Main Memory: Address Translation

CS 4410
Operating Systems

Cornell University
Founded A.D. 1865
Cornell CIS
Computing and Information Science

Address Translation

- Paged Translation
- Efficient Address Translation
  - Multi-Level Page Tables
  - Inverted Page Tables
  - TLBs

Downsides to Paging

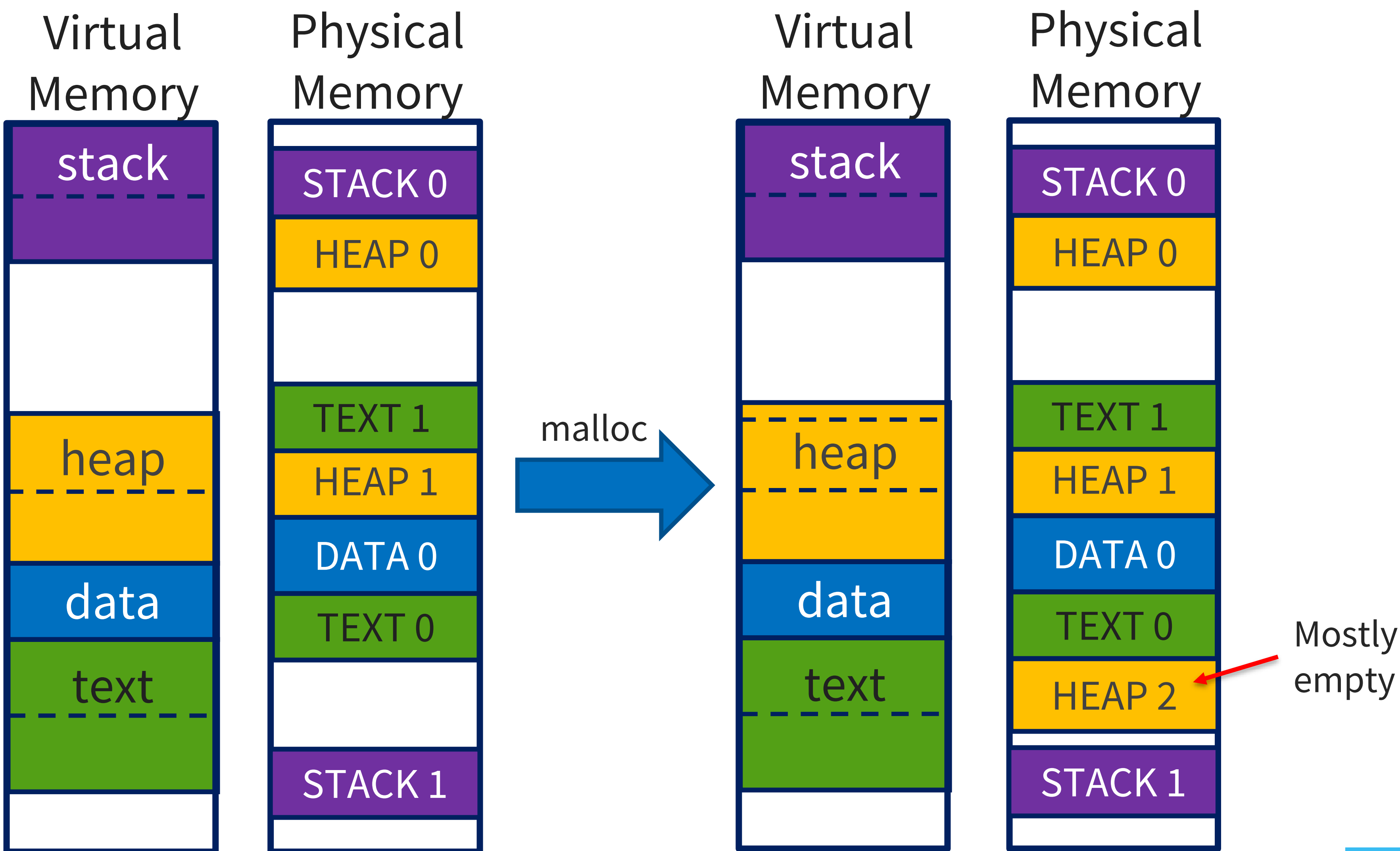
Memory Consumption:

- Internal Fragmentation
  - Make pages smaller? But then…
- Page Table Space: consider 32-bit address space, 4KB page size, each PTE 8 bytes
  - How big is this page table?
  - How many pages in memory does it need?

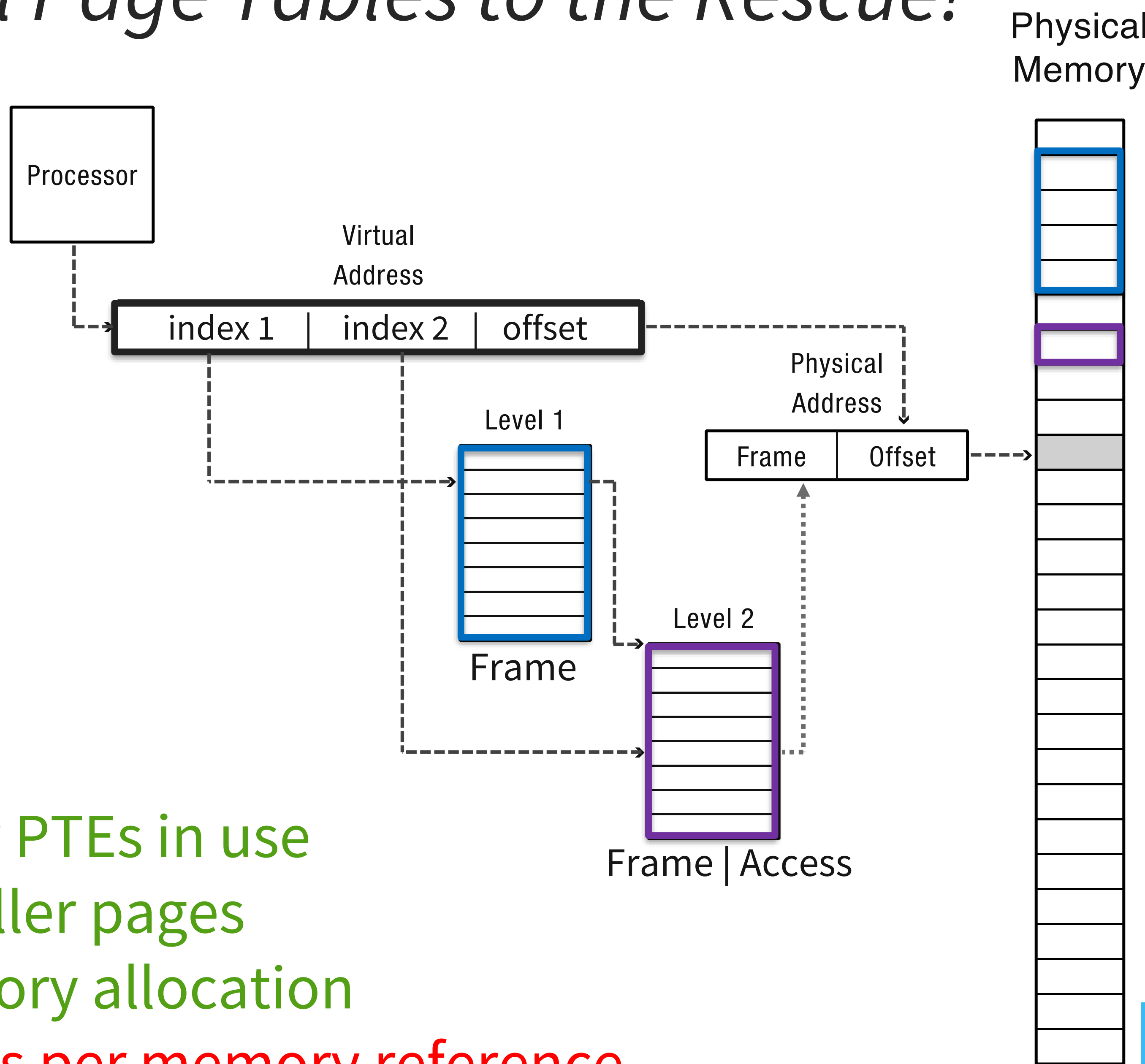
Performance: every data/instruction access requires *two* memory accesses:

- One for the page table
- One for the data/instruction

Internal Fragmentation Example



Multi-Level Page Tables to the Rescue!



+ Allocate only PTEs in use
+ Can use smaller pages
+ Simple memory allocation
− ***more*** lookups per memory reference

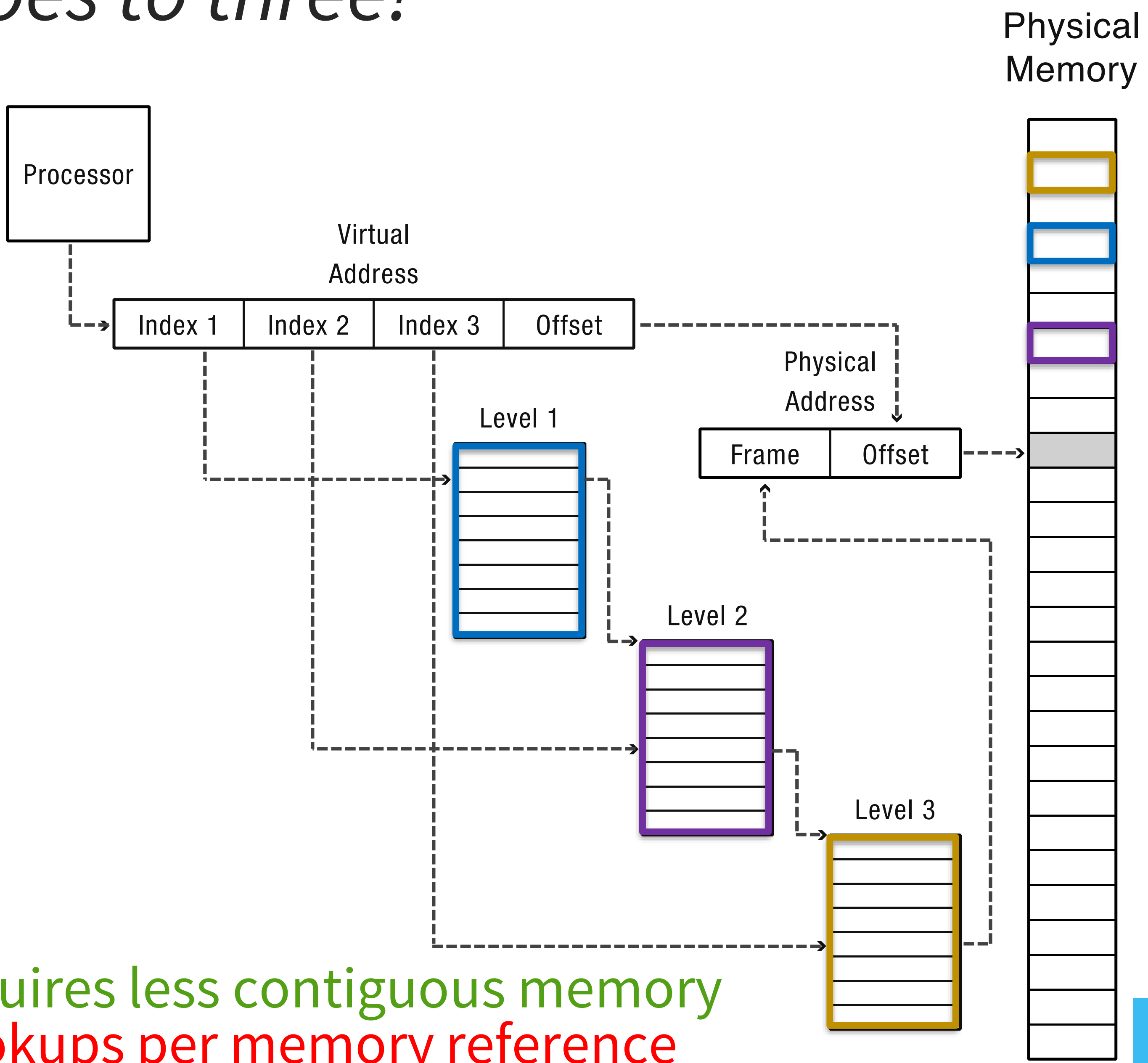
Two-Level Paging Example

32-bit machine, 1KB page size

- Logical address is divided into:
  - a page offset of 10 bits (1024 = 2^10)
  - a page number of 22 bits (32-10)
- Since the page table is paged, the page number is further divided into:
  - a 12-bit first index
  - a 10-bit second index
- Thus, a logical address is as follows:

page number		page offset
index 1	index 2	offset
12	10	10

This one goes to three!



+ First Level requires less contiguous memory

− ***even more*** lookups per memory reference

Complete Page Table Entry (PTE)

Valid	Protection R/W/X	Ref	Dirty	Index

Index is an index into:

- table of memory frames (if bottom level)
- table of page table frames (if multilevel page table)
- backing store (if page was swapped out)

Synonyms:

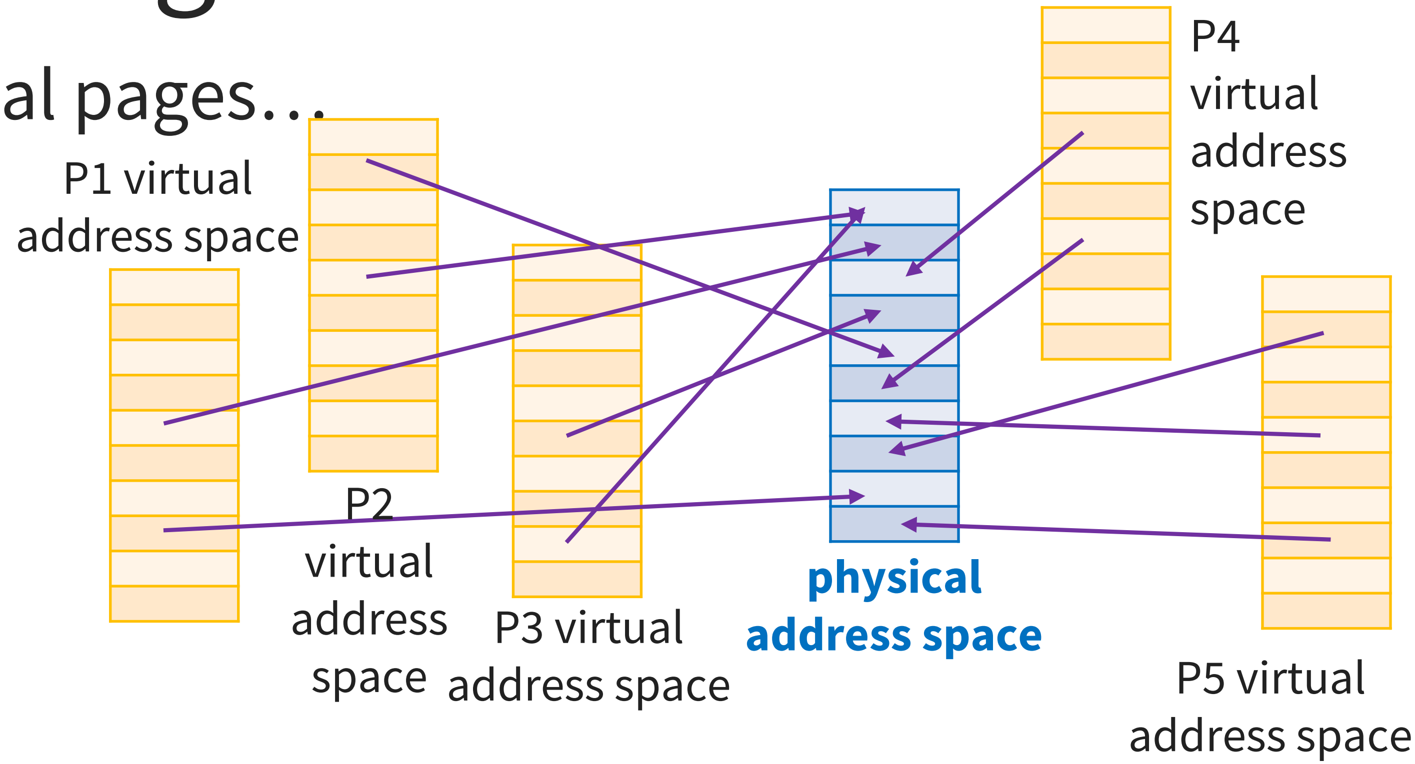
- Valid bit == Present bit
- Dirty bit == Modified bit
- Referenced bit == Accessed bit

Address Translation

- Paged Translation
- Efficient Address Translation
  - Multi-Level Page Tables
  - Inverted Page Tables
  - TLBs

Inverted Page Table: Motivation

So many virtual pages…



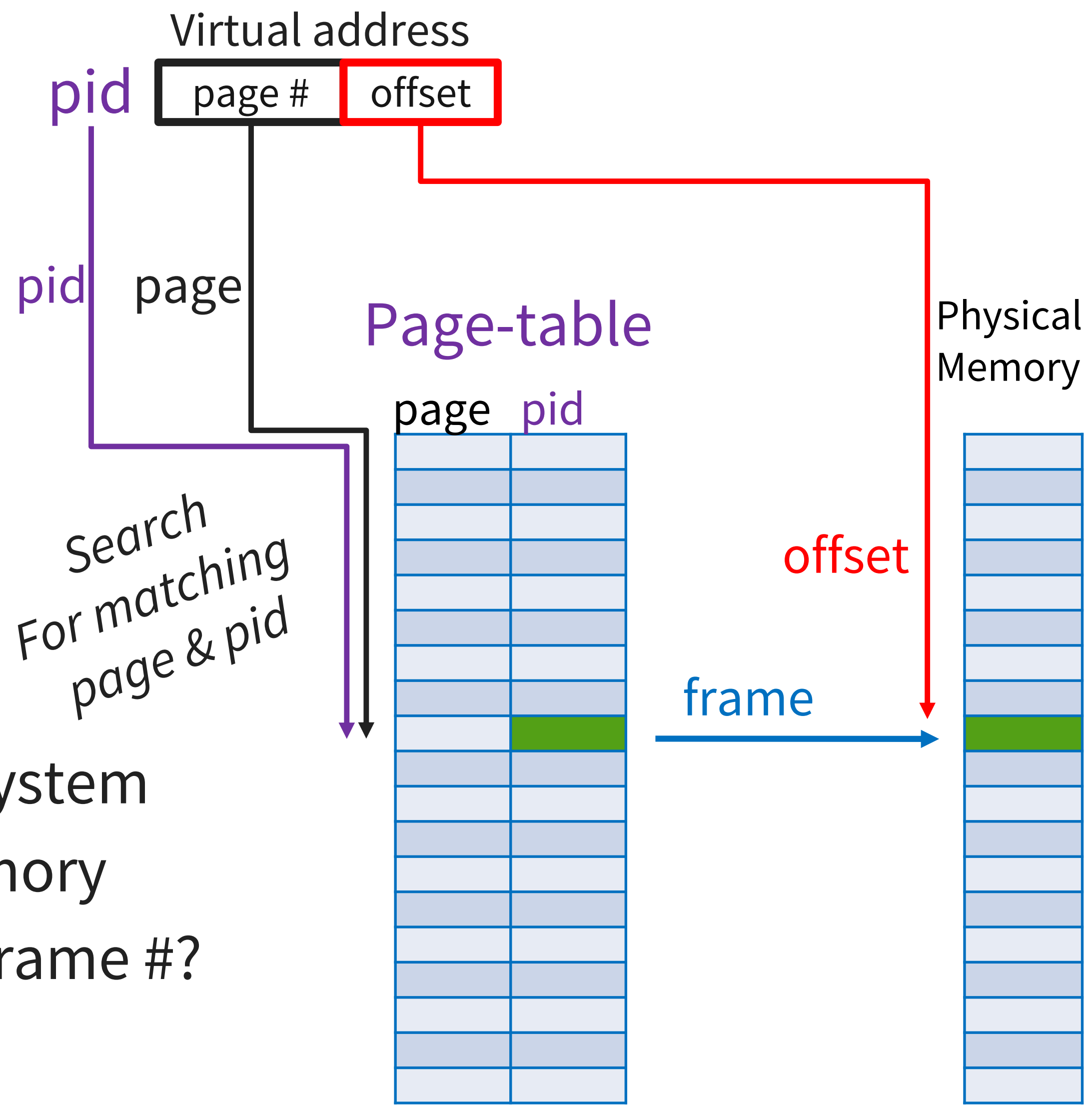
… comparatively few physical frames

Traditional Page Tables:

- map pages to frames
- are *numerous and sparse*

Why not map frames to pages? (How?)

Inverted Page Table: Implementation



Implementation:

- 1 Page Table for entire system
- 1 entry per frame in memory
- Why don't we store the frame #?

Not to scale! Page table << Memory

Inverted Page Table: Discussion

Tradeoffs:

↓ memory to store page tables

↑ time to search page tables

Solution: hashing

- hash(page,pid) → PT entry (or chain of entries)
- What about:
  - collisions…
  - sharing…

Address Translation

- Paged Translation
- Efficient Address Translation
  - Multi-Level Page Tables
  - Inverted Page Tables
  - TLBs

Page Table Memory Lookups

How many memory accesses per data/instruction access?

- One per level of the page table
- One for the data/instruction

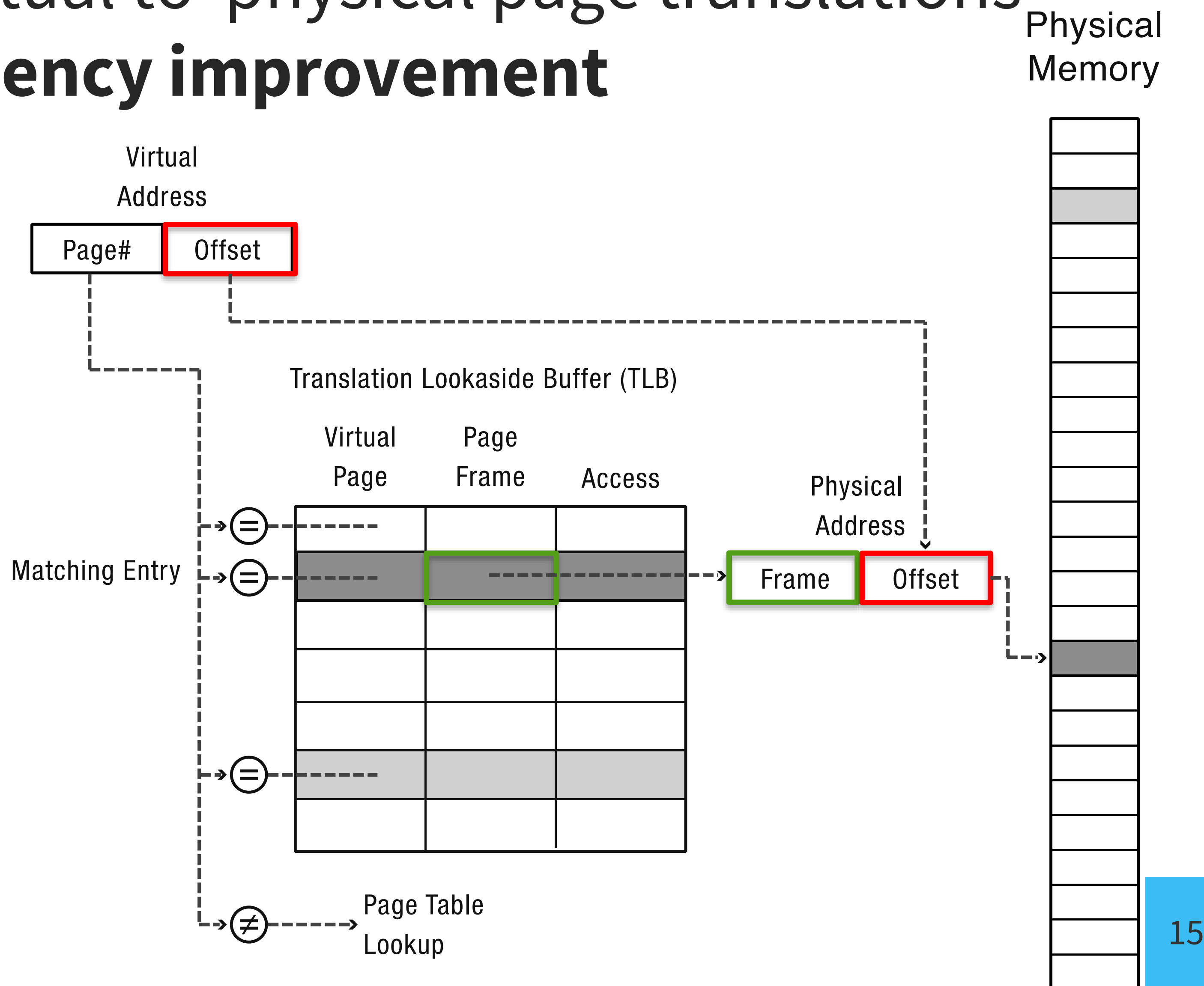
Workarounds

- CPU cache: Recently accessed data is still there, at virtual address
- Does this help for instructions?

Translation Lookaside Buffer (TLB)

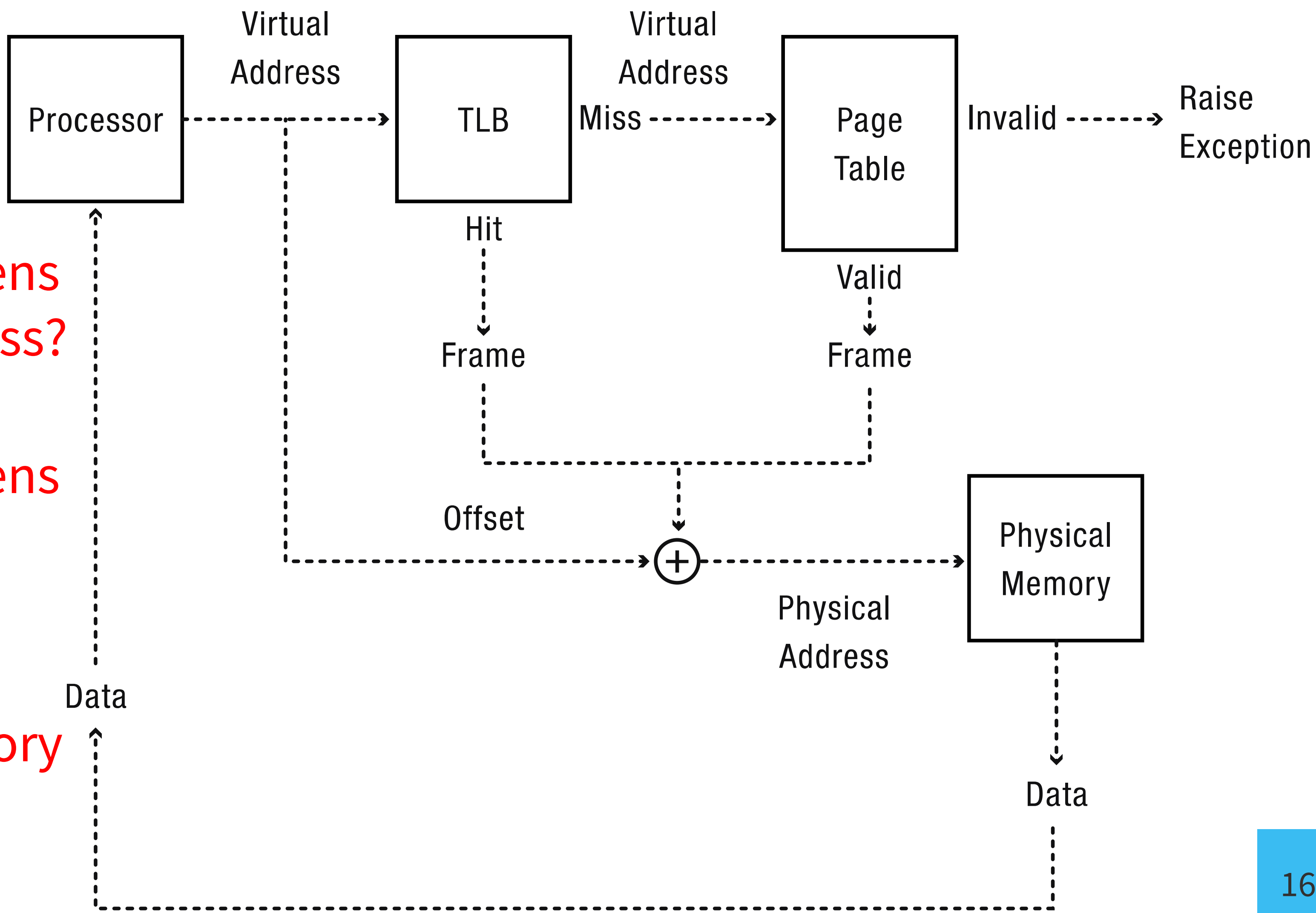
Cache of virtual to physical page translations

Major efficiency improvement



Address Translation with TLB

Access TLB before you access memory.



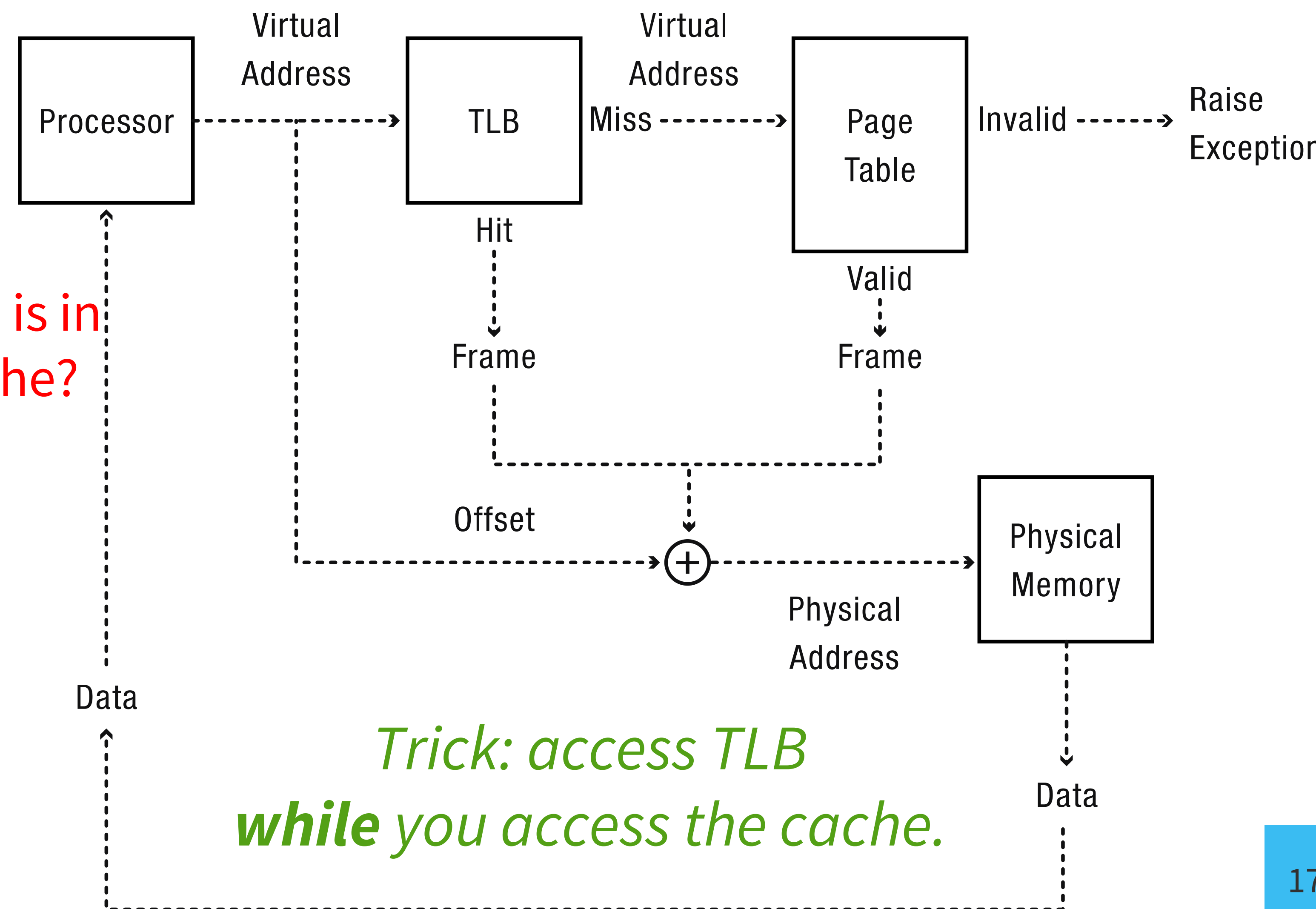
What happens on a TLB miss?

What happens on context switch?

When memory is freed?

Address Translation with TLB

Access TLB before you access memory.



What if data is in the CPU cache?

Trick: access TLB ***while*** *you access the cache.*

Address Translation Uses!

Process isolation

- Keep a process from touching anyone else’s memory, or the kernel’s

Efficient inter-process communication

- Shared regions of memory between processes

Shared code segments

- common libraries used by many different programs

Program initialization

- Start running a program before it is entirely in memory

Dynamic memory allocation

- Allocate and initialize stack/heap pages on demand

MORE Address Translation Uses!

Program debugging

- Data breakpoints when address is accessed

Memory mapped files

- Access file data using load/store instructions

Demand-paged virtual memory

- Illusion of near-infinite memory, backed by disk or memory on other machines

Next lecture

Checkpointing/restart

- Transparently save a copy of a process, without stopping the program while the save happens

Distributed shared memory

- Illusion of memory that is shared between machines