Virtual memory (II)

Hung-Wei Tseng

Recap: Demo revisited

```
Process C is using CPU: 4. Value of a is 685161796.000000 and address of a is 0x6010b0
Process A is using CPU: 4. Value of a is 217757257.000000 and address of a is 0x6010b0
Process B is using CPU: 4. Value of a is 2057721479.000000 and address of a is 0x6010b0
Process D is using CPU: 4. Value of a is 1457934803.000000 and address of a is 0x6010b0
Process C is using CPU: 4. Value of a is 685161796.000000 and address of a is 0x6010b0
Process A is using CPU: 4. Value of a is 217757257.000000 and address of a is 0x6010b0
Process B is using CPU: 4. Value of a is 2057721479.000000 and address of a is 0x6010b0
Process D is using CPU: 4. Value of a is 1457934803.000000 and address of a is 0x6010b0
```

The same processor!

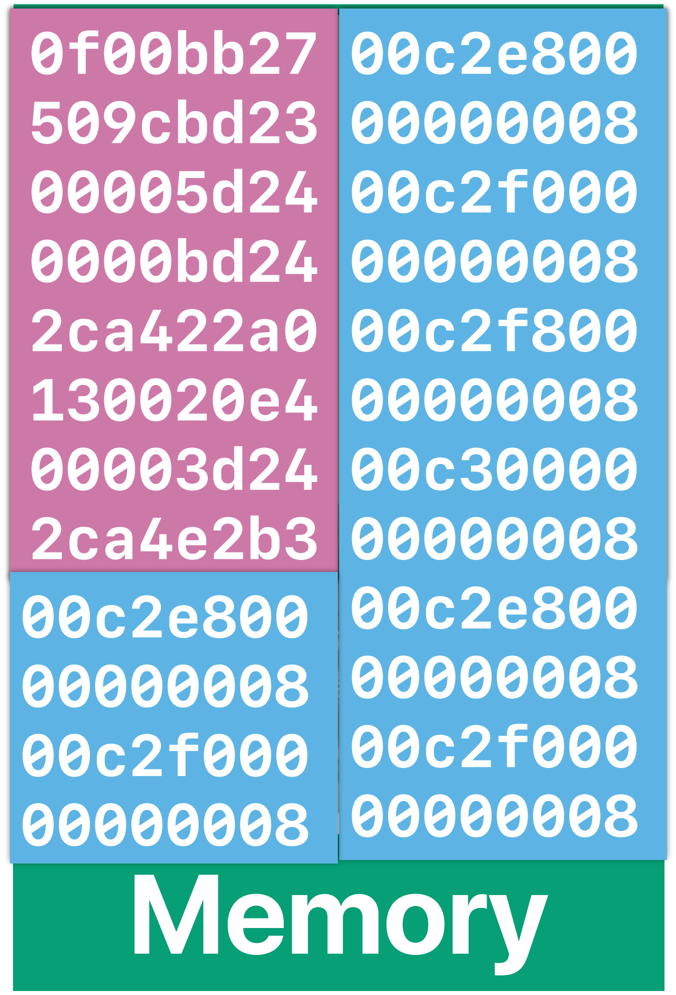
Different values

Different values are preserved

The same memory address!

Why: If we expose memory directly to the processor (I)

What if my program needs more memory?

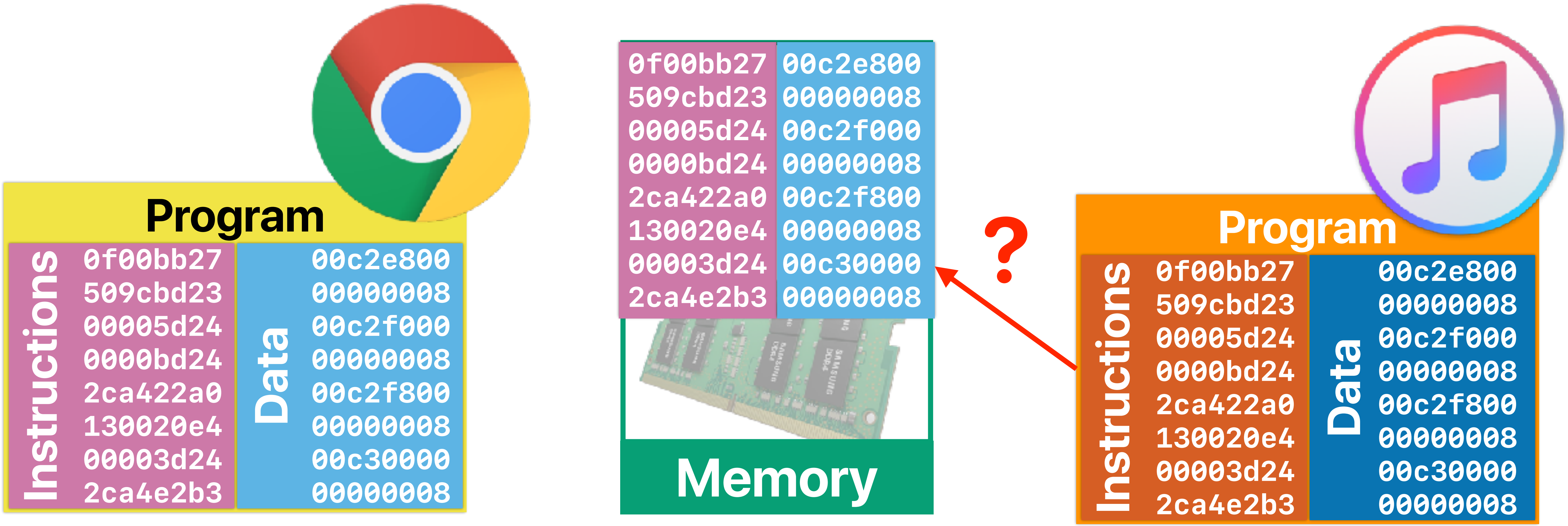


Why: If we expose memory directly to the processor (II)

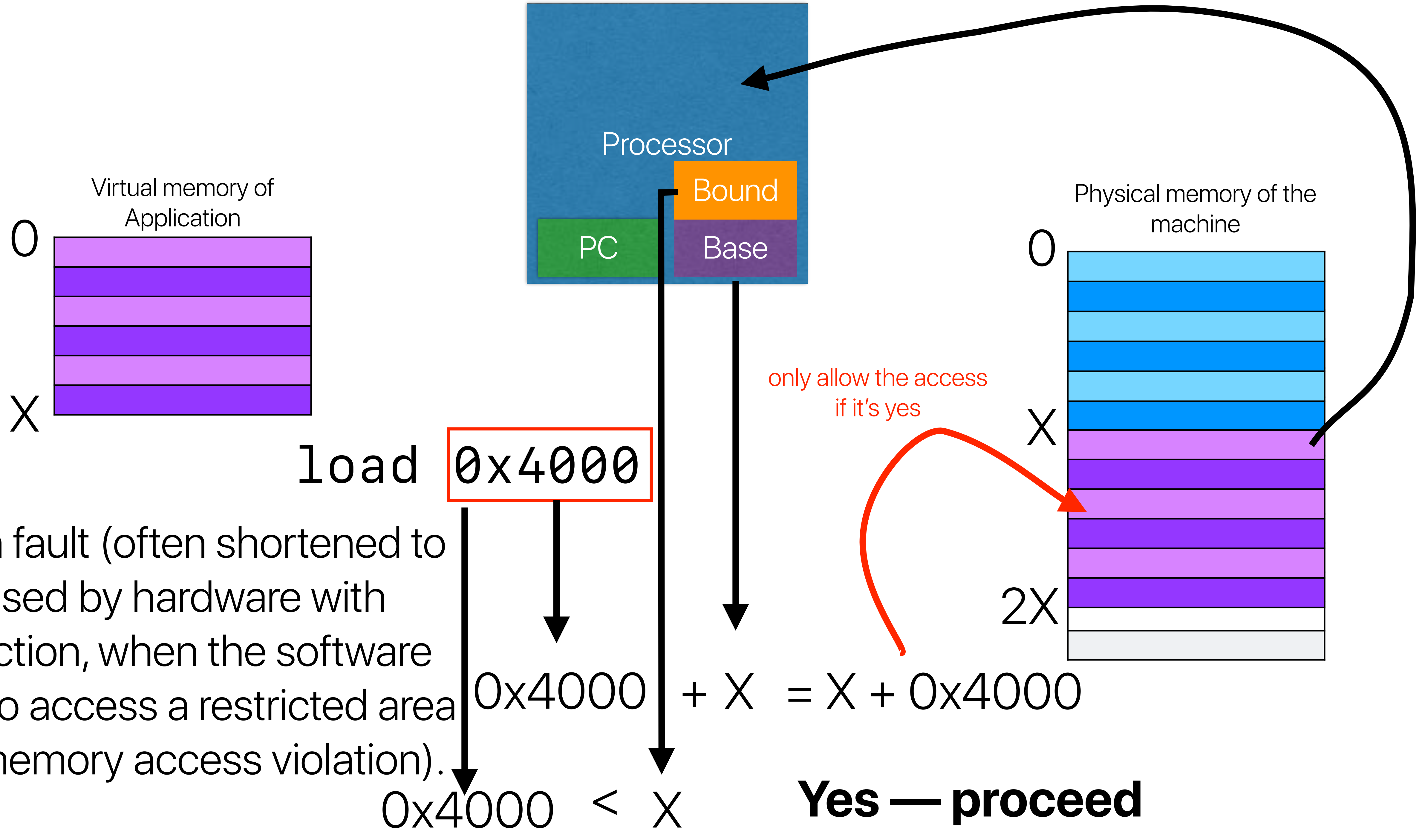
What if my program runs on a machine with a different memory size?

Why: If we expose memory directly to the processor (III)

What if both programs need to use memory?



Recap: Full picture of segmentation



a segmentation fault (often shortened to segfault), raised by hardware with memory protection, when the software has attempted to access a restricted area of memory (a memory access violation).

Yes — proceed

No — segmentation fault!!!

Current scoreboard

Red	Blue
9	12

Poll close in 1:30

Efficiency of Segmentation

- Regarding segments, how many of the followings are correct?
  ① Each segment must occupy contiguous physical memory locations
  ② The system must allocate and reserve the physical memory locations for a segment whenever the program using that segment is scheduled
  ③ An application can pre-allocate a large segment but turn out not using every byte in the segment
  ④ The system may not be able to allocate space for a segment even though the total capacity of available physical locations is sufficient

  A. 0
  B. 1
  C. 2
  D. 3
  E. 4

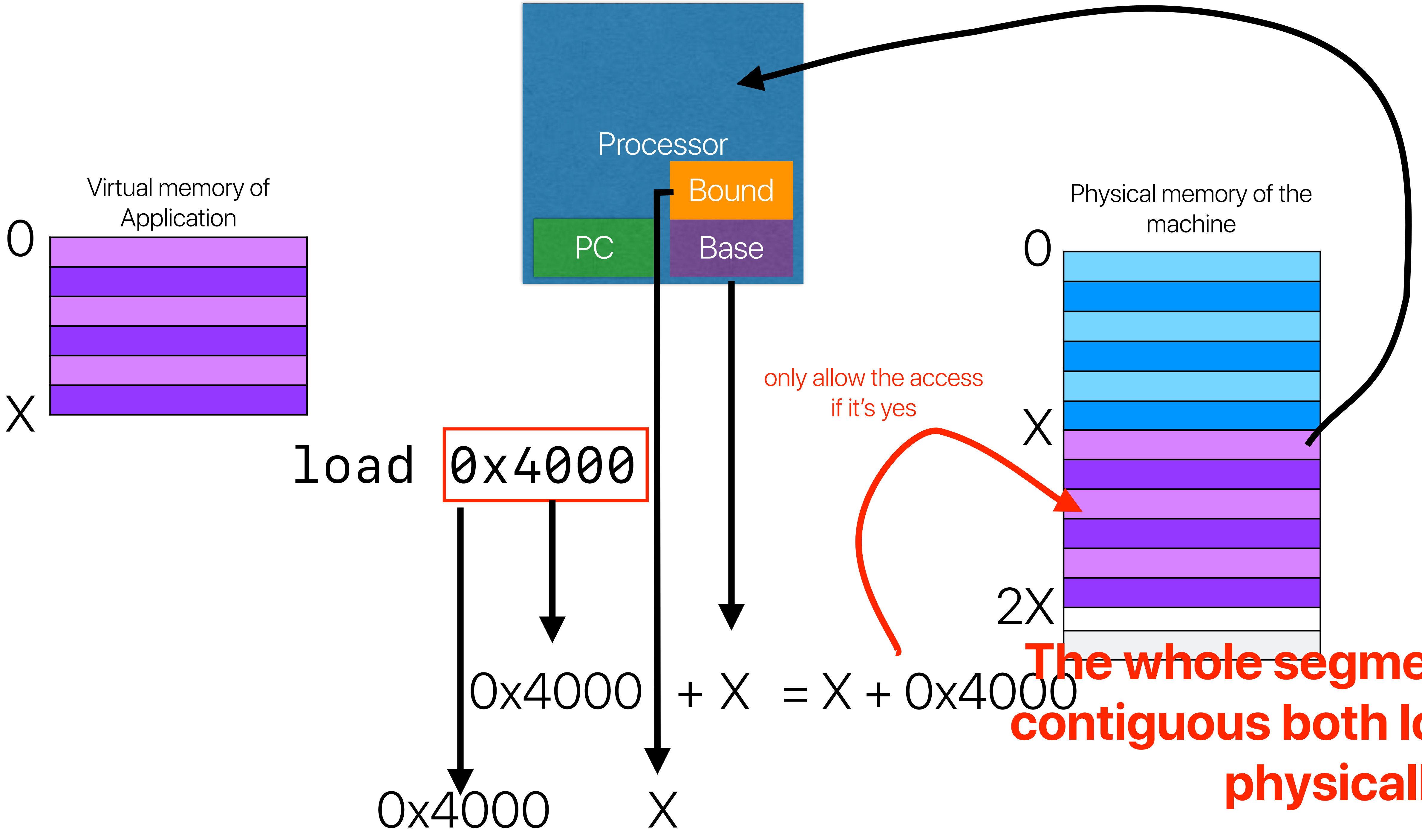
Poll close in 1:30

Efficiency of Segmentation

- Regarding segments, how many of the followings are correct?
  ① Each segment must occupy contiguous physical memory locations
  ② The system must allocate and reserve the physical memory locations for a segment whenever the program using that segment is scheduled
  ③ An application can pre-allocate a large segment but turn out not using every byte in the segment
  ④ The system may not be able to allocate space for a segment even though the total capacity of available physical locations is sufficient

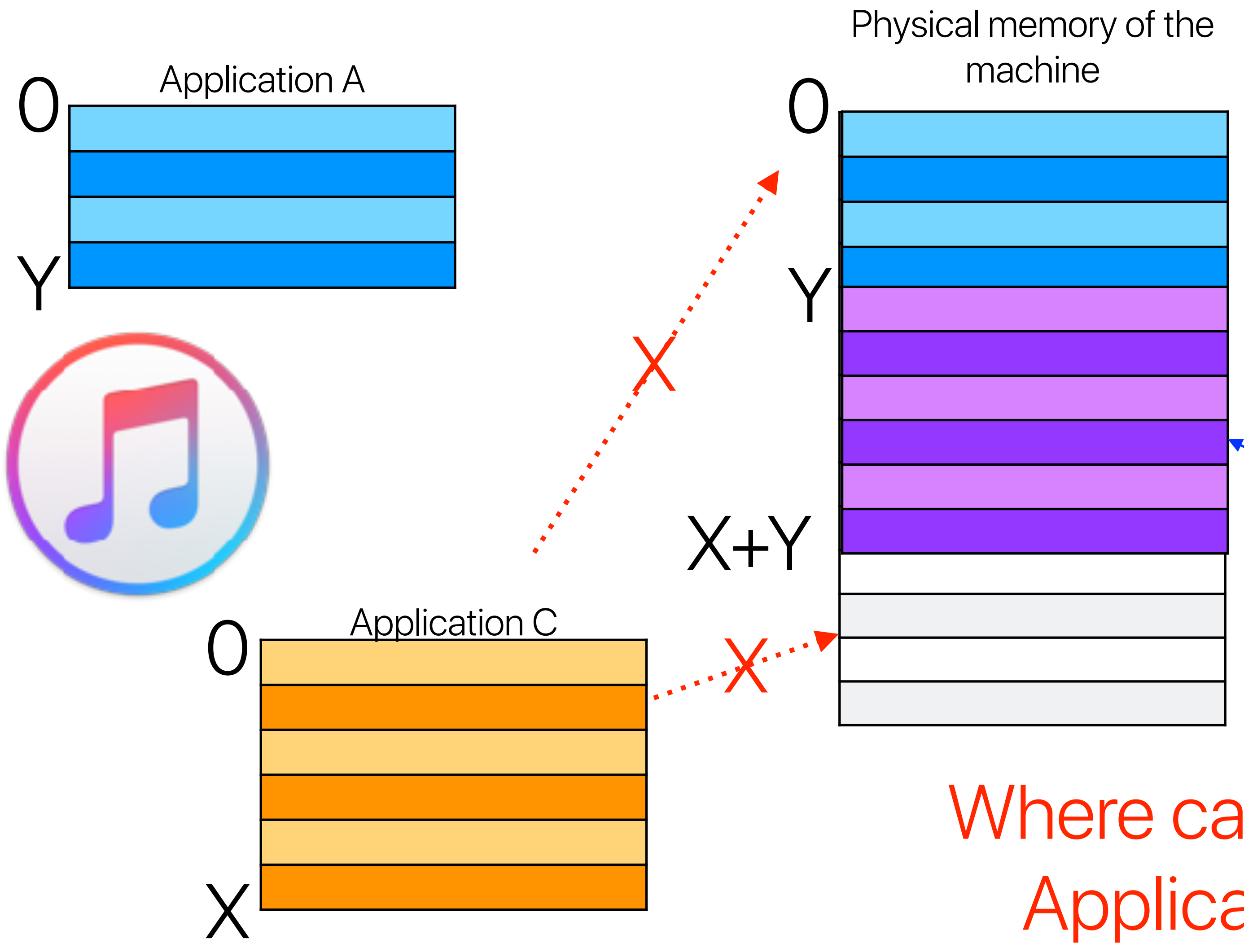
  A. 0
  B. 1
  C. 2
  D. 3
  E. 4

Recap: segmentation



The whole segment must be contiguous both logically and physically!

What if?



What if Application B only uses part of the allocated space?

Where can we map Application C?

External Fragment

Even though we have space, we still cannot map App. C

Internal Fragment

We waste some space in the allocated segment

11

Efficiency of Segmentation

- Regarding segments, how many of the followings are correct?
  ① Each segment must occupy contiguous physical memory locations
  ② The system must allocate and reserve the physical memory locations for a segment whenever the program using that segment is scheduled
  ③ An application can pre-allocate a large segment but turn out not using every byte in the segment
  ④ The system may not be able to allocate space for a segment even though the total capacity of available physical locations is sufficient

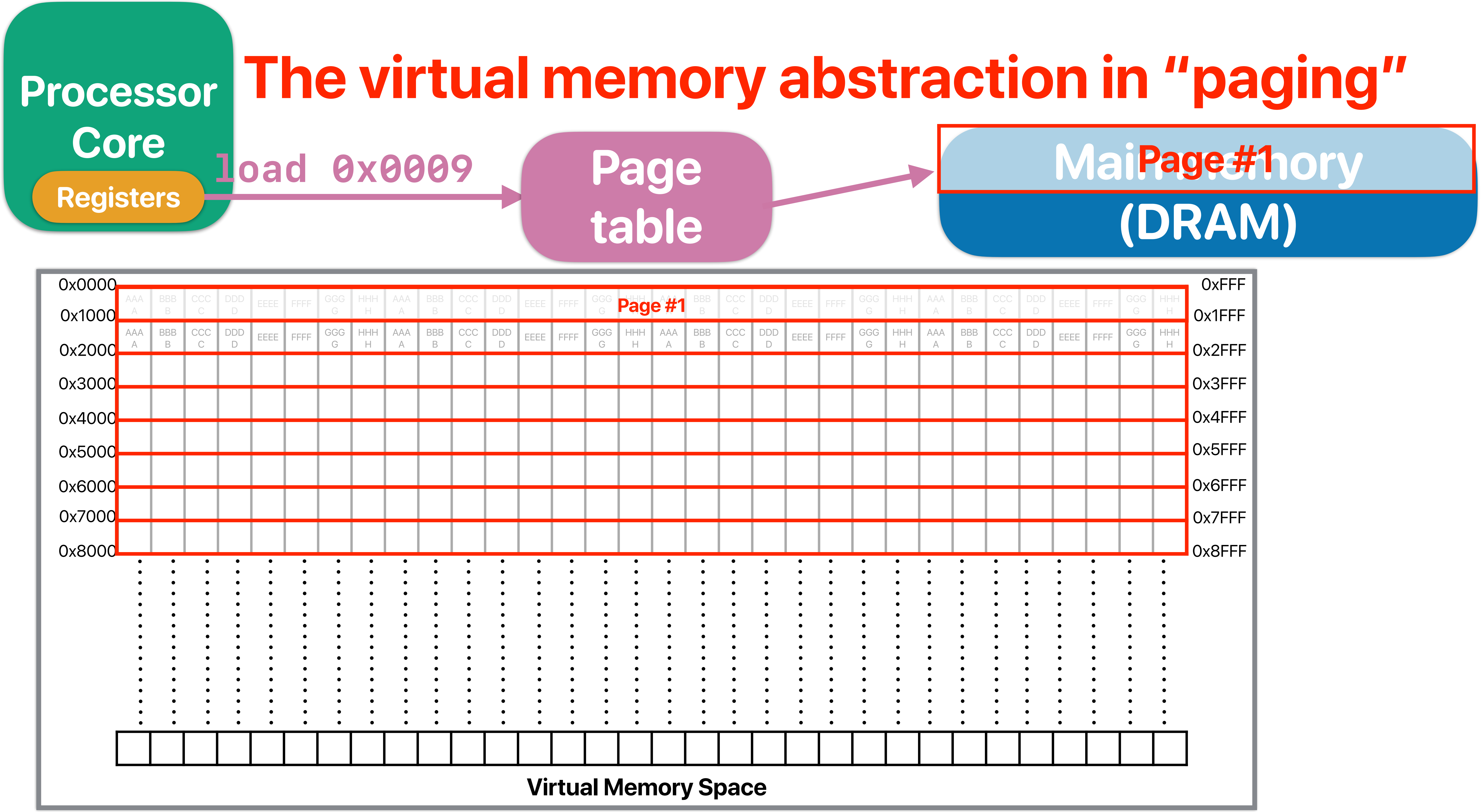
  A. 0
  B. 1
  C. 2
  D. 3
  E. 4

Outline

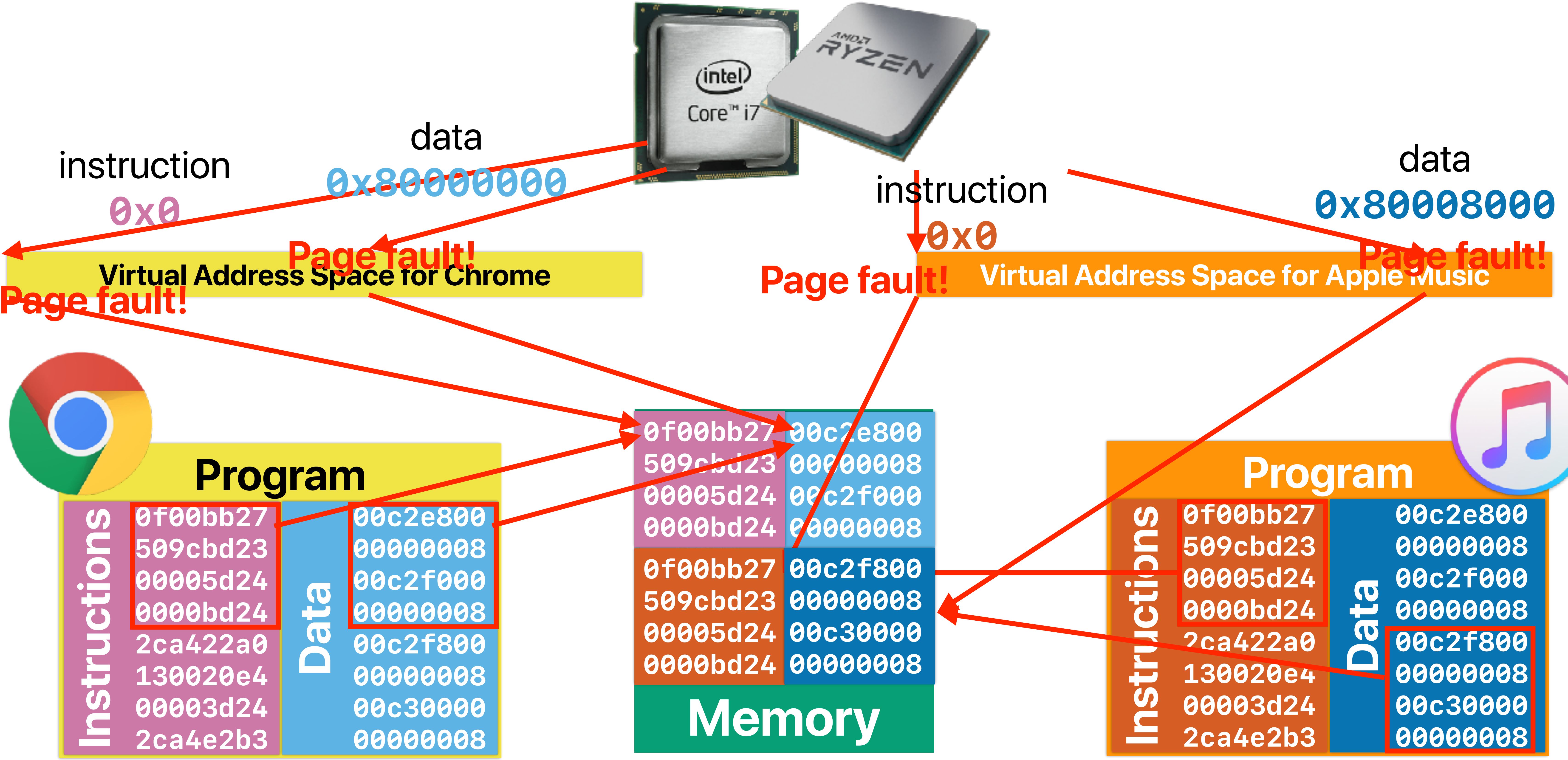
- Demand paging
- Making demand paging efficient
- Swapping

When to create a virtual to physical address mapping? — Demand paging

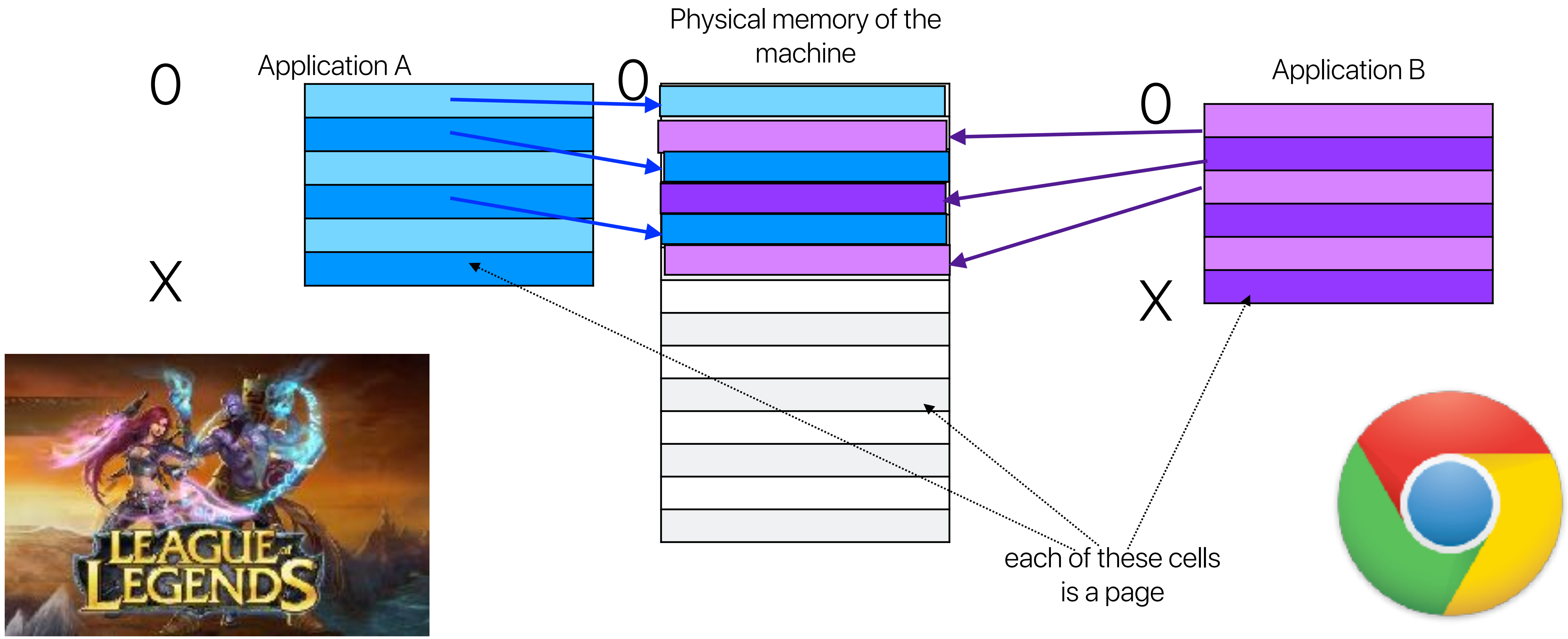
The virtual memory abstraction in "paging"



Demand paging



Demand paging



Terminology of Demand paging

- **Paging**: partition virtual/physical memory spaces into fix-sized pages
- **Page fault**: when the requested page cannot be found in the physical memory — created the demand of allocating pages!
- **Demand paging**: Allocate a physical memory page for a virtual memory page when the virtual page is needed (page fault occurs)
    - There is also **shadow paging** used by embedded systems, mobile phones — they load the whole program/data into the physical memory when you launch it

Poll close in 1:30

Segmentation v.s. demand paging

- How many of the following statements is/are correct regarding segmentation and demand paging?
  ① Segments can cause more external fragmentations than demand paging
  ② Paging can still cause internal fragmentations
  ③ The overhead of address translation in segmentation is higher
  ④ Consecutive virtual memory address may not be consecutive in physical address if we use demand paging
  A. 0
  B. 1
  C. 2
  D. 3
  E. 4

Poll close in 1:30

Segmentation v.s. demand paging

- How many of the following statements is/are correct regarding segmentation and demand paging?
  - ① Segments can cause more external fragmentations than demand paging
  - ② Paging can still cause internal fragmentations
  - ③ The overhead of address translation in segmentation is higher
  - ④ Consecutive virtual memory address may not be consecutive in physical address if we use demand paging

  A. 0

  B. 1

  C. 2

  D. 3

  E. 4

Segmentation v.s. demand paging

- How many of the following statements is/are correct regarding segmentation and demand paging?
  ① Segments can cause more external fragmentations than demand paging — the main reason why we love paging!
  ② Paging can still cause internal fragmentations — within a page
  ~~③~~ ✗ The overhead of address translation in segmentation is higher — you need to provide finer-grained mapping in paging — you may need to handle page faults!
  ④ Consecutive virtual memory address may not be consecutive in physical address if we use demand paging

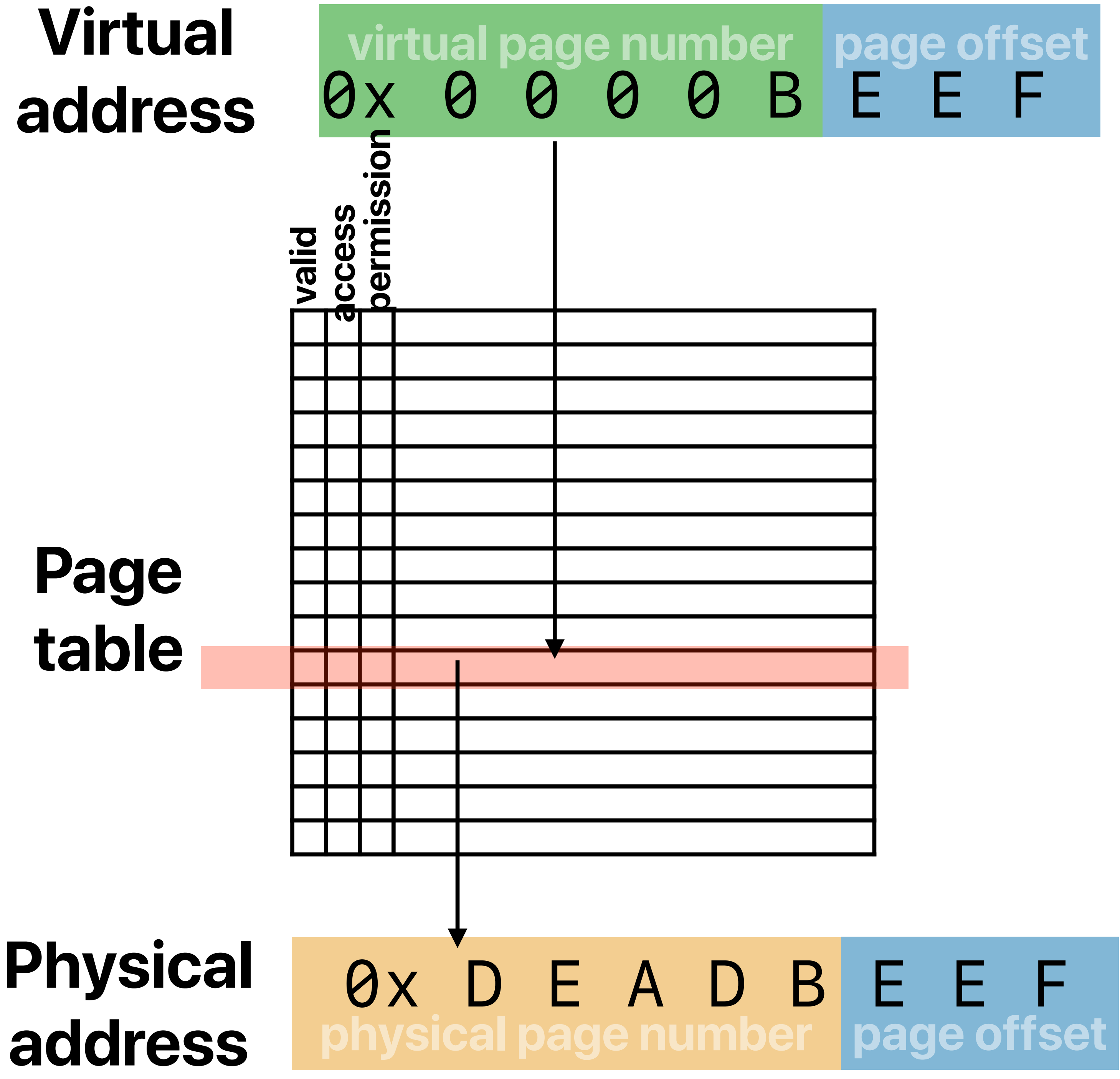
  A. 0
  B. 1
  C. 2
  D. 3
  E. 4

We haven't seen pure/true implementation of segmentations for a while, but we still use segmentation fault errors all the time!

Address translation in demand paging

Address translation

- Processor receives virtual addresses from the running code, main memory uses physical memory addresses
- Virtual address space is organized into “pages”
- The system references the **page table** to translate addresses
    - Each process has its own page table
    - The page table content is maintained by OS
- In addition to valid bit and physical page #, the page table may also store
    - Reference bit
    - Modified bit
    - Permissions



Poll close in 1:30

Size of page table

- Assume that we have 32-bit virtual address space, each page is 4KB, each page table entry is 4 bytes, how big is the page table for a process?
  A. 1MB
  B. 2MB
  C. 4MB
  D. 8MB
  E. 16MB

Poll close in 1:30

Size of page table

- Assume that we have 32-bit virtual address space, each page is 4KB, each page table entry is 4 bytes, how big is the page table for a process?
  A. 1MB
  B. 2MB
  C. 4MB
  D. 8MB
  E. 16MB

Size of page table

- Assume that we have 32-bit virtual address space, each page is 4KB, each page table entry is 4 bytes, how big is the page table for a process?
  A. 1MB
  B. 2MB
  C. 4MB
  D. 8MB
  E. 16MB

The size of each entry in the page table

Number of entries in the page table

$$\boxed{4\ bytes} \times \boxed{\frac{4\ GB}{4\ KB}} = 4 \times 1\ M = 4\ MB$$

What if we have 16 processes?

4 MB * 16 = 64MB

— we need a separate page table for each process

Poll close in 1:30

Size of page table

- Assume that we have **64-bit** virtual address space, each page is 4KB, each page table entry is 8 bytes (64-bit addresses), what magnitude in size is the page table for 32 processes?
   A. MB — 2^{20} Bytes
   B. GB — 2^{30} Bytes
   C. TB — 2^{40} Bytes
   D. PB — 2^{50} Bytes
   E. EB — 2^{60} Bytes

Poll close in 1:30

Size of page table

- Assume that we have **64-bit** virtual address space, each page is 4KB, each page table entry is 8 bytes (64-bit addresses), what magnitude in size is the page table for 32 processes?
  A. MB — 2^{20} Bytes
  B. GB — 2^{30} Bytes
  C. TB — 2^{40} Bytes
  D. PB — 2^{50} Bytes
  E. EB — 2^{60} Bytes

Size of page table

- Assume that we have **64-bit** virtual address space, each page is 4KB, each page table entry is 8 bytes (64-bit addresses), what magnitude in size is the page table for 32 processes?

  A. MB — 2^{20} Bytes
  B. GB — 2^{30} Bytes
  C. TB — 2^{40} Bytes
  D. PB — 2^{50} Bytes
  E. EB — 2^{60} Bytes

$$8\ bytes \times \frac{2^{64}\ B}{4\ KB} = 2^{3}B \times \frac{2^{64}\ B}{2^{12}\ B} = 2^{55}\ B = 32\ PB$$

$$32\ PB \times 32 = 2^{60}B = 1\ EB$$

Address translation (cont.)

- Page tables are too large to be kept on the chip (millions of entries)

  — **space overhead: surpasses cache capacity**

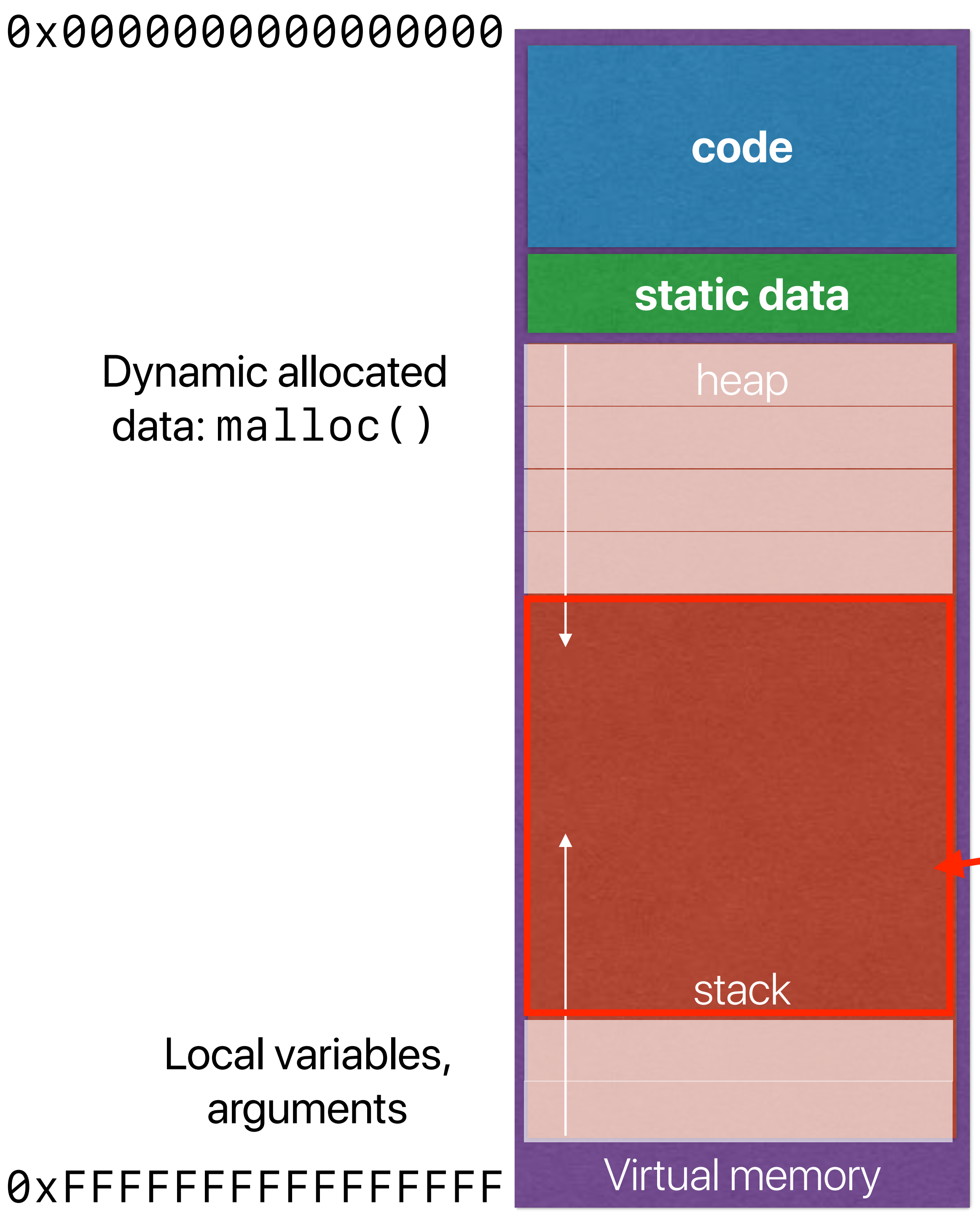
- Instead, the page tables are kept in memory

  — **memory access overhead**

  — **space overhead: can be bigger than physical main memory when address space is large**

Smaller page tables

Do we really need a large table?



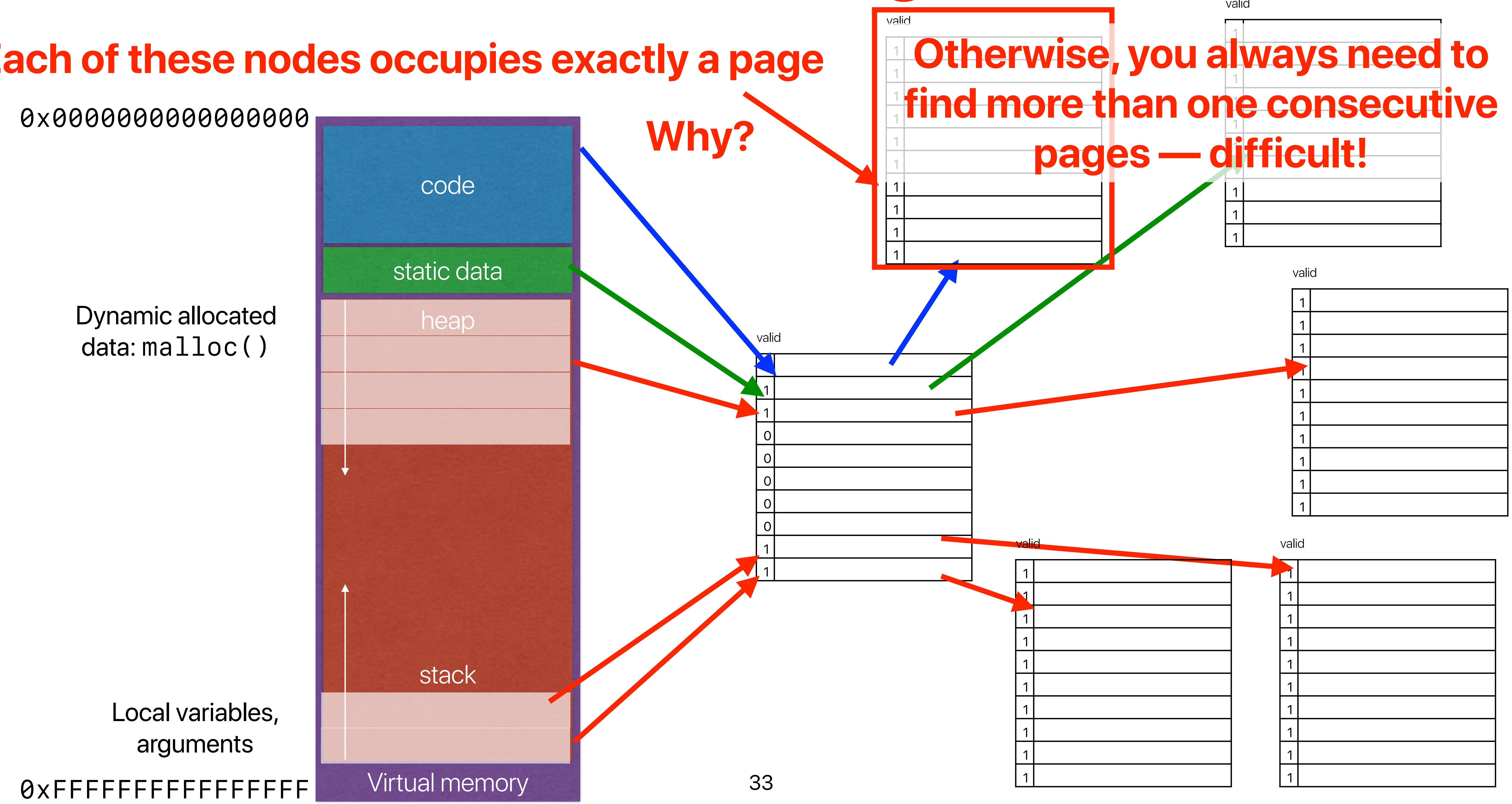
Your program probably never uses this huge area!

Hierarchical page table

Each of these nodes occupies exactly a page

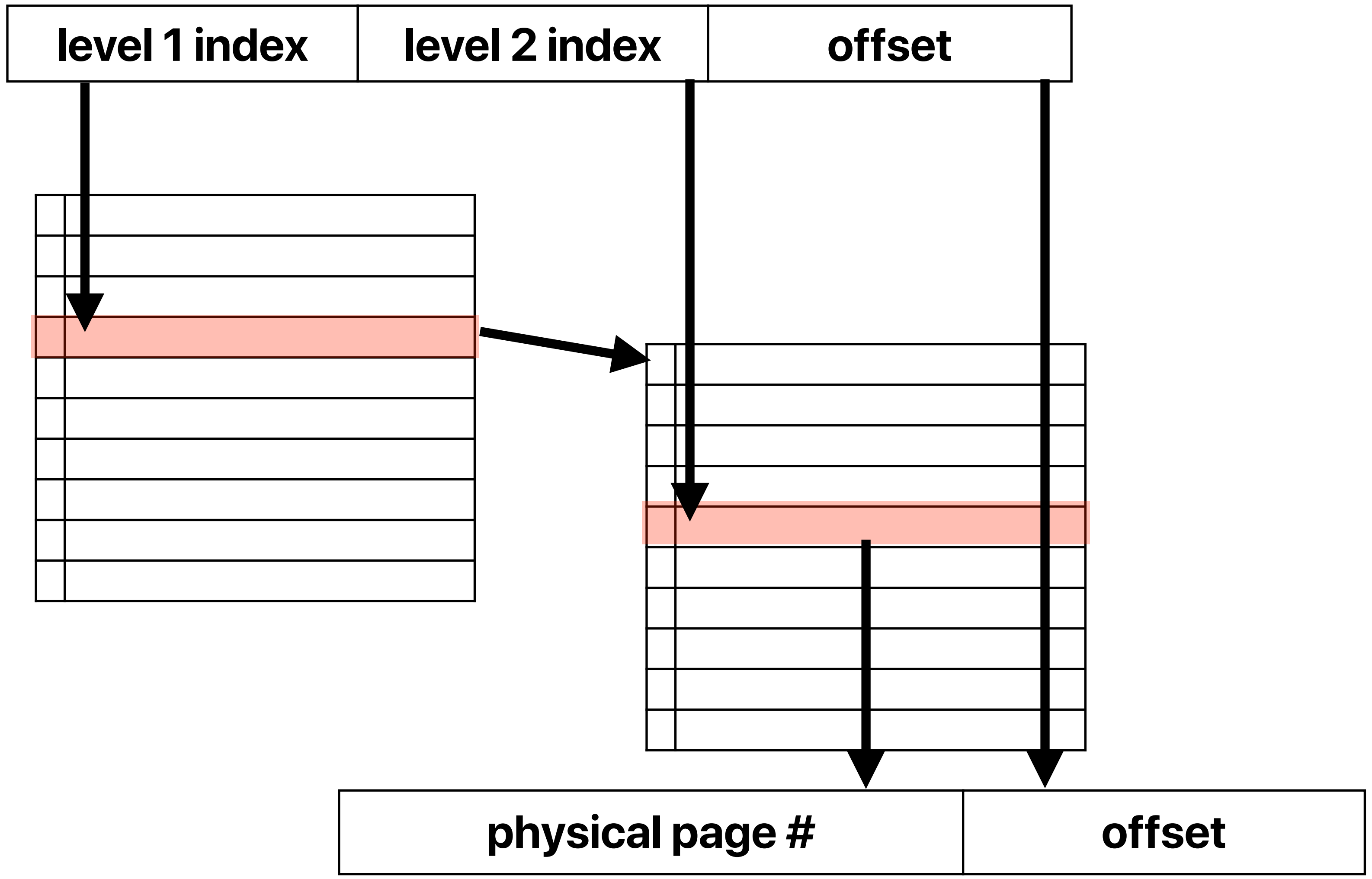
Why?

Otherwise, you always need to find more than one consecutive pages — difficult!



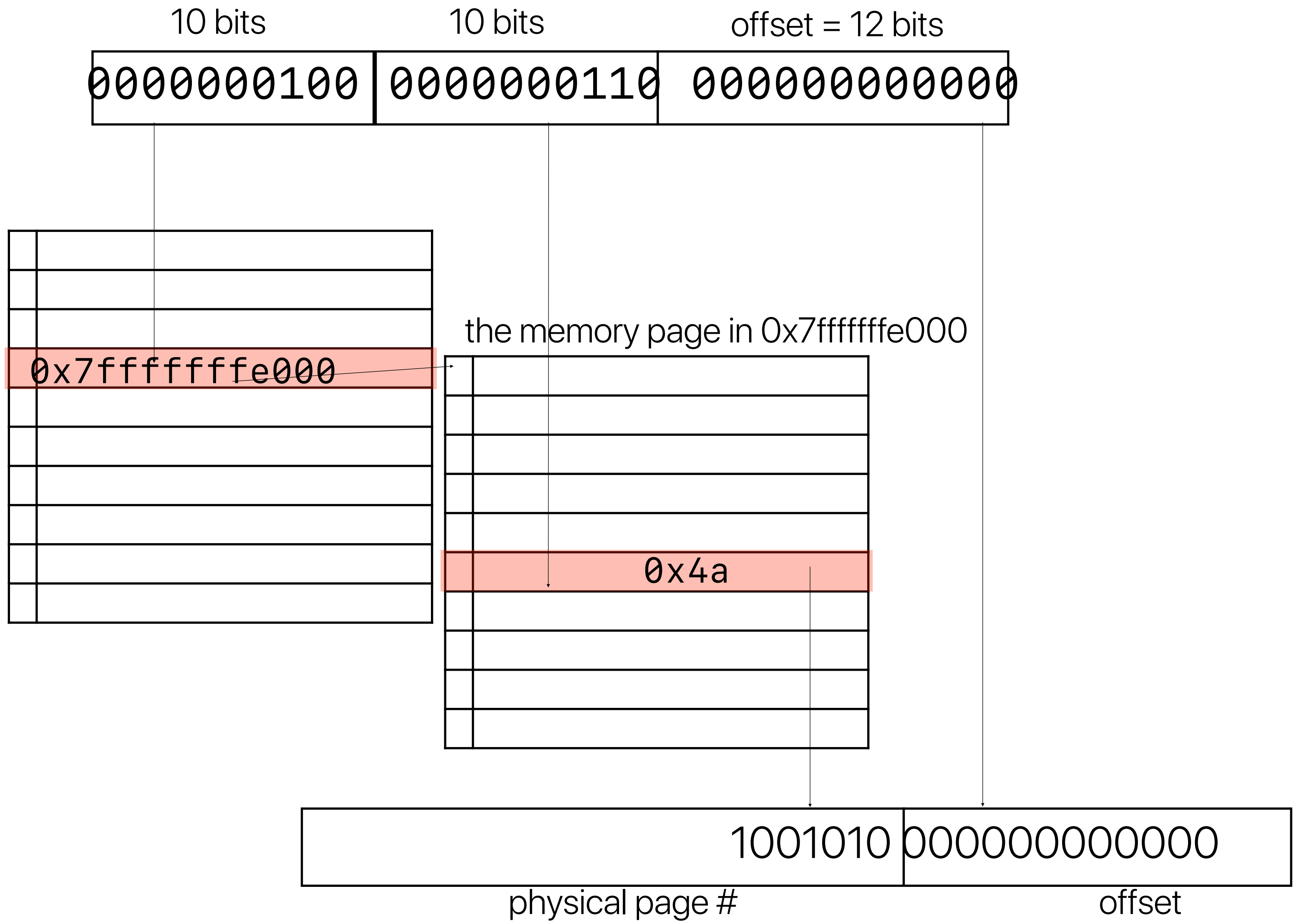
Hierarchical page table

- Break the virtual page number into several pieces
- If one piece has N bits, build an 2^N-ary tree
- Only store the part of the tree that contain valid pages
- Walk down the tree to translate the virtual address



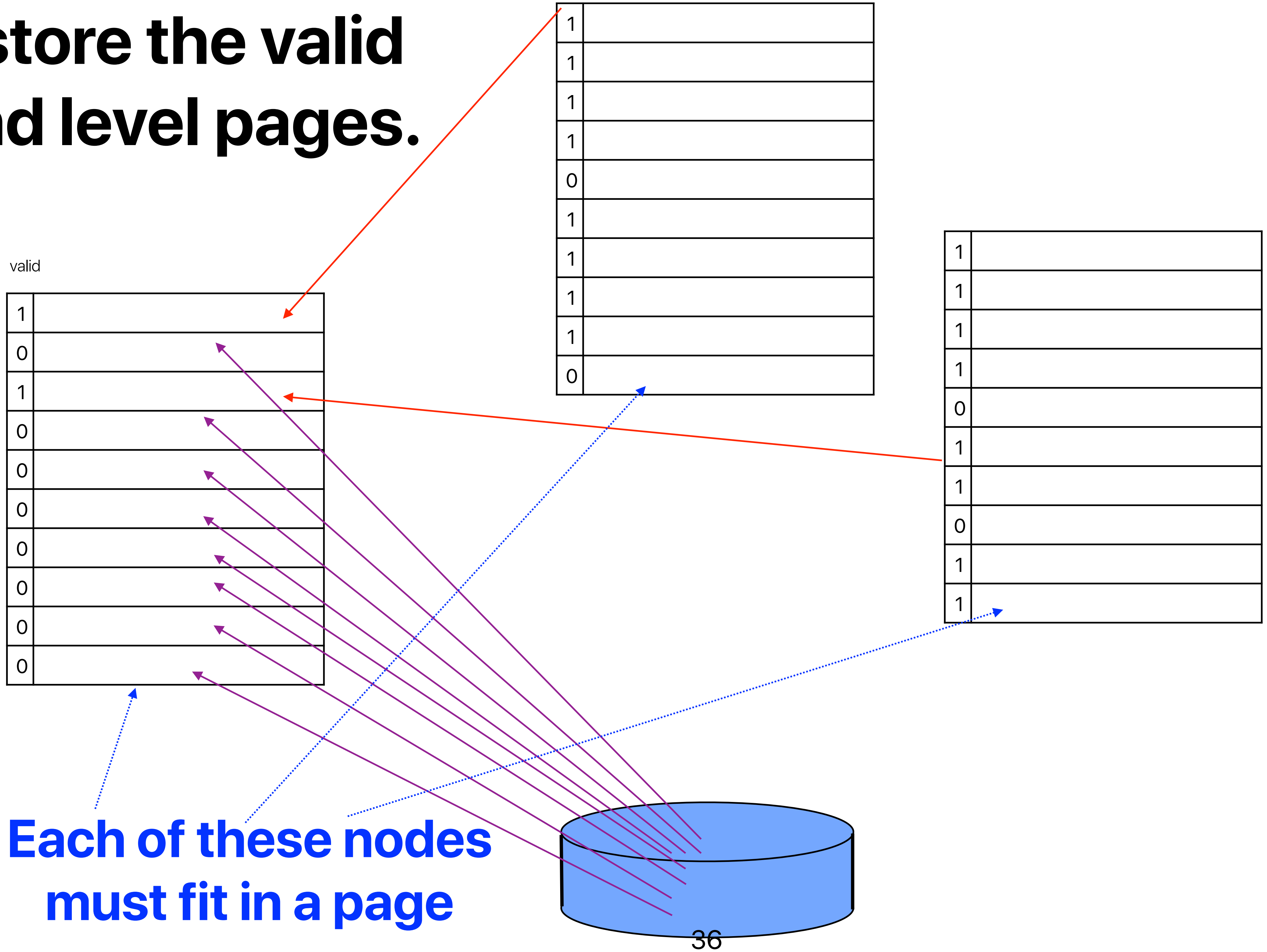
Page table walking example

- Two-level, 4KB, 10 bits index in each level
- If we are accessing 0x1006000 now…



Hierarchical page table

- **Only store the valid second level pages.**



Poll close in 1:30

How many levels do we need?

- Assume that our system uses hierarchical page table with 4KB page size under 64-bit virtual address space and each PTE is 8B in size. How many levels of indexes do we need for the hierarchical page table?
    A. 2
    B. 3
    C. 4
    D. 5
    E. 6

Poll close in 1:30

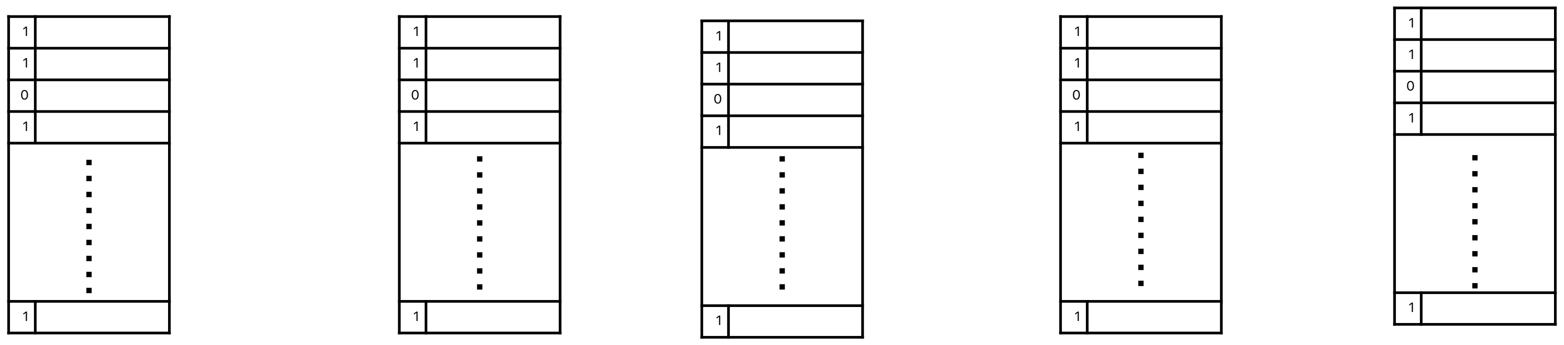
How many levels do we need?

- Assume that our system uses hierarchical page table with 4KB page size under 64-bit virtual address space and each PTE is 8B in size. How many levels of indexes do we need for the hierarchical page table?
  A. 2
  B. 3
  C. 4
  D. 5
  E. 6

How many levels do we need?

- Assume that our system uses hierarchical page table with 4KB page size under 64-bit virtual address space and each PTE is 8B in size. How many levels of indexes do we need for the hierarchical page table?

7 bits index	9 bits index	9 bits index	9 bits index	9 bits index	9 bits index	12 bits offset

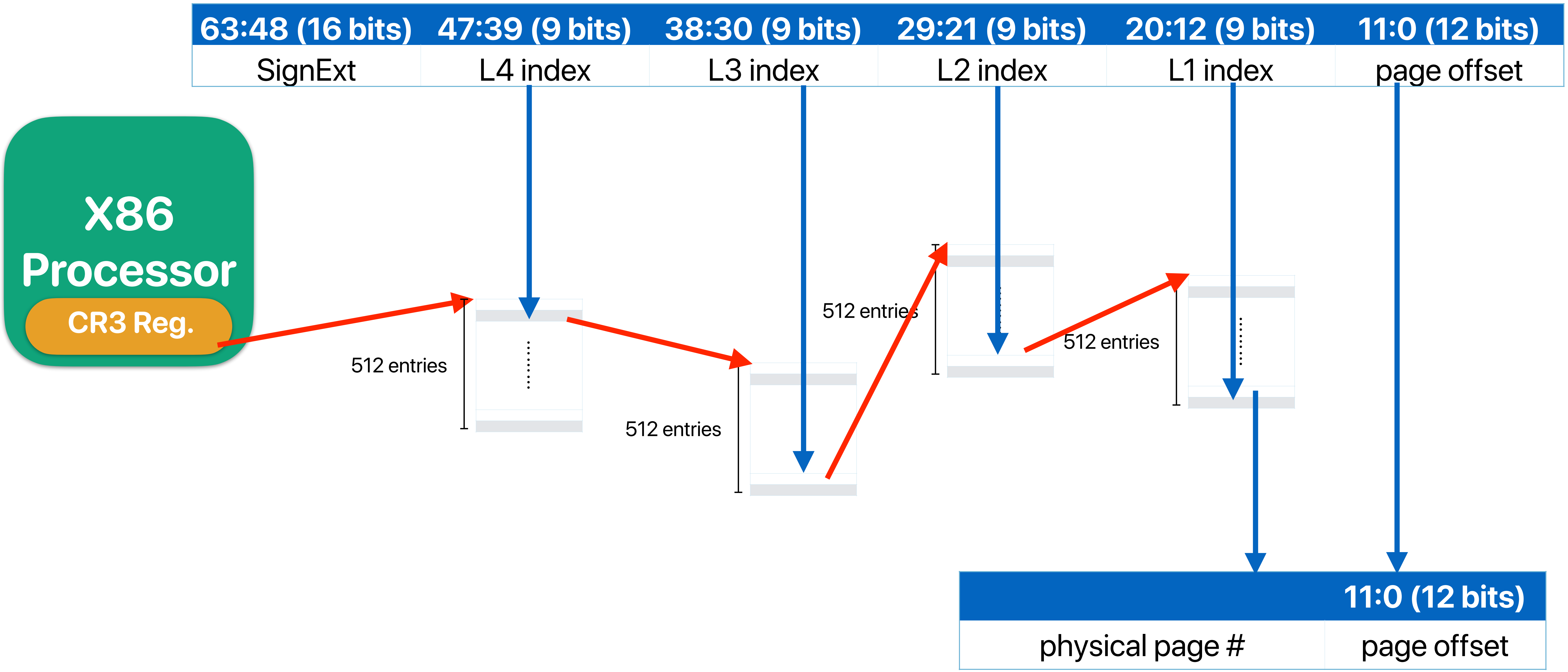


$\frac{4096}{8}=512=2^9$ $\frac{4096}{8}=512=2^9$ $\frac{4096}{8}=512=2^9$ $\frac{4096}{8}=512=2^9$ $\frac{4096}{8}=512=2^9$

How many levels do we need?

- Assume that our system uses hierarchical page table with 4KB page size under 64-bit virtual address space and each PTE is 8B in size. How many levels do we need for the hierarchical page table?
  A. 2
  B. 3
  C. 4
  D. 5
  E. 6

Case study: Address translation in x86-64



Announcement

- Reading quizzes due next Tuesday
- New office hour
  - M 3p-4p and Th 9a-10a
  - Use the office hour Zoom link, not the lecture one
- Project released
  - Groups in 2
  - **Pull the latest version — had some changes for later kernel versions** https://github.com/hungweitseng/CS202-ResourceContainer
  - **Install an Ubuntu Linux 16.04.07 VM as soon as you can!**
  - **Please do not use a real machine — you may not be able to reboot again**
- Midterm
  - Will release on 2/10/2021 0:00am and due on 2/15/2021 11:59:00pm
  - You will have to find a consecutive, non-stop 80-minute slot with this period
  - One time, cannot reinitiate — please make sure you have a stable system and network
  - **No late submission is allowed**

Computer
Science &
Engineering

202

つづく