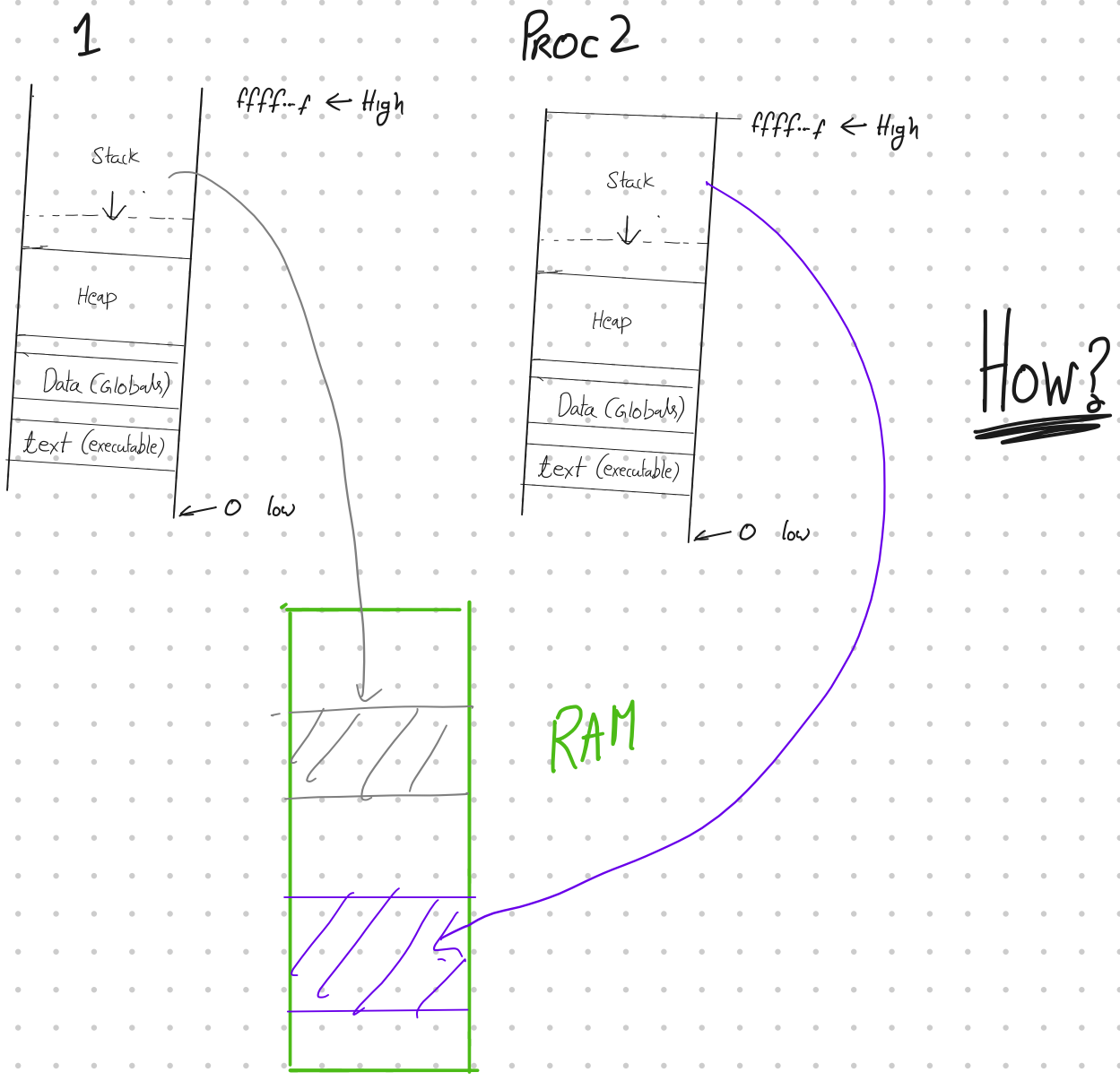
CS 202 (-002) Lecture 11: Virtual Memory I

Announcements

- Tuesday: Legislative Monday
  - ↳ No class
- Lab 3 due Friday

Virtual Memory

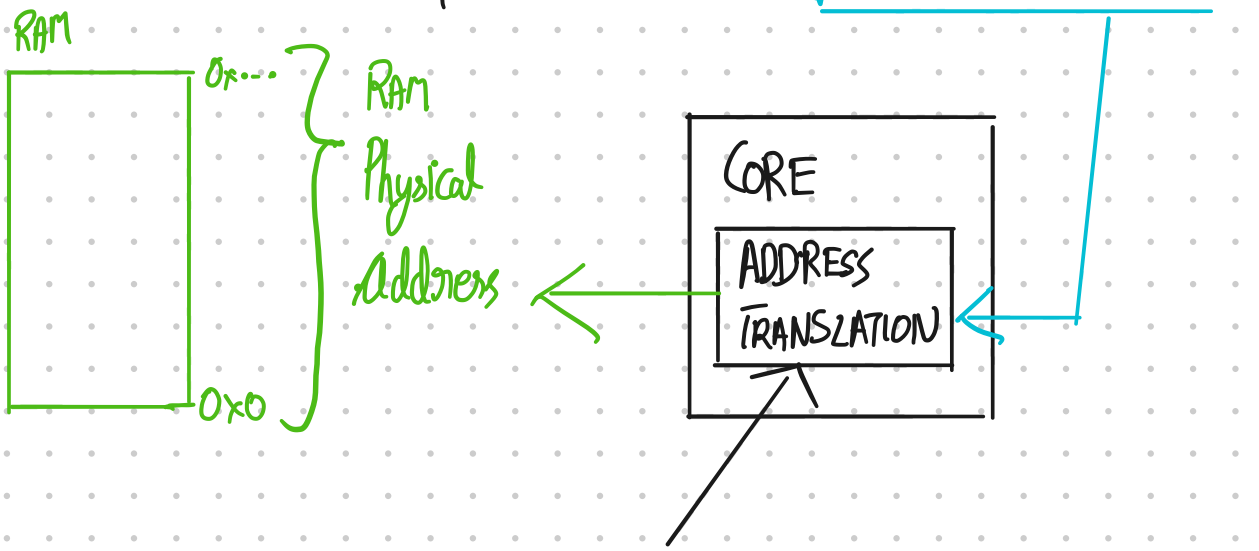
Lec 2/3: Process



Core Idea

0x2001f0 movq (0xffffff20), %RAX [%RIP = 0x2001f0+...]

↳ Nearly all addresses used by the processor are VIRTUAL ADDRESSES



Operating system Kernel controls VA to PA mapping

Control over mapping ENABLES

(1) Isolation: One process cannot affect another's memory*

(2) Transparency/programmability/...:

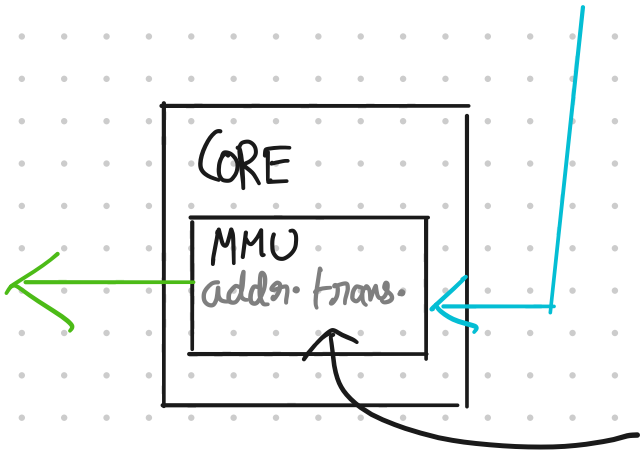
- Program does not need to worry about where/when it is loaded

Proc1 & Proc2 can both use address 0x20000

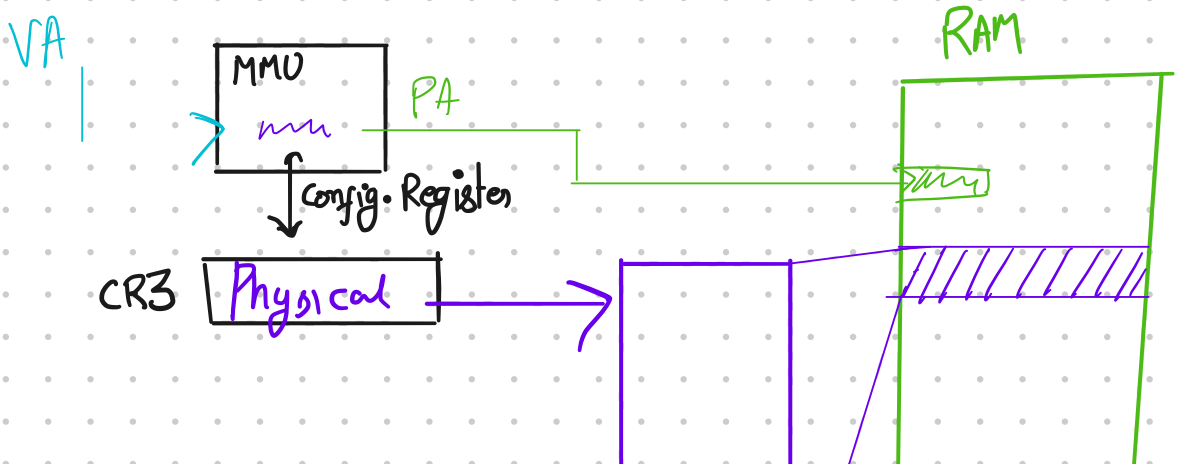
- Program can believe there is more memory than reality (Swapping)

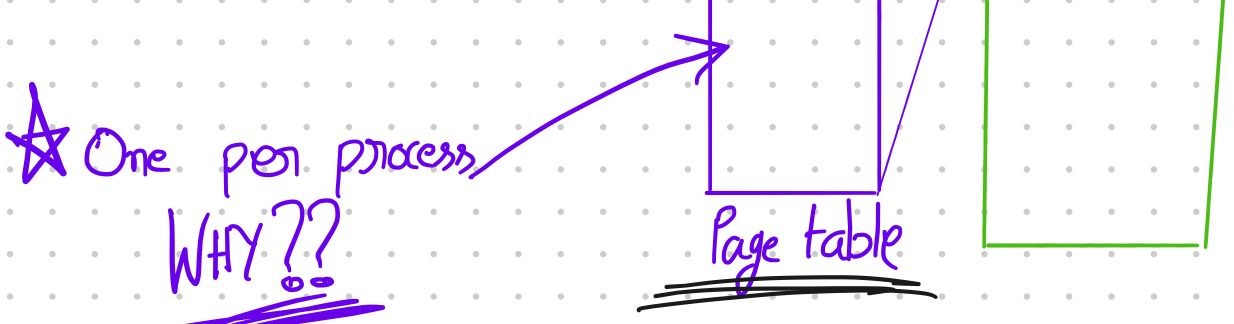
③ Resource efficiency

④ Sharing b/w processes ← Return to later.



Paging





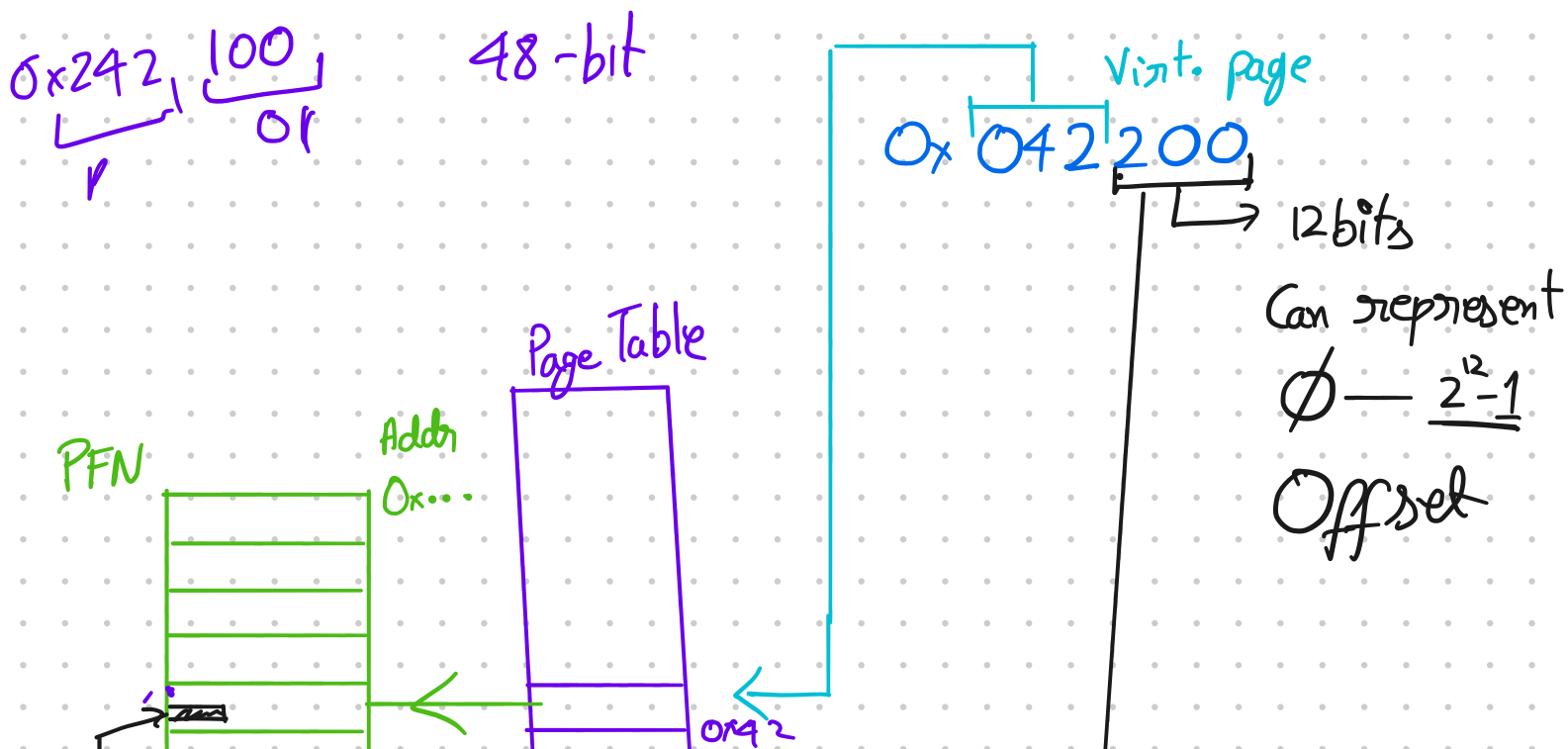
Goals: Efficiency: Limit size of pagetable
Limit translation complexity

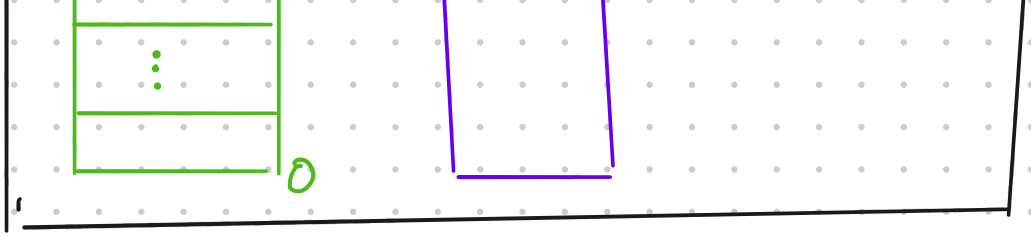
How? Partition physical memory into fixed size pages.

↳ Often 4KB ≡ 4096 bytes ≡ 2^{12} bytes

Pagetable maps

Virtual page to physical page





Q1. Number of mappings assuming 32-bit addresses?

- Addresses $0 - 2^{32}$

$\Rightarrow$ $2^{32}/2^{12}$ Pages $= 2^{20}$ pages

$= 1048576$

Assuming 48-bit

$2^{36} = \sim 68$ billion

Observations

① Array of 1 million or 68 billion items is large

② Most processes do not need 4 GB (2^{32} bytes) of memory.

How to make page tables more efficient

↳ Use a tree!