第 1 章　计算机系统基础知识

本章主要介绍计算机系统的基本组成、计算机中数据的表示和运算、计算机系统硬件基础组成、指令系统以及多媒体系统等基础知识。

1.1　计算机系统的基本组成

计算机系统是由硬件系统和软件系统组成的，通过运行程序来协同工作。计算机硬件是物理装置，计算机软件是程序、数据和相关文档的集合。计算机系统的基本组成如图 1-1 所示。

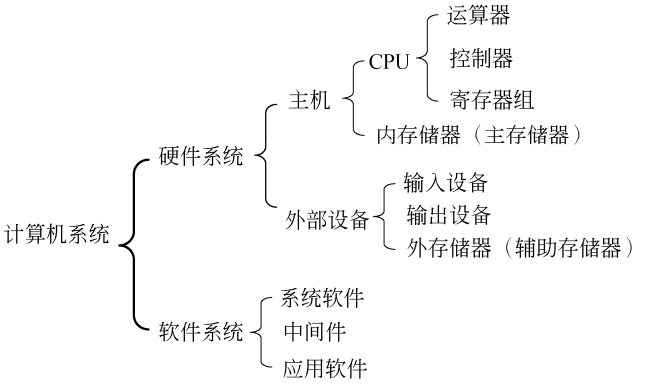

图 1-1　计算机系统的基本组成

1．计算机硬件

基本的计算机硬件系统由运算器、控制器、存储器、输入设备和输出设备五大部件组成，随着网络技术的发展和应用，通信部件也成为计算机系统的基本组件。运算器和控制器及其相关部件已被集成在一起，统称为中央处理单元（Central Processing Unit，CPU）。CPU 是硬件系统的核心，用于数据的加工处理，能完成各种算术、逻辑运算及控制功能。

运算器是对数据进行加工处理的部件，它主要完成算术和逻辑运算。控制器的主要功能则是从主存中取出指令并进行分析，控制计算机的各个部件有条不紊地完成指令的

功能。

存储器是计算机系统中的记忆设备，分为内部存储器（Main Memory，MM，简称内存、主存）和外部存储器（简称外存，辅存）。相对来说，内存速度快、容量小，一般用来临时存储计算机运行时所需的程序、数据及运算结果。外存容量大、速度慢，可用于长期保存信息。寄存器是 CPU 中的存储器件，用来临时存放少量的数据、运算结果和正在执行的指令。与内存储器相比，寄存器的速度要快得多。

习惯上将 CPU 和主存储器的有机组合称为主机。输入/输出（I/O）设备位于主机之外，是计算机系统与外界交换信息的装置。所谓输入和输出，都是相对于主机而言的。输入设备的作用是将信息输入计算机的存储器中，输出设备的作用是把运算结果按照人们所要求的形式输出到外部设备或存储介质上。

2．计算机软件

计算机软件是指为管理、运行、维护及应用计算机系统所开发的程序和相关文档的集合。如果计算机系统中仅有硬件系统，则只具备了计算的基础，并不能真正计算，只有将解决问题的步骤编制成机器可识别的程序并加载到计算机内存开始运行，才能完成计算。

软件是计算机系统中的重要组成部分，通常可将软件分为系统软件、中间件和应用软件。系统软件的主要功能是管理系统的硬件和软件资源，应用软件则用于解决应用领域的具体问题，中间件是一类独立的系统软件或服务程序，常用来管理计算资源和网络通信，提供通信处理、数据存取、事务处理、Web 服务、安全、跨平台等服务。

3．计算机分类

计算机技术的发展异常迅速，将更多的元件集成到单一的半导体芯片上，使得计算机变得更小，功耗更低，速度更快。

（1）个人移动设备（Personal Mobile Device，PMD）。指一类带有多媒体用户界面的无线设备，如智能手机、平板电脑等。

（2）桌面计算机。桌面计算机的产品范围非常广泛，包括低端的上网本、台式计算机、笔记本计算机以及高配置的工作站，核心部件是基于超大规模集成电路技术的 CPU。台式计算机和笔记本计算机属于微型计算机，常用于一般性的办公事务处理等，工作站则是一种高档的微型计算机，通常配有高分辨率的大屏幕显示器及容量很大的内存储器和外部存储器，具备强大的数据运算与图形、图像处理能力，主要面向工程设计、动画制作、科学研究、软件开发、金

融管理、信息服务、模拟仿真等专业应用领域。

（3）服务器。不同于桌面计算机，服务器代替了传统的大型机，主要提供大规模和可靠的文件及计算服务，强调可用性、可扩展性和很高的吞吐率。

（4）集群/仓库级计算机。集群机是将一组桌面计算机或服务器用网络连接在一起，运行方式类似于一个大型的计算机。将数万个服务器连接在一起形成的大规模集群称为仓库级计算机。

（5）超级计算机。超级计算机的基本组成在概念上与个人计算机无太大差异，但规格高，性能要强大许多，具有很强的计算能力，但是能耗巨大。我国的超级计算机主要有银河、天河、曙光、神威四个系列。例如，神威·太湖之光由 40 个运算机柜和 8 个网络机柜组成，共有 40960 块处理器，每一块处理器相当于 20 多台常用笔记本计算机的计算能力。

（6）嵌入式计算机。嵌入式计算机是专用的，是针对某个特定的应用，如针对网络、通信、音频、视频或针对工业控制，对功能、可靠性、成本、体积、功耗有严格要求的计算机系统。日常生活中常见的微波炉、洗衣机、数码产品、网络交换机和汽车中都采用嵌入式计算机技术。

1.2　数据的表示及运算

1.2.1　计算机中数据的表示

在计算机内部，数值、文字、声音、图形图像等各种信息都必须经过数字化编码后才能被传送、存储和处理。所谓编码，就是采用少量的基本符号，选用一定的组合原则，来表示大量复杂多样的信息。基本符号的种类和这些符号的组合规则是一切信息编码的两大要素。例如，用 10 个阿拉伯数码表示数字，用 26 个英文字母表示英文词汇等，都是编码的典型例子。

1．进位计数制及其转换

在采用进位计数的数字系统中，如果只用 r 个基本符号表示数值，则称其为 r 进制（Radix-r Number System），r 称为该数制的基数（Radix）。不同数制的共同特点如下。

（1）每一种数制都有固定的符号集。例如，十进制数制的基本符号有十个：0，1，2，…，9。二进制数制的基本符号有两个：0 和 1。

（2）每一种数制都使用位置表示法。即处于不同位置的数符所代表的值不同，与它所在位置的权值有关。例如，十进制数 1234.55 可表示为

$$1234.55 = 1\times10^{3} + 2\times10^{2} + 3\times10^{1} + 4\times10^{0} + 5\times10^{-1} + 5\times10^{-2}$$

计算机中常用的进位数制有二进制、八进制、十进制和十六进制，如表 1-1 所示。

表 1-1 常用计数制

进 位 制	二 进 制	八 进 制	十 进 制	十 六 进 制
规则	逢二进一	逢八进一	逢十进一	逢十六进一
基数	r =2	r =8	r =10	r =16
数符	0，1	0，1，2，…，7	0，1，2，…，9	0，1，2，…，9，A，B，…，F
权	2^i	8^i	10^i	16^i
形式表示符	B	O	D	H

可以看出，十进制计数制中权的值恰好是基数 10 的某次幂，其他计数制同理。因此，对任何一种进位计数制，其表示的数都可以写成按权展开的多项式，在此基础上实现不同计数制的相互转换。

1）十进制计数法与二进制计数法的相互转换

在十进制计数制中，r=10，基本符号为 0，1，2，…，9。

在二进制计数制中，r=2，基本符号为 0 和 1。二进制数中的一个 0 或 1 称为 1 位（bit）。

将十进制数转换成二进制数时，整数部分和小数部分分别转换，然后再合并。十进制整数转换为二进制整数的方法是“除 2 取余”；十进制小数转换为二进制小数的方法是“乘 2 取整”。

【例 1-1】 把十进制数 175.71875 转换为相应的二进制数。

算式	商	余数
175 / 2	87	1
87 / 2	43	1
43 / 2	21	1
21 / 2	10	1
10 / 2	5	0
5 / 2	2	1
2 / 2	1	0
1 / 2	0	1

$175_{10} = 10101111_2$

算式	乘积
0.71875 * 2	1.43750
0.4375 * 2	0.8750
0.875 * 2	1.750
0.75 * 2	1.50
0.5 * 2	1.0

$0.71875_{10} = 0.10111_2$

因此，$175.71875_{10} = 10101111.10111_2$

在熟悉 2 的整幂次情况下，可将十进制数写成按二进制数权的大小展开的多项式，按权值从高到低依次取各项的系数就可得到相应的二进制数。

$$(175.71875)_{10} = 2^7+2^5+2^3+2^2+2^1+2^0+2^{-1}+2^{-3}+2^{-4}+2^{-5}$$
$$= 10101111.10111_2$$

二进制数转换成十进制数的方法是：将二进制数的每一位数乘以它的权，然后相加，即可求得对应的十进制数值。

【例 1-2】 把二进制数 100110.101 转换成相应的十进制数。

$$(100110.101)_2 = 1\times2^5+0\times2^4+0\times2^3+1\times2^2+1\times2^1+0\times2^0+1\times2^{-1}+0\times2^{-2}+1\times2^{-3}$$
$$= 32+0+0+4+2+0+0.5+0+0.125$$
$$= 38.625$$

2）八进制计数法与十进制、二进制计数法的相互转换

八进制计数制的基本符号为 0，1，2，…，7。

十进制数转换为八进制数的方法是：对于十进制整数采用“除 8 取余”的方法转换为八进制整数；对于十进制小数则采用“乘 8 取整”的方法转换为八进制小数。

二进制数转换成八进制数的方法是：从小数点起，每三位二进制位分成一组（不足 3 位时，在小数点左边时左边补 0，在小数点右边时右边补 0），然后写出每一组的等值八进制数，顺序排列起来就得到所要求的八进制数。

【例 1-3】 将二进制数 10101111.10111 转换为相应的八进制数。

$$10\,101111.10111_2 = 010\ 101\ 111.101\ 110_2 = 257.56_8$$

依照同样的思想，将一位八进制数用三位二进制数表示，就可以直接将八进制数转换成二进制数。

二进制、八进制数和十六进制数之间的对应关系如表 1-2 所示。

表 1-2　二进制、八进制和十六进制数之间的对应关系

二进制	八进制	二进制	十六进制	二进制	十六进制
000	0	0000	0	1000	8
001	1	0001	1	1001	9
010	2	0010	2	1010	A
011	3	0011	3	1011	B
100	4	0100	4	1100	C
101	5	0101	5	1101	D
110	6	0110	6	1110	E
111	7	0111	7	1111	F

3）十六进制计数法与十进制、二进制计数法的相互转换

在十六进制计数制中，r=16，基本符号为 0，1，2，…，9，A，B，…，F。

十进制数可以转换为十六进制数的方法是：十进制数的整数部分“除 16 取余”，十进制

数的小数部分“乘 16 取整”。

由于一位十六进制数可以用 4 位二进制数来表示，因此二进制数与十六进制数的相互转换就比较容易。二进制数转换成十六进制数的方法是：从小数点开始，每 4 位二进制数为一组（不足 4 位时，在小数点左边时左边补 0，在小数点右边时右边补 0），将每一组用相应的十六进制数符来表示，即可得到正确的十六进制数。

【例 1-4】 将二进制数 10101111.10111 转换为相应的十六进制数。

$(1010\ \ 1111.1011\ \ 1)_2 = AF.B8_{16}$

2．二进制运算规则

（1）加法：二进制加法的进位规则是“逢二进一”。

0+0=0　　1+0=1　　0+1=1　　1+1=0（有进位）

（2）减法：二进制减法的借位规则是“借一当二”。

0–0=0　　1–0=1　　1–1=0　　0–1=1（有借位）

（3）乘法：

0×0=0　　1×0=0　　0×1=0　　1×1=1

3．机器数和码制

各种数据在计算机中表示的形式称为机器数，其特点是采用二进制计数制，数的符号用 0、1 表示，小数点隐含表示而不占位置。机器数对应的实际数值称为数的真值。

对于带符号数，机器数的最高位是表示正、负的符号位，其余位则表示数值。若约定小数点的位置在机器数的最低数值位之后，则是纯整数；若约定小数点的位置在机器数的最高数值位之前（符号位之后），则是纯小数。无符号数是指全部二进制位均代表数值，没有符号位。

为了便于运算，带符号的机器数可采用原码、反码和补码、移码等不同的编码方法。

1）原码表示

数值 X 的原码记为$[X]_{原}$，如果机器字长为 n（即采用 n 个二进制位表示数据），则最高位是符号位，0 表示正号，1 表示负号，其余的 n–1 位表示数值的绝对值。数值零的原码表示有两种形式：$[+0]_{原}$=00000000，$[-0]_{原}$=10000000。

【例 1-5】 若机器字长 n 等于 8，则

$[+1]_{原}$=00000001　　$[-1]_{原}$=10000001

$[+127]_{原}$=01111111　　$[-127]_{原}$=11111111

$[+45]_{原}$=00101101　　$[-45]_{原}$=10101101

$[+0.5]_{原}$=0◊1000000　　$[-0.5]_{原}$=1◊1000000（其中◊是小数点的位置）

2）反码表示

数值 X 的反码记作$[X]_{反}$，如果机器字长为 n，则最高位是符号位，0 表示正号，1 表示负号，其余的 n–1 位表示数值。正数的反码与原码相同，负数的反码则是其绝对值按位求反。数值 0 的反码表示有两种形式：$[+0]_{反}$=00000000，$[-0]_{反}$=11111111。

【例 1-6】 若机器字长 n 等于 8，则

$[+1]_{反}$=00000001　　$[-1]_{反}$=11111110

$[+127]_{反}$=01111111　　$[-127]_{反}$=10000000

$[+45]_{反}$=00101101　　$[-45]_{反}$=11010010

$[+0.5]_{反}$=0◊1000000　　$[-0.5]_{反}$=1◊01111111（其中◊是小数点的位置）

3）补码表示

数值 X 的补码记作$[X]_{补}$，如果机器字长为 n，则最高位为符号位，0 表示正号，1 表示负号，其余的 n–1 位表示数值。正数的补码与其原码和反码相同，负数的补码则等于其反码的末尾加 1。在补码表示中，0 有唯一的编码：$[+0]_{补}$=00000000，$[-0]_{补}$=00000000。

【例 1-7】 若机器字长 n 等于 8，则

$[+1]_{补}$=00000001　　$[-1]_{补}$=11111111

$[+127]_{补}$=01111111　　$[-127]_{补}$=10000001

$[+45]_{补}$=00101101　　$[-45]_{补}$=11010011

$[+0.5]_{补}$=0◊1000000　　$[-0.5]_{补}$=1◊1000000（其中◊是小数点的位置）

相对于原码和反码表示，n 位补码表示法有一个例外，当符号位为 1 而数值位全部为 0 时，它表示整数-2^{n-1}，即此时符号位的 1 既表示负数又表示数值。

设计补码时，有意识地引用了模运算在数理上对符号位的处理，利用模的自动丢弃实现了符号位的自然处理。

用补码表示数时，由于符号位和数值部分一起编码，很难从码值形式直接判断真值的大小。例如，45>−45，而其补码 00101101 在形式上小于 11010011。

4）移码表示

移码表示法是在数 X 上增加一个偏移量来定义的，常用于表示浮点数中的阶码。如果机器字长为 n，在偏移量为 2^{n-1} 时，只要将补码的符号位取反便可获得相应的移码表示。偏移量也可以是其他值。采用移码表示时，码值大者对应的真值就大。

【例 1-8】 若机器字长 n 等于 8，则

$[+1]_{移}$=10000001　　$[-1]_{移}$=01111111

$[+127]_{移}$=11111111　　$[-127]_{移}$=00000001

$[+45]_{移}$=10101101　　$[-45]_{移}$=01010011

$[+0]_{移}$=10000000　　$[-0]_{移}$=10000000

4．定点数和浮点数

1）定点数

所谓定点数，就是表示数据时小数点的位置固定不变。小数点的位置通常有两种约定方式：定点整数（纯整数，小数点在最低有效数值位之后）和定点小数（纯小数，小数点在最高有效数值位之前）。

设机器字长为 n，各种码制表示下的带符号数的范围如表 1-3 所示。当机器字长为 n 时，定点数的补码和移码可表示 2^n 个数，而其原码和反码只能表示 2^n-1 个数（0 表示占用了两个编码），因此，定点数所能表示的数值范围比较小，运算中很容易因结果超出范围而溢出。

表 1-3　机器字长为 n 时各种码制表示的带符号数的范围

码　制	定 点 整 数	定 点 小 数
原码	$-\left(2^{n-1}-1\right) \sim +\left(2^{n-1}-1\right)$	$-\left(1-2^{-(n-1)}\right) \sim +\left(1-2^{-(n-1)}\right)$
反码	$-\left(2^{n-1}-1\right) \sim +\left(2^{n-1}-1\right)$	$-\left(1-2^{-(n-1)}\right) \sim +\left(1-2^{-(n-1)}\right)$
补码	$-2^{n-1} \sim +\left(2^{n-1}-1\right)$	$-1 \sim +\left(1-2^{-(n-1)}\right)$
移码	$-2^{n-1} \sim +\left(2^{n-1}-1\right)$	$-1 \sim +\left(1-2^{-(n-1)}\right)$

2）浮点数

浮点数是小数点位置不固定的数，浮点表示法能表示更大范围的数。

在十进制中，一个实数可以写成多种表示形式。例如，83.125 可写成 $10^3\times0.083125$ 或 $10^4\times0.0083125$ 等。同理，一个二进制数也可以写成多种表示形式。例如，二进制数 1011.10101 可以写成 $2^4\times0.101110101$、$2^5\times0.0101110101$ 或 $2^6\times0.00101110101$ 等。

一个含小数点的二进制数 N 可以表示为更一般的形式：

$$N=2^E\times F$$

其中，E 称为阶码，F 为尾数，这种表示数的方法称为浮点表示法。

在浮点表示法中，阶码通常为带符号的纯整数，尾数为带符号的纯小数。浮点数的表示格式一般如下：

阶符	阶码	数符	尾数

很明显，一个数的浮点表示不是唯一的。当小数点的位置改变时，阶码也相应改变，因此可以用多种浮点形式表示同一个数。

浮点数所能表示的数值范围主要由阶码决定，所表示数值的精度则由尾数决定。

为了提高数据的表示精度，当尾数的值不为 0 时，规定尾数域的最高有效位应为 1，这称为浮点数的规格化表示，否则修改阶码同时左移或右移小数点的位置，使其变为规格化数的形式。

简单来说，规格化就是将尾数的绝对值限定在区间[0.5, 1）。

（1）若尾数 $F \geqslant 0$，则其规格化的尾数形式为 F=01×××…×，其中×可为 0，也可为 1，即将尾数 F 的范围限定在区间[0.5, 1）。

（2）若尾数 $F<0$，则其规格化的尾数形式为 F=10×××…×，其中×可为 0，也可为 1，即将尾数 F 的范围限定在区间（–1, –0.5]。

3）工业标准 IEEE 754

IEEE 754 是由 IEEE 制定的有关浮点数的工业标准，被广泛采用。该标准的表示形式如下：

S	P	M

其中，S 为数的符号位，为 0 时表示正数，为 1 时表示负数；P 为指数（阶码），用移码表示（偏移值为 $2^{p-1}-1$，p 为阶码的位数）；M 为尾数，用原码表示。

对于阶码为 0 或 255 的情况，IEEE 754 标准有特别的规定：若 P 为 0 且 M 为 0，则表示真值±0（正负号和数符位有关）。如果 P = 255 并且 M 是 0，则这个数的真值为±∞（与符号位有关）；如果 P = 255 并且 M 不是 0，则这不是一个数（NaN）。

目前，计算机中主要使用 3 种形式的 IEEE 754 浮点数，如表 1-4 所示。

表 1-4　3 种形式的 IEEE 754 浮点数

参　　数	单精度浮点数	双精度浮点数	扩充精度浮点数
浮点数字长	32	64	80
尾数长度	23	52	64
符号位长度	1	1	1
阶码长度	8	11	15
指数偏移量	+127	+1023	+16 383
可表示的实数范围	$10^{-38} \sim 10^{38}$	$10^{-308} \sim 10^{308}$	$10^{-4932} \sim 10^{4932}$

在 IEEE 754 标准中，对于单精度浮点数和双精度浮点数，约定小数点左边隐含有一位，通常这位数就是 1，因此尾数为 1.××…×。

【例 1-9】 利用 IEEE 754 标准将数 176.0625 表示为单精度浮点数。

解：首先将该十进制数转换成二进制数，即 $(176.0625)_{10}=(10110000.0001)_2$，其次对二进

制数进行规格化处理，即 10110000.0001=1◊01100000001×2^7。这就保证了最高位为 1，而且小数点应当在◊位置上，将最高位去掉并扩展为单精度浮点数所规定的 23 位尾数，得到01100000001000000000000。

然后求阶码，上述表示中指数为 7，用移码表示为 10000110（偏移量是 127，因此偏移后的指数值为 7＋127=134）。

最后，得到$(176.0625)_{10}$的单精度浮点数表示形式：

0 10000110 01100000001000000000000

5．十进制数与字符的编码表示

数值、文字和英文字母等都被认为是字符，任何字符被录入计算机后，都必须转换成二进制表示形式，称为字符编码。

用 4 位二进制代码表示一位十进制数，称为二-十进制编码，简称 BCD 编码。因为 2^4=16，而十进制数只有 0～9 这 10 个不同的数符，故有多种 BCD 编码。根据 4 位代码中每一位是否有确定的权来划分，可分为有权码和无权码两类。

应用最多的有权码是 8421 码，即 4 个二进制位的权从高到低分别为 8、4、2 和 1。无权码中常用余 3 码和格雷码。余 3 码是在 8421 码的基础上，把每个数的代码加上 0011 后构成的。格雷码的编码规则是相邻的两个代码之间只有一位不同。

常用的 8421BCD 码、余 3 码、格雷码与十进制数的对应关系如表 1-5 所示。

表 1-5　8421BCD 码、余 3 码、格雷码与十进制数的对应关系

十 进 制 数	8421BCD 码	余 3BCD 码	格　雷　码
0	0000	0011	0000
1	0001	0100	0001
2	0010	0101	0011
3	0011	0110	0010
4	0100	0111	0110
5	0101	1000	1110
6	0110	1001	1010
7	0111	1010	1000
8	1000	1011	1100
9	1001	1100	0100

6．ASCII 码

ASCII 码（American Standard Code for Information Interchange，美国标准信息交换代码）已被国际标准化组织 ISO 采纳，成为一种国际通用的信息交换标准代码。基本的 ASCII 码采用 7 二进制位，即 $d_6d_5d_4d_3d_2d_1d_0$ 对字符进行编码：低 4 位组 $d_3d_2d_1d_0$ 用作行编码，高 3 位组 $d_6d_5d_4$ 用作列编码。基本的 ASCII 字符代码表如表 1-6 所示。

表 1-6　7 位 ASCII 代码表

$d_3d_2d_1d_0$ 位（低 4 位）	$d_6d_5d_4$ 位（高 3 位）							
	000	**001**	**010**	**011**	**100**	**101**	**110**	**111**
0000	NUL	DLE	SP	0	@	P	、	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	,	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	'	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	↑	n	~
1111	DI	US	/	?	O	↓	o	Del

根据 ASCII 码的构成格式，可以很方便地从对应的代码表中查出每一个字符的编码。例如，字符 0 的 ASCII 码值为 0110000（$2^5+2^4=48$），字符 a 的 ASCII 码值为 1100001（$2^6+2^5+2^0=97$）。

7．汉字编码

在计算机中处理汉字，必须先将汉字代码化，即对汉字进行编码。汉字处理包括汉字的编码输入、汉字的存储和汉字的输出等环节。

西文是拼音文字，基本符号比较少，比较容易编码，在计算机系统中输入、内部处理、存

第 2 章　操作系统基础知识

2.1　操作系统概述

操作系统(Operating System，OS)是计算机系统中必不可少的核心系统软件，其他软件（如编辑程序、汇编程序、编译程序、数据库管理系统等系统软件，以及大量的应用软件）是建立在操作系统的基础上，并在操作系统的统一管理和支持下运行。操作系统是用户与计算机之间的接口，用户可以通过操作系统提供的功能访问计算机系统中的软硬件资源。

1．操作系统的作用、特征与功能

操作系统有效地组织和管理系统中的各种软、硬件资源，合理地组织计算机系统工作流程，控制程序的执行，并且向用户提供一个良好的工作环境和友好的接口。

操作系统的 4 个特征是并发性、共享性、虚拟性和不确定性。从传统的资源管理的观点来看，操作系统的功能可分为五大部分：进程管理、文件管理、存储管理、设备管理和作业管理。

（1）进程管理。实质上是对处理机的执行“时间”进行管理，采用多道程序等技术将 CPU 的时间合理地分配给每个任务。主要包括进程控制、进程同步、进程通信和进程调度。

（2）文件管理。主要包括文件存储空间管理、目录管理、文件的读写管理和存取控制。

（3）存储管理。是对主存储器“空间”进行管理，主要包括存储分配与回收、存储保护、地址映射（变换）和主存扩充。

（4）设备管理。实质是对硬件设备的管理，包括对输入输出设备的分配、启动、完成和回收。

（5）作业管理。包括任务、界面管理、人机交互、图形界面、语音控制和虚拟现实等。

操作系统提供系统命令级的接口，供用户组织和控制自己的作业运行。操作系统还提供编程一级接口，供用户程序和系统程序调用操作系统功能。

2．操作系统分类及特点

操作系统分为批处理操作系统、分时操作系统、实时操作系统、网络操作系统、分布式操作系统、微机操作系统和嵌入式操作系统等。

1）批处理操作系统

批处理操作系统分为单道批处理和多道批处理。

单道批处理操作系统是一种早期的操作系统，“单道”的含义是指一次只有一个作业装入内存执行。作业由用户程序、数据和作业说明书（作业控制语言）三部分组成。当一个作业运行结束后，随即自动调入同批的下一个作业，从而节省了作业之间的人工干预时间，提高了资源的利用率。

多道批处理操作系统允许多个作业装入内存执行，在任意一个时刻，作业都处于开始点和终止点之间。每当运行中的一个作业由于输入/输出操作需要调用外部设备时，就把 CPU 交给另一道等待运行的作业，从而将主机与外部设备的工作由串行改变为并行，进一步避免了因主机等待外设完成任务而浪费宝贵的 CPU 时间。多道批处理系统主要有 3 个特点：多道、宏观上并行运行、微观上串行运行。

2）分时操作系统

在分时操作系统中，一个计算机系统与多个终端设备连接。分时操作系统是将 CPU 的工作时间划分为许多很短的时间片，轮流为各个终端的用户服务。例如，一个带 20 个终端的分时系统，若每个用户每次分配一个 50 ms 的时间片，则每隔 1s 即可为所有的用户服务一遍。因此，尽管各个终端上的作业是断续地运行的，但由于操作系统每次对用户程序都能做出及时的响应，因此用户感觉整个系统均归其一人占用。

分时系统主要有 4 个特点：多路性、独立性、交互性和及时性。

3）实时操作系统

实时是指计算机对于外来信息能够以足够快的速度进行处理，并在被控对象允许的时间范围内做出快速反应。实时系统对交互能力要求不高，但要求可靠性有保障。为了提高系统的响应时间，对随机发生的外部事件应及时做出响应并对其进行处理。

实时系统分为实时控制系统和实时信息处理系统。实时控制系统主要用于生产过程的自动控制，如数据自动采集、武器控制、火炮自动控制、飞机自动驾驶和导弹的制导系统等。实时信息处理系统主要用于实时信息处理，如飞机订票系统、情报检索系统等。

实时系统与分时系统除了应用的环境不同，主要有以下三点区别。

（1）系统的设计目标不同。分时系统是设计成一个多用户的通用系统，交互能力强；而实时系统大都是专用系统。

（2）交互性的强弱不同。分时系统是多用户的通用系统，交互能力强；而实时系统是专用系统，仅允许操作并访问有限的专用程序，不能随便修改，且交互能力差。

（3）响应时间的敏感程度不同。分时系统是以用户能接收的等待时间为系统的设计依据，而实时系统是以被测物体所能接受的延迟为系统设计依据。因此，实时系统对响应时间的敏感程度更强。

4）网络操作系统

网络操作系统是使联网计算机能方便而有效地共享网络资源，为网络用户提供各种服务的软件和有关协议的集合。因此，网络操作系统的功能主要包括高效、可靠的网络通信；对网络中共享资源（在 LAN 中有硬盘、打印机等）的有效管理；提供电子邮件、文件传输、共享硬盘和打印机等服务；网络安全管理；提供互操作能力。

主要的网络操作系统有 UNIX、Linux 和各种版本的 Windows Server 系统。

5）分布式操作系统

分布式计算机系统是由多个分散的计算机经连接而成的计算机系统，系统中的计算机无主、次之分，任意两台计算机可以通过通信交换信息。通常，为分布式计算机系统配置的操作系统称为分布式操作系统。

分布式操作系统能直接对系统中各类资源进行动态分配和调度、任务划分、信息传输协调工作，并为用户提供一个统一的界面，标准的接口，用户通过这一界面实现所需要的操作和使用系统资源，使系统中若干台计算机相互协作完成共同的任务，有效控制和协调诸任务的并行执行，并向系统提供统一、有效的接口的软件集合。

分布式操作系统是网络操作系统的更高级形式，它保持网络系统所拥有的全部功能，同时又有透明性、可靠性和高性能等特性。

6）微型计算机操作系统

微型计算机操作系统简称微机操作系统，常用的有 Windows、Mac OS 和 Linux。Windows 操作系统是 Microsoft 公司开发的图形用户界面、多任务、多线程操作系统。Mac OS 操作系统是美国苹果计算机公司为其 Macintosh 计算机设计的操作系统。Linux 是一套免费使用并可自由传播的类 UNIX 操作系统，由世界各地成千上万的程序员设计和实现，其目的是建立不受任何商品化软件版权制约的、全世界都能自由使用的 UNIX 兼容产品。

7）嵌入式操作系统

嵌入式操作系统运行在嵌入式智能芯片环境中，对整个智能芯片以及它所操作、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。其主要特点：

（1）微型化。从性能和成本角度考虑，希望占用资源和系统代码量少，如内存少、字长短、运行速度有限、能源少（用微小型电池）。

（2）可定制。从减少成本和缩短研发周期考虑，要求嵌入式操作系统能运行在不同的微处理器平台上，能针对硬件变化进行结构与功能上的配置，以满足不同应用需要。

（3）实时性。嵌入式操作系统主要应用于过程控制、数据采集、传输通信、多媒体信息及关键要害领域需要迅速响应的场合，所以对实时性要求高。

（4）可靠性。系统构件、模块和体系结构必须达到应有的可靠性，对关键要害应用还要提

供容错和防故障措施。

（5）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。

嵌入式实时操作系统有很多，常见的有 VxWorks、μClinux、PalmOS、WindowsCE、μC/OS-II 和 eCos 等。

促使操作系统发展的因素主要有 3 个方面：第一，硬件的不断地升级与新的硬件产品出现，需要操作系统提供更多更复杂的支持；第二，新的服务需求，操作系统为了满足系统管理者和用户需求，需要不断扩大服务范围；第三，修补操作系统自身的错误，操作系统在运行的过程中其自身的错误也会不断地被发现，因此需要不断地修补操作系统自身的错误（即所谓的“补丁”）。需要说明的是，在修补的过程中也可能会产生新的错误。

2.2　进程管理

进程管理也称为处理机管理，其核心是如何合理地分配处理机的时间，提高系统的效率。在计算机系统中有多个并发执行的程序，采用“程序”这个静态的概念已经不能描述程序执行时动态变化的过程，所以引入了“进程”。

2.2.1　基本概念

1．程序执行时的特征

前趋图是一个有向无循环图，由结点和有向边组成，结点代表各程序段的操作，而结点间的有向边表示两个程序段操作之间存在的前趋关系（“→”）。程序段 P_i 和 P_j 的前趋关系表示成 $P_i \rightarrow P_j$，其中 P_i 是 P_j 的前趋，P_j 是 P_i 的后继，其含义是 P_i 执行结束后 P_j 才能执行。例如，图 2-1 为一个包含 3 个程序段的前驱图，其中输入是计算的前驱，计算是输出的前驱。

图 2-1　3 个程序段的前驱图

程序顺序执行时的主要特征有顺序性、封闭性和可再现性。其中，顺序性是指程序的各程序段严格按照规定的顺序执行；封闭性是指程序运行时系统内的资源只受该程序控制而改变，执行结果不受外界因素的影响；可再现性是指只要程序执行环境和初始条件相同，程序多次执行的结果相同。

若在计算机系统中采用多道程序设计技术，则主存中的多道程序可处于并发执行状态。对于图 2-1 中有 3 个程序段的作业，虽然其中有前趋关系的各程序段不能在 CPU 和输入输出各部件上并行执行，但是同一个作业内没有前趋关系的程序段或不同作业的程序段可以分别在 CPU 和各输入输出部件上并行执行。

例如，某计算机系统中有一个 CPU、一台输入设备和一台输出设备，每个作业具有 3 个程序段：输入 I_i、计算 C_i 和输出 P_i（i=1,2,3）。图 2-2 为 3 个作业的各程序段并发执行的前驱图。从图 2-2 中可以看出，I_2 与 C_1 并行执行，I_3、C_2 与 P_1 并行执行，C_3 与 P_2 并行执行。其中，I_2、I_3 受到 I_1 的间接制约，C_2、C_3 受到 C_1 的间接制约，P_2、P_3 受到 P_1 的间接制约；而 C_1、P_1 受到 I_1 的直接制约，C_2、P_2 受到 I_2 的直接制约，C_3、P_3 受到 I_3 的直接制约。

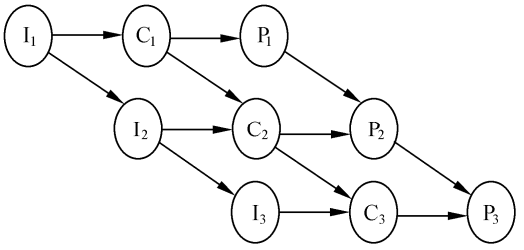

图 2-2　各程序段并发执行的前驱图

程序并发执行时的主要特征如下。

（1）失去了程序的封闭性。

（2）程序和机器执行程序的活动不再一一对应。

（3）并发程序间具有相互制约性。

2．进程的组成

进程（Process）是程序的一次执行。进程通常由程序、数据和进程控制块（Process Control Block，PCB）组成。其中，PCB 是进程存在的唯一标志，其主要内容如表 2-1 所示。

表 2-1　PCB 的内容

信　息	含　义
进程标识符	标明系统中的各个进程
状态	说明进程当前的状态
位置信息	指明程序及数据在内存或外存的物理位置
控制信息	参数、信号量和消息等
队列指针	链接同一状态的进程
优先级	进程调度的依据
现场保护区	将处理机的现场保护到该区域以便再次调度时能继续正确运行

进程的程序部分描述了进程需要完成的功能。假设一个程序能被多个进程同时共享执行，那么这部分就应该以可再入码的形式编制，它是程序执行时不可修改的部分。进程的数据部分包括程序执行时所需的数据及工作区，这部分只能为一个进程所专用，是进程的可修改部分。

3．进程的状态及其状态间的切换

1）三态模型

在多道程序系统中，进程的运行是走走停停，在处理器上交替运行，状态也不断地发生变

化，因此进程一般有 3 种基本状态：运行、就绪和阻塞，如图 2-3 所示，也称三态模型。

- 运行：当一个进程在处理机上运行时，称该进程处于运行状态。显然，对于单处理机系统，处于运行状态的进程只有一个。
- 就绪：一个进程获得了除处理机外的一切所需资源，一旦得到处理机即可运行，则称此进程处于就绪状态。
- 阻塞：也称等待或睡眠状态，一个进程正在等待某一事件发生（例如，请求 I/O 而等待 I/O 完成等）而暂时停止运行，这时即使把处理机分配给该进程，它也无法运行，故称该进程处于阻塞状态。

2）五态模型

事实上，对于一个实际的系统，进程的状态及其转换将更复杂。例如，引入新建态和终止态构成了五态模型，如图 2-4 所示。

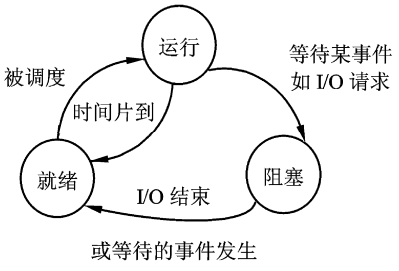

图 2-3　进程的三态模型

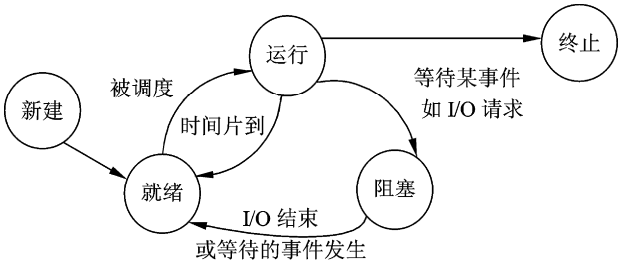

图 2-4　进程的五态模型

图 2-4 中，新建态对应于进程刚刚被创建且没有被提交的状态，并等待系统完成创建进程的所有必要信息。因为创建进程时分为两个阶段，第一个阶段为一个新进程创建必要的管理信息，第二个阶段让该进程进入就绪状态。由于有了新建态可以增加调度的灵活性，即操作系统可以根据系统的性能和内存容量的限制推迟新建态进程的提交。

类似地，进程的终止也可分为两个阶段，第一个阶段等待操作系统进行善后处理，第二个阶段释放内存。设置终止态的目的是防止系统进行善后处理时引起资源分配不当等问题。

2.2.2　进程控制

进程控制是指对系统中所有进程从创建到消亡的全过程实施有效的控制。在操作系统中通过设置一套控制机构对进程实施控制，其主要功能包括创建一个新进程，撤销一个已经运行完的进程，改变进程的状态，实现进程间的通信。进程控制是由操作系统内核（Kernel）中的原语实现的。

原语（Primitive）是指由若干条机器指令组成的、用于完成特定功能的程序段。原语的特点是在执行时不能被分割，即原子操作要么都做，要么都不做。内核中所包含的原语主要有进程控制原语、进程通信原语、资源管理原语以及其他原语。属于进程控制方面的原语有进程创建原语、进程撤销原语、进程挂起原语、进程激活原语、进程阻塞原语以及进程唤醒原语等。不同的操作系统，内核所包含的功能不同，但大多数操作系统的内核都包含支撑功能和资源管理的功能。

2.2.3 进程通信

在多道程序环境的系统中，存在多个可并发执行的进程，因此进程间必然存在资源共享和相互合作的问题。进程通信是指各个进程交换信息的过程。

1．同步与互斥

通常，同步是合作进程间的直接制约问题，互斥是申请临界资源进程间的间接制约问题。

1）进程间的同步

多个并发执行的进程都以各自独立的、不可预知的速度向前推进，但是有时需要在某些确定点上协调相互合作进程间的工作。例如，进程 A 向缓冲区送数据，进程 B 从缓冲区取数据加工，当进程 B 要取数据加工时，必须是进程 A 完成了向缓冲区送数据的操作，否则进程 B 必须停下来等待进程 A 的操作结束。可见，进程间的同步是指进程间完成一项任务时直接发生相互作用的关系。

2）进程间的互斥

在多道程序系统环境中，各进程可以共享各类资源，但有些资源一次只能供一个进程使用，称为临界资源（Critical Resource，CR），如打印机、共享变量等。进程间的互斥是指系统中各进程互斥使用临界资源。

3）临界区管理的原则

临界区（Critical Section，CS）是进程中对临界资源实施操作的那段程序。对互斥临界区管理的 4 条原则如下。

- 有空即进。当无进程处于临界区时，允许进程进入临界区，并且只能在临界区运行有限的时间。
- 无空则等。当有一个进程在临界区时，其他需要进入临界区的进程必须等待，以保证进程互斥地访问临界资源。
- 有限等待。对要求访问临界资源的进程，应保证进程等待有限时间后进入临界区，以免陷入“饥饿”状态。

- 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，以免进程陷入“忙等”状态。

2．信号量机制

信号量机制是荷兰学者 Dijkstra 于 1965 年提出，该机制是一种有效的进程同步与互斥工具。目前信号量机制有了很大的发展，主要有整型信号量、记录型信号量和信号量集机制。本章只介绍整型信号量。

1）整型信号量与 PV 操作

信号量是一个整型变量，根据控制对象的不同被赋予不同的值。信号量分为如下两类。

- 公用信号量。实现进程间的互斥，初值为 1 或资源的数目。
- 私用信号量。实现进程间的同步，初值为 0 或某个正整数。

信号量 S 的物理意义：若 S≥0，表示某资源的可用数；若 S<0，则其绝对值表示阻塞队列中等待该资源的进程数。

对系统中的每个进程，其工作的正确与否不仅取决于它自身的正确性，而且与它在执行中能否与其他相关进程正确地实施同步互斥有关。

PV 操作是实现进程同步与互斥的常用方法。P 操作和 V 操作是低级通信原语，在执行期间不可分割。其中，P 操作表示申请一个资源，V 操作表示释放一个资源。

P 操作的定义：S:=S–1，若 S≥0，则执行 P 操作的进程继续执行；若 S<0，则置该进程为阻塞状态（因为无可用资源），并将其插入阻塞队列。P 操作可用如下过程表示，其中 Semaphore 表示所定义的变量是信号量。

```
Procedure P(Var S:Semaphore);
        Begin
          S:=S–1;
          If S<0 then W(S)          {执行 P 操作的进程插入等待队列}
        End;
```

V 操作的定义：S:=S+1，若 S>0，则执行 V 操作的进程继续执行；若 S≤0，则从阻塞状态唤醒一个进程，并将其插入就绪队列，然后执行 V 操作的进程继续。V 操作可用如下过程表示。

```
Procedure V(Var S:Semaphore);
        Begin
          S:=S+1;
          If S≤0 then R(S)          {从阻塞队列中唤醒一个进程}
        End;
```

2）利用 PV 操作实现进程的互斥

令信号量 mutex 的初值为 1，进入临界区时执行 P 操作，退出临界区时执行 V 操作。这样，进入临界区的代码段如下：

```
P(mutex)
  临界区
V(mutex)
```

【例 2-1】 两个并发执行的程序段完成交通流量的统计，其中“观察者”P1 识别通过的车辆数，“报告者”P2 定时将观察者的计数值清“0”。用 PV 操作实现的交通流量统计程序如下：

```
        P1                                  P2
L1: if 有车通过 then                  L2:begin
      begin                                 P(mutex);
        P(mutex);                           PRINT COUNT;
        COUNT:=COUNT+1;                     COUNT:=0;
        V(mutex);                           V(mutex);
      end                                 end
      GOTO L1;                            GOTO L2;
```

3）利用 PV 操作实现进程的同步

进程的同步是由于进程间的合作而引起的相互制约问题。实现进程同步的一种方法是将一个信号量与消息相联系，当信号量的值为 0 时表示希望的消息未产生，否则表示希望的消息已经来到。假定用信号量 S 表示某条消息，进程可以通过调用 P 操作测试消息是否到达，调用 V 操作通知消息已准备好。典型的应用是单缓冲区的生产者和消费者同步问题。

【例 2-2】 生产者进程 P1 不断地生产产品送入缓冲区，缓冲区可以存放一件产品。消费者进程 P2 不断地从缓冲区中取出产品消费。利用 PV 操作实现进程 P1 与 P2 之间的同步问题，需要设置几个信号量，信号量初值为多少？

本题需要设置两个信号量 S1 和 S2。其中：S1 表示缓冲区是否空闲，初值=1，表示缓冲区空可以将产品送入缓冲区；S2 表示缓冲区有无产品，初值=0（初始时缓冲区是无产品的）。

P1 和 P2 的同步过程如图 2-5 所示。

【例 2-3】 假设有一个生产者和一个消费者，缓冲区可存放 n 件产品。生产者不断地生产产品，消费者不断地消费产品。如何用 PV 操作实现生产者和消费者的同步？

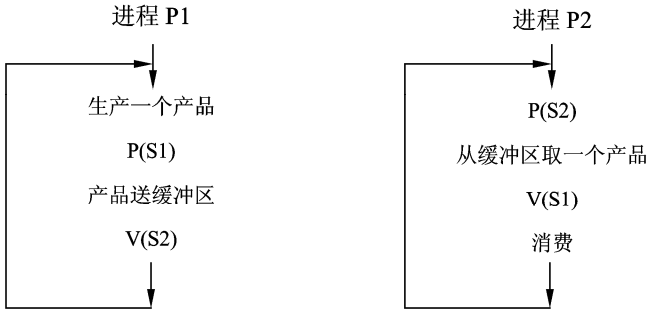

图 2-5　单缓冲区的同步控制

本题可以通过设置 3 个信号量 S、S1 和 S2，用 PV 操作实现生产者和消费者的同步。其中，S 是一个互斥信号量，初值为 1，因为缓冲区是一个互斥资源，所以需要进行互斥控制；S1 表示缓冲区中空闲单元数（大于 0 表示可以将产品放入），初值为 n；S2 表示缓冲区的产品数，初值为 0，其同步过程如图 2-6 所示。

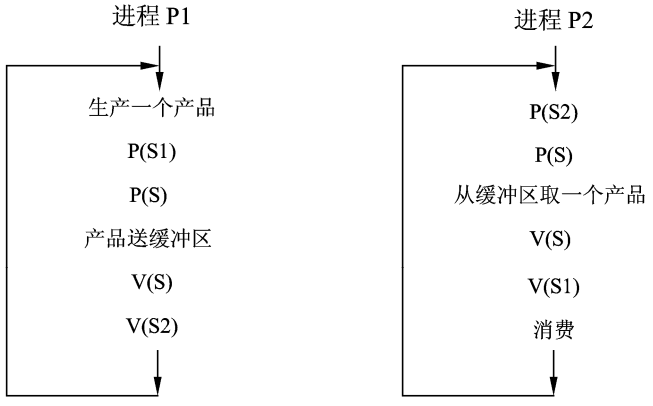

图 2-6　n 个缓冲区的同步控制

3．高级通信

根据进程间交换信息量的多少和效率的高低，进程通信的方式分为低级方式和高级方式。PV 操作属于低级通信方式，若用 PV 操作实现进程间通信，则存在如下问题。

（1）编程难度大，通信对用户不透明，即需要用户利用低级通信工具实现进程间的同步与互斥，而且，PV 操作使用不当还容易引起死锁。

（2）效率低，生产者每次只能向缓冲区放一个消息，消费者只能从缓冲区取一个消息。

为了提高通信效率，能传递大量数据，减轻程序的复杂度，系统引入了高级通信方式。高级通信方式主要分为共享存储模式、消息传递模式和管道通信。

（1）共享存储模式。相互通信的进程共享某些数据结构（或存储区），实现进程之间的通信。

（2）消息传递模式。进程间的数据交换以消息为单位，程序员直接利用系统提供的一组通信命令（原语）来实现通信。如 Send(A)、Receive(A)。

（3）管道通信。管道是用于连接一个读进程和一个写进程，以实现它们之间通信的共享文件（pipe 文件也称管道文件）。向管道提供输入的发送进程（即写进程），以字符流的形式将大量的数据送入管道；而接收进程可从管道接收大量的数据。

4．直接和间接通信

直接通信是将消息直接发送给指定进程，因此，Send 和 Receive 原语中应指出进程名字。其调用格式如下：

```
Send(Who,Message)        发送消息给指定进程或一组进程
Receive(Who,Message)     从约定进程接收消息
```

间接通信是以信箱为媒体来实现通信的，接收信件的进程只需设立一个信箱，若干个进程都可以向同一个进程发送信件。因此，Send 和 Receive 原语中应给出信箱名。其调用格式如下：

```
Send(N,M)                将信件 M 发送到信箱 N 中
Receive(N,X)             从信箱 N 中取一封信存入 X
```

有些系统还提供带标记的发送，用 Tag 可以指定进程是否要等到接收进程取到信息再继续运行。一般接收进程总是要等待消息到达后才继续运行。其调用格式如下：

```
Send(Who,Message,tag)
```

2.2.4 进程调度

1．三级调度

在某些操作系统中，一个作业从提交到完成需要经历高、中、低三级调度。

（1）高级调度。又称为“长调度”“作业调度”“接纳调度”，它决定处于输入池中的哪个后备作业可以调入主系统做好运行的准备，成为一个或一组就绪进程。系统中一个作业只需经过一次高级调度。

（2）中级调度。又称为“中程调度”“对换调度”，它决定处于交换区中的就绪进程哪个可