Final Review Part 4

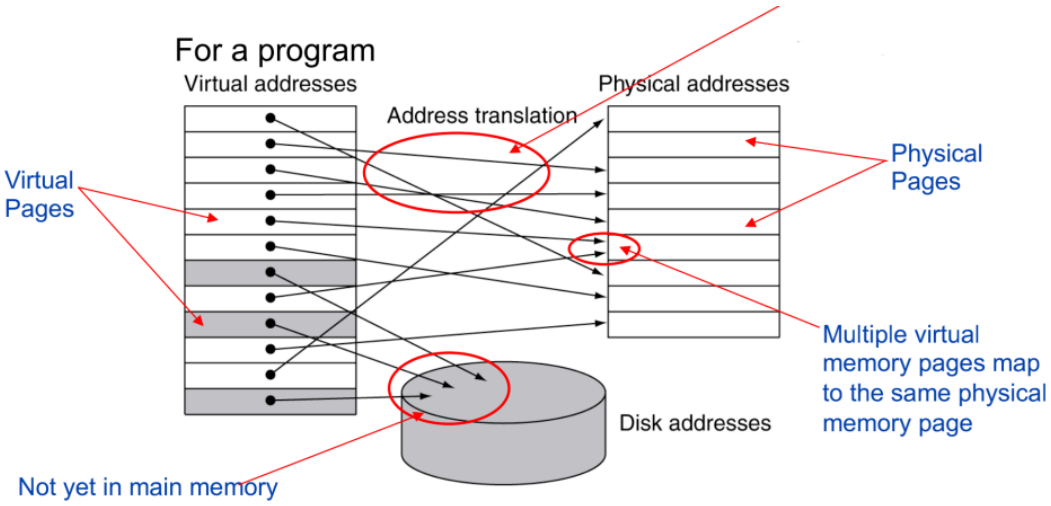
Virtual Memory

Why Virtual Memory

- Computer may have a huge program that requires memory to be much larger than cache and main memory.
- Computer may run multiple programs. They may share the main memory, but we don't want them to communicate with each other.
- CPU interacts with main memory through cache, but we don't want it be bothered by memory issues.

What is Virtual Memory

- An imaginary, huge and fast memory from CPU's perspective – mapped to physical memory.
- Each program has a virtual memory space.
- Mapping is done by CPU or OS translating specific virtual addresses to specific physical addresses.
- If a requested page is not in main memory, it should be fetched from **swap space** in disk (but time consuming).

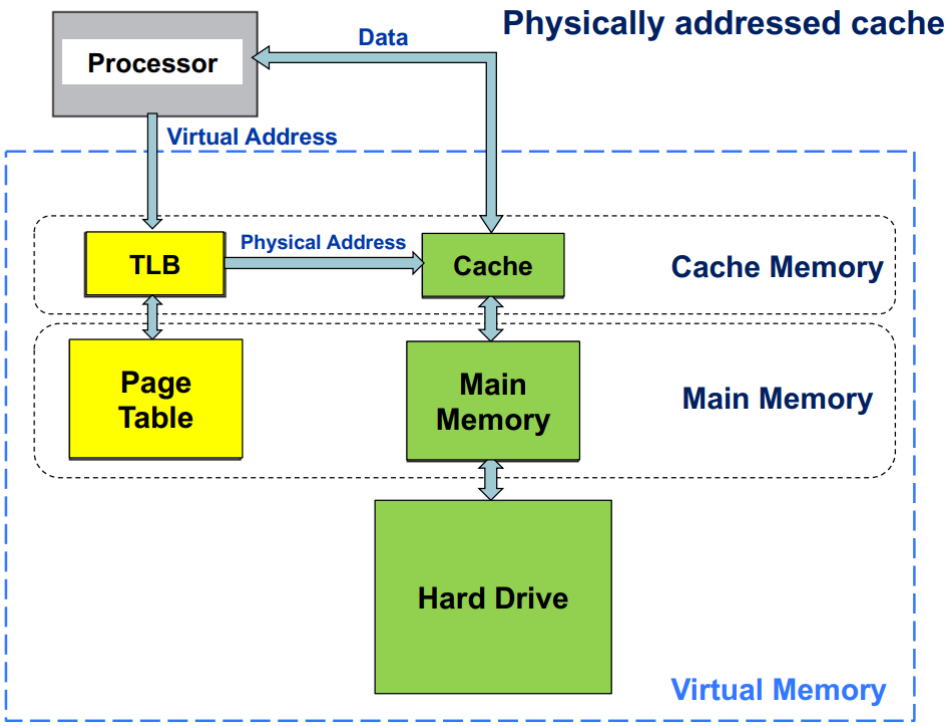


Page, Page Table and Page Fault

- In virtual memory context, the data transfer unit is **page** (larger than a block).
- Use **page offset** to locate each byte in a page.
- **Page table** is used for translating **virtual page number (VPN)** into **physical page number (PPN)**. Use VPN as an index to locate the corresponding PPN.
- **Page fault**: The requested page does not exist in the main memory. Need to go to disk to fetch it.

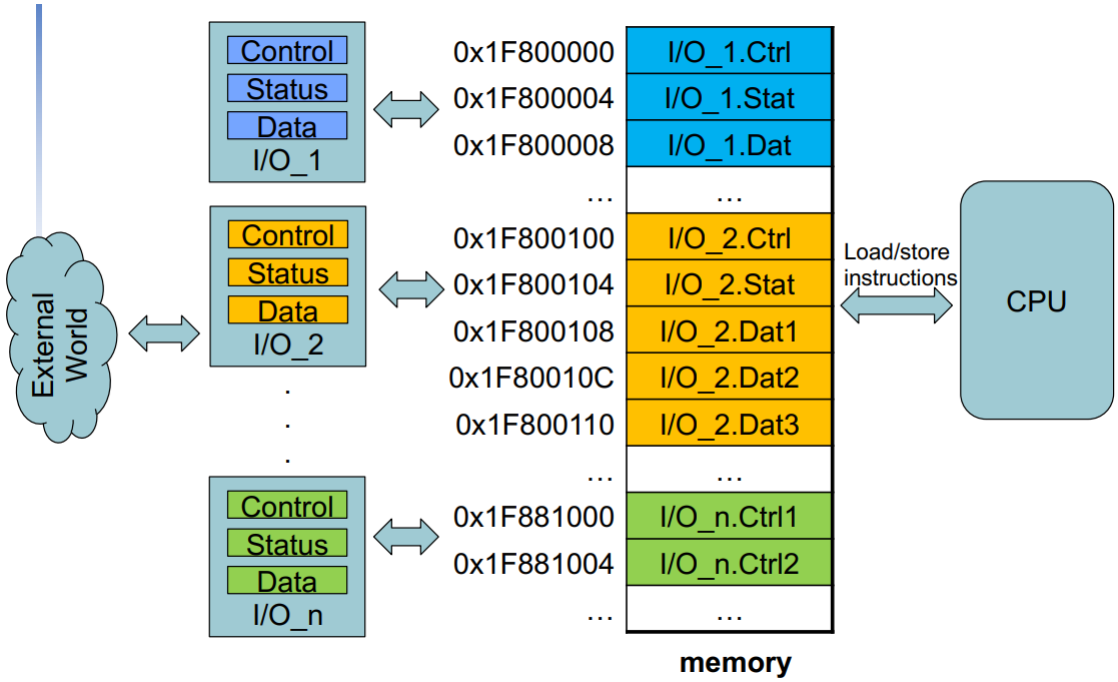
Translation Look-aside Buffer (TLB)

- (TLB ← page table) = (cache ← main memory)
- Located inside CPU
- Fully associative
- Tag field used as VPN



IO & Interfaces

- IO control register:
  - command register: cause device to do something
  - status register: indicate what the device is doing or has done and occurrence of errors
  - data register: write (transfer data to an I/O device), read (transfer data from an I/O device)
- Memory mapped IO
  - I/O registers are connected to memory locations
  - I/Os are accessed as regular memory locations



- Polling vs. Interrupt
  - Polling: periodically check I/O status register (waste CPU time)
  - Interrupt: notify processors when a device is ready or error occurs
- Direct memory access
  - Processor sets up DMA controller by providing device ID, starting address in memory, number of bytes to transfer, triggering events
  - I/O controller interrupt or CPU (software) requests data transfer to start
  - DMA controller transfers between memory & I/O device autonomously without supervision of CPU
  - DMA Controller interrupts to call CPU attention on completion or critical events or error