C菜单设计与实现

孙劲光　单亚飞　编著

东北大学出版社

前　言

当今的大学校园里，绝大多数以C程序设计作为大学生接受计算机文化教育的基础。在有限的课时内，他们学习程序设计的基本思想和方法，并具备了一定的解决问题的能力和编程技巧。但如何运用所学的知识作进一步的设计与开发呢？学生们为此深感困惑。

在这本书中，首先从学习Turbo C的学生们最熟悉的Turbo C的集成环境的界面入手，介绍Turbo C的编程模式、BIOS功能调用、DOS系统功能调用、外设接口和设备的操作控制及字符屏幕设计，完成文本模式下界面的设计。其次，当使用众多图形处理软件，却无法生成所需要的真正高级的图形信息时，我们通过Turbo C的图形屏幕的设计带领大家投入到编写绘制图形的软件设计中去，让学生体会到他们完全有可能编制出优秀的绘图程序及软件。为学生们学习面向对象的基本思想，并通过C++完成图形界限的设计，达到掌握面向对象程序设计与开发的技术，奠定一个良好的基础。我们希望通过上述“三步曲”，能够满足莘莘学子们孜孜以求的心愿。

今天，我们非常高兴地将多年来通过第二课堂与大学生们共同学习与交流的结晶汇集成册，真诚地希望在激发大家提高应用计算机的兴趣的同时，能够使大家的计算机应用水平上一个新的台阶。

编著者

2001年12月

目　录

第 1 章
计算机基础知识

1.1 计算机系统概述

计算机系统包括硬件和软件两部分。硬件包括电路、插件板、机箱等；软件则是为了运行、管理和维护计算机而编制的各种程序的总和。

1.1.1 硬 件

典型的计算机结构如图 1-1 所示。

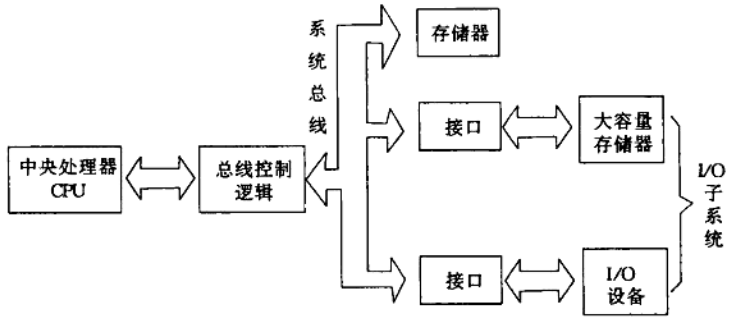


图 1-1 计算机结构

中央处理器 CPU（Central Processing Unit）包括运算器和控制器两部分。运算器执行所有的算术和逻辑运算指令；控制器则负责全机的控制工作，它负责把指令逐条从存储器中取出，经译码分析后向全机发出取数、执行、存数等控制命令，以保证计算机正确地完成程序所要求的功能。

存储器（Memory）是计算机的记忆部件。人们编写的程序（由指令序列组成）存放在这里后才被计算机执行，它也用来存放程序中所用的数据、信息及中间结果。

I/O（Input/Output）子系统一般包括 I/O 设备及大容量外存储器两类外部设备。

系统总线用来传送各种信息，它把 CPU、存储器和 I/O 设备连接起来。系统总线包括数据线、地址线和控制线三种。控制线规定总线的动作，数据线传送信息，地址线指出

信息的来源和目的地。系统总线的工作由总线控制逻辑负责指挥。

1.1.2 软 件

计算机软件是计算机系统的重要组成部分，它由系统软件和应用软件两大类组成。系统软件是用户使用计算机时所必需的；应用软件则是用户自行编制的各种程序。图 1-2 表示了计算机软件层次的结构。

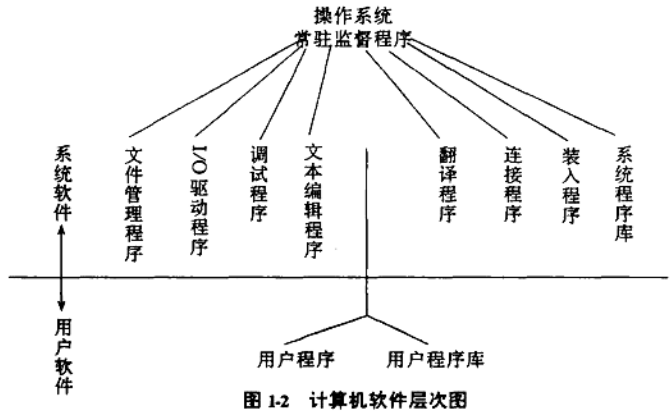


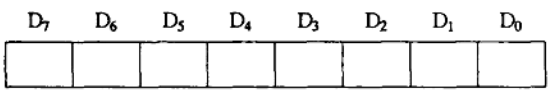
图 1-2 计算机软件层次图

1.2 存储器

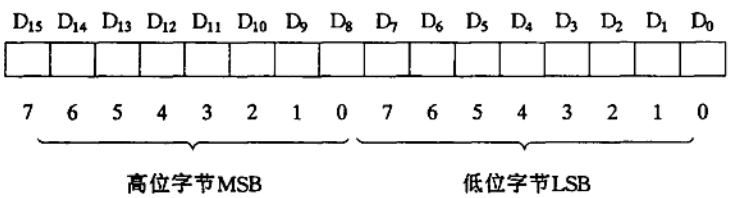
1.2.1 地址及内容

计算机存储信息的基本单位是一个二进制位（Bit），一位可存储一个二进制数：0 和 1。

8 位的二进制组成一个字节，其每位上的编号为：



以 IBM PC 机为例，其字长为 16 位，由 2 个字节组成，其每位的编号为：



为了正确地存放或取得内存中的信息，存储器中的每一个字节均分配了一个编号，该编号被称为地址。在机器内部，地址用二进制无符号整数表示，编制地址从 0 开始编号，顺序地每次加 1，书写时为方便起见，常常采用十六进制格式。

每个字节单元有一个二进制数表示地址，那么 16 位二进制数可以表示的字节单元为

216个，范围为 0～65535，所表示的总量为 64KB，使用十六进制编号则为 0000H～FFFFH。

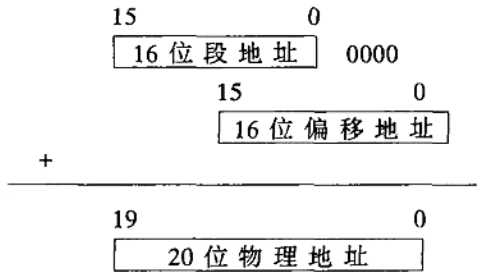
大部分数据都是以字为单位存储的，当一个字存入存储器时，占据了相邻的两个字节，其中低位字节存入地址低的单元，高位字节存入高地址单元，一个字单元的地址采用低地址来表示，且字单元的地址均以偶数地址开始。

1.2.2　存储器地址分段

在计算机的存储空间中，每个存储单元都有一个惟一的地址来表示。CPU 在访问存储器时，首先确定每个单元的物理地址后，才从该单元中取出数据。

由于一个字长可访问的最大存储空间为 64KB，而 IBM PC 机的最大存储容量为 1MB，所以，若要访问 1MB 字节空间的存储器，必须将 20 位地址用 16 进制表示出 1MB 的地址范围 00000H～FFFFFH。因此在 IBM PC 机里采用了存储器地址分段的方法。

程序员在编制程序时，从 0 地址开始，按每 16 个字节分为一小段，每个段内字节数最大可达 64KB，这样段内地址可以用 16 位二进制数来表示，并将其作为 20 位地址中的后 16 位；每个段的起始地址即段地址，由于它的低 4 位一定都是 0，因此将其向左偏移 4 位后，就使得段地址只有高 16 位值；每段的段内地址称为偏移地址。这样，物理地址的计算方法可以表示为：



即：把段地址左移 4 位再加上偏移地址值就形成了物理地址。或写成

16d×段地址＋偏移地址＝物理地址

1.3　中央处理器

中央处理器 CPU 由运算器和控制器两部分组成，它的任务是执行存放在存储器中的指令序列。在 IBM PC 机中它就是一个微处理芯片 8088。它的组成见图 1-3。

算术逻辑部件 ALU（Arithmetic Logic Unit）用来进行算术运算和逻辑运算。

控制逻辑负责全机的控制工作，如总线控制、存储器取指令或数据、指令进行译码、发出执行指令的命令、结果存入存储器等。

寄存器在 CPU 中起着至关重要的作用，对它进行存取数据的速度比存储器要快得多。

数据寄存器（AX，BX，CX，DX）用来暂时存放计算过程中所用到的操作数、结果或其他信息。它们可以以字的形式进行访问，也可以以字节的形式进行访问。当以字节的

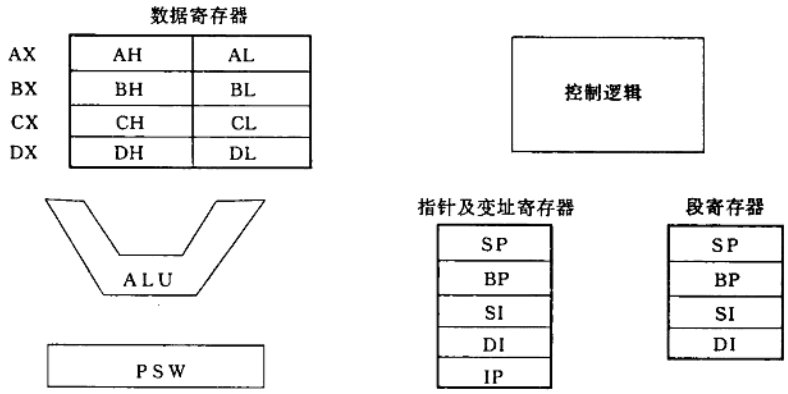


图 1-3 IBM PC 机的 CPU 组成

形式进行访问时，每个寄存器被分解为高位字节和低位字节（AH，AL，BH，BL，CH，CL，DH，DL）。IBM PC 机中的这四个数据寄存器都是通用寄存器，但习惯上它们又都具有着专用的情况。

AX（Accumulator）作为累加器及 I/O 指令的传送信息，所以它是算术运算的主要寄存器。

BX（Base）计算存储器地址经常作为基址寄存器。

CX（Count）用在循环和串处理中的隐含计数器。

DX（Data）在双字长运算时与 AX 组合存放一个双字长的数，AX 放低位字，DX 放高位字；在 I/O 操作中 DX 用来存放端口地址。

指针及变址寄存器（SP，BP，SI，DI）是四个 16 位寄存器，它们以字为单位使用时，既可以像数据寄存器一样在运算过程中存放操作数，同时它更经常用来在段内寻址时提供偏移地址。

段寄存器（CS，DS，SS，ES），每个寄存器可以确定一个段的起始地址。代码段 CS（Code Segment）存放当前正在运行的程序；数据段 DS（Data Segment ）存放当前运行程序所用的数据，如果程序中使用了串处理指令，则其源操作数也存放在数据段中；堆栈段 SS（Stack Segment）开辟了一个以先进后出方式访问的存储区；附加段 ES（Extra Segment）是附加数据段，它是一个辅助的数据区，也是串处理指令的目的操作数存放区。各段在存储器中的分配是由操作系统负责的，每个段可以独立地占用 64KB 存储区，也允许重叠。如果程序中的四个段都在 64KB 范围之内，则程序员只需在程序首部按操作系统的分配设定段寄存器的值即可。如果程序的某一段在程序运行过程中超过 64KB 存储空间，则需动态地修改段寄存器的内容。

PSW（Program Status Word）是一个 16 位的程序状态字寄存器，它由条件标志和控制标志所组成。

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				OF	DF	IF	TF	SF	ZF		AF		PF		CF

条件标志包括：

OF（Overflow Flag）溢出标志，在运算过程中，如操作数超出了机器能表示的范围，则称该位为 1，否则为 0；

SF（Sign Flag）符号标志，记录运算结果的符号，结果为负时置 1，否则为 0；

ZF（Zero Flag）零标志，运算结果为 0 时该位为 1，否则为 0；

CF（Carry Flag）进位标志，记录运算时从最高有效位产生的进位值，最高位有效时进位为 1，否则为 0；

AF（Auxiliary carry Flag）辅助进位标志，记录运算时第 3 位产生的进位值，当有进位时置 1，否则为 0；

PF（Parity Flag）奇偶标志，当结果操作数中 1 的个数为偶数时该位为 1，否则为 0。

控制标志包括：

DF（Direction Flag）方向标志，当 DF 位为 1 时，操作指针为 - 1，否则操作指针为 + 1；

IF（Interrupt Flag）中断标志，当 IF 为 1 时，允许中断正在执行的程序，否则关闭中断；

TF（Trap Flag）陷阱标志，用于单步操作。当 TF 为 1 时，每条指令执行完毕后产生陷阱，由系统操作计算机，否则 CPU 正常工作。

1.4　Turbo C 的存储及混合编程模式

1.4.1　Turbo C 的存储模式

IBM PC 系列机的微处理器的 Turbo C 可以按六种不同的计算机内存和管理方式来编译程序。这六种模式分别是微型（Tiny）、小型（Small）、中型（Medium）、紧凑型（Compact）、大型（Large）和巨型（Huge）。

Tiny Model 将所有段地址寄存器设置为同一个值，所有的寻址都用 16 位来进行。这种编译方式产生的运行文件最小，执行速度最快。通常情况下，按这种模式编译的文件可以用 DOS 命令 EXE2BIN 转换为 .COM 文件。

Small Model 是 Turbo C 的缺省模式，适用于多种编程任务。这种方式下仍然采用 16 位偏移量来寻址，但文件的代码段、数据段、堆栈段及附加段的段地址是分开的。这样被编译的文件可以占用 128K 内存，分为代码段和数据段，寻址时与 Tiny Model 模式相同，但程序可以大一倍。

Medium Model 用于编译大程序，这时文件的代码不限制存储在一个段内，可以采用 20 位寻址多个段。但堆栈段、附加段和数据段被限制在一个段内，段内只能采用 16 位寻址。这种方式适用于数据量小的大程序，程序的执行速度要慢很多，但不低于该程序调用的有关函数的执行速度；在执行数据调用时的操作速度与小型模式相同。

Compact Model 与 Medium Model 互补，这种模式下代码被限制在一个段内，但数据段可以占多个段。这种模式适用于文件较短但数据较多的程序。文件执行速度与小型模式

相同。

Large Model 允许文件代码段和数据段占有大于一段的内存，静态数据被限制在 64KB 以内。这种方式用于需处理大量数据的大程序，它的执行速度也大大慢于上述几种模式。

Huge Model 除静态数据不再被限制在 64KB 之内外，其余与 Large Model 相同。

一般说来，除非有特殊要求需要选用其他模式外，编译时应使用小型模式。当选择不同模式进行编译时，要给编译程序合适的指令。在使用 Turbo C 集成开发环境方式时，使用 Options/Compiler 来选择存储模式；在使用命令行 TCC 方式进行编译时，应使用_mt，_ms，_mm，_mc，_ml 或_mh 来选择微型、小型、中型、紧凑型、大型或巨型存储模式。

1.4.2 混合编程模式

一般情况下，即使只有一个数据存放在另一个段中，需要使用紧凑型而不是小型模式编译，这将出现一部分程序真正需要 20 位寻址，导致整个程序的执行速度下降。通常这种情况可能以各种方式进行调整，然而在 Turbo C 中可以使用“段跨越”修饰符及段修饰符来完成。

1.4.2.1 “段跨越”修饰符

“段跨越”修饰符是 Turbo C 提供的增强功能，它由 far、near 和 huge 三个修饰符组成。这些修饰符可以作用于指针或函数。当作用于指针时，影响数据的查询方式；作用于函数时，影响函数的调用和返回。

“段跨越”修饰符在使用时，写在基本类型说明符之后和变量之前。

例：int far * next;

（1）far（远程）

当需要去访问数据段之外的某个内存区域时，far 是最常用的存储模式段跨越修饰符。如果把程序编译成某种大型数据模式，访问数据的速度很慢。若针对数据段外的内存用 far 说明一些远程指针，而编译仍用较小型的模式，问题就可以解决。这样可以使程序仅在涉及那些确实在数据段外的寻址操作中具有附加操作。

far 作用于函数的情况不太常用，一般只局限于一些特殊编程情况，这时函数不在当前代码段内。例如一个在 ROM 中的基本程序。这时 far 能够确保正确地调用和返回这些语句段。

对于 Turbo C 中的 far 指针，有三点需要记住：

① 指针的运算仅作用于偏移量；

② 因为只检查偏移量部分，因此两个远程指针不能在同一个关系表达式中使用；

③ 如果需要比较两个 20 位地址指针，必须使用 huge 指针。这是因为 = = 和！ = 使用 20 位全地址。

（2）near（近程）

near 指针是一个 16 位偏移量，它用适当的段地址来确定实际内存位置。near 修饰符迫使 Turbo C 把指针视为 16 位偏移量指向在 DS 中保存的段。在使用中型、大型或巨型存储模式编译一个程序时可以使用 near 修饰符，用来超越所使用的模式。

把 near 作用于一个函数时，编译会把该函数视为在小型模式下编译的函数方式来处理。当一个函数是在微型、小型或紧凑型模式下编译，函数调用时将一个 16 位返回地址压入栈内。在大型模式下编译的程序将会把一个“段地址：偏移量”形式的 20 位地址压入栈内。因此，在一个由大型模式编译的程序中，那些高次递归函数应该说明为 near 型，以便节省空间，提高运行速度。

（3）huge（特大）

huge 指针在 far 指针的功能上又补充了两点：

① 它的段地址是规范的，因此两个 huge 型指针的比较是有意义的。

② 当一个 huge 指针增大时，段地址和偏移量可能都改变；但不会出现像 far 指针那样的“折回”问题。

1.4.2.2　Turbo C 的段修饰符

除了 far、near 和 huge 之外，Turbo C 支持 4 个段寻址修饰符_cs，_ds，_ss 和_es。

当这类修饰符应用于指针说明时，使这个指针成为所对应的这个段的 16 位偏移量，即如下语句

```
int _es *ptr;
```

中的 ptr 是放在附加段的 16 位偏移量。

另外，这些修饰符只用在一些十分特殊的场合。

1.5　主机与外部设备的通信

1.5.1　外设接口

计算机运行时的程序和数据都要通过输入设备送入内存，程序运行的结果要通过输出设备送给用户，所以输入、输出设备是计算机必不可少的组成部分。

外部设备与主机的通信是通过外设接口进行的。每个接口包括一组寄存器。这些寄存器有三种不同的用途。

① 数据寄存器：用来存放外设与主机间传送的数据，实际上起到一个缓冲器的作用。

② 状态寄存器：用来保存外设或接口的状态信息，以便 CPU 在必要时测试外设状态，了解外设的工作情况。

③ 命令寄存器：CPU 给外设或接口的控制命令通过此寄存器传送到外部设备。

外部设备都具有以上三种类型的寄存器，每个接口所配备的寄存器的数据是根据设备的需要确定的。为了方便主机访问外设，外设中的每个寄存器均被给予了一个地址，称为端口（Part）地址，又称为端口号，因此组成了一个独立于内存储器的 I/O 地址空间。IBM PC 机的 I/O 地址空间可达 64KB，所以端口地址的范围是 0000H～FFFFH，用 16 位二进制代码来表示。

IBM PC 机的端口地址分配情况见表 1-1。

表 1-1 IBM PC机的 I/O 端口地址分配

地　址	外 部 设 备
00－0F	DMA 芯片 8237A
20－21	中断控制器 8259A
40－43	时钟/定时器
60－63	可编程序外围接口芯片 8255A
200－20F	游戏适配器
321－32F	硬磁盘控制器
378－37A	并行接口打印机适配器
3B0－3BF	单色显示和并行打印机适配器
3C0－3DF	彩色/图形适配器
3F0－3F7	软磁盘控制器
3F8－3FE	异步通讯适配器（Primary）
2F8－2FE	异步通讯适配器（Alternate）

1.5.2 BIOS 及 DOS 功能调用

为了便于用户对外设进行编程，IBM PC 机提供了两种类型的例行程序供用户调用。一种是 BIOS（Basic Input/Output System），另一种是 DOS（Disk Operating System）功能调用。它们都是系统编制的子程序，通过中断方式调用所需要的子程序去执行，执行完毕后返回原来的程序继续执行。这些例行程序有的完成一次简单的外设信息传送，如从键盘输入一个字符或字节送至显示器等；也有的完成相当复杂的一次外设操作，如从磁盘读写一个文件等。总之，操作系统把一些复杂的外设操作编成例行程序，用户简单使用中断指令就可以进入这些例行程序，完成所需要的外设操作。BIOS 和 DOS 功能调用虽然都是系统提供的例行程序，但是它们之间又有差别。BIOS 存放在机器的只读存储器 ROM 中，可以把它看成是机器硬件的一个组成部分；DOS 功能调用是操作系统 DOS 的一个组成部分，它在开机时由磁盘装入存储器，每个例行程序可以一次或多次调用 DOS 以完成比 BIOS 更高级的功能。

为了方便地调用 BIOS 或 DOS 系统功能，系统将每一个例行程序作为中断（CPU 中止正在执行的程序，而转去处理特殊事件的操作）程序，并且在系统启动后所占用的低 1.5KB 字节（0000H～05FFH）中的最低 1KB 字节（0000H～03FFH）内存放 256 项中断向量，每一中断向量对应一种中断类型程序且具有一个中断号，每项占用四个字节，其中两个字节存放中断处理程序的段地址，另两个字节存放偏移地址。向量表中的段地址及偏移地址按中断类型号顺序存放，所以每类中断向量的地址可由中断类型号乘以 4 计算出来。当使用不同类型中断时，只需提供例行程序所需的寄存器值，即可调用该中断例行程序。表 1-2 列出了系统中断向量地址的分配情况。BIOS 功能调用见表 1-3。

DOS 系统功能调用的中断号为 21H，其功能见表 1-4。

表 1-2　　中断向量地址一览表

中断向量存储地址	中断号	功　　能
一、8088 中断向量		
0－3	0	除以零
4－7	1	单步(用于 DEBUG)
8－B	2	非屏蔽中断
C－F	3	断点指令
10－13	4	溢出
14－17	5	打印屏幕
18－1F	6,7	保留
二、8259 中断向量		
20－23	8	定时器
24－27	9	键盘
28－2B	A	彩色/图形
2C－2F	B	异步通讯(secondary)
30－33	C	异步通讯(primary)
34－37	D	硬磁盘
38－3B	E	软磁盘
3C－3F	F	并行打印机
三、BIOS 中断		
40－43	10	屏幕显示
44－47	11	设备检验
48－4B	12	测定存储器容量
4C－4F	13	磁盘 I/O
50－53	14	串行通讯口 I/O
54－57	15	盒式磁带 I/O
58－5B	16	键盘输入
5C－5F	17	打印机输出
60－63	18	BASIC 入口代码

续表 1-2

中断向量存储地址	中断号	功　　能
64－67	19	引导装入程序
68－6B	1A	日时钟
四、提供给用户的中断		
6C－6F	1B	Ctrl—Break 控制的软中断
70－73	1C	定时器控制的软中断
五、数据表指针		
74－77	1D	显示器参量表
78－7B	1E	软盘参量表
7C－7F	1F	图形表
六、DOS 中断		
80－83	20	程序结束
84－87	21	DOS 系统功能调用
88－8B	22	结束退出
8C－8F	23	Ctrl—Break 退出
90－93	24	严重错误处理
94－97	25	绝对磁盘读功能
98－9B	26	绝对磁盘写功能
9C－9F	27	驻留退出
A0－BB	28－2E	DOS 保留
BC－BF	2F	打印机
C0－FF	30－3F	DOS 保留
七、BASIC 中断		
100－17F	40－5F	保留
180－19F	60－67	用户软中断
1A0－1FF	68－7F	保留
200－217	80－85	由 BASIC 保留
218－3C3	86－F0	BSAIC 中断
3C4－3FF	F1－FF	保留

表 1-3　　BIOS 中断

中断号	AH	功　能	调 用 参 数	返回参数
10	0	设置显示方式	AL＝00　40×25 黑白方式 AL＝01　40×25 彩色方式 AL＝02　80×25 黑白方式 AL＝03　80×25 彩色方式 AL＝04　320×200 彩色图形方式 AL＝05　320×200 黑白图形方式 AL＝06　640×200 黑白图形方式 AL＝07　80×25 单色文本方式 AL＝08　160×200 16 色图形(PCjr) AL＝09　320×200 16 色图(PCjr) AL＝0A　640×200 16 色图形(PCjr) AL＝0B　保留 EGA AL＝0C　保留 EGA AL＝00　640×200 16 色图形(PCjr) AL＝0D　320×200 彩色图形 EGA AL＝0E　640×200 彩色图形 EGA AL＝0F　640×350 黑白图形 EGA AL＝10　640×350 黑白图形 EGA AL＝11　640×480 单色图形 EGA AL＝12　640×480 16 色图形 EAG AL＝13　320×200 256 色图形 EGA AL＝40　80×30 彩色文本(CGE400) AL＝41　80×50 彩色文本(CCE400) AL＝42　640×400 彩色文本(CGE400)	
10	1	置光标类型	$(CH)_{0-3}$＝光标起始行 $(CL)_{0-3}$＝光标结束行	
10	2	置光标位置	BH＝页号 DH, DL＝行, 列	
10	3	读光标位置	BH＝页号	CH＝光标起始行 DH, DL＝行, 列
10	4	读光笔位置		AH＝0 光笔未触发 AH＝1 光笔触发
10	5	置显示页	AL＝页号	
10	6	屏幕初始化或上卷	AL＝上卷行数　　AL＝0 整个窗口空白 BH＝卷入行属性 CH＝左上角行号　　CL＝左上角列号 DH＝右下角行号　　DL＝右下角列号	

续表 1-3

中断号	AH	功　　能	调 用 参 数	返回参数
10	7	屏幕初始化或下卷	AL＝下卷行数　　AL＝0 整个窗口空白 BH＝卷入行属性 CH＝左上角行号　　CL＝左上角列号 DH＝右下角行号　　DL＝右下角列号	
10	8	读光标位置的字符和属性	BH＝显示页	AH＝属性 AL＝字符
10	9	在光标位置显示字符及其属性	BH＝显示页　　BL＝属性 CX＝字符重复次数　AL＝字符	
10	A	在光标位置显示字符	BH＝显示页 AL＝字符CX＝字符重复次数	
10	B	置彩色调板 (320＊200 图形)	BH＝彩色调板 ID BL＝和 ID 配套使用的颜色	
10	C	写像素	DX＝行(0－199) CX＝(0－639) AL＝像素值	
10	D	读像素	DX＝行(0－199)　　CX＝列(0－639)	AL＝像素值
10	E	显示字符(光标前移)	AL＝字符 BL＝前景色	
10	F	取当前显示方式		AH＝字符列数 AL＝显示方式
10	10	设置调色板	AL＝设置调色板 AL＝00 设置单个调色板寄存器 BH＝装入调色板的颜色代码 BL＝装入的属性寄存器索引号(0－F) AL＝01 设置边框颜色 BH＝边界区颜色代码 AL＝02 设置调色板寄存器和边框 [ES]:[DX]＝数据表地址 AL＝03 闪烁/亮度位开关 BL＝0 允许背景加亮 BL＝1 允许前景加亮 AL＝07 读单个调色板寄存器 BL＝要读的调色板寄存器号 BH＝从被选择的调色板读颜色代码 AL＝08 读边框颜色选择寄存器 AL＝09 读调色板寄存器和边框颜色 ES＝指向缓冲区段地址 DX＝指向缓冲区偏移地址 AL＝10 设置单个颜色寄存器 BX＝要设置的颜色寄存器号 CH＝绿色强度(0～63) CL＝蓝色强度(0～63) DL＝红色强度(0～63) AL＝12 设置多个颜色寄存器 BX＝要设置的第一个颜色寄存器号 CX＝要设置的颜色寄存器数 ES:DX＝包含颜色代码的缓冲区地址	

续表 1-3

中断号	AH	功　能	调 用 参 数	返回参数
10	10	设置调色板	AL＝13 选择颜色表工作页 BL＝0 选择分页方式 BH＝0 选择 64 色的 4 页 BH＝1 选择 16 色的 16 页 BL＝1 选择颜色表工作页 BH＝0～3 在 4 页方式下 BH＝0～15 在 16 页方式下 AL＝15 读单个颜色寄存器 BX＝要读的颜色寄存器号 AL＝17 读多个颜色寄存器 BX＝所读的颜色寄存器的起始号 CX＝所读的颜色寄存器的总数 ES:DX＝存放读出数据的内存缓冲区地址 AL＝1A 读当前颜色表工作页号 AL＝1B 颜色/灰度转换 BX＝第一个要被转换的颜色寄存器号 CX＝被转换的颜色寄存器的个数	
10	13	显示字符串	ES:BP＝串地址　　CX＝串长度 DH,DL＝起始行，　列　　BH＝页号 AL＝0　BL＝属性 串:char,char,…… AL＝1　BL＝属性 串:char,char,…… AL＝2　　串:char,char,…… AL＝3　　串:char,char,……	光标返回起始位置 光标跟随移动 光标返回起始位置 光标跟随移动
11		设备检验		AX＝返回值 bit0＝1,配有磁盘 bit1＝1,协处理器 bit4,5 ＝01,40×25BW(彩色) ＝10,80×25BW(彩色) ＝11,80×25BW(黑白) bit6,7＝软盘驱动器号 bit9,10,11＝RS－232 板号 bit12＝游戏适配器 bit13＝串行打印机 bit14,15＝打印机号
12		测定存储器容量		AX＝字节数(KB)
13	0	软盘系统复位		
13	1	读软盘状态		AL＝状态字节
13	2	读磁盘	AL＝扇区数 CH,CL＝磁道号,扇区号 DH,DL＝磁头号,驱动器号 ES:BX＝数据缓冲区地址	读成功： AH＝0 AL＝读取的扇区数 读失败： AH＝出错代码
13	3	写磁盘	AL＝扇区数 CH,CL＝磁道号,扇区号 DH,DL＝磁头号,驱动器号 ES:BX＝数据缓冲区地址	写成功： AH＝0 AL＝读取的扇区数 写失败： AH＝出错代码