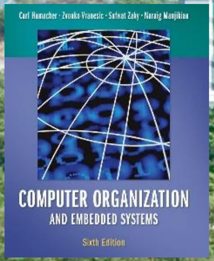
香港中文大學
The Chinese University of Hong Kong

CSCI2510 Computer Organization
Lecture 09: Virtual Memory

Ming-Chang YANG
mcyang@cse.cuhk.edu.hk

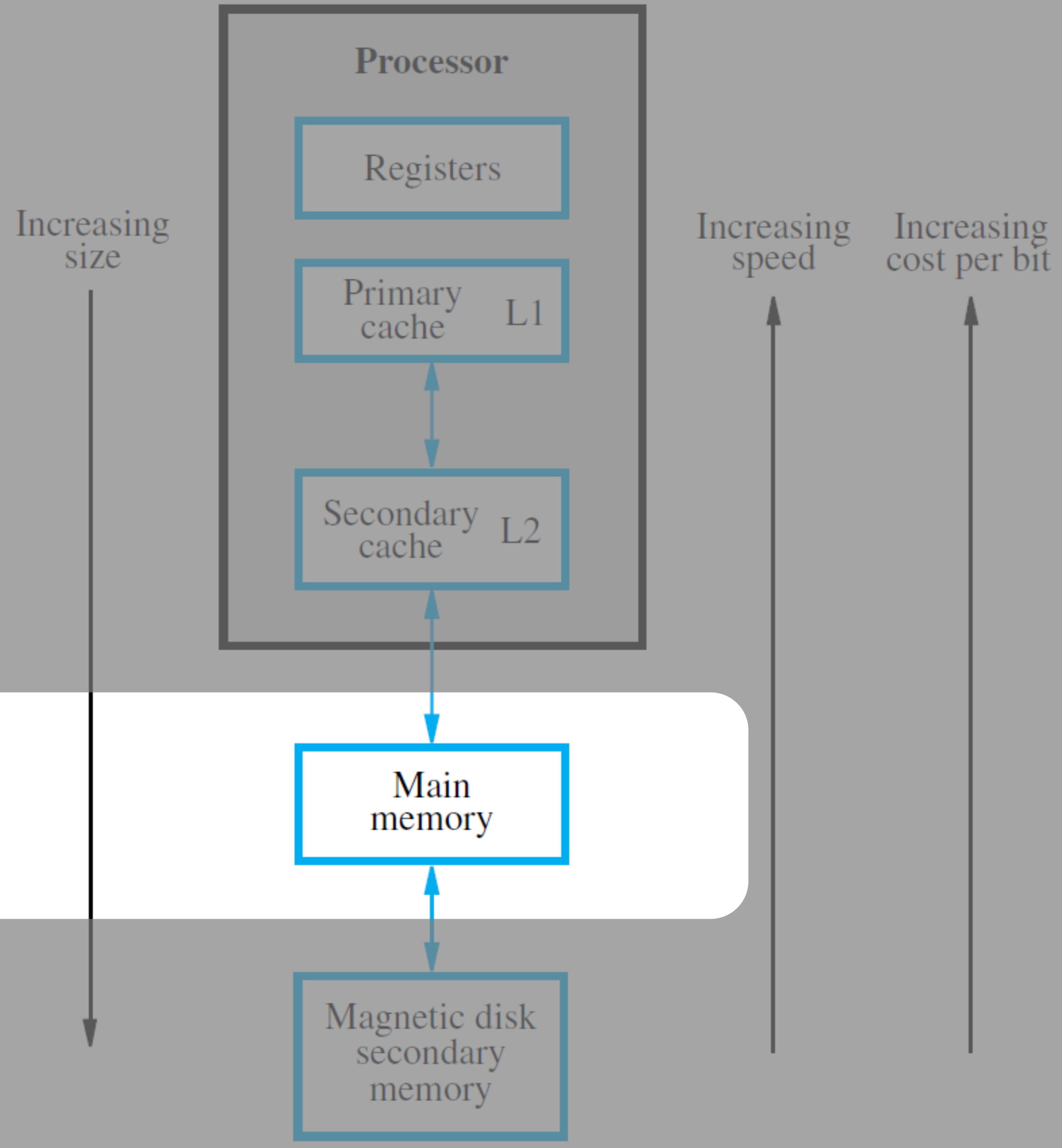


Reading: Chap. 8.8

Recall: Memory Hierarchy

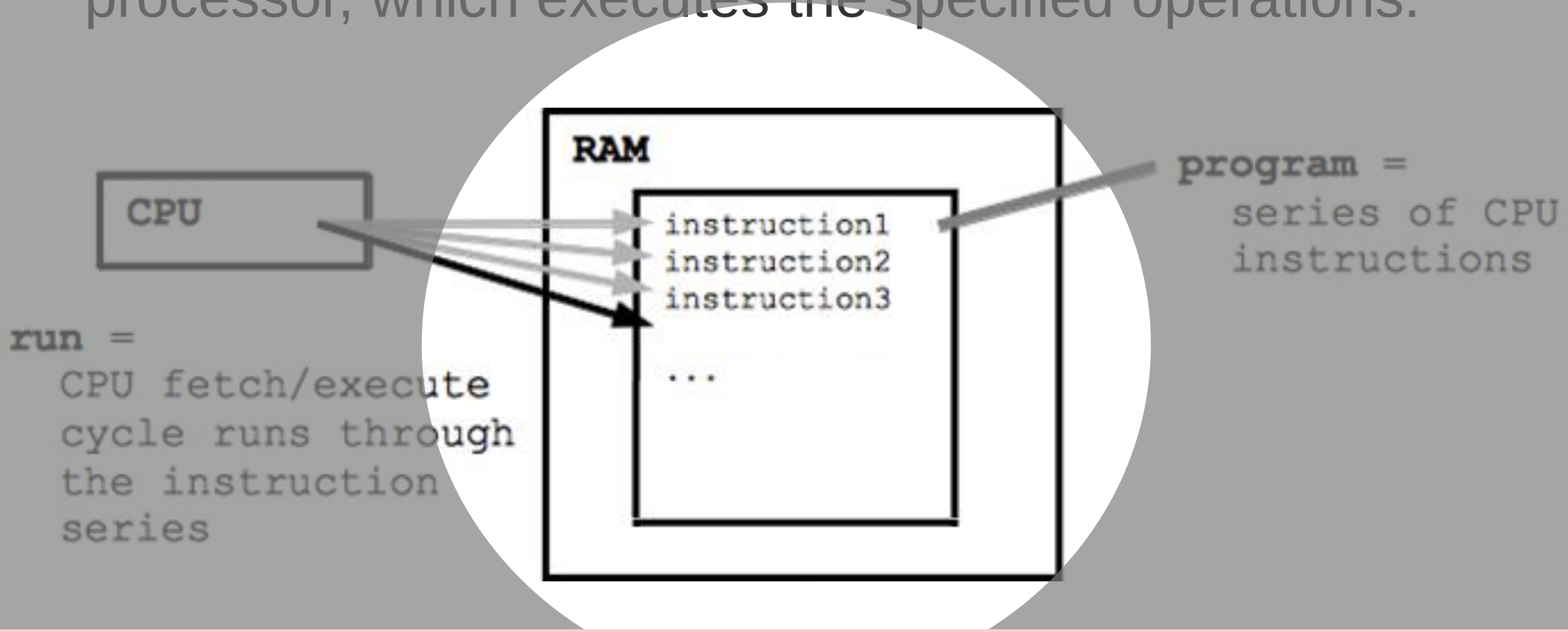
Processor

- Register: SRAM
- L1, L2 cache: SRAM
- Main memory: SDRAM
- Secondary storage: Hard disks or NVM



Recall: Instructions & Program

- A computer is governed by instructions.
  - To perform a given task, a program consisting of a list of machine instructions is stored in the memory.
    - Data to be used as operands are also stored in the memory.
  - Individual instructions are brought from the memory into the processor, which executes the specified operations.



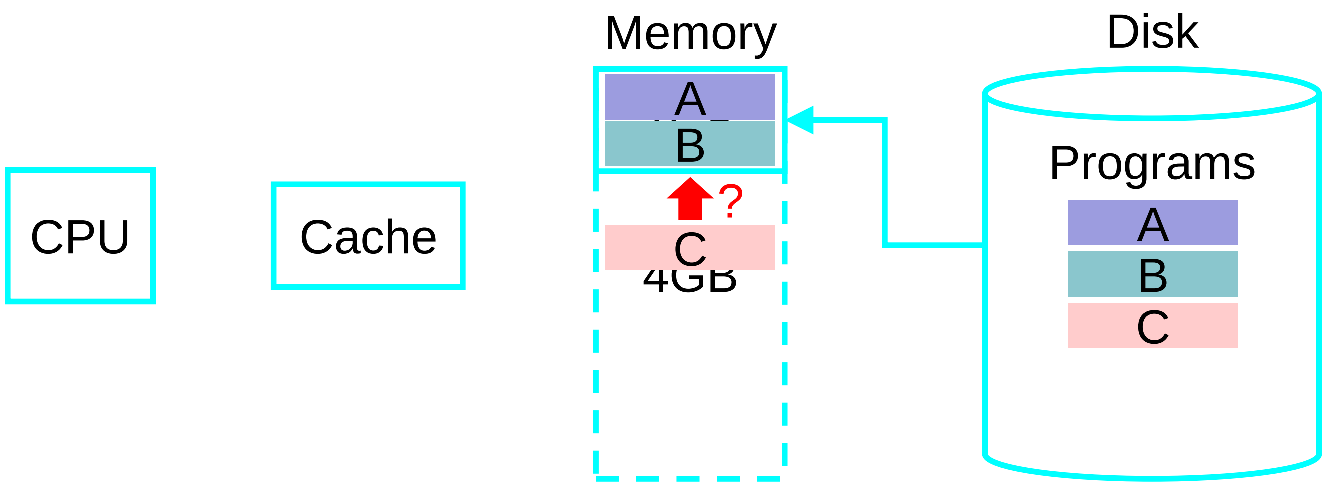
Question: What if the memory space is NOT large enough?

Outline

- Why Virtual Memory?
- MMU: Virtual-to-Physical Address Translation
  - Page Table
  - Translation Lookaside Buffer (TLB)
  - Page Fault

Why Virtual Memory?

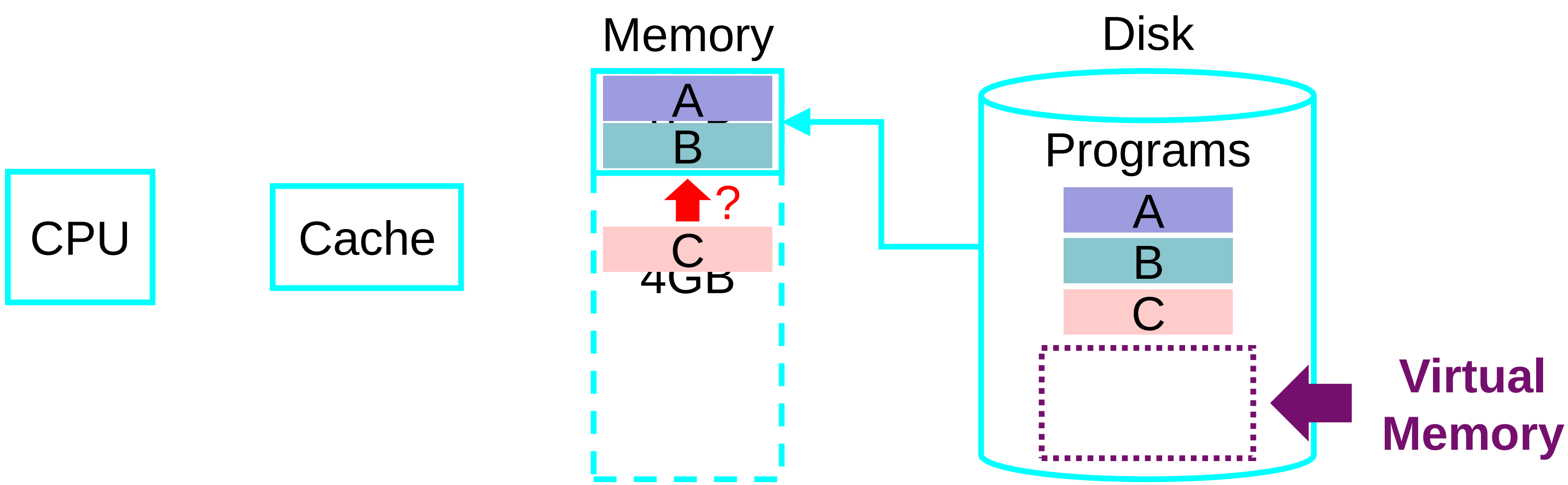
- Physical memory may **not** be as large as the "possible space" that can be addressed by a CPU.
  - E.g., a processor can address 4 GB with 32-bit address, but the space of installed main memory may only be 1GB.



- What if we want to concurrently run many programs in which the required memory capacity is **larger than** the installed memory capacity?
  - A running program is called a process (controlled by OS).

An Intuitive Solution: Virtual Memory

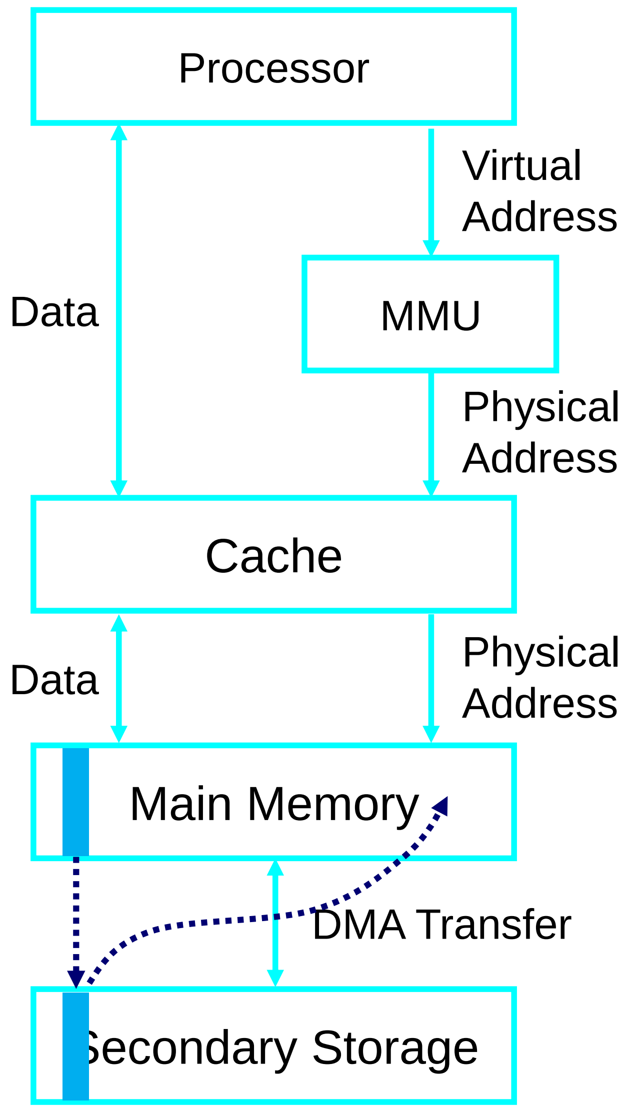
- What can we do?
  - Move some memory "parts" to a special space of disk (e.g., 500MB), then we have 500MB of "free" memory for use.
  - What if later on, those instructions/data in the saved 500MB part of memory are needed again?
  - We need to "free" some other memory parts in order to move the instructions/data back from the disk …



Basic Concept of Virtual Memory (1/2)

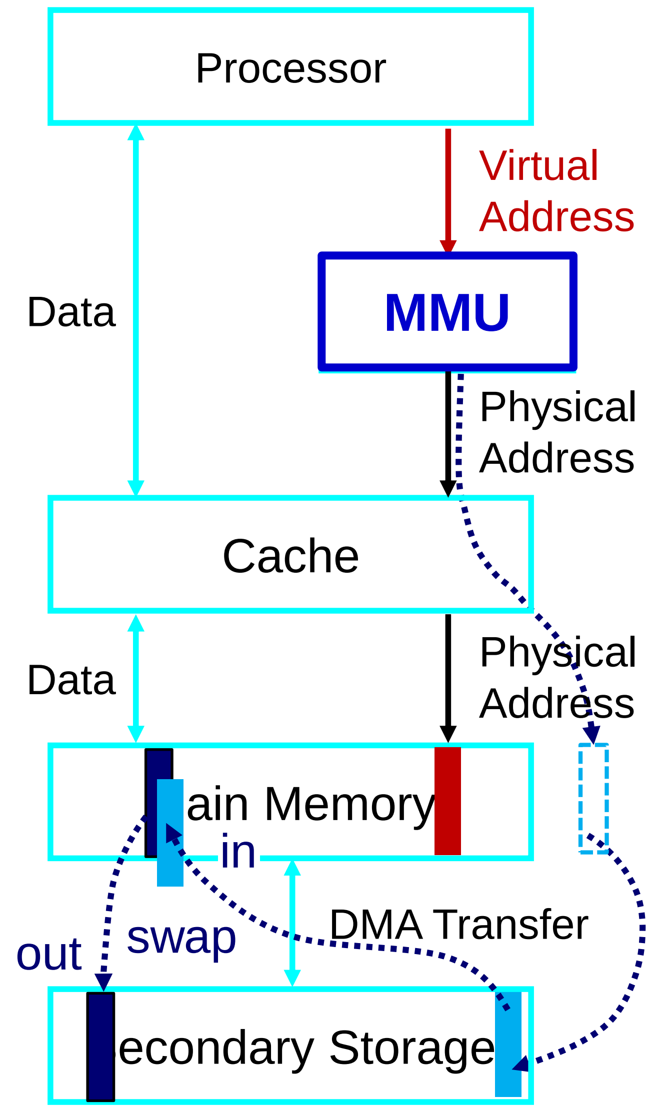
- **Virtual Memory**:
  - Store some parts of processes into the secondary storage, when there is insufficient physical memory.
  - Load them back into suitable main memory locations as needed.

  → Virtually increase the main memory space!
- This is done automatically by the operating system (OS).
- Application program does not need to know the existence of virtual memory.



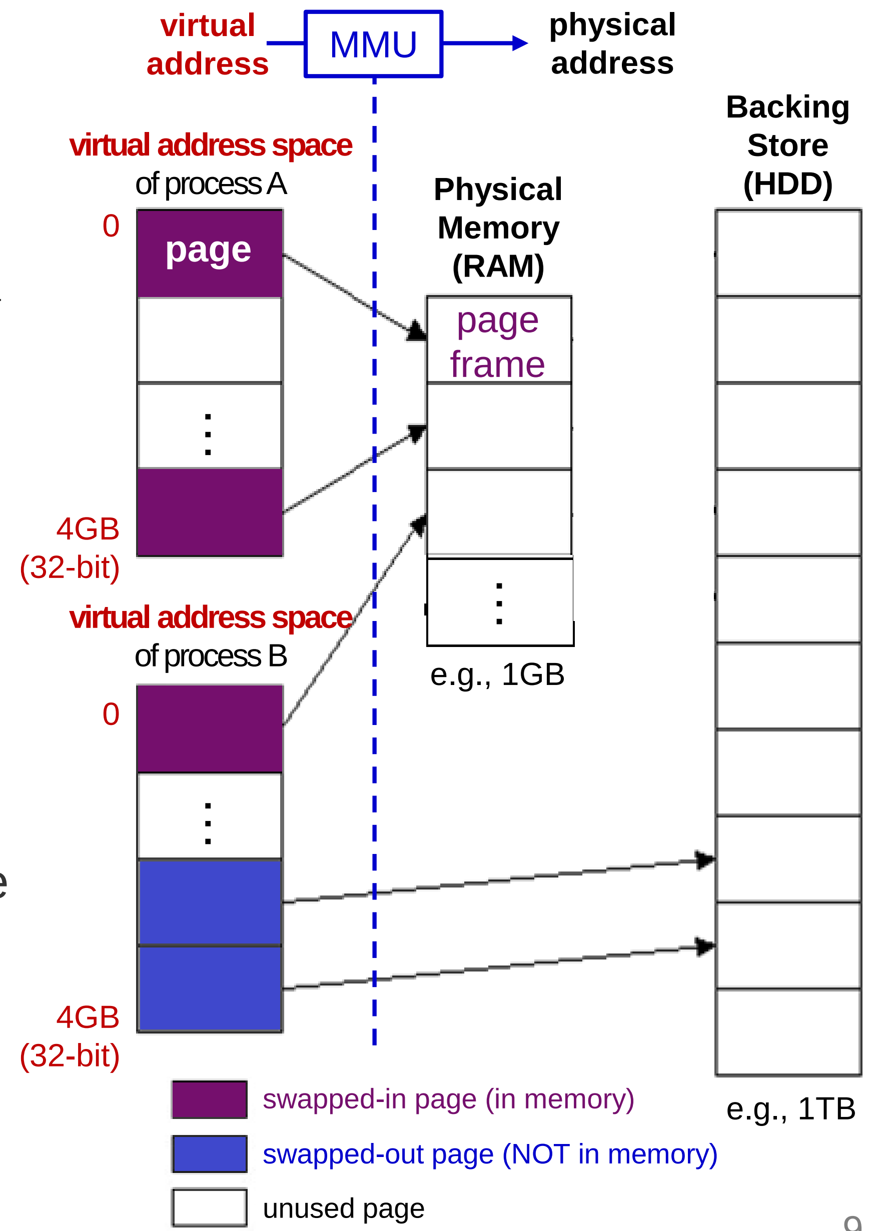
Basic Concept of Virtual Memory (2/2)

- When virtual memory is used: processor uses virtual addresses.
  - If a virtual address refers to a physical memory space: Access the memory content directly.
  - Otherwise: Bring the content from storage to memory for accessing.
    - Swap in & swap out
  - Cache will be checked first based on the **physical address**.
- **Memory Management Unit**
  - A hardware component to translate virtual addresses to physical addresses.



Virtual-to-Physical Address Translation

- Let each process have its own virtual address space.
  - The virtual address space of each process is often set as the maximal addressing space (e.g. 4GB).
- Each process is divided into fixed-sized pages.
  - The page size is ranging from 2KB to 16KB in practice.
    - Too small? Too much time will be spent getting pages from disk.
    - Too big? A large portion of the page may not be used.
- A main memory area that can hold one page is a page frame.



Outline

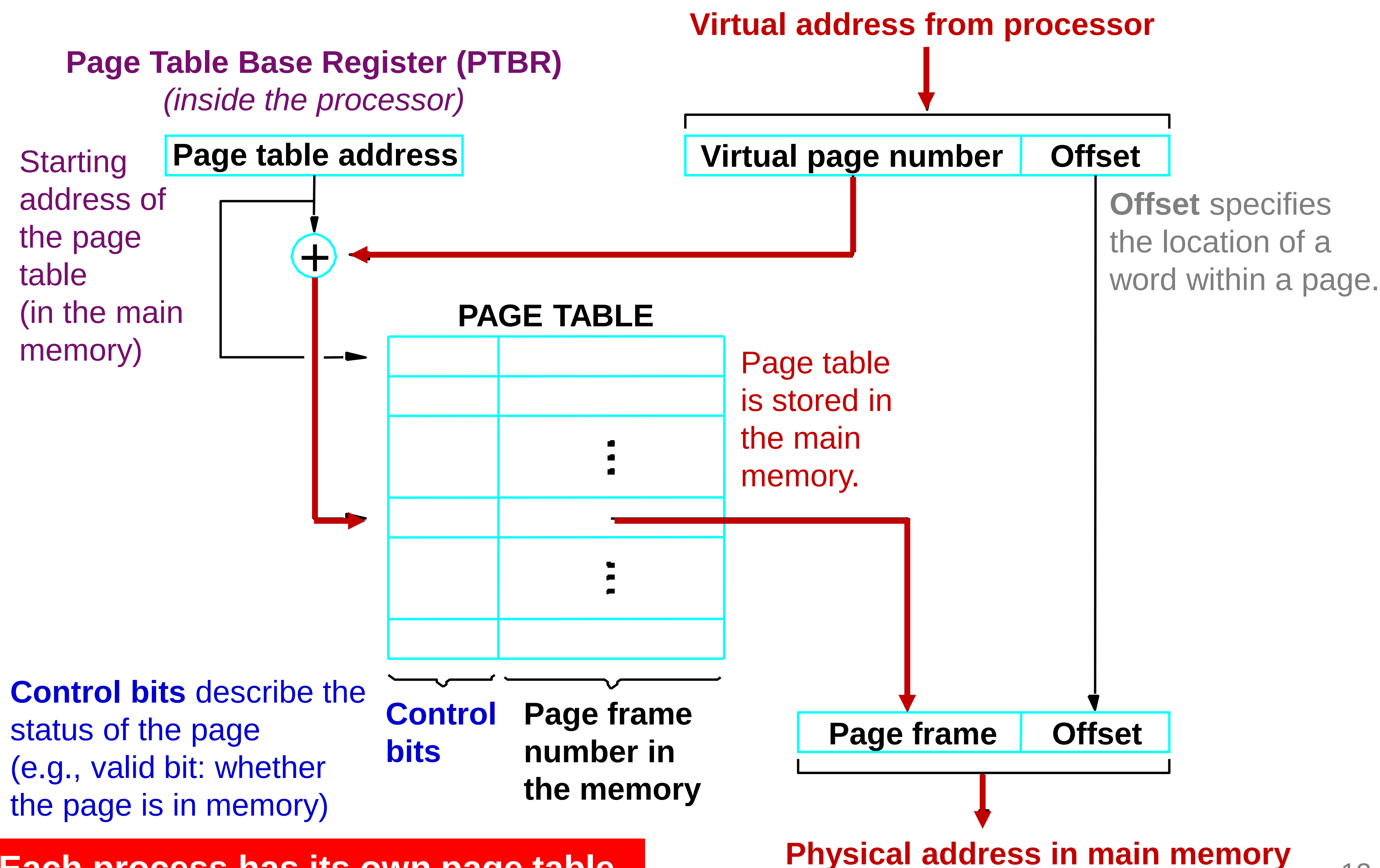
- Why Virtual Memory?
- MMU: Virtual-to-Physical Address Translation
  - Page Table
  - Translation Lookaside Buffer (TLB)
  - Page Fault

Page Table

- **Page Table**: Maintain the virtual-to-physical address translation information for each page.
  - Each process has its own table (virtual address space).
  - Page table is stored in the main memory.
  - Starting address of the page table is stored in a page table base register (PTBR) inside the processor.

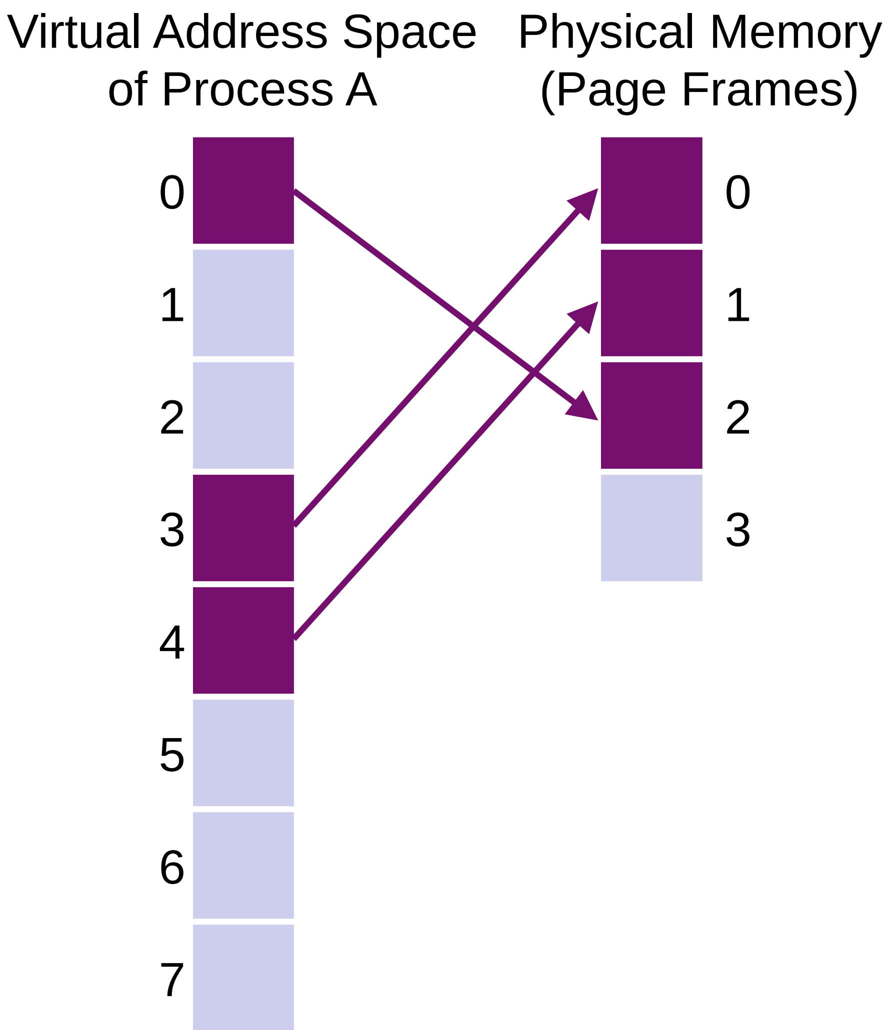
- How to index an entry of the page table in memory?
  - Processor uses virtual addresses.
    - MS (high order) Bits: The virtual page number.
    - LS (low order) Bits: The offset to specify the location of a particular byte (or word) within a page.
  - **Page Table Walk**: Virtual page number + PTBR

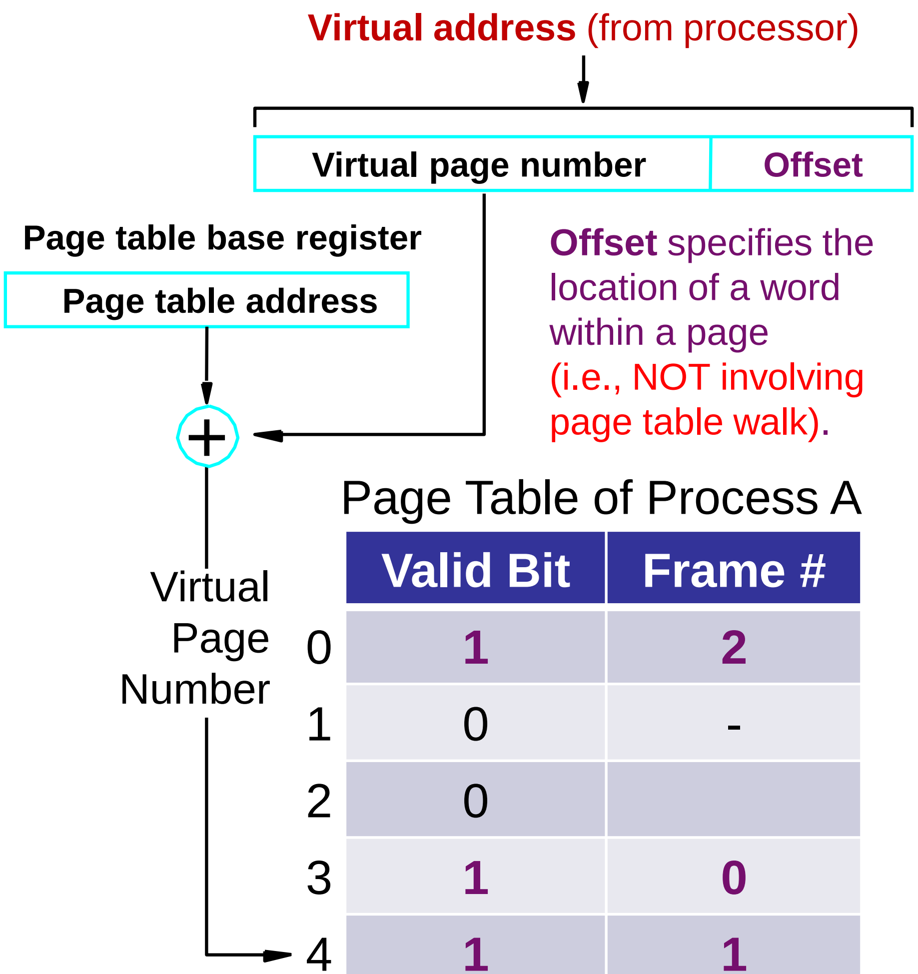
Page Table Walk



Each process has its own page table.

Example of Page Table Walk





Page Table of Process A

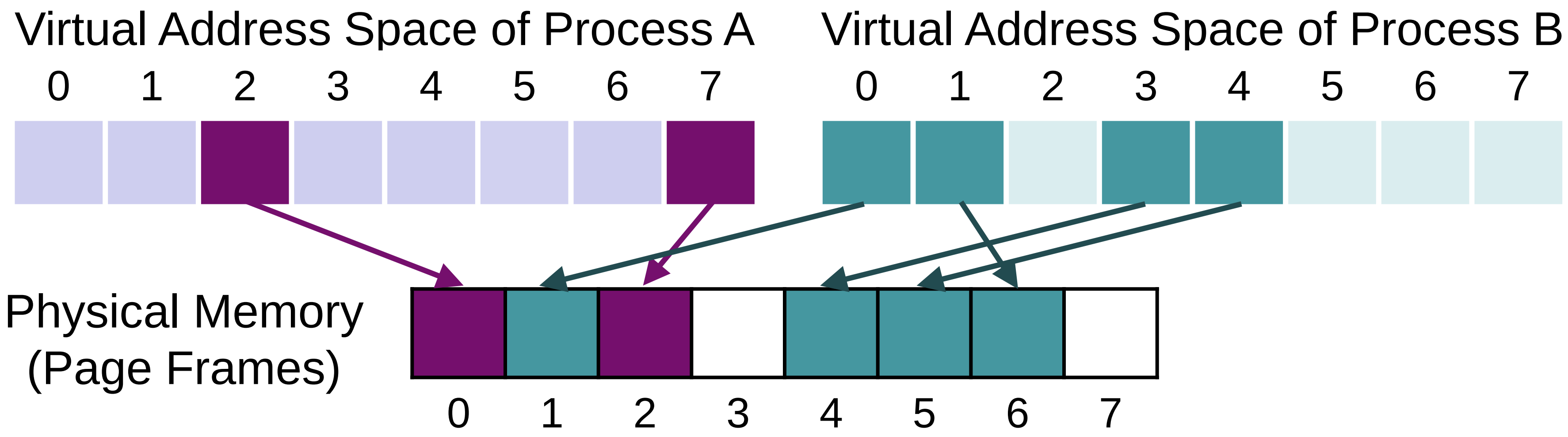
Virtual Page Number	Valid Bit	Frame #
0	**1**	**2**
1	0	-
2	0	
3	**1**	**0**
4	**1**	**1**
5	0	-
6	0	-
7	0	-

Each process has its own page table.

Class Exercise 9.1

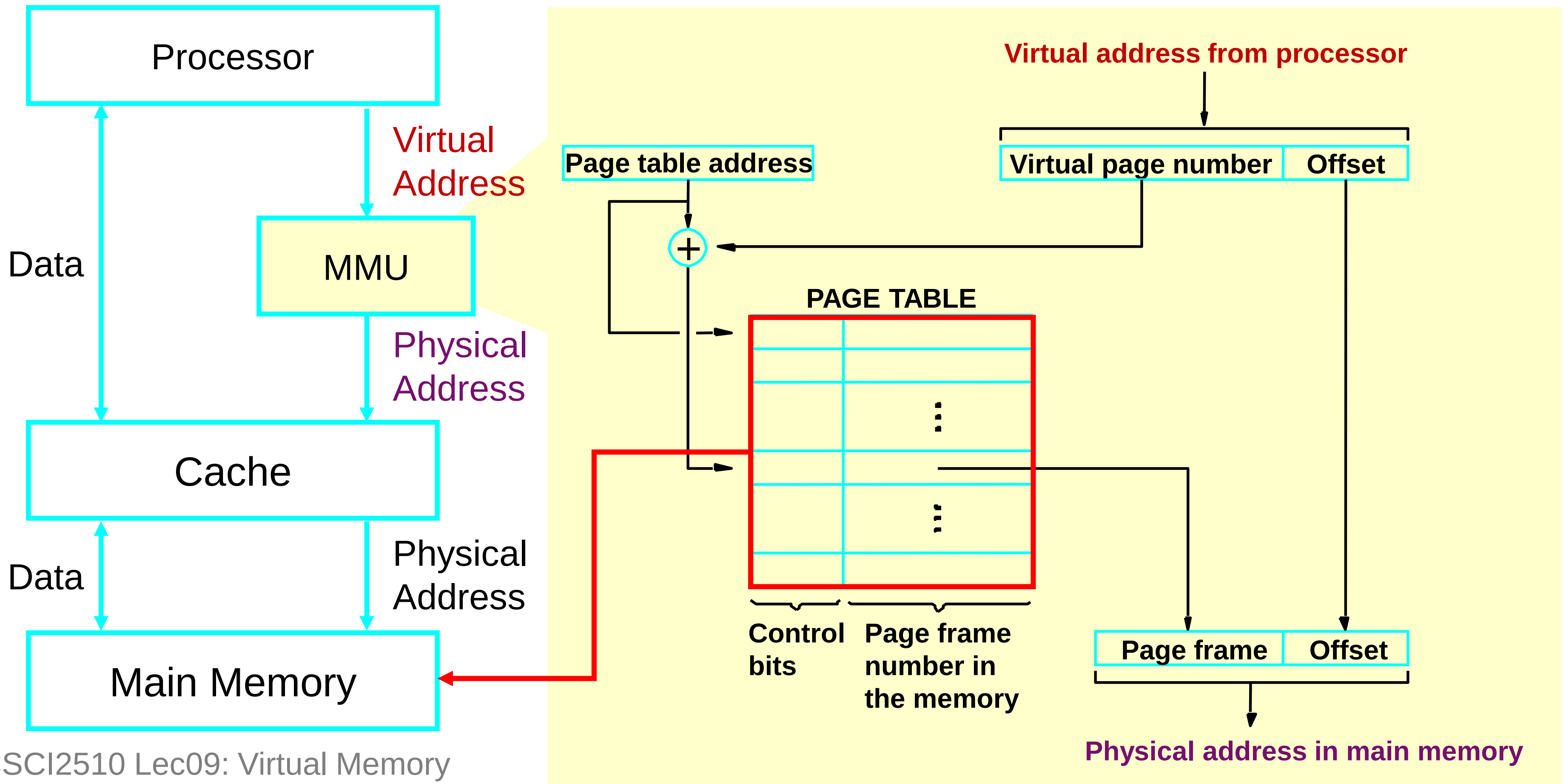
Student ID: ____________ Date:
Name: ________________ ________

- Please draw the page tables for processes A and B:



Something about Page Table

- The page table is used for every read/write access.
- The page table is large and stored in main memory.
- But main memory is slow (compared with cache) …

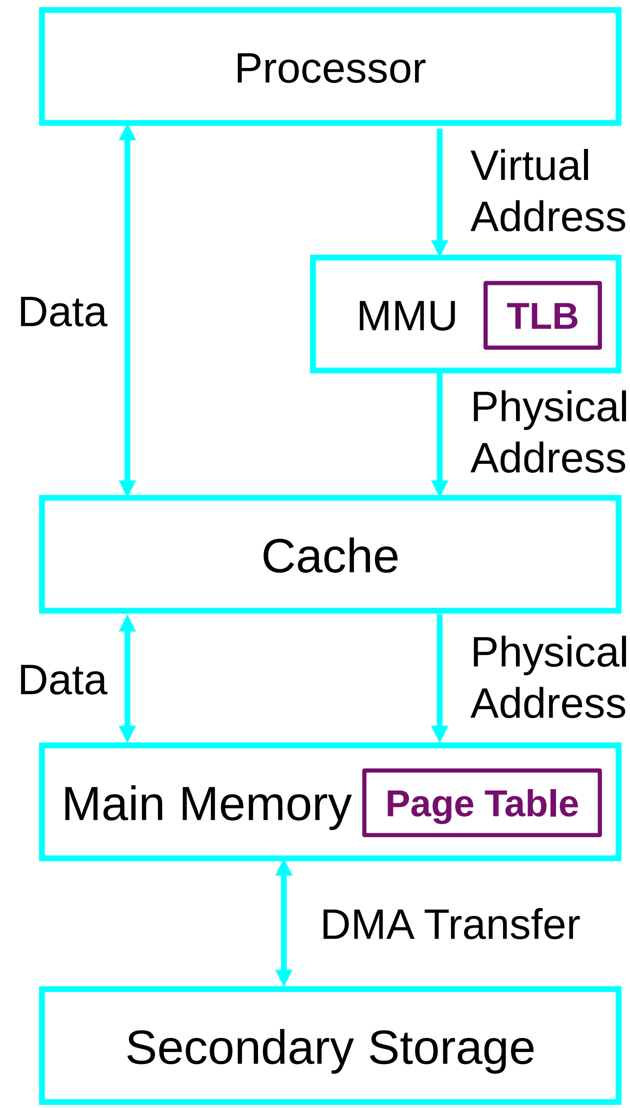


Outline

- Why Virtual Memory?
- MMU: Virtual-to-Physical Address Translation
  - Page Table
  - Translation Lookaside Buffer (TLB)
  - Page Fault

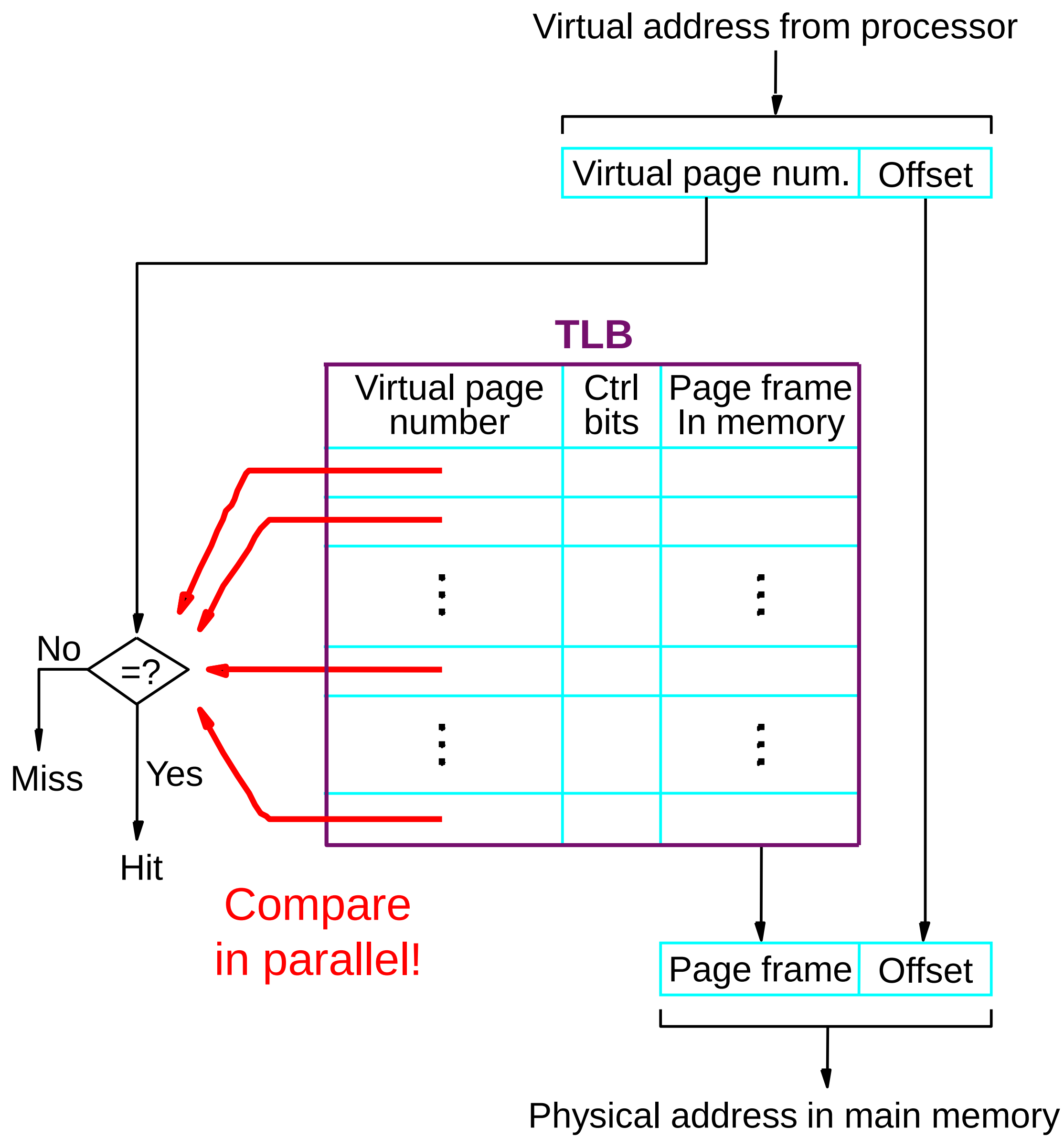
How to Speed Up? Cache of PTEs

- **Translation Lookaside Buffer (TLB)**: A cache of the page table entries (PTEs) in the MMU.
  - Associative or set-associative schemes are normally used.
  - Processor must keep TLB and page table information consistent.
- With TLB, we do **not** have to look up the page table for every memory accesses!



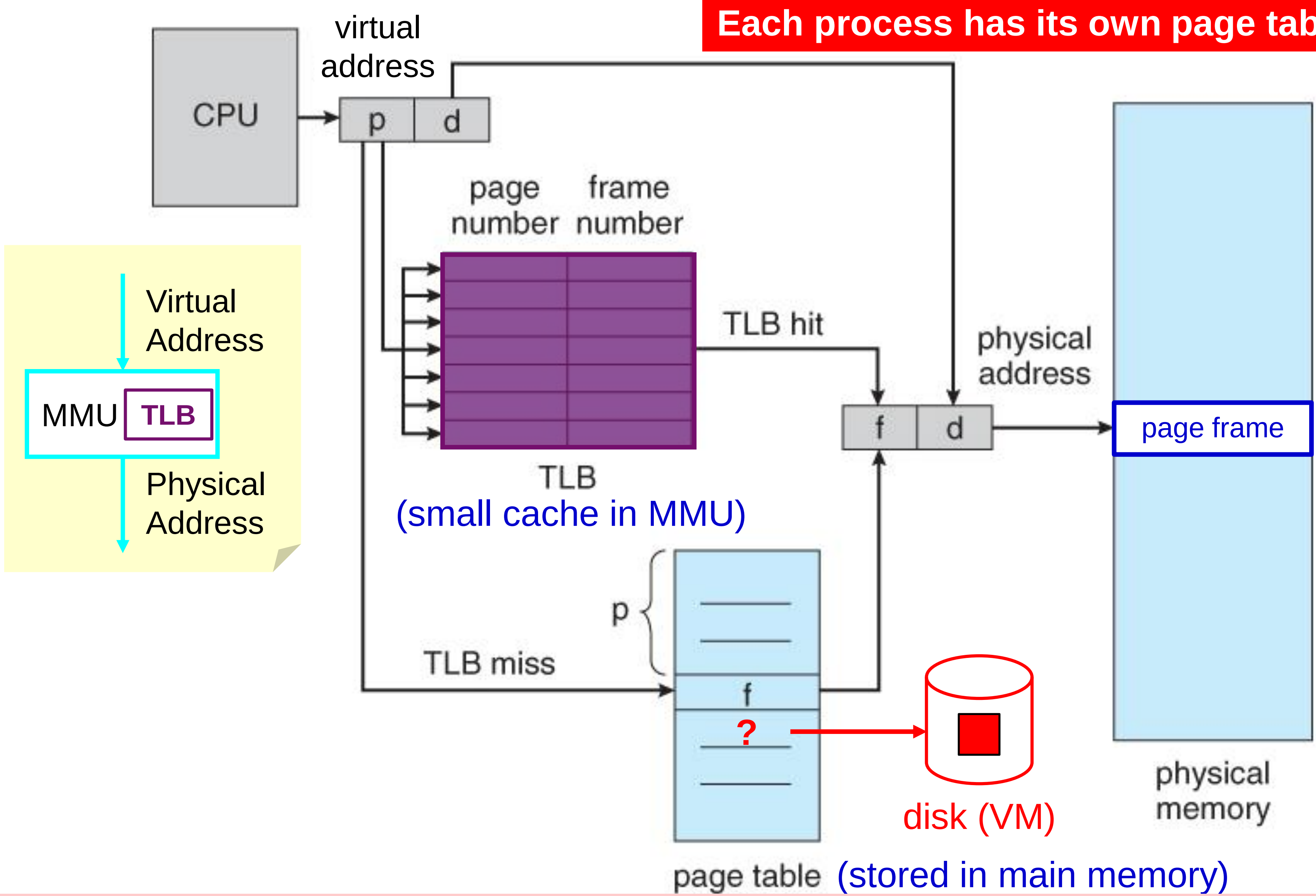
Translation Lookaside Buffer (TLB)

- Given a virtual address, MMU first looks up **TLB**.
- If available (**hit**):
  - Using the cached PTE in **TLB**.
- Otherwise (**miss**):
  - Obtaining PTE from the **page table**.
    - Which is stored in the main memory.
  - Updating **TLB**.



Example of TLB

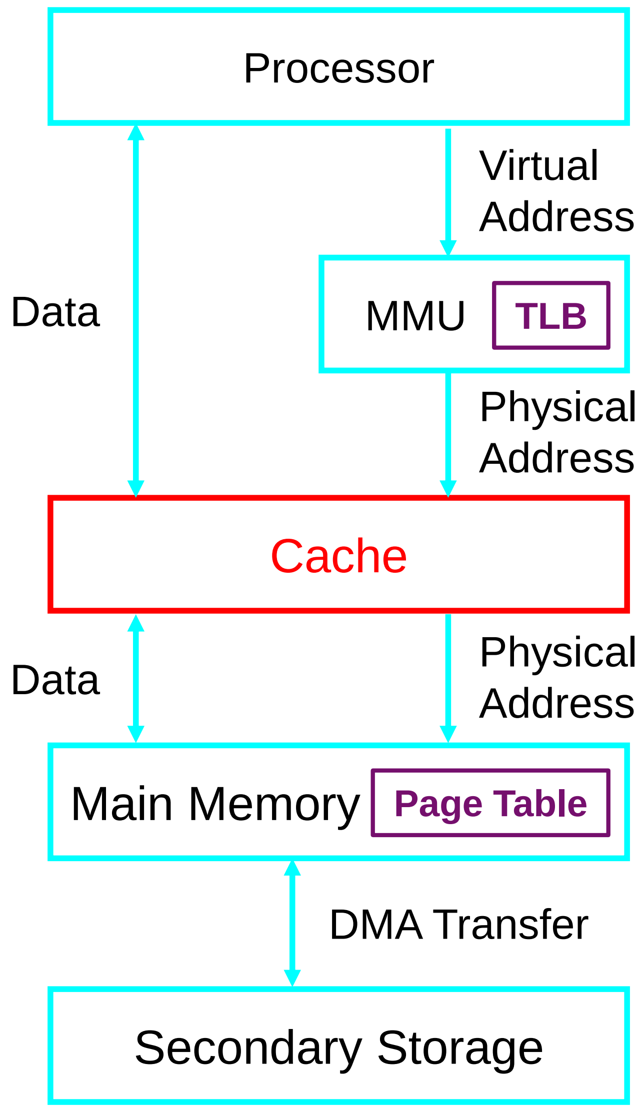
Each process has its own page table.



Question: What if the requested page is not in memory?

Class Exercise 9.2

- Please elaborate the difference between TLB and cache.

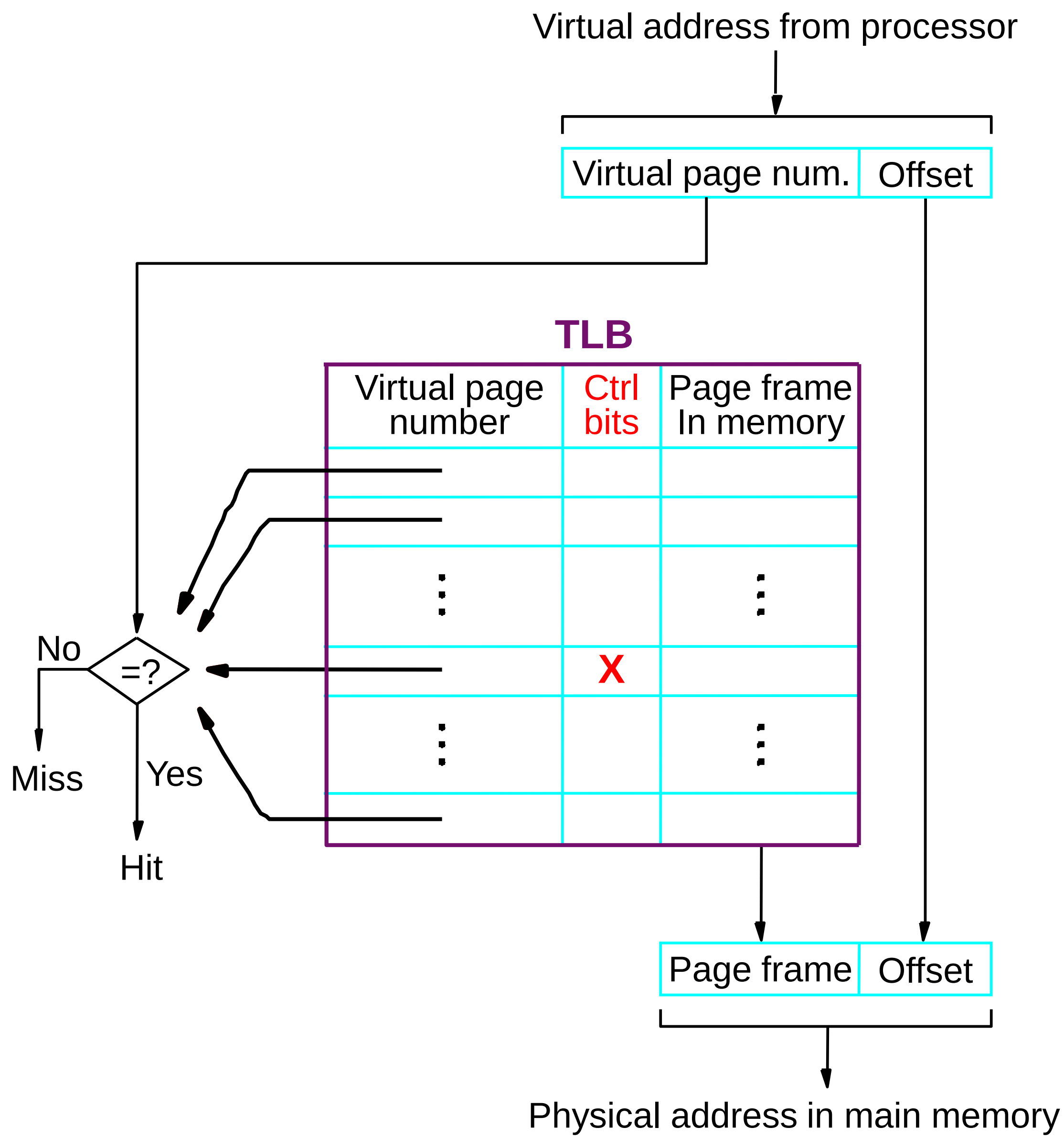


Outline

- Why Virtual Memory?
- MMU: Virtual-to-Physical Address Translation
  - Page Table
  - Translation Lookaside Buffer (TLB)
  - Page Fault

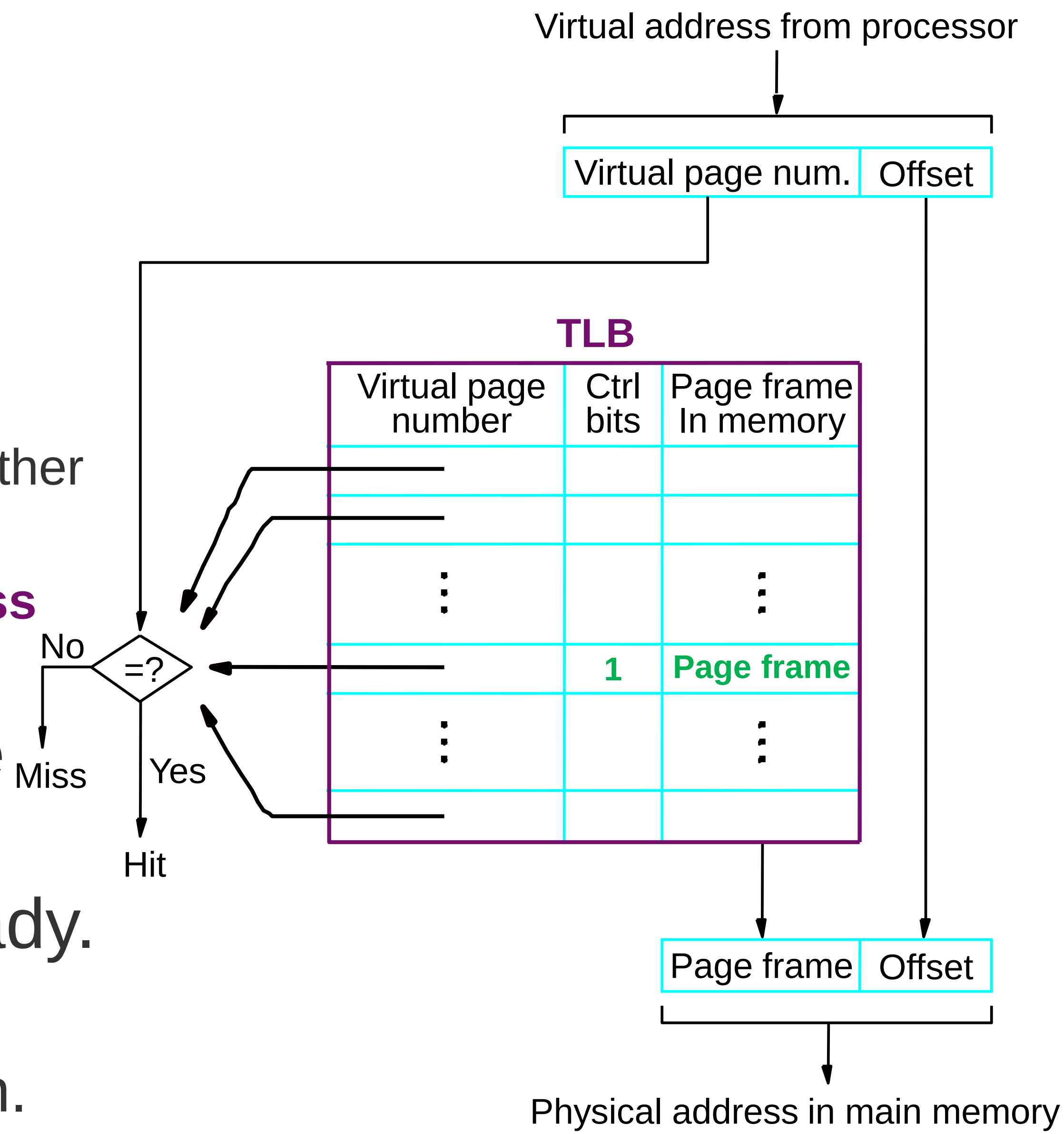
Page Fault (1/2)

- **Page Fault**: If the requested page is **not** in memory.
  - How to know? Checking the control bits in the page table entry (PTE).
  - MMU generates a page fault.
  - The process is suspended.
  - The control gives to the operating system (OS).

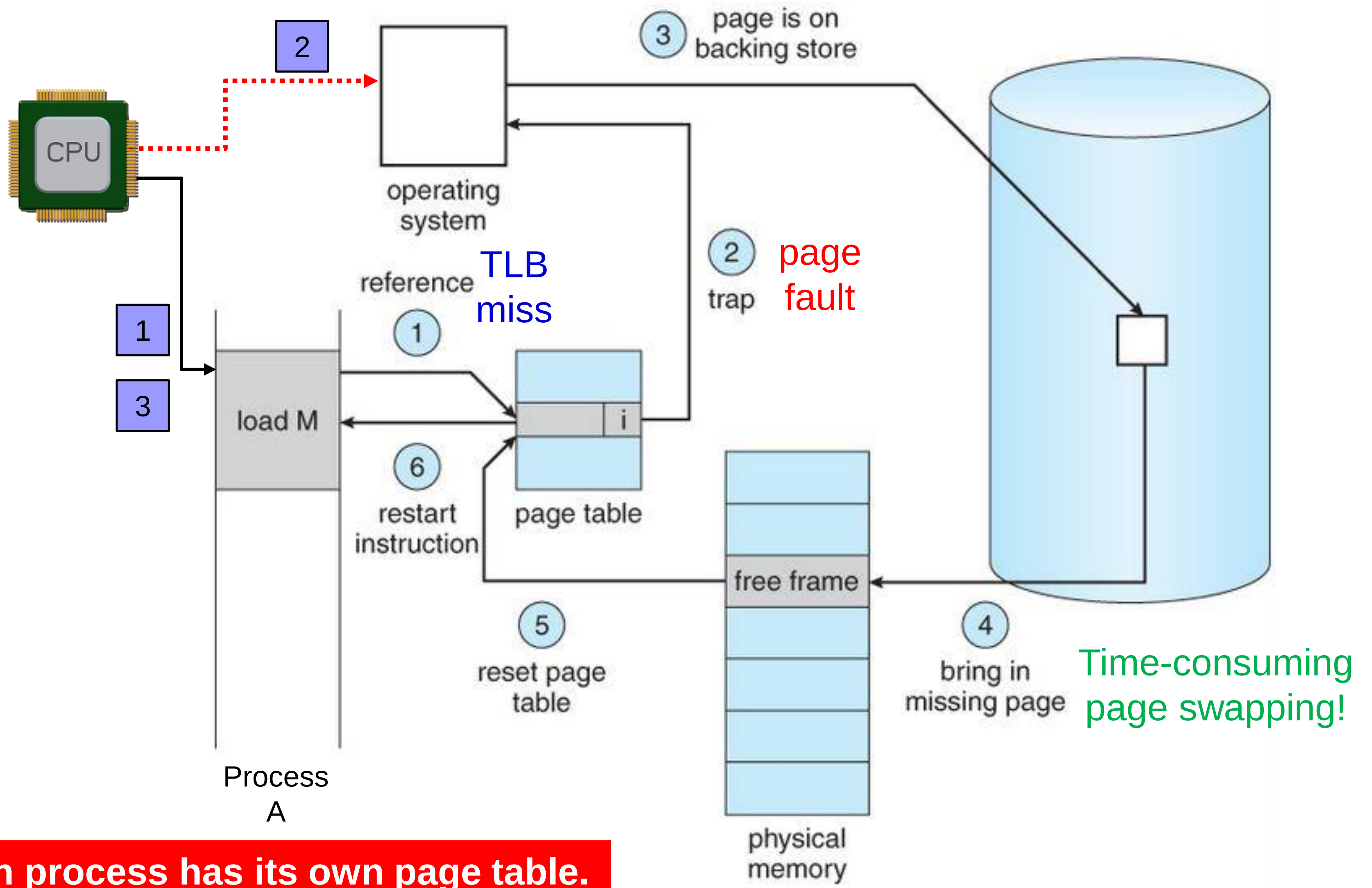


Page Fault (2/2)

- **OS** must **swap** the requested page from disk into memory.
  - **Page swapping** may take a long time.
    - OS may schedule another process to run.
    - **Direct memory access (DMA)** can help.
- **OS** must **resume** the suspended process when the **page** is ready.
  - It re-executes the suspended instruction.



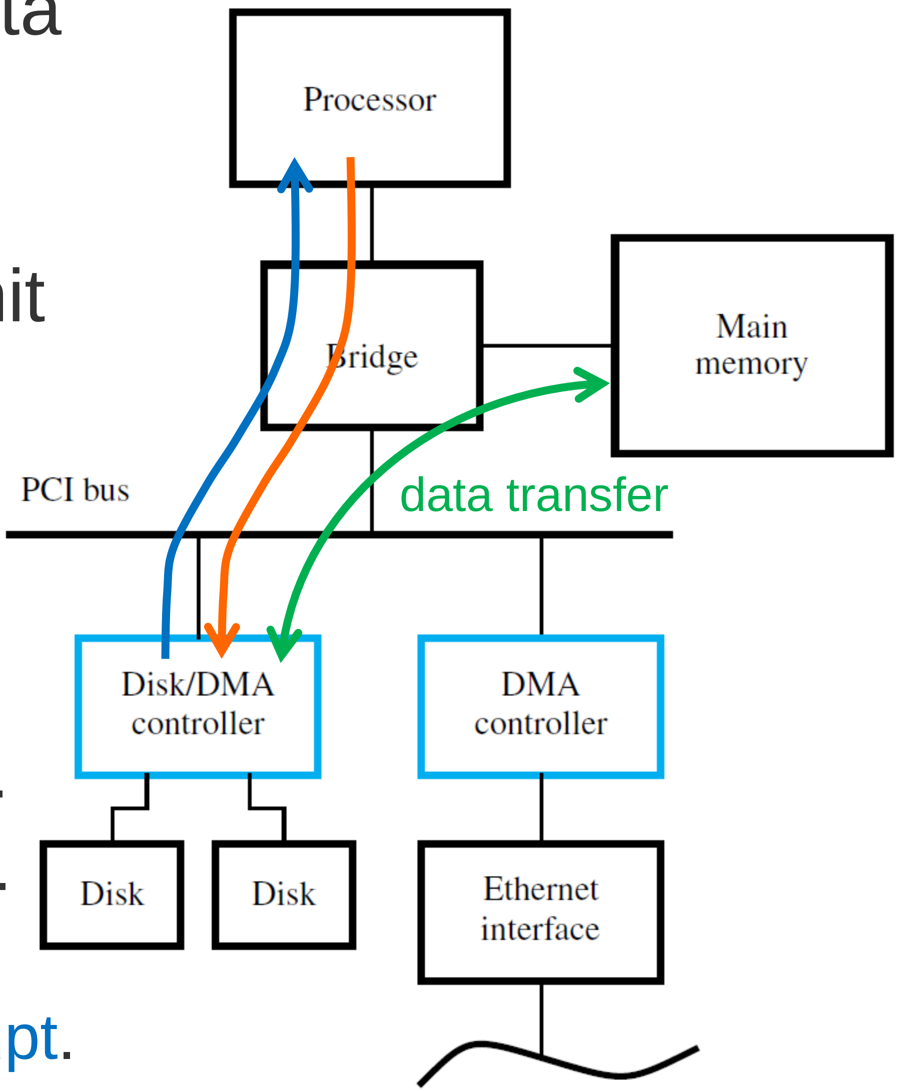
Steps in Handling a Page Fault



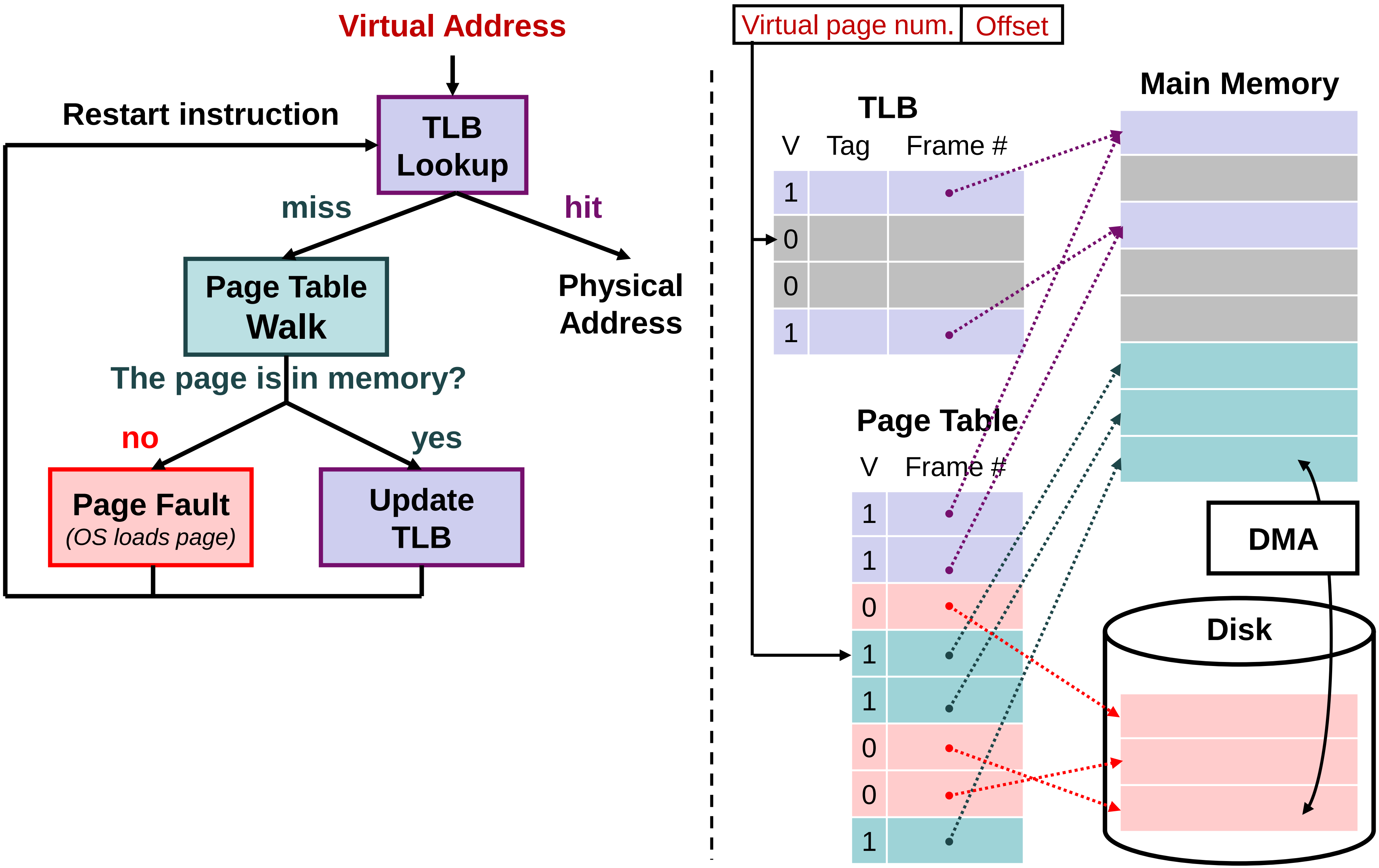
Each process has its own page table.

Direct Memory Access (DMA)

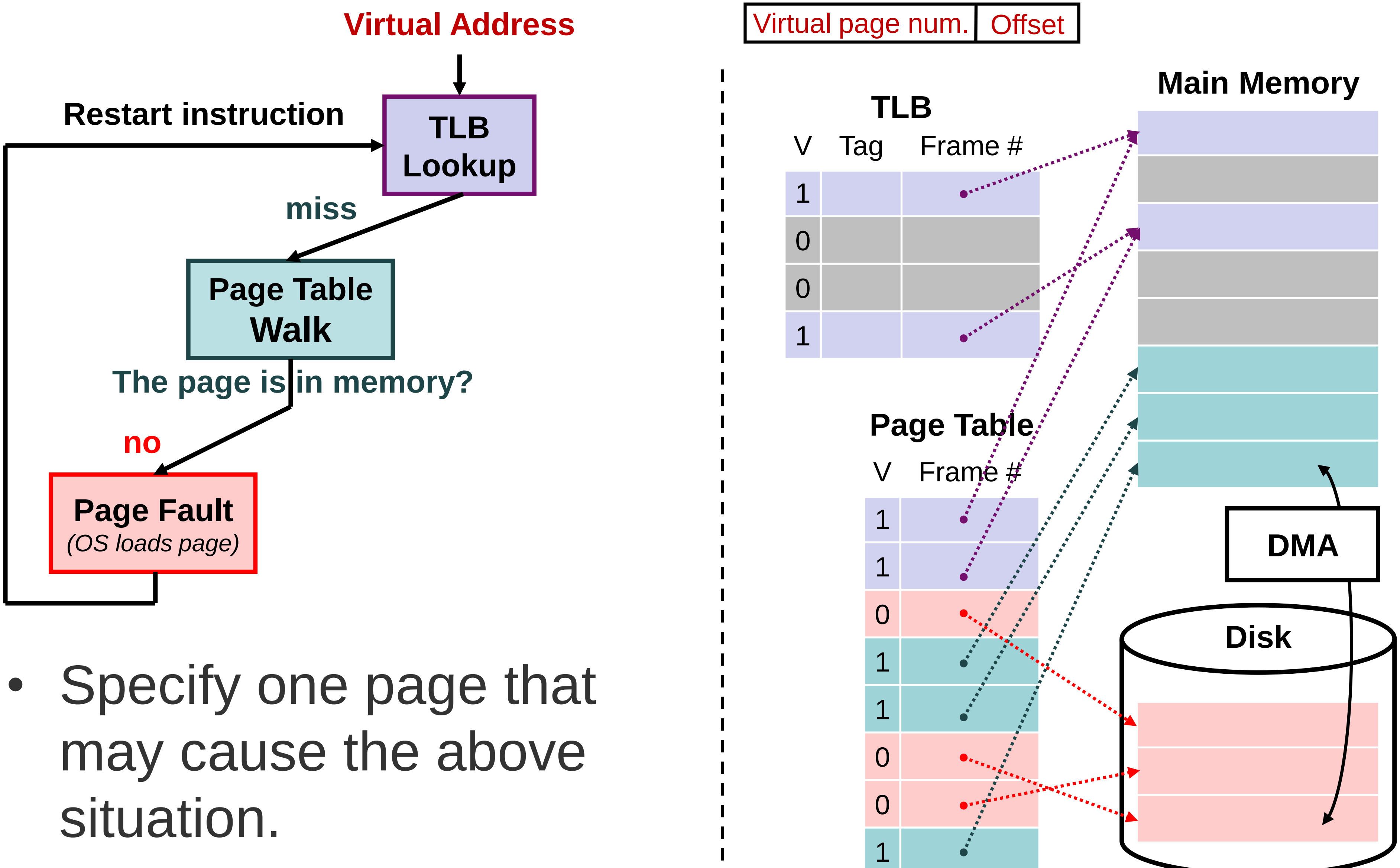
- **Goal**: Transfer blocks of data directly between the main memory and I/O devices.
- **DMA** is a special control unit to manage such transfers.
  - Without involving CPU.
  - Under the control of OS.
- **DMA Operations**:
  - Processor initiates a transfer.
  - DMA proceeds the operation.
  - When finished, DMA informs the CPU by raising an interrupt.



Putting All Pieces Together



Class Exercise 9.3



- Specify one page that may cause the above situation.

Summary

- Why Virtual Memory?
- MMU: Virtual-to-Physical Address Translation
  - Page Table
  - Translation Lookaside Buffer (TLB)
  - Page Fault