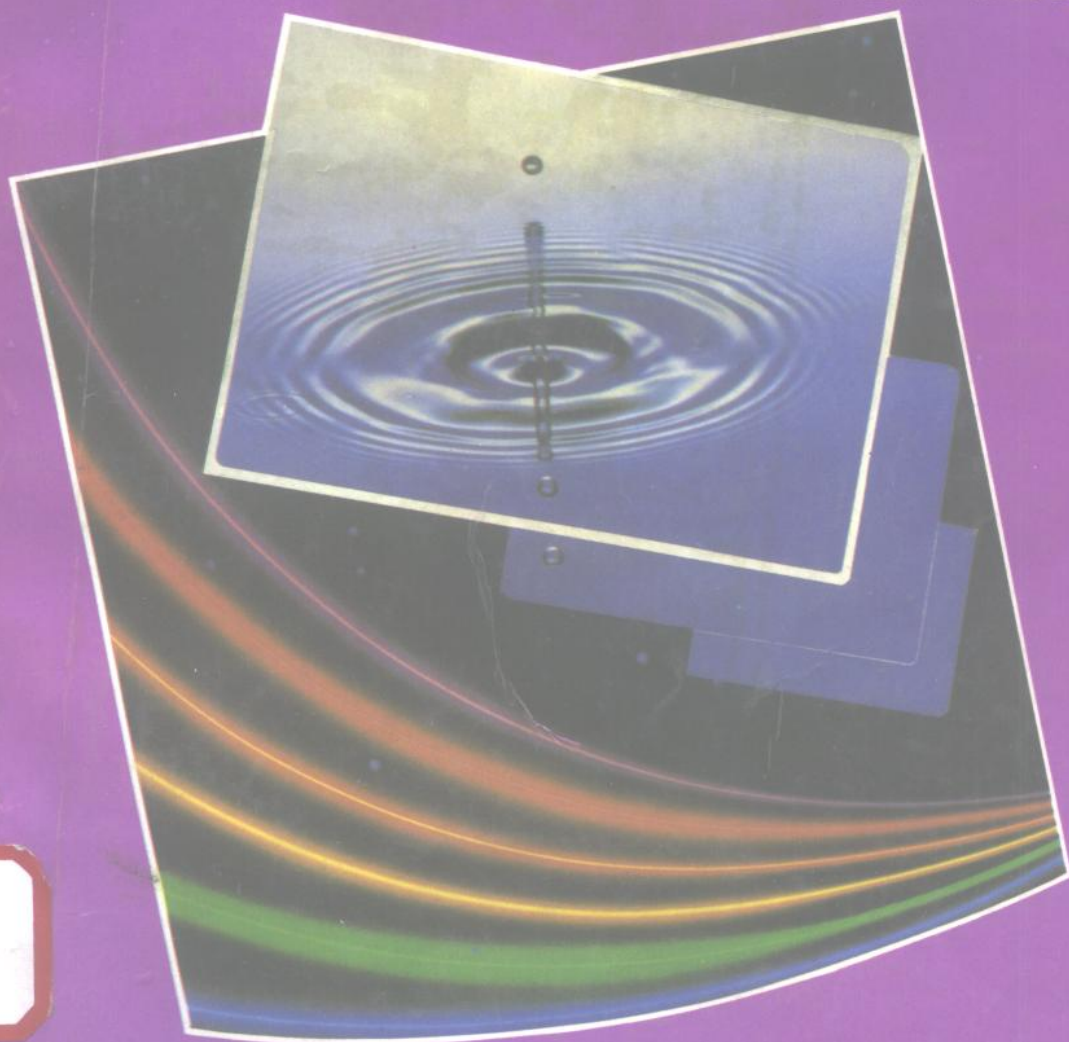


Turbo C/Borland C++

用户界面 程序设计

周升锋 李立新 孙传俊 编著



西安交通大学出版社

Turbo C/Borland C++

用户界面程序设计

周升锋 李立新 孙传俊 编著

西安交通大学出版社

内容提要

本书从实例入手,详细介绍了如何用 Turbo C/Borland C 进行用户界面程序设计的一些技术和技巧。主要包括:屏幕颜色的程序设计、点阵汉字及矢量汉字的多种显示技术、弹出式窗口设计、各种菜单的设计技术、鼠标器编程、数据的输入/输出程序设计、图符及图符菜单设计、屏幕特技设计、声音及音乐程序设计、以及动画程序设计共十个部分。

书中所提供的所有源程序均在 Turbo C 2.0 和 Borland C++ 2.0 下调试通过,可直接为软件开发者所采用,对软件开发者有启迪和帮助作用。

本书适合从事计算机软件开发的工程技术人员及大中专院校师生阅读参考。

(陕)新登字 007 号

Turbo C/Borland C++

用户界面程序设计

周升锋 李立新 孙传俊 编著

责任编辑 叶 涛

*

西安交通大学出版社出版

(西安市咸宁西路 28 号 邮政编码 710049)

西安理工大学印刷厂印装

陕西省新华书店经销

*

开本 787×1092 1/16 印张:16.625 字数:401 千字

1994 年 12 月第 1 版 1994 年 12 月第 1 次印刷

印数:1——5000

ISBN7-5605-0648-8/TP·91 定 价:15.00 元
实例软盘:25.00 元

前 言

用户界面(UI)是连接计算机和操作者的纽带,操作者与计算机的信息交换就是通过一个个界面来完成的。随着计算机软件技术的发展,用户界面的设计越来越成为软件设计的重点。一个成功的软件必定有良好的用户界面。在整个软件的开发过程中,用户界面的工作量是很大的,大约要占整个软件工作量的40%—60%。从某种程度上可以说:没有用户界面也就没有了软件,用户界面的好坏,不仅关系到软件的质量,而且对一个软件的被认可程度与成功起着巨大的作用。当今的计算机软件界面越来越漂亮,令计算机软件开发人员耳目一新,非常希望了解这些界面的设计技术。目前,系统地介绍用户界面设计方面的书籍还不多见。

C语言是目前颇为流行的一种计算机语言,其代码效率高、与系统十分接近,Turbo C以其小巧、快速、高效的集成开发环境及丰富的图形功能赢得了广大计算机用户的喜爱;Borland公司1991年推出的Borland C++ 2.0比Turbo C功能更强大,提供了更加丰富的函数资源。所以用Turbo C和Borland C++开发软件的用户越来越多。

为此,作者将近几年来从事C语言开发工作积累的经验 and 知识,用Turbo C/Borland C作为编程工具,从实际出发,系统介绍用户界面程序设计的一些技术和技巧,期望对广大读者的编程能够收到事半功倍的效果。

本书内容包括:

- 屏幕颜色的程序设计:包括屏幕16种颜色的使用、改变VGA设置的16种颜色及256种颜色,半色调明暗处理技术、立体块及按钮设计;
- 汉字显示技术:讲述中西文DOS下显示彩色汉字的方法,点阵汉字与矢量汉字的显示、汉字的放大旋转与倾斜显示、小字库的建立技术、立体字与空心字、256色汉字显示技术;
- 文本窗口、图形窗口及弹出式窗口程序设计;
- 关于鼠标的程序设计,鼠标的功能及函数设计,鼠标的编程应用;
- 下拉式菜单设计:中西文DOS下的中西文下拉式菜单及弹出式菜单的设计、用鼠标选择立体按钮菜单、鼠标与键盘兼容的下拉式菜单设计、交替运行其它语言程序时的菜单设计;
- 数据的输入/输出程序设计;

- 图符的创建及图符菜单的程序设计;
- 屏幕特技设计: 包括特技清屏、整屏弹出特技、连续颜色变换技术、屏幕的淡入淡出技术以及屏幕填充图案设计;
- 声音及音乐程序设计;
- 动画程序设计

共十个部分。书中列举大量的实例,全部用汉字注释,每个例程均是作者编制并且全部在 Turbo C 2.0 和 Borland C++ 2.0 下调试通过,可直接为广大读者所采用。

所有程序收集在一张高密软盘中。

本书不是 C 语言的入门教材,但也没有采用很复杂的数据结构进行描述,只要系统地学习过 C 语言并了解最基本的 DOS, BIOS 功能调用,就可以顺利阅读本书。

在本书的整个编著过程中,得到了张建教授的大力支持和帮助,李舜酩老师、孟宪皆老师以及赵庆志老师也给予了很多的帮助。在此,特向他们表示衷心地感谢!

由于水平的关系,加之时间仓促,错误之处当请批评指正。

编著者

1994 年 8 月

目 录

第 1 章 有关的基础知识及说明

- 1.1 软件及硬件要求 (2)
- 1.2 目录结构及说明 (2)
- 1.3 图形方式设置及图形独立运行程序的建立 (2)
- 1.4 在 Turbo C/Borland C 中使用 DOS, BIOS 中断 (5)
- 1.5 关于位操作 (8)
- 1.6 关于端口操作函数 (8)

第 2 章 关于颜色的程序设计

- 2.1 颜色的基本知识与使用原则..... (11)
 - 2.1.1 颜色的类型与混合..... (11)
 - 2.1.2 颜色方程式..... (11)
 - 2.1.3 颜色的功能..... (11)
 - 2.1.4 颜色使用的基本原则..... (12)
- 2.2 VGA 图形显示适配器简介 (12)
 - 2.2.1 VGA 的显示模式 (12)
 - 2.2.2 VGA 的结构 (14)
 - 2.2.3 VGA 的寄存器 (15)
 - 2.2.4 VGA 的数模转换器 (16)
- 2.3 屏幕 16 种颜色的使用 (17)
 - 2.3.1 文本窗口颜色的设置..... (17)
 - 2.3.2 图形方式下颜色的设置..... (19)
- 2.4 改变 VGA 设置的 16 种颜色 (21)
- 2.5 半色调明暗处理技术..... (25)
- 2.6 VGA 256 色编程技术 (27)
- 2.7 改变 VGA 设置的 256 种颜色 (31)

2.8 颜色与视觉效果——立体块及按钮设计	(33)
第3章 汉字显示技术	
3.1 中文 DOS 下的汉字显示	(40)
3.1.1 利用 C 语言的库函数显示彩色汉字	(40)
3.1.2 利用 BIOS 显示彩色汉字	(41)
3.2 西文 DOS 下的汉字显示	(45)
3.2.1 点阵汉字	(46)
3.2.1.1 16 点阵汉字的显示	(46)
3.2.1.2 24 点阵汉字的显示	(48)
3.2.1.3 点阵汉字的无级放大、旋转与倾斜显示	(51)
3.2.1.4 小字库的建立	(57)
3.2.2 矢量汉字	(65)
3.2.2.1 矢量汉字的显示	(65)
3.2.2.2 矢量汉字的无级放大、旋转与倾斜显示	(70)
3.2.2.3 矢量汉字小字库的建立	(73)
3.3 立体汉字与空心汉字	(78)
3.4 256 色汉字显示	(79)
第4章 窗口程序设计	
4.1 文本窗口	(83)
4.2 图形窗口	(85)
4.3 弹出式窗口系统	(86)
4.3.1 弹出式窗口	(86)
4.3.2 弹出式窗口的改进	(92)
第5章 关于鼠标的程序设计	
5.1 鼠标的功能介绍及功能函数设计	(104)
5.1.1 鼠标的常用功能介绍	(104)
5.1.2 鼠标的基本函数设计	(105)
5.2 鼠标的应用编程	(107)
第6章 菜单程序设计	
6.1 西文 DOS 下的菜单设计	(113)
6.1.1 西文 DOS 下的西文下拉式菜单的设计	(113)
6.1.2 西文 DOS 下的汉字下拉式菜单设计	(124)
6.1.3 西文 DOS 下的汉字弹出式下拉菜单设计	(135)
6.2 中文 DOS 下的汉字下拉式菜单设计	(143)
6.3 用鼠标选择立体按钮菜单	(151)
6.4 鼠标和键盘兼容的下拉式菜单设计	(157)
6.5 交替运行其它语言程序时的菜单设计	(168)
第7章 数据输入/输出程序设计	
7.1 标准输入/输出函数	(175)

7.1.1	格式化输入输出函数	(175)
7.1.2	非格式化输入输出函数	(179)
7.2	文件的输入/输出函数	(182)
7.2.1	标准文件函数	(182)
7.2.2	非标准文件函数	(187)
7.3	可编辑的数据输入程序设计	(189)
7.4	图形方式下的数据输入	(193)
7.5	图形方式下的文本输出	(196)
7.5.1	文本输出函数	(196)
7.5.2	有关文本字体、字型和输出方式的设置	(197)
7.5.3	用户对文本字符大小的设置	(199)
7.6	格式化数据输出	(203)
7.6.1	格式化文本输出	(204)
7.6.2	文本覆盖	(204)
7.6.3	突出显示文本	(204)
7.7	西文字符的放大、旋转与倾斜显示	(207)
7.7.1	矢量字库 *.CHR 的结构分析	(207)
7.7.2	放大、旋转与倾斜	(208)
第 8 章	图符及图符菜单	(213)
8.1	创建图符	(213)
8.2	图符菜单设计	(218)
第 9 章	屏幕特技设计	(225)
9.1	特技清屏	(225)
9.2	整屏弹出特技	(228)
9.3	连续颜色变换技术	(230)
9.4	屏幕的淡入淡出技术	(233)
9.5	屏幕图案设计	(236)
第 10 章	声音及音乐程序设计	(241)
10.1	发声函数及音乐程序设计	(241)
10.2	振铃的声响技术	(243)
第 11 章	动画程序设计	(246)
附录一	书中程序出现章节对照表	(255)
附录二	本书常用的 ASCII 码表	(256)
参考文献		(257)

有关的基础知识及说明

第 **1** 章

本章内容:

- 1.1 软件及硬件要求
- 1.2 目录结构及说明
- 1.3 图形方式设置及图形独立运行程序的建立
- 1.4 在 Turbo C/Borland C 中使用 DOS, BIOS 中断
- 1.5 关于位操作
- 1.6 关于端口操作函数

1.1 软件及硬件要求

本书以 Turbo C 2.0 和 Borland C++ 2.0 为蓝本,因此,必须有一套完整的 Turbo C 2.0 或 Borland C++ 2.0。除此之外,要顺利使用本书例程,应具有以下软硬件配置:

硬件 80286 以上主机或兼容机

VGA(TVGA)彩色高分辨率(640×480)图形显示器

标准键盘

[鼠标]

软件 DOS 3.0 以上版本操作系统

UCDOS 3.0 汉字操作系统

[华光排版系统的矢量汉字库 SLP 文件]

另外,本书介绍的函数和许多语句是 C 语言的标准,即其它 C 语言也有,但作者未对它们进行区分,而叙述成 Turbo C/Borland C 的函数和语句,特此说明。

1.2 目录结构及说明

本书将 Turbo C 或 Borland C 按下面的目录结构进行叙述:

ROOT—TC(BC)	—	INCLUDE	头文件目录
	—	LIB	库文件目录
	—	BGI	BGI 和 CHR 文件目录
	—	[USER]	用户文件目录,也可直接放在 TC(BC)子目录下。

Turbo C 2.0 的扩展名为: .C; Borland C++ 2.0 的扩展名为: .CPP 或 .C(默认为 .CPP);

Turbo C 2.0 较 Borland C++ 2.0 具有更快的编译速度,但语法检查不够严格。作者建议 Borland C++ 2.0 的用户编程时先在 Turbo C 2.0 下调试程序,然后在 Borland C 2.0 下编译。

1.3 图形方式设置及图形独立运行程序的建立

本书中大部分程序是在图形方式下编制的。Turbo C/Borland C 提供了非常丰富的图形函数,所有图形函数的原形均在 graphics.h 中,使用图形函数时要确保有显示器图形驱动程序 *.BGI,同时应将集成开发环境 Options/Linker 中的 Graphics Lib 选为 on,只有这样才能保证正确使用图形函数。

使用图形函数之前,应首先将屏幕设置为图形模式,可用下面的函数:

```
void far initgraph(int far * GraphDriver,int far * GraphMode,char * Path);
```

其中,GraphDriver 和 GraphMode 分别表示图形驱动器和模式,Path 是指图形驱动程序所在的目录路径。对 VGA 显示适配器,其图形模式的符号常数及对应的分辨率如表 1-1。

表 1-1 VGA 图形驱动器、模式的符号常数及数值

图形驱动器		图形模式		色 调	分辨率
符号常数	数 值	符号常数	数 值		
VGA	9	VGALO	0	16 色	640×200
VGA	9	VGAMED	1	16 色	640×350
VGA	9	VGAHI	2	16 色	640×480
DETECT	0	用于硬件测试			

当 GraphDriver = DETECT 时,表示用于硬件测试,对 VGA 取屏幕最高分辨率 640×480。

图形驱动程序由 Borland 公司提供,文件扩展名为 BGI。EGA,VGA 图形适配器的驱动程序为 EGAVGA. BGI。

另外,Turbo C/Borland C 提供了退出图形状态的函数 closegraph(),其调用格式如下:

```
void far closegraph(void);
```

调用该函数后可退出图形状态而进入文本方式(默认方式),并释放用于保存图形驱动程序和字体的系统内存。

Turbo C/Borland C 的图形库是由 graphics. h,GRAPHICS. LIB,*.BGI,*.CHR 四部分组成的,其中 VGA 的驱动程序名为 EGAVGA. BGI。*.CHR 文件有四个,分别为:TRIP. CHR(三倍笔划字体),LITT. CHR(小号笔划字体),SANS. CHR(无衬线笔划字体)和 GOTH. CHR(黑体笔划字体),对于用 initgraph() 函数直接进行图形初始化程序,在编译和连接时并没有将相应的驱动程序(*.BGI)装入到可执行程序,当程序运行到 initgraph()语句时,再从该函数中的第三个形式参数 char * Path 中所规定的路径中去寻找相应的驱动程序。若没有驱动程序,则在 TC(BC)子目录中去找,如果 TC(BC)子目录中仍没有或 TC(BC)子目录不存在,将会出现下列错误:

BGI Error: Graphics not initialized (use 'initgraph').

而对于图形方式下的文字输出,若当前目录下无相应的 *.CHR 文件,则输出的字体及文本大小的定义将无效。

因此,为了使用方便,应该建立一个不需要驱动程序或字体文件就能独立运行的可执行图形程序,我们可用下列步骤(以 EGAVGA 为例):

(1) 将驱动程序 EGAVGA. BGI 转换成 EGAVGA. OBJ 的目标文件:

确保 BGIOBJ. EXE 在 TC(BC)子目录下,在 TC(BC)子目录下,键入:

```
BGIOBJ BGI\EGAVGA
```

将字体文件 *.CHR 转换成 OBJ 文件(以 TRIP 为例):

BGIOBJ BGI\TRIP (另外三个 CHR 文件方法相同)

(2) 将以上建立的 OBJ 文件加入到 GRAPHICS.LIB 库文件中:

在 TC(BC)子目录下,键入:

TLIB LIB\GRAPHICS.LIB+EGAVGA (也可用 PRJ,TLINK)

TLIB LIB\GRAPHICS.LIB+TRIP

说明:

如果不希望 EGAVGA.OBJ 文件和 TRIP.OBJ 等文件加在 GRAPHICS.LIB 中时,可以采用下面的方法将其从 GRAPHICS.LIB 中去除掉:

TLIB LIB\GRAPHICS.LIB-EGAVGA.OBJ

TLIB LIB\GRAPHICS.LIB-TRIP.OBJ

(3) 在程序中 initgraph()函数调用之前,加上一句:

registerbgidriver(EGAVGA __driver);

该函数告诉连接程序在连接时把 EGAVGA 的驱动程序装入到用户的执行程序当中;

对于文字字体,在调用之前,要加:

registerbgifont(符号名);

表示将程序连接时,把字体文件装入到用户程序。

有关驱动程序和字体名字如表 1-2。

表 1-2 驱动程序和字体名字表

驱动程序文件 (*.BGI)	registerbgidriver 符号名	字体文件 (*.CHR)	registerbgifont 符号名
CGA	CGA __driver	TRIP	triplex __font
EGAVGA	EGAVGA __driver	LITT	small __font
HERC	Here __driver	SANS	sansserif __font
ATT	ATT __driver	GOTH	gothic __font

经过以上处理,编译连接后的执行程序在任何目录或其它兼容机上都可运行。

作过上述几步之后,初始化屏幕为图形模式的函数可以写成:

```
void InitGra(void){
    int GraphDrive = DETECT,GraphMode;          /* 定义为 DETECT 方式 */
    registerbgidriver(EGAVGA __driver);          /* 登记的图形驱动名 */
    registerbgifont(triplex __font);              /* 使用三倍笔划字体 */
    registerbgifont(small __font);                /* 使用小号笔划字体 */
    registerbgifont(sansserif __font);            /* 使用无衬线笔划字体 */
    registerbgifont(gothic __font);               /* 使用黑体笔划字体 */
    initgraph(&GraphDrive,&GraphMode,"");
}
```

1.4 在 Turbo C/Borland C 中使用 DOS, BIOS 中断

Turbo C/Borland C 提供了一些与操作系统有关的函数,这些函数使用起来比较灵活,可实现许多高级功能。

1. DOS 软中断功能调用

DOS 的软中断(21H 中断)具有强大的功能, Turbo C/Borland C 可用函数 `intdos()` 直接访问这些系统调用,其调用格式为:

```
int intdos(union REGS * Inregs, union REGS * Outregs);
```

该函数表示用 `Inregs` 指向的联合中的内容所确定的 DOS 系统调用,执行一次 DOS 21H 中断,并将结果存入指定的联合中。该函数的头文件为 `dos.h`,联合 `REGS` 的定义如下:

```
union REGS {  
    struct WORDREGS x;  
    struct BYTEREGS h;  
}
```

其中, `WORDREGS` 和 `BYTEREGS` 的结构分别为:

```
struct WORDREGS {  
    unsigned int ax, bx;  
    unsigned int cx, dx;  
    unsigned int si, di;  
    unsigned int cflag, flags;  
}  
  
struct BYTEREGS {  
    unsigned char al, ah;  
    unsigned char bl, bh;  
    unsigned char cl, ch;  
    unsigned char dl, dh;  
}
```

下面举例说明 `intdos()` 函数的用法: 读取系统日期并显示。

在 DOS 功能调用中, 读取系统日期的子功能是 2AH:

入口参数:

AH = 2AH; 得到系统日、星期、月、年。

出口参数:

CX = 年(1980-2099)

DH = 月(1—12)

DL = 日(1—31)

AL = 星期(0=星期天; 1=星期一等)

下面是利用 Turbo C 的 `intdos()` 函数写成的源程序:

```

/*
    RDSYSDT1.C —— 读取系统日期并输出,用 intdos()函数
*/
#include "dos.h"
#include "conio.h"
int main()
{
    union REGS In,Out;          /* 定义结构变量 */
    In.h.ah = 0x2a;             /* 设置 DOS 21H 类中断功能调用 2AH */
    intdos(&In,&Out);            /* 执行中断 */
    printf("%4d 年%2d 月%2d 日\n",Out.x.cx,Out.h.dh,Out.h.dl);
    printf("星期%1d\n",Out.h.al);
    getch();
    return 0;
}

```

2. 关于 BIOS, DOS 软中断调用的函数 int86()

BIOS 除了打印机(17H 中断)和键盘(16H 中断)之外,还有显示器驱动(10H 中断)、软盘服务中断(13H)等。Turbo C/Borland C 提供了关于这些中断的函数 int86(),其调用格式为:

```
int int86(int intnum, union REGS * In, union REGS * Out);
```

其中,intnum 为 BIOS 的软中断,联合 REGS 与上节中所述相同。该函数用 In 指向的联合变量中的内容所确定的系统调用,执行一次 intnum 规定的软中断,并将结果放入 Out 指向的联合中,该函数的头文件为 dos.h。

只要将 intnum 定义为 0x21,则该函数与 intdos()函数功能相同。

将上例中的函数 intdos()改用 int86(),即变成:

```

/*
    RDSYSDT2.C —— 读取系统日期并输出,用 int86()函数
*/
#include "dos.h"
#include "conio.h"
int main()
{
    union REGS In,Out;          /* 定义结构变量 */
    In.h.ah = 0x2a;             /* 设置 DOS 21H 类中断功能调用 2AH */
    int86(0x21,&In,&Out);        /* 执行中断 */
    printf("%4d 年%2d 月%2d 日\n",Out.x.cx,Out.h.dh,Out.h.dl);
    printf("星期%1d\n",Out.h.al);
    getch();
    return 0;
}

```

3. 本书中用到的 DOS 及 BIOS 中断

(1) 输出字符功能 09H(INT 10H)

入口参数:

AH = 09H

BH = 页号

CX = 字符数

AL = 输出的字符

BL = 输出字符的属性

出口参数: 无。

(2) 写图形象素功能 0CH(INT 10H)

提供一种在图形模式中与设备独立的,但很慢的处理象素的方法。

入口参数:

AH = 0CH;

BH = 页面号

AL = 颜色

CX = 横坐标 X

DX = 纵坐标 Y

执行: INT 10H

出口参数: 无。

(3) 读屏幕上点(x,y)图形象素的功能

入口参数:

AH = 0DH

CX = x

DX = y

出口参数: AL = 象素值(范围取决于图形方式)

(4) 读 VGA 设置的颜色功能 1015H(INT 10H)

入口参数:

AH = 10H

AL = 15H

BX = 颜色号

执行: INT 10H

出口参数:

DH = 红色分量

CH = 绿色分量

CL = 蓝色分量

(5) 设置 VGA 初始设置的颜色 1010H(INT 10H)

入口参数:

AH = 10H

AL = 10H

BX = 颜色号

DH = 红色分量

CH = 绿色分量

CL = 蓝色分量

执行: INT 10H

出口参数: 无。

1.5 关于位操作

C 语言与其它高级语言的不同是它完全支持按位运算符。这与汇编语言的位操作有些相似。

Turbo C/Borland C 中按位运算符如表 1-3。

表 1-3 按位操作运算符

操作符	作用
&	位逻辑与
	位逻辑或
^	位逻辑异或
~	位逻辑反
>>	右 移
<<	左 移

按位运算是对于字节或字中的实际位进行检测、设置或移位,它只适用于字符型和整型变量以及它们的变体,对其它数据类型不适用。

位移运算符“>>”和“<<”是指将变量中的每一位向右或向左移动,其通常形式为:

右移: 变量名 >> 位的位数

左移: 变量名 << 移位的位数

经过移位以后,一端的位被“挤掉”,另一端空出的位以 0 填补。所以, Turbo C/Borland C 中的移位不是循环移动的。

1.6 关于端口操作函数

Turbo C/Borland C 可以对端口操作,其中包括: 从端口读入数据,写数据到端口。

```
int inport(int portid);
```

该函数从参数 portid 指定的输入硬件端口读入一个字的低字节,从 portid+2 中读入高

字节。

```
unsigned char inportb(int portid);
```

该函数从参数 portid 指定的输入硬件端口读入一个字节,函数的原型在 dos.h。

```
void outport(int portid,int value);
```

该函数将给定字 value 的低字节写在 portid 所指的输出端口中,高位字节写在 portid+1 所指定的输出端口中。

```
void outportb(int portid,unsigned char value);
```

该函数将给定的字节值 value 输出到 portid 所指定的输出端口中。