Virtual Address = 0x0020

VPN | VPO

~~00~~00 000000 | 10 0000

$|VA| = 14$ bits $\Rightarrow$ $|VM| = 2^{14} = 16$ KB

$|PA| = 12$ bits $\Rightarrow$ $|PM| = 2^{12} = 4$ KB

$|Page| = 64$ bytes $\Rightarrow$ $|PO| = 6$ bits $\Rightarrow$ $64 = 2^6$

	VPN	VPO
VA	8	6

$|VPN| = 8$ bits $\Rightarrow$ # of virtual Page $= 2^8 = 256$ = # of PTE in PT

	PPN	PO
PA	6	6

$|PPN| = 6$ bits $\Rightarrow$ $2^6 = 64$ Physical Page

TLB : 16 entries (PTE) , 4-way associative

$16 = 4 \times S \Rightarrow S = 4$

	Tag	PPN	V			
00						
01						
10						
11						

TLB Tag	TLBI
6	2

Virtual Memory (VM) is an array of N contiguous bytes stored on disk.

The contents of the array on disk are cached in physical memory (DRAM cache)

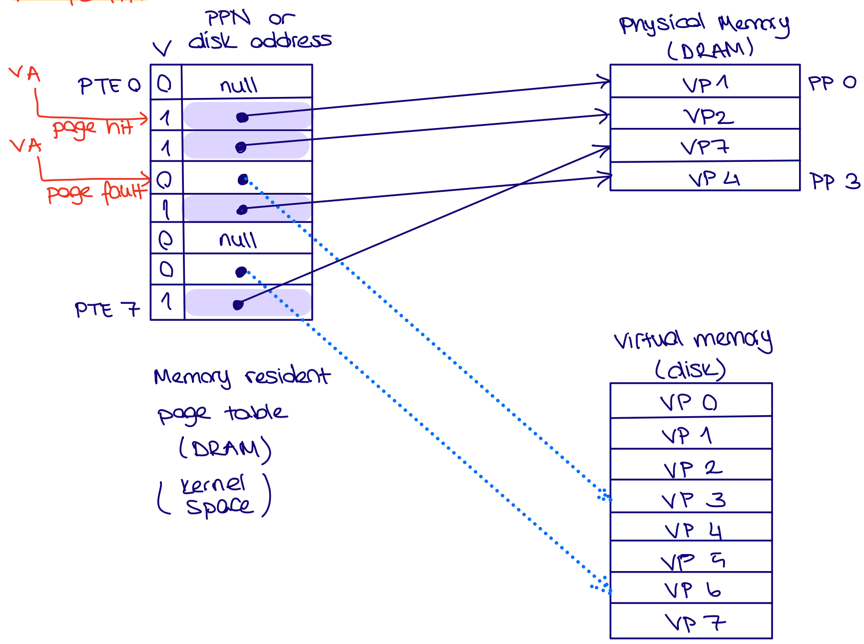
defer

Write back rather than write through → write immediately to memory

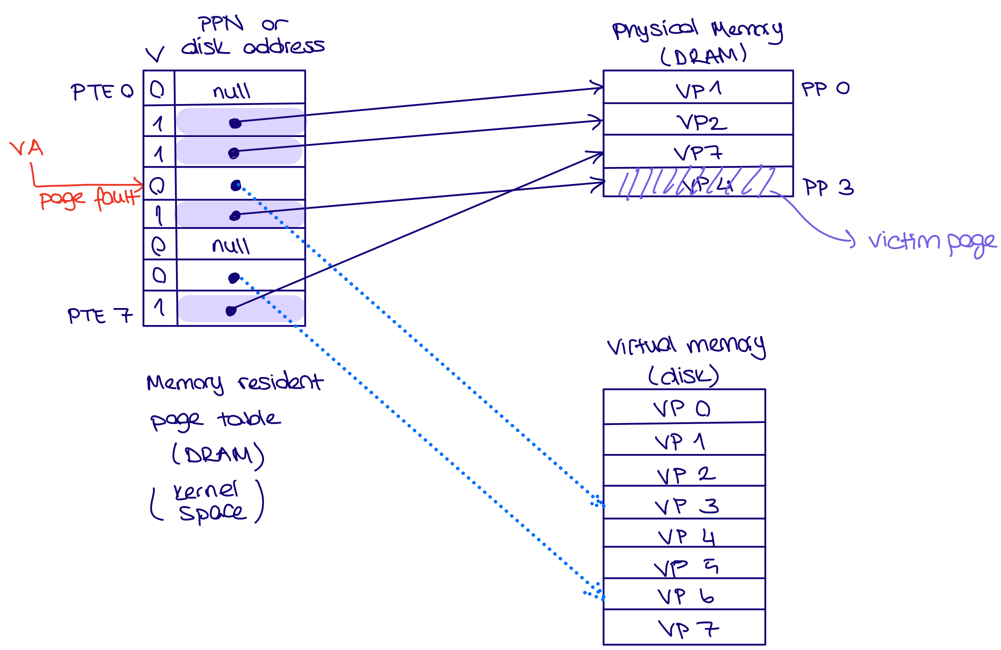
↳ defer write to memory until replacement of line ⟹ need a dirty bit (line different from mem or not)

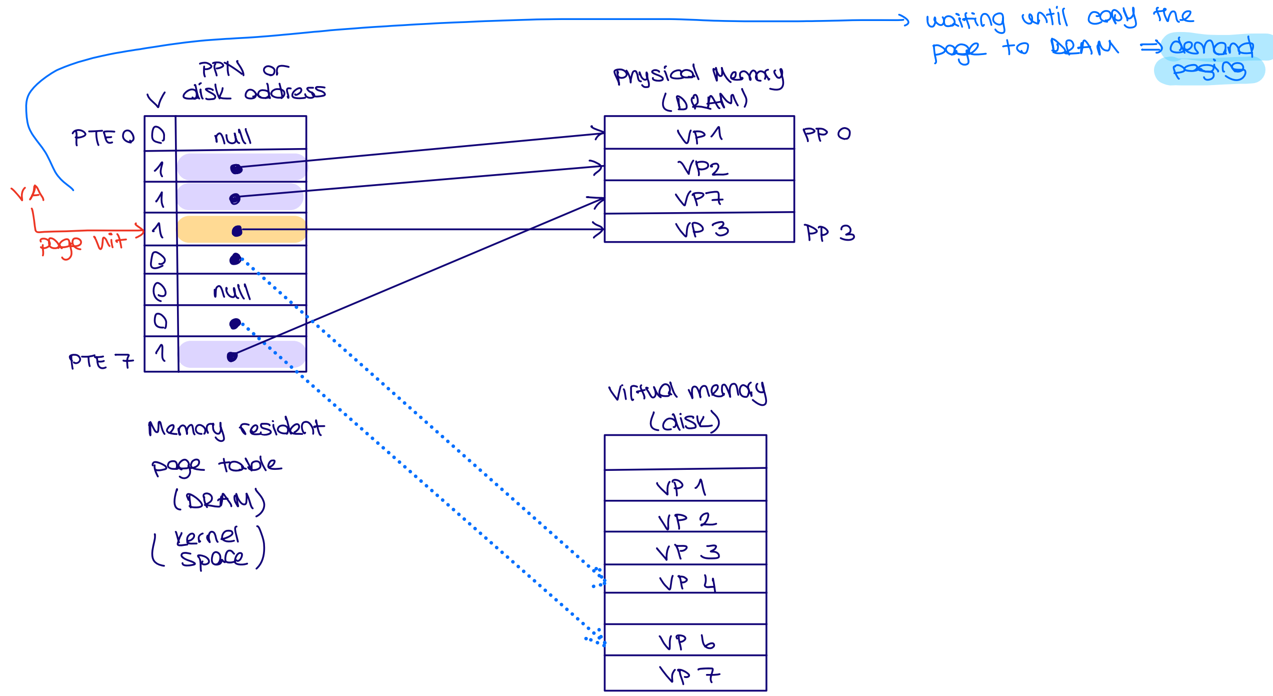
D: 0 ⟹ mem has old value
D: 1 ⟹ write back

PAGE HIT

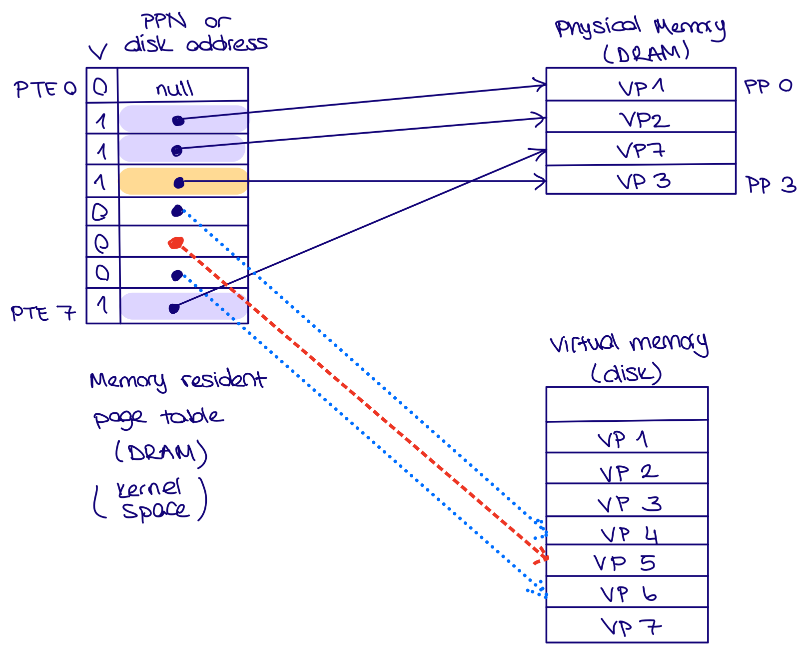


PAGE FAULT





ALLOCATING NEW PAGE OF VM → VP 5



VM works because of locality.

At any point in time, programs tend to access a set of active VPs called working set.

if (working set size < main memory size)

Good performance for one process after compulsory misses.

if (sum (working set sizes) > main memory size)

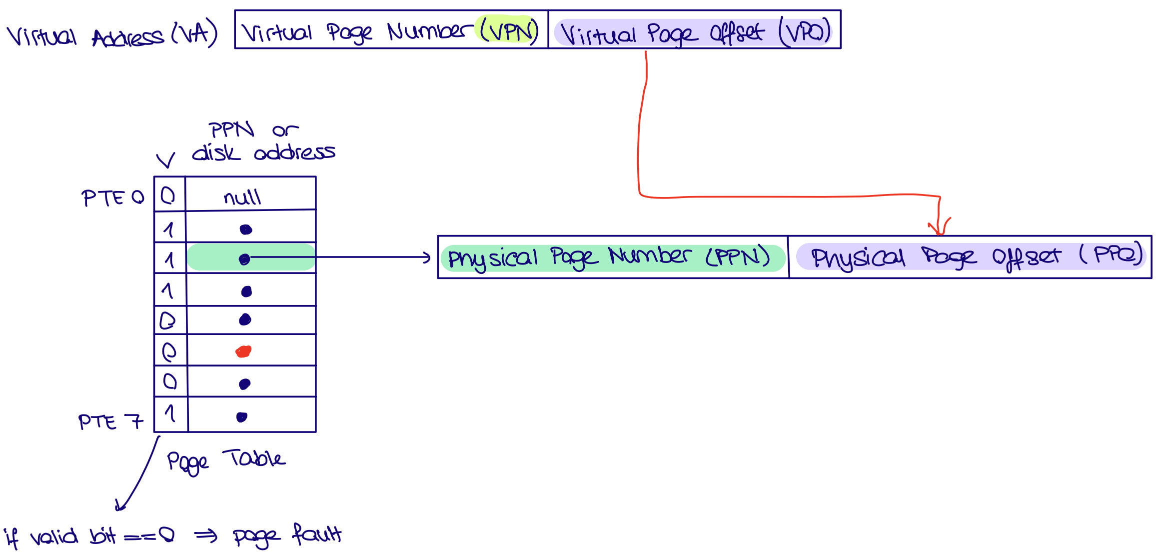
Trashing → performance meltdown → pages are copied in and out continuously

Each process has its own virtual address space

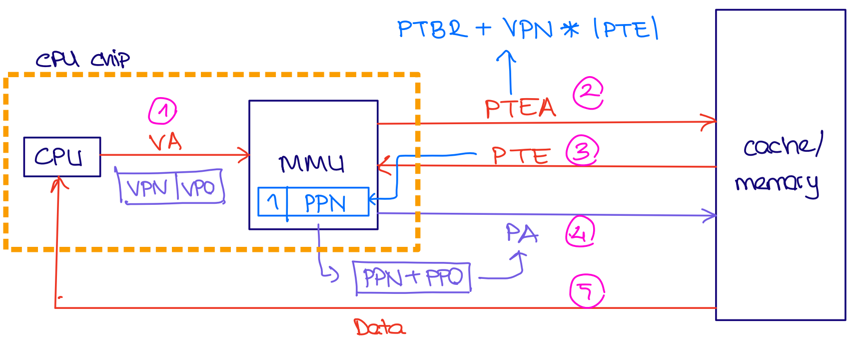
Each virtual page can be stored in different physical pages at different times

Virtual pages for different processes can be mapped to the same physical page.

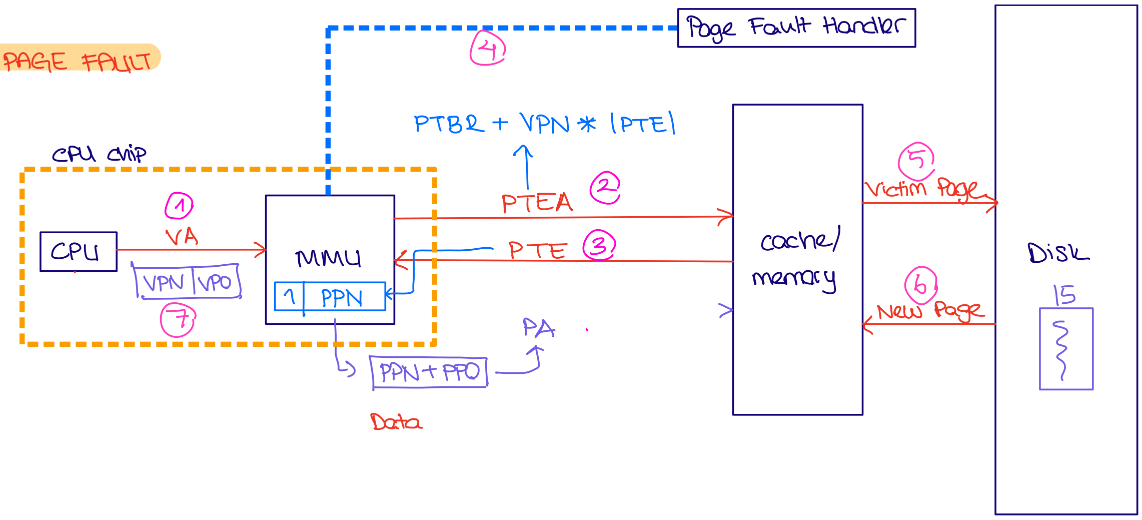
ADDRESS TRANSLATION



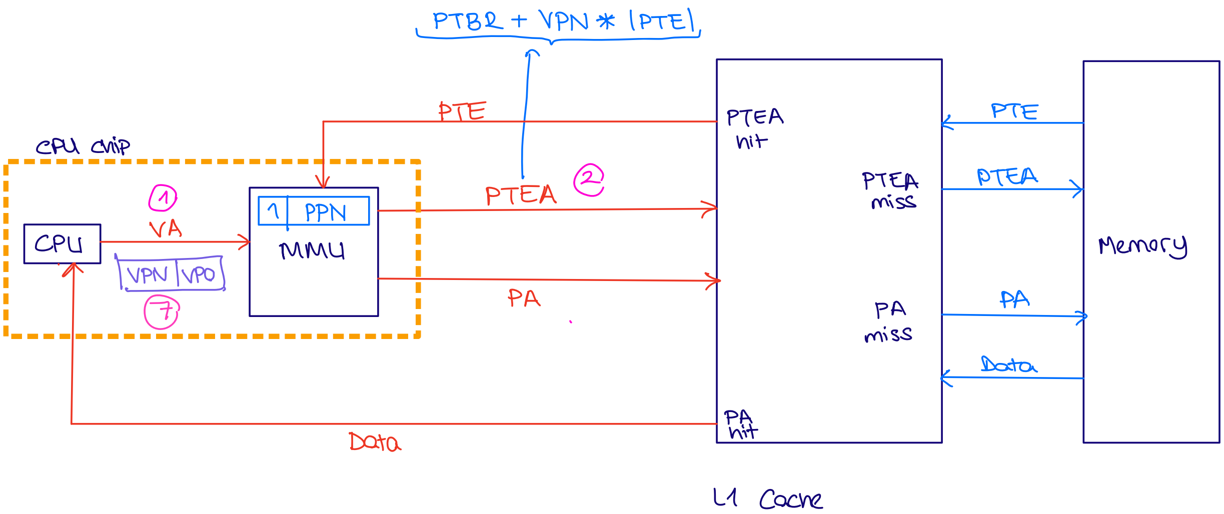
PAGE HIT



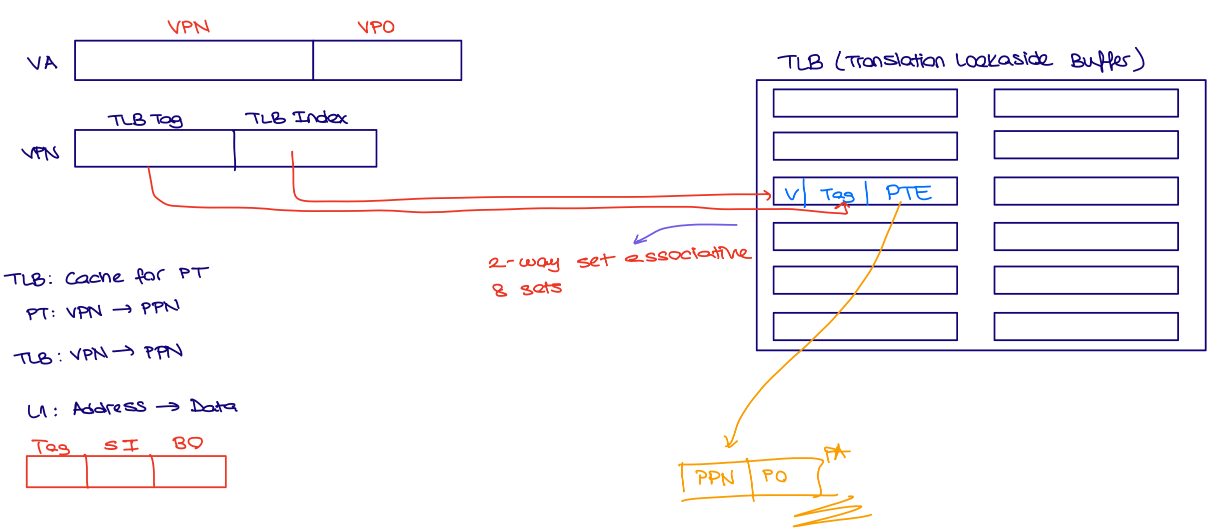
PAGE FAULT



CACHE AND VM



TRANSLATION LOOKASIDE BUFFER



TLB: Cache for PT

PT: VPN → PPN

TLB: VPN → PPN

L1: Address → Data

Tag	SI	BO

32-bit system IA32

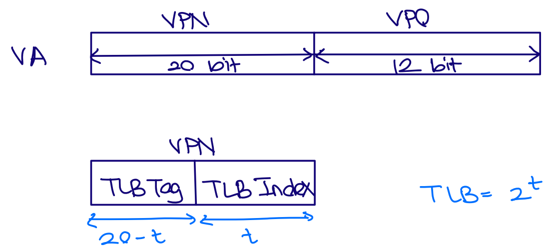
$|VM| = 2^{32} = 4\,GB$

$|VA| = 32$-bits

$|Page| = 4\,KB = 2^{12}$ bytes

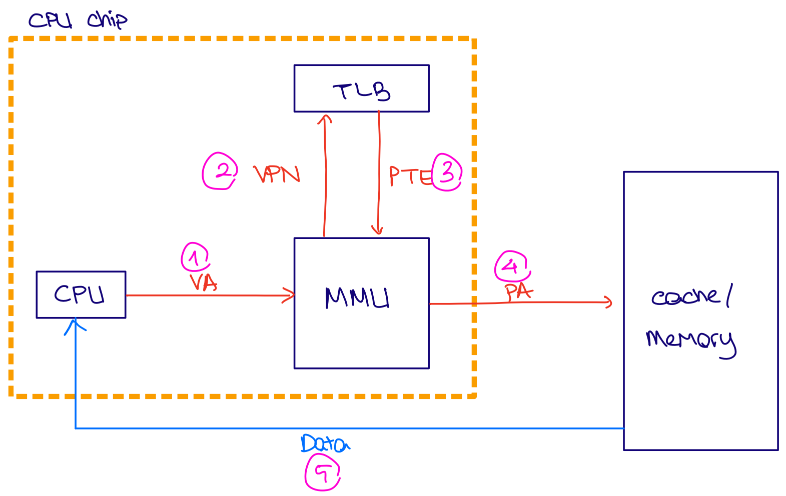
$|PM| = 1\,GB = 2^{30}$

$|PA| = 30$ bits

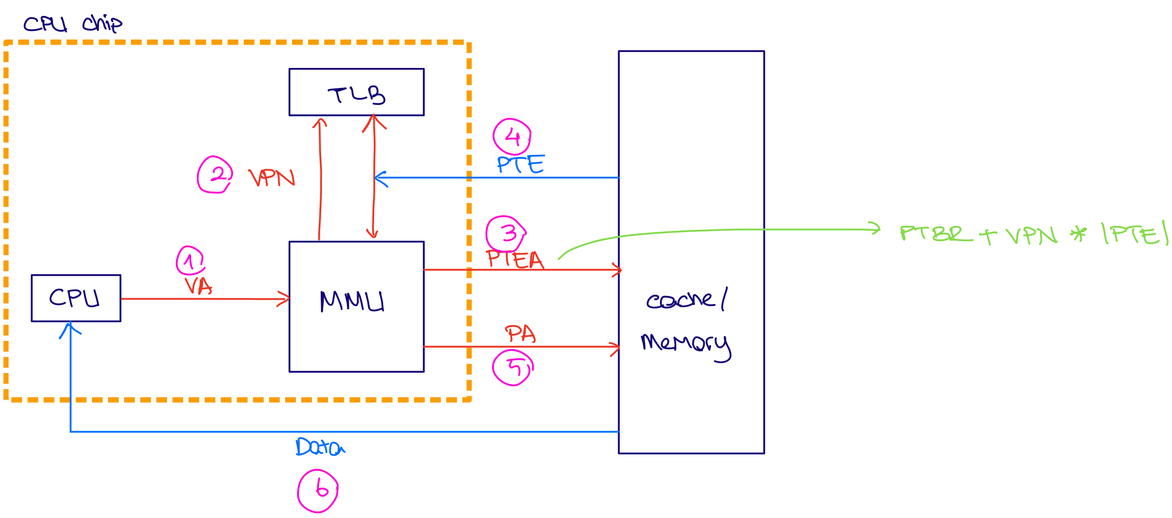


TLB = 2^t sets

TLB HIT + PAGE HIT

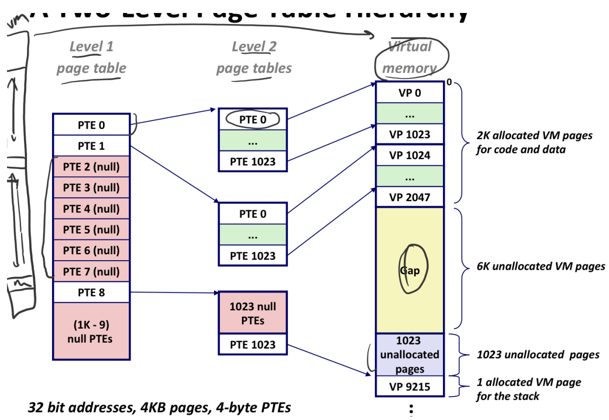


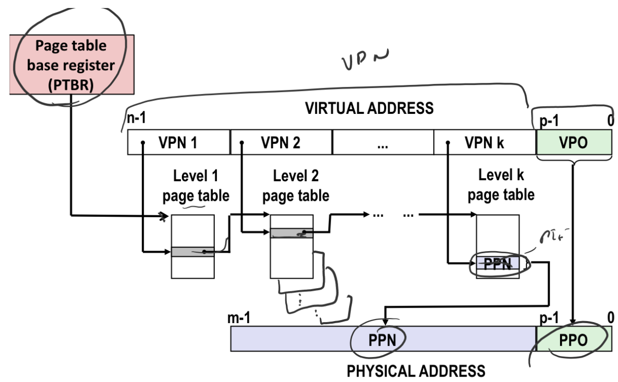
TLB MISS + PAGE HIT



MULTI-LEVEL PAGE TABLES

2-level Page Table
- Level 1 Table ⇒ each PTE points to a page table (always memory resident)
- Level 2 Table ⇒ each PTE points to a page (paged in and out)





ÖRNEKLER

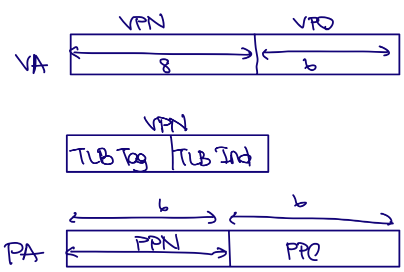
Memory Addressing

14-bit virtual addresses $|VA| = 14$

12-bit physical addresses $|PA| = 12$

Page size = 64 bytes $\Rightarrow 2^6 \Rightarrow |VPO| = 6 = |PPO|$

$|VPN| = 8$



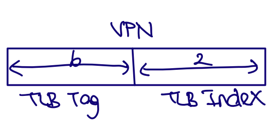
TLB

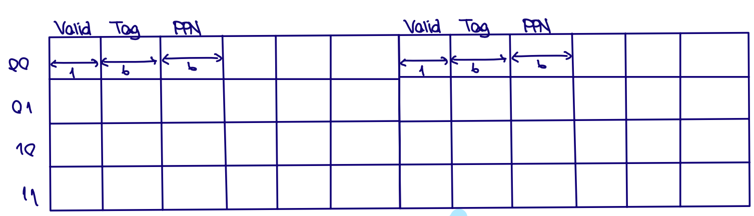
16 entries in total

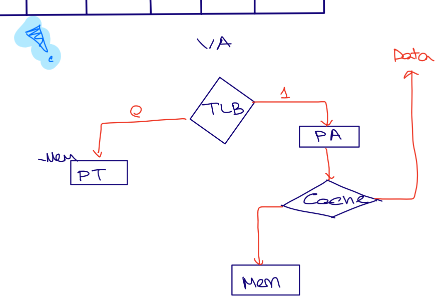
4-way associative

⟩ 4 set $\Rightarrow 2^2 \Rightarrow$ 2 bit for TLB Index

in each set I have 4 lines







Memory System Cache

16 lines, 4 byte block size $|CO| = 2^{2} \Rightarrow 2$

Physically addressed $\Rightarrow$ 12 bits $= |PA|$

Direct mapped $|CI| = 16 = 2^4 \Rightarrow 4$

PA

C Tag	Cache Index	Cache Offset
6	4	2
PPN	PPO	

Virtual Address $\Rightarrow$ 0x03D4

0000 11 11 | 01 0100

VPN | VPO

0x0F

VA

VPN	VPO
8	6

VPN

6	2
TLB Tag	TLB Index

000011 | 11

TLB Tag TLB Ind $\Rightarrow$ PPN = 0D

$\equiv$ 0x03 $\equiv$ 0x3

PA

PPN	PPO
6	6

PPN + PPO $\Rightarrow$ 001101 | 0101 | 00

C tag CI CO

$\equiv$ 13 = 0x0D $\equiv$ 5 $\equiv$ 0

PA

C Tag	Cache Index	Cache Offset
6	4	2
PPN	PPO	

$\boxed{36}$

Virtual Address $\Rightarrow$ 0x0020

00 | 00 0000 00 | 10 0000

VPN = 0x0 VPO = 0x20

VA

VPN	VPO
8	6

VPN

6	2
TLB Tag	TLB Index

000000 | 00

TLB Tag TLBI

= 0x0 = 0x0 $\Rightarrow$ valid = 0

$\hookrightarrow$ PT $\Rightarrow$ 28 = PPN

PA

PPN	PPO
6	6

0010 1000 | 1000 | 00

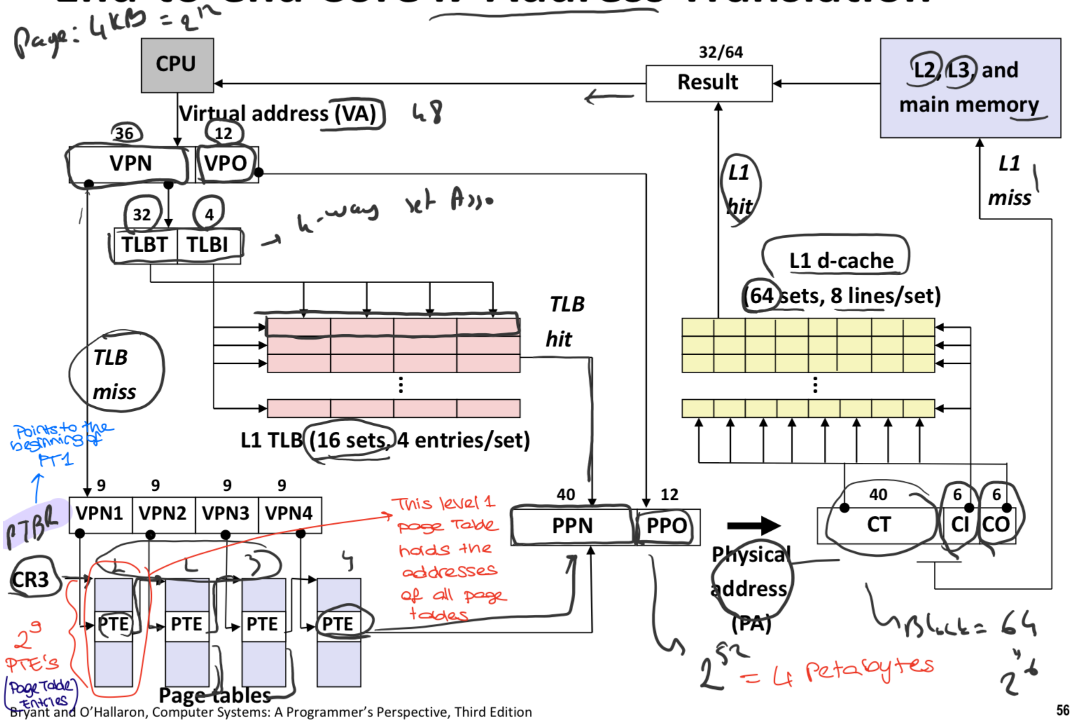
C Tag CI CO

0x28 = 0x8 0x0

PA

C Tag	Cache Index	Cache Offset
6	4	2
PPN	PPO	

End-to-end Core i7 Address Translation



CORE i7 Level 1-3 Page Table Entries

40 bits = |PPN| → but it points to the other levels PT → |PPN| = |PTE|

63	62–52	51–12	11–9	8	7	6	5	4	3	2	1	0
XD	Unused	Page table physical base address	Unused	G	PS		A	CD	WT	U/S	R/W	P=1
Available for OS (page table location on disk)												P=0

Virtual Address Space of a Linux Process

VM