Introduction

Operating Systems

- An **Operating System** is system software that manages computer hardware, software resources, and provides common services to all computer programs.
- Operating Systems provide a layer of **abstraction** that programs can use to perform operations that are **independent** of the physical hardware.
- There are **two main goals** for any operating system:
  1. Provide a **user-friendly environment**, that allows the user to execute their desired programs.
  2. Manage system resources as **efficiently** as possible.

Types of Operating Systems

- There are many types of operating systems, ranging from **general-purpose operating systems**, to **embedded operating systems**.
- A **general-purpose operating system** is an operating system that support process management, memory management, IO devices, a file system, and a user interface. That can solve a wide range of problems.
- An **embedded operating system** is a specialized operating system designed to perform a specific task for a specific device. They often lack many features that general-purpose operating systems have.
- There are two ways operating systems can be viewed:
  1. **The User View** is concerned with what the end user will be using the operating system for.
  2. **The System View** is concerned with the way the operating system will control programs, and manage resources.

The Operating System Kernel

- The **kernel is the core of an operating system**, and is a process that is **always running** when the system is on.
- The kernel **facilitates interactions** between **hardware components** and **software applications**.

Hardware Controllers

- The **physical hardware components** of a computer system are managed by a **controller** which acts as an **intermediary** between the device and the rest of the system.
- **Device controllers** work by **handling raw signals coming from the CPU and directing the hardware accordingly**.
- **Controllers** contain a **buffer** that is responsible for **communicating data between the devices they control, and the rest of the system**.
- The **controllers** are connected to the **system bus**, which gives the controllers access to **shared memory** that can be used to communicate with other components.
- **Drivers** are a special type of software that **manage device controllers**.
- To sum it up, **controllers** handle signals from the CPU and access shared memory. Whereas **drivers** are responsible for managing the device.

System Events

System Events

- An **event** is an **action** or **occurrence** recognized by software.
- Operation system are **event driven**.
- There are three main categories of events:
  1. **Hardware Interrupts** are events that are raised by **hardware devices**. They can occur at any time.
  2. **Software Interrupts (Traps)** are events that are raised by **programs to invoke an operating system functionality**.
  3. **Exceptions** are events that are **generated automatically** by the processor as the result of an **illegal instruction / operation**.
- There are two types of exception events:
  1. **Faults** are **exceptions** that the program **can recover from**.
  2. **Aborts** are **exceptions** that the program **cannot, or are very difficult to recover from**.

Hardware Interrupts

- A **hardware interrupt** is an **electronic alerting signal** that is **sent to the process** from an **external device**.
- If the **interrupt is permitted** by the **processor**, the **processor will stop executing it's current instructions, save it's state, and execute a function** known as an **interrupt handler**.
- **Interruptions** are often **temporary** and the **process can resume what it was doing before the interrupt**.
- The process of **the processer storing it's state** so that it can be **resumed later** is known as **context switching**.
- To handle **interruptions** from **several devices** an **interrupt controller is used**.
- **Interrupt controllers** provides a **programmable governing policy** that allows **software to determine which devices can interrupt the process at any specific time**.
- **Interrupt controllers** also allow **device controllers** to define a **device specific interrupt handler routines**.
- Different **interrupt signals** are given different **priorities** to avoid conflicts with **simultaneous interrupt signals**.
- The order **simultaneous interrupt signals** are sent to the processor is **controlled by the interrupt controller**.
- The **processor** users an **interrupt descriptor table (IDT)** to reference the **interrupt handler** that corresponds with each **interrupt signal**.
- The **IDT** has 256 entries in the table.
- In **multi-processor systems** an **Advanced Programmable Interrupts Controller (APIC)** is used to communicate **interrupt signals between processors**.

Types of Interrupts

- There are two types of hardware interrupts:
    1. **Non-Maskable Interrupts** - Non-Maskable Interrupts are **interrupts** that **cannot be ignored**. They are generally reserved for **unrecoverable errors**.
    2. **Maskable Interrupts** - Maskable Interrupts are **interrupts** that **can be ignored, or delayed by the processor**. They are generally used for **device controller requests**.

System Memory

Computer Memory

- In computing **memory is a device** that is used to store information.
- There are **different types of memory** that are used for **different purposes**.
- The following are common types of memory in computers:
    1. A **register** is a **small amount of volatile high-speed memory** contained directly **inside the processor**. **Registers** are used to store data **needed during processing**.
    2. **Cache** is a **volatile temporary memory** location inside **hardware** that makes **retrieving data from the computer's memory more efficient**. It stores recently accesses data.
    3. **Main Memory (Primary Memory)** is a **volatile large and fast memory** which his used to **store programs and data during runtime**. The **processor has direct access to the main memory**.
    4. **Secondary Memory** is a **non-volatile, long-term storage**. It is used to keep **data and programs indefinitely**.
    5. **Electrically Erasable Programmable Read-Only Memory (EEPROM)** is a special type of **non-volatile read-only memory**, *usually stored on the system's motherboard*, that is responsible for **storing the systems BIOS**.
- With computer memory, there is a **trade off** between **storage capacity** and **access time**. In the list above, as you go from 1 to 5, **capacity increases**, but **access time decreases**.

Programmed Input / Output

- **Programmed Input / Output** is a way of **moving data** between **devices**, where **all of the data passes through the processor**.
- This type of IO is **inefficient** because it takes a **large amount of time** for the processor to **perform IO**, when it could be doing other operations.
- A more **modern** way to perform IO is with **direct memory access**.

Interrupt-Initiated Input / Output

- **Interrupt-Initiated Input / Output** is a **more efficient** way than **Programmed I/O** to perform IO.
- **Interrupt-Initiated Input / Output** works by **interrupting the processor**, when the data is **ready to be transferred**, instead of **blocking the processor while it prepares the data to be transferred**.
- **All of the data still has to go through the processor**, but it **blocks the processor for less time**.

Direct Memory Access (DMA)

- **Direct Memory Access** is a capability provided by the **computer bus**, it allows for **data** to be sent directly from an **attached device** to the memory on the **computer's motherboard**.
- Typically a **specified portion of memory** is **designated** as an area to be used for **direct memory access**.
- **DMA does not required the processor to transfer data**.

Computer System Architecture

Processors

- **Computer Systems** may have **one or more processors**.
- **Most systems** have a **single processor**, however **multi-processor systems** are becoming **more common**.
- **Advantages** of having **multiple processors** include **increased throughput, economy, reliability, etc**.
- There are **two ways** an operating system can **handle multiple processors**:
  1. **Asymmetric Multiprocessing** is when each processor is assigned a **specific task**.
  2. **Symmetric Multiprocessing** is when each processor performs **any tasks**.
- **Modern processors** have **multiple cores**, allowing them to perform **multiple operations simultaneously**.

Clustered Systems

- **Clustered Systems** are similar to **multi-processor systems**. They are **several computer systems that work together**.
- They are usually **connected by sharing a storage-area network**, and provide a **high-availability service which survives failures**.
  - **Asymmetric Clustering** has **one machine** in **hot-standby mode**.
  - **Symmetric Clustering** has **multiple nodes running applications** and monitoring each other.

Time Slicing

- To allow a **single processor** to execute **several processes "at the same time"** we use **time slicing**.
- **Time slicing** gives each process **a specific amount of time to execute it's tasks** and then moves on to the next.

Dual-Mode Operation

- **Dual-Mode Operation** allows the **operating system to protect itself and other system components**.
- **Dual-Mode Operation** consists of **two modes**:
  1. **User Mode**, which allows users to perform regular computational tasks.
  2. **Kernel Mode**, which allows to kernel to perform privileged operations.

- To facilitate the modes a **mode bit is provided by the hardware**, giving the ability to determine when the system is running **user code**, or **kernel code**.
- When **user programs** invoke a **system call**, the **kernel mode is activated**. When the **system call returns**, the **mode is reset to user mode**.
- It is **not possible** for **user programs** to manually **change the mode bit**. It can only be performed by **the kernel**.