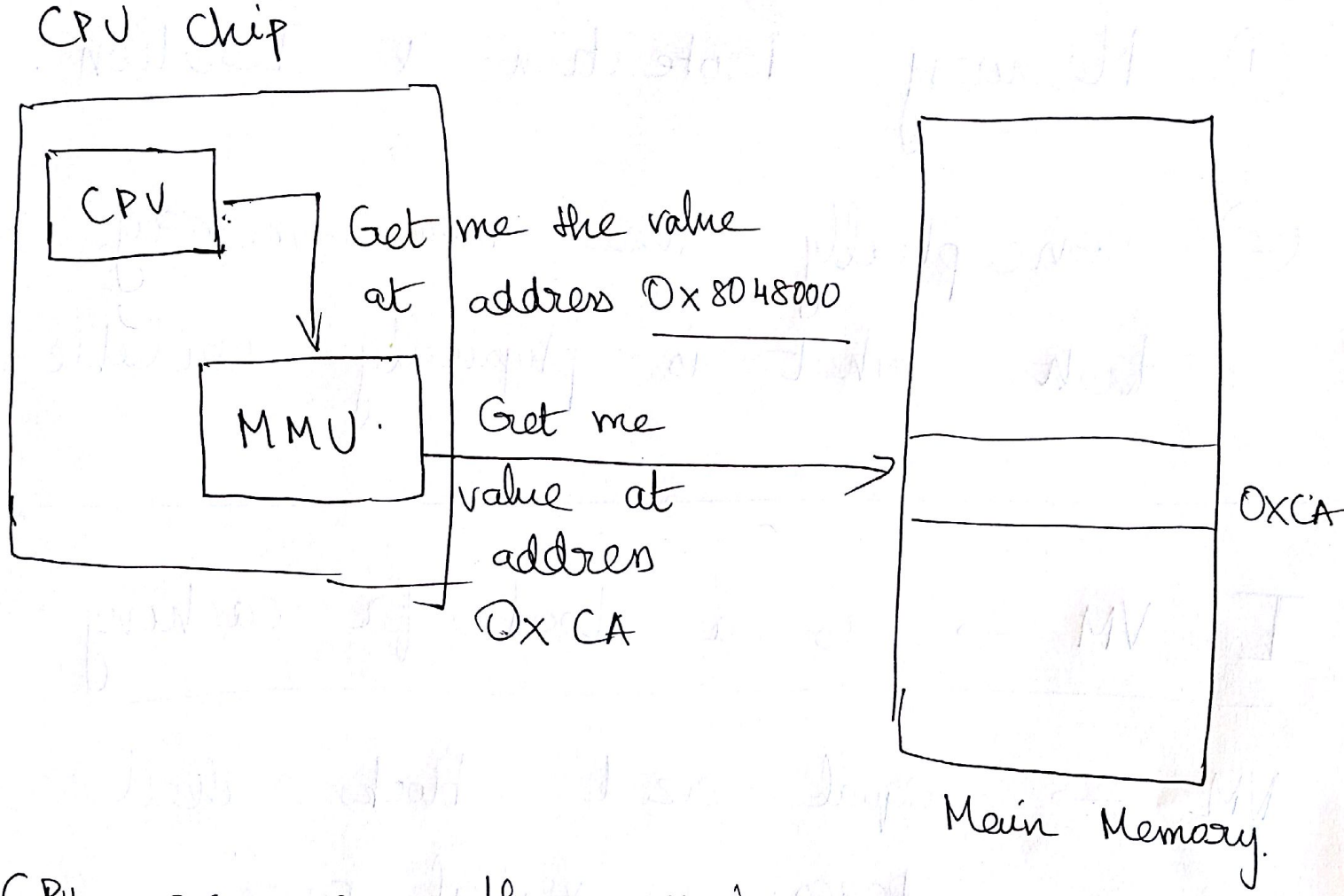
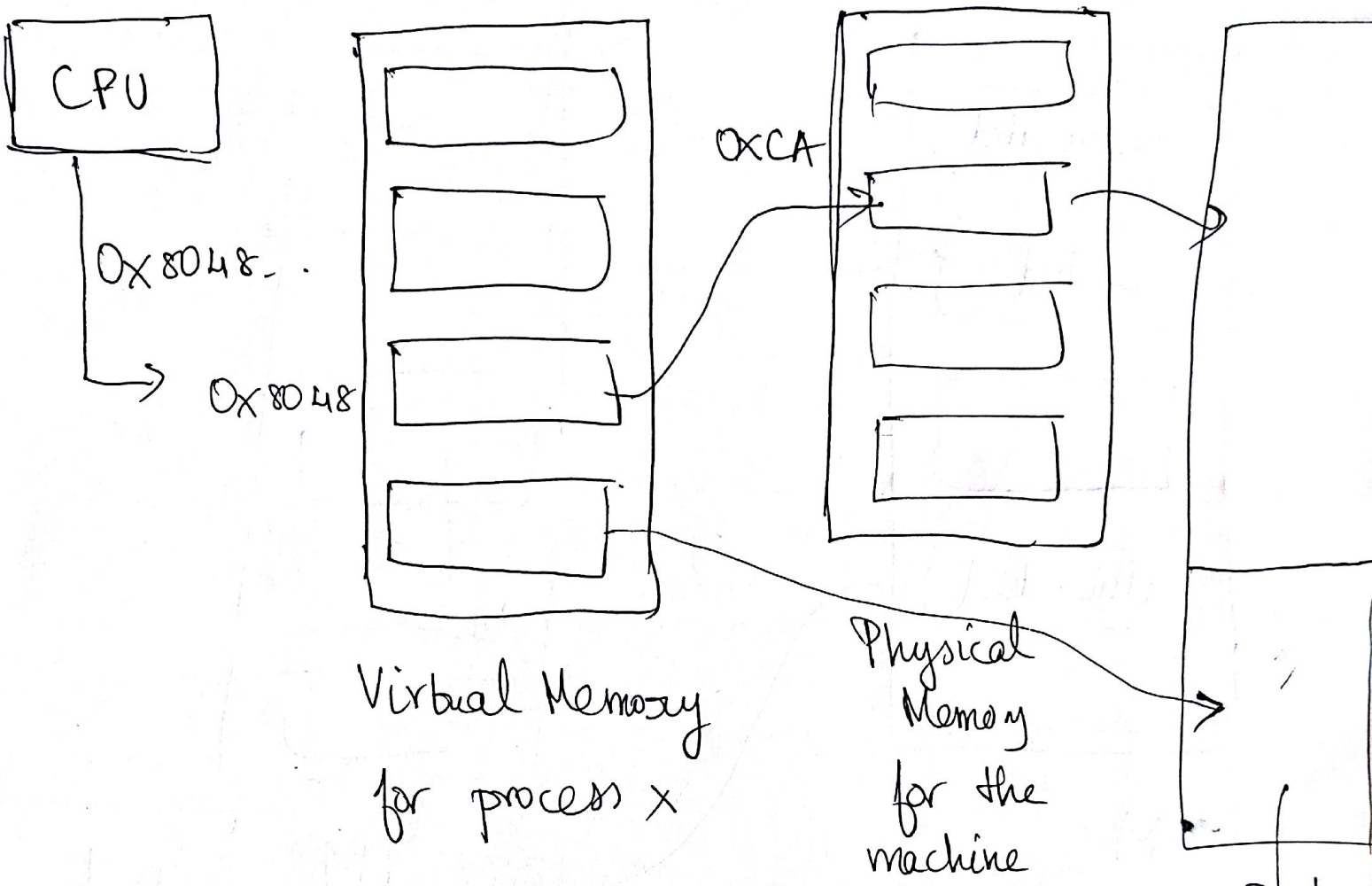
4/18

Virtual Memory / Virtual Addressing.



CPU accesses the main memory by using a virtual address. which is translated into a physical address.

↓ Conceptually



VM for a process on a 32 bit machine can be upto 4 GB.

But what if the physical memory of RAM?

Swap space

Primary benefits of VM ?

(1) Memory Protection & Isolation.

(2) Conceptually use more memory that what is physically available.

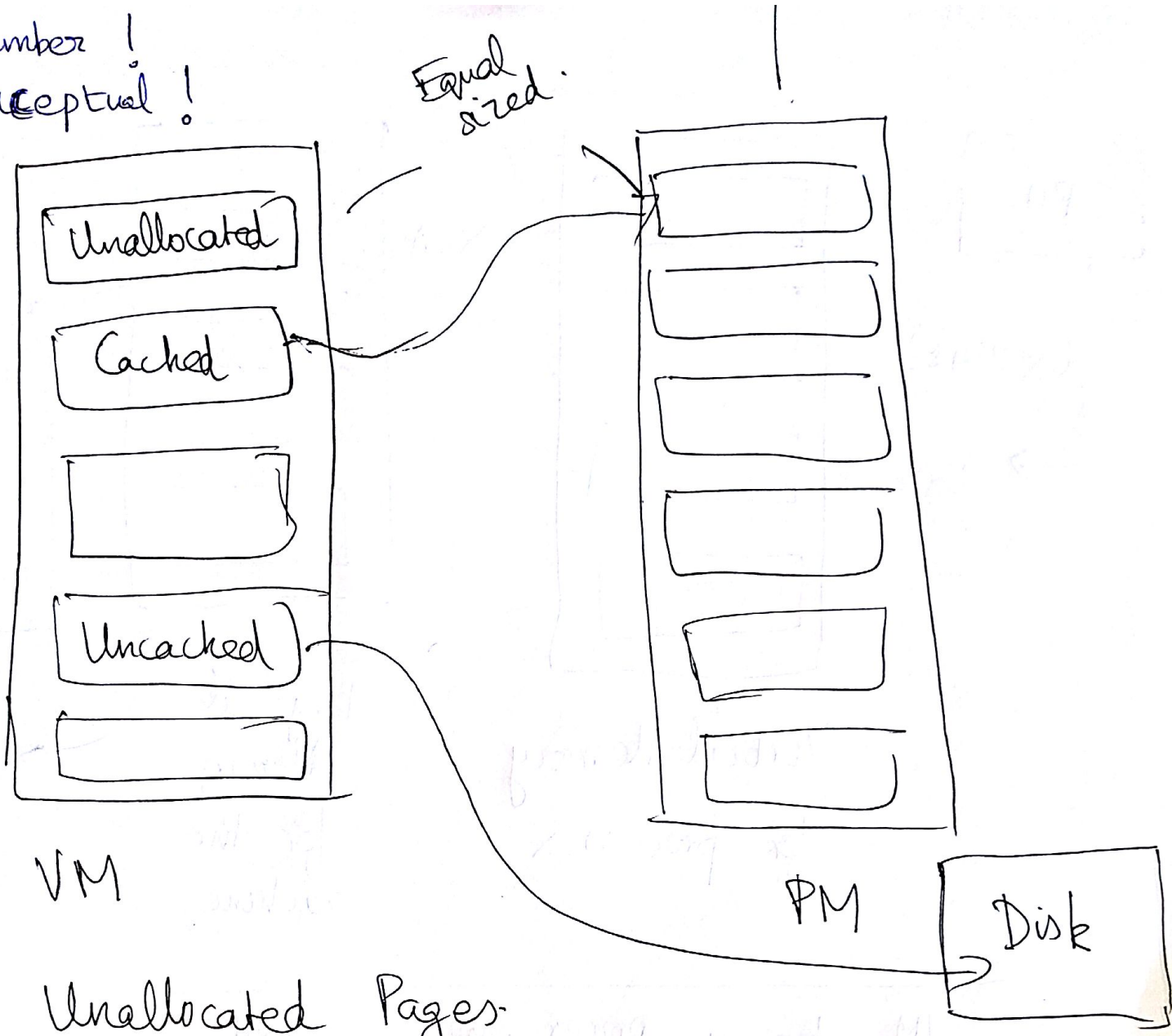
I VM → as a tool for caching

VM → equal sized blocks called pages. → virtual pages.

Physical pages
(or)
page frames.

(See next page)

Remember !
Conceptual !



(1) Unallocated Pages.

→ have no data associated with them. // Occupy no space on the disk.

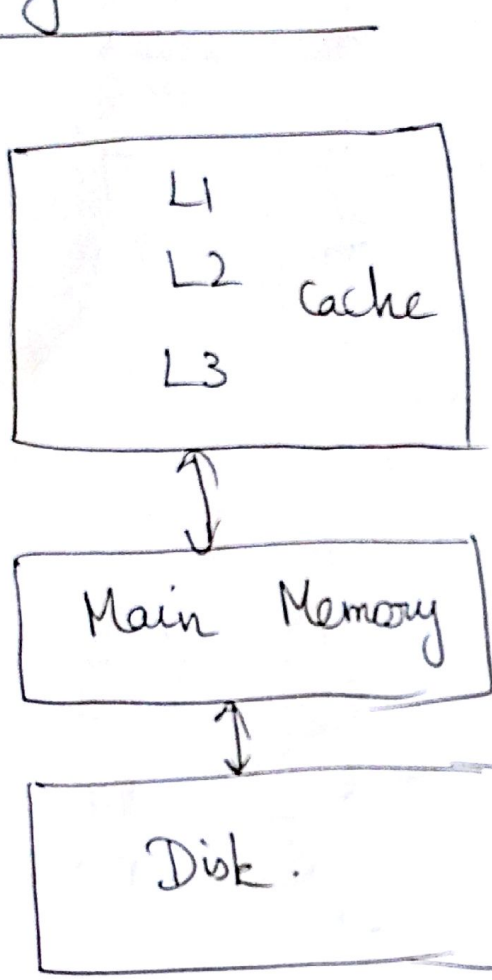
(2) Cached Pages.

→ Allocated pages currently in the physical memory

(3) Uncached pages

→ Allocated pages that are not cached in the PM

Design choices



Static.

SRAM

Dynamic.

10 times slower

DRAM

100k times slower

Misses here are very expensive.

Design is driven by this

1. Large page sizes (4KB to 2MB)
2. Fully Associative. → A virtual page can be placed anywhere (in any physical page) (No conflict misses).
3. Replacement Algorithm is much more important!

(4) Large Write times → Write back policy (instead of write through).

To represent what the page status is and where the page is

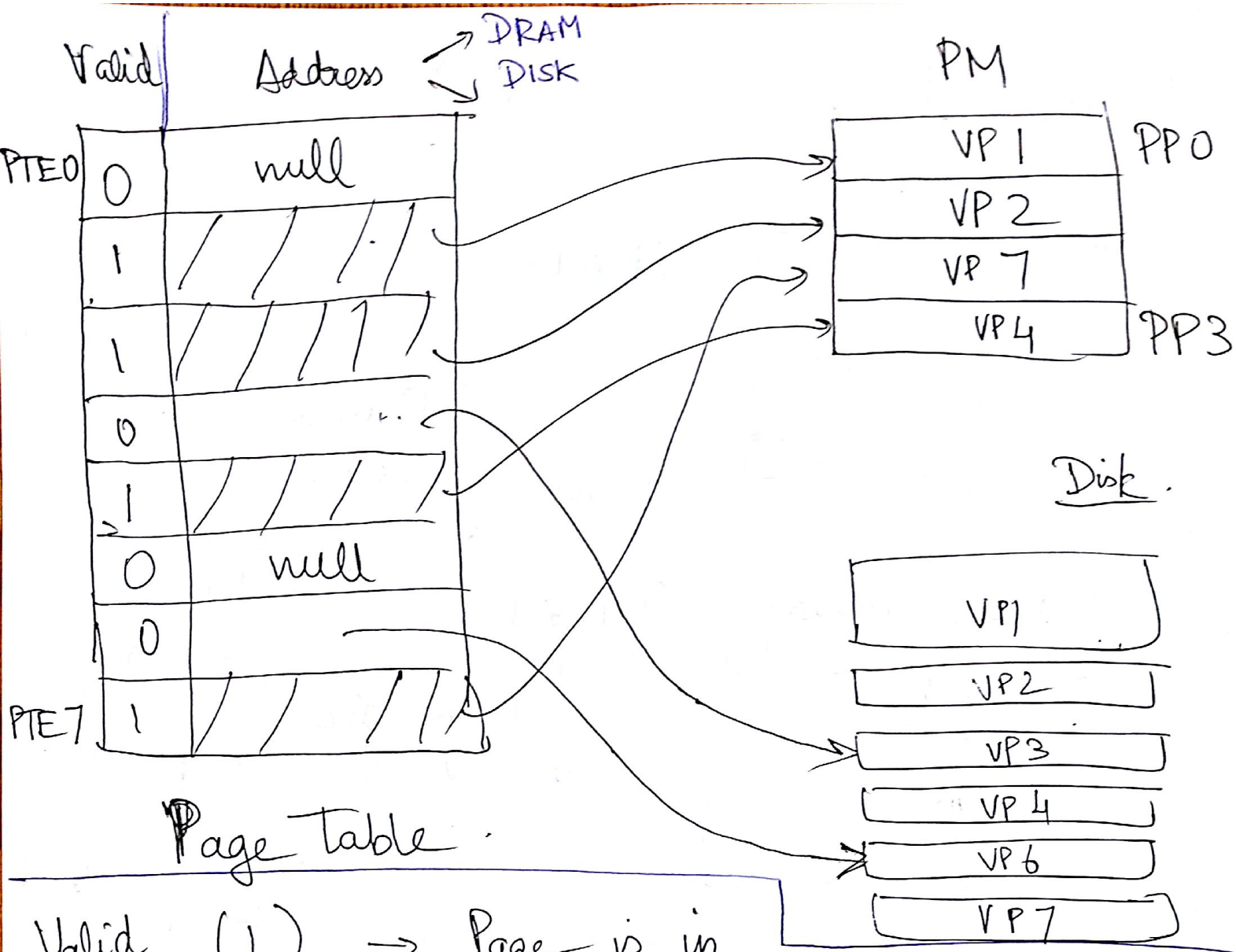
→ PAGE TABLE !

- resides in DRAM
- an array of page table entries (PTE)
  - valid bit
  - n-bit address field.

Who uses this page table ?

The MMU.

→ read the page table to convert a VA to a PA.



Valid (1) → Page is in the DRAM. (Address indicates the start of the physical page)

Valid (0) & Address is null
Page is unallocated

Valid (0) & some address
Page is on the disk (uncached).

Page hit → read a page that is cached Eg. VP1

Page fault → (A DRAM cache miss).

→ CPU needs something from VP3.

→ Address translator —reads→ PTE 3.

not cached in DRAM.

So Page fault!

↳ starts an exception handler in the OS.

Chooses a Victim Page and "Swaps it out" to the disk.

and "swaps in" PTE 3 to the DRAM.

(Figures 9.6 & 9.7)

Wait till a page fault happens to swap → demand paging

↓

Swap based on demand.

Locality !

→ guarantees that working set will be a small subset.

What if the working set of pages > Physical memory ?

↓

thrashing

Swapping in and out a bunch of pages repeatedly.

↓

Small set of active pages (resident set)

What to do when thrashing happens ?