
SEAGATE


Western Digital


SEAGATE

IOWA STATE UNIVERSITY
Department of Electrical and Computer Engineering

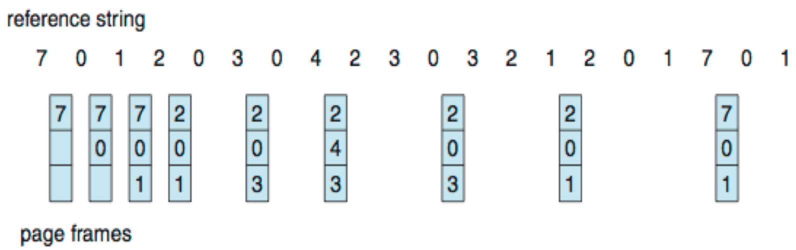
Lecture 26: Introduction to I/O

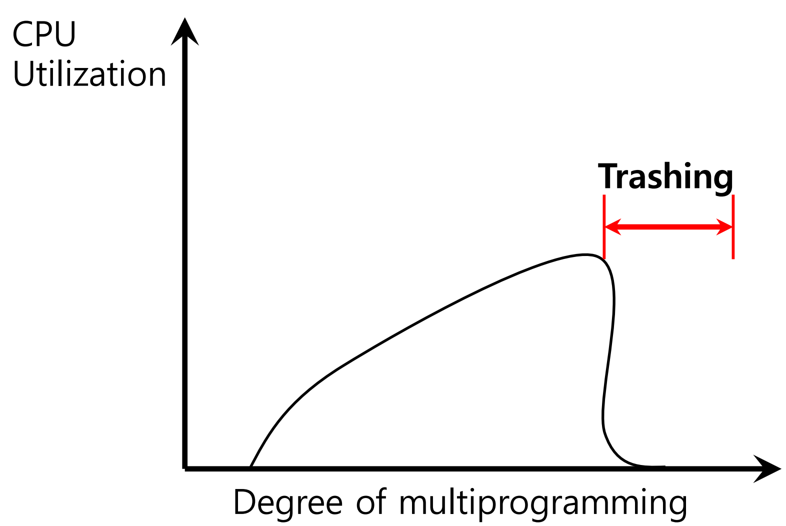
Agenda

- Recap
- Introduction to I/O
  - I/O Device
  - I/O Bus
  - I/O Interrupt
  - DMA
  - Device Driver

Recap

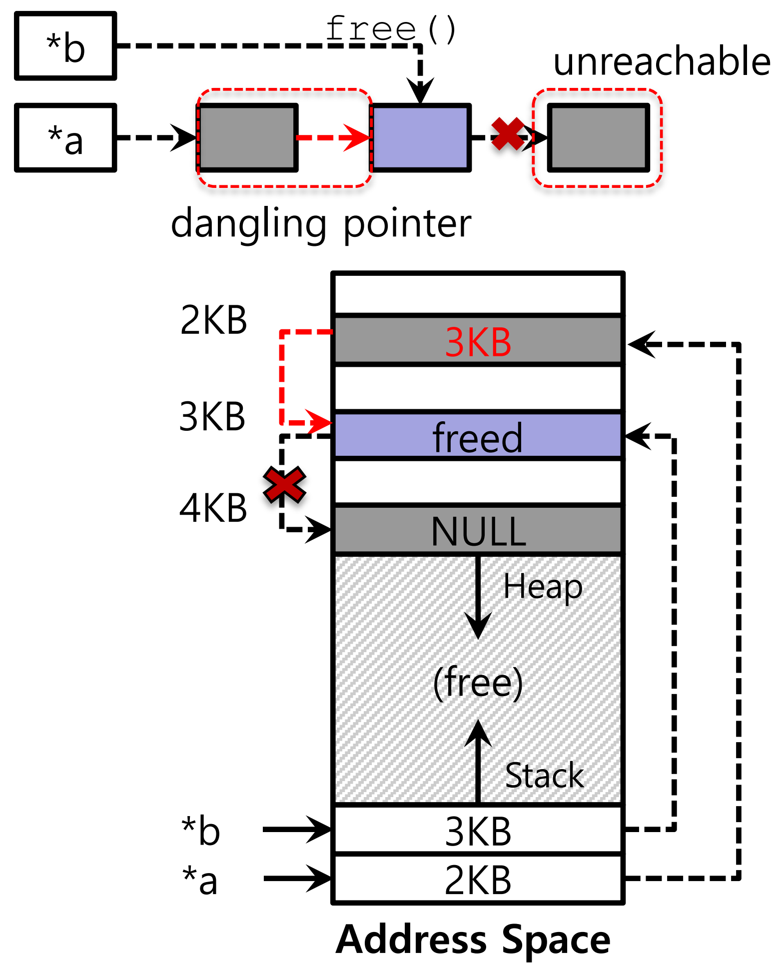
- Page Replacement Algorithms
  - The Optimal Algorithm
  - FIFO Algorithm
  - LRU Algorithm
  - Clock Algorithm
- Prefetching & Thrashing





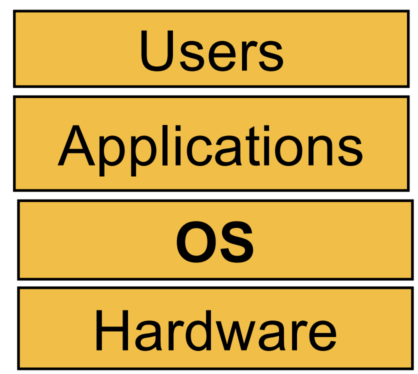
Recap

- Memory APIs
  - malloc()/free()
- Misuse of memory APIs has led to various bugs in practice
  - Forget to allocate/de-allocate/initialize
  - Dangling pointer
  - Double free
  - …



Recap

- OS
  - a resource manager
  - a control program
  - an extended/virtualized machine with abstraction
- Common OS Abstractions for HW
  - CPU
    - process and/or thread
  - Memory
    - address space
  - Disks
    - files



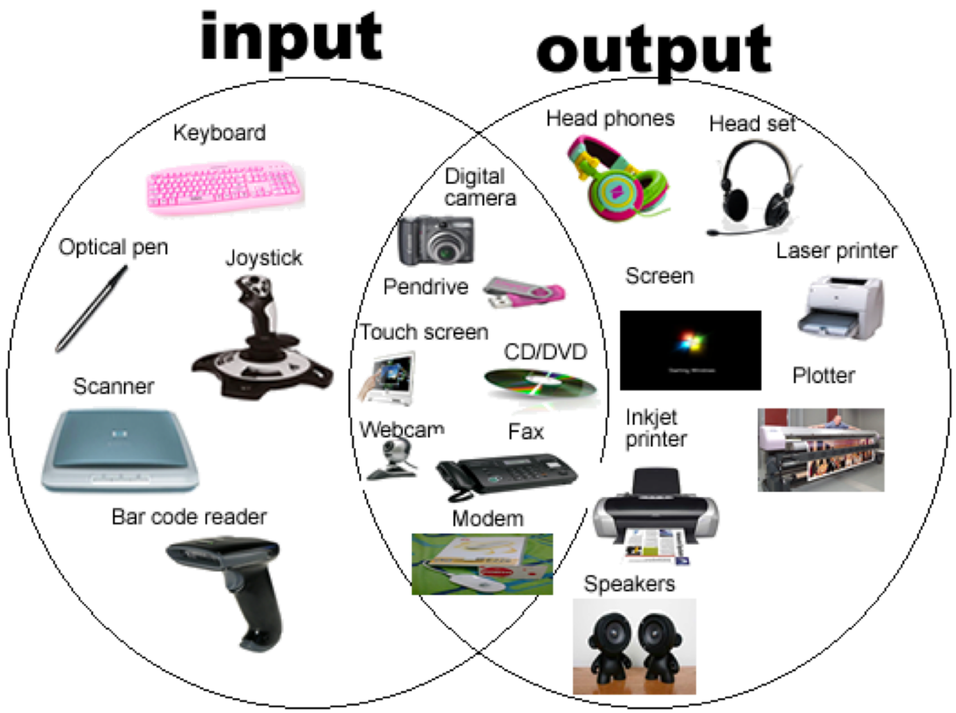
Agenda

- ~~Recap~~

- Introduction to I/O
  - I/O Device
  - I/O Bus
  - I/O Interrupt
  - DMA
  - Device Driver

Input/Output (I/O) Devices

- Critical for a computer to interact with users, the environment, and other systems
  - e.g., disks, USB, printer, keyboard



Input/Output (I/O) Devices

- Vary a lot in terms of speed
  - typically slower than memory

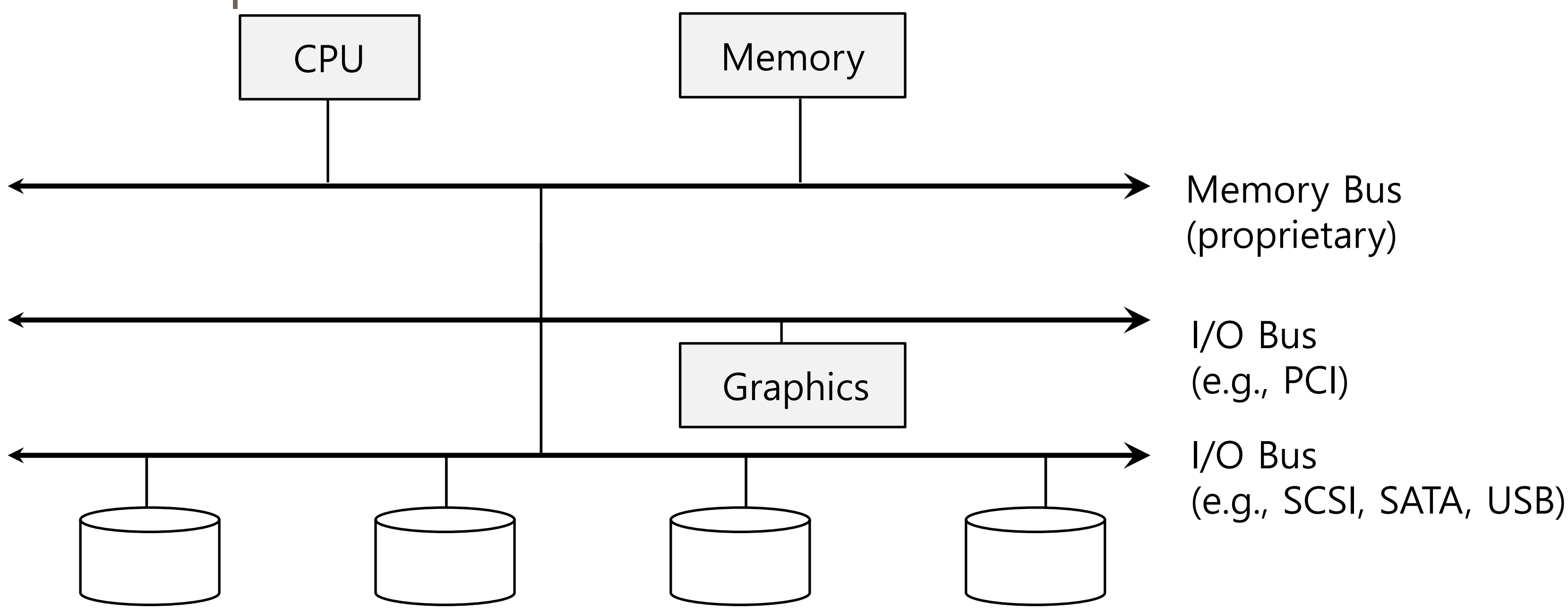
Device	Data rate
Keyboard	10 bytes/sec
Mouse	100 bytes/sec
56K modem	7 KB/sec
Scanner at 300 dpi	1 MB/sec
Digital camcorder	3.5 MB/sec
4x Blu-ray disc	18 MB/sec
802.11n Wireless	37.5 MB/sec
USB 2.0	60 MB/sec
FireWire 800	100 MB/sec
Gigabit Ethernet	125 MB/sec
SATA 3 disk drive	600 MB/sec
USB 3.0	625 MB/sec
SCSI Ultra 5 bus	640 MB/sec
Single-lane PCIe 3.0 bus	985 MB/sec
Thunderbolt 2 bus	2.5 GB/sec
SONET OC-768 network	5 GB/sec

Input/Output (I/O) Devices

- Two basic types from OS perspective
  - Block devices
    - stores information in fixed-size blocks, each one with its own address
    - All transfers are in units of one or more entire (consecutive) blocks
      - can read or write each block independently of all the other ones
    - E.g., Hard disks, CDROM, USB
  - Character devices
    - delivers or accepts a stream of characters (bytes), without any block structure
    - not addressable and does not have any seek operation
    - E.g., printer, network interface card (NIC)

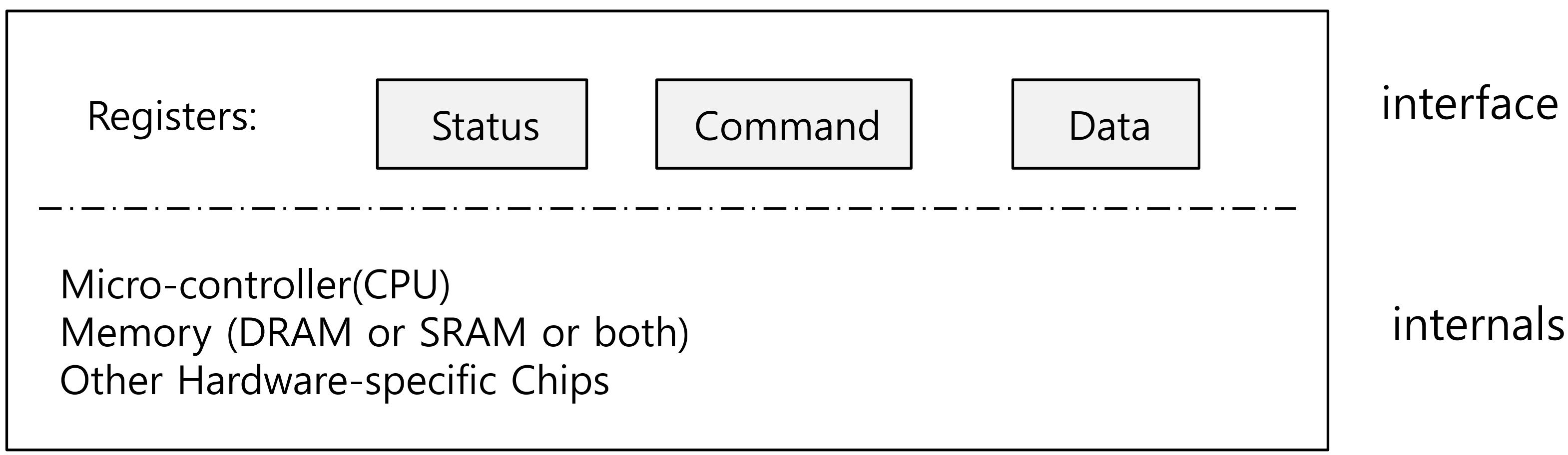
I/O Bus

- Buses
  - Data paths to enable information transfer between CPU(s), Memory, and I/O devices
  - I/O Bus
    - Data path that connects a CPU to an I/O device



Basic Structure of I/O Device

- Two parts
  - **Hardware interface** allows the system software to control its operation.
  - **Internals:** implementation details invisible to host



Basic Structure of I/O Device

- Hardware interface
  - **status register**
    - See the current status of the device
  - **command register**
    - Tell the device to perform a certain task
  - **data register**
    - Pass data to the device, or get data from the device

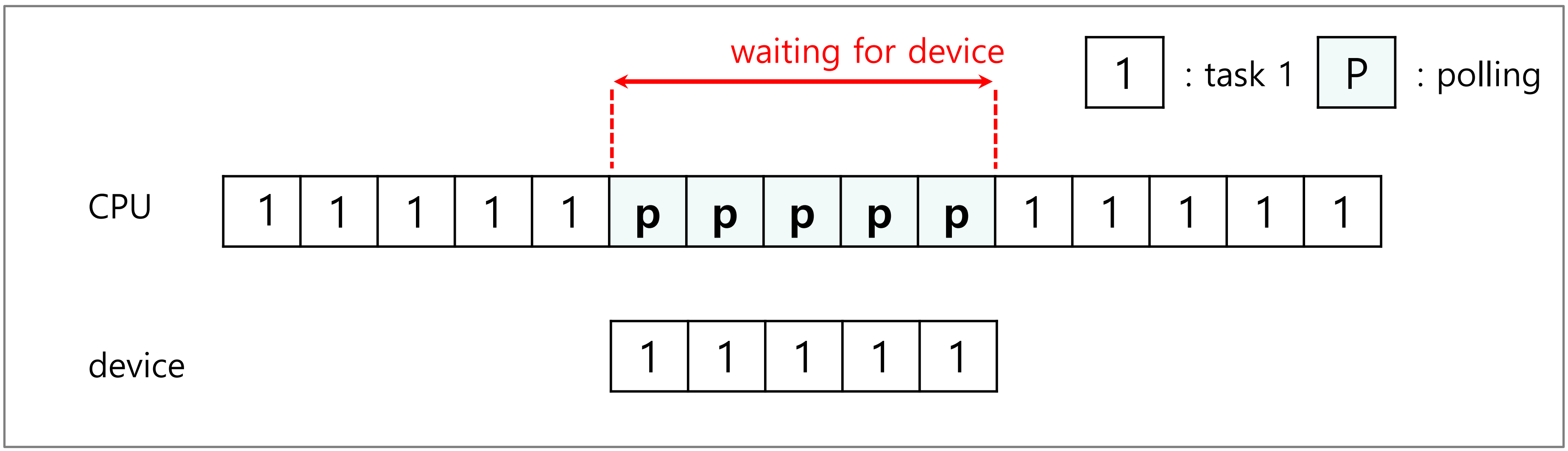
- By reading and writing above three registers, OS can control device behavior

Device Interaction

- Two basic ways
  - **programmed I/O**
    - OS reads/writes data from/to device registers through specific **I/O instructions**
      - e.g., `in` and `out` instructions on x86
  - **memory-mapped I/O**
    - Device registers available as if they were memory locations
    - The OS uses regular **memory instructions** to access the device
      - `load` (to read) or `store` (to write)

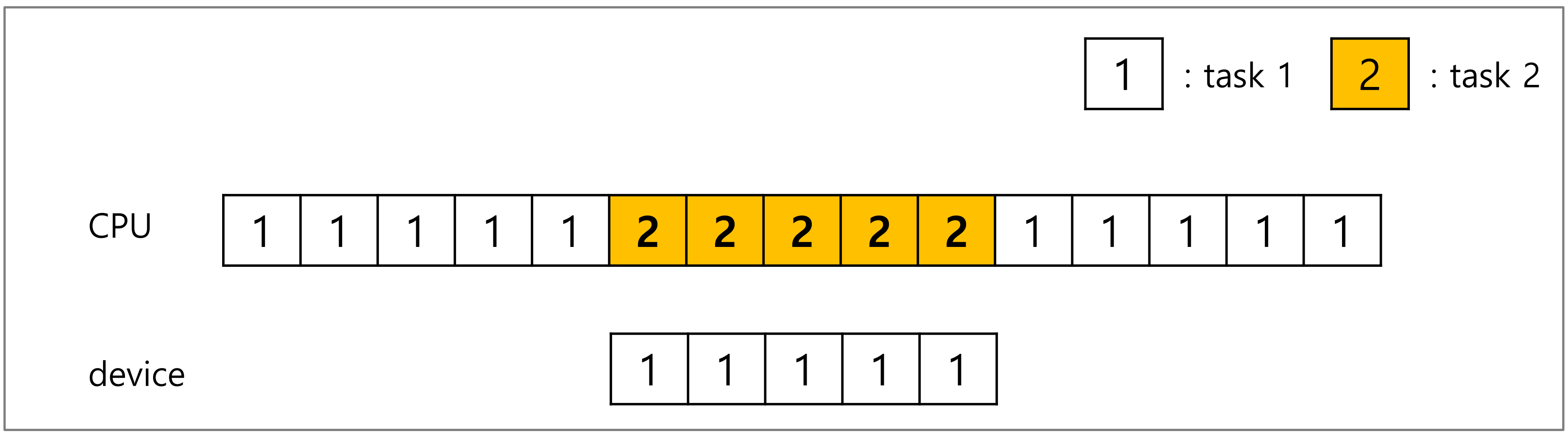
Polling

- OS waits until the device is ready by repeatedly reading the status register
  - simple
  - wastes CPU time for polling



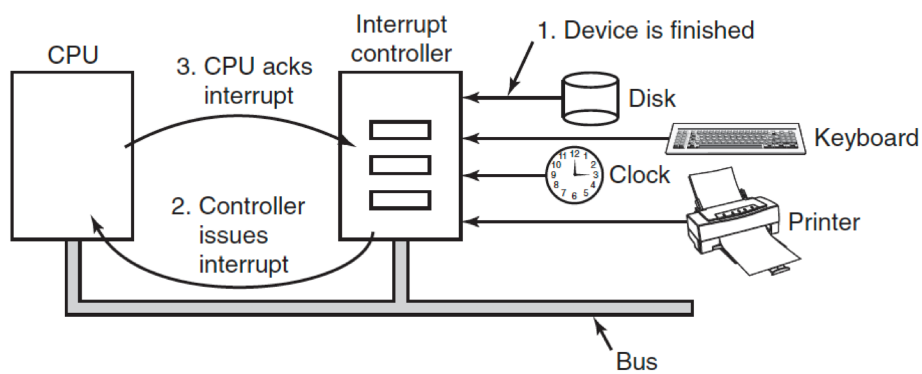
Interrupts

- Put the I/O request process to sleep and context switch to another process
  - When the device finished, wake up the process waiting for the I/O by **interrupt**
  - CPU and device can perform tasks in parallel
  - may be worse than polling (due to context switch)



Interrupts

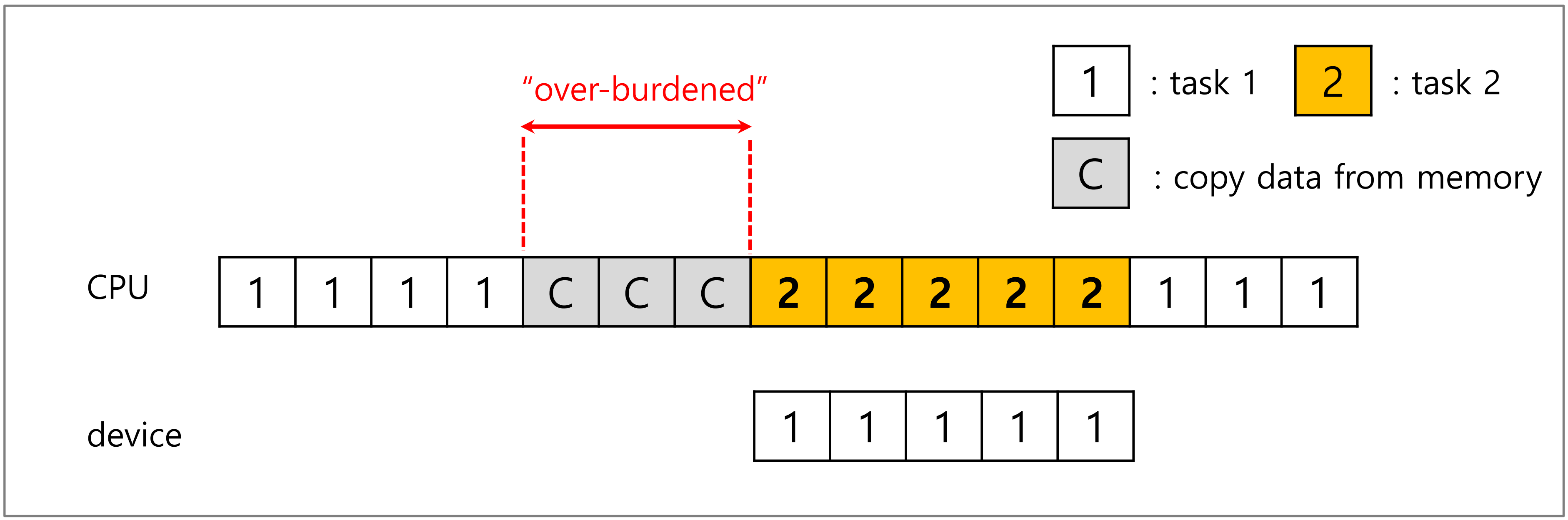
- A more detailed view
  - device asserts a signal on a bus line
  - interrupt controller detects the signal & decides what to do
    - e.g., puts a number on the address lines specifying which device wants attention and asserts a signal to interrupt the CPU
  - CPU detects the signal and jump to a service routine
    - use the number on the address lines as an index into a table called **interrupt vector** to fetch a new PC



    - need to save some context before jumping to the service routine

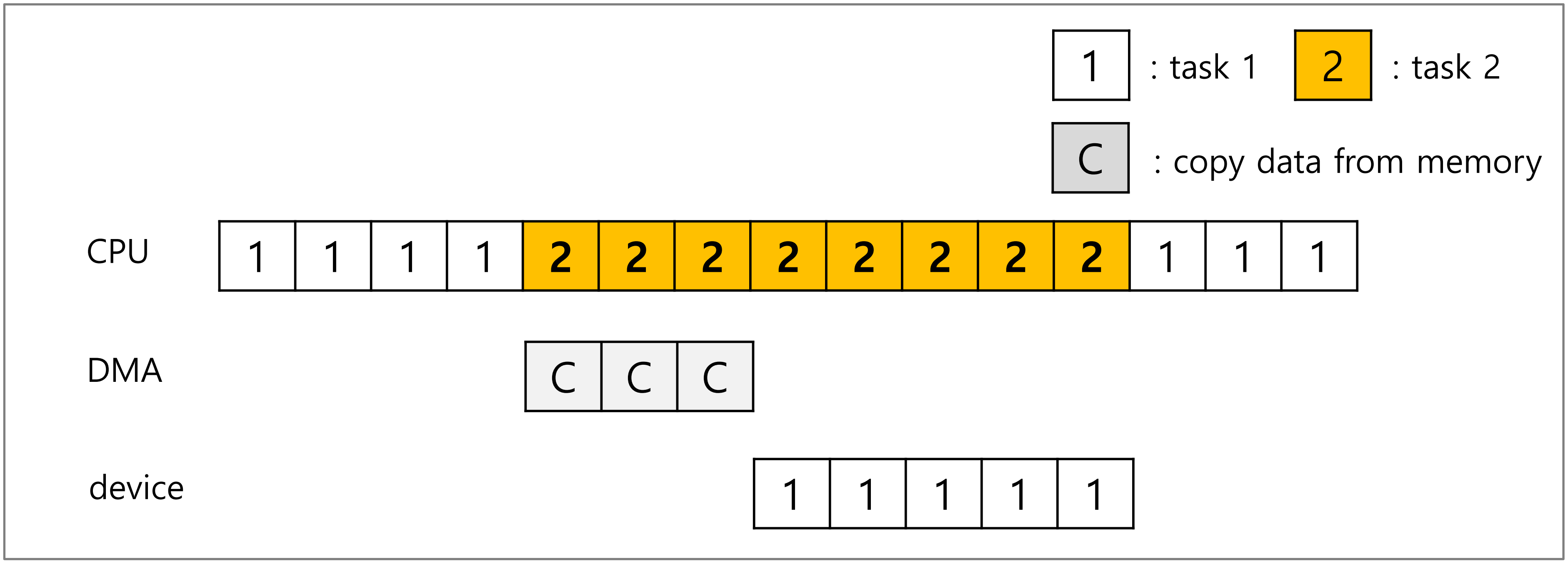
DMA (Direct Memory Access)

- Transfer a large chunk of data between memory & device may consume many CPU cycles



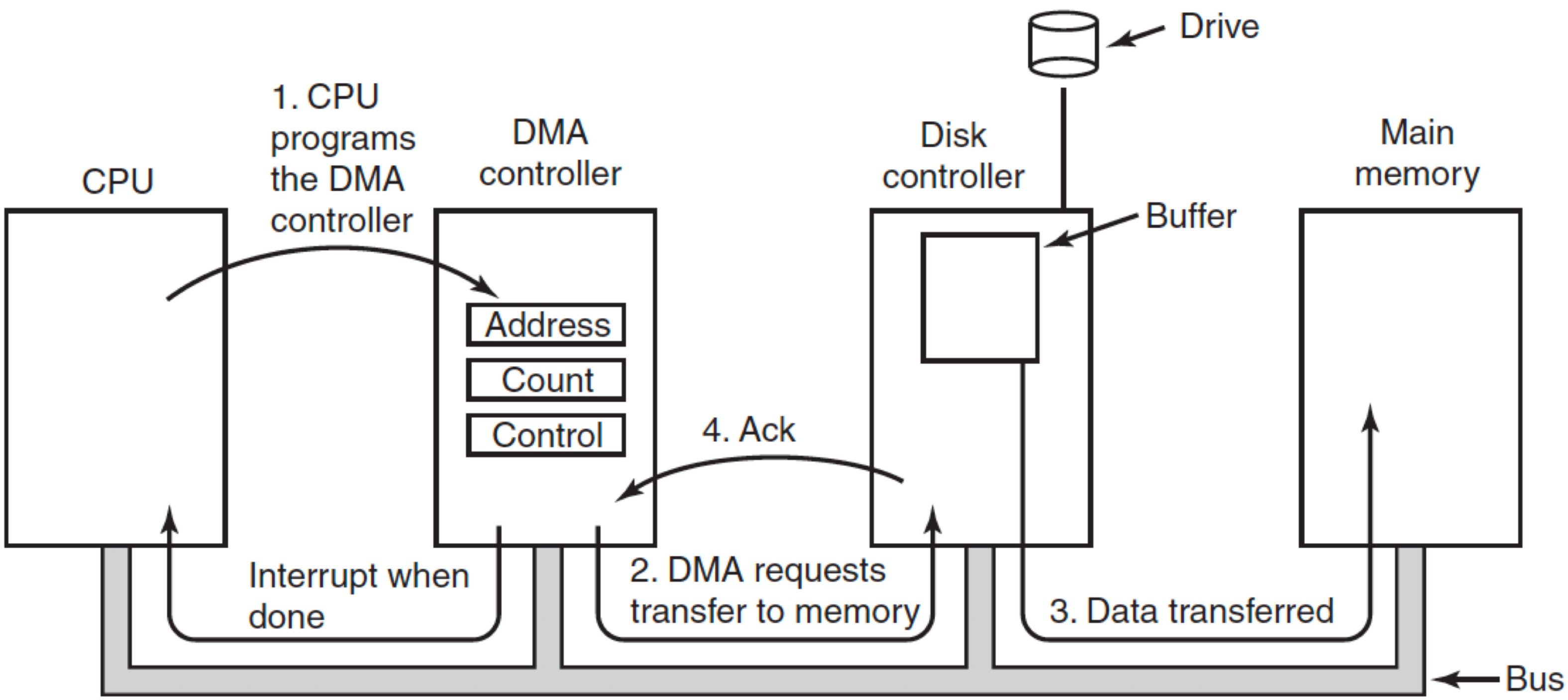
DMA (Direct Memory Access)

- Use a DMA controller to perform the data transfer
  - CPU initiates the transfer
  - DMA performs the transfer (CPU performs other task)



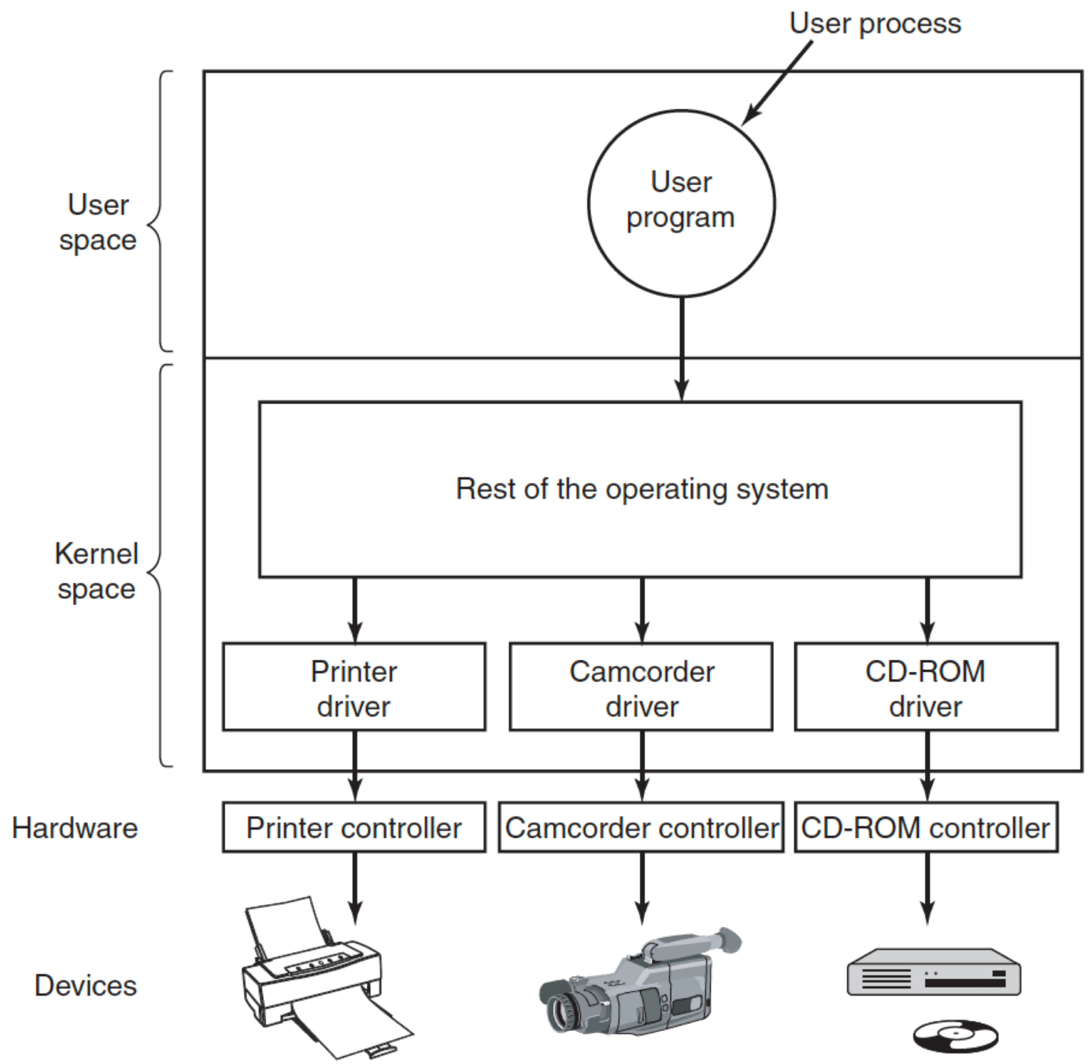
DMA (Direct Memory Access)

- A more detailed DMA operation



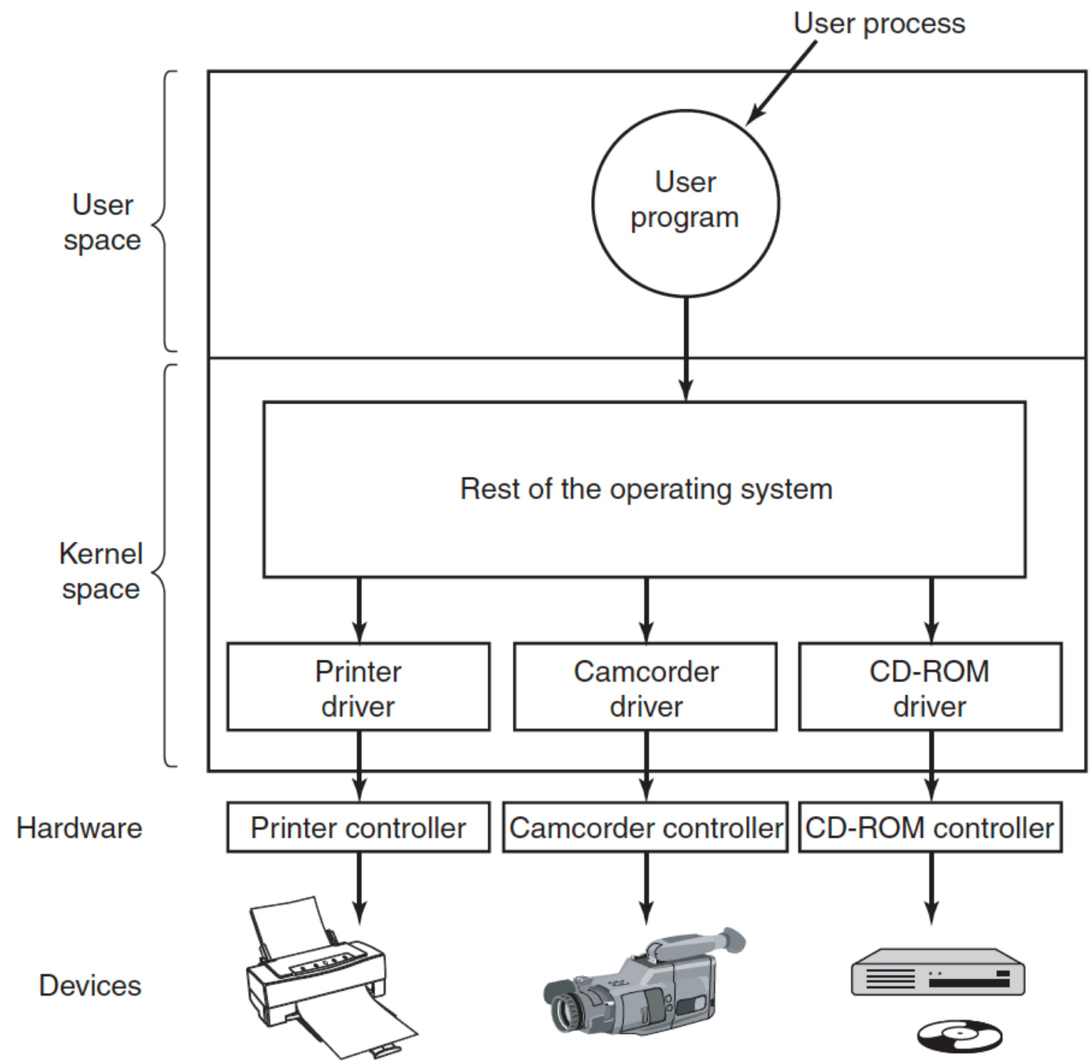
Device Drivers

- The software module in the OS that directly talk with device
  - device-specific
    - device manufacturers usually need to provide the corresponding drivers
  - can be installed/ removed separately
    - kernel module in Linux



Device Drivers

- Over 70% of OS code is in device drivers
  - primary contributor to kernel bugs/ crashes



Agenda

- ~~Recap~~
- ~~Introduction to I/O~~
    - ~~I/O Device~~
    - ~~I/O Bus~~
    - ~~I/O Interrupt~~
    - ~~DMA~~
    - ~~Device Driver~~

Questions?

*acknowledgement: slides include content from “Modern Operating Systems” by A. Tanenbaum, “Operating Systems Concepts” by A. Silberschatz etc., “Operating Systems: Three Easy Pieces” by R. Arpaci-Dusseau etc., and anonymous pictures from internet.