C语言与图形处理

严桂兰　刘甲耀　编著

华东化工学院出版社

(沪)新登字 208 号

责任编辑 欧聿红
责任校对 黄黎峥

C 语言与图形处理
C Yuyan yu Tuxing Chuli
严桂兰 刘甲耀 编者
华东化工学院出版社出版
(上海市梅陇路 130 号)
新华书店上海发行所发行
浙江上虞科技外文印刷厂排版
上海长阳印刷厂印刷
开本 850×1168 1/32 印张 8 字数 213 千字
1993 年 3 月第 1 版 1993 年 3 月第 1 次印刷
印数 1—5000 册

ISBN 7-5628-0269-6/TP·24 定价:6.60 元

目　录

第一章　图形模式与图形驱动程序

应用 Turbo C图形系统对图形进行处理，首先要了解图形模式与图形驱动程序。因此，本章叙述三个问题：

(1) 图形显示器与图形模式；

(2) 图形驱动程序与图形适配器；

(3) 图形驱动程序和图形模式的检查与登记。

§1-1　图形显示器与图形模式

一、图形显示器

用计算机进行图形处理，亦即要在显示器上有图形输出显示，则必须配置有图形适配器，对于仅配有单模式、单色的屏显适配器的系统，不可能显示图形，而只能显示文本。

图形显示器的可显示模式有两种，即文本模式和图像模式。文本模式的显示，在一般的 Turbo C 语言中均有介绍，可参阅文献[2]。

常用的显示器的性能如表 1-1 所示。

表 1-1　常用显示器的性能

适配器类型	分　辨　率	颜色(种类)
CGA(Color Graphics Adapter)	320×200 640×200	4 2
Herclus(Herclus monochrome graphics adapter)	720×348	1
EGA(Enhanced Graphics Adapter)	640×200 640×350	16 16
VGA(Video Graphics Array)	640×200 640×480	16 16

二. 图形模式

1. 分辨率

在图形模式下，整个屏幕均按显示器的分辨率分成点阵，如表1-1所示。

2. 坐标

在图形模式下，左上角为(0,0)，x 轴是从左到右(0～639)，y 轴是从上到下(0～349，对EGA)。

3. 图形窗口

在图形模式下，可以用 Turbo C 提供的窗口定义函数来定义任意位置和大小的长方形区域，称之为图形窗口或视口。当图形程序作图时，仅在窗口内有效，而在窗口之外的屏幕是不可接触的。所有图形输出均与当前窗口有关。而窗口不一定是整个屏幕，如果没有定义窗口则就是整个屏幕。

4. 窗口坐标

除图形窗口定义的函数是绝对坐标之外，其他函数的坐标均是相对于当前图形窗口的左上角(0,0)而言。

三、图形模式的编程准备

为了绘制各式各样的图形，Turbo C 提供了 70 多个图形函数(这些函数定义在 graphics.h 中，包含在 graphics.lib 中)、各种显示器的图形驱动程序的 obgi 文件以及几种笔划型字符模的 ·chr 文件，要正确地编译和连接使用图形函数的程序，必须在 options 菜单的 linker 的子菜单中选择 graphics lib on。

§1-2 图形驱动程序与图形适配器

一、图形驱动器符号常数与规定值

Turbo C 提供了几种图形驱动程序可支持的图形适配器，其符号常数与规定值如表 1-2 所示。

表 1-2 图形驱动器符号常数与规定值

符号常数	数 值	含 义
DETECT	0	根据硬件测试结果,自动装入相应驱动程序
CGA	1	CGA 显示器
MCGA	2	Multi Color Graphics Array 显示器
EGA	3	EGA 显示器
EGA64	4	EGA64 显示器
EGAMONO	5	EGA 单色显示器
IBM8514	6	IBM8514 显示器
HERCMONO	7	Herclus 显示器
ATT400	8	AT&T400 行图形显示器
VGA	9	VGA 显示器
PC3270	10	PC 3270 显示器

二、图形模式的符号常数

对于同样的图形显示器还有几种可选择的图形模式，可能是不同的分辨率或不同的颜色设置等等，图形模式的符号常数如表 1-3 所示。

表 1-3 图形模式的符号常数

适 配 器	图形模式	模式值	列×行	色调	页 数
CGA	CGAC0	0	320×200	C0	1
	CGAC1	1	320×200	C1	1
	CGAC2	2	320×200	C2	1
	CGAC3	3	320×200	C3	1
	CGAHI	4	640×200	2色	1
MCGA	MCGAC0	0	320×200	C0	1
	MCGAC1	1	320×200	C1	1
	MCGAC2	2	320×200	C2	1

续表

适配器	图形模式	模式值	列 × 行	色调	页 数
	MCGAC3	3	320×200	C3	1
	MCGAMED	4	640×200	2色1	1
	MCGAHI	5	640×480	2色1	1
EGA	EGALO	0	640×200	16色	4
	EGAHI	1	640×350	16色	2
EGA64	EGA64LO	0	640×200	16色	1
	EGA64HI	1	640×350	4色	1
EGAMON	EGAMONHI	0	640×350	2色	1(卡中有64K)
				2色	4(卡中有256K)
HERC	HERCMONOHI	0	720×348	2色	2
ATT400	ATT400C0	0	320×200	C0	1
	ATT400C1	1	320×200	C1	1
	ATT400C2	2	320×200	C2	1
	ATT400C3	3	320×200	C3	1
	ATT400MED	4	640×200	2色	1
	ATT400HI	5	640×400	2色	1
VGA	VGALO	0	640×200	16色	4
	VGAMED	1	640×350	16色	2
	VGAHI	2	640×480	16色	1
PC3270	PC3270HI	0	720×350	2色	1
IBM8514	IBM8514LO	0	640×480	256色	
	IBM8514HI	1	1024×768	256色	

三、颜色的符号常数与调色数

1．颜色的符号常数

关于颜色的设置，Turbo C 有相应的函数，有关颜色设置的符号常数(函数 Setpalette 的颜色编号)如表 1-4 所示。

表 1-4 颜色的符号常数

颜色	CGA(仅用作背景颜色)		EGA/VGA	
	符号名	数值	符号名	数值
黑	BLACK	0	EGA-BLACK	0
蓝	BLUE	1	EGA-BLUE	1
绿	GREEN	2	EGA-GREEN	2
青	CYAN	3	EGA-CYAN	3
红	RED	4	EGA-RED	4
洋红	MAGENTA	5	EGA-MAGENTA	5
棕	BROWN	6	EGA-BROWN	20
洋灰	LIGHTGRAY	7	EGA-LIGHTGRAY	7
深灰	DARKGRAY	8	EGA-DARKGRAY	56
浅蓝	LIGHTBLUE	9	EGA-LIGHTBLUE	57
浅绿	LIGHTGREEN	10	EGA-LIGHTGREEN	58
浅青	LIGHTCYAN	11	EGA-LIGHTCYAN	59
浅红	LIGHTRED	12	EGA-LIGHTRED	60
浅洋红	LIGHTMAGENTA	13	EGA-LIGHTMAGENTA	61
黄	YELLOW	14	EGA-YELLOW	62
白	WHITE	15	EGA-WHITE	63

2. 调色数

CGA 显示器有 C0～C3 四种模式可供选用（用于 CGA 的低分辨率显示），可有四种颜色，但不同的模式可使用不同的颜色组合，因而称之为调色数，而背景颜色可从表 1-4 所示的颜色列中任选一种。

CGA 的调色数（调色板）的色调符号常数（即调色板与颜色值）如表 1-5 所示。

表 1-5 CGA的色调符号常数

调色数	赋予点的颜色			
	0	1	2	3
C0	背景	浅绿	浅红	黄
C1	背景	浅青	浅洋红	白
C2	背景	绿	红	黄
C3	背景	青	洋红	浅灰

§1-3 图形驱动程序和图形模式的检查与登记

一、图形驱动程序和模式的检查

1。检查硬件配置确定所用的图形驱动程序和模式

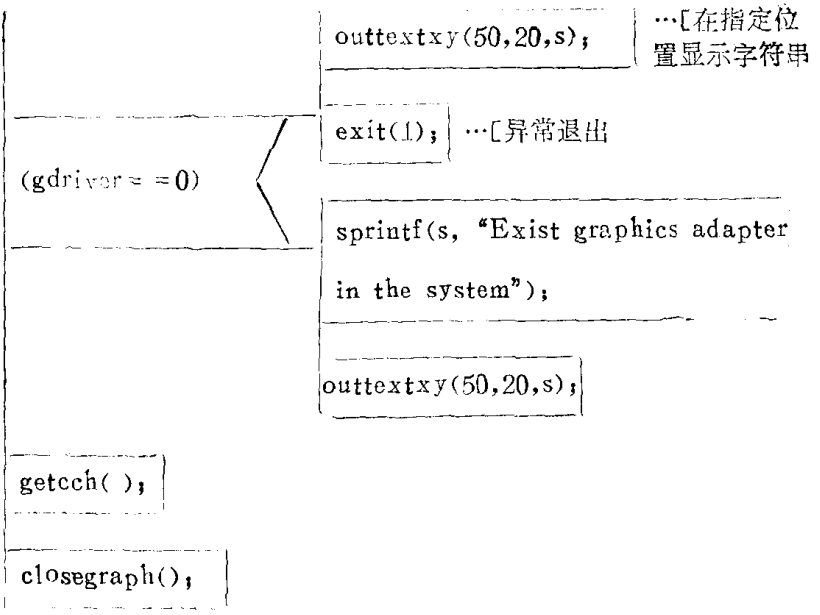
void far detectgraph (int far *graphdriver, int far *graphmode);

函数 detectgraph( )在计算机有图形适配器的情况下，确定图形适配器的类型，若系统有图形适配器，则返回适合于适配器的图形驱动器代码，用 graphdriver 所指向的整型量表示，该函数把 graphmode 所指的变量设置为适配器所能支持的最高分辨率。若系统无图形适配器，则 graphdriver 所指的变量为－2。

【例 g1-1】 检查系统是否有图形适配器

(1) PAD

```
#include "graphics.h"
main( )
char s[80];
int gdriver=DETECT, gmode;
initgraph(&gdriver, &gmode, "A:\LIB");   …[图形系统初始化
detectgraph(&gdriver, &gmode);           …[检查是否有图形适配器
                sprintf(s,"No graphics
                adapter in the system");  …[格式化输出到s
```



(2) PAD⇒程序

```
#include "graphics.h"
main()
{
    char s[80];
    int gdriver=DETECT, gmode;
    initgraph(&gdriver,&gmode,"A:\LIB");
    detectgraph(&gdriver,&gmode);
    if (gdriver==0)
    {
        sprintf(s,"No graphics adapter in the system");
        outtextxy(50,20,s);
        exit(1);
    }
    else
    {
        sprintf(s, "Exist graphics adapter in the system");
```

```
        outtextxy(50,20,s);
    }
    getch();
    closegraph();
}
```

(3) 输出结果

Exist graphics adapter in the system

2. 检查当前的图形模式

int far getgraphmode(void);

函数 getgraphmode( )返回当前的图形模式，其返回值关系到当前的视频驱动器，返回值将是表 1-3 所示值之一（定义在 graphics.h中）。

例如，下列语句将显示有关图形驱动器当前图形模式的代码：

【例 g1-2】 显示有关图形驱动器当前图形模式的代码

(1) PAD

```
#include"graphics.h"

main()

char s[80];

int gdriver, gmode, "A:\LIB"();

sprintf(s,"graphics mode is%d", getgraphmode());

outtextxy(50,20,s);

getch();

closegraph();
```

(2) PAD⇒程序

```
#include "graphics.h" main()
{
    char s[80];
    int gdriver = DETECT, gmode;
    initgraph (&gdriver,&gmode, "A:\LIB");
    sprintf(s,"graphics mode is %d",getgraphmode());
    outtextxy(50,20,s);
    getch();
    closegraph();
}
```

(3) 输出结果

```
graphics mode is 4
```

二、登记连接时所用的图形驱动器代码

int registerbgidriver(void(*driver)(void));

函数 registerbgidriver() 告诉图形系统，图形驱动器已被连接，不必再寻找相应的磁盘文件。

三、登记连接时所用的字体

int registerbgifont(void(*font)(void));

函数 registerbgifont()告诉图形系统，字体已被连接，不必再寻找相应的磁盘文件。

第二章　图形系统的初始化与图形模式的控制

要应用 Turbo C 图形系统对图制进行处理，首先就要启动图形系统，即图形系统初始化，处理工作完毕还得关闭图形系统。另外，在处理过程中可能还要对图形模式进行控制、设置图形缓冲区和分配图形内存。因此，本章叙述四个问题：

(1) 图形系统的初始化；

(2) 图形系统的关闭；

(3) 图形模式的控制；

(4) 图形缓冲区的设置以及图形内存的分配与释放。

§2-1　图形系统的初始化

由于图形显示器的种类繁多，其控制方式也各有所异，所以要显示图形就需先装入相应的驱动程序。另外，由于每种图形显示器又各有几种不同的图形显示模式，因此，要显示图形，就必须确定所用的显示模式，也就是说，要显示图形，需先装入相应的驱动程序和决定所用的显示模式，这就是图形系统初始化，亦即启动图形系统。为了从盘上装入图形驱动程序，并对图形系统进行初始化，Turbo C 提供了专门的函数：

```
void far initgraph (int *graphdriver, int far
    *graphmode, char far *path);
```

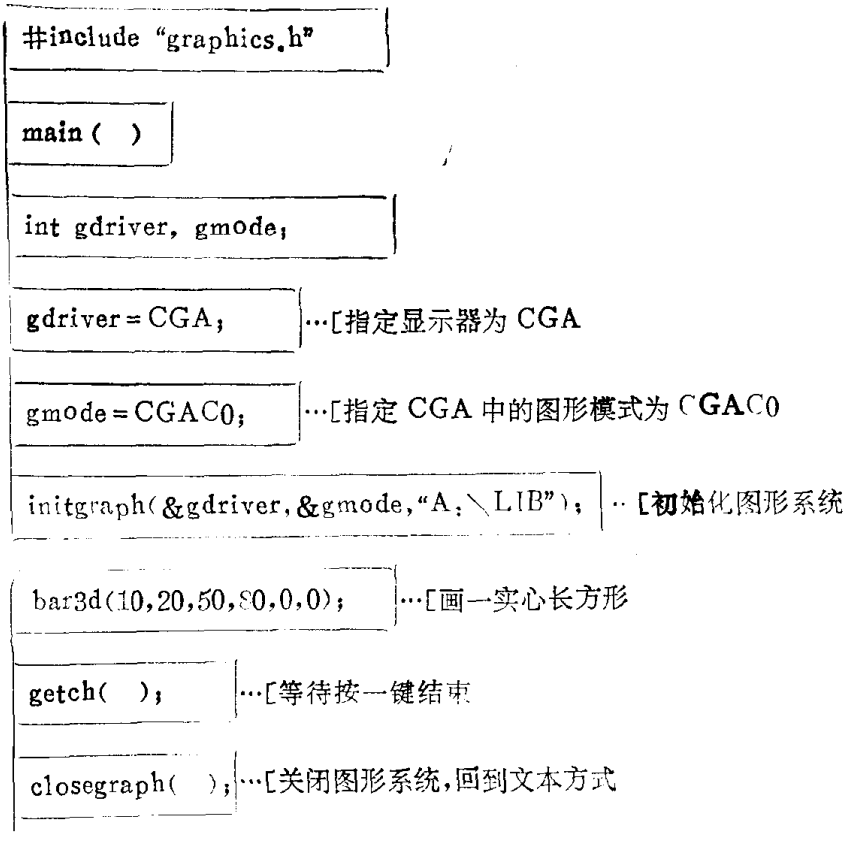
其中，graphdriver 和 graphmode 分别表示图形驱动器和图形模式，它们定义在 graphics.h 中，其有关常数如表 1-2 和表 1-3 所示；path为图形驱动器所在的路径名，即图形驱动程序放在哪个目录中，可简单地使用“”来表示在当前目录下寻找驱动程序(BGI)。

一、一般初始化处理

若已知所用图形显示器的种类和所用的图形模式，则图形系统初始化很简单。

【例 g2-1】

(1) PAD



(2) PAD⇒程序

```
#include "graphics.h"
main( )
{
        int gdriver, gmode;
        gdriver=CGA;
        gmode=CGAC0;
```

```
        initgraph(&gdriver,&gmode,"A:\LIB");
        bar3d(10,20,50,80,0,0);
        getch( );
        closegraph( );
}
```

(3) 输出结果

(4) 注意事项

(1) 有图形显示器的程序，在链接时并不装入图形驱动程序，只是当该程序运行到初始化函数 initgraph( )时，才到 path 中去寻找相应的图形驱动程序，并将其装入内存，然后再运行，这是 Turbo C 的规定。

(2) 当所编的程序不知道将用什么样的显示器或要用于不同的图形显示器时，可用 §1-3 所介绍的测试函数 detectgraph( ) 来测定。

【例 g2-2】

(1) PAD