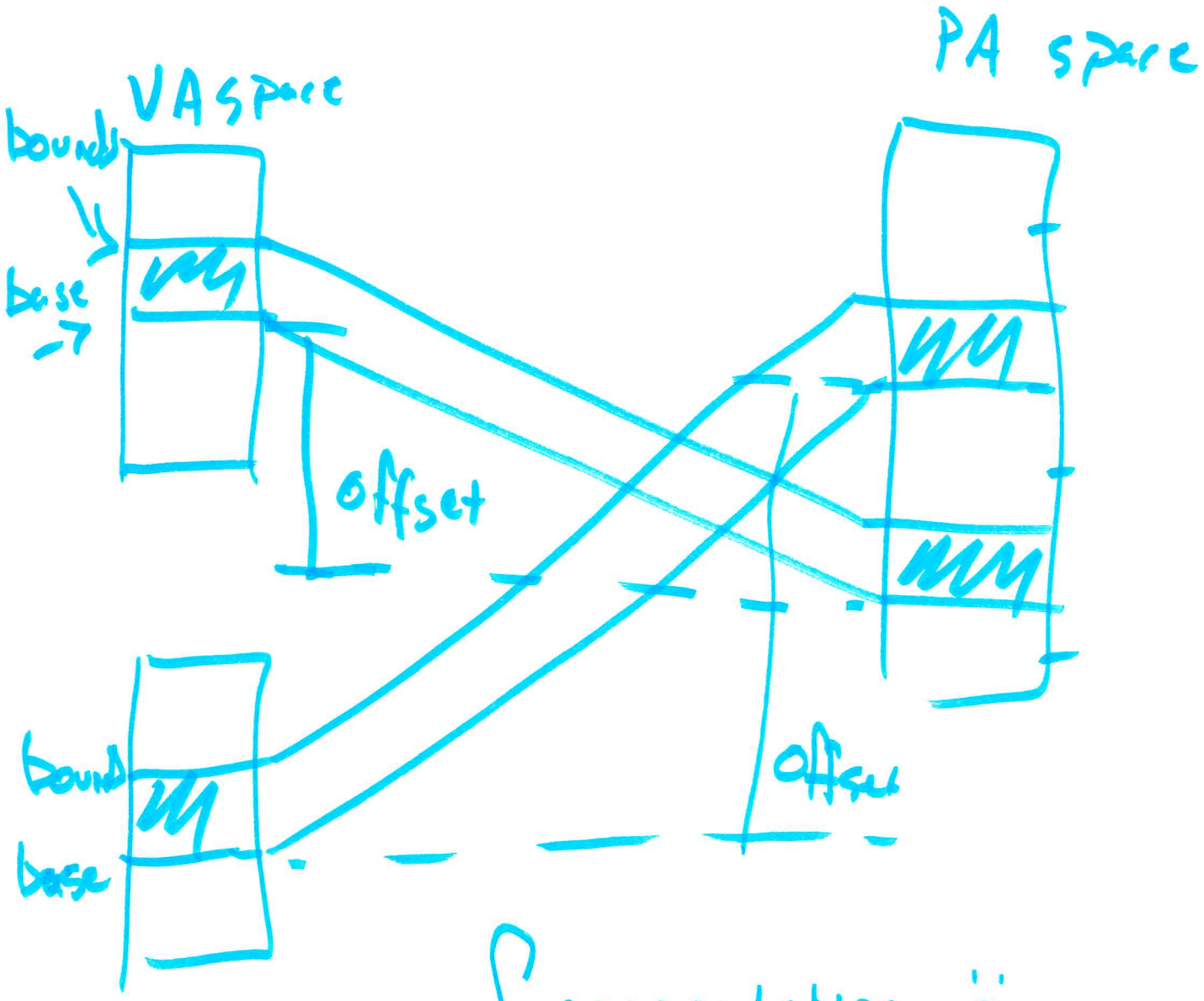
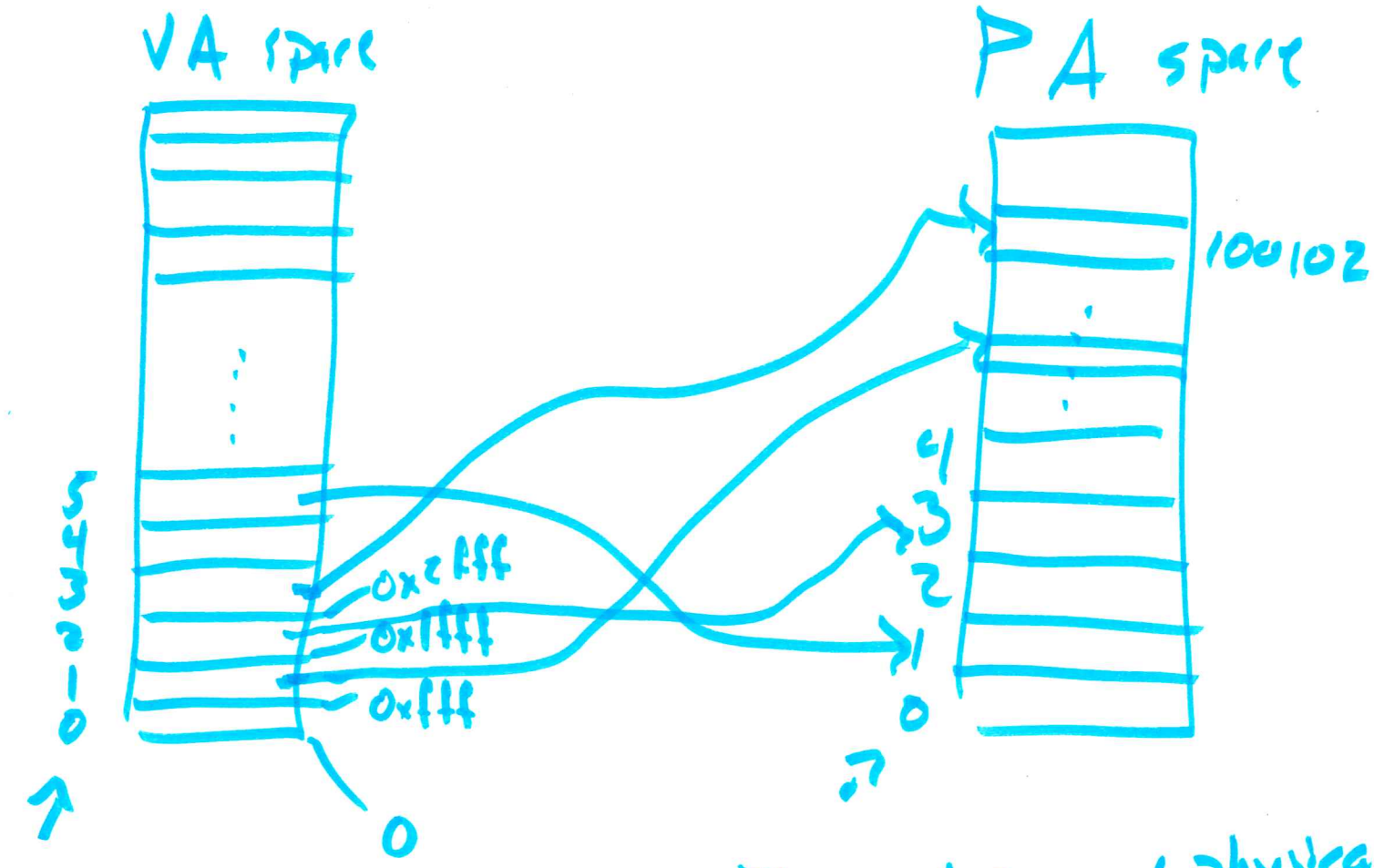
Page-based virtual memory

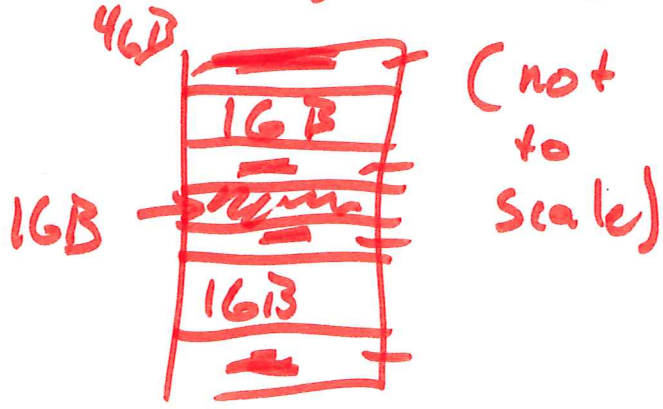


fragmentation :(

Paging - break address up into same-sized chunks (4 kB)



- fragmentation problem (w/ segmentation)

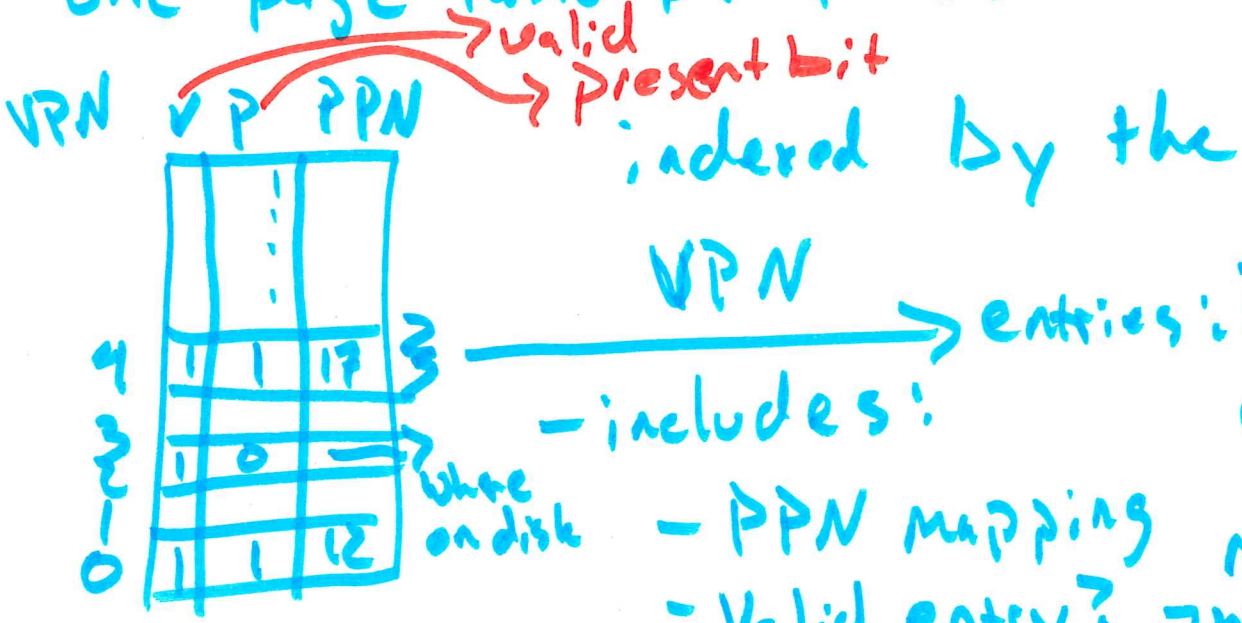


How to translate from VA to PA

Virtual address → MMU → physical address

Page table

- MMU looks up translation in page table
- one page table per process



- includes:
  - PPN mapping
  - Valid entry?
  - Permissions (read, write, execute)

Present bit → is this on disk or in memory?

physical

Memory as large cache for disk

Virtual memory makes it possible

page faults → OS look up if page is valid (possibly on disk)

→ OS will allocate a physical page → possibly moving a page to disk

→ OS will copy requested page into memory

→ OS updates the page table(s)

Swapping

page faults → caused by "page misses"

MMU looks up VPN in the page table → in memory (valid & present)

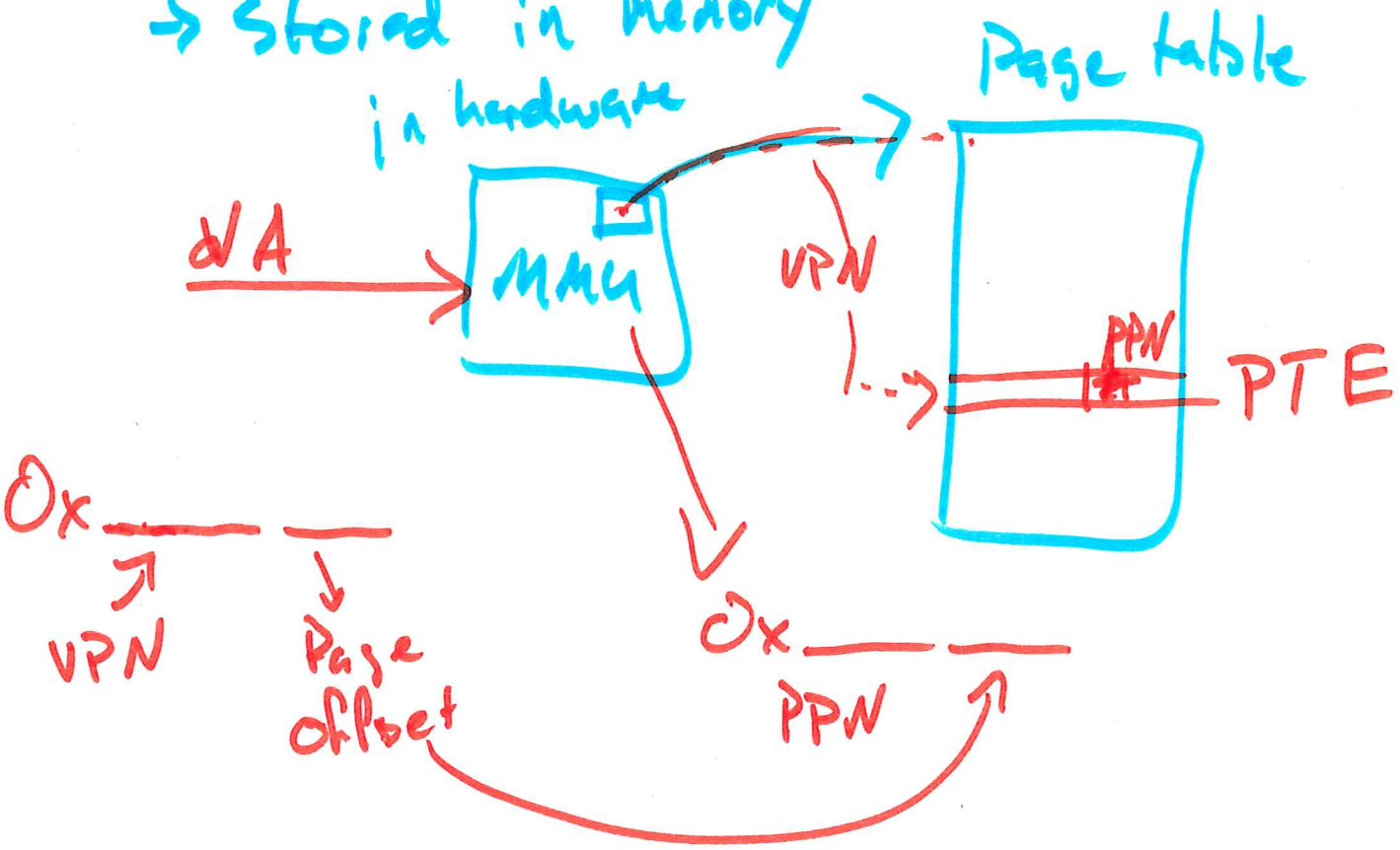
↳ "page hit"

→ not in memory page miss page fault

Where is the page table?

→ stored in memory

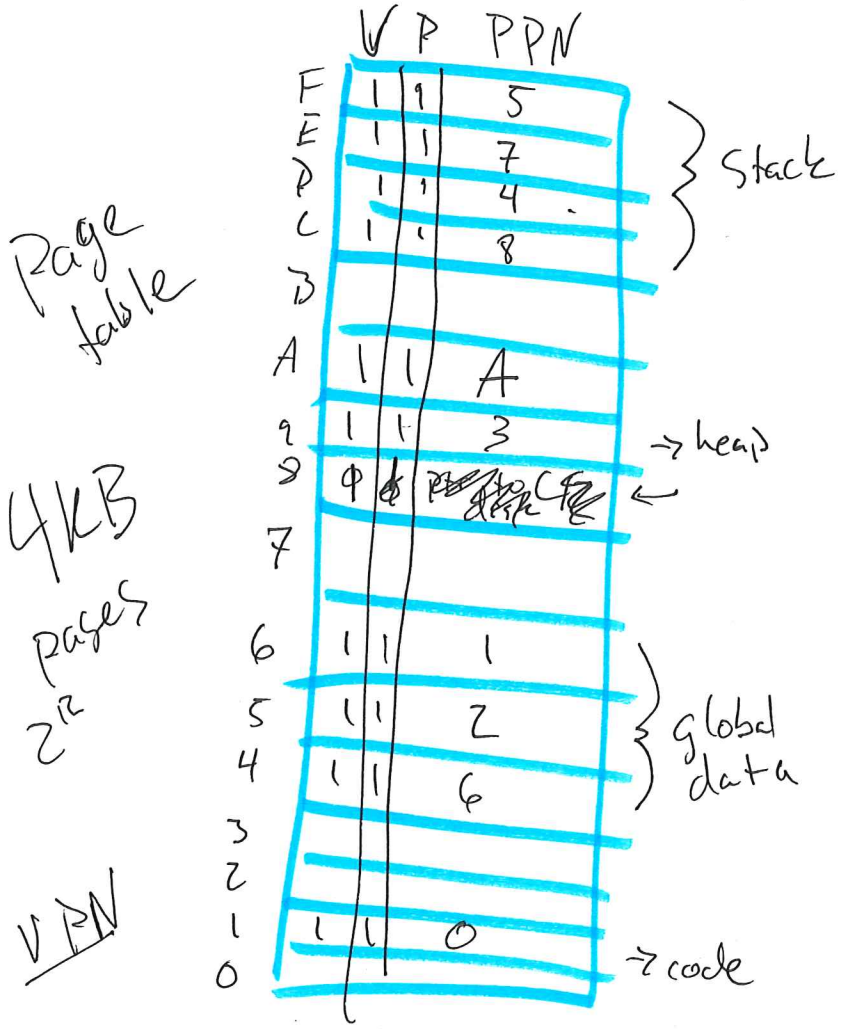
in hardware



MMU has a pointer to the page table → physical address (CR3)

each process has unique page table and unique PT pointer

Multi-level page table



0x8A14 → 0xCA14

0x8 = VPN, A14 = offset

~~Invalid~~

If the MMU finds an invalid entry → Page fault

or not present

CPU → MMU → 0x4010

0xD010

VPN

↳ page offset