency of 0.5ns and can capture 90% of the desired data/instructions.
 - the 2nd-level at latency of 5 ns and can capture $\frac{1}{CPU\ cycle\ time} = \frac{1}{4 \times 10^9} = 0.25\ ns$ of the desired data/instructions.
- What's the average CPI (pick the closest one)?

- A. 5
- B. 10
- C. 30
- D. 50

$$CPI_{average} = 100\% \times [2 + (1 - 90\%) \times (20 + (1 - 60\%) \times 55)] + 20\% \times [2 + (1 - 90\%) \times (20 + (1 - 60\%) \times 55)] = 8.44\ cycles$$



L1? L2? L3?

Can we really “predict” upcoming data accurately (e.g., 90%) with such small caches?

The image displays two side-by-side CPU-Z software windows. The left window is for an AMD Ryzen 7 7700X (Code Name: Raphael) and the right is for an Intel Core i7 14700K (Code Name: Raptor Lake). Both windows show their respective processor details, including core speed, multiplier, bus speed, and instruction sets. The cache sections are highlighted with red boxes.

Processor Cache (AMD Ryzen 7 7700X):

- L1 Data: 8 x 32 KB
- L1 Inst.: 8 x 32 KB
- Level 2: 8 x 1024 KB
- Level 3: 32 MBytes

Processor Cache (Intel Core i7 14700K):

- L1 Data: 8 x 48 KB + 12 x 32 KB
- L1 Inst.: 8 x 32 KB + 12 x 64 KB
- Level 2: 8 x 2 MB + 3 x 4 MB
- Level 3: 33 MBytes

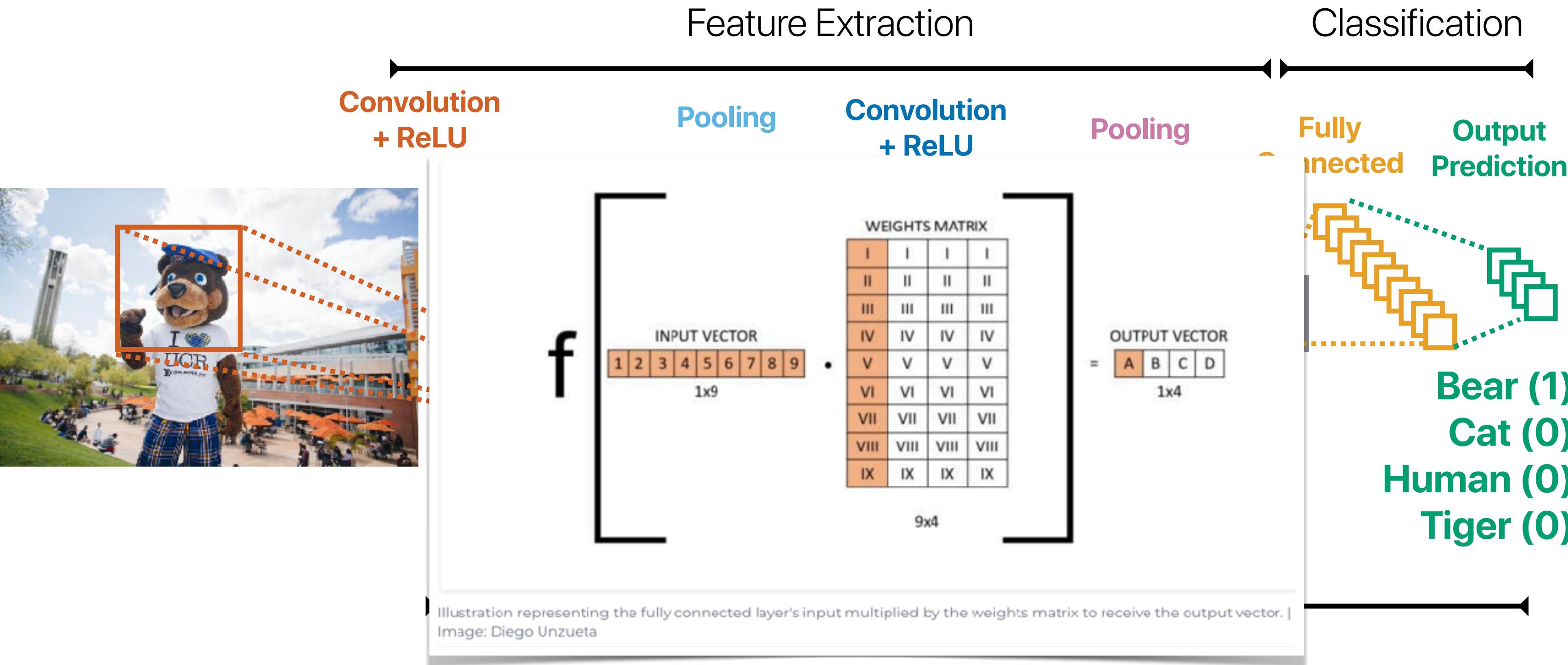
CPU-Z Version: Ver. 2.09.0.x64 (left) and Ver. 2.08.0.x64 (right)

Take-aways: inside out our memory hierarchy

- Memory access time is the most critical performance problem
 - One memory operation is as expensive as 50 arithmetic operations
 - Processor has to fetch instructions from memory
 - We have an average of 33% of data memory access instructions!
- Hierarchical caching with small amount of SRAMs will work if we can efficiently capture data and instructions

The predictability of your code

The Machine Learning Inference Pipeline





Locality of data

- Which description about locality of arrays `matrix` and `vector` in the following code is the **most accurate**?

```
for(uint64_t i = 0; i < m; i++) {  
    result = 0;  
    for(uint64_t j = 0; j < n; j++) {  
        result += matrix[i][j]*vector[j];  
    }  
    output[i] = result;  
}
```



- A. Access of `matrix` has temporal locality, `vector` has spatial locality
- B. Both `matrix` and `vector` have temporal locality, and `vector` also has spatial locality
- C. Access of `matrix` has spatial locality, `vector` has temporal locality
- D. Both `matrix` and `vector` have spatial locality and temporal locality
- E. Both `matrix` and `vector` have spatial locality, and `vector` also has temporal locality

Data locality

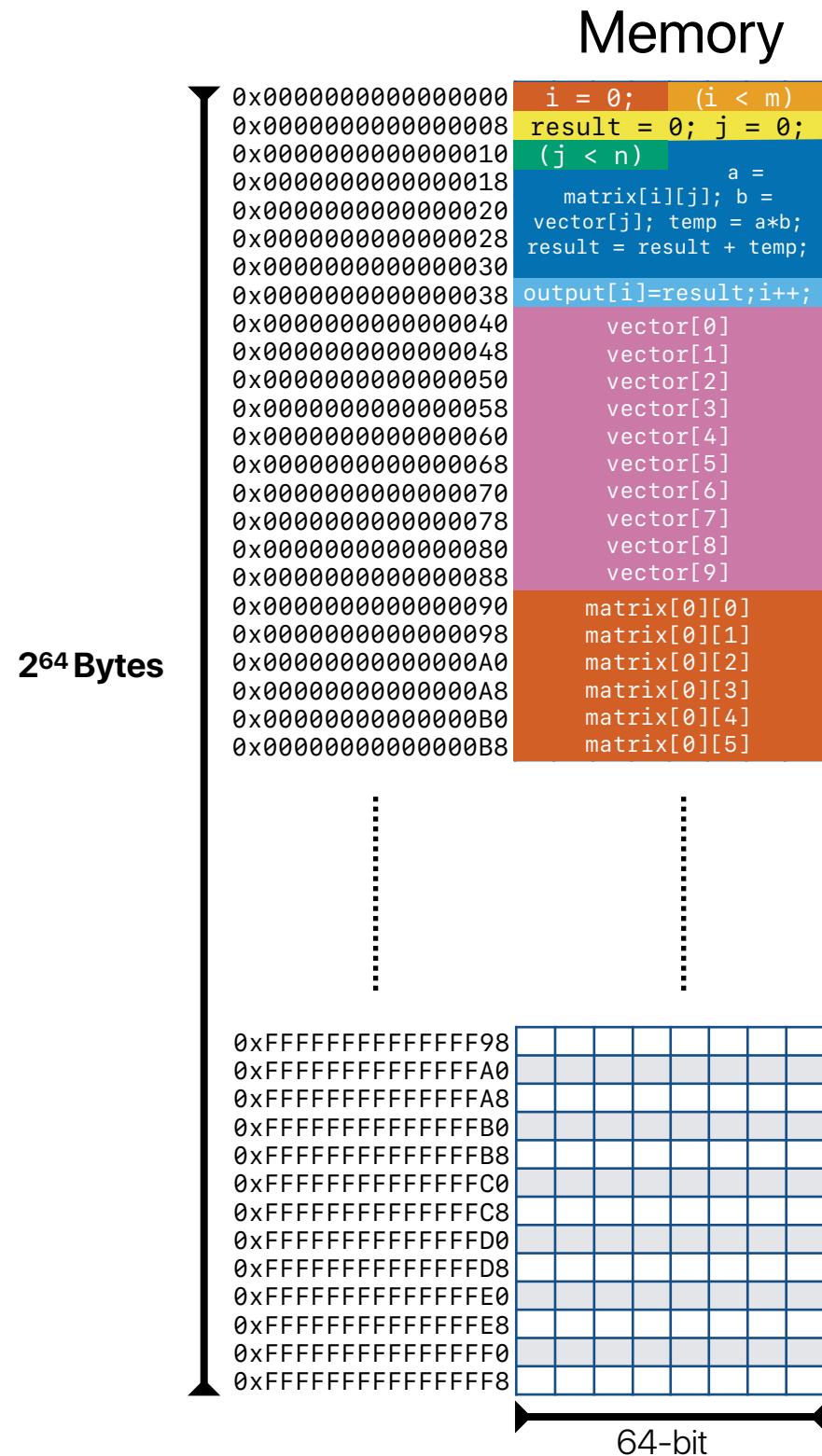
- Which description about locality of arrays `matrix` and `vector` in the following code is the **most accurate**?

```
for(uint64_t i = 0; i < m; i++) {  
    result = 0;  
    for(uint64_t j = 0; j < n; j++) {  
        result += matrix[i][j]*vector[j];  
    }  
    output[i] = result;  
}
```

spatial locality:
`matrix[0][0], matrix[0][1], matrix[0][2], ...`
`vector[0], vector[1], ..., vector[n]`
temporal locality:
`reuse of vector[0], vector[1], ... ,`

- A. Access of `matrix` has temporal locality, `vector` has spatial locality
- B. Both `matrix` and `vector` have temporal locality, and `vector` also has spatial locality
- C. Access of `matrix` has spatial locality, `vector` has temporal locality
- D. Both `matrix` and `vector` have spatial locality and temporal locality
- E. Both `matrix` and `vector` have spatial locality, and `vector` also has temporal locality

Data/instructions in the memory



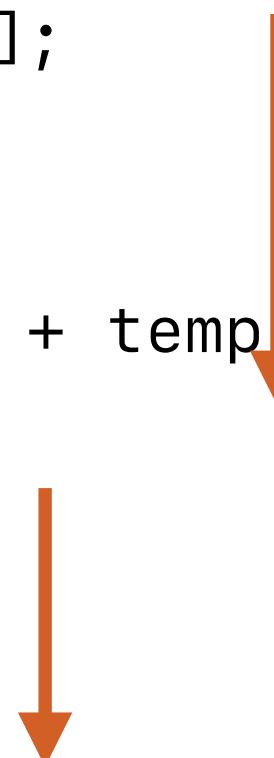
Code also has locality

```
for(uint64_t i = 0; i < m; i++) {  
    result = 0;  
    for(uint64_t j = 0; j < n; j++) {  
        result += matrix[i][j]*vector[j];  
    }  
    output[i] = result;  
}
```

repeat many times —
temporal locality!

```
i = 0;  
while(i < m) {  
    result = 0;  
    j = 0;  
    while(j < n) {  
        a = matrix[i][j];  
        b = vector[j];  
        temp = a*b;  
        result = result + temp;  
    }  
    output[i] = result;  
    i++;
```

keep going to the
next instruction —
spatial locality



Locality

- Spatial locality — application tends to visit nearby stuffs in the memory
 - Code — the current instruction, and then the next instruction

Most of time, your program is just visiting a very small amount of data/instructions within a given window

- Temporal Locality — application revisit the same thing again and again
 - Code — loops, frequently invoked functions
 - Typically tens of static instructions — at most several KBs
 - Data — program can read/write the same data many times (e.g., vectors in matrix-vector product)

Take-aways: inside out our memory hierarchy

- Memory access time is the most critical performance problem
 - One memory operation is as expensive as 50 arithmetic operations
 - Processor has to fetch instructions from memory
 - We have an average of 33% of data memory access instructions!
- Hierarchical caching with small amount of SRAMs will work if we can efficiently capture data and instructions
- Caching is possible! Most of time, we only work on a small amount of data!
 - Spatial locality
 - Temporal locality

Outline

- Designing a cache that captures the predictability
- The A, B, Cs of caches
- Estimating how cache friendly is our code

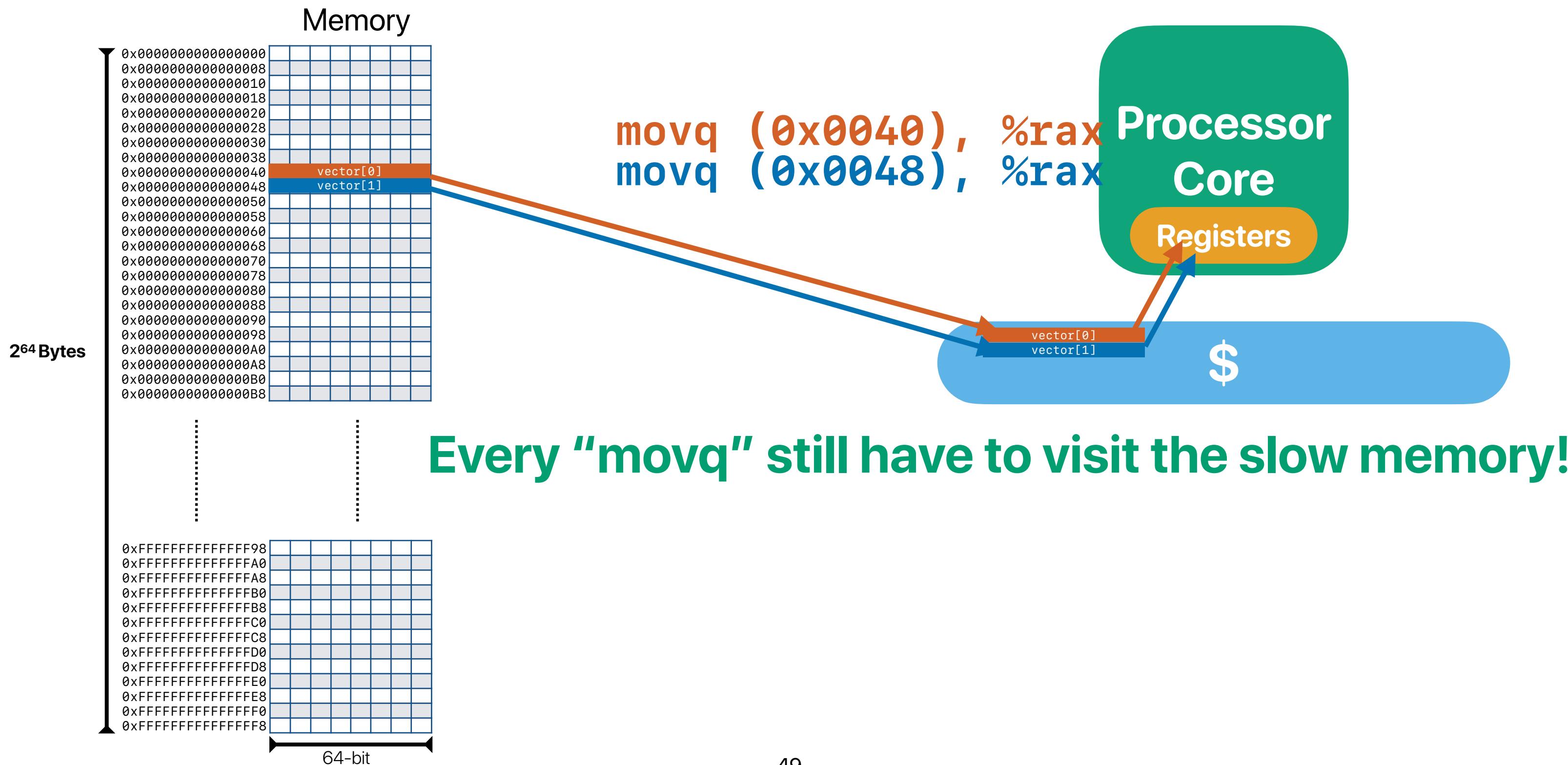
Designing a hardware to exploit locality

- Spatial locality — application tends to visit nearby stuffs in the memory

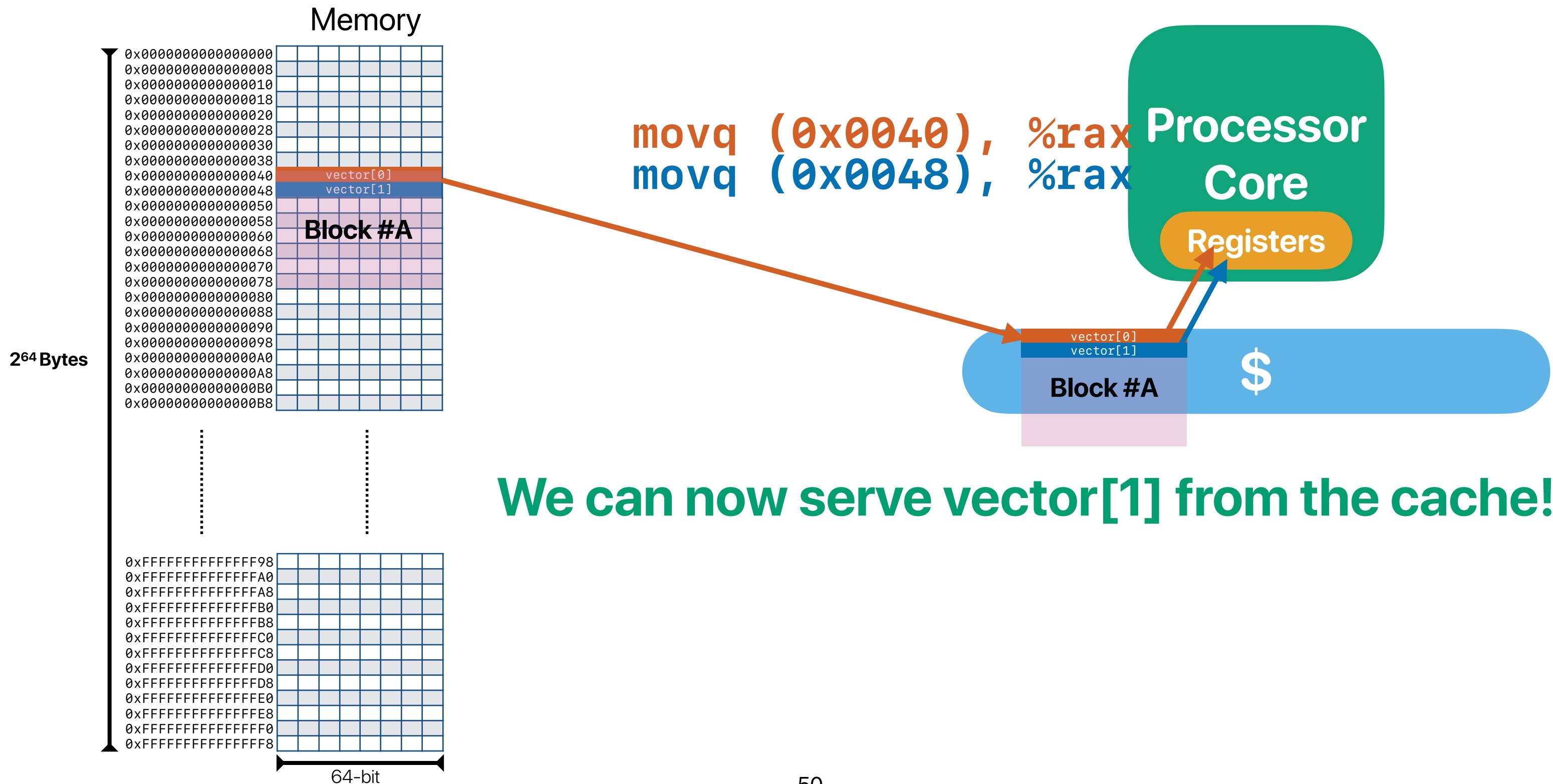
**We need to “cache consecutive memory locations” every time
— the cache should store a “block” of code/data**

- Temporal locality — application revisit the same thing again and again
 - Code — loops, frequently invoked functions
 - Typically tens of static instructions — at most several KBs
 - Data — program can read/write the same data many times (e.g., vectors in matrix-vector product)

If we just cache what has requested



If we can cache a block of data



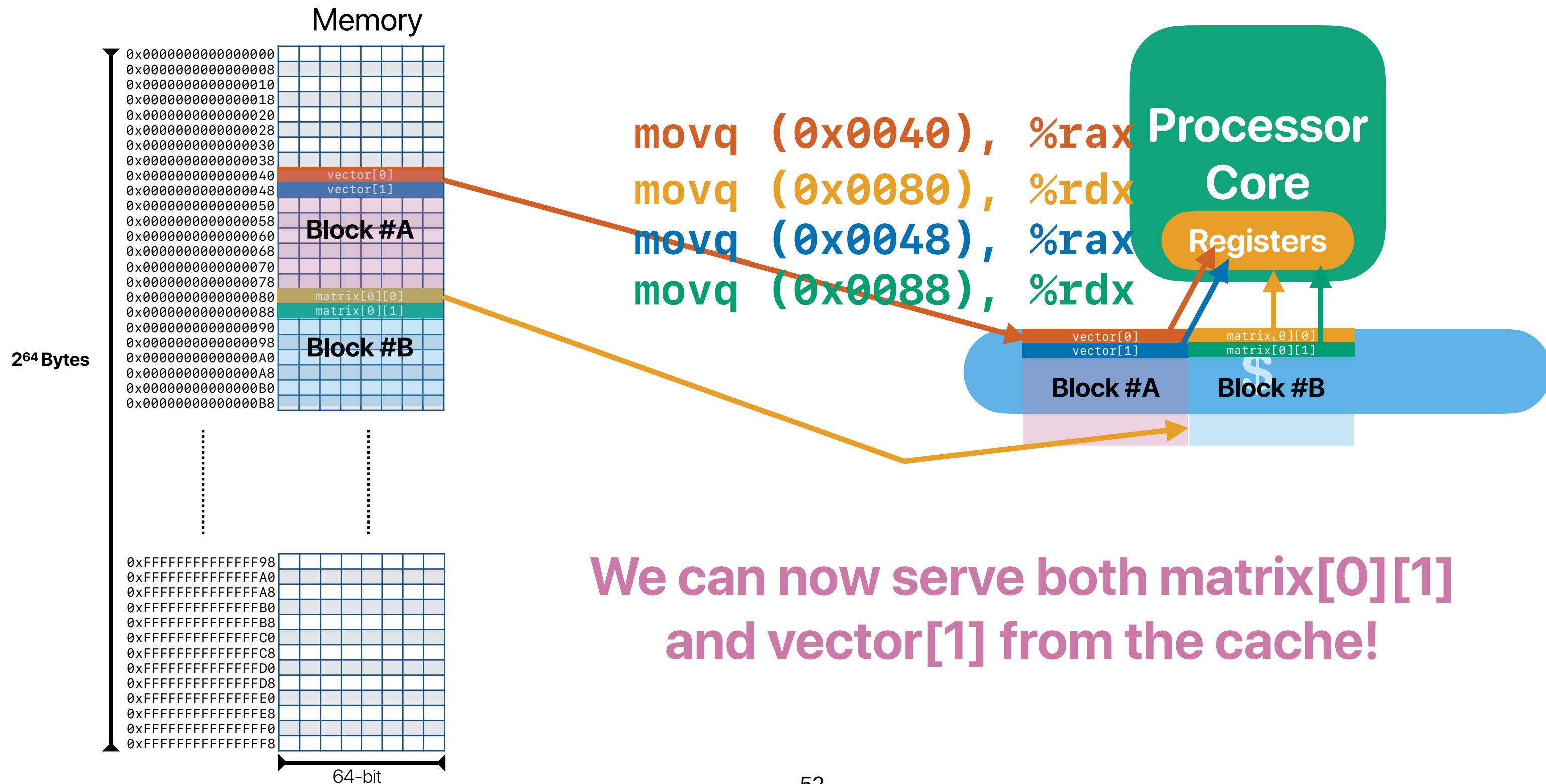
Recap: Locality

- Which description about locality of arrays `matrix` and `vector` in the following code is the **most accurate**?

```
for(uint64_t i = 0; i < m; i++) {  
    result = 0;  
    for(uint64_t j = 0; j < n; j++) {  
        result += matrix[i][j]*vector[j];  
    }  
    output[i] = result;  
}
```

Simply caching one block
isn't enough

We need to cache multiple blocks of data!



Designing a hardware to exploit locality

- Spatial locality — application tends to visit nearby stuffs in the memory

We need to “cache consecutive memory locations” every time
—**the cache should store a “block” of code/data**

- Temporal locality — application revisit the same thing again and again

We need to “cache frequently used memory blocks”

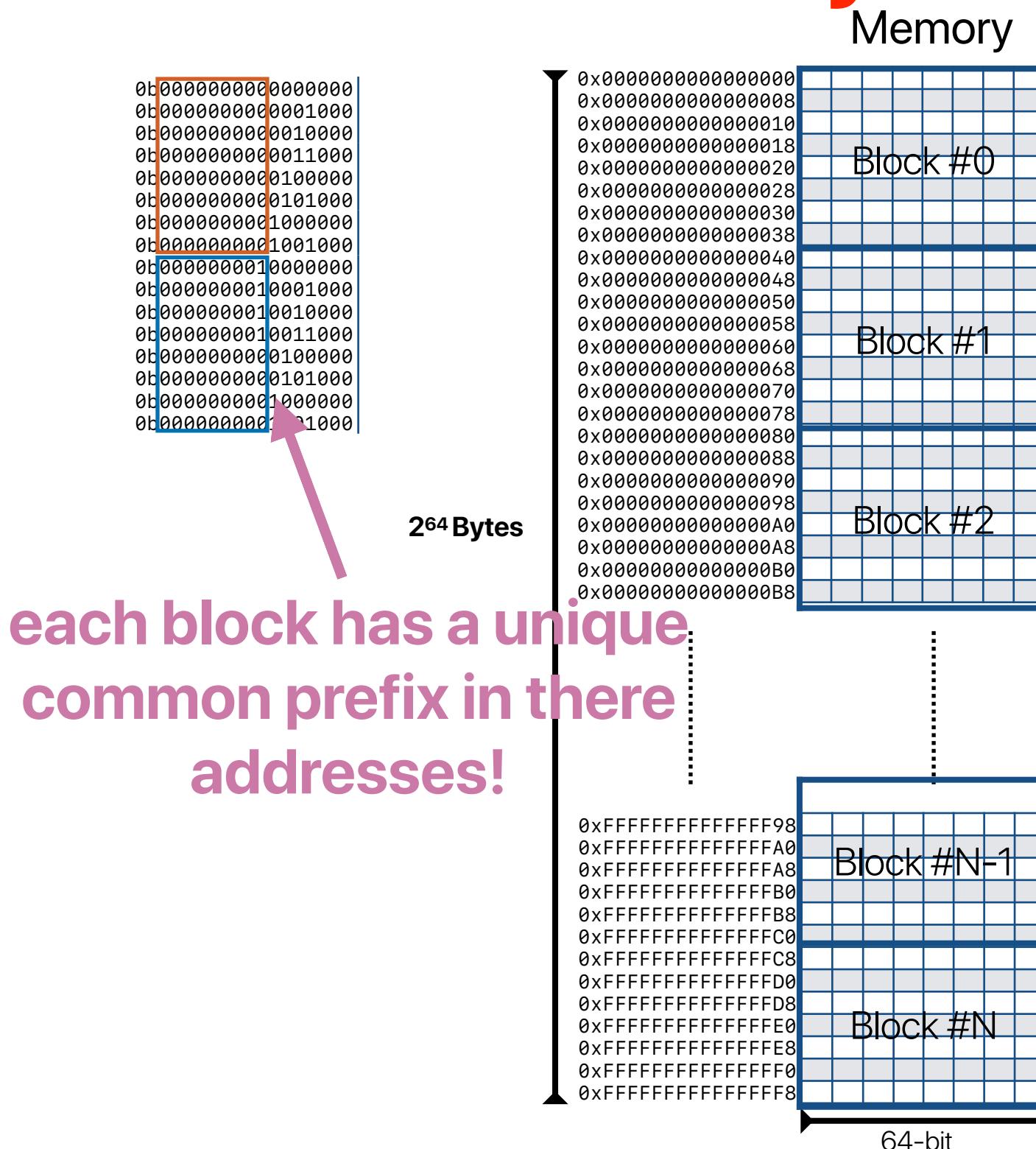
- Typically tens of static instructions — at most several KBs
- the cache should store a few blocks**
- Data — program can read/write the same data many times (e.g., vectors in matrix-vector product)
- the cache must be able to distinguish blocks**

How to tell who is there?

?

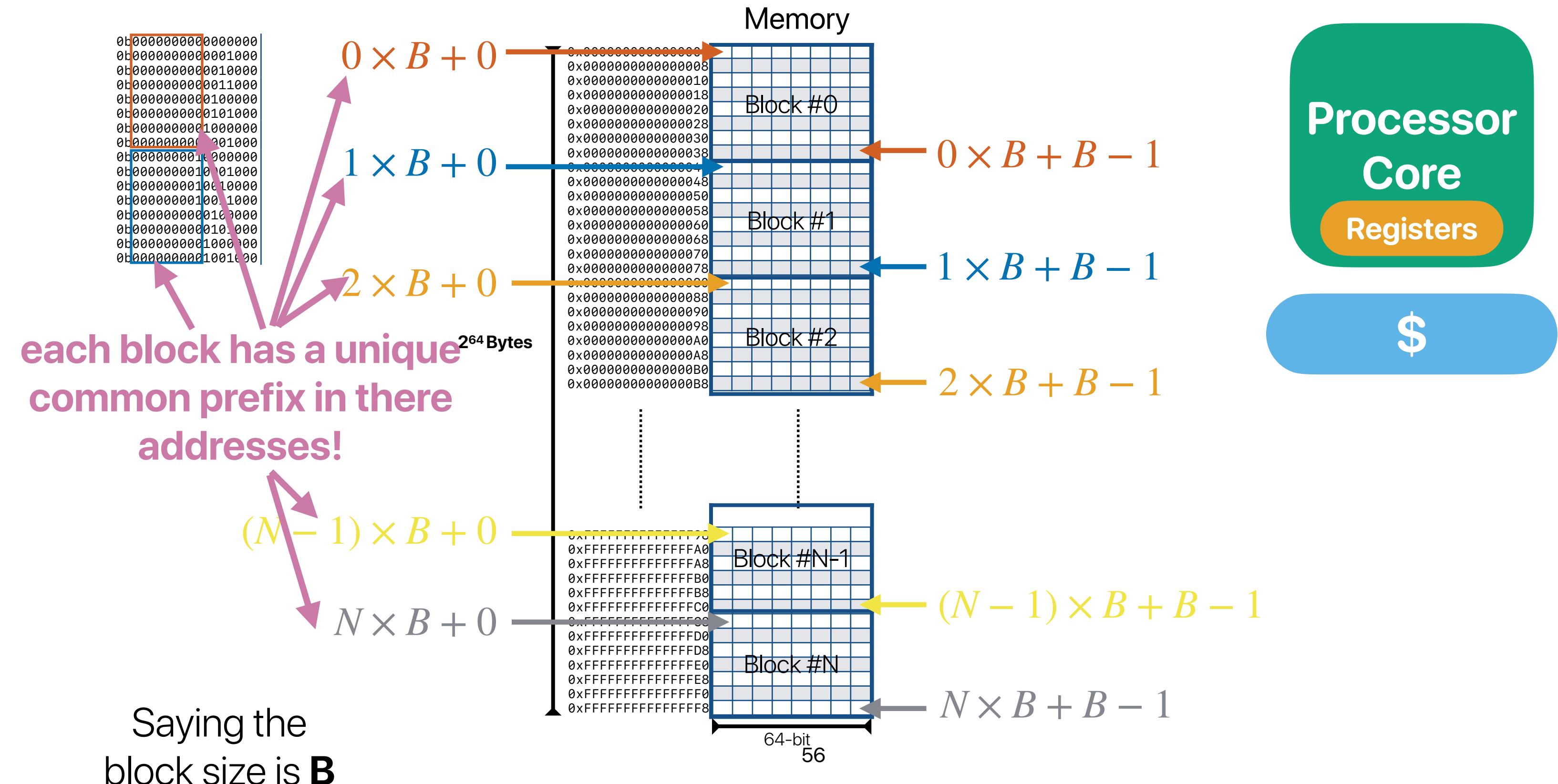
?	0123456789ABCDEF
?	This is CS 203:
	Advanced Compute
	r Architecture!
	This is CS 203:
	Advanced Compute
	r Architecture!
	This is CS 203:
	Advanced Compute
	r Architecture!
	This is CS 203:
	Advanced Compute
	r Architecture!
	This is CS 203:
	Advanced Compute
	r Architecture!
	This is CS 203:

Partition memory addresses into fix-sized chunks



\$

Partition memory addresses into fix-sized chunks



How to tell who is there?

the common address prefix in each block

tag array

0x000	This is CS 203: Advanced Compute Architecture!
0x001	This is CS 203: Advanced Compute Architecture!
0xF07	This is CS 203: Advanced Compute Architecture!
0x100	This is CS 203: Advanced Compute Architecture!
0x310	This is CS 203: Advanced Compute Architecture!
0x450	This is CS 203: Advanced Compute Architecture!
0x006	This is CS 203: Advanced Compute Architecture!
0x537	This is CS 203: Advanced Compute Architecture!
0x266	This is CS 203: Advanced Compute Architecture!
0x307	This is CS 203: Advanced Compute Architecture!
0x265	This is CS 203: Advanced Compute Architecture!
0x80A	This is CS 203: Advanced Compute Architecture!
0x620	This is CS 203: Advanced Compute Architecture!
0x630	This is CS 203: Advanced Compute Architecture!
0x705	This is CS 203: Advanced Compute Architecture!
0x216	This is CS 203: Advanced Compute Architecture!



Processor Core

Registers

How to tell whether

block offset
tag

1w 0x0008

1w 0x4048

0x404 not found,
go to lower-level memory

		Valid Bit	Dirty Bit	data
		tag		0123456789ABCDEF
1	1	0x000		This is CSE13:
1	1	0x001		Advanced Compute
1	0	0xF07		r Architecture!
0	1	0x100		This is CS 203:
1	1	0x310		Advanced Compute
1	1	0x450		r Architecture!
0	1	0x006		This is CS 203:
0	1	0x537		Advanced Compute
1	1	0x266		r Architecture!
1	1	0x307		This is CS 203:
0	1	0x265		Advanced Compute
0	1	0x80A		r Architecture!
1	1	0x620		This is CS 203:
1	1	0x630		Advanced Compute
1	0	0x705		r Architecture!
0	1	0x216		This is CS 203:

Announcement

- Make a reservation for practice examine at the testing center
 - Please schedule through
<https://testcenter.cs.ucr.edu/Make-Change-Delete-Reservations>
 - You may start scheduling today
 - Instructor/TA does not handle the scheduling
 - First come first serve — the same policy for midterm/final
 - This will help you get familiar with the environment for midterm and final
 - It's your responsibility if you miss a question due to any unfamiliarity of the testing environment
- Reading quiz #3 due next **Tuesday** before the lecture
- Assignment #1 due **tonight**
 - You should run the performance measurement yourself and calculate results based on that — everyone should have a different answer
 - We do not support people on the last minute. Please carefully plan your time and respect the instructor/TA
- Assignment 2 and Programming Assignment 2 are both released
 - You should start working on them as soon as possible
 - Make sure you submit to the right place (we won't open them until the assignment 1 due)



Computer Science & Engineering

203

つづく

