Lecture 1: Operating System

Necessary stuff

- Linux as a platform to use
  - Virtual Box / VMWare
  - Any stable distro
  - Ubuntu
    - apt-get install --no-install-recommends lubuntu-desktop
    - apt-get install lxde
  - Netbeans IDE
- Do file handling in C/C++
  - The 0^{th} assignment

What is an Operating System?

- A program that acts as an intermediary between a user of a computer and the computer hardware
- Operating system goals:
  - Make the computer system convenient to use
  - Use the computer hardware in an efficient manner

Computer System Structure

- Computer system can be divided into four components:
  - Hardware – provides basic computing resources
    - CPU, memory, I/O devices
  - System Software
    - **Operating System** - Controls and coordinates use of hardware among various applications and users
  - Application programs – define the ways in which the system resources are used to solve the computing problems of the users
    - Word processors, compilers, web browsers, database systems, video games
  - Users
    - People, machines, other computers

What Operating Systems Do

- An operating system is a resource manager.
- An OS has different components that manage different resources:

1. Process Management
2. Memory Management
3. File Management
4. Disk Management

Process Management

- A **process** is a program in execution.

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

- Creating and deleting both user and system processes
- Scheduling processes and threads on the CPUs
- Suspending and resuming processes
- Providing mechanisms for **process synchronization**
- Providing mechanisms for **inter-process communication**

Memory Management

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

• Providing memory protection, i.e. it ensures that a process does not interfere with the memory of another process.

• Keeping track of which parts of memory are currently being used and which process is using them

• Allocating and deallocating memory space as needed

• Deciding which processes (or parts of processes) and data to move into and out of memory

File System Management

A file is a collection of related information.

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

- Creating and deleting files.
- Creating and deleting directories to organize files.
- Providing an interface to the programmer to access a file.
- Providing access control.
- Mapping files onto mass storage.

Mass Storage Management

- Free-space management
- Storage allocation
- Disk scheduling
- Partitioning

Other functionalities of the OS

Other functionalities of the OS are:

- I/O management
- Network Management
- providing security and protection

Operating System Definition

- OS is a **resource allocator**
  - Manages all resources
  - Decides between conflicting requests for efficient and fair resource use
- OS is a **control program**
  - Controls execution of programs to prevent errors and improper use of the computer

Kernel

- The kernel is the core part of the operating system.

- The kernel has absolute control over all resources of the computer it is running on.

- The kernel is part of the OS that manages all resources.

- To ensure proper functioning of the system, the kernel is always memory resident.

System Calls

- A **system call** is the programmatic way in which a computer program requests a service from the kernel of the operating system it is executed on. A system call is a way for programs to **interact with the operating system**.

- A computer program makes a system call when it makes a request to the operating system's kernel.

- It provides an interface between a process and operating system to allow user-level processes to request services of the operating system.

- Some examples of system calls are:

Process-related System Calls: fork(), exit(), wait()

File-related System Call: open(), close(), read(), write()

Direct Memory Access (DMA)

- To read and write data to a device, such as a disk, instead of having the CPU perform this task, the CPU delegates this task to the device controller. It's known as direct memory access (DMA).

- Direct memory access (DMA) allows an input/output (I/O) device to send or receive data directly to or from the main memory, bypassing the CPU.

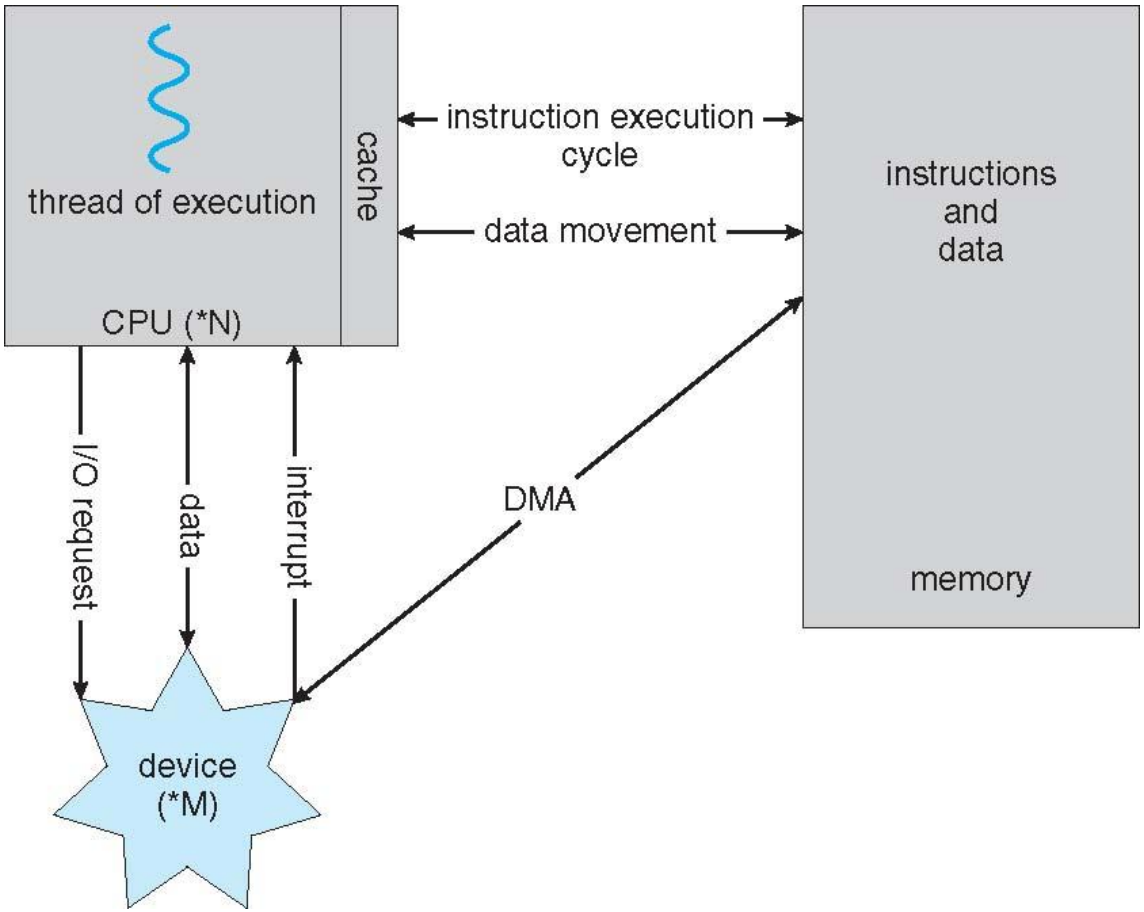
- For example, suppose that a process wants to read a file from the disk. Instead of having CPU read the file and store it in RAM, the CPU can ask the disk controller to read the file and store it in RAM.

Direct Memory Access (DMA)

- While the Disk controller is performing its task, the CPU can be assigned to another process.

- When the device controller has finished reading or writing the data to the device it controls, it generates an interrupt to the CPU, signaling that the task has been finished.

DMA



A von Neumann architecture

Device Drivers & Device Controllers

- The operating system interacts with devices using **device driver**.
- Each device has a **device controller** that controls that device.
- The OS gives instructions to a device controller via the device driver.
- When the device has finished executing the operation that is was required to do, its controller gives signal to the OS via an interrupt (Modern operating systems are interrupt-driven).

Dual-mode

- There are some instructions that are **privileged**, such as: disabling/enabling interrupts, setting the timer, etc.
- If the OS allows a user program to access privileged instructions, the result may be damaging.
- So, the normal user programs are not allowed to execute privileged instructions. The OS ensures this using dual modes of execution.
- **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
    - ✓ Some instructions designated as **privileged**, only executable in kernel mode
    - ✓ System call changes mode to kernel, return from call resets it to user

Dual-mode

- User programs are run only in the user-mode in which trying to execute a priviliged instruction will cause an error/exception.
- Privileged instruction can only be run in the kernel mode (privileged mode).
- System calls are also executed in the kernel mode since the instructions in a system call are privileged.

Transition from User to Kernel Mode