FoxBase+下打印机控制命令的发送

宁波市机械工业局计算机室　　朱孟海

一、问题的提出

在个人计算机数据库管理系统中，FoxBase+以其极快的处理速度、高效的编程语言，与dBASEⅢ数据库管理系统完全兼容，更加强大的处理功能，有良好可移植性和易学易用等特点而深受广大用户青睐。

但是在FoxBase+(包括dBASE系列)中，由于系统采用以ASCⅡ值为0的字符作为字符串或字符串变量的结束标识，因此在采用以FoxBase+作为程序设计语言来编制管理信息系统时，ASCⅡ码为0的字符就不能被系统所识别(也就是说系统把它作为字符串的结束符而"吃掉")，这种情况在某些场合下是不正常的。

如EPSON LQ-1600K打印机控制命令集中，为了设置0字距就得向打印机发送以上指令"28 83 0 0"，要置双向打印则发送"27 85 0"等命令。在FoxBase+程序中，为了能够正确地控制打印机的操作，ASCⅡ码为0的字符必须被解释成特殊字符CHR(0)，而不能被FoxBase+系统所"吃掉"，否则就不能达到程序员所预期的效果。这正是本文需要解决的问题。

二、设计思想

在FoxBase+中，系统向我们提供了调用汇编语言子程序的方法，这样使得在FoxBase+下，通过二进制模块正确地向打印机发送控制命令成为实现。

FoxBase+提供了调入、执行和释放二进制模块的命令，具体如下：

- LOAD＜二进制模块文件名＞
- CALL＜二进制模块文件名＞[WITH＜参数＞]
- RELESSE MODULE ＜二进制模块文件名＞

对于各条命令的意义不再多述。这里主要讲一下CALL命令，CALL命令能够运行已装入内存的二进制模块，它可以带有参数，该参数的起始地址由DS:BX寄存器对来指定。如果没有参数，则BX寄存器的内容为0。二进制程序通过参数与FoxBase+相互交换信息，参数可以是FoxBase+系统认可的任意类型的内存变量，其中以字符型表达式最为常见，也最便于存取。当参数为字符型表达式时，在二进制程序中可以检查BX寄存器的内容是否为零来判别的参数串结束与否，若BX的值为零则表示参数已经结束。

正是因为FoxBase+向我们提供那样的办法，才有可能利用汇编语言子程序来正确地向打印机传送控制命令。本文所述方法，参数一律采用十进制形式，各参数之间以逗号","分隔，其意义为控制打印机的一条指令。

三、使用环境

硬件环境：IBM PC／XT、AT、长城系列及其它兼容机

软件环境：中／英文DOS 2.0及以上版本运行、FoxBase+2.0或2.1、dBASEⅢ PLUS

四、注意事项

众所周知，打印机的控制命令随着执行过不同的汉字打印机驱动程序后而改变，上面讲的EPSON LQ-1600K打印机控制命令指的是打印机本身所拥有的命令，对于不同的驱动程序请参考有关说明。

五、源程序清单

```
;文件名：MODEPRN.asm
;
;调用规则：
;       在FoxBase+中如下使用
;       LOAD MODEPRN
;       ......
;       CALL MODEPRN WITH ＜参数＞
;       例如LQ--1600K打印机可用CALL MODEPRN
WITH
```

```
        '27,43,45,13'来设置 1/8 英寸行距
;           type    MODEPRN.asm
;
code_seqment    segment byte
                assume cs:code_segment
begin           proc far
                jmp start
;           =====数据区======
save_ds     dw      0
save_bx     dw      0
counter     db      0; 传送过来的参数个数
end flag    db      0; 参数结束标识: 1--已结束,0--
                    未结束
buffer      db      254 dup(0) ; 参数保存区
            db      0
;           ----------
start :
;           -----保护调用现场-----
            mov     word ptr cs:save_ds, ds
            mov     word ptr cs:save_bx, bx
;           -------传参数--------
            mov     ax,cs
            mov     es,ax
            lea     di,cs:buffer
            mov     cs:counter,0
            mov     cs:end_flag,0
load_paameter:
            call    filter
            cmp     cs:end_flag,0
            jne     next_step
            call    convert
            mov     byte ptr es:[di],al
            inc     di
            inc     cs:counter
            mov     cs:end_flag,0
            je      load_paraneter
next_step:
            mov     cs:counter,0 ;无调用参数
            je      exit_print
;           ------测试打印机状态-------
test_printer:mov    ah,02h
            xor     dx,dx
            int     17h
            cmp     ah,90h ;打印机正常
            jnz     exit_print
ready_printer:mov   ax,cs
            mov     dx,ax
mode_printer:xor ch,ch
            mov     cl,counter
            lea     si, buffer
print_a_char:
            xor     dx,dx
            xor     ah,ah
            mov     al, byte ptr[si]
            int     17h
            inc     si
            loop    print_a_char
exit_print:
            mov     ds, word ptr cs:save_ds
            mov     bx, word ptr cs:save_bx
            ret
filter      proc ; 过滤空格、逗号
filt_space: mov     al, byte ptr [bx]
            cmp     al ,','
            je      meeting_comma
            cmp     al, 00h
            je      end_continue
            cmp     al,''
            jne     return_filter
            inc     bx
            jmp     filt_space
meeting_comma: inc bx
            jmp     return_filter
end_continue: mov   cs:end_flag,1
return_filter: ret
filter      endp
convert     proc  ; 取参数值
            mov     sl, uan
            xor     ax,ax
            mov     cx,ax
char_to_data: mov cl, byte ptr [bx]
            cmp     cl,','
            je      return_convert
            cmp     cl, 00h
            je      end_convert
            cmp     cl,''
            je      convert_next
            sub     cl,'0'
            mul     si
            add     ax,cx
convert_next: inc   bx
            jmp     char_to_data
end_convert: mov    cs: end_flag,1
return_convert: ret
convert     endp
begin       endp
code_segment ends
            end     begin
```