Part I Overview

Chapter 1: Introduction

What is an Operating System?

- **A computer system can be roughly divided into the *hardware*, the *operating system*, the *application programs*, and the *users*.**
- **The *hardware* (*i.e.*, CPU, memory, I/O devices) provides the basic computing resources.**
- **The *application programs* (*e.g.*, web browsers, word processors, and compilers) are used to solve user problems.**
- **The *operating system* controls and coordinates the use of hardware among various application programs.**

User View

- **PC Users: an operating system is a program that provides an easy-to-use interface for using the hardware.**
- **Mainframe/Minicomputer Users: an operating system is a program that helps maximize the system resource utilization.**
- **Workstation Users: an operating system is a program designed to compromise between individual usability and resource utilization.**

System View

- ***A Resource Manager*: the operating system allocates and reclaims the system hardware resources to and from user programs.**
- ***A Control Program*: the operating system controls the execution of user programs to prevent errors and improper use of the computer.**
- **There is *no* universal definition of what is an operating system. A common definition is that the operating system is the one program running at all times on the computer (*i.e.*, the *kernel*). All other programs are application programs.**

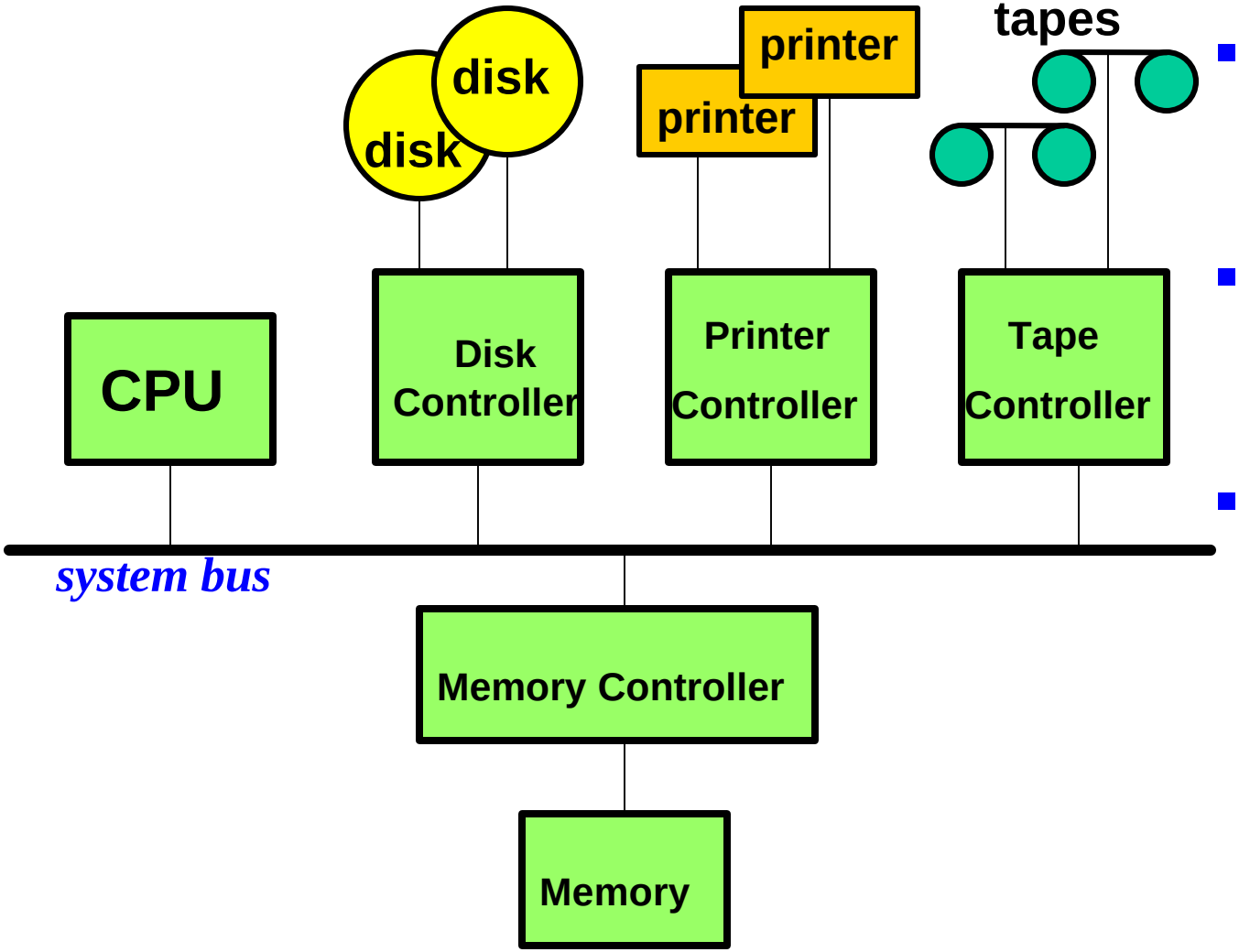
The Goals of an Operating System

- **Primary Goal: *efficient* operation of the computer system.**
- **In some systems, it is *easy of use*.**
- **In general, efficiency is far more important than easy of use.**

Computer-System Organization

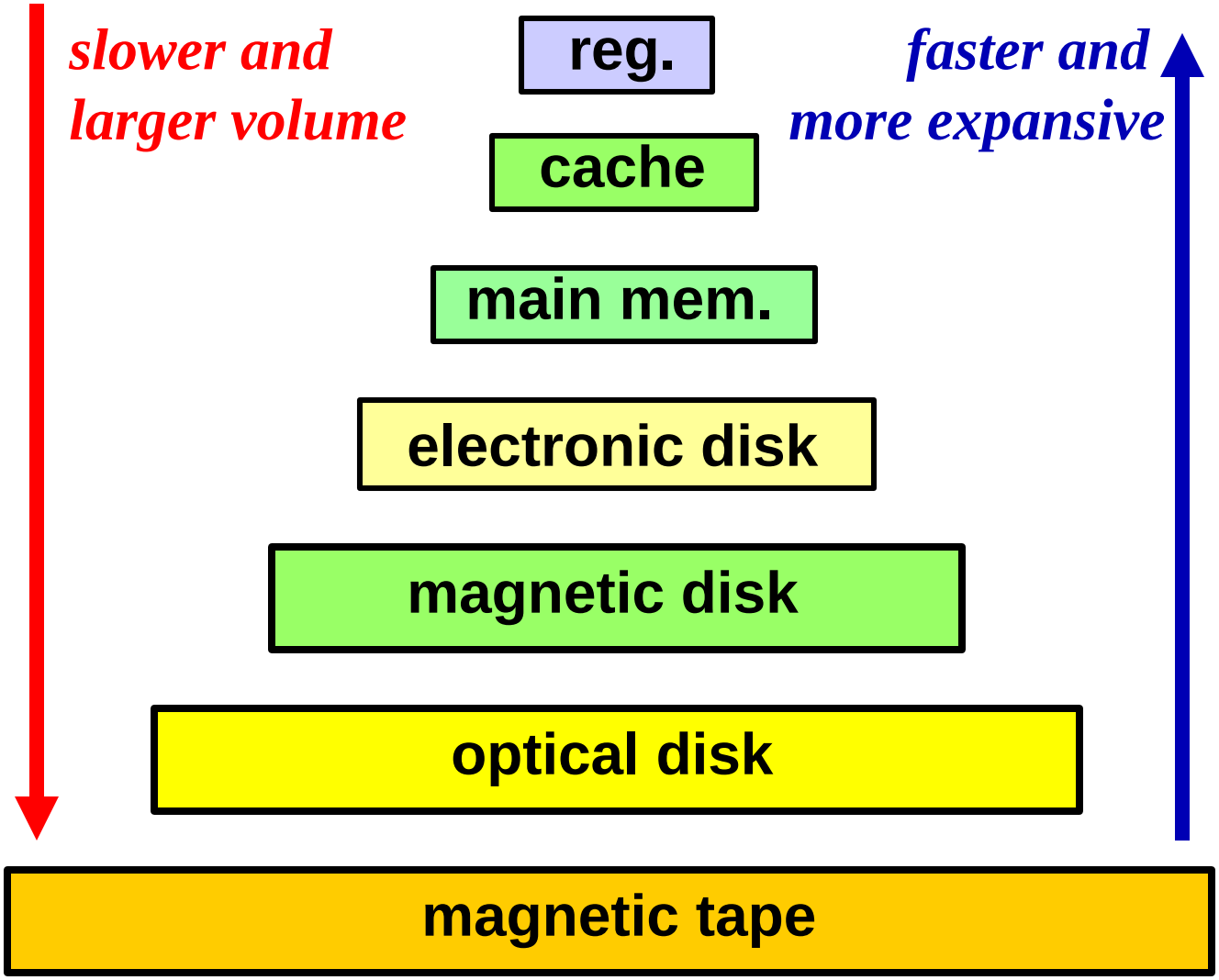
- **Computer-System Operation**
- **Storage Structure (von Neumann architecture):**
  - **CPU-Memory**
  - **Fetch-decode-execute-store**
- **I/O structure**
- **More on the next few slides**

Computer System Operation



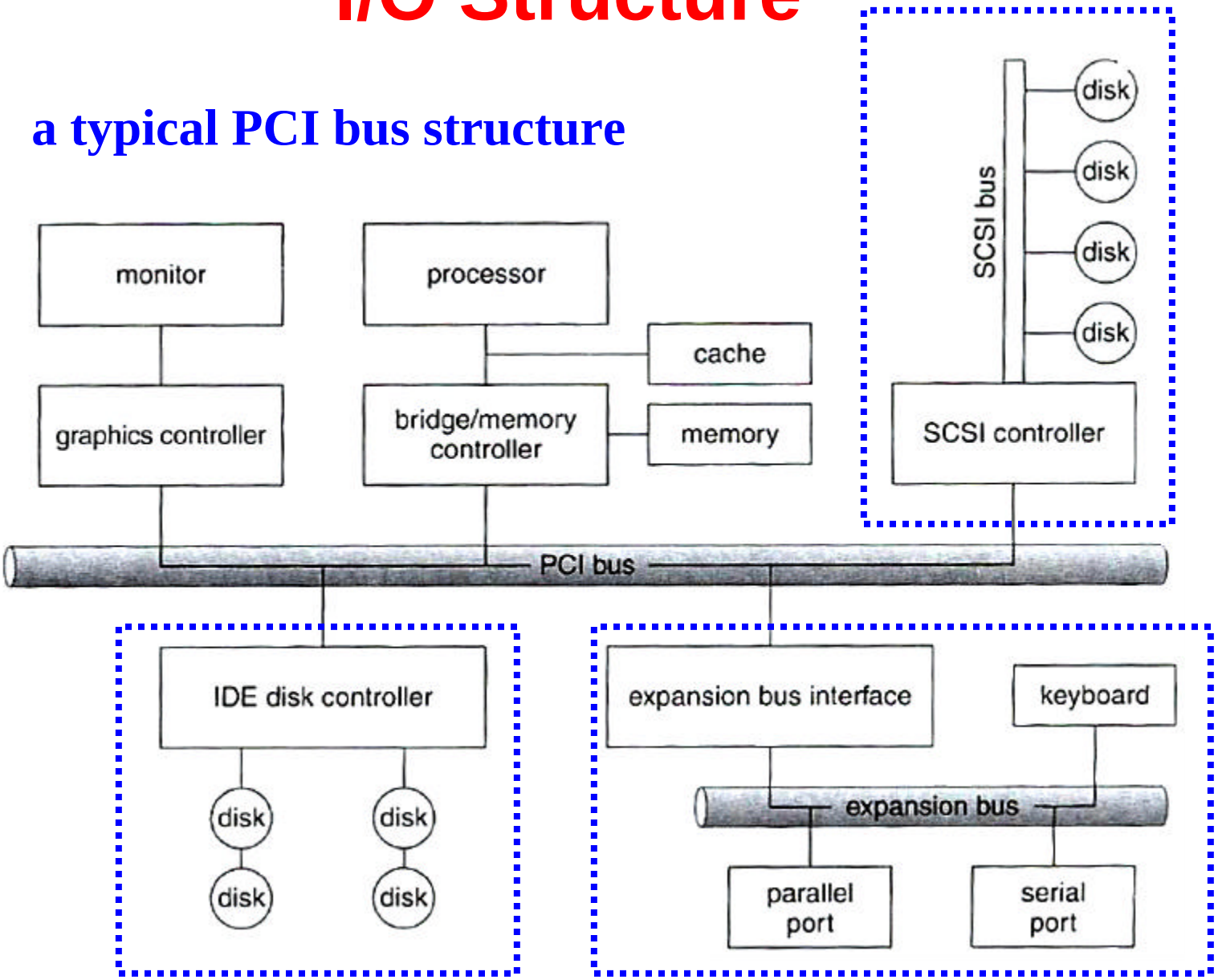
- How does the OS get into system?
- How do we request for services?
- How does the OS know something has happened?

Storage Hierarchy



I/O Structure

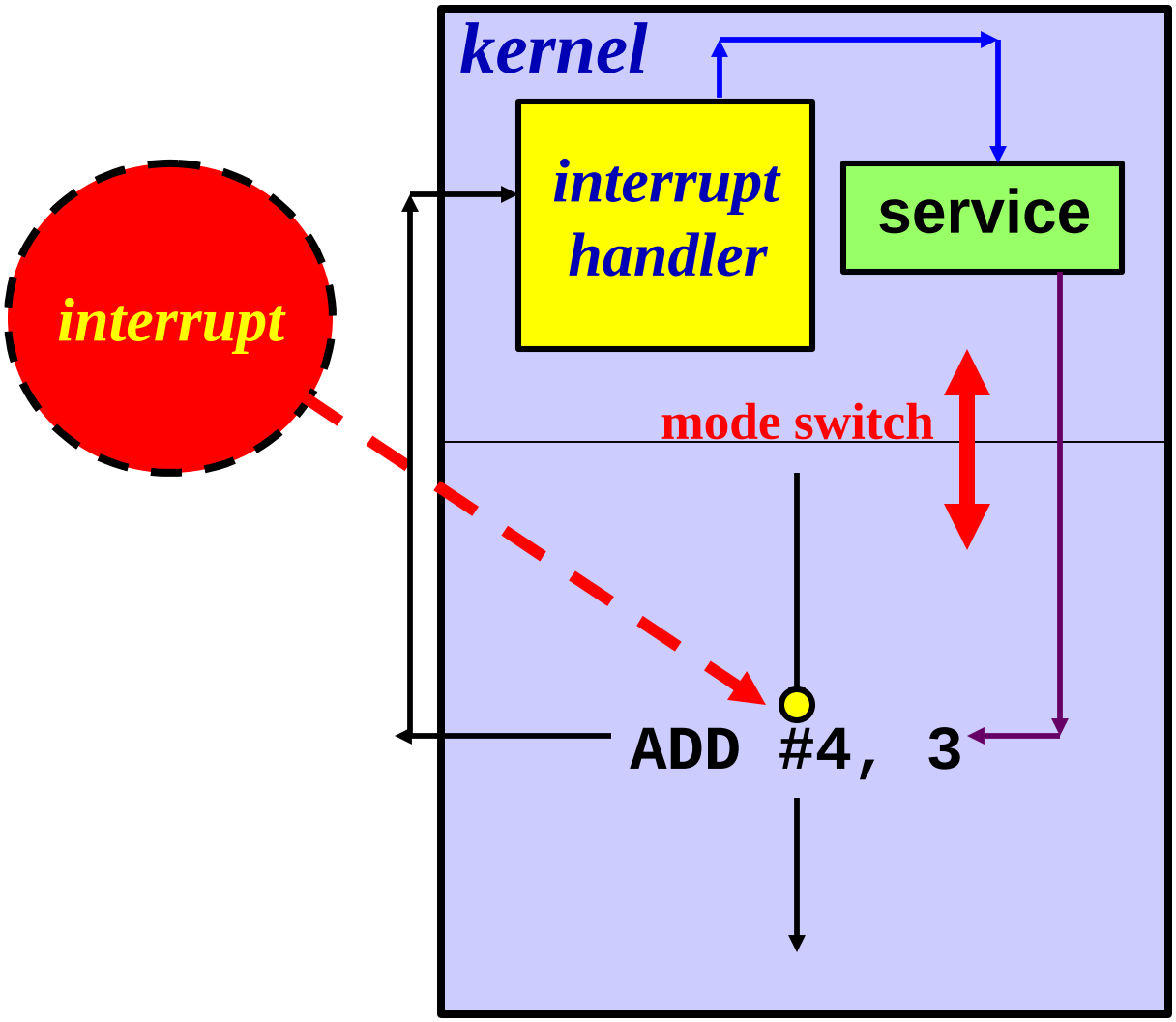
a typical PCI bus structure



Interrupt and Trap

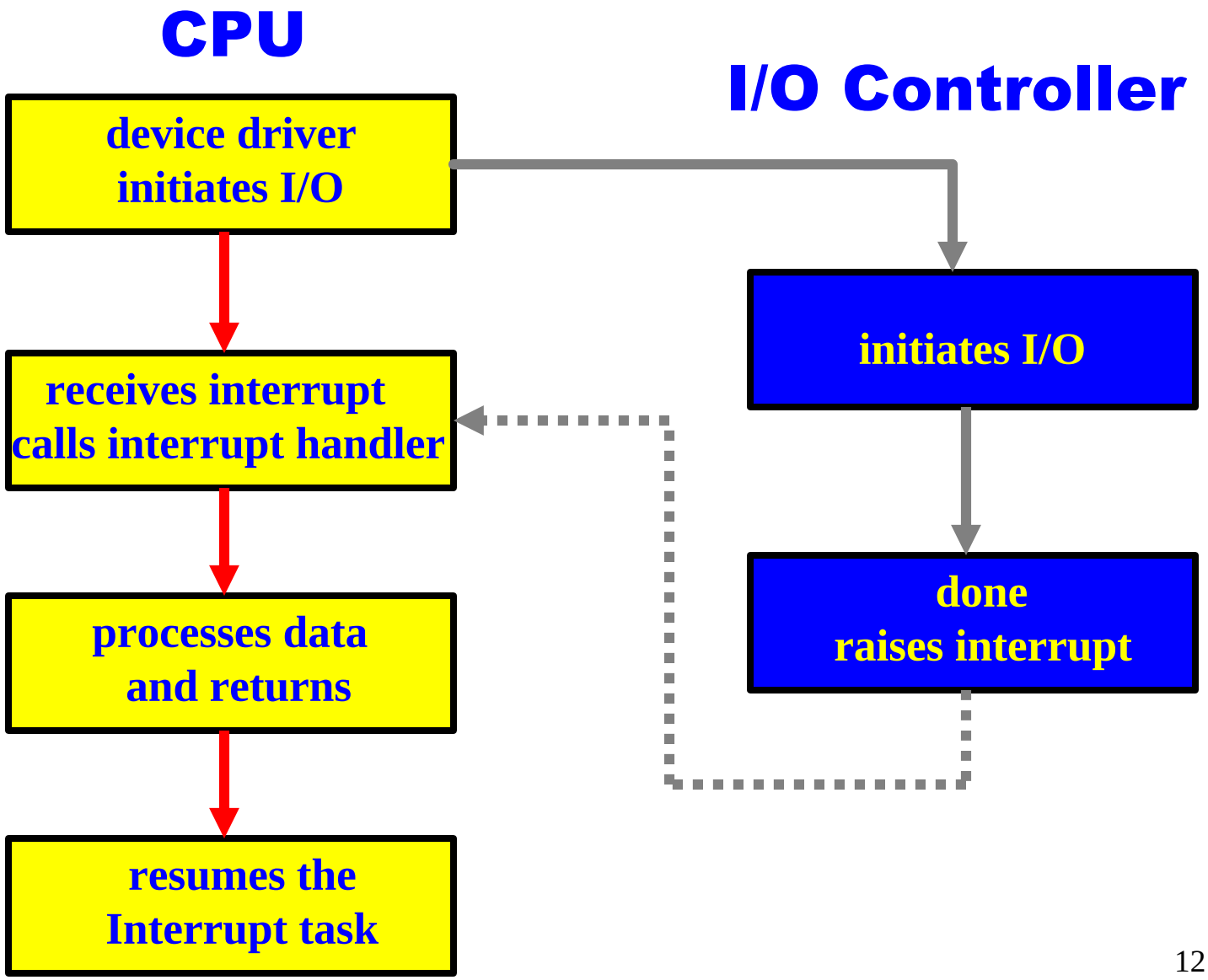
- **An event that requires the attention of the OS is an *interrupt*. These events include the completion of an I/O, a keypress, a request for service, a division by zero and so on.**
- **Interrupts may be generated by hardware or software.**
- **An interrupt generated by software (*i.e.*, division by 0) is usually referred to as a *trap*.**
- **Modern operating systems are *interrupt driven*, meaning the OS is in action only if an interrupt occurs.**

What is Interrupt-Driven?



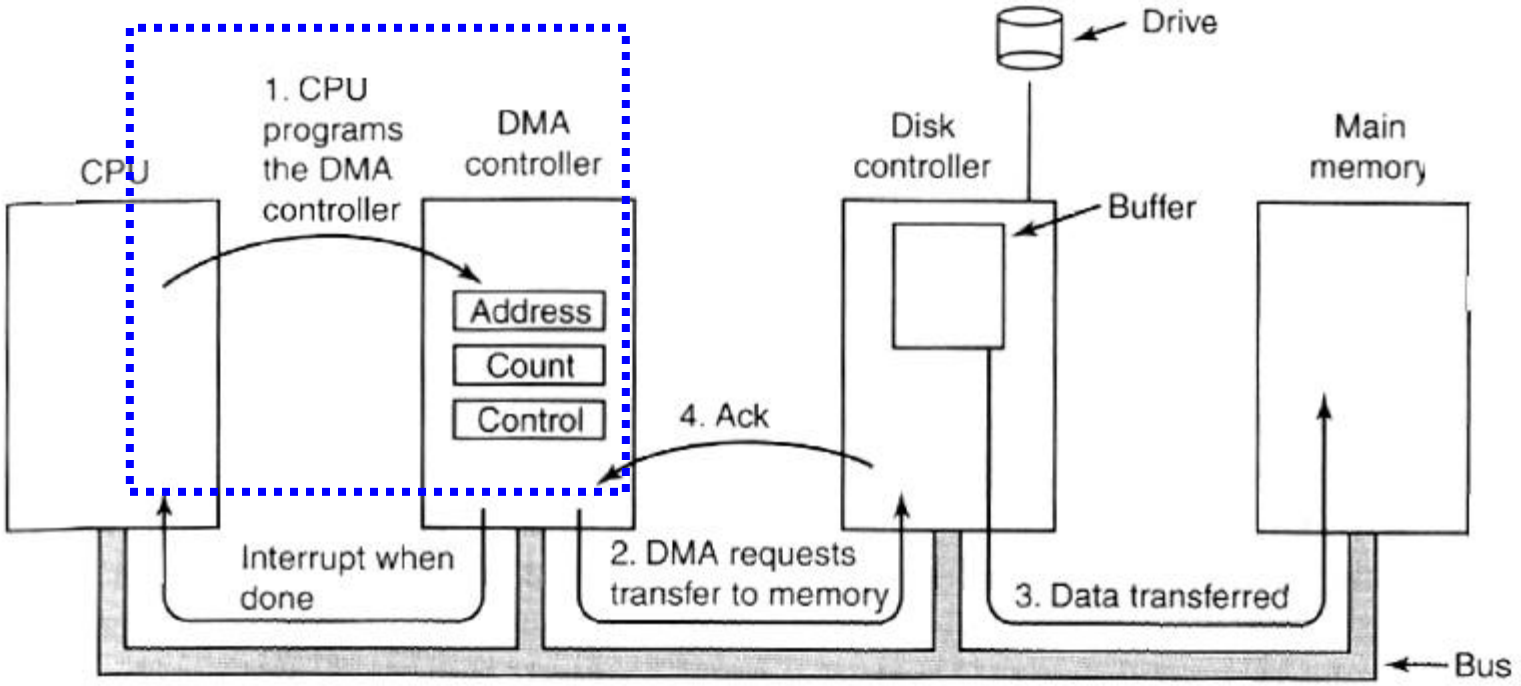
- The OS is activated by an interrupt.
- The executing program is suspended.
- Control is transferred to the OS.
- Program continues when the service completes.

I/O Interrupt



Direct Memory Access

- For large volume data transfer, most systems use direct memory access to avoid burdening the CPU.
- The CPU gives the controller (1) disk address, (2) memory address for storing the block, and (3) a byte count. Then, the CPU goes back to work.



Computer-System Architecture

- **Single-Processor Systems**
- **Multiprocessor Systems**
- **Clustered Systems**

****details will be presented in next few pages****

Multiprocessor Systems: 1/3

- **Multiprocessor systems have more than one CPUs.**
- **Advantages:**
  - **Increased throughput: gets more jobs done**
  - **Economy of scale: Because of resource sharing, multiprocessor systems is cheaper than multiple single processor systems.**
  - **Increased reliability: the failure of one processor will not halt the whole system.**

Multiprocessor Systems: 2/3

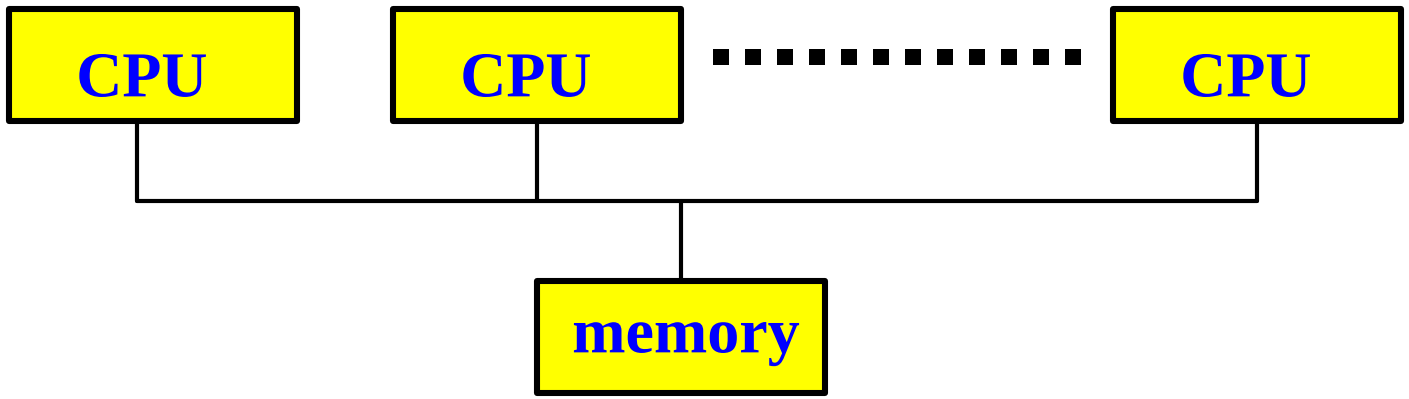
Tightly Coupled Systems or Parallel Systems

- **Asymmetric Multiprocessing:**
  - **Each processor is assigned a specific task.**
  - **A master processor controls the system. The other processors either look to the master for instructions or have predefined tasks.**
  - **Thus, this defines a master-slave relationship.**
- **Symmetric Multiprocessing (SMP):**
  - **Each processor runs an identical copy of OS**
  - **These copies communicate with each other.**

Multiprocessor Systems: 3/3

Tightly Coupled Systems or Parallel Systems

- **Symmetric Multiprocessing (SMP):**
  - **Each processor performs all tasks within the OS**
  - **All processors are peers; no master-slave relationship exists**

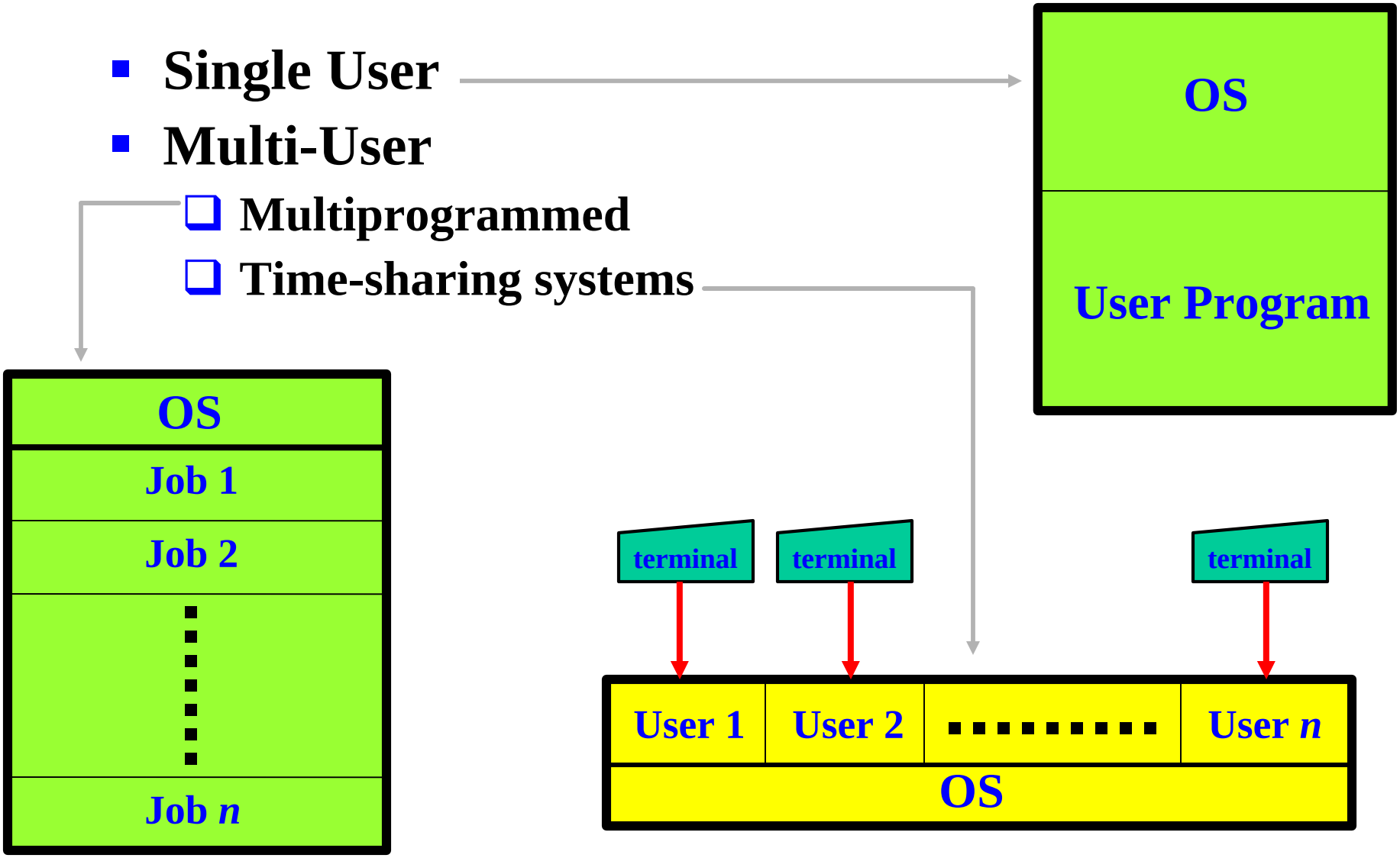


Clustered Systems

- **A clustered system has two or more individual systems connected by a local area network (LAN) or interconnection network.**
- **The key of clustered system is high availability.**
- **Asymmetric Clustering:**
  - **One machine is in hot-standby mode while others are running applications.**
  - **The hot-standby machine (*i.e.*, does nothing but) monitors other machines and becomes active if one server fails.**
- **Symmetric Clustering:**
  - **Two or more hosts are running applications and monitor each other.**
  - **This is more efficient as it uses all available hardware.**

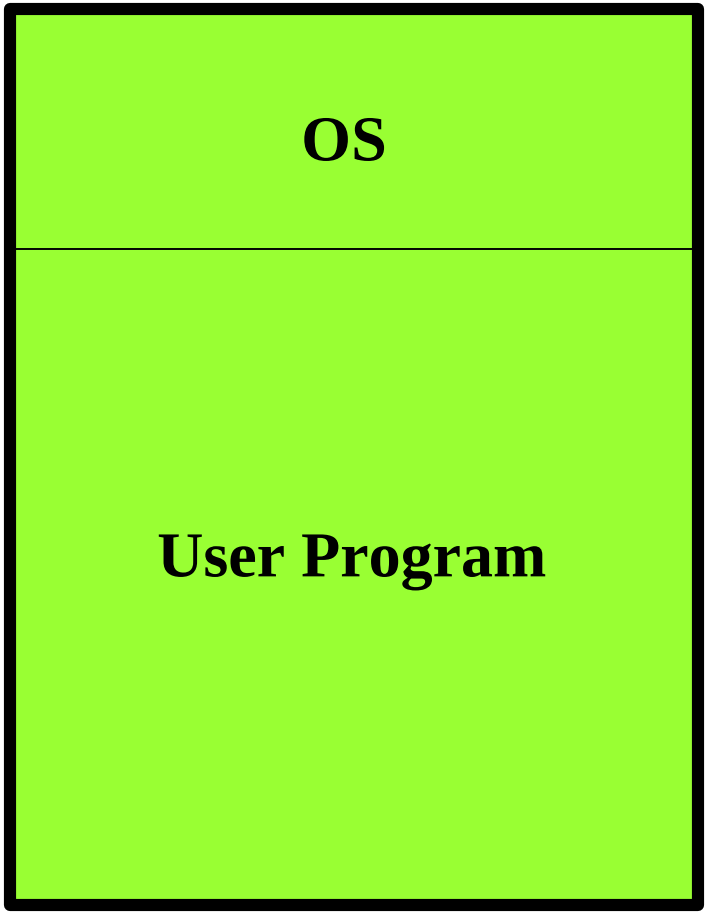
Operating-System Structure 1/4

- Single User
- Multi-User
  - Multiprogrammed
  - Time-sharing systems



Operating-System Structure: 2/4

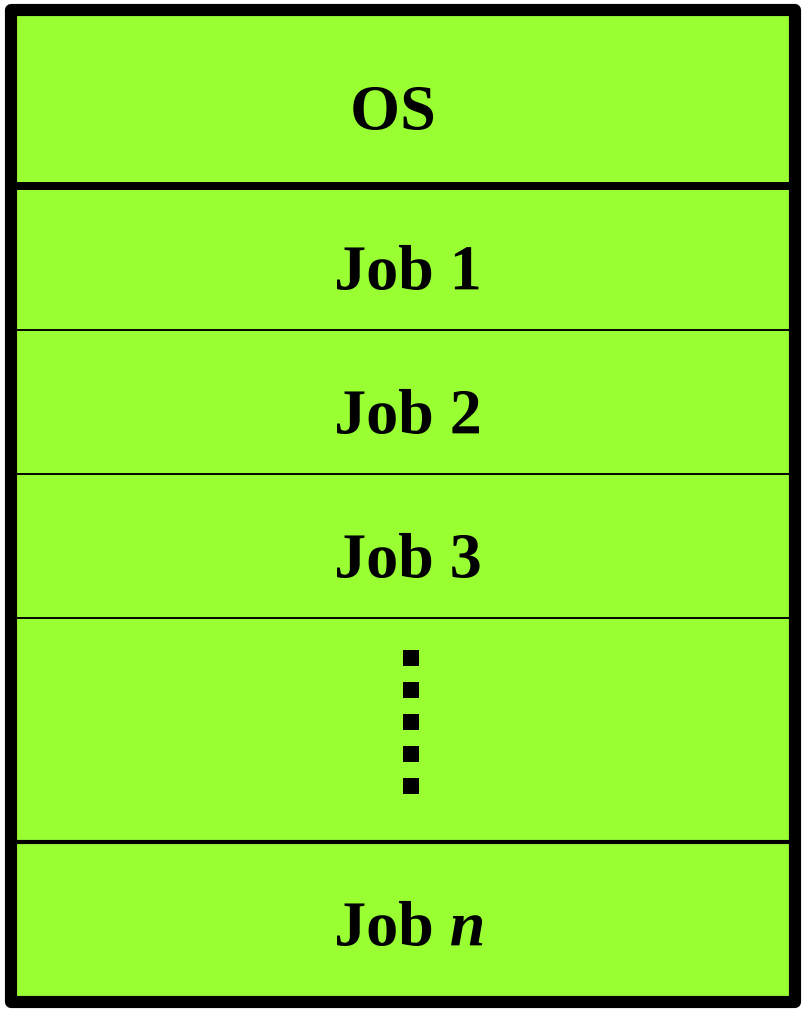
Single User System



- Users submit programs to an operator.
- Programs are *batched*.
- A *job scheduler* determines which program runs next.
- *Very inefficient* because the CPU is *idle* when the running program is doing I/O.

Operating-System Structure: 3/4

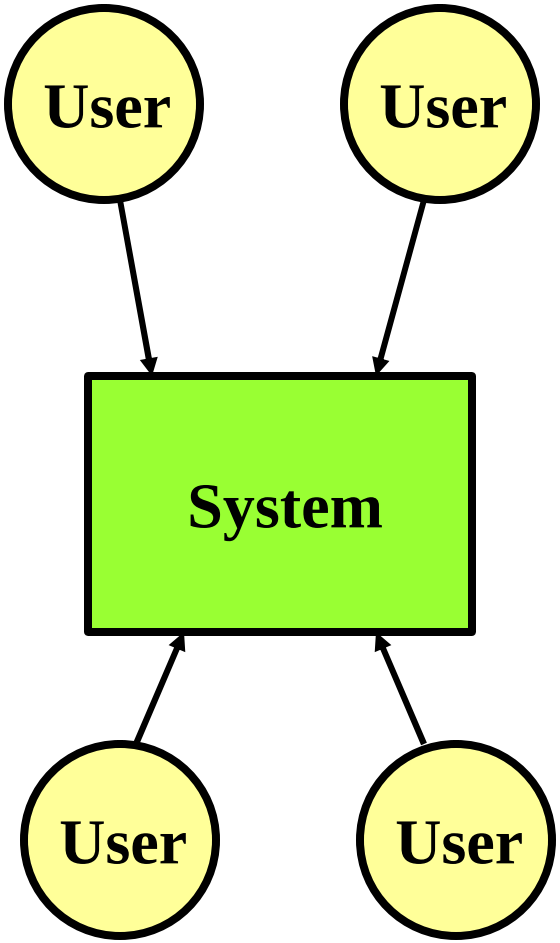
Multiprogrammed Systems



- **Several programs may run in the system at *the same time*.**
- **A *CPU scheduler* selects a job to use the CPU.**
- **A *job scheduler* selects jobs to enter the system.**
- **While a job is doing I/O, another can use the CPU. Thus, it is more *efficient*.**

Operating-System Structure: 4/4

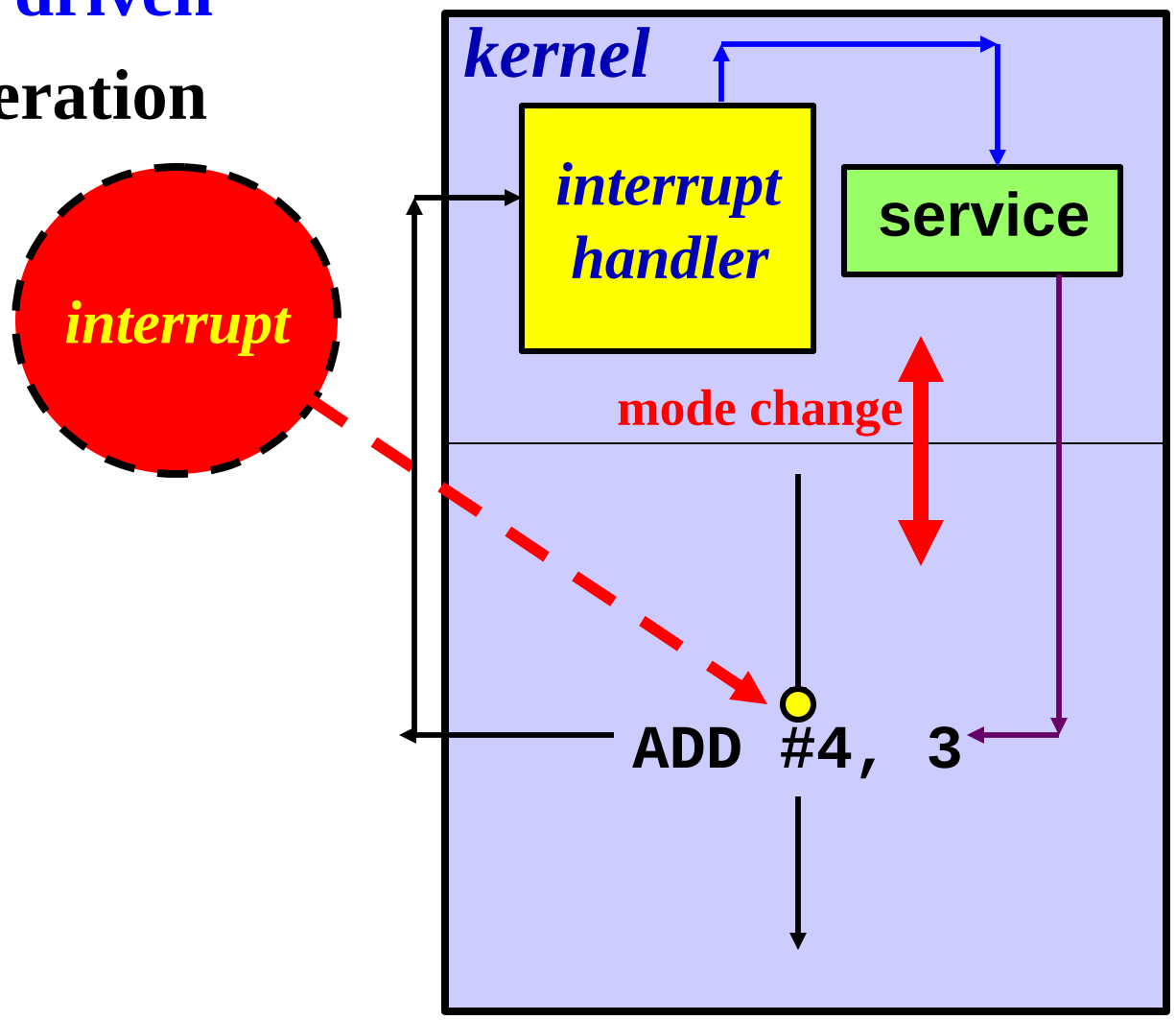
Time-Sharing Systems



- Users provide instructions to the OS or programs directly using a keyboard or a mouse.
- The *response time* should be short.
- Thus, *time-sharing* systems maximize *usability* while *batch* systems maximize *resource usage*.

Operating-System Operations

- OS is interrupt driven
- Dual Mode Operation
- Timer



Dual-Mode Operation

- **Modern CPUs have two execution modes: *user* mode and *supervisor* (or system, monitor, privileged) mode, controlled by a mode bit.**
- ***The OS runs in supervisor mode and all user programs run in user mode*.**
- **Some instructions that may do harm to the OS (*e.g.*, doing I/O) are *privileged instructions*. Privileged instructions can only be used in the supervisor model.**
- **When the control switches to the OS (*resp.*, a user program), execution mode will be changed to supervisor (*resp.*, user) mode.**

Timer

- **Because the operating system must maintain the control over the CPU, we must prevent a user program from getting the CPU forever or not calling system service.**
- ***Use an interval timer*! An interval timer is a count-down timer.**
- **Before a user program runs, the OS sets the interval timer to certain value. Once the interval timer counts down to 0, an interrupt is generated and the OS can make appropriate action.**

Distributed Systems: 1/3

- **A distributed system is a collection of physically *separate*, possible *heterogeneous* computer systems that are *networked* to provide the users with access to the various resources that the system maintains.**
- **By being able to communicate, distributed systems are able to share computational tasks, and provide a rich set of features to users.**

Distributed Systems: 2/3

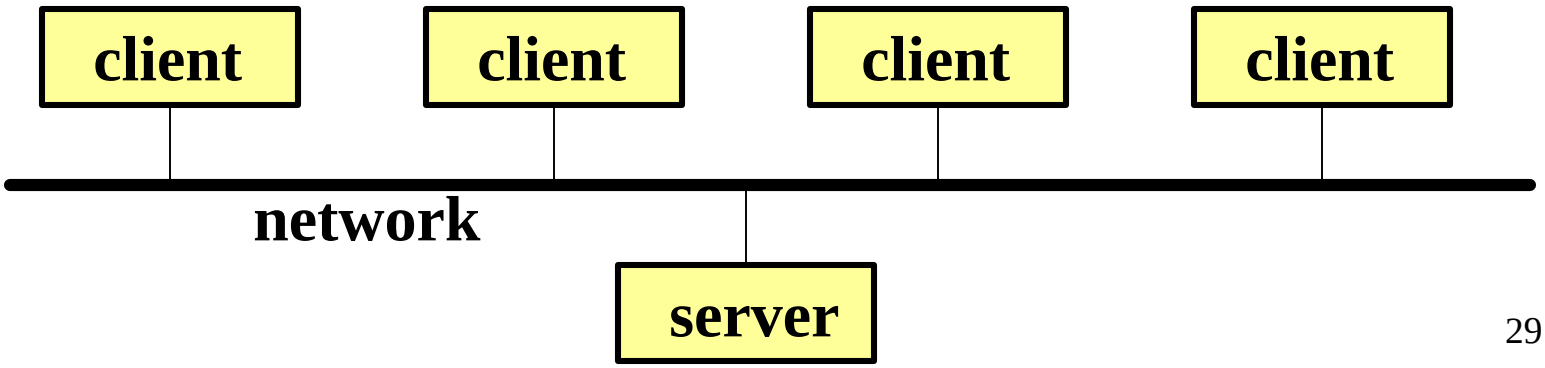
- **A network operating system is an OS that**
  - **provides features such as file sharing across the network**
  - **includes a communication scheme that allows different processes running on different computers to exchange messages**
- **A computer in a network operating system acts *autonomously* from all other computers on the network, although it is aware of the network and is able to communicate with other computers.**

Distributed Systems: 3/3

- **A distributed operating system is less autonomous.**
- **Different operating systems communicate closely enough to provide the illusion that only a *single operating system controls the network*.**

Client-Server Computing

- **Some systems are assigned as servers to satisfy requests generated by client systems.**
- **Client-server systems are specialized distributed systems.**
- **In general, there are two types of server systems**
  - **Computer-Server Systems: provide an interface to which clients can send requests to perform an action. Then, execute the action and send back the results.**
  - **File-Server Systems: provide a file system interface where clients can create, update, read and delete files.**



Real-Time Operating Systems

- **A real-time operating system has well-defined, fixed time constraints.**
- **Processing *must* be done within the defined constraints, or the system will fail.**
- ***Hard Real-Time Systems* guarantee that critical tasks be completed on time.**
- ***Soft Real-Time Systems* prioritize critical tasks. That is, a critical task get priority over other tasks, and retains that priority until it completes.**
- **Embedded systems almost always run real-time operating systems.**