

廈門大學



《汇编语言》实验报告

(五)

姓 名 宋浩元

学 号 37220232203808

学 院 信息学院

专 业 软件工程

2024 年 11 月

1 实验目的

- (1) 进一步学习 8086 的硬指令，完成基本的程序设计，并融合一定的算法思想；
- (2) 重点掌握分支和循环结构的程序设计方法和技巧。

2 实验环境

Windows11 环境下的 masm 与 DOSBOX；

3 实验内容

- (1) 试编制一个程序段完成图 1 中的流程图所规定的功能。

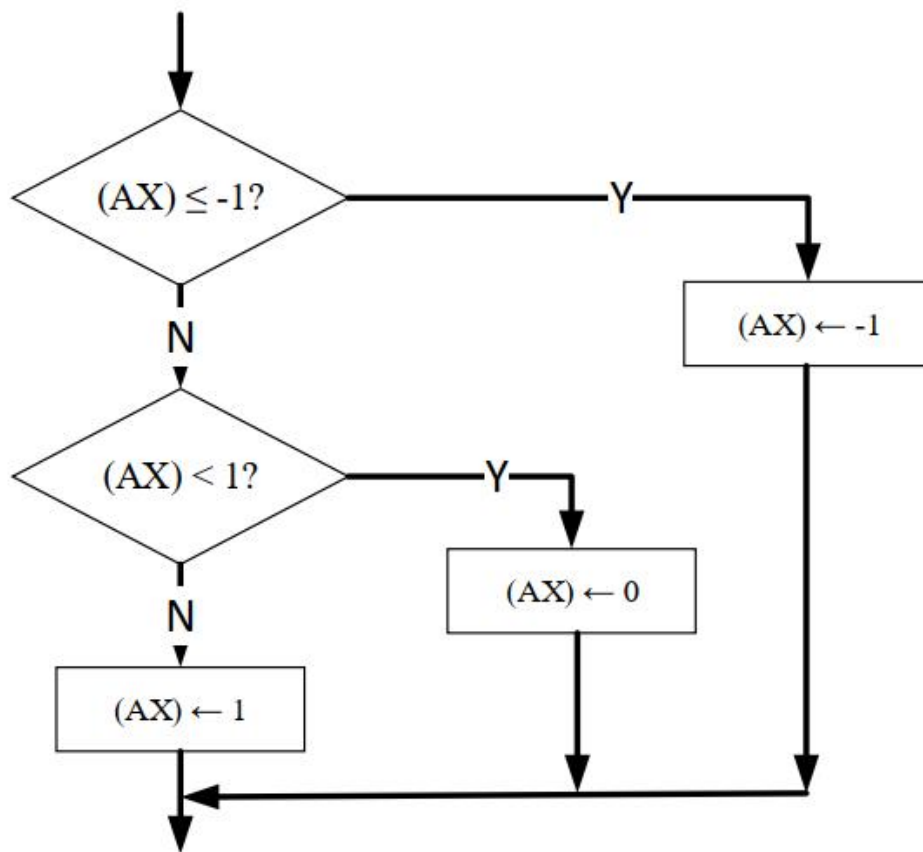


图 1

(2) 要求测试在 STATUE 中的一个字节，如果第 1、3、5 位均为 1 则转移到 ROUTINE_1；如果此三位中有两位为 1 则转移到 ROUTINE_2；如果此三位只有一位为 1 则转移到 ROUTINE_3；如果此三位全为 0 则转移到 ROUTINE_4。试画出流程图，并编制相应的程序段。

(3) 编写程序要求将一个字节数据以十六进制数形式显示， 要求如下：

- 使用换码指令 XLAT
- 不使用换码指令

(4) 已知用于 LED 数码管显示的代码表为：

LEDTABL DB 0C0H, 0F9H, 0A4H, 0B0H, 99H, 92H, 82H, 0F8H

E DB 80H, 90H, 88H, 83H, 0C6H, 0C1H, 86H, 8EH

依次表示 0~9 和 A~F 这 16 个数码的显示代码。现编写一个程序，实现将 lednum 中的(0~9 和 A~F) 转换成对应的 LED 显示代码。

(5) bufX、bufY 和 bufZ 是 3 个有符号十六进制数，编写一个比较相等关系的程序。

- 如果这 3 个数都不相等，则显示 0。
- 如果这 3 个数中有两个数相等，则显示 1。
- 如果这 3 个数都相等，则显示 2。

(6) 已定义了两个整数变量 A 和 B，试编写程序完成下列功能：

- 若两个数中有一个是奇数，则将奇数存入 A 中，偶数存入 B 中；
- 若两个数均为奇数，则将两数均加 1 后存回原变量；
- 若两个数均为偶数，则两个变量均不改变。

4 实验具体实现

实验一：

在 START 标签后，首先初始化 DS 寄存器。然后将 AX 寄存器初始化为 0。接着使用 CMP 和 JLE、JL 指令实现流程图中的判断和赋值功能。最后，通过 MOV AH, 4CH 和 INT 21H 返回操作系统。

```

DATAS SEGMENT
    ; 数据段可以在这里定义变量等，本程序不需要
DATAS ENDS

STACKS SEGMENT
    ; 堆栈段可以在这里定义堆栈空间，本程序不需要
STACKS ENDS

CODES SEGMENT
    ASSUME CS:CODES, DS:DATAS, SS:STACKS
START:
    MOV AX, DATAS
    MOV DS, AX

    ; 此处输入代码段代码
    MOV AX, 0
    CMP AX, -1      ; 比较AX是否小于等于 -1
    JLE SET_MINUS_1 ; 如果小于等于 -1, 跳转到SET_MINUS_1
    CMP AX, 1       ; 比较AX是否小于1
    JL SET_ZERO     ; 如果小于1, 跳转到SET_ZERO
    ; 如果都不满足, 设置AX为1
    MOV AX, 1
    JMP END_PROGRAM ; 跳转到程序结束
SET_MINUS_1:
    MOV AX, -1      ; 设置AX为 -1
    JMP END_PROGRAM ; 跳转到程序结束
SET_ZERO:
    MOV AX, 0       ; 设置AX为0
END_PROGRAM:
    MOV AH, 4CH
    INT 21H
CODES ENDS
    END START

```

实验二：

首先初始化数据段寄存器 DS。然后取 STATUE 中的字节到 AL 寄存器，并通过 AND 操作保留第 1、3、5 位。接着通过一系列 CMP 和 JE 指令判断这三位中 1 的个数，根据不同情况跳转到不同的子程序（ROUTINE_1、ROUTINE_2、ROUTINE_3、ROUTINE_4）。

```

DATAS SEGMENT
    STATUE DB 0
DATAS ENDS

STACKS SEGMENT
    DW 100 DUP(?)
STACKS ENDS

CODES SEGMENT
    ASSUME CS:CODES, DS:DATAS, SS:STACKS
START:
    MOV AX, DATAS
    MOV DS, AX

    ; 取STATUE中的字节进行测试
    MOV AL, STATUE
    AND AL, 00101010B ; 只保留第1、3、5位
    CMP AL, 00101010B ; 检查这三位是否全为1
    JE ROUTINE_1
    CMP AL, 00010100B ; 检查这三位是否有两位为1
    JE ROUTINE_2
    CMP AL, 00000010B ; 检查这三位是否只有一位为1
    JE ROUTINE_3
    JMP ROUTINE_4

ROUTINE_1:
    ; ROUTINE_1的代码
    JMP END_PROGRAM

ROUTINE_2:
    ; ROUTINE_2的代码
    JMP END_PROGRAM

ROUTINE_3:
    ; ROUTINE_3的代码
    JMP END_PROGRAM

ROUTINE_4:
    ; ROUTINE_4的代码

END_PROGRAM:
    MOV AH, 4CH
    INT 21H

```

实验三：

用 xlat 换行版本：

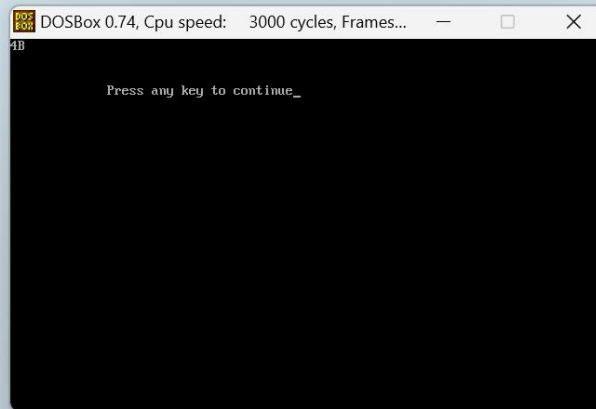
```

;用xlat
.model small
.stack
.data
hex db 4bh
ascii db 30h,31h,32h,33h,34h,35h,36h,37h,38h,39h
      db 41h,42h,43h,44h,45h,46h

.code
.startup
    mov bx,offset ascii ;取得表的偏移地址，为换码做准备
    ;显示高位
    mov al,hex
    mov cl,4
    sar al,cl
    xlat ;al<--ds:[bx+al]
    mov dl,al
    mov ah,02h
    int 21h
    ;显示低位
    mov al,hex
    and al,0fh ;高位为0
    xlat
    mov dl,al
    mov ah,02h
    int 21h

    .exit 0
end

```



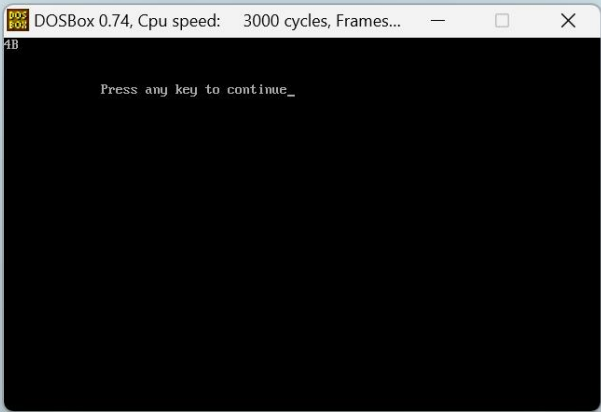
不用换行版本：

```
;不换行的版本
.model small
.stack
.data
hex db 4bh

.code
.startup
;显示高位
mov al,hex
mov cl,4
shr al,cl
mov dl,al
call disp
;显示低位
mov dl,hex
and dl,0fh ;将高位置0
call disp
.exit 0

;显示字符的子程序
disp PROC
    cmp dl,9
    jbe next ;如果小于等于9,就直接加30h,如果是字母,就需要另外加7
zimu:
    add dl,7h
next:
    add dl,30h
;显示
    mov ah,02h
    int 21h
    ret
disp ENDP

end
```



实验四：将 lednum 中的 (0~9 和 A~F) 转换成对应的 LED 显示代码。


```

stack      segment stack
            db 100h dup(?)
stack
            ends
data       segment
ledtable   db 0c0h,0f9h,0a4h,0b0h,99h,92h,82h,0f8h
            db 80h,90h,88h,83h,0c6h,0c1h,86h,8eh
lednum     db 3
data       ends
code       segment 'code'
assume     cs:code,ss:stack,ds:data
start:     mov ax,data
            mov ds,ax
            lea bx,ledtable
            mov al,lednum
            xlat
            mov ax,4c00h
            int 21h
code       ends
            end start
|

```

实验五：

```

DATAS SEGMENT
    D DB 3 DUP(?); 将数据首先保存到D数组
    D1 DW ?
    D2 DW ?
    D3 DW ?
    D4 DB 30h
    MESSAGE DB 'please input three number(hex, 1Byte):',13,10,'$'
;此处输入数据段代码
DATAS ENDS

STACKS SEGMENT
;此处输入堆栈段代码
STACKS ENDS

CODES SEGMENT
    ASSUME CS:CODES,DS:DATAS,SS:STACKS
START:
    MOV AX,DATAS
    MOV DS,AX
    LEA DX,MESSAGE
    MOV AH,09H ;dos 9号功能调用输出字符串,以$结尾
    INT 21H ; 输出提示
    MOV CX, 3H ;控制输入3个数
    MOV SI,OFFSET D;利用寄存器间接寻址
INPUT: MOV BX,0; 将输入的数据暂时保留BX,初始化位0
    MOV AH,01H ; dos 1号功能调用输入1个字符到AL
    INT 21H ; 读入字节高位
    CMP AL, 'a' ; 输入的是a..f
    JB C1
    SUB AL,20H; 变成小写字母
C1: CMP AL, 'A' ;
    JB C2
    SUB AL,7H; 与数字'9'隔7
C2: SUB AL,30H
    MOV BL,AL
    SHL BL,1
    SHL BL,1
    SHL BL,1
    SHL BL,1; 循环左移4位,移到高位
    INT 21H
    CMP AL, 'a' ; 输入的是a..f
    JB C3
    SUB AL,20H; 变成小写字母

```

```

DOSBox 0.74: Cpu speed: 3000 cycles, Frames...
please input three number(hex,1Byte):
12
34
56
Press any key to continue_

```

```

C1: CMP AL, 'A' ;
    JB C2
    SUB AL, 7H ; 与数字 '9' 隔7
C2: SUB AL, 30H
    MOV BL, AL
    SHL BL, 1
    SHL BL, 1
    SHL BL, 1
    SHL BL, 1 ; 循环左移4位, 移到高位
    INT 21H
    CMP AL, 'a' ; 输入的是a..f
    JB C3
    SUB AL, 20H ; 变成小写字母
C3: CMP AL, 'A' ;
    JB C4
    SUB AL, 7H ; 与数字 '9' 隔7
C4: SUB AL, 30H
    ADD BL, AL
    MOV [SI], BL ; 将数据送往内存D的位置
    INC SI ; si+1
    MOV AH, 02H ; dos 2号功能输出1个字符
    MOV DL, 13 ; 输出字符
    INT 21H ; 输出\n使得表示将光标移至行首,dos系统下需要这样做
    MOV DL, 10
    INT 21H ; 输入\n 换行
    LOOP INPUT ; 循环
    MOV SI, OFFSET D
    MOV AX, 0H ; 因为AX16,D8位, 用A1,所以ax初始化为0
    MOV AL, BYTE PTR [SI]
    MOV D1, AX
    MOV AL, BYTE PTR [SI+1]
    MOV D2, AX
    MOV AL, BYTE PTR [SI+2]
    MOV D3, AX ; 将数据分别送往D1,D2,D3
    MOV AX, D1
    CMP AX, D2
    JE L1 ; D1, D2相等跳转L1位置
    CMP AX, D3
    JE L2
    MOV AX, D2
    CMP AX, D3
    JE L2
    JMP L

```

DOSBox 0.74; Cpu speed: 3000 cycles, Fram
 please input three number(hexa41Byte):
 11
 11
 11
 2
 Press any key to continue

实验六:

两个偶数: 不变

```

1 DATA SEGMENT
2     A DW 4
3     B DW 4
4 DATA ENDS
5 STACK SEGMENT
6     DW 10 DUP(?)
7 TOP LABEL WORD
8 STACK ENDS
9 CODE SEGMENT
10    ASSUME CS:CODE,SS:STACK
11 MAIN PROC FAR
12    MOV AX,STACK
13    MOV SS,AX
14    LEA SP,TOP
15    MOV AX,DATA
16    MOV DS,AX
17    MOV AX,A
18    MOV BX,B
19    TEST AX,1
20    JZ L1
21    TEST BX,1

```

DOSBox 0.74, Cpu speed: 3000 cycles, Frames...

```

-t
AX=4C04 BX=0004 CX=006D DX=0000 SP=0014 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=0771 CS=0773 IP=003B NU UP EI PL ZR NA PE NC
0773:003B CD21 INT 21
-t
AX=4C04 BX=0004 CX=006D DX=0000 SP=000E BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=0771 CS=F000 IP=14A0 NU UP DI PL ZR NA PE NC
F000:14A0 FB STI
-t
AX=4C04 BX=0004 CX=006D DX=0000 SP=000E BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=0771 CS=F000 IP=14A1 NU UP EI PL ZR NA PE NC
F000:14A1 FE3B ??? [BX*SI] DS:0004=
-d 0000
0770:0000 04 00 04 00 00 00 00 00-00 00 00 00 00 00 00 .....
0770:0010 00 00 00 00 00 04 4C 00 00-A1 14 00 F0 A6 01 3D 00 .....L...=
0770:0020 73 07 46 72 00 00 00 00-00 00 00 00 00 00 00 s..r.....
0770:0030 B8 71 07 8E D0 8D 26 14-00 B8 70 07 8E D8 A1 00 .q...&...p...
0770:0040 00 8B 1E 02 00 A9 01 00-74 11 F7 C3 01 00 74 19 .....t.....
0770:0050 40 43 A3 00 00 89 1E 02-00 EB 0E F7 C3 01 00 74 @C.....t
0770:0060 08 93 A3 00 00 89 1E 02-00 B4 4C CD 21 00 00 00 .....L.!...
0770:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....

```

两个奇数：分别加一存回原变量

```

1 DATA SEGMENT
2     A DW 3
3     B DW 3
4 DATA ENDS
5 STACK SEGMENT
6     DW 10 DUP(?)
7 TOP LABEL WORD
8 STACK ENDS
9 CODE SEGMENT
10    ASSUME CS:CODE,SS:STACK
11 MAIN PROC FAR
12    MOV AX,STACK
13    MOV SS,AX
14    LEA SP,TOP
15    MOV AX,DATA
16    MOV DS,AX
17    MOV AX,A
18    MOV BX,B
19    TEST AX,1
20    JZ L1
21    TEST BX,1

```

DOSBox 0.74, Cpu speed: 3000 cycles, Frames...

```

-t
AX=0004 BX=0004 CX=006D DX=0000 SP=0014 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=0771 CS=0773 IP=0039 NU UP EI PL NZ NA PO NC
0773:0039 B44C MOV AH,4C
-t
AX=4C04 BX=0004 CX=006D DX=0000 SP=0014 BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=0771 CS=0773 IP=003B NU UP EI PL NZ NA PO NC
0773:003B CD21 INT 21
-t
AX=4C04 BX=0004 CX=006D DX=0000 SP=000E BP=0000 SI=0000 DI=0000
DS=0770 ES=0760 SS=0771 CS=F000 IP=14A0 NU UP DI PL NZ NA PO NC
F000:14A0 FB STI
-d 0000
0770:0000 04 00 04 00 00 00 00 00-00 00 00 00 00 00 00 .....
0770:0010 00 00 00 00 00 00 00 00-00 00 04 4C 00 00 3D 00 .....L...=
0770:0020 73 07 02 72 00 00 00 00-00 00 00 00 00 00 00 s..r.....
0770:0030 B8 71 07 8E D0 8D 26 14-00 B8 70 07 8E D8 A1 00 .q...&...p...
0770:0040 00 8B 1E 02 00 A9 01 00-74 11 F7 C3 01 00 74 19 .....t.....
0770:0050 40 43 A3 00 00 89 1E 02-00 EB 0E F7 C3 01 00 74 @C.....t
0770:0060 08 93 A3 00 00 89 1E 02-00 B4 4C CD 21 00 00 00 .....L.!...
0770:0070 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 .....

```

一个奇数一个偶数：

