Lecture1 Introduction to Operating System

0. Course Introduction

Course Information

- Lecturer: Yinqian ZHANG (yinqianz@acm.org)
- Lab instructor: Yun SHEN (sheny@mail.sustech.edu.cn)
- Open Office Hour: Tues. 14:30 ~ 16:30

Content of This Course

- Theory of operating systems
  - Virtualization, concurrency, persistence
  - Management of CPU, memory, I/O and storage
- Implementation of Operating System
  - Build an operating system kernel by yourself (uCore)
  - Run your OS on (emulated) RISC-V CPU (via QEMU)

Goals of This Course

- Process concepts and CPU scheduling
- Memory hierarchy and memory management
- Process control blocks, system calls, context switching, interrupts, exception control flows
- Process synchronization, inter-process communication, threads
- Multi-threaded programming
- File systems, disk scheduling algorithm, I/O
- Security

Prerequisites

- C Programming
- Data structure and algorithm analysis
- Computer Organization

Reference Books and Organization of this Course

- **Operating System Concepts**, 9th Edition, Abraham Siberschatz et. al. (a.k.a. the Dinosaur Book)
  - Organized by the resources OS manages

        - CPU management (Processes, synchronization, scheduling, deadlocks)
        - Memory management (Physical memory, virtual memory)
        - I/O management (I/O subsystems, storage and file systems)
- **Operating Systems: Three Easy Pieces**, Remzi H. ArpaciDusseau and Andrea C. Arpaci-Dusseau
    - https://pages.cs.wisc.edu/~remzi/OSTEP/
    - Organized by the functionalities of OS
        - Virtualization (Process, scheduling, memory address space, swapping)
        - Concurrency (Threads, locks, semaphores)
        - Persistent (I/O, storage, file systems)

Gradings

- Lecture participation: 10%
    - Get 100 points in quiz
- Lab participation: 15%
    - In-class assignments
- Assignment: 40%
    - Written and coding
- Mid-term exam: 15%
- Final exam: 20%

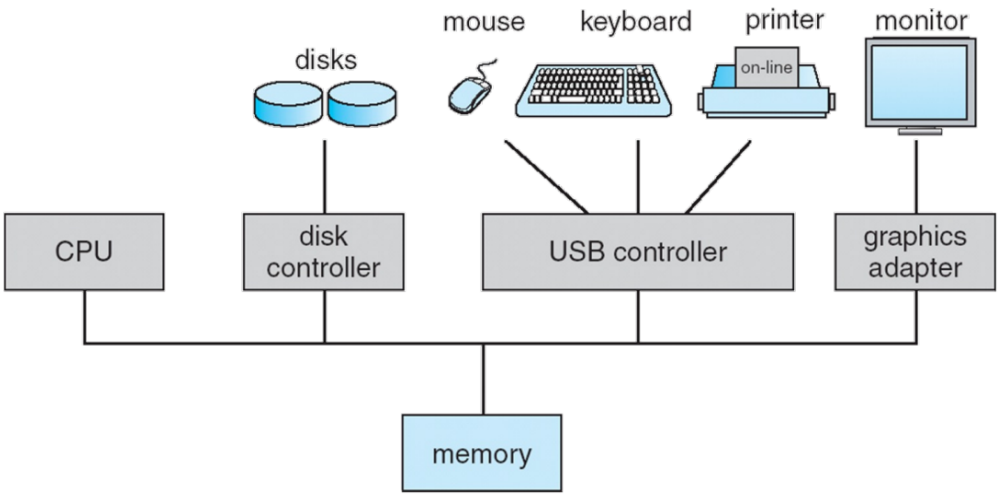
How to prepare

- Come to class
    - Content in exams $\subseteq$ what I explain in class $\supseteq$ content on slides $\nsubseteq$ textbook
- Visit Sakai
- Prepare to code
    - You will build your own OS kernel

1. What is a computer?

[Informal definition] A computer is a digital electronic machine that can be **programmed** to carry out **sequences of arithmetic or logical operations automatically**.

Computer System Organization

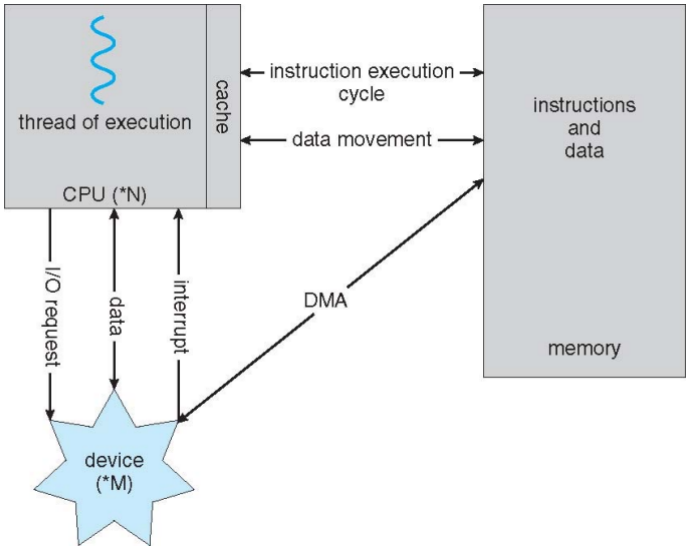


- One or more CPUs, device controllers connect through common bus providing access to **shared memory**
- **Concurrent**(并发的) **execution** of CPUs and devices competing for memory cycles

How a Modern Computer Works - von Neumann Architecture

John von Neumann (1903~1957)

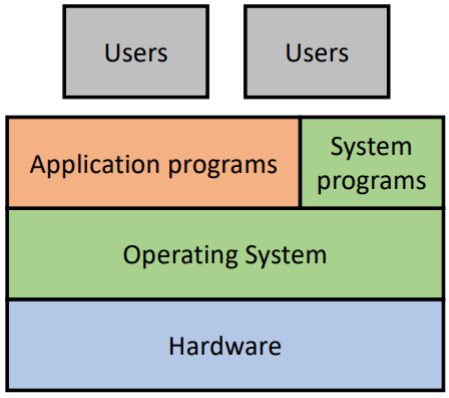
- Mathematician, computer scientist, physicist, chemist
- Known for invention of modern computer architecture and game theory



- a **single**, **shared memory** for programs and data
- a **single bus** for memory access
- an **arithmetic** unit
- and a **program control unit**

Structure of a Computer System

Computer system can be divided into **four** components



Components	Functions
Hardware	Provides basic computing resources CPU, memory, I/O devices
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

2. What is an operating system?

Definition

> Operating system was once called **supervisor** or **master control program**

A group of software that makes the computer operate **correctly** and **efficiently** in an **easy-to-use** manner.

- Execute user programs and make solving user problems **easier**
- Make the computer system convenient to use
- Use the computer hardware in an efficient manner (**hardware abstraction**)
    - it provides **abstraction** of the hardware

- macOS
- Windows
- Linux

More function of OS system

- It includes a **software program** called **kernel**
    - manages all the **physical devices** (e.g., CPU, RAM and hard disk)
    - exposes some functions such as **system calls**(系统调用) for others to configure the kernel or build software (e.g., C library) on top
- It includes other ”helper” programs
    - **shell**: render a simple command-line user interface with a full set of commands
    - **GUI** (Graphic User Interface): render a user-friendly interface with icons representing files and folders
    - **Browser**: visit websites (it depends whether it is part of OS or not)

- It is a **resource manager**
  - Managing CPUs, memory, disks, I/O devices (keyboards, USB drive, sensors, …)
  - Arbitrator(仲裁人) of conflicting requests for efficient and fair resource use
- It is a **control program**
  - Controls execution of programs to prevent **errors** and **improper use** of the computer
    - error: prevent from crashing
    - improper use: unsecure applications

What does an OS do - Functionalities

- **Virtualization**
  - Virtualize CPU: Run **multiple programs on a single CPU** (as if there are many CPUs)
  - Virtualize memory: Give each **process** (or programs if you will) the illusion of **running in its own memory address space**
- **Concurrency**
  - Run **multi-threaded programs** and make sure they execute correctly
- **Persistence**
  - Write data (from volatile SRAM/DRAM) into **persistent storage**
  - Performance, crash-resilience

3. History of OS

Evolution of OS

- **Early OS**: A library to handle low-level I/O
- **Atlas computing system**: **system calls** that raise the hardware privilege level
- **UNIX**: support of multi-programming and memory protection
- **PC era**: better security and useability
  - Disk Operating System (DOS), Mac OS, Windows, Linux
- **Smart phones**: user-facing applications, more sensors
  - iOS, Android, …

Brief History of UNIX

- Influenced by Multics system from MIT
- Originally by Ken Thompson and Dennis Ritchie at Bell Labs
    - Support meta-level programming with shell and pipe
    - Written in easy-to-understand C programming language
- Evolves to Berkeley Systems Distribution (BSD)
    - Advanced virtual memory, file system, and networking subsystems
- Commercial versions of UNIX
    - SunOS from Sun Microsystems, AIX from IBM, HPUX from HP, and IRIX from SGI.
- Mac OS has UNIX at its core
- Ideas and principles of UNIX inspire Linus Torvalds
    - The Linux Operating System

Brief History of Windows

- Windows 1 (1985): Graphic user interface on MS-DOS
- Windows 2 (1987): Support overlapping windows
- Windows 3 (1990): Run MS-DOS programs on Windows
- Windows 3.1 (1992): TrueType fonts support
- Windows 95 (1995): Start menu and button
- Windows 98, ME, 2000, XP, Vista, 7, 8, 8.1, 10

4. OS Concept

Process (进程)

- A process is a **program in execution**
    - Program is a **passive** entity and process is an **active** entity
- Process needs resources to accomplish its task
    - CPU, memory, I/O, files
    - Process termination requires reclaim(回收) of any reusable resources
- Process executes instructions **sequentially**, one at a time, until completion
    - Single-threaded process has **one program counter** specifying location of next instruction to execute
    - Multi-threaded process has **one program counter per thread**
- Typically, system has many processes, some user, some operating system running concurrently on one or more CPUs

    - Concurrency by multiplexing(多路复用) the CPUs among the processes/threads

Process Management

- Creating and deleting both user and system processes
- Suspending(挂起) and resuming processes
- Providing mechanisms for process synchronization
- Providing mechanisms for process communication
- Providing mechanisms for deadlock handling

Memory

- **DRAM (Dynamic Random Access Memory)** is the **main memory** used for all desktop, laptops, servers, and mobile devices
- **CPU** only directly interacts with the **main memory** during execution
    - All data in memory before and after processing
    - All instructions in memory in order to execute
- OS **manages the main memory for kernel and processes**
    - OS dictates(规定) which process can access which memory region

Memory Management

- Memory management determines what is in memory when
    - Optimizing CPU utilization and computer response to users
- Memory management activities
    - Keeping track of which parts of memory are currently being used and by whom
    - Deciding which processes (or parts thereof) and data to move into and out of memory
    - Allocating and deallocating memory space as needed

Storage Management

- OS provides uniform, logical view of information storage
    - **Abstracts physical properties** to logical storage unit - **file**
    - Each **medium** is controlled by **device** (i.e., disk drive, tape drive)
        - Varying properties include access speed, capacity, data-transfer rate, access method (sequential or random)
- File-System management
    - Files usually organized into **directories**
    - **Access control** on most systems to determine who can access what

- OS activities include
    - Creating and deleting files and directories
    - Primitives to manipulate files and directories
    - Mapping files onto secondary storage
    - Backup files onto stable (non-volatile) storage media

I/O Subsystem

- One purpose of OS is to hide peculiarities(特性) of hardware devices from the user
- I/O subsystem responsible for
    - Memory management of I/O including
        - **buffering** (storing data temporarily while it is being transferred)
        - **caching** (storing parts of data in faster storage for performance)
    - General device-driver interface
    - Drivers for specific hardware devices

Protection and Security

- **Protection**: any mechanism for **controlling access** of processes or **users to resources** defined by the OS
- **Security**: defense of the system against internal and external attack
    - Huge range, including denial-of-service, worms, viruses, identity theft, theft of service
- OS determines which users can do what
    - **User identities** (user IDs, security IDs) include name and associated number, one per user
        - User ID then associated with all files, processes of that user to determine access control
    - **Group identifier** (group ID) allows set of users to be defined and controls managed, then also associated with each process, file
    - Privilege escalation(提升) allows user to change to effective ID with more rights