《计算机世界报》1992 年 1 月 8 日版，p．87，1992 年 1 月

键盘缓冲区直接存取例程

清华大学计算机系　郑　方

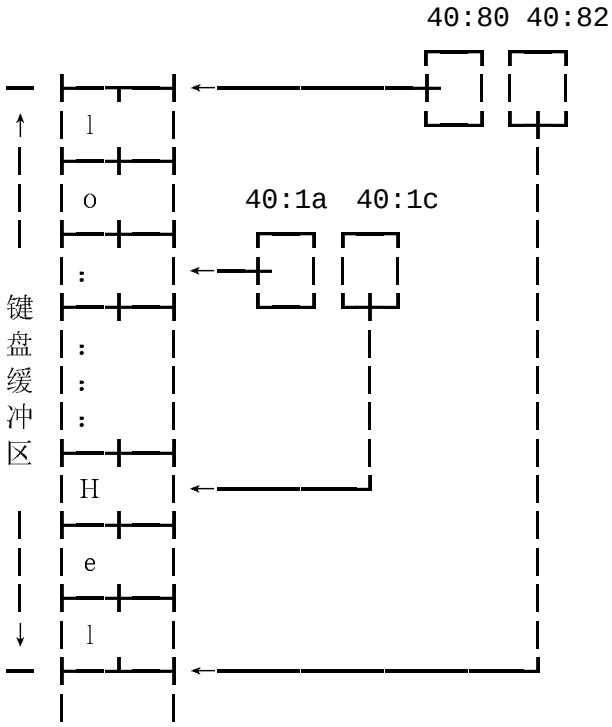
FoxBase+ 中有一个叫 keyboard() 的函数，它把函数的参数（即一个字符串）直接送到键盘缓冲区（Keyboard Buffer）中去，就好象用户用键盘敲入一样。这个函数无疑给编程者带来很大的方便。其中一个最明显的应用是“宏替换”。当我们在键盘中断服务例程（ISR）中捕获某一特定的键时，我们可以把它对应的字符串直接送到键盘缓冲区中去，如同键入的是这个字符串。比如：

特定字符	解释串
Alt_C	copy
Alt_P	pctools <ENTER>
Alt_D	dir

本文将给出用 C 语言编写的具有同样功能的函数 send_keyboard()。　这个函数对 DOS-BIOS 的通讯数据区直接进行存取，因而使用很方便。

BIOS 在 40h 段有一块数据区专门用于记录一些参数或进行通讯，它称为 BIOS 通讯区。DOS 以及 BIOS 的功能调用几乎都要对这个数据区进行读写。在这个通讯区中，有一个 32 字节的区域，它用以作为键盘缓冲区。

键盘缓冲区是一个循环队列，如图所示。循环队列的头和尾的地址分别在 40：80、40：82　两个字空间中保存；而键盘缓冲区中用以存字符的首尾指针在 40：1A、40:1C　两个字空间中保存。



键盘缓冲区中存有 “Hello” 时的情形

DOS-BIOS 的键盘中断服务例程对此缓冲区进行访问。当缓冲区首尾指针相等时，表示缓冲区空；而尾指针指向头指针所指地址的前一个字时，表示缓冲区满。读字符操作总是先从头指针处移走字符，然后使头指针下移；向缓冲区送字符的操作总是先把字符送到尾指针所指的地址处，再把尾指针下移。在指针下移过程中，当指针指向缓冲区尾时，指针要回转指向缓冲区头。可以这样说，头指针指向下一个要移走的字符在缓冲区中的位置，尾指针指向下一个要放入的字符在缓冲区中的位置。

在键盘缓冲区中，每个字符占一个字的空间，其中低字节是字符的 ASCII 码（或扩展码 0），而高字

节为字符的扫描码。汉字没有扫描码，但在前后两半汉字的扫描码位置上分别存放 90h、91h 以示标识。

有了上面的知识，键盘缓冲区的直接存取就很容易编程实现。下面给出两个函数。其中函数 clear_keyboard() 用于把键盘缓冲区清除。

函数 send_keyboard(str) 用于把字符串 str 送入键盘缓冲区。函数 clear_keyboard() 很简单，只要把缓冲区头尾指针置为相等即可。

变量 chinese 用于记录当前要存放的是汉字还是一般的字符。静态函数 scancode() 返回字符 c 的 ASCII 码（c>0 且 c<128）或汉字标识（0x90 或 0x91）。该函数认为当 c >= 128 时为汉字，但当要发送的字符串为可显示字符串时，这一缺陷不成问题。

函数 send_keyboard() 首先判断键盘缓冲区中有多少剩余空间，然后根据要发送的字符串的长度确定实际欲发送的字符个数。最后逐个把字符存入键盘缓冲区中。

下面给出用 Microsoft C 5.0 实现的两个函数。

```
/*=================================================================*
  File Name   :  Keyboard.C (Lib)
  Programmer  :  Zheng Fang
  Date        :  08-04-1991
 *=================================================================*/

#include  <string.h>
#include  <stdlib.h>
#include  <bios.h>

typedef   unsigned  far    *PFU;
typedef   int       far    *PFI;
typedef   char      far    *PFC;

PFU kb_begin  = (PFU)0x00400080;  /* 键盘缓冲区头 */
PFU kb_end    = (PFU)0x00400082;  /* 键盘缓冲区尾 */
PFU kb_head   = (PFU)0x0040001a;  /* 缓冲区头指针 */
PFU kb_tail   = (PFU)0x0040001c;  /* 缓冲区尾指针 */
PFC dos_data  = (PFC)0x00400000;  /* 数据区长指针 */

/*---------------- clear keyboard buffer ----------------------*/

void   clear_keyboard()
{
       _disable();
       *kb_head = *kb_tail;
       _enable();
}

/*---------------- send a string to the keyboard buffer --------*/

int send_keyboard(str)
char   *str;
{
       int    left, num, k, chinese;
       unsigned char scancode(unsigned char, int *);

       _disable();

       if ((left = *kb_head - *kb_tail) <= 0)
              left += (*kb_end - *kb_begin);
       num = min(left/2-1, strlen(str));

       chinese = 0;
```

```
        for (k=0; k < num; k++)
        {
               dos_data[(*kb_tail)++] = str[k];
               dos_data[(*kb_tail)++] = scancode(str[k], &chinese);
               if (*kb_tail == *kb_end)
                      *kb_tail = *kb_begin;
        }

        _enable();

        return(num);
}

/* return the scan code of the character c */
static
unsigned char scancode(unsigned char c, int *chinese)
{
    static char scan_code[128] = {
         0,  30,  48,  46,  32,  18,  33,  34,
        14,  15,  28,  37,  38,  28,  49,  24,
        25,  16,  19,  31,  20,  22,  47,  17,
        45,  21,  44,  26,  43,  27,   7,  12,

        57,   2,  40,   4,   5,   6,   8,  40,  /* !"#$%&'*/
        10,  11,   9,  13, 51,  74,  52,  53,  /*()*+,-./*/
        11,   2,   3,   4,   5,   6,   7,   8,  /*01234567*/
         9,  10,  39,  39,  51,  13,  52,  53, /*89:;<=>?*/
         3,  30,  48,  46,  32,  18,  33,  34, /*@ABCDEFG*/
        35,  23,  36,  37,  38,  50,  49,  24, /*HIJKLMNO*/
        25,  16,  19,  31,  20,  22,  47,  17, /*PQRSTUVW*/
        45,  21,  44,  26,  43,  27,   7,  12, /*XYZ[\]^-*/
        41,  30,  48,  46,  32,  18,  33,  34, /*`abcdefg*/
        35,  23,  36,  37,  38,  50,  49,  24, /*hijklmno*/
        25,  16,  19,  31,  20,  22,  47,  17, /*pqrstuvw*/
        45,  21,  44,  26,  43,  27,  41,  141 /*xyz{|}~ */
    };

    if (!*chinese)
    {
        if (c < 128)
        {
            return(scan_code[c]);
        }
        else
        {  /* 1st half of chinese character */
            *chinese = 1;
            return(0x90);
        }
    }
    else
    {      /* 2nd half of chinese character */
        *chinese = 0;
        return(0x91);
    }
}
```