为FoxBASE创建绘图功能的捷径

淮南新庄孜矿监测中心 马林

图形较数据而言更具有直观性，对于一个应用软件，是否具有良好的图形处理功能，是用户和软件开发者都极为重视的一个问题。目前许多数据库管理软件都栖息在 FoxBASE 和 dBASE 系统下，FoxBASE 和 dBASE 都存在着不支持图形处理功能的缺陷，使得图形处理上很不方便。

通常利用 RUN 命令运行用其他高级语言编写的绘图程序，但 RUN 命令存在着内存开销大、数据通讯繁杂等问题。也有介绍用 GWBIOS 3.00 中提供的虚拟图形设备驱动文件 GRD. SYS 或是直接调用 GWBIOS 3.00中显示输出模块的30H～36H 功能而实现图形处理功能的，依赖于微机的硬件设置，对于非长城系列微机此法就行不通。

一、方案选择

利用汇编语言编写图形处理程序供 FoxBASE 调用不失为一个方法，但这要求对微机有全面系统的了解，而且程序设计相当复杂，远非一般用户所能掌握。

对于其他 PC 兼容机是否如同长城系列微机一样有现成的图形包可供利用呢?我们知道 BIOS 中只提供了屏幕绘点例程，显然图形处理仅此一项功能是不够的。拓宽视野，把着眼点转移到微机支持的外部高级语言上，或许会有帮助。

目前普遍使用的程序设计语言中，支持图形功能的也有不少。其中 Turbo pascal 以其众多的优点，倍受用户的青睐。在3. 0版本中，增添了图形处理功能，其中有画点、线、图、弧、涂色、保存图形、输出图形、以指定“纹理”（即点的模式）填充矩形块等子程序。值得一提的是，Turbo Pascal 3. 0强，绘图处理是作为外部模块调用的。其内容放在 GRAPH. P 和 GRAPH. BIN 两个文件中，GRAPH. P 实际上只是定义 GRAPH. BIN 中的外部机器码例程，分析 GRAPH. BIN 中各个子程序或函数的入口参数及入口地址（用户可将 GRAPH. BIN 清单打印出来，以备查阅），而 GRAPH. BIN 才是真正的绘图执行代码。分析 GRAPH. BIN 可知，它上一可浮动程序，也就是说可以将其嵌入执行程序的任何位置。利用这一特性，我们就可在 FoxBASE 状态下，调用汇编子程，再由汇编子程调用 FRAPH. BIN 中的图形处理例程，而实现屏幕绘图功能。

二、具体实现

FoxBASE 提供了二进制程序的调用功能，通过下面三条语句完成加载、调用和清除汇编子程：

LOAD＜二进制文件名＞

CALL＜模块名＞[WITH＜字符串表达式＞/＜内存变量＞]

RELEASE MODULE＜模块名＞

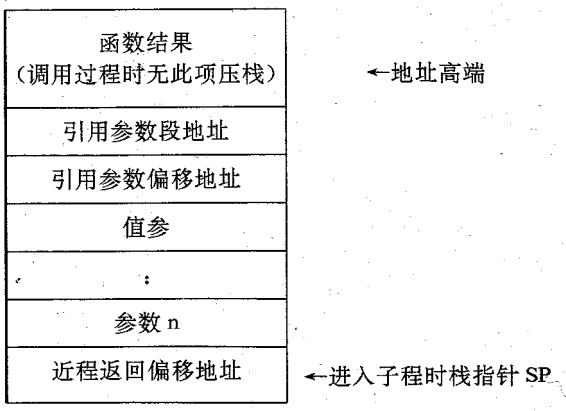
在用 WITH 选择项传递参数时，传递变量虽然可以是任意类型，本例中以字符串形式传递更为简洁有效。因为在 FoxBASE 中运用 STR ()、VAL () 函数可以方便地进行 C、N 型变量之间的转换，从而避开了繁杂的浮点运算，且通过字符串的合并运算可实现多个参数的传递。

可在 FoxBASE 下构造统一格式字符串，形式如下：

“命令号，参数1，参数2，……，参数 n”

进入 CALL 模块时，DS：BX 指向由 WITH 传递字符串参数的第一字节，汇编子程完成类型转换，而后根据命令号转 GRAPH. BIN 中相应子程。

调用 GRAPH. BIN 中过程工函数时，参数是通过堆栈进行传递的。在编写调用 Tubro Pascal 子程的汇编程序之前，需要对 Pascal 调用外部过程或函数时，参数进栈的情况有清楚的了解。以下是 Pascal 调用外部过程或函数时压栈的情况：



如果形式参数是值参，那么参数的实际值压入栈中，如果是引用参数（VAR 说明的），则参数的地址（段地址和偏移地址）被压入堆栈。利用汇编程序，模拟 Turbo Pascal 调用时的压栈操作，就可正确地调用 GRAPH. BIN 中的子程。

根据以上原则我们编写了 FOXGRP. ASM（程序清单附后），PARAMETER 子程序的功能是将数值字符串转换为十六位二进制数。数值字符串中可包含正负号和小数点，对小数自动四舍五入取整。GRAPH. BIN 中有几个很具有价值的函数，调用时要求函数值返回到 FxoBASE 下，在程序设计中，就要区分开调用的是过程还是函数。这里我们采用保存栈指针的方法，前面已看出函数的压栈情况与过程不同，另一方面，保存栈指针而后恢复之，可以防止用户因给定参数项个数错误而引起的堆栈混乱，致使程序向不可预料。GRAPH. BIN 中函数返回时，值在 AX 寄存器中，仍然要将其转换字符串形式并存回到 DS：BX 所指的传递数据区。由于 FoxBASE 要求不得改变由 CALL WITH 语句传递的以 DS：BX 为指针的内存变量的长度，应尽量保证传递的字符串变量的长度要大于反馈函数值的长度，若反馈数值字符串超长，将会覆盖数据区的其他数据。

FOXGRP. ASM 编辑完成后，经 MASM 编译，LINK 链接，EXE2BIN 转换成 FOXGRP. BIN，还要利用 DOS 的 COPY 命令（一定要有/B 选择项）将其与 GRAPH. BIN 合并而成一新文件，可执行：

COPY /B FOXGRP. BIN+GRAPH. BIN FOXGRP. BIN

三、命令简介

有关详细介绍，请参阅科海培训中心出版的《TURBO PASCAL 3. 0 参考手册》一书的第十九章，这里只做一简单列表：

置320×200单色图形模式　“0”
置320×200彩色图形模式　“1”
置640×200单色图形模式　“2”
640×200模式颜色　“3，Color”
设置调色板　“4，Palette”
设置背景颜色　“5，BackgroundColor”
图形窗口　“6，X1，X1，Y2，Y2”
画点　“7，X，Y，COLOR”
画线　“8，X1，Y1，X2，Y2，COLOR”
设置颜色转换表　“9，C1，C2，C3，C4”
画弧　“10，X，Y，Angle，Radius，Color”
画圆　“11，X，Y，Radijus，Cloir”
保存图象　“12，Segment，Offset，X1，Y1，X2，Y2”
输出图象　“13，Segment，Offset，x，y”
取点的颜色（函数）　“14，X，Y”
填充现行窗口　“15，Color”
区域填充　“16，X，Y，FillColor，BorderCloir”
以 Pattern 模式填充矩形　“17，X1，Y1，X2，Y2，Color”
设置 Pattern 模式填　“18，P1，P2，P3，P4，P5，P6，P7，P8”
龟回移　“19，Distance”
清除现行窗口　“20”

龟前移	“21，Distance”
取龟的方向（函数）	“22”
隐藏龟	“23”
龟复位	“24”
定义不环绕	“25”
笔放下	“26”
笔抬起	“27”
设置龟方向	“28，Angle”
置笔的颜色	“29，Color”
移动龟	“30，X，Y”
显示龟	“31”
龟向左转	“32，Angle”
龟向右转	“33，Angle”
龟延迟	“34，Delay”
龟图窗口	“35，X，Y，Width，Hight”
龟在窗口中可见（逻辑函数）	“36”
定义环绕	“37”
龟的 X 坐标（函数）	“38”
龟的 Y 坐标（函数）	“39”

四、结束语

该方法仅利用5KB 字节就完成了近40个过程和函数的调用，模块可随时调入或撤消，节省内存开销，同时参数传递极为方便。我们已成功地将其运用到我单位开发的应用软件中，绘制了好几种统计图形，且图文同时显示使用户界面也得到了改善。

上述程序在 dBASE Ⅲ Plus 中同样可以使用。

FOXGRP. ASM 程序清单：

```
CSSEG   SEGMENT
        ASSI，E         CS：CSSEG
MAIN    PROC            FAR
        PUSH            DS                  ；保存 FoxBASE 数据段
        PUSH            BX
        OR              BX，BX              ；检查有无参数？
        JZ              END—RUN             无 WITH 参数则返回
        MOV             CS：Save—SP，SP     保存堆栈指针
        CALL            PARAMETER           ；计算第一参数（即命令号）
        MOV             SI，CX
        CMP             SI，39              ；命令号超出39？
        JG              END—RUN
        CMP             SI，18              ；为 Pattern 命令号？
        JZ              PTN—PROC
NO—More：
        MOV             AX，SI              ；
        SHL             AX，1               ；命令号＊3得到 GRAPH. BIN
        ADD             SI，AX              ；中相应入口地址
        ADD             SI，OFFSET Graphics ；再加上修正偏移量
        CALL            SI                  ；调用对应命令子程
        CMP             SP，CS：Save—sp     ；检测堆栈指针有无改变？
        JZ              END—RUN             ；栈指针改变，表示有返回参数
        MOV             SP，CS：Save—SP     ；恢复堆栈指针
```

```
        POP     BX
        POP     DS
        XOR     CX, CX
More—Digit:
        XOR     DX, DX
        PUSH    CX
        MOV     CX, 10
        DIV     CX                      ; AX/10: AX=商  DX=余数
        POP     CX
        ADD     DL, 30                  ; 余数化为十进制 ASCLL 码字符
        PUSH    DX                      ; 字符压栈中
        INC     CX                      ; CX 压栈计数+1
        CMP     AX, 0                   ;
        JNZ     More—Digit
Digit—POP
        POP     AX                      ; 字符出栈
        MOV     [BX], AL                ; 存入参数传递区
        INC     BX
        LOOP    Digit—POP
        MOV     [BX], AH                ; 置结束标记 (NULL)
        JMP     RUN—POP
END—RUN:
        POP     BX
        POP     DS                      ; 恢复 FoxBASE 数据段
RUN—RET:
        RET                             ; 远程返回
MAIN    ENDP
PTN—PROC:
        PUSH    SI
        MOV     SI, OFFSET PATTERN
More—PNT:
        CMP     AL, 0                   ; 参数项结束否?
        JZ      NO—More—PNT
        CALL    PARAMETER               ; 计算随后的参数
        MOV     CS:[SI], CL             ; 存入 PATTERN 数据区
        INC     SI
        CMP     SI, OFFSET PNT—DATA—END     ; 超过8字节否?
        JB      More—PNT
NO—More—PNT:
        POP     SI
        MOV     AX, OFFSET PATTERN
        PUSH    CS                      ; PATTERN 数据段地址入栈
        PUSH    AX                      ; PATTERN 数据偏移地址入栈
        JMP     No—More
; ..................................
        Save—SP     DW          0
        PATTERN     DB          8 DUP (0)
        PNT—DATA—END LABEL      NEAR
;
;   ENTRANCE:    DS: BX=ADDRESS OF NUMBER STRING
;   EXIT:        AL=THE LAST CHAR OF A NUMBER STRING
;                CX=VALUE OF A NUMBER STRING
;   REGSTER:           SI        UN—CHANGE
;
```

```
PARAMETER       PROC        NEAR
        XOR     CX, CX
        XOR     DI, DI      ;"+","-" 正负号出现标记
        MOV     DX, CX      ;"." 小点出现标记
NextChar:
        MOV     AL, [BX]
        INC     BX
        CMP     AL, " "     ; 为 SPACE?
        JZ      NextChar
        CMP     AL, "."     ; 为小数点?
        JNZ     NumberChar
        MOV     AL, [BX]
        INC     CX
NIINC:
                INC     DX
        JMP     NextChar
NumberChar:
        CMP     AL, "0"
        JB      NotNumber
        CMP     AL, "9"
        JA      NotNumber
        OR      DL, DL
        JNZ     NextChar
        AND     AX, 000FH   ; 屏蔽去高位，变为数值
        PUSH    AX
        MOV     AX, 10
        MUL     CX
        POP     CX
        ADD     CX, AX      ; CX 为字符串数值
        JMP     NextChar
NotNumber:
        CMP     AL, "+"
        JNZ     IsMinus
        XOR     DI, DI
        JMP     NextChar
IsMinus:
        CMP     AL, "-"
        JNZ     TransRet
        MOV     DI, 01
        JMP     NextChar
TransRet:
        CMP     DI, 0
        JZ      PlusOn
        NEG     CX
PlusOn:
        RET
PARAMETER       ENDP
; ..................................
Graphics LABEL NEAR; GRAPH. BIN 入口修正偏移量
CSSEG ENDS
        END     MAIN
```