Lecture 2
Chapter 1
OS Operation and Functions

Dual-Mode and Multimode Operation

- In order to ensure the proper execution of the operating system, we must be able to distinguish between the execution of operating-system code and user defined code.
- There are two separate ***modes of operation:*** ***user mode*** and **kernel mode (also called supervisor mode, system mode, or privileged mode).**
- A bit, called the **mode bit,** is added to the hardware of the computer to indicate the current mode: kernel (0) or user (1). With the mode bit, we can distinguish between a task that is executed on behalf of the operating system and one that is executed on behalf of the user.

- At system boot time, the hardware starts in kernel mode.
- The operating system is then loaded and starts user applications in user mode.
- Whenever a trap or interrupt occurs, the hardware switches from user mode to kernel mode (that is, changes the state of the mode bit to 0).
- Thus, whenever the operating system gains control of the computer, it is in kernel mode.

- **The dual mode of operation provides us with the means for protecting the operating system from errant user.**

- We accomplish this protection by designating some of the machine instructions that may cause harm as **privileged instructions.**
- The hardware allows privileged instructions to be executed only in kernel mode. 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.

- Examples of a privileged instructions include I/O control, timer management, and interrupt management.

- **The concept of modes can be extended beyond two modes.**

- CPUs that support Virtualization frequently have a separate mode to indicate when the **virtual machine manager (VMM)—**and the virtualization management software—is in control of the system.
- In this mode, the VMM has more privileges than user processes but fewer than the kernel. It needs that level of privilege so it can create and manage virtual machines, changing the CPU state to do so.

The life cycle of instruction execution

- Initial control resides in the operating system, where instructions are executed in kernel mode.
- When control is given to a user application, the mode is set to user mode.
- Eventually, control is switched back to the operating system via an interrupt, a trap, or a system call.

  (System calls provide the means for a user program to ask the operating system to perform tasks reserved for the operating system on the user program's behalf).

- When a system call is executed, it is typically treated by the hardware as a software interrupt. Control passes through the interrupt vector to a service routine in the operating system, and the mode bit is set to kernel mode. The system-call service routine is a part of the operating system.
- The kernel examines the interrupting instruction to determine what system call has occurred; a parameter indicates what type of service the user program is requesting. Additional information needed for the request may be passed in registers, on the stack, or in memory.
- The kernel verifies that the parameters are correct and legal, executes the request, and returns control to the instruction following the system call.

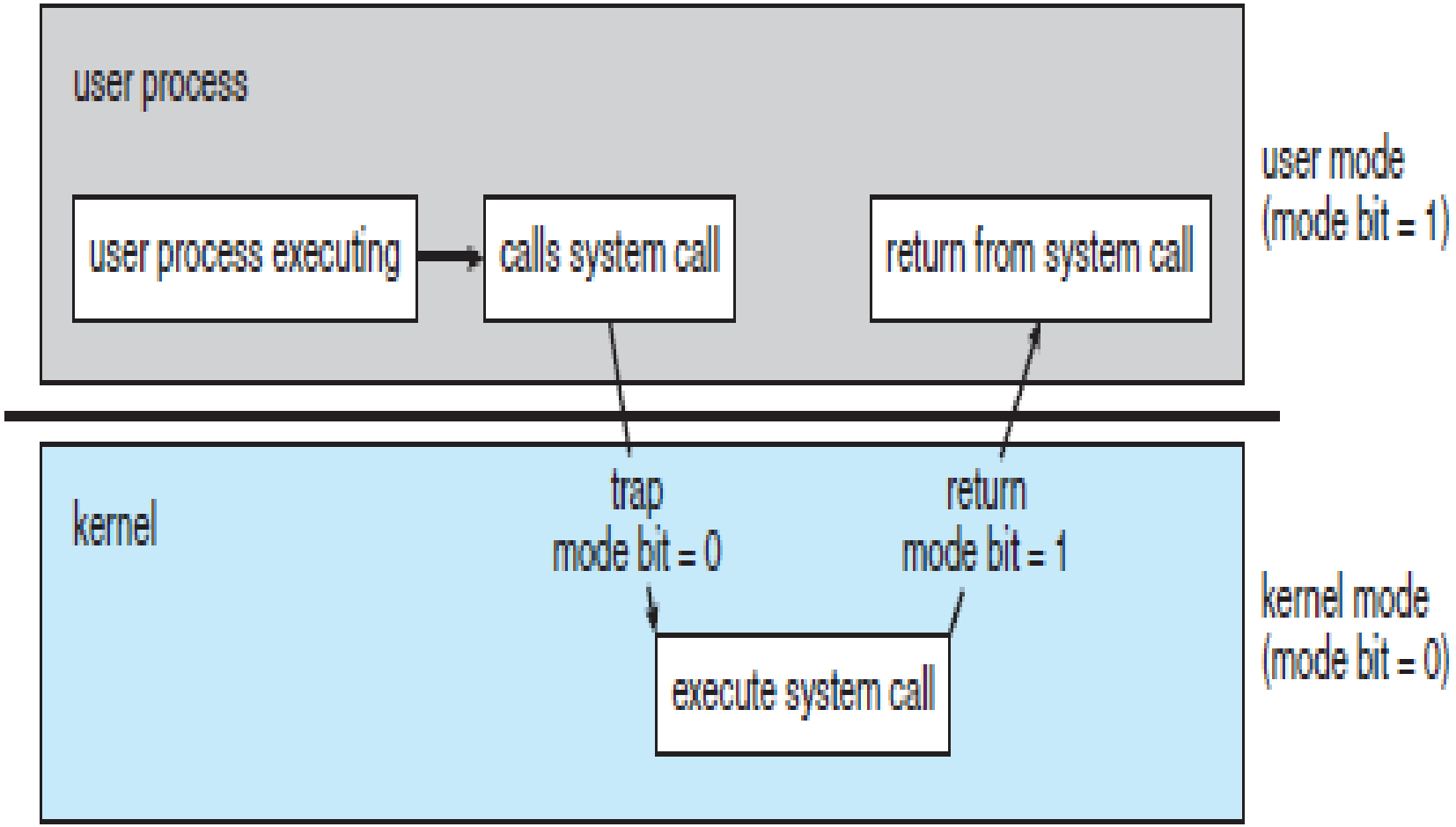

Figure 1.10 Transition from user to kernel mode.

Timer

- We must ensure that the operating system maintains control over the CPU.
- We cannot allow a user program to get stuck in an infinite loop or to fail to call system services and never return control to the operating system.
- To accomplish this goal, we can use a **timer.** A timer can be set to interrupt the computer after a specified period.

OS Functions Overview

Process Management

- A program is a ***inactive entity,*** like the contents of a file stored on disk, whereas a process is an ***active entity.***
- A process needs certain resources—including CPU time, memory, files, and I/O devices—to accomplish its task. These resources are either given to the process when it is created or allocated to it while it is running.

- A process is the unit of work in a system. A system consists of a collection of processes, some of which are operating-system processes the rest of which are user processes.
- All these processes can potentially execute concurrently—by multiplexing on a single CPU, for example.

- The operating system is responsible for the following activities in connection with process management:

1. Scheduling processes and threads on the CPUs
2. Creating and deleting both user and system processes
3. Suspending and resuming processes
4. Providing mechanisms for process synchronization
5. Providing mechanisms for process communication

Memory Management

- The operating system is responsible for the following activities in connection with memory management:

1. Keeping track of which parts of memory are currently being used and who is using them
2. Deciding which processes (or parts of processes) and data to move into and out of memory
3. Allocating and deallocating memory space as needed

Storage Management

- To make the computer system convenient for users, the operating system provides a uniform, logical view of information storage.
- The operating system abstracts from the physical properties of its storage devices to define a logical storage unit, the **file.**
- The operating system maps files onto physical media and accesses these files via the storage devices.

File-System Management

The operating system is responsible for the following activities in connection with file management:

1. Creating and deleting files
2. Creating and deleting directories to organize files
3. Supporting primitives for manipulating files and directories
4. Mapping files onto secondary storage
5. Backing up files on stable (nonvolatile) storage media

Mass-Storage Management

The operating system is responsible for the following activities in connection with disk management:

1. Free-space management
2. Storage allocation
3. Disk scheduling

I/O Systems

The OS I/O subsystem consists of several components:

1. A memory-management component that includes buffering, caching, and spooling
2. A general device-driver interface
3. Drivers for specific hardware devices

Protection and Security

If a computer system has multiple users and allows the concurrent execution of multiple processes, then access to data must be regulated.

For that purpose, mechanisms ensure that files, memory segments, CPU, and other resources can be operated on by only those processes that have gained proper authorization from the operating system.

- **Protection** is any mechanism for controlling the access of processes or users to the resources.
- A protection-oriented system provides a means to distinguish between authorized and unauthorized usage.

- **Security** is used to defend a system from external and internal attacks.
- Such attacks spread across a huge range and include viruses and worms, denial-of service attacks.