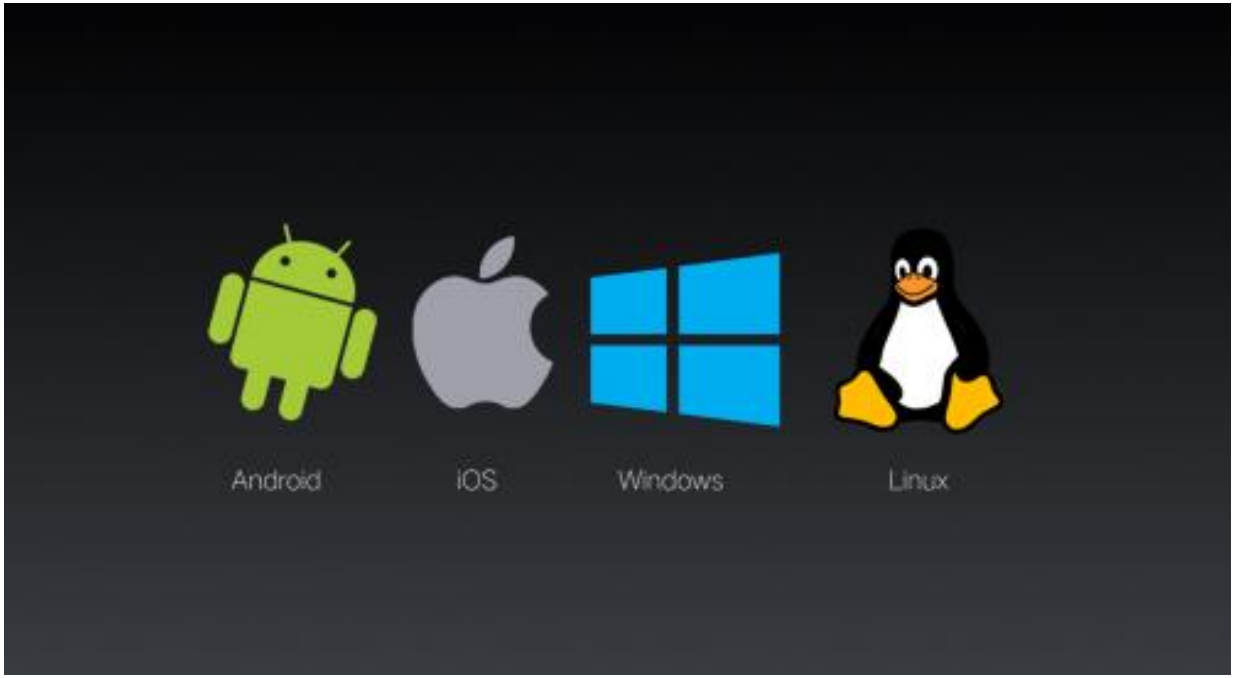
第2章 操作系统概述

- 主要内容
  - 2.1 操作系统的目标和功能
  - 2.2 操作系统的发展
  - 2.3 主要的成就
  - 2.4 现代操作系统的特征
  - 2.5 虚拟机
  - 2.6 针对多处理器和多核的操作系统设计考虑因素
  - 2.7 微软Windows系统简介
  - 2.8 传统的UNIX系统
  - 2.9 现代UNIX系统
  - 2.10 Linux操作系统

2.1 操作系统的目标和功能

- 操作系统是控制应用程序执行的程序，并充当应用程序和计算机硬件之间的接口。
- 操作系统的目标：
  - 方便
  - 有效
  - 扩展能力



2.1 操作系统的目标和功能

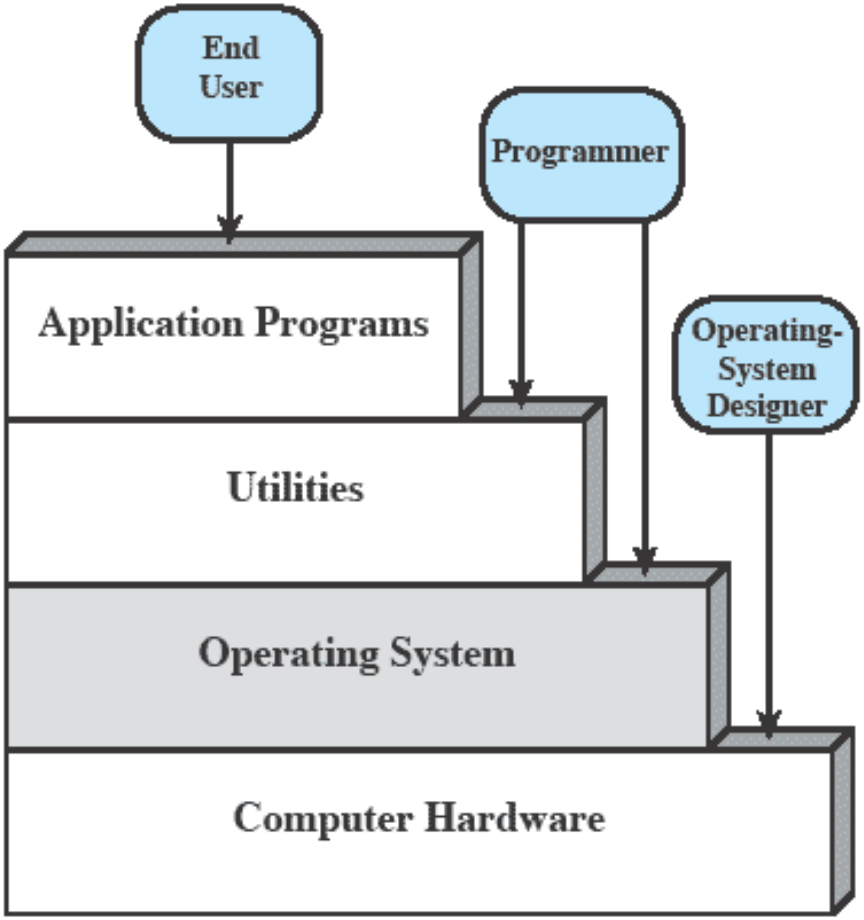
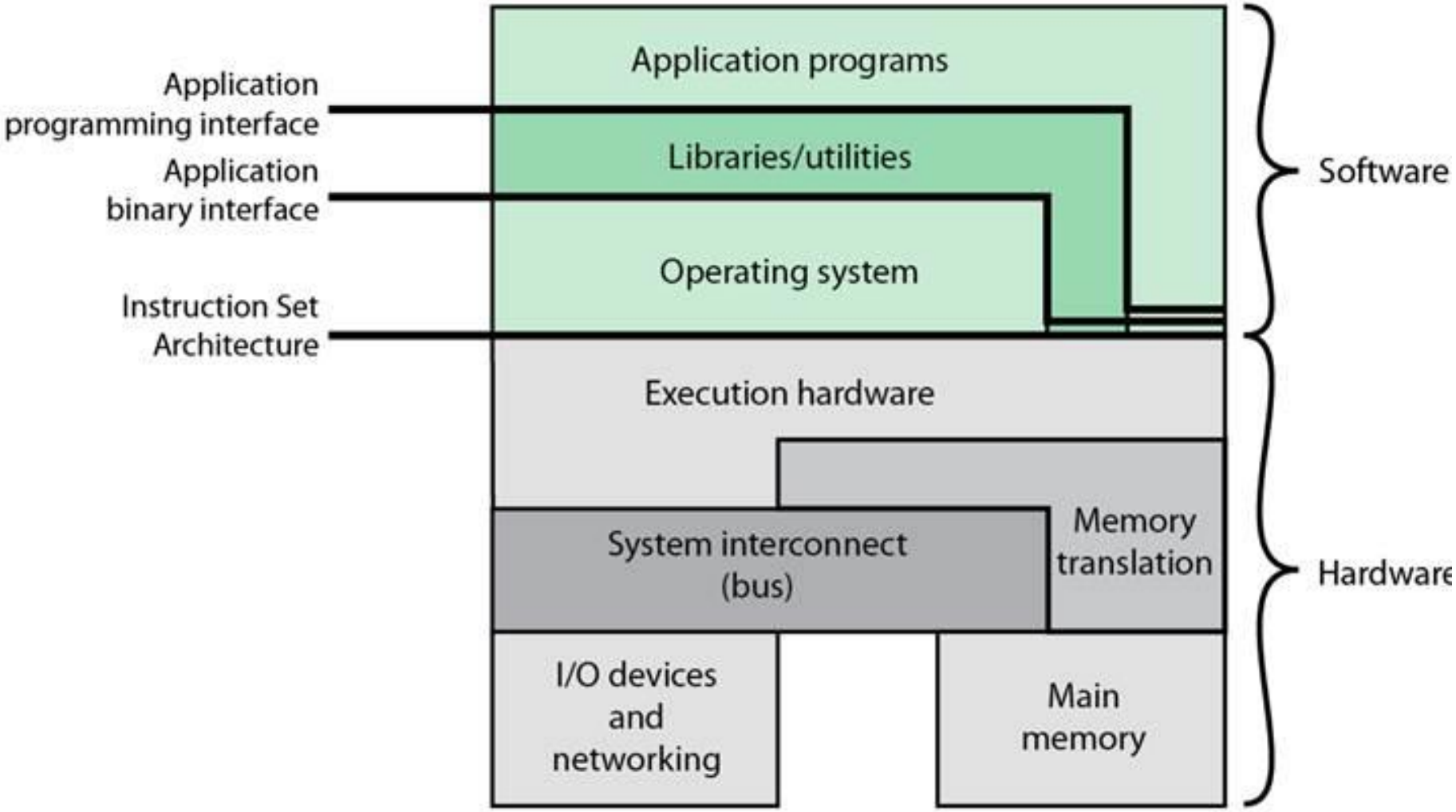


Figure 2.1 Layers and Views of a Computer System

2.1.1 作为用户/计算机接口的操作系统



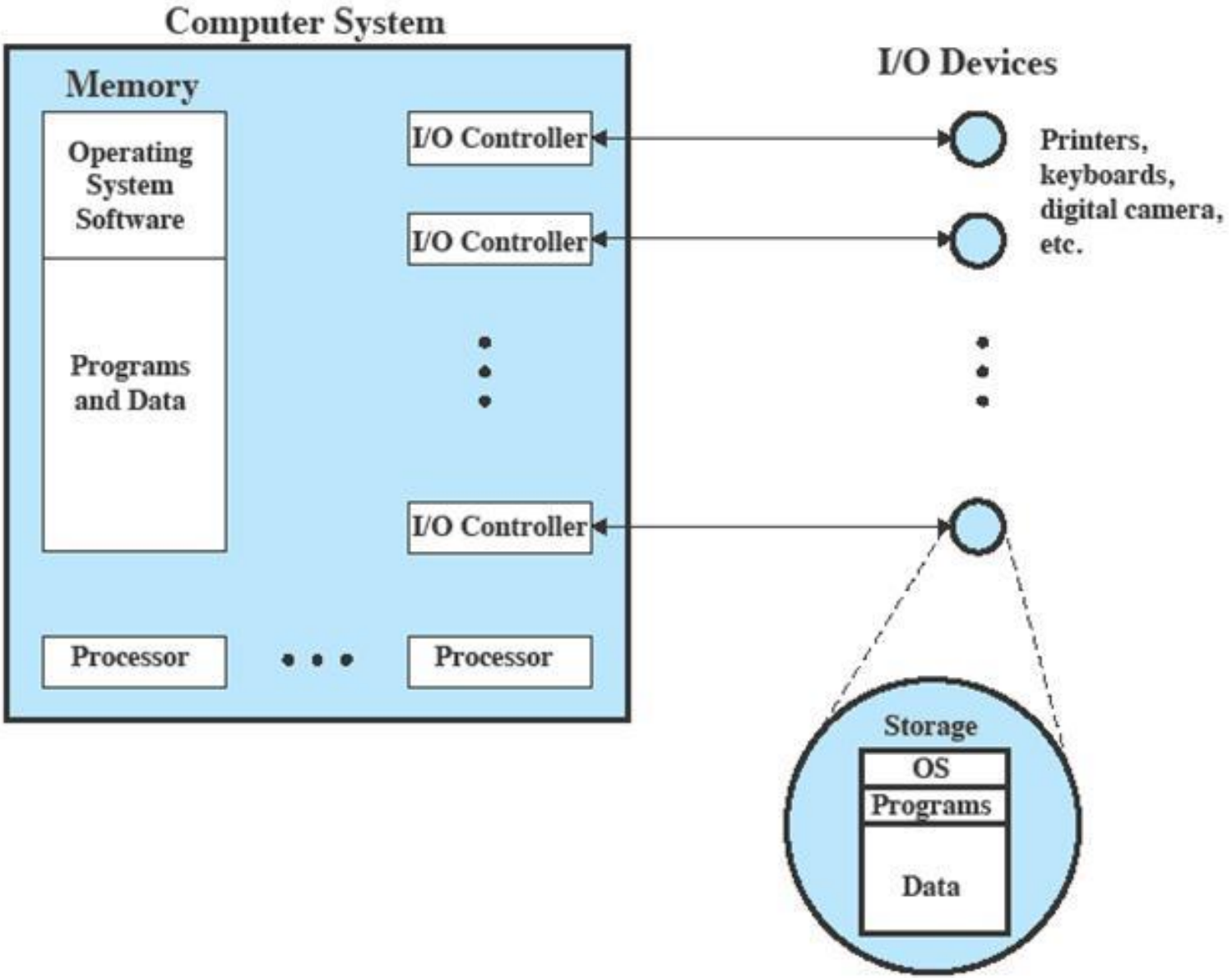
操作系统提供的服务

- 程序开发
- 程序运行
- I/O设备访问
- 文件访问控制
- 系统访问
- 错误检测和响应
- 记账

典型计算机系统中的三种重要的接口

- 指令系统体系结构（ISA）
  - 定义了计算机遵循的机器语言指令系统，该接口是硬件与软件的分界线。
- 应用程序二进制接口（ABI）
  - 定义了程序间二进制可移植性的标准。
- 应用程序编程接口（API）
  - 允许应用程序访问系统的硬件资源和服务。

2.1.2 作为资源管理器的操作系统



2.1.3 操作系统的易扩展性

- 一个重要的操作系统应该能够不断发展，原因：
  - 硬件升级和新型硬件的出现
  - 新的服务
  - 纠正错误
- 对操作系统设计的要求：
  - 采用模块化的结构
  - 清楚地定义模块间的接口
  - 备有说明文档

提问

1. 操作系统的目标之一是在构造操作系统时，应允许在不妨碍服务的前提下，有效地开发、测试和引入新的系统功能。这属于操作系统的（）目标。

A. 安全　　B. 有效　　C. 扩展能力　　D. 方便

2. 判断：操作系统和普通计算机软件的作用是相同的。

A. 正确　　B. 错误

3. 操作都会存在错误，人们会根据错误引入相应的补丁，补丁本身也可能会引入新的错误，这就要求系统具有良好的（）。

A. 有效性　　B. 方便性　　C. 可控性　　D. 易扩展性

4. （）给程序开发者提供了程序开发需要用到的接口，以方便其访问系统资源。

A. ISA　　B. API　　C. ABI　　D.

5. 判断：内核程序通常包含了操作系统中最常用的功能。

A. 正确　　B. 错误

2.2 操作系统的发展 2.2.1 串行处理

- 串行处理时期（20世纪40年代到50年代中期），没有操作系统，用户必须顺序访问计算机。存在两个主要问题：
  - 调度：大多数装置都使用一个硬拷贝的登记表预订机器时间。
  - 准备时间：加载编译器+源程序+目标程序+公用函数链接等等。

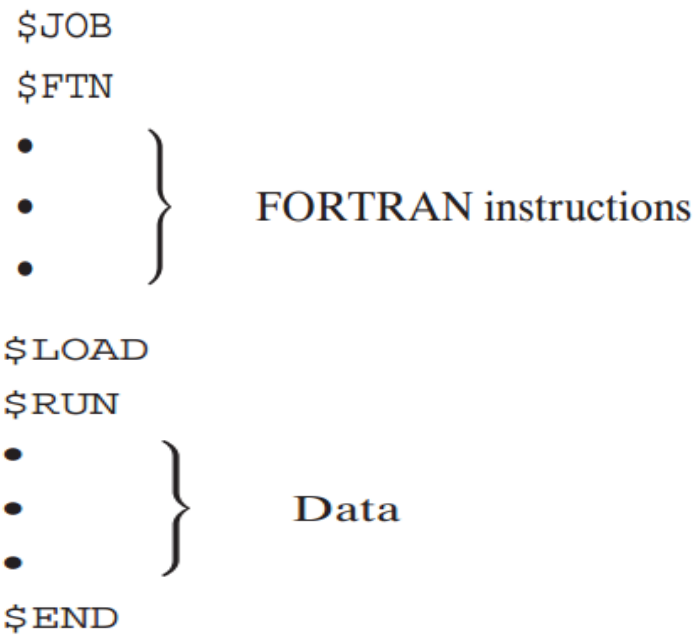
2.2.2 简单批处理系统

- 第一个操作系统（第一个批处理操作系统）
  - 20世纪50年代中期，General Motors开发，用于IBM701
- 中心思想：监控程序
  - 用户作业-计算机操作员-将作业组织成批-输入-监控程序
  - 每个程序处理完后返回到监控程序，同时，监控程序自动加载下一个程序。
- 作业控制语言（JCL）
  - 为监控程序提供指令。每个作业中的指令以JCL的基本形式给出。

2.2.2 简单批处理系统

从监控程序和处理器两个角度分析理解该系统：

- 监控程序角度
  - 常驻监控程度
- 处理器角度



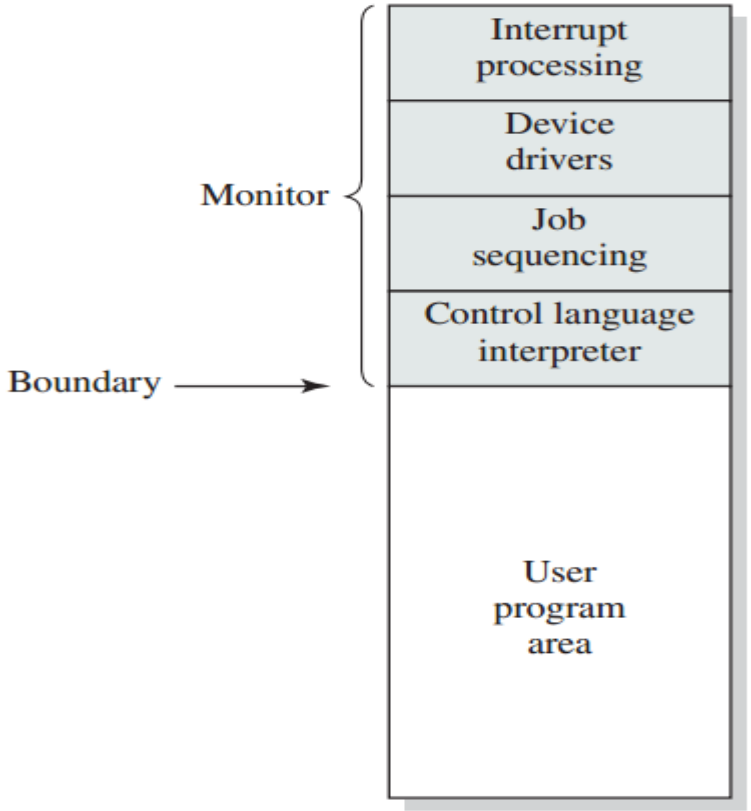


Figure 2.3 **Memory Layout for a Resident Monitor**

2.2.2 简单批处理系统

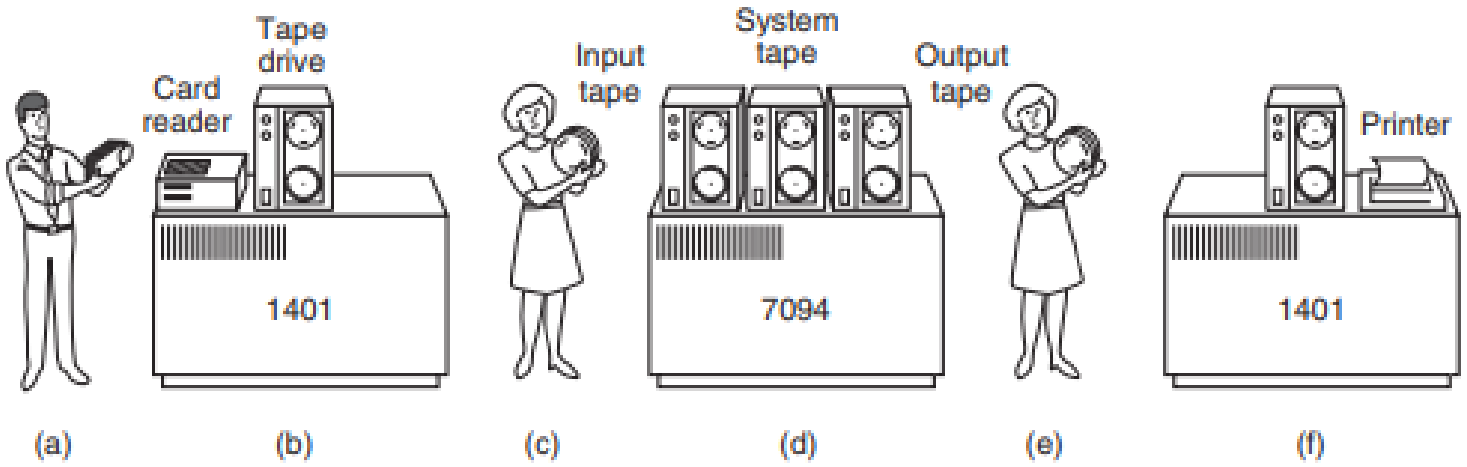
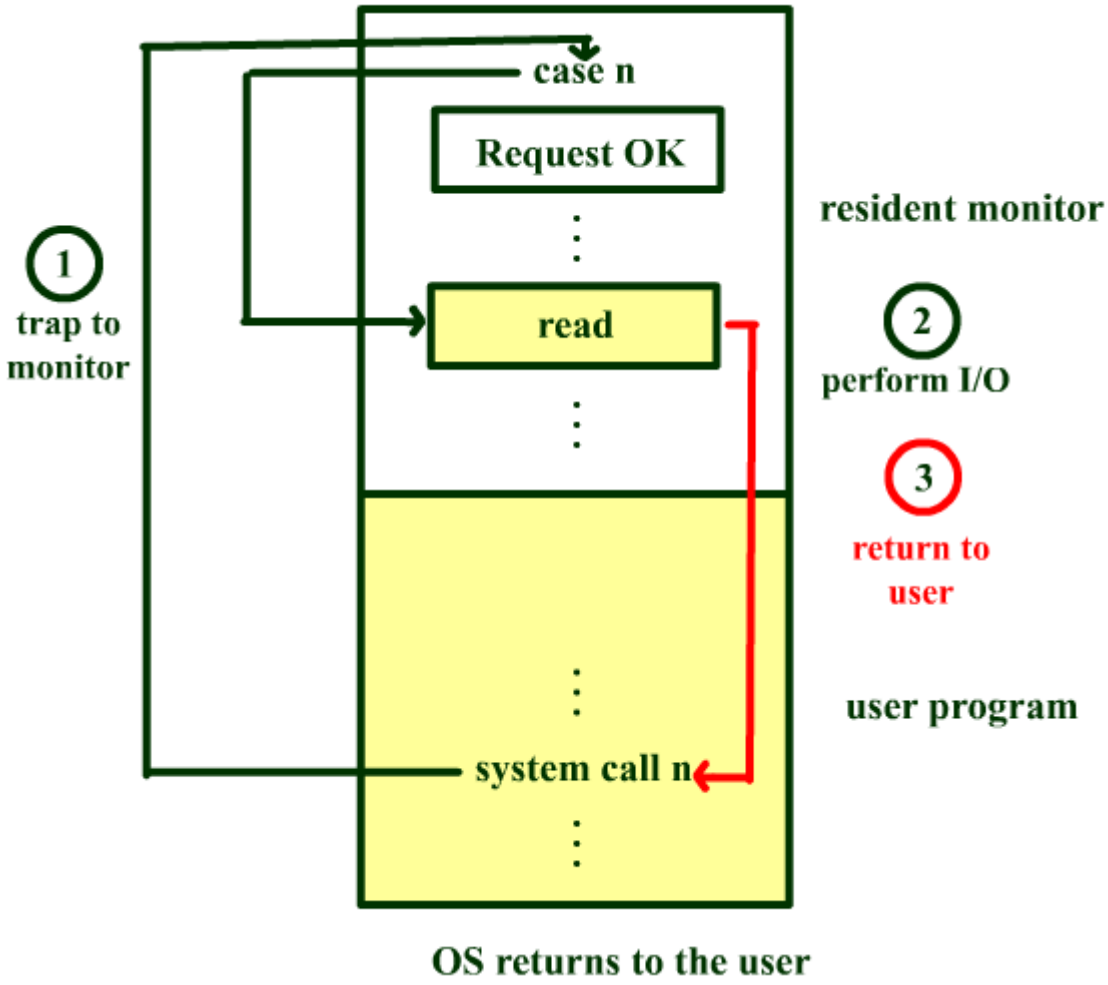


Figure 1-3. An early batch system. (a) Programmers bring cards to 1401. (b) 1401 reads batch of jobs onto tape. (c) Operator carries input tape to 7094. (d) 7094 does computing. (e) Operator carries output tape to 1401. (f) 1401 prints output.

2.2.2 简单批处理系统


case n
Request OK
read
system call n
resident monitor
1
trap to monitor
2
perform I/O
3
return to user
user program
OS returns to the user

2.2.2 简单批处理系统

- 涉及到的硬件功能
  - 内存保护
  - 定时器
  - 特权指令
  - 中断—早期的计算机模型没有中断能力
- 用户模式（用户态）–用户程序
- 内核模式（内核态）–监控程序

提问

6. 判断：监控程序是早期的一类操作系统，整个监控程序必须处于内存当中，以读取作业。

A. 正确　　B. 错误

7. 判断：在简单批处理系统中，用户程序以用户态模式执行，不可以访问受保护的内存区域，但可以执行极少数的特权指令。

A. 正确　　B. 错误

8. 判断：在简单批处理系统中，用户程序和监控程序交替执行。

A. 正确　　B. 错误

2. 2. 3 多道批处理系统.

Read one record from file	15 μs
Execute 100 instructions	1 μs
Write one record to file	15 μs
TOTAL	31 μs

Percent CPU Utilization $= \frac{1}{31} = 0.032 = 3.2\%$

Figure 2.4 System Utilization Example

2.2.3 多道批处理系统.

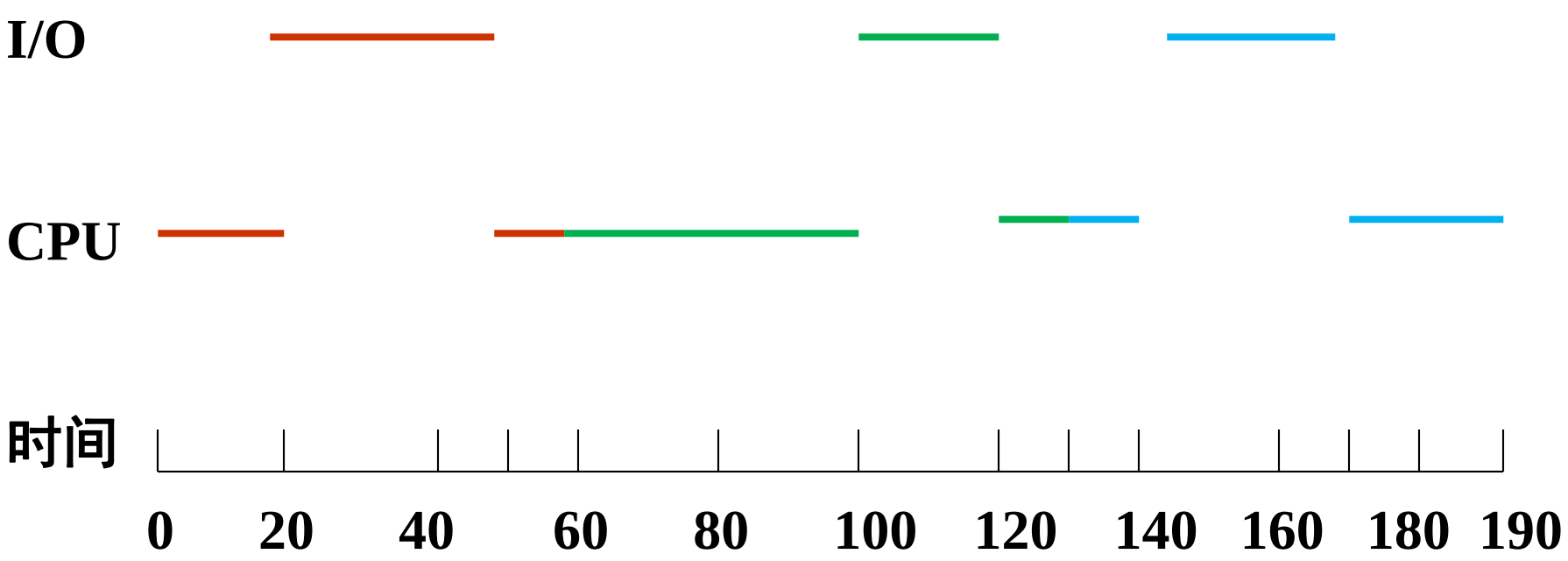
- 多道程序设计（多任务处理）
  - 内存同时保存多个程序，当一个作业需要等待I/O时，处理器可以切换到另一个不需要等待I/O的作业。
  - 提高CPU的利用率。
  - 需要中断技术、内存管理、进程调度等方面的支持。

多道程序设计例

- 若主存中有3道程序A、B、C，它们按A、B、C优先次序运行，各程序的计算轨迹为：
  - A：计算（20）、I/O（30）、计算（10）
  - B：计算（40）、I/O（20）、计算（10）
  - C：计算（10）、I/O（30）、计算（20）
- 如果三道程序都使用相同设备进行I/O（即程序用串行方式使用设备，调度开销忽略不计）。试分别画出单道和多道运行的时间关系图。两种情况下，CPU的平均利用率各为多少？

A: 计算（20）、I/O（30）、 计算（10）
A: 计算（40）、I/O（20）、 计算（10）
A: 计算（10）、I/O（30）、 计算（20）

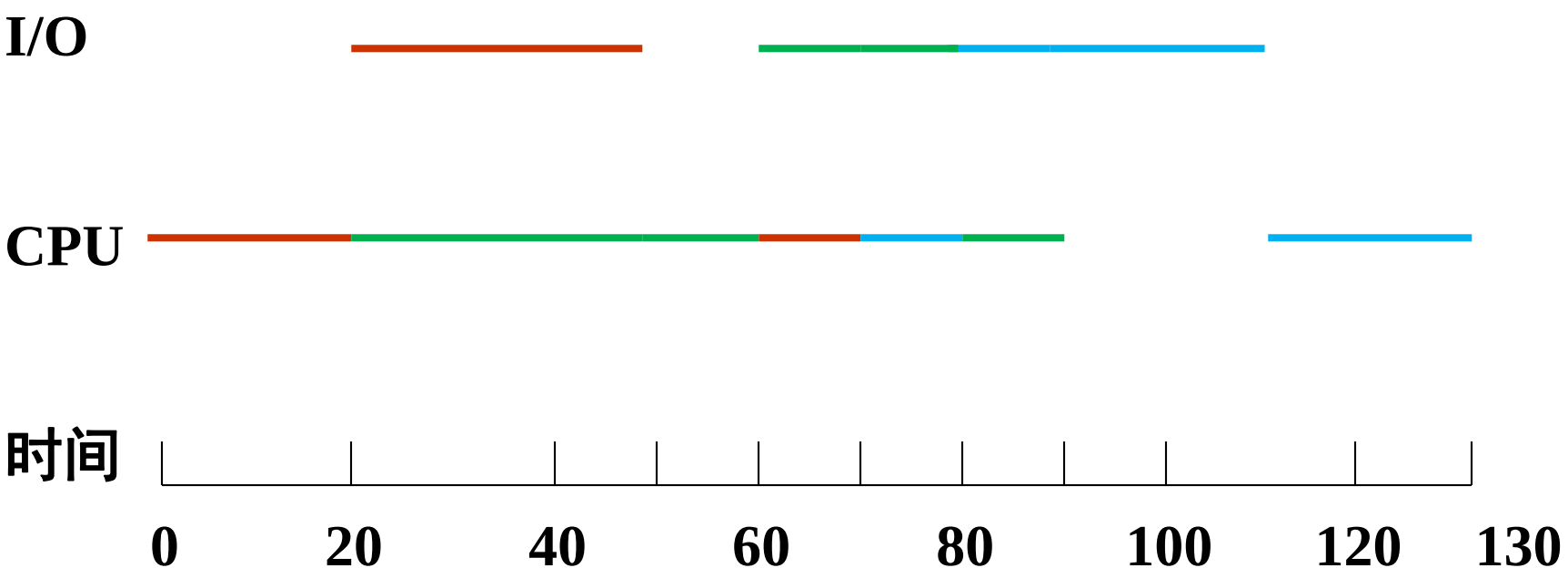
单道运行时间关系图



单道CPU利用率为(190－80)/190＝57.9%

多道运行时间关系图（不抢占）

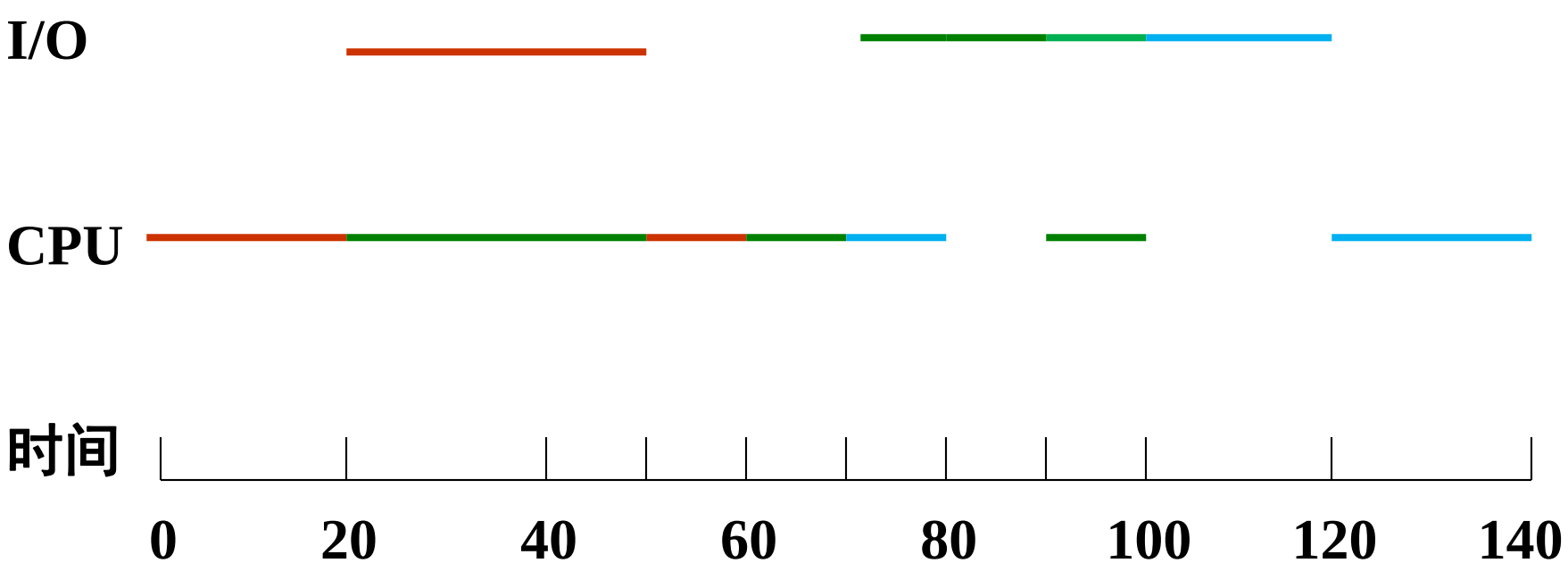
A: 计算（20）、I/O（30）、 计算（10）
A: 计算（40）、I/O（20）、 计算（10）
A: 计算（10）、I/O（30）、 计算（20）



多道CPU利用率为(130－20)/130＝84.6%

多道运行时间关系图（抢占）

A: 计算（20）、I/O（30）、 计算（10）
A: 计算（40）、I/O（20）、 计算（10）
A: 计算（10）、I/O（30）、 计算（20）



多道CPU利用率为(140－30)/140＝78.6%

多道程序设计示例

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

多道程序设计示例

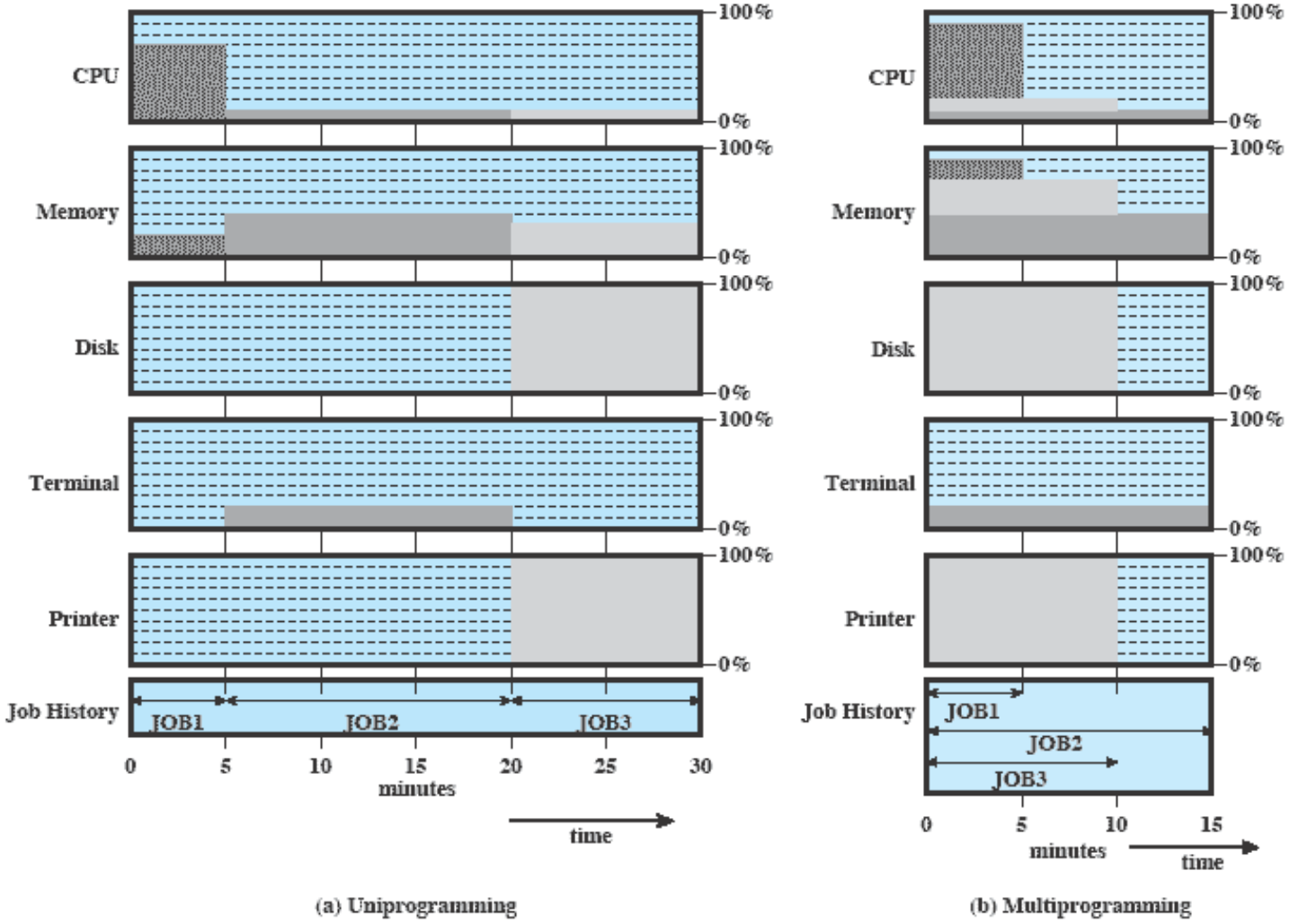
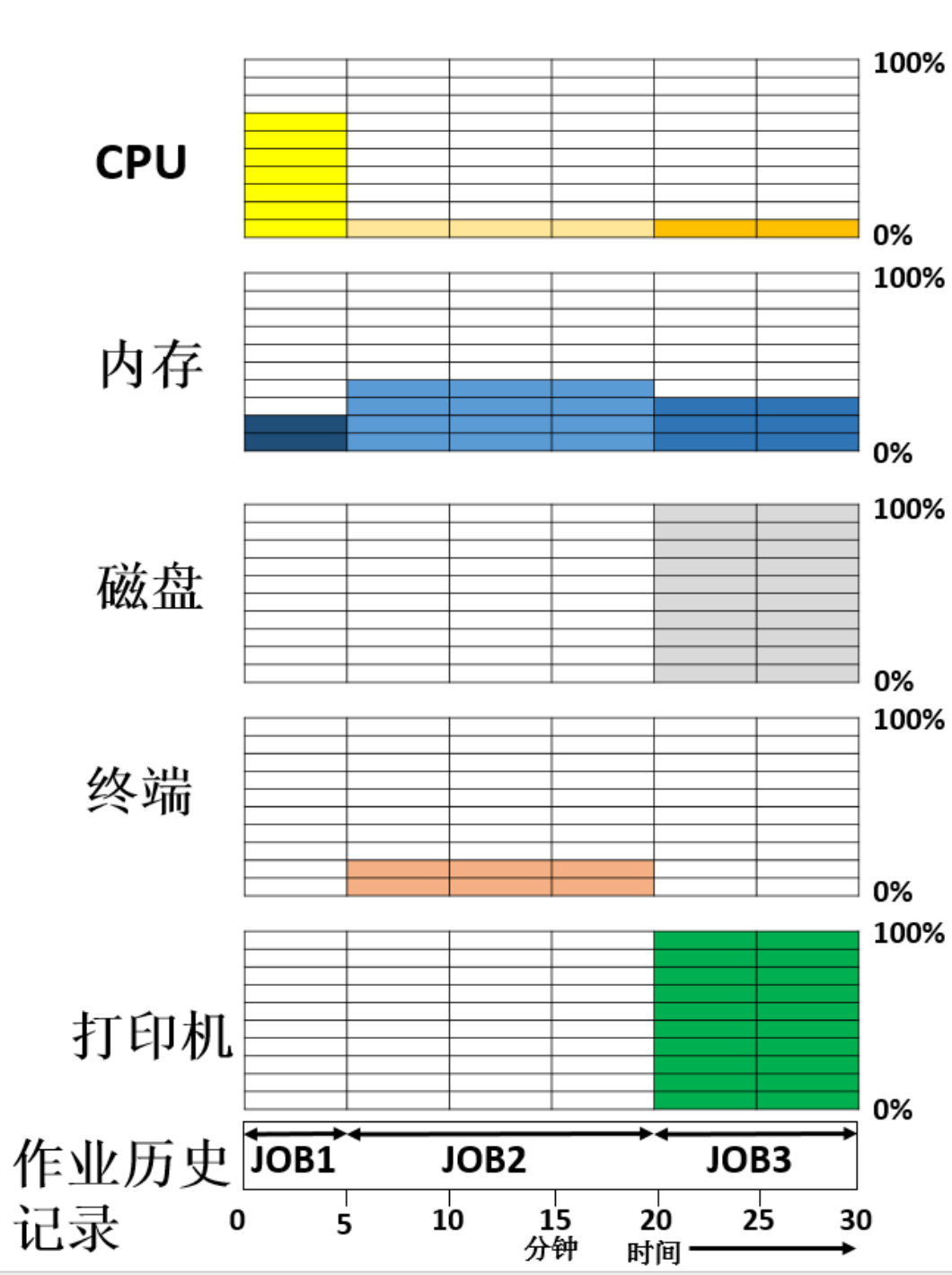
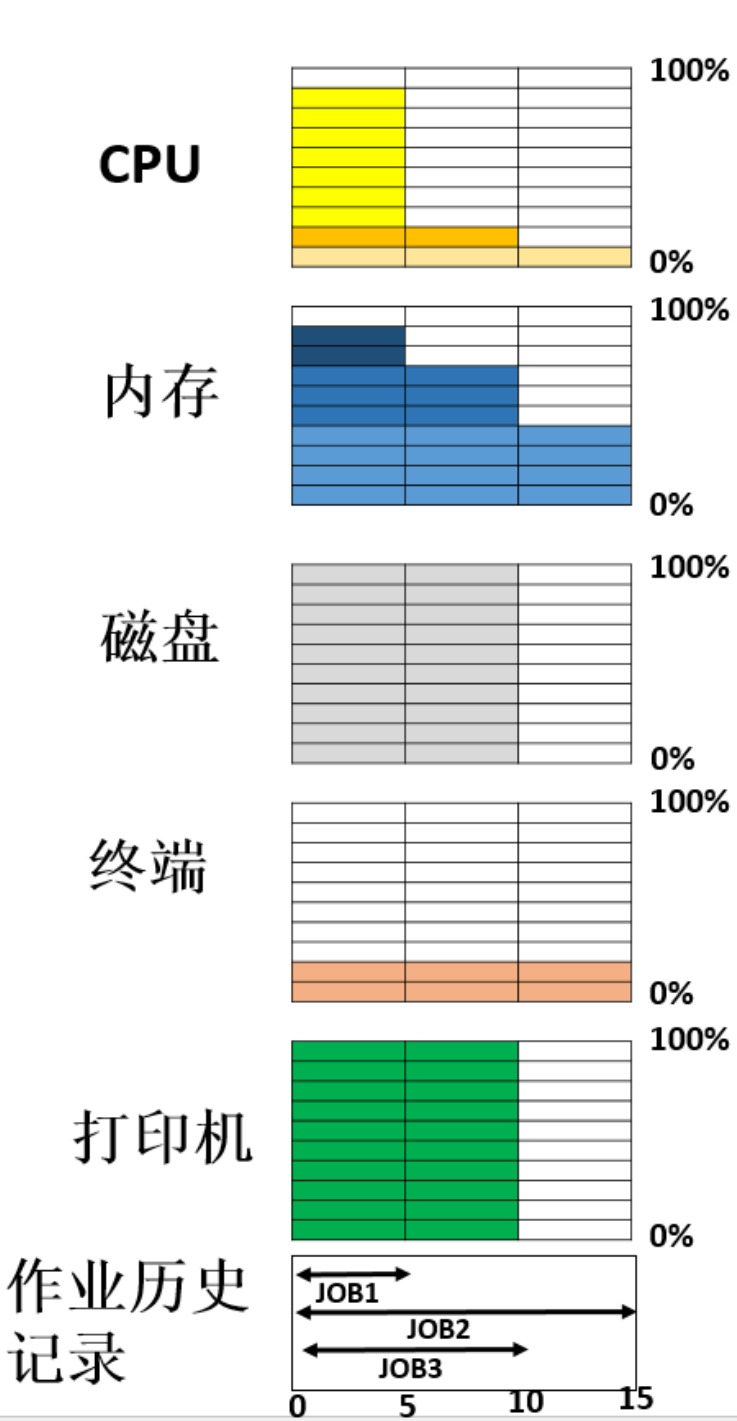


Figure 2.6 Utilization Histograms


CPU
100%
0%
内存
100%
0%
磁盘
100%
0%
终端
100%
0%
打印机
100%
0%
作业历史记录
JOB1
JOB2
JOB3
0
5
10
15
20
25
30
分钟
时间


CPU
100%
0%
内存
100%
0%
磁盘
100%
0%
终端
100%
0%
打印机
100%
0%
作业历史记录
JOB1
JOB2
JOB3
0
5
10
15

多道程序设计示例

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

2.2.4 分时系统

- 为什么要有分时系统?
  - 批处理用户不能干预自己程序的运行，无法得知程序的运行情况，不利于程序调试和排错。
- 分时系统
  - 允许多个联机用户同时使用一个计算机系统进行交互式计算。
  - 时钟中断，时间片技术。
- 分时和多道程序设计引发的新问题：
  - 作业的相互干扰
  - 文件系统的保护
  - 处理资源的竞争

2.2.4 分时系统

批处理多道程序设计和分时的比较

	批处理多道程序设计	分时
主要目标	充分使用处理器	减少响应时间
操作系统指令源	作业控制语言 作业提供的命令	终端输入的命令

2. 2. 4 分时系统

JOB1→JOB2→JOB3→JOB4

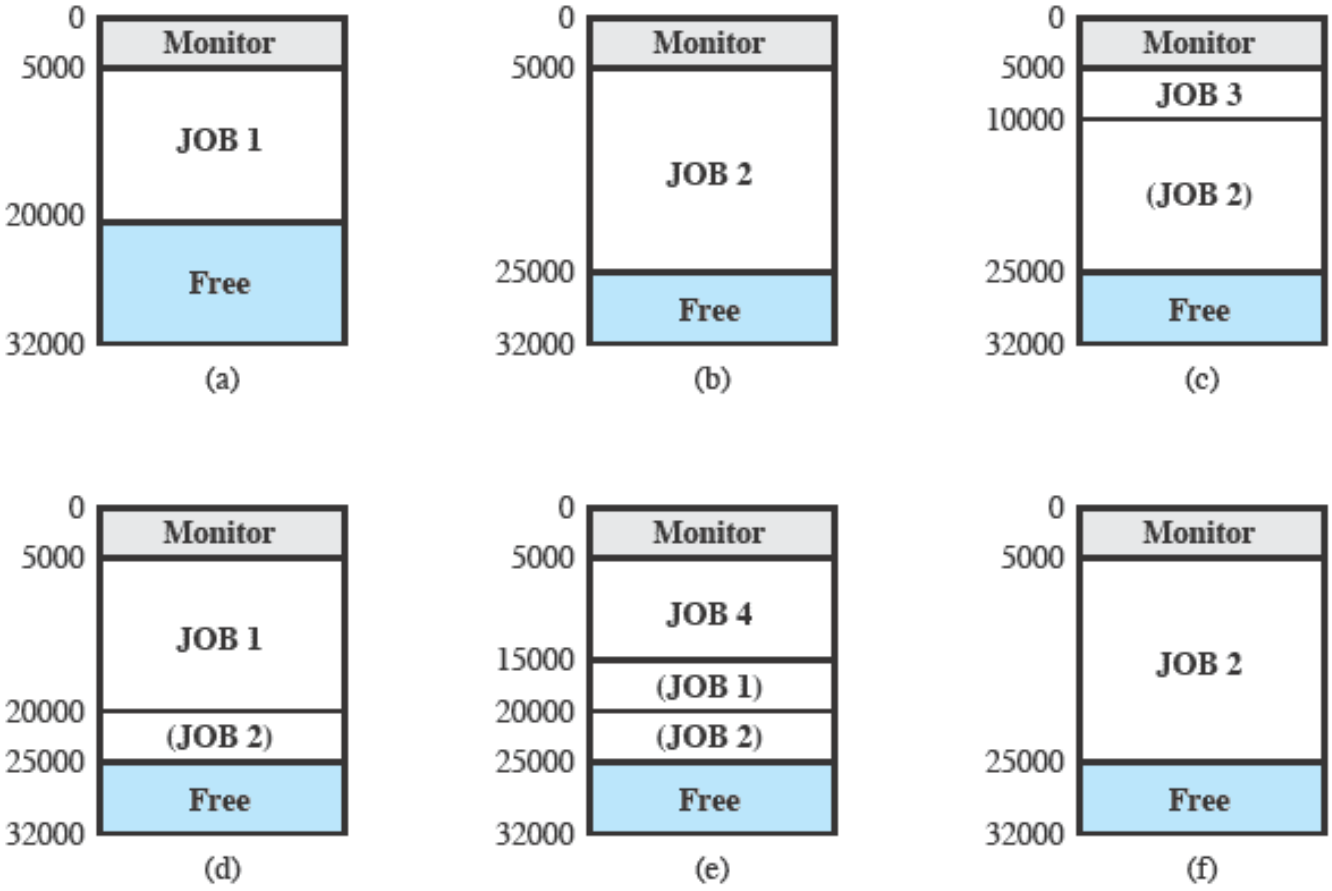


Figure 2.7 CTSS Operation

提问

9. 判断：批处理多道程序设计的主要目标是充分利用处理器，它允许多个联机用户同时使用一个计算机系统进行交互式计算。

A. 正确　　B. 错误

10. 分时系统的指令源是（）

A. 作业控制语言命令　　B. 作业提供的命令　　C. 终端键入的命令

2.3 主要的成就

- 操作系统开发中四个重要的理论进展
  - 进程
  - 内存管理
  - 信息保护和安全
  - 调度和资源管理

2.3.1 进程

- 定义
  - 一个正在执行的程序。
  - 计算机中正在运行的程序的一个实例。
  - 可以分配给处理器并由处理器执行的一个实体。
  - 由单一顺序的执行线索、一个当前状态和一组相关的系统资源所描述的活动单元。
- 问题
  - 同步
  - 互斥
  - 不确定的程序操作
  - 死锁

2.3.1 进程

- 组成
  - 一段可执行的程序
  - 程序所需要的相关数据
  - 程序的执行上下文
- 进程和程序的区别
  - 进程是动态的
  - 程序是静态的

2.3.1 进程

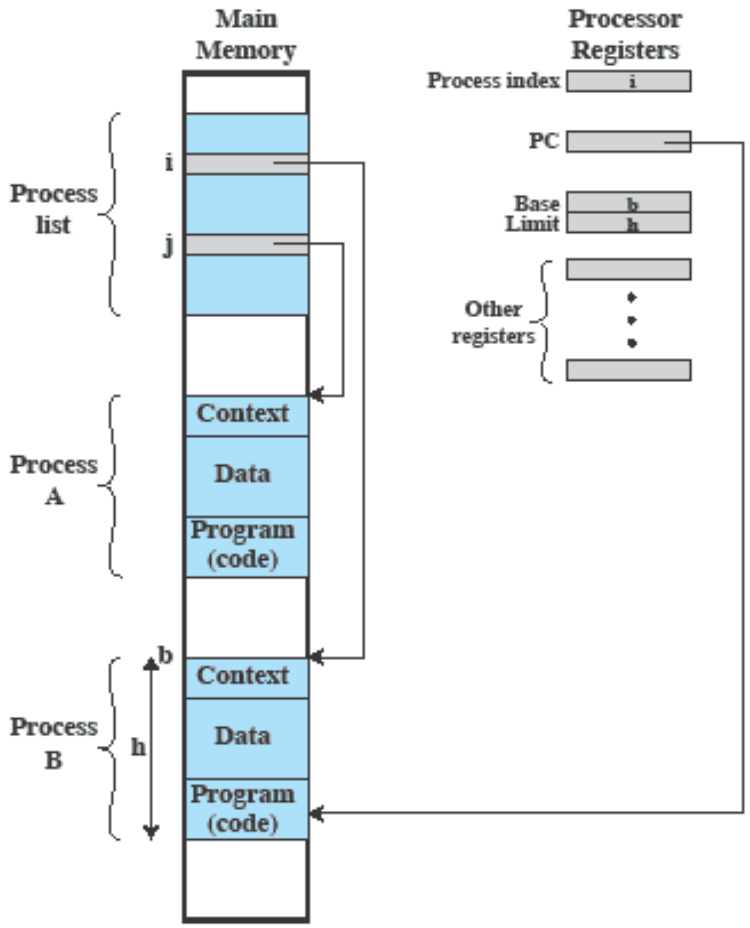
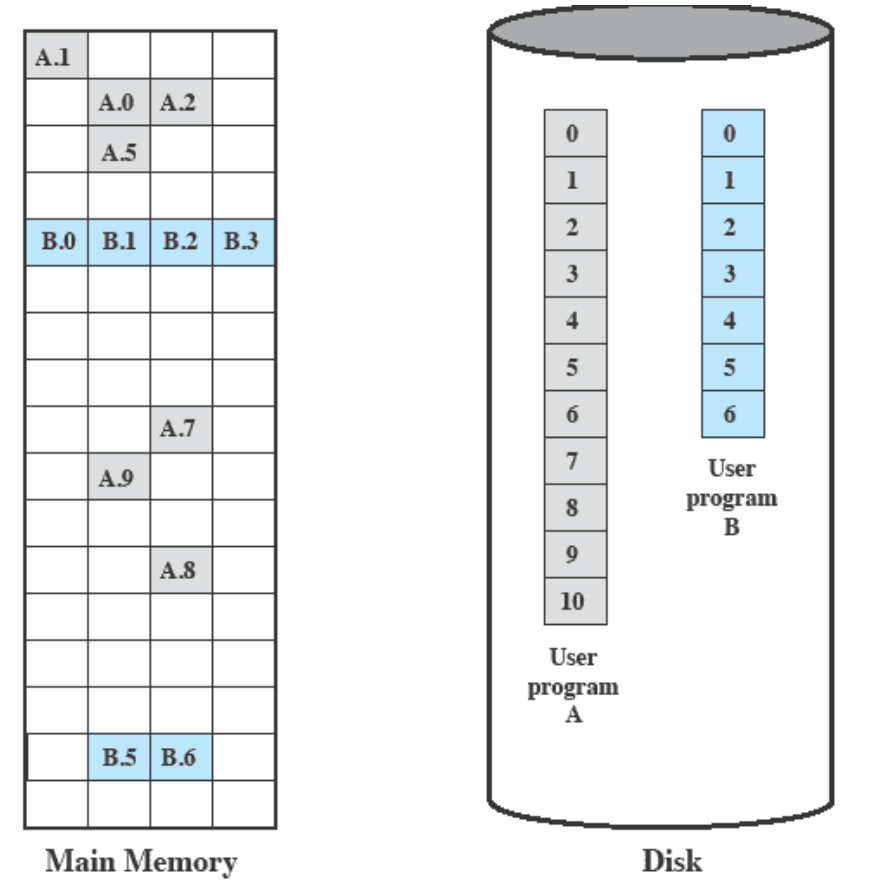


Figure 2.8 Typical Process Implementation

2.3.2 内存管理

- 管理责任
  - 进程隔离
  - 自动分配和管理
  - 支持模块化程序设计
  - 保护和访问控制
  - 长期存储
- 文件系统
  - 长期存储
- 虚存机制
  - 允许程序从逻辑的角度访问存储器，不用考虑物理内存上可用的空间数量。

2.3.2 内存管理



Main memory consists of a number of fixed-length frames, each equal to the size of a page. For a program to execute, some or all of its pages must be in main memory.

Secondary memory (disk) can hold many fixed-length pages. A user program consists of some number of pages. Pages for all programs plus the operating system are on disk, as are files.

Figure 2.9 Virtual Memory Concepts

2.3.2 内存管理

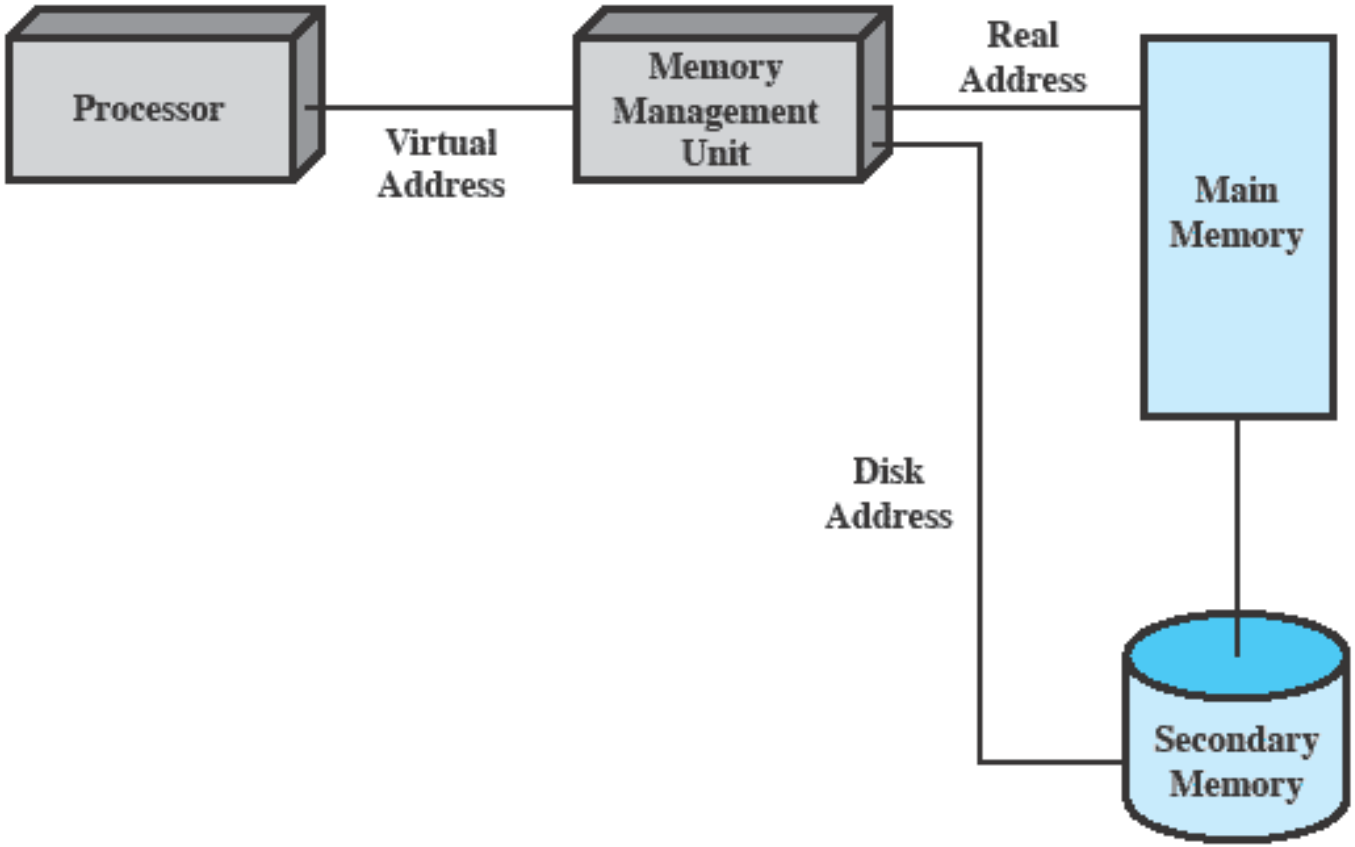


Figure 2.10 Virtual Memory Addressing

2.3.3 信息保护和安全

- 与操作系统相关的安全和保护问题
  - 可用性：保护系统不被打断。
  - 保密性：保证用户不能读到未授权访问的数据。
  - 数据完整性：保护数据不被未授权修改。
  - 认证：涉及用户身份的正确认证和消息或数据的合法性。

2.3.4 调度和资源管理

- 资源分配和调度策略需考虑的因素：
  - 公平性
  - 有差别的响应性
  - 有效性
- 资源分配和调度策略
  - 短程队列（short-term queue）
    - ☒短程调度器（分派器dispatcher）
      - 时间片轮转
      - 分配不同优先级
  - 长程队列（long-term queue）

2.3.4 调度和资源管理

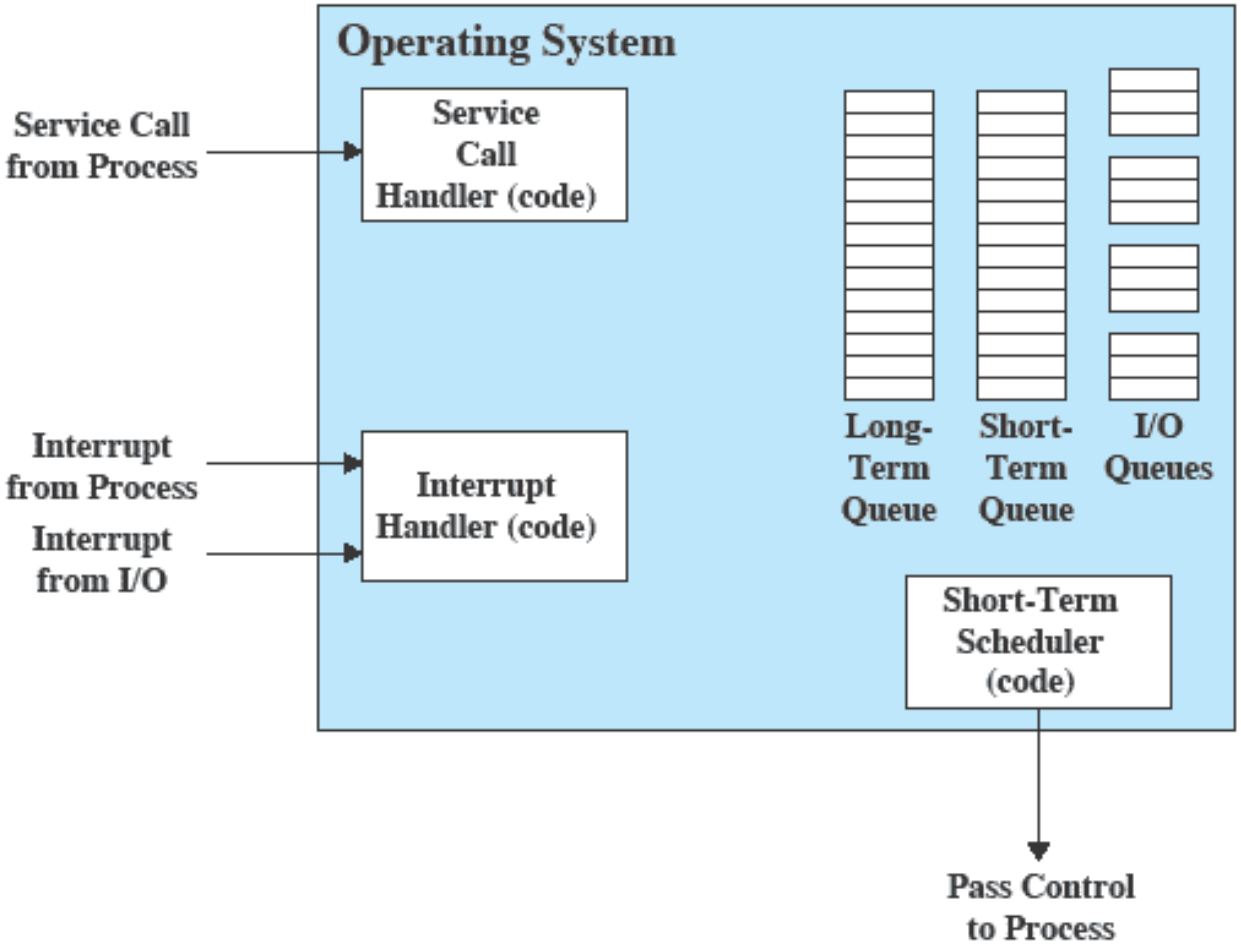


Figure 2.11 Key Elements of an Operating System for Multiprogramming

2.4 现代操作系统的特征

- 微内核体系结构
  - 只给内核分配最基本的功能
  - 其他操作系统服务由运行在用户态下的进程提供
  - 使设计简单、灵活
- 多线程
  - 可分派的工作单元
  - 进程?
- 对称多处理
  - 多处理器的存在对用户的透明的
- 分布式操作系统
- 面向对象设计
  - 其原理用于给小内核增加模块化扩展

提问

11. 判断：程序就是进程。

A. 正确　　B. 错误

12. 操作系统用（）来管理和控制进程所需的内部数据。

A. 执行上下文　B. IR寄存器　　C. PC　　D. 数据寄存器

13. 为了保证多个用户可以正确使用一个共享的资源，操作系统应该具备（）。

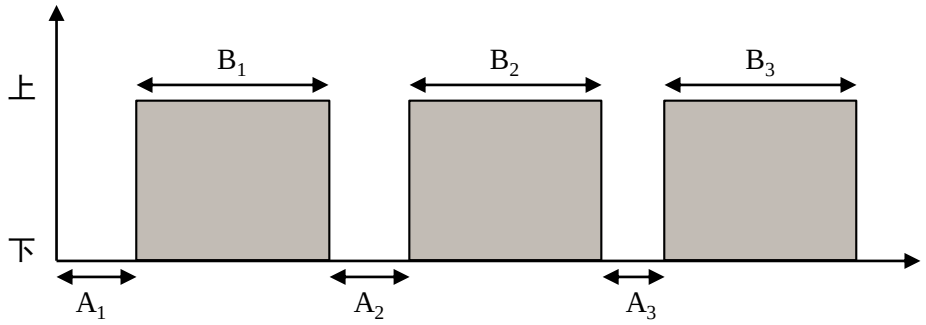
A. 同步机制　　B. 互斥机制　　C. 可控机制　　D. 内存管理

14. 判断：处理器通过虚拟地址直接访问内存。

A. 正确　　B. 错误

2.5 容错性

- 容错性是指系统或部件在发生软/硬件错误时，能够。这种能力通常会涉及一定程度的冗余。
- 基本概念
  - 可靠性
  - 平均失效时间
  - 平均修复时间
  - 可用性



$$MTTF=\frac{B_1+B_2+B_3}{3}$$

$$MTTR=\frac{A_1+A_2+A_3}{3}$$

2.5 容错性

- 错误
  - 永久性错误
  - 临时性错误
    - ☒瞬时性错误
    - ☒间歇性错误

- 冗余
  - 空间（物理）冗余
  - 时间冗余
  - 信息冗余

2.5 容错性

- 操作系统机制
  - 进程隔离
  - 并发控制
  - 虚拟机
  - 检测点和回滚机制

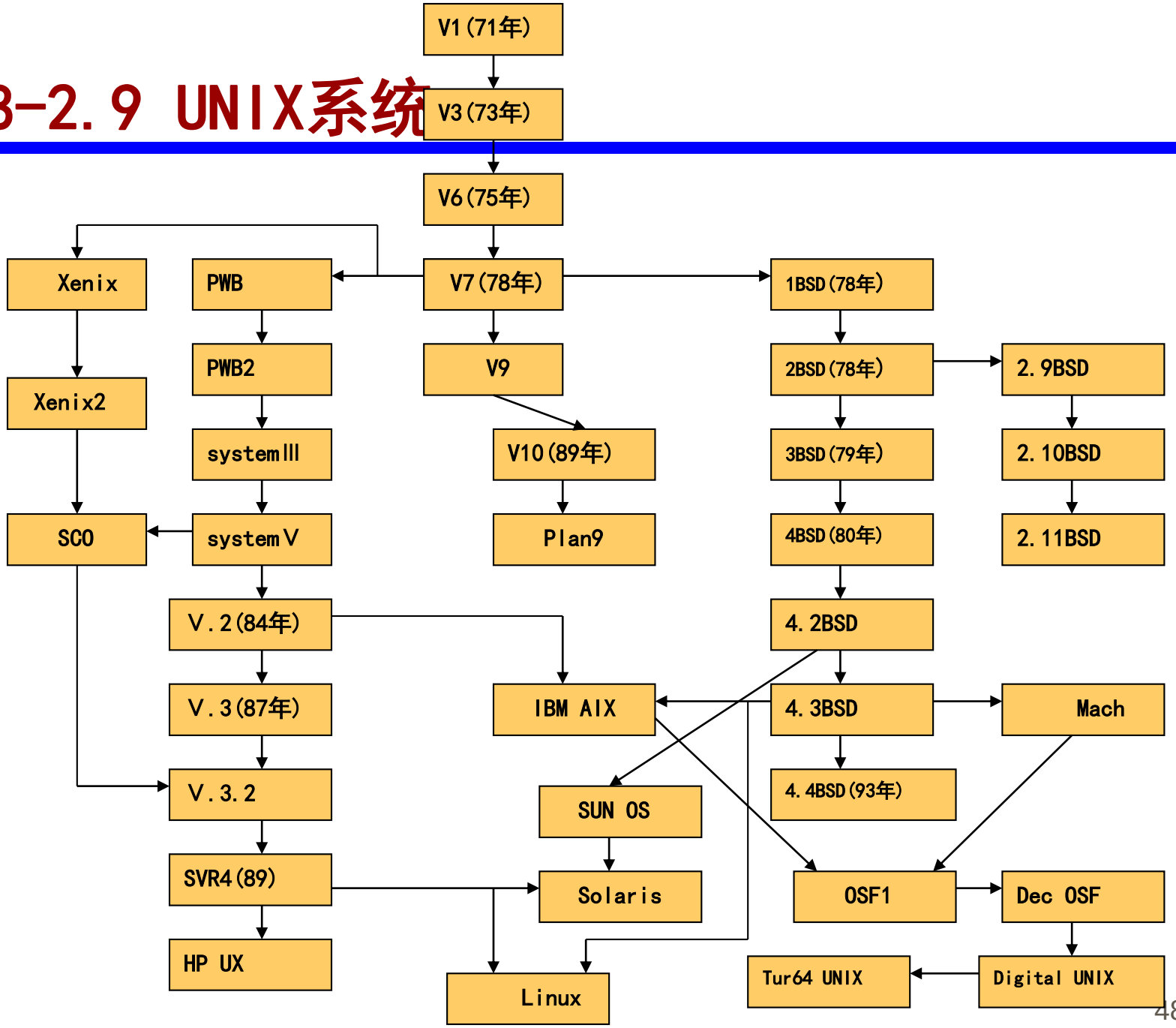
2.6 针对多处理器和多核的操作系统设计考虑因素

- 对称多处理器计算机的操作系统设计考虑因素
  - 并发进程或线程
  - 调度
  - 同步
  - 内存管理
  - 可靠性和容错性
- 多核计算机的操作系统设计考虑因素
  - SMP系统上的所有设计问题
  - 潜在的并行规模问题
    - ☒应用层并行
    - ☒虚拟机方式

2.7 微软Windows系统简介

- MS-DOS
  - Windows 3.0
- Windows 95
  - Windows 98
  - Windows Me
- Windows NT 3.1
  - NT 4.0
  - Windows 2000
  - Windows XP
  - Windows Vista
  - Windows Server 2008
  - Windows 7
  - Windows Server 2008 R2
- 面向云计算的NT版本Windows Azure

2.8-2.9 UNIX系统


V1(71年)
V3(73年)
V6(75年)
V7(78年)
Xenix
PWB
Xenix2
PWB2
systemIII
SCO
systemV
V.2(84年)
V.3(87年)
V.3.2
SVR4(89)
HP UX
V9
V10(89年)
Plan9
1BSD(78年)
2BSD(78年)
3BSD(79年)
4BSD(80年)
4.2BSD
2.9BSD
2.10BSD
2.11BSD
IBM AIX
4.3BSD
4.4BSD(93年)
Mach
SUN OS
Solaris
OSF1
Dec OSF
Linux
Tur64 UNIX
Digital UNIX

2.10 Linux操作系统

- Linux是由芬兰藉科学家Linus Torvalds于1991年编写完成的操作系统内核。
- 许多人对Linux进行改进、扩充、完善，做出了关键性贡献。Linux由最初一个人写的原型变成在Internet上由无数志同道合的程序高手们参与的一场运动。
- Linux操作系统的技术特点
  - 自由软件
  - 内核质量高
  - 高度模块化
  - 易于配置

作业

- 复习题2.1，2.2，2.3
- 补充：在单CPU和两台I/O设备（I1和I2）的多道程序设计环境下，同时投入3个作业运行。其执行轨迹如下：

  Job1：I2(30ms)，CPU(10ms)，I1(30ms),CPU(10ms)，I2(20ms)
  Job2：I1(20ms)，CPU(20ms)，I2(40ms)
  Job3：CPU(30ms)，I1(20ms)，CPU(10ms)，I1(10ms)

  设CPU，I1和I2都能并行工作，作业优先级从高到低依次为 Job1，Job2，Job3，优先级高的作业可以抢占优先级低的作业的CPU，但不可抢占I1和I2。求：
  （1）从作业投入到完成，CPU的利用率。
  （2）I1和I2的设备利用率。

重点

- 操作系统的概念及目标
- 监控程序的概念及其与处理器的关系
- 多道程序设计概念中关于CUP利用率和设备利用率的计算
- 进程的概念，进程与程序的关系