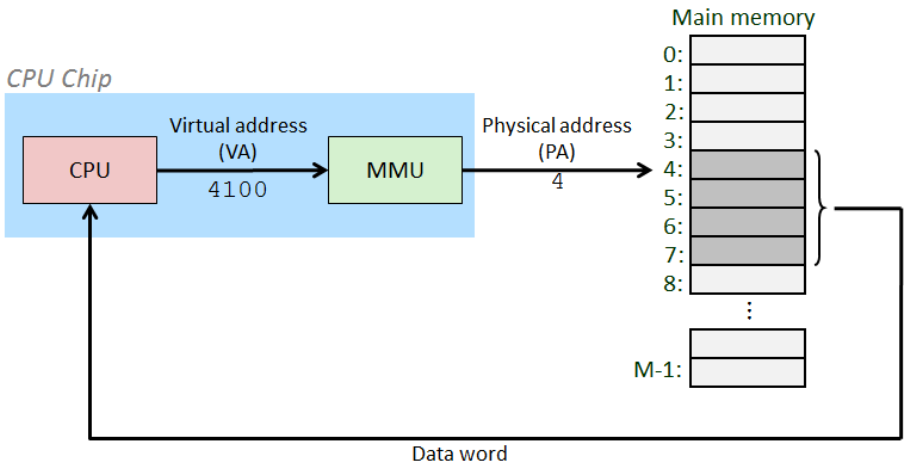
The Memory Hierarchy

Virtual Memory Concepts



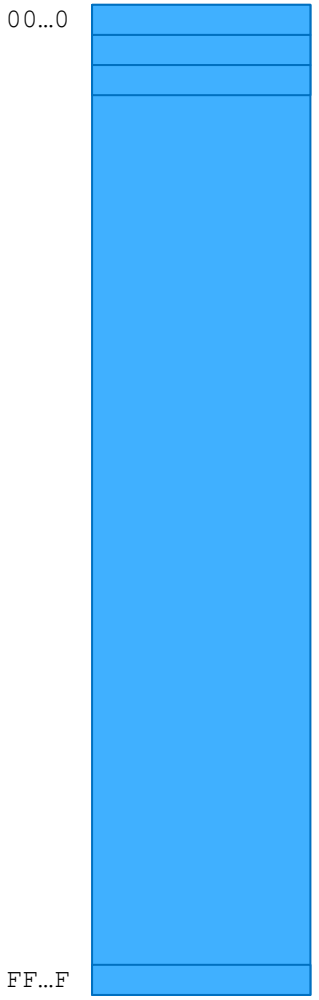
Virtual Memory: Theory

- **Virtual Memory(VM): Overview and motivation**
- VM as a tool for caching
- VM as a tool for memory management
- VM as a tool for memory protection
- Address translation
- Allocation

Acknowledgement: slides based on the cs:app2e material

Memory Up Until Now

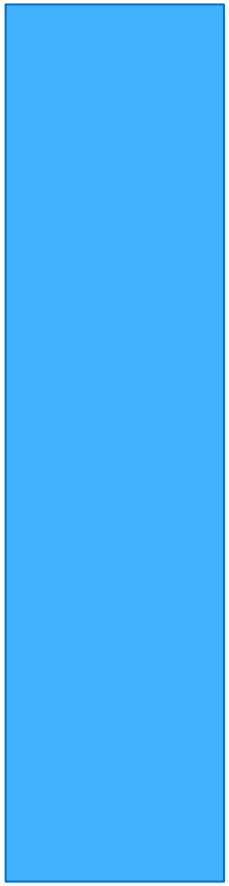
- Programs refer to virtual memory addresses
  - mov (%ebx), %eax
  - conceptually a very large array of bytes
  - implemented with a hierarchy of different memory types
  - system provides private address space to each process
- Allocation: Compiler and run-time system
  - where different program objects should be stored
  - all objects allocated within a single virtual address space
- Why would we need virtual memory?



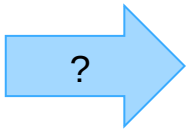
CSE 컴퓨터공학부 Department of Computer Science & Engineering

Problem 1: How does everything fit?

virtual memory:
64-bit addresses,
16 exabytes



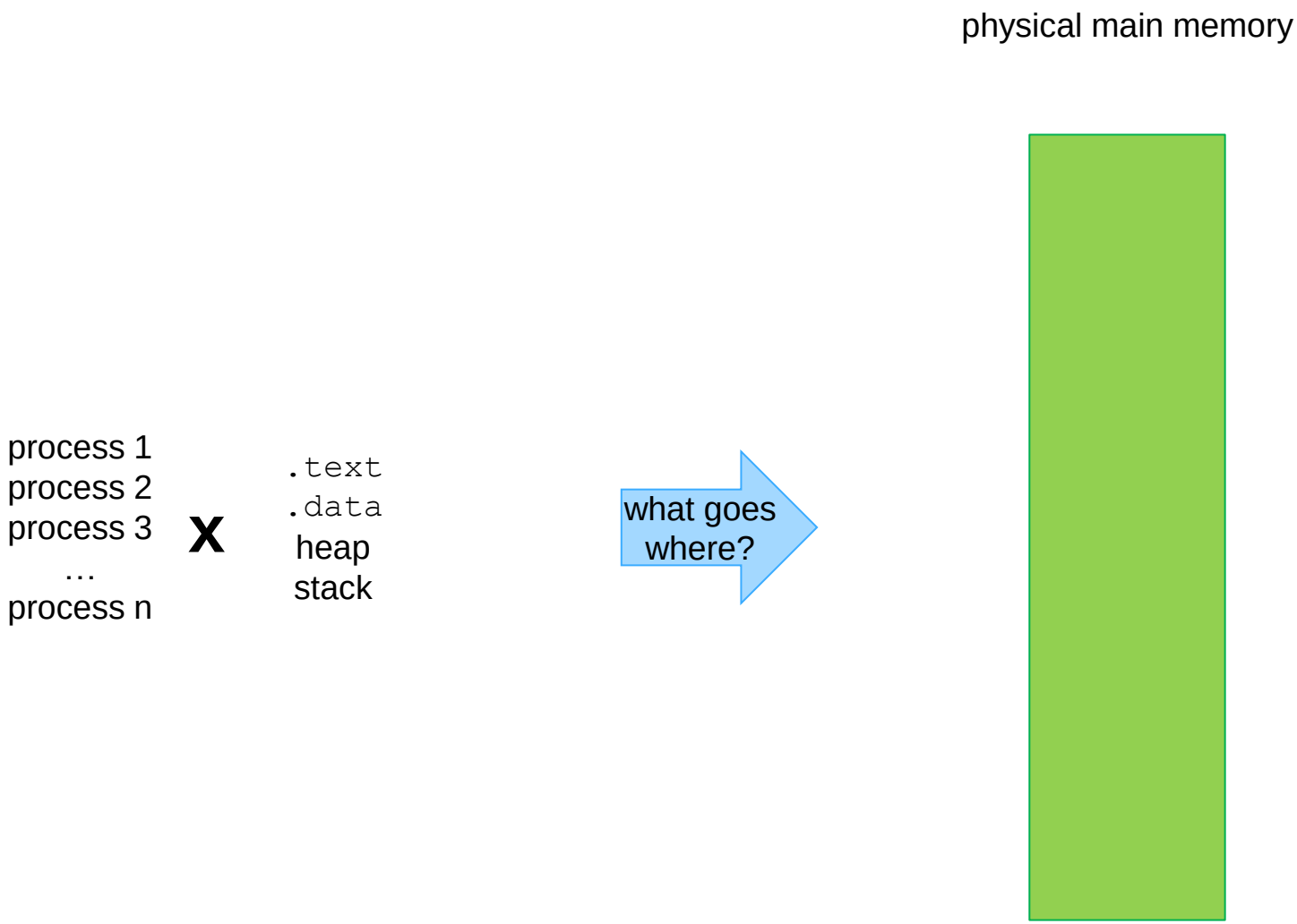
physical memory:
a few gigabytes



…and there are many processes

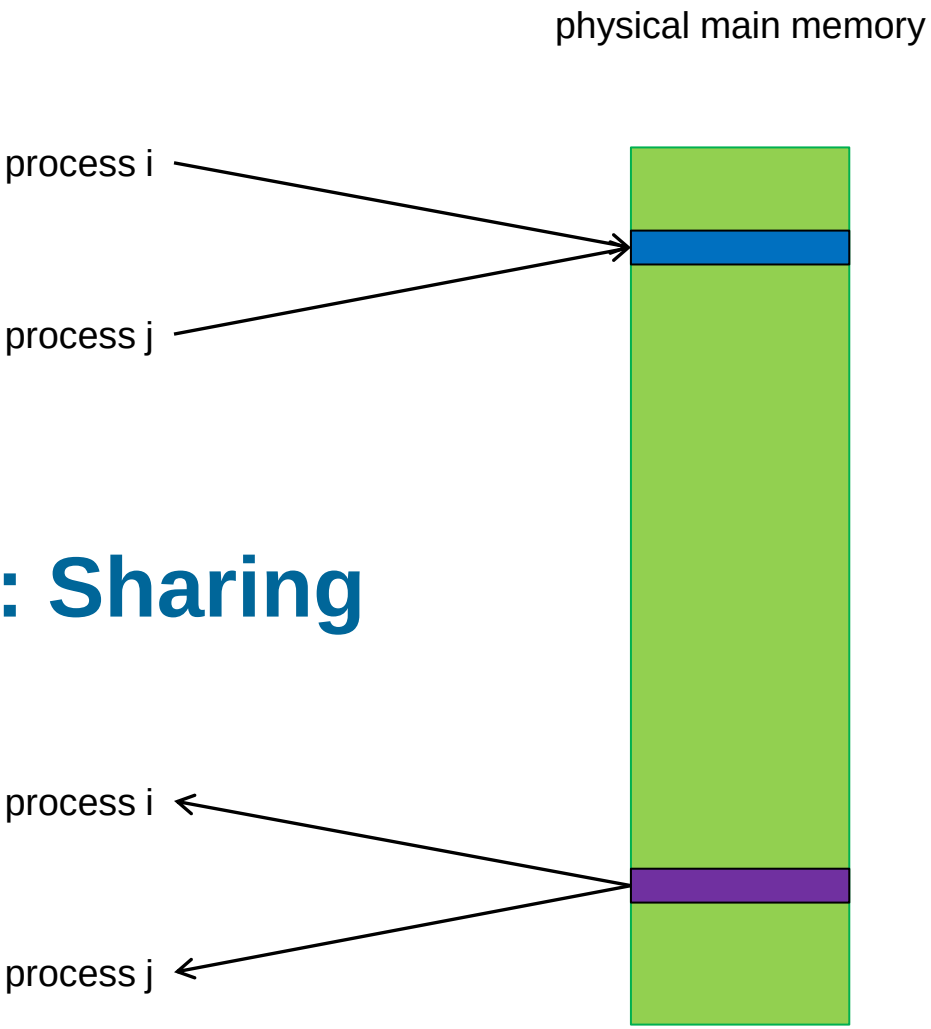
CSE 컴퓨터공학부
Department of Computer Science & Engineering

Problem 2: Memory Management



CSE 컴퓨터공학부 Department of Computer Science & Engineering

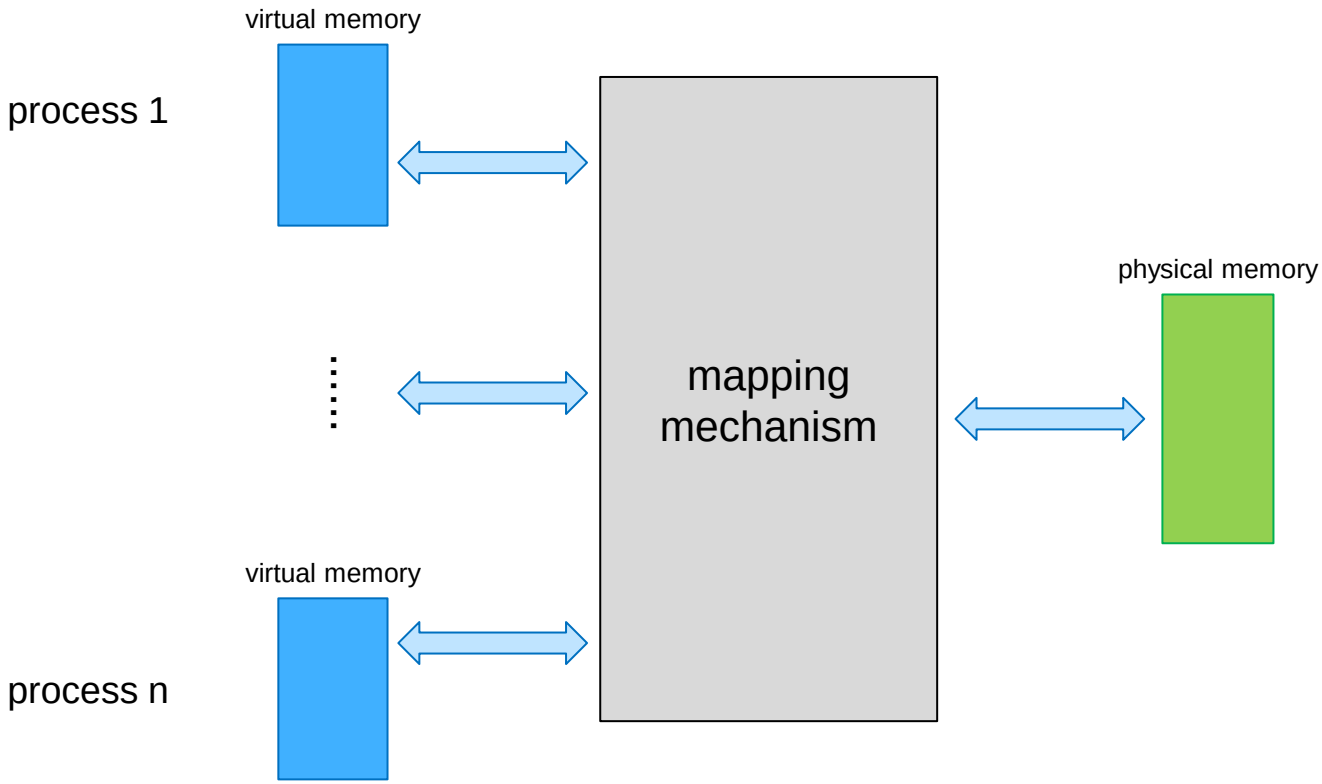
Problem 3: Protection



Problem 4: Sharing

CSE 컴퓨터공학부 Department of Computer Science & Engineering

Solution: Indirection



- Each process gets its own private memory space
- Solves all previous problems

CSE 컴퓨터공학부 Department of Computer Science & Engineering

Address Spaces

- Linear address space: Ordered set of contiguous non-negative integer addresses:
  {0, 1, 2, 3 … }

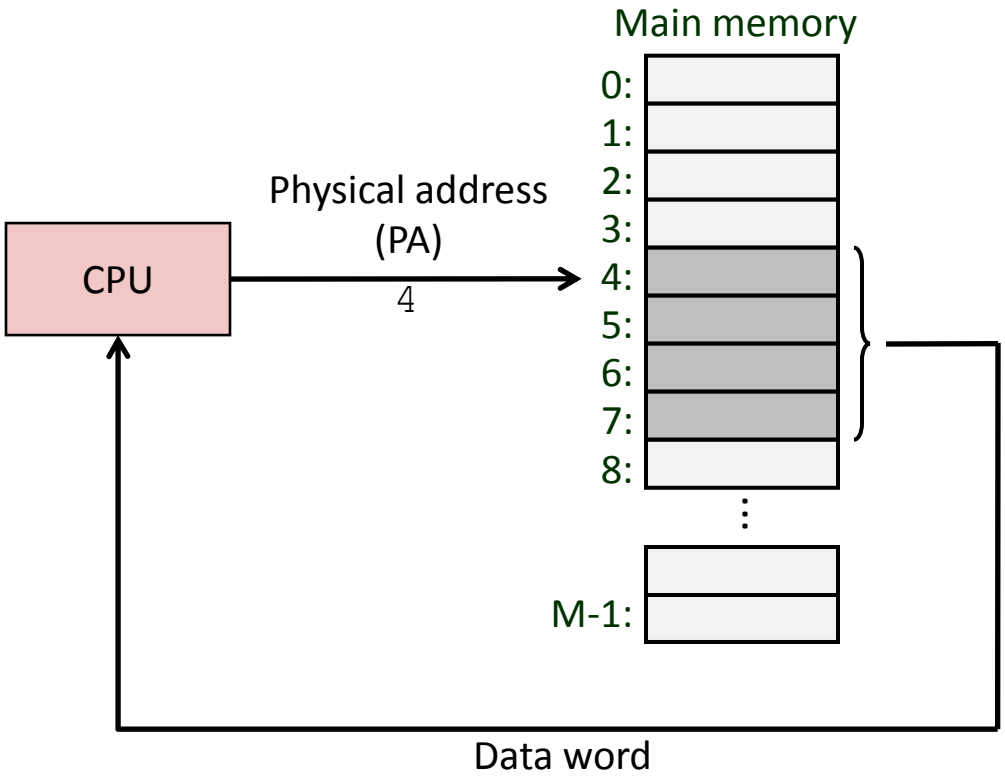
- Virtual address space: Set of $N = 2^n$ virtual addresses
  {0, 1, 2, 3, …, N-1}

- Physical address space: Set of $M = 2^m$ physical addresses
  {0, 1, 2, 3, …, M-1}

- Clean distinction between data (bytes) and their attributes (addresses)
- Each object can now have multiple addresses
- Every byte in main memory:
  one physical address, one (or more) virtual addresses

CSE 컴퓨터공학부 Department of Computer Science & Engineering

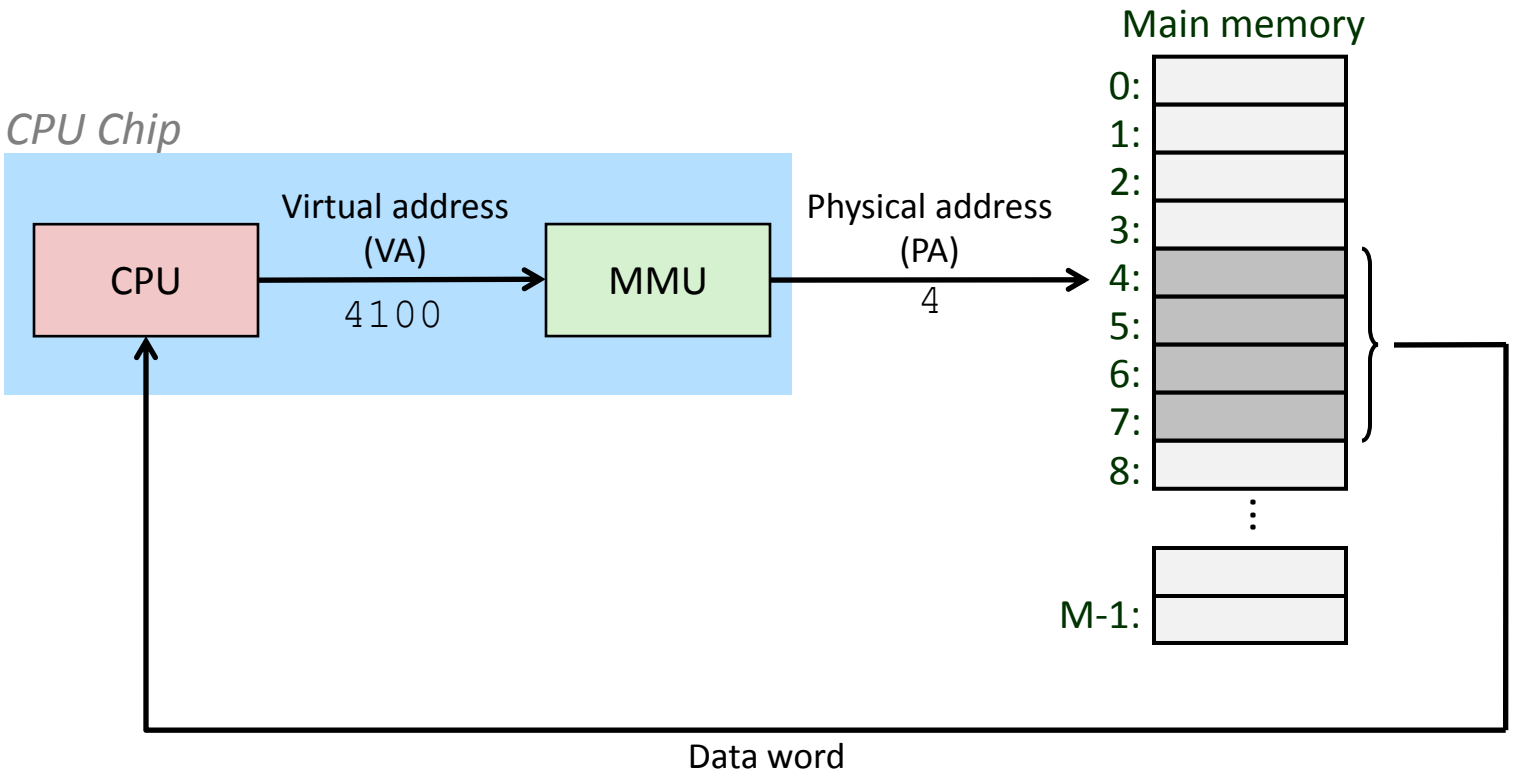
A System Using Physical Addressing



- Still used in “simple” systems like embedded microcontrollers in devices like cars, elevators, and digital picture frames

CSE 컴퓨터공학부 Department of Computer Science & Engineering

A System Using Virtual Addressing



- Used in all modern servers, desktops, and laptops
- One of the great ideas in computer science

CSE 컴퓨터공학부 Department of Computer Science & Engineering

Why Virtual Memory (VM)?

- Efficient use of limited main memory
  - uses DRAM as a cache for the parts of a virtual address space
    - some non-cached parts stored on disk
    - some (unallocated) and non-cached parts stored nowhere
  - keep only active areas of virtual address space in memory
    - transfer data back and forth as needed

- Simplified memory management for programmers
  - Each process gets the same full, uniform linear address space

- Isolated address spaces
  - One process can't interfere with another's memory
    - because they operate in different address spaces
  - User program cannot access privileged kernel information
    - different sections of address spaces have different permissions

CSE 컴퓨터공학부 Department of Computer Science & Engineering

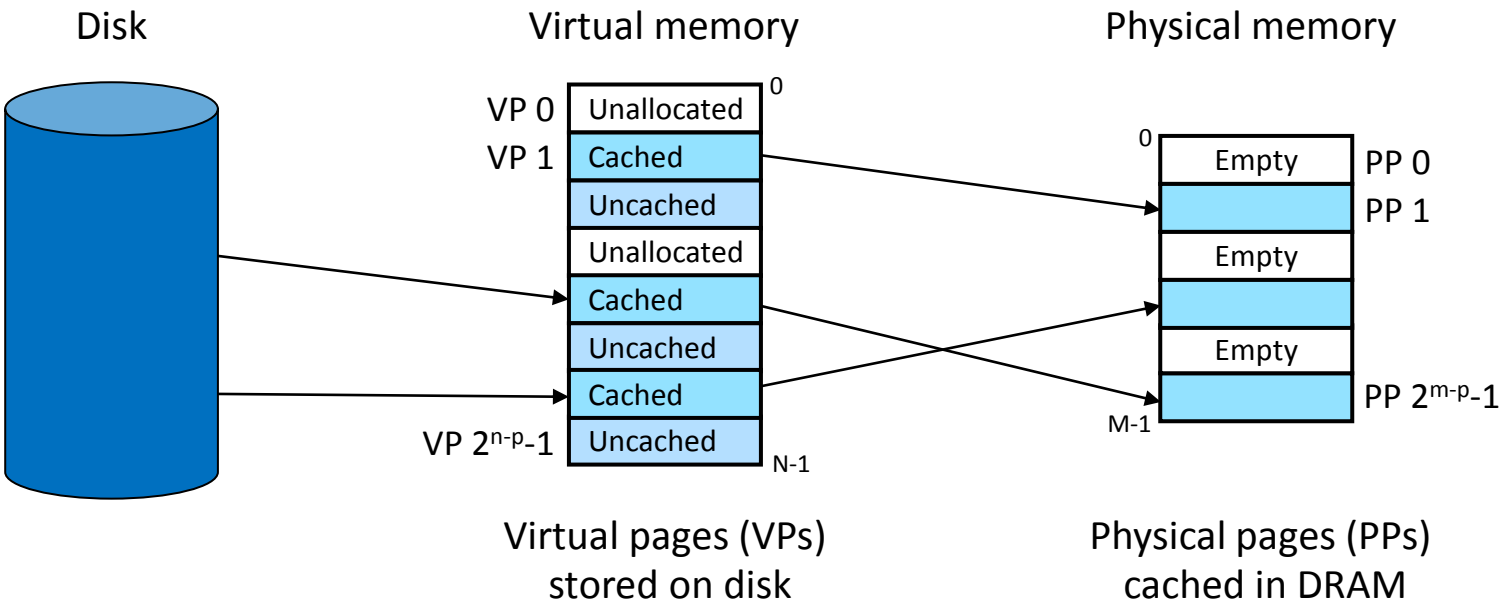
Virtual Memory: Theory

- Virtual Memory(VM): Overview and motivation
- **VM as a tool for caching**
- VM as a tool for memory management
- VM as a tool for memory protection
- Address translation
- Allocation

CSE 컴퓨터공학부 Department of Computer Science & Engineering

VM as a Tool for Caching

- Virtual memory is an array of N contiguous bytes stored on disk
  - conceptually: stored somewhere on disk
- Physical memory = cache for allocated virtual memory
- Cache blocks are called pages (size is P = 2^p bytes)



CSE 컴퓨터공학부 Department of Computer Science & Engineering

DRAM Cache Organization

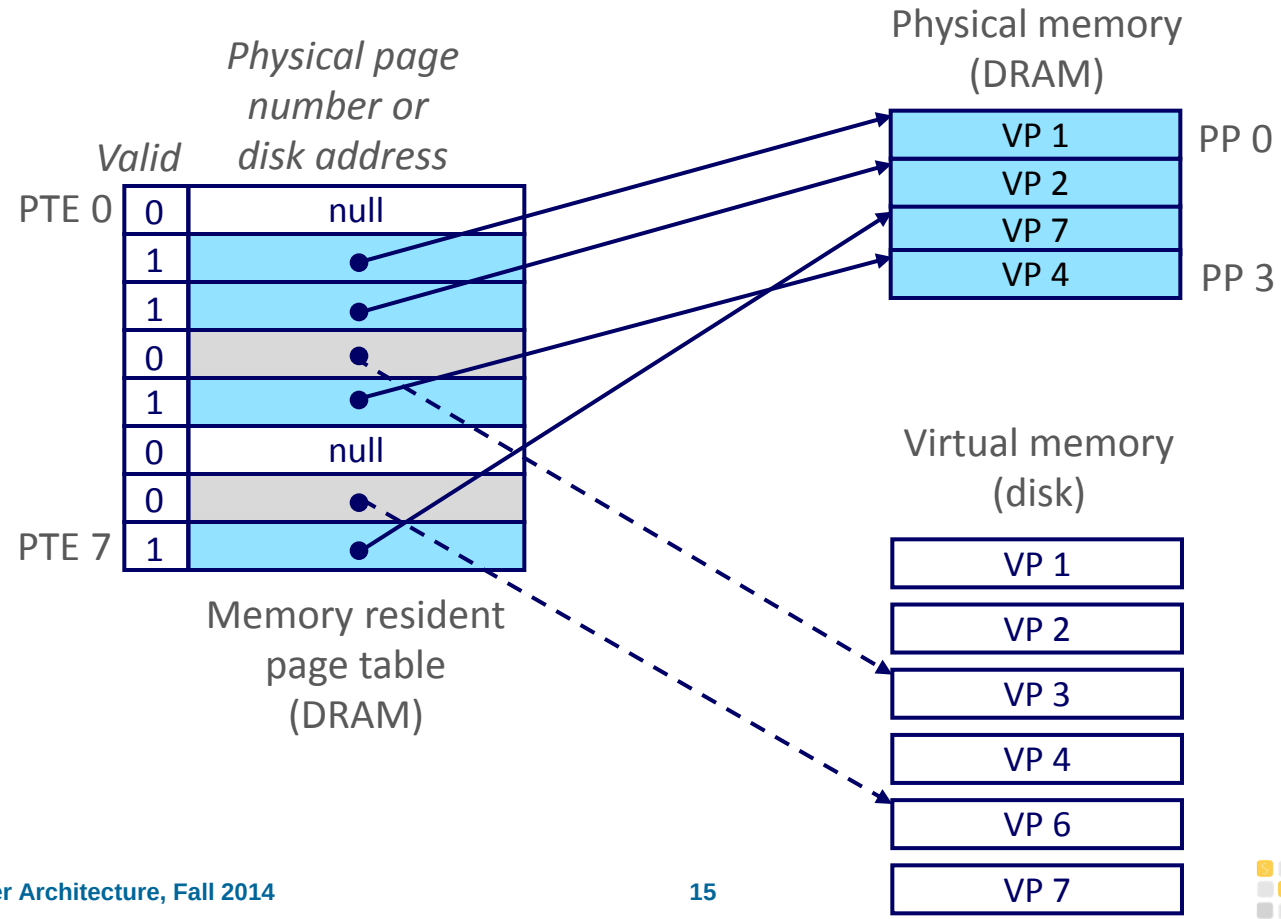
- DRAM cache organization driven by the enormous miss penalty
  - DRAM is about 10x slower than SRAM
  - Disk is about 10,000x slower than DRAM

- Consequences
  - Large page (block) size: typically 4-8 KB, sometimes 4 MB
  - Fully associative
    - Any VP can be placed in any PP
    - Requires a “large” mapping function – different from CPU caches
  - Highly sophisticated, expensive replacement algorithms
    - Too complicated and open-ended to be implemented in hardware
  - Write-back rather than write-through

CSE 컴퓨터공학부 Department of Computer Science & Engineering

Address Translation: Page Tables

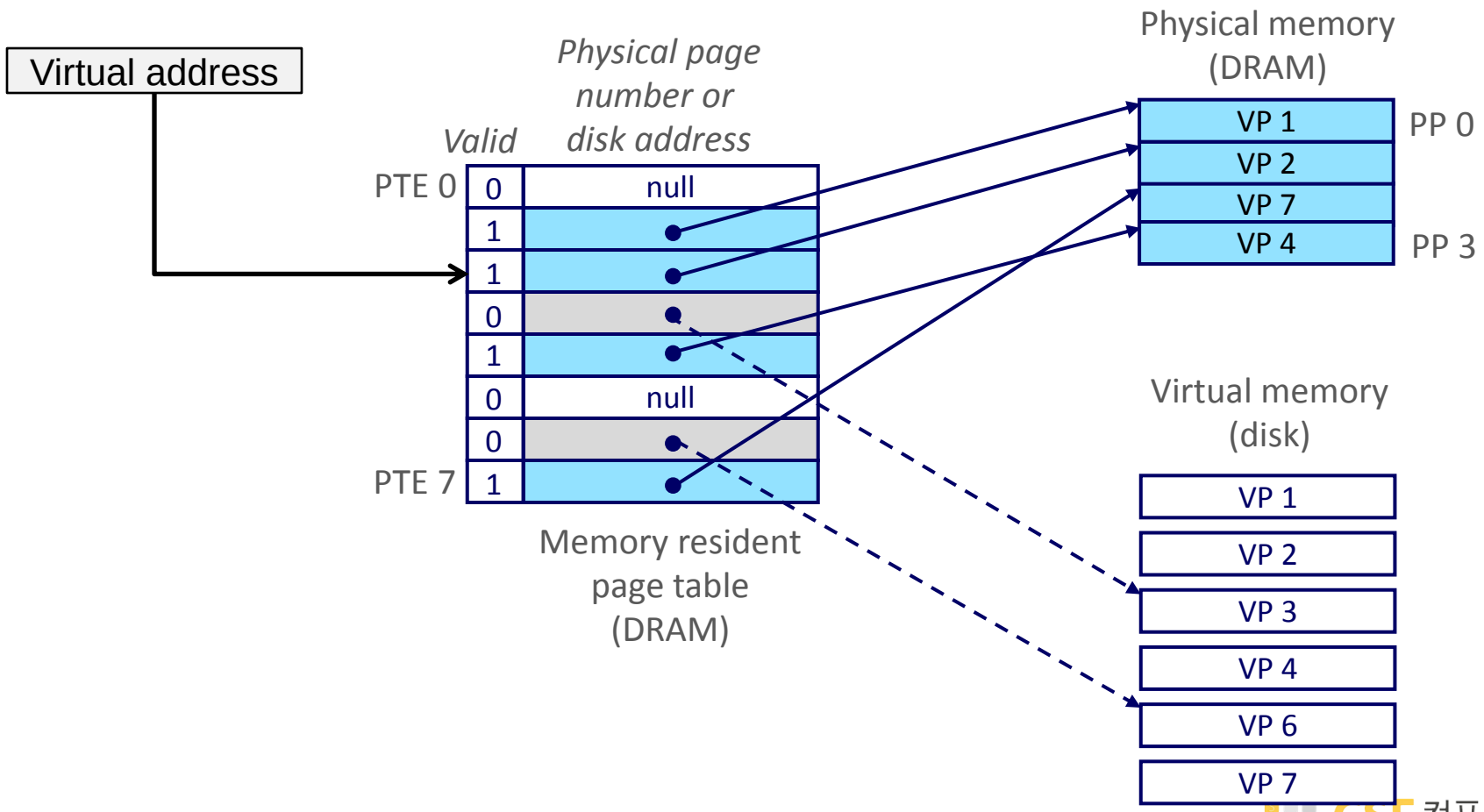
- A **page table** is an array of page table entries (PTEs) that maps virtual pages to physical pages.
  - Per-process kernel data structure in DRAM



CSE 컴퓨터공학부 Department of Computer Science & Engineering

Page Hit

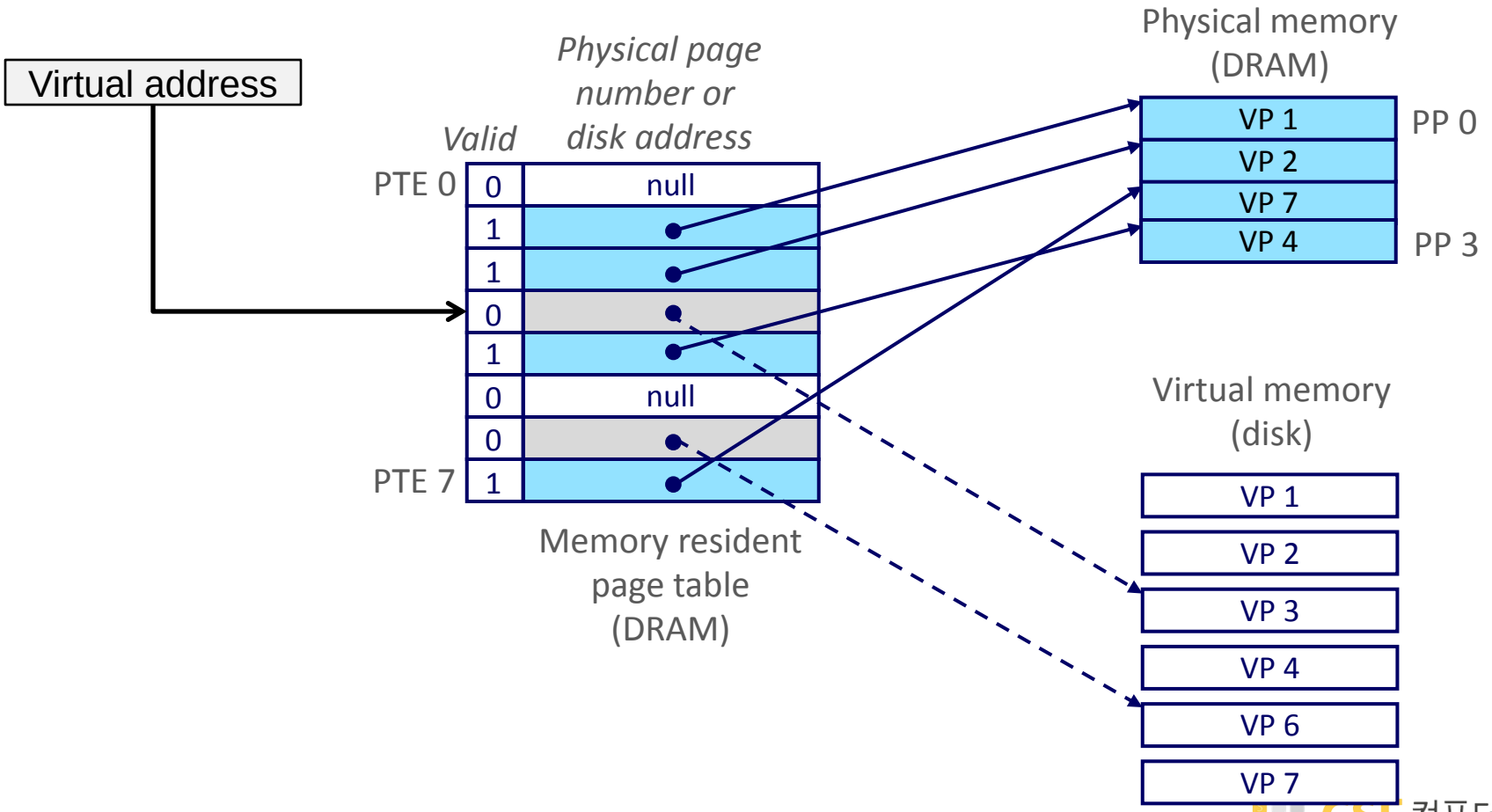
- Reference to VM word that is in physical memory (DRAM cache hit)



CSE 컴퓨터공학부 Department of Computer Science & Engineering

Page Miss

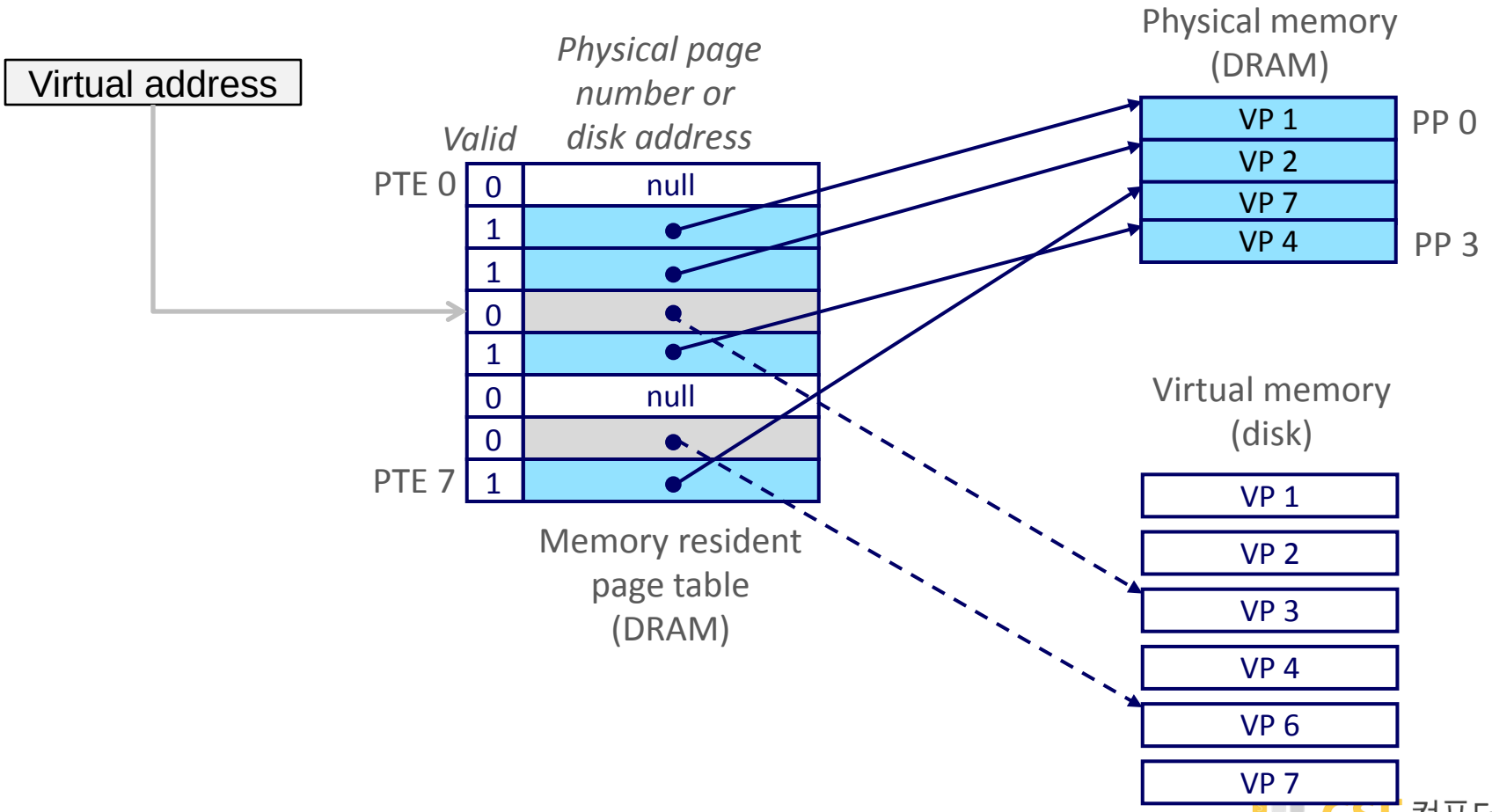
- Reference to VM word that is not in physical memory
  - in the context of VM called “page fault”



CSE 컴퓨터공학부 Department of Computer Science & Engineering

Handling a Page Fault

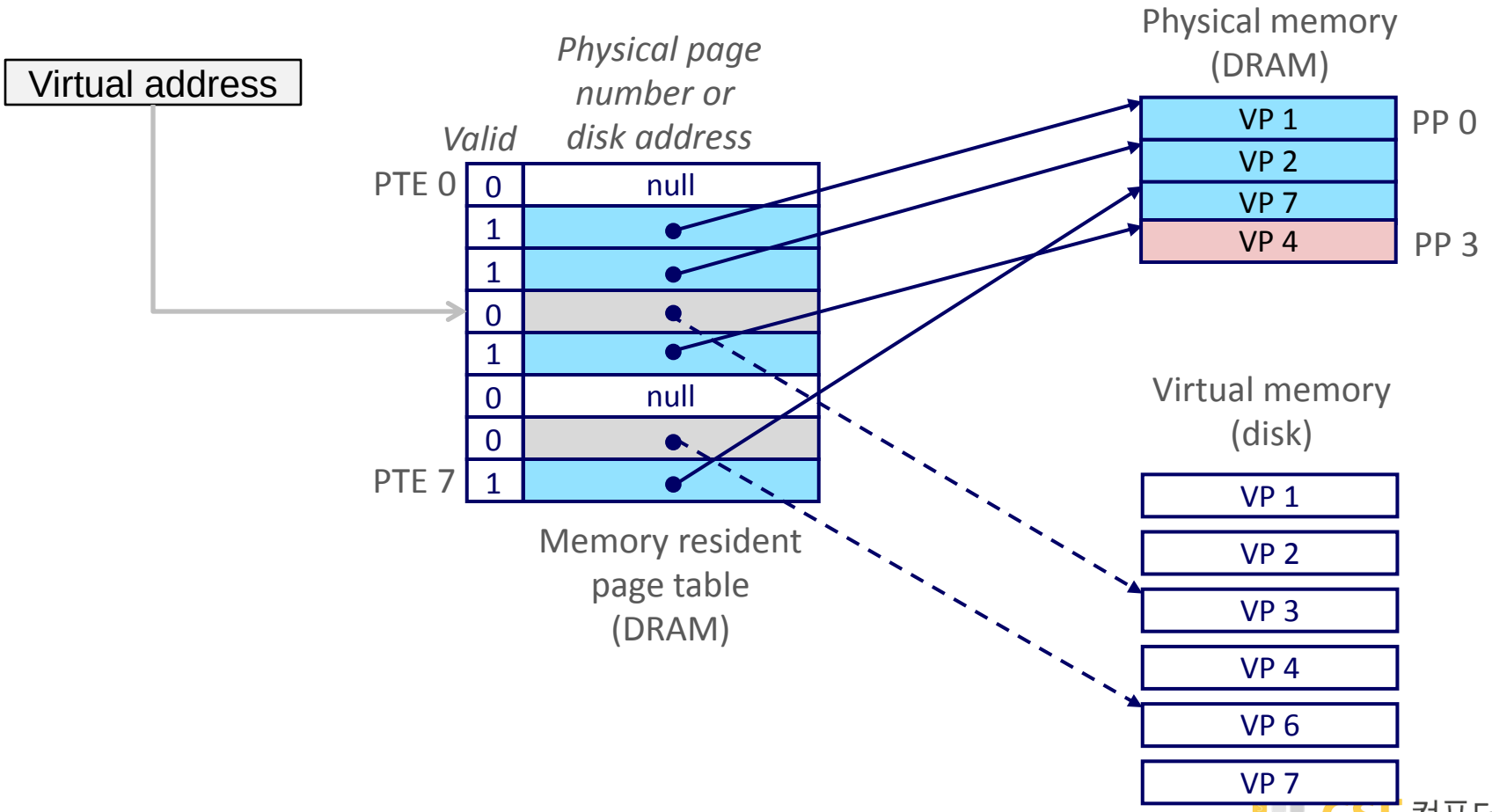
- Page miss causes page fault (an exception)



CSE 컴퓨터공학부 Department of Computer Science & Engineering

Handling a Page Fault

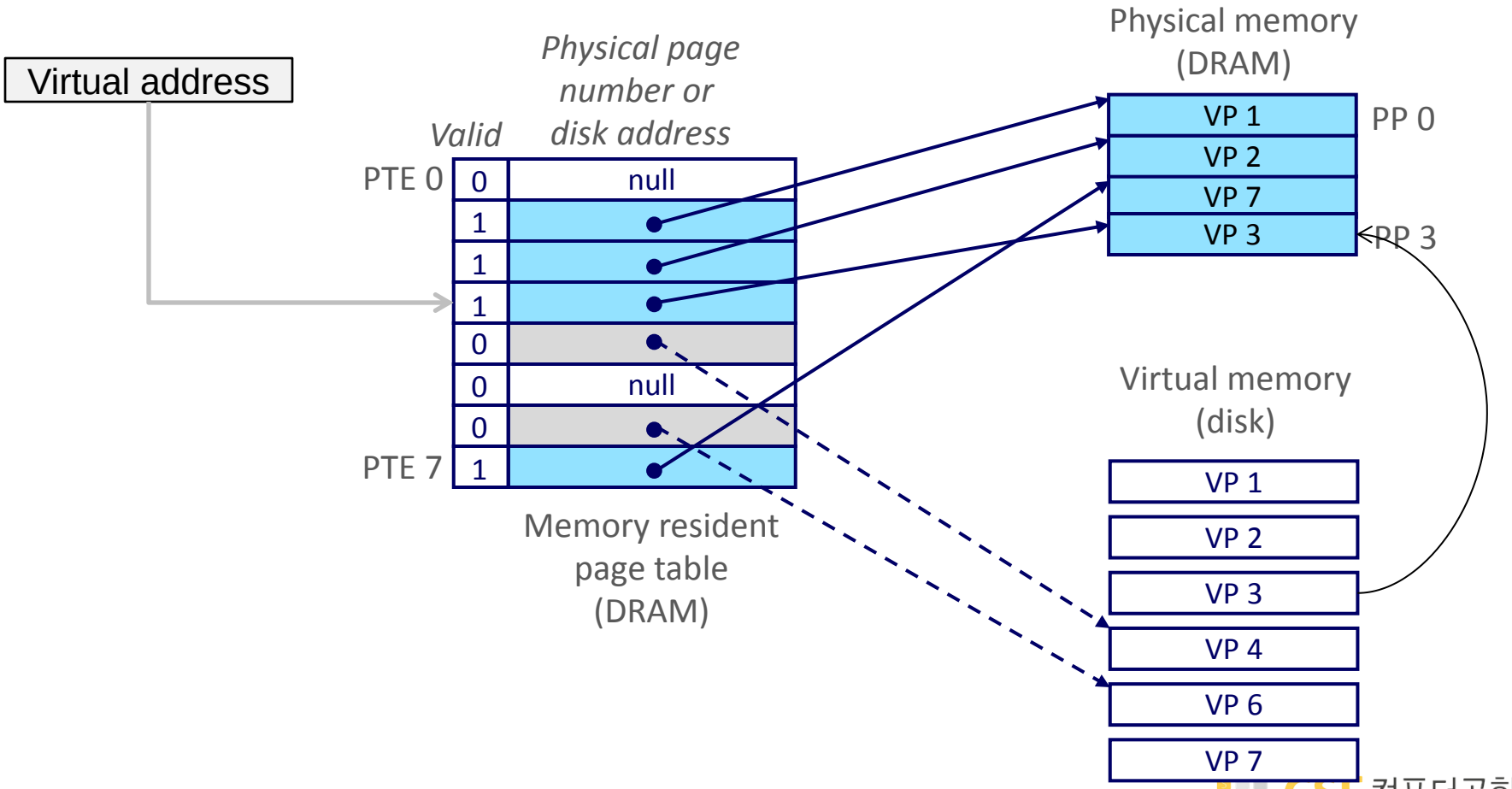
- Page fault handler selects a victim to be evicted (here VP 4)



CSE 컴퓨터공학부 Department of Computer Science & Engineering

Handling a Page Fault

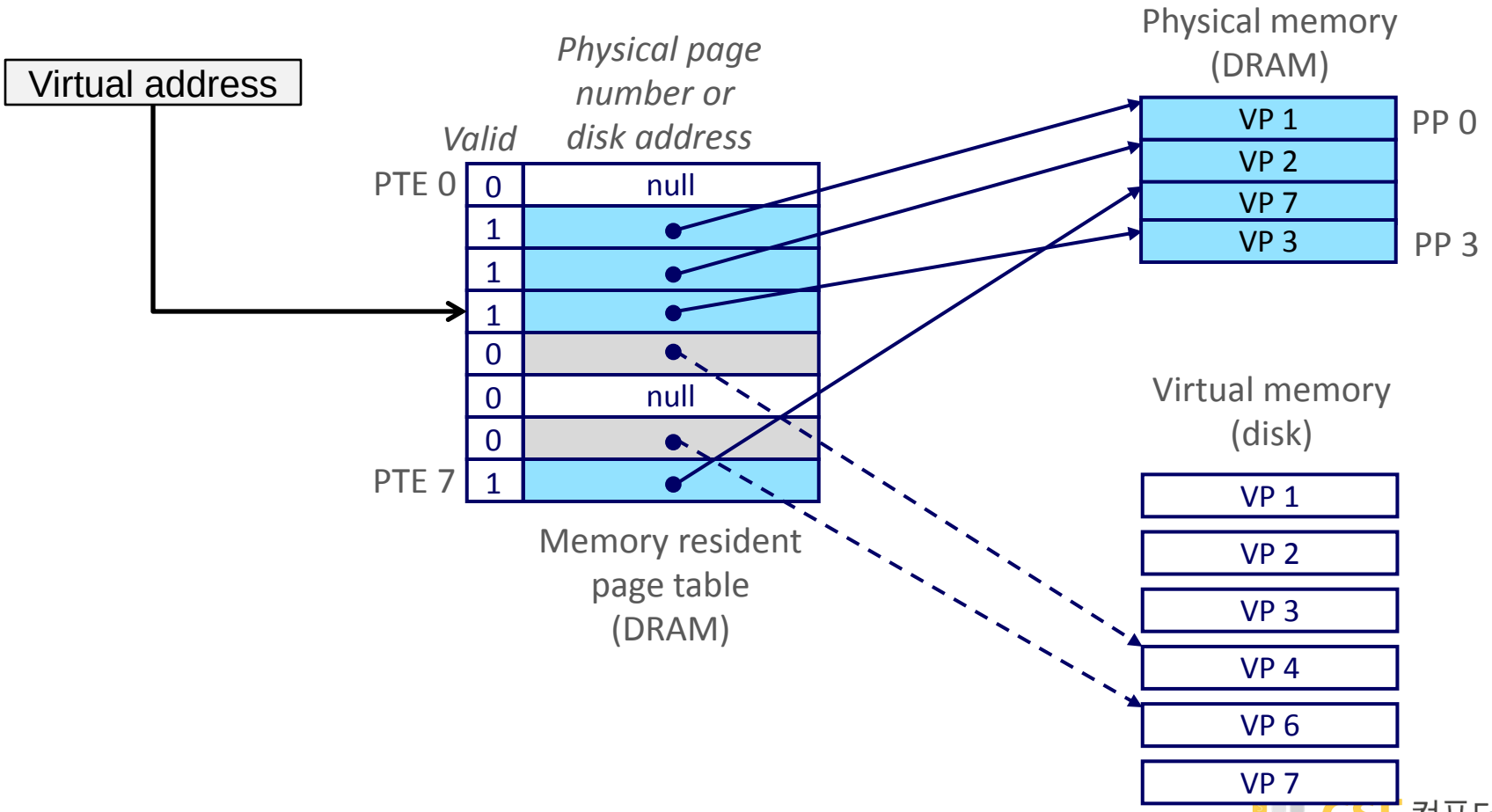
- Page fault handler selects a victim to be evicted (here VP 4) and loads the requested page into physical memory



CSE 컴퓨터공학부 Department of Computer Science & Engineering

Handling Page Fault

- Offending instruction is restarted: page hit!



CSE 컴퓨터공학부 Department of Computer Science & Engineering

Why does it work? Locality

- Virtual memory works because of locality

- At any point in time, programs tend to access a set of active virtual pages called the **working set**
  - Programs with better temporal locality have smaller working sets

- If (working set size < main memory size)
  - Good performance for one process after compulsory misses

- If (SUM(working set sizes) > main memory size)
  - **Thrashing**: Performance meltdown where pages are swapped (copied) in and out continuously

CSE 컴퓨터공학부 Department of Computer Science & Engineering

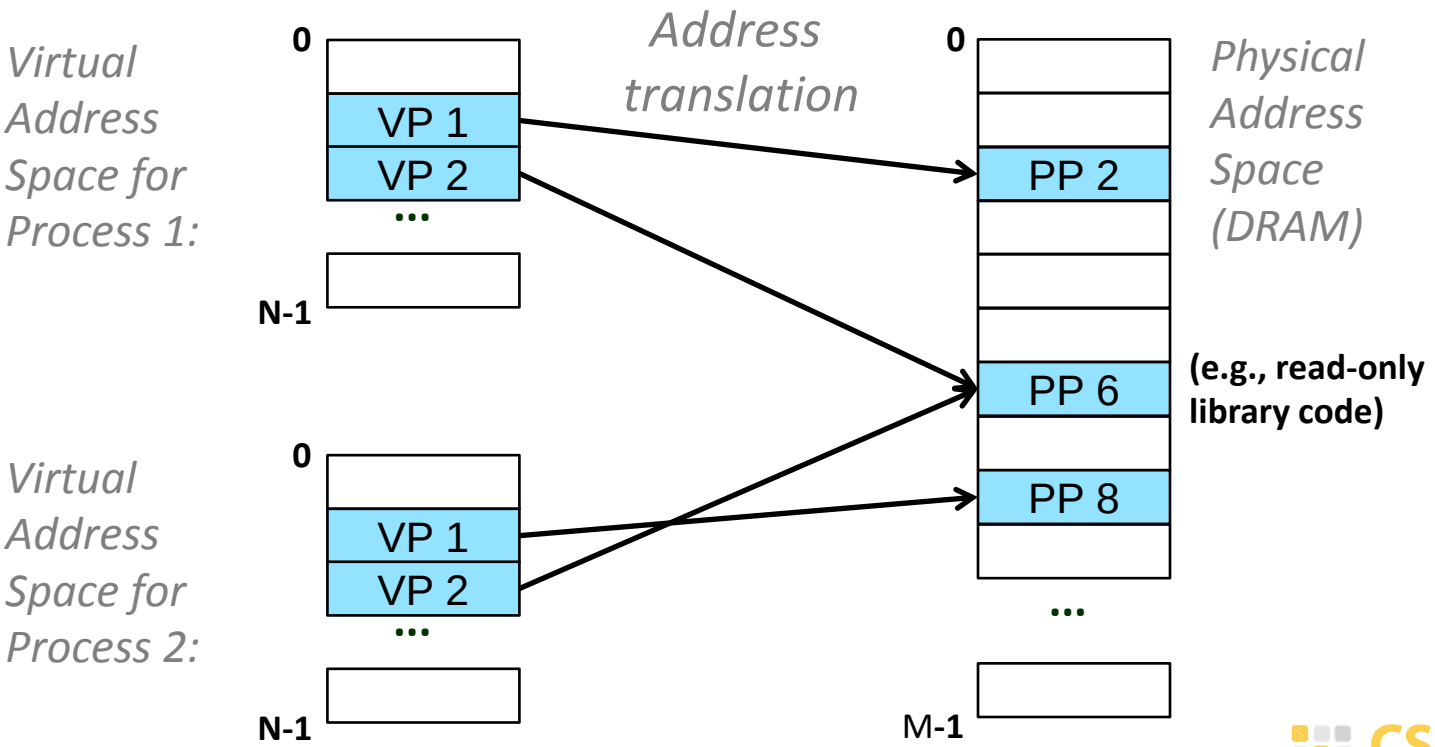
Virtual Memory: Theory

- Virtual Memory(VM): Overview and motivation
- VM as a tool for caching
- **VM as a tool for memory management**
- VM as a tool for memory protection
- Address translation
- Allocation

CSE 컴퓨터공학부 Department of Computer Science & Engineering

VM as a Tool for Memory Management

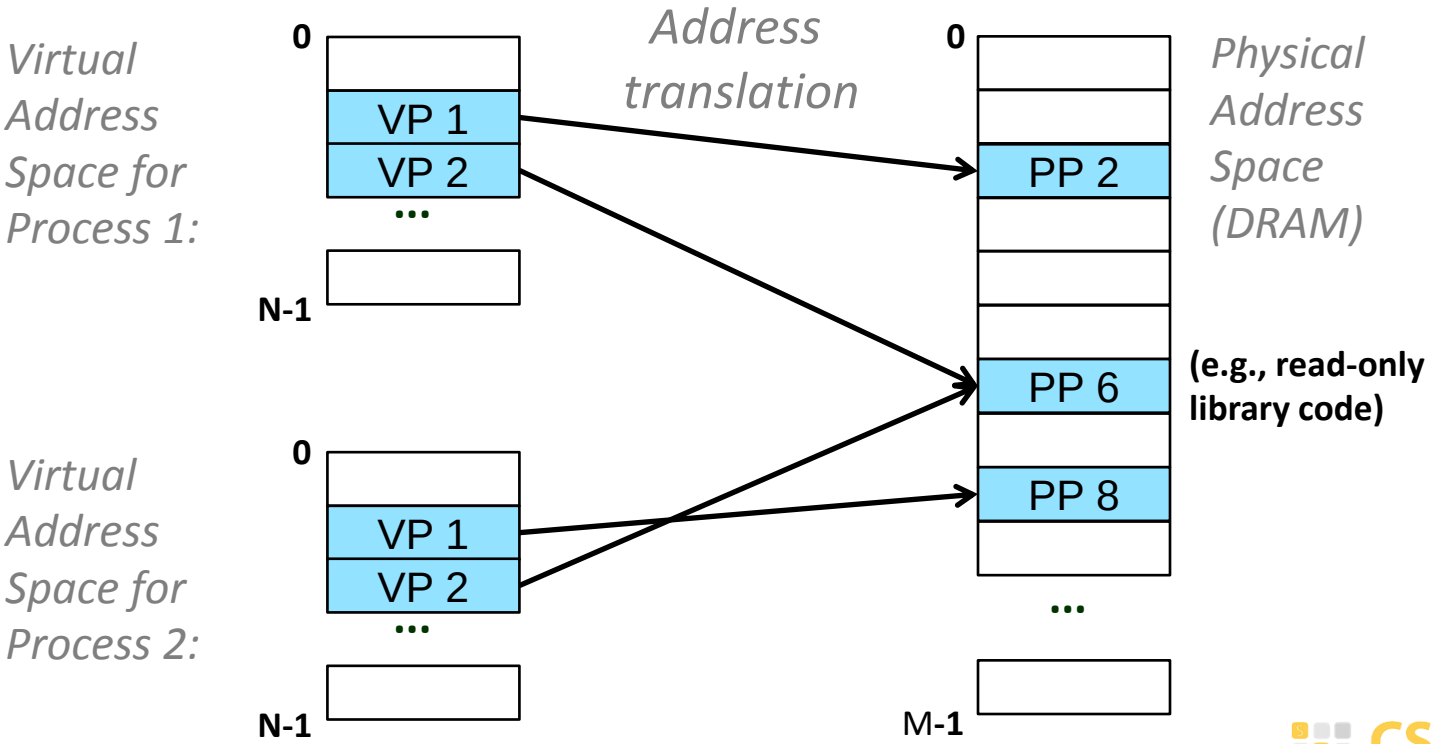
- Key idea: each process has its own virtual address space
  - It can view memory as a simple linear array
  - Mapping function scatters addresses through physical memory
    - Well chosen mappings simplify memory allocation and management



CSE 컴퓨터공학부 Department of Computer Science & Engineering

VM as a Tool for Memory Management

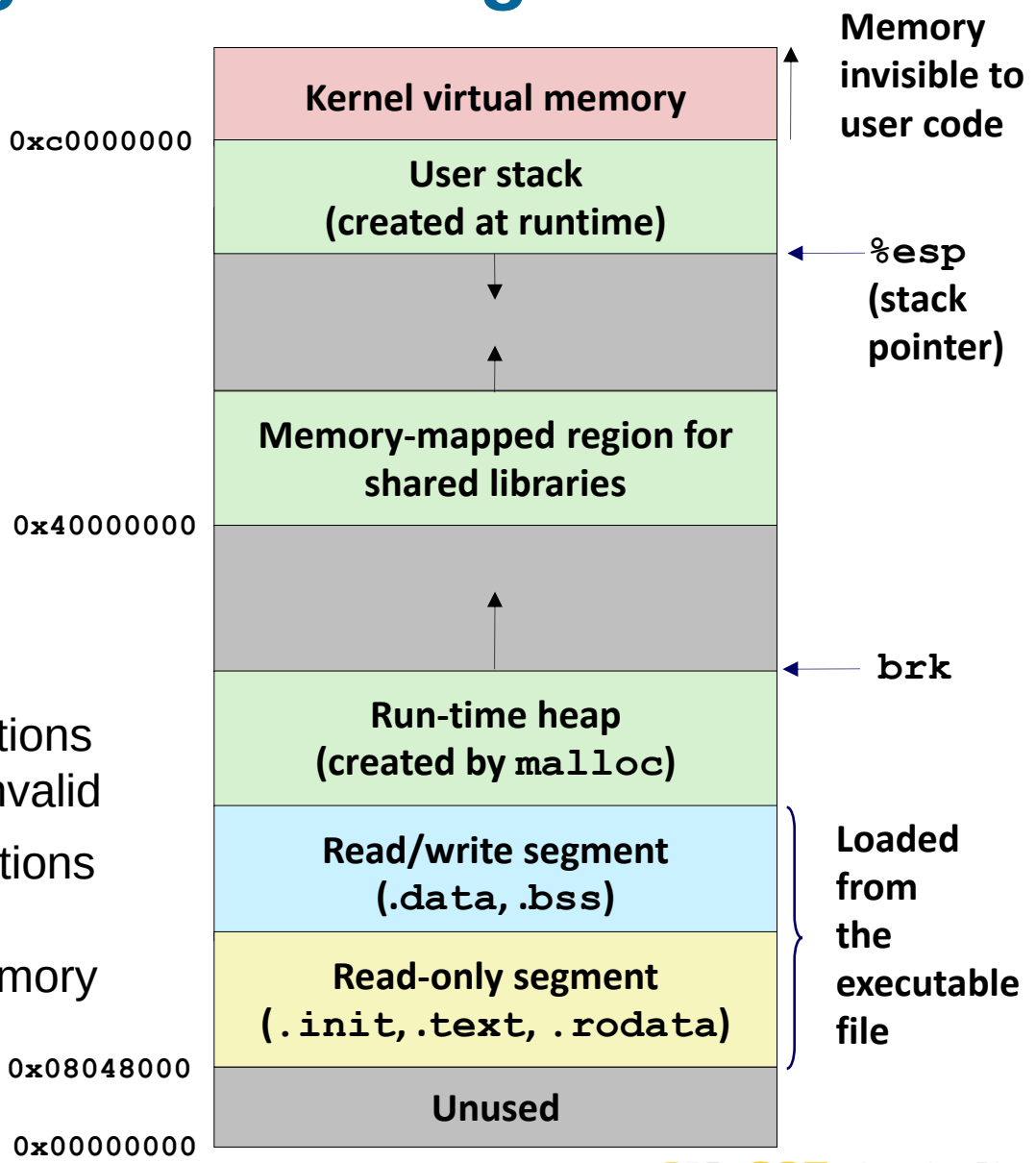
- Memory allocation
  - Each virtual page can be mapped to *any* physical page
  - A virtual page can be stored in different physical pages at different times
- Sharing code and data among processes
  - Map virtual pages to the same physical page (here: PP 6)



CSE 컴퓨터공학부 Department of Computer Science & Engineering

Simplifying Linking and Loading

- Linking
  - Each program has similar virtual address space
  - Code, stack, and shared libraries always start at the same address
- Loading
  - `execve()` allocates virtual pages for .text and .data sections = creates PTEs marked as invalid
  - The `.text` and `.data` sections are copied, page by page, on demand by the virtual memory system



CSE 컴퓨터공학부 Department of Computer Science & Engineering

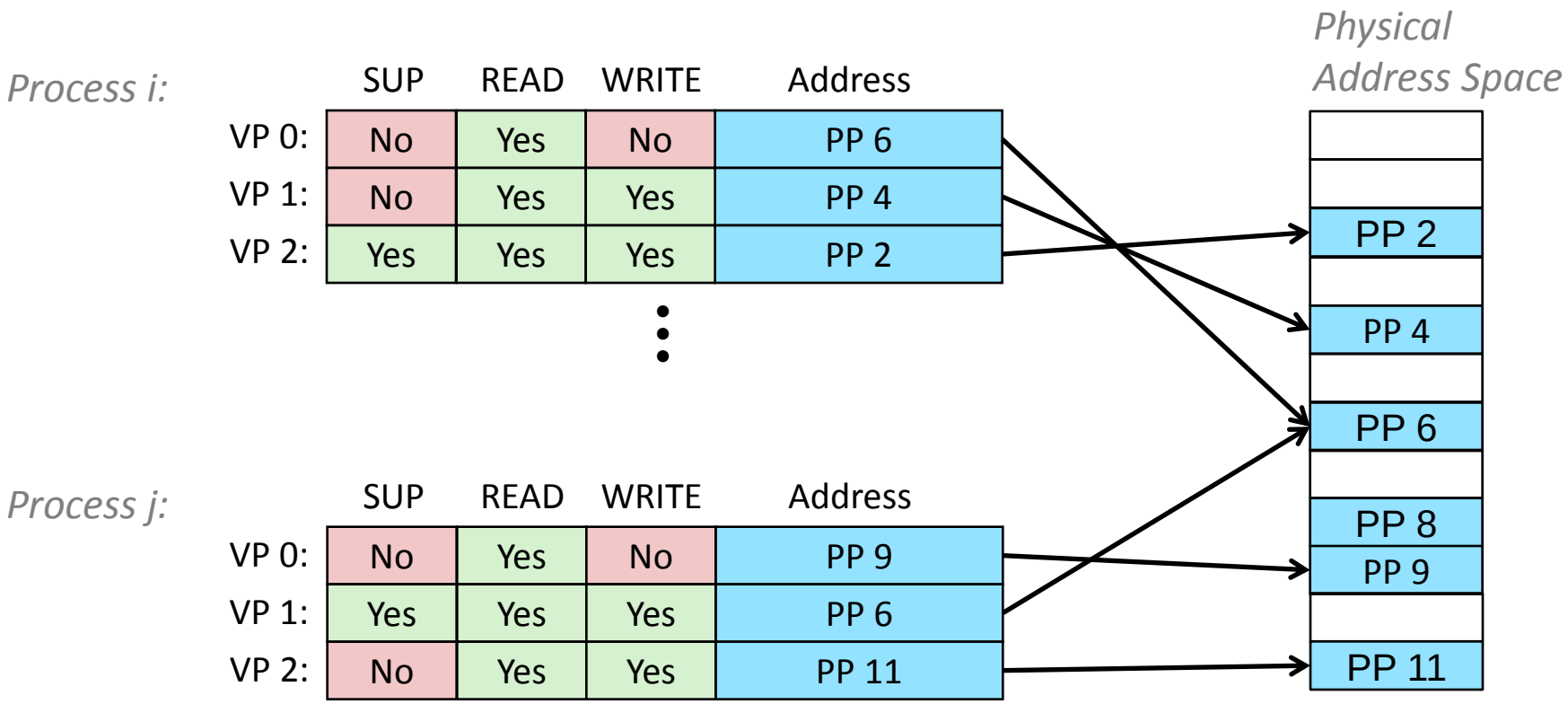
Virtual Memory: Theory

- Virtual Memory(VM): Overview and motivation
- VM as a tool for caching
- VM as a tool for memory management
- **VM as a tool for memory protection**
- Address translation
- Allocation

CSE 컴퓨터공학부 Department of Computer Science & Engineering

VM as a Tool for Memory Protection

- Extend PTEs with permission bits
- Page fault handler checks these before remapping
  - If violated, send process SIGSEGV (segmentation fault)



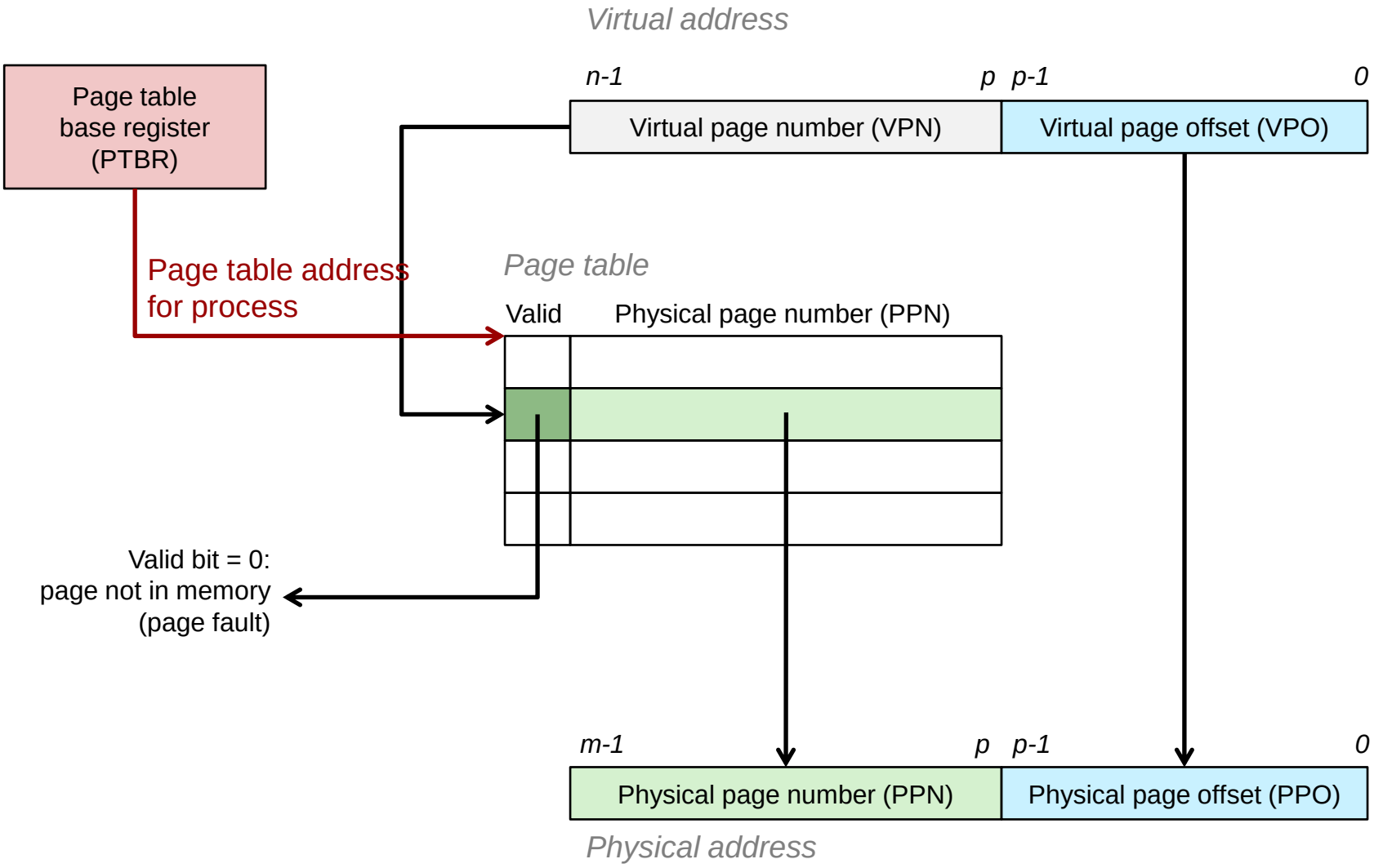
CSE 컴퓨터공학부 Department of Computer Science & Engineering

Virtual Memory: Theory

- Virtual Memory(VM): Overview and motivation
- VM as a tool for caching
- VM as a tool for memory management
- VM as a tool for memory protection
- **Address translation**
- Allocation

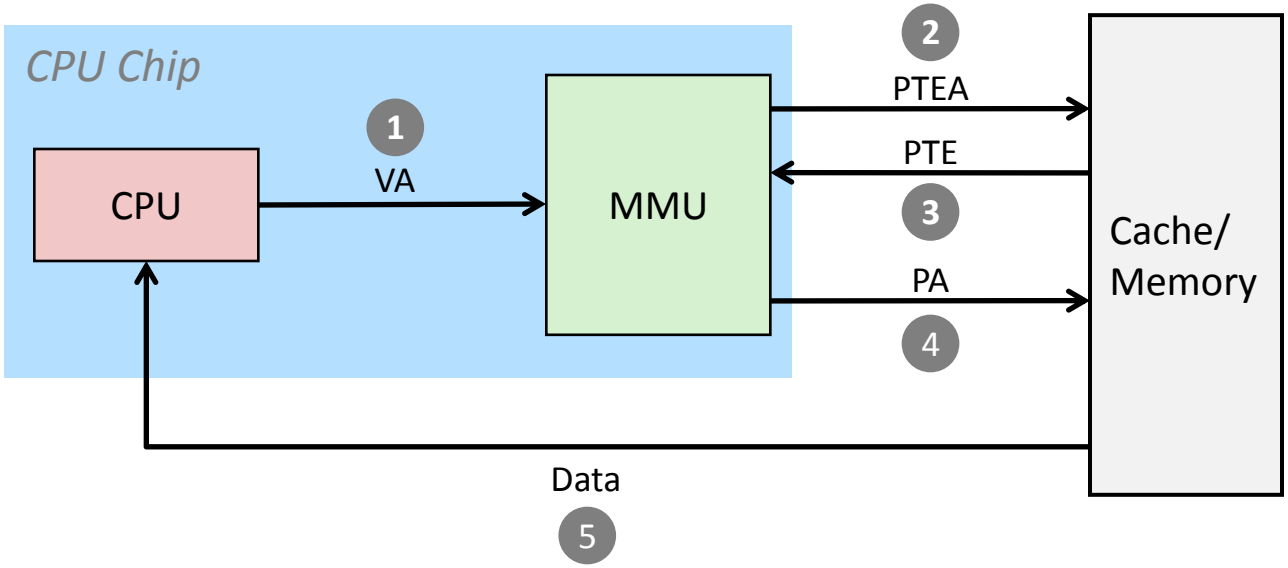
CSE 컴퓨터공학부 Department of Computer Science & Engineering

Address Translation With a Page Table



CSE 컴퓨터공학부 Department of Computer Science & Engineering

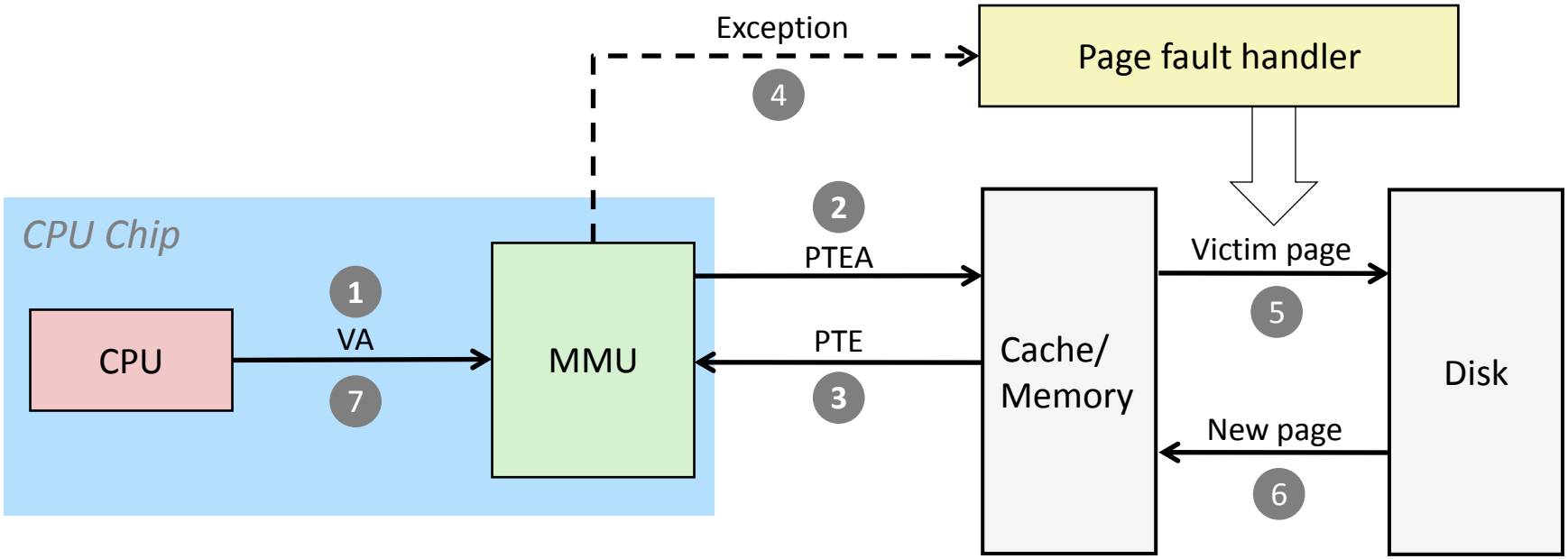
Address Translation: Page Hit



- 1) Processor sends virtual address to MMU
- 2-3) MMU fetches PTE from page table in memory
- 4) MMU sends physical address to cache/memory
- 5) Cache/memory sends data word to processor

CSE 컴퓨터공학부 Department of Computer Science & Engineering

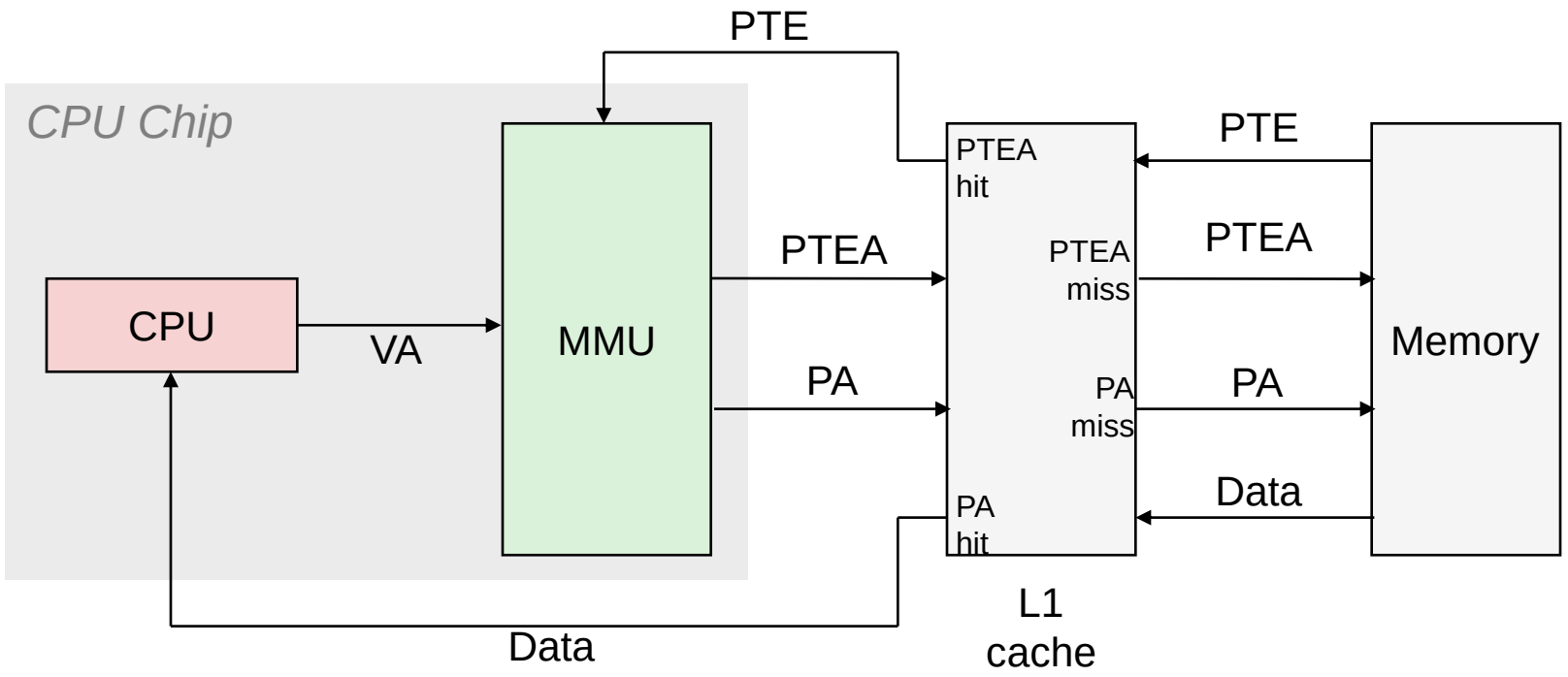
Address Translation: Page Fault



- 1) Processor sends virtual address to MMU
- 2-3) MMU fetches PTE from page table in memory
- 4) Valid bit is zero, so MMU triggers page fault exception
- 5) Handler identifies victim (and, if dirty, pages it out to disk)
- 6) Handler pages in new page and updates PTE in memory
- 7) Handler returns to original process, restarting faulting instruction

CSE 컴퓨터공학부 Department of Computer Science & Engineering

Integrating VM and Cache



VA: virtual address, PA: physical address, PTE: page table entry, PTEA = PTE address

CSE 컴퓨터공학부 Department of Computer Science & Engineering

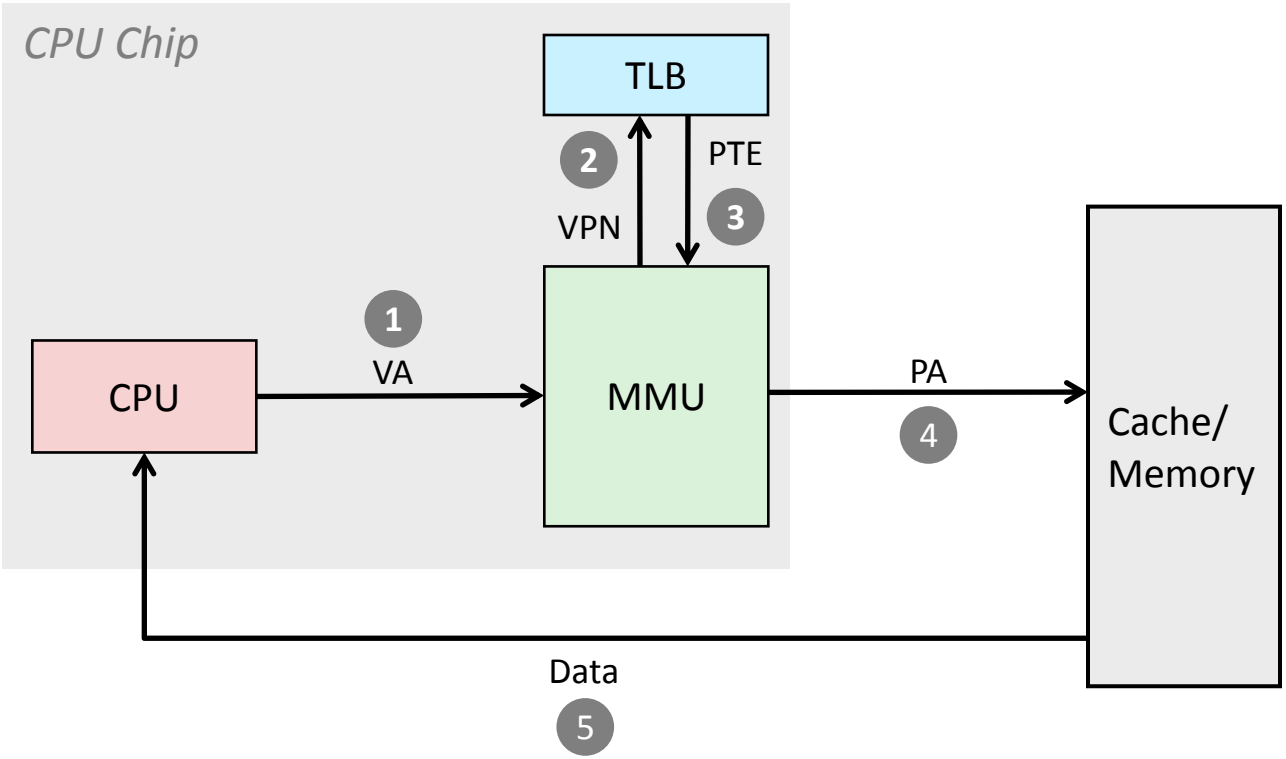
Speeding up Translation with a TLB

- Page table entries (PTEs) are cached in L1 like any other memory word
  - PTEs may be evicted by other data references
  - PTE hit still requires a small L1 delay

- Solution: **Translation Lookaside Buffer (TLB)**
  - Small hardware cache in MMU
  - Maps virtual page numbers to physical page numbers
  - Contains complete page table entries for small number of pages

CSE 컴퓨터공학부 Department of Computer Science & Engineering

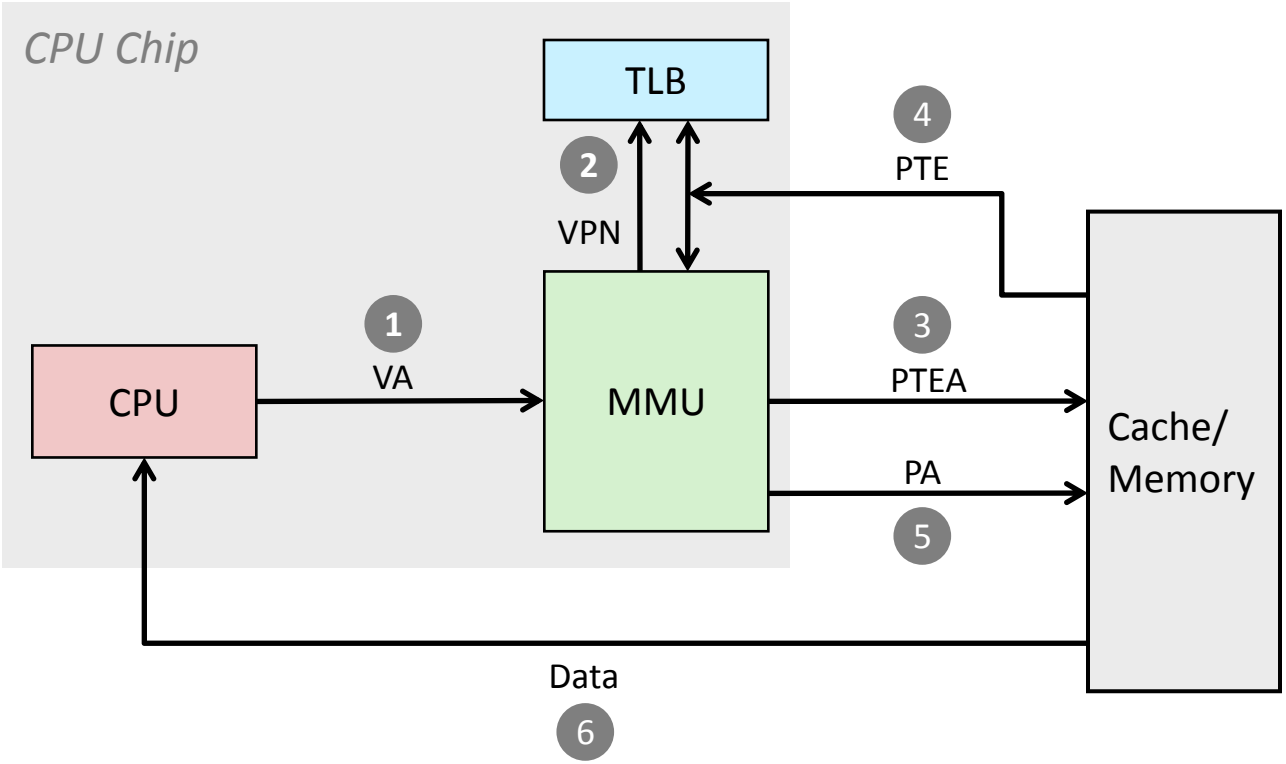
TLB Hit



- A TLB hit eliminates a memory access

CSE 컴퓨터공학부 Department of Computer Science & Engineering

TLB Miss

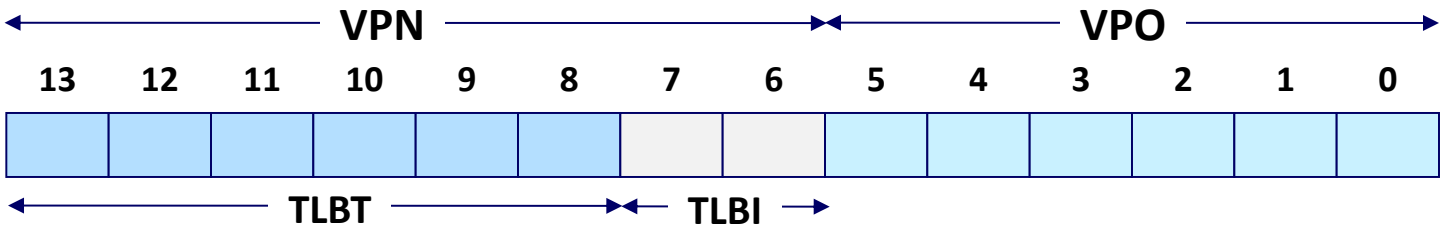


- A TLB miss incurs one (or more) additional memory access(es) (the PTE)
  - fortunately, TLB misses are rare.

CSE 컴퓨터공학부 Department of Computer Science & Engineering

Virtual Addresses and TLB Index/Tag

- The TLB is a small cache with a high associativity
  - $T = 2^t$ sets
  - TLBI (TLB Index) = *t* least significant bits of VPN
  - TLBT (TLB Tag) = remaining bits of VPN



Set	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid	Tag	PPN	Valid
0	03	–	0	09	0D	1	00	–	0	07	02	1
1	03	2D	1	02	–	0	04	–	0	0A	–	0
2	02	–	0	08	–	0	06	–	0	03	–	0
3	07	–	0	03	0D	1	0A	34	1	02	–	0

CSE 컴퓨터공학부 Department of Computer Science & Engineering

Summary

- Programmer's view of virtual memory
  - Each process has its own private linear address space
  - Cannot be corrupted by other processes

- System view of virtual memory
  - Uses memory efficiently by caching virtual memory pages
    - Efficient only because of locality
  - Simplifies memory management and programming
  - Simplifies protection by providing a convenient interpositioning point to check permissions

CSE 컴퓨터공학부 Department of Computer Science & Engineering

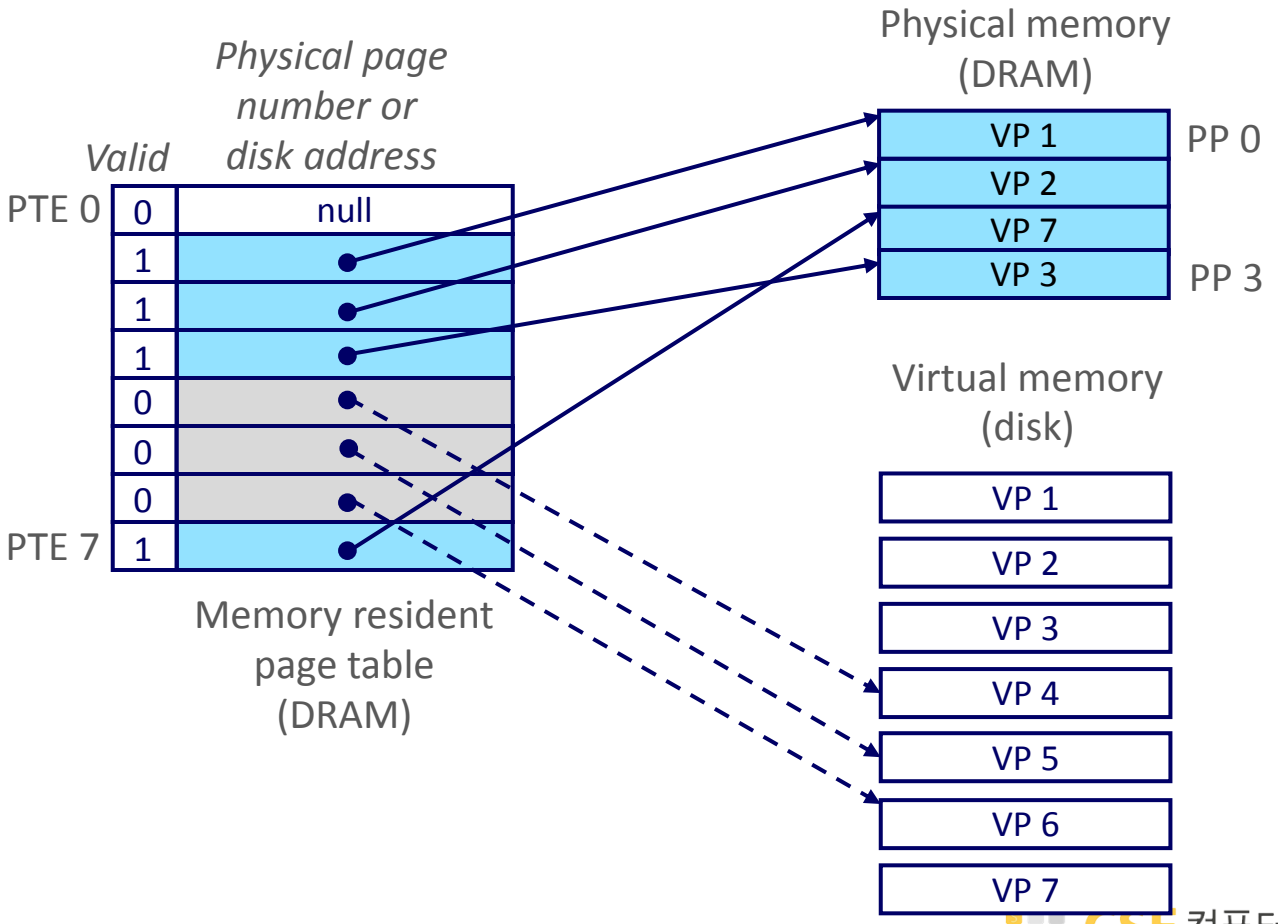
Virtual Memory: Theory

- Virtual Memory(VM): Overview and motivation
- VM as a tool for caching
- VM as a tool for memory management
- VM as a tool for memory protection
- Address translation
- **Allocation**

CSE 컴퓨터공학부 Department of Computer Science & Engineering

Allocating Virtual Pages

- Example: Allocating VP 5
  - kernel allocates VP 5 on disk and points PTE 5 to it



CSE 컴퓨터공학부 Department of Computer Science & Engineering

Memory System Summary

- L1/L2 Memory Cache
  - purely a speed-up technique
  - behavior invisible to application programmer and (mostly) OS
  - implemented entirely in hardware

- Virtual Memory
  - supports many OS-related functions
    - process creation, task switching, protection, sharing
  - software
    - allocates/shares physical memory among processes
    - maintains high-level tables tracking memory type, source, sharing
    - handles exceptions, fills in hardware-defined mapping tables
  - hardware
    - translates virtual addresses via mapping tables, enforcing permissions
    - accelerates mapping via translation cache (TLB)

CSE 컴퓨터공학부 Department of Computer Science & Engineering