要 Server 口令的键盘锁定状态，输入口令字并回车，闪烁即停止，键盘恢复工作。

若因记忆错误或外因性信息突变致口令失效，正确方法是清除口令。

近些年，安装使用、禁用口令开关或跨接(跳线)器的名牌微机渐多，出厂设置每为使用口令状态，允许设置口令。若将该开关相应的拨钮推至禁用位置，微机在启动过程中，自动初始化 CMOS RAM 口令单元，可单一地清除口令，不波及其余信息，属处理非常情况的最佳选择。

AST、COMPAQ、HP...微机都有此类开关，均详列随机手册.谨以 AST PP3 微机为例简述清除口令的操作步骤如下：

1.关机数分钟后，从主机拔下电源线，用钥匙开机箱锁，卸下机盖。

2.用竹签或其他绝缘体，将位于主板前面、软盘驱动器左侧的 4 位开关 SW1 上标'2' 的拨钮推至'ON'。

3.加盖锁机，接上电源线后开机，进入系统即表示口令已清除。(重设密码应关机，将开关或跨接器恢复原状，再次运行 SETUP)

AST PⅡ 型取 8 位开关 SW1'2'，AST PA4 是 SW1'5'，而 AST PⅢ 机改用位于主板左下角的跨接器 E6，需拔除跨接块断开连接(禁用口令)；早年的 AST P386 对应为 E1 跨接器，位于主板右列短扩展槽下方，原跨接标'PWD'侧(使用口令)，需移接标'E1'侧(禁用口令)。

COMPAQ DESKPRO／M386 及 486 需拨动开关电源下方的 SW'2'至'ON'，其他型号的开关位置有变动，或在开关电源左侧等。其 PL 及 EPL 系列微机的清除口令跨接器均标 P14，位于主板中线上段附近，拔除跨接块，经十秒钟口令被清除，原位插妥跨接块，启动微机即可重设口令。

HP 微机设计独特，例如 HP Vectra 486 使用／禁用口令的(独位)开关安装在多功能卡左上角，设置禁用也是由'OFF'拨至'ON'。

至于无清除口令装置的主板，宜检查在电池附近有无两针式或三针式标志'CMOSRAM RESET'的跨接器，前者出厂设置跨接块仅插 Pin 1，空旷 Pin 2；三针式则跨接 Pin 1—Pin 2，空旷 Pin 3，均处于非复位状态。瞬间跨接两针式复位器的 Pin 1—Pin 2 或三针式的 Pin 2—Pin 3 可使 CMOS RAM 迅即归零，口令一并清除，似也优于断离电池法。

仿真计算器的通用 C 程序

王晓武 (总后司令部战勤计划局)

读者在使用 WPS 及 pctools 等工具软件时，可方便地弹出计算器窗口来仿真使用计算器的功能。也许很多软件作者也设想在自己开发的软件中为用户提供这种计算器仿真功能，如读者在开发自己的 MIS 系统或表处理软件及文本检索软件等应用软件时，也应该具有一个热键弹出的计算器窗口，为用户提供仿真计算器的功能，从而改善应用软件的用户友好性。笔者介绍一个可供 C 语言及各种计算机语言调用的通用计算器仿真程序。该程序用 MS C6.0 语言编写，在 DOS5.0 环境下通过；源程序清单附后，各种子函数及语句的功能程序注释比较详细。可在各种汉字操作系统下供软件作者调用。

说明：如果要在 C 语言程序中调用此程序，最好是将其 main 改为用户自定义的子函数名，将其融入读者自己的程序中调用。若要在其他语言程序中调用，应采用 DOS 的程序加载功能实现，例如在 FoxBASE 应用程序中用以下的命令格式调用：

RUN JSQ [行 列]

JSQ.C 源程序清单如下：

```
#include <stdio.h>
#include <stdlib.h>
#include <bios.h>
#include <malloc.h>
char far *address;
unsigned char *p;
void cur_lock(int flag)        /*隐去或显示光标*/
{
   union REGS r,o;
   r.h.ah=1; r.h.ch=0;
   r.h.cl=0; r.h.bh=0;
   if(flag==1){ r.h.ch=23; r.h.cl=25;}
   int86(0x10,&r,&o);
}
void goto_xy(int x,int y)      /*置光标位置*/
{
   union REGS r;
   r.h.ah=2; r.h.bh=0;
   r.h.dh=(char)x; r.h.dl=(char)y;
   int86(0x10,&r,&r);
}
get_key( )                     /*获取按键值*/
```

```
{
    union REGS r; r.h.ah = 0;
    return int86(0x16,&r,&r);
}
void write_char(int x,int y,char ch,int attrib)    / * 显示字符 * /
{
    union REGS r;
    goto_xy(x,y);
    r.h.ah = 9; r.h.bh = 0;
    r.x.cx = 1; r.h.al = ch;
    r.h.bl = (char)attrib;
    int86(0x10,&r,&r);
}
void write_video(int x,int y,char * p,int attrib)      / * 显示字符串
* /
{
    register int i;
    union REGS r;
    for (i = y; * p;i++){
      if ( * p = = '\n')
         { goto_xy(x+1,2);break; }
      goto_xy(x,i);
      r.h.ah = 9; r.h.bh = 0;
      r.x.cx = 1; r.h.al = * p++;
      r.h.bl = (char)attrib;
      int86(0x10,&r,&r);
    }
}
void save_popup(int startx,int starty)      / * 保存屏幕 * /
{
    register int i,j;
    int endx,endy;
    unsigned char * buf_ptr;
    union REGS r;
    endx = startx+11; endy = starty+32;
    p = (unsigned char * )malloc(2 * (endx-startx) * (endy-starty));
    if(!p) exit(1);
    buf_ptr = p;
      for (j = startx;j< endx;j++)
         for(i = starty;i< endy;i++){
           goto_xy(j,i);
           r.h.ah = 8; r.h.bh = 0;
           int86(0x10,&r,&r);
           * buf_ptr++ = r.h.al;
           * buf_ptr++ = r.h.ah;
    }
}
void restore_popup(int startx,int starty)    / * 恢复屏幕 * /
{
    register int i,j;
    int endx,endy;
    unsigned char * buf_ptr;
    union REGS r; buf_ptr = p;
    endx = startx+11; endy = starty+32;
    for (j = startx;j< endx;j++)
         for(i = starty;i< endy;i++){
         goto_xy(j,i); r.h.ah = 9;
         r.h.bh = 0; r.x.cx = 1;
         r.h.al = * buf_ptr++;
         r.h.bl = * buf_ptr++;
         int86(0x10,&r,&r);
      }
    free(p);
}
void main(int argc,char * argv[ ])
{
int startx,starty,endx,endy;
int i,k = 0,k1 = 0;
union inkey{ char ch[2]; int i; }c;
double dc = 0.0,dr = 0.0,dm = 0.0,j = 0.0;
char ch = '',ch1 = '',dec = '';
address = (char far * ) 0x00400017L; / * NumLock ON * /
* address = * address   0x20;
cur_lock(0);        / * 隐去光标 * /
startx = atoi(argv[1]);
starty = atoi(argv[2]);
if(startx > 12) startx = 12;
if(starty > 48) starty = 48;
endx = startx+10; endy = starty+32;
save_popup(startx,starty);      / * 保存屏幕 * /
write_video(startx, starty,"    +-----------------+ ",0x3f);
write_video(startx+1, starty," |                   | ",0x3f);
write_video(startx+2, starty," |---------+---------| ",0x3f);
write_video(startx+3, starty," | 7   8   9 |*   MC  | ",0x3f);
write_video(startx+4, starty," | 4   5   6 |/   MR  | ",0x3f);
write_video(startx+5, starty," | 1   2   3 |-   M-1 | ",0x3f);
write_video(startx+6, starty," | 0   .   = |+   M+  | ",0x3f);
write_video(startx+7, starty," | C: Clear|ESC  Exit | ",0x3f);
write_video(startx+8, starty," +-----------------+",0x3f);
write_video(startx+9,starty+2,"                    ",0x1f);
write_video(startx+1, starty+2,"                    ",0x1f);
for(i = 1;i< 10;i++) write_video(startx+i,starty+30," ",0x1f);
goto_xy(startx+1,starty+7);
printf("%18.4f",dc);
while(1)
{
c.i = get_key( );
if(dr > 999999999999.9999||dc > 999999999999.9999) / * 位数超出范
围 * /
```

```
    {
    dr=0.0; dc=0.0; ch=' '; dec=' ';
    goto_xy(24,0);
    printf("value out of range\7");
    }
if(c.ch[0]>='0'c.ch[0]<='9'&&dec!='.'&&k<10) /* 接收整数数字 */
    {
    j=c.ch[0]-48; dc=dc*10+j;
    k++; if(k>9) k=0;
    goto_xy(startx+1,starty+7);
    printf("%18.4f",dc);
    }
if(c.ch[0]>='0'&&c.ch[0]<='9'&&dec=='.'&&k1<4) /* 接收小数数字 */
    {
    j=c.ch[0]-48;
    switch(k1){
        case 0: dc+=j/10; break;
        case 1: dc+=j/100; break;
        case 2: dc+=j/1000; break;
        case 3: dc+=j/10000; break;
        } k1++;
    if(k1>3) k1=0;
    goto_xy(startx+1,starty+7);
    printf("%18.4f",dc);
    }
if(ch=='')      /* 初次运算处理 */
    switch(c.ch[0]){
        case '+': case '-': case '*': case '/':
        if(dr==0.0) dr=dc;
        dc=0.0; dec=' '; k=0; k1=0;
        write_char(startx+1,starty+5,c.ch[0],0x0f);
        break;
        }
    if(ch!='')      /* 非初次运算处理 */
    switch(c.ch[0]){
        case '+': case '-': case '*': case '/':
        switch(ch){
            case '+': dr=dr+dc; break;
            case '-': dr-=dc; break;
            case '*': dr=dr*dc; break;
            case '/': if(dc!=0.0) dr=dr/dc; break;
            }
        dc=0.0; k=0; k1=0; dec=' ';
        write_char(startx+1,starty+5,ch,0x0f);
        goto_xy(startx+1,starty+7);
        printf("%18.4f",dr);
        break;
        }
if((c.ch[0]=='='||c.ch[0]==13)&&dr!=0)      /* 取当前运算结果 */
    {
    switch(ch){
        case '+': dr=dr+dc; break;
        case '-': dr-=dc; break;
        case '*': dr=dr*dc; break;
        case '/': if(dc!=0.0) dr=dr/dc; break;
        }
    write_char(startx+1,starty+5,' ',0x0f);
    goto_xy(startx+1,starty+7);
    printf("%18.4f",dr);
    k=0; k1=0; dec=' '; dc=0.0;
    }
switch(c.ch[0]){      /* 接收有效运算符 */
    case '+': case '-': case '*': case '/': case '=':
    ch=c.ch[0];
    switch(c.ch[0]){
        case '+': case '-': case '*': case '/':
        write_char(startx+1,starty+5,c.ch[0],0x0f);
        break;
        }
    break;
    case '.': dec='.'; break;
    }
if((c.ch[0]=='c'||c.ch[0]=='C')&&ch1=='')/* 清除处理 */
    {
    dr=0.0; dc=0.0; ch=' '; dec=' ';
    write_char(startx+1,starty+5,' ',0x0f);
    goto_xy(startx+1,starty+7);
    printf("%18.4f",dc);
    }
if(ch1=='m')      /* 接收累加运算符 */
    switch(c.ch[0]){
        case '+': case '-': case 'C': case 'c': case 'R': case 'r':
        ch1=c.ch[0]; break;
        }
if(ch1!='')      /* 累加运算处理 */
    switch(c.ch[0]){
        case '+': case '-': case 'C': case 'c': case 'R': case 'r':
        if(dr==0&&dc!=0) dr=dc;
        switch(ch1){
            case '+': dm=dm+dr; break;
            case '-': dm=dm-dr; break;
            case 'C': case 'c': dm=0.0;
                write_video(startx+1,starty+2,"  ",0x0f);
                break;
            case 'R': case 'r':
                write_char(startx+1,starty+5,c.ch[0],0x0f);
```

```
                if(dm==0.0)
                write_video(startx+1,starty+2,"",0x0f);
                goto_xy(startx+1,starty+7);
                printf("%18.4f",dm);
                break;
            }
        dr=0.0; dc=0.0; ch1='\'; ch='\'; k=0; k1=0;
        break;
        }
if(c.ch[0]=='M'||c.ch[0]=='m')      /* 显示累加运算标志
*/
    {
    ch1='m';
    write_char(startx+1,starty+2,'M',0x0f);
    }
if(ch=='=')       /* 错误按键处理 */
    switch(c.ch[0]){
        case '+': case '-': case '*': case '/':
        dr=0.0; ch='\'; dec='\'; break;
        }
if(c.ch[0]==27) break;       /* 结束返回 */
}
restore_popup(startx,starty);       /* 恢复屏幕 */
address=(char far *)0x00400017L;  /* NumLock OFF */
*address=*address&0xdf;
cur_lock(1);       /* 恢复光标 */
}
```