ECE437/CS481

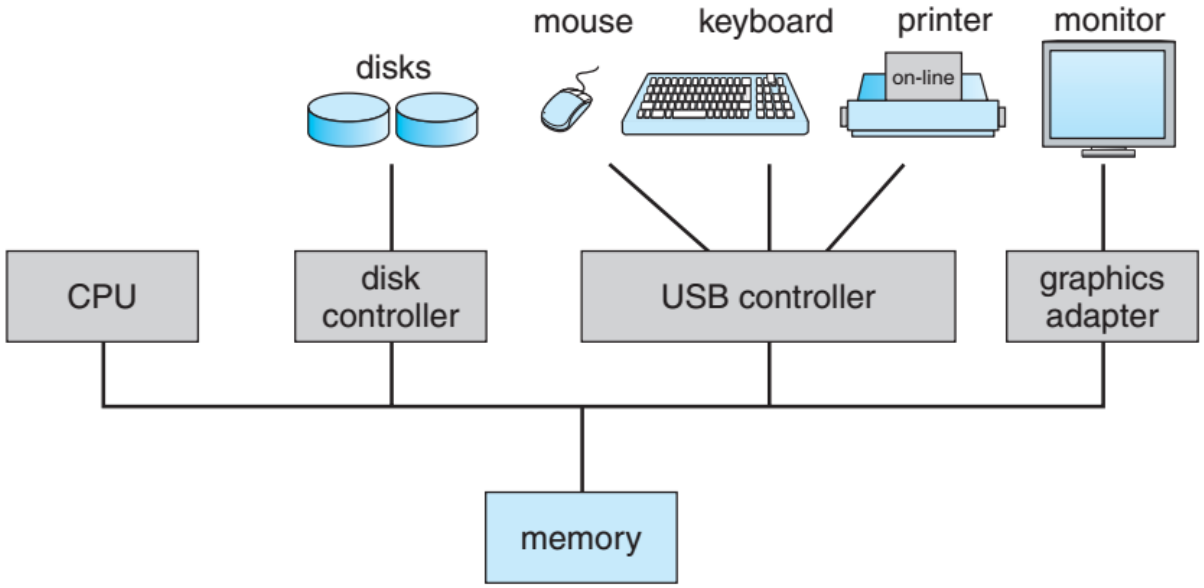
INTRODUCTION TO OS
COMPUTER ARCHITECTURE SUPPORT

Xiang Sun

The University of New Mexico

Computer System & OS

- CPU & I/O devices could execute concurrently--Asynchronous I/O
  - Device controller is in charge of a particular device type
    - ✓ may be able to handle multiple devices
    - ✓ act as a intermediary between the device and the OS
    - ✓ informs CPU about I/O completion by issuing interrupt
  - Differences between device controller and device driver?



I/O Devices & OS

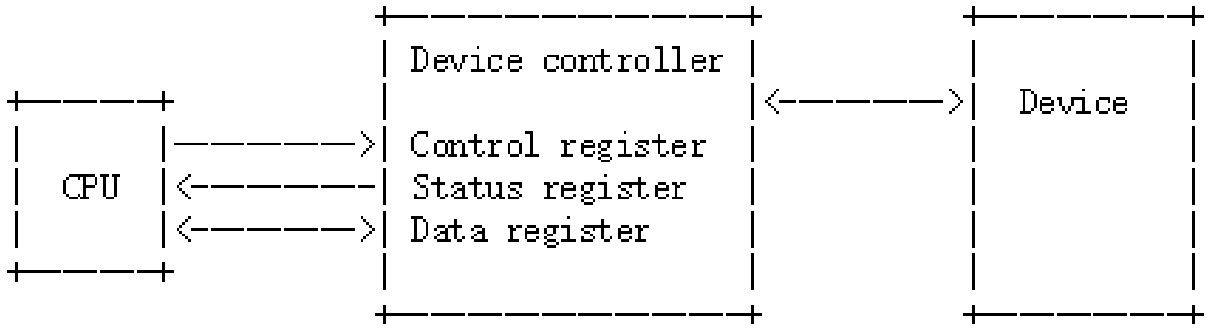
❑ Different types of I/O devices

- Character device
  - ✓ sending and receiving single characters
  - ✓ typical devices: mouse, keyboard, printer
- Block device
  - ✓ sending and receiving entire blocks of data
  - ✓ typical devices: disk drives, flash memory
- Network devices
  - ✓ sending and receiving packets
  - ✓ typical devices: network card

I/O Devices & OS

Three types of registers in devices controller

- **Status registers**
  - ✓ provide status information (e.g., busy, ready) to the CPU about I/O devices
  - ✓ read-only, i.e. the CPU can only read their bits, and cannot change them.



- **Configuration and control registers**
  - ✓ enable CPU to configure and control the device
  - ✓ bits in configuration registers may be write-only (i.e., CPU can change them, but not read them back)
  - ✓ bits in control registers can be both read and written

- **Data registers**
  - ✓ read data from or send data to the I/O device

I/O Devices & OS

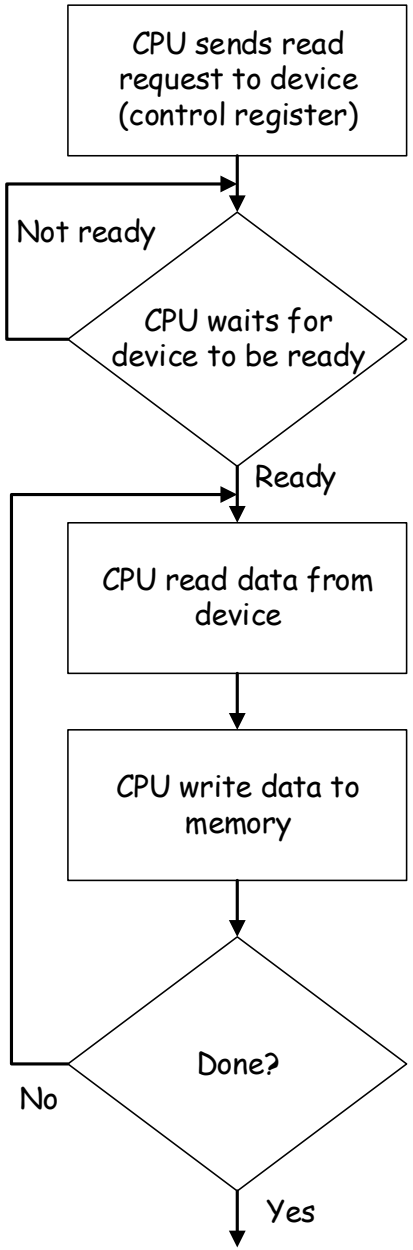
- Three ways of managing data transfers between IO devices and memory
  - Programmed I/O (PIO)
  - Interrupt-driven I/O
  - Direct memory access (DMA)

I/O Devices & OS

- Programmed I/O

  ---Assuming that the CPU tries to transfer data from hard disk to the memory

  1) The CPU issues a request to the control register of the I/O device (i.e., hard disk)
  2) The CPU waits the I/O device to become ready (e.g., to move the disk head to the right place)
     - The CPU continuously check the status register of the I/O device to see if it is ready-----polling
     - The CPU cannot do anything but wait
  3) The CPU reads data from the I/O device
  4) The CPU writes data to the memory
  5) Step 3&4 continuous until all the data are transferred to the memory

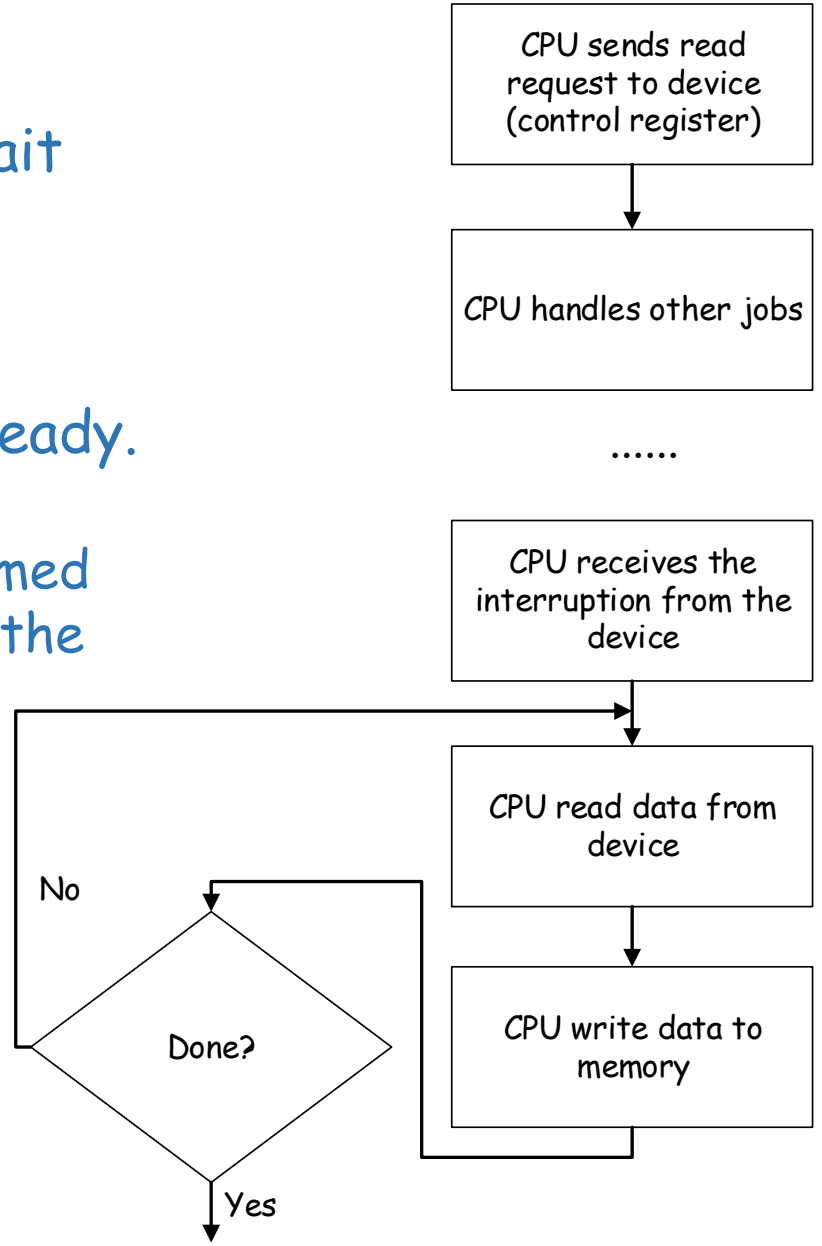


I/O Devices & OS

- Interrupt-driven I/O
  ---Solving the problem of the processor having to wait for a slow device.

  1) Instead of waiting, the CPU continues to handle other jobs. The device interrupts the CPU it is ready.

  2) The data transfer are still the same as with programmed I/O (i.e., Step 3-5) --- The CPU still has to involve in the data transferring.



I/O Devices & OS

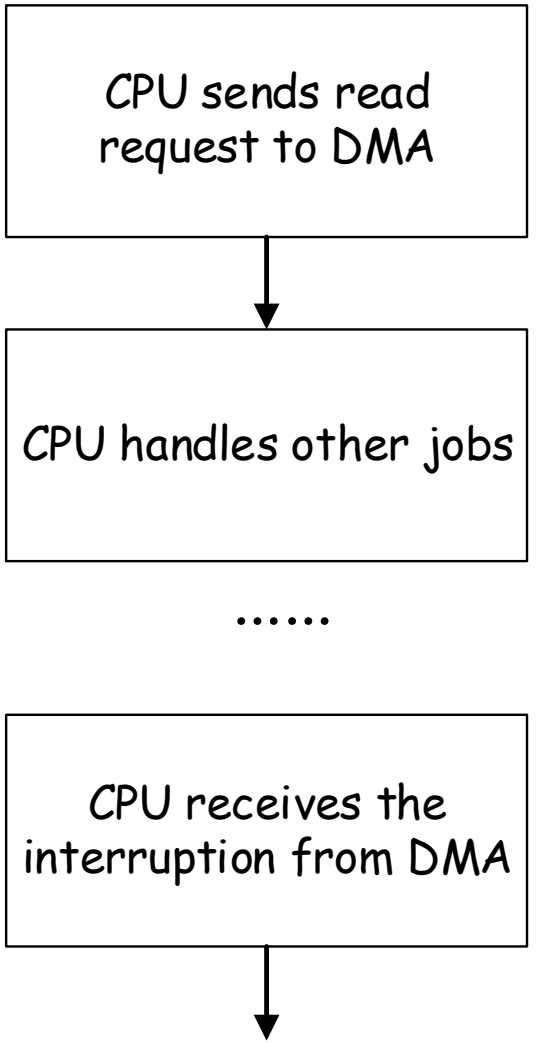
- Direct memory access (DMA)
  ---benefit the CPU for bulk data movement

  - DMA is a simple processor/controller which does most of the functions (related to data transferring) that the CPU would otherwise have to handle.

  1) The CPU asks the DMA controller to transfer data between a device and memory. After that, the CPU continues to handle other jobs.

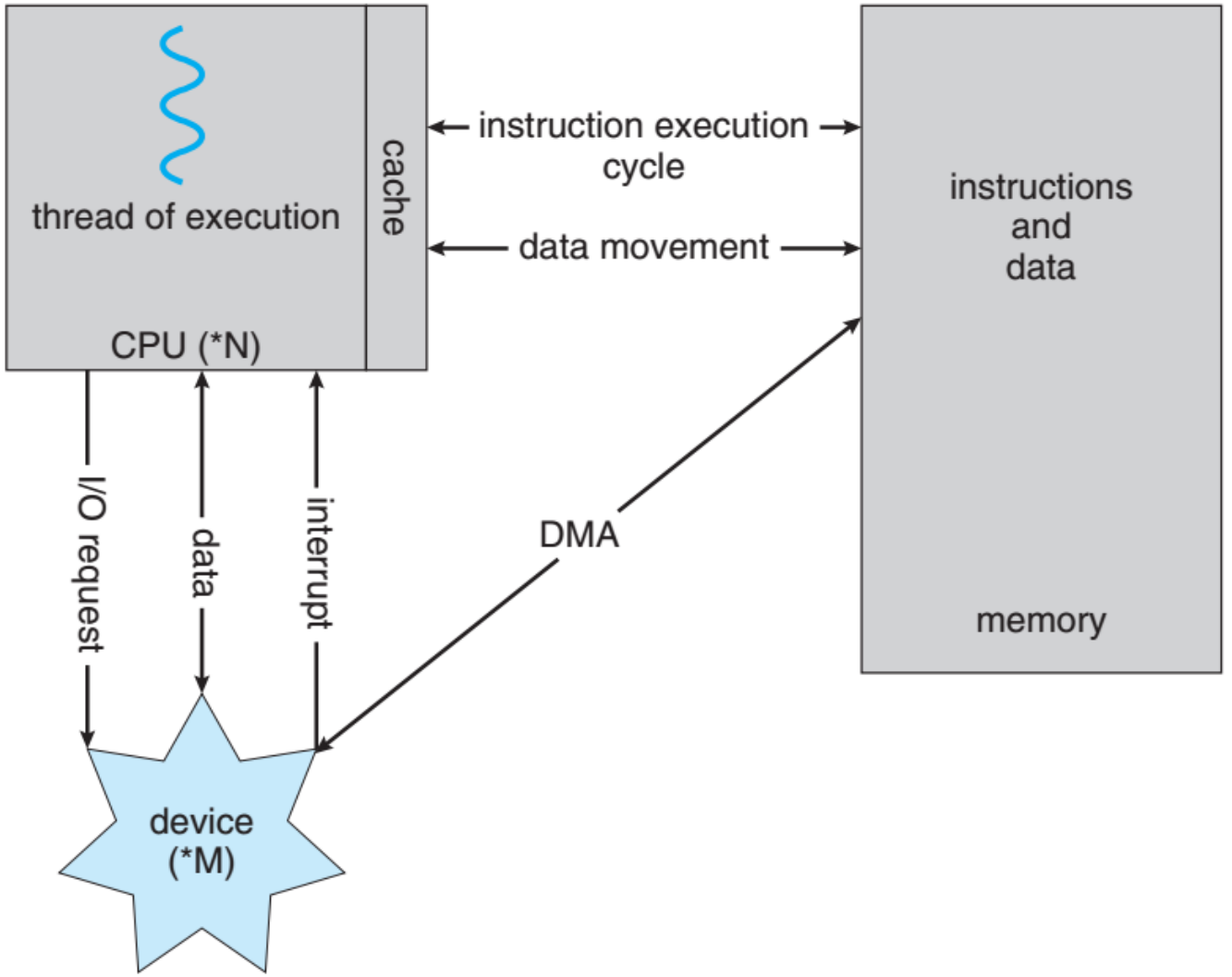
  2) The DMA controller issues requests to the I/O device, waits, and manages the data transfer between the I/O device and memory

  3) Once finished, the DMA controller interrupts the CPU.



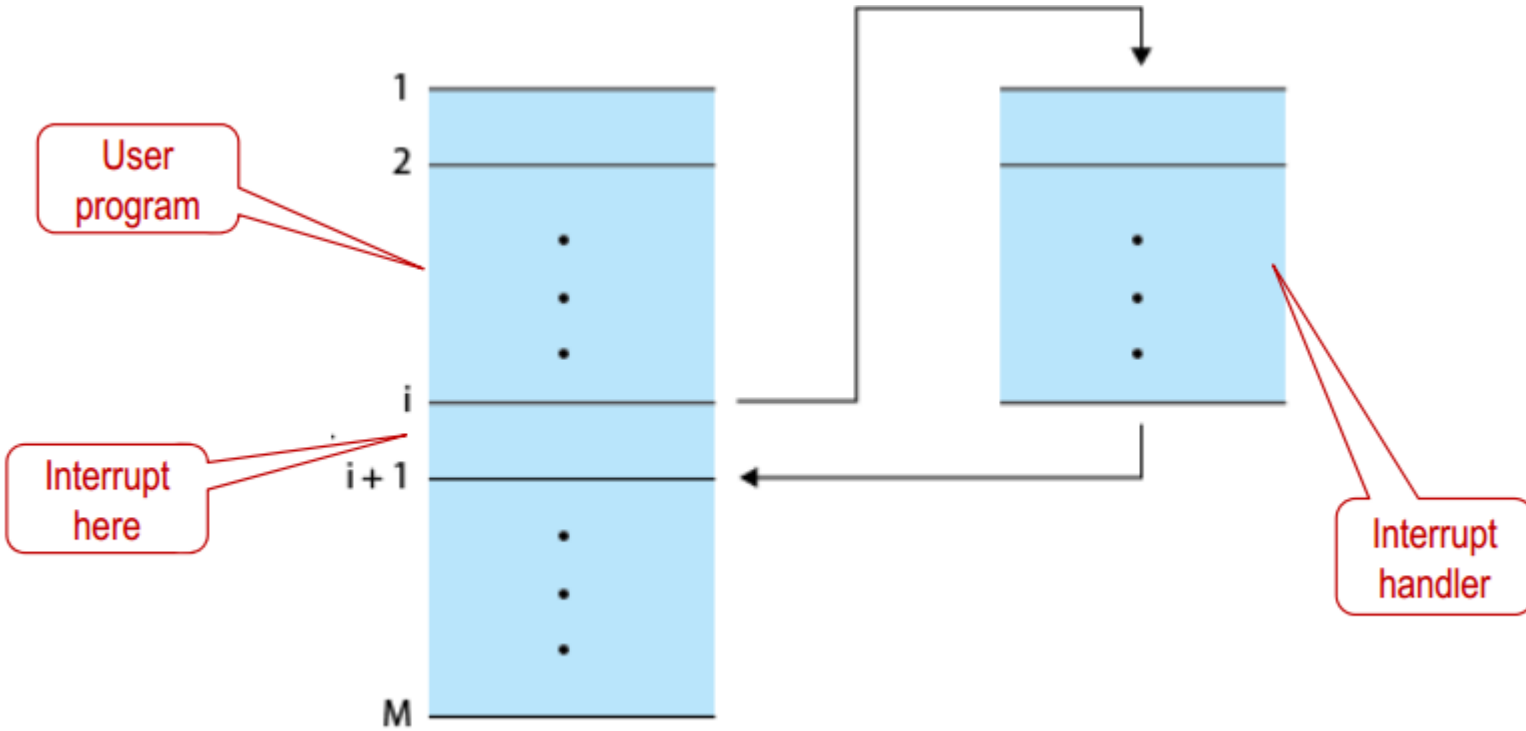
I/O Devices & OS

- Interplay of CPU, memory, and I/O devices



Interrupts in OS

- Interrupt – OS is interrupt driven!
  - Suspend the normal sequence of execution
  - Improve CPU utilization & application responses
  - OS must save the environment of the interrupted instruction by storing:
    - ✓ Status registers or PSW (Program Status Word)---status of CPU.
    - ✓ PC (Program Counter)---the address of the next instruction to be fetch for execution.



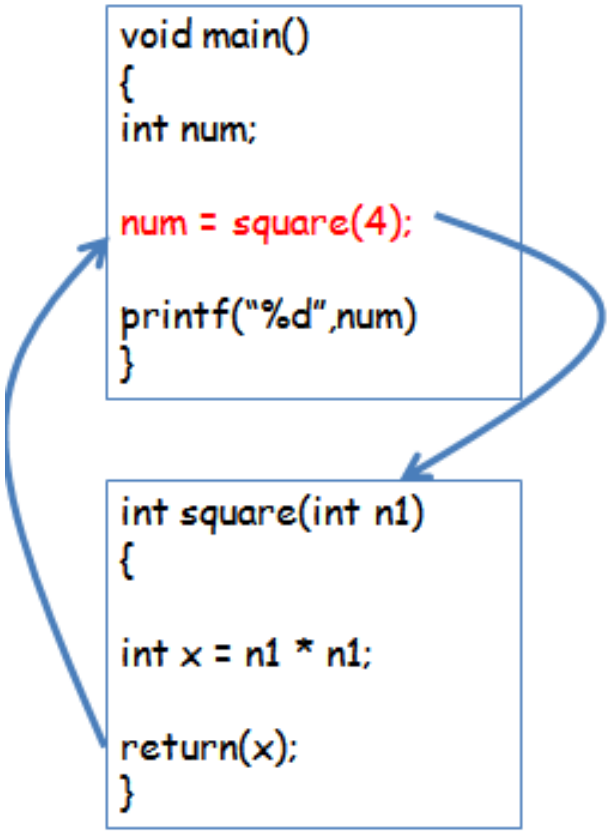
Interrupts in OS

- Interrupt
  - Interrupt transfers to Interrupt Service Routine (ISR)—Interrupt handler
    - A software component in OS.
    - ✓ Interrupt vector == the addresses of all the service routines.
    - ✓ Service routine == what action should be taken for interrupt.
  - Interrupts generated by
    - ✓ Hardware: I/O devices (e.g., click of mice), A timer within CPU chip.
    - ✓ Software: incurring errors (e.g., divide by zero, arithmetic overflow), requests via a system call.
      Question: System call = Software interrupts?
    - ✓ A software-generated interrupt is also called a trap.

Interrupts in OS

- Interrupt Service Routine (ISR) and function call
  - Function call: the instructions are executed from a new address, e.g., execute a function/method.
  - Major differences between ISR and function call

ISR	Function Call
Random	Expected or User Programmed
Execution based on priority	Execution based on sequential order
Cannot have arguments and return values	Can have arguments and return values

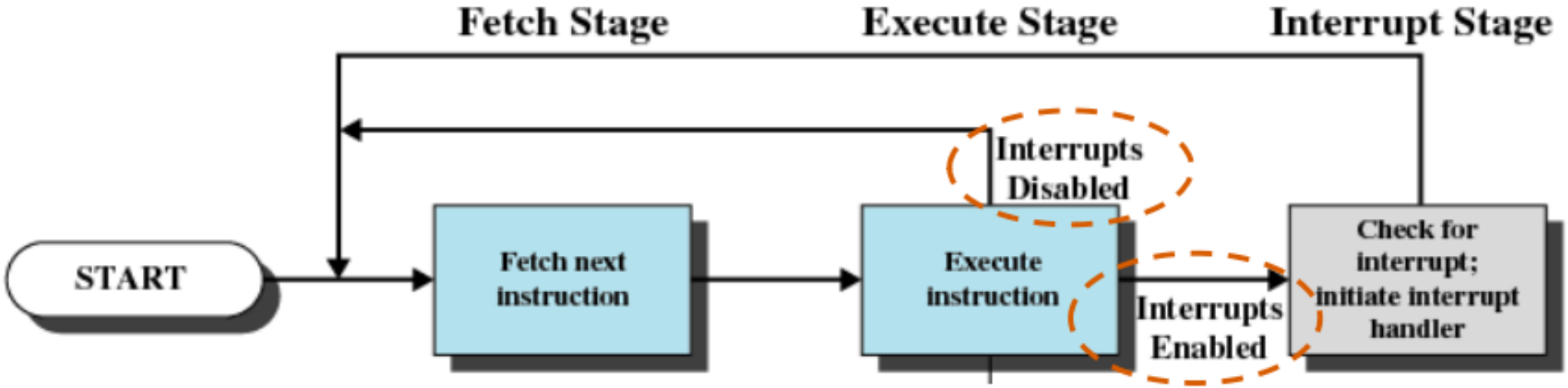


Interrupts in OS

Interrupt

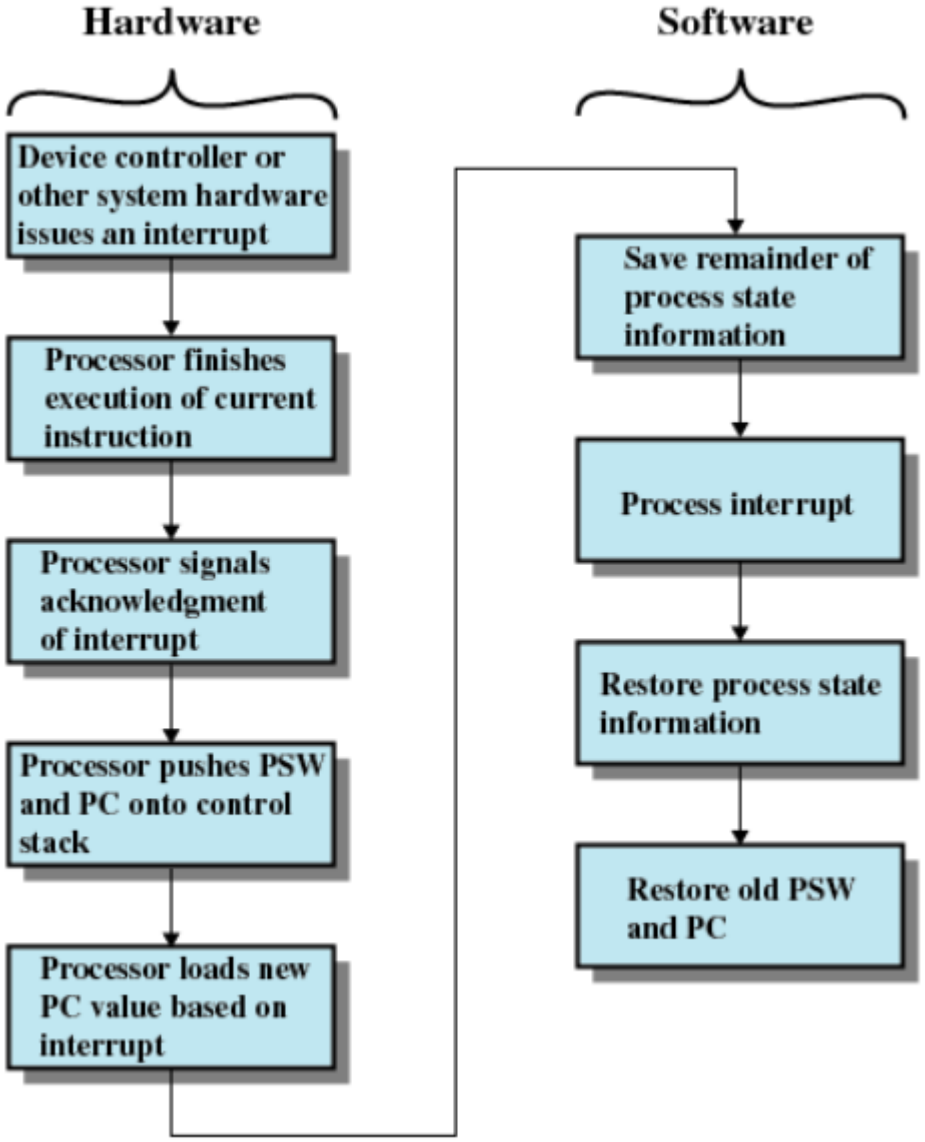
- To prevent a lost interrupt, all incoming interrupts are disabled while an interrupt is being processed. --- Non-preemption

- Instruction cycle for Interrupts:.



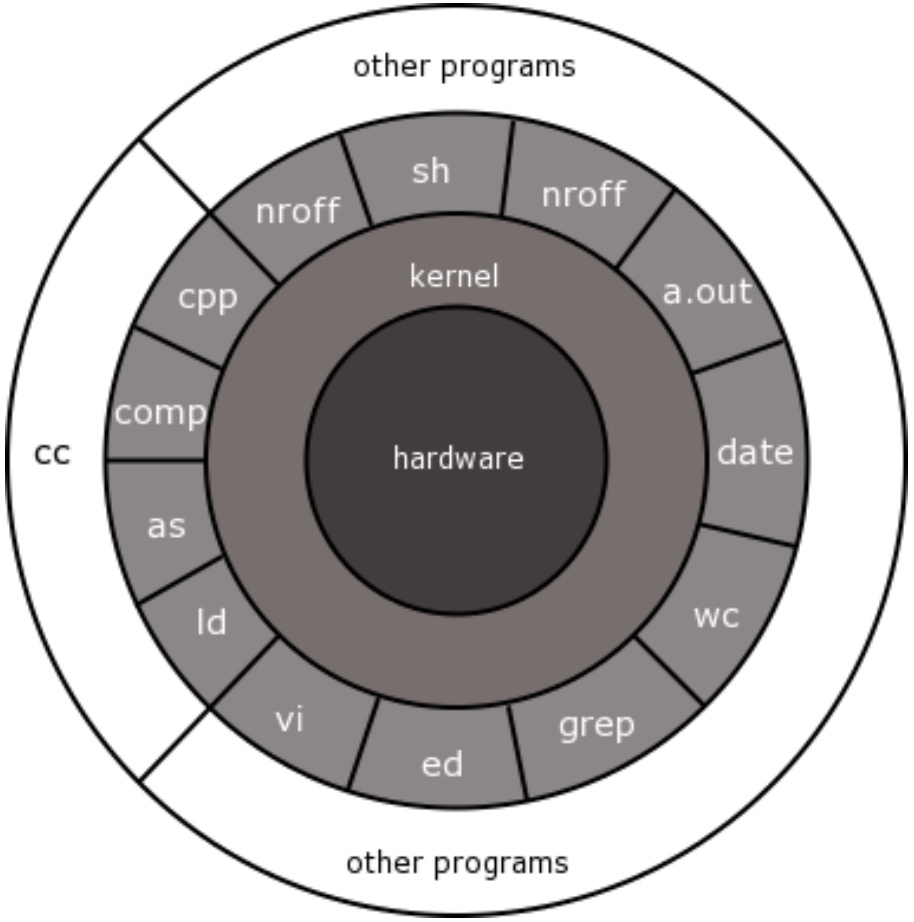
Interrupts in OS

- Summary of handling interrupts (from hardware)



OS & Kernel

- Operating System
  - Kernel+ Utility Software
- Kernel--core component of the system
  - Controls execution of processes
  - Schedules processes onto CPU(s)
  - Manages memory
  - Manages file systems
  - Manages access to devices
  - ……

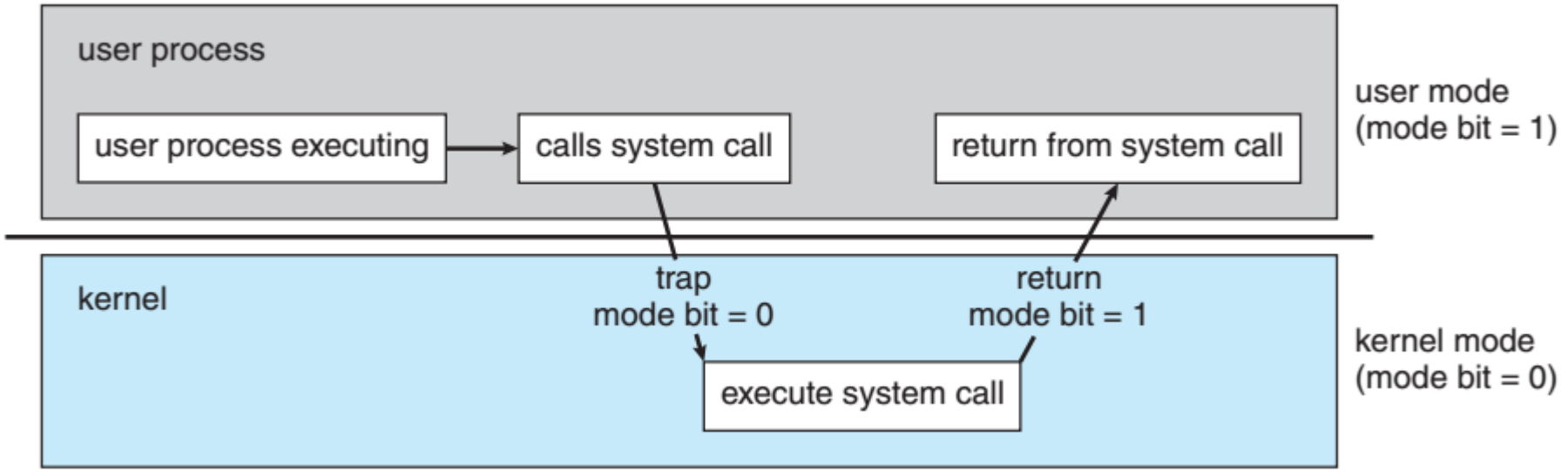


Kernel Mode VS. User Mode

- A processor has two modes: user mode and kernel mode
  - User mode (slave, restricted mode)
    - ✓ A task is executed on behalf of the user
    - ✓ High restriction (cannot access hardware)
    - ✓ Exception only crash single process
    - ✓ Only access its own virtual memory
  - Kernel mode (master, privileged, system, supervisor mode)
    - ✓ A task is executed on behalf of OS
    - ✓ Low restriction (can access hardware)
    - ✓ Exception may crash the entire OS
    - ✓ Can access the entire memory

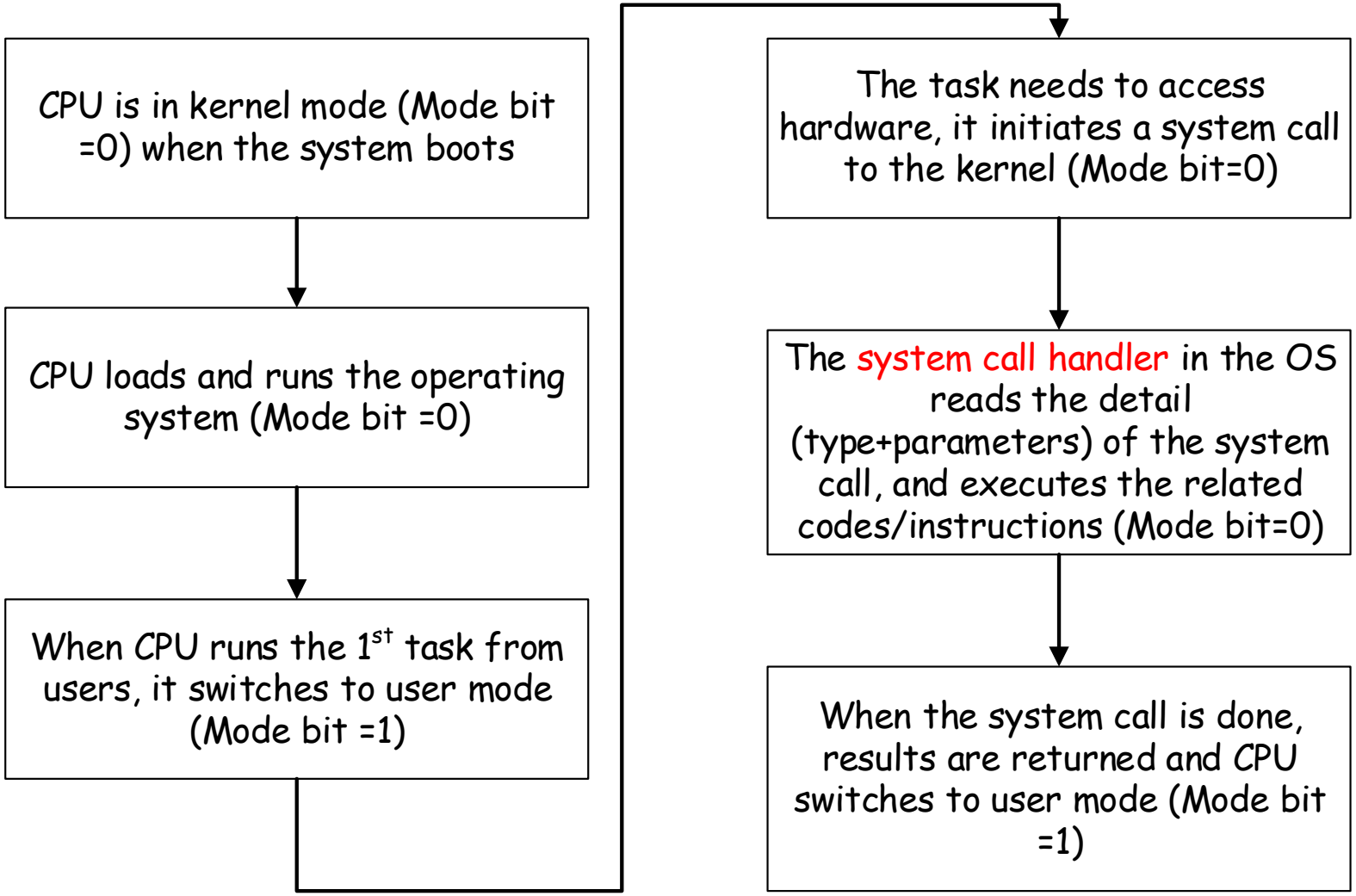
Kernel Mode VS. User Mode

- Switch between kernel mode and user mode
  - Mode bit is used to indicate if the processor is in kernel mode (0) or user mode (1)
    - ✓ Stored in a register (e.g., Program Status Word (PSW) register)
  - System calls are used to switch from kernel mode and user mode



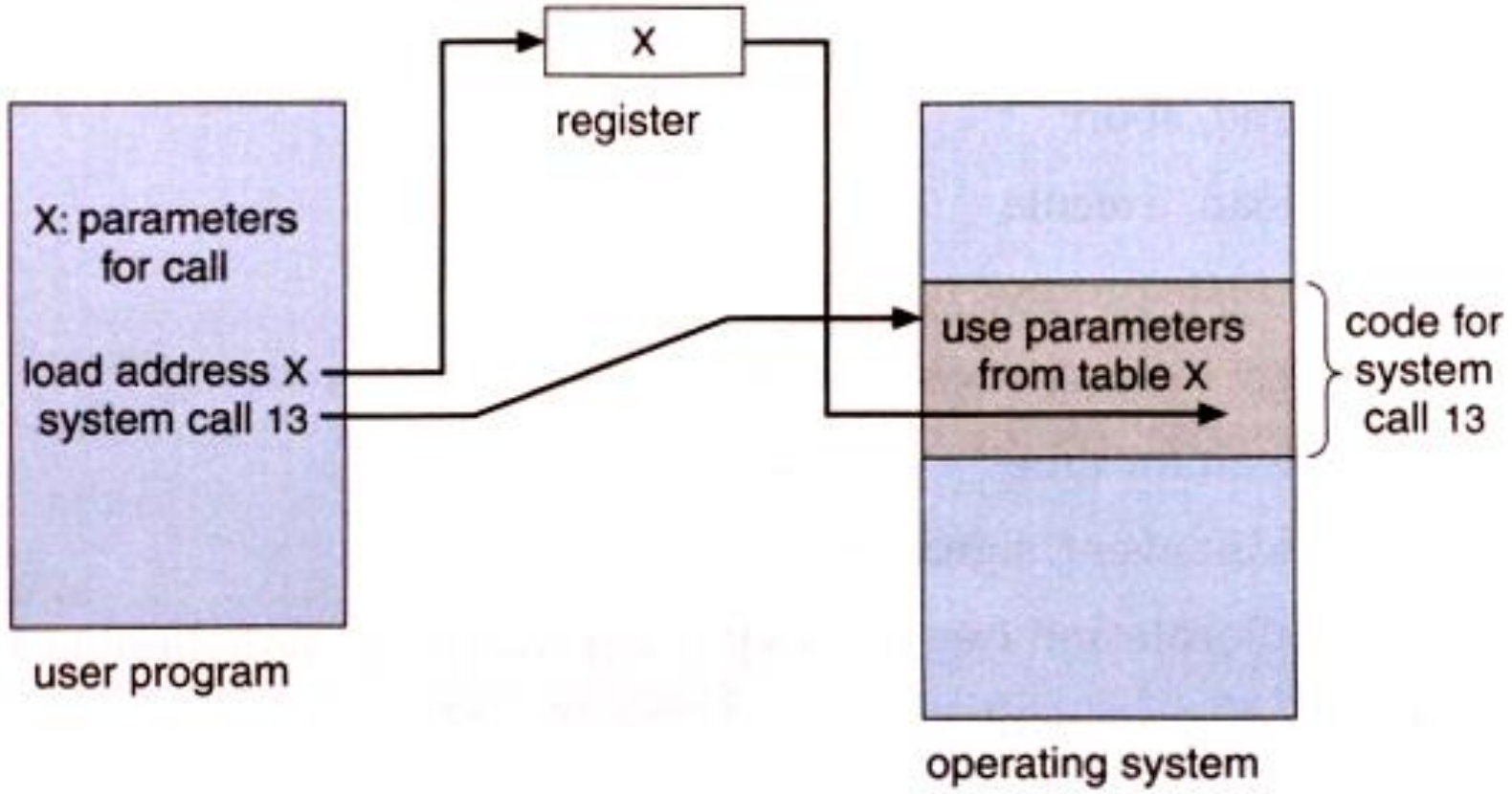
Kernel Mode VS. User Mode

- Switch between kernel mode and user mode



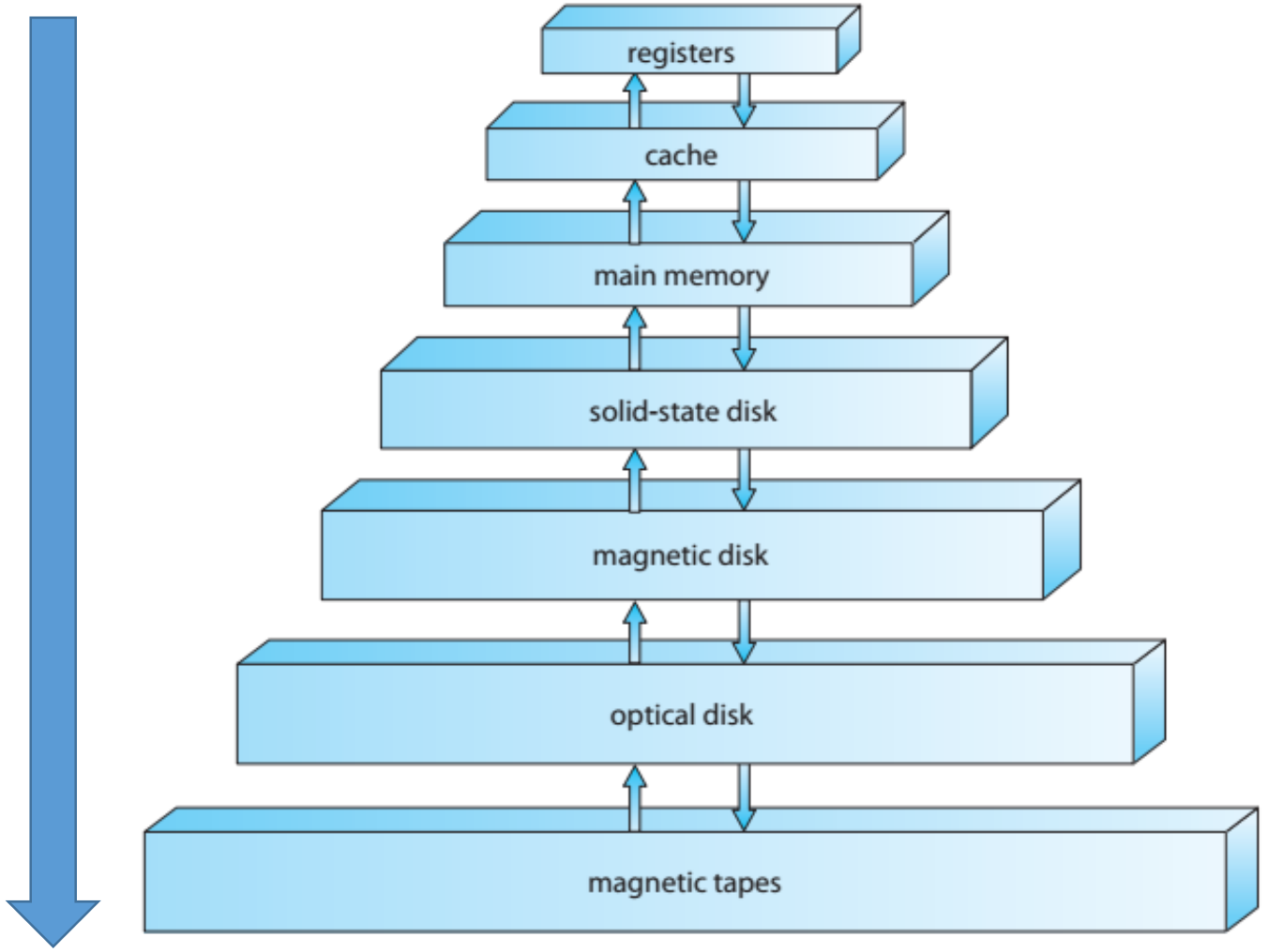
Kernel Mode VS. User Mode

- Switch between kernel mode and user mode
  - Parameters of the system call can be passed in registers.
  - If there are more parameters than registers, parameters can be stored in a block and the block address can be passed as a parameter to a register.



OS & Storage Devices

- Storage systems organized in hierarchy
  - Decreasing cost per bit
  - Increasing capacity
  - Increasing access time
  - Increase size of the transfer unit



OS & Storage Devices

❑ Performance of Various Levels of Storage

Level	1	2	3	4	5
Name	registers	cache	main memory	solid state disk	magnetic disk
Typical size	< 1 KB	< 16MB	< 64GB	< 1 TB	< 10 TB
Implementation technology	custom memory with multiple ports CMOS	on-chip or off-chip CMOS SRAM	CMOS SRAM	flash memory	magnetic disk
Access time (ns)	0.25 - 0.5	0.5 - 25	80 - 250	25,000 - 50,000	5,000,000
Bandwidth (MB/sec)	20,000 - 100,000	5,000 - 10,000	1,000 - 5,000	500	20 - 150
Managed by	compiler	hardware	operating system	operating system	operating system
Backed by	cache	main memory	disk	disk	disk or tape