experiment 3

实验三

结构程序设计

汇编语言程序和采用其他高级语言编写的程序的结构一样，主要有顺序结构、分支结构、循环结构、子程序结构和宏结构 5 种（上机实验环境和实验步骤见主教材第 11 章）。

任务 1　顺序和分支结构程序设计

顺序结构程序是最简单的一种程序结构，即写在前面的指令先执行，后面的指令后执行，适用于简单的应用场合。分支结构程序可根据条件进行选择。

➢ 任务目标

1. 牢记转移指令和比较指令的格式及用法。

2. 正确描述汇编语言宏观架构，建立初步的汇编思维。

3. 正确描述分支结构的执行过程，进一步加深对分支结构的理解。

4. 正确使用顺序和分支结构解决实际问题，养成良好的编程习惯，具备程序员的职业素养。

5. 对程序进行查错与排错，具备一定的程序调试能力和技巧。

6. 熟练保存和备份程序，树立保密意识，养成知保密、懂保密、善保密的良好习惯。

➢ 任务实施

通过对主教材的学习，编写具有顺序结构和分支结构的程序以解决实际问题。


实验三 1.1

【任务 1.1】 有 3 个有符号数，分别为 10H、67H 和 02H，编写源程序计算 VAL＝67H/02H＋10H。

分析：在数据段中定义 3 个字节变量，依次存储为 10H、67H 和 02H。除数 02H 为字节类型，被除数 67H 须扩展为字类型，商为字节类型，最后结果为字节类型。

源程序 compute.asm 如下。

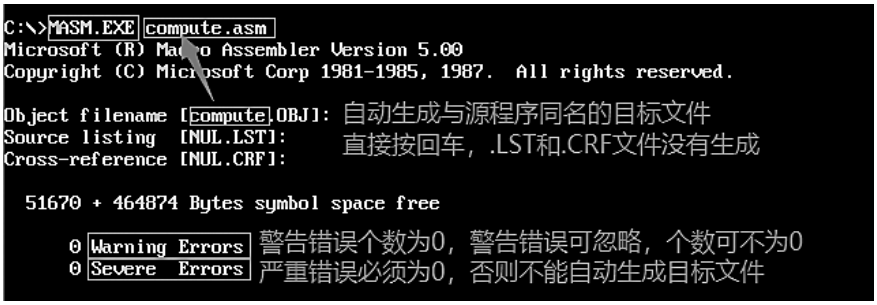
```
DATA SEGMENT
    A DB 10H,67H,02H
    VAL DB ?
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE,DS: DATA
START:
    MOV AX,DATA
    MOV DS,AX
    MOV AL,A+1          ;AL=67H
    CBW                 ;AX=0067H
    MOV BL,A+2H         ;BL=02H
    IDIV BL             ;AX 除以 BL,商送 AL,余数送 AH
    MOV CH,A            ;CH=10H
    ADD AL,CH           ;AL=CH+AL
    MOV VAL,AL          ;将 AL 中的内容传送至 VAL 指向的字节单元
    MOV AH,4CH
    INT 21H
CODE ENDS
    END START
```

上机执行过程如下。

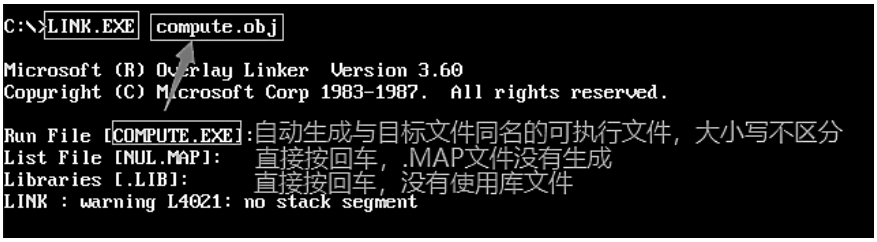
第一步：在编辑工具（记事本、DEVC 或 EDIT 等）中输入源程序

compute.asm。

第二步：用汇编程序 MASM.EXE 汇编 compute.asm 源程序，严重错误个数为 0 表示汇编成功，否则需要修改源程序；继续执行第二步，直到严重错误个数为 0。

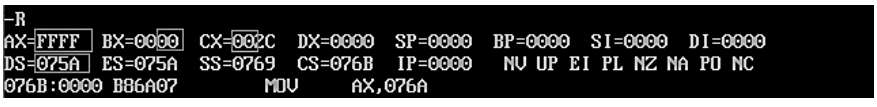


第三步：用链接程序 LINK.EXE 链接文件 compute.obj。



第四步：用调试程序 DEBUG.EXE 调试文件 compute.exe，使用反汇编命令 U 可以查看数据段段基址和一些无用指令。

第五步：指令执行前，用 R 命令观察源程序中使用的寄存器 AX、DS、BL 和 CH 中的内容为 AX＝FFFFH、DS＝075AH、BL＝00H 和 CH＝00H（这样指令执行前后寄存器中的内容可以对比）。



指令执行前，用D命令观察数据段0000H、0002H和0003H指向的字节单元中的内容分别为10H、67H和02H（这样指令执行前后数据段中的内容可以对比）。

```
-
-D 076A:0000 000F
076A:0000  10 67 02 00 00 00 00 00-00 00 00 00 00 00 00 00   .g..............
-
```

段基址一定是为源程序分配的段基址076AH

第六步：用T命令执行第1、2条指令后，AX=076AH，DS=076AH。

```
-T=0000

AX=076A  BX=0000  CX=002C  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076B  IP=0003   NV UP EI PL NZ NA PO NC
076B:0003 8ED8          MOV     DS,AX
-T

AX=076A  BX=0000  CX=002C  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0005   NV UP EI PL NZ NA PO NC
076B:0005 A00100        MOV     AL,[0001]                       DS:0001=67
```

用T命令执行第3条指令后，AL=67H。

```
-T

AX=0767  BX=0000  CX=002A  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0008   NV UP EI PL NZ NA PO NC
076B:0008 98            CBW
-
```

用T命令执行第4条指令后，被除数扩展为字类型，AX=0067H。

```
-T

AX=0067  BX=0000  CX=002C  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0009   NV UP EI PL NZ NA PO NC
076B:0009 8A1E0200      MOV     BL,[0002]                       DS:0002=02
```

用T命令执行第5条指令后，除数传送至BL=02H。

```
-T

AX=0067  BX=0002  CX=002A  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000B   NV UP EI PL NZ NA PO NC
076B:000B F6FB          IDIV    BL
-
```

用T命令执行第6条指令后，AX/BL=0067H/02H，商为33H。

```
-T

AX=0133  BX=0002  CX=002A  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000D   NV UP EI PL NZ NA PO NC
076B:000D 8A2E0000      MOV     CH,[0000]                       DS:0000=10
-
```

用T命令执行第7条指令后，将0000H指向的字节单元中的内容传送至CH，CH=10H。

```
-t
命令大小写不区分

AX=0133  BX=0002  CX=102A  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0011   NV UP EI PL NZ NA PO NC
076B:0011 02C5          ADD     AL,CH
-
```

用 T 命令执行第 8 条指令后，将 AL 和 CH 求和，结果传送至 AL。

```
-T

AX=0143  BX=0002  CX=102A  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0013   NV UP EI PL NZ NA PO NC
076B:0013 A20300        MOV     [0003],AL                       DS:0003=00
-
```

用 T 命令执行第 9 条指令后，将 AL 中的内容传送至 0003H 指向的字节单元。

```
-T

AX=0143  BX=0002  CX=102A  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0016   NV UP EI PL NZ NA PO NC
076B:0016 B44C          MOV     AH,4C
```

```
-D DS:0000 000F
076A:0000  10 67 02 43 00 00 00 00-00 00 00 00 00 00 00 00   .g.C............
-
-
```

用 T 命令执行第 10 条指令后，AH＝4CH；用 P 命令执行 INT 21H 结束程序，返回操作系统。

```
-T

AX=4C43  BX=0002  CX=102A  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0018   NV UP EI PL NZ NA PO NC
076B:0018 CD21          INT     21
-P

C:\>_
```

【任务 1.2】 设 TABLE 表存放 1～9 的平方，NUM 存放 1～9 中的任意一个数，编写源程序，将 NUM 中数的平方传送至 BL。


实验三 1.2

分析：该题须在 TABLE 表中查找 NUM 中数的平方，可将 TABLE 表的首地址传送至 BX，NUM 中的数传送至 AL，然后用 XLAT 指令即可实现该任务。

源程序 xlat.asm 如下。

```
DATA SEGMENT
    TABLE DB 0,1,4,9,16,25,36,49,64,81
    NUM DB 5
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE,DS: DATA
```

```
START:
    MOV AX,DATA
    MOV DS,AX
    LEA BX,TABLE   ;TABLE 首地址传送至 BX,BX=0000H
    MOV AL,NUM     ;NUM 指向的字节单元中的内容传送至 AL,AL=5
    XLAT TABLE     ;将 BX+AL 指向的字节单元中的内容传送至 AL,AL=25(19H)
    MOV BL,AL      ;BL=19H
    MOV AH,4CH
    INT 21H
CODE ENDS
    END START
```

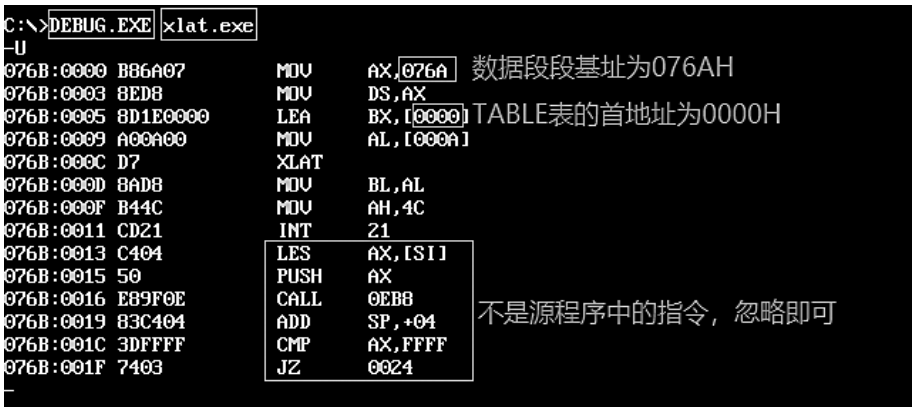
上机执行过程如下。

第一步：在编辑工具（记事本、DEVC 或 EDIT 等）中输入源程序 xlat.asm。

第二步：用汇编程序 MASM.EXE 汇编源程序 xlat.asm，严重错误个数为 0 表示汇编成功，否则需要修改源程序，然后继续执行第二步，直到严重错误个数为 0。

第三步：用链接程序 LINK.EXE 链接文件 xlat.obj。

第四步：用调试程序 DEBUG.EXE 调试文件 xlat.exe，使用反汇编命令 U 可以查看数据段段基址和一些无用指令。



第五步：指令执行前，用 R 命令观察源程序中使用的寄存器 AX、DS 和 BX 中的内容为 AX＝FFFFH、DS＝075AH、BX＝0000H（这样指令执行前后寄存器中的内容可以对比）。

```
-R
AX=FFFF  BX=0000  CX=0023  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076B  IP=0000   NV UP EI PL NZ NA PO NC
076B:0000 BB6A07        MOV     AX,076A
-
```

指令执行前，用 D 命令观察数据段 TABLE 表 0000H～0009H 指向的字节单元中的内容分别为 00H～51H，NUM 字单元中的内容为 05H（这样指令执行前后数据段中的内容可以对比）。

```
-
-   不要使得DS，因为DS中的值为075AH，而给源程序中数据段分配的段基址为076AH
-D 076A:0000   000F          十六进制
076A:0000  00 01 04 09 10 19 24 31-40 51 05 00 00 00 00 00   ......$1@Q......
-   十进制数 0  1   4   9 16 2536 49  64 81 NUM
-
-
```

第六步：用 T 命令执行第 1、2 条指令后，AX＝076AH，DS＝076AH。

```
-T

AX=076A  BX=0000  CX=0023  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076B  IP=0003   NV UP EI PL NZ NA PO NC
076B:0003 8ED8          MOV     DS,AX
-T

AX=076A  BX=0000  CX=0023  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0005   NV UP EI PL NZ NA PO NC
076B:0005 8D1E0000      LEA     BX,[0000]                       DS:0000=0100
-
```

用 T 命令执行第 3、4 条指令后，BX＝0000H，AL＝05H。

```
-T

AX=076A  BX=0000  CX=0023  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0009   NV UP EI PL NZ NA PO NC
076B:0009 A00A00        MOV     AL,[000A]                       DS:000A=05
-
-
-T

AX=0705  BX=0000  CX=0023  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000C   NV UP EI PL NZ NA PO NC
076B:000C D7            XLAT
-
```

用 T 命令执行第 5、6 条指令后，将 BX＋AL＝0000H＋05H＝0005H 指向的字节单元中的内容 19H 传送至 AL，AL＝19H，BL＝19H。

```
-T

AX=0719  BX=0000  CX=0023  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000D   NV UP EI PL NZ NA PO NC
076B:000D 8AD8          MOV     BL,AL
-
-T

AX=0719  BX=0019  CX=0023  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000F   NV UP EI PL NZ NA PO NC
076B:000F B44C          MOV     AH,4C
```

用 T 命令执行第 7、8 条指令后，程序正常结束。

```
-T

AX=4C19  BX=0019  CX=0023  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0011   NV UP EI PL NZ NA PO NC
076B:0011 CD21          INT     21
-P

Program terminated normally
-
```

实验三 1.3

【任务 1.3】 设 X 和 Y 字节单元分别存放有符号数 89H 和 02H，编写源程序，求两个数中的最小数并传送至 MIN 字单元。

分析：89H 和 02H 为有符号数，89H 转换为二进制是 1000 1001B，最高位为 1，则为负数；02H 转换为二进制数是 0000 0010B，最高位为 0，则为正数；所以最小数为 89H。

源程序 min.asm 如下。

```
DATA SEGMENT
    X DB 89H
    Y DB 02H
    MIN DB ?
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE,DS: DATA
START:
    MOV AX,DATA
    MOV DS,AX
    MOV AL,X        ;X 指向的字节单元中的内容 89H 传送至 AL
    MOV BH,Y        ;Y 指向的字节单元中的内容 02H 传送至 BH
    CMP AL,BH       ;AL-BH,
    JG L1           ;AL 大于 BH,跳转到 L1,BH 为小数
        MOV MIN,AL
        JMP NEXT
    L1:
        MOV MIN,BH
    NEXT:
    MOV AH,4CH
    INT 21H
CODE ENDS
```

```
END START
```

上机执行过程如下。

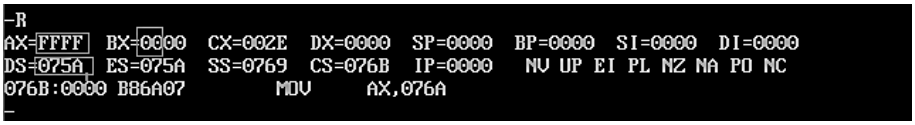
第一步：在编辑工具（记事本、DEVC 或 EDIT 等）中输入源程序 min.asm。

第二步：用汇编程序 MASM.EXE 汇编源程序 min.asm，严重错误个数为 0 表示汇编成功，否则需要修改源程序，然后继续执行第二步，直到严重错误个数为 0。

第三步：用链接程序 LINK.EXE 链接文件 min.obj。

第四步：用调试程序 DEBUG.EXE 调试文件 min.exe，使用反汇编命令 U 可以查看数据段段基址和一些无用指令。

第五步：指令执行前，用 R 命令观察源程序中使用的寄存器 AX、DS 和 BH 中的内容为 AX＝FFFFH、DS＝075AH、BH＝00H（这样指令执行前后寄存器中的内容可以对比）。



指令执行前，用 D 命令观察数据段 X、Y 和 MIN 字节单元中的内容分别为 89H、02H 和 00H（这样指令执行前后数据段中的内容可以对比）。

第六步：用 T 命令执行第 1、2 条指令后，AX＝076AH，DS＝076AH。

```
-T=0000

AX=076A  BX=0000  CX=002E  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076B  IP=0003   NV UP EI PL NZ NA PO NC
076B:0003 8ED8          MOV     DS,AX
-T

AX=076A  BX=0000  CX=002E  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0005   NV UP EI PL NZ NA PO NC
076B:0005 A00000        MOV     AL,[0000]                       DS:0000=89
-
```

用 T 命令执行第 3、4 条指令后，AL＝89H，BH＝02H。

```
-T

AX=0789  BX=0000  CX=002E  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0008   NV UP EI PL NZ NA PO NC
076B:0008 8A3E0100      MOV     BH,[0001]                       DS:0001=02
-T

AX=0789  BX=0200  CX=002E  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000C   NV UP EI PL NZ NA PO NC
076B:000C 3AC7          CMP     AL,BH
-
```

用 T 命令执行第 5～7 条指令后，MIN 字节单元存放最小数 89H。

```
-T  CMP AL,BH    AL小于BH

AX=0789  BX=0200  CX=002E  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=000E   NV UP EI NG NZ NA PE NC
076B:000E 7F06          JG      0016
-T   JG L1     AL不大于BH，所以没有跳转到L1，顺序往下执行

AX=0789  BX=0200  CX=002E  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0010   NV UP EI NG NZ NA PE NC
076B:0010 A20200        MOV     [0002],AL                       DS:0002=00
-
-
-
-
-T   MOV MIN,AL  将最小值AL送MIN单元

AX=0789  BX=0200  CX=002E  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0013   NV UP EI NG NZ NA PE NC
076B:0013 EB05          JMP     001A
-
-
-D DS:0000 000F  MIN
076A:0000  89 02 89 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

用 T 命令执行第 8、11 和 12 条指令后，程序正常结束。

```
-T   JMP NEXT 跳转到NEXT位置，执行返回DOS中断调用

AX=0789  BX=0200  CX=002E  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=001A   NV UP EI NG NZ NA PE NC
076B:001A B44C          MOV     AH,4C
-T

AX=4C89  BX=0200  CX=002E  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=001C   NV UP EI NG NZ NA PE NC
076B:001C CD21          INT     21
-P

Program terminated normally
-
```

【任务 1.4】 从键盘输入 1～3 中任意一个数，编写源程序，在屏幕上分别显示“1OK”“2OK”或“3OK”；如果输入其他字符，则显示“*”。


实验三 1.4

分析：该题有 3 种情况，相当于有 3 个分支，如图 3.1 所示。

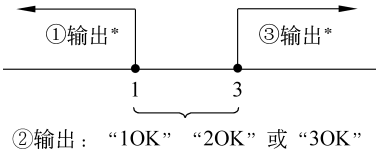


图 3.1 3 种情况

源程序 output.asm 如下。

```
DATA SEGMENT
    STR DB "OK$"
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE,DS: DATA
START:
    MOV AX,DATA
    MOV DS,AX
    MOV AH,01H                ;将功能号 01H 传送至 AH
    INT 21H                   ;实现从键盘输入数据传送至 AL
    CMP AL,'1'                ;AL-'1'
    JL  L1                    ;AL 低于'1',跳转到 L1,图 3.1①输出 *
        CMP AL,'3'            ;AL-'3'
        JA  L1                ;AL 高于'3',跳转到 L1,图 3.1③输出 *
            MOV DL,AL
            MOV AH,02H
            INT 21H
            LEA DX,STR        ;图 3.1② 输出“1OK”“2OK”或“3OK”
            MOV AH,09H
            INT 21H
            JMP NEXT
    L1:
        MOV DL,'*'
        MOV AH,02H
        INT 21H
```

```
    NEXT:
    MOV AH,4CH
    INT 21H
CODE ENDS
    END START
```

上机执行过程如下。

第一步：在编辑工具（记事本、DEVC 或 EDIT 等）中输入源程序 output.asm。

第二步：用汇编程序 MASM.EXE 汇编源程序 output.asm，严重错误个数为 0 表示汇编成功，否则需要修改源程序，然后继续执行第二步，直到严重错误个数为 0。

第三步：用链接程序 LINK.EXE 链接文件 output.obj。

第四步：程序中有输出指令，可直接执行可执行文件，输出结果。

➢ **知识拓展**

1. BUF 字节单元中有一个有符号数，判断其正负，若为正数，则显示“＋”；若为负数，则显示“－”；若为 0，则显示 0。

2. 设 W、X、Y、Z 和 V 是字节变量，计算(X×Y＋360－Z)/V，并将结果传送至 W。

任务 2 循环结构程序设计

➢ 任务目标

1. 牢记循环指令、串操作和比较指令的格式及用法。

2. 正确描述循环结构的执行过程，进一步加深对循环结构的理解。

3. 正确使用循环结构解决实际问题，树立“不积跬步，无以至千里；不积小流，无以成江海”的思想。

➢ 任务实施

通过前面的学习，编写具有循环结构的程序以解决实际问题。

【任务 2.1】 X 字节单元中的内容为 83H，编写源程序，将其二进制数显示到屏幕上。


实验三 2.1

分析：83H 转换为二进制数是 1000 0011B，先显示高位，再显示低位。可采用 SHL 指令一位一位地移出，然后调用 02H 号功能显示输出；但是 02H 号功能只能显示输出字符型数据，所以须得到 0 和 1 的 ASCII 码 30H 和 31H。如图 3.2 所示，循环移动 8 次即可将二进制数显示到屏幕上。

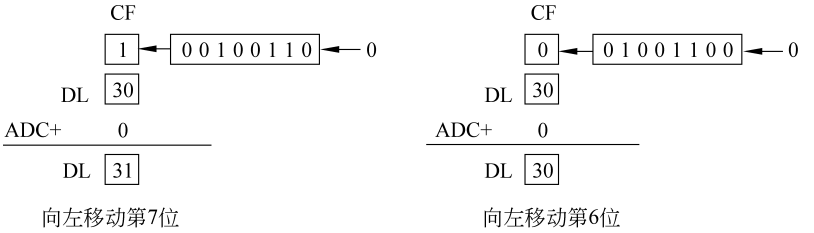


图 3.2 获得 0 和 1 的 ASCII 码的执行过程

源程序 boutput.asm 如下。

```
DATA SEGMENT
    X DB 83H
```

```
DATA ENDS
CODE SEGMENT
    ASSUME CS: CODE,DS: DATA
    START:
      MOV AX,DATA
      MOV DS,AX
      MOV BL,X                ;将 X 字节单元中的内容 83H 传送至 BL
      MOV CX,8                ;循环次数传送至 CX
      L1:
          MOV DL,30H          ;30H 传送至 DL
          SHL BL,1            ;将 BL 中的内容逻辑左移 1 位,结果传送至 BL
          ADC DL,0            ;将 DL+0+CF 的结果传送至 DL
          MOV AH,02H
          INT 21H
      LOOP L1
      MOV AH,4CH
      INT 21H
CODE ENDS
    END START
```

上机执行过程如下。

第一步：在编辑工具（记事本、DEVC 或 EDIT 等）中输入源程序 boutput.asm。

第二步：用汇编程序 MASM.EXE 汇编源程序 boutput.asm，严重错误个数为 0 表示汇编成功，否则需要修改源程序，然后继续执行第二步，直到严重错误个数为 0。

第三步：用链接程序 LINK.EXE 链接文件 boutput.obj。

第四步：程序中有输出指令，可直接执行可执行文件，输出结果。

```
C:\>boutput.exe
10000011
```

实验三 2.2

【任务 2.2】 BUF 字节单元中存放了 5 个无符号数，求 5 个数中所有偶数的累加和。

分析：通过逻辑右移指令将 BUF 字节单元中的数据向右移动 1 位，然后判断 CF；如果 CF＝1，则该数为奇数，否则为偶数。

源程序 sum.asm 如下。

```
DATA SEGMENT
    BUF DB 83H,18H,90H,11H,53H
    LEN EQU $-BUF          ;获得 BUF 缓冲区数据个数
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE,DS: DATA
START:
    MOV AX,DATA
    MOV DS,AX
    MOV AL,00H             ;将 00H 传送至 AL,存放累加和
    LEA BX,BUF             ;将 BUF 缓冲区首地址传送至 BX
    MOV CX,LEN             ;将数据个数传送至 CX
    L1:
        MOV DL,[BX]        ;将 BX 指向的字节单元中的内容传送至 DL
        SHR DL,1           ;DL 中的内容右移 1 位,结果传送至 DL
        JC  L2      ;CF=1,说明 BX 指向的字节单元中的内容是奇数,跳转到 L2
          ADD AL,[BX]      ;说明 BX 指向的字节单元中的内容是偶数,累加到 AL
        L2:
          INC BX           ;BX 自增 1,BX 指向下一个字节单元
    LOOP L1
    MOV AH,4CH
    INT 21H
CODE ENDS
    END START
```

上机执行过程如下。

第一步：在编辑工具（记事本、DEVC 或 EDIT 等）中输入源程序 sum.asm。

第二步：用汇编程序 MASM.EXE 汇编源程序 sum.asm，严重错误个数为 0 表示汇编成功，否则需要修改源程序，然后继续执行第二步，直到严重错误个数为 0。

第三步：用链接程序 LINK.EXE 链接文件 sum.obj。

第四步：程序中没有输出指令，用调试程序 DEBUG.EXE 调试文件 sum.

exe，使用反汇编命令 U 可以查看数据段段基址和一些无用指令。

```
C:\>DEBUG.EXE sum.exe
-U
076B:0000 B86A07        MOV     AX,076A
076B:0003 8ED8          MOV     DS,AX
076B:0005 B000          MOV     AL,00
076B:0007 8D1E0000      LEA     BX,[0000]
076B:000B B90500        MOV     CX,0005
076B:000E 8A17          MOV     DL,[BX]   该指令所在位置为000EH位置
076B:0010 D0EA          SHR     DL,1
076B:0012 7202          JB      0016
076B:0014 0207          ADD     AL,[BX]
076B:0016 43            INC     BX
076B:0017 E2F5          LOOP    000E      CX不等于0，循环指令需跳转到000EH位
076B:0019 B44C          MOV     AH,4C
076B:001B CD21          INT     21
076B:001D FFFF          ???     DI
076B:001F 7403          JZ      0024      不是源程序中指令，忽略即可
-
```

第五步：指令执行前，用 R 命令观察源程序中使用的寄存器内容（这样指令执行前后寄存器中的内容可以对比）。

```
-R
AX=FFFF  BX=0000  CX=002D  DX=0000  SP=0000  BP=0000  SI=0000  DI=0000
DS=075A  ES=075A  SS=0769  CS=076B  IP=0000   NV UP EI PL NZ NA PO NC
076B:0000 B86A07        MOV     AX,076A
-
```

指令执行前，用 D 命令观察数据段 BUF 字节单元中的内容。

```
-D 076A:0000 000F
076A:0000  83 18 90 11 53 00 00 00-00 00 00 00 00 00 00 00   ....S...........
-
```

第六步：用 T 命令执行第 1 次循环，BUF 字节单元中的第 1 个数据 83H 不是偶数，BX 自增 1，指向第 2 个单元。

```
076B:000E 8A17          MOV     DL,[BX]                         DS:0000=83
-T

AX=0700  BX=0000  CX=0005  DX=0083  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0010   NV UP EI PL NZ NA PO NC
076B:0010 D0EA          SHR     DL,1
-T

AX=0700  BX=0000  CX=0005  DX=0041  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0012   OV UP EI PL NZ AC PE CY
076B:0012 7202          JB      0016
-T                                          83H为奇数，应BX自增1，指向
                                            下一个字节单元
AX=0700  BX=0000  CX=0005  DX=0041  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0016   OV UP EI PL NZ AC PE CY
076B:0016 43            INC     BX
-T
          BX自增1
AX=0700  BX=0001  CX=0005  DX=0041  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0017   NV UP EI PL NZ NA PO CY
076B:0017 E2F5          LOOP    000E
```

用 T 命令执行第 2 次循环，BUF 字节单元中的第 2 个数据 18H 是偶数，累加到 AL，AL＝18H，然后 BX 自增 1，指向第 2 个单元。

用 T 命令执行第 3 次循环，BUF 字节单元中的第 3 个数据 90H 是偶数，累加到 AL，AL＝18H＋90H＝A8H，然后 BX 自增 1，指向第 3 个单元。

用 T 命令执行第 4 次循环，BUF 字节单元中的第 4 个数据 11 是奇数，BX 自增 1，指向第 4 个单元。

```
076B:000E 8A17          MOV     DL,[BX]                         DS:0003=11
-T

AX=07A8  BX=0003  CX=0002  DX=0011  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0010   NV UP EI PL NZ NA PE NC
076B:0010 D0EA          SHR     DL,1
-T

AX=07A8  BX=0003  CX=0002  DX=0008  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0012   NV UP EI PL NZ AC PO CY
076B:0012 7202          JB      0016
-T
AX=07A8  BX=0003  CX=0002  DX=0008  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0016   NV UP EI PL NZ AC PO CY
076B:0016 43            INC     BX
-T

AX=07A8  BX=0004  CX=0002  DX=0008  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0017   NV UP EI PL NZ NA PO CY
076B:0017 E2F5          LOOP    000E
```

CF=1，说明11H是奇数，BX自增1，指向下一个单元

用 T 命令执行第 5 次循环，BUF 字节单元中的第 5 个数据 53H 是奇数，BX 自增 1，指向第 5 个单元，最后所有偶数的累加和为 A8H，存放至 AL 中。

```
076B:000E 8A17          MOV     DL,[BX]                         DS:0004=53
-T

AX=07A8  BX=0004  CX=0001  DX=0053  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0010   NV UP EI PL NZ NA PO CY
076B:0010 D0EA          SHR     DL,1
-T

AX=07A8  BX=0004  CX=0001  DX=0029  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0012   NV UP EI PL NZ AC PO CY
076B:0012 7202          JB      0016
-T
AX=07A8  BX=0004  CX=0001  DX=0029  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0016   NV UP EI PL NZ AC PO CY
076B:0016 43            INC     BX
-T

AX=07A8  BX=0005  CX=0001  DX=0029  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0017   NV UP EI PL NZ NA PE CY
076B:0017 E2F5          LOOP    000E
```

CF=1，说明53H为奇数，BX自增1，BX指向下一个单元

用 T 命令执行循环结果，程序正常结束。

```
-T

AX=07A8  BX=0005  CX=0000  DX=0029  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=0019   NV UP EI PL NZ NA PE CY
076B:0019 B44C          MOV     AH,4C
-T

AX=4CA8  BX=0005  CX=0000  DX=0029  SP=0000  BP=0000  SI=0000  DI=0000
DS=076A  ES=075A  SS=0769  CS=076B  IP=001B   NV UP EI PL NZ NA PE CY
076B:001B CD21          INT     21
-P

Program terminated normally
-
```

【任务 2.3】 BUF 字节单元中存放了 10 个无符号数，在这 10 个数中查找是否存在 12H，如果 12H 存在，则输出 YES，否则输出 NO。


实验三 2.3

源程序 search.asm 如下。

```
DATA SEGMENT
    BUF DB 23H,38H,60H,13H,12H,88H,90H,11H,45H,66H
    LEN EQU $-BUF
    MSG1 DB "YES$"
    MSG2 DB "NO$"
DATA ENDS
CODE SEGMENT
ASSUME CS: CODE,DS: DATA
START:
    MOV AX,DATA
    MOV DS,AX
    LEA BX,BUF          ;将 BUF 缓冲区首地址传送至 BX
    MOV CX,LEN          ;将数据个数传送至 CX
    MOV SI,-1
    L1:
      INC SI            ;SI 自增 1
      MOV DL,[BX+SI]    ;将 BX+SI 指向的字节单元中的内容传送至 DL
      CMP DL,12H        ;DL-12H
    LOOPNE L1           ;结果不相等,继续循环至 L1
    JZ L2
      LEA DX,MSG2
      JMP NEXT
    L2:
      LEA DX,MSG1
    NEXT:
      MOV AH,09H
      INT 21H
    MOV AH,4CH
    INT 21H
CODE ENDS
    END START
```

上机执行过程如下。

第一步：在编辑工具（记事本、DEVC 或 EDIT 等）中输入源程序 search.asm。

第二步：用汇编程序 MASM.EXE 汇编源程序 search.asm，严重错误个数为 0 表示汇编成功，否则需要修改源程序，然后继续执行第二步，直到严重错误个数为 0。

第三步：用链接程序 LINK.EXE 链接文件 search.obj。

第四步：程序中有输出指令，可直接执行可执行文件，输出结果。

```
C:\>search.exe
YES
```


实验三 2.4

【任务 2.4】 STR1 字节单元中存放了字符串“LOVE”，编写源程序，将 STR1 串从右到左依次传送至 STR2 字节单元。

分析：源串 STR1 的末偏移地址传送至 SI，SI＝0003H；目的串 STR2 的末偏移地址传送至 DI，DI＝0003H；方向标志 DF＝1。每重复一次 MOVS 指令，SI 和 DI 自减 1；重复 4 次后，STR1 中的内容传送至 STR2，最后 SI 和 DI 的值为 FFFFH。执行过程如图 3.3 所示。

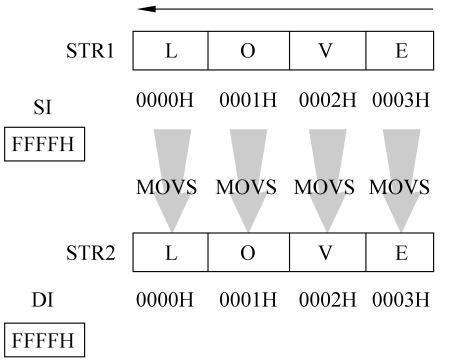


图 3.3　MOVS 指令的执行过程

源程序 movs.asm 如下。

```
DATA SEGMENT
    STR1 DB "LOVE"              ;源串
```