CS 1217

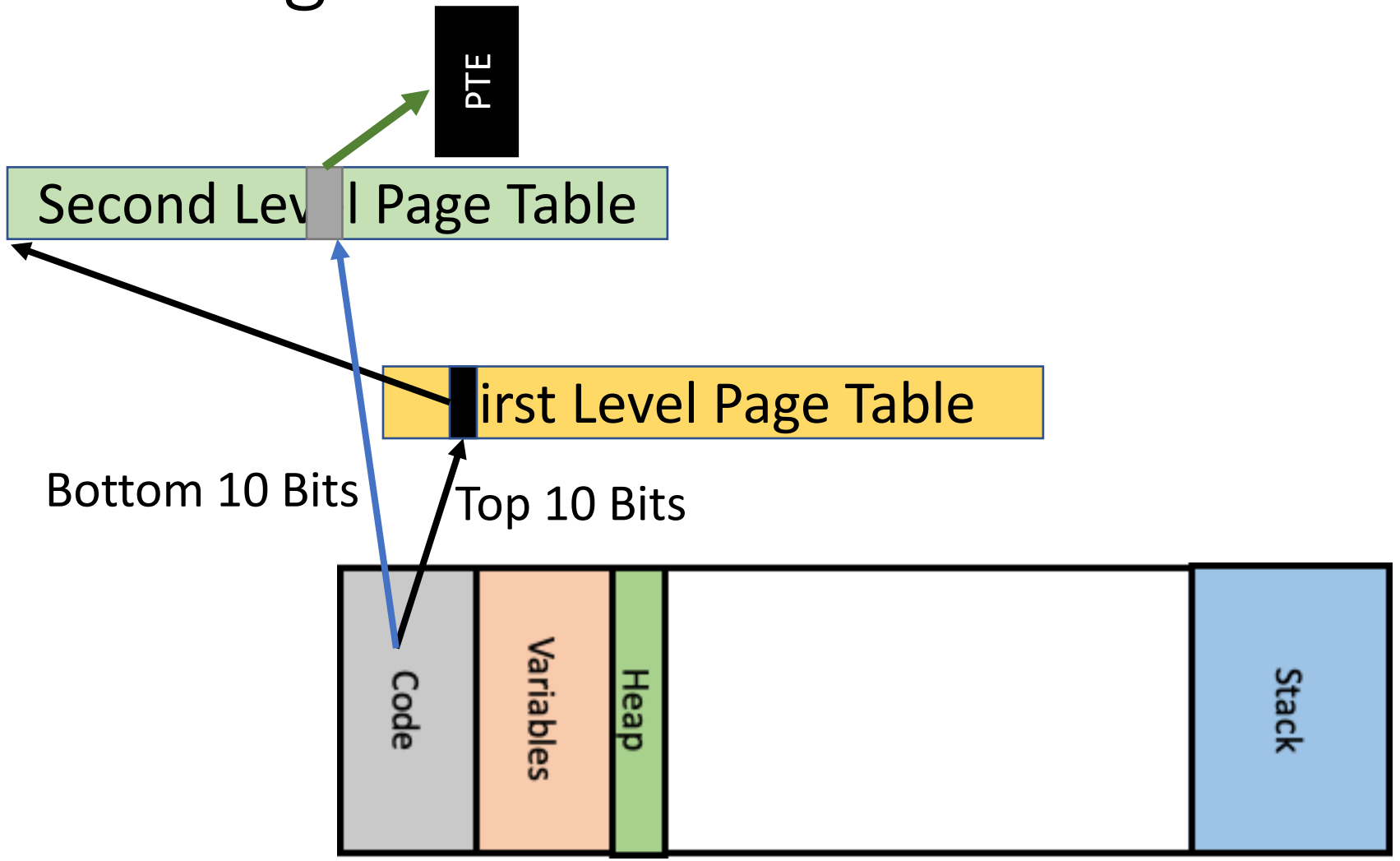
Lecture 17: Page Tables wrap, Paging, Intro to Swapping

Multi Level Page Tables

- Insight: Utilize the sparsity of virtual address space
- Key Idea: Break VPN into multiple parts, each used as an index at a separate level of the tree.

- Example: with 4K pages VPN is **20** bits. Use **top 10** bits as index into **top-level** page table, **bottom 10 bits** as index into second-level page table

Multi Level Page Tables



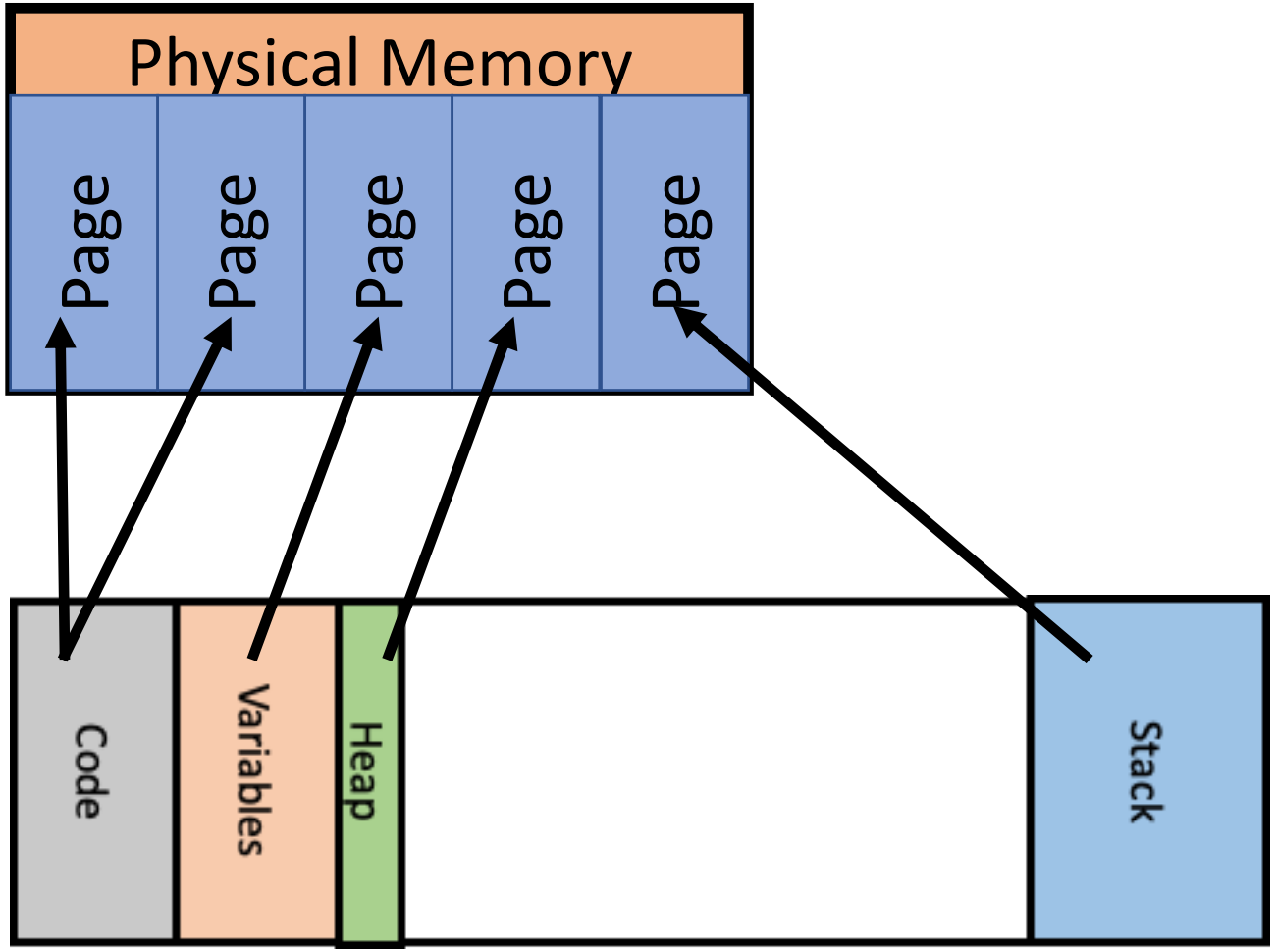
Multi Level Page Tables

- Time Complexity:
- O(c). Constant number of look ups per translation depending on tree depth

- Space Complexity:
- Depends on sparsity of address space, but better than flat.

Out of Memory

- But what happens when we **run out** of physical memory?



Options

- **Fail**
  - don’t load the process (**exec()**)
  - don’t create a new process (**fork()**)
  - refuse to allocate more heap (**sbrk()**)
  - kill the process if it is trying to allocate more stack space

- **Create more space**, preserving the contents of memory for later use.

Swapping

- Swap Space: The on-disk space that operating systems typically place data stored in memory in order to *borrow* memory.

- Swapping: Process of moving (swapping) data back and forth between memory and disk

- Goal: make it **feel** like the system has memory that is as *large* as the size of the disk and as *fast* as actual RAM

Page Fault vs. TLB Fault

- **TLB Fault**
  - a required virtual to physical address translation is not in the TLB.

- **Page Fault**
  - the contents of a virtual page are either not initialized or not in memory
- Does every Page Fault generate a TLB Fault?

- Does every TLB Fault generate a Page Fault?

Swap Out Process

- **Remove** the translation from the TLB, if it exists
- **Copy** the contents of the page to disk.
- **Update** the page table entry to indicate that the page is on disk

Things about TLBs

- Where do entries in the TLB come from?

- What happens if a process tries to access an address that is **not** in the TLB?

TLB Management

- Two ways : software vs hardware
  - We have assumed the former in our discussions
- On some architectures the MMU will search the page table **itself** to locate virtual-to-physical address translations missing from the TLB.
- Pros?
  - Faster
- Cons?
  - The H/W now has to **know** the structure of page tables; limits optimizations that the OS can do with them
- What happens to TLB faults in the case of H/W managed TLBs?
  - the kernel never sees TLB faults: they are handled in hardware

Swap Out Process