Operating Systems

Chapter 2 Operating System Overview

Agenda

2.1 Operating System Objectives and Functions

2.2 The Evolution of Operating Systems

2.3 Major Achievements

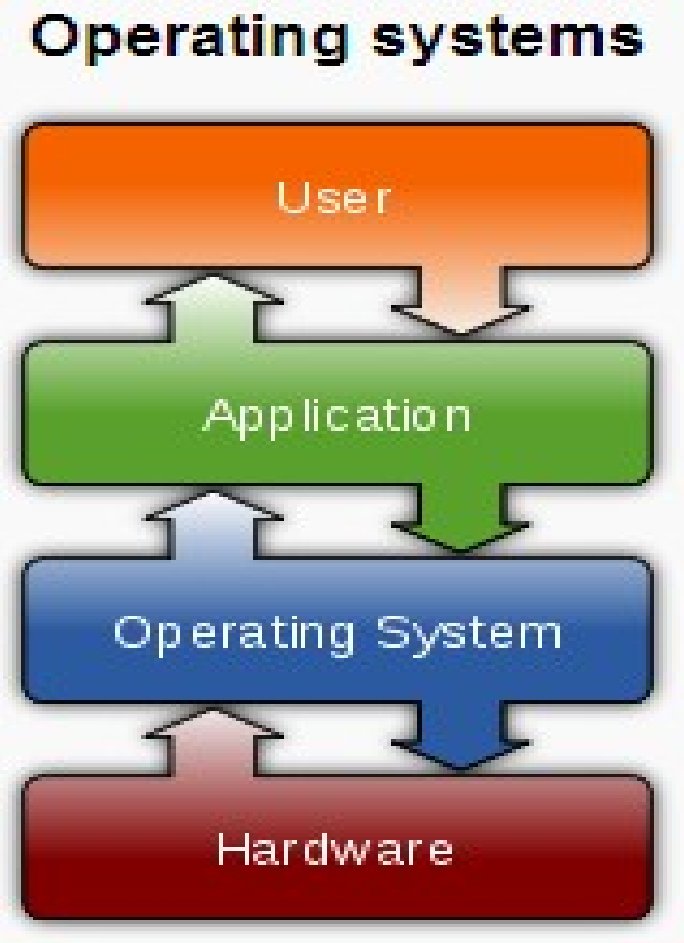
2.4 Developments Leading to Modern Operating Systems

2.5 Load of APP & OS

2.6 Other topic

2.1Operating System Objectives and Functions (1/6)

- **Operating System definition**: A program that
  - controls the execution of application programs
  - Acts as an interface between applications and hardware



2.1Operating System Objectives and Functions (2/6)

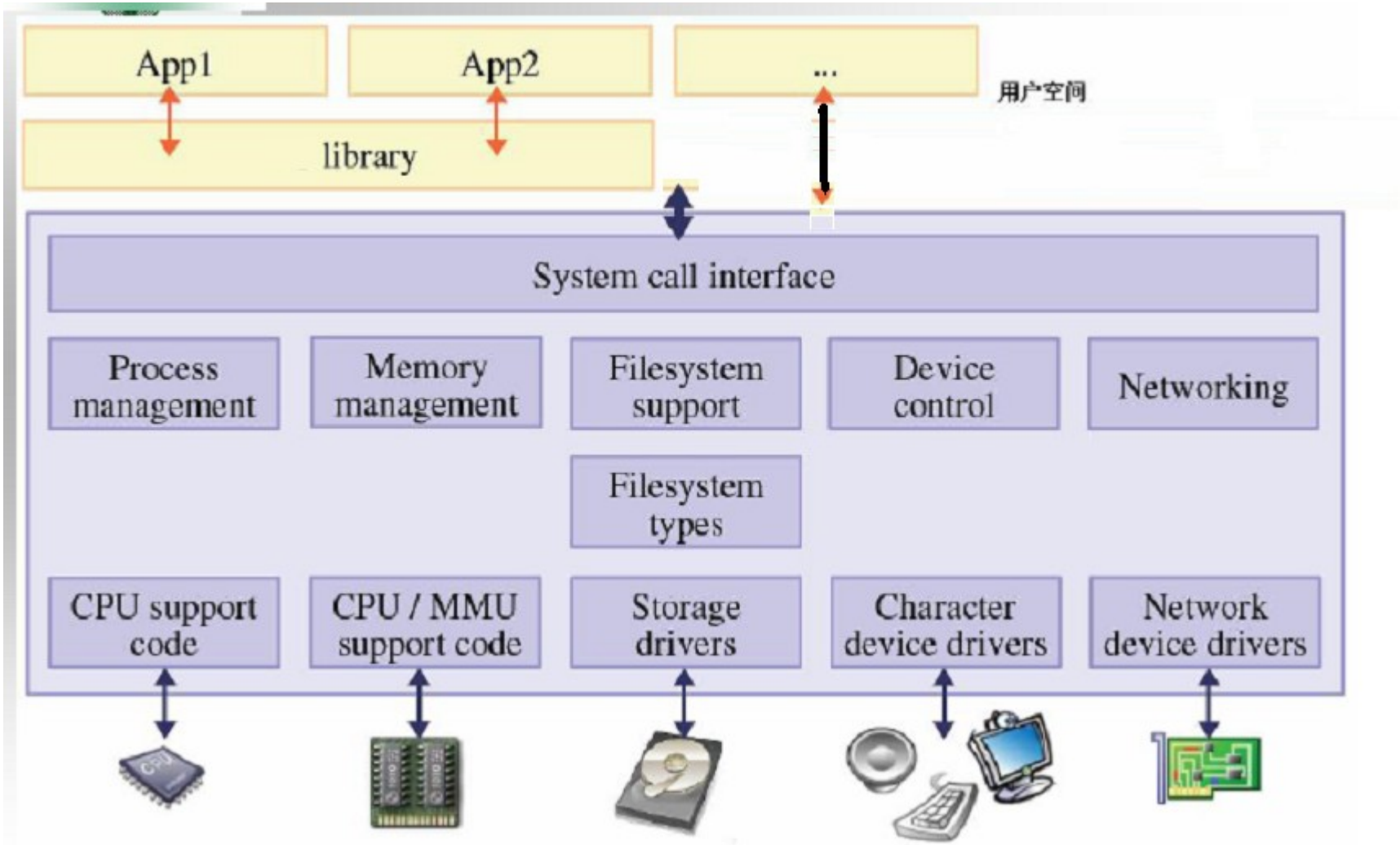
- The Operating System as Resource Manager
  - Responsible for managing resources/Hardware
  - Work in the same way as ordinary computer software
    - It is program that is executed
  - Operating system frequently relinquishes(放弃) and regains control of the processor

2.1Operating System Objectives and Functions (3/6)

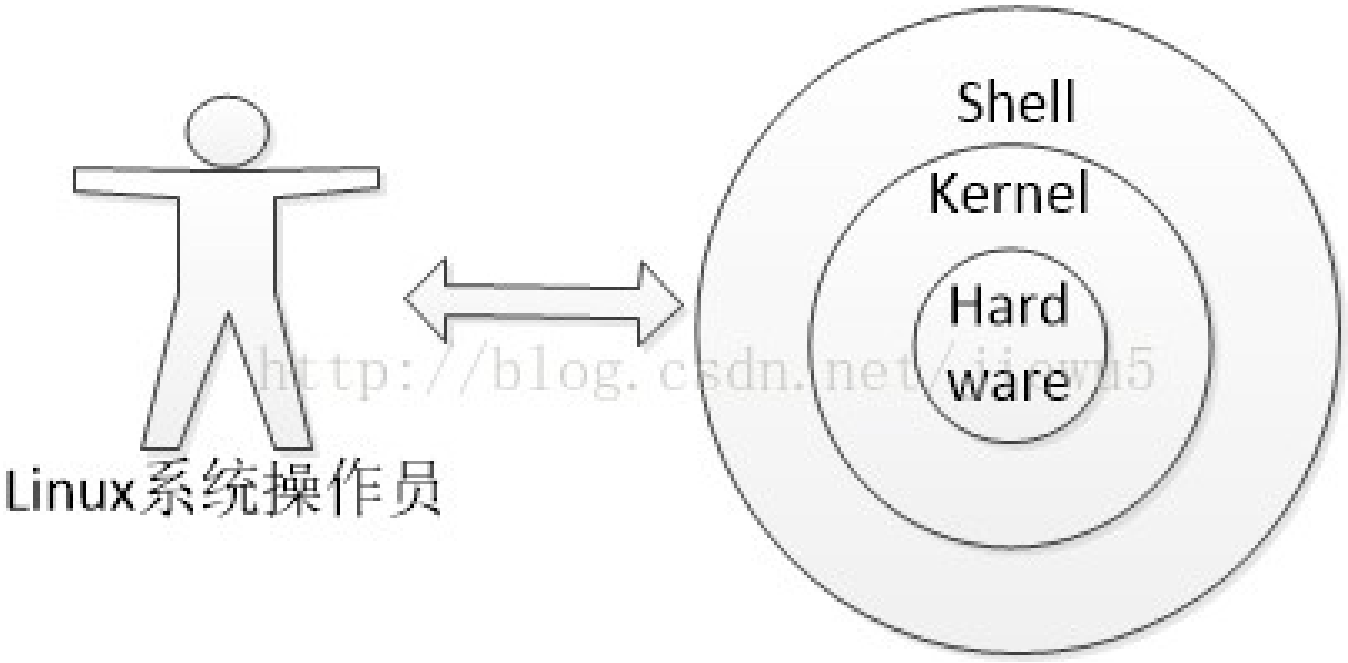
- **Operating System Objectives**
  - As User/Computer Interface --Convenience/ 方便
  - As Resource Manager--Efficiency / 有效
  - As System Software--Ability to evolve / 扩展
    - Hardware upgrades plus new types of hardware
    - New services ： Windows Subsystem for Linux
    - Fixes Bug
    - How ： Modularization / OS Architecture

- 2.1Operating System Objectives and Functions (4/6)

Compare with Fig2.2 on textbook



2.1Operating System Objectives and Functions (5/6)



2.1Operating System Objectives and Functions (7/6)

- Shell(外壳)
  - CLI :command line interface
  - GUI:graphical user interface
- Kernel(内核)
  - Portion of operating system that is in main memory
  - Contains most frequently used functions
  - Also called the nucleus （ 核子 ）
    - Linux ubuntu 4.15.0-142-generic #uname -a
      - 第一个组数字： 4, 主版本号
      - 第二个组数字： 15, 次版本号，当前为稳定版本
      - 第三个组数字： 0, 修订版本号
      - 第四个组数字： 142 ，当前内核版本（ 4.15.0 ）的第 142 次微调 patch

Agenda

2.1 Operating System Objectives and Functions

2.2 The Evolution of Operating Systems

2.3 Major Achievements

2.4 Developments Leading to Modern Operating Systems

2.5 Load of APP & OS

2.6 Other topic

2.2 The Evolution of Operating Systems(1/17)

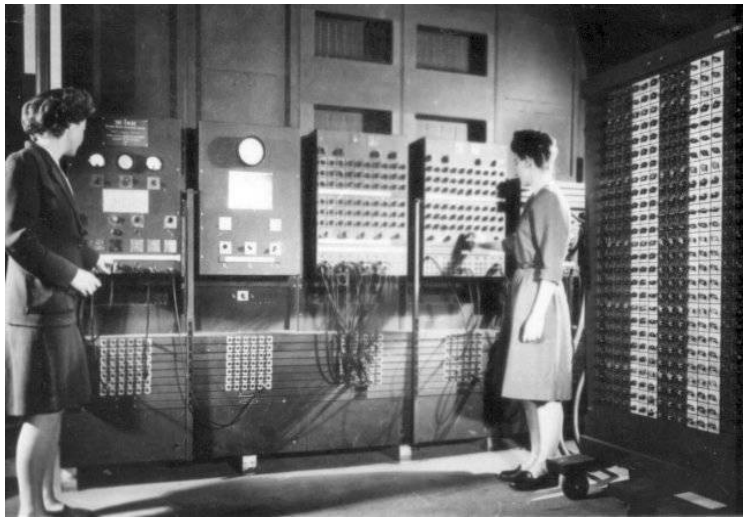
- Why learn it?
  - This history is itself interesting and also serves the purpose of providing an overview of OS principles.
  - In attempting to understand the key requirements for an OS and the significance of the major features of a contemporary OS, it is useful to consider how operating systems have evolved over the years

2.2 The Evolution of Operating Systems(2/17)

- 阶段 1 ：提高昂贵计算机的使用率
  - 1 Serial Processing: No Operating Systems 串行处理
  - 2 Simple Batch Systems 简单批处理
  - 3 Multiprogrammed Batch Systems 多道批处理
  - 4 Time-Sharing Systems 时分系统
- 阶段 2 ： PC 普及
  - 易用性 GUI/ 安全性
- 阶段 3 ：网络普及
  - 分布式系统
- Future
  - 服务云计算，大数据

2.2 The Evolution of Operating Systems(3/17)

- 1 Serial Processing: No Operating Systems



- When 1940s-1950s
- How it work
- Features
- Problems

- 操作 ENIAC

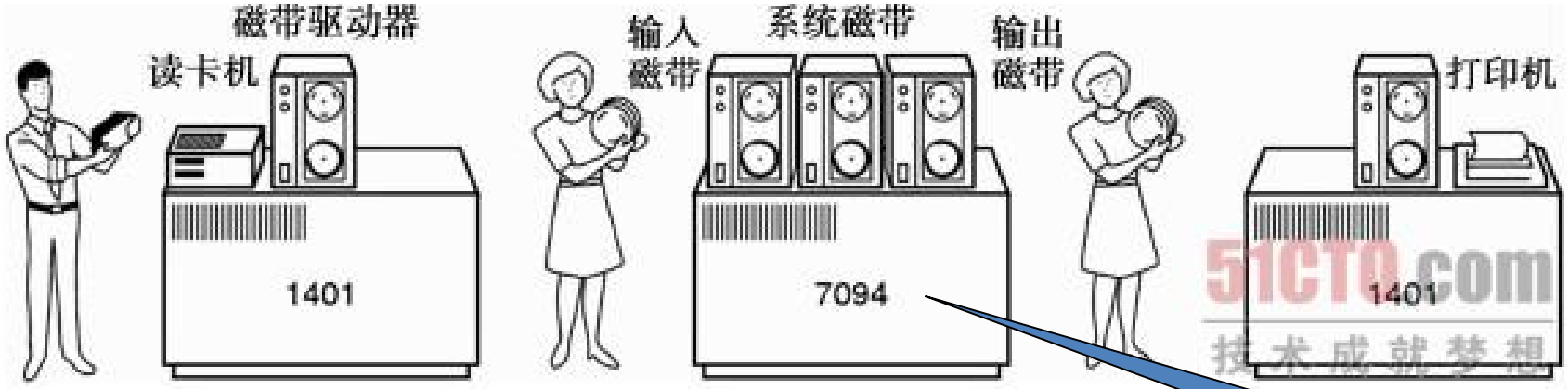
2.2 The Evolution of Operating Systems(4/17)

- 1 Serial Processing: No Operating Systems

  - 命名： the user have access to the system in series
  - 特点： the programmer interacted directly with the computer hardware
  - 如何操作： Machines run from a console with display lights, toggle switches 转换开关 , input device, and printer
  - 问题
    - Scheduling: Most installations used a hardcopy sign-up sheet to reserve computer time
    - Setup time: included loading the compiler, source program, saving compiled program, and loading and linking

2.2 The Evolution of Operating Systems(5/17)

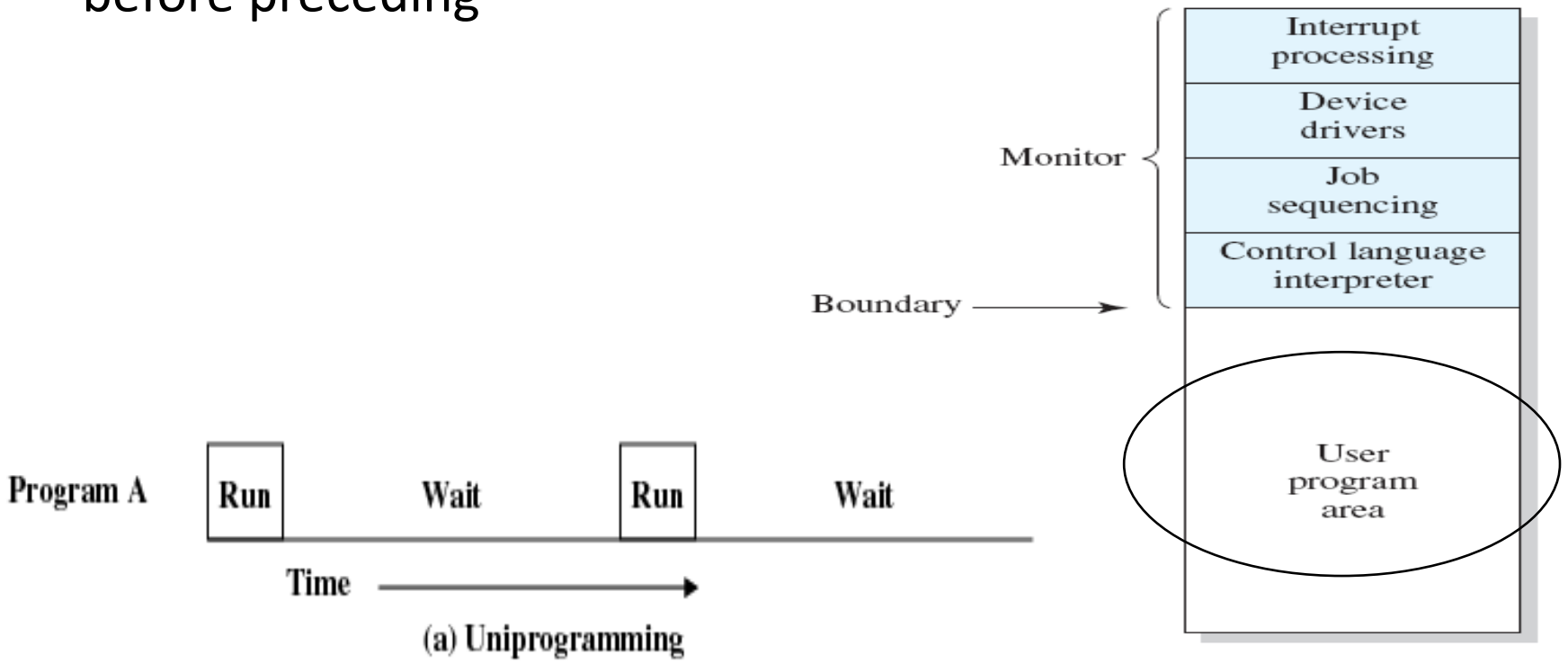
- 2 Simple Batch Systems
  - Feature: without interactive from users
  - When 1950s-1960s



IBM 7094 简单 / 单任务批处理系统

2.2 The Evolution of Operating Systems(6/17)

- Simple Batch Systems
  - Only one job in memory at a time
  - Processor must wait for I/O instruction to complete before preceding



2.2 The Evolution of Operating Systems(7/17)

- The central idea is
  - Batch: made up by many jobs from users
    - A job may use several program
  - the use of a piece of software known as the monitor that controls the sequence of events.
- The Process
  - The monitor reads a job from the batch
  - Then control is passed to this job
  - When the job is completed, it returns control to the monitor
  - The monitor reads in the next job

2.2 The Evolution of Operating Systems(8/17)

- Job Control Language (JCL)
  - Special type of programming lang
  - Provides instruction to the monit
    - What compiler to use
    - What data to use
    - …….

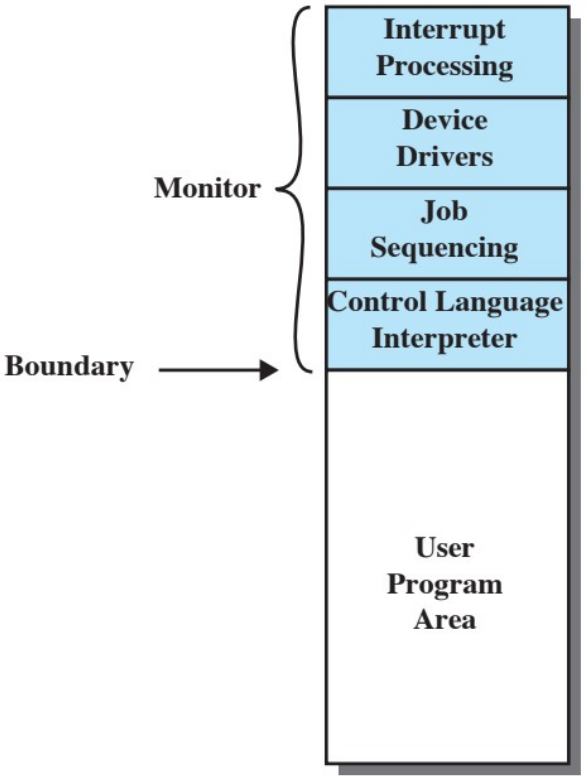


Figure 2.3 Memory Layout for a Resident Monitor

2.2 The Evolution of Operating Systems(9/17)

- New: CPU mode
  - User program executes in **user mode(用户模式)**
    - Certain instructions may not be executed
    - Some memory can not be accessed
  - Monitor executes in **system mode(系统模式)**
    - **Kernel mode(内核模式)**
    - Privileged instructions are executed
    - Protected areas of memory may be accessed

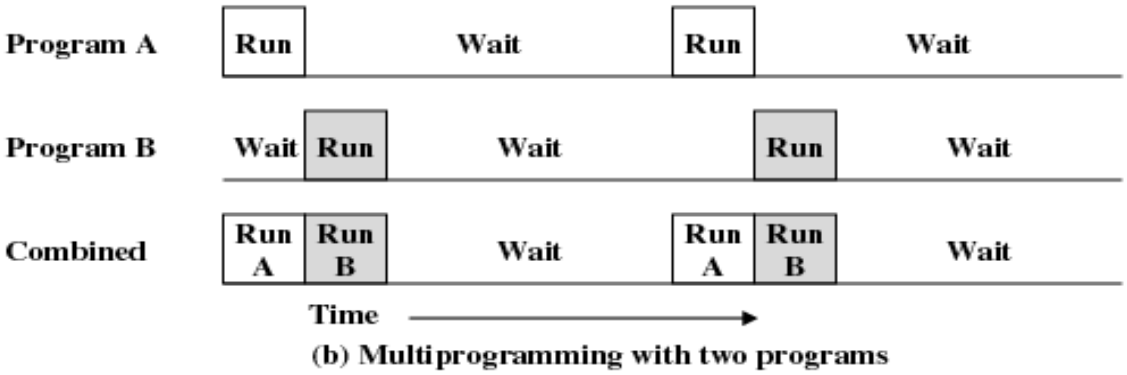
- Disadvantage?

2.2 The Evolution of Operating Systems(10/17)

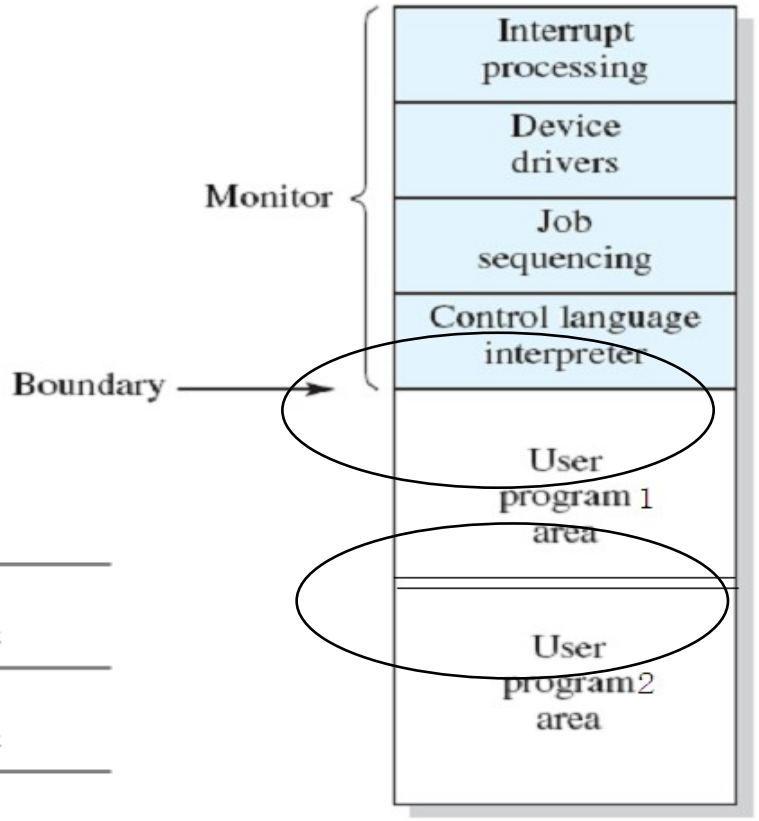
- Hardware Features
  - Memory protection
    - Do not allow the memory area containing the monitor to be altered
  - Timer
    - Prevents a job from monopolizing 独占 the system
  - Privileged instructions
    - Certain machine level instructions can only be executed by the monitor
  - Interrupts
    - This feature gives the OS more flexibility in relinquishing 放弃 control to and regaining control from user programs

2.2 The Evolution of Operating Systems(11/17)

- 3 Multiprogrammed Batch Systems
  - Multiple jobs in memory at a time
- Why ?
  - I/O Devices Slow
    - CPU not Busy



(b) Multiprogramming with two programs



2.2 The Evolution of Operating Systems(12/17)

- Example: 3 个作业 / 程序
  - 各自情况描述 : MM 250MB

Table 2.1 Sample Program Execution Attributes

	JOB1	JOB2	JOB3
Type of job	Heavy compute	Heavy I/O	Heavy I/O
Duration	5 min	15 min	10 min
Memory required	50 M	100 M	75 M
Need disk?	No	No	Yes
Need terminal?	No	Yes	No
Need printer?	No	No	Yes

2.2 The Evolution of Operating Systems(13/17)

- Example: 3 个作业 / 程序 (cont.)
  - Utilization Histograms(利用率直方图)

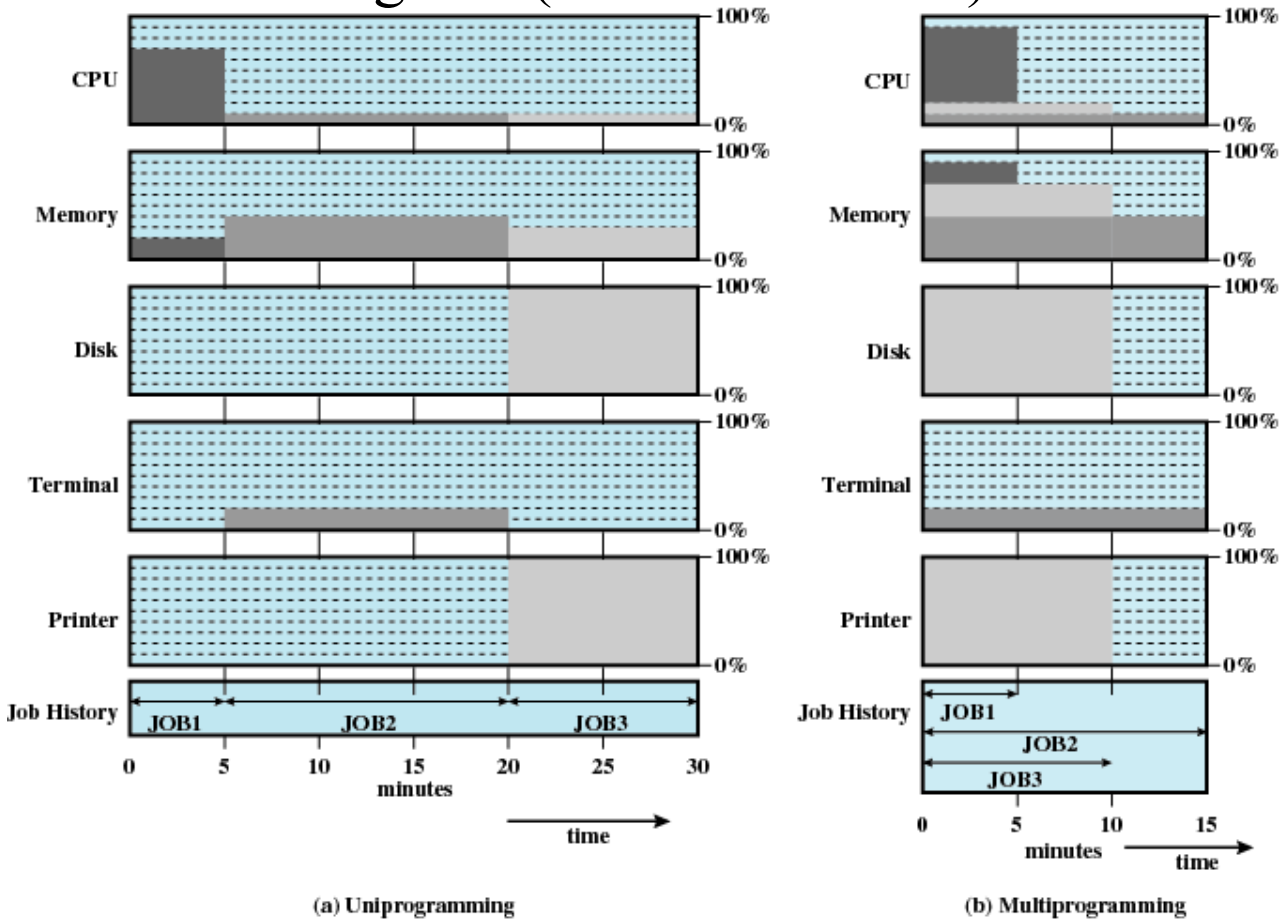


Figure 2.6 Utilization Histograms

2.2 The Evolution of Operating Systems(14/17)

- Example: 3 个作业 / 程序 (cont.)
  - Effects of Multiprogramming on Resource Utilization

Table 2.2 Effects of Multiprogramming on Resource Utilization

	Uniprogramming	Multiprogramming
Processor use	20%	40%
Memory use	33%	67%
Disk use	33%	67%
Printer use	33%	67%
Elapsed time	30 min	15 min
Throughput	6 jobs/hr	12 jobs/hr
Mean response time	18 min	10 min

2.2 The Evolution of Operating Systems(15/17)

- 4. Time-Sharing Systems

- When: 1960s~
- Why?
  - Multiple users simultaneously access the system(mainframe computer 大型主机)through terminals
  - Requirement: handle multiple interactive users/jobs
- Idea
  - Processor's time is shared among multiple users/jobs
    - Time slicing 时间片（技术）
- Products:CTSS 兼容分时系统

2.2 The Evolution of Operating Systems(16/17)

- Batch Multiprogramming VS Time Sharing

Table 2.3 Batch Multiprogramming versus Time Sharing

	Batch Multiprogramming	Time Sharing
Principal objective	Maximize processor use	Minimize response time
Source of directives to operating system	Job control language commands provided with the job	Commands entered at the terminal

2.2 The Evolution of Operating Systems(17/17)

- 当代操作系统实例
  - 封闭微软： DOS, Windows 系列
    - 95,NT,98,me,2000,xp,vista…window 10, window 11
  - 开放 Unix:
    - 主要分支 Unix BSD(Berkeley Software Distribution 版，开源，可商用)
    - 进一步演变 solaris,mac OS
  - 开源 Linux(仿 unix ，但是完全独立)、鸿蒙操作系统
    - redhat, 红旗， ubuntu, openeuler, 麒麟
    - 继续扩展变化 Android

Agenda

2.1 Operating System Objectives and Functions

2.2 The Evolution of Operating Systems

2.3 Major Achievements

2.4 Developments Leading to Modern Operating Systems

2.5 Load of APP & OS

2.6 Other topic

2.3 Major Achievements(1/12)

1 The Process

2 Memory Management

3 Information Protection and Security

4 Scheduling and Resource Management

5 System Structure

2.3 Major Achievements(2/12)

- 1. Processes

- Processes consists of three components
  - An executable program /code
  - Associated data needed by the program
  - Execution context of the program (the core)
    - All information the operating system needs to manage the process
- The process is realized/implemented as a data structure
  - All information the operating system needs to manage the process is stored in the data structure

2.3 Major Achievements(3/12)

- An example

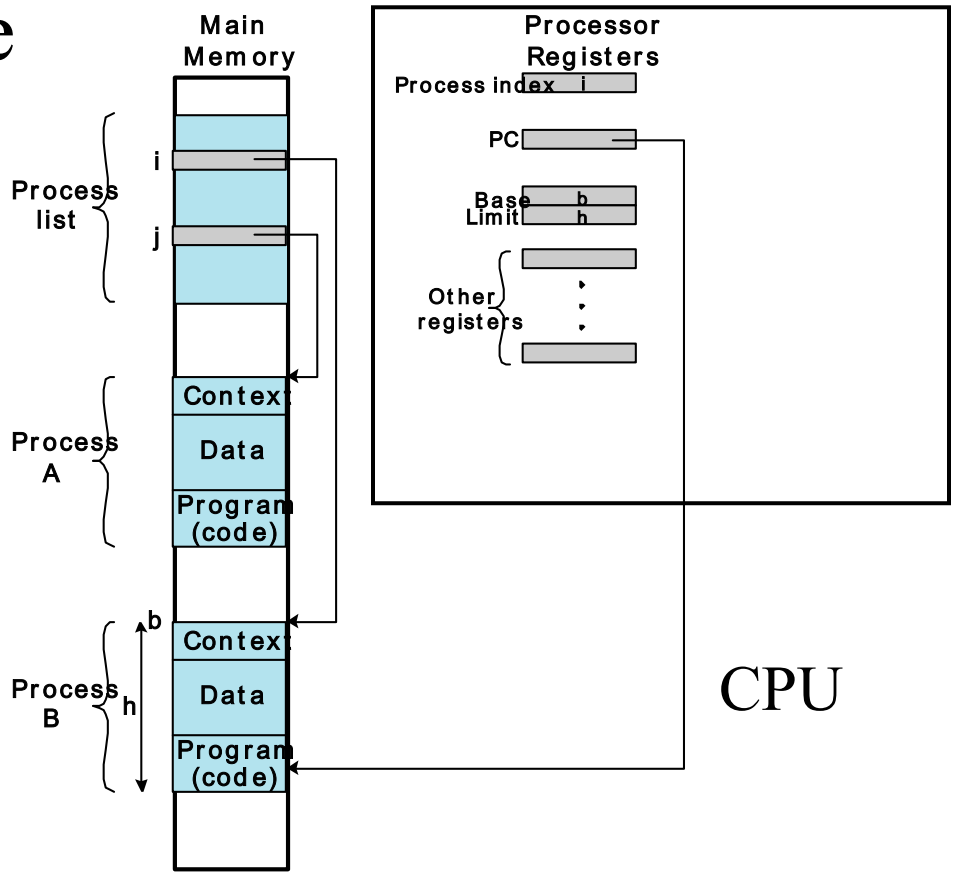


Figure 2.8 Typical Process Imple

2.3 Major Achievements(4/12)

- 2. Memory Management

- OS has five storage management responsibilities:
  - Process isolation(进程隔离)
  - Automatic allocation and management(自动分配和管理)
  - Support of modular programming(模块化程序设计)
  - Protection and access control (保护与存取控制)
  - Long-term storage(长期存储)

2.3 Major Achievements(5/12)

- Virtual Memory / VM
  - Allows programmers to address memory from a logical point of view
  - No hiatus(脱节) between the execution of successive processes while one process was written out to secondary store and the successor process was read in
- Page
  - Allows process to be comprised of a number of fixed-size blocks, called pages
  - Each page may be located any where in main memory

2.3 Major Achievements(6/12)

- Virtual Memory Addressing
  - Storage system consists of main memory and auxiliary 辅助 memory

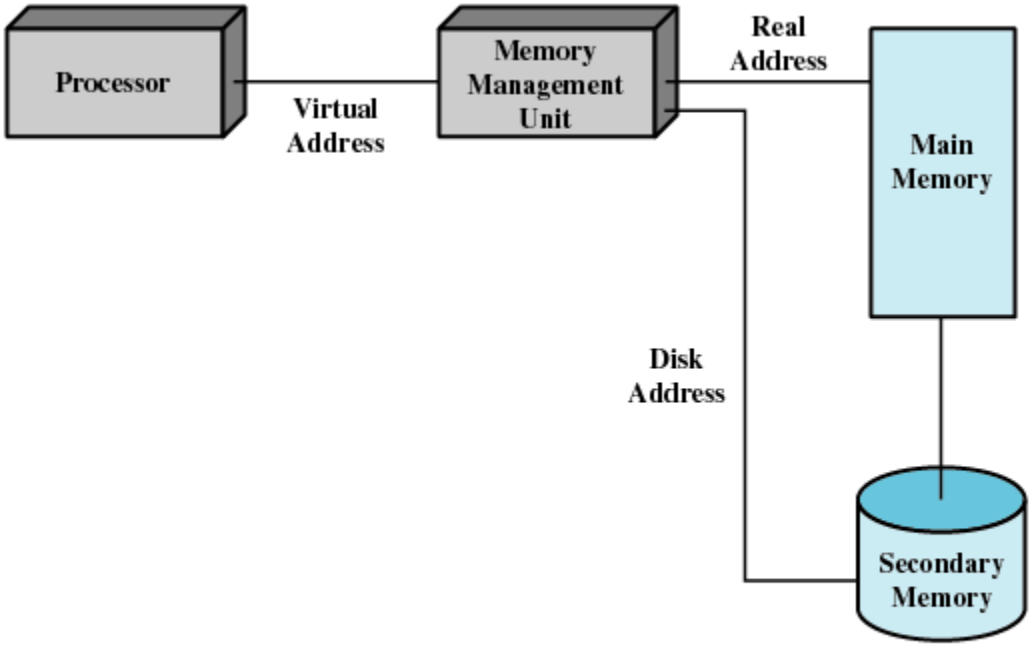


Figure 2.10 Virtual Memory Addressing

2.3 Major Achievements(7/12)

- 3. Information Protection and Security

- Availability 不被中断
- Confidentiality 保密性，授权
- Data integrity 不被未授权篡改
- Authenticity 认证

2.3 Major Achievements(8/12)

- 4. Scheduling and Resource Management

- Manage the resources (such as CPU, Main Memory, I/O devices)
- Schedule/ 调度 these resources to various active processes/ 进程

2.3 Major Achievements(9/12)

- The factors that should be considered
  - Fairness(公平性)
    - Give equal and fair access to resources
  - Differential responsiveness(有差别的响应性)
    - Discriminate 区别 among different classes of processes
  - Efficiency(高效性)
    - Maximize throughput(吞吐量), minimize response time(响应时间), and accommodate as many users as possible

2.3 Major Achievements(10/12)

- 5. System Structure

- The size and complexity of operating systems has grown.
  - MIT CTSS: 32,000 36-bit words Compatible Time-Sharing System
  - Windows
    - Windows NT 4.0: 16 million lines of code
    - Windows 2000: more than 32 million lines of code
    - Windows XP: 35 million lines of code
    - Windows Vista: 50 million lines of code
  - Linux 2.6.27: 10 million lines of code

2.3 Major Achievements(11/12)

- Led to four unfortunate but all-too-common problems
  - late delivery
  - performance unexpected
  - latent 潜在的 bugs
  - vulnerable 易受攻击的 to a variety of security attacks
- Solution ：
  - The software must be modular and modules must have well-defined interfaces
  - Levels

2.3 Major Achievements(12/12)

Table 2.4 Operating System Design Hierarchy

Level	Name	Objects	Example Operations
13	Shell	User programming environment	Statements in shell language
12	User processes	User processes	Quit, kill, suspend, resume
11	Directories	Directories	Create, destroy, attach, detach, search, list
10	Devices	External devices, such as printers, displays, and keyboards	Open, close, read, write
9	File system	Files	Create, destroy, open, close, read, write
8	Communications	Pipes	Create, destroy, open, close, read, write
7	Virtual memory	Segments, pages	Read, write, fetch
6	Local secondary store	Blocks of data, device channels	Read, write, allocate, free
5	Primitive processes	Primitive processes, semaphores, ready list	Suspend, resume, wait, signal
4	Interrupts	Interrupt-handling programs	Invoke, mask, unmask, retry
3	Procedures	Procedures, call stack, display	Mark stack, call, return
2	Instruction set	Evaluation stack, microprogram interpreter, scalar and array data	Load, store, add, subtract, branch
1	Electronic circuits	Registers, gates, buses, etc.	Clear, transfer, activate, complement

Agenda

2.1 Operating System Objectives and Functions

2.2 The Evolution of Operating Systems

2.3 Major Achievements

2.4 Developments Leading to Modern Operating Systems

2.5 Load of APP & OS

2.6 Other topic

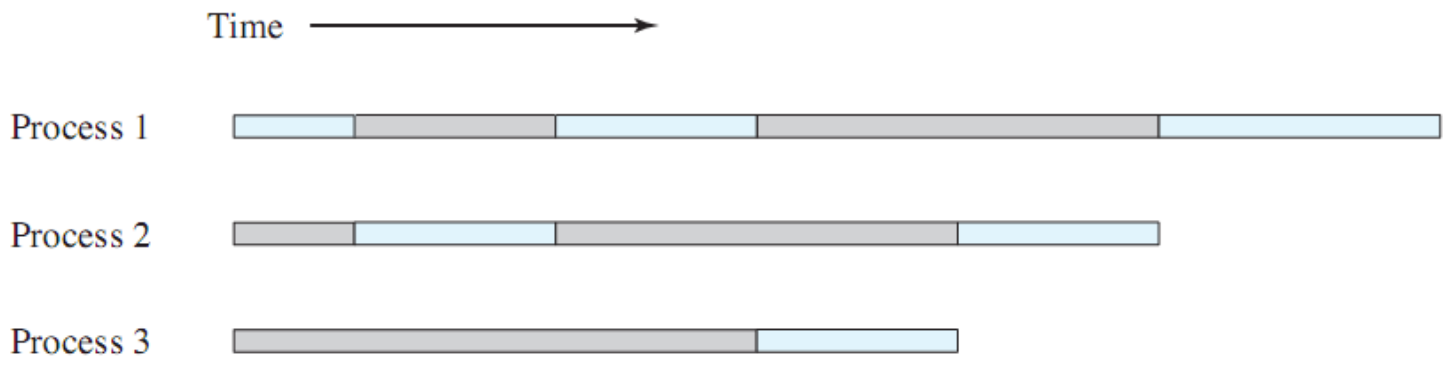
2.4 Modern Operating Systems (1/3)

- Background
  - new developments in hardware
  - new applications
  - new security threats

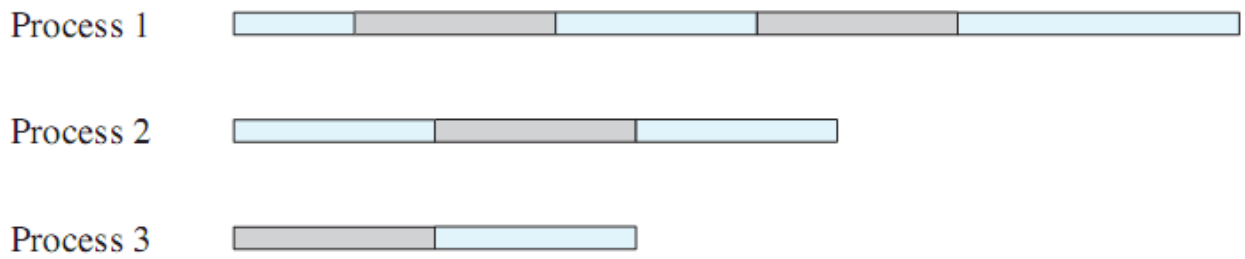
2.4 Modern Operating Systems (1/3)

- Five features are mentioned in this chapter
  - Microkernel architecture
    - 对比 OS 的单内核完成所有管理工作，它含基本功能
  - Multithreading
    - 进程含多线程，资源占用少，调度单位
  - Symmetric multiprocessing (SMP) 对称多处理
    - 一般具有 multiple processors ，且完全相同对等
    - 性能提升，对抗处理器失效，配置灵活
  - Distributed operating systems : chapter 14~15: 不讲
    - 网络一组独立计算机虚拟为用户的一台完整计算机
  - Object-oriented design

2.4 Modern Operating Systems (1/3)



(a) Interleaving (multiprogramming, one processor)



(b) Interleaving and overlapping (multiprocessing; two processors)

Agenda

2.1 Operating System Objectives and Functions

2.2 The Evolution of Operating Systems

2.3 Major Achievements

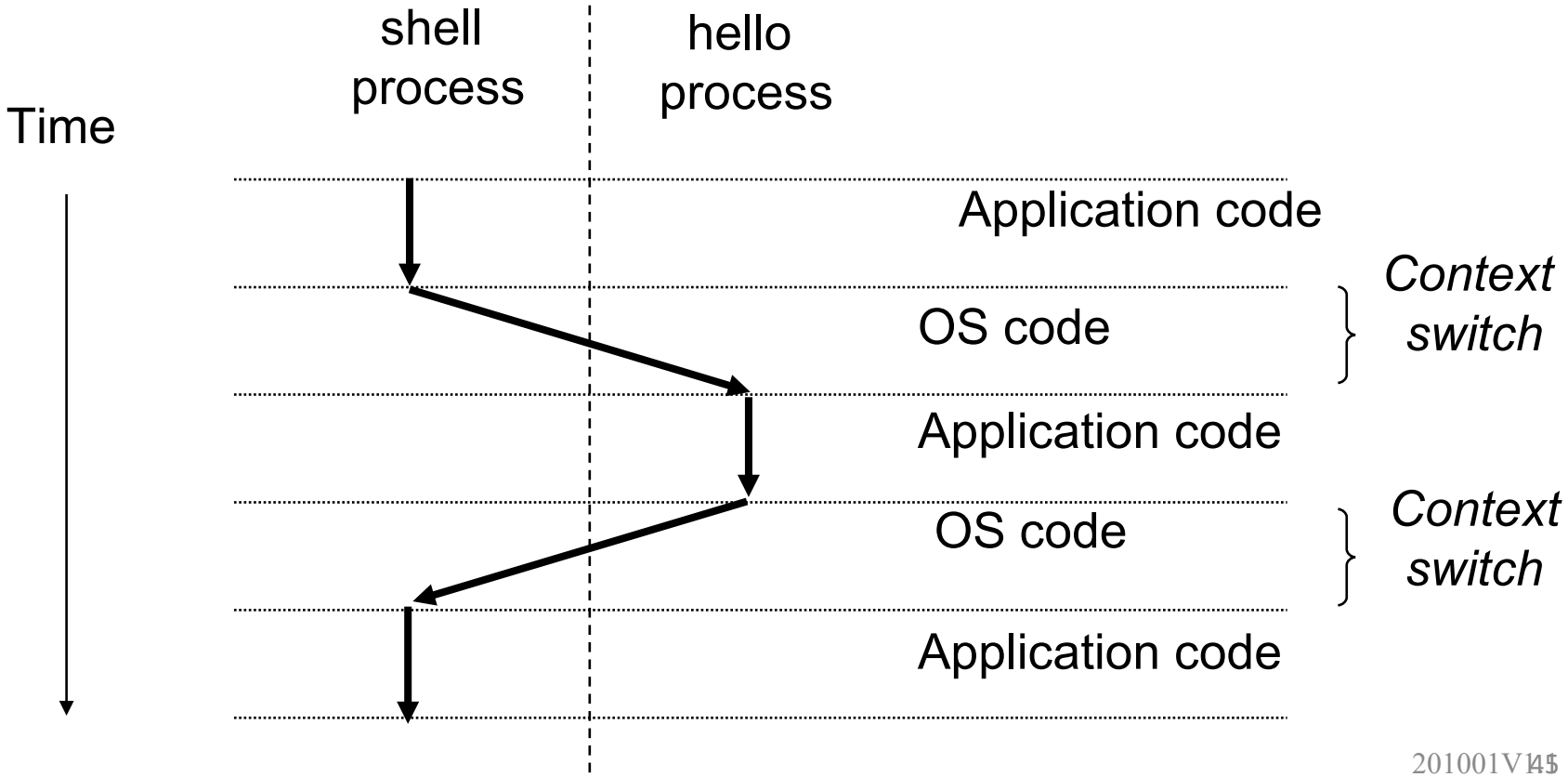
2.4 Developments Leading to Modern Operating Systems

2.5 Load of APP & OS

2.6 Other topic

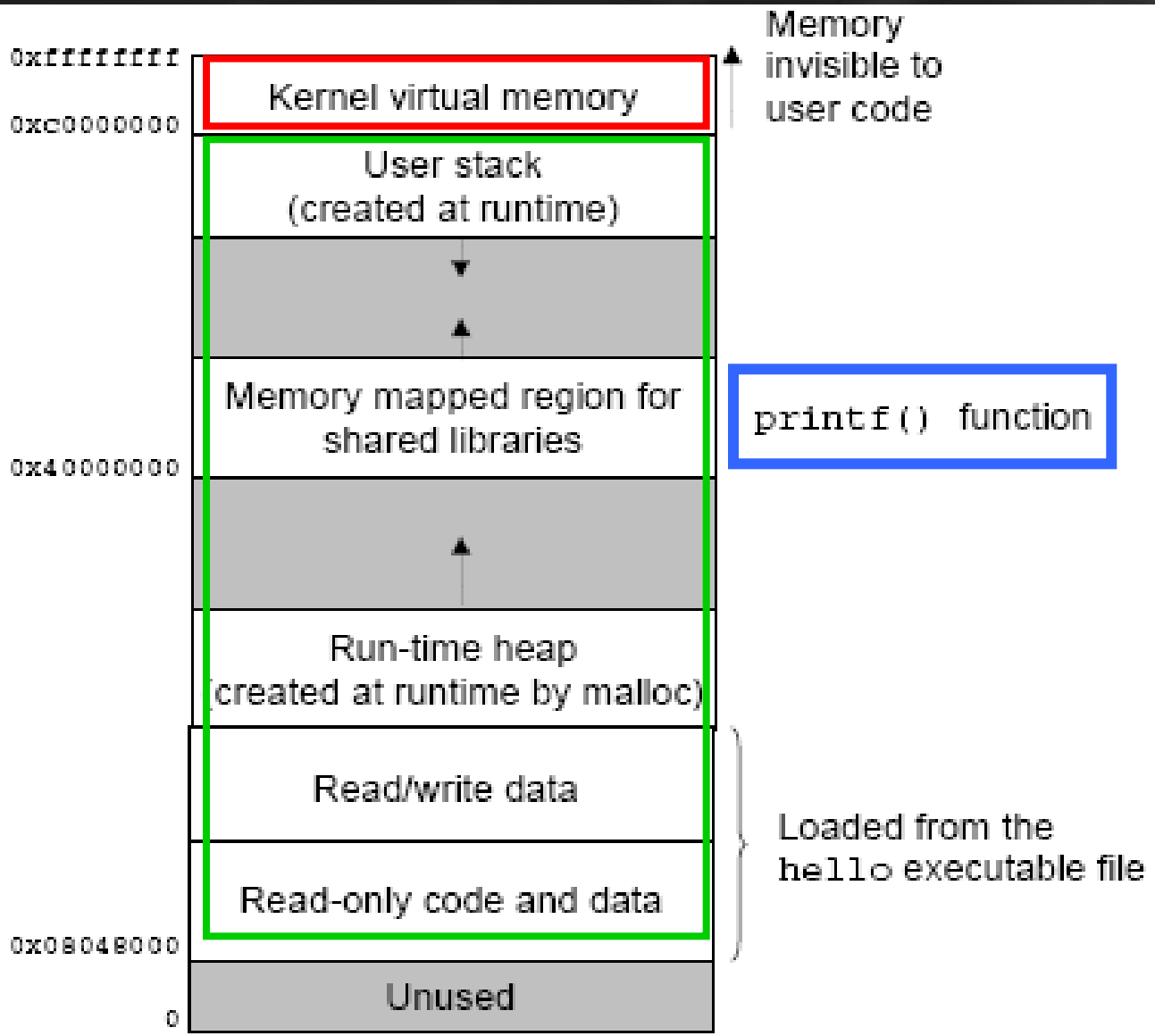
2.5 Load of hello.exe

- Process
- Virtual Memory
- File

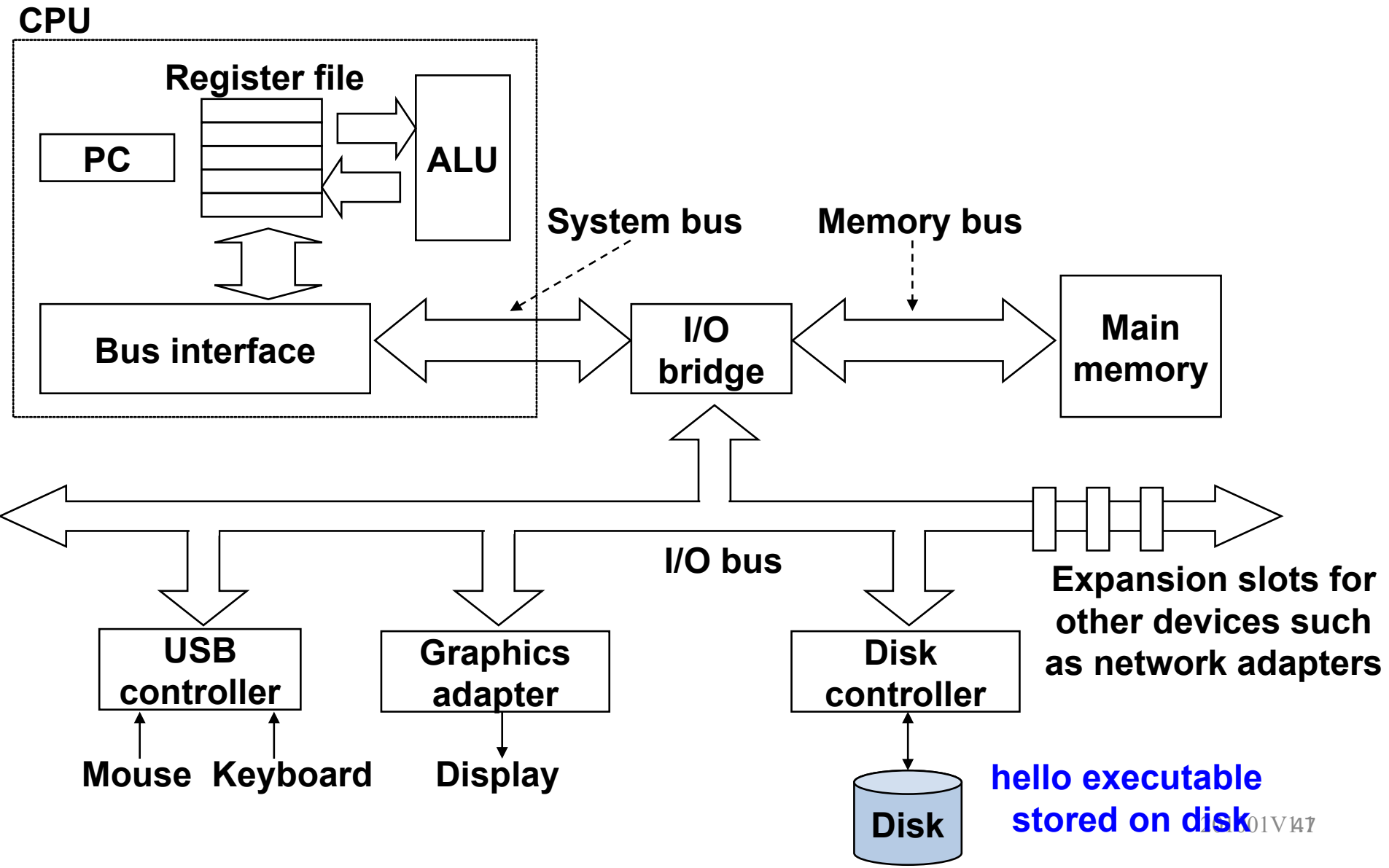


2.5 Load of hello.exe

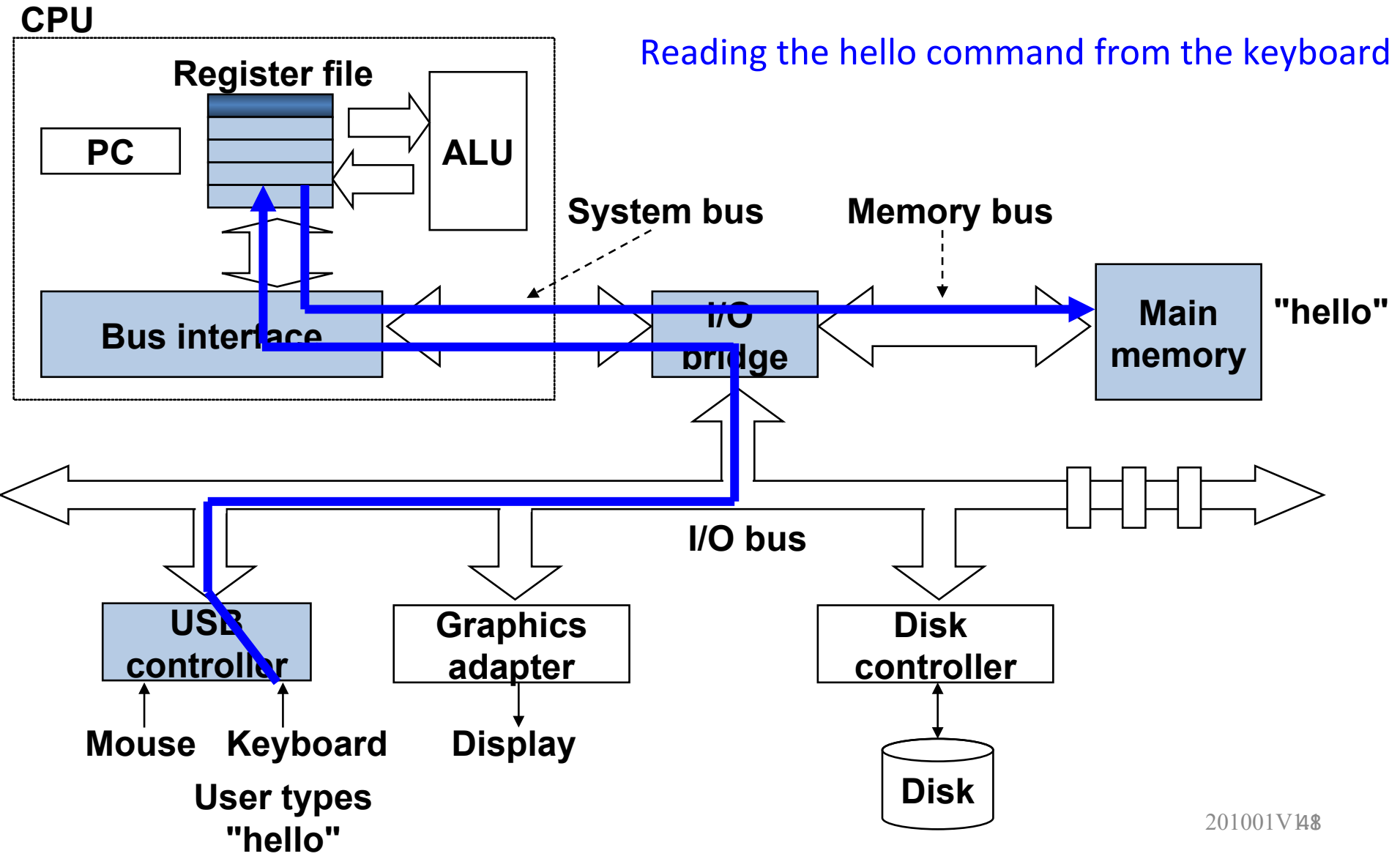
- Process
- Virtual Memory
- File



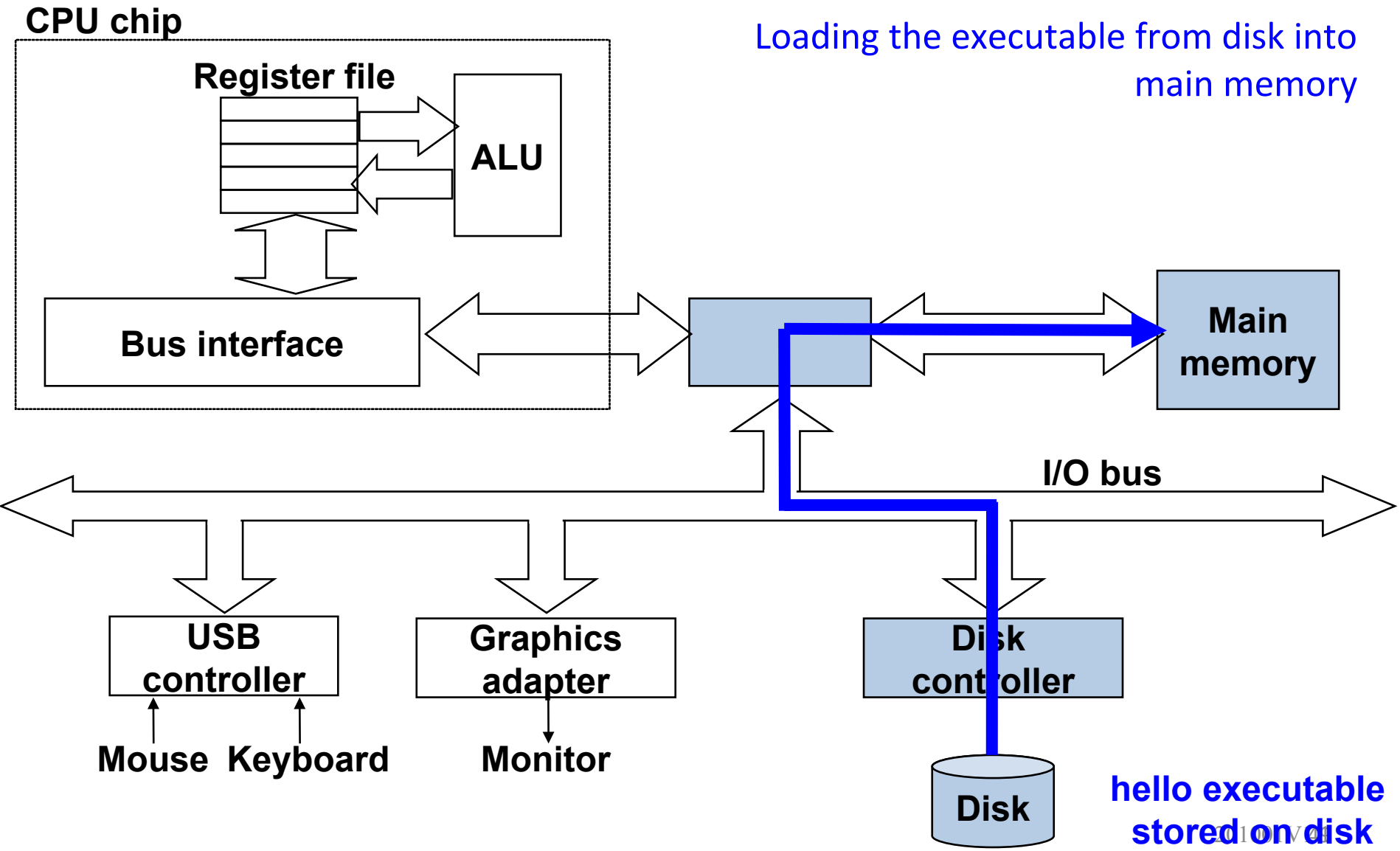
2.5 Load of hello.exe



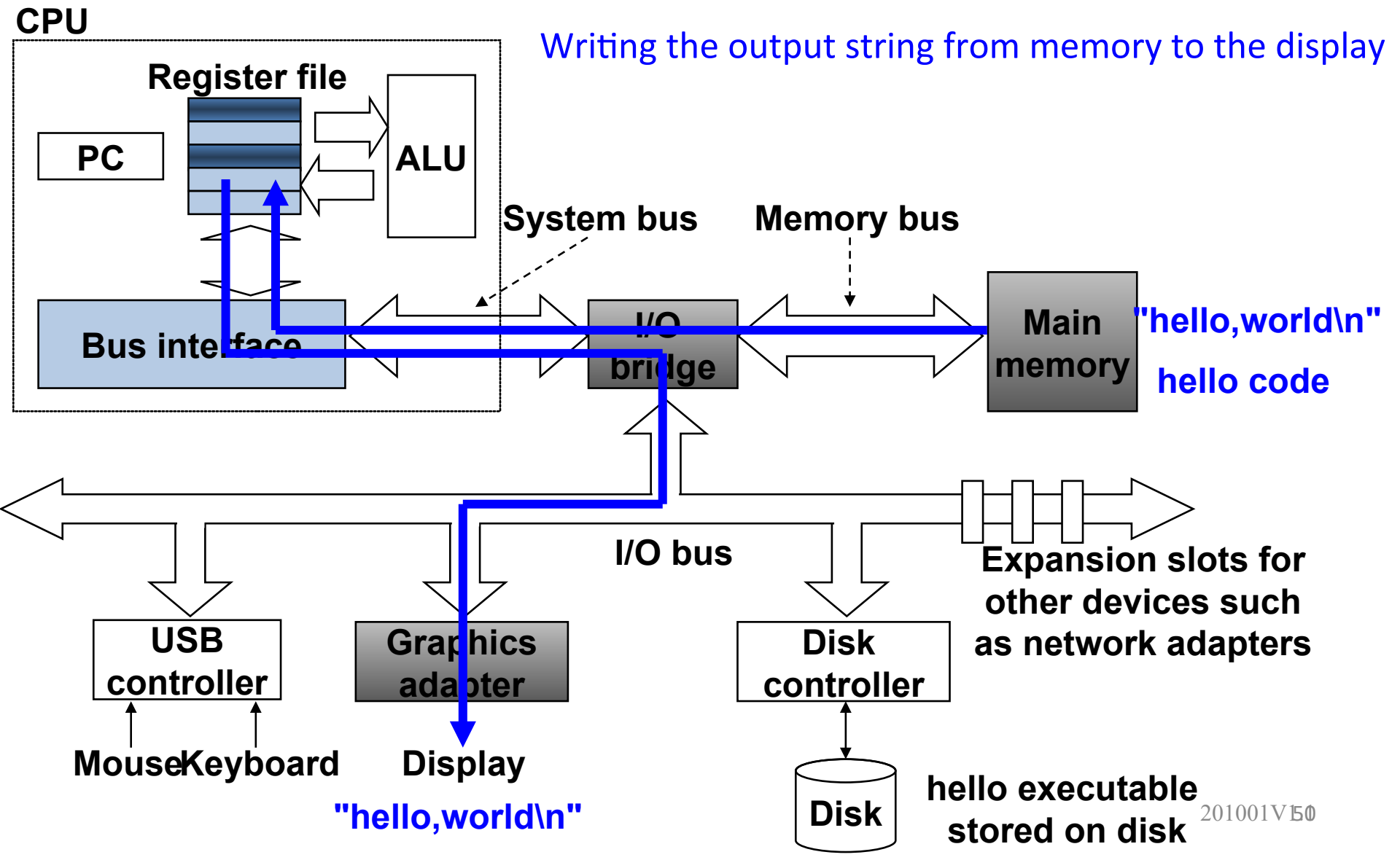
2.5 Load of hello.exe



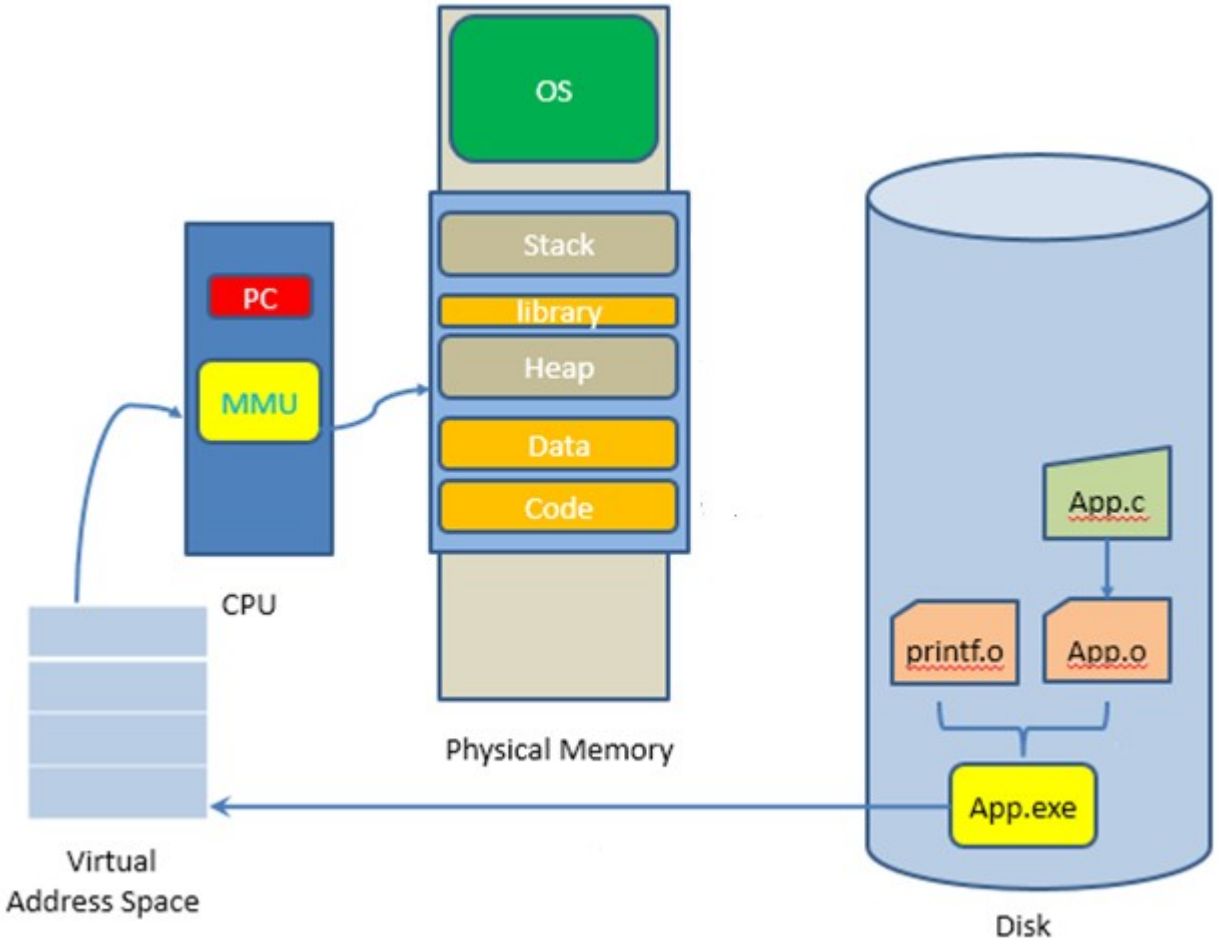
2.5 Load of hello.exe



2.5 Load of hello.exe



2.5 Load of OS



The load of

- OS
- Application

2.5 Load of OS

- 加电 -> 读 BIOS-> 读加载程序 -> 读 OS 内核映像文件
- 加电后第一条指令从 MEM 的 ROM 部分读出（ MEM=ROM+RAM), 即 BIOS(Basic Input Output System)
- 实模式： CS:IP=0XF000:FFF0
- 实模式地址 CS*16+IP, 最大访问 1M

物理内存	地址
空闲	4G
BIOS 启动固件	1M
	64KB
空闲	0

2.5 Load of OS

- 引导的最终目的是从磁盘加载 OS ，因此 BIOS 需要建立以下功能
  - 基本输入输出：磁盘和键盘，显示
  - 系统设置：系统从哪个盘启动， usb ，硬盘，光盘？
  - 开机自检程序
  - 系统启动程序

2.5 Load of OS

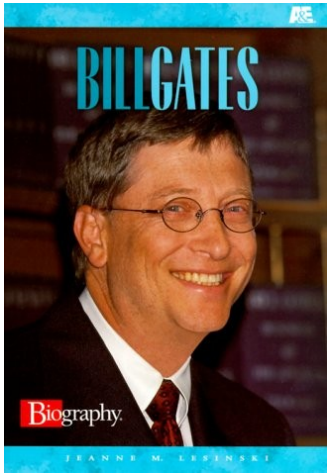
- BIOS 将加载程序从磁盘的引导扇区 512 个字节读到内存指定的 0x7c00, 跳转到 CS:IP=0000:7C00 执行
- 加载程序将操作系统代码和数据从硬盘加载到内存，然后跳转到操作系统起始地址（即内核执行地址）

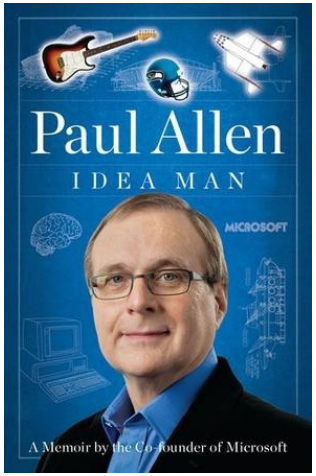
物理内存	地址
空闲	4G
BIOS 启动固件	~1M
加载程序	0x7c00~
BIOS 数据	0

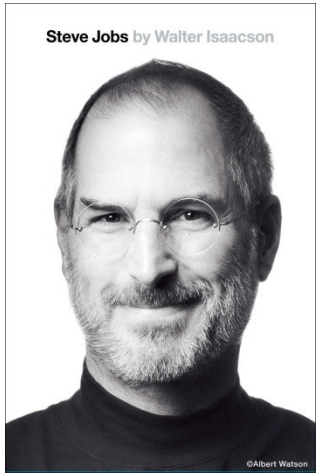
2.6 Other topic

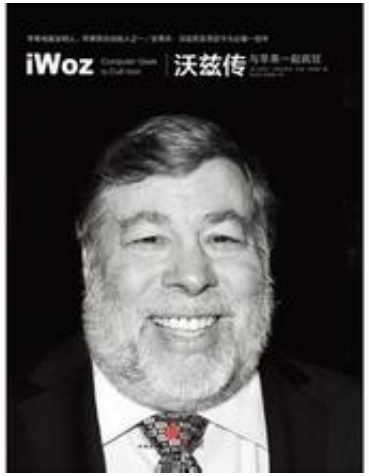
How is Operating System (OS) developed?

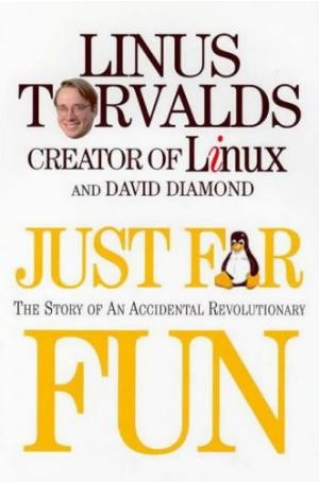
操作系统的发展：相关书籍：人物传记











Microsoft

Apple

Linux

2.6 Other topic

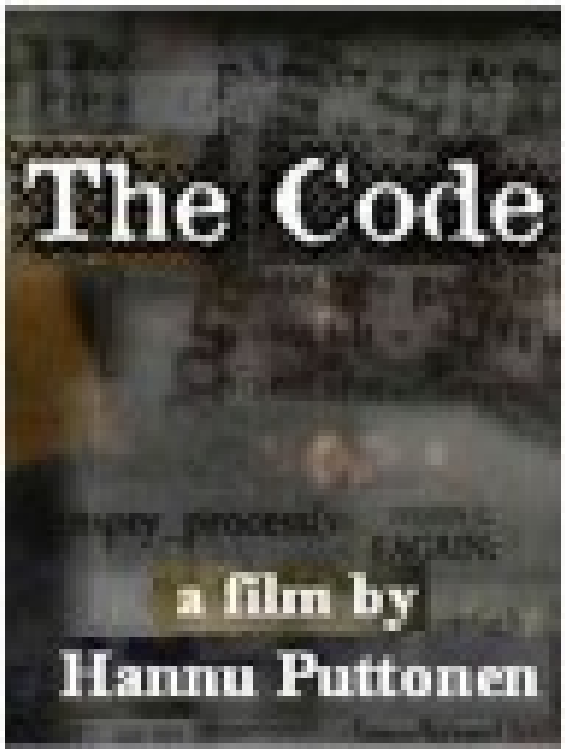
How is Operating System (OS) developed?

操作系统的发展：纪录片

2.6 Other topic

How is Operating System (OS) developed?

操作系统的发展：纪录片



Q:

- 假设有一台多道程序的计算机，每个作业都有相同的特征，即在一个计算周期 T 里， I/O 占用第一和第四个 1/4 周期，处理器占有第二和第三个 1/4 周期。而各个作业占用的 I/O 设备各不相同。如果计算机被任何一个作业单独占用的话，完成作业总共需要 1 个计算周期。假设使用时间片轮转进程调度策略，而且 I/O 操作能与处理器操作重叠，不同的 I/O 设备可以同时运行。定义以下变量：
  周转时间 = 完成一个作业的实际时间
  吞吐量 = 每周期 T 完成的作业的数量的平均值
  处理器使用率 = 处理器处于活动状态 (非等待) 的时间在总时间中占的百分比

Q:

- 若有两个作业同时要求运行，作业 I 的优先级较高，作业都只运行一次。进程状态转换时的操作系统开销忽略不计，且时间片的大小远小于 T。
  计算：
  (1) 作业 1 和作业 2 各自的周转时间。
  (2) 从作业开始运行，到两个作业都已完成的瞬间，计算机系统的吞吐量和处理器使用率。
- （ 3 ）如果是前 T/2 是 I/O, 后 T/2 是处理器时间呢？

- 多道批处理和时分系统的任务调度策略是否一致？