第一章　微机图形设计概论

随着计算机技术的进步，计算机图形技术的发展也愈来愈受到人们的重视。计算机的应用也逐步由数值计算，数据处理领域向信息处理和知识处理领域拓宽。计算机的应用不断提出各种各样的要求又进一步促进了计算机科学技术的发展和提高。计算机图形学这一新的分支学科的出现，就是计算机应用与计算机技术相互促进的一个范例。在这一章中，我们简单介绍计算机图形学的发展及应用和计算机图形设计的基本方法。

第一节　计算机图形学的发展及应用

一、图形信息的计算机处理

与其它形态的信息相比，图形具有直观明了，含义丰富等种种优点。因此它有着广泛的用途。虽然图形的表示、生成、处理、存贮、检索和管理等要比文字复杂得多，但用计算机处理图形信息比传统的手工或机械方式提高了一大步，它使图形的用途更加广泛，更加有效，而成本越来越低。

与图形信息的计算机处理有关的计算机分支学科有三个，它们是图象处理、模式识别和计算机图形学。

①图象处理(Image Processing)

可见或不可见的图形(图象)经过量化后送入计算机，由计算机按应用的需要进行图象增强、复原、分割、重建、编码、存贮、传输等处理，需要把加工处理后的图形(图象)重新输出，这个过程统称为"图象处理"。

②模式识别(Pattern Recognition)

图形信息输入计算机后，先对它进行特征抽取等预处理，然后用统计判定方法或语法分析方法对图形作出识别，最后由计算机按照人们的使用要求给出图形的分类或描述。

③计算机图形学(Computer Graphics)

使用计算机建立、存贮、处理某个对象的模型，并根据模型产生该对象图形输出的有关理论、方法与技术，称为"计算机图形学"。这里所说的对象(Object)，可以是各种实在的物体，如汽车、房屋、机械零件等，也可以是抽象的或者是假想的事物，如天气形势、人口分布、世界各国经济发展增长速度等。无论何种对象，计算机图形学的主要任务是先对这些对象进行描述(建模)，然后对描述这些对象的一组数据或过程进行种种处理，产生能正确反映这些对象某种性质的图形输出，如汽车外形图、零件加工图、大楼布局图、人口分布图等等……。图形生成的方式有被动(Passive)和交互式(Interactive)两种，前者指图形生成过程中操作员无法对图形进行操纵和控制，它主要使用在以绘图机作为输出设备的早期系统中。交互方式则允许操作人员控制和操纵模型的建立和图形的生成过程，模型及其图形可以边生成，边显示，边修改，直到产生出满意的模型和图形为止。目前，大多数图形系统的工作方式几乎都以

交互式为主。

图象处理、模式识别和计算机图形学这三门学科，它们之间的相互关系可以粗略地用图1-1来表示。

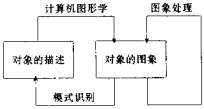


图1-1

二、图形系统的组成

从程序员的角度来看，所有图形系统在概念上均由四个部分组成。

1. 应用数据结构(Application Data Atructure)

应用数据结构实质上是一些数据文件，其中保存着欲生成图形的那些对象的全部描述信息，这些信息包括：用于定义该对象所有组成部分的形状和大小的几何信息及有关的拓扑信息，用于说明与该对象图形有关的属性信息，如色彩、纹理、表面性质等，以及实际问题中还需要涉及的其它一些非几何数据，如材料、单价、加工要求等，它们往往存放在数据库中。

能够正确地表达出一个对象的性质、结构和行为的所有描述信息，称为这个对象的模型。计算机图形最感兴趣的主要是这个对象的几何性质(形状、大小、位置、结构等)，因此，用于刻画被处理对象几何性质的描述信息就构成了它们的几何模型。

2. 图形应用软件

图形应用软件是系统中的核心部分，它是图形技术在各种不同应用中的抽象。其主要功能大体概括如下：

①根据从图形输入设备经由图形支撑软件送来的命令和数据，构造或修改被处理对象的模型。

②从应用数据结构(模型)中取出该对象的几何数据及有关属性，按照应用的要求对它们进行处理，然后使用图形支撑软件所提供的各种功能，生成该对象的图形并在输出设备上输出。

③与图形显示并无直接关系的一些其它处理功能，如性能模拟、分析计算、后处理、用户接口、系统维护等。

3. 图形支撑软件

图形支撑软件通常由一组公用的图形子程序所组成，它扩展了系统中原有的高级语言和操作系统的图形处理功能，特别是采用标准图形软件如 PHIGS、GKS、CGI 等之后，图形应用软件的开发将得到如下三个方面的好处：

①与设备无关。在标准图形软件基础上开发的各种图形应用软件，不必关心具体设备的物理性能和参数。它们可以在不同的硬件系统之间方便地进行移植和运行。

②与应用无关。标准图形软件的各种图形输入输出处理功能，综合考虑了多种应用的不同需要，因此有很好的适应性。

③ 具有较高性能。标准图形软件能够提供多种图形输出原语(Graphic output Primitives),如线段、圆弧、折线、曲线、标志、填充区域、图象、文字等,能处理各种类型图形输入设备的操作,可以允许对图形分段,也可以对图形进行种种变换,因此,应用程序能以较高的起点进行开发。

4. 图形设备

图形系统中的外围设备除大容量外存贮器,通讯控制器等常规设备外,还有图形输出和图形输入设备。图形输出设备有显示器和图形硬拷贝设备两类;图形输入设备的种类繁多,按照它们的逻辑功能可分成定位设备、选择设备、描画设备等若干类。通常,一种物理设备往往兼具几种逻辑功能。在交互系统中,图形的生成、修改、标注等人机交互操作,都是由用户通过图形输入设备进行控制的,表 1-1,表 1-2 是常用设备的分类,供系统配置时参考:

表 1-1 常用图形输出设备的分类

图形显示器	CRT 显示器	光栅扫描图形显示器 随机扫描显示器(刷新式) 随机扫描图形显示器(存贮管式)
	其它显示器	液晶显示器(LCD) 等离子平板显示器 发光二极管显示器(LED) 激光显示器
图形硬拷贝输出	绘图仪	平板绘图机 滚筒式绘图机(静电式,机械式)
	图形打印机	点阵式(机械)打印机 喷墨式打印机 蜡差打印机
	其它设备	缩微胶片输出设备 复印输出设备 录象设备(录象带,录象盘)

表 1-2 常用图形输入设备的分类

逻辑输入设备	对应的主要物理设备
定位设备(locator)	数字化仪,鼠标器,操纵杆等
描画设备(Stroke)	数字化仪(手动,自动)
拾取设备(pick)	光笔,鼠标器
命令选择设备(choice)	按钮,功能键
数值输入设备(valuator)	可调电位器,数字盘,键盘
字符输入设备(String)	键盘,字符阅读器,语音识别装置

三、计算机图形学的应用

计算机图形学是研究如何使用计算机这个工具来描述物体并生成物体的图形。利用计算机生成图形,这是人们自从发明照相技术和电视技术以来产生图形的一种重要手段。与照相技术或电视技术相比,它的主要优点有:

* 计算机不但能生成实际存在的具体对象的图形,而且还能生成假想的或者抽象的对象的图形,例如人口分布图、天气图、经济增长趋势图等。

* 计算机不仅能生成静态图形,而且还能生成随地域、时间、位置或其它参数而变化的动态图形。

* 计算机生成图形的过程中，允许人们随时进行修改和各种控制。

因此，计算机图形学有着广阔的应用范围和发展前景。下面是它一些有代表性的应用领域：

①在科学和工程计算及事务处理中，常常使用图形来表示数值计算或数据处理的结果。例如函数的图形、分子模型、框架结构、应力或场强的分布等。用图形来表示大量的数值结果，不但形象、直观，而且能反映出各种参数相互之间的关系，比较容易揭示事物的本质和规律。最近几年，“n 形象化计算技术”(Computing Visualisation)的兴起就是图形学与大型科学计算相结合的结果。

②各种地理信息和自然现象的图形表示，如地形图、天气图、海洋图、石油开采图、人口密度图等，利用计算机制作地图(Computerized Cartography)是计算机在古老的地理科学中的重要应用之一。

③计算机辅助制图和辅助设计(Computer Aided Drafting and Design)，简称为(CADD)，这是计算图形学最早、也是最重要的一个应用领域。目前，计算机辅助设计在机械、电子、电气、建筑、轻工等各种工程领域都有着广泛的应用，并取得明显的经济效益。

④计算机模拟和动画。利用计算机模拟某个系统或某种现象和过程，这是一种进行系统分析和仿真的有效手段。图形显示在计算机模拟中有着极重要的作用，尤其是动画技术。用计算机进行实时模拟，则对图形显示技术有很高的要求。

⑤过程控制。在过程控制中，图形显示器可用来显示被控对象(炼油厂、发电厂等)有关环节的状态，操作人员通过图形交互技术进行各种调节或处理所发生的意外事故。由于图形显示清晰明了、形象直观，比指示灯、数字仪表等包含的信息量大得多，因而现场操作人员的工作效率得到很大提高。

⑥办公自动化和电子出版技术。图形显示技术在办公自动化和事务处理中的使用，有助于数据及其相互关系的有效表达，因而有利于人们进行正确的决策。利用电子计算机进行资料、文稿、书刊、手册等的编写、修改、制图、制表、分页、排版，这是对传统活字印刷技术进行的重大变革，没有图形显示技术的支持，这种电子出版技术的实现是不可能的。

计算机图形学的应用远远不止上述六个方面，它在艺术、广告、数学、游戏、软件工程等许多方面都有着很好的应用。近几年来，随着图形设备价格的下降和图形显示技术的发展，特别是个人计算机图形功能的增强和高性能图形工作站(工程工作站)的出现，计算机图形正在得到更加广泛的应用。

计算机图形学是计算机场科学中一个比较年轻的分支学科，它的核心技术是如何建立所处理对象的模型并生成该对象的图形，其主要研究内容大体上可以概括为如下几个方面：

①几何模型构造技术(Geometric Modelling)。例如各种不同类型几何模型二维、三维、分数维(Fractal Model)的构造方法及性能分析，曲线与曲面的表示与处理，专用或通用模型构造系统的研究，等等。

②图形生成技术(Image Synthesis)，例如线段、圆弧、字符、区域填充的生成算法，以及隐线/隐面消除、光照模型、浓淡处理(Shading)、纹理、阴影、灰度与色彩等各种逼真感图形技术。

③图形操作与处理方法(Picture Manipulation)。例如图形的开窗、裁剪、平移、旋转、放大、缩小、投影等各种几何变换操作方法及其软件或硬件实现技术。

④图形信息的存贮、检索与交换技术，例如图形信息的各种内外表示方法、组织形式、存取技术、图形数据库的管理、图形信息的通信，等等。

⑤人机交互及用户的接口技术。例如新型定位设备、选择设备的研究，各种交互技术如构造技术、命令技术、选择技术、响应技术等的研究，以及用户模型、命令语言、反馈方法、窗口系统等用户接口技术的研究。

⑥动画技术。研究实现高速动画的各种软、硬件方法，开发工具，动画语言等。

⑦图形输出设备与输出技术。例如各种图形显示器(图形卡、图形终端、图形工作站等)逻辑结构的研究，实现高速图形功能的专用芯片(ASIC)的开发，图形硬拷贝设备(特别是彩色硬拷贝设备)的研究等。

⑧图形标准与图形软件包的研究开发。如制订一系列国际图形标准，使能满足图形应用软件开发工作的需要，并使图形应用软件摆脱对硬设备的依赖性，允许在不同系统之间进行移植。

总之，计算机图形学的研究内容是十分丰富的。许多研究工作已经进行了多年，取得了不少成果。随着计算机技术的进步和图形显示技术应用领域的扩大和深入，计算机图形学的研究、开发与应用还将得到进一步的发展。

第二节　计算机图形设计基本方法

一、概述

在 IBM PC 微机的各种应用中，许多都与图形显示有关。例如计算机辅助设计、电子印刷、过程监控、办公自动化、图象处理、辅助教学，计算机模拟等等。为了开发各种与图形有关的应用程序，必须要有必要的图形设备和有关的图形支撑软件。IBM PC 机中图形应用软件，图形支撑软件及图形硬设备之间的关系如图 1-2 所示。其中，最底层的 BIOS(基本输入/输出系统)只提供了一组简单的子程序，功能相当有限，且完全与硬件有关。图形设备驱动程序是 MS—DOS 的一个部分，它是 DOS 与物理设备之间的一个接口，由于它是按照 MS 公司规范编写的独立程序模块，因此修改、扩充或更换驱动程序极为方便。通常，不同的显示卡各有其相应的驱动程序，甚至同一显示卡的不同显示模式也有各自的驱动程序，这些驱动程序与上层软件保持统一的数据和程序接口，更换图形设备时，只需在系统中安装相应的驱动程序，整个系统就可以正常运行。此外，驱动程序的功能也比 BIOS 扩充了许多，从而使上层软件的开发更为方便。图形子程序库是更高级的支撑软件，它可以分成两类：一类是各种程序设计语言所专用的子程序库，如 MSC(Misrosoft C)Turbo C、Turbo PASCAL、BASIC 等语言各自使用的库程序；另一类是按国际标准或公司标准开发的图形子程序库，如 GKS、CGI、CGM、PostScript 和 MS－Windows SDK。这些图形子程序功能丰富，通过性强，不依赖于具体的设备和系统，与多种程序设计语言均有接口，因此，在此基础上开发的图形应用软件不仅性能好，而且易于移值。

开发图形应用程序时，可以根据应用的要求、操作环境以及所使用的图形设备，选择某一种支撑软件来进行。表 1-3 是使用不同级别的支撑软件进行应用开发的对比。

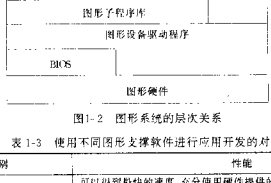


图1-2 图形系统的层次关系

表 1-3 使用不同图形支撑软件进行应用开发的对比

支撑软件级别	性能
无支撑软件(直接程序设计)	可以得到最快的速度,充分使用硬件提供的功能;编程复杂,程序几乎无兼容性
BIOS	提供简单绘图功能,有 BIOS 级兼容性,但功能差,编程麻烦,速度有影响
图形驱动程序	图形功能增加,隔离了设备的物理特性,使用较方便,兼容性有提高
程序语言专用图形子程序库	功能较多,编程方便,效率稍低,通用性差,有设备级的兼容性
标准图形软件和窗口软件	功能丰富,编程方便,通用性好,有系统级兼容性,但开销大,效率低

在以后的几章中我们将详细讨论在不同的支撑软件级别下编程的方法,在这章中我们只作简单介绍。

二、图形显示器的直接程序设计

在开发图形支撑软件、应用软件的过程中,用户程序常常需要直接访问显示器硬件,这种直接对图形硬件进行控制和处理的做法,就称为直接程序设计。这种方法速度快,能充分发挥硬件的性能,但开发出来的程序兼容性差,很难不经修改就可在另一种显示器上正常运行。由于直接程序设计方法是设计、扩充和修改 BIOS 功能,开发设备驱动程序,分析、移植和改造现有程序必不可少的手段,它又与具体设备的逻辑结构、性能参数等密切相关,因此不同的显示器上实现不同图形功能的编程方法有:CGA 的直接程序设计,EGA/VGA 的直接程序设计,还有在不同显示器下 BIOS 程序设计。在此我们仅对这种方法的原理和原则作概括性介绍。

(一)显示器硬件与系统的接口

使用直接程序设计方法编程需要直接读写显示器的硬件寄存器和显示存贮器,这些读写操作分别通过输入输出指令(或语句)和访问指令(或语句)来完成。

尽管 PC AT 和某些 386 机的外部数据总线宽度为 16 位或 32 位,但大部分显示器仍以字节(8 位)作为与 CPU 通讯的单位。另外,多数图形卡的显示存贮器地址均在主存贮器空间的 A0000～BFFFF 范围之内,而它们的控制寄存器所占用的 I/O 端口地址范围是 3B0～3DF,所以,地址总线与它们的接口分别是 20 位和 10 位。

显示器的控制和状态寄存器是使用输入/输出指令(IN/OUT)进行读写的,它们用端口地址(Port Address)进行选择。由于控制和状态寄存器数目很多,而系统分配给显示器的端口地址数量有限,因此,常常采用间接选择的方法,即预先将这些寄存器编号(称为索引号),并添加一个索引寄存器和一个数据寄存器。需要访问某个寄存器时,先使用输出指令把该寄

存器的编号送入索引寄存器，然后再使用输入或输出指令对数据寄存器进行读写操作。该读写操作实际上是对指定的那个寄存器进行的(如图 1-3)。

(二)寄存器读写操作

对图形显示器的控制必须通过寄存器的读写操作来完成。使用汇编语言时，主要有以下四条输入输出指令：

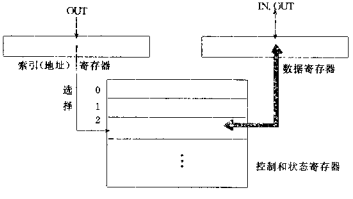


图1-3 寄存器的间接存取

IN AL,DX; 从 DX 指出的端口寄存器读一个字节送 AL

IN AX,DX, 从 DX 指出的端口寄存器中读入一个字送 AX

OUT DX,AL; 把 AL 中的字节写入 DX 指出的端口寄存器

OUT DX,AX; 把 AX 中的字写入 DX 指出的端口寄存器

前面已经说过，图形显示器一般均以字节单位与 COU 交换数据，它们所含的寄存器长度也都是 8 位，所以，使用“IN AX,DX”和“OUT DX,AX 指令时，数据传送实际上分成两次完成，且对应的寄存器也是两个，端口地址分别是(DX)和(DX)+1。

在采用间接方法选择显示器的寄存器时，通常总把索引寄存器的端口号和数据寄存器的端口号安排成为相互连续。因此，当需要向索引号为 Index 的寄存器输出一个数据字节 Data 时，可以使用下面的指令序列：

```
MOV AL,Index        ;AL 中放索引号
MOV DX,Port         ;DX 中放索引寄存器端口地址
OUT DX,AL           ;向索引寄存器输出索引号
MOV AL,Data         ;AL 中放输出数据
INC DX              ;DX 中是数据寄存器端口地址
OUT DX,AL           ;向索引号指定的寄存器输出数据
```

使用“OUT DX,AX”指令后，上述过程可以简化为：

```
MOV AL,Index
MOV AH,Data
MOV DX,Port
OUT DX,AX
```

这样就加快了处理的速度。

在高级语言所编写的程序中需要直接对显示器的寄存器进行读写时，不同的语言有不

同的方法，一般可以有三种选择。下面以 MSC 和 Turbo C 为例作简单的介绍。

①直接使用高级语言提供的输入输出语句(MSC)。MSC 提供的输入输出语句有：

```
b=inp(port);          /* 从端口 port 中读入一个字节放入 b */
w=inpw(port);         /* 从端口 port 中读入一个字放入 w */
outp(port b);         /* 把字节 b 写入端口 port */
outw(port w);         /* 把字 w 写入端口 port */
```

其含义与前面介绍的汇编语言输入输出指令相同，但它们可在 C 语言源程序中直接使用

②在高级语言源程序中插入一段汇编程序(Turbo C)

使用这种方法时，被插入的汇编语言程序，每一条指令前都须冠以编译命令 asm，以便在编译时被特殊处理。例如：

```
main()
  {
     asm mov al,index
     asm mov dx,port
     asm mov ah,datd
     asm out dx,ax
  }
```

需要注意，有些寄存器(如 BP，ES 等)在使用时需预先保护入栈，使用后再恢复原状，否则将影响程序的正确运行。

③高级语言中调用汇编子程序目标模块，汇编子程序的标准格式如下：

```
_TEXT   SEGMENT BYTE PUBLIC'CODE'
        ASSUME CS:_TEXT,DS:NOTHINT
MYPROG  PROC    NEAR        ;MYPROC 是任意的汇编程序名
        PUSH    BP
        MOV     BP,SP       ;取得参数的指针
        PUSH    SI
        PUSH    DI          ;BP,SI,DI,ES 和 DS 必须保护入栈…
        …
        MOV     AX,[BP+4],取得参数 P1
        MOV     BX,[BP+6],取得参数 P2
        MOV     CX,[BP+8],取得参数 P3
        …
        POP     DI
        POP     SI
        MOV     SP,BP
        POP     BP          ;恢复被保护的寄存器
        RET
MYPROG  ENDP
 TEXT   ENDS
        END
```

于是，在 C 语言源程序中就可以以 MYPROG(P1,P2,P3)的形式调用上述汇编子程序

了。无论使用何种方式，对显示器的寄存器进行读写操作时，必须注意以下几点：

* 对有关寄存器的格式和含义应有清楚而透彻的了解，其中不应修改的位应保持不变。

* 用于控制 CRT 扫描参数的一组寄存器，用户程序一般不直接处理，因为若处置不当，会造成 CRT 损坏。

* 改变某些控制寄存器状态时，特别是一些"只写"寄存器状态时，软件应在主存中留下该寄存器状态的副本。

* 在速度特别重要的场合，编制的程序要优化。

(三)显示存贮器操作

对显示存贮器进行读写操作的目的主要是为了生成、修改或保存被显示图形的位图。由于 PC 机显示存贮器是 CPU 主存贮器的一个组成部分，所以，几乎大部分指令都可以对显示存贮器进行操作，其中用得最多的是：

MOV　　DEST,SRC;字节或字的传送指令

MOVSB　　DEST,SRC;字节串的传送指令

MOVSW　　DEST,SRC;字串的传送指令

其中源地址 SRC 和目的地址 DEST 既可以是内存也可以是显存，当两者均为显存时，则可实现位图在显存内部的移动。一些算术和逻辑指令则用于对显存内容进行修改，它们常和 MOV 指令配合使用。

有些高级语言也可直接对(显示)存贮器进行存取操作，例如 MSC 语言提供了下列语句：

```
movedata(srcseg,srcoff,dstseg,dstoff,n)
```

它把从 srcseg:srcoff 开始的长度为 n 的一串字节传送到 dstseg:dstoff 开始的存贮区域中去。下面是把变量 i 值写入显示器 B8000 处的一个程序片段：

```
{int i=0x30,struct SREGS src;
  segread(src);              /*取得段寄存器的值*/
  srcseg=src.ds;          /*使 srcseg 为 DS 寄存器的值*/
  srcoff=(int)i;           /*使 srcseg:srcoff 指向 i 的值*/
  dstseg=0×B800;dstoff=0×0000;   /*使 dstseg:dstoff 指向 B8000*/
  movedata(srcseg,srcoff,dstseg,dstoff,i);
}
```

其中 SREGS 是 MSC 语言预先定义的段寄存器结构类型 9 参见后面的(bios.h)中的描述，它的四个域分别表示四个段寄存器，例如 src.ds 为 DS 寄存器。

与寄存器操作一样，直接对显示存贮器的存取操作也可以通过在 C 语言源程序中插入一段汇编程序或调用一个汇编程序来实现。无论哪一种方法，存取显示贮器的操作步骤大体如下：

①了解显示器当前的显示模式及工作页面；

②根据要处理的象素或字符编码(及属性)的屏幕坐标位置计算出它们在显示存贮器中的地址；

③判断显示存贮器是否处于 CPU 可访问状态(例如是否回扫状态)，若否，则等待；

④读出象素或字符编码所在的字节；

⑤选择该字节中所需处理的位进行修改，其它无关的位保持不变；

⑥把该字节写回显存原处。

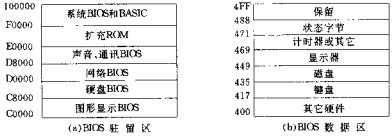
其中，步骤③有些显示器需由软件进行，有些则由显示器硬件自动完成，若不加处理，则会引出屏幕画面闪烁不定。

三、基于 BIOS 的图形程序设计

(一)BIOS 概述

IBM PC 机中有一组固化在只读存贮器中的子程序，它们主要用来管理各种输入输出设备，称为基本输入输出系统(Basic Input Output System)，简称为 BIOS。

BIOS 常驻在系统存贮器的高区，其空间分布大体如图 1-4 所示。BIOS 中用来管理键盘、打印机、异步通讯、时钟等设备的基本子程序一般都装在主机板的 ROM 中，称为系统 BIOS，硬盘、网络、图形显示器等的基本子程序则分别装在各自的控制卡上。系统加电启动时，系统 BIOS 负责检查其它控制卡上有无 BIOS 存在，若有的话，则执行必要的初始操作(例如安装各自的中断向量)。



(a)BIOS 驻 留 区　　(b)BIOS 数 据 区

图1-4　BIOS在存贮器中的分布

BIOS 的主要用途有两个：一是向高层软件提供一组有一定功能、易于使用的子程序，减少了编程的困难；二是为不同厂家的外部设备提供互相兼容的一种方法，因为不同性能参数的设备，只要它们的 BIOS 功能完全兼容。

BIOS 的调用方法很简单。每个设备的控制程序使用一种软中断来调用(见表 1-4)，控制程序中的不同功能(子程序)则由 AH 和 AL 寄存器来区分。中断向量中存放着相应的设备控制程序的起始地址，它的长度为 4 个字节，低字节是段地址，高字节是段内偏移量，它们是在系统初始启动时装入的。与显示器有关的软中断共两个：INT10 和 INT5。

表 1-4　部分 BIOS 软中断号(十六进制)

软中断号	设备控制程序	中断向量地址
10	图形显示器	40
13	磁盘	4C
14	异步通讯	50
16	键盘	58
17	打印机	5C
11	设备状态	44

续表

软中断号	设备控制程序	中断向量地址
12	存贮器大小	48
1A	时间/日期	68
5	屏幕打印	14
19	初始起动	64

(二)显示器 BIOS 的功能与调用方法

IBM PC 所配置的各种显示器若自带 BIOS 程序时,其 BIOS 程序一般驻留在地址空间 C0000－3FFF 范围之内,它除了包含一组供上层软件调用的基本子程序外,还包括必要的字库和各种控制信息。不同的显示器,BIOS 所提供的子程序数目和功能不完全一样,但高性能的显示器与低性能的显示器之间往往有"向下兼容性",例如 EGA 与 CGA 兼容,VGA 与 EGA、CGA 兼容。显示器 BIOS 所提供的功能大体上有如下几组:

①控制功能。例如显示模式的选择,工作页面的选择,光标形态与位置的控制等。

②查询功能。查询现行显示模式及工作页面,查询光标的位置,查询光标位置处的字符及属性,查询显示器的配置信息等。

③字符输出功能。输出单个字符,输出字符串,仿真 TTY(电传打字机)方式输出字符等。

④图形输出功能。例如画点、画线等,多数显示器 BIOS 只提供画点功能,其它功能需自行扩充。

⑤彩色控制功能。设置调色器,设置边框色,保存及修改彩色表等。

⑥字库处理功能。装入字库,选择工作字库,读取字库信息等。

显示器 BIOS 提供的上述功能都是通过 INT10 调用的。AH 寄存器中存放着指定的功能号,其余寄存器则用来存放必要的输入参数。返回值也放在寄存器中。以模式选择功能为例,它的功能号为 0,输入参数是模式编号,用 AL 寄存器指出,无返回值。因此,设置显示器模式的汇编程序是:

```
MOV AH,0          ;指定功能号
MOV AL,mode       ;指定设置的模式
INT 10H           ;调用显示器 BIOS
```

用 C 语言编写的程序需要调用 BIOS 功能时,可以把上述汇编程序段插入 C 语言程序中,就像第一节介绍的做法。也可以使用 MSC 提供的标准库函数 int86 来调用系统中任一软中断。由于调用 BIOS 功能的软中断都是使用寄存器传递参数的,所以 C 语言必须通过某种机构读写寄存器。下面是 MSC 关于寄存器联合和段寄存器结构的定义(在 bios.h 文件中):

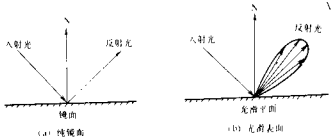


图 3-81 镜面反射

实用时，常采用余弦函数的幂次来模拟一般光滑表面的镜面反射光的空间分布，其值为

$$I_{ps}\cos^n\theta$$

其中

I_{ps}为镜面反射方向上的镜面反射光亮度；

θ为镜面反射方向和视线方向的夹角；

n 为镜面反射光的会聚指数。物体越光滑，其镜面反射光的会聚程度较高(n 值大)。

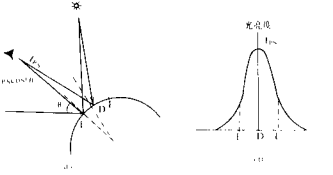


图 3-82 用于光滑球面的 Phone 光照模型

图 3-82 为镜面反射用于一个光滑球面时的情形。图(a)中 D 点处镜面反射方向和视线方向一致，$\theta=0$，D 处呈现明亮的高光。面在 E 和 E′点，θ变大，便观察者接受到的镜面反射光急剧下降，图(b)给出了镜面反射分量的明暗过渡曲线。

对于一特定的物体表面，以上三种分量所占的比例具有一定的值。若令 k_a，k_d 和 k_s 分别表示环境反射、漫反射和镜面反射分量的比例系数，则一个实用的光照明模型可以表示如下：

$$I = k_aI_{pa} + \sum[kdI_{pd}\cos i + ksI_{ps}\cos^n\theta]$$

其中符号 $\sum$ 表示对所有特定光源求和。

该式又称为 phong 模型，式中的 I_{pa}，I_{pd}，I_{ps}和 I 都是光谱量。为避免光谱计算，可将上式转换至光栅图形显示器的 RGB 三基色颜系统，这时 phong 模型可写成

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = k_a \begin{bmatrix} r \\ P_a \\ g_{pa} \\ b_{pa} \end{bmatrix} + \sum \left[k_d \begin{bmatrix} r_{pd} \\ g_{pd} \\ b_{pd} \end{bmatrix} \cos i + k_s \begin{bmatrix} r_{ps} \\ g_{ps} \\ b_{ps} \end{bmatrix} \cos^n\theta \right]$$

```
MOV     BP,[BP+14]
INT     10H             ;执行显示器 BIOS
POP     ES
POP     BP              ;恢复 ES 和 BP
RET
_Display _BIOS ENDP
```

在C语言源程序中调用上述汇编程序非常方便。下面是一个简单的例子，它先通过2#功能把光标定位在第10行第32列的位置，然后调用E#功能输出一个字符串。

```
test _Display _BIOS( )
{int i;
  static char string[]="print string using BIOS function E";
  display _BIOS(0x0200,0,0,0x0a20);  /* 把光标移到 ab 行 20h 列的位置处 */
  for(i=0;string[i]! =NULL;i++)  /* 使用功能 E 显示一个字符串 */
  display _BIOS(0x0E00+string[i]);
```

(三)BIOS的扩充方法

由于BIOS中INT10的图形功能较弱，完全采用INT10编制图形应用程序的效率较低，要求应用程序直接对硬件程序设计时难度大、兼容性又差，为此，可以采用扩充BIOS功能的方法，即预先利用直接编程方法按一定接口规则编一组子程序，然后将它们扩充到BIOS中去，供编制应用程序的人员使用。

扩充BIOS INT10功能的方法有两种：一种是直接修改和扩充固化在ROM中的BIOS程序，例如Color400显示器、0520CH和CEGA中西文图形显示器等均是如此，这种做法需要修改显示卡硬件，它是由生产厂完成的；另一种是采用全软件的做法，它显得更加方便灵活。办法是找一个PC机系统软件不使用的软中断号(一般INT 40—INT FF之间，参见表1-5)，然后编制一个可执行的汇编程序，它由三部分组成，即

(1)装入程序，它的功能是：

①把原INT10的中断向量传送到某个指定的中断号(例如INT60)中去；

②使用DOS的系统调用(功能号=25h)为INT10装入新的中断向量地址；

③使用DOS的系统调用(功能号=31h)结束本程序的执行并把它留在存贮器中；

(2)新的INT10软中断总控程序，每当调用INT10时，它的工作过程如下：

①保护有关的寄存器入栈；

②判断AH寄存器中的功能号；

③若为INT10原有功能号(假设不超过20h)，则调用INT60，执行原来的INT10 BIOS程序，完毕后执行⑤

④若为INT10扩充的新功能(假设依次为80h，81h，…)，则调用程序中第(3)部分的有关子程序；

⑤恢复各寄存器原先的值，中断返回。

表 1-5 DOS 的存贮器低区布局

地　址	内　容
005FF 00500	DOS 数据区
00400	BIOS 数据区
00100	用户指定的中断向量(INT40—FF)
00080	DOS 使用的中断向量(INT20　3F)
00040	BIOS 使用的中断向量(INT10—1F)
00000	硬件中断向量(INT0—0F)

(3)执行 INT10 扩充功能的一组子程序。

下面是采取这种方法扩充 INT10 功能的一个汇编程序框架,读者可根据实际需要自行修改和扩充。这种做法在 MS—DOS 中有一定的代表性,它常常被称为"结束驻留法"(TSR 法,Terminate Stay Resident)。

```
CodeSeg    SEGMENT
           ASSUME CS:CodeSeg
LOADING    PROC    FAR
           XOR     AX,AX
           MOV     DS,AX
           MOV     AX,DS:[10H*4]          ;将 INT10 中断向量保存至 INT60 处
           MOV     DS:[60H*4],AX
           MOV     AX,DS:[10H*4+2]
           MOV     DS:[60H*4+2],AX
           MOV     AX,CS                  ;DS:DX 是程序的入口地址
           MOV     DS,AX
           MOV     DX,OFFSET NEW_INT10
           MOV     AX,2510H               ;安装 INT10 的新中断向量
           INT     21H
           MOV     AH,31H
           MOV     DX,OFFSET Bottom
           MOV     CL,4
           SHR     DX,CL                  ;DX 是本程序的长度
           ADD     DX,11H
           INT     21H                    ;结束并驻留
LOADING    ENDP
NEW_INT10  PROC    ;INT10 的总控程序
           PUSH    AX
           PUSH    BX
           PUSH    CX
```

```
                PUSH    DX
                PUSH    BP
                PUSH    DS
                PUSH    ES
                CMP     AH,20H              ;判断功能号是否超过 20h
                IG      I10_80              ;处理原 INT10 功能
                INT     60H
                JMP     I10_END
I10_80:         MOV     BL,AH
                SUB     BL,80H
                SHL     BX,1                ;BX=(funcnon # -80) * 2
                CALL    CS:[BX+I10_80H]
I10_END:        POP     ES
                POP     DS
                POP     BP
                POP     DX
                POP     CX
                POP     BX
                POP     AX
                IRET
NEW_INT10       ENDP
I10_80H         DW      FUNC80              ;功能号 80h 的子程序入口地址
                DW      FUNC81              ;功能号 81h 的子程序入口地址
                ...
                与 INT10 扩充功能对应的一组子程序
                ...
Bottom          LABEL WORD
Codeseg         ENDS
                END
```

使用 BIOS 进行图形应用程序的开发，虽然由于 BIOS 屏蔽了显示器硬件的许多物理特性，不再需要程序员直接与寄存器和显示存贮器打交道，因而大大减少了编程的难度，也给软件的可移性创造了条件。但是，由于 BIOS 所提供的图形功能相当有限，而且级别也比较低，例如它们都直接使用屏幕坐标来描述几何图元，而不同显示器(甚至不同模式)的屏幕坐标取值范围各不相同，这就给图形的移植带来了麻烦。再加上调用 BIOS 功能时需要使用寄存器来传递参数和返回值，因而对高级语言编程有许多不便，而且程序可读性、可维护性也较差，这些都是在 BIOS 级上开发图形应用程序的不足之处。

四、设备驱动程序及虚拟图形设备

从上面的介绍中可以看出，直接对图形硬件进行程序设计或使用 BIOS 进行程序设计，

应用程序可以使用的功能和有关的参数与具体的物理设备有关，这就给应用程序的开发带来种种不便。为此提出了虚拟图形设备的概念。所谓虚拟图形设备是程序设计者概念上的一种逻辑设备，不论何种物理设备，它们都使用统一的虚设备坐标系，具有一组通用的图形输出功能，程序设计时，每一种功能的调用以及参数传递的格式都有标准的形式。虚拟图形设备是图形系统中常用的一种技术，它为图形软件及其应用程序的开发提供了一个与设备无关的接口，便于程序的移植，有关如何在虚拟设备上编写程序的方法将在以后的章节中有详细描述。

五、高级程序设计语言的图形设计及图形库

使用驱动程序实现的虚拟图形设备，虽然能够满足应用程序对图形功能以及设备无关性的基本需要，但由于软中断要使用寄存器传递参数，用户需要记住功能号及存放参数的寄存器名，调用方法不够直观。另外，驱动程序一般是用汇编语言书写的，功能可能还不能满足用户程序的需要，这些问题可以通过设计图形子程序库的方法来解决。

(一)图形的设计方法及其特点

一些程序设计语言本身含有图形输出语句，例如 BASIC 语言和页面描述语言 PostScript 等均设有图形语句，用户程序可直接使用这些图形语句显示图形。另外一些语言则配备有图形子程序库，用来实现图形功能。例如，Turbo C 就有一个图形子程序库 GRAPHICS。使用 Turbo C 编写程序时，可以使用这个库完成图形输出功能。

编写图形应用程序，可直接使用这些图形语句或图形函数来显示图形。一般情况下，高级程序设计语言所含有的图形输出语句基本上能满足应用程序对图形功能的要求。

使用高级语言进行图形编程的最大优点是易学、易懂，对于一般具有一定计算机基础知识的人来说，比较容易掌握。

对于如何利用高级语言中的图形语句、图形函数，以后的章节中也有详细的描述，在此就不进行更深的讨论。

(二)图形库

利用汇编语言编写图形程序是一种底层开发图形功能的有利手段。特别是在开发图形系统软件时，用汇编语言编写图形元素的基本程序供应用程序使用是非常方便的。有些高级语言中具有自己的图形库函数，如 Turbo C、Microsoft C、Turbo Pascal BASIC 等。下面我们介绍一下 Turboc 中的图形子程序库。

1. Turbo C 的图形子程序库

Turbo C 有一个图形子程序库 GRAPHICS，库中的函数所使用的数据常量及数据结构在文件 GRAPHICS. H 中都有定义，所以在使用图形子程序库时，必须在程序首部嵌入 GRAPHICS. H 文件。使用格式为：

```
#include(graphics.h)
```

Turbo C 图形库中共有 40 多个函数，这些函数按完成的功能分为 8 组：图元素的输出，属性设置，文字处理，图象操作，图形工作状态的查询，坐标变换，象素操作，调色板。其中图元素的输出包括有对直线、矩形、圆/椭圆、圆弧/椭圆弧、扇形和填充区域等。矩形、扇形、圆/椭圆可以是实心、空心，填充类型可以是单色，也可以为图案。

下面我们运用 Turbo C 中提供的图形库函数来画一些简单的图形

```
/ * drawing graphics with Thrbo C * /
#include<graphics. h>
main(      )
{
int graphdriver=DETECT,graphmode;/ * Autodetection * /
struct arccoordstyp arcinfo;
int xasp,yasp
long xlong;
initgraph(&graphdriver,&graphmode,"␣");/ * initidlix graphics * /
/ * draw a 90 degree arc with radius of 50 * /
arc(150,0,89,50);
/ * get the coordinates of the arc and connect ends * /
getarccoorde(&arcinfo);
line(arcinfo. xstart,arcinfo. ystart,arcinfo. xend,arcinfo. yend);
/ * draw a circle * /
circle(150,150,100);
/ * draw an ellipse inside the circle * /
ellipse(150,150,0,359,100,50);
/ * draw and fill a pielice * /
setcolor(WHITE);/ * white outline * /
setfillstyle(SOLID_FILL,LIGHTRED);
pieslice(100,100,135,225,49);
setfillsytle(SOLID FILL,WHITE);
pieslice(100,100,225,360,449);
/ * draw"square"rectangle * /
getaspectratio(&xasp,&yasp);
xlong=(100L * (long)yasp)/(long)xasp;
rectangle(0,0,(int)xlong,100);
closegraph(    );
}
```

2. 用户图形子程序库设计

用户也可以根据自己的需要为高级语言开发图形子程序库。自己开发的程序库中，子程序的名称及调用规程可以自行定义，或仿效一些语言(如 Turbo C)的图形库。有可能的话也可以使用国际标准 CGI 格式。如果选用 CGI 或 CGI 的子集实现库程序，就会使图形系统兼容性进一步提高。有关 CGI 的介绍，在以后章节中描述。

六、二维图形标准 GKS

随着技术的发展，市场上出现了与 PC 机配套的大量不同类型的图形设备。它们的性能越来越好，成本越来越低。但是，由于图形设备的种种类繁多，性能参数又有很大差异，因此开