Week 9.b
CS6640
11/02 2023
https://naizhengtan.github.io/23fall/

1. Virtual memory intro
2. Paging?
3. Page table?
4. Today's virtual memory

Admin

- midterm
- lab5 (next Monday)
- lab4: the first 3 exercises

1. VMem intro

Q: WHY? PMem?

isolate process' memory

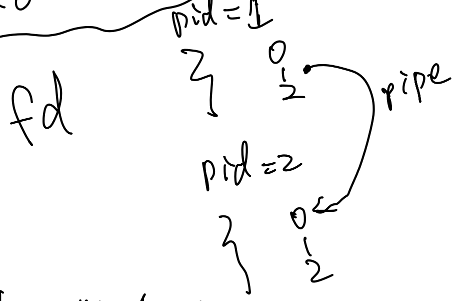
security

- benefits:

(a) programmability

① self-owned huge contiguous memory ✓

② multiplexing addresses. 0x800000



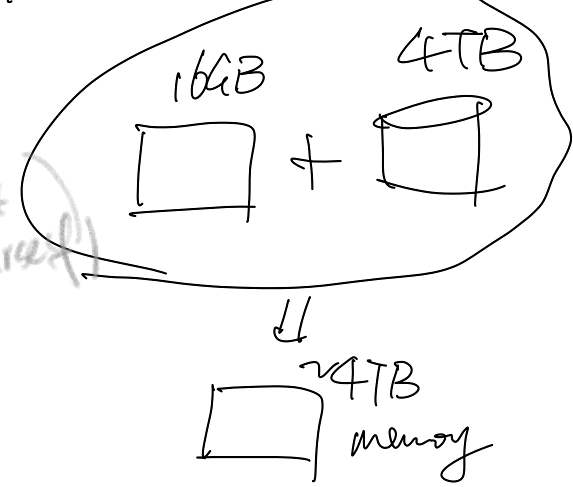
(b) protection

① separate address spaces X

② access control Priv level: M/S/U
OP: r/w/x

(part)

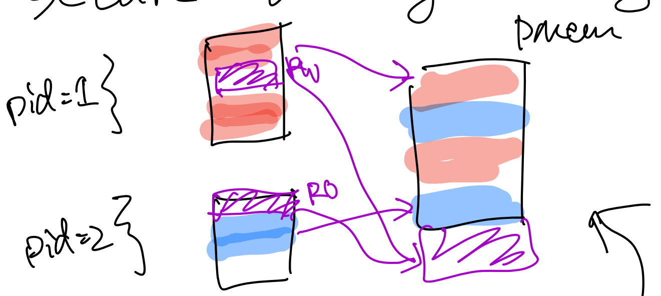
[meltdown
spectre]



(c) effective use of resources

① overcommitting memory ✗ (do it yourself)

② secure sharing memory

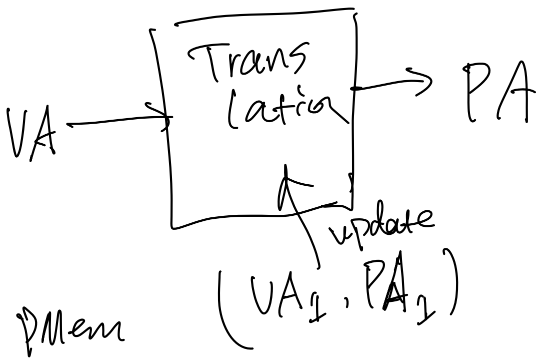


Q: PMem can achieve points? Why, why not?

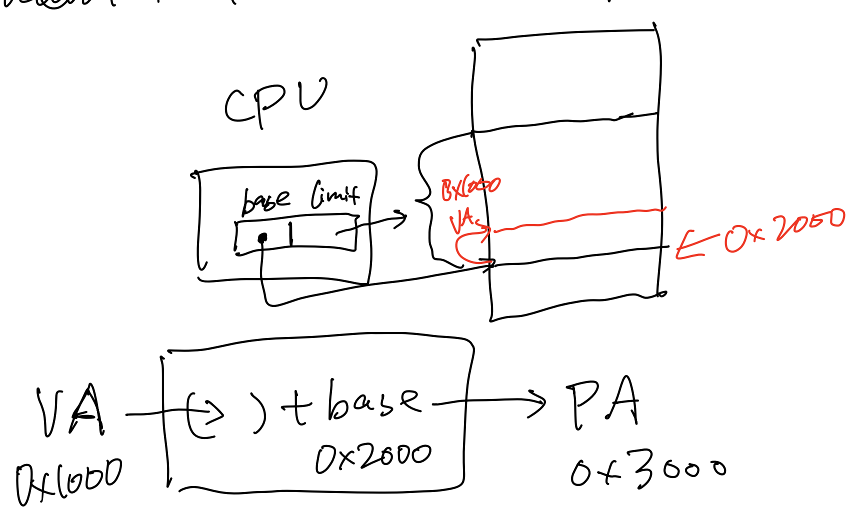
VMem → ① translation ⇐
→ ② protection (week7)

2. Paging?

- the translation problem:
    VA => PA
  and hope this translation
    (i) runs fast,
    (ii) has small memory overhead,
    (iii) can be updated quickly.



① Segmentation



Q: Cons of Seg?

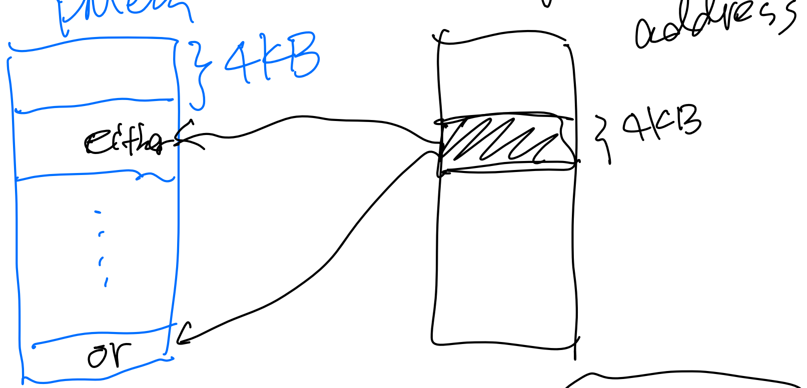
- fragmentation
- complexity

② pages

PMem

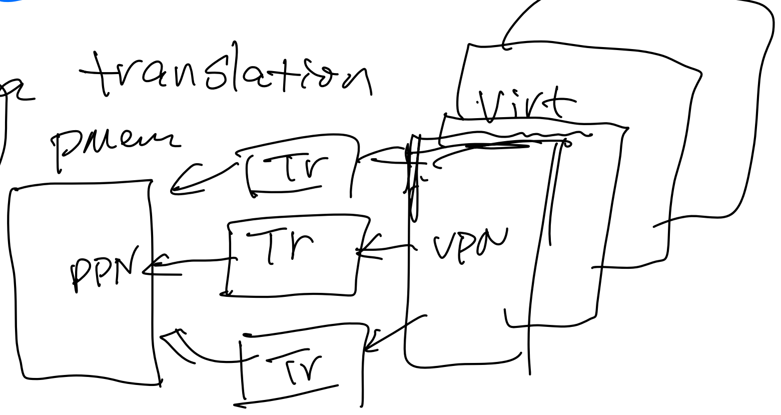
Virtual

$$\begin{tikzcd} VA \arrow[r] \arrow[d] & PA \arrow[d] \\ VPN \arrow[r] & PPN \end{tikzcd}$$

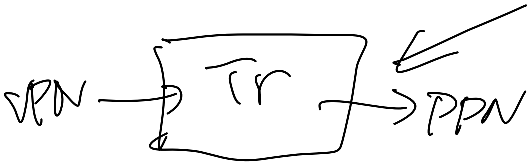


③ "Who" owns a translation

a group of processes share a translation?
⇓
threads.

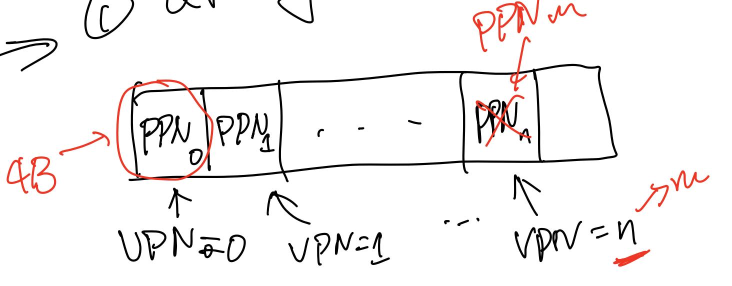


3. Page table?



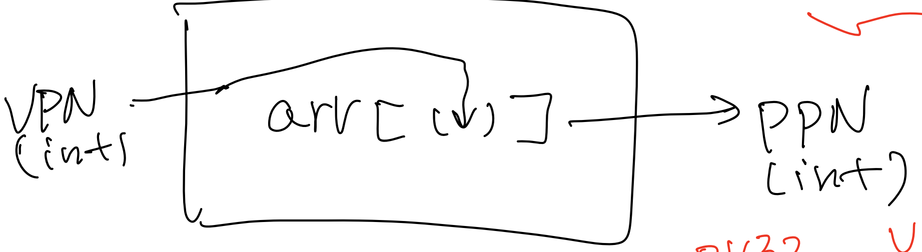
data structure

① array



2^{20}

Translation



$2^{32}B = 4GB$? → fast (v)

(v) ⇒ small (?) →

? ↗ update (v)

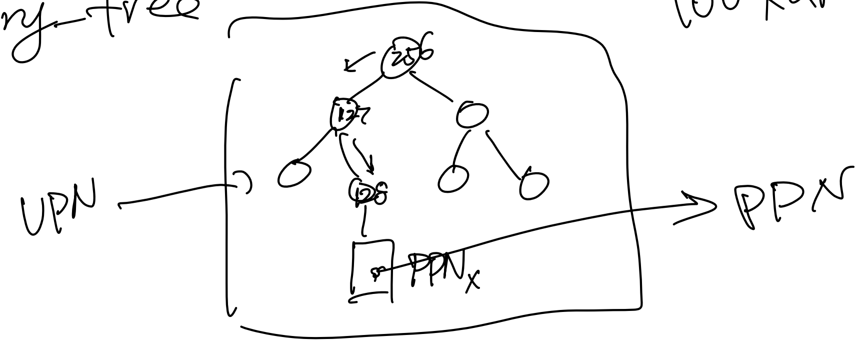
RV32. VA (32bit) offset=12bits

VPN: 20bit) (64-12) = 52

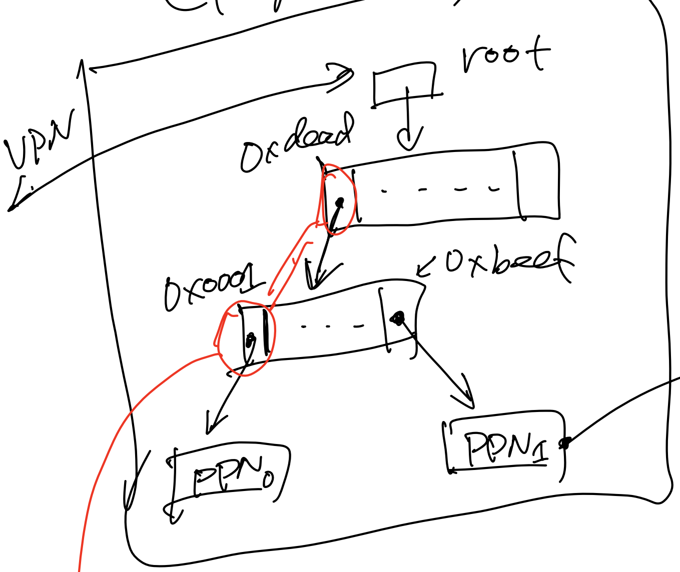
size(arr) = $2^{20} \times 4B = 4MB$ per process

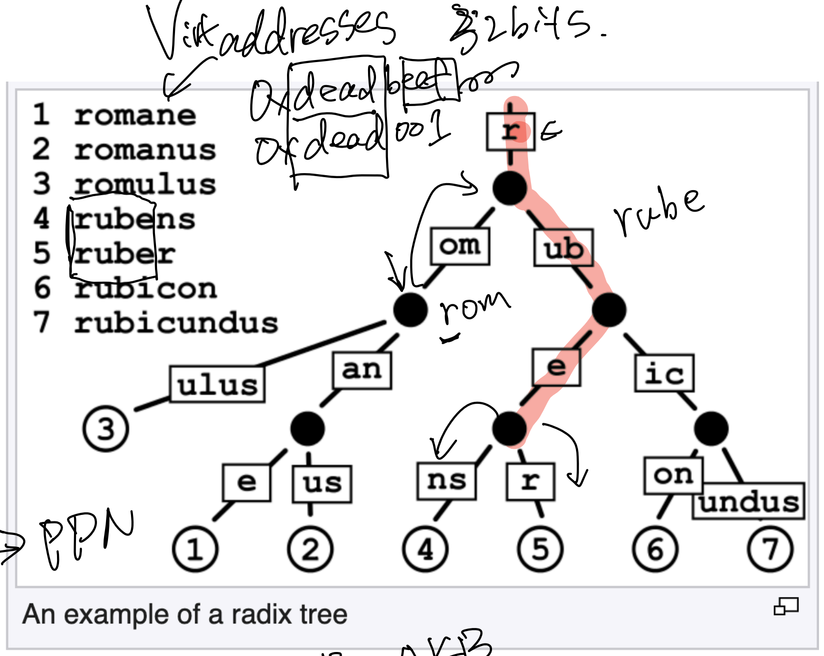
100 × 4MB = 400MB

② binary tree • VPN=128



③ radix tree (Page table)



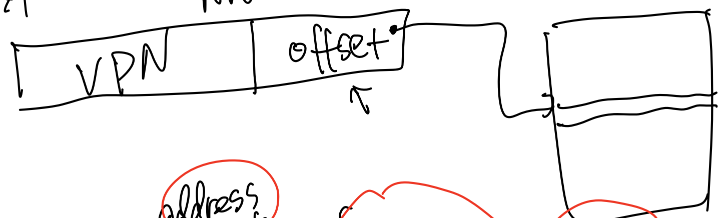


1 romane
2 romanus
3 romulus
4 rubens
5 ruber
6 rubicon
7 rubicundus

An example of a radix tree

VA (32-bit) $1B * 2^{12} = 4KB$

$MAU * 2^{offset} = PageSize$



⇙ Rv32

- page table design space:
  * offset (12 bits)
  * page size (4KB)
  * address length (32 bits)
  * addressable memory unit (1B)
  * depth of the PT (2 layer)
  * PTE size (4B)

page table entry

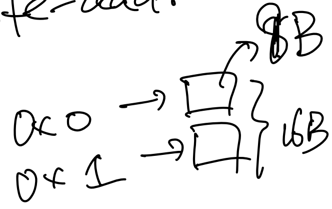
$$2^{\text{address length}} = \left(\frac{\text{Page Size}}{\text{PTE Size}}\right)^{\text{depth}} \cdot \text{page size}$$

fan out of the tree

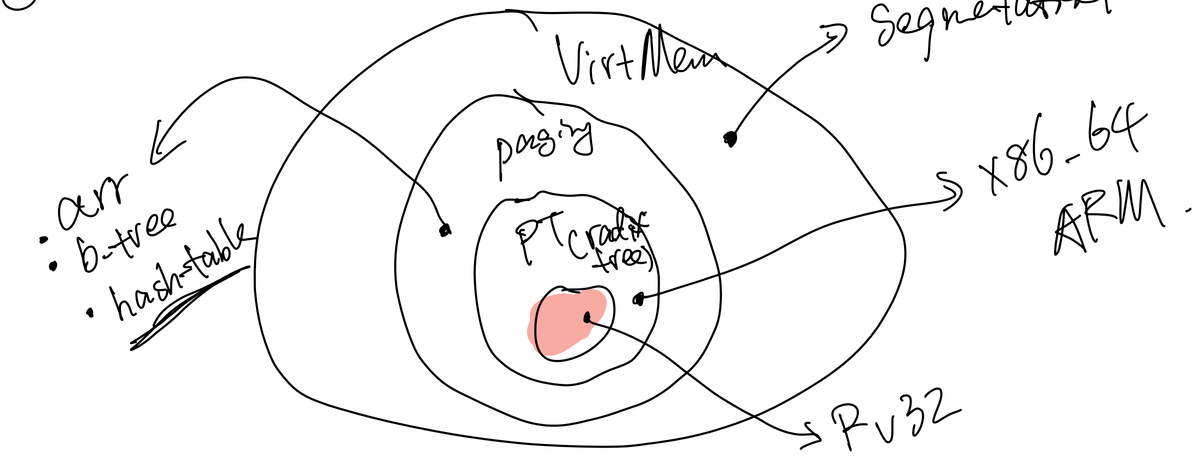
$$\frac{4KB}{4B} = 1024$$

$$2^{\overset{VA}{32}} B = (1024)^2 \cdot 4KB = 4GB$$

Byte-addressable



4. today's Virtual mem



VirtMem
segmentation
paging
PT (radix tree)
x86_64
ARM.
Rv32
· arr
· b-tree
· hash-table