实 验 报 告

PB12000646 陈思杰

【源代码】

```
;IRQ1，IRQ2 级联单中断应用实验 实验程序清单（8259_2.ASM）

;宏定义
IRQ10_IVADD EQU 01C8H          ;IRQ10 对应的中断矢量地址
IRQ10_OCW1 EQU 0A1H            ;IRQ10 对应 PC 机内部 8259 的 OCW1 地址
IRQ10_OCW2 EQU 0A0H            ;IRQ10 对应 PC 机内部 8259 的 OCW2 地址
IRQ10_IM EQU 0FBH              ;IRQ10 对应的中断屏蔽字

IRQ7_IVADD EQU 003CH           ;IRQ7 对应的中断矢量地址
IRQ7_OCW1 EQU 021H             ;IRQ7 对应 PC 机内部 8259 的 OCW1 地址
IRQ7_OCW2 EQU 020H             ;IRQ7 对应 PC 机内部 8259 的 OCW2 地址
IRQ7_IM EQU 07FH               ;IRQ7 对应的中断屏蔽字

;堆栈以及数据段
STACK1 SEGMENT STACK
DW 256 DUP(?)
STACK1 ENDS

DATA SEGMENT
MES DB 'Press any key to exit!',0AH,0DH,0AH,0DH,'$'
CS10_BAK DW ?                  ;保存 IRQ10 原中断处理程序入口段地址的变量
IP10_BAK DW ?                  ;保存 IRQ10 原中断处理程序入口偏移地址的变量
IM10_BAK DB ?                  ;保存 IRQ10 原中断屏蔽字的变量
CS7_BAK DW ?                   ;保存 IRQ7 原中断处理程序入口段地址的变量
IP7_BAK DW ?                   ;保存 IRQ7 原中断处理程序入口偏移地址的变量
IM7_BAK DB ?                   ;保存 IRQ7 原中断屏蔽字的变量
```

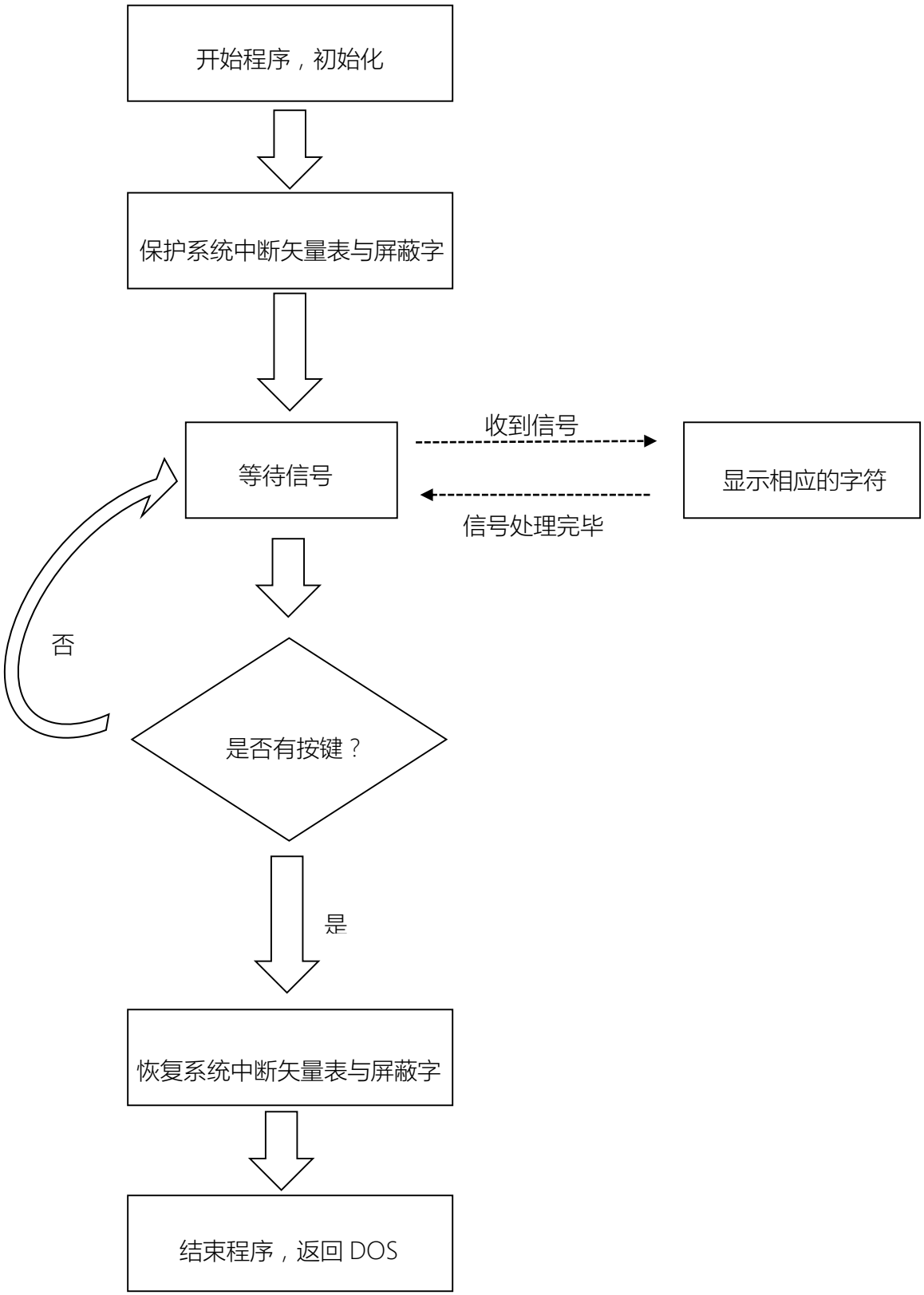
```
DATA ENDS


;代码段
CODE SEGMENT
                ASSUME CS:CODE,DS:DATA
START:          MOV AX,DATA
                MOV DS,AX

                MOV DX,OFFSET MES       ;显示退出提示
                MOV AH,09H
                INT 21H

        ;保存系统的中断矢量表
        SAVE:
                CLI
                MOV AX,0000H            ;替换 IRQ10 的中断矢量
                MOV ES,AX
                MOV DI,IRQ10_IVADD
                MOV AX,ES:[DI]
                MOV IP10_BAK,AX         ;保存 IRQ10 原中断处理程序入口偏移地址
                MOV AX,OFFSET MYISR10
                MOV ES:[DI],AX          ;设置当前中断处理程序入口偏移地址
                ADD DI,2
                MOV AX,ES:[DI]
                MOV CS10_BAK,AX         ;保存 IRQ10 原中断处理程序入口段地址
                MOV AX,SEG MYISR10
                MOV ES:[DI],AX          ;设置当前中断处理程序入口段地址
                MOV DX,IRQ10_OCW1       ;设置中断屏蔽寄存器，打开 IRQ10 的屏蔽位
                IN AL,DX
                MOV IM10_BAK,AL         ;保存 IRQ10 原中断屏蔽字
                AND AL,IRQ10_IM
                OUT DX,AL
```

```
        MOV AX, 0000H               ;替换 IRQ7 的中断矢量
        MOV ES, AX
        MOV DI, IRQ7_IVADD
        MOV AX, ES:[DI]
        MOV IP7_BAK, AX             ;保存 IRQ7 原中断处理程序入口偏移地址
        MOV AX, OFFSET MYISR7
        MOV ES:[DI],AX              ;设置当前中断处理程序入口偏移地址
        ADD DI, 2
        MOV AX, ES:[DI]
        MOV CS7_BAK, AX             ;保存 IRQ7 原中断处理程序入口段地址
        MOV AX, SEG MYISR7
        MOV ES:[DI],AX              ;设置当前中断处理程序入口段地址
        MOV DX, IRQ7_OCW1           ;设置中断屏蔽寄存器，打开 IRQ7 的屏蔽位
        IN AL, DX
        MOV IM7_BAK,AL              ;保存 IRQ7 原中断屏蔽字
        AND AL,IRQ7_IM
        OUT DX,AL
        STI

;直到有按键按下之前一直等待中断的到来
WAIT1:
        MOV AH,1                    ;判断是否有按键按下
        INT 16H
        JZ WAIT1                    ;无按键则跳回继续等待，有则退出

;有按键按下，则恢复系统中断矢量表
RECVSYS:
        CLI
        MOV AX, 0000H               ;恢复 IRQ10 原中断矢量
        MOV ES, AX
        MOV DI, IRQ10_IVADD
        MOV AX, IP10_BAK            ;恢复 IRQ10 原中断处理程序入口偏移地址
        MOV ES:[DI],AX
        ADD DI, 2
```

```
                MOV AX, CS10_BAK        ;恢复 IRQ10 原中断处理程序入口段地址
                MOV ES: [DI],AX
                MOV DX, IRQ10_OCW1      ;恢复 IRQ10 原中断屏蔽寄存器的屏蔽字
                MOV AL, IM10_BAK
                OUT DX, AL

                MOV AX, 0000H           ;恢复 IRQ7 原中断矢量
                MOV ES, AX
                MOV DI, IRQ7_IVADD
                MOV AX, IP7_BAK         ;恢复 IRQ7 原中断处理程序入口偏移地址
                MOV ES:[DI],AX
                ADD DI, 2
                MOV AX, CS7_BAK         ;恢复 IRQ7 原中断处理程序入口段地址
                MOV ES:[DI],AX
                MOV DX, IRQ7_OCW1       ;恢复 IRQ7 原中断屏蔽寄存器的屏蔽字
                MOV AL, IM7_BAK
                OUT DX, AL
                STI
        QUIT:
                MOV AX,4C00H            ;返回到 DOS
                INT 21H



MYISR10         PROC NEAR               ;处理 IRQ1 中断处理的程序 MYISR10
                PUSH AX
                MOV AL, 31H             ;调用 BIOS 调用在屏幕上显示一个 1
                MOV AH, 0EH
                INT 10H
                MOV AL, 20H
                INT 10H
        OVER10: MOV DX, IRQ10_OCW2      ;向 PC 机内部 8259 发送两次中断结束命令
                MOV AL, 20H
                OUT DX, AL
```

```
                MOV AL, 20H
                OUT 20H, AL
                POP AX
                IRET
                MYISR10 ENDP

MYISR7          PROC NEAR           ;处理 IRQ2 中断处理的程序 MYISR7
                PUSH AX
                MOV AL, 32H         ;调用 BIOS 调用在屏幕上显示一个 2
                MOV AH, 0EH
                INT 10H
                MOV AL, 20H
                INT 10H
        OVER7:  MOV DX, IRQ7_OCW2   ;向 PC 机内部 8259 发送一次中断结束命令
                MOV AL, 20H
                OUT DX, AL
                POP AX
                IRET
                MYISR7 ENDP

CODE ENDS
END START
```

【流程图】


开始程序，初始化
保护系统中断矢量表与屏蔽字
等待信号
收到信号
显示相应的字符
信号处理完毕
否
是否有按键？
是
恢复系统中断矢量表与屏蔽字
结束程序，返回 DOS

【实验心得】

经过这一系列的实验，我对汇编语言和 8086CPU 及其外设多了许多深入的认识。

汇编语言作为一种较为低级的语言，相对于 C 这样的高级语言，其语法与结构都要简单得多。8086CPU 的指令系统虽然丰富，但大部分很少用到。但其缺点也是显而易见的，比如对于阶乘程序，在 C 中很容易以递归方式完成，但是在汇编语言中，受到储存空间的限制，同样的算法就变得很难实现。而对其较大整数的结果输出，也体现了汇编语言设计难度高、可读性差、可维护性差的特点。

在对硬件编程方面，与 LabView 这种高级编程环境相比，汇编语言的语法和指令都更为显得简单。但为此付出的代价是巨大的，汇编语言从更低的层次操作硬件，因而要求程序设计者了解大量关于硬件的信息，其中不少甚至是无法通过软件方式获得的，比如在编写音乐播放程序时需要知道外接时钟的频率，在编写延时程序时需要较精确地了解单条指令的执行时间。这些都是在使用高级语言时被包装的细节，是从来不需要考虑的。

实验中总共涉及了 4 种 8086CPU 的外围可编程芯片，这些芯片看似功能简单，但组合起来却可以完成一些诸如播放音乐，之前认为非常高级的任务。这几种芯片虽然互相之间功能差异很大，但是其操作方式却非常相似，几乎都是复位后先设置方式控制字然后设置操作控制字的模式，掌握一种芯片的使用后就很容易掌握其余几种，这可能也是 Intel 成功的原因之一吧。