

P4 Discussion Section

TA: Hayden Coffey

Updated: 03/04/25

P4: Learning Objectives

Understand paging in xv6.

Understand complexities of supporting multiple page sizes.

P4: Project Overview

Integrating huge pages (4 MB) into the xv6 memory management system.

—

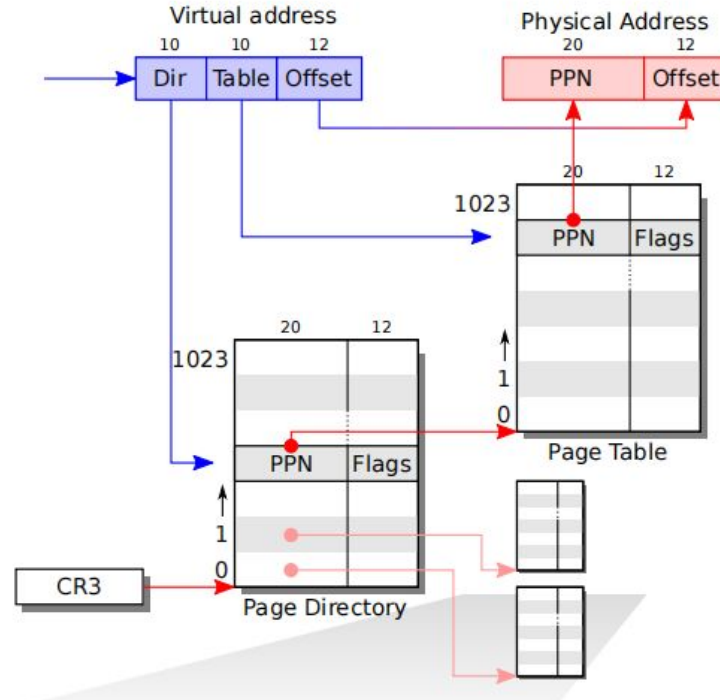
Can work with a partner (can be in other lecture section).

Due 03/20/25.

XV6 Memory Management Overview

xv6 Page Table

4 KB Pages



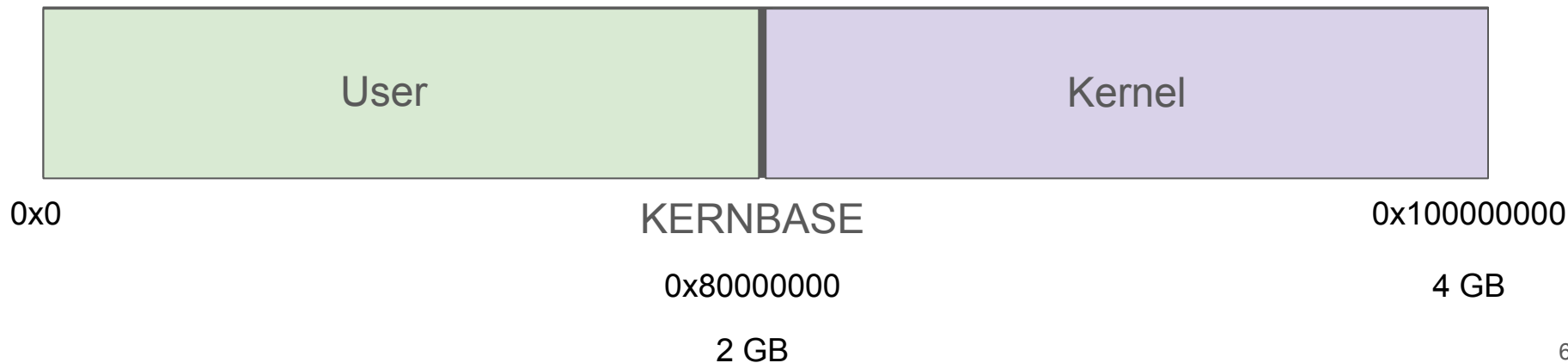
xv6 Memory Basics

4 KB page size

32 bit **virtual** address space

Virtual memory is split between kernel and user space program

Virtual Address Space



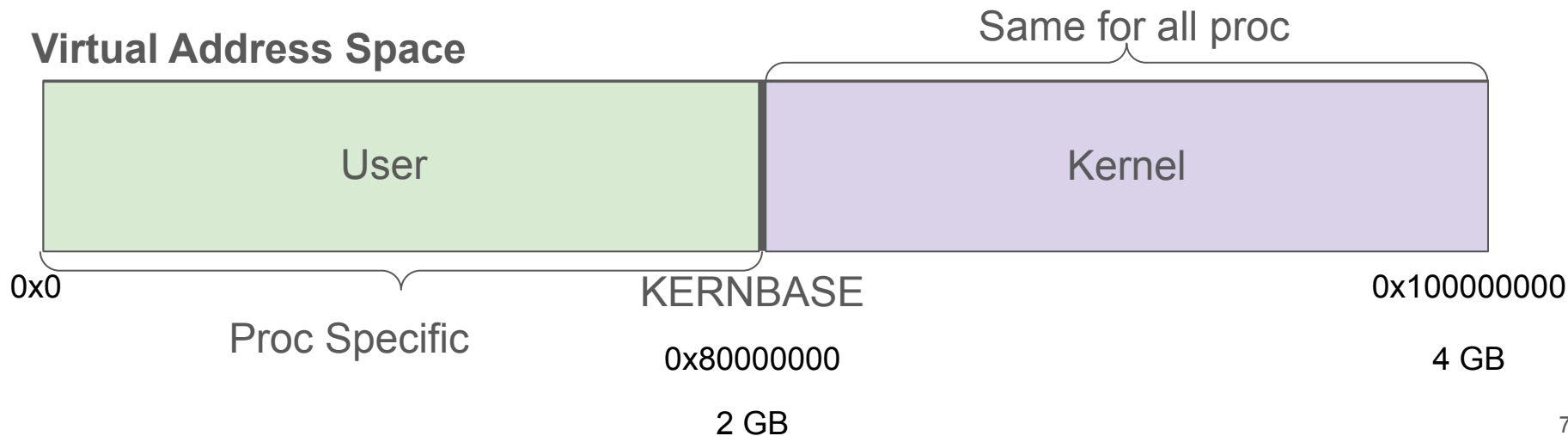
xv6 Memory Basics

4 KB page size

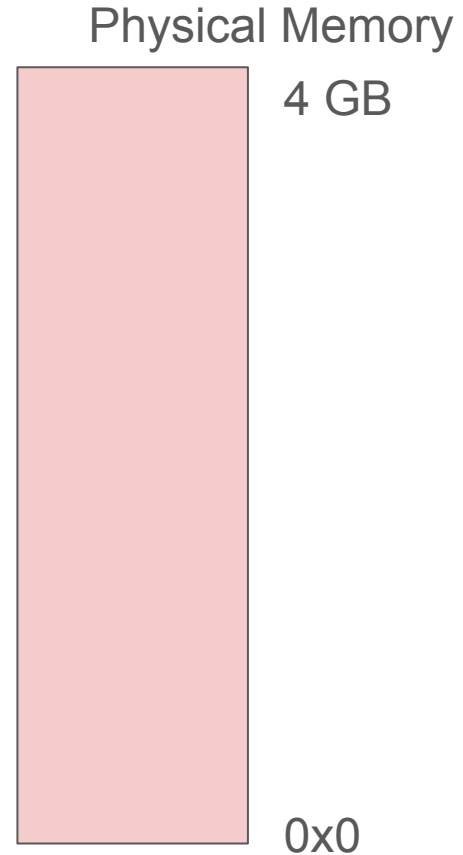
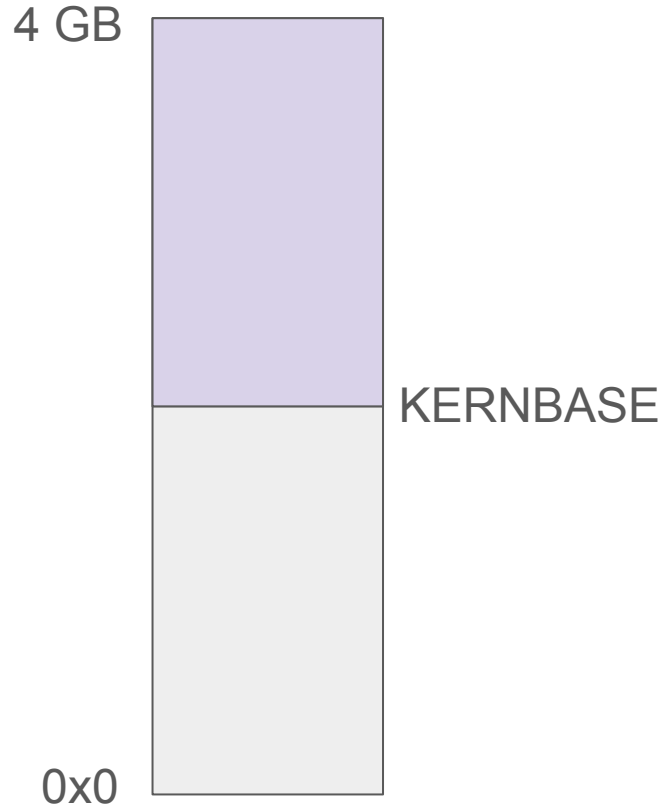
32 bit **virtual** address space

Virtual memory is split between kernel and user space program

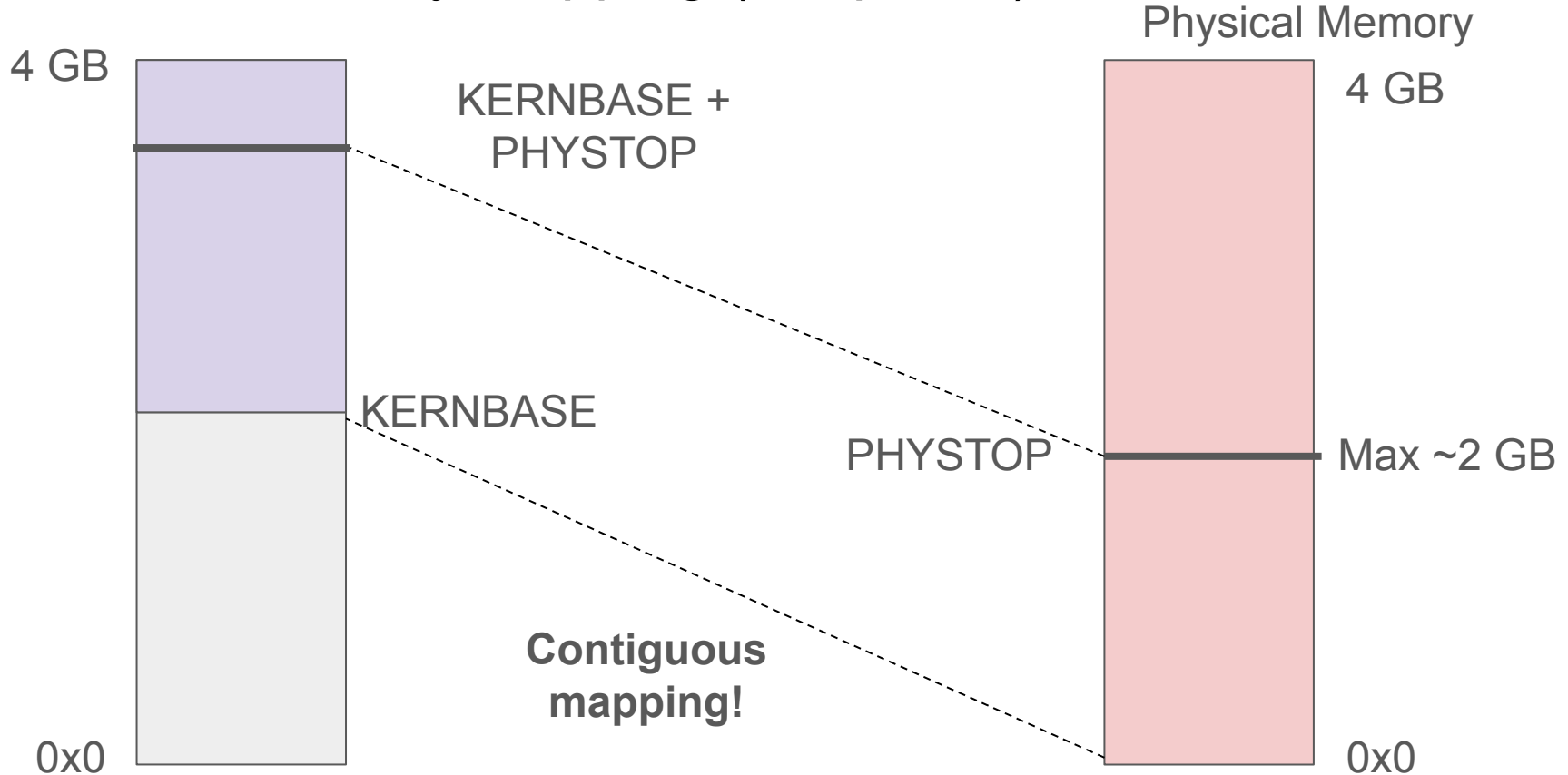
Virtual Address Space



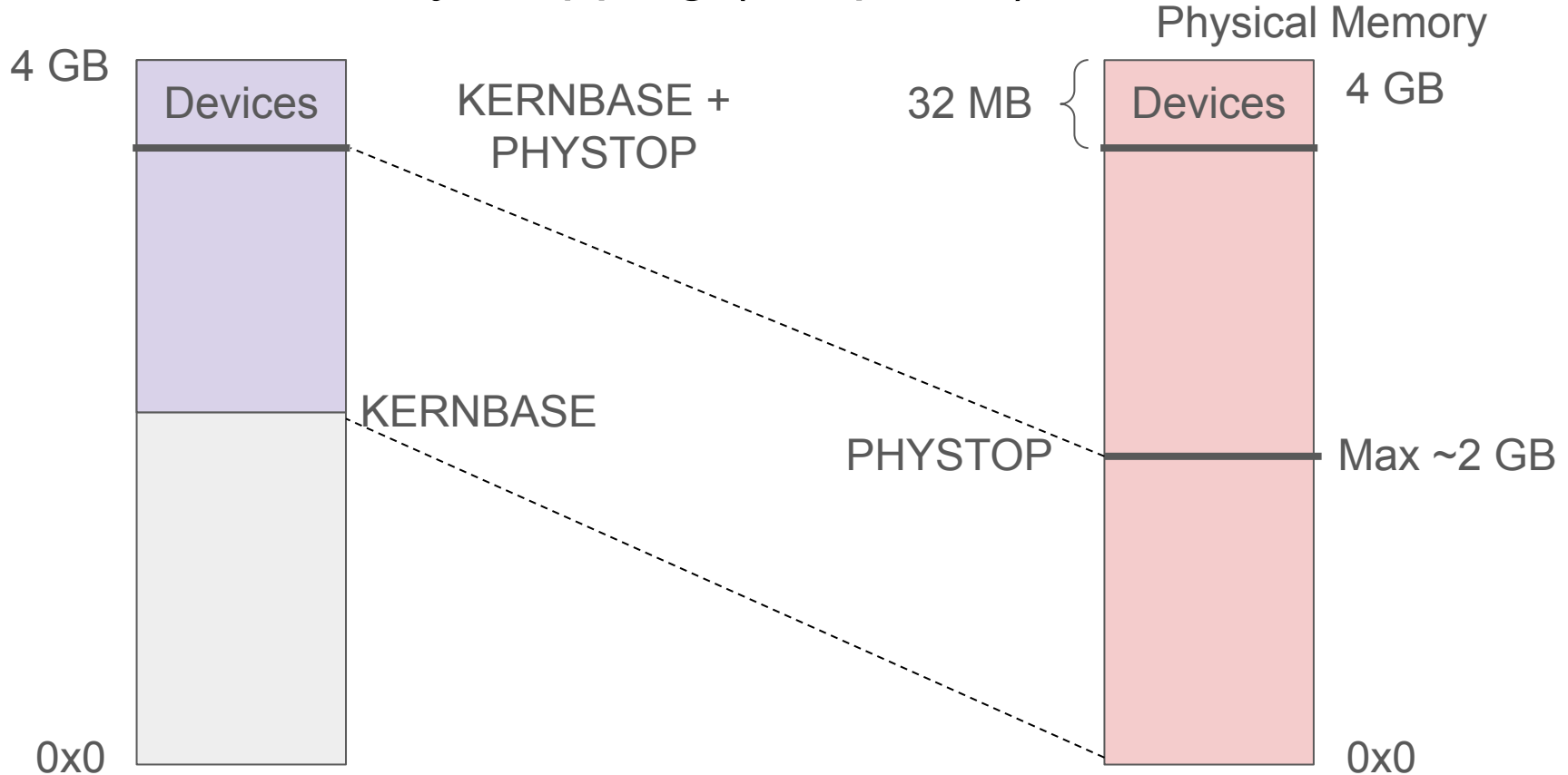
Kernel Memory Mapping (Simplified)



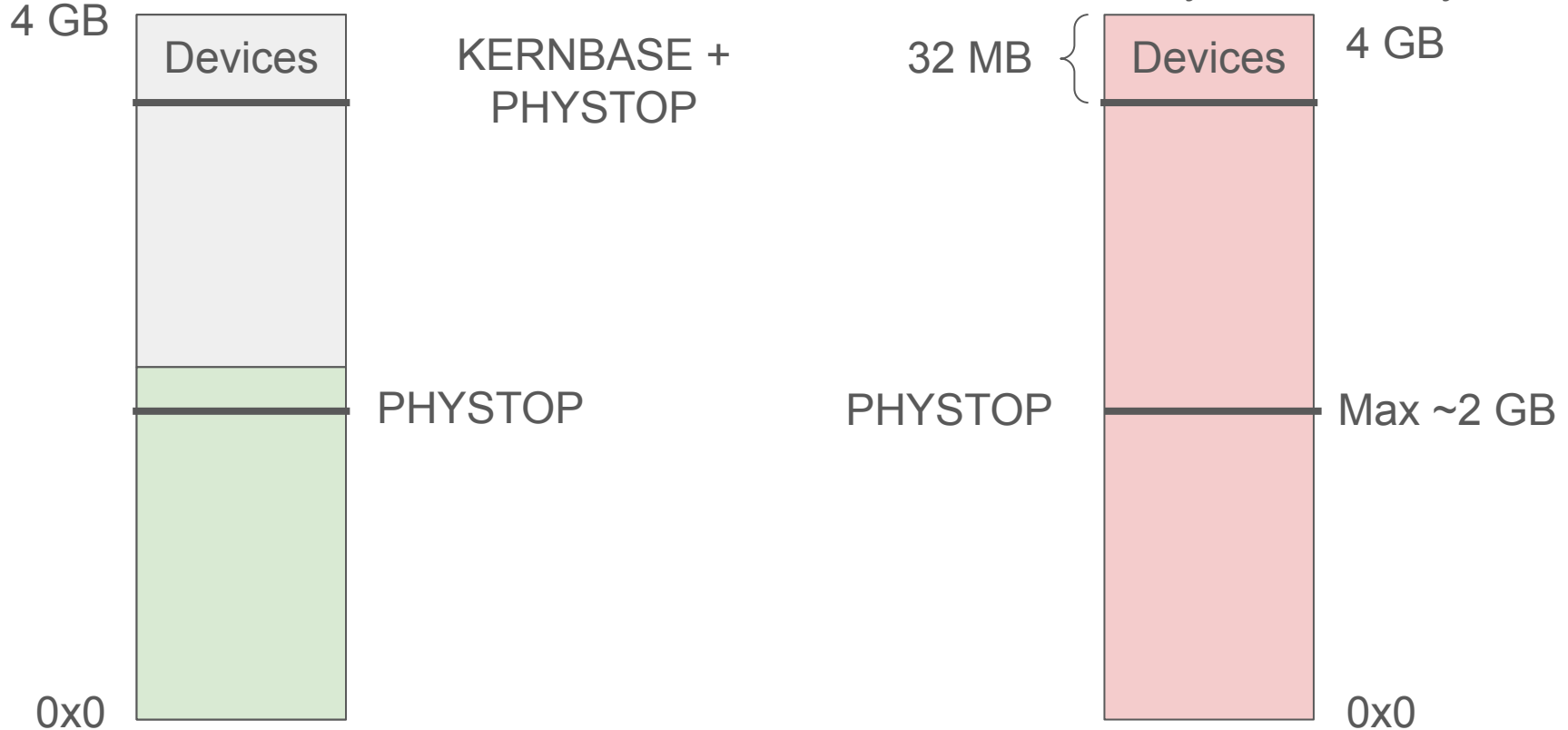
Kernel Memory Mapping (Simplified)



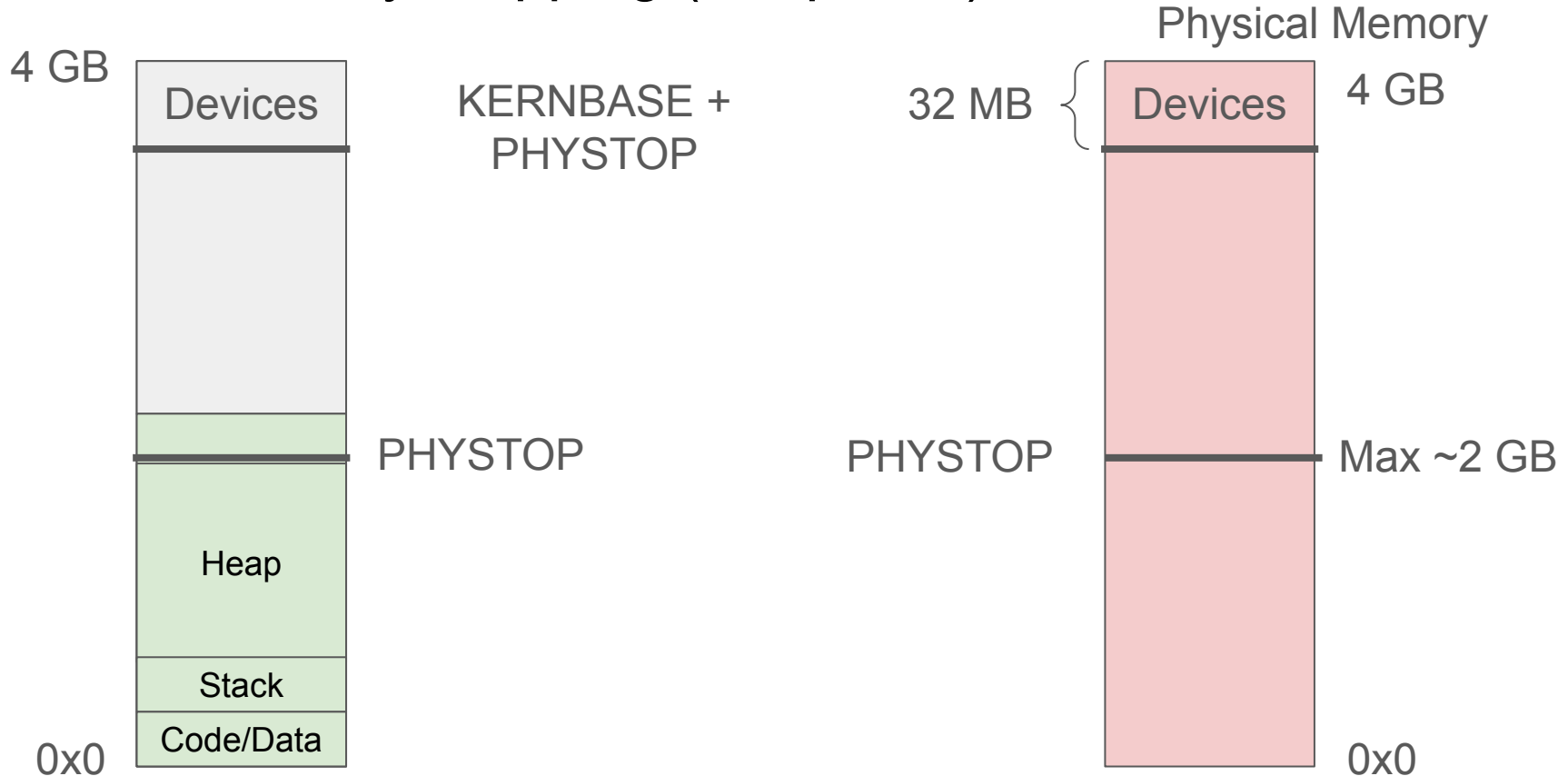
Kernel Memory Mapping (Simplified)



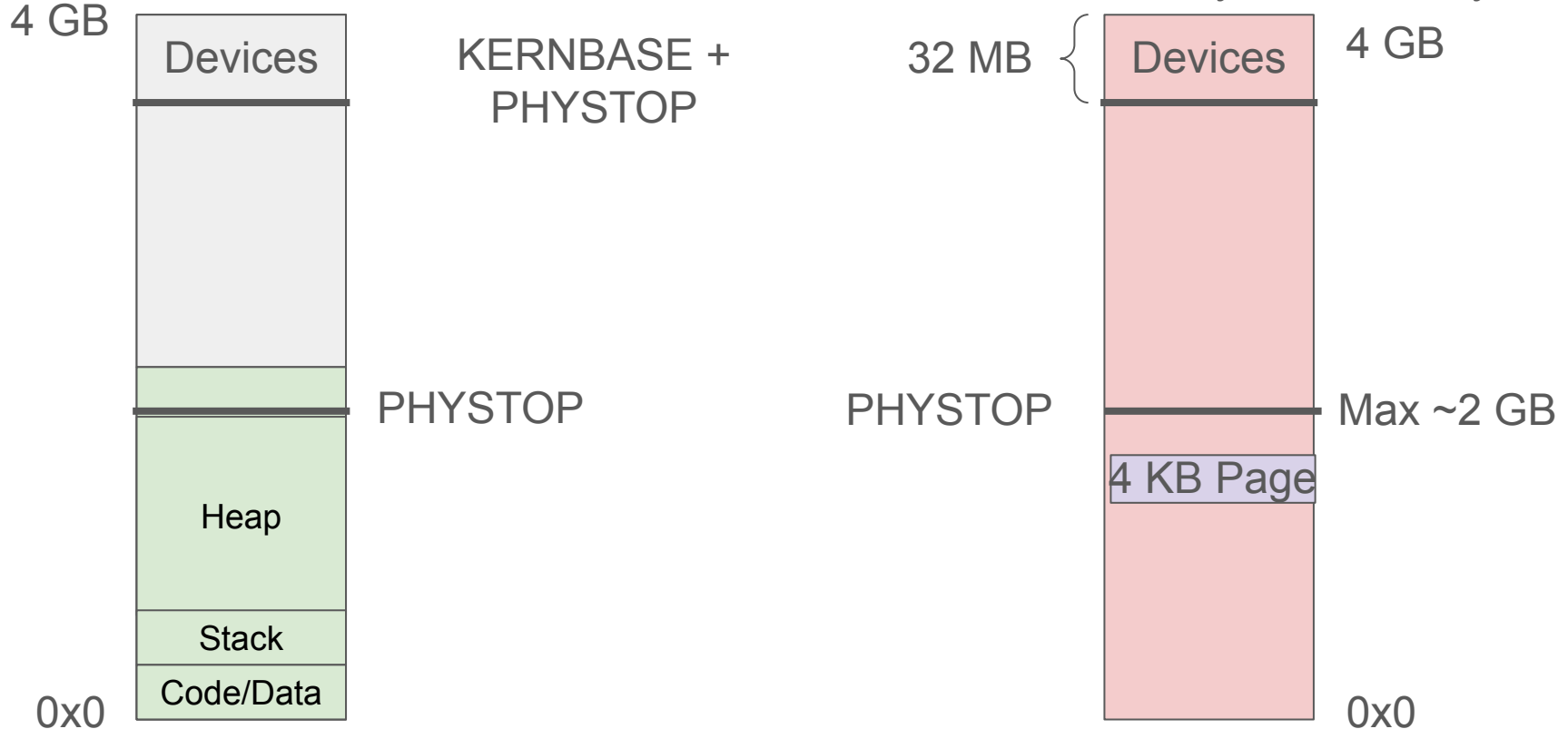
User Memory Mapping (Simplified)



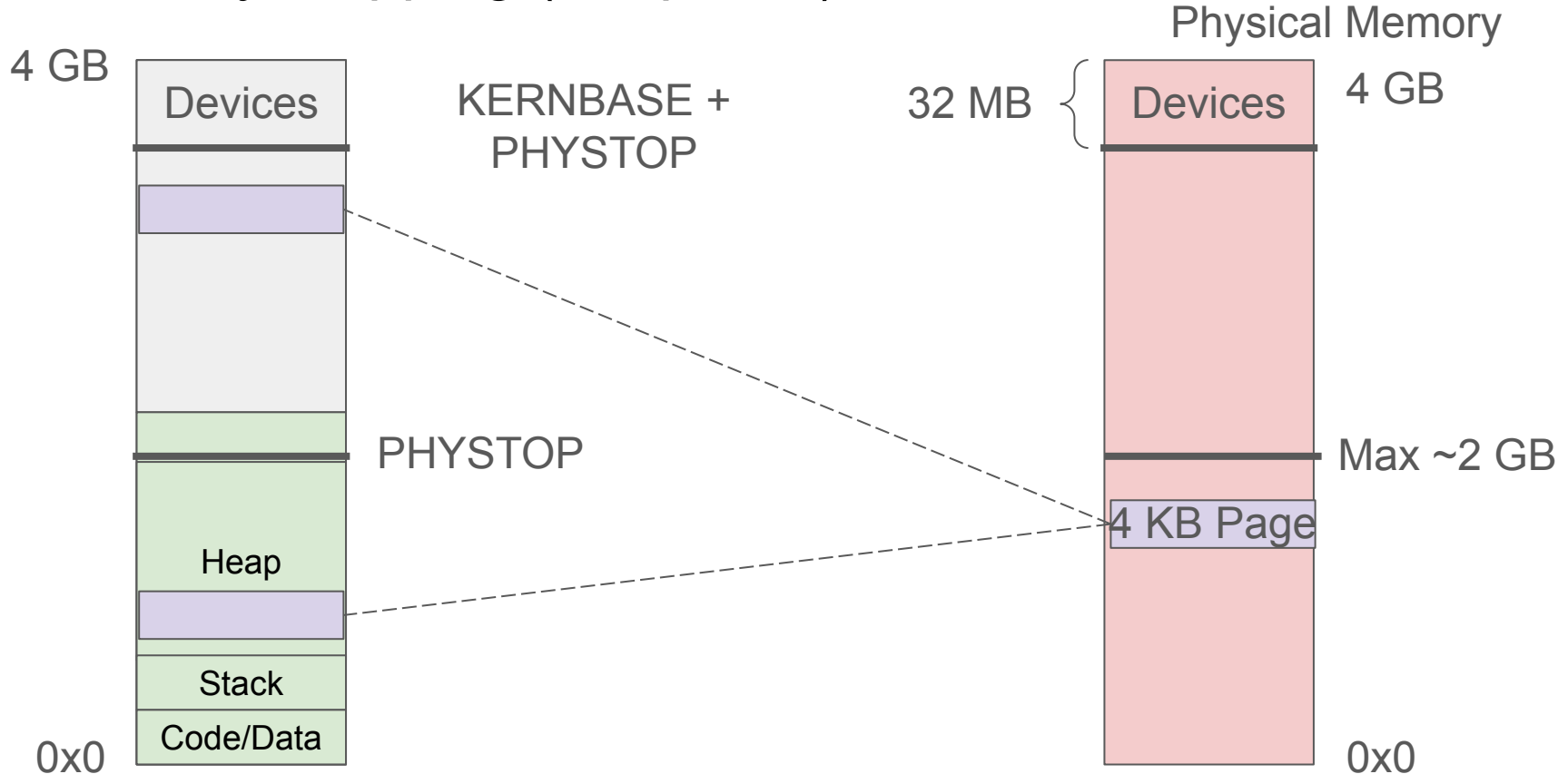
User Memory Mapping (Simplified)



Memory Mapping (Simplified)



Memory Mapping (Simplified)

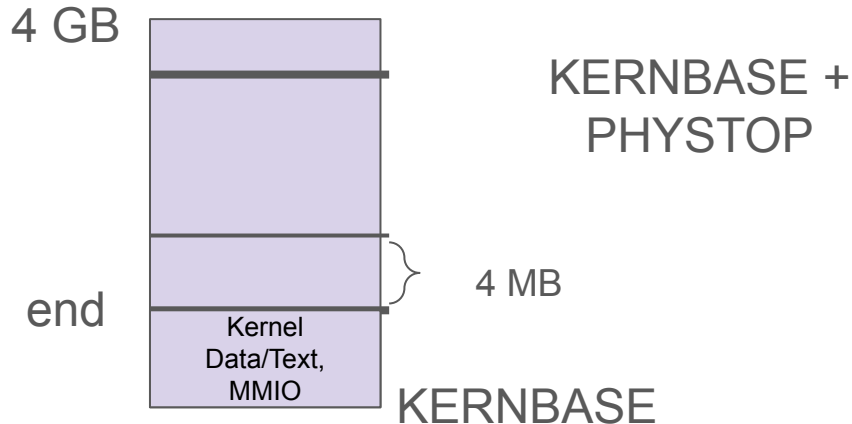


Memory Init (Look in main.c)

Kernel needs to create its own page table!

After boot, kernel starts with only first 4 MB of memory available.

kinit1(): Chops up 4 MB into 4 KB free pages.

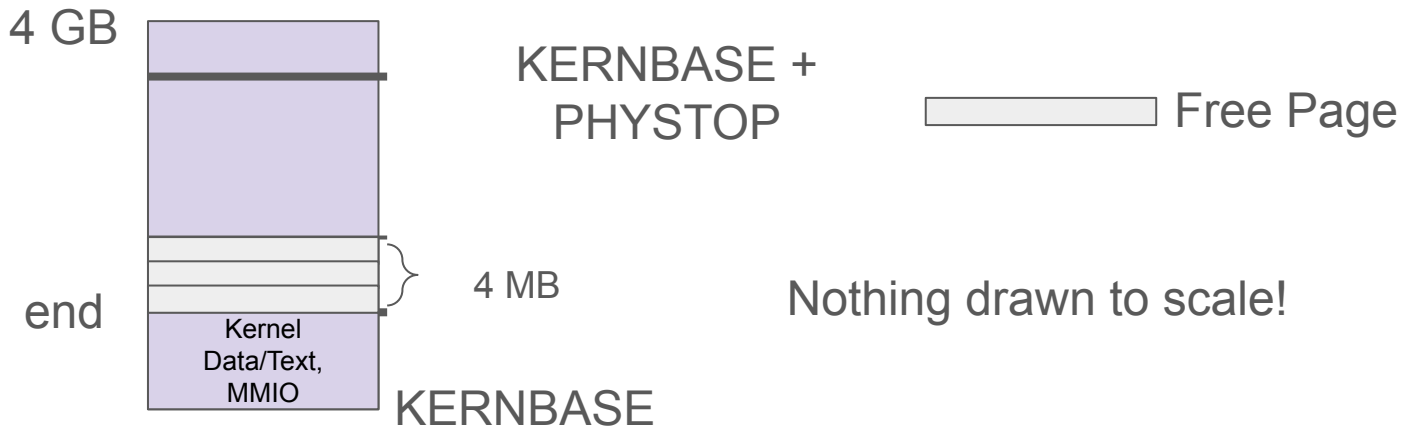


Memory Init (Look in main.c)

Kernel needs to create its own page table!

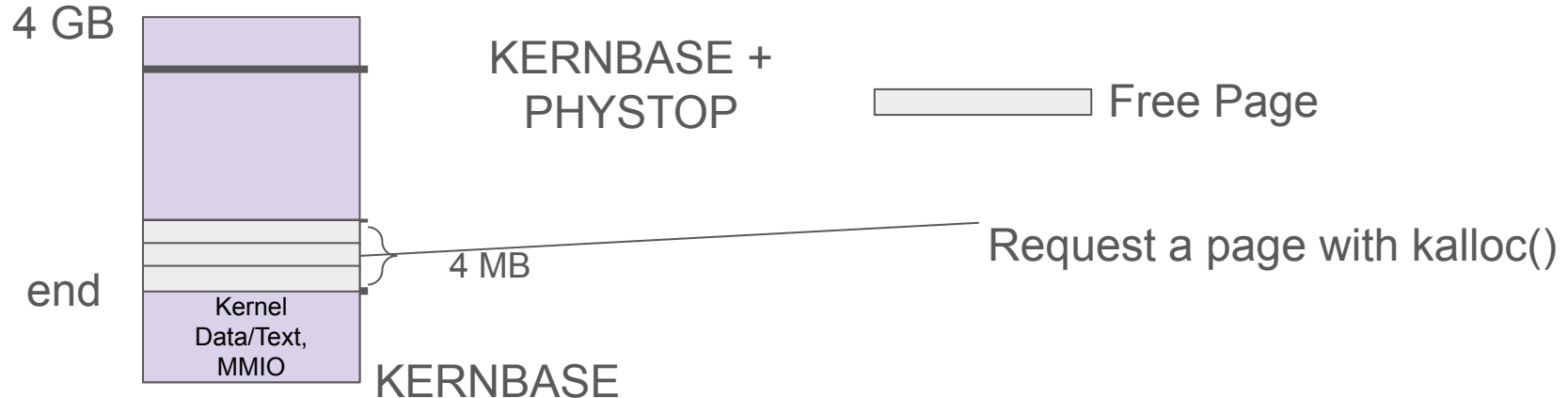
After boot, kernel starts with only first 4 MB of memory available.

kinit1(): Chops up 4 MB into 4 KB free pages with freerange().



Memory Init (Look in main.c)

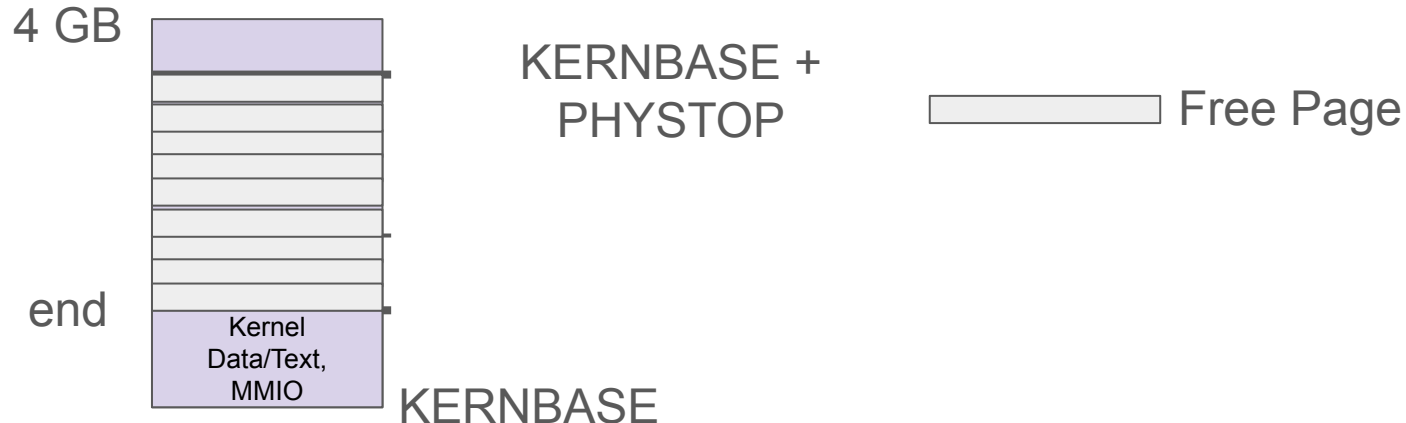
kvmalloc(): Use starting free pages to build kernel part of page table.



Memory Init (Look in main.c)

kvmalloc(): Use starting free pages to build kernel part of page table.

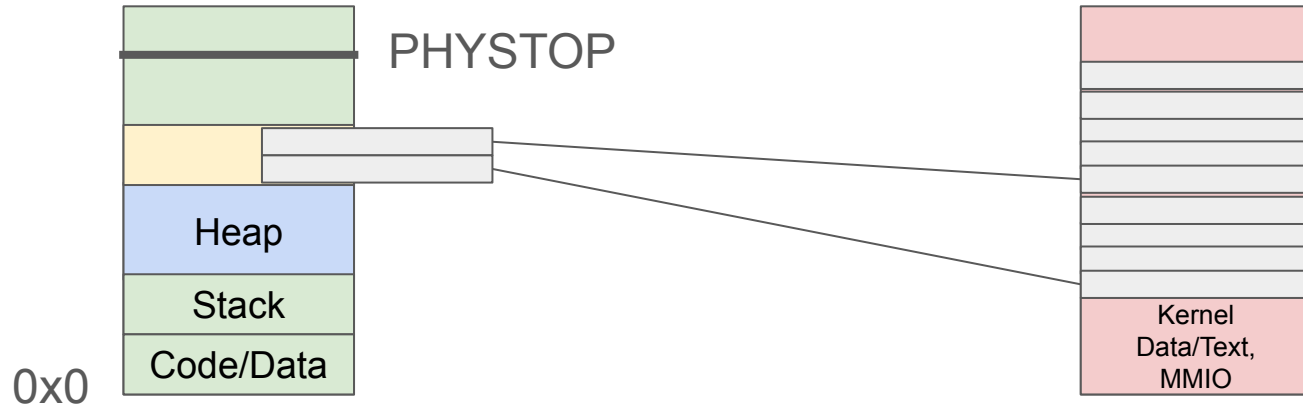
kinit2(): Adds rest of memory to free page list with freerange().



How does a process get memory?

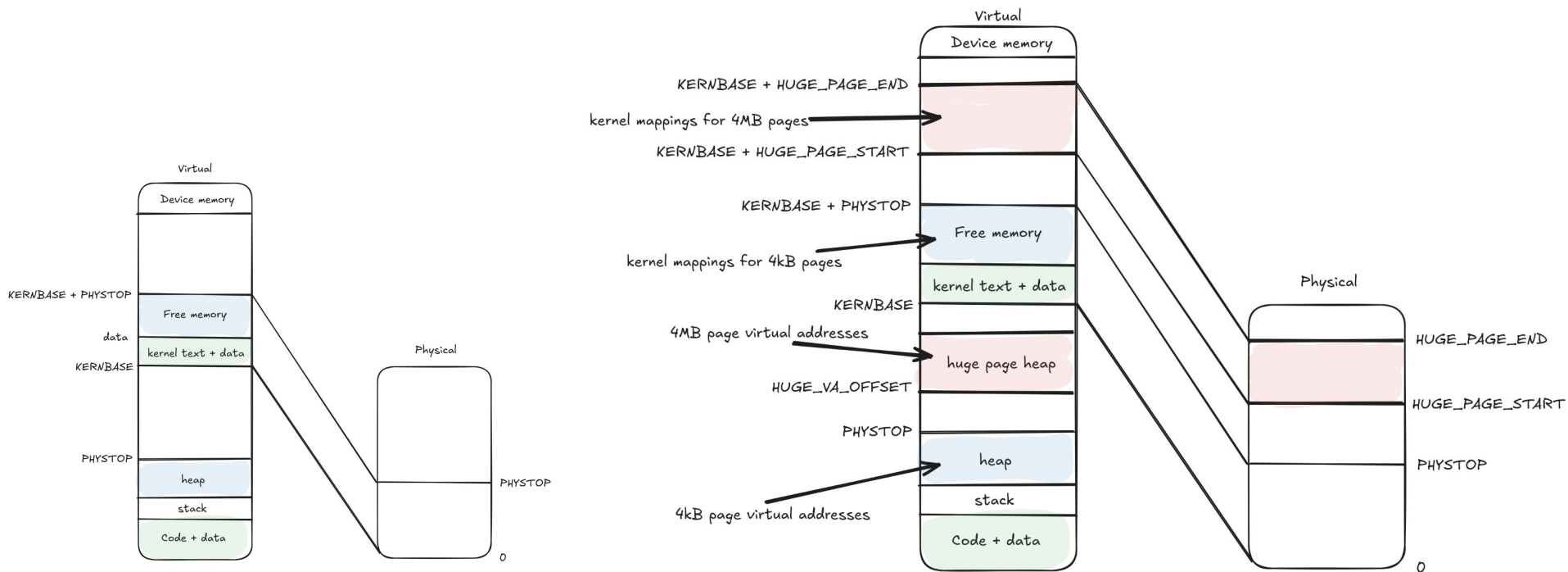
malloc(): If more pages needed, calls sbrk()

sbrk() -> growproc() -> allocvm



Project 4:

Introducing huge pages (4 MB) into xv6



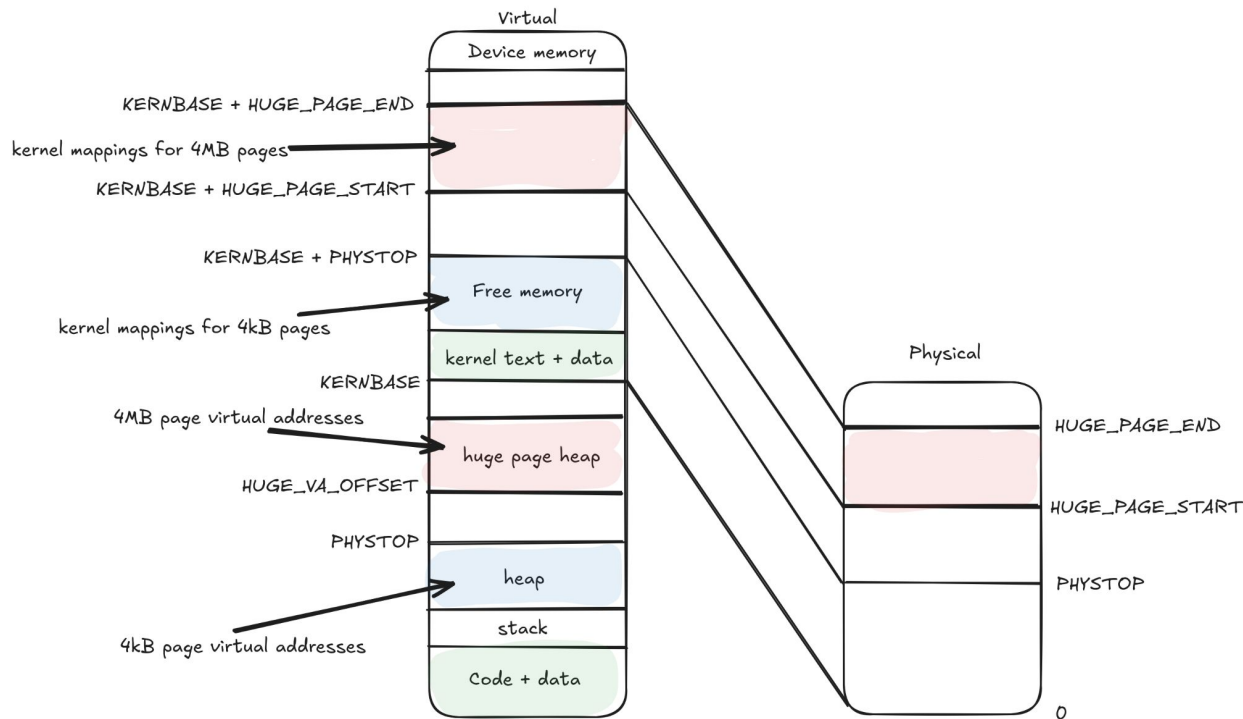
Project 4:

Step 1: Enable huge page allocations

khugeinit()

khugealloc()

khugefree()



Project 4:

Step 2: Integrate into memory management system.

`vmalloc(n, flag)`: Can base on `malloc` (`umalloc.c`)

`flag` -> `VMALLOC_SIZE_BASE` or `VMALLOC_SIZE_HUGE`

Project 4:

Step 3: Transparent Huge Page Allocation

Update malloc to use huge pages for allocations ≥ 1 MB.

Sources

<https://github.com/zarif98sjs/xv6-memory-management-walkthrough>

<https://www.cse.iitb.ac.in/~mythili/os/notes/old-xv6/xv6-memory.pdf>