Amirkabir University of Technology
(Tehran Polytechnic)

Operating Systems

Multiprogramming and Dual-mode

Seyyed Ahmad Javadi

sajavadi@aut.ac.ir

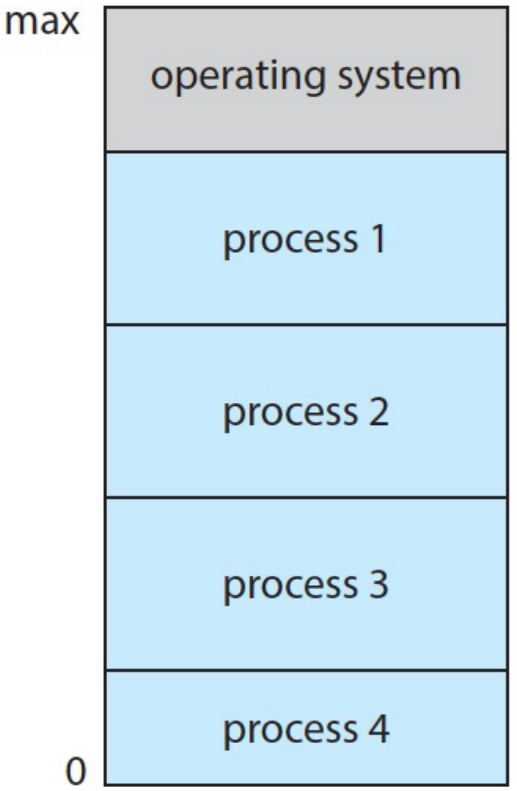
Fall 2021

Multiprogramming (Batch System) (cont.)

- Multiprogramming organizes multiple jobs (code and data) -->
  - CPU always has one to execute.
- A subset of total jobs

  in system is kept in memory.
- One job selected and run via job scheduling.
- When job has to wait (I/O for example),

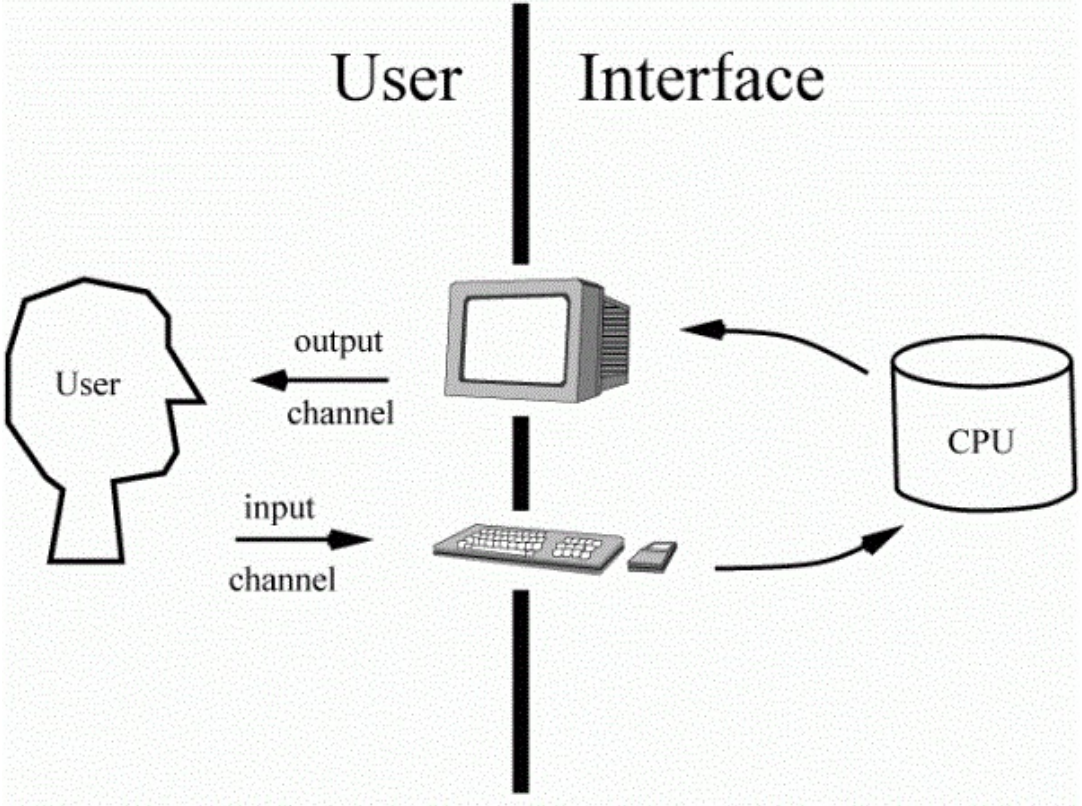
  OS switches to another job.



Memory layout for a multiprogramming system

Multiprogramming (Batch System)

- Single user/program cannot always keep CPU and I/O devices busy.



Multiprogramming (Batch System) (cont.)

- Single user/program cannot always keep CPU and I/O devices busy.
- Examples

Program	CPU-intensive	Memory-intensive	I/O-intensive
Random Number Generator	?	?	?
Microsoft word	?	?	?
QuickTime Player (a long 4K video)	?	?	?

Multitasking (Timesharing)

- A logical extension of Batch systems
- The CPU ***switches jobs so frequently*** that users can interact with each job while it is running, creating **interactive** computing.
  - Response time should be < 1 second.
  - Each user has at least one program executing in memory ⇨ process.
  - If several jobs ready to run at the same time ⇨ CPU scheduling.
  - If processes don't fit in memory, swapping moves them in&out to run.
  - Virtual memory allows execution of processes not completely in memory.

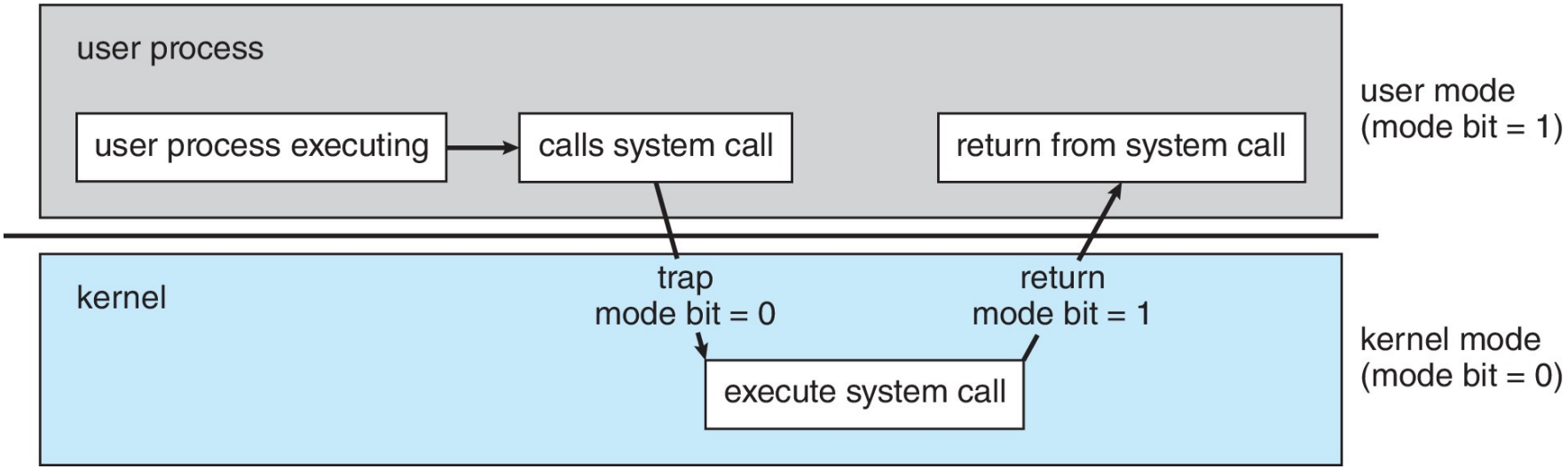
https://www.geeksforgeeks.org/difference-between-job-task-and-process/

Dual-mode Operation

- **Dual-mode** operation allows OS to protect itself and other system components.
  - **User mode** and **kernel mode**
- **Mode bit** provided by hardware
  - Provides ability to distinguish when system is running user code or kernel code.
  - When a user is running ⇨ mode bit is “user”.
  - When kernel code is executing ⇨ mode bit is “kernel”.

Dual-mode Operation (Cont.)

- How do we guarantee that user does not explicitly set the mode bit to “kernel”?
  - System call changes mode to kernel, return from call resets it to user.



Types of Instructions

- Instructions are divided into two categories:
  - The ***non-privileged instruction*** instruction is an instruction that ***any application or user can execute***.
  - The ***privileged instruction*** is an instruction that ***can only be executed in kernel mode***.
- Instructions are divided in this manner because privileged instructions ***could harm the kernel***.

http://web.cs.ucla.edu/classes/winter13/cs111/scribe/4a/

Examples of instructions

Instruction	Type
Reading the status of Processor	?
Set the Timer	?
Sending the final printout of Printer	?
Remove a process from the memory	?

Examples of non-privileged instructions

- Reading the status of Processor
- Reading the System Time
- Sending the final printout of Printer

https://www.geeksforgeeks.org/privileged-and-non-privileged-instructions-in-operating-system/

Examples of privileged instructions

- I/O instructions and halt instructions
- Turn off all Interrupts
- Set the timer
- Context switching
- Clear the memory or remove a process from the memory
- Modify entries in the device-status table

https://www.geeksforgeeks.org/privileged-and-non-privileged-instructions-in-operating-system/

Privileged instructions

If an attempt is made to execute a privileged instruction in user mode

The hardware ***does not execute the instruction*** but rather treats it as ***illegal*** and ***traps*** it to the ***operating system.***