Operating Systems!

Memory: Approaches to Memory Management (Part 2)

Prof. Travis Peters

Montana State University

CS 460 - Operating Systems

Fall 2020

https://www.cs.montana.edu/cs460

Some diagrams and notes used in this slide deck have been adapted from Sean Smith's OS courses @ Dartmouth. Thanks, Sean!

Today

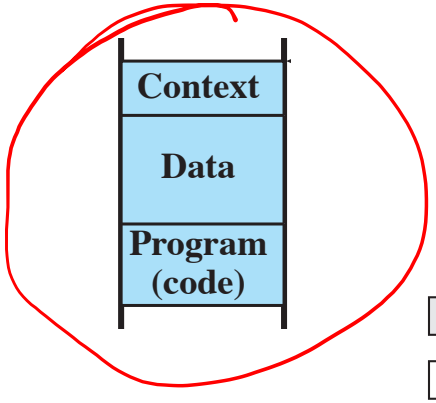
- Announcements
  - ~~Yalnix~~ Project Updates
  - *Exam 1:* ***Nice work! Working on grading...***

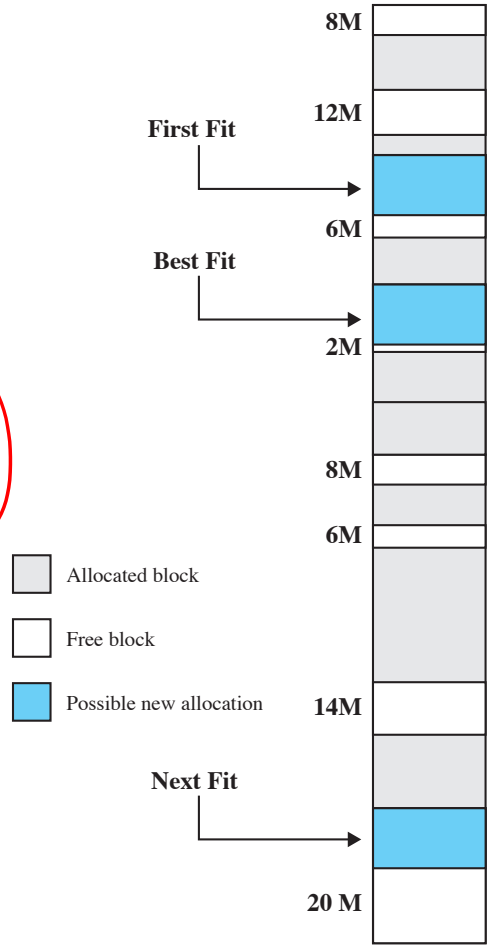
- Upcoming Deadlines
  - **Sunday [10/18/2020] @ 11:59 PM (MST)** - PA3 (Group Assignment)
  - **Sunday [10/25/2020] @ 11:59 PM (MST)** - Project Proposal

Today *(cont.)*

- Agenda
  - ~~Revisiting Memory Basics~~
  - ~~Logical vs. Physical Addresses~~
  - ~~Contiguous Memory & "Fit" Strategies~~
  - Paging / Non-Contiguous Memory
  - Address Translation
  - Demos (**fault.c**)

  ➔ Virtual Memory



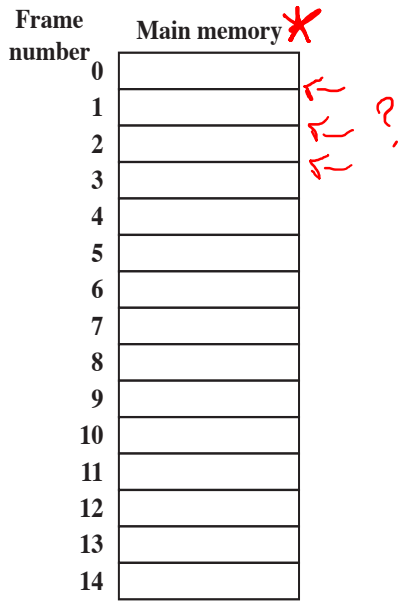


Paging

We don’t have to keep everything in a contiguous region!

Paging

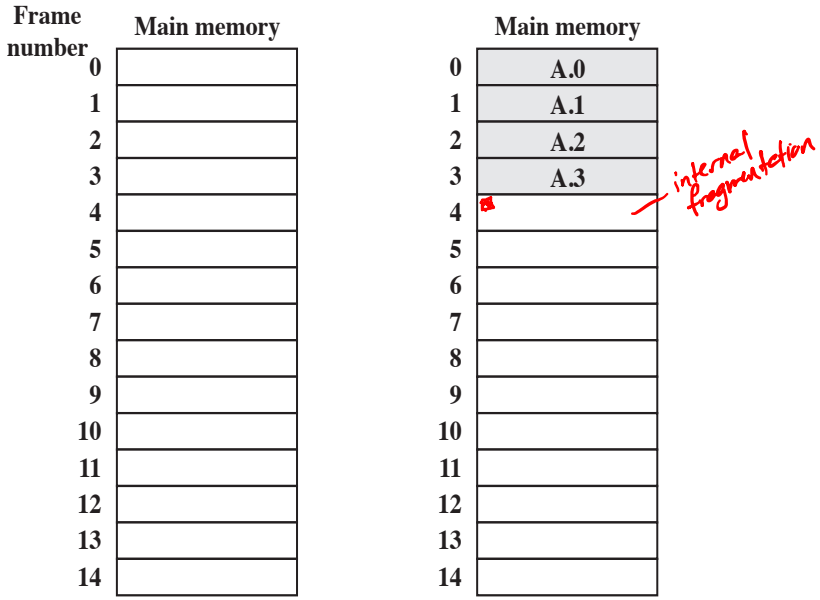
- Partition MM into **small, fixed-size chunks** of the same size
- Assign chunks of processes ("**pages**") into available chunks of MM ("**frames**")
- No more external fragmentation!
- Only minimal internal fragmentation!



(a) Fifteen Available Frames

Paging

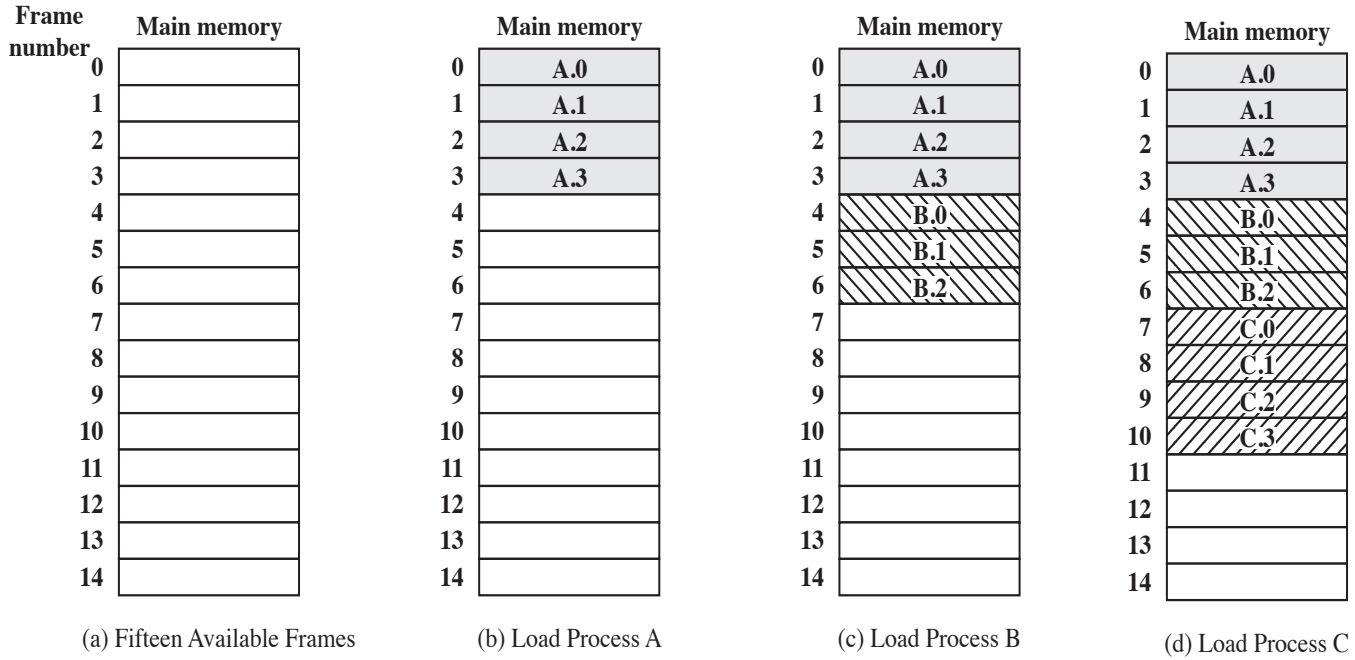
- Partition MM into **small, fixed-size chunks** of the same size
- Assign chunks of processes ("**pages**") into available chunks of MM ("**frames**")
- No more external fragmentation!
- Only minimal internal fragmentation!



(a) Fifteen Available Frames (b) Load Process A

Paging

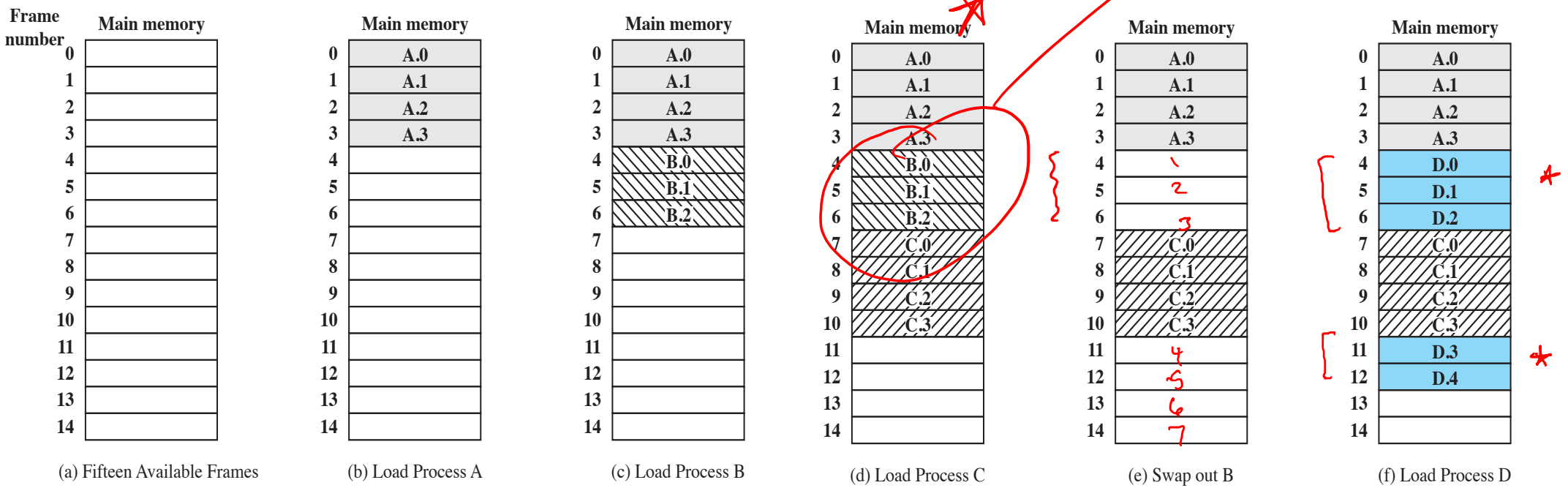
- Partition MM into **small, fixed-size chunks** of the same size
- Assign chunks of processes ("**pages**") into available chunks of MM ("**frames**")
- No more external fragmentation!
- Only minimal internal fragmentation!



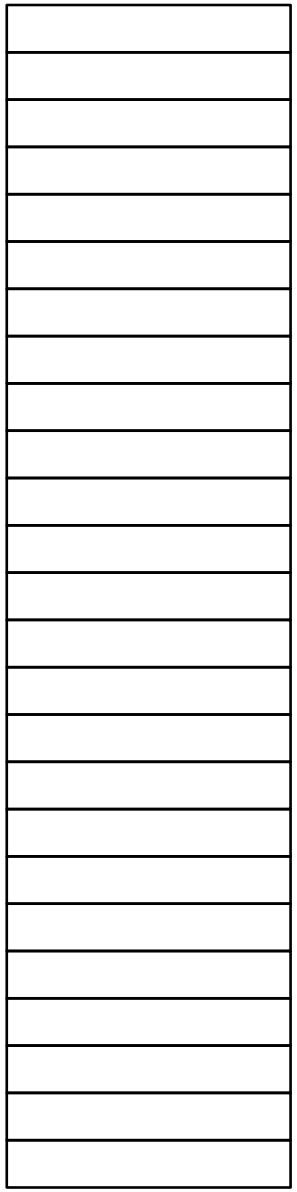
(a) Fifteen Available Frames (b) Load Process A (c) Load Process B (d) Load Process C

Paging

- Partition MM into **small, fixed-size chunks** of the same size
- Assign chunks of processes ("**pages**") into available chunks of MM ("**frames**")
- No more external fragmentation!
- Only minimal internal fragmentation!



high
("foxes")



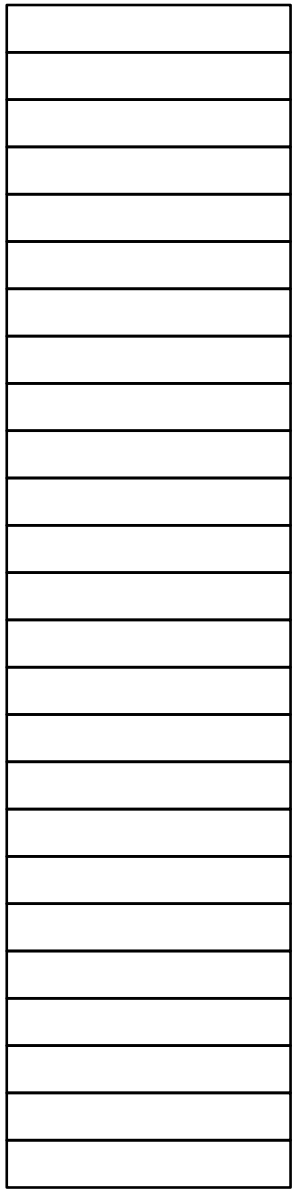
stack

(divided into "pages")

heap

code/text

"Logical Memory"



(divided into "frames")

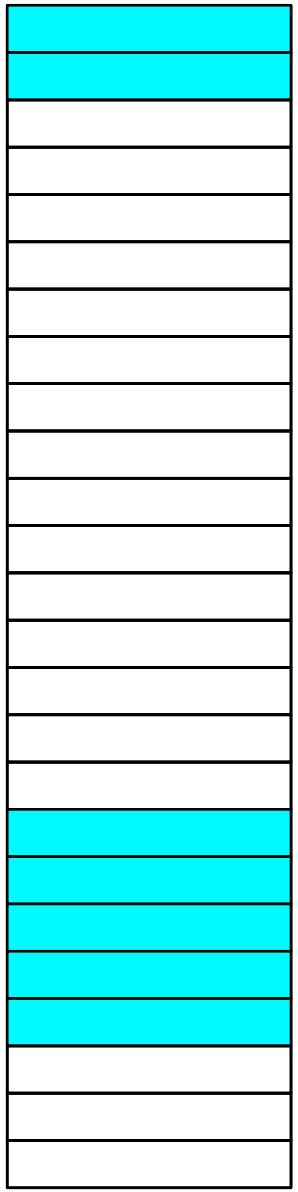
"Physical Memory"

P1's needs some memory...

0xFFFF...

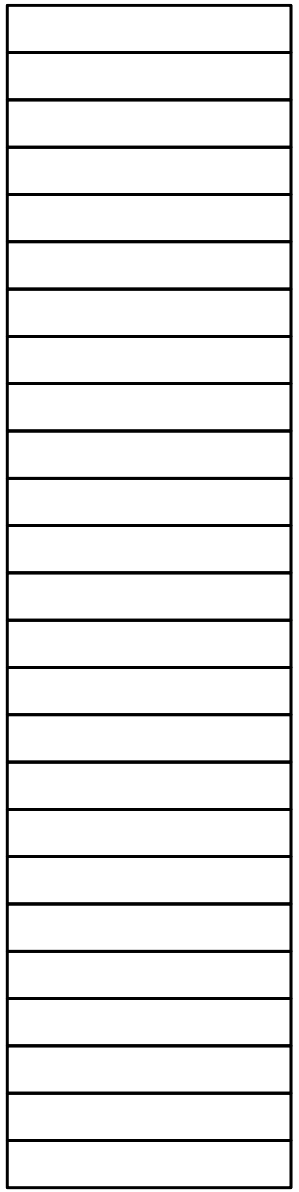
high ("foxes")

(divided into "pages")



0x0000

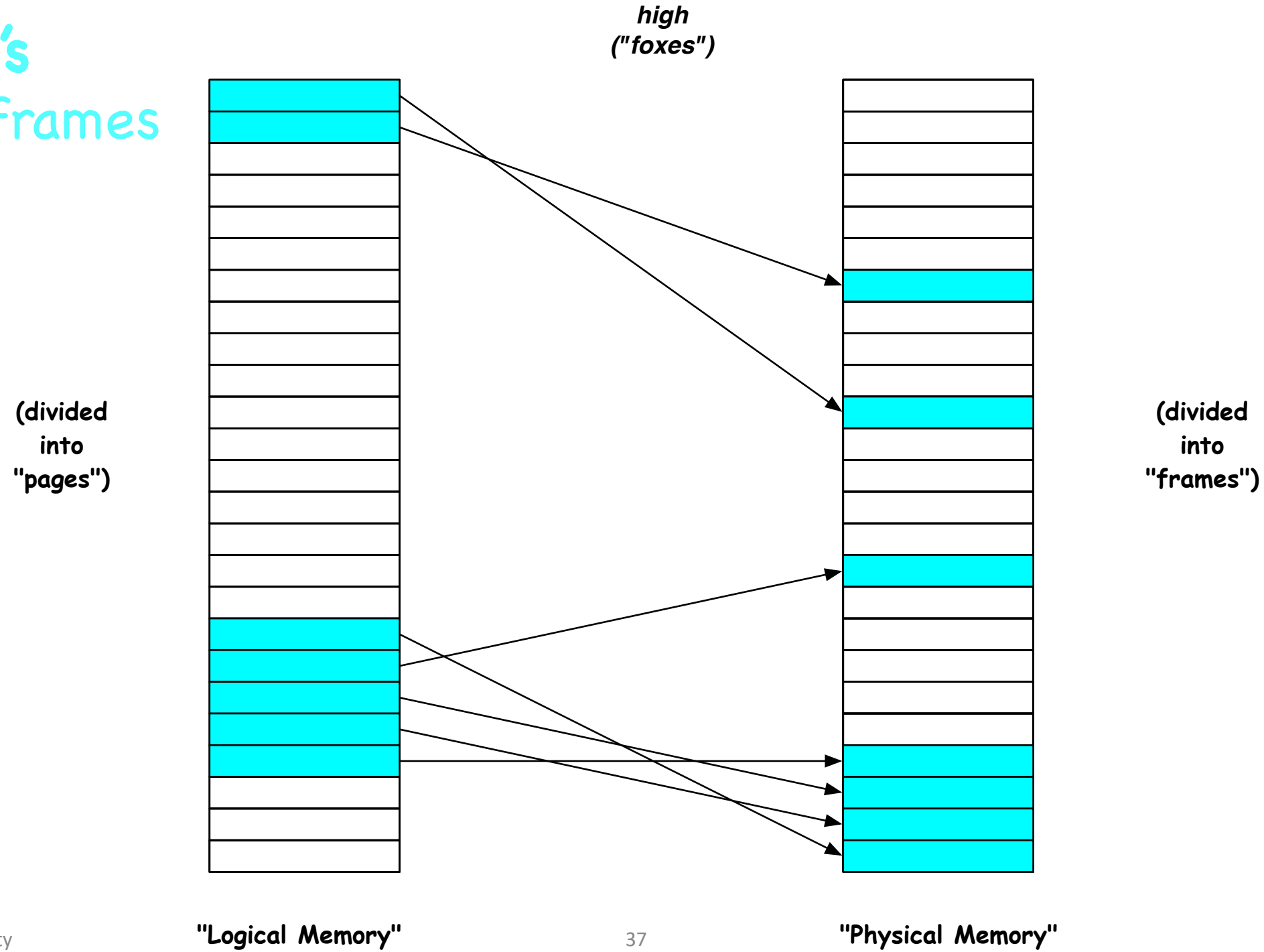
"Logical Memory"



(divided into "frames")

"Physical Memory"

P1's pages/frames



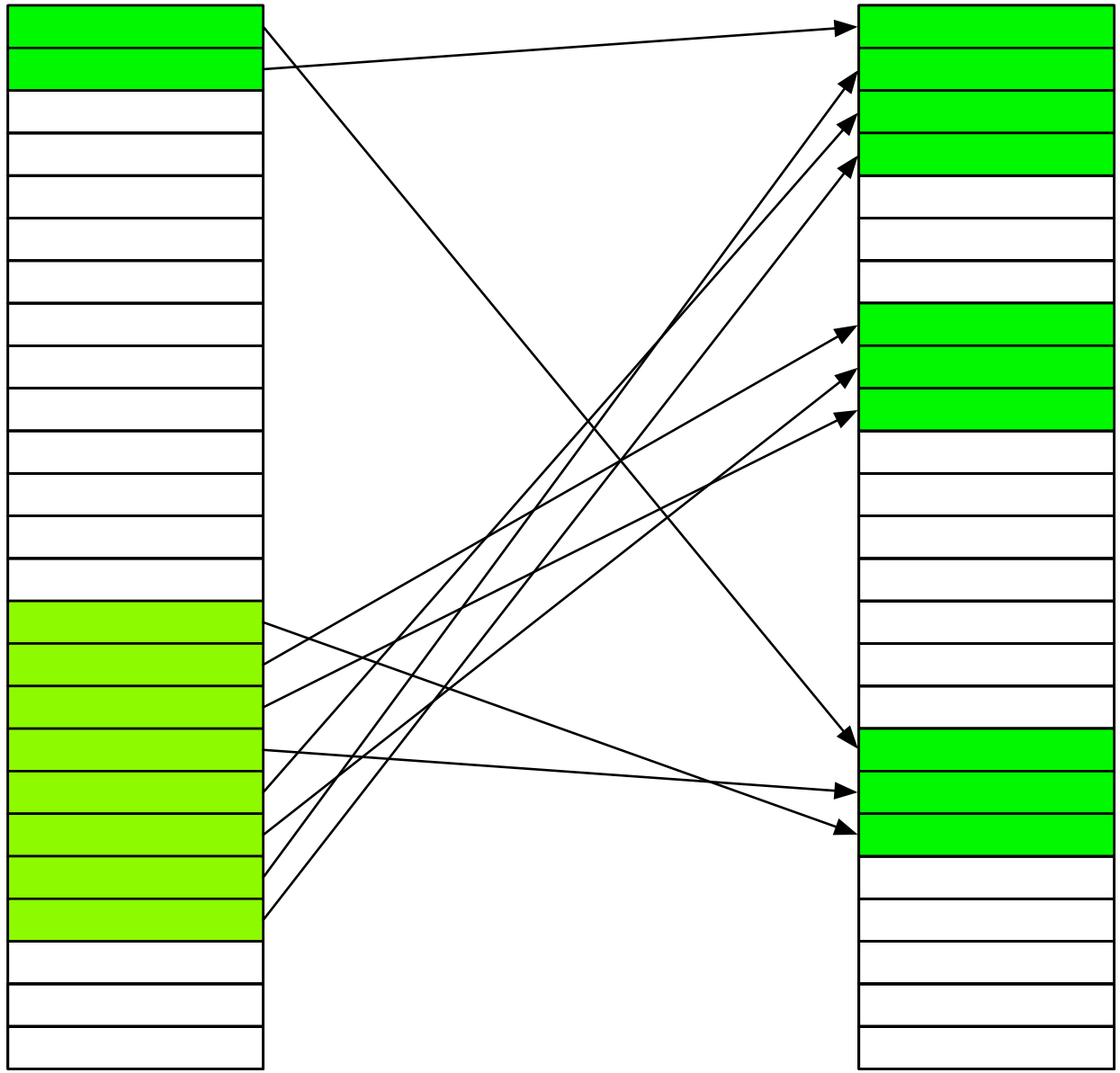
P2's pages/frames

Stack

high ("foxes")

(divided into "pages")

Heap



(divided into "frames")

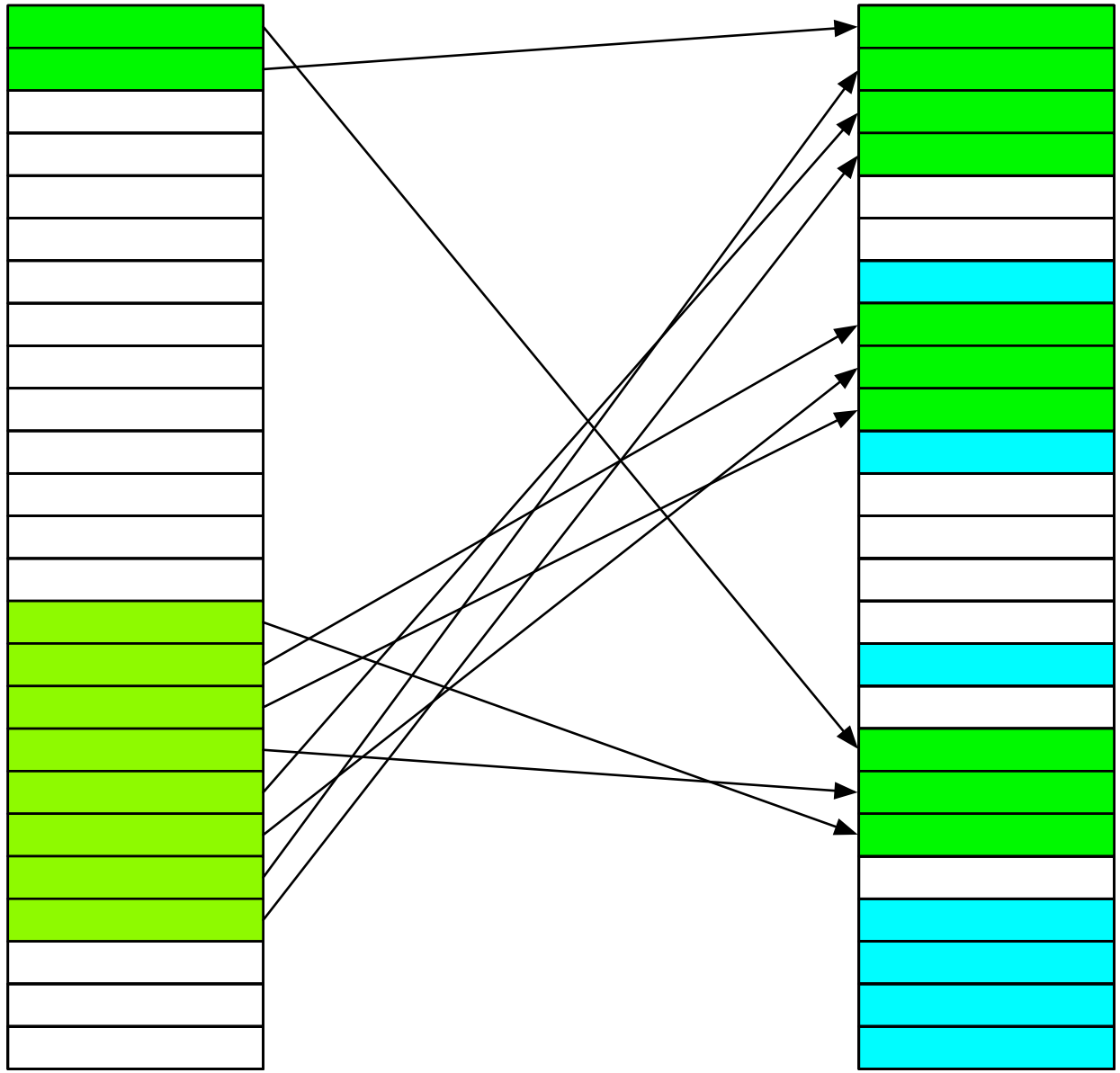
"Logical Memory"

"Physical Memory"

P2
running...

(divided
into
"pages")

high
("foxes")



"Logical Memory"

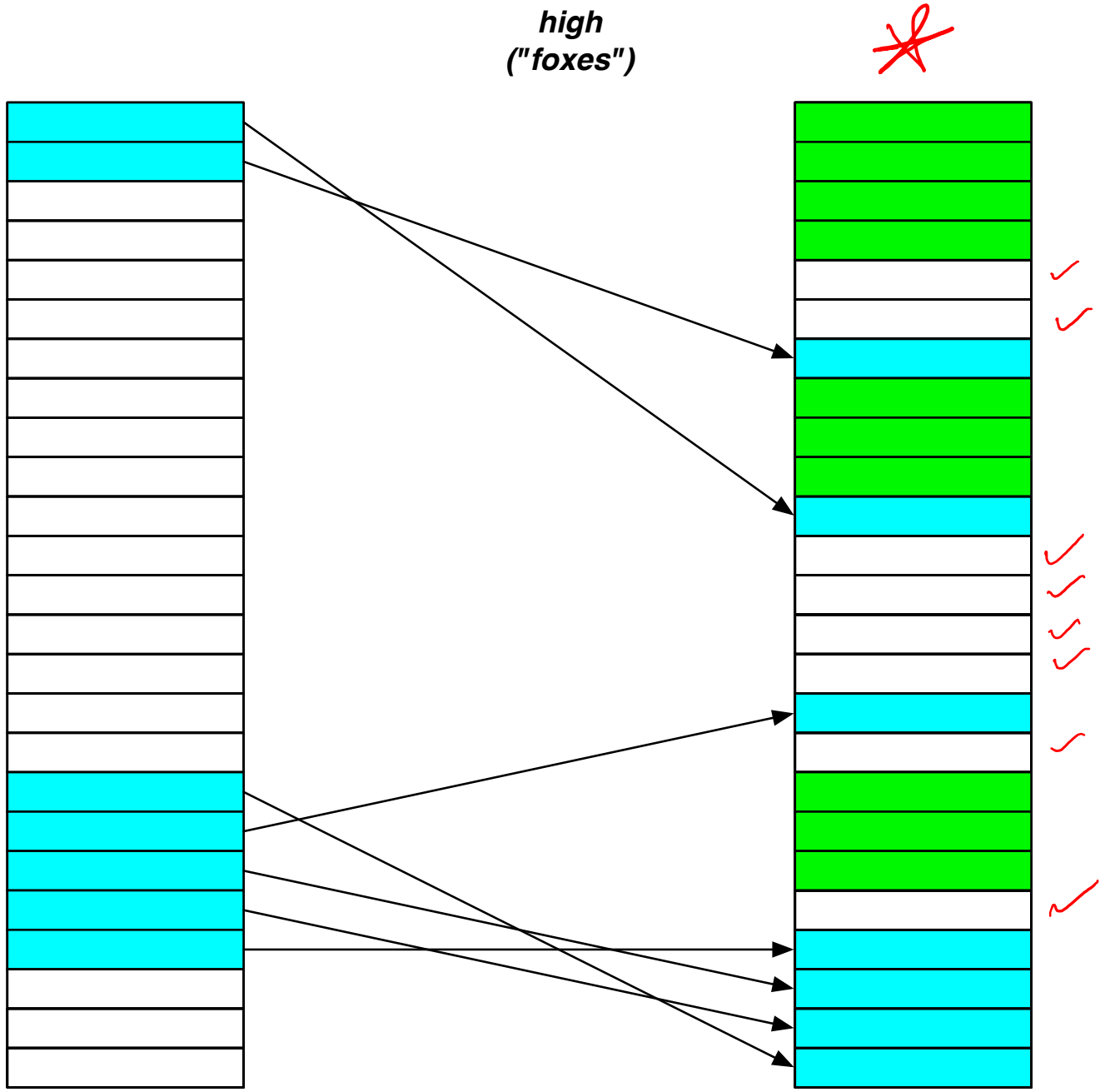
"Physical Memory"

(divided
into
"frames")

P1+P2
in memory!

P1
running...

high
("foxes")



(divided
into
"pages")

(divided
into
"frames")

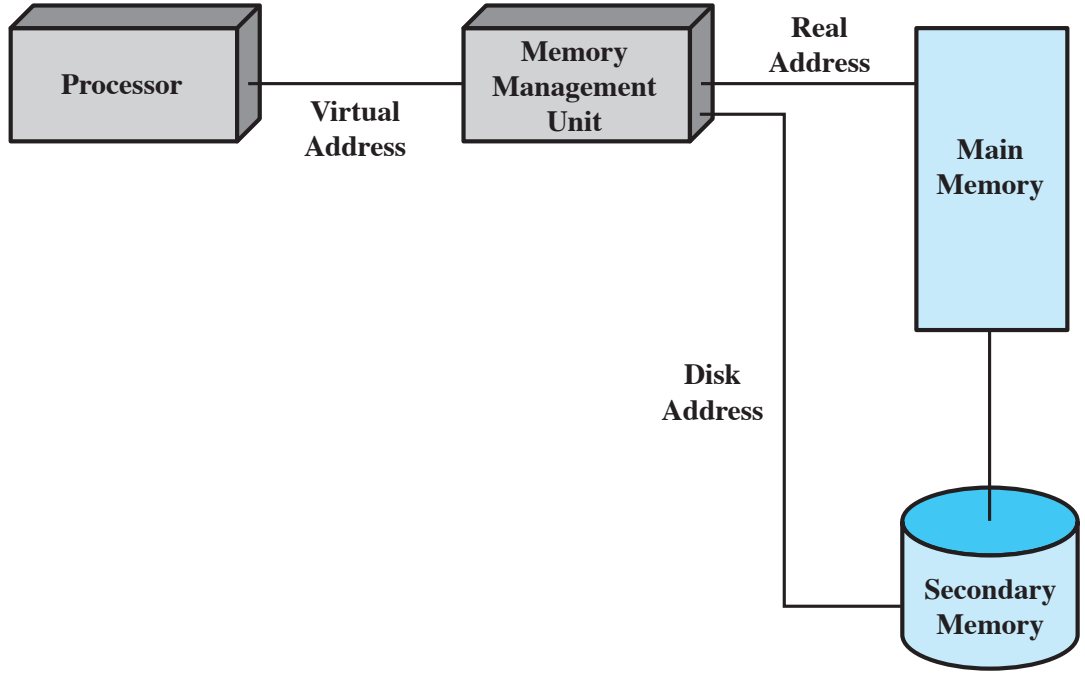
"Logical Memory"

"Physical Memory"

P1+P2
in memory!

Outstanding Issues

- How does the hardware and OS do address translation now?
- How do we keep track of all the free frames?
- What if there aren't enough free frames?
- Do we have to bring ***ALL*** of a process's logical address space into physical memory ***now***?
- Do we have to keep it all in physical memory always?
- Can there be parts of the logical address space that never go into memory?
- Can we put other things into a process's logical address space?
- When we allocate into noncontiguous physical memory like frames, what about space utilization?

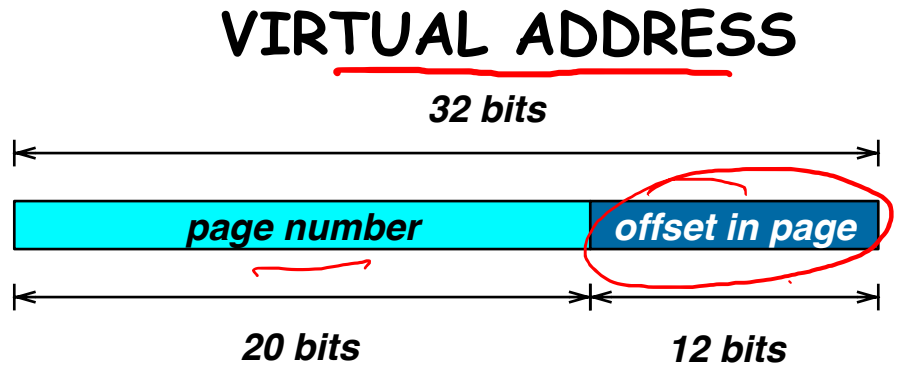


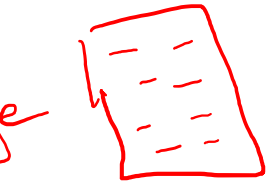
Address Translation

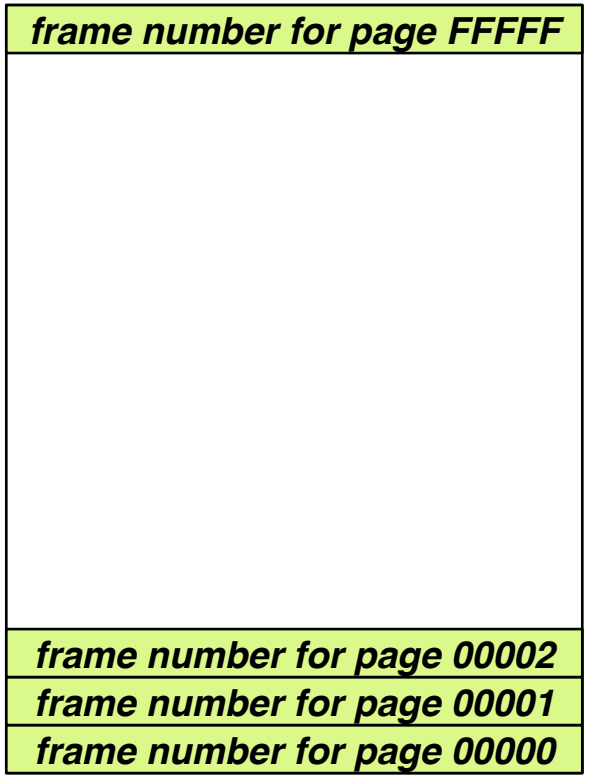
How do we go
from a **virtual address**
to a **physical address**?

Address Translation

- Example MMU address translation
- Single-level page table
- 32-bit address space
- 4 kB pagesize

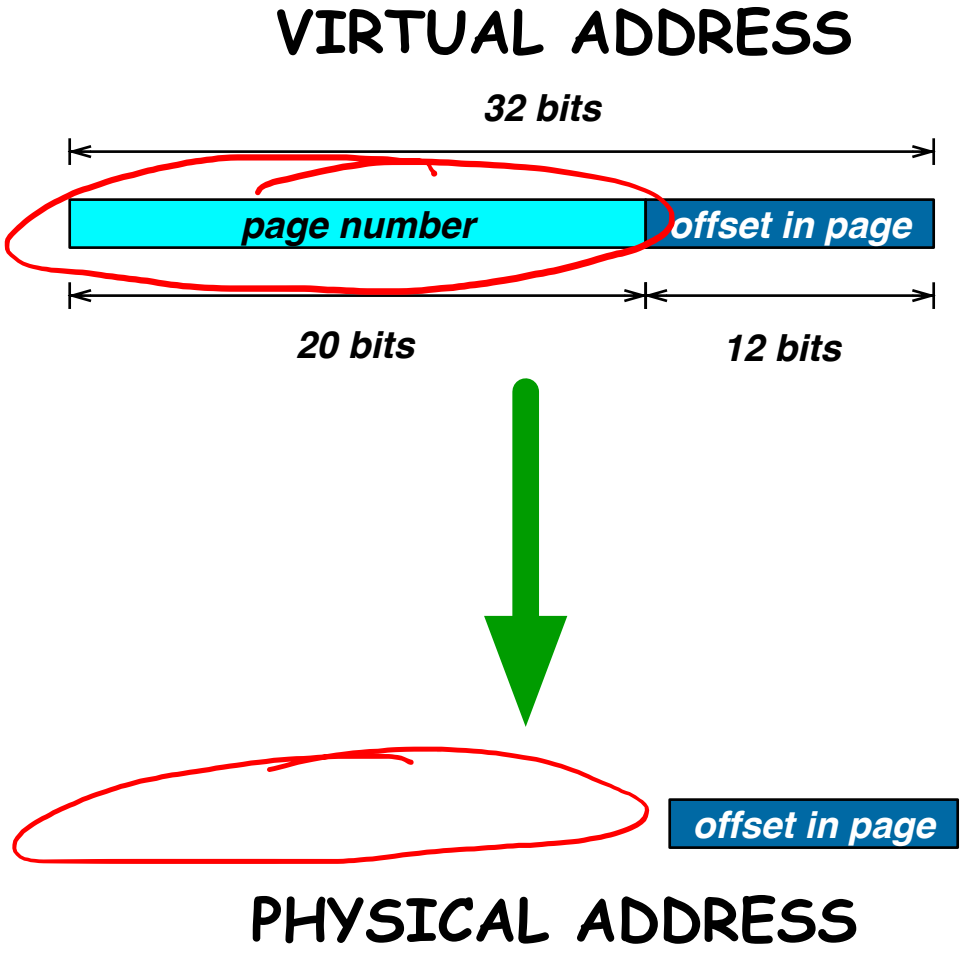


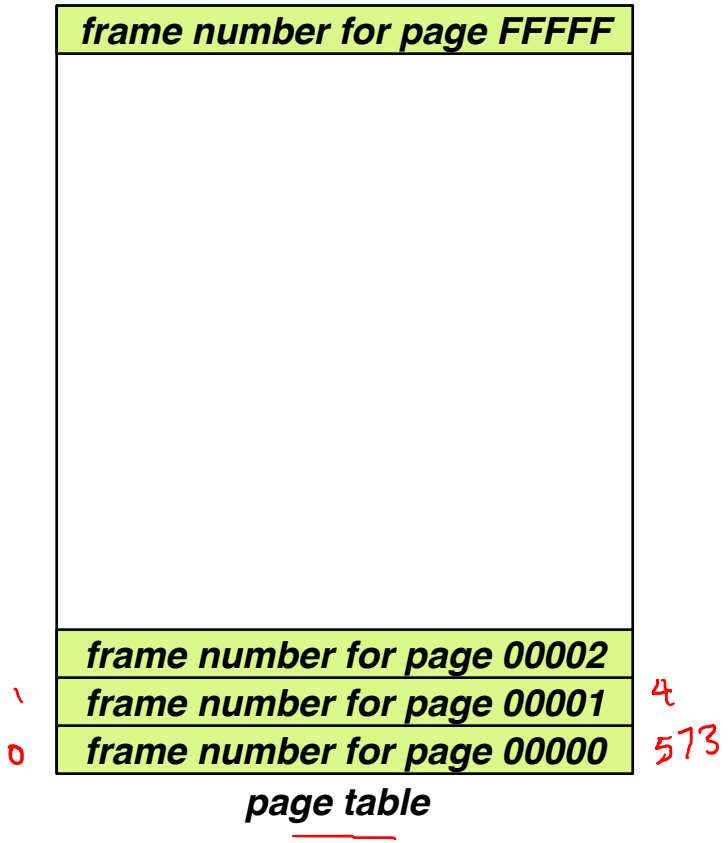




Address Translation

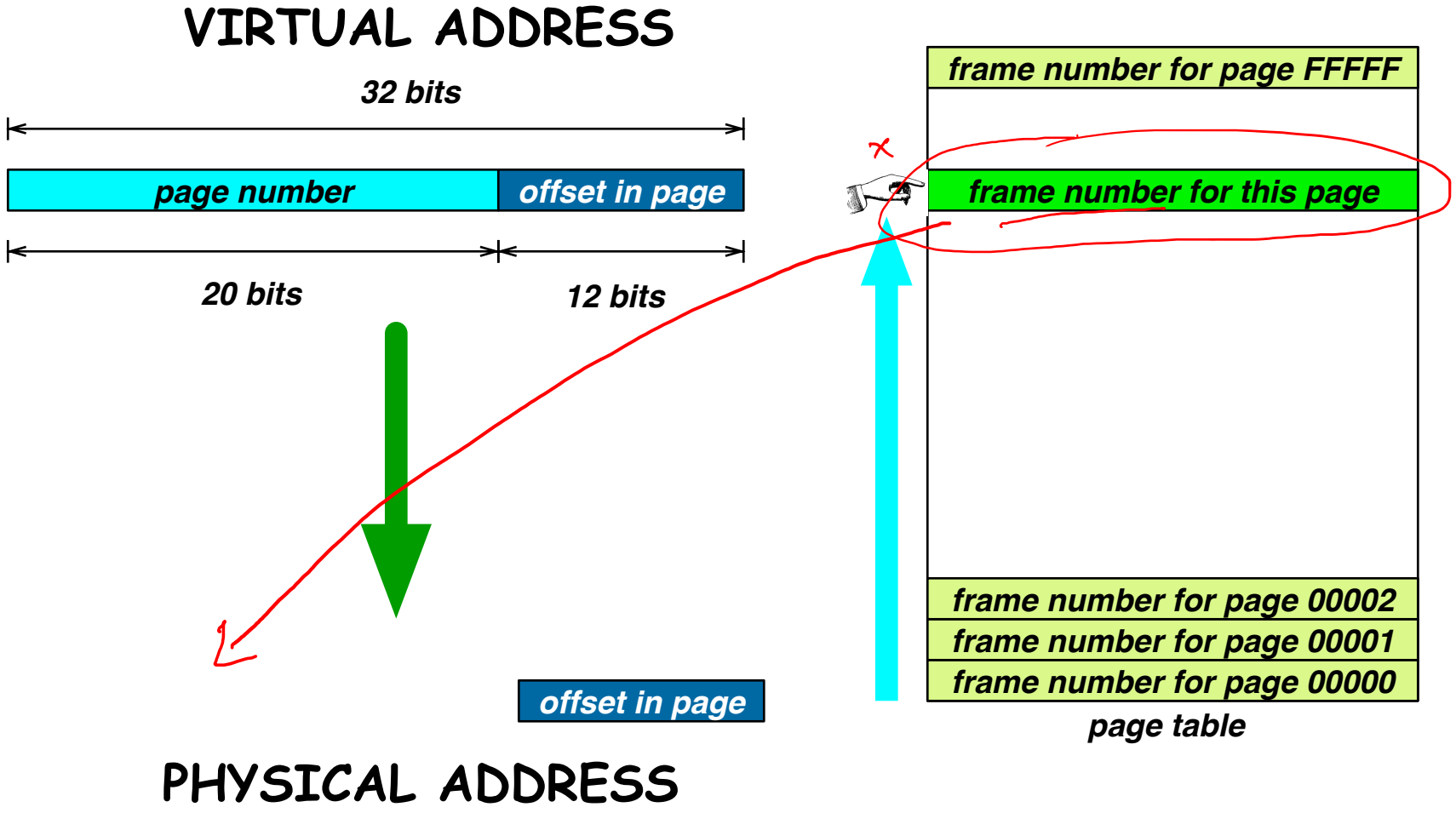
- Example MMU address translation
- Single-level page table
- 32-bit address space
- 4 kB pagesize





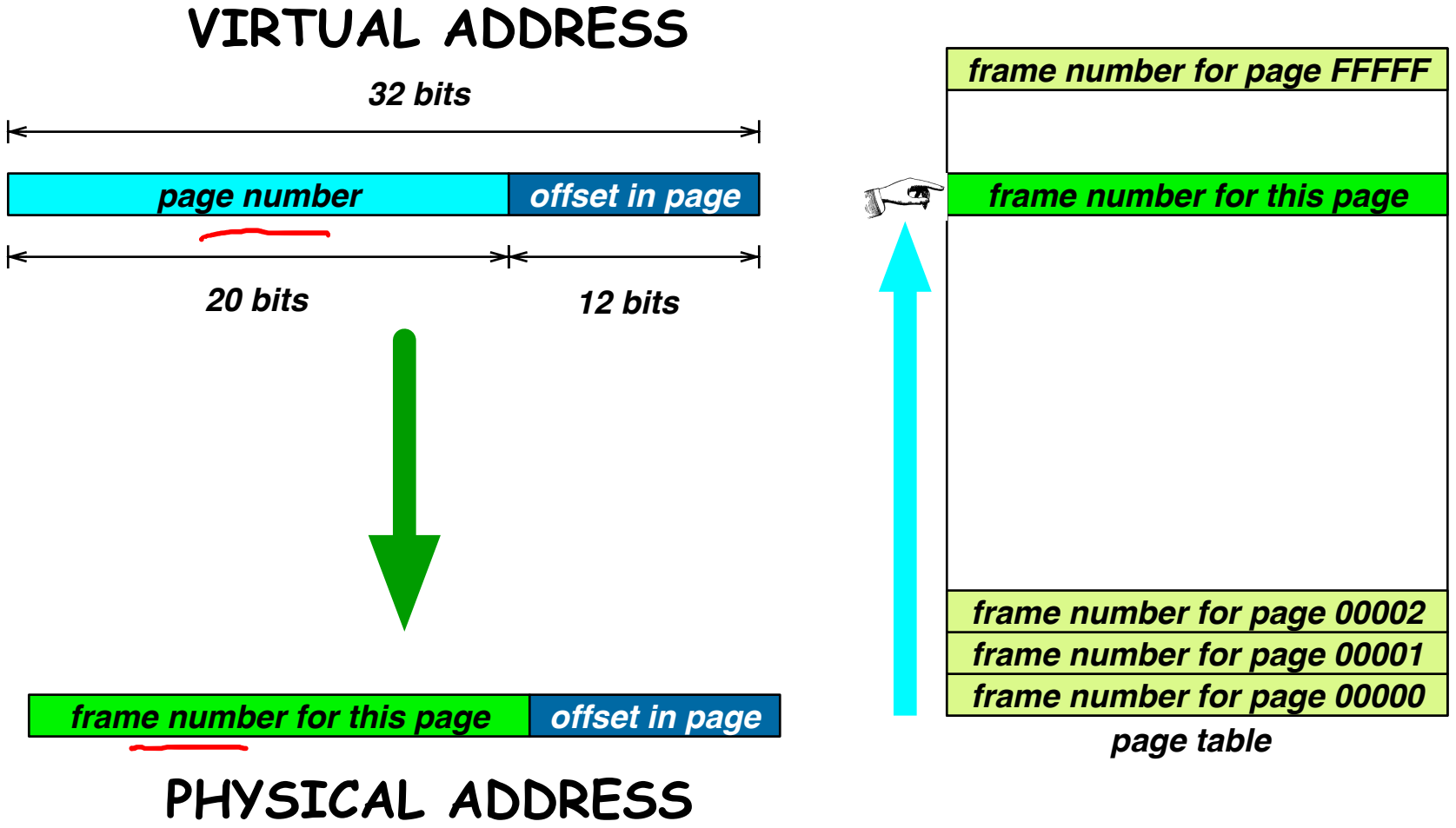
Address Translation

- Example MMU address translation
- Single-level page table
- 32-bit address space
- 4 kB pagesize



Address Translation

- Example MMU address translation
- Single-level page table
- 32-bit address space
- 4 kB pagesize



Probing the Address Space

Exploring address spaces with **fault.c**