

实验十一 定时中断

学号： 04022212

姓名： 钟 源

一、实验任务和实验结果

请写出**已通过验收**的各个实验任务的具体内容、调试通过的源程序（**加注释**）和实验结果。

实验结果请截图，并加以必要说明。

1.任务：

定时/计数器 8253 每隔 55ms 发一次定时中断请求信号(其中断类型为 1CH)，CPU 响应中

断后转去执行 TIMERINTS 中断服务程序。

基本功能：

本次实验任务为改写定时中断(中断类型为 1CH)的中断服务程序，要求在定时中断服务程

序中累计中断次数，每计到 50 次定时中断就在显示器上显示字符串“SUN”。

主程序：从屏幕左上角到右下角循环显示“太阳”图形，并判断字符串“SUN”的显示次数是

否到十次，到十次就结束程序返回 DOS。

附加功能：

1) 将累计定时中断次数改为设定一个定时的时间（如定时 0.5 秒、1 秒、2 秒……），每次

定时时间到了就显示一次字符串 1（显示字符串 1 的内容自己设置，希望同学们尽量不要雷同，

字符串前面加上显示次数，后面加个空格）；

2) 定时显示 12 次字符串后，不等 25 行“太阳”图形显示完，就立即返回 DOS；

3) 在本次实验中加入实验十的键盘中断，在程序执行过程中，如果有按键，就显示一个字

符串 2（显示字符串 2 的内容也自己设置，尽量不要雷同，字符串前面加上按键次数，后面加个

空格）；如果定时时间到了就显示一次字符串 1（字符串前面加上显示次数，后面加个空格）。按

键次数或定时显示次数只要有一个到 12 次了，就结束程序返回 DOS；

注：按键先到 12 次和定时显示先到 12 次 这两种情况都要试一下。

4)将“太阳”图标和字符串改为彩色显示(彩色背景彩色字符),如,红底白字,蓝底黄字,.....。

注：“太阳”图形、定时显示的字符串 1 和按键显示的字符串 2 最好用不同的颜色。

5) 其他：鼓励同学们自己增加其他的附加功能。

2.源程序及注释：

```
DATA SEGMENT
;提示消息
MSG1      DB      ?,?, '-MAKA  BAKA'
SLENGTH1  EQU      $-MSG1
MSG2      DB      ?,?, '-WUXI  DIXI'
SLENGTH2  EQU      $-MSG2
;计数器
COUNT    DB  ?           ;计数次数
LOS       DB  ?
KEYN      DB  ?
;计数上限
LIM1      EQU  12D        ;键盘中断计数上限
LIM2      EQU  12D        ;定时中断计数上限
LINE      DB  25          ;最大显示行数

TIME      DB  3;
          DB  ?;
          DB  3  DUP(?);
T         DB  10;
N         DB  18;
TXT1      DB  "Please input TIME in units of 0.1s(5<=TIME<=50):$";文本
提示 1
ERROR0    DB  0DH,0AH,'Wrong Input Format!',$'           ;错误
提示 0
ERROR1    DB  0DH,0AH,'The Input is too small!',$'        ;错误
提示 1
ERROR2    DB  0DH,0AH,'The Input is too big!',$'           ;错误
提示 2
DATA ENDS

STACK SEGMENT STACK
          DW  300H DUP(?)
```

```
STACK ENDS
```

```
CODE SEGMENT
```

```
    ASSUME CS:CODE,SS:STACK,DS:DATA
```

```
;显示太阳的子程序
```

```
START:
```

```
;给段寄存器赋值
```

```
    MOV AX,STACK
```

```
    MOV SS,AX
```

```
    MOV AX,DATA
```

```
    MOV DS,AX
```

```
    MOV AX,0
```

```
    MOV ES,AX
```

```
    LEA SI,TXT1           ;取 TXT1 地址
```

```
    MOV DX,SI             ;赋值地址给 DX
```

```
    MOV AH,09             ;功能号 09H
```

```
    INT 21H               ;类型号 21H, 显示 DS: DX 指向的以$结尾的字符串
```

```
;输入 TIME 并检验
```

```
    LEA SI,TIME;
```

```
    MOV DX,SI             ;赋值地址给 DX
```

```
    MOV AH, 0AH;
```

```
    INT 21H;              输入
```

```
    CALL INPUTError;      判断输入格式是否错误
```

```
    CALL ASCtoNUM;        把缓存区 ASC 码转换成 NUM
```

```
    CALL VALUEError;      判断时间符合实际
```

```
;TIMEtoN
```

```
    PUSH SI
```

```
    PUSH AX
```

```
    PUSH BX
```

```
    LEA SI,T
```

```
    SUB AX,AX;
```

```
    MOV AL,[SI];
```

```
    MOV BL,100D
```

```
    MUL BL;
```

```
    MOV BL,55D
```

```
    DIV BL;
```

```
    LEA SI,N
```

```
    MOV [SI],AL
```

```
    POP BX
```

```
    POP AX
```

```

        POP SI

        MOV AH, 0AH;
        SUB AX,AX;
        SUB BX,BX;
        SUB CX,CX;
        SUB DX,DX;
        SUB SI,SI;
        SUB DI,DI;
;保存原 1CH 中断向量
        MOV AX,ES:[1CH*4]
        PUSH AX
        MOV AX,ES:[1CH*4+2]
        PUSH AX
;置 1CH 中断向量
        CLI
        MOV AX,OFFSET TIMERINTS
        MOV ES:[1CH*4],AX
        MOV AX,SEG TIMERINTS
        MOV ES:[1CH*4+2],AX
        STI
;保存原 09H 的中断
        MOV AX,ES:[09H*4]
        PUSH AX
        MOV AX,ES:[09H*4+2]
        PUSH AX
        PUSH DS
;置 09H 中断向量
        CLI
        MOV AX, SEG KEYINT
        MOV DS, AX
        MOV DX, OFFSET KEYINT
        MOV AL, 09H
        MOV AH, 25H
        INT 21H
        STI
        POP DS;
;计数值置 0
        ;MOV MSG2,0
        MOV LOS,0
        MOV COUNT,0
        MOV KEYN,0
;调用 DISP1 显示太阳
AGAIN:

```

```

CALL FAR PTR DISP1
CMP KEYN,LIM1;与 12 比较
JAE NEXT
CMP LOS,LIM2;与 12 比较
JAE NEXT
JMP AGAIN
NEXT:
;恢复 09H 中断
CLI
POP ES:[09H*4+2]
POP ES:[09H*4]
STI
;恢复 1CH 向量
CLI
POP ES:[1CH*4+2]
POP ES:[1CH*4]
STI
EXIT:
MOV AH,4CH
INT 21H

;—————INPUTError 子程序—————
;判断输入格式是否合规的子程序
INPUTError PROC
    MOV BX, OFFSET TIME+2;    BX 取到数组的首地址
    MOV CL, 00H;            开始 CL 置零

    MOV AL, [BX];            提取 BX 对应的值
    CALL NUMerror;            判断是否是数字
    INC BX;                    判断下一字节

    MOV AL, [BX];            提取 BX 对应的值
    CALL NUMerror;            判断是否是数字

    CMP CL, 00H;            CL=0 说明格式正确
    JE RIGHT ;                正确跳转

ERR0:
    MOV DX, OFFSET ERROR0
    MOV AH, 9
    INT 21H;                在屏幕显示 ERROR0
    JMP START;                跳回开头
RIGHT:
    RET

```

```

INPUTError ENDP

;-----NUMError 子程序-----
;判断数字错误的子程序
NUMError PROC
    CMP AL, 39H;
    JA ERR_num;
    CMP AL, 30H;
    JB ERR_num;    以上都是比较 ASC 码，30，39 为数字在 ASC 码表中范围
    RET
ERR_num:
    MOV CL, 01H;    其 ASC 码若不在数字范围，CL 置 1
    RET
NUMError ENDP

;-----ASCtoNUM 子程序-----
;ASC 变压缩 NUM 的子程序
ASCtoNUM PROC
    PUSH SI;
;转换个位
    MOV BX, OFFSET TIME+2;    指针指向数据区
    MOV SI, OFFSET T;
    SUB CX,CX;
    MOV CL, [BX+1];
    SUB CL, 30H
;转换十位
    MOV AL,[BX];
    SUB AL,30H
;整合相加
    MOV DL,10;
    MUL DL;转换十位
    ADD AX,CX;整合相加
    MOV [SI],AL;储存
    SUB CX,CX;
    MOV CL,[SI];储存在 CL，以便 VALUEError 子程序的判断

    POP SI;
    RET
ASCtoNUM ENDP

;-----VALUEError 子程序-----
;判断时间符合实际的子程序
VALUEError PROC
    CMP CL, 5D;    将行数和 25 比较

```

```

        JB    tooSMALL;           >25 则跳转
        CMP   CL, 50D;           将行数和 25 比较
        JA    tooBIG;           >25 则跳转
        JMP   GONEXT;

tooSMALL:
        MOV   DX, OFFSET  ERROR1
        MOV   AH, 9
        INT   21H;               在屏幕显示 ERROR1
        JMP   START;            跳回开头

tooBIG:
        MOV   DX, OFFSET  ERROR2
        MOV   AH, 9
        INT   21H;               在屏幕显示 ERROR2
        JMP   START;            跳回开头

GONEXT:
        RET

VALUEerror ENDP

```

;—————DISP1 子程序—————

```

DISP1 PROC FAR
        PUSH  SI
        PUSH  AX
        PUSH  BX
        PUSH  CX
        PUSH  DX
;BIOS 中断调用，屏幕显示
;读当前显示状态
        MOV   AH,15
        INT   10H
;设置显示模式 40*25 黑白
        MOV   AH,0
        INT   10H
        MOV   DX,0 ;行号为 0，列号为 0

REPT:
;附加任务，12 次显示 SUN 后立即停止返回 DOS
        CMP   LOS,LIM2;与 12 比较
        JB    K1
        JMP   D_EXIT1
;附加任务，12 次键盘后立即停止返回 DOS
K1:
        CMP   KEYN,LIM1;与 12 比较
        JB    K2
        JMP   D_EXIT1
K2:

```

```

;设置光标位置
    MOV AH,2
    INT 10H
;写字符及属性到当前光标位置处
    MOV AL,0FH
    MOV CX,1
    MOV BL,01001111B    ;红底白字
    MOV AH,9
    INT 10H
;延时
    CALL DELAY
;清除屏
    MOV AL, 03    ;设置显示方式(80*25 彩色文本)
    MOV AH, 0
    INT 10H        ;清屏
    INC DH          ;行号+1
    ADD DL, 2        ;列号+2
    SUB BX,BX
;查看是否到了最大行数
    MOV SI, OFFSET LINE;    BX 取到数组的首地址
    CMP DH, [SI]
    JB REPT        ;如果没到 LINE 行，则跳转到 REPEAT 更新光标位置；到了 LINE 行则
退出子程序
D_EXIT1:
    POP DX
    POP CX
    POP BX
    POP AX
    POP SI
    RET
DISP1 ENDP
;—————DISP2 子程序—————
;显示字符串 MSG1 的子程序
DISP2 PROC    FAR
    PUSH SI
    PUSH DX
    PUSH CX
    PUSH BX
    PUSH AX
;读取当前光标位置
    MOV AH, 3
    MOV BH, 0
    INT 10H

```



```

    INC DL          ;列号+1
    MOV AH, 2
    INT 10H         ;将光标设在下一个位置，避免覆盖原字符

    LEA SI, MSG1    ;将字符串 MSG1 的首地址装入 SI
    MOV BYTE PTR[SI], 30H
    MOV BL, LOS
    CMP LOS, 10
    JB XIAOYU1
DAYU1:
    MOV BYTE PTR[SI], 31H
    SUB BL, 10
XIAOYU1:
    MOV AL, BL
    OR  AL, 30H
    MOV BYTE PTR[SI+1], AL
    MOV CX, SLENGTH1 ;将字符串 MSG1 的长度装入 CX
NEXTC1:
    MOV BL, 01111000B ;白底黑字
    MOV AH, 09H
    INT 10H

    LODSB           ;从内存中的 SI 指向的位置读取一个字节，并将其存入 AL 寄存器，同时 SI 寄存器递增。
    MOV AH, 0EH     ;设置 AH 寄存器为 0EH，表示显示字符和属性。
    MOV BX, 1
    INT 10H
    LOOP NEXTC1
;弹出
    POP AX
    POP BX
    POP CX
    POP DX
    POP SI
    RET
DISP2    ENDP

;-----DISP3 子程序-----
;附加任务：显示字符串 MSG2 的子程序
DISP3 PROC FAR
;压栈
    PUSH SI
    PUSH DX
    PUSH CX

```

```

    PUSH BX
    PUSH AX
;读取当前光标位置
    MOV AH, 3
    MOV BH, 0
    INT 10H

    INC DL          ;列号+1
    MOV AH, 2
    INT 10H        ;将光标设在下一个位置，避免覆盖原字符

    LEA SI, MSG2    ;将字符串 MSG 的首地址装入 SI
    MOV BYTE PTR[SI],30H
    MOV BL,KEYN

    ;附加功能：加入按键次数 KEY
    CMP KEYN,10
    JB XIAOYU2
DAYU2:
    MOV BYTE PTR[SI],31H
    SUB BL,10
XIAOYU2:
    MOV AL,BL
    OR  AL,30H
    MOV BYTE PTR[SI+1],AL
    MOV CX, SLENGTH2 ;将字符串 MSG2 的长度装入 CX
NEXTC2:
    MOV BL,00011110B ;蓝底黄字
    MOV AH,09H
    INT 10H

    LODSB           ;从内存中的 SI 指向的位置读取一个字节，并将其存入 AL 寄存器，同时 SI 寄存器递增。
    MOV AH, 0EH     ;设置 AH 寄存器为 0EH，表示显示字符和属性。
    MOV BX, 1
    INT 10H
    LOOP NEXTC2
;弹出
    POP AX
    POP BX
    POP CX
    POP DX
    POP SI
    RET

```

DISP3 ENDP

;—————DELAY 子程序—————

;延时 500ms 子程序

DELAY PROC

 ;压栈

 PUSH DX

 PUSH CX

 PUSH BX

 PUSH AX

 ;中断调用延时

 MOV AH, 2CH

 INT 21H ;DOS 调用系统时间

 MOV BL, DL ;存储当前秒数

 ADD BL, 50

 MOV BH, DH ;存储当前百分秒数

 ADD BH, 1

DL_BEGIN:

 INT 21H ;调用系统时间

 CMP DH, BH ;比较当前秒数与 BH

 JAE DL_END ;若大于 BH, 则完成延时, 退出程序

 ;CMP DL, BL ;比较当前百分秒数与 BL

 ;JAE DL_END ;若大于 BL, 则完成延时, 退出程序

 JMP DL_BEGIN ;反之, 则继续调用系统时间

DL_END:

 ;弹出

 POP AX

 POP BX

 POP CX

 POP DX

 RET

DELAY ENDP

;—————TIMERINTS 子程序—————

;设置计时器中断服务子程序

TIMERINTS PROC FAR

 PUSH SI

 PUSH AX

 CLI

 MOV SI,OFFSET N

 MOV AL,[SI]

 INC COUNT

 CMP COUNT,AL ;定时

 JNE GO1

```

    MOV COUNT,0      ;清零
    MOV SI,OFFSET MSG1
    INC LOS;计数 10 次停止
    CALL FAR PTR DISP2
G01:
    POP AX
    POP SI

    STI
    IRET
TIMERINTS ENDP

;—————KEYINT 子程序—————
;设置键盘中断服务程序
KEYINT PROC FAR
;保护现场
    PUSH SI
    PUSH DX
    PUSH BX
    PUSH AX
;关中断
    CLI
;读取键盘扫描码
    IN AL,60H
    MOV AH,AL
;输出一个正脉冲
    IN AL,61H
    OR AL,80H
    OUT 61H,AL
    AND AL,7FH
    OUT 61H,AL
;判断：无键按下则执行退出键盘中断
    TEST AH,80H
    JNE G02
;有键按下则显示 MSG2，KEYN 同时计数
    INC KEYN
    CALL FAR PTR DISP3
G02:
;EOI 命令
    MOV AL,20H
    OUT 20H,AL

    POP AX
    POP BX

```

```
POP DX
POP SI
STI
IRET
KEYINT ENDP

CODE ENDS
END START
```

3.程序设计思路:

(1) 提示用户输入时间:

- ① 在数据段定义了提示文本以及用于存储输入时间的缓冲区。
- ② 使用 INT 21H 中断调用功能 0AH 来从键盘输入时间，并将其保存在缓冲区中。
- ③ 调用子程序 INPUTError 检查输入格式是否正确，ASCtoNUM 将 ASCII 码转换为数字，VALUEError 判断输入的时间是否在合理范围内。

字，VALUEError 判断输入的时间是否在合理范围内。

(2) 设置中断向量和计时器:

- ① 保存了原来的 1CH 和 09H 中断向量。
- ② 设置了新的计时器中断向量和键盘中断向量。
- ③ 计时器中断服务程 TIMERINTS 在每次计时器中断触发时，调用 DISP2 显示字符串 MSG1，并增加计数。
- ④ 键盘中断服务程序 KEYINT 在键盘按下时，调用 DISP3 显示字符串 MSG2，并增加计数。

(3) 显示字符串和检测计数器:

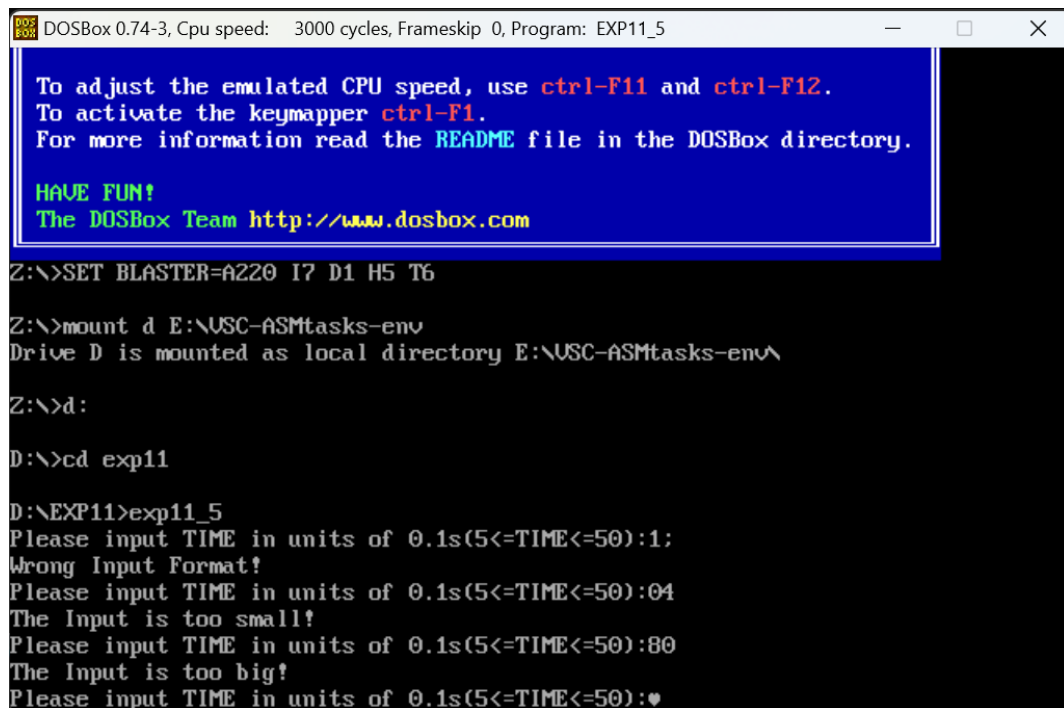
- ① 使用 DISP1 显示太阳，同时检查计数是否达到预设的上限。
- ② 在主程序中循环调用 DISP1，直到达到预设的计数上限。

(4) 清理和退出：

在退出前，恢复原来的中断向量，然后调用 INT 21H 退出到 DOS。

总的来说，该程序除了完成了规定的附加功能外，还实现了用户自己决定并输入中断间隔时间，并检测输入是否有效的功能。

4.实验结果：



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: EXP11_5

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

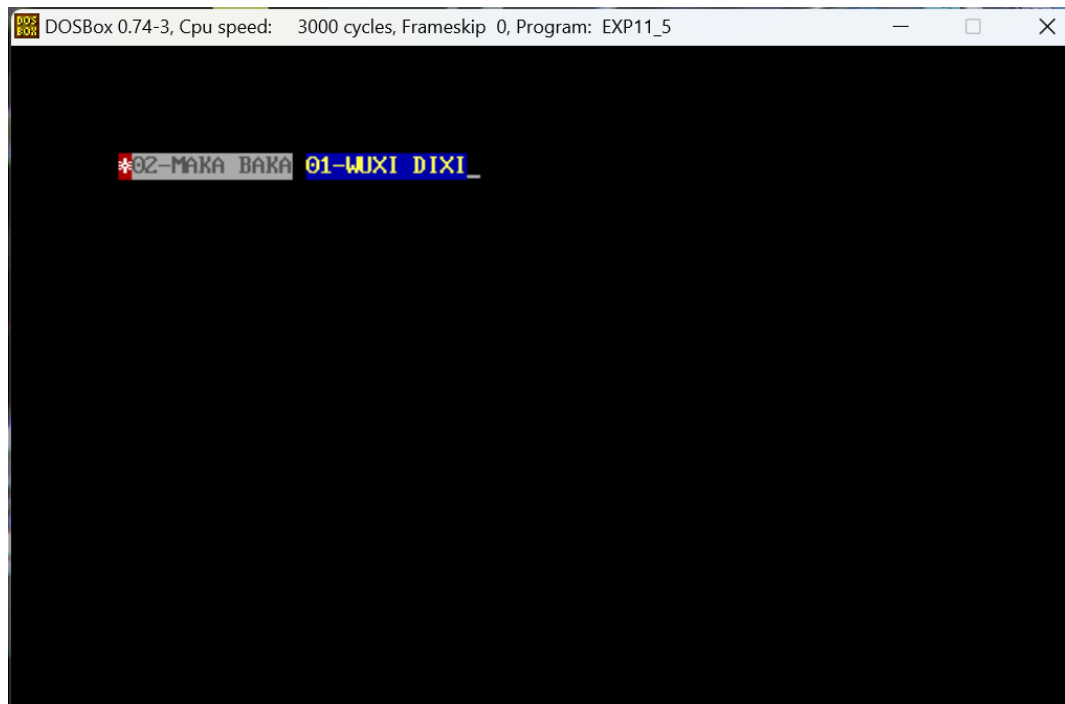
Z:\>mount d E:\USC-ASMTasks-env
Drive D is mounted as local directory E:\USC-ASMTasks-env\

Z:\>d:

D:\>cd exp11

D:\EXP11>exp11_5
Please input TIME in units of 0.1s(5<=TIME<=50):1;
Wrong Input Format!
Please input TIME in units of 0.1s(5<=TIME<=50):04
The Input is too small!
Please input TIME in units of 0.1s(5<=TIME<=50):80
The Input is too big!
Please input TIME in units of 0.1s(5<=TIME<=50):*
```

实现了检验输入间隔时间是否出错，并相应报错的功能。



实现了用户输入间隔显示中断时间、定时显示 12 次字符串后，不等 25 行“太阳”图形显示完，就立即返回 DOS；按键次数或定时显示次数只要有一个到 12 次就结束程序返回 DOS、彩色显示“太阳”图标和字符串等功能。

二、实验总结（实验中遇到的问题、解决方法和实验收获）

1. 通过 DOS 系统功能调用 25H 设置中断向量时，应先将 DS 寄存器压栈保护，然后再赋值中断服务程序的段地址，设置过后出栈还原 DS。

2. 在显示“太阳”子程序中分别增加按键次数、显示次数与 10 的比较，可以在“太阳”符号显示 25 行之前就结束程序。

3. 调试时发现，小太阳图标运动过快，于是我应用调用系统时间的方式将延时设为 1s，使显示更清晰。

三、回答思考题：

1. 如果要求定时 2 秒左右显示一次字符串，那么，定时中断应该累计多少次显示一次字符串？

答：应累积 $2000/55$ 约为 36 次，再显示字符。

事实上，我设计的附加功能也应用了该算法，即累计 $T/55$ (T 单位为 ms) 次就显示一次字符串。

2. 完成附加功能的同学请回答：

(1) 在完成附加功能 3)(加上实验十的键盘中断)时，如果程序运行结束返回 DOS 后，键盘不能正常使用，请分析一下可能会有哪些原因（至少写出三种原因）。

答：① 压栈和出栈（PUSH 和 POP）没有对应，如果不一一对应，会导致子程序结束时无法正常返回 CS: IP 或 IP 值。

② 在处理中断时没有一层层结束中断，直接返回 DOS，也会导致无法正常返回 CS: IP 或 IP 值。

③ 未能恢复原中断向量，导致自定义的中断程序还残留在中断向量表中。

(2) 在完成附加功能 3)时，如果在按键显示字符串 2 时，各显示字符之间有延时，会遇到显示字符串 2 的字符中间会插入定时中断显示的字符串 1 的情况，请分析一下出现这种情况的原因，并说明如何修改程

序才能避免这种情况。

答：如果选择单个字符方式输出字符串确实会出现这种问题，但如果一次显示完字符串就不会出现该问题。可以通过调用功能号为 09H、向量号为 21H 的中断，并将 CX 设置为要输出字符串的长度，则能有效避免该问题。