第 1 章

引　论

计算机技术发展到今天，从个人计算机到巨型计算机，无一例外都配置了一种或多种操作系统（Operating System，OS）。计算机系统由硬件和软件两部分组成，操作系统是计算机系统中最重要的系统软件，也是配置在计算机硬件上的第一层软件，是对硬件系统的首次扩充。它作为计算机硬件和计算机用户之间的中介，为应用程序提供使用基础，并成为整个计算机系统的控制中心。在现代计算机系统中，如果不安装操作系统，很难想象如何使用计算机。操作系统不仅将仅有硬件的裸机改造成为功能强、使用方便灵活、运行安全可靠的虚拟机，来为用户提供良好的使用环境，而且采用有效的方法组织多个用户共享计算机系统中的各种资源，最大限度地提高了系统资源的利用率。

1.1 操作系统的概念

1.1.1 什么是操作系统

1．引子

计算机程序是如何运行的呢？首先，需要先进行编程，而编写程序是需要以计算机程序设计语言作为基础的。对大多数编写程序的人来说，使用的编程语言称为高级程序设计语言，如 C、C++、Java 等。但由于计算机并不识别用高级语言编写的程序，因此编写好的程序还需要将它编译成计算机能够识别的机器语言程序，而这需要编译程序或汇编程序的帮助才能完成。其次，编译好的机器语言程序需要加载（调入内存并将程序中的逻辑地址变成可以执行的内存物理地址）到内存中形成一个可执行的程序，即进程（见第 2 章），而这需要操作系统的帮助。进程需要在计算机芯片 CPU（中央处理器）上执行才是真正的执行，而将进程调度到 CPU 上运行也是由操作系统完成的。最后，在 CPU 上执行的机器语言指令需要变成能够在一个个时钟脉冲周期里执行的基本操作，这需要基本硬件指令系统及相关计算机硬件的支持，而整个程序的执行过程还需要操作系统提供服务和程序语言提供运行环境。这样，一个从程序到微指令执行的过程就完成了，图 1-1 给出了程序演变的全过程。

我们只是从一个线性角度来看待从编程开始到最终输出结果的演变，而没有考虑各种因素之间的穿插和交互过程。事实上，程序可以直接用机器语言或汇编语言这种称为“低级”的语言编写。用机器语言编写的程序无须经过编译程序的翻译就可以直接在 CPU 上运行，而用汇编语言编写的程序则还需经过汇编程序翻译后才能加载执行。

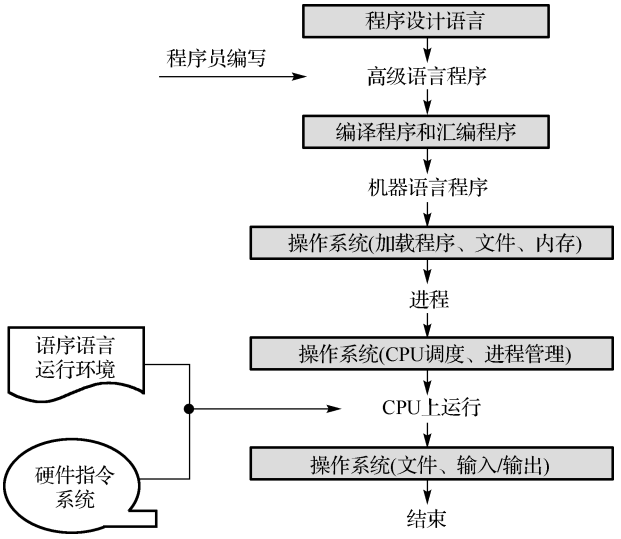

图 1-1　由程序到结果的演变

由图 1-1 的描述可以看出，程序的运行涉及四个方面：① 程序设计语言；② 编译程序；③ 操作系统；④ 硬件指令系统（计算机硬件系统）。而操作系统在程序的执行过程中具有关键作用。

2．操作系统的定义

计算机系统由硬件系统和软件系统两大部分组成，硬件系统是计算机赖以工作的实体，软件系统则保证了计算机系统的硬件部分按用户指定的要求协调地工作。

计算机硬件系统由中央处理器（Central Processing Unit，CPU）、内存储器、外存储器和各种输入输出设备组成，它提供了基本的计算机资源。只有硬件的计算机称为裸机。

计算机硬件由软件来控制。按与硬件相关的密切程度，通常将计算机的软件分为系统软件和应用软件两类。用户直接使用的软件通常为应用软件，而应用软件一般需借助系统软件来指挥计算机的硬件完成其功能。

那么，操作系统是什么呢？英文中的 Operating System 意为掌控局势的一种系统，也就是说，计算机里的一切事情均由 Operating System 来掌控（管理）。现在我们面临两个问题：第一个问题，操作系统到底是什么东西？第二个问题，操作系统到底操控（管理）什么事情？

由图 1-1 可以大致得到第一个问题的答案：操作系统是介于计算机硬件和应用软件之间的一个软件系统，即操作系统的下面是硬件平台，而上面则是应用软件（如图 1-2 所示）。

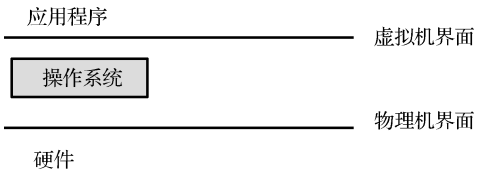

图 1-2　操作系统上下界面

下面分析第二个问题，操作系统掌控的事情当然是计算机上或计算机里发生的一切事情。最早的计算机并没有操作系统，而是直接由人来操控。随着计算机复杂性的增加，人们已经不能胜任直接掌控计算机了，于是编写出操作系统这个“软件”来掌控和管理计算机。这个掌控有着多层深远的意义。

（1）由于计算机的功能不断趋向完善和复杂，操作系统所掌控的事情也就越来越多，越来越复杂，即操作系统必须掌控计算机的所有软硬件资源，并使计算机的工作变得有序。这是早期驱动操作系统不断改善的根本原因。

（2）既然操作系统是专门掌控计算机的，那么计算机上发生的所有事情必须得到操作系统的允许，但由于所设计的操作系统不可能做到尽善尽美，因此给攻击者造成可乘之机，而操作系统设计人员和攻击者之间的博弈是当前驱动操作系统改善的一个重要动力。

（3）为了更好地掌控计算机上发生的所有事情，同时也为了更好地满足人们对操作系统越来越苛刻的要求，操作系统必须不断地完善自己。

也就是说，从计算机管理的角度看，操作系统的引入是为了更加充分、有效地使用计算机系统资源，也就是合理地组织计算机的工作流程，有效地管理和分配计算机系统的硬件和软件资源，同时注意操作系统自身的安全与完善。从用户使用的角度看，操作系统的引入是为了给用户使用计算机提供一个良好的界面，使之既方便又安全。

因此，操作系统是掌控（管理）计算机上所有事情的系统软件，它需要完成以下5种功能。

（1）控制和管理计算机系统的所有硬件和软件资源。

（2）合理地组织计算机的工作流程，保证计算机资源的公平竞争和使用。

（3）方便用户使用计算机。

（4）防止对计算机资源的非法侵占和使用。

（5）保证操作系统自身的正常运转。

长期以来，（1）～（3）项一直是操作系统定义的内容，但随着越来越多对计算机系统恶意攻击的事件以及越来越多的计算机病毒出现，操作系统自身的安全越来越受到人们的重视。所以，现在的操作系统的定义又增加了（4）和（5）两项。

任何计算机，只有在安装了相应的操作系统后才构成一台可以使用的计算机系统，用户才能方便地使用计算机。只有在操作系统的支持下，计算机的各种资源才能安全、方便、合理地分配给用户使用，各种软件（编译程序、数据库程序、网络程序以及各种应用程序等）才能安全、高效、正常地运行。操作系统性能的高低直接决定了计算机整体硬件性能能否得到充分发挥。操作系统本身的安全性和可靠性在一定程度上决定了整个计算机系统的安全性和可靠性。操作系统在整个计算机系统中的地位如图1-3所示。

由图 1-3 可以看出，计算机系统具有层次结构，其中操作系统是在硬件基础上的第一层软件，是其他软件和硬件之间的接口。因此，操作系统是最重要的系统软件，它控制和协调各用户程序对硬件的使用。实质上，用户在使用计算机时直接面对的并不是计算机的硬件，而是应用软件，由应用软件在“幕后”与操作系统打交道，再由操作系统指挥计算机完成相应的工作。

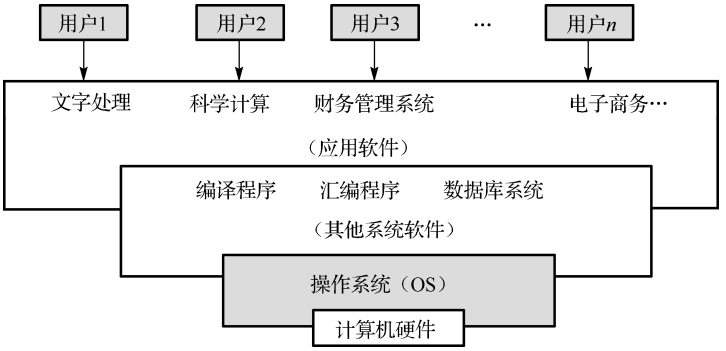

图 1-3　操作系统在计算机系统中的地位

3．操作系统的设计目标

目前存在多种类型的操作系统，不同类型的操作系统其设计目标各有侧重。一般来说，设计的操作系统应达到以下 6 个目标。

（1）方便性。操作系统应提供统一且界面友好的用户接口，以方便用户使用计算机。

（2）有效性。操作系统应能有效分配和管理计算机的软、硬件资源，使系统资源得到充分利用。此外，操作系统应能合理地组织计算机的工作流程，改善系统性能，并提高系统运行效率。

（3）可扩展性。操作系统应具备较好的可扩展性，以适应计算机硬件和计算机网络等发展的需要。可扩展性表现在能否很容易地增加新的模块。

（4）开放性。操作系统应遵循国际标准进行设计，构造一个统一的开放环境，以便不同厂家生产的计算机和设备能通过网络集成，且能正确、有效地协调工作，实现应用程序的可移植性和互操作性。

（5）可靠性。可靠性包括正确性、健壮性和安全性。操作系统除应满足正确性这一基本要求外，还应满足确保操作系统自身正常工作的健壮性，以及如何防止非法操作和入侵的安全性。

（6）可移植性。可移植性是指将程序从一个计算机环境转移到另一个计算机环境中仍能正常运行的特性。由于操作系统开发是一项非常庞大的工程，为了避免重复工作及缩短软件研发周期，现在操作系统设计已将可移植性作为一个重要目标。

1.1.2　操作系统的主要功能

为了高效地使用计算机软、硬件资源，提高计算机系统资源的利用率和方便用户使用，在计算机系统中都采用多道程序设计技术。多道程序设计技术是指内存中同时放入多道程序（进程）交替运行并共享系统资源，当一道程序（进程）由于某种原因（如输入/输出请求）而暂停执行时，CPU 立即转去执行另一道程序（进程）；这样，不仅使 CPU 得到充分利用，而且也提高了 I/O 设备和内存的利用率。在引入了多道程序设计技术后，使得操作系统具有多道程序同时（进程）运行且宏观上并行、微观上串行的特点，而操作系统也正是随着多道程序设计技术的出现而逐步发展起来的。要保证内存中多道程序的正常运行，在技术上需要解决以下 5 个问题。

（1）在多道程序之间应如何分配 CPU，使得 CPU 既能满足各程序运行的需要，又能有较高的利用率？此外，一旦将 CPU 分配给某程序后，应何时回收且如何回收？

（2）如何为每道程序分配必要的内存空间，使它们各得其所但又不会因相互重叠而丢失信息？此外，还要防止因某道程序出现异常而导致其他程序被破坏的情况出现。

（3）系统中可能有多种类型的 I/O（输入/输出）设备供多道程序共享，应如何分配这些 I/O 设备？如何做到既方便用户对设备的使用，又能提高设备的利用率？

（4）在现代计算机系统中通常都存放着大量的程序和数据，应如何组织它们才便于用户使用并保证数据的安全性和一致性？

（5）系统中的各种应用程序，有的属于计算型，有的属于 I/O 型，有些既重要又紧迫，有些又要求系统及时响应，这时系统应如何组织这些程序（作业）的工作流程？

实际上，这些问题就是操作系统的核心内容。因此，操作系统应具有以下 4 个方面的管理功能：处理器管理、存储管理、设备管理及文件管理。此外，随着网络技术的不断发展，操作系统还应具备相应的网络功能。同时，为了方便用户使用计算机，操作系统还应向用户提供方便、友好的用户接口。

1．处理器管理

处理器管理主要是指对计算机系统中央处理器（CPU）的管理，其主要任务是对 CPU 进行分配，并对其运行进行有效地控制与管理。

为了提高计算机的利用率，操作系统采用了多道程序技术。为了描述多道程序的并发执行引入了进程的概念，进程可看作正在执行的程序，通过进程管理来协调多道程序之间的关系，以使 CPU 资源得到最充分的利用。在多道程序环境下，CPU 的分配与运行是以进程为基本单位的。随着并行处理技术的发展，为了进一步提高系统并行性，使并发执行单位的粒度更细以降低并发执行的代价，操作系统又引入了线程（Thread）的概念。对 CPU 的管理和调度最终归结为对进程和线程的管理和调度，它的主要功能包括进程控制和管理、进程同步与互斥、进程通信、进程死锁、线程控制和管理以及 CPU 调度。

2．存储管理

存储管理是指对内存空间的管理。程序要运行就必须由外存装入内存，当多道程序被装入内存共享有限的内存资源时，存储管理的主要任务就是为每道程序分配内存空间，使它们彼此隔离互不干扰，尤其是当内存不够用时，要通过虚拟技术来扩充物理内存，把当前不运行的程序和数据及时调出内存，需要运行时再将其由外存调入内存。存储管理的主要功能包括内存分配、内存保护、地址变换和内存扩充。

3．设备管理

设备管理是指计算机中除 CPU 和内存之外的所有输入输出设备（I/O 设备，也称外部设备，简称外设）的管理。其首要任务是为这些设备提供驱动程序或控制程序，以便用户不必详细了解设备及接口的细节就可以方便地对设备进行操作。设备管理的另一个任务就是通过中断技术、通道技术和缓冲技术使外部设备尽可能与 CPU 并行工作，以提高设备的使用效率。为了完成这些任务，设备管理的主要功能包括外部设备的分配与释放、缓冲区管理、共享型外部设备的驱动调度、虚拟设备等。

4．文件管理

文件是计算机系统中除 CPU、内存、外部设备等硬件设备之外的另一类资源，即软件资源。程序和数据以文件的形式存放在外存储器（如磁盘、光盘、磁带、优盘）上，需要时再把它们装入内存。文件管理系统的主要任务是有效地组织、存储和保护文件，使用户方便、安全地访问它们。文件管理的主要功能包括文件存储空间管理、文件目录管理、文件存取控制和文件操作等。

5．用户接口

为了方便用户使用，操作系统向用户提供了使用接口。接口通常以命令、图形和系统调用等形式呈现给用户，前两种形式供用户通过键盘、鼠标或屏幕操作，后一种形式供用户在编程时使用。用户接口的主要功能包括命令接口管理、图形接口管理（图形实际上是命令的图形化表现形式）和程序接口管理。用户通过这些接口能方便地调用操作系统功能。

6．网络与通信管理

随着计算机网络的迅速发展，网络功能已成为操作系统的重要组成部分。现代操作系统都注重为用户提供便捷、可靠的网络通信服务。为此，网络操作系统至少应具有以下 3 种管理功能。

（1）网络资源管理。计算机联网的主要目的之一是共享资源，网络操作系统应能够实现网上资源共享，管理用户程序对资源的访问，保证网络信息资源的安全性和完整性。

（2）数据通信管理。计算机联网后，站点之间可以互相传送数据。数据通信管理为网络应用提供必要的网络通信协议，处理网络信息传输过程中的物理细节，同时通过通信软件，按照网络通信协议完成网络上计算机之间的信息传输。

（3）网络管理。包括网络性能管理、网络安全管理、网络故障管理、网络配置管理和日志管理等。

1.1.3　操作系统的基本特征

目前存在多种类型的操作系统，不同类型的操作系统有各自的特征，但它们都具有并发性、共享性、虚拟性和不确定性等共同特征。在这些共同特征中，并发是操作系统中最重要的特征，而其他三个特征都是以并发为前提的。

1．并发性

并发性（Concurrence）是操作系统最重要的特征。并发性是指两个或两个以上的事件或活动在同一时间间隔内发生（注意，不是同一时刻）。也就是说，在计算机系统中同时存在多个进程，从宏观上看，这些进程是同时运行并向前推进的；从微观上讲，任何时刻只能有一个进程执行，如果在单 CPU 条件下，那么这些进程就是在 CPU 上交替执行的。

操作系统的并发性能够有效地改善系统资源的利用率，提高系统的效率。例如，一个进程等待 I/O 时，就让出 CPU，并由系统调度另一个进程占用 CPU 运行。这样，在一个进程等待 I/O 时，CPU 就不会空闲，这就是并发技术。

操作系统的并发性会使操作系统的设计和实现变得更加复杂。例如，以何种策略选择下一个可执行的进程？怎样从正在执行的进程切换到另一个等待执行的进程？如何将内存中各交替执行的进程隔离开来，使之互不干扰？怎样让多个交替执行的进程互通消息并协作完成任务？如何协调多个交替执行的进程对资源的竞争？多个交替执行的进程共享文件数据时，如何保证数据的一致性？为了更好地解决这些问题，操作系统必须具有控制和管理进程并发执行的能力，必须提供某种机制和策略进行协调，从而使各并发进程能够顺利推进并获得正确的运行结果。此外，操作系统要充分发挥系统的并行性，就要合理地组织计算机的工作流程，协调各类软、硬件资源的工作，充分提高资源的利用率。

注意，并发性与并行性是两个不同的概念。并发性是指两个或多个程序在同一时间段内同时执行，即宏观上并行（同时执行），微观上串行（交替执行）；而并行性则是指同时执行，如不同硬件（CPU 与 I/O 设备）同时执行。

2．共享性

共享性（Sharing）是操作系统的另一个重要特征。在内存中并发执行的多个进程可以共同使用系统中的资源（包括硬件资源和信息资源）。资源共享的方式可以分为以下两种。

（1）互斥使用方式

互斥使用方式是指当一个进程正在使用某种资源时，其他欲使用该资源的进程必须等待，仅当这个进程使用完该资源并释放后，才允许另一个进程使用这个资源，即它们只能互斥地共享该资源，因此这类资源也称互斥资源。系统中的有些资源，如打印机、磁带机的使用就只允许互斥使用。

（2）同时使用方式

系统中有些资源允许在同一段时间内被多个进程同时使用，这里的“同时”是宏观意义上的。典型的可供多个进程同时使用的资源是磁盘，可重入程序（可以供多个用户同时运行的程序）也是可被同时使用的，如编译程序。

共享性和并发性是操作系统两个最基本的特征，它们互为依存。一方面，资源的共享是因为程序的并发执行而引起的，若系统不允许程序并发执行，自然也就不存在资源共享的问题。另一方面，如果系统不能对资源共享实施有效的管理，则必然会影响到程序的并发执行，甚至程序无法并发执行，操作系统也就失去了并发性，导致整个系统的效率低下。

3．虚拟性

虚拟性（Virtual）的本质含义是指将一个物理实体映射为多个逻辑实体。前者是实际存在的；后者是虚拟的，是一种感觉性的存在。例如，在单 CPU 系统中虽然只有一个 CPU 存在，且每一时刻只能执行一道程序，但操作系统采用了多道程序技术后，在一段时间间隔内，从宏观上看有多个程序在运行，给人的感觉好像是有多个 CPU 在支持每一道程序运行。这种情况就是将一个物理的 CPU 虚拟为多个逻辑的 CPU。

4．不确定性

在多道程序设计环境下，不确定性（Nondeterminacy）主要表现在以下三个方面。

（1）在多道程序环境中，允许多个进程（程序）并发执行，但由于资源等因素的限制，每个进程的运行并不是“一气呵成”的，而是以“走走停停”的方式执行。内存中的每个进程何时开始执行，何时暂停，以什么速度向前推进，每个进程需要多少时间才能完成都是不可预知的。

（2）并发程序的执行结果也可能不确定，即对同一程序和同样的初始数据，其多次执行的结果可能是不同的。因此，操作系统必须解决这个问题，即保证在相同初始条件下，重复执行同一个程序时都不受运行环境的影响，而得到完全相同的结果。

（3）外部设备中断、I/O 请求、程序运行时发生中断的时间等都是不可预测的。

1.2 操作系统的逻辑结构和运行模型

1.2.1 用户态和内核态的划分

设想一下，如果操作系统的可靠性和安全性出了问题，可能的结果是造成整个系统的崩溃，这将影响系统的所有用户程序。但如果一个用户程序的可靠性和安全性出了问题，所造成的损失只不过是该用户程序崩溃而操作系统将继续运行，这就保证了系统下的其他用户程序不受影响。

不言而喻，操作系统的重要性要远远大于用户程序。那么如何保证操作系统的重要性呢？通常的办法是采用内核态和用户态两种模式。内核态是指操作系统程序运行的状态，在该状态下可以执行操作系统的所有指令（包括特权指令），并能够使用系统的全部资源。用户态是指用户程序运行的状态，在该状态下所能执行的指令和访问的资源都将受到限制。

内核态和用户态各有优势：运行于内核态的程序可以访问的资源多，但可靠性、安全性要求高，维护管理比较复杂；用户态程序可以访问的资源有限，但可靠性、安全性要求低，编写程序和维护管理都比较简单。

一般来说，如果一个程序能够运行于用户态，那么就应该让它在用户态下运行，只有在迫不得已的情况下才允许程序在内核态下运行。凡是涉及计算机本身运行的事情都应该在内核态下运行，凡是只与用户数据和应用相关的部分则应在用户态下运行。另外，对时序要求特别高的操作，如中断等也应在内核态下完成。

那么，什么样的功能应在内核态下实现呢？首先，从保障计算机安全的角度来说，CPU 和内存的管理必须在内核态下实现。诊断与测试程序也应在内核态下实现，因为诊断和测试需要访问计算机的所有资源，否则无法判断计算机是否正常工作。I/O 管理也是一样，因为要访问各种设备和底层数据结构，所以也必须在内核态下实现。

对于文件管理来说，可以一部分放在用户态下，一部分放在内核态下。文件系统本身的管理必须放在内核态下，否则任何人都有可能破坏文件系统的结构；用户文件（程序和数据）的管理则可放在用户态下。另外用户程序、编译程序、编辑程序、网络管理的部分功能等，自然都可以放在用户态下执行。图 1-4 描述了 Windows 操作系统的内核态与用户态的界线。

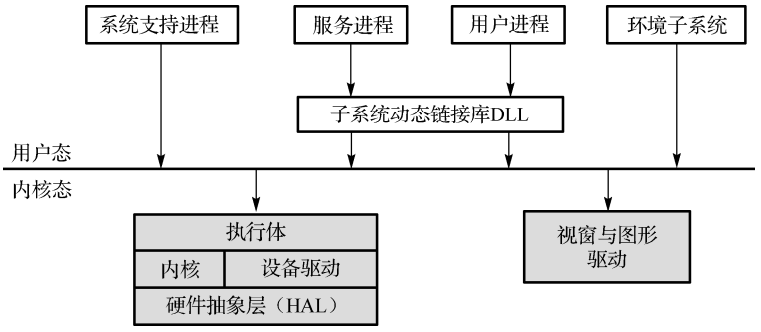

图 1-4 Windows 操作系统的内核态与用户态界线

1.2.2 操作系统的逻辑结构

操作系统的结构与操作系统的发展历史类似，经历了好几个阶段。在操作系统刚出现时，人们并没有意识到操作系统的存在，也没有将那些简单的库函数称为操作系统。那时候想到什么功能就把这个功能作为一个库函数加入进来，并没有对所有的功能进行统筹兼顾的计划。显然，那时候的操作系统是杂乱无章的，没有什么结构可言。

随着操作系统不断地发展与完善，人们对操作系统的认识逐步加深，出现了不同的逻辑结构。根据内核的组织结构，可以将操作系统的逻辑结构划分为单内核、分层式和微内核三种。

1．单内核结构

随着操作系统的不断演化而逐渐有了一些结构，各种功能归为不同的功能模块（程序），每个功能模块相对独立，却又通过固定的界面相互联系。任何一个功能模块都可以调用其他功能模块的服务，整个操作系统呈现出单内核结构。虽然有些内核的内部又划分成若干模块或层次，但内核在结构上可以看成一个整体（如图 1-5 所示）。操作系统运行在内核态下，为用户提供服务。在单内核结构中，模块间的交互是通过直接调用相应模块中的函数，而不是通过消息传递来实现的，所有模块都在相同的内核空间中运行，内核代码高度集成。

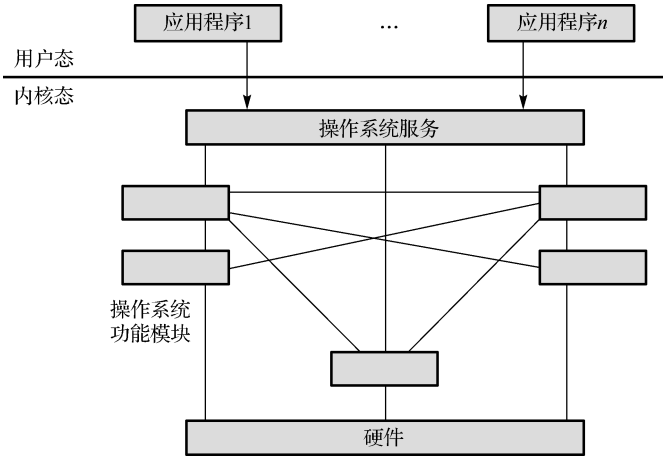

图 1-5 单内核操作系统结构

单内核结构的优点是结构紧密，模块间可以方便地进行组合以满足不同的需要，灵活性较好，效率高。单内核结构的缺点是对模块功能的划分往往不能精准确定，模块的独立性较差；模块之间的调用关系复杂，导致系统结构不清晰，正确性和可靠性不容易保证，系统维护比较困难。

2．分层式结构

单内核结构的操作系统有很多缺点：功能模块之间的关系复杂，修改任意一个功能模块可能导致许多其他功能模块都要随之修改，从而导致操作系统的设计开发困难。并且，这种没有层次关系的网状联系容易造成循环调用，从而形成死锁，导致操作系统的可靠性降低。于是，人们熟悉的层次关系被引入到操作系统的设计中。

分层式结构的设计思想是操作系统被划分成若干模块，这些模块按照功能调用次序分成若干层；每一层的程序只能使用其低层模块提供的功能和服务，即低层为高层服务，高层可以调用低层的功能，反之则不允许。按照分层结构设计的操作系统不仅系统结构清晰，而且不会出现循环调用（如图 1-6 所示）。

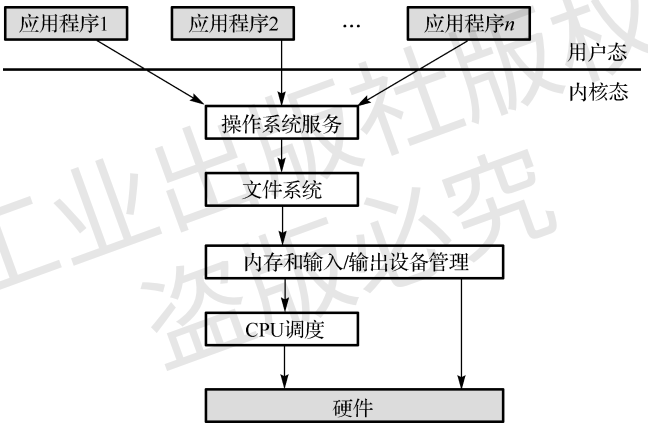

图 1-6　分层式操作系统结构

分层式结构的主要优点是按照单向调用关系以层为单位组织各模块（程序），使得模块之间的依赖、调用关系更加清晰和规范，对一个分层进行修改不会影响到其他层次，系统的调试和验证比较容易，系统的正确性更容易得到保证，系统中间的接口也会减少。当然，要在具有单向依赖关系的各分层之间实现通信则系统的开销较大，因此系统的效率会受到一定的影响。

1968 年，E.W.Dijkstra（迪杰斯特拉）按分层式结构开发了 THE 操作系统。

3．微内核结构

从图 1-5 和图 1-6 可以看出，操作系统的所有功能都在内核下运行，这会带来以下两个问题。

（1）操作系统的所有服务都需要进入内核态才能使用，而从用户态转换为内核态是需要花费 CPU 时间的，这就造成了操作系统的效率低下。在操作系统比较简单时，这个问题并不突出，但是随着操作系统功能和复杂性的增加，这个问题显得非常严重。

（2）在内核态下运行的程序可以访问所有资源，因此对其安全性和可靠性要求很高。在操作系统规模很小时，将其设计得可靠和安全并不困难，但随着操作系统越来越庞大，破坏者水平越来越高，操作系统的可靠性和安全性要求就变得很难达到。于是，人们又想出了一个办法，仅将系统的一些核心功能放入内核（因此称为微内核，此时的内核态也称为核心态），而将其他功能都移到用户态运行。这样就同时提高了效率和安全性（如图 1-7 所示）。

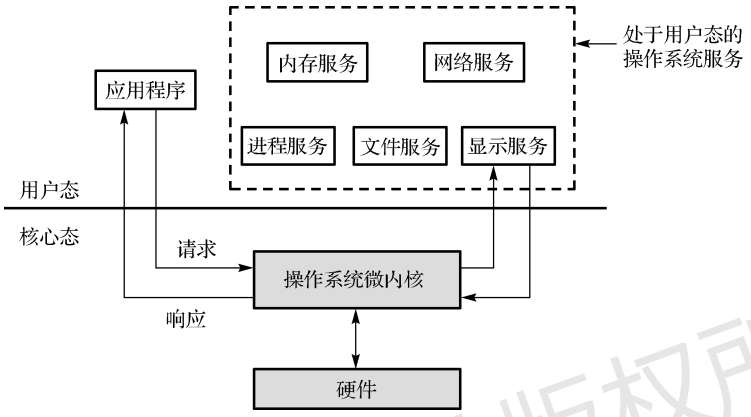

图 1-7 微内核的操作系统结构

微内核结构的设计思想是将操作系统分成两部分：一部分是运行在核心态下的微型内核，它提供系统最基本的功能，只完成极少的核心态任务，如进程管理和调度、内存管理、消息传递和设备驱动等，内核构成了操作系统的基本部分；另一部分是一组服务器进程，它们运行在用户态下，并以客户-服务器方式提供服务，如文件管理服务、进程管理服务、存储管理服务、网络通信服务等，操作系统的绝大部分功能由这组服务器进程提供。在微内核结构中，客户与服务器进程之间采用消息传递机制进行通信，通信过程借助内核实现，即由内核接收来自客户的请求，再将该请求传送至相应的服务器进程，同时，内核也接收来自服务器进程的应答，并将此应答回送给请求的用户。

微内核结构的优点是：① 为进程的请求提供了一致性接口，不必区分内核级服务和用户级服务，所有服务均采用消息传递机制提供（见 3.7 节）；② 具有较好的可扩充性和易修改性，增加新服务或替换老服务时，只需增加或替换服务器（进程）；③ 可移植性好，与 CPU 有关的代码集中在微内核中，将系统移至新平台（不同计算机）时修改较少；④ 对分布式系统提供有力支持，客户给服务器进程发送消息，不必知道服务器进程驻留在哪台机器上。

微内核结构的缺点是：虽然运行效率比分层式结构有所提升，但仍然不够高，这是因为进程之间必须通过内核的通信机制才能相互通信。

上面介绍的三种操作系统的结构各有优缺点，当前的趋势是微内核模式，至于这个微内核到底有多“微小”，则无统一的标准。例如，美国卡内基梅隆大学开发的 MACH 操作系统的内核非常小，而微软公司的 Windows XP 的内核就大得多。

1.2.3 操作系统的运行模型

操作系统本身是一组程序，这组程序按照什么方式运行称为操作系统的运行模型。操作系统有以下三种运行模型。

1．独立运行的内核模型

操作系统有自己的独立存储空间和独立运行环境。例如，有自己的核心栈，其执行过程不与应用程序（进程）发生关联。若操作系统具有这种运行模型，则当用户进程被中断或发出系统调用时，被中断运行的进程现场被保存到该进程的运行现场区，内核接收控制权并开始执行，且内核程序总是运行在自己的核心栈上。

在这种模型下，操作系统作为一个独立实体在内核模式下运行，因此内核程序要并发执行很困难，进程的概念只适合应用程序。独立运行的内核模型出现在早期的操作系统中。

2．嵌入到用户进程中执行的模型

为了提高内核程序的并发性，操作系统在创建用户进程（程序）时，为它分配了一个核心栈，该核心栈可以将操作系统的内核程序嵌入到用户进程中运行。若操作系统具有这种模型，则当用户进程发出系统调用或遇到中断时，CPU 转到内核态（核心态）下运行，控制权转移给操作系统且用户进程的现场被保护，此时启用刚被中断用户进程的核心栈来作为内核程序执行过程中调用的工作栈。需要注意的是，整个过程中只是发生了 CPU 的状态转移（从用户态转变为内核态），而进程的现场切换却并没有发生，即认为内核程序属于当前用户进程的一部分而嵌入到当前用户进程中执行。

3．作为独立进程运行的模型

操作系统的小部分核心功能（进程切换和通信、底层存储管理、中断处理等）仍然在内核态（核心态）下运行，而操作系统的大部分功能则由一组独立的服务器进程提供（见微内核结构），这组服务器进程运行在用户态下。

1.3 操作系统的形成与发展

操作系统不断改进与发展是由以下两个因素驱动的。

（1）硬件成本的不断降低。

（2）计算机的功能和复杂性不断提高。

以硬盘为例，IBM 公司制造的第一张硬磁盘直径达到 2 米，造价 100 多万美元，但容量仅 1MB，而现今一个容量 100GB 的硬盘成本只有几十美元。最初，计算机的组件虽然巨大但数量少，且功能单一；现今，随着硬件质量和数量的提升，使得计算机的功能更加全面、复杂。硬件成本的下降和计算机复杂性的提高，推动了操作系统的发展。成本的降低意味着同样的价格可以买到更先进的计算机，而复杂性的提高则需要操作系统管理的能力也随之提升。这些变化使得操作系统从最初仅仅几百或几千行代码的独立库函数，发展到如今多达 4000 万行代码的操作系统（如 Windows XP），而某些 Linux 版本的代码行数更加庞大。操作系统的发展历史可以划分为以下三个时期。

1.3.1 操作系统的形成时期

1．手工操作阶段

从第一台计算机诞生（1946 年）至 20 世纪 50 年代中期的计算机属于第一代计算机，

构成计算机的主要元器件是电子管，计算机运算速度慢、设备少，操作系统尚未出现。这时的计算机操作方式是，由用户（即程序员）采用人工操作方式直接使用计算机硬件系统，由手工控制作业（程序）的输入/输出，通过控制台开关启动程序运行。到了 20 世纪 50 年代，出现了穿孔卡片和纸带，用户（同时也是计算机的操作员）将事先编写（最初采用机器语言编写，后来采用汇编语言编写）好的程序和数据穿孔在纸带或卡片上，再将纸带或卡片装入纸带输入机或卡片输入机，并启动输入机将程序和数据输入计算机，然后通过控制台开关操控计算机运行该程序。当程序运行结束且将计算结果在打印机上输出后，操作员再卸下纸带或卡片并取走打印结果，这时后续用户才可以依次重复前述上机操作过程来运行各自的程序。手工操作阶段计算机的工作过程如图 1-8 所示。

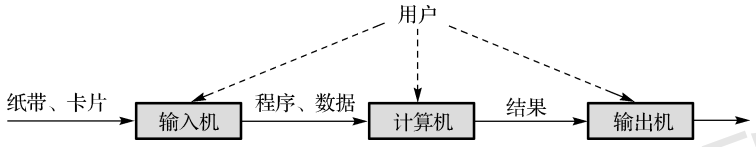

图 1-8 手工操作阶段计算机的工作过程

这个时期的代表机型为美国宾夕法尼亚大学与其他机构合作制造的第一台电子计算机 ENIAC。在 ENIAC 刚制造出来的时候，没有人知道计算机是怎么回事，所以没有操作系统的整体概念，唯一能想到的就是提供一些标准命令供用户使用，这些标准命令集合就构成了原始操作系统 SOSC（Single Operator，Single Console）。SOSC 设计的目的是满足基本功能并提供人机交互。在这种操作系统下，任何时候只能做一件事，即不支持并发和多道程序运行。由于操作系统本身只是一组标准库函数，因此无法主动管理计算机资源，只能被动地等待操作员输入命令再运行，即输入一条命令就执行一个对应的库函数。因此，这种原始操作系统根本称不上是操作系统。

手工操作方式存在着以下三方面的缺点。

（1）用户独占全部计算机资源。此时，用户既是程序员又是操作员，计算机及其全部资源只能由上机用户独占，资源利用率低。如打印机在装卸纸带和计算过程中被闲置。

（2）CPU 等待人工操作。当用户进行程序装入或结果输出等人工操作时，CPU 及内存等资源处于空闲，严重降低了计算机资源的利用率。

（3）CPU 和 I/O 设备串行工作。所有设备均由 CPU 来控制，CPU 向设备发出命令后设备开始工作，而此时 CPU 处于等待状态。当 CPU 工作时，I/O 设备处于等待 CPU 命令的状态，即 CPU 和 I/O 设备不能同时工作。

手工操作方式本身耗费了大量时间，而在这个时间里计算机只能等待，因此严重降低了计算机资源的利用率，即出现了严重的人机矛盾。随着 CPU 速度的迅速提高，导致系统规模的不断扩大，而 I/O 设备的速度却提高缓慢，人机矛盾越来越突出，手工操作越来越影响计算机的效率。

为了缓解人机矛盾，有人提出了“自动作业（程序）定序”的思想，即按顺序依次将作业（程序）装入计算机的这种转换装入工作，不再由手工操作完成，而是由计算机自动实现。作业（程序）自动转换装入技术的实现导致了操作系统的雏形——监控程序的产生。

2．监控程序阶段（早期批处理阶段）

1947 年，Bell（贝尔）实验室的 William B.Shockley、John Bardeen 和 Walter H.Brattain 发明了晶体管，开辟了电子时代新纪元，晶体管的发明极大地改变了计算机的性能。随着第二代晶体管数字计算机的出现，计算机运算速度显著提高，手工操作的低速度与计算机运算的高速度形成了强烈的反差，使得采用手工操作使用计算机方式的弊端更加突出。由于手工操作方式浪费了大量的 CPU 时间，因此严重地影响了 CPU 的利用率，使得在缓慢的手工操作方式下，计算机运行速度的提高并不明显。为了减少作业（程序）间用手工转换装入的时间，提高 CPU 的利用率，20 世纪 50 年代中期，出现了监控程序干预下的单道批处理系统，即完全由计算机控制实现作业间的转换装入。单道批处理系统是操作系统的雏形，其工作过程如图 1-9 所示。

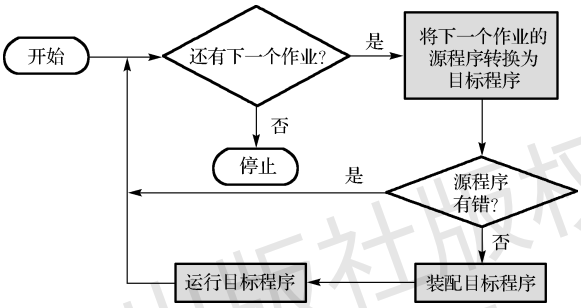

图 1-9　单道批处理系统的处理流程

操作员按照作业的性质组织一批作业，并将这批作业统一由纸带或卡片输入到磁带上，再由监控程序将磁带上的作业一个接一个装入内存投入运行，即作业由装入内存到运行结束各个环节均实现自动处理。这个处理过程首先由监控程序将磁带上的第一个作业装入内存，并将运行控制权交给该作业；其次当该作业运行结束（即处理完成）时，又将控制权交还给监控程序，再由监控程序将磁带上的第二个作业调入内存；然后按照这种方式使作业一个接一个自动得到处理，直至磁带上的所有作业全部处理完毕。由于作业的装入、启动运行等操作都由监控程序自动完成，而无须再由用户手工实施，因此 CPU 和其他系统资源的利用率都得到了显著提高。由此看出，虽然操作系统的功能还很有限，但已经开始主动地管理计算机资源了。

20 世纪 50 年代中期，第一个批处理操作系统（同时也是第一个操作系统）由 General Motors 开发并用于 IBM 701 计算机上。这个概念随后经过一系列的改进，出现了：IBM 开发的 FORTRAN 监视系统，用于 IBM 709 计算机上；IBM 开发的基于磁带的工作监控系统 IBMSYS，用于 IBM 7090 和 7094 计算机上；密歇根大学开发的 UMES，用于 IBM 7094 计算机上。

在当时，世界上最先进的计算机是 IBM 7094。IBM 将 IBM 7094 作为礼物，分别赠送给密歇根大学和麻省理工学院。密歇根大学坐落在密歇根湖和伊犁湖畔，IBM 的高管喜欢组织帆船比赛，每次帆船比赛都需要使用计算机来安排赛程、计算成绩、打印名次等。因此，IBM 在捐赠计算机时有一个要求，平时归学校使用，一旦进行帆船比赛就得停下一切计算任务为 IBM 服务。这当然使得学校很恼火，因为那个时候很难在程序执行中间停下来（由于没有现代操作系统中能够实现中断功能的硬件和软件，故无法从程序的中断处再

恢复正常执行），只要停下来，程序的执行就必须从头再来。于是，在 1959 年密歇根大学的 R.M.Graham、Bruce Arden 和 Bernard Galler，开发出当时著名的 UMES 系统（密歇根大学执行系统），它是一个能够保存中间结果的操作系统，即程序在中断执行时，将此时程序运行的中间结果和现场信息保存起来，待该程序再次投入运行时，先恢复保存的中间结果并复原中断时的现场信息，然后程序就可以不受影响的由被中断运行的断点处继续正常向下执行了。有了这个系统，密歇根大学的计算机运行基本上不受 IBM 帆船比赛带来的中断影响。

单道批处理是在解决人机矛盾的过程中出现的，它解决了依次执行的作业之间自动转换装入的问题。随着 CPU 与 I/O 设备在速度上的差异日益扩大，CPU 因等待 I/O 操作而空闲的时间越来越多，CPU 与 I/O 设备在速度上不匹配的矛盾也越来越突出。因此，单道批处理系统仍然不能很好地利用系统资源。

1.3.2 操作系统的成熟时期

1．多道批处理操作系统

第三代计算机的特点是用集成电路（Integrated Circuit，IC）代替了分立的晶体管元件，因此这段时期被称为“中小规模集成电路计算机时代”。集成电路是把多个电子元器件集中在只有几平方毫米的基片上形成的逻辑电路，第三代计算机的基本电子元件是每个基片上集成几个到十几个电子元件（逻辑门）的小规模集成电路，以及每片上几十个元件的中规模集成电路。第三代计算机已经开始采用性能优良的半导体存储器取代磁芯存储器，运算速度提高到每秒几十万至几百万次基本运算。因此，第三代计算机的运行速度更快，内存容量及设备的数量和种类都大大增加。为了更好地发挥硬件的功能并满足各种应用需求，迫切需要一个功能强大的监控程序（管理程序）来控制计算机系统的所有操作，并管理计算机系统的所有软、硬件资源。

20 世纪 60 年代，随着中断技术和通道技术的出现，多道程序设计技术成为现实。借助多道程序设计技术，人们成功设计出了具有一定并发处理能力的监控程序，并在此基础上进一步形成了功能更加强大的系统程序集合，出现了真正意义上的操作系统。典型的多道批处理操作系统是 IBM 的 OS/360(M)。

在多道批处理系统中，用户提交的作业都先放在外存上并排成一个队列，称为“后备作业队列”，然后由作业调度程序按一定的算法从后备作业队列中选择若干个作业调入内存，使它们共享 CPU 及系统中的各种资源。

为什么设计操作系统必须引入多道程序设计技术呢？在单道系统（如早期的单道批处理系统）中，任何时间内存中仅有一个作业，CPU 与其他硬件设备串行工作，导致许多资源空闲，系统性能差。图 1-10 显示了 CPU、输入机及打印机的串行工作情况。由图 1-10 可以看出，当输入机或打印机工作时，CPU 必须等待。

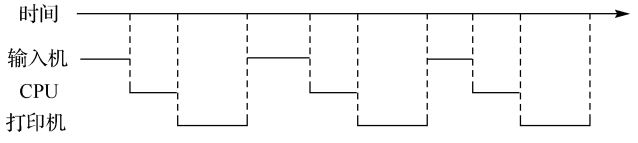

图 1-10 单道系统中程序的运行情况

多道程序设计是指允许多个程序同时进入计算机内存，并采用交替执行方法使它们运行。尽管从微观上看，这些程序交替执行轮流使用唯一的 CPU，但从宏观上看，这些程序是同时运行的。在操作系统设计中，引入多道程序设计技术可以提高 CPU 的利用率，充分发挥计算机硬件的并行能力。图 1-11 显示了多道系统中 A、B、C 三个程序的运行情况。图 1-11 中显示程序 A、B、C 在运行过程中的部分操作是并行的，即真正做到了 CPU 的执行与 I/O 设备的操作同时进行。

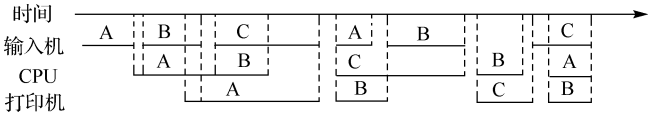

图 1-11　多道系统中 A、B、C 三个程序的运行情况

必须注意：多道程序设计中程序的数量不是任意的，它受程序中 I/O 占用时间的比例、内存大小及用户响应时间等诸多因素的影响。

进行多道程序设计时，需要解决以下三个问题。

（1）程序浮动与存储保护问题。程序浮动是指程序能从一个内存位置移动到另一个内存位置，而不影响其运行。存储保护指多个程序共享内存时，要求每个程序只能访问授权的区域。

（2）CPU 的调度和管理问题。多道程序对 CPU 的使用是交替进行的，这就要求对每个程序何时使用 CPU，以及怎样使用 CPU 等环节进行安排和管理。

（3）其他资源的管理和调度问题。多道程序除能共享 CPU 资源外，同样能共享系统中的其他资源，如何保证资源的合理分配，以及正确使用这些资源也是需要解决的问题。

多道批处理操作系统 OS/360(M)是由密歇根大学为 IBM 开发的，它运行在 IBM 第三代计算机 System/360、System/370、System 4300 上。OS/360 在技术和理念上都是划时代的操作系统，首次引入了内存分段管理的思想，它同时支持商业和科学应用，而此前的操作系统则只支持科学计算。IBM 随后对 OS/360(M)进行了改进，使其逐渐演变为一个功能强大、性能可靠的操作系统。这个改进的版本被命名为 OS/390，它提供了资源管理和共享，允许多个 I/O 同时进行，以及 CPU 运行和磁盘操作可以并发执行。OS/390 得到了广泛的商业应用。

多道批处理系统的缺点是：① 作业平均周转时间长。作业的周转时间是指从作业进入系统开始到运行完成并退出系统所经历的时间。由于每个作业都需要在作业队列中排队依次等待系统的处理和执行，因此作业的周转时间较长。② 无交互性。用户一旦提交了作业就失去了对该作业的控制（交由系统进行控制），不能再与自己的作业进行交互，这使程序的修改与调试都极不方便。

2．分时操作系统

如果说推动多道批处理系统形成和发展的主要动力是提高资源利用率和系统吞吐量的话，那么推动分时系统形成和发展的主要动力则是用户的需求。也就是说，分时系统是为了满足用户需求而形成的一种新型操作系统。用户需求具体表现在以下三个方面。

（1）人机交互。每当程序员写好一个新程序时，都需要上机进行调试。由于新编程序难免有错需要修改，因而希望能像早期使用计算机时一样对它进行直接控制，即以边运行边修改的方式对程序中的错误进行修改。因此，希望能进行人机交互。

（2）共享主机。在 20 世纪 60 年代计算机非常昂贵，不可能像现在这样每人独占一台计算机，而只能是由多个用户共享一台计算机，但用户在使用计算机时，应能够像自己独占计算机一样，不仅可以随时与计算机交互，而且感觉不到其他用户也在使用计算机。

（3）便于用户上机。用户在使用计算机时希望能通过自己的终端直接将作业送到计算机上进行处理，并能对自己的作业进行控制。

分时系统是指计算机系统由若干用户共享，在一台主机上连接了多个带有显示器和键盘等设备的终端，允许多个用户同时通过自己的终端以交互方式使用计算机。由于上述原因产生了分时系统，系统将 CPU 的执行时间分割为一个个时间段（称为时间片），然后轮流分配给每个用户使用，每个用户程序每次只能在 CPU 上运行一个时间片。由于轮转间隔的时间很短，使用户几乎感觉不到这种间隔，因此对用户来讲好像整个计算机系统由他独占。

分时的思想于 1959 年，由麻省理工学院正式提出，并在 1962 年开发出第一个分时操作系统 CTSS（Compatible Time Sharing System），并成功运行在 IBM 7094 机上，它能支持 32 个交互式用户同时工作。分时操作系统最著名的应用是 MULTICS 和 UNIX。麻省理工学院将密歇根大学开发出来的 UMES（批处理操作系统）移植到自己的 IBM 7094 中，后来大家觉得只保存中间结果还不是最好的办法，毕竟频繁地保存中间结果等帆船比赛结束后再进行重载（接着上次程序运行暂停处继续向下执行）仍然很麻烦，于是就想开发一个可同时支持多个用户上机的分时操作系统，以便一劳永逸地解决同一时间内只能允许一个用户上机的问题。1965 年，在美国国防部的支持下，麻省理工学院将密歇根大学开发过 UMES 的 R.M.Graham 请来作为主持，与来自 Bell 实验室、DEC（美国数字仪器公司）和麻省理工学院的设计人员一起开始了 MULTICS（分时操作系统）的研制。不过 MULTICS 还没有开发出来，团队内部就出现了分歧，Bell 实验室的几个人自立门户开发出了 UNIX 操作系统，并因此获得了图灵奖。而 UNIX 的出现则使得 MULTICS 从一面世就难以立足。

虽然 UNIX 的辉煌掩盖了 MULTICS 的光芒，但 MULTICS 引入了许多现代操作系统概念的雏形，如分时处理、远程联机、段页式虚拟存储器、文件系统、多级反馈调度、保护环安全机制、多 CPU 管理、多种程序设计环境等，对后来操作系统的设计有着极大的影响。

UNIX 操作系统是 AT&T 公司 Bell 实验室的 Ken Thompson 和 Dennis Ritchie 于 1969—1970 年间研制成功的一个分时操作系统。其后 Dennis Ritchie 又成功研制了 C 语言，并将 UNIX 用 C 语言重新改写并加以实现。在此后许多年里 UNIX 不断发展完善，目前它几乎已运行于从巨型计算机到微型计算机的各种硬件平台，成为多用户系统事实上的工业标准，被公认为开放式系统结构的核心。

1.3.3 操作系统的进一步发展时期

20 世纪 80 年代后，随着微电子技术的迅速发展，大规模及超大规模集成电路技术得到了广泛应用。计算机工业获得了井喷式的发展，计算机硬件不断升级换代，计算机的体系结构更加灵活多样，各种新计算机和新操作系统不断出现和发展，计算机和操作系统领域均进入了一个百花齐放、百家争鸣的时代。尤其是微型计算机得到了迅速发展，推动了微机操作系统的出现和发展。由于微型计算机迅速普及，主要使用对象也发生了改变，计算机的使用对象趋于个人化，导致进行操作系统设计时，将如何方便用户使用计算机放在了更重要的地位。系统的操作界面朝着更加方便用户与计算机交互的方向发展，传统的字

符界面逐渐被图形用户界面所取代。这段时期的微机操作系统种类繁多，功能强大，以DOS、Windows、OS/2、UNIX 等为典型代表。

20 世纪 90 年代后期，随着网络的出现，促进了网络操作系统和分布式操作系统的诞生，计算机应用逐步向网络化、分布式及智能化方向发展，推动操作系统进入一个新的发展时期。各种网络操作系统、多 CPU 操作系统、分布式操作系统及嵌入式操作系统纷纷出现，功能日新月异。操作系统的设计观念也发生了改变，由主要追求如何提高系统资源的利用率，转变到同时要考虑使用方便及提高人工效率等因素。对于网络操作系统来说，其任务是将多个计算机虚拟成一个计算机。传统的网络操作系统在现有操作系统的基础上增加了网络功能；而分布式操作系统则从一开始就把对多计算机的支持考虑进来，由于该操作系统是重新设计的操作系统，因此比网络操作系统的效率高。分布式操作系统除提供传统操作系统的功能外，还提供多计算机协作的功能。

随着计算机不断普及，操作系统的功能将变得越来越复杂。在这种趋势下，操作系统的发展面临着两个方向的选择：一是朝着微内核方向发展；二是朝着大而全的全方位方向发展。微内核操作系统虽然有不少人在研究，但获得工业界认可的并不多，这方面的代表是 MACH 系统。对工业界来说，操作系统正朝着多功能、全方位方向发展，某些 Linux 版本已有 2 亿行代码。

鉴于大而全的操作系统管理起来越加复杂，现代操作系统采取的都是模块化的管理方式，即一个小的内核加上模块化的外部管理功能。例如，最新的 SOLARIS 将操作系统划分为核心内核和可装入模块两个部分，其中核心内核分为中断管理、引导和启动、陷阱管理、CPU 管理；可装入模块分为调度类、文件系统、可加载系统调用、可执行文件格式、流模块、设备和总线驱动程序等。

Windows 将操作系统划分成内核（Kernel）、执行体（Executive）、视窗和图形驱动以及可装入模块。Windows 执行体又划分为 I/O 管理、文件系统缓存、对象管理、热插拔管理器、能源管理器、安全监视器、虚拟内存、进程与线程、配置管理器、本地过程调用等。而且，Windows 还在用户层设置了数十个功能模块，可谓功能繁多、结构复杂（如图 1-12 所示）。

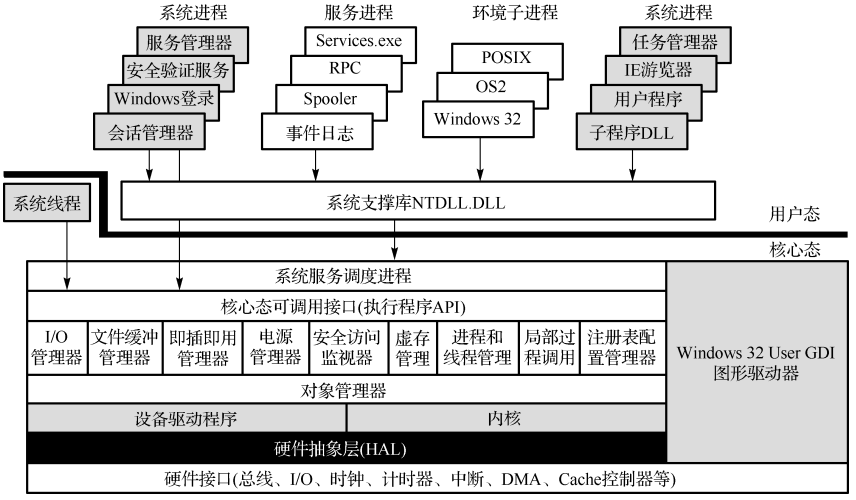

图 1-12　Windows XP 系统结构

进入 21 世纪以来，操作系统发展的一个新动态是虚拟化技术和云操作系统的出现。虚拟化技术和云操作系统虽然听上去有点不易理解，但它们只不过是传统操作系统和分布式操作系统的延伸和深化。虚拟机扩展的是传统操作系统，即将传统操作系统的一个虚拟机变成多个虚拟机，从而同时运行多个传统操作系统。云操作系统扩展的是分布式操作系统，这种扩展具有两层含义：分布式范围的扩展；分布式从同源到异源的扩展。虚拟机技术带来的最大好处是闲置计算机资源的利用，而云操作系统带来的最大好处是分散的计算机资源整合和同化。

1.4 主要操作系统的类型

为了在特定计算机硬件环境下满足对计算机的使用要求，在计算机发展的不同时期，出现了不同类型的操作系统，这些操作系统在用户面前呈现出不同的处理方式和运行特征。可以按照功能、特点及使用方式将操作系统划分为若干种类型，而批处理操作系统、分时操作系统和实时操作系统则是操作系统的三种基本类型。需要注意的是，随着操作系统技术的不断发展，操作系统的功能越来越综合化和多元化，操作系统类型的划分界限越来越模糊，现代主流的操作系统一般具有多种功能，能适应多方面应用的需要。

1.4.1 批处理操作系统

在计算机的早期使用中，批处理是各种计算机的中心主机最主要的工作方式，这些主机上配置的操作系统就是批处理操作系统。

在批处理系统中，用户提交给计算机的工作常被称为作业。一个作业通常由程序、数据和作业说明书组成。当用户将作业提交给操作员以后，为了减少作业处理过程中的时间浪费，操作员先将作业按其性质进行分组（分批），然后以组（批）为单位将作业提交给计算机，由计算机自动完成这批作业的装入、执行并输出运行结果。

根据内存中允许存放作业的个数，批处理操作系统又分为单道批处理操作系统和多道批处理操作系统。

早期的批处理操作系统是单道批处理操作系统，其特征是一批作业自动按提交顺序依次装入内存执行，每次只允许一个作业进入内存运行，先提交的作业总是先完成。在单道批处理系统中，整个系统的资源被进入内存的作业独占使用，因此资源利用率很低。例如，当运行的作业进行 I/O 操作时，由于内存中无其他作业，CPU 只能等待，导致 CPU 的利用率很低。

为了提高 CPU 和其他系统资源的利用率，在批处理操作系统设计中引入了多道程序设计技术，于是形成了多道批处理操作系统。多道批处理操作系统仍然一次自动完成一批作业的处理，但允许多个作业同时进入内存并发执行。在多道批处理操作系统中，作业的运行次序与作业的提交顺序没有严格的对应关系，先提交的作业有可能后完成，因为作业的执行顺序是由调度算法确定的。多道批处理操作系统的资源利用率很高，这是因为当一个正在运行的作业需要等待 I/O 时，操作系统就调度另一个作业执行。

现在的批处理操作系统一般指多道批处理操作系统。多道批处理操作系统的优点是资源利用率高，系统吞吐量（即系统在单位时间内完成的工作总量）大。缺点是作业平均周转时间长，用户与计算机的交互能力差，不利于程序的开发与调试。

1.4.2 分时操作系统

尽管批处理操作系统具有效率高的优点，但用户在脱机方式（用户给计算机提交了作业后就不再对作业进行控制）下工作，却无法干预自己的程序运行，不能掌握程序运行的进展情况，不利于程序调试和排错，使用计算机非常不方便。而用户对计算机系统的期望是使用方便，能人机交互，多个用户能以共享方式同时使用一台计算机。于是，在操作系统设计中，同时融合了多道程序设计技术和分时技术，出现了分时操作系统。今天，分时操作系统已经成为最流行的一种操作系统，几乎所有的现代通用操作系统都具备分时操作系统的功能。

1．工作方式

在分时操作系统管理下，计算机的工作方式是一台主机连接了若干台终端（如图 1-13 所示），每台终端供一个用户使用，这些用户通过各自的终端向系统发出命令请求，系统采用时间片轮转法的方式，来接收每个用户的命令并处理服务请求，同时通过交互方式在终端上向用户显示处理的结果，用户则根据系统回复的结果继续发出下一道命令。由此看出：

（1）分时系统为用户提供交互命令；

（2）分时系统采用分时方法，为多个终端用户服务；

（3）分时方法是将 CPU 的执行时间划分为若干个时间片；

（4）分时系统以时间片为单位轮流为各终端用户服务。

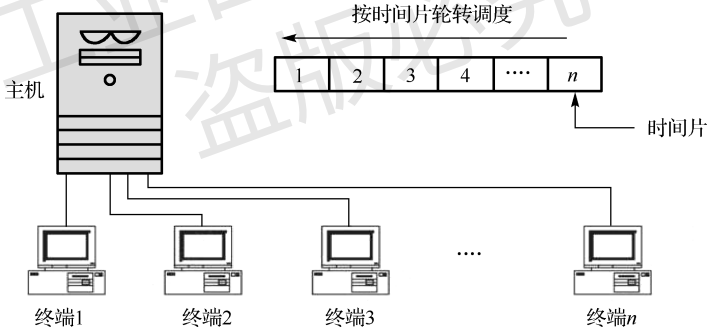

图 1-13　分时系统示意

2．时间片

操作系统将 CPU 的执行时间划分为若干个片段，这些片段称为时间片。操作系统以时间片为单位，使 CPU 轮流为各终端用户服务，每次服务的时间长度为一个时间片，如 0.1 秒（其特点是利用人的错觉，使人感觉不到 CPU 同时也在为其他终端用户服务）。每个用户程序一次只能执行一个时间片，若时间片用完而程序尚未执行完，则挂起等待（放弃 CPU，有可能还要将程序自身由内存对换至外存），直到系统分配给它的下一时间片到来时再继续执行。

由于调试程序的终端用户通常只发出简短的命令，因此每个终端用户的每次需求都能得到及时响应。也就是说，每个终端用户多路分时使用一个 CPU，不同之处在于每个终端用户都有一台联机终端。

响应时间是衡量分时操作系统性能的一个重要指标，其影响因素很多，如 CPU 的速度、联机终端的数目、时间片的长短、系统调度开销大小及对换信息量（内外存之间传送数据的数量）的多少等。响应时间可以利用可重入代码（不可被任意修改的代码，可同时供多个用户使用，由此减少了对换信息量的时间），及其他减少对换信息量的方法来进行改善，而时间片长短则可以通过控制终端的数目来调整。

3．分时操作系统的特性

分时操作系统主要具有以下 4 个特征。

（1）独立性。由于分时操作系统采用时间片轮转法，使一台计算机同时为若干个终端用户服务，因此客观效果是这些用户彼此独立，互不干扰，每个用户感觉好像自己独占了计算机。

（2）同时性（多路性）。从宏观上看，多个终端用户在同时使用一台计算机。

（3）交互性。分时操作系统的用户（终端用户）是联机用户，各终端用户可以采用人机对话的方式与自己的程序对话，直接控制程序运行。

（4）及时性。系统对终端用户的请求能够在足够短的时间内得到响应。这一特性与计算机 CPU 的处理速度、分时系统中联机终端用户的个数，以及时间片的长短密切相关。

4．分时操作系统与批处理操作系统的区别

分时操作系统和多道批处理操作系统虽然都有基于多道程序设计技术的共性，但还存在以下 4 点不同。

（1）追求目标不同。批处理操作系统以提高资源利用率与系统吞吐量为主要目标，分时操作系统则以满足用户的人机交互需求及方便用户使用计算机为主要目标。

（2）适应作业不同。批处理操作系统适用于非交互型的大型作业，而分时系统则适用于交互型的小型作业。

（3）作业的控制方式不同。批处理系统由用户利用作业控制语言（Job Control Language，JCL）书写作业控制说明书并预先提交给系统，处理过程属于脱机工作；分时系统是交互型系统，由用户从键盘上输入操作命令来控制作业，处理过程属于联机工作。

（4）资源的利用率不同。批处理操作系统可合理安排不同负载的作业，使各种资源均衡使用，利用率较高；在分时操作系统中，当多个终端作业使用同类型编译程序和公共子程序（都属于可重入代码）时，系统调用它的开销较小。

1.4.3 实时操作系统

1．实时操作系统的提出

随着计算机技术的不断发展，计算机的应用领域日益扩大。20 世纪 60 年代后期，计算机已广泛应用于控制与商业事务处理等领域。这些新出现的应用中一些任务往往在时间上带有紧迫性，要求在规定的时间内完成响应和处理，这类任务称为实时（及时）任务。虽然多道批处理操作系统和分时操作系统获得了较高的资源利用率与快速的响应时间，但对这类实时任务仍难以满足要求。即实时任务对计算机系统提出了新的要求，要求计算机系统能够及时响应外部事件的请求，并在规定的时限内完成对该事件的处理，同时有效地

控制所有实时任务协调一致运行。这种应用需求导致了实时操作系统的出现。

实时操作系统是指具有实时（及时）特性，能够支持实时控制与实时信息处理的操作系统。典型的实时系统有三种：过程控制系统、信息查询系统以及事务处理系统。过程控制系统主要用于生产过程的自动控制、自动驾驶及武器自动控制等；信息查询系统主要用于实时信息查询，即当计算机同时接收来自不同终端的提问和服务请求时，系统必须在很短的时间内做出响应；事务处理系统除能够对终端用户的实时请求做出及时响应外，还必须对系统中的数据文件及时更新，如火车或飞机订票系统、银行业务处理系统就是这类系统的典型代表。

从实现上看，实时系统分为硬实时系统与软实时系统两种类型。硬实时系统保证关键任务按时完成，这一目标要求系统内所有延迟都有限制，包括从获取存储数据到要求操作系统完成的任何操作都有严格的时间要求。硬实时系统一般没有绝大多数高级操作系统的功能，因为这些功能常常将用户与硬件分开，导致难以估计操作系统所需的运行时间。因此，硬实时系统与分时操作系统是相互矛盾的，两者不可混合使用。软实时系统是一种限制较弱的实时系统，这类系统中关键实时任务的优先级要高于其他任务的优先级。但软实时系统没有严格的时间界限，它在工业过程控制等领域的应用可能是很危险的。软实时系统实现时，需要提供硬实时系统不能支持的高级操作系统的功能，且可以与其他类型的系统混合使用。

2．实时操作系统的主要功能和特征

实时操作系统主要具有以下三个功能。

（1）实时时钟管理。提供系统日期与时间以及定时和延时等时钟管理功能。

（2）过载保护。在支持多任务的实时系统中，实时任务启动的数目在某些时刻超出系统的处理能力时，系统要通过相应的措施，例如，延迟或丢弃不重要的任务来保证实时性强的重要任务能够及时得到处理。

（3）高可靠性和安全性。提供容错能力（如故障自动复位）和冗余备份（双机，关键部件）等。

实时操作系统主要具有以下 4 个特征。

（1）及时响应和处理。实时操作系统就是为缩短系统的响应和处理时间而设计的操作系统。因此设计操作系统时，特别是实时控制系统，必须首先考虑及时响应和处理。

（2）安全可靠。尽管批处理操作系统和分时操作系统也要求安全可靠，但实时操作系统对系统的安全性和可靠性要求更高。对过程控制系统尤其是重大控制项目，如航天航空、药品与化学反应、武器控制等，任何疏忽都可能导致灾难性的后果，因此系统中必须有相应的容错机制。对信息查询和事务处理系统，则要求保证信息与数据的完整性。

（3）交互能力有限。实时操作系统是人为干预较少的监督和控制系统，虽然也提供人机交互，但交互操作应根据不同的应用对象与不同的应用要求而加以限制。用户只能访问系统中特定的专用服务程序，不能像分时操作系统一样向终端用户提供多方面的服务。

（4）多路性。实时操作系统也具有多路性，过程控制系统一般具有现场多路采集、处理和控制执行的功能，信息查询和事务处理则允许多个终端用户同时向系统提出服务请求，每个用户都会得到独立的服务响应。

3．分时操作系统和实时操作系统的区别

分时操作系统和实时操作系统主要具有以下三方面的区别。

（1）设计目标不同。分时操作系统为多用户提供一个通用的交互方式，实时操作系统则是为特殊用途提供的专用系统。

（2）交互性强弱不同。分时操作系统交互性强，实时操作系统交互性弱。

（3）响应时间要求不同。分时操作系统以用户能接受的响应时间为标准，实时操作系统则与受控对象及应用场合有关，响应时间变化范围很大。

1.4.4 微机操作系统

1．微机操作系统的特点

微型计算机的出现引发了计算机产业革命，该计算机迅速进入社会的各个领域，拥有巨大的使用量和最广泛的用户群。配备在微型计算机上的操作系统称为微机操作系统，也称个人计算机（Personal Computer，PC）系统，一般指的是安装在个人计算机上的图形界面操作系统软件。微机操作系统具有以下 4 个特点。

（1）微机操作系统基本上是根据用户使用键盘和鼠标发出的命令进行工作的，对人的动作和反应在时序上的要求并不是很严格。

（2）从应用环境来看，微机操作系统面向复杂多变的各类应用。

（3）从开发界面来看，微机操作系统为开发人员提供了一个“黑箱”，让开发人员通过一系列标准的系统调用来使用操作系统的功能。

（4）微机操作系统相对于嵌入式操作系统来说，显得比较庞大、复杂。

2．微机操作系统的分类

随着微机的 CPU 字长从 8 位、16 位、32 位，发展到 64 位，依次出现了 8 位、16 位、32 位及 64 位微机操作系统。按其性能可以划分成以下微机操作系统。

（1）单用户单任务操作系统

单用户单任务操作系统的含义是在同一时间只允许一个用户上机且只允许运行一个用户程序，计算机的所有资源归一个程序使用。在刚刚出现个人计算机的时候，从需要与别人共享小型机（分时系统）资源变为由一个人拥有个人计算机全部资源的感觉很好。由于个人计算机由一人独享，所以分时操作系统的许多功能就无须存在。因此个人计算机（微机）操作系统又回到了计算机最初的标准函数库，这就是单用户单任务操作系统，它是最简单的微机操作系统，主要配备在 8 位微机和 16 位微机上，其典型代表是 CP/M 和 MS-DOS，CP/M 主要配置在 8 位微机上，而 MS-DOS 则主要配置在 16 位微机上。

（2）单用户多任务操作系统

单用户多任务操作系统的含义是在同一时间只允许一个用户上机，但允许同时运行多个用户程序，人们在独享了个人计算机一段时间后发现，没有分时功能的操作系统使一些事情无法完成。这是因为虽然只有一个人在使用机器，但这个人可能想同时做几件事，例如，同时运行几个程序，而没有分时功能这是不可能的。于是，需要对微机操作系统进行改善，这样就将各种分时的功能又加到操作系统中，从而形成了单用户多任务操作系统。

在该系统中，由于多个并发执行的程序共享系统资源，因此系统的性能得到明显改善。目前，在 32 位微机上运行的操作系统主要是单用户多任务操作系统并支持分时操作，其中最具代表性的是 OS/2 和 Microsoft Windows 家族。

（3）多用户多任务操作系统

多用户多任务操作系统的含义是允许多个用户通过各自的终端同时使用一台主机，且允许每个用户同时运行多个程序，共享主机的各类资源。在大、中、小型计算机上配备的操作系统都是多用户多任务操作系统。目前占主流地位的 32 位微机也有不少配置了多用户多任务操作系统，其中最具代表性的是 UNIX 和 Linux。

近年来，微机操作系统得到了进一步发展，以 Windows、OS/2、Macintosh 和 Linux 为代表的新一代微机操作系统，具有图形用户界面（Graphic User Interface，GUI）、单用户多任务、多用户多任务、虚拟存储管理、网络通信支持、数据库支持、多媒体支持、应用编程支持 API 等功能。

3．微机操作系统的特性

目前使用的微机操作系统具有以下 4 方面特性。

（1）开放性。支持不同系统互联，支持分布式处理和支持多 CPU 系统。

（2）通用性。支持应用程序的独立性与在不同平台上的可移植性。

（3）高性能。随着硬件性能的提升、64 位机的逐步普及以及 CPU 速度的进一步提高，微机操作系统中引进了以前在中、大型机上才能实现的技术，支持虚拟存储器、多线程及对称处理器 SMP，促使计算机系统性能大大提高。

（4）采用微内核结构。提供基本支撑功能的内核极小，大部分操作系统功能由内核之外运行的服务器（服务程序）来实现。

1.4.5　网络操作系统

随着社会进入信息化时代，计算机技术、通信技术及信息处理技术得到了快速发展并推动了计算机网络的出现，自 1980 年以来，计算机网络已经得到了飞速的发展。现代计算机网络的格局是，通过高速网络（含 Internet），将个人计算机群、工作站、批处理系统、分时系统、有时甚至是实时系统等计算机相互连接而成为一个大的计算机网络，实现网络上资源和信息的共享。网络操作系统以及将要介绍的分布式操作系统除具有传统操作系统应有的 CPU 管理、存储器管理、设备管理和文件管理功能外，还应提供以下 4 种功能。

（1）实现网络中各节点机之间的通信。

（2）实现网络中硬、软件资源的共享。

（3）提供多种网络服务软件。

（4）提供网络用户的应用程序接口。

1．网络操作系统的工作模式

计算机网络指一些互联的自治计算机的集合。所谓自治计算机是指计算机具有独立处理能力，而互联则表示计算机之间能够实现通信和相互合作。通过计算机网络，可以将地理上分散的、具有独立功能的若干计算机与终端设备通过通信线路连接起来，实现数据通信与资源共享。网络操作系统主要有以下两种工作模式。

（1）客户机-服务器（Client/Server，C/S）模式

客户机-服务器模式中网络分成两类站点：一类作为网络控制中心或数据中心的服务器提供文件打印、通信传输、数据库等各种服务；另一类是本地处理与访问服务器的客户机。这是比较常见的 C/S 工作模式，其通用结构如图 1-14 所示。

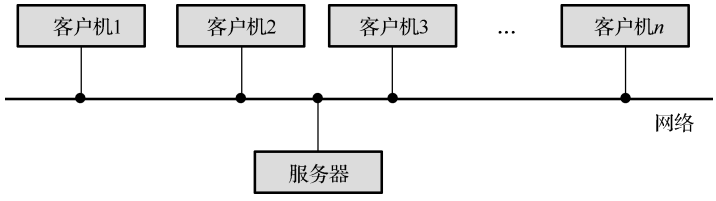

图 1-14 客户机-服务器系统的通用结构

服务器可大致分为计算服务器和文件服务器。计算服务器提供了一个接口，以接收用户所发送的执行操作请求，然后执行操作并将操作的结果返回给客户机。文件服务器则提供文件系统接口，以便客户机能够创建、更新、访问和删除文件服务器中的文件。

（2）对等（Peer-to-Peer）模式

在对等模式中，网络所有站点都是对等的，每个站点既可作为服务器，又可作为客户机。

2．网络操作系统的功能

网络操作系统是计算机网络中最重要的系统软件，它能够控制计算机在网络中传送信息和共享资源，并能为网络用户提供各种所需的服务。网络操作系统具有以下 5 种功能。

（1）网络通信。在源主机和目标主机之间实现无差错的数据传输，这是计算机网络最基本的功能。

（2）资源管理。对网络中的所有软、硬件共享资源（文件、硬盘及打印机等）实施有效管理，协调多个用户使用共享资源，保证数据的安全性和一致性。

（3）网络服务。在（1）和（2）的基础上，为方便用户而直接向用户提供多种有效服务，如电子邮件服务、文件传输服务、共享硬盘服务以及共享打印服务等。

（4）网络管理。最基本的任务是安全管理，既要确保存取数据的安全性，又要保证系统出现故障时数据的安全性。此外，还包括对网络性能进行监视，对网络使用情况进行统计，以便为提高网络性能、进行网络维护和记账等提供必要的信息。

（5）互操作能力。现代网络操作系统都提供了在一定范围内的互操作功能。所谓互操作是指在客户机-服务器的局域网环境下，连接到服务器上的多种客户机和主机不仅能与服务器通信，而且还能以透明方式访问服务器上的文件系统。而在互联网环境下的互操作，是指不同网络之间的客户机不仅能通信，而且也能以透明的方式访问网络中的文件服务器。

目前，计算机网络操作系统主要有三大主流：UNIX、Linux 和 Windows。

1.4.6 多 CPU 操作系统

1．多 CPU 操作系统引入的原因

要改善计算机系统的性能，除提高计算机元器件的速度外，另一条途径是改进计算机

系统的体系结构。最主要的方法是通过增加系统 CPU 的数量来实现任务的并行处理。

早期的计算机系统基本上都是单 CPU 系统。20 世纪 70 年代以后，打破了单 CPU 体系结构垄断的局面，出现了多 CPU 体系结构。近年来，推出的大、中、小型计算机，大多数采用了多 CPU 体系结构，甚至在高档微型计算机中也出现了这种趋势。引入多 CPU 系统（MPS）的原因主要有以下三点。

（1）增加系统的吞吐量。随着系统中 CPU 数量的增加，可以使系统在较短的时间内完成更多的工作。

（2）节省投资。在达到相同处理能力的条件下，采用具有 *n* 个 CPU 的系统要比使用 *n* 台独立的计算机更节省费用。这是因为这 *n* 个 CPU 是安装在同一个机箱内且使用同一个电源，并共享内存、打印机等资源。

（3）提高系统的可靠性。多 CPU 系统通常具有系统重构功能。当系统中的某个 CPU 发生故障时，系统能立即将在该 CPU 处理的任务迁移到其他一个或多个 CPU 上继续处理，整个系统仍然能够正常运行，只是系统性能有所降低。

可以从不同角度对多 CPU 系统分类。根据 CPU 之间耦合的紧密程度，可以将多 CPU 系统划分为紧密耦合多 CPU 系统与松散耦合多 CPU 系统。根据多 CPU 系统中 CPU 的功能是否相同，可以将多 CPU 系统划分为同构对称型多 CPU 系统（SMPS）与异构非对称型多 CPU 系统（AMPS）。

2．多 CPU 操作系统的原理及特点

目前多 CPU 系统采用了两种芯片结构：多处理器和多核。多处理器指一个体系结构上放置多个 CPU，而多核则指在同一块芯片（CPU）上放置多个核（Core），即执行单元。多核和多 CPU 的区别是多核结构更加紧凑，成本在同等执行单元数量的情况下更少，功耗更低。

多 CPU 计算机的出现，打破了单 CPU 环境下的许多操作系统设计的正确性或可靠性。本书后面章节讲述的进程和内存管理的许多策略和机制都是针对单 CPU 环境的，这些策略和机制在多 CPU 环境下要么不能正确运行，要么效率太低，或两者兼而有之。

虽然在一台计算机里安装多个 CPU 能够提升计算机的性能，但基于多种原因，CPU 的执行单元并没有得到充分使用。如果 CPU 不能正常读取数据（计算机始终存在的总线/内存瓶颈），其执行单元的利用率就会明显下降。这是因为目前大多数执行线程缺乏 ILP（Instruction-Level Parallelism，指令级并行，即多条指令同时执行）支持，这些都造成了目前 CPU 的性能没有得到全部发挥。因此，英特尔公司（Intel Corporation）采用了超线程技术让一个 CPU 同时执行多重线程。超线程技术是在一个 CPU 上同时执行多个程序来共享这个 CPU 内的资源，它可以在同一时间内让用户程序使用芯片的不同部分，从而提高 CPU 的使用效率。

例如，当一台计算机里有两个 CPU 时，虽然采用超线程技术能同时执行两个线程，但它并不像两个真正的 CPU 那样每个 CPU 都有独立的资源。当两个线程都同时需要该 CPU 中的某一资源时，其中一个就要暂时停止并让出资源，直到另一个线程使用完该资源后才能继续。因此超线程的性能并不等于两个 CPU 的性能，解决这一问题的办法就是采用多核结构。多核结构就是在一个 CPU 中布置两个执行核，即两套执行单元，而其他部

分则两个核共享。这样，由于布置了多个核其指令级并行才是真正的并行，而不是超线程结构的半并行。

此外，多个 CPU 是不可能同时启动的，必须有先后次序之分，因为不能使多个 CPU 同时执行 BIOS（基本输入输出系统）里面的指令（BIOS 不支持多线程），所以除一个 CPU 外，必须让其他 CPU 处于中断屏蔽状态，即 CPU 的启动是有次序的。因此在实际处理中有一个 CPU 被定为启动 CPU，而其他 CPU 则作为应用 CPU。

既然一个系统中有多个 CPU，则这些 CPU 之间总需要进行某种通信以实现任务协调。这种协调既可能是 CPU 本身的需要，也可能是运行在它们之上的进程和线程之间的需要。在多 CPU 之间通信自然也可以发送信号（见 3.3 节），不过这个信号不再是内存中的一个对象（变量），因为如果这样则无法及时引起另一个 CPU 的注意，而要引起其注意就要采用中断的方法（这与单 CPU 中的进程通信完全不同）。对称多 CPU 系统是通过高级可编程中断控制器（APIC）来协调这些 CPU 之间中断的机制并实现多 CPU 通信的。

在单 CPU 环境下，一次只能有一个程序正在执行；而在多 CPU 环境下，由于多个执行核或 CPU 的存在，多个程序可以真正地同时执行。那么如何保证程序执行的正确性呢？解决的方法仍然像单 CPU 系统那样，对涉及临界资源（见 3.1.3 节）的程序必须互斥执行，即保证该段程序的执行是原子操作（见 2.3.2 节）。不过这里的原子操作与单 CPU 环境下的原子操作有所不同，它必须保证跨越所有 CPU 的原子性，即一个 CPU 执行时，不允许另一个 CPU 执行此段程序（相当于单 CPU 环境下的临界区）。

多 CPU 环境下与单 CPU 环境下的最大不同是可以有多个线程或进程真正地同时执行。对于多 CPU 环境下的进程调度来说，就是使每个 CPU 有自己的就绪队列（见 2.2.2 节），该队列里面又可以按照不同的优先级分为多个子队列，就如同单 CPU 环境下的就绪队列一样。多 CPU 环境下，一个线程或进程可以排在任何一个 CPU 的就绪队列上，并且只允许排在其中一个 CPU 的就绪队列上，而不允许排在两个以上 CPU 的就绪队列上。对不同优先级线程或进程来说，调度策略也是优先级高的线程或进程优先调度，而在就绪队列中同一优先级子队列里，通常采用时间片轮转调度。

3．多 CPU 操作系统的类型

多 CPU 操作系统目前有以下三种类型。

（1）主从式。主从式操作系统安装在一台拥有主 CPU 的主机上，用来管理整个系统的资源，并分配任务给从 CPU。

（2）独立监督式。与主从式操作系统不同，独立监督式操作系统中，每个 CPU 均有各自的管理程序（核心）。

（3）浮动监督式。该方式中有一台 CPU 作为执行操作系统全面管理功能的“主 CPU”，但根据需要，“主 CPU”是可浮动的，即可以从一台 CPU 切换到另一台 CPU。

1.4.7 分布式操作系统

1．分布式系统简介

分布式系统是通过网络将多个分散的处理单元连接起来，并在分布式处理软件的支持下构成一个整体而形成的系统。在分布式系统中，系统的处理和控制分散在系统的各个处

理单元上，系统的所有任务可以动态地分配到各个处理单元上执行。分布式系统在整个系统范围内实现资源的动态分配和管理，各个处理单元既高度自治，又相互协同，能够有效控制和协调多个任务的并行执行。若分布式系统的每个处理单元是计算机，则称其为分布式计算机系统；若处理单元只是 CPU 和存储器，则称其为分布式（处理）系统。分布式操作系统与处理和控制功能都高度集中在一台计算机上的单机操作系统相比，其主要区别在于资源管理、进程通信和系统结构等方面。

分布式系统中的各台计算机之间没有主从之分，且任意两台计算机都可以通过通信交换信息。系统的资源为所有用户共享，系统中的若干台计算机可以互相协作来完成一个共同的任务。在分布式系统中，有一个全局的操作系统称为分布式操作系统，它负责整个系统（包括每个处理单元）的资源分配、调度、任务划分、信息传输及协调控制等工作，并为用户提供统一的界面和标准接口。系统运行过程中每个操作具体在哪个处理单元上执行，使用哪个处理单元的资源都由分布式操作系统决定，用户无须知道。也就是说，系统的访问过程对用户是透明的，尽管分布式系统实际上由多个 CPU 组成，但在用户看来，就像是普通的单 CPU 系统。

2．分布式系统发展的原因

分布式系统的迅速发展，主要有以下 4 个原因。

（1）它可以解决组织机构分散而数据需要相互联系的问题。

（2）如果一个组织机构需要增加新的相对自主的组织单位来扩充机构，则分布式数据库系统可以在对当前机构影响最小的情况下进行扩充。

（3）均衡负载的需要。采用使局部应用达到最大的原则来进行数据的分解，这使得各 CPU 之间的相互干扰降到最低。因此负载在各 CPU 之间进行分担也可以避免临界瓶颈。

（4）相同规模的分布式数据库比集中式数据库系统可靠性更高。

3．分布式系统的功能

分布式系统除具有通常操作系统的主要功能外，还应该包括以下 5 种功能。

（1）分布式进程通信。由分布式操作系统所提供的一些通信原语（见 2.3.2 节）来实现。与计算机网络类似，分布式操作系统中必须有通信规程，计算机之间的发信、收信都将按规程进行。

（2）分布式文件系统。允许通过网络互联使不同计算机上的用户共享文件，即能让运行它的所有主机共享，并可以管理操作系统内核及文件系统之间的通信。

（3）分布式进程迁移。由进程原来运行的计算机（称为源计算机）向目的计算机（准备迁往的计算机）传送足够数量的有关进程状态的信息，使进程能在另一台计算机上运行。

（4）分布式进程同步。分布式系统中各 CPU 没有共享内存和统一的时钟，因此分布式进程同步必须对不同 CPU 上所发生的事件进行排序，还应该配置性能较好的分布式同步算法和机制。

（5）分布式进程死锁。在分布式系统中，也可能会因进程竞争资源而引起死锁（见 3.8 节）。因此，也需要对进程死锁进行预防和处理。

4．分布式系统与网络系统的区别

对分布式系统来说，除具有网络操作系统的功能之外，还因各节点机不存在主从或层次关系，因此增加了控制机构的复杂性。首先，由于各节点的自治性，它们之间发生冲突的概率要高得多，使得同步问题变得更加复杂，死锁问题也难以处理；其次，由于系统透明性的要求，使得系统故障的检测与用户操作的检查都变得更加困难。分布式系统与网络系统的区别主要表现在以下 5 个方面。

（1）分布性。在分布式系统中有一个统一的分布式操作系统，由它统一管理整个分布式系统的资源；而网络系统中每个节点都可以有自己的网络操作系统。

（2）并行性。分布式操作系统可以将任务动态分配到不同的处理单元上并行处理；而网络操作系统中每个用户的任务通常在本地处理。

（3）透明性。分布式系统的访问过程对用户是透明的，例如，用户要访问某个文件，只需要提供文件名而不需要知道文件存放在哪个站点；而网络系统的访问过程对用户是不透明的，若用户要访问某个文件，则必须提供文件名和文件存放的位置。在分布式系统中，系统结构对用户是透明的，用户把整个系统看成是一个单一的计算机系统，完全看不到系统是由多台计算机构成的事实；而网络系统中，系统结构对用户是不透明的，用户确切知道系统是由多台计算机构成的这一事实。

（4）共享性。在分布式系统中，各站点的资源可以供全系统共享；而在网络系统中，一般只有服务器上的部分资源可以供全网共享。

（5）健壮性。分布式系统具有健壮性，若某站点出现故障，则在该站点上处理的任务可以自动迁移到其他站点完成；而网络系统的健壮性相对要差一些，若服务器出现故障，则有可能导致全网瘫痪。

著名的分布式操作系统包括由荷兰自由大学开发的 Amoeba，以及由 AT&T 公司 Bell（贝尔）实验室开发的 Plan 9 等。

1.4.8 嵌入式操作系统

随着信息技术的快速发展与 Internet 的广泛使用，出现了多种类型的信息电器产品，所有信息电器都与嵌入式（计算机）系统应用有关。在嵌入式（计算机）系统中，硬件不再以物理独立的装置和设备形式出现，而是大部分或者全部隐藏和嵌入到各种应用系统中，实现软、硬件的一体化。由于嵌入式系统的应用环境与其他计算机系统的应用环境存在较大的差别，因此推动了嵌入式软件和嵌入式操作系统的出现。

根据 IEEE（电气和电子工程师协会）的定义，嵌入式系统是“控制、监视或者辅助装置、机器和设备运行的装置”。从中可以看出嵌入式系统是软件和硬件的综合体，还可以涵盖机械等附属装置。目前国内一个普遍被认同的对嵌入式系统的定义是：以应用为中心，以计算机技术为基础，软、硬件可裁剪，适用于应用系统对功能、可靠性、成本、体积、功耗要求严格的专用计算机系统。

嵌入式操作系统是运行在嵌入式应用环境中，对整个系统以及所操作和控制的各种部件、装置等资源进行统一协调、处理、指挥和控制的系统软件。由于它仍然是一种操作系统，因此具有通用操作系统的功能，包括与硬件相关的底层软件及操作系统核心功能，功能强大的还提供图形界面、通信协议、小型浏览器等。但由于嵌入式操作系统硬件平台的

局限性、应用环境的多样性与开发手段的特殊性，它与一般操作系统相比又有很大不同，主要区别和特点表现在以下 4 个方面。

（1）微型化。嵌入式操作系统运行的硬件平台可用内存少，往往不配置外存，微 CPU 字长短且运算速度有限，能够提供的资源较少，外部设备和被控制设备千变万化。因此无论从性能还是从成本角度考虑，都不允许它占用很多资源。即嵌入式操作系统的系统代码量要少，应在保证应用功能的前提下，以微型化作为出发点来设计嵌入式操作系统的结构与功能。

（2）可定制。嵌入式操作系统运行的平台多种多样，应用更是五花八门。因此嵌入式操作系统表现出专业化的特点，并要求它能运行在不同微 CPU 平台上，能针对硬件变化进行结构与功能的配置，以满足不同的应用需求。

（3）实时性。嵌入式操作系统广泛应用于过程控制、数据采集、传输通信、多媒体信息及要求迅速响应的场合，且实时响应要求严格。因此，实时性是其主要特点之一。

（4）易移植性。为了提高系统的易移植性，通常采用硬件抽象层（Hardware Abstraction Level，HAL）和板级支撑包（Board Support Package，BSP）的底层设计技术。HAL 提供了与设备无关的特性，能屏蔽硬件平台的细节和差异，向操作系统上层提供统一接口，保证了系统的可移植性。

嵌入式操作系统与应用环境密切相关，至今已有几十种嵌入式操作系统面世。按应用范围划分，可把它分成通用型嵌入式操作系统与专用型嵌入式操作系统，前者可适用多种应用领域，比较有名的有 Windows CE、VxWorks 和嵌入式 Linux；而后者则面向特定的应用场合，如适用于掌上计算机的 Plan OS、适用于移动电话的 Sysbian 等。

1.5 操作系统安全性概述

1.5.1 操作系统安全的重要性

随着计算机应用领域的不断扩展，人们对计算机系统的依赖越来越高。为了使用方便，越来越多的组织和个人将一些重要的信息保存在计算机系统中，并通过计算机网络进行信息传输。在这种情况下，如何确保信息在计算机中安全地存放，以及信息如何安全地在网络中传输，已成为重要且必须解决的问题。在影响信息安全的诸多因素中，操作系统的安全尤为重要，这是因为操作系统是计算机系统资源的管理者和人们使用计算机的接口，若没有相应的安全保护措施，整个计算机系统的安全自然得不到保证。没有操作系统的安全，就谈不上整个系统的安全，就不能解决计算机网络、数据库及其他各种应用中的信息安全问题。

操作系统安全是整个计算机系统安全的基础，也是计算机系统安全的必要条件。操作系统的安全包括对系统重要资源（存储器、文件系统）的保护和控制，是计算机系统安全的原始基石。现代操作系统允许资源共享，这就使被共享的资源成为攻击的首要目标，一旦攻破操作系统的防御，就获得了计算机系统保密信息的存取权。

操作系统的安全问题十分复杂，不仅与计算机系统软、硬件的安全性有关，而且受操作系统构建方式等多方面因素的影响。操作系统中往往存在着多个风险点，任何一个风险

点出问题，都可能导致出现安全事故。尤其需要注意的是，操作系统的安全问题往往呈现出动态性特征，因为随着信息技术的发展，攻击者的攻击手段层出不穷，也许今天的主要攻击手段，到了明天就会被一种新的攻击手段替代。因此对待操作系统安全性问题，人们不可能找到一种一劳永逸的解决方案。

1.5.2 操作系统的安全观点

操作系统的安全性对不同的角色来说，其关注点也有所不同。对于设计者来说，主要关注的问题有：操作系统的安全机制如何从一开始就纳入到操作系统中？操作系统中哪些资源需要保护？如何建立最可靠的安全机制？如何分层次、分步骤地实现安全机制？难度如何？用户是否对这些安全机制感兴趣？如何对操作系统的安全进行评价？是否对操作系统提供不同的安全级别以供用户选择等。

对于使用者来说，其关注的内容包括安全机制是否方便？费用如何？耗时多少？如何保护操作系统的资源？这些安全机制是否可靠？有没有漏洞？对安全漏洞有无后期修补措施等。这就要求在设计操作系统时通盘考虑，综合设计者和使用者所关注的内容，既要考虑与应用系统的安全机制无缝连接，也要考虑操作系统可能使用的各种安全机制。对于这些众多的安全性因素，往往需要采用系统工程的方法予以解决。例如，可以采用层次化方法对系统安全的功能按层次进行组织，即首先将操作系统的安全问题划分成若干个安全主题作为最高层；然后将各安全主题划分成若干个子功能作为次高层，依次划分下去，直至最低层为一组最小可供选择的安全功能，通过使用多层次的安全功能来保证整个操作系统各方面的安全。

目前在实现操作系统安全工程时，都遵循了适度安全准则，即根据实际需要实现适度的安全目标。这样做的理由是：① 对安全问题的全面覆盖难度以实现；② 对安全问题进行全面覆盖所需要的资源和成本令人难以接受。

1.5.3 实现操作系统安全性的基本技术

面对各种安全威胁，操作系统必须采用有效的安全机制，以抵御人为恶意攻击可能对操作系统造成的危害，确保操作系统的安全。否则，来自操作系统本身的漏洞会使整个计算机系统的安全措施变得毫无用处。目前有效的安全机制主要有以下 7 个。

（1）身份鉴别。常作为计算机和网络安全的第一道防线，操作系统使用注册和登录机制对用户的身份进行核准和验证。

（2）存取控制。当主体访问一个客体时，操作系统根据该客体的存取控制表检查该主体是否有相应的访问权限。

（3）最小特权管理。为了操作系统安全，每个主体只能拥有与其操作相符的必要最小特权集，不能赋予用户、进程超越其执行任务所必须特权之外的任何其他特权。

（4）硬件保护。采用硬件方式对内存和进程运行进行保护。

（5）安全审计。对系统中有关安全的活动进行记录、检查及审核，即作为一种事后追踪的手段来保持操作系统的安全。

（6）入侵检测。这是一种积极主动的安全防护技术，即识别针对计算机系统或网络资源的恶意攻击企图和行为，并对此做出反应。

（7）数据加密。对系统中存储和传输的数据进行加密，只有被授权者才能对加密后的数据进行解密。这样，即使攻击者截获了数据也无法知道其内容，从而保证了系统信息资源的安全。

操作系统在最初设计时根本没想到有人会破坏计算机系统，当后来发现有人试图利用计算机进行非法操作时，就迫不得已修改操作系统来添加安全防范功能。每当操作系统提高安全措施，攻击者也会改进他们的攻击手段，这样循环往复，造成了操作系统安全水平和攻击者的攻击水平不断交替上升。随着人们不断提高对信息安全的重视程度，如何构建可靠、可用和安全的操作系统成为一个十分重要的课题。而对可靠、可用和安全的追求无疑将使操作系统变得更加复杂，操作系统的规模也随之不断地扩大，从 UNIX 的 1400 行代码，到 Windows XP 的 4000 万行代码，这完全是一种爆炸式增长。而爆炸式增长的后果就是操作系统设计已经变得极为复杂和困难，安全性越来越难以保证。关于操作系统安全性的讨论已经有专门的课程讲述，在此不做深入讨论。

1.6 操作系统运行基础

现代计算机系统结构如图 1-15 所示，它由 CPU 和若干设备控制器通过共同的总线相连而成，该总线提供了对共享内存的访问。每个设备控制器负责一种特定类型的设备，如磁盘控制器、视频控制器、键盘控制器等。CPU 与设备控制器可并发工作，并竞争使用内存，为了确保对共享内存的有序访问，需要提供内存控制器来实现对内存的同步访问控制。操作系统作为系统的管理程序，它的运行需要相应的硬件环境支持，下面主要介绍支持操作系统运行的 CPU 硬件特性、中断和系统调用等概念。

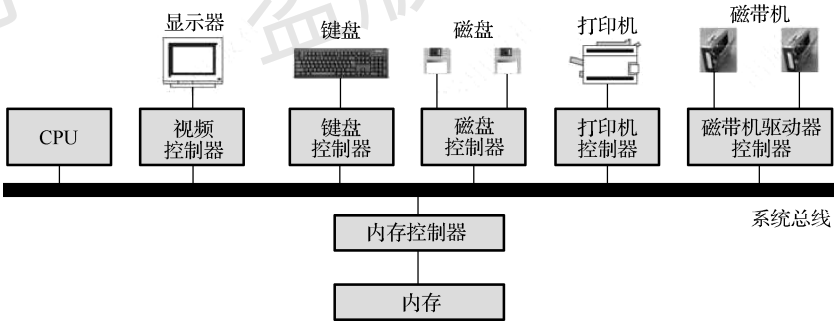

图 1-15　现代计算机系统结构

1.6.1　处理器及工作模式

1．中央处理器

CPU（中央处理器）一般由集成在一片或几片大规模或超大规模集成电路中的运算器、控制器、寄存器和高速缓存构成。

运算器主要负责指令中的算术和逻辑运算，由算术逻辑单元（ALU）、累加寄存器、数据缓冲寄存器和条件状态寄存器等组成，运算器是计算机计算的核心。

控制器是计算机系统的控制中心，主要控制程序运行的流程，包括取指令、维护 CPU 状态、实现 CPU 与内存的交互等。

寄存器是 CPU 内部指令处理过程中暂存数据、地址以及指令信息的存储设备，在计算机的存储系统中它具有最快的访问速度。

高速缓存处于 CPU 和物理内存之间，一般由控制器中的内存管理单元（MMU）来管理，它的访问速度快于内存但低于寄存器。利用程序执行的局部性原理使得高速的指令处理和低速的内存访问得以匹配，从而提高了 CPU 的利用率。

一个计算机系统中可能只有一个 CPU，也可能有多个 CPU。只包含一个 CPU 的计算机系统称为单 CPU 系统，含有多个 CPU 的计算机系统称为多 CPU 系统。

CPU 内部的寄存器通常分为以下两类。

（1）用户可见寄存器

用户可见寄存器通常对所有程序都适用，包括系统程序和应用程序。这类寄存器由高级语言编译器（编译程序）通过算法分配并使用，以减少程序访问内存的次数。用户可见寄存器主要包括以下四种。

① 通用寄存器。可根据程序设计者指定的功能存放数据，或者作为某种寻址方式所需的寄存器使用。通用寄存器可参与算术、逻辑运算，并保存运算结果。

② 数据寄存器。用于存放操作数，通常被程序员分配给各种函数。

③ 地址寄存器。用于存放数据或指令的内存地址。

④ 条件码寄存器。条件码是 CPU 根据运算结果由硬件设置的位，用于保存 CPU 操作结果的各种标记。通过对条件码进行测试，可以使执行转入当前运行程序的不同分支。

（2）控制和状态寄存器

控制寄存器被 CPU 控制部件使用，主要用于控制 CPU 的操作。大部分控制寄存器对用户是不可见的，一部分控制寄存器可在某种特权模式（由操作系统使用）下访问。计算机系统中的控制寄存器主要包括以下 5 种。

① 程序计数器（PC）。记录将要取出的指令地址，程序计数器具有计数的功能，当遇到转移指令时 PC 的值可被修改。

② 指令寄存器（IR）。存储最近取出的且马上就要执行的指令。

③ 存储器地址寄存器（MAR）。存放欲访问的内存单元的地址。

④ 存储器数据寄存器（MDR）。存放欲存入内存中的数据或最近由内存中读出的数据。

⑤ 程序状态字寄存器（Program Status Word，PSW）。记录 CPU 的运行模式和状态信息等，如中断允许/禁止位、CPU 优先级、运行模式（内核态还是用户态）以及其他各种控制位等。

在支持多种中断的 CPU 中，通常还有一组中断寄存器，每个中断寄存器指向一个中断处理程序。

操作系统必须了解所有的寄存器，在多道程序运行环境下，操作系统经常需要暂停正在运行的程序，并启动另一个程序运行。为了保存和恢复被中断程序的现场信息，每次中断当前程序运行时，操作系统必须保存所有寄存器的内容。当被中断的程序再次运行时，通过重新装入这些保存的寄存器内容来恢复被中断程序中断时的现场信息，以便使该程序能够从中断处继续正常运行。

2．特权指令

在多道程序环境下，为了保障计算机系统的运行安全，将计算机系统中的指令分为两类：特权指令和非特权指令。能引起系统损害的机器指令称为特权指令，否则称为非特权指令。在操作系统模式（内核态）下可执行特权指令和非特权指令，在用户模式（用户态）下只能执行非特权指令。在操作系统内核中，使用特权指令来对系统资源进行分配和管理，包括改变系统工作方式，检测用户的访问权限，修改虚拟存储管理的段表、页表（见 4.7 节），完成进程的创建和切换等。特权指令拥有最高权限，如果使用不当则可能导致系统崩溃。因此特权指令不直接提供给用户使用。

在单用户单任务的计算机系统中，因为运行程序不会影响其他程序（内存仅有该运行程序），所以不需要设置特权指令。在多任务的计算机系统中，特权指令是必不可少的，如中断屏蔽指令，建立存储保护指令，启动设备指令，修改虚拟存储管理的段表、页表等指令。特权指令主要有以下 4 种。

（1）有关 I/O 的指令。

（2）访问程序状态字寄存器的指令。

（3）存取特殊寄存器（如用于内存保护的寄存器）的指令。

（4）其他访问系统状态与直接访问系统资源的指令等。

特权指令的规定既保障了系统的安全，也使操作系统拥有了对计算机系统中所有软、硬件资源的控制权和管理特权。

3．CPU 状态

在 CPU 运行过程中，CPU 时而执行操作系统程序，时而执行用户程序。那么 CPU 如何知道当前执行的是哪种程序呢？这就依赖于 CPU 状态的标识。在多道程序环境下，操作系统引入程序状态字（PSW）来区分 CPU 的不同状态，即根据运行程序对资源和机器指令的使用权限将 CPU 设置为不同的工作状态。CPU 的工作状态也称 CPU 模式，多数系统将 CPU 的工作状态划分为两种：内核态和用户态。

（1）内核态

内核态也称为管态或核心态，是指操作系统程序运行的状态。当 CPU 处于内核态时，可以执行包括特权指令在内的全部指令并使用系统所有资源，当 CPU 处于内核态时具有改变 CPU 工作状态的能力。

操作系统的内核指令一般为特权指令。操作系统的内核是计算机上配置的底层软件，主要包括与硬件密切相关的模块，如系统时钟管理程序、中断处理程序与设备驱动程序等，以及一些操作系统中运行频率较高的程序，如进程管理、内存管理和设备管理的相关程序等。不同的操作系统，其内核的定义也有所不同，大多数操作系统的内核主要包括以下 4 方面内容。

① 时钟管理。时钟是计算机系统中最重要的外部设备，操作系统需要通过时钟为用户提供标准的系统时间。此外，通过时钟中断可以实现进程的切换，例如，在分时系统中，通过时间片轮转来实现进程的切换。

② 中断机制。在中断机制中只有小部分功能属于操作系统内核，这部分功能主要包括中断现场的保护与恢复、转移控制权到中断处理程序等。

③ 原语。操作系统的底层通常是一些可被调用的公用小程序，每个小程序完成一个规定的操作。这些小程序一般是操作系统中最接近硬件的部分，而且它们的运行具有原子性，即其运行过程不允许被中断，所以被称为原语（Atomic Operation）。

④ 用于系统控制的数据结构管理。系统中用来记录系统各种状态信息的数据结构有很多，如作业控制块、进程控制块、消息队列、缓冲区、内存分配表等，对这些数据结构的访问和维护必须由操作系统内核来完成。

（2）用户态

用户态也称为算态或目态，是指用户程序运行的状态。当 CPU 处于用户态时只能执行非特权指令，并且只能访问当前运行进程（运行的用户程序）的地址空间，这样才能有效地保护操作系统内核及内存中其他用户程序不受该运行进程（程序）的侵害。

（3）内核态与用户态的转换

中断和异常是 CPU 状态从用户态转换到内核态（核心态）的唯一途径。用户程序在执行中发生中断或异常时，CPU 响应中断或异常并交换程序状态字（暂停用户程序的执行，准备执行相应的中断处理程序），此时会导致 CPU 的状态从用户态转换为内核态。而中断或异常处理的程序（中断处理程序）则运行在内核态下。

在操作系统层面上，通常关心的是内核态和用户态的软件实现和切换。当 CPU 处于内核态时，可以通过修改程序状态字（PSW）直接进入用户态运行。当 CPU 处于用户态时，如果需要切换到内核态，则一般是通过访管指令或系统调用来实现。访管指令或系统调用是一条具有中断性质的特殊机器指令，用户程序使用它们通过中断进入内核态来运行操作系统程序完成指定的操作。访管指令或系统调用的主要功能表现在以下三个方面。

① 通过中断实现从用户态到内核态的改变。

② 在内核态下由操作系统代替用户完成其请求（用户指定的操作）。

③ 操作系统完成指定操作后再通过修改程序状态字（PSW）由内核态切换回用户态。

（4）程序状态字（PSW）

程序状态字寄存器是计算机系统的核心部件，也是控制器的一部分，主要为处理中断而设置。中断发生时，系统中断当前运行程序的执行，并将该程序的 PSW 及运行现场信息保存于内存，然后取出新程序（中断处理程序）的 PSW 来响应中断。新的 PSW 指出，为处理该中断所应执行的中断处理程序，据此 CPU 执行这个中断处理程序。一旦此中断处理程序执行完毕，系统又从内存取回所保存的被中断程序的 PSW 及运行现场信息，复原被中断程序的运行环境，然后恢复被中断程序的执行。

每个 CPU 都有专门的程序状态字寄存器来存放 CPU 的状态信息。一个程序占用 CPU 运行时，该程序的 PSW 将送入程序状态字寄存器，程序状态字寄存器用来存放两类信息：一类是反映当前指令执行结果的各种状态信息，称为状态标志，如进位标志位（CF）、溢出标志位（OF）、结果正负标志位（SF）、结果是否为 0 标志位（ZF）、奇偶标志位（PF）等；另一类存放控制信息，称为控制状态，如中断标志位（IF）、CPU 的工作状态位（内核态还是用户态）等。

PSW 主要用来控制指令的执行顺序，并保留与运行程序相关的各种信息。不同 CPU 中控制寄存器的组织方式可能不同，CPU 中可以不设置专门的程序状态字寄存器，但总会有一组控制寄存器与状态寄存器实际上承担着程序状态字寄存器的作用。

1.6.2 中断技术

中断（Interrupt）机制是实现多道程序并发执行的重要条件，也是现代计算机系统的重要组成部分。中断是指程序在执行过程中 CPU 对系统发生的某个事件做出的一种反应。中断具有以下三个特点。

（1）中断是随机的。

（2）中断是可恢复的。

（3）中断是自动处理的。

中断发生时，CPU 暂停正在执行的程序并保存其运行现场信息，然后自动转去执行相应的中断处理程序，中断处理完成后返回被中断程序的断点处（同时恢复为被中断程序所保存的现场信息）继续执行或者重新调度其他程序执行。在计算机系统中引入中断的目的主要有两个：一是解决 CPU 与 I/O 设备的并行工作问题；二是实现实时控制。中断在计算机系统中的应用越来越广泛，它可以用来处理系统中的任何突发事件，如请求系统服务、程序出错、硬件故障和网络通信等。用户程序执行系统调用，或者出现 I/O 通道和设备（见第 5 章）产生的内部和外部事件时，都需要通过中断机制产生中断信号来启动系统内核工作。

1．中断分类

根据中断源和中断事件的性质，可以将中断划分为不同的类型，如图 1-16 所示。

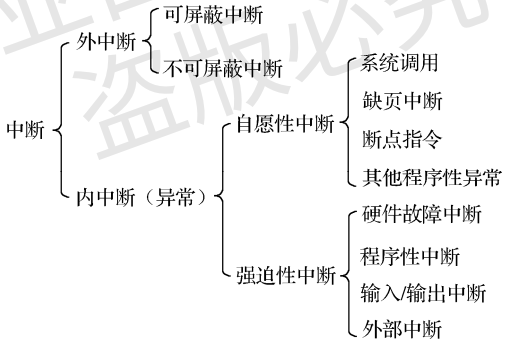

图 1-16 中断分类

（1）外中断。通常指硬中断，指来自 CPU 之外的外部设备通过硬件请求方式产生的中断，如被 I/O 设备触发的中断。外中断是一种强迫性中断，主要包括外部中断、I/O 中断等。由于外中断是由随机发生的异步事件引起的，它与 CPU 当前正在执行的任务无关，因此通常也称为异步中断。

外中断可以发生在用户态，也可以发生在内核态。不同的外中断通常具有不同的优先级，当 CPU 处理高一级的中断时，会部分或全部屏蔽较低优先级的中断，外中断一般在两条机器指令之间响应。外中断又可分为不可屏蔽中断和可屏蔽中断。不可屏蔽中断在当前指令执行结束后，就会无条件立即予以响应；可屏蔽中断则根据中断允许标志位是否允许来决定 CPU 是否响应中断。可屏蔽中断通常用于 CPU 与外设之间的数据交换。

（2）内中断。通常也称为异常（Exception），是由 CPU 在执行指令过程中检测到一个

或多个错误或异常条件引起的，用于表示 CPU 执行指令时本身出现算术溢出、0 作除数、取数时的奇偶校验错、访存（访问内存）指令越界或执行了一条所谓的“异常指令”（用于实现系统调用）等情况。这时中断当前执行的程序，转到相应的错误处理程序或异常处理程序。最早的中断和异常并没有区分，随着中断发生原因和处理方式差别的越发明显，才有了中断和异常的区分。内中断由 CPU 中控制器里的硬件中断装置产生，中断处理程序所提供的服务是当前运行程序所需要的，如内存访问错误、单步调试和被 0 除等。内中断处理程序在当前运行进程（程序）的上下文中执行。

内中断不可屏蔽，一旦发生必须立即响应处理。内中断通常发生在用户态，允许在指令执行期间进行响应并且允许多次响应。

根据中断事件的性质，可以将内中断划分为两大类：强迫性中断和自愿性中断。

① 强迫性中断。与当前运行程序的执行完全异步，中断是由随机事件和外部请求所引发的，引起强迫性中断的事件不是当前运行程序所期待的。强迫性中断事件主要包括硬件故障中断、程序性中断、外部中断和 I/O 中断等。

② 自愿性中断。用户程序在运行过程中请求操作系统为其提供某种功能服务，通过执行一条访管指令而引起的中断，称为“访管中断”或“陷阱”（Trap）。常见的自愿性中断有创建进程、分配内存、打开文件、信号量操作、发送或接收消息等。自愿性中断事件是当前运行程序所期待的，是用户在程序中有意安排的中断，它表示用户程序对操作系统的某种需求。当用户需要操作系统提供某种服务时，就使用访管指令或系统调用（见 1.6.3 节）产生自愿性中断来达到需求的目的。

访管中断是通过执行访管指令而产生的中断。用户程序执行时，如果执行的是访管指令就产生一个访管中断，并通过陷入指令将 CPU 的状态从用户态转换为内核态，然后将处理权移交给操作系统中的一段特殊代码（系统调用子程序），这一过程称为陷入。访管指令在用户态下执行，它包括两个部分：操作码和访管参数。操作码指出该指令是访管指令；访管参数则描述具体的访管要求。不同的访管参数对应不同的中断服务要求，就像机器指令的操作码一样，操作系统通过分析访管指令中的访管参数，调用相应的系统调用子程序为用户服务，系统调用功能完成后再返回用户态继续运行用户程序。

不同类型的中断具有不同的优先级。一般情况下，上述各种中断优先级从高到低的顺序依次为：硬件故障中断、自愿性中断、程序性中断、外部中断和 I/O 中断。

大部分内中断是由软件方法产生的，通过软件方法来模拟硬件中断，以实现宏观上异步执行效果的中断称为软中断，如“信号量”是一种软中断机制，是操作系统内核（或其他进程）对某个进程的中断。Windows 系统中由内核发出用于启动线程调度、延迟过程调用的 Dispatch/DPC 和 APC 等中断也是软中断的典型应用。

2．中断向量

每个中断有一个唯一的与其对应的中断向量号（通常为中断类型号），并按照中断向量号从小到大的顺序放在中断向量表中。因此，根据中断向量号可以得到该中断向量在中断向量表中的位置。

中断向量表一般存放在内存中的固定区域。例如，较为简单的 PC-DOS 系统的中断向量表，占用了系统内存最低端的 1KB 空间，共存储了 256 个中断向量，如图 1-17 所示。

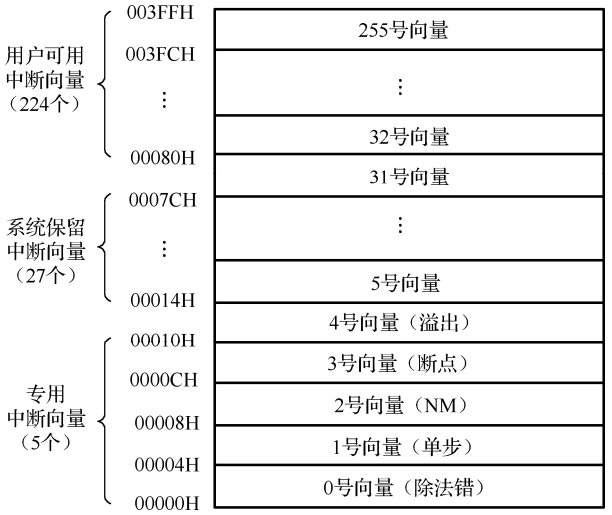

图 1-17　早期的 PC-DOS 系统中断向量表

系统调用的中断向量在访管指令中给出，其他中断或异常的中断向量一般是由计算机硬件或操作系统预先分配和设置的。因此系统可以根据中断向量的不同，来为不同的中断请求提供不同的中断服务。

3．中断响应与处理

中断系统是现代计算机系统的核心组成部分，通常由两部分组成：硬件中断装置和软件中断处理程序。硬件中断装置是中断系统的机制部分，负责捕获中断源发出的各种中断请求，并以一定方式响应中断源，然后将 CPU 控制权交给特定的中断处理程序；软件中断处理程序是中断系统的策略部分，负责辨别中断类型并执行相应的中断处理操作。

（1）中断响应

在硬件中断装置中，使用中断寄存器保存来自各个中断源的中断请求。中断寄存器的每一位称为一个中断位，当用户程序需要系统提供服务时就向 CPU 发出中断请求，设置对应的中断位。

CPU 接收到来自于不同中断源的中断请求后，需要及时地响应中断。某一时刻可能有多个中断源向 CPU 提出中断请求，但在任何时刻 CPU 只能响应一个中断。因此，中断系统需要按照各个中断源的优先级选择具有高优先级的中断进行响应。

CPU 如何发现中断信号呢？为了发现系统中的中断信号，在 CPU 的控制部件中设置了一个能检测中断的硬件机构——中断扫描机构，该机构在每条指令执行周期内的最后一个机器周期扫描中断寄存器，查询是否有中断信号到来。若无中断信号，则 CPU 继续执行下一条指令；若有中断信号，则硬件中断装置将该中断位按规定编码（中断码）送入程序状态字寄存器中相应的位，并通过交换中断向量引出中断处理程序，如图 1-18 所示。

（2）中断处理

无论是外中断还是内中断，CPU 响应和处理中断的过程基本相同，即根据中断源的“中断向量”，找到中断处理程序在内存中的入口地址，然后执行相应的中断处理程序。现代计算机系统的中断处理过程如下。

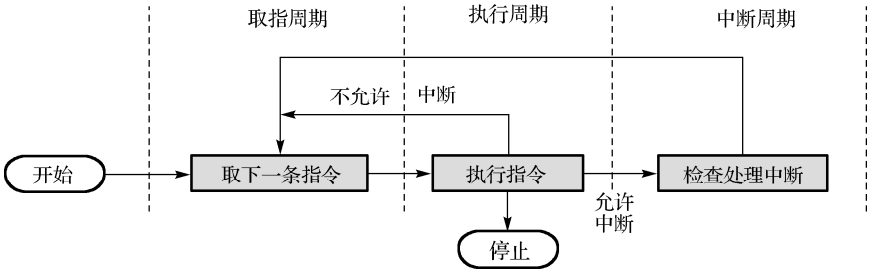

图 1-18 中断发现过程

① 保护被中断进程的现场信息。为了在中断处理结束后能使进程正确地返回到中断点，系统必须保存当前 CPU 的程序状态字 PSW 和程序计数器 PC 等内容。

② 分析中断原因，转去执行相应的中断处理程序。在多个中断请求同时发生时，处理优先级最高的中断源发出的中断请求。

③ 恢复被中断进程的现场信息，CPU 继续执行原来被中断的进程。

中断处理过程包括硬件操作和中断处理程序操作两部分。CPU 在响应中断请求之后到执行中断处理程序之前需要经历一个过渡期，这就是中断周期。在中断周期中，由硬件进行关中断、保存断点、寻找中断处理程序入口地址等操作。一个简单的 I/O 中断处理流程如图 1-19 所示。其中，CS 为代码段寄存器，PC 为程序计数器。

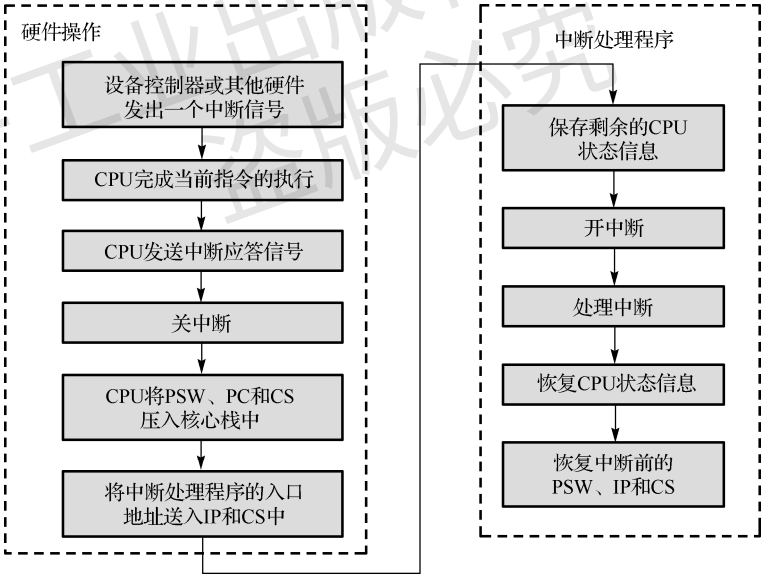

图 1-19 简单的 I/O 中断处理流程

中断处理是中断执行的主要部分。不同中断源的中断服务要求是不同的，因此所对应的中断处理程序也不同，操作系统内核通过调用中断处理程序完成用户请求的服务。中断处理结束后，为了能重新返回到被中断程序的断点处继续执行，需要在中断处理程序的最后安排一条中断返回指令，并通过该指令将保存在核心栈中的 PC、CS 和 PSW 弹出，从而恢复被中断程序断点处的地址和 PSW 等内容（恢复断点处现场信息），使得 CPU 能够转到被中断程序的断点处继续正常运行。

对于不同类型的中断，中断处理结束后执行的去向可能会不一样。对于内中断，异常处理程序结束的返回点将随异常类型的不同而不同；对于用户程序执行指令出错而产生的程序性中断（如除 0 操作），异常处理一般不会回到原用户程序；对于访管中断，异常处理结束后返回到原用户程序执行当前访管指令的下一条指令处继续执行；而对缺页故障（见第 4 章），异常处理结束后会返回原用户程序执行发生异常的那条指令去重新执行。

4．多重中断

当 CPU 正在执行一个中断处理时，如果有另一个优先级更高的中断请求到达，则 CPU 为响应这个更紧迫的新中断请求，而使正在运行的中断处理程序中断运行，这个过程称为中断嵌套。允许在中断处理过程中响应新的中断请求称为多重中断。

在支持多重中断的计算机系统中，若某一时刻系统中有多个中断源同时发出中断请求，那么中断系统应如何响应这些中断请求呢？为了使系统能及时响应并处理所有中断源的中断请求，中断硬件系统会根据中断事件的紧迫程度将中断源分为若干个级别，称为中断优先级。中断源的中断优先级决定了其被响应的优先顺序，中断事件的紧迫程度通常根据中断源得不到响应可能产生错误的严重程度来确定。在不丢失中断请求的前提下，系统把紧迫程度相当的中断源归为同一个中断优先级。中断优先级的确定通常由计算机的硬件排队电路实现。例如，在 PC 机中最多允许 256 个中断或异常，共分为 5 类，它们的响应顺序从高到低依次是：复位、异常、软件中断、不可屏蔽中断和可屏蔽中断。

多重中断的处理原则是当多个不同优先级的中断同时发生时，CPU 按照中断优先级由高到低的顺序响应各个中断源，并且规定：高优先级中断可以中断低优先级中断的中断处理程序的运行，正在运行的中断处理程序不能被新的同级或低优先级的中断请求所中断。

1.6.3 系统调用

操作系统作为介于计算机硬件和应用软件之间的一个层次，隐藏了底层硬件的物理特性差异和复杂的处理细节，并向上层提供方便、有效和安全的接口，那么用户以什么方式来使用这些接口呢？

对于现代计算机系统，用户可以通过键盘、鼠标等来使用操作系统提供的命令接口和图形接口，这是一种既简单又普遍的计算机使用方式。

但是操作系统提供的命令接口和图形接口功能有限，而用户在学习、生活、工作中要求计算机处理的问题是各种各样的。因此还需要软件设计开发人员编写应用软件，来满足用户的各种应用需求，所以操作系统必须提供用户编程使用的接口。此外，从事编译系统、数据库系统等系统软件的开发人员也要使用这些接口。目前，程序接口是通过系统调用（System Call）来实现的。

程序的运行空间分为用户空间和内核空间，在逻辑上它们之间是相互隔离的。因此在用户空间（用户态）运行的用户程序不能直接访问内核空间（内核态）的内核数据，也无法直接调用执行在内核空间的内核函数；否则，系统的安全性和可靠性就得不到保证。系统调用是操作系统提供给用户程序的特殊接口，用户程序通过系统调用通知系统自己需要执行内核空间的内核函数。此时由系统实现用户态到内核态的切换，并代替用户程序在内核空间执行内核函数及访问内核数据，待执行完内核函数后再由系统完成内核态到用户态

的切换，从而继续执行用户程序。为保护系统内核数据不被破坏，同时又为用户程序提供一个良好的运行环境，操作系统内核提供了一组具有特定功能的内核函数，并通过一组称为系统调用的接口呈现给用户使用。

1．系统调用概念

系统调用包含以下两个方面的含义。

（1）在操作系统内核中设置了一组用于实现各种系统功能的子程序（内核函数）。

（2）用户程序在用户态执行中需要系统内核提供服务时能够使用这组子程序。

操作系统把系统调用子程序（内核函数）组织在内核中且运行在内核态下，用户程序不能直接访问。操作系统设计了一条特殊的指令称为访管指令（如 DOS 中的 int 21H，UNIX 中的 trap 等是供汇编语言程序使用的访管指令），用户程序通过这条访管指令来调用内核的系统调用子程序。

系统调用通常也称为“陷阱”，它的执行过程与普通函数的调用很相似，但用户态运行的程序只有通过系统调用才能进入操作系统内核，而普通函数调用则由函数库或用户自己提供且只能运行于用户态。

访管指令的主要功能已在前面介绍过，那么系统调用的主要功能表现在以下三个方面。

（1）产生一个访管中断，把 CPU 工作状态由原来的用户态切换到内核态。

（2）执行对应的系统调用子程序（内核函数）。

（3）系统调用子程序运行完成后将 CPU 工作状态切换回用户态。

访管指令通常采用主程序-子程序的关系模式，按调用-返回方式实现（有的系统采用基于消息传递的通信方式实现）。CPU 在执行用户程序的访管指令时，用户程序暂停执行，CPU 转去执行该系统调用对应的系统调用子程序，该系统调用子程序执行完成后将结果返回给用户程序并恢复用户程序的运行，即继续执行用户程序中位于该访管指令之后的后续指令。

2．系统调用的实现过程

系统调用的实现机制主要依靠计算机硬件，一般用汇编代码描述，并通过库函数使其他程序能够使用，如 C 语言程序。不同机器的系统调用命令格式和功能定义可能有所不同，但处理流程基本相同。操作系统把系统调用中的各子程序进行统一编写，称为功能号。对于已经公开的系统调用，它们的功能号（如果是高级语言接口，则对应的是函数名）、所实现的功能、入口参数、返回结果等详细说明写成一份专门的文档，连同操作系统软件一起提交给用户，用户在编写程序时可以参考这份说明文档。

如图 1-20 所示，各系统调用子程序的地址与功能号登记在入口地址表中，CPU 执行用户程序中的系统调用后进入访管指令的处理程序。系统调用的实现过程如下。

（1）系统产生软中断，CPU 由用户态切换为内核态，保存用户程序的现场信息。

（2）分析功能号并在入口地址表中查找对应的系统调用子程序（内核函数），有时还需要进行安全控制检查。

（3）执行系统调用子程序并得到结果。

（4）恢复用户程序的现场信息，CPU 切换回用户态并返回结果，必要时进行安全检查。

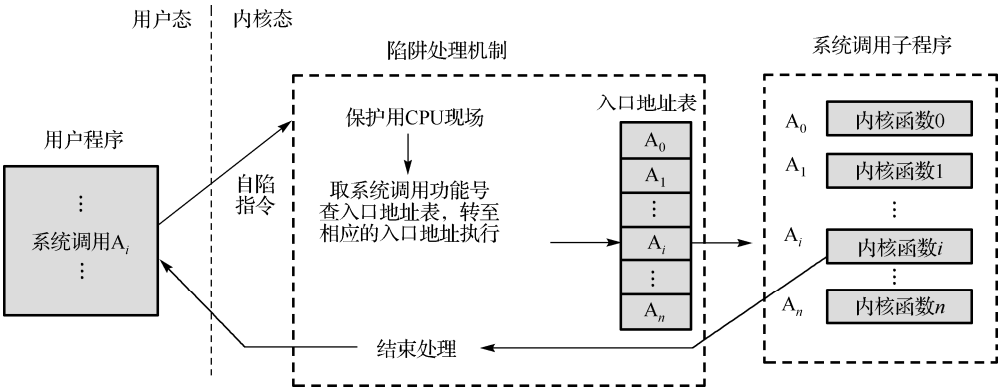

图 1-20　系统调用实现过程示意

在多任务操作系统中，系统调用子程序供多个用户共享使用，并且可能存在多个用户程序，同时调用同一个系统调用的情况。操作系统还需要设计专门的管理和控制机制，以保证彼此调用都能各自独立地正确运行。

此外，对单用户、单任务操作系统（如 DOS）来说，只有在一个系统调用执行完成后，才能开始另一个系统调用，即“内核不可重入”。

3．系统调用与一般用户子程序的区别

系统调用作为子程序与用户编写的程序中的子程序有哪些区别？从操作系统角度来看，它们所处的 CPU 工作状态不同，系统调用运行在内核态下，而用户子程序则运行在用户态下；系统调用的执行产生中断即访管中断（陷阱），而用户子程序的运行则不会产生中断；系统调用子程序的代码与调用者的程序代码是分开的、各自独立的，而用户子程序代码与调用者的程序代码在同一进程地址空间；不同用户程序可以共享使用同一个系统调用，而用户子程序通常不能由其他用户程序调用。

此外，操作系统还提供了一组实用程序，如编辑器、计算器等，它与操作系统没有很大关系，但却是人们管理、使用计算机不可缺少的功能，所以操作系统设计人员实现了这些功能，连同操作系统软件一起提供给用户。系统调用与操作系统实用程序的区别是系统调用不能单独运行（由用户程序调用），并且运行在内核态下，而实用程序可以单独运行并且运行在用户态下。

习　题　1

一、单项选择题

1．从用户的观点看，操作系统是_____。

A．用户与计算机之间的接口

B．控制和管理计算机资源的软件

C．合理地组织计算机工作流程的软件

D．由若干层次的程序按一定的结构组成的有机体

2．操作系统在计算机系统中位于_____之间。

A．CPU 和用户　　B．CPU 和内存

C．计算机硬件和用户　　D．计算机硬件和软件

3．下列选项中，_____不是操作系统关心的主要问题。

A．管理计算机裸机

B．设计、提供用户程序与计算机硬件系统的界面

C．管理计算机系统资源

D．高级程序设计语言的编译程序

4．操作系统的逻辑结构不包含_____。

A．混合型结构　　B．单内核结构

C．分层式结构　　D．微内核结构

5．相对于单内核结构，采用微内核结构的操作系统具有许多优点，但_____并不是微内核的优势。

A．使系统更高效　　B．想添加新服务时不必修改内核

C．使系统更安全　　D．使系统更可靠

6．操作系统的三种基本类型是_____。

A．批处理系统、分时操作系统及网络操作系统

B．分时系统、实时操作系统及分布式操作系统

C．批处理系统、分时操作系统及实时操作系统

D．网络操作系统、批处理系统及分时操作系统

7．现代操作系统的基本特征是_____、资源的共享和操作的异步性。

A．多道程序设计　　B．中断处理

C．程序的并发执行　　D．实现分时与实时处理

8．若把操作系统看作计算机资源的管理者，下列的_____不属于操作系统所管理的资源。

A．程序　　B．内存　　C．CPU　　D．中断

9．批处理操作系统首先要考虑的问题是_____。

A．灵活性和可适应性　　B．交互性和响应时间

C．周转时间和系统吞吐量　　D．实时性和可靠性

10．_____不是分时操作系统的基本特征。

A．同时性　　B．独立性　　C．实时性　　D．交互性

11．在设计实时操作系统时，_____不是重点考虑的问题。

A．及时响应、快速处理　　B．高安全性

C．高可靠性　　D．提高系统资源的利用率

12．下面叙述中正确的是_____。

A．批处理作业必须有作业控制信息

B．分时系统不一定都具有人机交互功能

C．从响应时间的角度看，实时系统与分时系统差不多

D．由于采用了分时技术，用户可以独占计算机资源

13．_____操作系统允许在一台主机上同时连接多台终端，多个用户可以通过各自的终端同时交互地使用计算机。

A．网络　　B．分布式　　C．分时　　D．实时

14．操作系统的不确定性是指_____。

A．程序运行结果的不确定性　　B．程序运行次序的不确定性

C．程序多次运行时间的不确定性　　D．A～C 都是

15．多道程序设计技术是指_____。

A．在实时系统中并发运行多个程序

B．在分布式系统中同一时刻运行多个程序

C．在一台 CPU 上同一时刻运行多个程序

D．在一台 CPU 上并发运行多个程序

16．当 CPU 执行操作系统内核代码时，称处理机处于_____。

A．自由态　　B．用户态　　C．内核态　　D．就绪态

17．操作系统有效的安全机制不包括_____。

A．身份鉴别　　B．硬件保护　　C．入侵检测　　D．计算机病毒防治

18．CPU 执行的指令被分为两类，其中一类称为特权指令，只允许_____使用。

A．操作员　　B．联机用户　　C．操作系统　　D．用户程序

19．中断的概念是指_____。

A．暂停 CPU 执行　　B．暂停 CPU 对当前运行程序的执行

C．停止整个系统的运行　　D．使 CPU 空转

20．用户程序在用户态下使用系统调用引起的中断属于_____。

A．硬件故障中断　　B．程序中断

C．访管中断　　D．外部中断

21．系统调用是_____。

A．用户编写的一个子程序　　B．高级语言中的库程序

C．操作系统中的一条命令　　D．操作系统向用户程序提供的接口

22．当操作系统完成用户请求的系统调用功能后，应使 CPU_____工作。

A．维持在用户态　　B．从用户态转到内核态

C．维持在内核态　　D．从内核态转到用户态

23．中断系统一般是由相应的_____组成的。

A．硬件　　B．软件　　C．硬件和软件　　D．A～C 都不是

24．计算机系统中判断是否有中断事件发生应在_____。

A．进程切换时　　B．执行完一条指令后

C．执行 P 操作后　　D．由用户态转入内核态时

25．在中断发生后，进入中断处理的程序属于_____。

A．用户程序　　B．可能是应用程序也可能是操作系统程序

C．操作系统程序　　D．既不是应用程序也不是操作系统程序

26．中断处理和子程序调用都要压栈以保护现场，中断处理一定会保存而子程序调用不需要保存其内容的是_____。

A．程序计数器　　B．程序状态字寄存器

C．数据寄存器　　D．地址寄存器

二、判断题

1．采用多道程序设计的系统中，系统中的程序道数越多则系统的效率越高。

2．应用软件是加在裸机上的第一层软件。

3．操作系统特征之一的“不确定性”是指程序运行结果是不确定的。

4．多道程序设计可以缩短系统中程序的执行时间。

5．操作系统的所有程序都必须常驻内存。

6．分层式结构的操作系统必须建立模块之间的通信机制，所以系统效率高。

7．微内核结构操作系统具有较高的灵活性和扩展性。

8．操作系统内核不能使用特权指令。

9．通常将 CPU 模式分为内核态（核心态）和用户态，这样做的目的是为了提高运行速度。

10．从响应的角度看，分时系统与实时系统的要求相似。

11．使计算机系统能够被方便地使用和高效地工作是操作系统的两个主要设计目标。

12．操作系统的存储管理就是指对磁盘存储器的管理。

13．分时操作系统允许两个以上的用户共享一个计算机系统。

14．实时操作系统只能用于控制系统而不能用于信息管理系统。

15．当 CPU 处于用户态时，它可以执行所有的指令。

16．访管指令为非特权指令，在用户态下执行时会将 CPU 转换为内核态。

17．系统调用与程序级的子程序调用是一致的。

18．用户程序有时也可以在内核态下运行。

19．执行系统调用时会产生中断。

20．系统调用返回时，由内核态变为用户态执行用户程序。

21．中断的处理是由硬件和软件协同完成的，各中断处理程序是操作系统的重要组成部分，所以对中断的处理是在内核态下进行的。

三、简答题

1．什么是操作系统，它有什么基本特征？

2．什么是多道程序设计技术？多道程序设计技术的特点是什么？

3．操作系统是随着多道程序设计技术的出现逐步发展起来的，要保证多道程序的正常运行，在技术上需要解决哪些基本问题？

4．如何理解操作系统的不确定性？

5．分时操作系统形成和发展的主要动力是什么？

6．批处理、分时和实时操作系统各有什么特点？

7．分时系统和实时系统有什么区别？设计适用实时环境的操作系统的主要困难是什么？

8．什么是分布式操作系统？它与网络操作系统有何不同？试说明分布式操作系统或网络操作系统在传统的操作系统管理模式上需要哪些改进。

9．简述操作系统内核及其功能。

10．简述分层式结构与单内核结构的异同。

11．简述微内核操作系统的主要特点。

12．处理机为什么要区分内核态和用户态两种操作方式？在什么情况下进行两种方式的转换？

13．在用户与操作系统之间存在哪几种类型的接口？它们的主要功能是什么？

14．简述中断处理过程。

15．叙述系统调用的概念和操作系统提供系统调用的原因。

16．简述系统调用的实现过程。