

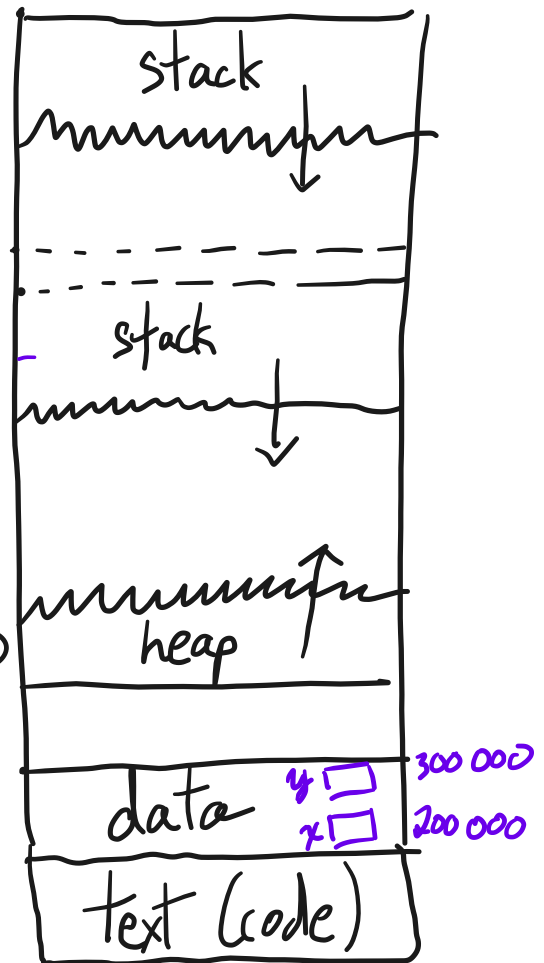
- 1. Last time
- 2. Intro to virtual memory
- 3. Paging
 - Intro
 - Key data structure: page table
 - Multilevel page table
 - Alternatives / Trade offs

2. Intro to virtual memory

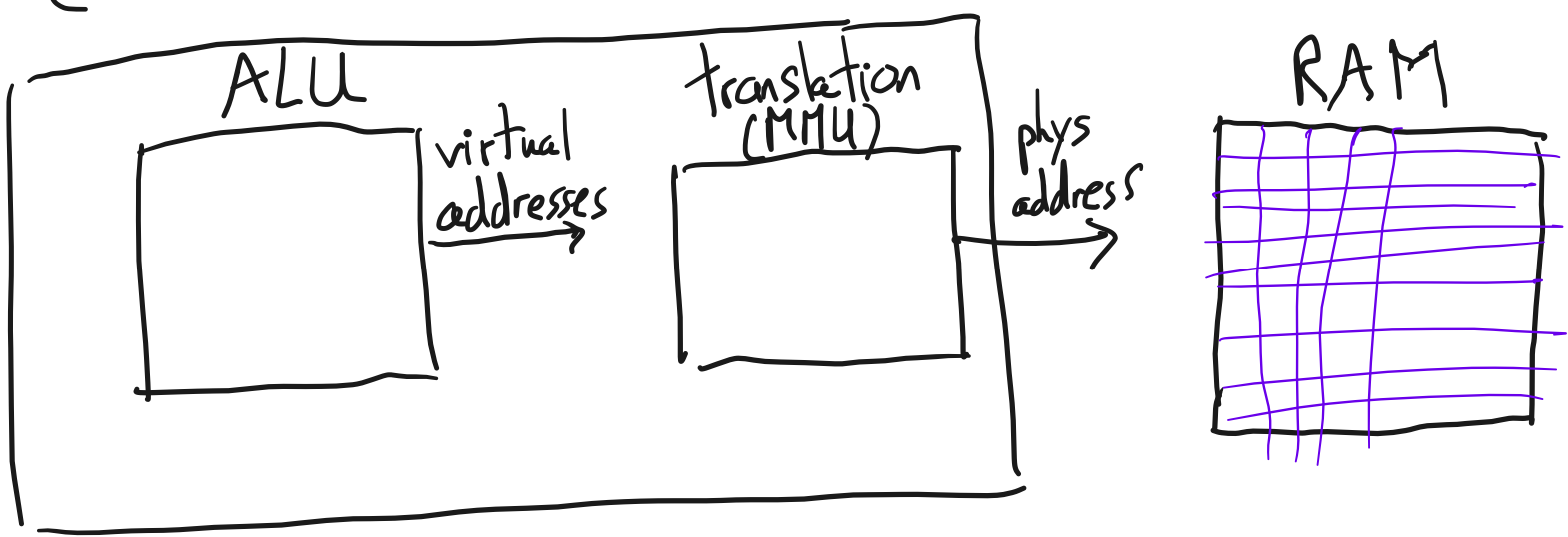
process "sees"

program excerpt: $y = x + 1$

<u>code address</u>	<u>instruction</u>
0x1500	movq 0x200000, %rax
0x1508	incq 1, %rax
0x1510	movq %rax, 0x300000



cpu



Benefits of virtual memory:

- Programmability

- (a) program thinks it has lots of memory
- (b) programs can use "easy" addresses: compiler and linker don't have to worry about where program lives in physical memory
- (c) multiple instances of a program can be loaded and not collide

1. tion

- protection
 - processes cannot read/write each other's memory
 - enables isolation (which is essential)

- effective use of resources

- sharing

- - - - -

How is translation implemented?

- hardware does it, in MMU

OS sets up data structures that the hardware "sees".

These data structures are per-process.

3. Paging

A. Intro

- Divide memory (virtual + physical) into fixed-size chunks

- These chunks are called PAGES

- PAGE SIZE

- x86-64: $4096 \text{ B} = 4 \text{ KB} = 2^{12} \text{ bytes}$

8 bits =
1 byte

Aside, in this class:

2^{10} : kilo, ~ 1000

2^{20} : mega, $\sim 1 \text{ million}$

2^{30} : giga, $\sim 1 \text{ billion}$

2^{40} : tera, $\sim 1 \text{ trillion}$

2^{50} : peta, $\sim 1 \text{ quadrillion}$

(though, increasingly, there are other terms: kibi byte, mebi byte, etc.)

How many pages are there in a 32-bit (byte-addressed) architecture?



$$2^{32} \text{ bytes} \times \frac{1 \text{ page}}{2^{12} \text{ bytes}} = 2^{20} \text{ pages}$$

What if 48 bits are used to address memory?

$$2^{48} \text{ bytes} \times \frac{1 \text{ page}}{2^{12} \text{ bytes}} = 2^{36} \text{ pages}$$
$$= 2^{30} \times \underbrace{2^6}_{64}$$

Page 0: [0, 4095] } VPNS PPN

Page 1: [4096, 8191]

⋮

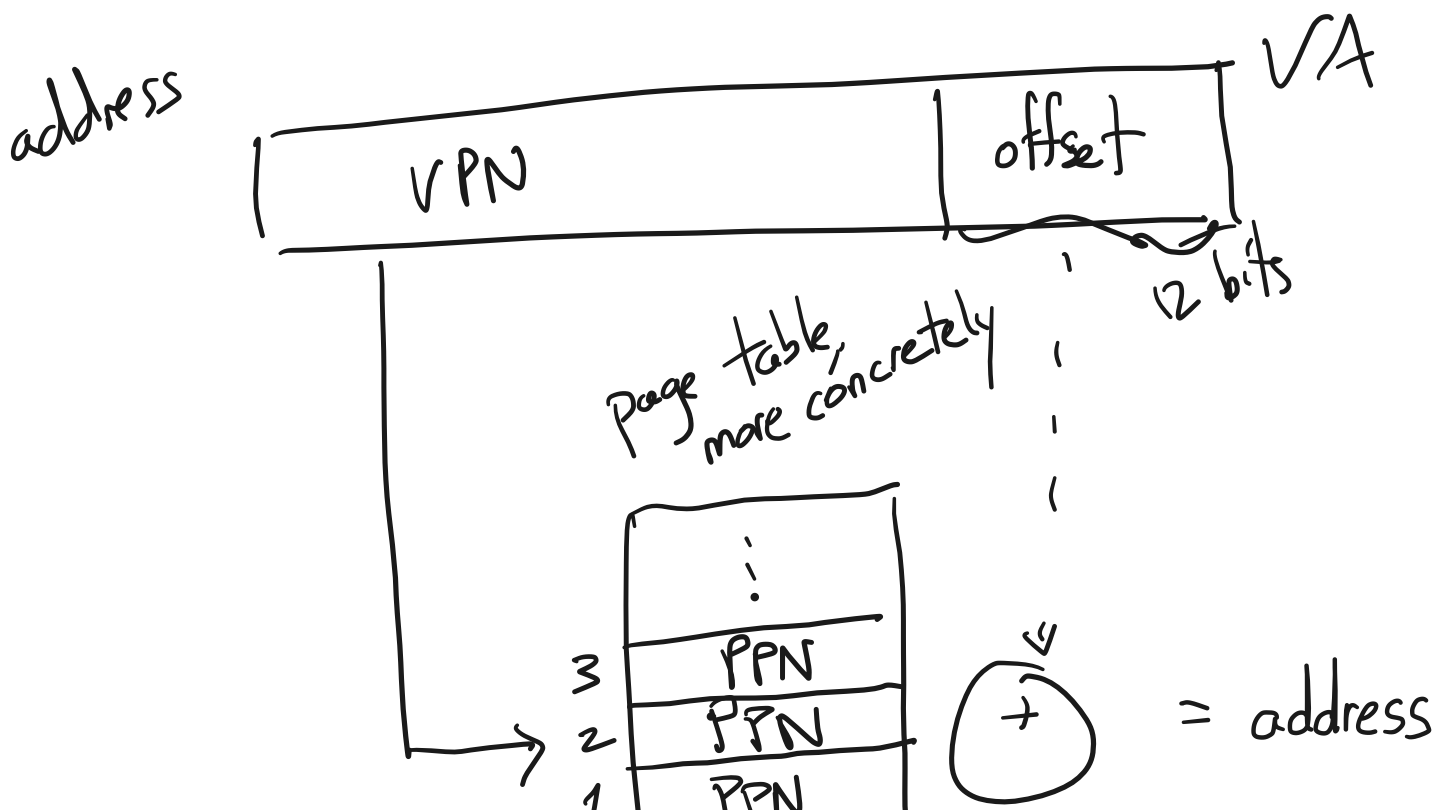
Page $2^{20}-1$:

[... .., $2^{32}-1$]

B. Key data structure: page table (per-process)

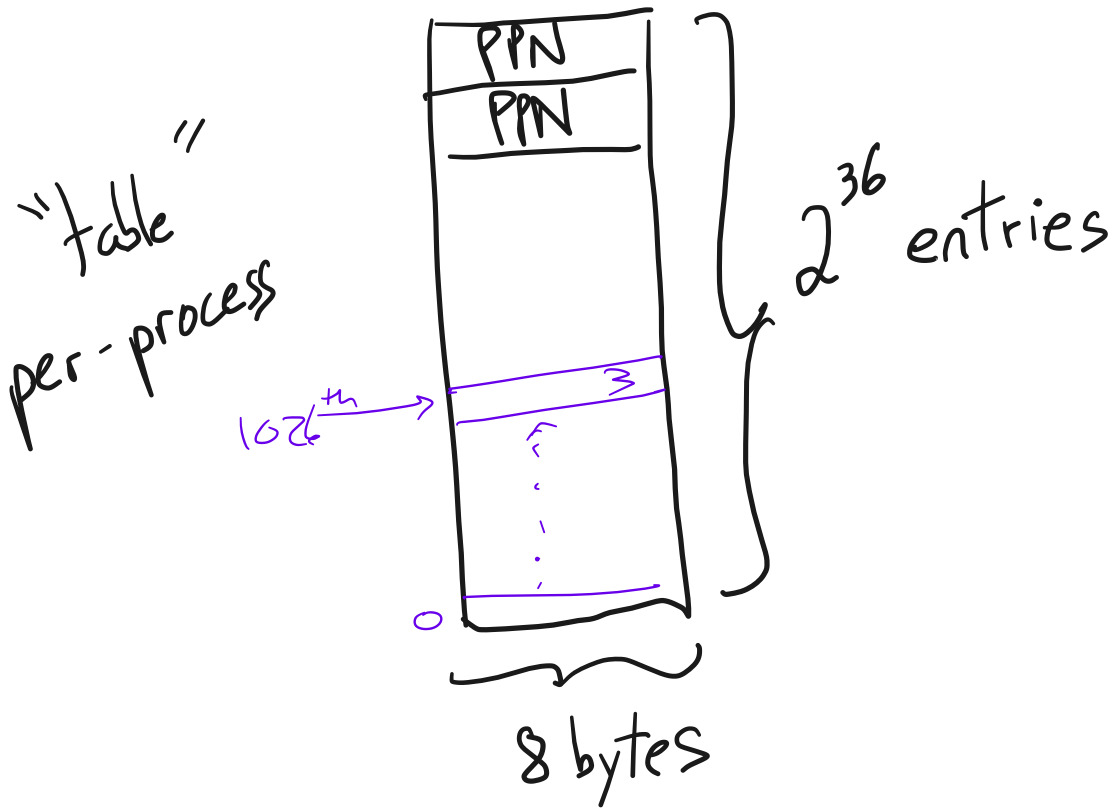
conceptually: a map from

VPN \rightarrow PPN





assume: 48-bit addresses, and 4KB pages (2^{12} bytes)



Ex: OS wants: a process to use address

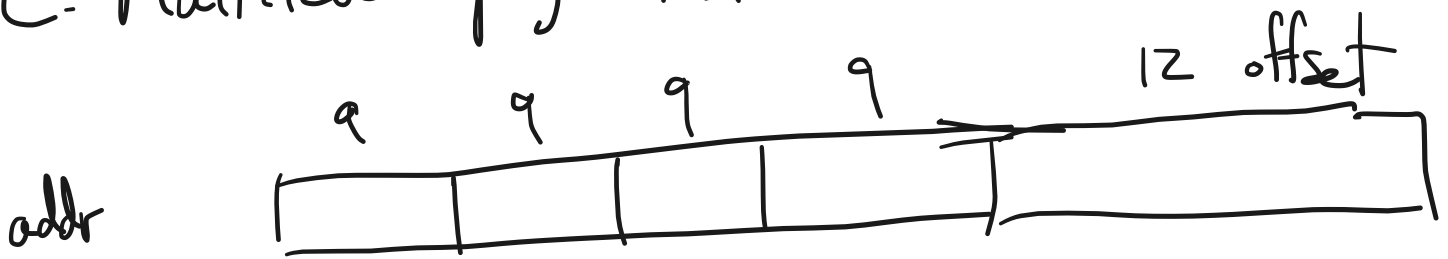
VA: $0x00402000$ to refer to

PA: $0x00003000$

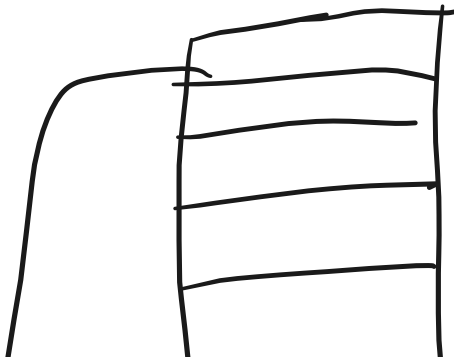
table[0x00402] = 0x00003

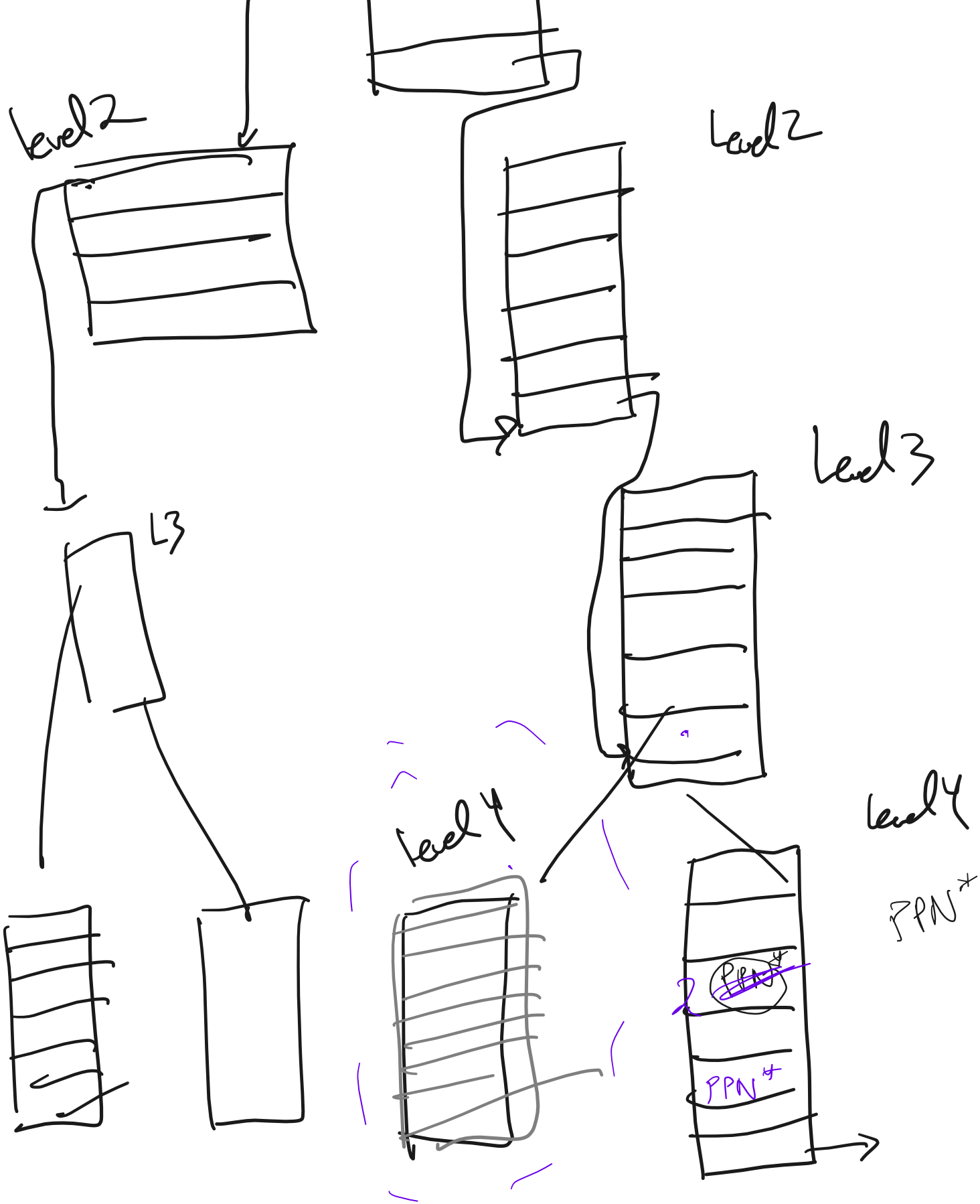
What's the issue?

C. Multilevel page tables



root level 1





Ex: we want the physical address range
 $[4\text{MB}, 6\text{MB} - 1]$ to appear at virtual address
 $[48, 17]$

range $[2^1 - 2MB, 2^2 - 1]$