学院：**信息学院**　学期：**2022 年秋季学期**　课程名称：**汇编语言程序设计**　指导教师：**周丽华**
专业：**计算机科学与技术**　年级：**2020**　姓名：**胡诚皓**　学号：**20201060330**

实验十 声音程序设计

一、实验目的

1. 掌握响铃符的使用方法。
2. 掌握利用 PC 扬声器发出不同频率声音的方法。
3. 熟悉软硬件结合程序的编写方法。

二、实验内容

1. 复习相关 DOS 系统功能调用。
2. 复习有关 8255 和 8253 的相关内容。
3. 熟悉实验原理。
4. 阅读程序 1：

```
DATA    SEGMENT
DATA1   DB      'INPUT NUMBER1-8 (QUIT: Ctrl-C)--$'
DATA    ENDS
STACK   SEGMENT  PARA   STACK   'STACK'
STA     DW32 DUP(?)
STACK   ENDS
CODE    SEGMENT
        ASSUME   CS:CODE, DS:DATA, SS:STACK, ES:DATA
START:   MOV   AX,DATA
        MOV   DS,AX
        MOV   ES,AX
KKK:     MOV   AH,02H
        MOV   DL,0DH
        INT      21H
        MOV   AH,02H
        MOV   DL,0AH
        INT      21H
    MOV       AH,09H
        MOV   DX,    OFFSET DATA1
        INT   21H
        MOV   AH,01H
        INT   21H
        CMP   AL,03H
        JZ      PPP
        CMP   AL,30H
```

```
        JBE   TTT
        CMP   AL,39H
        JA    TTT
        SUB   AL,30H
        XOR   AH,AH
        MOV      BP,AX
GGG:    MOV   AH,02H
        MOV   DL,07H
        INT   21H
        mov     bx,     100
back:   mov     cx,     663
        PUSH    AX
WAITF1:  IN      AL,61H
        AND     AL,10H
        CMP     AL,AH
        JE      WAITF1
        MOV     AH,AL
        LOOP    WAITF1
        POP     ax
        dec     bx
        jnz     back
        DEC     BP
        JNZ   GGG
TTT:    JMP   KKK
PPP:    MOV   AX,4C00H
        INT   21H
CODE    ENDS
        END   START
```

5. 阅读程序 2：

```
DATA       SEGMENT
COUNT      DW    1
MESS       DB    'The bell is ring!', 0DH, 0AH, '$'
DATA       ENDS
CODE       SEGMENT
           ASSUME   CS: CODE, DS: DATA, ES: DATA
MAIN       PROC     FAR
```

学院：**信息学院**　学期：**2022 年秋季学期**　课程名称：**汇编语言程序设计**　指导教师：**周丽华**

专业：**计算机科学与技术**　年级：**2020**　姓名：**胡诚皓**　学号：**20201060330**

```
START:    MOV     AX,     DATA
          MOV     DS,     AX
          MOV     AL,     1CH
          MOV     AH,     35H
          INT     21H
          PUSH    ES
          PUSH    BX
          PUSH    DS
          MOV     DX, OFFSET   RING
          MOV     AX,     SEG   RING
          MOV     DS,     AX
          MOV     AL,     1CH
          MOV     AH,     25H
          INT     21H
          POP     DS
          IN      AL,     21H
          AND     AL,     11111110B
          OUT     21H,    AL
          STI
          MOV     DI,     60000
DELAY:    MOV     SI,     60000
DELAY1:   DEC     SI
          JNZ     DELAY1
          DEC     DI
          JNZ     DELAY
          POP     DX
          POP     DS
          MOV     AL,     1CH
          MOV     AH,     25H
          INT     21H
          MOV     AH,     4Ch
          INT     21H
MAIN      ENDP
RING:     PUSH    DS
          PUSH    AX
          PUSH    CX
          PUSH    DX
```

```
          MOV     AX,      DATA
          MOV     DS,      AX
          STI
          DEC     COUNT
          JNZ     EXIT
          MOV     DX,     OFFSET   MESS
          MOV     AH,     09H
          INT     21H
          MOV     DX,     100
          IN      AL,     61H
          AND     AL,     0FCH
SOUND:     XOR     AL,     02
          OUT     61H,    AL
          MOV     CX,     0F400H
WAIT1:     LOOP    WAIT1
          DEC     DX
          JNE     SOUND
          MOV     COUNT, 18
EXIT:      CLI
          POP     DX
          POP     CX
          POP     AX
          POP     DS
          sti
          IRET
CODE       ENDS
          END     START
```

6. 阅读程序 3：

```
STACK   SEGMENT PARA    STACK   'STACK'
       DB      64 DUP(0)
STACK   ENDS
DSEG    SEGMENT
MUS_FREQ        DW      330,294,262,294,3 DUP(330)
                DW      3 DUP (294),330,392,392
                DW      330,294,262,294,4 DUP(330)
                DW      294,294,330,294,262,-1
```

```
MUS_TIME        DW      6 DUP(25),50
                DW      2 DUP(25,25,50)
                DW      12 DUP(25),100
DSEG    ENDS

CSEG    SEGMENT
        ASSUME  CS:CSEG,SS:STACK,DS:DSEG
MUSIC   PROC
        MOV     AX,DSEG
        MOV     DS,AX
        LEA     SI,MUS_FREQ
        LEA     BP,DS:MUS_TIME
FREQ:
        MOV     DI,[SI]
        CMP     DI,-1
        JE      END_MUS
        MOV     BX,DS:[BP]
        PUSH    AX
        PUSH    BX
        PUSH    CX
        PUSH    DX
        PUSH    DI
        MOV     AL,0B6H
        OUT     43H,AL
        MOV     DX,12H
        MOV     AX,348CH
        DIV     DI
        OUT     42H,AL
        MOV     AL,AH
        OUT     42H,AL
        IN      AL,61H
        MOV     AH,AL
        OR      AL,3
        OUT     61H,AL
        add     bx,bx
        add     bx,bx
back:   mov     cx,     663
```

```
        PUSH    AX
WAITF1:
        IN      AL,61H
        AND     AL,10H
        CMP     AL,AH
        JE      WAITF1
        MOV     AH,AL
        LOOP    WAITF1
        POP     ax

        dec     bx
        jnz     back
        MOV     AL,AH
        OUT     61H,AL
        POP     DI
        POP     DX
        POP     CX
        POP     BX
        POP     AX
        ADD     SI,2
        ADD     BP,2
        JMP     FREQ
END_MUS:
        MOV     AX,4C00H
        INT     21H
MUSIC   ENDP
CSEG    ENDS
        END     MUSIC
```

7. 修改程序 3 中的数据，演奏其它乐曲。

三、实验结果（截图）

1. 程序 1 的功能分析

程序 1 是一个响铃程序，使用了响铃符（ASCII 码为 2）进行发声。程序会要求用户输入 1-8 之间的数字（记为 N），之后会响铃 N 次。程序中有输入验证，但是有点问题，1-9 都会被认为是有效输入。每次响铃之间的等待时间利用了 61H 端口的第五个比特位信息实现延时。细节上的理解详见下面代码中的注释：

```
DATA    SEGMENT
```

```
    DATA1   DB  'INPUT NUMBER1-8 (QUIT: Ctrl-C)--$'
DATA    ENDS

STACK   SEGMENT PARA    STACK   'STACK'
    STA DW 32 DUP(?)
STACK   ENDS

CODE    SEGMENT
    ASSUME  CS:CODE, DS:DATA, SS:STACK, ES:DATA
START:
    ; 设置段寄存器
    MOV AX, DATA
    MOV DS, AX
    MOV ES, AX
KKK:
    ; 输出换行回车
    MOV AH, 02H
    MOV DL, 0DH
    INT 21H
    MOV AH, 02H
    MOV DL, 0AH
    INT 21H
    ; 输出提示字符串
    MOV AH, 09H
    MOV DX, OFFSET DATA1
    INT 21H
    ; 读取输入
    MOV AH, 01H
    INT 21H
    ; 输入 Ctrl-C
    CMP AL, 03H
    JZ  PPP
    ; 输入不在 1~9 内就跳转 TTT
    CMP AL, 30H
    JBE TTT
    CMP AL, 39H
    JA  TTT
```

```
    ; 将数字的 ASCII 转为值
    SUB AL, 30H
    XOR AH, AH
    MOV BP, AX
GGG:
    ; 输出响铃符
    MOV AH, 02H
    MOV DL, 07H
    INT 21H
    ; BX 为等待循环的次数控制
    mov bx, 100
back:
    ; CX 控制每一次的等待循环中等待的时钟刷新次数
    ;（事实上只是个大概，并不绝对精确）
    mov cx,663
    PUSH AX
WAITF1:
    ; 取得 61H 端口第 5 个比特的信息，是随时钟刷新而变化的
    IN  AL,61H
    AND AL,10H
    ; 检查时钟是否反转
    CMP AL,AH
    JE WAITF1
    ; 时钟反转则将此时的状态记录在 AH 中
    MOV AH,AL
    ; 一次时钟刷新结束，跳转，开始检测下一次
    LOOP WAITF1
    ; CX 次时钟刷新完成，即一次等待循环完成
    ; 此时将 BX-1，开始下一次等待循环
    POP ax
    dec bx
    jnz back
    ; BX 次等待循环完成
    DEC BP
    JNZ GGG
TTT: ; 输入不合法就要求重新输入
    JMP KKK
```

```
PPP:  ; 退出代码
    MOV AX, 4C00H
    INT 21H
CODE    ENDS
    END START
```

2. 程序 2 的功能分析

程序 2 利用 1CH 的定时器软中断,使得在一定时间内一段段不断进行响铃发声。将 RING 标签处的程序入口地址作为 1CH 的中断向量，这样定时器每次进入软中断时，就能调用 RING 处代码进行发声。发声时使用了“振荡”的方法，不断开启关闭声音一定次数完成一次“长时间”发声。需要注意的是，为了使各轮次发声可以清晰地辨别，使用 COUNT 变量进行计数，使得每触发 18 个定时器软中断进行一轮响铃发声。细节上的理解详解下面代码中的注释：

```
DATA    SEGMENT
    COUNT DW 1
    MESS DB 'The bell is ring!', 0DH, 0AH, '$'
DATA    ENDS
CODE    SEGMENT
    ASSUME   CS: CODE, DS: DATA, ES: DATA
MAIN    PROC    FAR
START:
    ; 设置段寄存器
    MOV AX, DATA
    MOV DS, AX
    ; 获得 1CH 中断向量的地址存在 ES:BX
    MOV AL, 1CH
    MOV AH, 35H
    INT 21H
    ; 保护获得的地址
    PUSH ES
    PUSH BX
    PUSH DS
    ; 获得标签 RING 代码处的段地址和偏移地址存于 AX:DX
    MOV DX, OFFSET RING
    MOV AX, SEG RING
    ; 设置中断向量 1CH 为 DS:DX，即标签 RING 处
    MOV DS, AX
```

```
    MOV AL, 1CH
    MOV AH, 25H
    INT 21H
    ; 恢复 DS
    POP DS
    ; 获取中断开关情况，若定时器中断关，则将其开启
    IN  AL, 21H
    AND AL, 11111110B
    OUT 21H, AL
    ; 开中断
    STI

; 使用 SI、DI 做两层循环，通过单纯循环完成延时
; 在这个延时中，定时器不断触发 1CH 的软中断，反复响铃
    MOV DI, 60000
DELAY:
    MOV SI, 60000
DELAY1:
    DEC SI
    JNZ DELAY1
    DEC DI
    JNZ DELAY
    ; 恢复寄存器内容，并将 1CH 中断向量原先指向的地址恢复
    POP DX
    POP DS
    MOV AL, 1CH
    MOV AH, 25H
    INT 21H
    ; 程序结束
    MOV AH, 4Ch
    INT 21H
MAIN    ENDP

RING:
    ; 保护寄存器
    PUSH DS
    PUSH AX
```

```
    PUSH CX
    PUSH DX
    ; 切换到正常的数据段上
    MOV AX, DATA
    MOV DS, AX
    STI
    DEC COUNT
    JNZ EXIT
    ; 显示字符串
    MOV DX, OFFSET MESS
    MOV AH, 09H
    INT 21H
    ; 响铃次数
    MOV DX, 50
    IN AL, 61H
    ; 清零低 2 位
    AND AL, 0FCH
SOUND:
    ; 翻转第二位，开启或关闭声音
    XOR AL, 02
    OUT 61H, AL
    ; 等待循环
    MOV CX, 0F400H
WAIT1:
    LOOP WAIT1
    DEC DX
    JNE SOUND
    ; 每 COUNT 个 1CH 中断响铃一次
    MOV COUNT, 18
EXIT:
    ; 关中断
    CLI
    POP DX
    POP CX
    POP AX
    POP DS
    ; 开中断
```

```
    sti
    ; 从栈中弹出 IP、CS 和标志位寄存器
    IRET
CODE  ENDS
    END START
```

3. 程序 3 的功能分析

程序 3 不使用响铃符进行发声，而是直接给入扬声器数据进行特定频率的发声。事实上，通过预定义的频率及播放的时间长度，程序 3 实现了播放“玛丽有只小绵羊”的曲子。主要使用到了 43H 端口（用于初始化扬声器设置）、42H 端口（用于设置扬声器的发声数据，即频率）、61H 端口（低两位用于设置扬声器的开关与数据读取的开关）。细节处的理解参见下面代码中的注释：

```
STACK   SEGMENT PARA    STACK   'STACK'
    DB 64 DUP(0)
STACK   ENDS

DSEG    SEGMENT
    MUS_FREQ    DW 330,294,262,294,3 DUP(330)
                DW 3 DUP (294),330,392,392
                DW 330,294,262,294,4 DUP(330)
                DW 294,294,330,294,262,-1
    MUS_TIME    DW 6 DUP(25),50
                DW 2 DUP(25,25,50)
                DW 12 DUP(25),100
DSEG    ENDS

CSEG    SEGMENT
    ASSUME  CS:CSEG,SS:STACK,DS:DSEG
MUSIC   PROC
    ; 设置段寄存器
    MOV AX,DSEG
    MOV DS,AX
    ; 取得频率及时间的偏移地址
    LEA SI,MUS_FREQ
    LEA BP,DS:MUS_TIME
FREQ:
    ; 判断是否到最后一个音
```

```
    ; DI 存音调，BX 存音长
    MOV DI,[SI]
    CMP DI,-1
    JE END_MUS
    MOV BX,DS:[BP]
    ; 保护寄存器
    PUSH AX
    PUSH BX
    PUSH CX
    PUSH DX
    PUSH DI
    ; 向 43H 端口输出 182，初始化扬声器的设置
    MOV AL,0B6H
    OUT 43H,AL
    ; 计算音调号=12348H/频率，得到的结果在 AX 中
    MOV DX,12H
    MOV AX,348CH
    DIV DI
    ; 42H 端口宽度为 8 位，AX 需要分两次传入，先低字节再高字节
    OUT 42H,AL
    MOV AL,AH
    OUT 42H,AL
    ; 读取 61H 端口内容，并把低两位置为 1，使得扬声器发声
    IN  AL,61H
    MOV AH,AL
    OR AL,3
    OUT 61H,AL
    ; bx=bx*4
    add bx,bx
    add bx,bx
back:
    ; 等待 663 个时钟刷新为一个等待循环
    mov cx, 300
    ; 此时 AH 为 61H 端口原状态，AL 为发声状态
    PUSH AX
WAITF1:  ; 等待一个时钟刷新时间（非精确）
    ; 取时钟刷新切换位
```

```
    IN AL,61H
    AND AL,10H
    CMP AL,AH
    JE  WAITF1
    MOV AH,AL
    LOOP WAITF1
    POP ax
    ; 按照音长进行等待，等待期间一直发声
    dec bx
    jnz back
    ; 向 61H 端口输出其原来未发声的状态
    MOV AL,AH
    OUT 61H,AL
    ; 恢复寄存器
    POP DI
    POP DX
    POP CX
    POP BX
    POP AX
    ; 播放下一个音
    ADD SI,2
    ADD BP,2
    JMP FREQ
END_MUS:  ; 退出代码
    MOV AX,4C00H
    INT 21H
MUSIC   ENDP
CSEG    ENDS
    END MUSIC
```

4. 修改程序 3 中的数据，演奏其它乐曲

修改 MUS_FREQ 与 MUS_TIME 处的值，使其演奏“两只老虎”，修改后的数据段如下，其他部分与程序 3 保持一致：

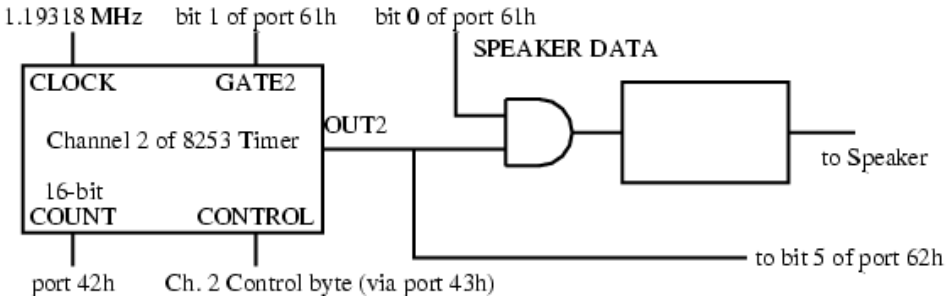
```
DSEG    SEGMENT
    MUS_FREQ    DW 2 DUP(262,294,330,262)
                DW 2 DUP(330,349,391)
                DW 2 DUP(391,440,391,349,330,262)
```

```
                DW 330,196,262
                DW 294,196,262,-1
    ; 一拍设为 12
    MUS_TIME    DW 2 DUP(12, 12, 12, 12)
                DW 2 DUP(12,12,24)
                DW 2 DUP(6,6,6,6,12,12)
                DW 2 DUP(12,12,24)
DSEG    ENDS
```

四、实验报告要求（习题）

1. 扬声器发声原理简述

PC 中有内置的扬声器，能发出不同频率的“哔”，可以向对应端口传入频率数据，然后将扬声器打开，从而实现播放。一般来说，会使用 8253 计时器芯片的 Channel 2 生成特定频率的振荡让扬声器发声。给入的频率数据是一个字长的值，由于输入端口宽度为 8bit，因此需要分两次输入。给入的频率数据并不是频率本身的值，而是介于 0-65535 之间的值，称为频率数。振荡器芯片生成的基本频率为 1193180Hz，这与频率数之间呈反比，也就是频率数和实际频率值的乘积始终为 1193180。例如当频率数取最大 65535 时，产生的频率为 1193180/65535=18.2Hz。下面是一个框图，表示了 PC 内置扬声器部分的结构。



上图指出了使用扬声器进行发声时要用到的一些端口。输出 OUT2 是 8253-5 定时器芯片通道 2 的输出；控制端口 CONTROL 用于扬声器部分的功能初始化，可以在初始化扬声器时设置其工作模式及状态；数据输入端口 COUNT 用于给入扬声器播放波形的相关数据；GATE2 是 8253 计数器通道 2 的使能端；SPEAKER DATA 可以独立用于调制输出波形。

一般来说，要产生声音需要以下步骤：

（1）发送值 182 到端口 43H 以初始化设置扬声器

（2）将频率号发送到端口 42H，需要两次 OUT 发送，先低字节再高字节。然后将端口 61H 的低两比特位设置为 1（注意不要修改 61H 的其他 bit）

（3）等待一段时间让扬声器持续发声

(4)将端口 61H 的低两比特位设置为 0(同样注意不要修改 61H 的其他 bit)

2. 遇到的问题及解决方法

在进行此次声音程序设计的过程中，一开始在 DOSBOX 中汇编、链接、运行程序后没有任何的声音，响铃符也没有任何作用，在查看 DOSBOX 的配置文件后发现没有任何问题 MIDI 设备的选择和连接也没有任何问题，但就是无法发出声音。在查找相关资料以求解决该问题的过程中，发现了一个基于 DOSBOX 开发的增强版 dos 模拟器 DOSBox-X（官网：https://dosbox-x.com/）支持非常多的功能，甚至包括了切换不同 CPU（80186、80286、80386 等）的功能，最后转而使用之，问题完美解决。

五、个人体会与总结

本次实验中了解了扬声器的基本发声原理,同时这次实验中的等待延迟代码涉及到了与 CPU 时钟的交互。另外，在程序 2 中还涉及到了中断向量的设置，通过定时器来实现定时触发某段代码。在此过程中，使用到了 STI 和 CLI，即对 IF 寄存器的操作。需要注意的是，IF=0 表示的关中断只关闭外部可屏蔽中断。

根据中断源将中断分为外部中断与内部中断。外部中断,即外围设备的中断,又可分为不可屏蔽中断与可屏蔽中断。内部中断，则又可分为陷阱（人为预先设置）、故障和终止三种。