Cheatography

os Cheat Sheet

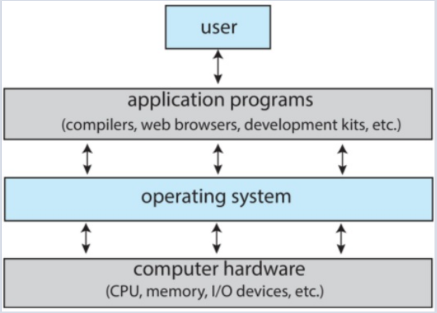
by ggravity via cheatography.com/211480/cs/45823/

lecture 1

computer system (layered architecture): consist of 4 components (user -> applicatoin program -> Operating system -> computer hardware) *Components communicate with their immediate neighbors only*

Roles of OS 1) Resource management: os = software that manage a computer's hardware 2) Serivices: act as interface application/user program <-> computer hardware

computer system


user
application programs
(compilers, web browsers, development kits, etc.)
operating system
computer hardware
(CPU, memory, I/O devices, etc.)

Hardware Organization

Processor: Controls the operation of the computer and performs its data processing functions. When there is only one processor, it is often referred to as the central processing unit (CPU).

Main memory: Stores data and programs, *volatile (lost power lost data)*

I/O modules (devices and controllers): Move data between the computer and its external environment.

System bus: for communication

CPU core (1 CPU =1 calculation)

A processor might consist of one core or many cores.
• Core: The basic component unit of the CPU to execute instructions.
• One core is needed to execute an instruction per time.
If there are N cores, we can executes up to N instructions at a time (called parallelism).

Running a computer create Process

Process = abstraction of executing a program (by the CPU)

computer working because of a process

Storage Structure

Electrically erasable programmable read-only memory (**EEPROM**) stores a bootstrap program, which loads the operating system when the computer is turned on.
Memory (including registers, cache, main memory) stores the ongoing instructions (codes) and temporary data that the CPU is executing. *store processes* , [volatile]
Secondary storage : stores programs & data [nonvolatile]
Tertiary storage : refers to any special proposed storages e.x., CD-ROM, magnetic tape [nonvolatile]

Hardware Interrupt/Polling

Interrupts: I/O Devices will signal the CPU. The CPU is free to do other work until signaled.
-> increasing CPU Utilization
Polling(without interrupt): The CPU repeatedly asks devices if they are done. The CPU is tied to the polling loop.
-> waste CPU cycles
in comparison with using *Interrupts* finished earlier

Multiprogramming and Multitasking

Multitasking [time-sharing]: CPU switches frequently from executing one process to executing another process,

Multiprogramming: maximize CPU utilization, keep the CPU busy at all times.

Dual-mode

Kernel mode(0) -> run OS , can access to ALL hardware (completely control the computer))

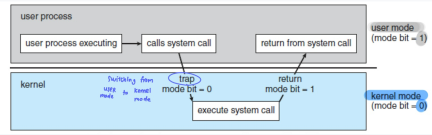
User mode(1) -> run application/user program, NOT allowed to access to hardware and system resources.

trap AKA. software interrupts =exception where the computer system switches from user mode -> kernel mode [systemcall]

context switching= CPU will switch executing from one process to another process

timer interrupt= Prevents user program to run forever.

Dual-mode



OS structure

Simple: most basic structur, Everything runs in kernel mode. ex. MS-DOS (in its early versions)supporting a single task/process

Monolithic: all OS services into a *single* kernel space ex. Linux
pros; communicate between components is fast -> efficient
Cons: Large kernel size.

Q&A

How user program use hardware

> the app make privileged jobs -> create system call to OS in kernel -> OS do that job

???

Tedious mechanism =

System call = when the CPU executes an instruction related to privileged job

previlegde = make change on I/O devices