Introduction to Operating Systems

Andrés Ponce

September 18, 2020

Introduction

An **operating system**(OS) is a piece of software that lies between the hardware and software layer inside a computer. The OS helps the machine allocate resources to processes that require them.

The OS allocates resources to different processes on a computer.

Interrupts

When we have an I/O request, the device controller will receive a signal to load and move the data to the CPU or main memory. When this process is done, the controller will send a signal to the CPU informing of a successful completion of the transfer.

There are wother ways of triggering a system interrupt. Sometimes the program might create an interrupt, known as a **software interrupt**. Other times, the hardware itself might create an interrupt. When this happens, there is a wire connected to the CPU which is directly activated.

Computer System Architecture

In days past, the comptuer was able to run on a single **core**, which executed a general instruciton set needed by application programs. There could also be some smaller and different processors such as **controllers**. [1]

However, recently there are multicore systems, which have more than one physical core. This means that they can execute more than one instruction at at a time. However, it depends on whether there are any resources to allocate available.

There are a couple useful distinctions:

- Random Access: This type of memory allows any part of its data to be accessed without having to first access the other parts. This memory is *volatile*, which means it will be wiped clean next time the computers boot.
- Solid State: A solid state drive(SSD) uses solid state technology and removes any moving components from the drive. This allows for faster read speeds since we don't have to wait for the platter to spin in the correct decision under the read/write head.

[1] A **controller** is a type of device which handles taking information from I/O devices and getting it to the CPU. A controller usually has its own buffer, where it stores the data it is responsible for fetching before sending it to memory or CPU.

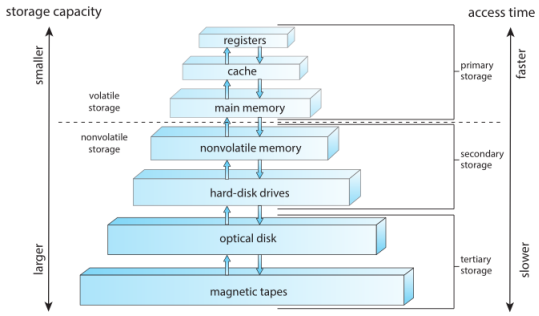

Figure 1: Memory hierarchy. As the speed of the memory increases, so does its cost. So we can only include a limited amount of very quick memory. Cache and registers usually are very limited and are located directly in the CPU.