第5章 资源管理

CHAPTER 5

众所周知，计算机的资源由硬件资源和软件资源两个部分组成。计算机硬件是组成计算机物理设备的总称，包括中央处理器、存储器、输入输出设备等，是计算机系统工作的物质基础；而软件是计算机硬件设备上运行的各种程序及其相关资料的总称。要实现用户的软件通过各种指令使计算机硬件高效有序地执行，需要一个管理系统既可以实现对计算机所有硬件的管理，又能方便地为应用软件提供各种接口指令或协议。操作系统就是介于计算机硬件和应用软件之间，实现对计算机系统资源进行控制与管理的软件。本章内容就是针对操作系统的控制和管理功能来阐述操作系统在计算机中的重要地位。

5.1 操作系统的概念

5.1.1 操作系统的定义

为了明确操作系统的定义，先来简单回顾一下操作系统的发展。操作系统的发展是随着计算机体系结构的变化而来的。

1. 操作系统的发展

1946年计算机诞生到20世纪50年代中期的计算机属于第一代计算机，这时的计算机体积庞大、速度慢，没有操作系统。由用户(程序员)采用手工方式直接控制和使用计算机硬件，即程序员编好程序后，首先需要使用穿孔机，将程序和数据制作成纸带或卡片。然后将这些纸带或卡片装入纸带输入机或卡片输入机，启动计算机。计算机从读卡机上将程序和数据读入，并开始运行程序。程序运行结束并取走结果后，才让另一个用户使用计算机。这种人工操作方式有以下缺陷，用户上机时独占全机资源，造成资源利用率不高，系统效率低；手工操作多，浪费处理器时间等。

20世纪50年代后期，随着晶体管计算机的广泛应用和计算机高级语言(FORTRAN、ALGOG、COBOL等)的出现，此时用户可以采用高级语言编写程序来控制计算机的执行。用户需要完成某一个任务，首先将程序写到纸上，然后将纸穿成卡片，再将卡片带到计算中心交给操作员。计算机运行完当前的任务之后，其结果由打印机输出。接着操作员从卡片盒中选择另一个任务(作业)交给计算机执行。在这个阶段，用户提交的任务在操作员的干预下成批执行。由于处理机的速度与手工操作设备的输入和输出的速度不相匹配，人们设计了监督程序(或管理程序)来实现任务的自动转换处理。这期间，每个任务由程序员提供

一组在某种介质(如纸、磁盘)上的任务信息(文件),包括任务说明书及相关的程序和数据。任务说明书由程序员提交给系统操作员,操作员集中一批用户提交的作业,由管理程序将这批作业从纸带或卡片机输入到磁带上,当一批作业输入完成后,管理程序自动把磁带上的第1个作业装入内存,并把控制权交给作业。该作业执行完成后,作业又把控制权交回管理程序,管理程序再调入磁带上的第2个作业到内存中执行。以此类推,直到所有作业完成。这种处理方式称为批处理方式。由于是串行操作,所以又称为单道批处理。

单道批处理系统内存中仅有一道任务,无法充分利用系统中的所有资源,导致系统中仍有许多资源空闲,设备利用率低,系统性能差。20世纪60年代中期,计算机体系结构发生了很大变化,由以CPU为中心的结构改变为以主存为中心,使在内存中同时装入多个作业(或任务)成为可能,使多道程序的概念成为现实。多道程序设计指允许多个程序同时进入一个计算机系统的主存储器并启动进行计算的方法。即计算机内存中可以同时存放多道(两个或以上相互独立的)程序,它们都处于开始和结束点之间。从微观上看是串行的,各道程序轮流使用CPU,交替执行。这样的处理方式提高了CPU的利用率,充分发挥计算机系统部件的并行性,现代计算机系统都采用了多道程序设计技术。

随着计算机系统和网络的进一步发展,为了适应计算机结构的变化,满足用户不断变化的应用需求,提高计算机系统资源的利用率等,操作系统也步入了更实用化的阶段,陆续出现了分时操作系统、实时操作系统、微机操作系统、嵌入式操作系统和网络操作系统等。感兴趣的读者可以参考其他书目,了解这类操作系统的详细情况。

2. 操作系统的定义

操作系统是最基本的系统软件,因为所有其他的系统软件(例如编译程序、数据库管理系统等语言处理器)和软件开发工具都是建立在操作系统的基础之上,它们的运行全都需要操作系统的支持。计算机启动后,通常先把操作系统装入内存,然后才启动其他的程序。

所谓操作系统(Operating System,OS)是由一些程序模块组成,用来控制和管理计算机系统内的所有资源,并且合理地组织计算机的工作流程,以便有效地利用这些资源,并为用户提供一个功能强大、使用方便的工作环境。

操作系统的定义同时说明了操作系统有如下任务。

1) 资源管理

任何一个计算机系统,不论是大型机、小型机,还是微机,都具有两种资源,即硬件资源和软件资源。硬件资源指计算机系统的物理设备,包括中央处理机、存储器和I/O设备;软件资源指由计算机硬件执行的、用以完成一定任务的所有程序及数据的集合,包括系统软件和应用软件。操作系统是最基本的系统软件,操作系统又是资源的管理者,用户程序使用资源都必须经过操作系统——调用操作系统的功能函数。操作系统管理资源的目标是发挥资源的最大效率,因此需要实现高效率的资源管理机制,包括分配、调度和共享机制。

2) 资源使用方便

既然用户程序必须通过操作系统来使用计算机资源,操作系统就必须提供方便、好用、安全和可靠的资源访问功能函数(或接口函数)。例如,所有I/O设备的驱动函数。操作系统为用户(计算机系统管理员、应用软件的设计人员等)提供了抽象的资源使用界面。抽象层次越高,用户使用资源就越方便,因为它让用户远离了资源访问的细枝末节。

3）资源的安全保障

操作系统在提供功能函数时，必须考虑到资源的安全使用问题，即用户是否有权限使用该资源。用户超越权限地使用资源，不仅会危害系统的正常运行，也会损害其他用户的利益。这也正是计算机病毒和攻击者想做的。因此操作系统必须建立有效的安全机制。

操作系统是计算机中的最基本核心软件，它位于计算机硬件与应用程序之间，提供给应用程序各种功能函数，同时控制硬件完成各种操作。

5.1.2 操作系统的分类

随着计算机技术和软件技术的发展，目前已经形成了各种类型的操作系统，以满足不同的应用要求。最常用分类方法（按照操作系统的用户服务方式分）主要有多道批处理操作系统、分时操作系统、实时操作系统、微机操作系统、网络操作系统、分布式操作系统等。

1. 多道批处理操作系统

为了解决 CPU 和 I/O 运行速度的差别，即作业占用 I/O 时，CPU 处于空闲状态的问题，产生了多道批处理系统。它一次将几个作业放入内存，宏观上看，同时有多个作业在系统中运行，而实际上这些作业是分时串行地在一台计算机上运行。但此时的操作系统具有了两大基本特征。

1）并发

指系统允许同时执行多个运行中的程序。如图 5-1 所示，3 道作业交替执行的情况，当一个作业进入 I/O 过程，CPU 并没有等待该作业 I/O 的结束，而是直接执行等待队列中的另外一个作业，提高了系统单位时间内的作业运行量。

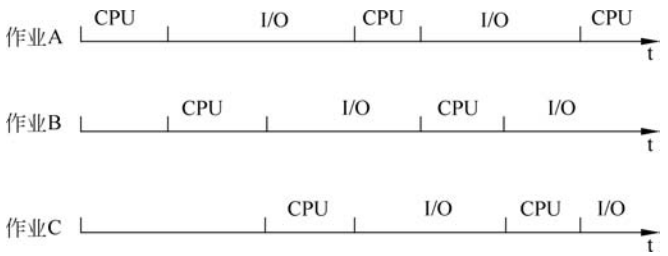


图 5-1 CPU 交替地执行 3 道作业

2）共享

指系统为多个运行中的程序提供共享资源。利用磁盘的速度比读卡机和打印机快的特点，将磁盘分为“输入井”和“输出井”，将多个作业的相关数据由 I/O 通道控制读卡机读入并存放到“输入井”。操作系统从“输入井”中读入作业的相关数据，启动作业运行。作业在运行过程中，也从“输入井”中输入数据，进行处理。同时，作业的输出数据都写入“输出井”中。当作业运行结束后，由 I/O 通道控制打印机，将作业在“输出井”中的全部输出数据打印出来，好像每个用户都拥有了一台打印机，实现了资源共享。

2. 分时操作系统

多道批处理操作系统虽然能提高机器的资源利用率，却存在一个重要的缺点，即程序的调试问题。由于一次要处理一批作业，作业的处理过程中，任何用户都不能和计算机进行交互。即使发现了某个作业有程序错误，也要等一批作业全部结束后脱机进行纠错。查明或

改正一处错误，往往需要运行多次作业，延长了程序的调试周期。正是这一矛盾，导致了分时操作系统应运而生。

在精确的时钟中断系统和速度更快的处理机等硬件的支持下，分时操作系统允许多个用户同时联机与系统进行交互通信，一台分时计算机系统连有若干台终端，多个用户可以在各自的终端上向系统发出服务请求，系统根据用户的请求完成指定的任务，并把执行结果返回。这样用户可以根据运行结果，再次通过终端决定下一步的请求指令。重复这个交互过程，直到每个用户实现自己的预定目标。由于处理机的速度很快，将时间片划分适当，从一个较长时间看，每一个用户似乎都独享主机。这样，对于每个用户都仿佛“独占”了整个计算机系统。具有这种特点的计算机系统称为分时系统。

分时系统具有多路性、交互性、独占性和及时性的特点。

3. 实时操作系统

实时操作系统指系统能够及时响应随机发生的外部事件，并在严格的时间范围内完成对该事件的处理，并能控制所有实时设备和实时任务协调一致地工作。所谓“外部事件”是指与计算机连接的设备向计算机发出的各种服务请求。实时操作系统根据控制对象不同又分为实时控制系统（如导弹导航）和实时信息处理系统（如机票查询）。

实时系统的特点如下：

（1）响应及时，一般在毫秒或微秒级，用于军事的实时系统对响应时间要求更高。

（2）可靠性高，系统效率放在次要地位。

（3）系统安全性放在第一位，交互性差或根本没有交互性。

（4）系统整体性强，很多实时系统同时又是分布式系统，具有分布式系统整体性强的优点。

4. 微机操作系统

从20世纪70年代以来，个人计算机得到普及。为了满足个人使用计算机的要求，微机操作系统诞生了。早期的微机操作系统是单用户单任务的命令行形式的操作系统，如MS-DOS。随着计算机技术的发展和用户需求的不断提高，微机操作系统具有图形用户界面、多用户、多任务、虚拟存储管理、网络通信支持等功能，最大限度地满足用户对计算机的使用需求。微机操作系统的出现改变了人们对计算机的传统观念，计算机开始走进人们生活和工作的各个领域。

5. 网络操作系统

网络操作系统是为计算机网络而配置的。计算机网络是把不同地点上分布的计算机按照网络体系结构协议标准设计的结构连接起来，实现资源共享。在网络范围内，网络操作系统用于管理网络通信和共享资源，协调各计算机上任务的运行，并向用户提供统一、有效方便的网络接口的软件。网络操作系统虽具有分布处理功能，但其控制功能却是集中在某个或某些主机或网络服务器中，即集中式控制方式。需要说明的是，在网络中各独立计算机仍有自己的操作系统，由它管理着自身的资源。只有在它们进行相互间的信息传递、使用网络中的可共享资源时，才会涉及网络操作系统。

6. 分布式操作系统

由多台计算机组成的特殊的计算机网络形成分布式计算机系统，分布式操作系统是为分布式计算机系统而配置的，它将物理上分布的具有自治功能的数据处理系统或计算机系

统互连起来,实现信息交换和资源共享,协作完成任务。分布式操作系统具有任务分配功能,可将多个任务分配到多个处理单元上,使这些任务并行执行,从而加速了任务的执行。其特征是系统中所有主机使用同一操作系统,可以实现资源的深度共享,系统具有透明性和自治性。

7. 嵌入式操作系统

计算机嵌入式操作系统不仅能用于建立像银行服务、图书馆管理、数字天气预报这样的大型应用系统,也能用于建立小型甚至微型的应用系统。例如,家电、手机、数码相机上的应用程序,甚至可以植入人体内部的医疗芯片。

嵌入式系统是在各种设备、装置或系统中,完成特定功能的软硬件系统。它们是一个大设备、装置或系统中的一部分,这个大设备、装置或系统可以不是"计算机"。通常工作在反应式或对处理时间有较严格要求的环境中,由于它们嵌入在各种设备、装置或系统中,因此称为嵌入式系统。在嵌入式系统中的操作系统,称为嵌入式操作系统。嵌入式操作系统具有实时操作系统的特征,功能及编程非常简便,支持特定的 I/O 设备及定制相关系统功能。

5.1.3 操作系统的特征

操作系统是一个十分复杂的系统软件。前面介绍的几种操作系统,虽然它们都有各自的特征,但也都具有以下几个共同特征。

1. 并发性

并发性是指在计算机系统中同时存在着若干个正在运行的程序,这些程序同时或交替地运行。从宏观上看,程序是同时向前推进的;从微观上看,程序是顺序执行的,在单 CPU 上是轮流执行。

2. 共享性

共享性指系统中的资源可供内存中多个并发执行的进程共同使用,即操作系统程序与多个用户程序共享系统中的各种软、硬件资源。

3. 虚拟性

虚拟性指通过某种技术把一个物理实体变成逻辑上的多个。例如,前面提到的分时操作系统,虽然只有一个 CPU,但每个终端用户却都认为各有一个 CPU 在专门为他服务,实现了虚拟处理。

4. 不确定性

不确定性指因为多个用户程序共享系统中各种资源,造成了系统中很多不确定性因素。操作系统控制下的多个作业的运行顺序和每个作业的运行时间是不确定的。

5.1.4 操作系统的功能

操作系统的基本功能就是管理各种计算机资源并给用户提供一种简便、有效地使用资源的手段,充分发挥各种资源的利用率。操作系统应具有的基本功能有进程管理、存储管理、设备管理、作业管理和文件管理。

1. 进程管理

进程代表一个运行中的程序,进程管理主要解决处理机的分配调度问题,有时也称为处理机管理。CPU 是计算机系统中最宝贵的硬件资源。为了提高 CPU 的利用率,操作系统

采用了多道程序技术,即系统中有多个进程同时运行。操作系统负责监督进程的执行过程,协调进程之间的资源分配及竞争等问题,以使 CPU 资源得到最充分的利用。

2. 存储管理

存储管理主要指内存资源的分配和使用管理,虽然 RAM 芯片的集成度不断提高,但受 CPU 寻址能力的限制,内存的容量仍有限。操作系统负责为每个进程分配内存空间,保证用户存放在内存中的程序和数据彼此隔离、互不侵扰,同时提供进程之间共享程序和数据的功能。当内存容量不足时,操作系统使用磁盘空间作为内存空间的扩展,实现虚拟存储系统。

3. 设备管理

设备管理主要是对计算机系统 I/O 设备(外部设备)进行分配、回收与控制。设备管理负责外部设备的分配、启动和故障处理,用户不必详细了解设备及接口的技术细节,就可以方便地对设备进行操作。

4. 文件管理

计算机系统中的软件资源(如程序和数据)是以文件的形式存放在外存储器(如磁盘、磁带)中,需要时再把它们装入内存。文件管理的任务是有效地支持文件的存储、检索和修改等操作,解决文件的共享、保密和保护问题,以使用户方便、安全地访问文件。

5. 作业管理

所谓作业是用户要求计算机处理的一个相对独立的任务。作业管理是操作系统提供给用户的最直接的服务。按照用户观点,操作系统是用户与计算机系统之间的接口。因此,作业管理的任务是为用户提供一个使用系统的良好环境,使用户能有效地组织自己的工作流程,并使整个系统能高效运行。

操作系统的各功能之间并非是完全独立的,它们之间存在着相互依赖的关系。

5.2 多道程序设计

多道程序设计是操作系统所采用的最基本、最重要的技术,其根本目的是提高整个系统的效率。衡量系统效率的尺度是系统吞吐量。所谓吞吐量是指单位时间内系统所处理作业(程序)的道数(数量)。如果系统的资源利用率高,则单位时间内所完成的有效工作就多,吞吐量就大。

5.2.1 并发程序设计

当一个作业进入内存后,它的计算工作就开始了。通常,计算机运行用户作业的方式有两种,顺序运行和并发运行。

1. 顺序运行方式

顺序运行方式是传统的程序设计方法,即作业的运行总是一个一个顺序来,完成一个作业后再运行下一个作业。一种最容易实现的方式,常见于早期的单道批处理系统中。其具有如下特点。

1) 独占性

任意一个程序运行中独占系统资源,即使某个程序只使用很少的资源,多余的资源也不

会分配给其他程序。

2）顺序性

程序所规定的动作在机器上严格地按顺序执行，每个动作的执行都以前一个动作的结束为前提条件。

3）封闭性

只有程序本身的动作才能改变程序的运行环境，不会受到任何其他程序和外界因素的干扰。

4）可再现性

程序的执行结果与其执行速度无关。只要输入的初始条件相同，就会得到相同的结果。这样当程序中出现了错误时，往往可重现错误，以便进行分析。

2. 并发运行

并发运行是多道程序系统中的一种运行方式。它运行多个程序共享 CPU，以并发方式进行运算。换句话说，在任一时刻，系统中不再只有一个活动，而存在着许多并行的活动。从程序活动方面看，则可能有若干程序同时或者相互穿插地在系统中执行，并且次序不是事先确定的。

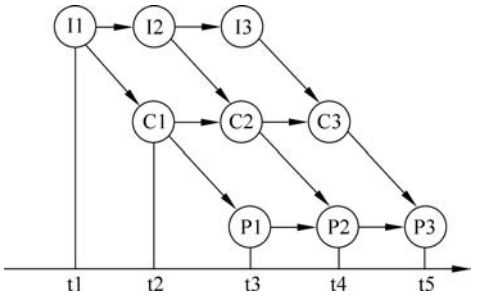


图 5-2　多道程序并发执行示意图

图 5-2 给出了一个假想的并发运行示例，1，2，3 分别为并发运行的 3 个程序。其中，每一个程序的运算可分为 3 个相对独立的处理。现假设这些处理都必须按先后顺序执行，那么这 3 个程序就可并发运行。图 5-2 中，除了 t1 和 t5 外，其余时间都有多个程序可供选择运行。并发运行方式具有以下基本特征。

1）异步性

每个程序的运行是不确定的，即启动时间不确定，结束时间也不确定。程序的整个运行过程呈现不连续状态，走走停停，停停走走。

2）共享性

当一个程序申请系统资源并得到满足时，该程序就可以获得运行机会。当它用完这些资源主动归还时，或者被迫暂时放弃某些资源时，释放的资源被其他程序获得。这样，就形成了资源供多道程序共享的局面。

3）相互制约性

这是多道程序系统的常见特征。从资源方面，当一个资源被某个程序占用，其他程序就不能使用该资源。从程序协作方面，一个程序需要利用另一个程序的结果，如果对方尚未加工完成，该程序就需要等待。从 CPU 的竞争方面，一部分程序获准在 CPU 上运行，另一部分程序就不能运行。这是一种既相互依赖又相互制约的关系。

4）不可重现性

该特征指程序的运行过程和运行结果不可以重现。即使程序的初始条件相同，也会因为运行时间不同，得到不同的运行结果。另外，从运行过程看，由于系统中处于驻留状态的作业很多，它们的运行都以未知的速度向前推进。因此，不可能通过重复性操作，将某一个

程序的运行轨迹再现出来。

5.2.2 进程

1. 进程的概念

进程作为操作系统的一个重要概念，是在20世纪60年代提出来的。在程序设计领域，“程序”一词的含义中具有浓厚的“代码”成分，比较适合描述静态文本，不能完整描述系统中多道程序的并发运行。为了描述程序运行的动态特性，“进程”概念被广泛接受。

所谓进程是具有一定独立功能的程序在某个数据集合上的一次运行活动，进程是系统进行资源分配和调度的一个独立单位。

从进程的定义来看，进程不是程序，而是程序的一次运行。当一个程序在计算机上运行时，便有一个进程存在；若程序不运行，就只有程序而没有进程了。另外，当一个程序运行多次时，必然会生成多个进程。

2. 进程的特性

进程具有以下5个特征。

1）动态性

进程有“生命周期”。它由“创建”而产生，由“撤销”而消亡。这是进程最重要的特征。

2）并发性

一个进程可以与其他进程并发执行。从系统的角度看，在一个时段内可以有多个进程同时存在并以不同的速度向前推进。而程序作为一种静态文本是不具备这种特征的。

3）独立性

进程是系统中的一种独立运行的基本单位，也是系统分配资源和调度的独立单位。

4）异步性

进程是按异步方式运行的，即它的推进速度是不可预知的。由于系统中允许多个进程并发执行，每一次调度的目标带有一定的随机性，且进程的运行规律是“走走——停停——走走”。因此，系统无法预知某一瞬间运行的是哪一个进程，以及它的推进速度怎样。

5）结构性

为了描述进程的动态变化过程，并使之能独立运行，系统为每一个进程设置一个进程控制块(PCB)。由程序代码、数据集及进程控制块组成进程实体。进程控制块是进程存在的标志。只要一个进程的控制块存在，无论该进程的程序代码和数据集是否在内存，都可以被系统控制和调度。

3. 进程的状态及其状态转换

根据进程在执行过程中的不同情况，通常可以将进程分成不同的状态：

1）就绪状态

指进程原则上是可以运行的，只是缺少CPU而不能运行，一旦把CPU分配给它，就可以立即投入运行。由于系统中可以活跃着多个进程，某一时刻同时处于就绪状态的进程可能不止一个，通常将它们排成一个队列，称为就绪队列。

2）运行状态

指进程已获得CPU，并且在CPU上执行的一种状态。在单处理机系统中，每个瞬间最多只能有一个进程处于执行状态；在多处理机系统中，则有小于或等于处理机数量的进程

处于运行状态。

3）等待状态

等待状态也称阻塞状态或睡眠状态。进程在前进的过程中，由于等待某种条件而不能运行时所处的状态。通常处于阻塞状态的进程也有多个，系统也将其排成一个队列，称为阻塞队列。

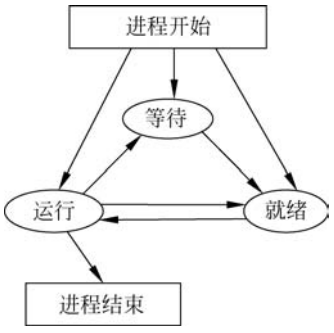


图 5-3　进程状态转换

进程的动态性质决定了进程的状态不可能固定不变，其状态随着自身的推进和外界条件的变化而变化。进程状态之间的转换如图 5-3 所示。

在运行状态的进程，一旦条件不满足时，将转为等待状态，或由于时间片用完，而转为就绪状态。处于等待状态的进程，一旦条件满足，将解除等待状态而转为就绪状态。处于就绪状态的进程，一旦分配到 CPU 就转为运行状态。最后当进程结束时，往往处于运行状态中。

4. 进程控制块

在操作系统中为进程定义了一个专门的数据结构，称为进程控制块(Process Control Block，PCB)。系统为每一个进程设置一个 PCB，PCB 是进程存在与否的唯一标志。当系统创建一个进程时，系统为其建立一个 PCB；然后利用 PCB 对进程进行控制和管理；当进程撤销时，系统收回它的 PCB，随之该进程也就消亡了。

进程控制块是一种数据结构，通常包括 3 类信息。

(1) 标识信息用于唯一的标识一个进程，可以分为用户使用的标识符和系统使用的内部标识符。常用的标识信息有进程标识符、父进程标识符、用户进程名、用户组名等。

(2) 现场信息用于保留一个进程在运行时存放在处理器现场中的各种信息。任何一个进程在让出处理器时，必须把此时的处理器现场信息保存到进程控制块中；而当该进程重新恢复运行时，也应恢复处理器现场。常用的现场信息有通用存储器的内容、控制寄存器的内容、用户堆栈指针和系统堆栈指针等。

(3) 控制信息用于管理和调度一个进程。常用的控制信息有进程的组成、调度、通信、以及资源使用和占用情况等。

进程控制块使用权限或修改权限均属于操作系统程序。操作系统根据 PCB 对并发执行的进程进行控制和管理，借助于进程控制块进程才能被调度执行。

5.2.3　进程之间的通信

进程是操作系统中可以独立运行的单位，但是由于处于同一个系统之中，进程之间不可避免地会产生某种联系。例如，竞争使用共享资源，而且有些进程本来就是为了完成同一个作业而运行的。因此，进程之间必须互相协调，彼此之间交换信息，这就是进程之间的通信。主要表现在进程同步、进程互斥、临界资源管理等方面。

1. 进程同步

进程同步指进程之间一种直接的协同工作关系，这些进程相互合作，共同完成一项任务。由于进程都是以独立的、不可预知的速度运行，对于相互协作的进程需要在某些协调点处协调它们的工作。当其中的某个进程先到达协调点，则暂停执行，等待其他协作进程，直

到其他协作进程达到协调点并给出协调信号后方可唤醒并继续执行。例如，有 A、B 两个进程，A 进程负责从键盘读数据到缓冲区，B 进程负责从缓冲区读数据进行计算。要完成取数据并计算的工作，A 进程和 B 进程要协同工作，即 B 进程只有等待 A 进程把数据送到缓冲区后才能进行计算，A 进程只有等待 B 进程发出已把缓冲区数据取走的信号之后才能从键盘向缓冲区中送数据，否则就会出现错误。这是进程同步的问题，如图 5-4 所示。

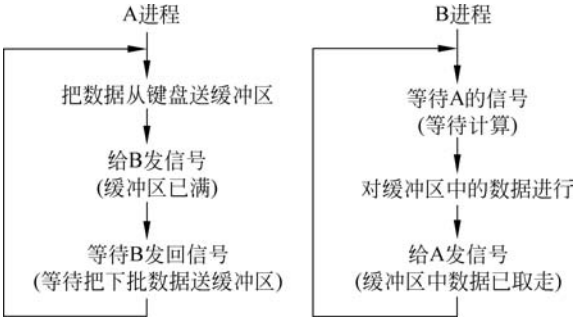


图 5-4　进程同步示意图

2. 进程互斥

系统中许多进程常常需要共享资源，而这些资源往往要求排他地使用，即一次只能为一个进程服务。因此，各进程间互斥使用这些资源，进程间的这种关系是进程的互斥。进程间的间接相互作用构成进程互斥。例如，多个进程在竞争使用打印机、一些变量、表格等资源时，表现为互斥关系。

系统资源共享是现代操作系统的基本功能，但是解决资源共享所面临的问题却因资源属性不同而有很大区别。有些系统资源是共享的，如磁盘；有些资源则是独享的，如打印机。显然独享性资源的共享处理与共享性资源的处理机制就不同。以打印机为例，设有 A，B 两个进程共享一台打印机，如果任它们自由使用，则打印出的结果会交织在一起，根本无法使用。考虑一种解决方案，让进程 A(或进程 B)一旦先占用打印机，就一直供其独享使用，直到其打印机使用完成为止，然后释放打印机，供其他进程使用。

系统中一些资源一次只允许一个进程使用，这类资源称为临界资源。在进程中访问临界资源的那一段程序称为临界区，要求进入临界区的进程之间构成互斥关系。为了保证系统中各并发进程顺利运行，对两个以上欲进入临界区的进程，必须实行互斥。为此，系统采取了一些调度协调措施。

1) 空闲让进

当该临界区没有进程进入时，则允许申请进入临界区的进程立即进入。

2) 忙则等待

当已有进程进入该临界区时，则其他申请欲进入临界区的进程只能等待。

3) 有限等待

对要求访问临界资源的进程，应保证能在有限时间内进入临界区，以免进程陷入“死等”状态。

4) 让权等待

当进程不能进入自己的临界区时，应立即释放处理机(让等待进程进入)，以免进程陷入

“忙等”状态。

3. 进程通信

并发进程在运行过程中,需要进行信息交换。交换的信息量可多可少,少的只是交换一些已定义的状态值或数值;多的则可交换大量信息,因此要引入高级通信原语,解决大量信息交换问题。高级通信原语不仅保证相互制约的进程之间的正确关系,还同时实现了进程之间的信息交换。目前常用的高级通信机制有消息缓冲通信、管道通信和信箱通信。

1) 消息缓冲通信

基本思想是系统管理若干消息缓冲区,用以存放消息。每当一个进程(发送进程)向另一个进程(接收进程)发送消息时,便申请一个消息缓冲区,并把已准备好的消息送到缓冲区,然后把该消息缓冲区插入到接收进程的消息队列中,最后通知接收进程。接收进程收到发送进程发来的通知后,从本进程消息队列中的一个消息缓冲区取出所需的信息,然后把消息缓冲区还给系统。

2) 管道通信

管道通信由 UNIX 首创,已成为一种重要的通信方式。管道通信以文件系统为基础。所谓管道,就是连接两个进程之间的一个打开的共享文件,专用于进程之间进行数据通信。管道通信的实质是利用外存来进行数据通信,故具有传送数据量大的优点,但通信速度较慢。

3) 信箱通信

为了实现进程间的通信,需设立一个通信机制——信箱,以传送、接收信件。当一个进程希望与另一进程通信时,就创建一个连接两个进程的信箱,通信时发送进程只要把信件投入信箱,而接收进程可以在任何时刻取走信件。

5.2.4 多道程序的组织

多道程序设计环境中,进程个数往往多于处理机数,这将导致多个进程互相争夺处理机。多道程序的组织工作就是要控制、协调进程对 CPU 的竞争,按照一定的调度算法,使某一个就绪进程获得 CPU 的控制权,转换成运行状态,实现对进程的调度,也称作处理机调度。

1. 进程调度

进程调度程序的一项重要工作是根据一定的调度算法从就绪队列中选出一个进程,把 CPU 分配给它。因此,调度算法的好坏直接影响到系统的设计目标和工作效率。通常,考虑调度算法的因素,主要是有利于充分利用系统的资源,发挥最大的处理能力;有利于公平地响应每个用户的服务请求;有利于操作系统的工作效率。常用的调度算法有:

1) 先来先服务

如果早就绪的进程排在就绪队列的前面,迟就绪的进程排在就绪队列的后面,那么先来先服务(First Come First Service,FCFS)总是把当前处于就绪队列之首的那个进程调度到运行状态。也就是说,它只考虑进程进入就绪队列的先后,而不考虑其他因素。FCFS 算法简单易行,但性能却不太好。

2) 最短进程优先

最短进程优先(Shortest Process First,SPF)的基本思想是进程调度程序总是调度当前

就绪队列中的下一个要求 CPU 时间最短的那个进程运行。和先来先服务算法相比，SPF 调度算法能有效地降低平均等待时间和提高系统的吞吐量。

3）时间片轮转法

常用于分时系统中，将 CPU 的处理时间划分成一个个时间片，轮流地调度就绪队列中的诸进程运行一个时间片。当时间片结束时，强迫运行进程让出 CPU，该进程进入就绪队列，等待下一次调度。同时，进程调度程序又去选择就绪队列中的一个进程，分配给它一个时间片，以投入运行。

4）优先级法

进程调度程序总是调度当前处于就绪队列中优先级最高的进程，使其投入运行。进程的优先级通常由进程优先数（整数）表示，数大优先级高，还是数小优先级高取决于规定。

2. 进程死锁

1）死锁的概念

在多道程序系统中，虽可通过多个进程的并发执行来改善系统的资源利用率和提高系统的处理能力，但可能发生一种危险——死锁。所谓死锁（Deadlock），是指多个（进程数大于或等于 2）进程因竞争资源而形成的一种僵持局面。若无外力作用，这些进程将永远不能再向前推进。例如，设有一台打印机和一台磁带机，有两个进程 P_1 和 P_2，它们分别占用打印机和扫描仪，当 P_1 申请已被 P_2 占用的扫描仪，而 P_2 申请已被 P_1 占用的打印机。这种情况出现时，P_1 和 P_2 僵持不下，称为进程死锁，如图 5-5 所示。

P_1：	P_2：
…	…
申请打印机	申请扫描仪
申请扫描仪	申请打印机
使用	使用
释放打印机	释放打印机
释放扫描仪	释放扫描仪
…	…

图 5-5 死锁状态示意图

从图 5-5 可以看出，产生死锁的原因有两个，一是系统内的资源数量不足，比如打印机，扫描仪有多台的话，就不会产生图 5-5 资源抢占的问题，而且这些资源都是独享的，即任何时刻只能有一个进程使用；二是系统内多个进程的推进速度不当，假如图 5-5 中 P_1 用完打印机并释放后，P_2 才开始运行，则两个进程都可以顺利执行。

2）产生死锁的必要条件

在一个计算机系统中，死锁的产生有如下 4 个必要条件。

（1）互斥使用（资源独占）。在一段时间内，一个资源只能由一个进程独占使用，若别的进程也要求使用该资源，则必须等待直至其占用者释放。

（2）不剥夺性（不可抢占）。进程所占用的资源在未使用完之前，不能被其他进程强行剥夺，而只能由占用进程自身释放。

（3）保持和请求。允许进程在不释放其已占用资源的情况下，继续请求并等待分配新的资源。

（4）循环等待。在进程资源图中存在环路，环路中的进程形成等待链。存在一个进程等待序列{P_1，P_2，…，P_n|，P_1 等待 P_2 占用的资源，P_2 等待 P_3 占用的资源，…，P_n 等待 P_1 占用的资源。

3）死锁的预防

通过破坏 4 个必要条件中的一个或多个以确保系统不会发生死锁。为此，可以采取下列 3 种预防措施：采用资源的静态预分配策略，破坏“保持和请求”条件；允许进程剥夺使用

其他进程占有的资源,从而破坏“不剥夺性”条件;采用资源有序分配法,破坏“环路”条件。这是解决发生死锁的一种方法。

4) 死锁的避免

即使死锁必要条件成立,也未必会发生死锁。因此死锁避免是在系统运行过程中小心地推进各进程,避免死锁的最终发生。最著名的死锁避免算法是 Dijkstra 提出的银行家算法。该算法对于进程发出的每一个系统能够满足的资源申请命令加以动态检查。如果发现分配资源后,系统进入不安全状态,则不予分配;若分配资源后,系统仍处于安全状态,则分配资源。所谓安全状态是指在 T0 时刻系统是安全的或系统处于安全状态,仅当存在一个由系统中所有进程构成的进程序列$<P_1,P_2,\cdots,P_n>$,对于每一个进程 $P_i(i=1,2,\cdots,n)$满足,它以后尚需要的资源数量不超过系统中当前剩余资源与所有进程 $P_j(j<i)$当前占有资源数量之和。如果不存在这样的序列,则说明系统处于一种不安全状态。与死锁预防策略相比,死锁避免策略提高了资源利用率,但增加了系统开销。

5) 死锁的解除

检测出系统处于死锁状态,就要将其消除,使系统恢复正常运行。常用的方法如下。

(1) 撤销进程。撤销死锁环中的一个或者多个进程,释放它们占用的资源,使其他进程能继续运行。

(2) 剥夺资源。从死锁进程中选一个进程,剥夺它的资源(一个或多个资源)但不撤销它,把这些资源分配给别的死锁进程,反复做这一工作直到死锁解除。

(3) 设置检查点。一个花费代价较高的解除方法,定时地记录各个进程的执行情况,一旦检查到死锁发生,让一个或多个进程回退到足以解除死锁的地步。

死锁的检查和解除都要花费很大的系统代价,可能影响进程处理效率,但是可以避免发生死锁而给系统带来更大的危害。

5.3 存储空间的管理

计算机系统中,存储空间是系统重要的组成部分,它用于存放计算机中所有的信息,包括程序、文件及数据。存储空间又分为主存储空间(内存)和辅助存储空间(外存)。内存是暂存性存储器,用来存放正在被 CPU 访问和处理的信息。外存是永久性存储器,用来存放长久保存的数据。存储器资源是直接影响系统处理效率的宝贵资源。因此,如何有效地管理存储器资源是操作系统要解决的核心问题之一。

5.3.1 内存储器的管理

CPU 能直接访问的空间是主存储空间,即内存。任何程序和数据必须装入内存之后,CPU 才能对它们进行操作,因而一个作业必须把它的程序和数据存放在内存才能运行,而且操作系统本身也要存放在内存中并运行。内存的划分如图 5-6 所示。存储管理主要是对内存中的用户区域进行管理。

1. 存储器管理的功能

存储器管理的主要任务是为系统提供良好的运行环境,方便用户使用不同类型的存储器,提高存储器的利用率,满足系统对于更大虚拟存储空间的需求。为实现这些任务,存储

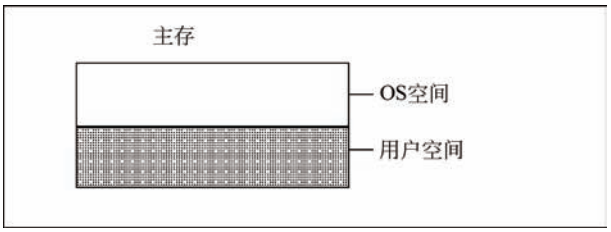


图 5-6 内存空间的划分

器管理必须具有存储分配与回收、地址重定位、存储保护和扩充的功能。

(1) 存储分配与回收。要清晰地掌握系统中所有存储器空间的状态,响应所有要求分配存储器空间的请求,具体分配满足应用需求的存储器空间,回收被进程释放的空间。

(2) 地址变换。存放在外存中的程序使用的是逻辑地址(虚地址),程序装入内存时要分配物理地址(实地址)。因此,将程序中的逻辑地址转换成进程中的物理地址是存储管理的主要功能之一。

(3) 存储扩充。由于多道程序的引入,使内存资源更为紧张,为了使用户在编制程序时不受内存容量的限制,可以在硬件支持下,将外存作为主存的扩充部分供用户程序使用,这就是内存扩充。内存扩充可以使用户程序得到比实际内存容量大得多的"内存"空间,从而极大地方便了用户。

(4) 存储保护。由于各个用户程序和操作系统同在内存,因而一方面要求各用户程序之间不能互相干扰,另一方面用户程序也不能破坏操作系统的信息。因此,为使系统正常运行,必须对内存中的程序和数据进行保护。

2. 存储分配方式

为了实现存储空间的统一管理,系统为所有的存储空间建立一张存储区管理表,用于记录存储区的使用状况。当分配存储区时,操作系统在存储区管理表中进行检索,找出满足实际需求的未使用存储区分配给申请的进程。被分配的存储区状态属性进行相应的修改。

常用的存储分配方式有 3 种。

1) 直接分配

程序员在编写程序时,直接使用主存的物理地址。

2) 静态分配

在程序装入前,一次性分配程序所需的存储空间。整个程序运行期间不会再发生变化。

3) 动态分配

系统在进程的运行过程中为其分配所需的内存,允许用户程序动态申请内存。

早期的操作系统采用静态的、连续的存储分配方式。现代的操作系统通常采用动态的、非连续的存储分配方式,并结合虚拟存储管理系统实现内存资源的主动回收。

3. 地址重定位

用户程序经过编译或汇编形成的目标代码,通常采用相对地址形式,其首地址为零,其余指令中的地址都是相对首地址而定的。这个相对地址就称为逻辑地址或虚拟地址。物理地址是内存中各存储单元的编号,即存储单元的真实地址,它是可识别、可寻址并实际存在的。

为保证 CPU 执行程序指令时能正确访问存储单元,需要将用户程序中的逻辑地址转换为运行时可由机器直接寻址的物理地址,这一过程称为地址映射或地址重定位。地址重定位也分为静态和动态重定位两种。

1) 静态地址重定位

在用户程序装入到内存的过程中,实现逻辑地址到物理地址的转换,以后在程序运行时不再改变。

2) 动态地址映射

当执行程序过程中要访问指令或数据时,才进行地址变换,又称动态重定位。动态重定位需要依靠硬件地址映射机制完成,比如重定位寄存器。动态重定位可以为装入的程序分配到不连续的存储空间。

图 5-7 给出了静态地址重定位的一个示意图。其中,a、b、c 是程序定义的 3 个变量。经编译后,在目标程序的地址空间为 a、b 和 c 分配了地址 m1、m1+1 和 m1+2。当程序运行时,操作系统将 a、b 和 c 的逻辑地址转换成能够在内存中存放和处理的物理地址(16 位操作系统)2FF0、3EC0 和 6DAA。

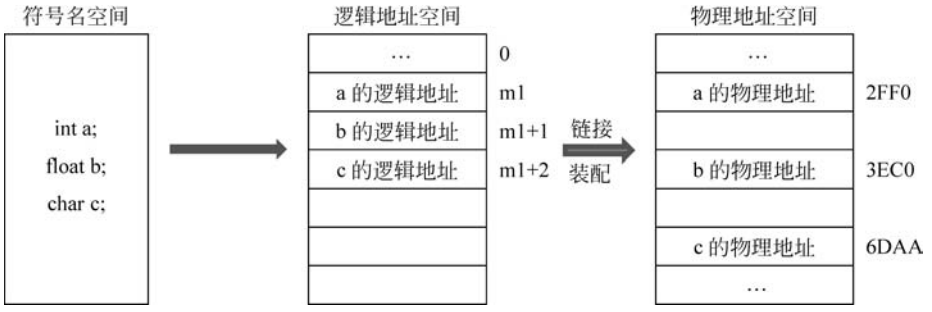


图 5-7 静态地址重定位示意图

4. 存储扩充

存储管理中面临很现实的一个问题是“地址空间超过内存空间的程序能否运行”。现代计算机系统中,大于内存空间的程序也能运行,这就是由存储管理的存储扩充功能实现的。目前最常用的扩充主存的方法有自动覆盖技术、交换技术和虚拟存储技术。

1) 自动覆盖技术

这种方法的主要思想是将大于内存空间的程序按逻辑功能划分为若干个小的程序段,这些程序段的最大特点是它们的体积小于主存空间且能够独立地运行;每次运行时只装入其中的一个程序段到内存,该段程序执行完后,下一个装入的程序段覆盖在当前程序段弃用的主存空间中;从而解决了在小内存空间中运行大地址空间程序的问题。自动覆盖技术以此达到扩充主存空间的目的。由于在此过程中既要解决存储空间分配与回收问题,还要解决程序段的调度和上下程序段的衔接问题,增加了存储管理的难度。

2) 交换技术

这种方法是对自动覆盖技术的改进,其目的是为了更加充分地利用系统的各种资源(包括内、外存储器、CPU 等)。采用交换技术的思路是当一个进程运行受阻时,将该进程交换出主存,再装入下一个可运行的进程,交换出去的进程状态又恢复为就绪态后,再重新将其

装入，以此达到扩充主存的目的。

3）虚拟存储技术

虚拟存储技术实质上是综合吸收了自动覆盖技术和交换技术而形成的一种存储管理技术。它把程序存在的地址空间与程序运行时用于存放程序的存储空间区分开来。在这种方法中，一个大程序分割为相互独立、逻辑上关联的小程序段，每个程序的地址空间也划分为若干个部分，它们在使用时加载到主存中；不用时，请求操作系统的交换功能，将其交换到外存中。因此，在有限存储空间的系统中可以运行大得多的应用程序。

例如，进程的某些程序段在进程整个运行期间，可能根本不使用（如出错处理等），因而没有必要调入内存；互斥执行的程序段在进程运行时，系统只执行其中一段，因而没必要同时驻留内存；在进程的一次运行中，有些程序段执行完毕，从某一时刻起不再使用，因而没必要再占用内存区域。

5. 存储分配管理方法

多道程序环境中，当多个程序提出申请分配存储空间的请求后，系统将采用怎样的分配策略呢？静态存储分配和动态存储分配在存储管理上有无区别？下面简单介绍在多道程序环境下，存储管理的不同方法和技术。

1）单一连续区分配

单一连续区分配法是像 MS-DOS 这样的单用户、单任务的操作系统采用的最基本、最简单的方法。其要点是把主存空间分割为两个固定的区域，一个区域固定地分配给操作系统，另一个区域分配给用户程序。其采用静态分配方式，不支持虚拟存储器技术。其存储空间的利用率低，造成存储资源的极大浪费。

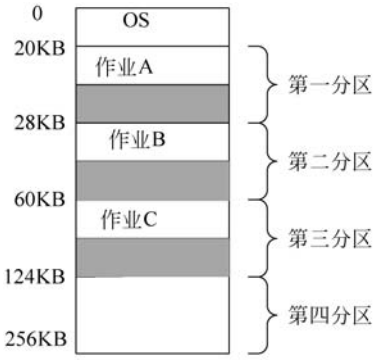
2）分区存储管理

分区存储管理是满足多道程序运行的最简单的存储管理方案，这种管理方法特别适用于小型机、微型机上的多道程序系统。其基本思想是将内存划分成若干个连续区域，称为分区。在每个分区中装入一个运行作业，用硬件措施保证各个作业互不干扰。分区的划分方式有固定分区方式，可变分区方式及可重定位分区方式。

（1）固定分区分配管理，也称静态分区，是事先将可分配的内存空间划分成若干个固定大小的连续区域，每个区域大小可以相同，也可以不同。当某一作业要调入内存时，存储管理程序根据它的大小，找出一个适当的分区分配给它。如果当时没有足够大的分区能容纳该作业，则通知作业调度程序挑选另一作业。

固定分区方式虽然简单，但由于一个作业的大小，不可能刚好等于某个分区的大小，故内存利用率不高。每个分区剩余的空白空间，称为“碎片”。例如，图 5-8 所示为固定分区管理示意图，图中阴影部分即为主存“碎片”。

（2）可变分区分配管理，也称动态分区，为了减少固定分区法造成的主存垃圾，人们想到了“量体裁衣”式的分配方法。这种方式在作业将要装入内存时，按作业的大小来划分分区。即根据作业需要的内存量查看内存是否有足够大的内存空闲区。若有，则按需要建立一个分区分配给该作业；若无，则令该作业等待。当某个程序运行完毕撤销时，其占用的分区空间被收回。一般情况下，系统把与其相邻的其他空闲区合并成一个更大的空闲区。由于分区的大小是按装入作业的实际需要量来定的，所以克服了固定分区的缺点，提高了内存的利用率。



(a)

分区号	起始地址	容量	状态
1	20KB	8KB	已分配
2	28KB	32KB	已分配
3	60KB	64KB	已分配
4	124KB	132KB	未分配

(b)

图 5-8　固定分区管理示意图

(3) 存储分配算法,主要用于查找主存中较为合适的空闲区域并进行分配。常用的算法有最先适应算法、最佳适应算法和最坏适应算法等。由于篇幅有限,感兴趣的读者可以参考相关书目了解算法详情。

3) 分页存储管理

分区管理时,一道作业要占用内存的一个或几个连续的分区。因此当内存的连续空闲区域不够存放一道作业时,就得大量移动已在内存中的信息。这不仅不方便,而且大大增加了系统的开销。为了克服上述管理的不足,1961 年曼彻斯特大学的 Arlas 研究小组,在 Atlas 计算机上,首先采用了分页存储管理技术。分页存储管理解决了不连续存储碎块再利用的问题。

(1) 分页存储管理,把内存划分成若干相同大小的存储区域,每个区域称为一个块;把用户作业地址空间也按同样大小分成若干页;系统以块为单位把内存分配给各作业的各个页,每个作业占有的内存块无须连续。

分页存储管理中,用户作业的地址空间本来是一个一维的连续地址空间,当它划分为页后,就变成二维空间。即用户作业的逻辑地址由页号和页内地址两部分组成。同理,在主存空间划分为块后,进程中的任何一个物理地址也是由两部分组成,即块号 B 和块内地址 D。装入程序分配存储空间时,系统通过动态地址转换机制(一种地址转换的硬件装置),就可以实现由逻辑地址空间到物理地址空间的转换。转换公式如下:

$$\text{绝对地址} = \text{块号} \times \text{块长度} + \text{块内地址} \tag{5-1}$$

分页系统中,为了保证在连续的逻辑地址空间中的作业能在不连续的物理地址下正确运行,系统为每个程序作业建立一个地址变换表,简称页表。页表中的每一个表项由两部分组成,页号和该页所对应的物理块号。程序作业的地址空间有多少页,它的页表中就登记多少行,且按逻辑页的顺序排列。页表存放在内存系统区内。为了方便查找页表表项,系统还设立了一个“控制寄存器”,用于存放 CPU 正在处理的程序所对应页表的起始地址及该程序的页数,格式为页数,页表起始地址。

例如,控制寄存器值为“3,1500”,表示当前正在处理的程序共有 3 页,它的页表起始地址为 1500。设程序的逻辑地址空间划分为 1024 字节大小的若干页,由管理程序将其分别分配给主存空间的第 2、第 3 和第 8 块。程序作业的具体任务是从逻辑地址为 2500 处取得

一个数据 999 到第 L 个记录中。图 5-9 给出了该例逻辑空间与主存空间的对应关系。由页表可知，程序的逻辑地址页面对应主存物理地址块号分别为第 2、第 3 和第 8 块。系统自动把地址码 2500 转换成两部分，即 2500＝2×1024＋452，其中 2 为页号，1024 是页的大小，452 是页内偏移量。产生物理地址时，系统通过控制寄存器确定页表的起始位置，然后找到页表中页号为 2 的表项，由此知对应的主存块号为 8。根据式(5-1)计算，就得到逻辑地址 2500 对应的 8644 这一物理地址，也就是 999 这一数据在主存中的实际存放位置。

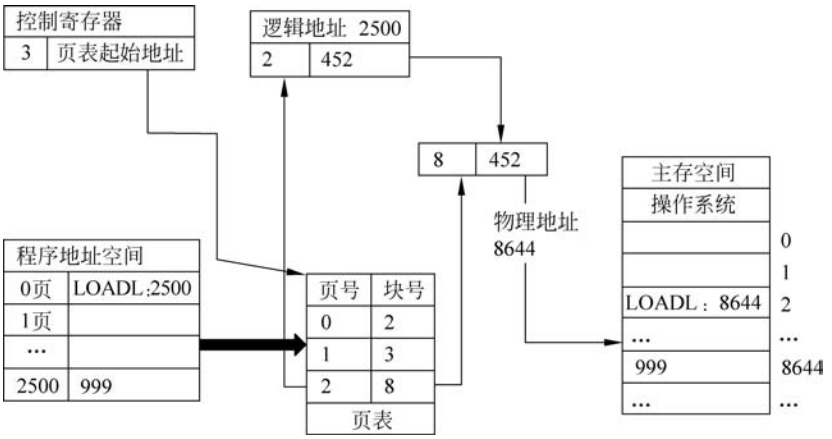


图 5-9　分页存储管理页表及地址转换示意图

分页存储管理有效解决了主存空间的“垃圾碎块”问题，易于实现代码段的共享，以及用户可以连续编址。由于采用硬件的动态变址机构，增大了系统的成本和开销，要求运行的程序必须整体装入内存。如果主存空间不足，系统将拒绝执行程序。

(2) 请求分页存储管理是对分页管理的改进。其核心是借助交换技术实现分页管理。首先，在进程开始执行前，不是装入其全部页面，只装入几个当前运行必需的部分页面；然后，根据进程执行时的需要，动态地装入其他页面。当主存已占满而新的页面需要装入时，则依据某种算法将暂时不用的页面交换到外存中。按实际需要装入页面的做法可避免装入那些在进程执行中用不着的页面，比起纯粹的页式管理，无疑节省了主存空间。为此需要将页表表项的结构扩充，如图 5-10 所示。

页号	物理块号	状态位	访问位	修改位	外存地址

图 5-10　请求分页页表表项示意图

表项各字段的作用如下：

① 页号和物理块号与普通分页页表相同。②状态位用于标记该页是否在主存中，如果不在主存，则产生调页请求。③访问位用于记录页面在指定时间内被访问的次数，或最近已有多长时间未被访问。④修改位用于标记页面调入内存后是否被修改过。分页存储管理机制中，页面在装入内存的同时在外存中保留该页面的备份。为了减少写磁盘的次数，对那些未经修改的页面，在换出主存时不执行写外存操作。⑤外存地址，即该页面的备份在外存中的存放位置。

4）分段存储管理

分页存储管理中为作业分配的主存空间地址可以是不连续的，但程序的逻辑空间地址仍然要求是连续的。在实际中，一个用户的程序往往由若干功能相对独立的模块组成，例如主程序模块、子程序模块和数据块等。这就要求编译链接程序将这些程序段按一维空间顺序线性地址排序，从而给程序和数据的共享带来困难。为解决此问题，人们提出了分段存储管理的思想。

分段存储管理下，每个用户程序可由若干段组成，每段可以对应于一个过程、一个程序模块或一个数据集合，段间的地址可以是不连续的，但每一段内的地址是连续的。分段存储管理就是以段作为基本单位的主存管理方法。作业的逻辑地址由段号和段内地址两部分组成，如图 5-11 所示。

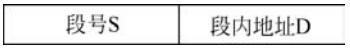


图 5-11　作业的逻辑地址

系统以段为单位进行内存分配，为每一个逻辑段分配一块连续的内存区域，逻辑上连续的段在内存中不一定连续存放。

为了实现逻辑地址到物理地址的变换，系统为每个用户程序建立一张段表，记录各段的段号、段长及内存起始地址等内容。用户程序有多少逻辑段，该段表里就登记多少行，且按逻辑段的顺序排列。段表存放在内存系统区里。

例如，某用户程序划分为 4 段，分别是主程序 Main、子程序 Add、数据块 Data 和工作区 Work。如图 5-12 所示。主程序中指令 Call[Add]|<Y>的功能是调用子程序 Add 中地址 Y 的指令，Add 对应 1 段，Y 对应 1 段内 300 地址单元，即[Add]|<Y>对应的逻辑地址是 1:300。同样主程序中指令 Load L1,[Data]|<X>的功能是将数据块 Data 中地址 X 的数据读入到变量 L1 中，Data 对应 2 段，X 对应 2 段内的 150 地址单元，即[Data]|<X>对应的逻辑地址是 2:150。分段存储管理模式下，图 5-12 中各程序段分段存储地址空间变换过程如图 5-13 所示。Call 1:300 中 1:300 的逻辑地址经地址变换转换后变为主存中的物理地址 8492(因为 1 段的起始地址是 8K，段内地址是 300，8×1024+300=8492)。

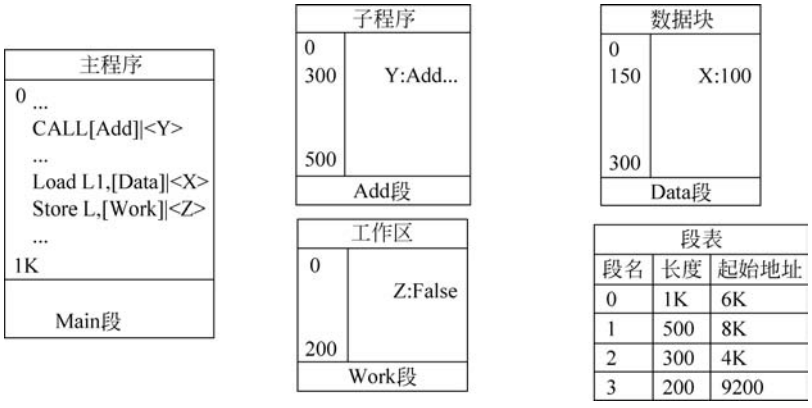


图 5-12　分段地址空间示意图

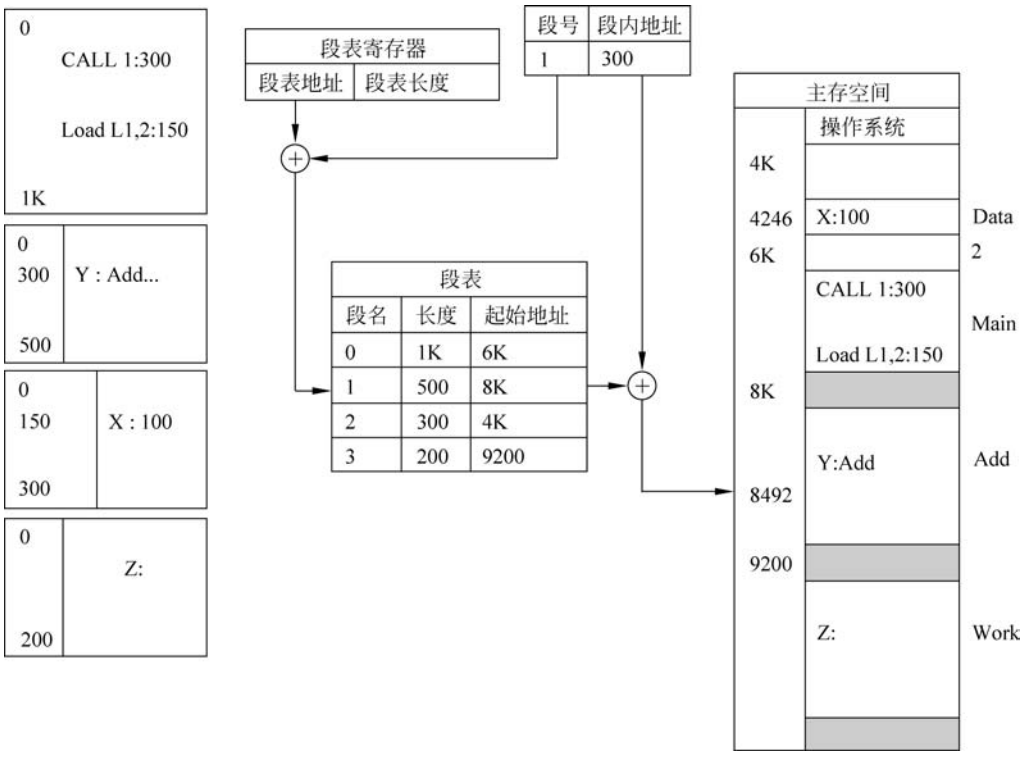


图 5-13　分段存储管理地址变换示意图

从实现技术上看，分段管理与分页管理很相似，但在概念上二者有本质的不同。段是用户可知的逻辑单位，它由用户在程序设计时确定，而页是用户不可知的物理单位，页的大小由操作系统事先确定；段的长度不固定，由用户根据实际问题的性质来划分确定，而页的长度固定，由系统确定。

分段存储管理的优点是便于模块化处理、分段共享、易于保护和动态链接等；缺点是增设硬件（地址转换机构）提高了系统成本，附加了地址转换功能和为段提供主存空间加大了系统的开销，由于段的尺寸较大而产生出较大的主存碎块。

5）段页式存储管理

为了获得分段方式和分页方式的优点，将分段和分页两种方法进行综合演变而形成的管理方式即为段页式存储管理。

段页式系统的基本原理是段式和页式原理的结合，即先将用户程序分为若干个段，再把每个段划分成若干页。其基本思想是用分段的方法来分配和管理虚拟存储器，而用分页方法来分配和管理实存储器。为了进行存储管理，系统为每个用户程序建立一张段表，用于记录各段的段号、页表起始地址和页表长度；为用户程序中的每一段各建立一张页表，用于记录该段中各页与物理块号之间的对应关系。

段页式存储管理中，系统在进行地址变换时，根据控制寄存器找到指定段号，再根据段号找到相应的页表，由页表中的页号得到对应的块号，再与页内地址计算得到块的物理地址，如图 5-14 所示。

段页式系统综合了段式和页式各自的优点。其缺点是增加了硬件成本，并且软件也变

得复杂,占用了不少处理机时间。此外,段表、页表和页内零头仍占用了不少存储空间。

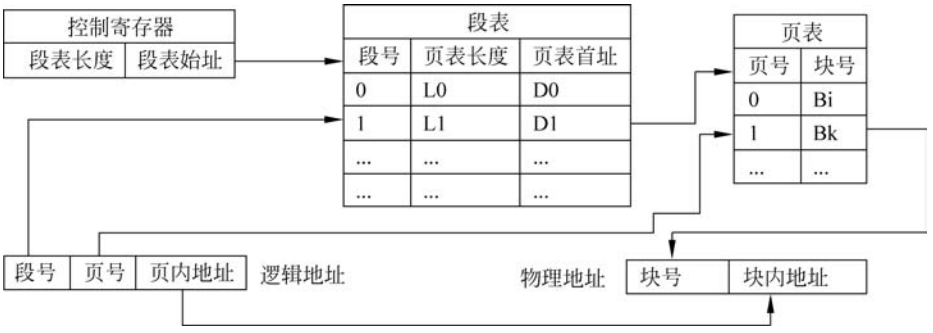


图 5-14 段页式存储管理地址转换示意图

5.3.2 外存储器中文件的组织结构

当人们在计算机上完成各种简单或复杂的计算和处理工作时,需要一个安全、可靠和共享的信息存储系统。用户并不希望了解自己数据存放的形式及存放时占用的空间等信息,用户只希望能够快速存取、共享和管理相关数据。操作系统实现的文件系统很好地解决了这个问题。文件系统是建立在存储设备上的信息存储管理系统,它负责管理和分配设备的存储空间,以文件为单位存储数据或程序。文件系统是计算机系统的信息记忆中心,是操作系统的重要组成部分。

1. 文件与文件系统

所谓文件是具有符号名的,在逻辑上具有完整意义的一组相关信息项的有序序列。它是一种在磁盘上保存信息而且能方便以后读取的方法。文件用符号名加以标识,这个符号名通常也称为文件名。文件名是用户在创建文件时确定的,并在以后访问文件时使用。其命名规则在各个操作系统中不尽相同。

所谓文件系统是操作系统中负责管理和存取文件信息的软件机构。包括与文件管理有关的软件、被管理的文件及实施文件管理所需的数据结构。从系统角度看,文件系统是对文件存储器的存储空间进行组织和分配,负责文件的存储并对存入的文件进行保护和检索的系统。具体地说,它负责为用户建立文件;存入、读出、修改、转储文件;控制文件的存取;当用户不再使用时撤销文件。

2. 文件的结构

文件的结构是指以什么样的形式去组织一个文件。从用户角度看到的文件组织形式,用户以这种形式存取、检索和加工有关信息的文件称为文件的逻辑结构。从系统存储角度组织的文件称为文件的物理结构。

1) 文件的逻辑结构

文件的逻辑结构有流式结构和记录式结构两种。

(1) 流式文件是有序字符的集合,其长度为该文件所包含的字符个数,所以又称为字符流文件。流式文件无结构,且管理简单,用户可以方便地对其进行操作。源程序、目标代码等文件属于流式文件。UNIX 系统采用的是流式文件结构。

(2) 记录式文件构成文件的基本单位是记录,记录式文件是一组有序记录的集合。记

录式文件可把记录按各种不同的方式排列，以便用户对文件中的记录进行修改、追加、查找和管理。记录式文件可分为定长记录文件和变长记录文件两种。

2）文件的物理结构

文件的物理结构指文件的内部组织形式，即文件在物理存储设备上的存放方法。它和文件的存取方法密切相关。文件的物理结构好坏，直接影响到文件系统的性能。因此，只有针对文件或系统的适用范围建立起合适的物理结构，才能既有效地利用存储空间，又便于系统对文件的处理。

根据文件空间中的存放形式，文件可分为连续文件（连续存放）、串联文件（链接存放）和索引文件（索引表存放）。

连续文件是一种最简单的文件物理结构，它把逻辑上连续的文件信息依次存放在连续编号的物理块中，只要知道文件在存储设备上的起始地址（首块号）和文件长度（总块数），就能很快地进行存取。这种结构的优点是访问速度快，缺点是增加文件长度困难。

串联文件是将逻辑上连续的文件分散存放在若干不连续的物理块中，每个物理块设有一个指针，指向其后续的物理块。只要指明文件的第一个块号，就可以按链指针检索整个文件。这种结构的优点是文件长度容易动态变化，其缺点是不适合随机存取访问。

索引文件的组织方式要求为每个文件建立一张索引表，表中的每个项目指出了文件的逻辑块号和与之对应的物理块号。索引表也以文件的形式存在磁盘上，只要给出索引表的地址，通过索引表就可以查找到文件信息的存放位置。这种结构有利于进行随机存取，并具备串联文件的所有优点。缺点是存储开销大，因为每个文件有一个索引表，而索引表也要占用存储空间。

3）文件的存取方式

文件的存取方法按照存取的顺序关系，通常分为顺序存取和随机存取。

对于记录式文件的存取，顺序存取是严格按照记录排列的顺序依次进行存取。磁带机只能采用顺序存取方式。

随机存取方法允许随机存取文件中的记录，而不管上次存取了哪一个记录。这种存取方式对很多应用程序是必须具有的，如数据库系统。

3. 文件目录

一个计算机系统中保存有许多文件，用户在创建和使用文件时只给出文件的名字，由文件系统根据文件名找到指定文件。为了便于对文件进行管理，设置了文件目录，用于检索系统中的所有文件。文件系统的基本功能之一就是负责目录的编排、维护和检索。因此，要求目录的编排便于寻址，并且要防止冲突和便于目录的迅速检索。

对于文件，操作系统仍然用控制块来管理，即为每一个文件开辟一个存储区，在里面记录着该文件的有关信息，该存储区域称为文件控制块（file control block，FCB）。FCB 是文件存在的标志，记录了系统管理文件所需要的全部信息。FCB 通常包括文件名、文件号、用户名、文件的物理位置、文件长度、记录大小、文件类型、文件属性、共享说明、文件逻辑结构和文件物理结构等信息。

把文件的 FCB 汇集在一起，就形成了系统的文件目录，每个 FCB 就是一个目录项，给定一个文件名，通过查找文件目录便可找到该文件对应的目录项（即 FCB）。

为了实现文件目录的管理，通常将文件目录以文件的形式保存在外存空间，这个文件就

称为目录文件。

文件目录的组织与管理是文件管理的一个重要方面。目前大多数操作系统,如 UNIX 等都采用多级目录结构,又称树形目录结构。图 5-15 是树形目录结构。

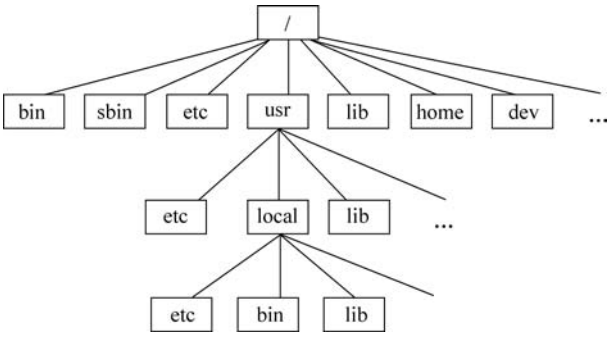


图 5-15 树形目录结构

树根结点称为根目录,根目录是唯一的,由它开始可以查找到所有其他目录文件和普通文件,根目录一般放在内存。从根结点出发到任一非叶结点或树叶结点都有且仅有一条路径,该路径上的全部分支组成一个全路径名。采用多级目录结构时,文件名为一个路径名,可以解决重名的问题。

多级目录结构的优点是便于文件分类,可为每类文件建立一个子目录;查找速度快,因为每个目录下的文件数目较少;可以实现文件共享。缺点是比较复杂。

4. 存储空间的管理

文件管理不仅需要为文件数据分配存储块,还需要为文件索引表和文件控制块分配存储空间。所以,需要采用合适的数据结构,描述设备上的空闲存储块信息,以便分配和回收设备存储块。存储空间的管理方法包括连续分配、链接分配、索引分配和位图分配。连续分配法主要应用在磁带之类的顺序存储设备上。下面以磁盘为对象,介绍后 3 种方法。

1) 链接分配(空闲块链)

系统将所有的空闲物理块连成一个链,用一个指针指向第一个空闲块,然后每个空闲块含有指向下一个空闲块的指针,最后一块的指针为空,表示链尾。外存空间的申请和释放以块为单位,申请时从链首取一块,释放时将其链入链尾。空闲块链节省内存,但申请释放速度较慢,实现效率较低。

2) 索引分配(空闲块表)

文件系统建立一张空闲块表,该表记录全部空闲的物理块,包括首空闲块号和空闲块个数。空闲块表方式特别适合于文件物理结构为顺序结构的文件系统。

建立新文件时,系统查找空闲块表,寻找合适的表项,分配一组连续的空闲块。如果对应表项所拥有的空闲块个数恰好等于所申请值,就将该表项从空闲块表中删去。当删除文件时,系统收回它所占用的物理块,考虑是否可以与原有空闲块相邻接,合并成更大的空闲区域,最后修改有关表项。

3) 位图分配(位示图)

使用位示图(Bitmap)表示所有存储块的分配状态。按照设备逻辑地址顺序建立位图单元与存储块的一一对应关系。每一个磁盘物理块对应一个二进制位,如果物理块为空闲,

则相应的二进制位为0；如果物理块已分配，则相应的二进制位为1。

申请磁盘物理块时，可在位示图中从头开始查找为0的字位，将其改为1，返回对应的物理块号；归还物理块时，在位示图中将该块对应的字位改为0。位示图描述能力强。一个二进制位就描述一个物理块的状态。位示图较小，可以复制到内存，使查找既方便又快速。位图分配是操作系统中经常采用的一种设备存储空间分配方法。

5. 文件的存取控制

文件系统的建立使多个用户共享存储设备的存储空间。因此，文件系统的一个重要任务是保护用户所存放的数据，保证用户在授权的条件下访问这些数据。文件系统的存取控制主要包括访问权限的定义、设置、审批和检查4个部分。

1）定义访问权限

根据系统安全级别的需求，定义访问权限的类别。操作系统中，文件的访问权限通常分成读、写和执行3种。在安全性较高的操作系统中，还会进一步细化访问权限。

2）设置访问权限

通常由文件所有者设置文件的访问权限，即规定哪些用户对文件拥有哪些访问权限。对某个文件的全部授权信息组成该文件的访问控制表，通常将用户分组或分类，为每类用户指定访问权限，以此简化访问控制表，节省空间开销。

3）审批访问权限

用户在打开文件时，申请其所需要的文件访问权限，以便能进行后续的文件操作。只有当申请的访问权限获得准许时，内核才完成打开文件的请求，建立一个文件资源对象，将获准的访问权限存放其中。

4）检查访问权限

审批权限只是实施文件访问控制的第一步。此后，每当用户进行一次文件操作，操作系统都需要检查该操作是否在用户获准的访问权限之内。任何超出用户获准权限的文件操作都不允许。

5.4 小结

计算机硬件是组成计算机的物理设备的总称，包括中央处理器、存储器，输入输出设备等。本章首先介绍了操作系统的概念，从操作系统的发展、定义、分类、特征、功能这几方面进行了详细介绍。而多道程序设计是操作系统采用的最基本、最重要的技术，其根本目的是提高整个系统的效率。多道程序系统主要从并发程序设计、进程、进程之间的通信、多道程序的组织几方面进行诠释。计算机系统中，存储空间是系统重要的组成部分，它用于存放计算机所有的信息，包括程序、文件及数据。存储结构分为内存储器和外存储器。如何有效地管理存储器资源是操作系统要解决的核心问题之一。

5.5 习题

1. 单项选择题

（1）计算机的操作系统是一种（　　）。

A. 应用软件　　B. 系统软件　　C. 工具软件　　D. 字表处理软件

(2) 工业过程控制系统中运行的操作系统最好是(　　)。

A. 分时系统　　B. 实时系统

C. 分布式操作系统　　D. 网络操作系统

(3) 对处理事件有严格时间限制的系统是(　　)。

A. 分时系统　　B. 实时系统

C. 分布式操作系统　　D. 网络操作系统

(4) 批处理系统的主要缺点是(　　)。

A. 没有交互性　　B. 系统资源利用率不高

C. 系统吞吐率小　　D. 不具备并行性

(5) 从资源处理的角度看,操作系统的功能是进行处理机管理、(　　)管理、存储管理、设备管理和文件管理。

A. 硬件　　B. 软件　　C. 作业　　D. 进程

(6) 进程和程序的根本区别在于(　　)。

A. 是不是调入到内存中

B. 是不是占有处理器

C. 是不是具有就绪、运行和等待 3 种状态

D. 是不是具有静态与动态特点

(7) 进程在 3 个基本状态中转换,肯定不会有的转换是(　　)。

A. 运行态⇨就绪态　　B. 阻塞态⇨运行态

C. 运行态⇨阻塞态　　D. 阻塞态⇨就绪态

(8) 在单处理器系统中,如果同时存在 10 个进程,则处于就绪队列中的进程最多有(　　)个。

A. 1　　B. 8　　C. 9　　D. 10

(9) 每一个进程在执行过程中的任一时刻,可以处于(　　)个状态。

A. 1　　B. 2　　C. 3　　D. 4

(10) 系统出现死锁的根本原因是(　　)。

A. 资源管理和进程推进顺序都不得当　　B. 系统中进程太多

C. 资源的独占性　　D. 作业调度不当

2. 填空题

(1) 操作系统的资源管理的功能可分为_____、_____、_____和_____ 4 个部分。

(2) 操作系统的基本特性包括_____、_____、_____。一个作业从进入系统到运行结束,一般要经历_____、_____、_____、_____ 4 种状态。

(3) 为了管理和调度作业,当作业收容到外存储器后,系统为每个作业建立一个_____,它详细记录每个作业的有关信息。

(4) 进程的基本状态是_____、_____和_____。

(5) 进程具有_____、_____、_____、制约性、结构性等 5 个特性。

(6) 从管理角度看,操作系统是管理资源的_____。

(7) 退出等待状态的进程将进入_____。

(8) 从人机交互方式来看,操作系统是用户与计算机软、硬件之间的_____。

(9) 操作系统是运行在计算机_____系统上的最基本的系统软件。

3. 判断题

(1) (　　)早期的批处理系统中,用户可以用交互式方式方便地使用计算机。

(2) (　　)批处理系统的主要优点是系统的吞吐量大、资源利用率高、系统的开销较小。

(3) (　　)资源共享是现代操作系统的一个基本特征。

(4) (　　)程序在运行时需要很多系统资源,例如内存、文件、设备等。因此操作系统以程序为单位分配系统资源。

(5) (　　)当一个进程从等待态变成就绪态,则一定有一个进程从就绪态变成运行态。

(6) (　　)当条件满足时,进程可以由阻塞状态直接转换为运行状态。

(7) (　　)当条件满足时,进程可以由阻塞状态转换为就绪状态。

(8) (　　)当条件满足时,进程可以由就绪状态转换为阻塞状态。

(9) (　　)当条件满足时,进程可以由运行状态转换为就绪状态。

(10) (　　)进程是具有一定独立功能的程序关于某个数据集合上的一次运行活动。