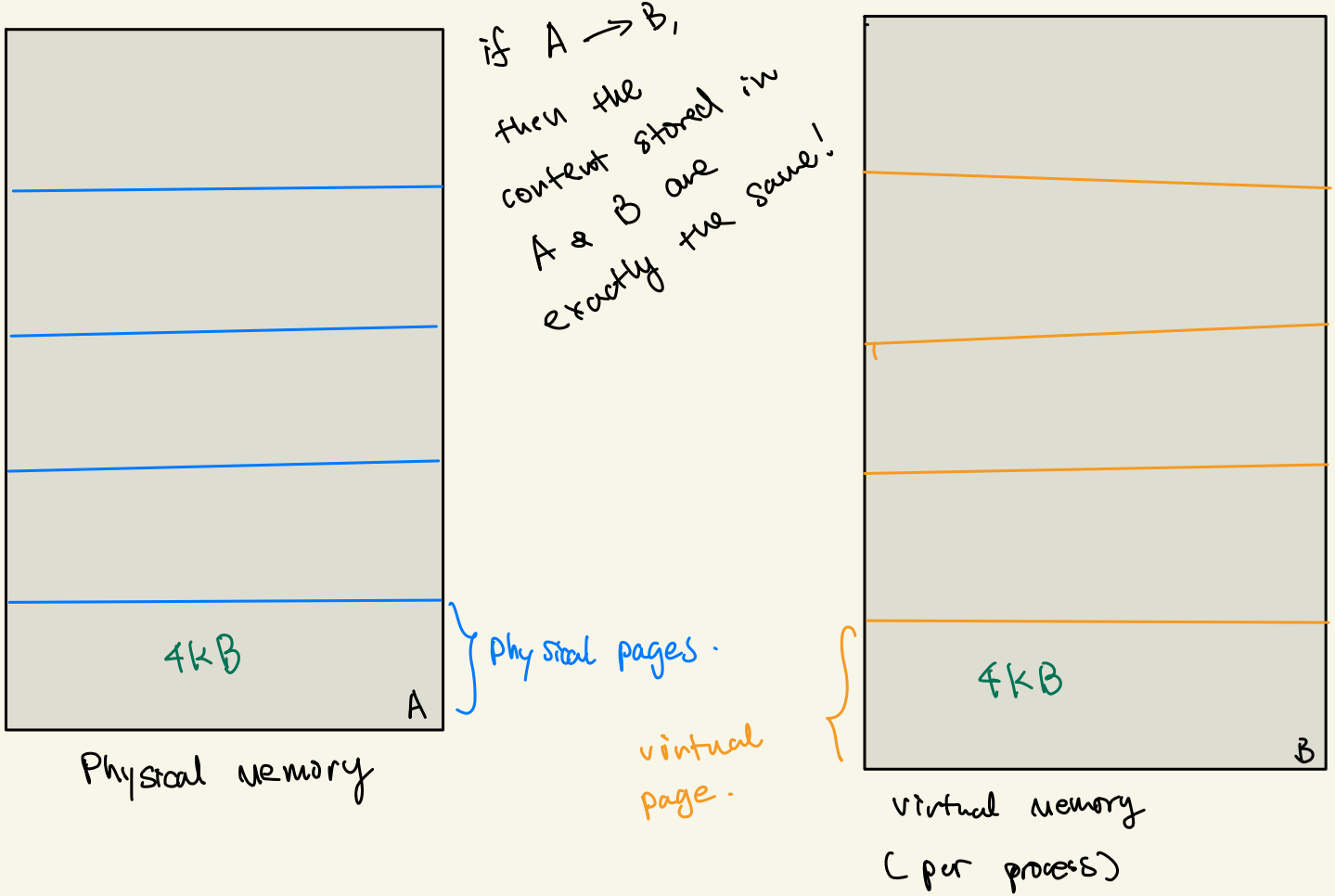
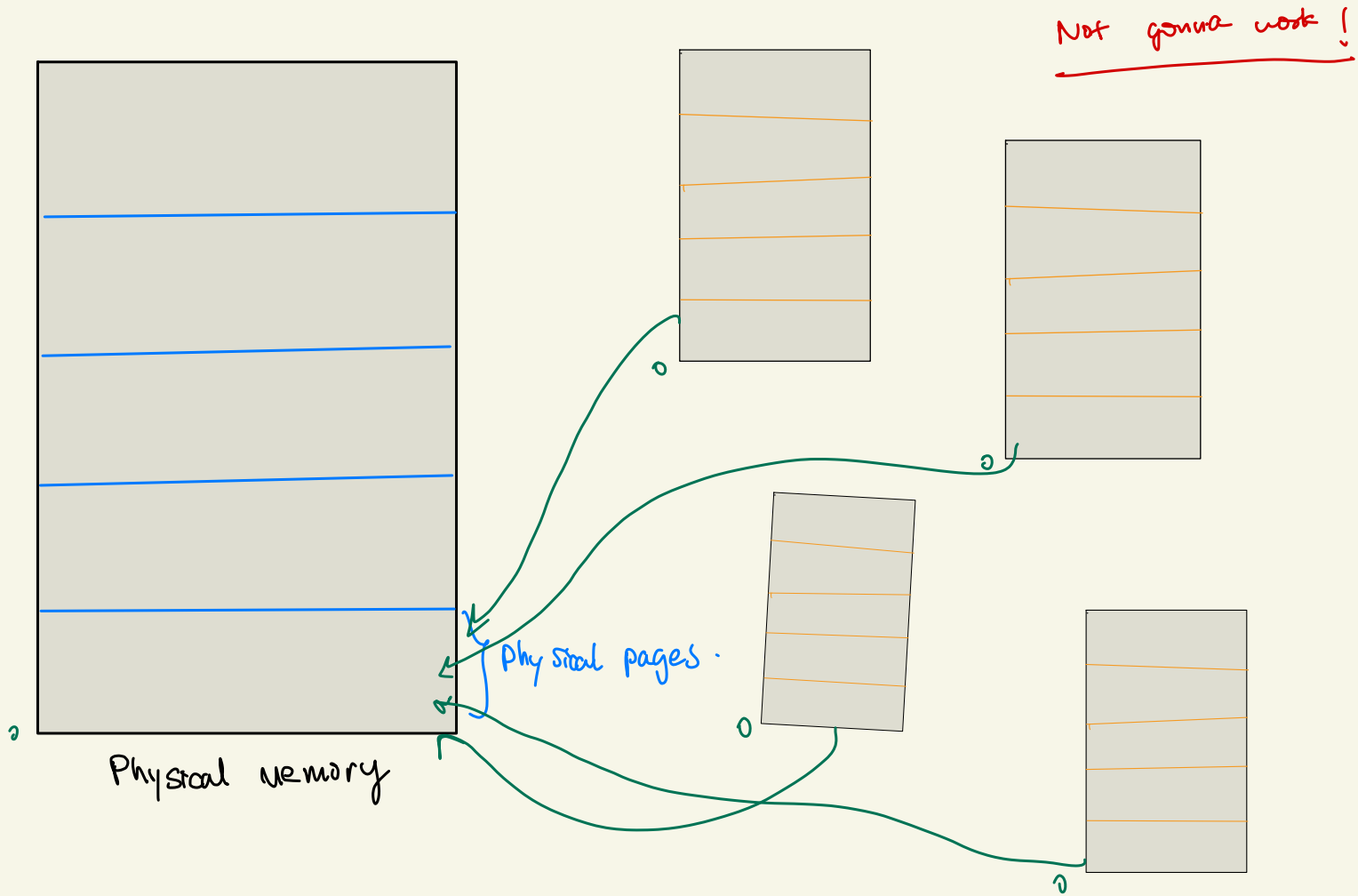
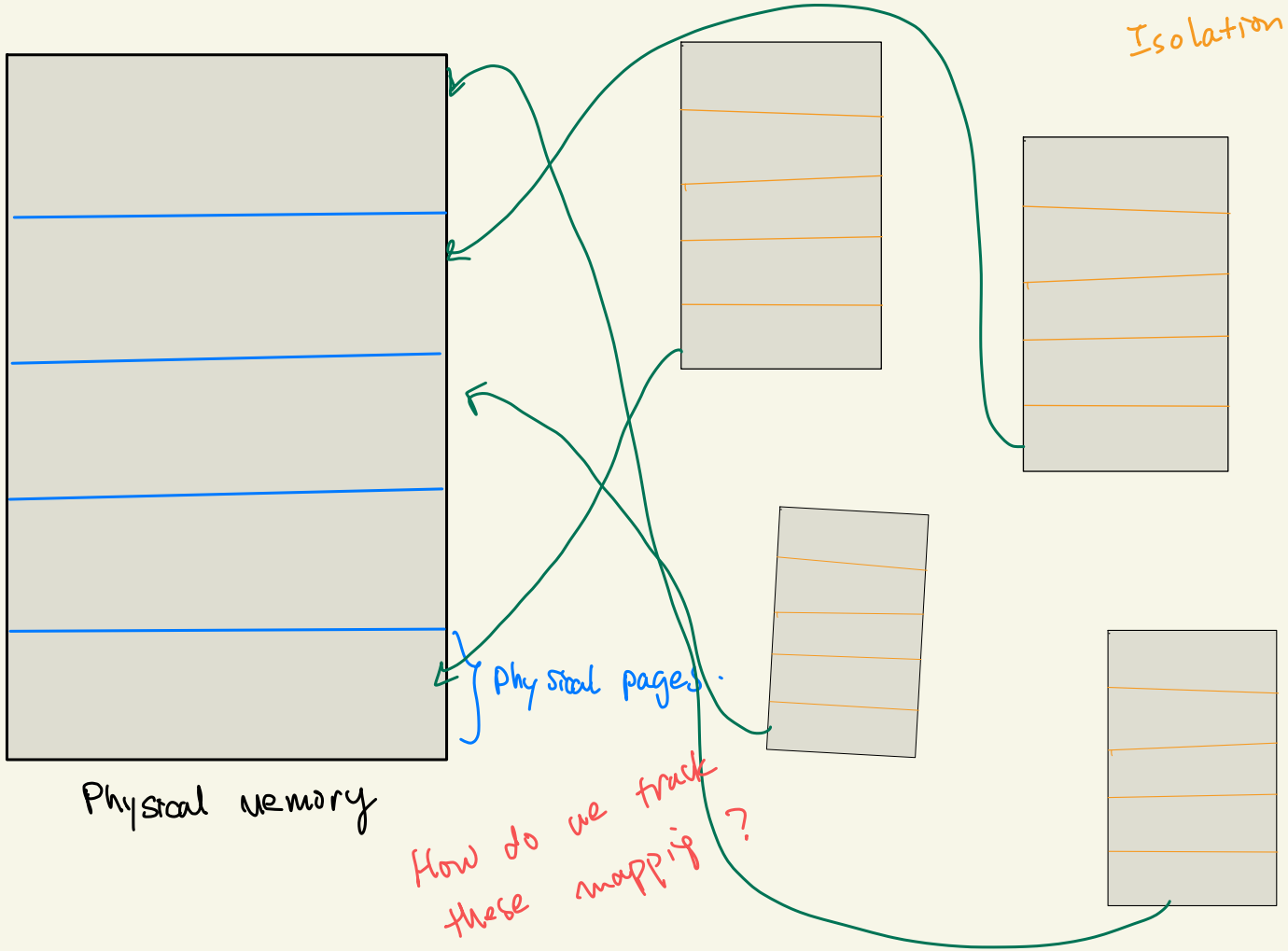

if A → B,
then the content stored in A & B are exactly the same!
4kB
A
Physical pages.
Physical memory
virtual page.
4kB
B
virtual memory
(per process)


Not gonna work !
Physical pages.
Physical memory


Isolation!
Physical pages
Physical memory
How do we track these mapping?

Page Table!

Virtual page i → Physical Page j.

(VPN) (PPN)

No, Page Table is per-process !!!

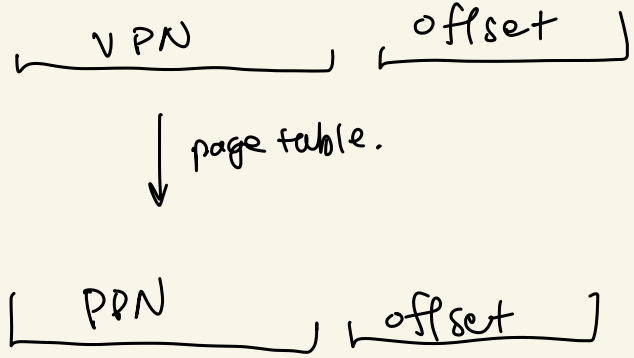
VA: 0000 ···· 0 1001 ··· 1

VPN 12-bit offset

PA: 000 ··· 10000 1 ··· 1

PPN 12-bit offset

So, inorder to find the PA of a VA, we do.



"isn't it possible to have the same VA for multiple different process?" → "doesn't it mean same VA gonna map to the same PA?"

What does a linear page table look like?
(naive version)

You can think of this as: dictionary [VPN, PPN]
↑ key ↑ value

but that require 2 entry per mapping.

So, we use table instead. ith index of the page table is the VPN #, the value correspond to the row index is the PPN #.

(i.e. this is just a array ...)

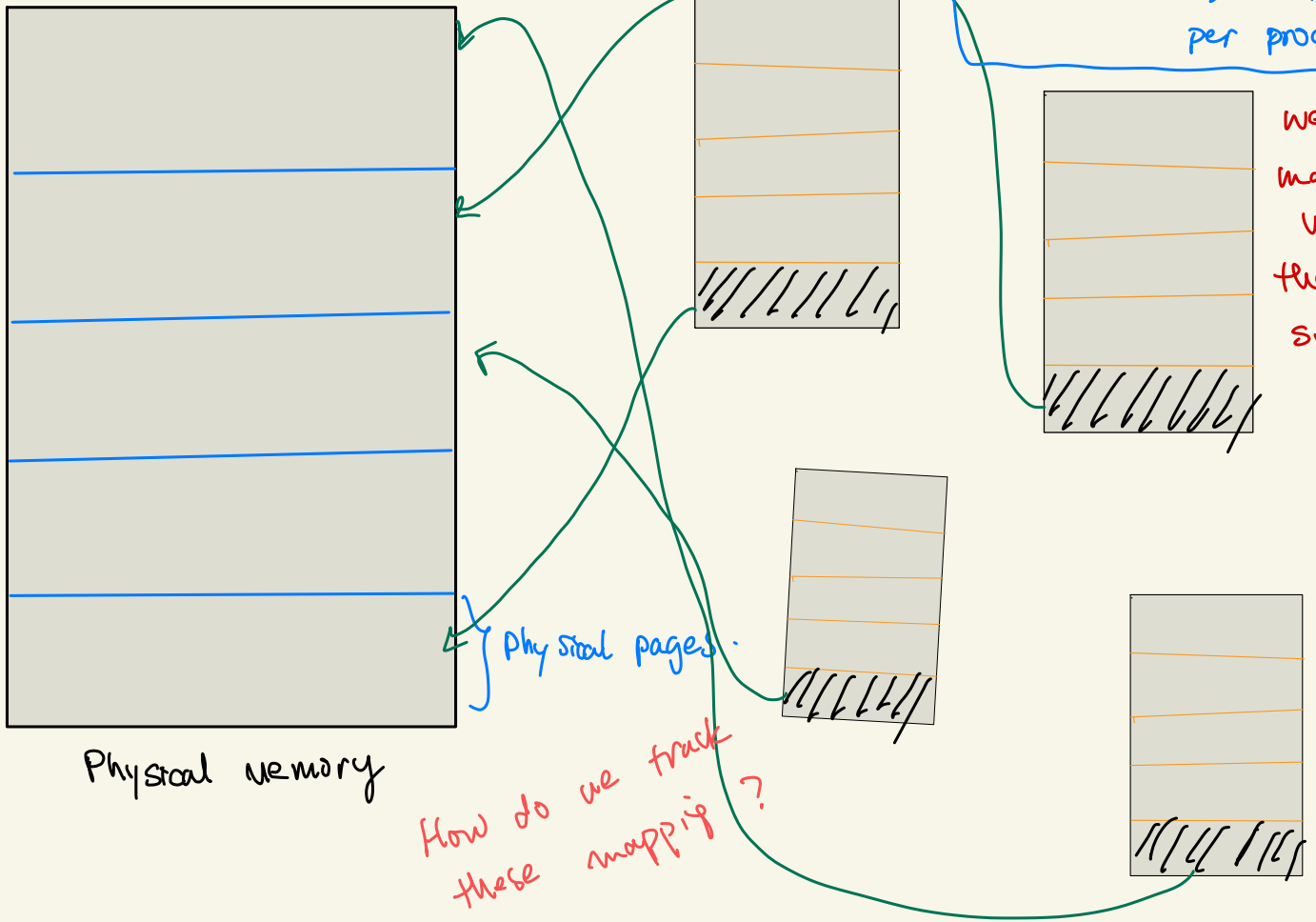
Problem of linear PT: too large!!!

Assuming each PPN contains 8 bytes.

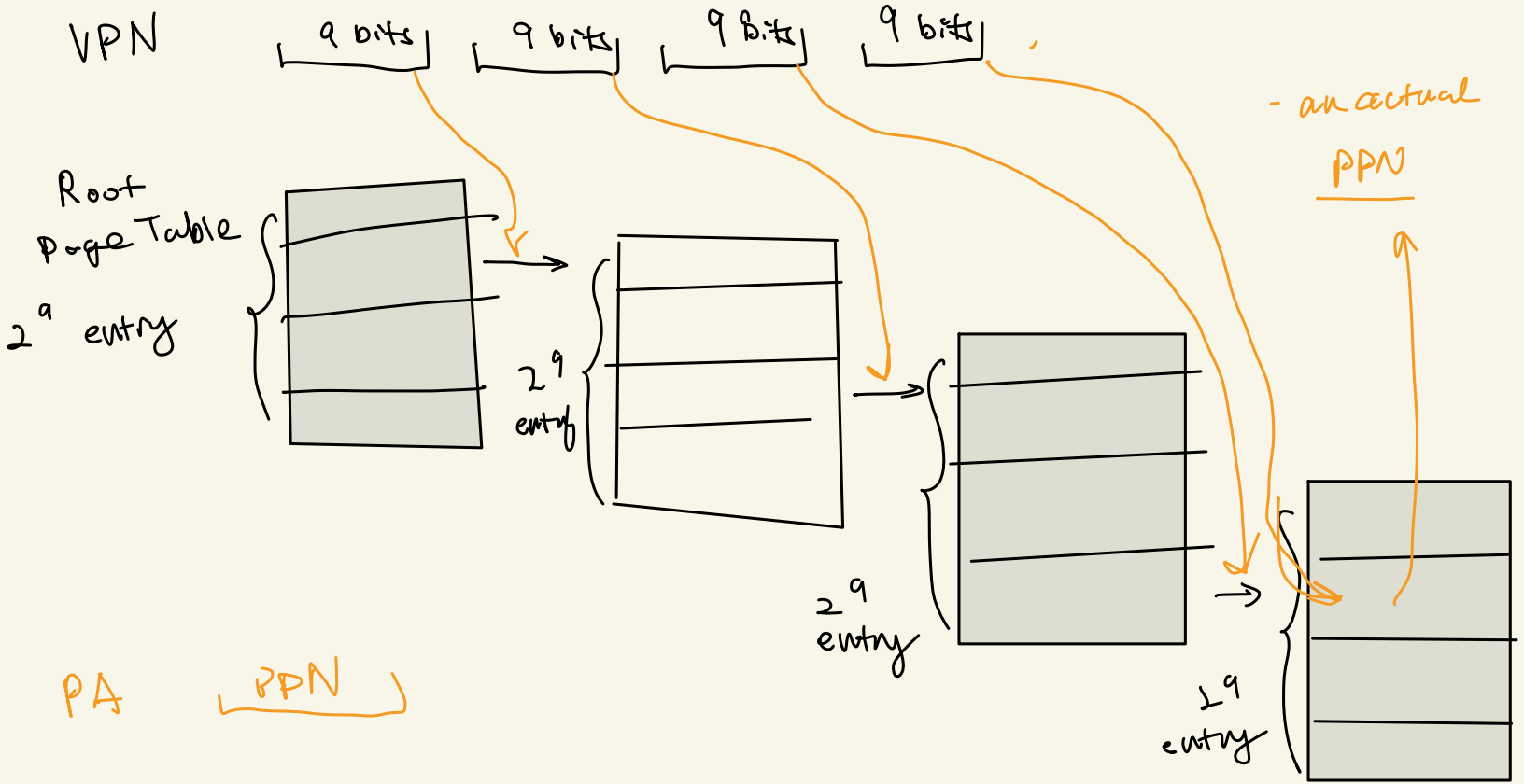
there are 2^{36} different VPN,

The size of the PT itself is $2^{36} \cdot 8 = 512$ GB!

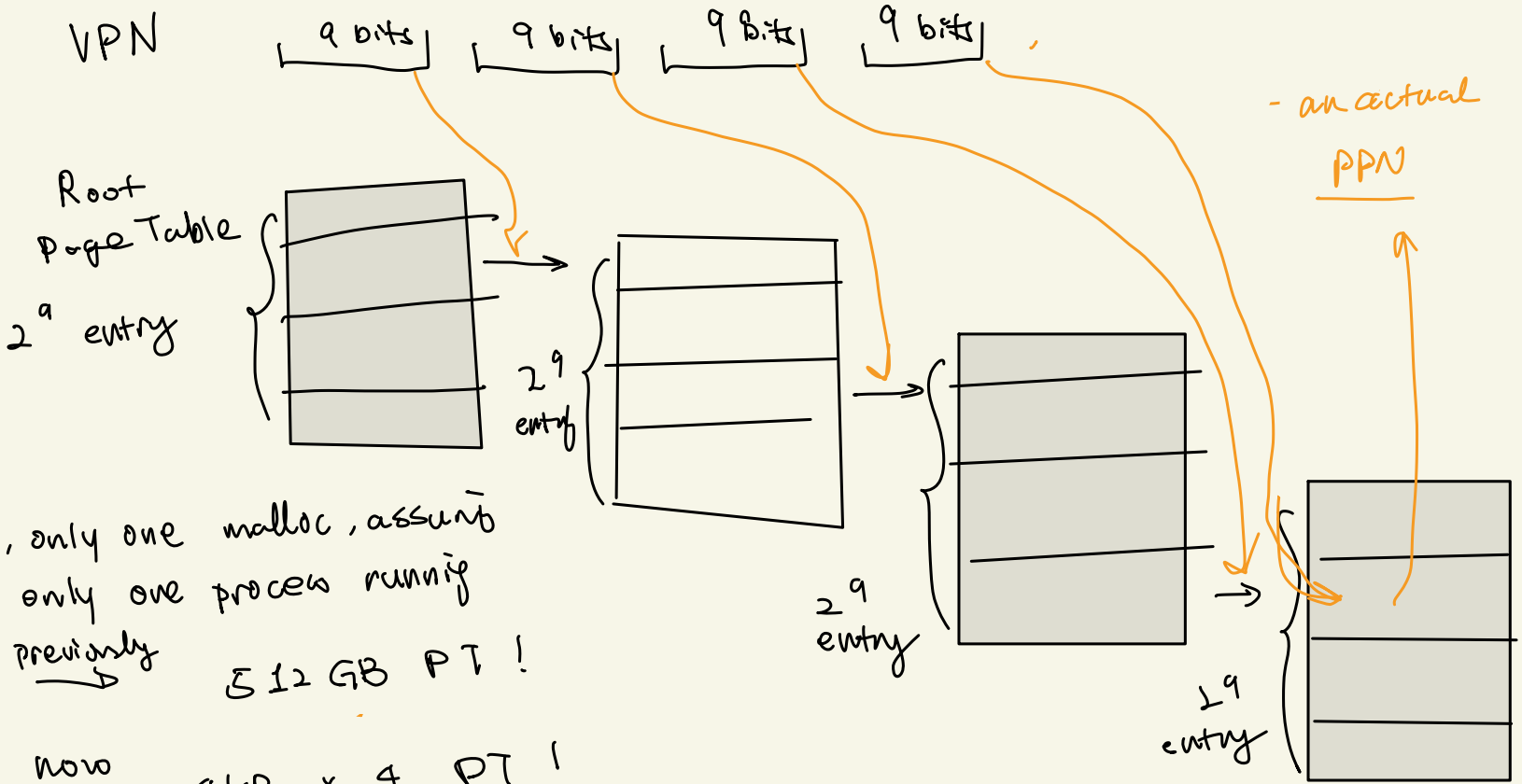
(and this is per-process)


32 bit VPN => 2^{32} bit of info
=> 4GB memory per process.
we rarely make full use of the 4GB space!
Physical pages.
Physical memory
How do we track these mapping?

Sparsity => Multi-level page table



Sparsity => Multi-level page table



If, only one malloc, assume only one process running previously

→ 512 GB PT !

now → 4kB x 4 PT !