微机原理与接口技术课程实践指导书

易 丛 琴　编著

上海海洋大学海洋智能信息实验教学示范中心

第一篇　汇编语言实验

实验一　实验平台搭建及“Hello World”的汇编语言实现

一、实验目的

1、能够在 win10 下实现 8086 汇编程序设计

2、实现 win32 汇编工作环境搭建，能在 win32 实现汇编程序设计

3、掌握 EMU 8086 软件的使用

二、实验环境

1、DOSBox

2、masm32v11r

3、EMU 8086

三、实验内容：

1、win10 下实现 8086 汇编程序

2、win32 汇编工作环境搭建

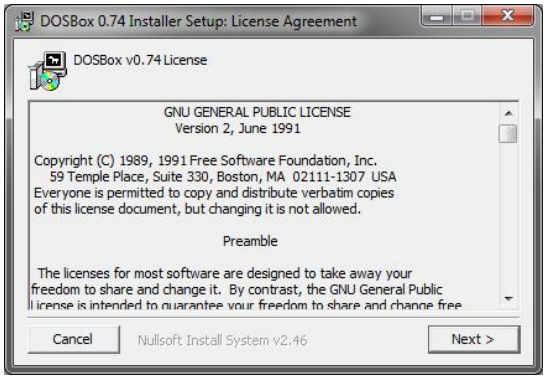
3、EMU 8086 软件使用

四、实验步骤

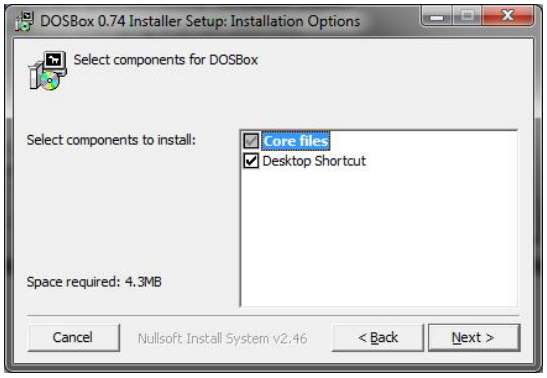
一、 win10 下实现 8086 汇编程序

4、 安装 DOSBox

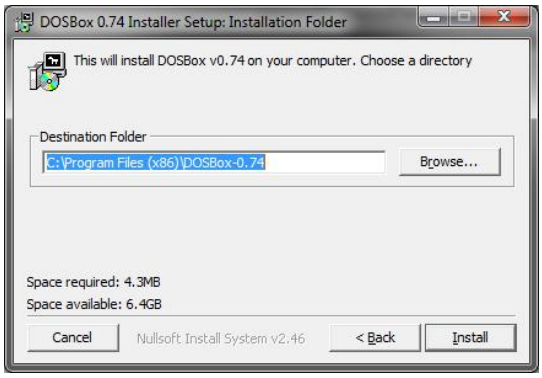
双击 DOSBox0.74-win32-installer.exe



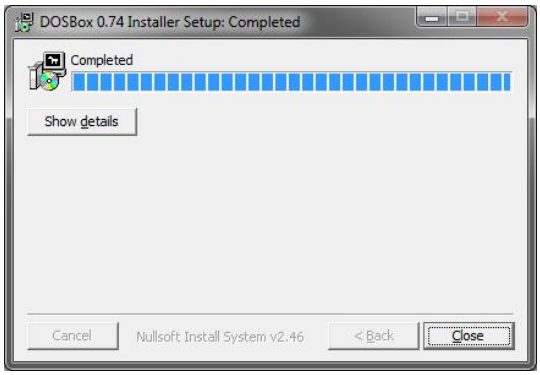
Next



Next

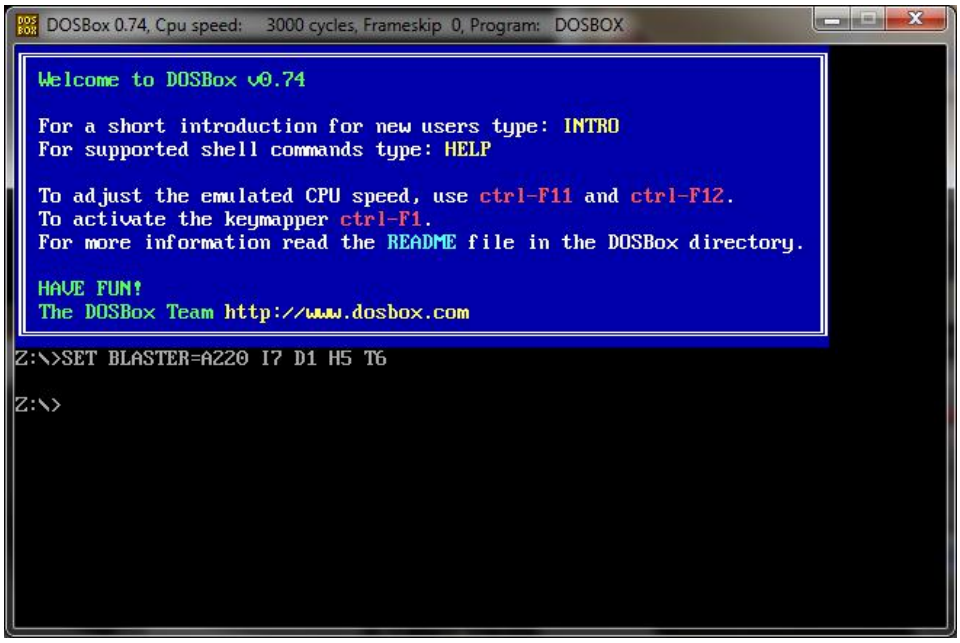


Install



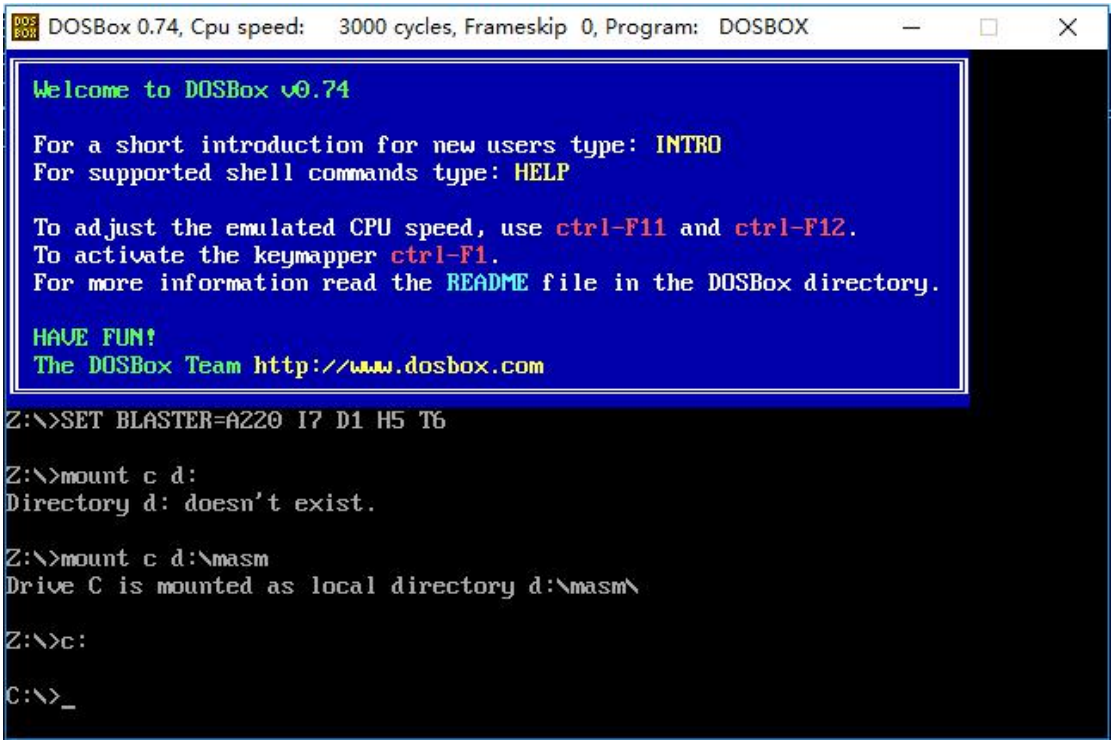
5、 运行 DOSBox

双击桌面的 DOSBox 快捷方式

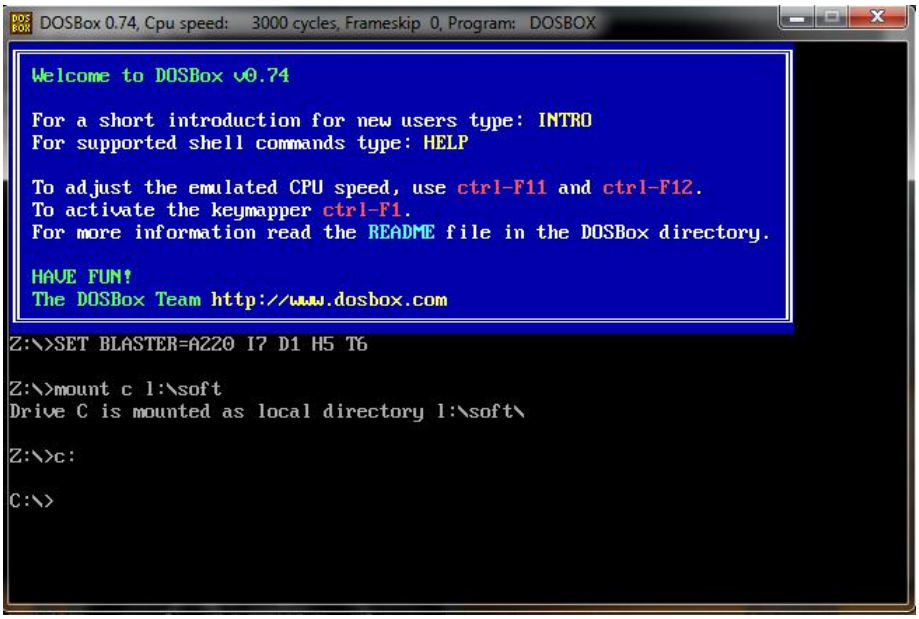


将 MASM 文件夹拷贝到一个目录下，比如 D:\MASM 下，然后将这个目录挂着为 DOSBox 的一个盘符下，挂载命令为

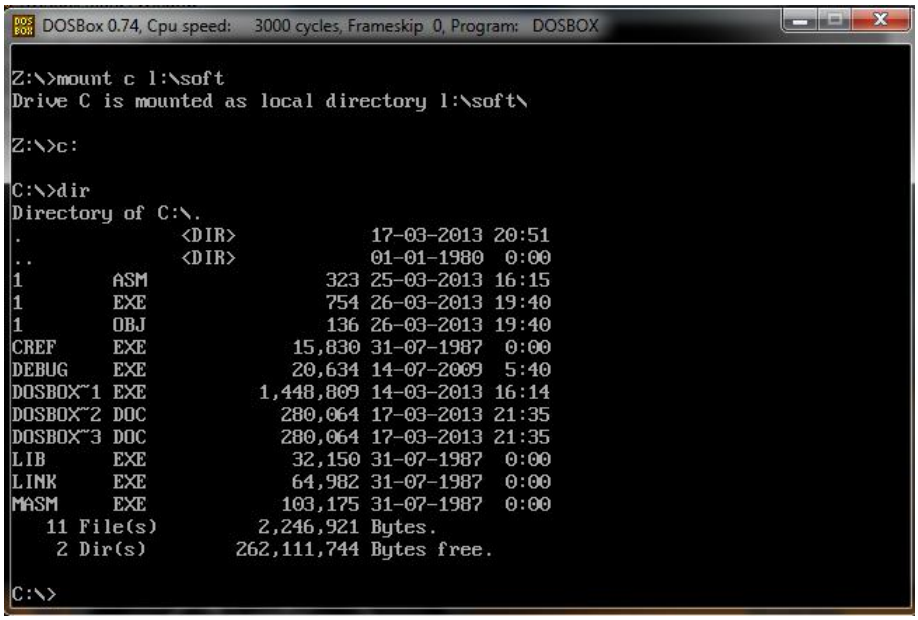
Mount c D:\MASM



切换到挂载的 c 盘



显示下一下 DOSbox 中 c 盘下面的文件 C:\>dir



之后就可以运行 MASM, LINK 了

2、8086 汇编程序实现

用 Notepad++编辑屏幕显示 helloworld!的汇编程序，命名为*.asm

```
assume cs:code,ds:datas
datas segment
str db 'helloWorld!','$'
datas ends
code segment
    mov ax,datas
    mov ds,ax
    lea dx,str   ；获取 str 的偏移地址
    mov ah,9 ；调用 9 号功能输出字符串
    int 21h

    mov ah,4ch
```

```
    int 21h
code ends
end
```

在 DosBox 中进行输入 masm　*.asm 进行编译，连续回车，link *.obj 进行连接。

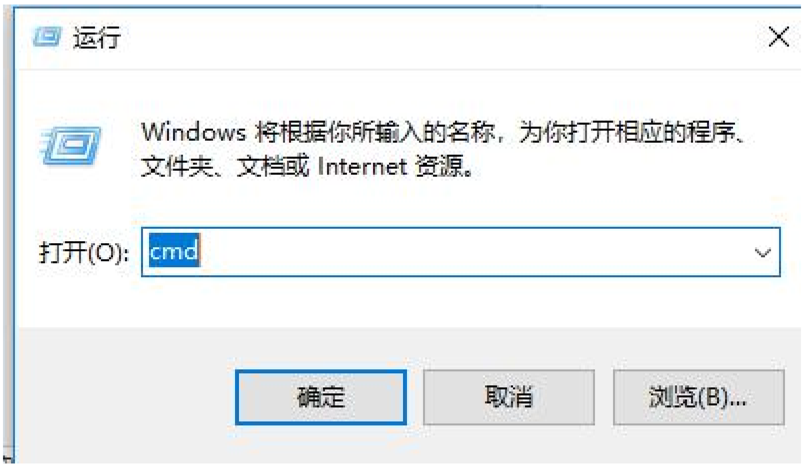
二、 win32 汇编工作环境搭建

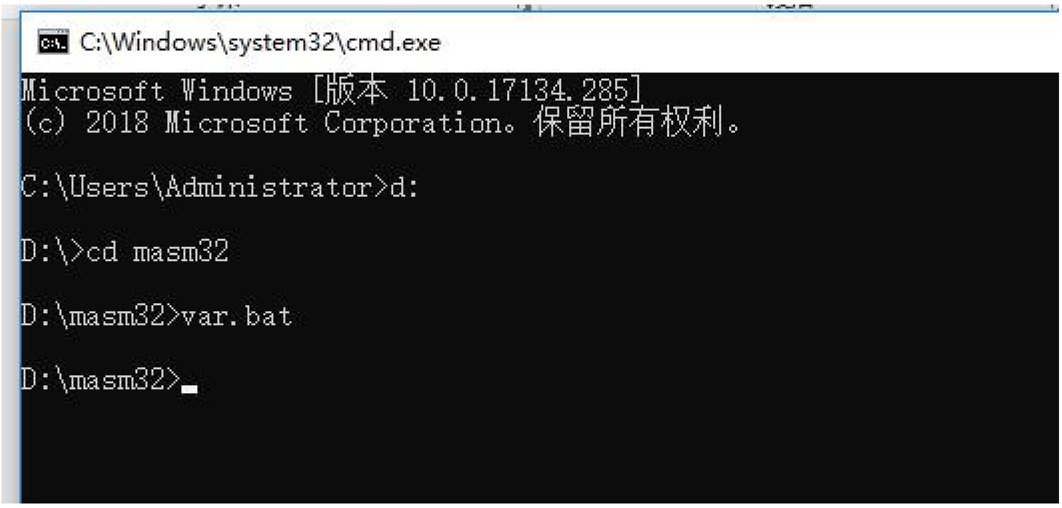
1、 安装 masm32v11r，注意安装于根目录；假设安装于 D 盘下

2、 设置环境变量：Notepad++编辑，存储为 Var.bat

```
@echo off
set Masm32Dir=D:\masm32
set include=%Masm32Dir%\Include;
set lib=%Masm32Dir%\lib;
set path=%Masm32Dir%\Bin;%Masm32Dir%;
```

3、





3、 win32 汇编实现“Hello world！”

masm32 下建一个文件夹如 source，再建一个 test 文件夹，保存 Hello.asm。

;>>>;

Hello.asm

; 使用 Win32ASM 写的 Hello, world 程序

;>>>;

使用 nmake 或下列命令进行编译和链接:

; ml /c /coff Hello.asm

```
; Link /subsystem:windows Hello.obj
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
        .386
        .model flat,stdcall
        option casemap:none
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; Include  文件定义
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
include         windows.inc
include         user32.inc
includelib  user32.lib
include         kernel32.inc
includelib  kernel32.lib
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;  数据段
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
        .data
szCaption db  'A MessageBox !',0
szText      db  'Hello, World !',0
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>;
代码段
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
        .code
start:
        invoke MessageBox,NULL,offset                szText,offset
szCaption,MB_OK
        invoke ExitProcess,NULL
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
        end     start
```

使用 ml /c /coff Hello.asm

; Link /subsystem:windows Hello.obj

运行 Hello 观察结果。

三、 EMU 8086 软件使用

EMU 8086 是一款基于 Windows 的集编辑、汇编、链接、调试运行于一体的可视化的 8086 CPU 仿真软件，有着与 8086 十分相近的功能。EMU8086 是交互式学习汇编语言（Assembly Language）、计算机结构（Computer Architecture）和逆向工程（Reverse Engineering）的完整仿真体系。其内部集成了汇编程序汇编器、链接器、虚拟硬件、参考资料、例程、学习指南等。EMU8086 是学习 Intel 8086 微处理器的理想工具，它模拟真实微处理器的每一步骤，并显示内部寄存器、存储器、堆栈、变量和标志寄存器，而且其中任何一个数值都可通过鼠标双击来改变。同时它还提供了微机显示器、直流步进电机、交通灯、LED 等虚拟外设。

以 HELLO 程序为例，参考例程如下所示：

```
;*************************************************************
DSEG    SEGMENT
MSG1    DB 'Hello,this is a sample program!',0DH,0AH,'$'
```

```
DSEG     ENDS
;*****************************************************************
SSEG     SEGMENT STACK
ST1      DB 100 DUP()
TDP      EQU $-ST1
SSEG     ENDS
;*****************************************************************
CSEG     SEGMENT
         ASSUME CS:CSEG,DS:DSEG,SS:SSEG
START: MOV AX,DSEG
         MOV DS,AX
         MOV AX,SSEG
         MOV SS,AX
         MOV SP,TDP
         MOV AH,9
         MOV DX,OFFSET MSG1
         INT 21H
         MOV AH,4CH
         INT 21H
CSEG     ENDS
         END START
;*****************************************************************
```

双击 EMU8086 图标，打开程序。此时会弹出 welcome 窗口，见图 1-1 所示，可将该窗口关闭直接进入编辑状态；也可点击它的 new 按钮，则会弹出代码模板选择框供用户选择（可选择 EXE 模板），见图 1-2 所示，一般点击 Cancel 按钮，直接进入编辑状态。在编辑窗口，可直接将源程序输入，见图 1-3 所示。

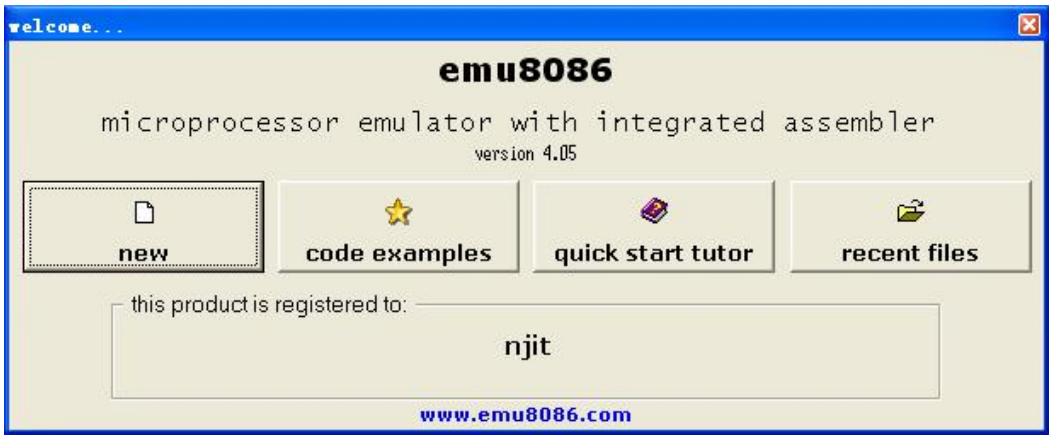


图 1-1 welcome 窗口

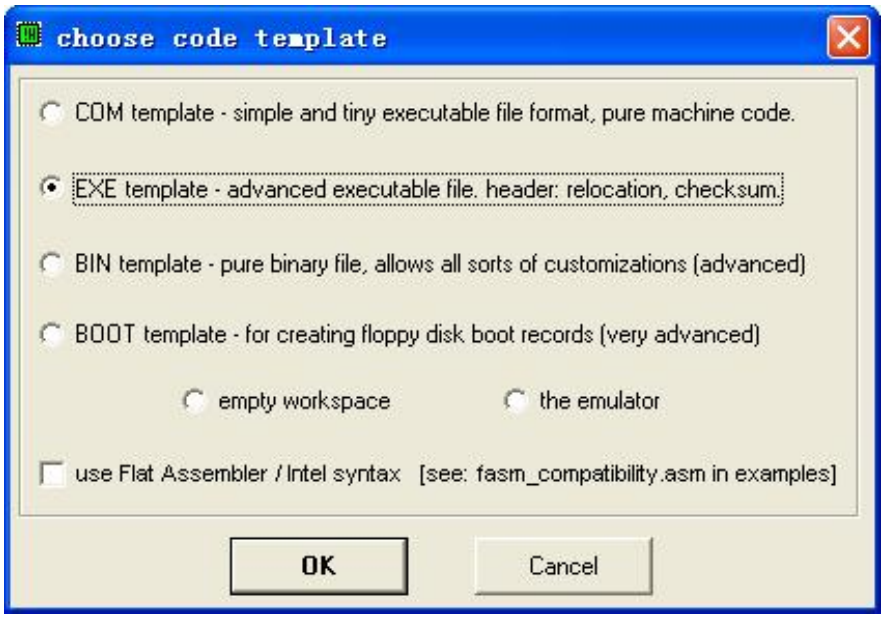


图 1-2 模板选择

- COM 模板——适用于简单且不需分段的程序，所有内容均放在代码段中，程序代码默认从 ORG 0100H 开始；

- EXE 模板——适用于需分段的复杂程序，内容按代码段、数据段、堆栈段划分。需要注意的是采用该模板时，用户不可将代码段人为地设置为 ORG　0100H，而应由编译器自动完成空间分配；
- BIN 模板——二进制文件，适用于所有用户定义结构类型；
- BOOT 模板——适用于在软盘中创建文件。

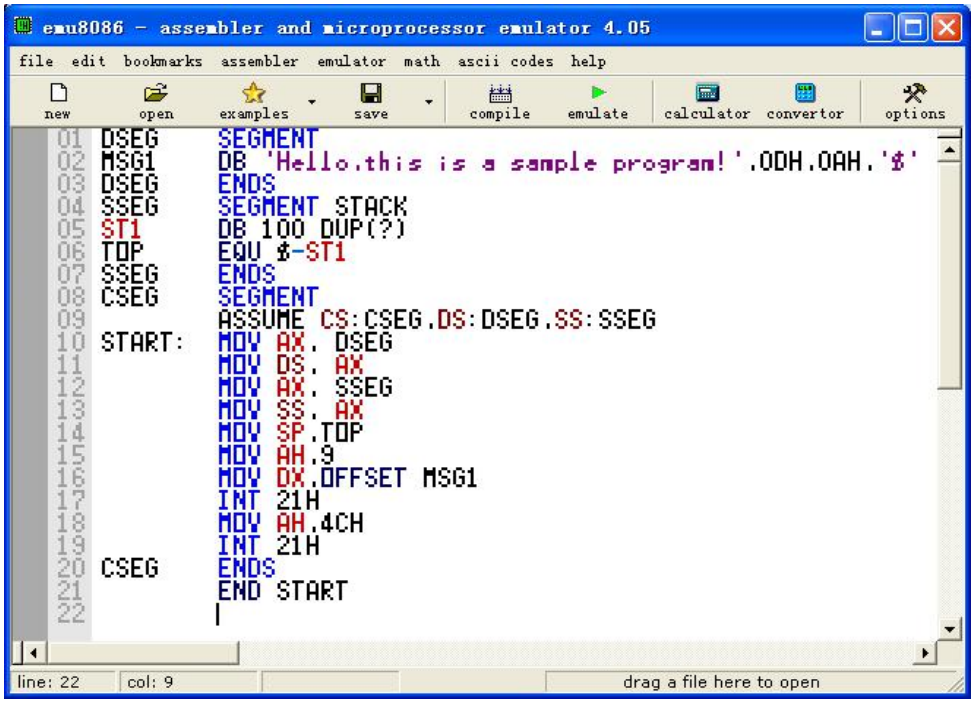


图 1-3 编辑窗口输入源程序

点击 compile（编译）按钮则进行汇编和链接，生成.EXE 文件，并提示存盘。

点击 emulate（仿真）按钮则可进入 8086 仿真器界面，见图 1-4 所示。界面中提供了寄存器窗口、内存单元窗口及反汇编窗口，下方一行按钮则提供了虚拟屏幕、源代码观察、复位、辅助工具、变量、DEBUG、堆栈及 FLAGS（标志寄存器）观察等功能。

用户可点击 run 连续运行程序，也可点击 single step 对程序进行单步

调试。

图 1-5 为 HELLO.EXE 的运行结果，用户通过虚拟屏幕观察，非常形象生动。

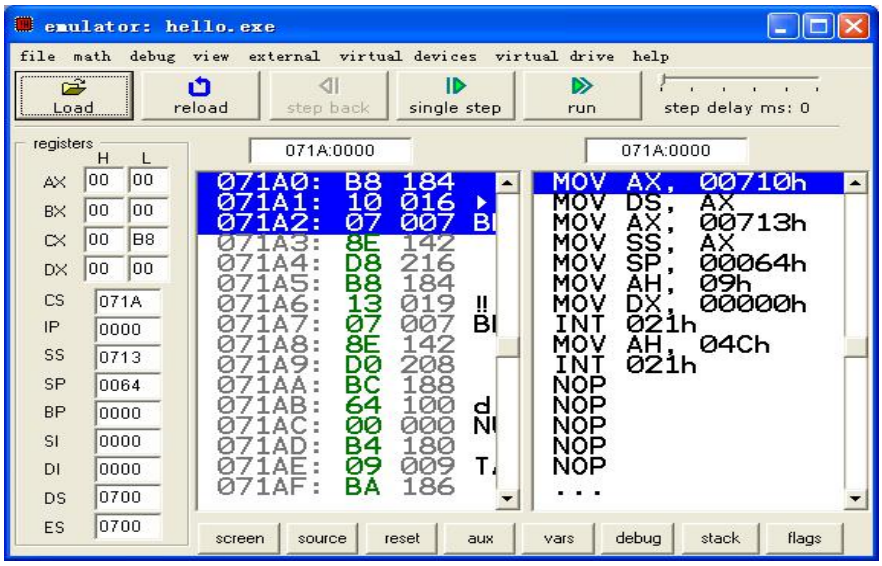
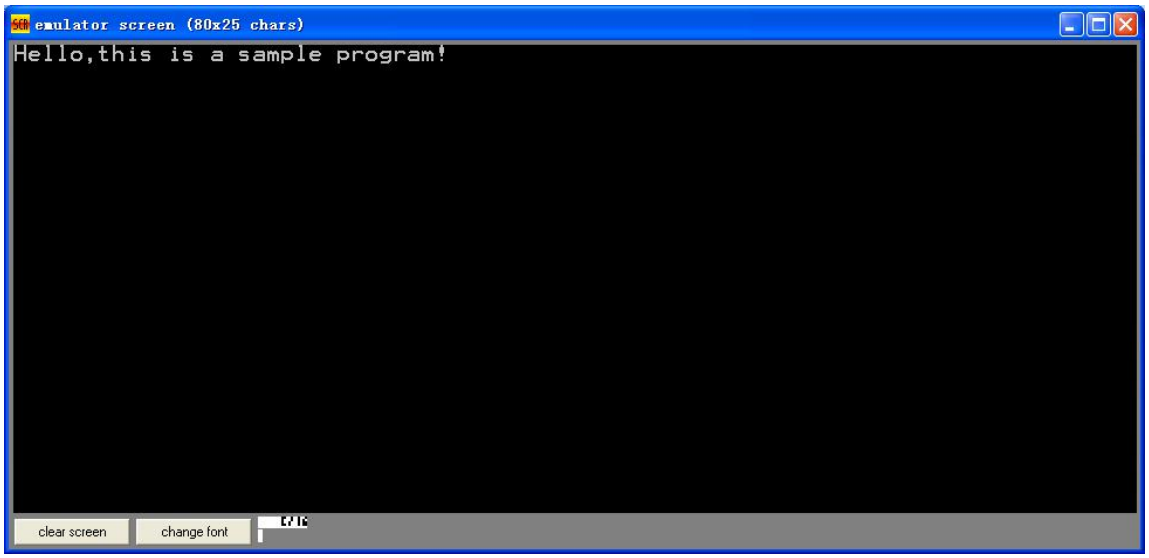


图 1-4 8086 仿真器界面



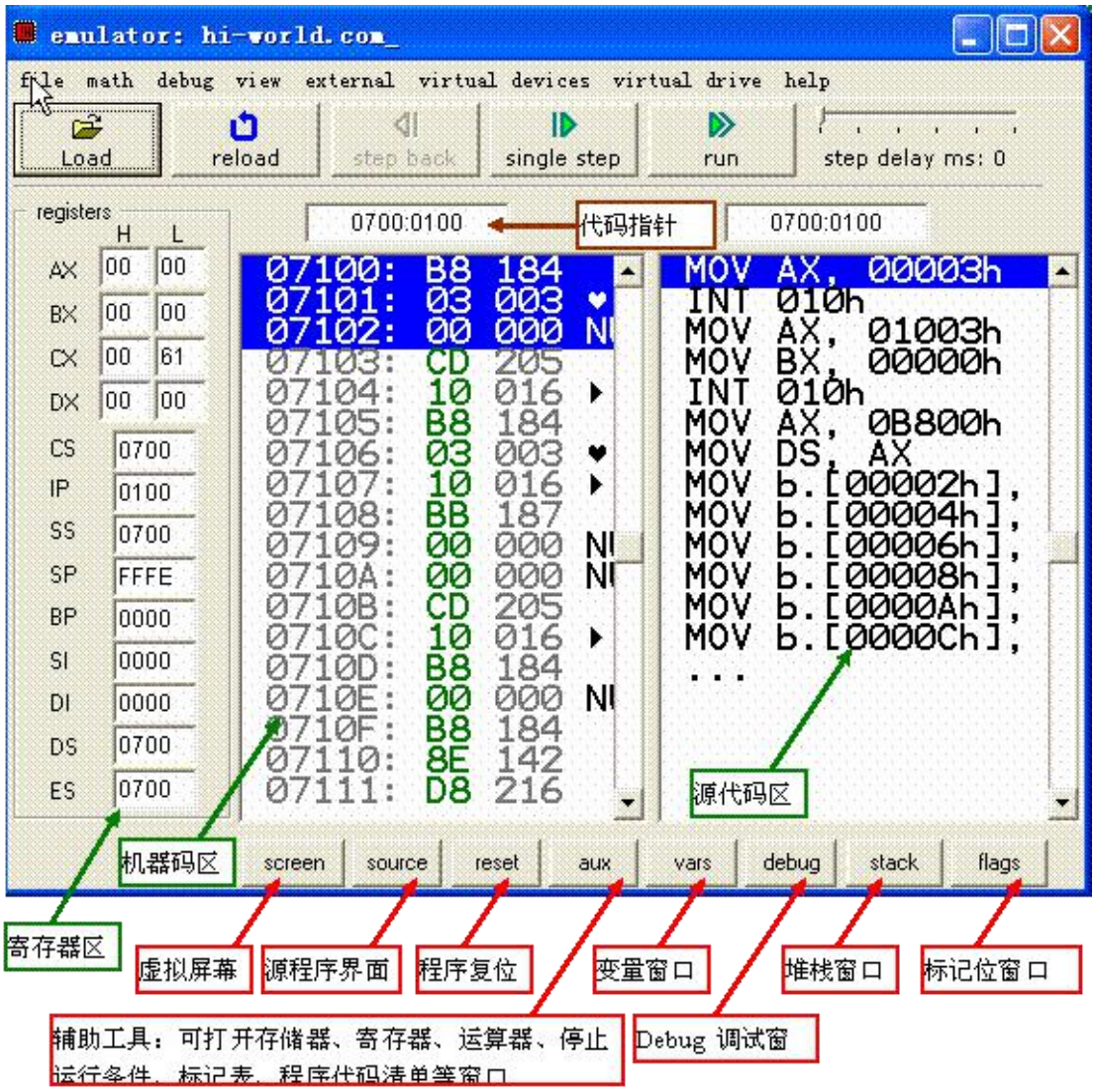


图 5 仿真器界面

图 1-5 HELLO.EXE 的运行结果

EMU 8086 提供了断点设置功能，见图 1-6 所示，先用鼠标选中要设置为断点的指令，然后点击菜单 debug——set break point，即可设置断点。若要清除断点，则点击菜单 debug——clear break point。

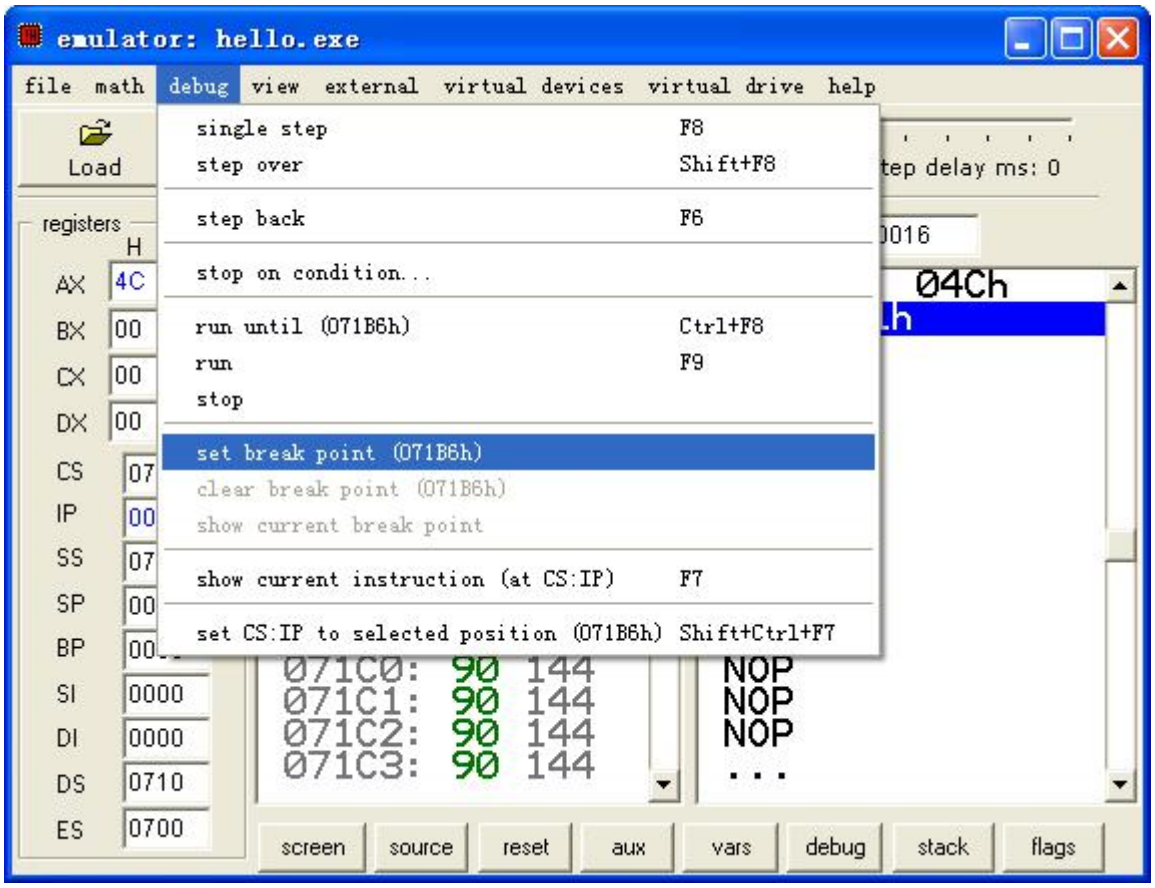


图 1-6 断点设置

EMU 8086 也支持 DEBUG 调试程序，仿真器界面下方的一行按钮中有 debug 按钮，点击打开后可输入 DEBUG 命令进行调试。见图 1-7 所示。

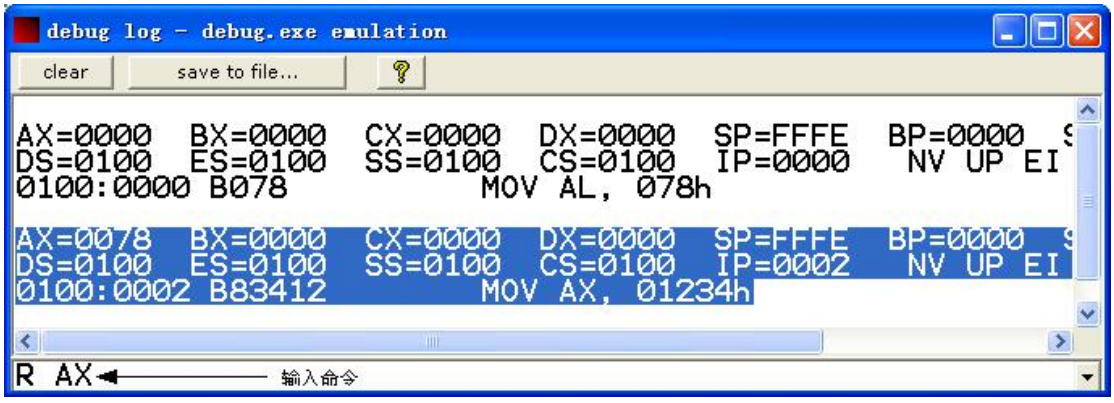


图 1-7 EMU 8086 提供的 DEBUG

五、实验报告要求

实验报告参考模板如下：

实验报告

题目：_____

学 院：信息学院

专 业：

班 级：

学 号：

姓 名：

年 月 日

一、实验目的（宋体四号加粗）

正文（正文 宋体小四，1.5 倍行距）

二、实验环境（宋体四号加粗）

三、实验内容（宋体四号加粗）

四、实验步骤（图文方式叙述）（宋体四号加粗）

五、实验结果及分析（遇到的问题与解决）（宋体四号加粗）

六、实验体会（宋体四号加粗）

实验二 数据传送实验

一、实验目的

1. 熟悉 8086 指令系统的数据传送指令及 8086 的寻址方式。
2. 利用 EMU8086 调试工具调试汇编语言程序。
3. 通过观察深入掌握数据传送类指令的功能。
4. 初步理解汇编语言程序设计方法。

三、实验环境

PC 机、EMU8086

三、实验内容

1、实现 MOV 指令操作

2、实现堆栈操作指令实验

3、实现寻址操作

四、实验步骤

1. MOV 指令实验

通过下述程序段的输入和执行来熟悉 EMU8086 的实用，并用单步调试的方式来观察每条指令执行的结果。

```
MOV BL,08H

MOV CL,BL

MOV AX,03FFH

MOV BX,AX

MOV DS:[0020H],BX
```

2. 堆栈操作指令实验

用一下程序段将一组数据压入堆栈区，并观察以三种方式出栈的结

果，并把程序调试的结果写入表 2-1 中。（**按照完整的汇编语言程序设计步骤进行**）

表 2-1 出栈后数据的变化

	第一种出栈方式	第二种出栈方式	第三种出栈方式
AX=			
BX=			
CX=			
DX=			

程序段如下：

```
MOV AX,0102H
        MOV BX,0304H
        MOV CX,0506H
        MOV DX,0708H
        PUSH AX
        PUSH BX
        PUSH CX
        PUSH DX
;*****************************************************************
```

第一种出栈方式：

```
        POP DX
        POP CX
```

```
        POP BX
        POP AX
;*****************************************************************
```

第二种出栈方式:

```
        POP AX
        POP BX
        POP CX
        POP DX
;*****************************************************************
```

第三种出栈方式:

```
        POP CX
        POP DX
        POP AX
        POP BX
```

3.将 DS:1000H 字节存储单元中的内容发送到 DS:2020H 单元中存放。试着分别用 8086 的直接寻址、寄存器间接寻址、寄存器相对寻址方式实现数据传送。

参考程序如下:

```
;*****************************************************************
; 第一种方式，直接寻址
    MOV [1000H],1234H     ;设置初始值
    MOV AX,[1000H]
```

```
        MOV [2020H],AX
;*****************************************************************
; 第二种方式，寄存器间接寻址
        MOV [1000H],1234H      ;设置初始值
        MOV BX,1000H
        MOV AX,[BX]
        MOV BX,2020H
        MOV [BX],AX
;*****************************************************************
; 第三种方式，寄存器相对寻址
        MOV [1000H],1234H      ;设置初始值
        MOV BX,1000H
        MOV AX,[BX]
        MOV BX,2000H
        MOV SI,20H
        MOV [BX+SI],AX
;*****************************************************************
```

4.设 AX 寄存器中的内容为 1111H，BX 寄存器的内容为 2222H，将 AX 寄存器中的内容与 BX 寄存器内容进行交换，然后再将 BX 寄存器中的内容与 DS:0010H 单元中的内容进行交换。试着编写程序段，并上机验证结果。参考程序如下：

```
;*****************************************************************
```

; 交换指令 XCHG

MOV AX,1111H ;设置初始值

MOV BX,2222H

XCHG AX,BX

XCHG BX,[0010H]

;***

五、实验练习题

1.设 DS=1000H,ES=2000H,对应的内存单元中的内容如图 9-12 所示。要求编写程序段，将图中所示数据段的 1 个字单元的内容传送到 AX 寄存器，附加段 1 个字节的内容传到 BX 寄存器。

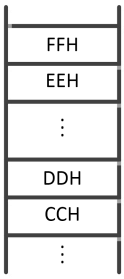


图 1-1 实验练习题 1 图

五、实验报告要求

同实验一

实验三 算术逻辑运算及移位操作实验

一、实验目的

1. 熟悉 8086 指令系统的逻辑运算指令和移位指令的功能。

2. 了解标志寄存器各标志位的意义和指令执行对它的影响。

3. 熟悉在 PC 上简历、汇编、链接、执行和调试 8086 汇编语言的全过程。

4. 初步理解汇编语言程序设计方法。

三、实验环境

PC 机、EMU8086

四、实验内容

1. 理解指令执行及其对标志位的影响

2.无符号字节数求和与求乘积程序设计

五、实验步骤

1、本实验共包含 5 段程序，程序段代码如表 3-1 至 3-5 所示。要求:

（1）自行定义所需要的逻辑段，并分别将表 3-1 中各段程序代码填写在代码段中;

（2）完成程序的汇编、链接，并在 EMU8086 环境中单步执行程序段;

（3）观察个程序段中每条指令的执行结果（寄存器或内容单元中数据的变化）及其对标志位的影响

表 3-1 程序段 1 执行结果及标志位

程序段 1:	执行结果	标志位					
		CF	ZF	SF	OF	PF	AF
MOV AX,1018H	AX=						
MOV SI,030AH	SI=						

	[SI]=						
	AL=						
	DX=						
	AX=						
	[20H]=						
	[SI]=						
	[SP]=						
	BX=						

表 3-2 程序段 2 执行结果及标志位

程序段 2:	执行结果	标志位					
		CF	ZF	SF	OF	PF	AF
MOV AX,0A0AH	AX=						
ADD AX,0FFFFH	AX=						
MOV CX,0FF00H	CX=						
ADC AX,CX	AX=						
SUB AX,AX	AX=						
INC AX	AX=						
OR CX,0FFH	CX=						
AND CX,0F0FH	CX=						
MOV DS:[10H],CX							

	[10H]=						

表 3-3 程序段 3 执行结果及标志位

程序段 3:	执行结果	标志位					
		CF	ZF	SF	OF	PF	AF

MOV BL,25H
MOV DS:BYTE PTR[10H],4
MOV AL,DS:[10H]
MUL BL
	BL=						

	[10H]=						
	AL=						
	AX=						

表 3-4 程序段 4 执行结果及标志位

程序段 4:	执行结果	标志位					
		CF	ZF	SF	OF	PF	AF

MOV DS:WORD PTR[10H],80H
MOV BL,4
MOV AX,DS:[10H]
DIV BL
	[10H]=						

	BL=						
	AX=						
	AX=						

表 3-5 程序段 5 执行结果及标志位

程序段 5:	执行结果	标志位					
		CF	ZF	SF	OF	PF	AF
MOV AX,0	AX=						

	AX=						
	AX=						
	AX=						
	AX =						
	AX=						
	AX=						
	AX=						
	AX=						
	AX =						

2、编写程序实现：用 BX 寄存器作为地址指针，为 BX 赋值 0010H；将 BX 所指向的内存单位开始连续存入 3 个无符号数（10H、04H、30H）；计算内存单元中的这 3 个数之和，并将结果存放到 0013H 单元中去；再求出这 3 个数之积，将乘积存放到 0014H 为首地址的单元中。写出完成此功能的程序段并上机验证结果。参考程序如下所示：

```
;***************************************************************
DSEG SEGMENT
        NUM DB 100 DUP(0)
DSEG    ENDS
CSEG SEGMENT
            ASSUME CS:CSEG, DS:DSEG
```

```
START:MOV AX,DSEG
        MOV DS,AX
;*************************************************************
; 程序段 1:
    MOV BX,0010H                ; 完成赋初值 10H、04H、30H
    MOV BYTE PTR[BX],10H
    INC BX
    MOV BYTE PTR[BX],04H
    INC BX
    MOV BYTE PTR[BX],30H
;*************************************************************
    MOV BX,0010H                ;完成 3 个数累加，并存放结果,[0013H]=44H
    MOV AL,[BX]
    INC BX
    ADD AL,[BX]
    INC BX
    ADD AL,[BX]
    INC BX
    MOV [BX],AL
;*************************************************************
    MOV BX,0010H                ;完成 3 个数累乘，并存放结
```

果,[0014H]=0COOH

```
    MOV AL,[BX]
    INC BX
    MUL [BX]
    INC BX
    MUL [BX]
    MOV [0014H],AX
; ***************************************************************
    CSEG ENDS
        END START
```

3、实验练习题

1.写出完成下述功能的程序段，并说明程序运行的最后结果 AX 值。

（1）传送 15H 到 AL 寄存器;

（2）再将 AL 的内容乘以 2;

（3）接着传送 15H 到 BL 寄存器;

（4）最后把 AL 的内容乘以 BL 的内容。

五、实验报告要求

同实验一

实验四　字符及字符串的输入输出实验

一、实验目的

1. 熟悉如何调用系统功能进行字符及字符串的输入和输出。

2. 掌握在 PC 机上建立、汇编、链接和运行 8088 汇编语言程序的过程。

3. 编写简单的算法程序。

二、实验环境

PC 机、EMU8086

四、实验内容

1. 寻找最大最小数程序实验

2. 实现代码转换实验

五、实验步骤

1、设内存缓冲区从 BUF 单元开始，存放若干个单字节数，其数据长度在 BUF 单元，要求找出最大数送 MAX 单元，最小数送 MIN 单元。

参考源程序:

```
;*************************************************************
;                        FILENAME: 5-1.ASM
;                        寻找最大最小数程序实验
; 设内存缓冲区从 BUF 单元开始，存放若干个单字节数，其数据长度在 BUF
; 单元，要求找出最大数送 MAX 单元，最小数送 MIN 单元。
; *************************************************************
DATA SEGMENT
    BUF DB  9, -4, 55, 78, -81, 0, 41, 124
    B1 EQU $-BUF
    MAX DB  0
    MIN DB  0
DATA ENDS
```

```
; ***************************************************************
CODE SEGMENT
    ASSUME CS: CODE, DS: DATA
START: MOV AX, DATA
       MOV DS, AX
       MOV CH, 0
       MOV CL, B1                      ;CX=数据个数
       DEC CX                          ;循环次数
       MOV AL, BUF                     ;第一个数据送 AL 寄存器
       MOV MAX, AL                     ;假设第一个数是最大数
       MOV MIN,AL                      ;假设第一个数是最小数
       MOV BX, OFFSET  BUF+1           ;下一个数据地址
 LAST:MOV AL, [BX]
      CMP AL, MAX                      ;比较
      JG GREAT
      CMP AL, MIN
      JL LESS
      JMP NEXT
GREAT:MOV MAX, AL                      ;大数->MAX
      JMP NEXT
LESS: MOV MIN, AL                      ;小数->MIN
NEXT: INC BX
      LOOP LAST
      MOV AH, 4CH
      INT 21H
 CODE ENDS
      END START
; ***************************************************************
```

2、从键盘输入 2 个十进制数组合成压缩 BCD 码存入 DL，再将压缩 BCD 码转换成 ASCII 码送 CRT 显示。

```
;***************************************************************
;                        FILENAME: 5-2.ASM
```

```
;                          代码转换程序实验
; 从键盘输入 2 个十进制数组合成压缩 BCD 码存入 DL, 再将压缩 BCD 码转换; 成 ASCII 码送 CRT 显示。
; 例如: 输入 12, 则显示 12; 输入 CD, 则显示 34
; '0'=30H  'A'=41H  'B'=42H 'a'=61H
; ************************************************************
CODE SEGMENT
   ASSUME CS:CODE
ST: MOV AH, 01H  ;键盘输入并显示, AL=输入字符
    INT 21H
    MOV CL, 4
    SHL AL, CL      ;输入的数据左移四位
    MOV DL, AL      ;暂存数据
    MOV AH, 01H      ;键盘输入并显示, AL=输入字符
    INT 21H
    AND AL, 0FH      ;清除刚输入的数据高 4 个位
    OR DL, AL      ;合成压缩 BCD 码
    MOV BL, DL      ;保护 DL 原值
    SHR DL, CL      ;处理压缩 BCD 码高位, 右移动 4 位
    OR DL, 30H      ;数字 0 的 ASCII 码为 30H, 把高半字节变成 ASCII
    MOV AH, 02H      ;显示十位数, DL=输出字符
    INT 21H
    MOV DL, BL      ;处理压缩 BCD 码低位, 变成 ASCII 输出
    AND DL, 0FH
    OR DL, 30H
    MOV AH, 02H      ;显示个位数
    INT 21H
    MOV AH, 4CH
```

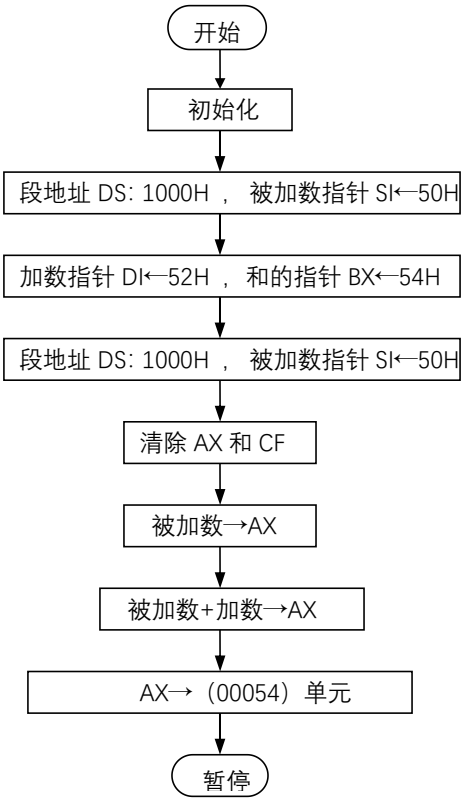
```
    INT 21H
CODE ENDS
    END ST
```

要求先编写汇编文件， EMU8086 采用单步调试的方法进行运行调试，任意输入两个数字，验证实验结果。

3、实验练习题

编写简单程序，用数据运算指令，对两个 16 位数做加法运算。这两个数从地址 10050H 开始连续存放，低位在低地址一端，结果放在这两个数之后。（自己在两个内存单元存入两个数并做加法验证）（清除 CF 用 CLC 指令）（暂停用 HLT 指令）

实验框图:



六、实验报告要求

同实验一

实验五　直线与分支程序设计实验

一、实验目的

1. 学习应用汇编语言进行加减运算的方法。

2. 学习提示信息的显示及键盘输入字符的方法。

3. 学会编写简单的直线和分支算法程序。

二、实验设备和软件

PC 机、EMU8086

二、实验内容

1、认真阅读程序提示及字符串的输入输出方法；

2、理解直线程序是顺序结构程序，根据本实验的编程提示和程序框架预先编写汇编语言源程序，并验证结果。

四、实验步骤

1. 直线程序设计

(1) 在 NUM 变量中定义了 5 个无符号字节数据 U、V、W、X、Y，再定义字节变量 Z。编写程序计算 Z= (U+V-W*X) /Y，并将结果输出显示到屏幕上。程序流程如图 6-1 所示。程序测试数据分别是: U=09H, V=16H, W=02H, X=03H, Y=05H。

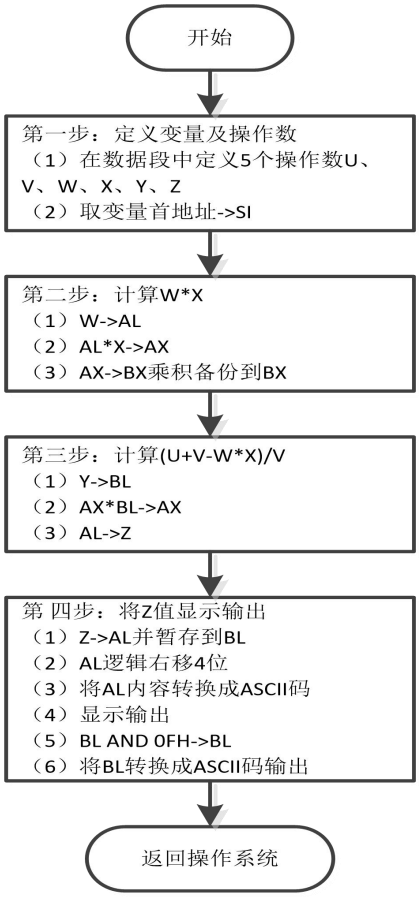


图 6-1 直线程序控制流程图

（2）若将上述 5 个字节数据修改为：U=70，V=23，W=42，X=17，Y=41，重新运行并解释原因。（自己修改程序编程）

```
;*************************************************************
;                          FILENAME: 6-1.ASM
;                          运算 Z= (U+V-W*X) /Y
; 在 NUM 变量中定义了 5 个无符号字节数据 U、V、W、X、Y，再定义字节变
; 量 Z。编写程序计算 Z= (U+V-W*X) /Y，并将结果输出显示到屏幕上。
; 程序流程如图 6-1 所示。程序测试数据分别是：U=09H，V=16H，W=02H，
; X=03H，Y=05H。
; *************************************************************
DSEG SEGMENT
    BUF  DB 09H，16H，02H，03H，05H,0
```

```
  DSEG ENDS
; ***************************************************************
CSEG SEGMENT
      ASSUME CS:CSEG, DS:DSEG
STATR:MOV AX,DSEG
      MOV DS,AX
; ***************************************************************
      MOV AL,BUF+2   ;做运算 Z= (U+V-W*X) /Y
      MUL BUF+3
      MOV BL,AL
; ***************************************************************
      MOV AX,0
      MOV AL,BUF
      ADD AL,BUF+1
      SUB AL,BL
; ***************************************************************
      MOV BL,BUF+4
      DIV BL
      MOV BUF+5,AL    ;结果存放在 AL 中
; ***************************************************************
      AND AL, 0FH     ;清除刚输入的数据高 4 个位
      OR  DL, AL      ;合成压缩 BCD 码
      MOV BL, DL      ;保护 DL 原值
      SHR DL, CL      ;处理压缩 BCD 码高位, 右移动 4 位
      OR DL, 30H      ;数字 0 的 ASCII 码为 30H, 把高半字节变成 ASCII
      MOV AH, 02H      ;显示十位数, DL=输出字符
      INT 21H
      MOV DL, BL      ;处理压缩 BCD 码低位, 变成 ASCII 输出
      AND DL, 0FH
      OR DL, 30H
      MOV AH, 02H      ;显示个位数
      INT 21H
; ***************************************************************
      MOV AH, 4CH
```

```
        INT 21H
CSEG    ENDS
        END STATR
```

2. 分支程序设计

从键盘输入一个十进制正整数 N（10≤N≤99），将其转换成为十六进制数，转换的结果显示在屏幕上。（键盘输入的内容都是 ASCII 的形式）

(1) 程序流程可以参照图 6-1 所示。其中“显示结果”处理框可编写成为子程序，其流程图如图 6-2 所示。

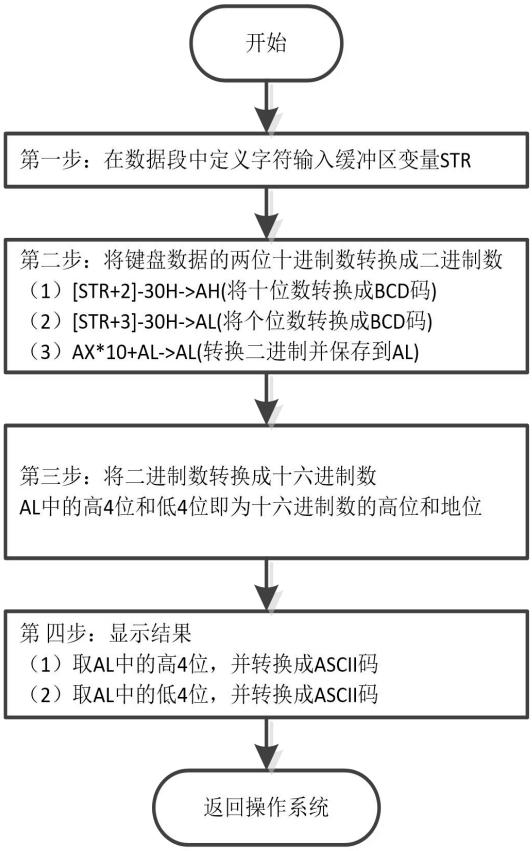


图 6-1 分支程序设计流程

图 6-2 AL 低 4 位转换为 ASCII 码并显示

(2) 字符 0～9 的 ASCII 码是 30H～39H，即在数值 0～9 的基础上加 30H；字符 A～F 的 ASCII 是 41H～46H，即在数值 A～F 的基础上加 37H。

参考程序如下：

```
;*************************************************************
;                          FILENAME: 6-1.ASM
```

; 从键盘输入一个十进制正整数 N (10≤N≤99)，将其转换成为十六进

;制数，转换的结果显示在屏幕上。

; 例如输入 12，显示 0C ； 输入 47，显示 2F

```
; ***************************************************************
DSEG SEGMENT
    STR   DB 3,0,3 DUP(0)
    MES   DB 'Input a decimal number(10-99):',0AH,0DH,'$'
    MES1  DB 0AH,0DH,'Show decimal number as hex;$'
    FORAL DB 0
DSEG ENDS
; ***************************************************************
CSEG SEGMENT
      ASSUME CS:CSEG, DS:DSEG
START:MOV AX,DSEG      ;初始化
      MOV DS,AX
; ***************************************************************
      LEA DX,MES        ;输出提示"Input a decimal number(10～99):"
      MOV AH,9
      INT 21H
; ***************************************************************
      LEA DX,STR        ;输入字符到 STR
      MOV AH, 10
      INT 21H
; ***************************************************************
      MOV AH,[STR+2]    ;将 ASCII 码的两位十进制数转换成为二进制数
      SUB AH,30H
      MOV AL,[STR+3]
      SUB AL,30H
; ***************************************************************
      SHL AH, 1          ;AL=AH*10+AL
      MOV BL,AH
      SHL AH,1
      SHL AH,1
      ADD AH,BL
      ADD AL,AH
      MOV FORAL,AL
```

```
; ***************************************************************
    LEA DX,MES1       ;输出提示"Show decimal number as hex:"
    MOV AH,9
    INT 21H
; ***************************************************************
    MOV AL,FORAL       ;转换并输出 AH
    MOV BL,AL
    MOV CL,4
    SHR BL,CL
    CMP BL,0AH
    JB  L1
    ADD BL,7
L1: ADD BL,30H
    MOV DL,BL
    MOV AH,2
    INT 21H
; ***************************************************************
    MOV AL,FORAL      ;转换并输出 AH
    AND AL,0FH
    CMP AL,0AH
    JB  L2
    ADD AL,7
L2: ADD AL,30H
    MOV DL,AL
    MOV AH,2
    INT 21H
```

3、练习

根据理论课程内容完成：编程计算以下 8 个数据的和，结果存在 ax 寄存器中：0123H，0456H，0789H，0abcH，0defH，0fedH，0cbaH，0987H。（要求及提示：分别使用数据段及代码段，可使用 bx 寄存器循环实现）

五、实验报告要求

同实验一

实验六 循环程序

一、实验目的

1）了解 8086 汇编程序的基本结构

2）掌握循环程序的基本设计方法。

二、实验设备

微型计算机、emu8086

三、实验内容

实现循环程序设计

四、实验步骤

循环程序是把一个程序段重复执行多次的程序结构。循环程序包括三个部分：初始化部分、循环体、循环控制部分。初始化部分用于对循环程序的参数（循环次数、控制条件、指针等）设置初值。循环体是要被重复执行的程序段。循环控制部分用于决定是否退出循环。循环控制指令可以是转移指令或 LOOP 指令。当已知循环次数或控制条件为 ZF 时，用 LOOP 指令控制循环是最简单的方法。

基础：1、 编程实现 1 到 100 之和。

2、 data 段中的 8 个字节如下：

```
data segment
   db 8,11,8,1,8,5,63,38
data ends
```

(1) 统计 data 段中数值为 8 的字节的个数，用 ax 保存统计结果。

(2) 统计 data 段中数值大于 8 的字节的个数，用 ax 保存统计结果。

提高： 1、实现对内存中五个数从小到大排列

基本要求源代码

```
 mov cx,4
 mov si,offset a
 call sort
 mov cx,5
 mov ah,02h
 mov si,offset a
 lp3:
mov dl,[si]
int 21h
 inc si
 INC S
 loop lp3
 HLT
A DW '5','7','1','9','2
 ; si string pointer
 ;cx length-1
sort proc
 lp2:
mov di,si
push cx
mov dl,[di]
 lp1:
```

```
inc di
inc di
cmp dl,[di]
jna j1
xchg dl,[di]
j1:loop lp1
pop cx
mov [si],dl
inc si
inc si
loop lp2
ret
sort endp
```

五、实验报告要求

1) 写出完整的程序。
2) 使用不同数据测试程序并解释运行结果。

实验七　子程序调用及 win32 编程

一、实验目的

1）了解子程序调用

2）掌握 win32 编程。

二、实验环境

微型计算机、masm32v11r

三、实验内容

1、根据理论课程内容完成：编程计算以下 8 个数据的和，结果存在 ax 寄存器中： 0123H，0456H，0789H，0abcH，0defH，0fedH，0cbaH，0987H。（要求及提示：分别使用数据段及代码段，可使用 bx 寄存器循环实现）

2、win32 环境下实现计算 Z=（X-Y+3）

3、设计一个子程序，功能是将一个字节的 BCD 码转换成二进制数。

四、实验步骤

1、将一个字节的 BCD 码转换成二进制数, 所用寄存器: CX, 入口参数: AL 存放两位 BCD 码, 出口参数: AL 存放二进制数。

主程序及子程序清单:

```
CODE        SEGMENT
            ASSUME      CS:CODE
START:      MOV         AL, 12H
            CALL        BCD2BIN
            HALT
BCD2BIN     PROC        NEAR                ;子程序
            PUSH        CX
            MOV         CH, AL
```

```
                AND         CH, 0FH          ;存低4位
                MOV         CL, 4
                SHR         AL, CL           ;高4位右移4位后乘10
                MOV         CL, 10
                MUL         CL
                ADD         AL, CH           ;高 4 位加低 4 位
                POP         CX
                RET
BCD2BIN         ENDP
CODE            ENDS
                END         START
```

2. 设有五个字数据存放在以 BUF 为首地址的内存单元中,要求采用调用多个字数据相加的子程序方法编程,和的低位字放在 RESULT 单元,和的高位字放在 RESULT+2 单元,并将结果显示在屏幕上。

```
STACK SEGMENT STACK

DB 1024 DUP(0)

STACK ENDS

DA TA SEGMENT

BUF DW 0F101H,110DH,52H,100H,456H COUNT =($-BUF)/2

RESULT DW 4 DUP(?),'$'

DA TA ENDS

CODE SEGMENT
```

```
ASSUME CS:CODE,DS:DA TA,SS:STACK WDADD PROC

PUSH DI

MOV AX, 0

MOV DX, 0

MOV DI, OFFSET BUF

NEXT2: ADD AX, [DI]

JNC NEXT1

INC DX

NEXT1: ADD DI, 2

LOOP NEXT2

POP DI

RET

WDADD ENDP

SHOW PROC

PUSH CX

PUSH DI

MOV CX, 04H

MOV DI, OFFSET RESULT
```

```
MOV BX, AX

ADD DI, 07H

BBB: MOV AX, BX

AND AX, 000FH

CMP AL, 0AH

JB QQQ

ADD AL, 07H

QQQ: ADD AL, 30H

MOV [DI], AL

DEC DI

PUSH CX

MOV CL, 04

SHR BX, CL

POP CX

LOOP BBB

MOV CX, 0004H

CCC: MOV AX, DX

AND AX, 000FH
```

```
CMP AL, 0AH
JB DDD
ADD AL, 07H
DDD: ADD AL, 30H
MOV [DI], AL
DEC DI
PUSH CX
MOV CL, 04H
SHR DX, CL
POP CX
LOOP CCC
POP DI
POP CX
RET
SHOW ENDP
BEGIN: MOV AX, DA TA
MOV DS, AX
MOV CX, COUNT
```

```
    CALL WDADD

    CALL SHOW

    MOV DX, OFFSET RESULT

    MOV AH, 09H

    INT 21H

    MOV AH, 4CH

    INT 21H

    CODE ENDS

    END BEGIN
```

3、设计要求：以下两个功能使用子程序实现。

•检测并清除数字字符；

•字符串中小写字母改为大写字母。

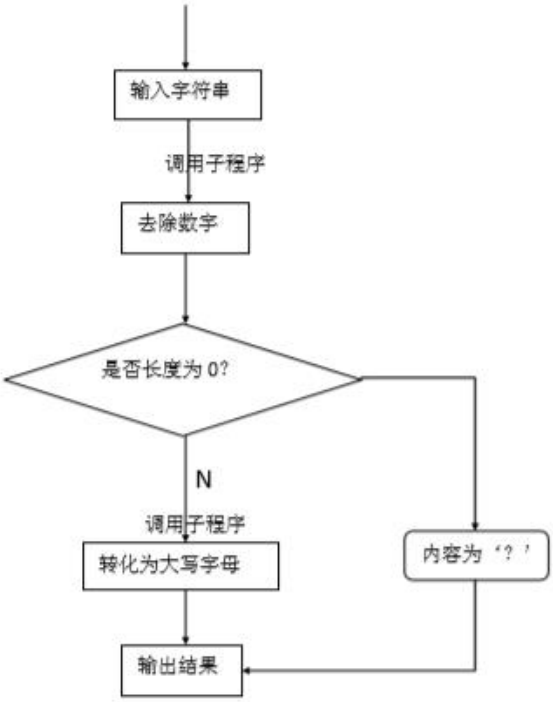
```
DATA SEGMENT
MAXLEN DB 20
INPTLEN DB 0
STR1 DB 10 DUP(0)
STR2 DB 10 DUP(0)
DATA ENDS

STACK SEGMENT STACK
    DW 40H DUP(?)
STACK ENDS

CODE SEGMENT
ASSUME CS:CODE,DS:DATA,SS:STACK
BEGIN:  MOV AX,DATA
        MOV DS,AX
        LEA DX,MAXLEN
        MOV AH,0AH
```

```
        INT 21H
        XOR CX,CX
        LEA SI,STR1
        LEA DI,STR2
        CALL CLRNUM
        CMP INPTLEN,0
        JZ L0
        CALL CHANGECH
        JMP OUTPUT

L0:     MOV [DI],'?'
        MOV [DI+1],'$'
OUTPUT: MOV DL,0DH
        MOV AH,02H
        INT 21H

        MOV DL,0AH
        MOV AH,02H
        INT 21H

        LEA DX,STR2
        MOV AH,09H
        INT 21H
        MOV AH,4CH
        INT 21H

CLRNUM  PROC
        PUSH DI
        PUSH SI
        PUSH AX
        PUSH CX
        MOV CL,INPTLEN

LOP:    CMP [SI],30H
        JAE L1
        JMP L2
L1:     CMP [SI],39H
        JA L2
        DEC INPTLEN
        JMP L3
L2:     MOV AL,[SI]
        MOV [DI],AL
        INC DI
L3:     INC SI
```

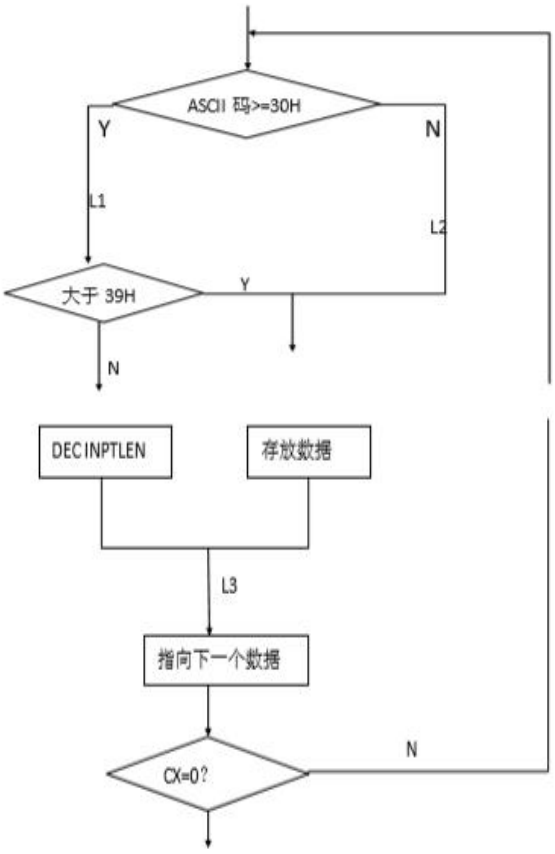
```
        LOOP LOP
        POP CX
        POP AX
        POP SI
        POP DI
        RET
CLRNUM  ENDP

CHANGECH PROC
        PUSH CX
        PUSH DI
        MOV CL,INPTLEN
LOP1:    CMP [DI],61H
        JAE  L4
        JMP L5
L4:     CMP [DI],7AH
        JA L2
        SUB [DI],20H
L5:     INC DI
        LOOP LOP1
        MOV [DI],'$'
        POP DI
        POP CX
        RET
  CHANGECH ENDP
```

（2）流程图

整个程序流程图：



子程序 CLRNUM 流程图



实验八　win32 汇编程序设计

一、实验目的

熟悉 win32 汇编语言程序设计

二、实验环境

微型计算机、masm32v11r

三、实验内容

1、先检查所在机器是否有 win32 汇编环境，如没有请参照实验一安装配置

2、实现 win32 汇编程序设计

四、实验步骤

1、在 notepad++中输入教材 P33 页例 3-1，编译链接，执行。（提醒：结果无法显示有可能是软件不知此中文，可将程序中的中文改成英文，link 时用控制台形式）

例 3-1　计算 $Z=(X-Y+3)$，其中 $X=10, Y=4$。

```
.686
.model flat, stdcall
option casemap :none
;################################################
include \masm32\include\windows.inc
include \masm32\include\kernel32.inc
includelib \masm32\lib\kernel32.lib
;################################################
.data
x         db  10
y         db  4
szPmt     db  '程序结果 = '
szResult  db  ?,0
_size     dd  (sizeof szPmt) + 2
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 代码段
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
.code
start:
        mov     al,x
        sub     al,y
        add     al,3
        add     al,30h
        mov     szResult,al
        invoke  GetStdHandle,STD_OUTPUT_HANDLE
        invoke  WriteConsole,eax,addr szPmt,(sizeof szPmt) + 2,addr _size,0
        invoke  ExitProcess,0
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
        end     start
```

程序在 32 位控制台环境的运行结果如图 3.4 所示。

2、输入 P51 页例 3-3，编译链接执行出结果。

例 3-3 例如数字 0～7 对应的格雷码为

序号	格雷码	十六进制值
0	000	00H
1	001	01H
2	011	03H
3	010	02H
4	110	06H
5	111	07H
6	101	05H
7	100	04H

要求从键盘输入一位 0～7 的十进制数码，把它变成格雷码再输出到显示器上。

分析：因为十进制数码与格雷码之间没有函数关系，所以就必须用查表指令来实现转换，不过需在数据段首先建立格雷码表。

源程序如下：

```
.686
.model flat, stdcall
option casemap :none
;#########################################################
include \masm32\include\windows.inc
include \masm32\include\kernel32.inc
```

3、输入 P77 页例 3-11，编译链接执行出结果。

例 3-11 判断从键盘输入的 ASCII 是字母还是数字，若是字母显示 Char，若是数字显示 Data。假设输入的字符只可能是字母或数字。

分析：通过查看 ASCII 表可以看出，若是字母，则其 ASCII 的 D_6 位为 1，若为数字，则其 ASCII 的 D_6 位为 0。

源程序如下：

```
.686
.model flat, stdcall
option casemap :none
;#########################################################
include \masm32\include\windows.inc
include \masm32\include\kernel32.inc
includelib \masm32\lib\kernel32.lib
;#########################################################
.data
_insize         dword 1
szInput         db  ?
szPmt           db  '请从键盘输入一个数字或字符,然后按回车键!',0
szOutC          db  'Char',0
szOutD          db  'Data',0
_outsize        dword sizeof szOutD
_size           dword 80
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;代码段
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
.code
start:
    invoke      GetStdHandle,STD_OUTPUT_HANDLE
    mov         ebx,eax
```

```
    invoke      WriteConsole,eax,offset szPmt,sizeof szPmt,offset _size,0
    invoke      GetStdHandle,STD_INPUT_HANDLE
    invoke      ReadConsole,eax,offset szInput,sizeof szInput,offset _insize,0
    ;从键盘输入一个数
    mov         dl,szInput
    mov         eax,ebx
    test        dl,40h                  ; 测试 ASCII 的 D6 位
    jz          la1
    invoke      WriteConsole,eax,offset szOutC,sizeof szOutC,offset _size,0
    jmp         la2
la1:
    invoke WriteConsole,eax,offset szOutD,sizeof szOutD,offset _size,0
la2:
    invoke      ExitProcess,0
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
includelib \masm32\lib\kernel32.lib
;##########################################################
.data
bGlm        db  00H,01H,03H,02H,06H,07H,05H,04H
_insize     dword ?
szInput     db  3 dup(0)
szResult    db  '其格雷码为 = ',?,0
_size       dword 13
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
; 代码段
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
.code
start:
        invoke GetStdHandle,STD_INPUT_HANDLE
        invoke ReadConsole,eax,addr szInput,lengthof szInput,addr _insize,0
        mov  al,szInput
        sub  al,30H
        mov  ebx,offset bGlm
        xlat
        add  al,30H
        mov  szResult+11,al
        invoke GetStdHandle,STD_OUTPUT_HANDLE
        invoke WriteConsole,eax,addr szResult,sizeof szResult,addr _size,0
        invoke ExitProcess,0
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
        end  start
```

运行结果如图 3.14 所示。

4、输入 P84 页例 3-16，编译链接执行出结果。

例 3-16 显示 9 个数字字母 1～9、26 个大写字母以及输入的任意数字字符，并按 Enter 键来结束本程序的运行。

分析：利用第一个循环伪指令完成 1～9 的显示；利用第二个循环完成大写字母的显示；利用第三个循环完成任意数字的显示，并监视 Enter 键结束循环。

源程序如下：

```
.686
.model flat, stdcall
option casemap :none                    ; case sensitive
;#########################################################
include \masm32\include\windows.inc
include \masm32\include\kernel32.inc
includelib \masm32\lib\kernel32.lib
;#########################################################
.data
szMsg1          db  '数字:',0
bNum1           db  '1',0
szMsg2          db  13,10,'字母:',0
bNum2           db  'A',0
szMsg3          db  13,10,'请从键盘输入任意数字,然后按 Enter 键结束!',0
_insize         dword 1
szInput         db  ?
_outsize        dword 80
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;代码段
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
.code
start:
    invoke    GetStdHandle,STD_OUTPUT_HANDLE
    mov       ebx,eax
    invoke    WriteConsole,eax,offset szMsg1,sizeof szMsg1,offset _outsize,0
    mov       ecx,9
    .repeat
```

```
  .repeat
  mov       eax,ebx                                   ;没有加这一句,无法正确显示
  push      ecx                                       ;没有加这一句,循环无法正确结束
  invoke    WriteConsole,eax,offset bNum1,sizeof bNum1,offset _outsize,0
  inc       bNum1
  pop       ecx                                       ;显示下一个数字
  .untilcxz
  mov       eax,ebx
  invoke    WriteConsole,eax,offset szMsg2,sizeof szMsg2,offset _outsize,0
  .repeat
  mov       eax,ebx
  invoke    WriteConsole,eax,offset bNum2,sizeof bNum2,offset _outsize,0
  inc       bNum2                                     ;显示下一个字母
  .until    bNum2>'Z'                                 ;显示到'Z'结束
  mov       eax,ebx
  invoke    WriteConsole,eax,offset szMsg3,sizeof szMsg3,offset _outsize,0
  .while 1
  invoke    GetStdHandle,STD_INPUT_HANDLE
  invoke    ReadConsole,eax,addr szInput,lengthof szInput,offset _insize,0
  mov       al,szInput
  .break    .if al == 13                              ;是 Enter 键吗?
  .continue .if (al<'0')||(al>'9')                    ;0~9 之间的任意数字
  mov       eax,ebx
```

```
    invoke      WriteConsole,eax,offset szInput,sizeof szInput,offset _outsize,0
    .endw
    invoke      ExitProcess,0
    ;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
end         start
```

运行结果如图 3.29 所示。

5、输入 P84 页例 3-16，编译链接执行出结果。

例 3-18 编写程序，分类统计出一个字符串中数字字符、字母和其他字符的个数。要求用子程序完成分类统计。

分析：从题目要求可以看出，子程序要完成分类统计，应把字符串作为入口参数，因为字符串可能很长，所以，须用内存单元作为入口参数的传递，这时用 EDX 来指定字符串的首地址。

源程序如下：

```
.686
.model flat, stdcall
option casemap :none                                    ; case sensitive
include \masm32\include\windows.inc
include \masm32\include\kernel32.inc
includelib \masm32\lib\kernel32.lib
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;数据段
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
.data
szStr       db 'JH2J8#0&',0
_outsize    dword 80
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;代码段
```

```
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
.code
start:
        mov      edx,offset szStr
        call     count
        invoke   ExitProcess,0
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
;count 子程序功能：分类统计出一个字符串中数字字符、字母和其他字符的个数
;入口参数：edx 指向被统计字符串
;出口参数：BX、CX 和 DI 分别保存数字字符、字母和其他字符的个数
;算法描述：首先判断字符是否在 0～9 的范围，是则 BX 递增；反之，若小于字
;符 0，说明肯定是其他字符，DI 递增；若大于字符 9，说明可能是字母，也可
;能是其他字符，为了判断简单，先把小写字母变成大写字母，然后判断字符是否
;在 A～Z 的范围，是则 CX 递增；反之，DI 递增
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
count   proc
        push     ax
        xor      bx,bx
        xor      cx,cx
        xor      di,di
again:  mov      al,[edx]
        inc      edx
        cmp      al,0
        je       close
        cmp      al,'0'
        jb       other
        cmp      al,'9'
        ja       LetterOrOther
        inc      bx
```

```
            jmp         again
    LetterOrOther:
            cmp         al,'a'
            jb          Nolower
            cmp         al,'z'
            ja          Nolower
            sub         al,20H
    Nolower:
            cmp         al,'A'
            jb          other
            cmp         al,'Z'
            ja          other
            inc         cx
            jmp         again
other:      inc         di
            JMP         again
close:      pop         ax
            ret
```

```
count       endp
;>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
            end         start
```

五、实验报告要求

同实验一

第二篇 接口技术

接口技术实验采用网上虚拟实验，实验网址为

http://www.vlab.cn/vlab/ckxm.php

实验九 查询输出方式实验

一、实验目的

1. 掌握 I/O 数据传送的基本原理；

2. 2.掌握查询方式的程序设计方法。

二、实验环境

北斗一号微机原理虚拟仿真实验系统

三、实验内容

利用查询方式，从 buffer 开始的单元中依次读取 10 个字节数，并存放到 43AH 端口。设状态端口为 43BH，其 D0 位为 0 时允许传送。

四、实验步骤

1. 电路连接

从“虚拟接口电路”列表中选择打开相应的实验台

若点击无法打开，请确认已正确下载安装最新客户端程序(查看最新版)，并且与网络服务器保持连接。

2. 程序调试

★ 进入 Windows 的 MS-DOS 方式下，使用 MASM 等工具（下载 MASM5.0）调试程序的步骤：

编辑程序

↓

用 MASM 汇编程序

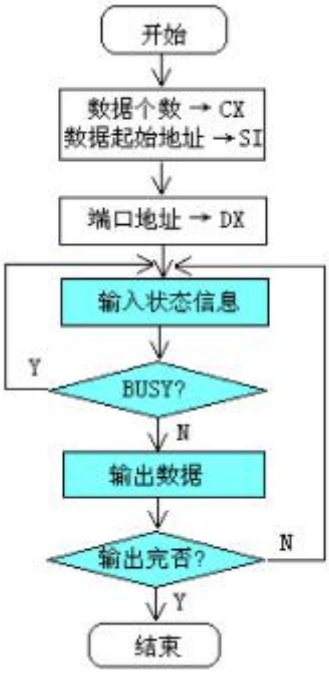
↓

用 LINK 连接程序

↓

运行程序

实验流程图



实验例程

```
Data     segment
Buffer db 1,2,3,4,5,6,7,8,9,10
Data     ends
Code     segment
         Assume cs:code,ds:data
Go:      mov ax,data
         Mov ds,ax
         Lea si,buffer
         Mov cx,10
L1:      Mov dx,43BH
         In   al,dx                ; 读状态端口
         Test al,00000001B
         Jnz L1
         Mov al,[si]
        Mov dx,43AH
         Out dx,al                 ; 数据输出
         Inc   si
         Loop L1
         Mov ah,4ch
         Int 21h
```

五、实验报告要求

同实验一

实验十 8255 工作方式与数据传送实验

一、实验目的

1、掌握 8255 工作方式 0、1、2 及其初始化方法；

2、学会使用 8255 实现 I/O 端口数据的输入输出；

3、能够实现 8255 双向数据传送

二、实验环境

北斗一号微机原理虚拟仿真实验系统

三、实验内容

1、利用 8255 的 PB 获得 8 个拨动开关状态，并利用 PA 口控制 LED 等实时指示；

2、设置 8255 位选通方式，实现通过外部按键选通控制 LED 灯显示拨动开关状态；

3、编程设置 8255 的 PA 口工作于双向传送方式，实现与外部 I/O 设备的数据发送和接收。

四、实验步骤

1、点击 8255 与数据传送中的自动连线；

2、利用 8255 的 PB 获得 8 个拨动开关状态，并利用 PA 口控制 LED 实时指示，点击左侧“汇编调试工具”，打开 8255P0.asm 程序；

3、点击“运行”后拨动开关，观察 LED 灯亮灭情况；

4、设置 8255 位选通方式，实现通过外部按键选通控制 LED 灯显示拨动开关状态，清除 P0 连线，调入 P1 连线，打开 8255P1.asm 程序并运行；

5、观察拨动开关、LED 灯及按键之间的关系；

6、设置 8255 的 PA 口工作于双向传送方式，实现与外部 I/O 设备的数据发送和接收。清除 P1 连线，调入 P2 连线，打开 8255P2.asm 程序并运行；

7、观察发送和接收数据之间的关系。

五、实验报告要求

同实验一

实验十一　8253 方波输出实验

一、实验目的

1. 了解 8253 定时器的硬件连接方法及时序关系；
2. 掌握 8253 的各种模式的编程及其原理。

二、实验环境

北斗一号微机原理虚拟仿真实验系统

三、实验内容

分别设置 8253 计数器/定时器 0、2 的工作方式，用示波器观察对应 OUT 输出波形：

1、定时器 0 工作方式方式 2

2、定时器 0 工作方式方式 3

3、定时器 2 工作方式方式 3

四、实验步骤

1. 电路连接

从“虚拟接口电路”列表中选择打开相应的实验台

若点击无法打开，请确认已正确下载安装最新客户端程序(查看最新版)，并且与网络服务器保持连接。

2. 程序调试

★ 进入 Windows 的 MS-DOS 方式下，使用 MASM 等工具（下载 MASM5.0）调试程序的步骤：

编辑程序

↓

用 MASM 汇编程序

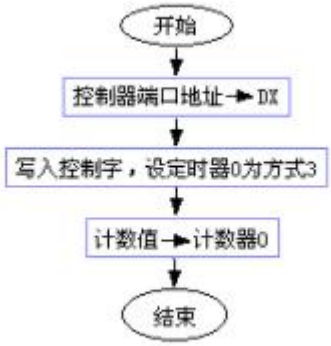
↓

用 LINK 连接程序

↓

运行程序

实验流程图:



调入程序:

```
data segment
tim_ctl equ 453h            ;8253  状态/命令口地址
timer0 equ 450h
mode02 db 34h
data ends
code segment
        assume    cs:code,ds:data
go:    mov ax,data
        mov ds,ax
        mov cx,10
        mov bx,5h
        mov dx,tim_ctl
        mov al,mode02
        out dx,al                ;定时器 0 工作在方式 2
        mov dx,timer0
        mov al,bl
        out dx,al
        mov al,bh
        out dx,al          ;计数初值为 0100h
        mov ah,4ch
        int 21h
code ends
        end    go
```

五、实验报告要求

同实验一

实验十一　8254 模拟电子琴实验

一、实验目的

1．了解音符的实现和扬声器发声原理。

2．掌握用 8254 定时/计数器使扬声器发声的编程方法。

二、实验环境

北斗一号微机原理虚拟仿真实验系统

三、实验内容

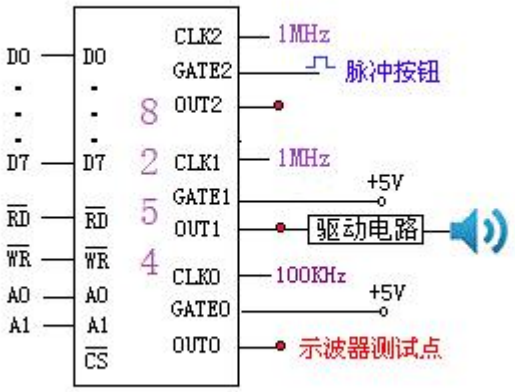
1．根据实验提供的音乐频率表和时间表，编写程序控制 8254，使其输出连接到扬声器上能发出相应的乐曲。

2．利用 8254 电路和小键盘实验台编程实现可模拟电子琴弹奏的功能。

四、实验步骤

1．实验电路

8254 定时/计数器的连接电路如图，其中 OUT1 输出接一个扬声器，其外接时钟



频率为 1MHz。

2．电路连接

从“虚拟接口电路”列表中选择打开相应的实验台

若点击无法打开，请确认已正确下载安装最新客户端程序，并且与网络服务器保持连接。

3．程序调试

★ 进入 Windows 的 MS-DOS 方式下，使用 MASM 等工具调试程序的步骤：

编辑程序

↓

用 MASM 汇编程序

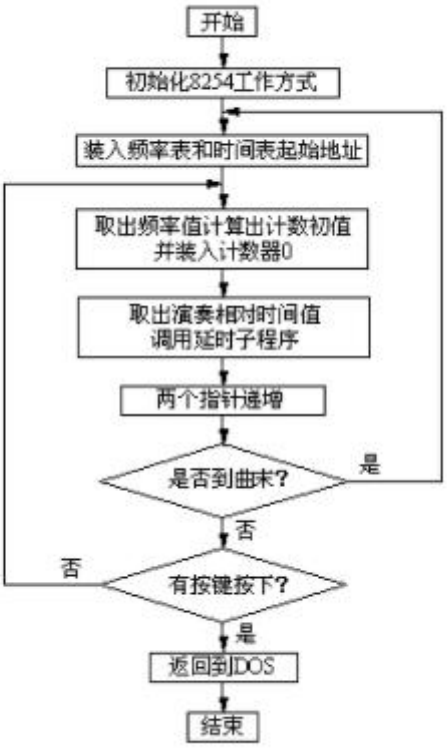
↓

用 LINK 连接程序

↓

运行程序

实验流程图：



实验例程：

```
;  演奏“友谊地久天长”——使用 8254 虚拟电路

DATA      SEGMENT

FREQ_LIST  DW  371,495,495,495,624,556,495,556,624   ;频率表

           DW  495,495,624,742,833,833,833,742,624

           DW  624,495,556,495,556,624,495,416,416,371

           DW  495,833,742,624,624,495,556,495,556,833

           DW  742,624,624,742,833,990,742,624,624,495

           DW  556,495,556,624,495,416,416,371,495,0

TIME_LIST  DB  4,6,2,4,4,6,2,4,4                      ;时间表

           DB  6,2,4,4,12,1,3,6,2

           DB  4,4,6,2,4,4,6,2,4,4

           DB 12,4,6,2,4,4,6,2,4,4

           DB  6,2,4,4,12,4,6,2,4,4

           DB  6,2,4,4,6,2,4,4,12

TIM_CTL    EQU 453H        ;8254 状态/命令口地址

TIMER1     EQU 451H        ;8254 定时计数器 1 地址

MODE00     db 70h

MODE03     db 76h

sound_FREQ dw  523,587,659,698,784,880,988,1046,1175,1318,1397,
1568,1760,1967

s          db "Playing a tune. Please wait a moment...",0dh,0ah,"$"

DATA      ENDS

code      segment
```

```
            assume  cs:code,ds:data
go:         mov ax,data
            mov ds,ax
            mov dx,offset s     ; 显示提示
            mov ah,9
            int 21h
            mov dx,TIM_CTL      ; 定时器1工作在方式3
            mov al,MODE03
            out dx,al
            LEA SI,OFFSET FREQ_LIST
            LEA DI,OFFSET TIME_LIST
m1:         cmp WORD PTR [si],0
            jz exit
            mov ah,0bh
            int 21h   ; 检测有无按键，返回AL=0FFh(有按键)或0(无按键)
            cmp al,0FFh
            jz Exit
            call sound
            call DALLY
            inc si
            inc si
            inc di
            loop m1
exit:
```

```
            mov dx,TIM_CTL       ; 定时器 1 工作在方式 0(关闭)
            mov al,MODE00
            out dx,al
            mov ah,4ch           ; 结束
            int 21h
DALLY    PROC          ;延时子程序。若电脑速度慢可减少 CX 或 AX 的值
            push ax
            push bx
            push cx
            mov bl,[di]
D0:         MOV CX,12H
D1:         MOV AX,02000H
D2:         DEC AX
            JNZ D2
            LOOP D1
            dec bl
            jnz d0
            pop cx
            pop bx
            pop ax
           RET
DALLY    ENDP

sound    PROC          ;发音
```

```
        push ax
        push bx
        push cx
        push dx
        MOV DX,0fH                    ;输入时钟为1MHz，1M = 0F4240H
        MOV AX,4240h
        DIV  WORD PTR [SI] ;取出频率值计算计数初值，0F4240H / 输出频率
        MOV cx,AX
        MOV dx,TIMER1        ; 定时器1计数初值
        MOV al,cl
        out dx,al
        MOV al,ch
        out dx,al
        pop dx
        pop cx
        pop bx
        pop ax
        ret
sound   ENDP
code ends
    END  go
```

五、实验报告要求

同实验一

实验十二　8259 中断控制实验

一、实验目的

1．掌握中断的工作原理，熟悉 8259 中断控制器基本工作原理及其初始化及工作方式的设定，熟悉实验中涉及到的各寄存器的使用方法；

2．2. 学会 8259 中断控制器接口电路的应用，掌握对 8259A 的初始化编程方法和中断服务程序的编写；

3.掌握对 8259A 的初始化编程方法和中断服务程序的设计方法。

二、实验环境

北斗一号微机原理虚拟仿真实验系统

三、实验内容

利用 8259 芯片完成中断控制程序的设计。每按动一次外部按键，产生一次中断信号向主 8259A 发出中断请求，并由中断服务程序在 PC 机显示器上显示一个字符":"。

四、实验步骤

1．电路连接

从“虚拟接口电路”列表中选择打开相应的实验台

若点击无法打开，请确认已正确下载安装最新客户端程序(查看最新版)，并且与网络服务器保持连接。

2．程序调试

★ 进入 Windows 的 MS-DOS 方式下，使用 MASM 等工具（下载 MASM5.0）调试程序的步骤：

编辑程序

↓

用 MASM 汇编程序

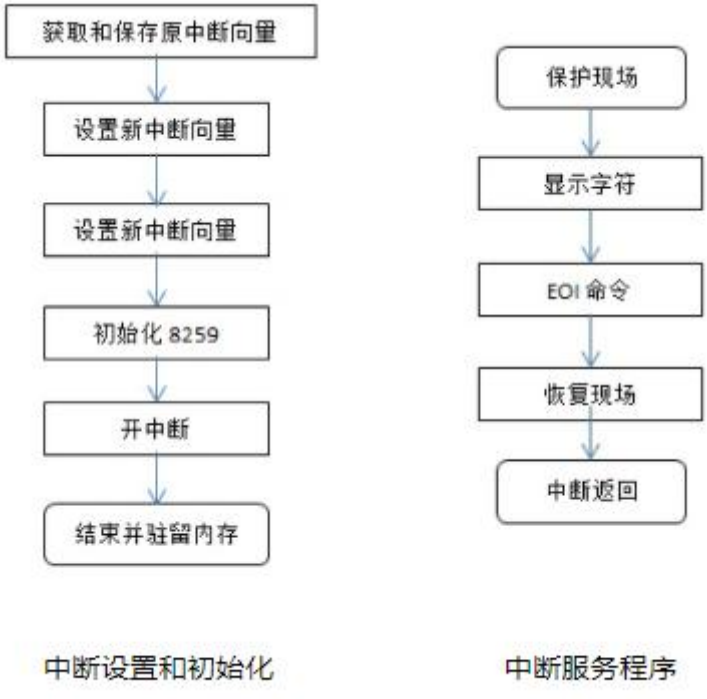
↓

用 LINK 连接程序

↓

运行程序

流程图



例程:

```
;用汇编编写一个 INT 38H  中断驻留程序，要求每响应一次中断显示一单字符“:”
code        segment
assume cs:code,ds:code
start:
            jmp go
s           db "INT 38H is ON !",0dh,0ah,"$"        ;程序功能提示
oldint dw    2 dup(?)
go:
            push        cs
            pop         ds
            mov dx,offset s
            mov ah,9
            int 21h                         ;显示提示
;取中断向量
            mov         ax,3538h
            int         21h
;保存原中断向量
            mov         oldint,bx
            mov         bx,es
            mov         oldint+2,bx
;置新的中断向量
            mov         dx,offset newint
            mov         ax,2538h
```

```
        int     21h
;初始化 8259
        mov al,13h
        mov dx,42ah
        out dx,al
        mov al,38h
        inc dx
        out dx,al
        mov al,09h
        out dx,al
        sti                             ;开中断
        mov     dx,offset endw
        sub     dx,offset start
        mov     cl,4
        shr     dx,cl
        add     dx,11h
        mov     ax,3100h                ;结束并驻留
        int     21h

; 中断服务程序
newint:
        push    ax
        push    bx
        push    cx
        push    dx
        call    dispchar
        mov dx,42ah                     ; EOI 结束命令
        mov al,20h
        out dx, al
        pop     dx
        pop     cx
        pop     bx
        pop     ax
        iret

dispchar    proc near   ; 显示字符
            mov     al,':'
            call    show
            call    curmove
            ret
dispchar    endp


curmove proc near       ; 光标后移
```

```
    push    ax
    push    bx
    push    cx
    push    dx
    mov     ah,3
    mov     bh,0
    int     10h
    inc     dl
    mov     ah,2
    int     10h
    pop     dx
    pop     cx
    pop     bx
    pop     ax
    ret
curmove endp

show   proc near
       push    ax
       push    bx
       push    cx
       push dx
       mov     ah,09h        ; 显示 al 字符
       mov     bx,2fh        ; 显示字符的属性：字符白、底色绿
       mov     cx,1
       int     10h
       pop   dx
       pop     cx
       pop     bx
       pop     ax
       ret
endw:
show   endp

code   ends

end start
```

五、实验报告要求

同实验一

实验十三　交通灯控制系统设计

一、实验目的

1. 掌握交通灯控制系统的设计思路和实现方法；

2. 能够实现 8255a 在控制系统中的应用；

3. 能够实现 8255a 工作方式和编程方法。

二、实验环境

北斗一号微机原理虚拟仿真实验系统

三、实验内容

据交通规则要求，对某一假想十字路口的交通灯实施控制。要求：

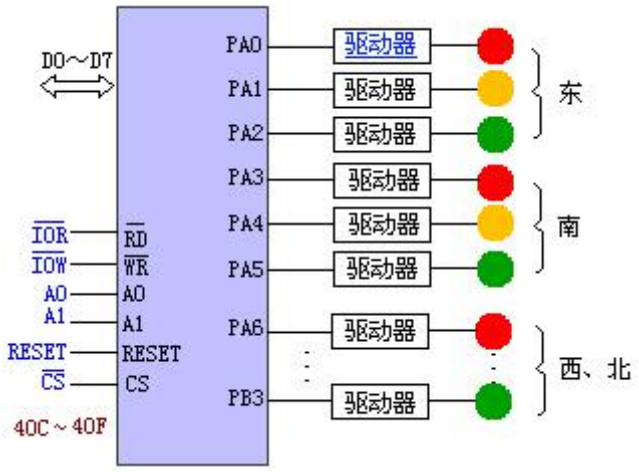
1. 了解目前交通灯的一般要求，并结合实际拟定一个控制方案；

2. 利用 8255a 设计硬件接口电路，以便用 pc 机实施控制；

3. 根据方案要求，确定程序流程和定时的方法，并编制和调试程序，完成设计。

四、实验步骤

1、利用 8255 输出若干控制信号，各信号经过驱动放大和隔离可分别用来控制其所对应交通灯的亮 / 灭， 若十字路口的交通灯分四组，均有"红、黄、绿"三种灯，共 12 个灯，则需要 8255 提供 12 个输出信号，电路结构如下图。



假设十字路口分东西和南北方向，交通灯分四组(东南西北分别编号为 1、2、3、4)，均有“红、黄、绿”三种灯，

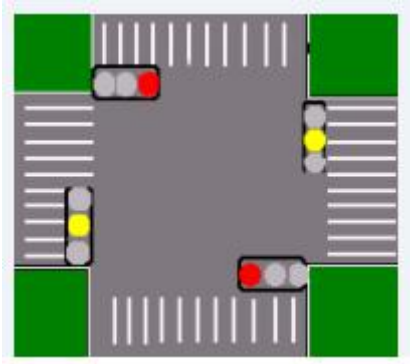
控制方案为：

1）南北向每次通行时间定为 30 秒，此时南北向“绿”灯亮，东西向“红”灯亮；

2）东西向每次通行时间定为 20 秒，此时南北向“红”灯亮，东西向“绿”灯亮；

3）“绿”、“红”灯转换过程为，“绿”灯灭→“黄”灯亮 3 秒→“红”灯亮(另一路“绿”灯亮)；

4）按[ESC]键可使程序结束。



将交通灯控制用的编码数据及其维持时间分别存放于 encode 和 remain 两个数据区中，每个编码数据含两个字节，

为 1 的位表示对应灯亮，各位与灯的对应关系如图：

数据编码：0000100001100001B(0861H)表示东西红灯亮、南北绿灯亮；

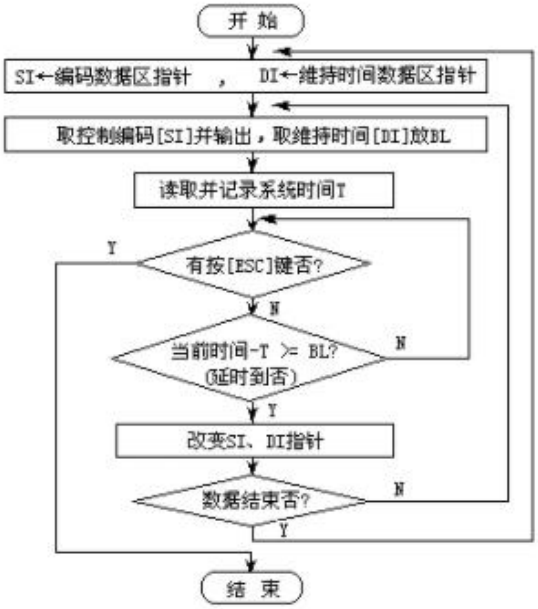
0000010001010001B(0451H)表示东西红灯亮、南北黄灯亮；

0000001100001100B(030cH)表示东西绿灯亮、南北红灯亮；

0000001010001010B(028aH)表示东西黄灯亮、南北红灯

亮。

流程图



按照实验原理及流程图实现程序设计。

五、实验报告要求

同实验一

实验十四 温度监控系统设计

一、实验目的

能够实现有关传感器对温度等现场信息监控系统的设计方法。

二、实验环境

北斗一号微机原理虚拟仿真实验系统

三、实验内容

对某环境温度进行实时监测和控制，要求系统具有如下基本功能：

1. 正常情况（温度<18℃）下，绿灯亮、红灯灭、控温设备（如空调）不工作；

2. 当温度≥18℃时，红灯亮、绿灯灭、控温设备仍不工作；

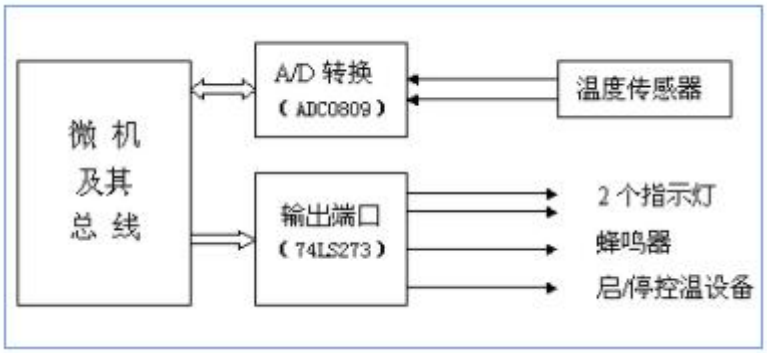
3. 当温度≥20℃时，给出警告声音并发出启动控温设备的控制信号，此后当温度回落到<20℃时，

灯和控温设备仍保持现有状态不变；

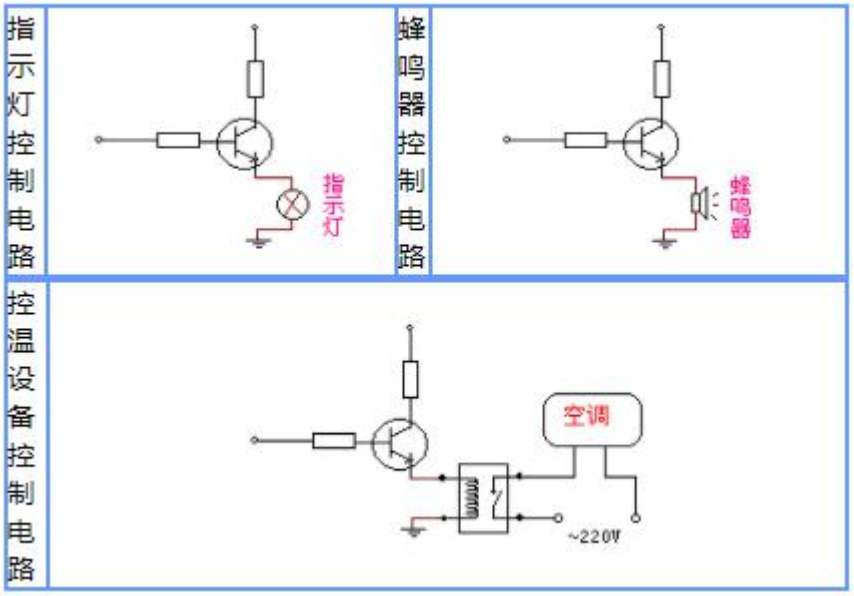
4. 当温度继续回落到<18℃时，红灯灭、绿灯亮，控温设备停止工作。

四、实验原理

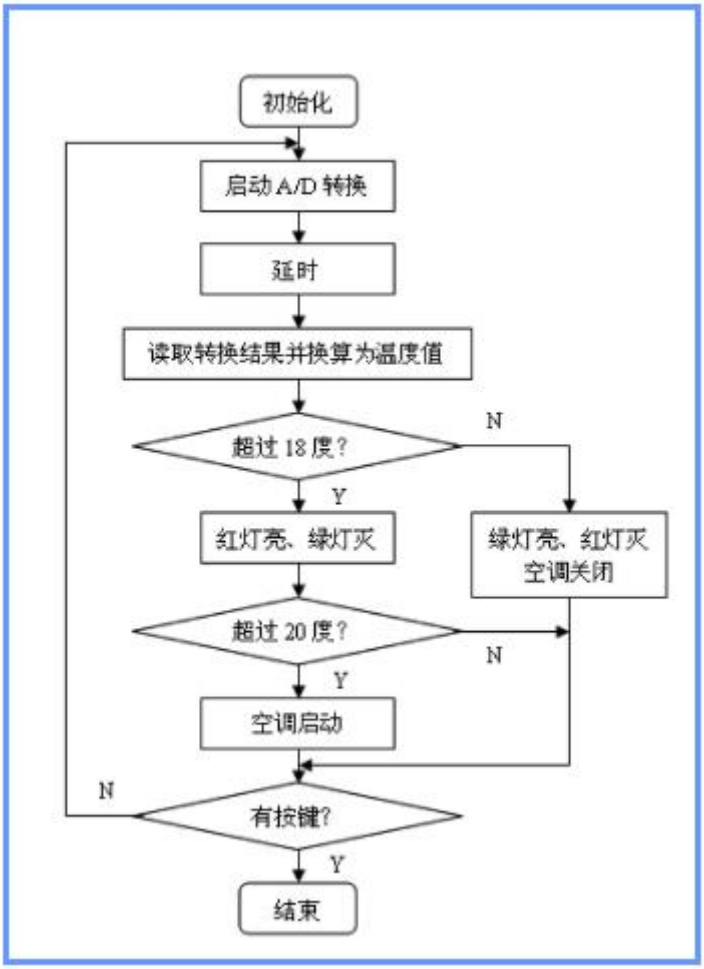
1、电路连接



2、外部控制电路



流程图



3、按照实验原理编制控制程序

五、实验报告要求

同实验一