西安电子科技大学

微机系统综合实验 课程实验报告

实验名称 实验二 数制转换与数值运算编程

网络与信息安全学院 2118021 班

姓名 盖乐 学号 21009200991

同作者_____

实验日期 2023 年 4 月 15 日

成 绩

指导教师评语:
指导教师: _____年____月____日

一、 实验要求

1. 将 ASCII 码表示的十进制数转换为二进制数，自行绘制流程图并编写程序。十进制表示为：

$$D_n \times 10^n + D_{n-1} \times 10^{n-1} + \cdots + D_0 \times 10^0 = \sum_{i=0}^{n} D_i \times 10^i \qquad (1)$$

D_i 代表十进制数 0，1，2，…，9；
上式转换为：

$$\sum_{i=0}^{n} D_i \times 10^i = (\cdots((D_n \times 10 + D_{n-1}) \times 10 + D_{n-2}) \times 10 + \cdots + D_1) \times 10 + D_0 \qquad (2)$$

由式（2）可归纳十进制数转换为二进制数的方法：从十进制数的最高位 D_n 开始作乘 10 加次位的操作，依次类推，则可求出二进制数的结果。

表 1.1 数制对应关系表

十六进制	BCD 码	二进制机器码	ASCII 码	七段码	
				共阳	共阴
0	0000	0000	30H	40H	3FH
1	0001	0001	31H	79H	06H
2	0010	0010	32H	24H	5BH
3	0011	0011	33H	30H	4FH
4	0100	0100	34H	19H	66H
5	0101	0101	35H	12H	6DH
6	0110	0110	36H	02H	7DH
7	0111	0111	37H	78H	07H
8	1000	1000	38H	00H	7FH
9	1001	1001	39H	18H	67H
A		1010	41H	08H	77H
B		1011	42H	03H	7CH
C		1100	43H	46H	39H
D		1101	44H	21H	5EH
E		1110	45H	06H	79H
F		1111	46H	0EH	71H

2. BCD 码转换为二进制数

将四个二位十进制数的 BCD 码存放于 3500H 起始的内存单元中，将转换的二进制数存入 3510H 起始的内存单元中，自行绘制流程图并编写程序。

3. 两个非压缩 BCD 数加法程序。

要求：被加数和加数均为 4 位数，从键盘输入，分别放在 SBCD1 和 SBCD2 开始的地址单元中。运算结果放在 SSUM 开始的地址单元中。并在虚拟终端上以“SBCD1+SBCD2=SSUM”格式进行显示，自行绘制流程图并编写程序。

4. 从键盘上输入任意两个不大于 2 位数的正整数，计算其乘积。结果在屏幕上显示，自行绘制流程图并编写程序。

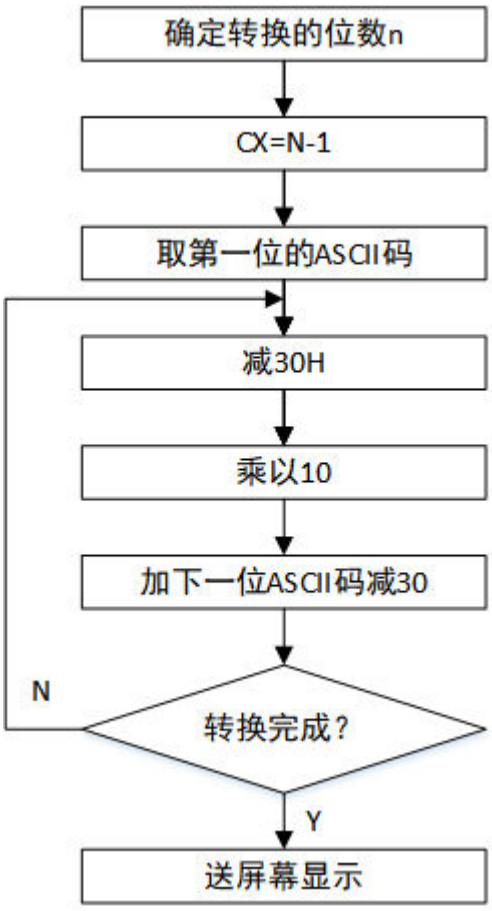
二、 实验目的

1. 掌握不同进制数及编码相互转换的程序设计方法，加深对数制转换的理解。
2. 熟悉在 EMU8086 集成环境中程序调试的方法。
3. 掌握使用运算类指令编程及调试方法。
4. 掌握运算类指令对各状态标志位的影响及其测试方法。
5. 学习使用软件监视变量的方法。

三、 实验代码及实验结果

实验一

1. 实验流程图：

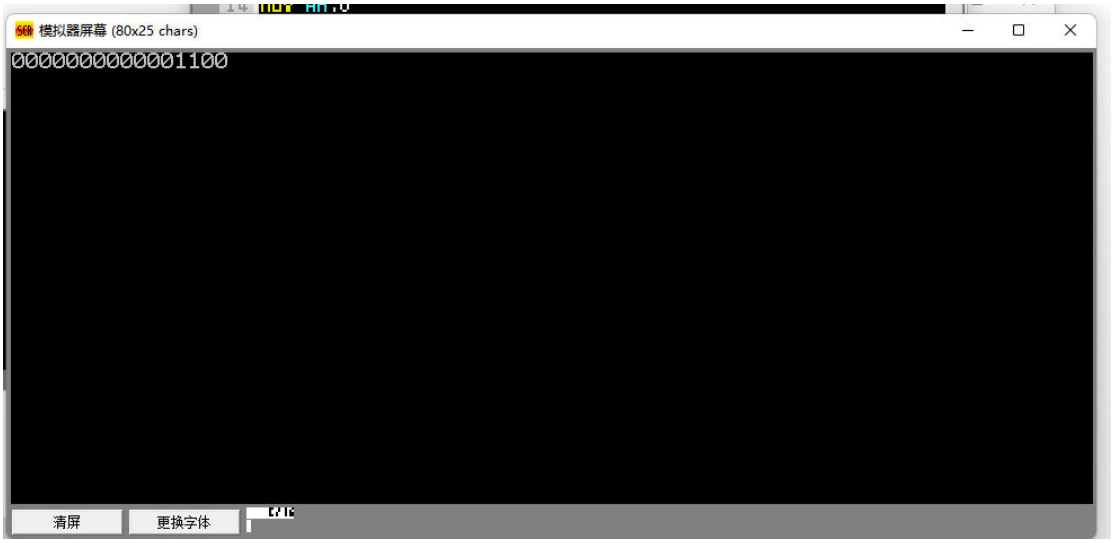


2. 实验代码：

```
DATA SEGMENT ;    定义源数据段
MSR DB '00012';      将十进制数 00012 放入 DB 中
LEN EQU $-MSR
DATA ENDS

CODE SEGMENT ;定义代码段
ASSUME CS:CODE, DS:DATA

START: MOV AX, DATA
MOV DS,AX
        MOV SI, OFFSET MSR
```

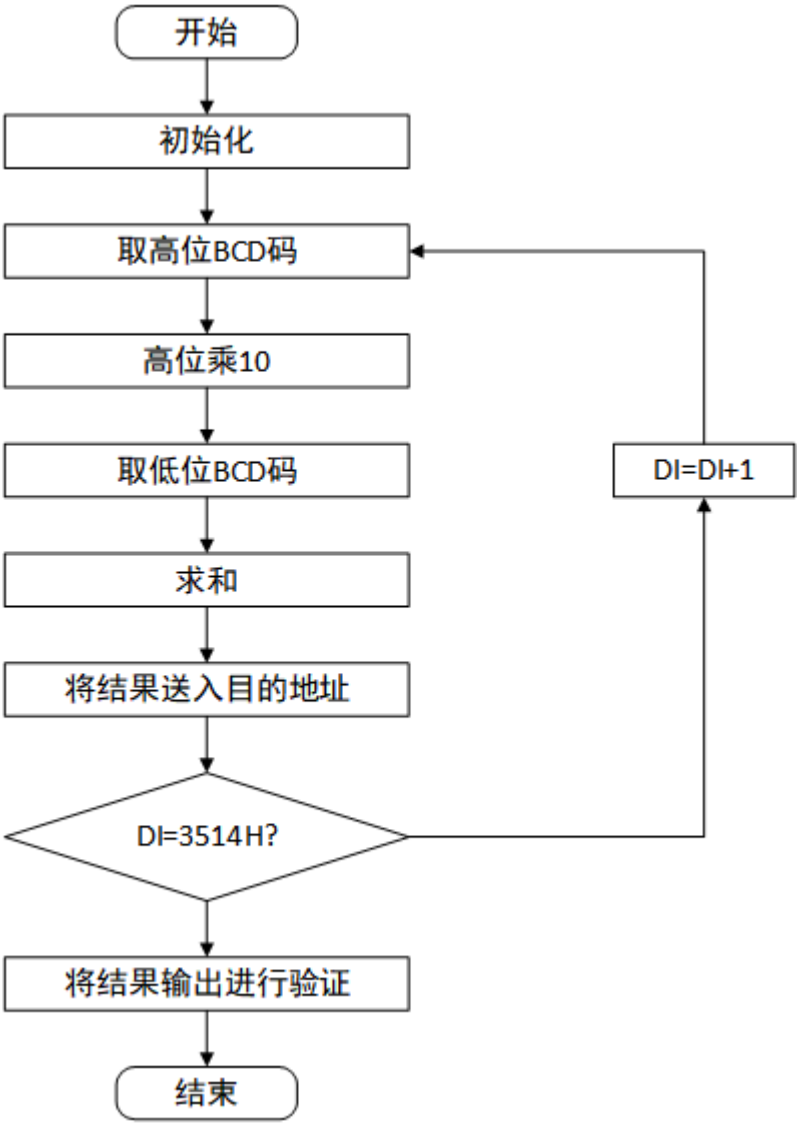
```
MOV CX,LEN-1
        MOV AL, DS:[SI] ;       开始传输数据
MOV AH,0
SUB AX,30H


NEXT1: MOV BX, 0AH
MUL BX ;    将 AX 中的数乘以 10，所得的结果高位保存到 DX 中，低位保存到 AX 中
        INC SI
MOV DL,DS:[SI]
        MOV DH,0
SUB DX,30H ;   将数字对应的 ASCII 码转换成响应的二进制数
        ADD AX,DX
        DEC CX
        JNZ NEXT1
MOV CL,16 ;    CL 用作循环 NEXT2 的计数器，因为 5 位十进制数转化成二进制只有十六位，所以循环次数为 16 次
        MOV BX,AX

NEXT2: ROL BX,1
        MOV DL,BL
        AND DL,01H
        ADD DL,30H ;   把 BX 中的数以二进制形式输出
        MOV AH,2H
        INT 21H
        DEC CL
JNZ NEXT2
MOV AX,4C00H
INT 21H

CODE ENDS
END START
```

3. 实验结果：


模拟器屏幕 (80x25 chars)
0000000000001100
清屏
更换字体

实验二

1. 实验流程图：

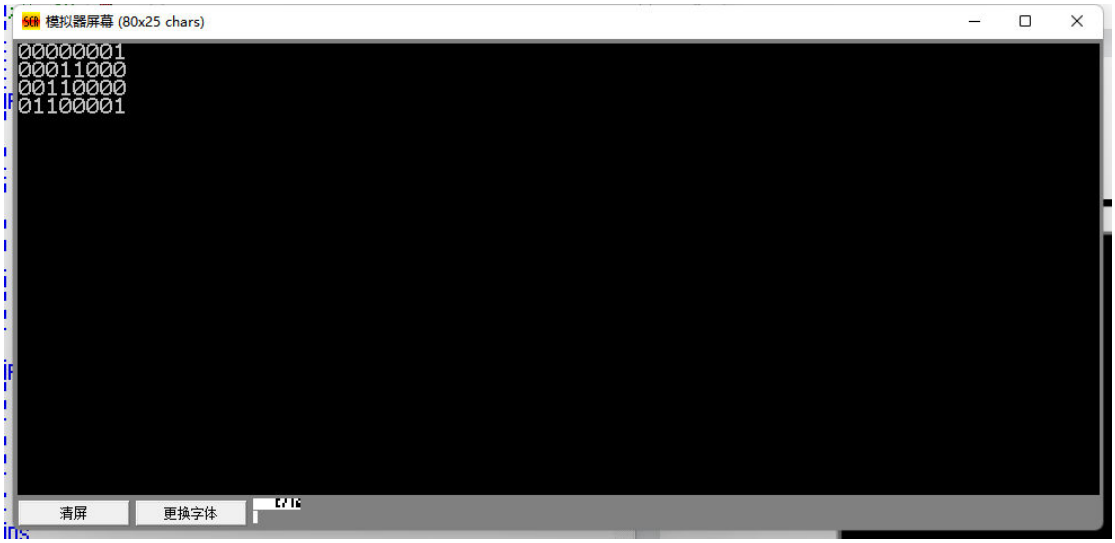

开始
初始化
取高位BCD码
高位乘10
取低位BCD码
求和
将结果送入目的地址
DI=3514H?
DI=DI+1
将结果输出进行验证
结束

2. 实验代码：

```
CODE SEGMENT
    ASSUME CS:CODE
START:
    MOV AX,0
    MOV DS,AX
    MOV SI,3500H;将 SI 置为 3500H,以便后续将数据送入 3500H 开始的内存单元
    MOV [SI],0000B
    MOV [SI+1],0001B
    MOV [SI+2],0010B
    MOV [SI+3],0100B
    MOV [SI+4],0100B
    MOV [SI+5],1000B
    MOV [SI+6],1001B
    MOV [SI+7],0111B;将四个二位十进制数的 BCD 码依次送入
    MOV DI,3510H;将 DI 置为 3510H,以便后续将数据送入 3510H 开始的内存单元
    MOV CX,4;共四个数循环次数位 4
LOOP_1:
    MOV AH,[SI];取一个数的高位
    MOV AL,[SI+1];取一个数的低位
    AAD;将 AX 中两个未压缩十进制数进行处理
    MOV [DI],AL;将最终结果送回 3510 开始的内存单元
    INC SI;
    INC SI;指向下一个数的高位
    INC DI;指向下一个位置
    LOOP LOOP_1
    MOV DI,3510H
LOOP_2:
    MOV BL,[DI]
    INC DI
    MOV CX,8;循环次数为 8
LOOP_3:
    MOV AL,BL
    AND AL,80H;每次只取最高位进行输出
    ROL AL,1
    ADD AL,30H;获取 ASCII 码
    MOV DL,AL
    MOV AH,02H
    INT 21H
    SAL BL,1;将数字左移一位
    LOOP LOOP_3
    MOV DL,0AH
    MOV AH,02H
    INT 21H;输出回车
```

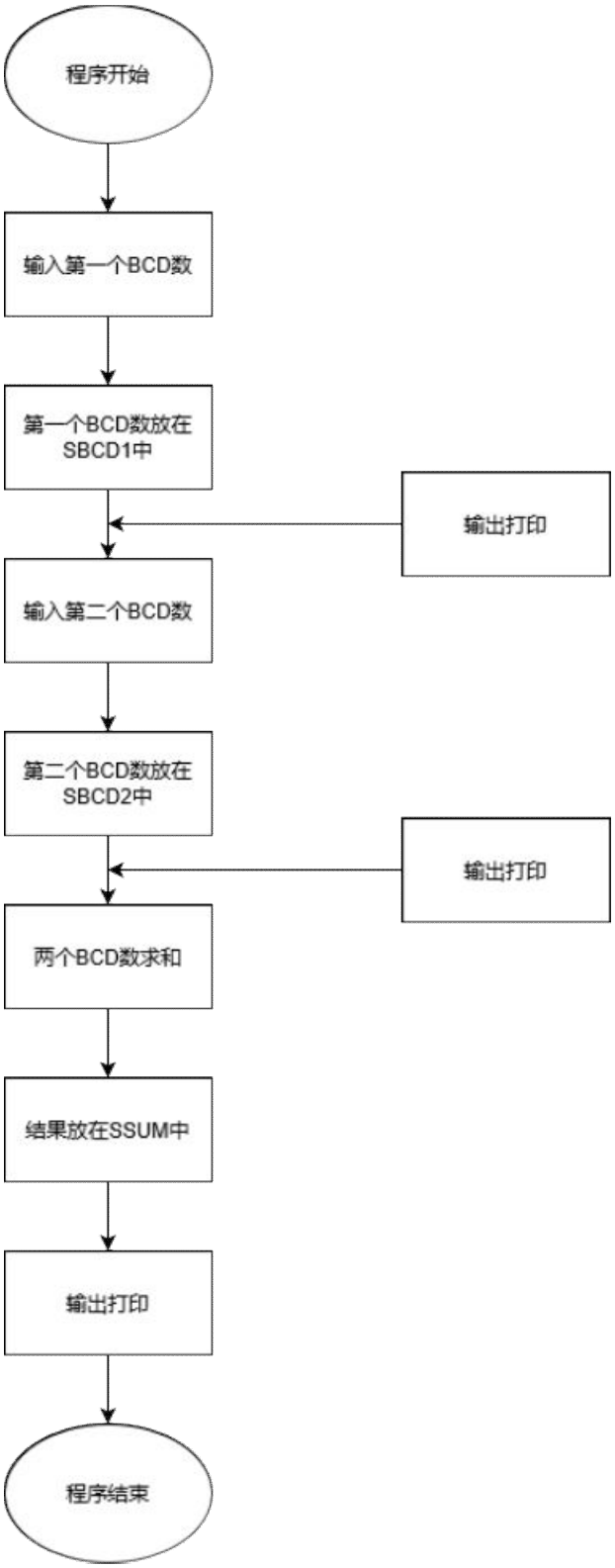
```
    MOV DL,0DH
    MOV AH,02H
    INT 21H;输出换行
    CMP DI,3514H
    JNE LOOP_2
CODE ENDS
END START
```

3. 实验结果：



实验三

1. 实验流程图：



2. 实验代码：

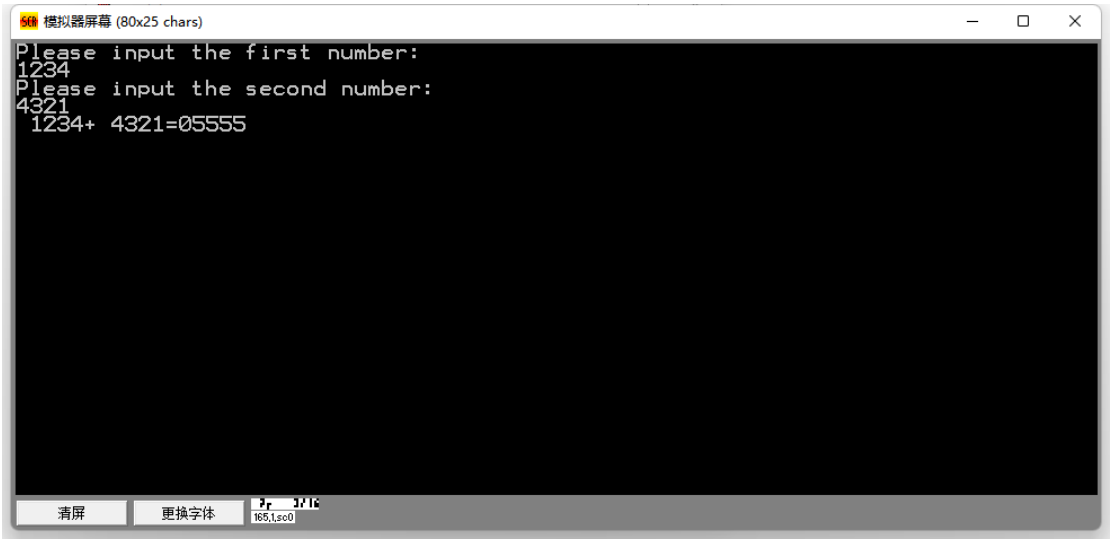
```
DATA SEGMENT
    STRING_1 DB 'Please input the first number:',0AH,0DH,'$'
    STRING_2 DB 0AH,0DH,'Please input the second number:',0AH,0DH,'$'
    SBCD1 DB 0AH,0DH,0,0,0,0,0,'+','$'
    SBCD2 DB 0,0,0,0,0,'=','$'
```

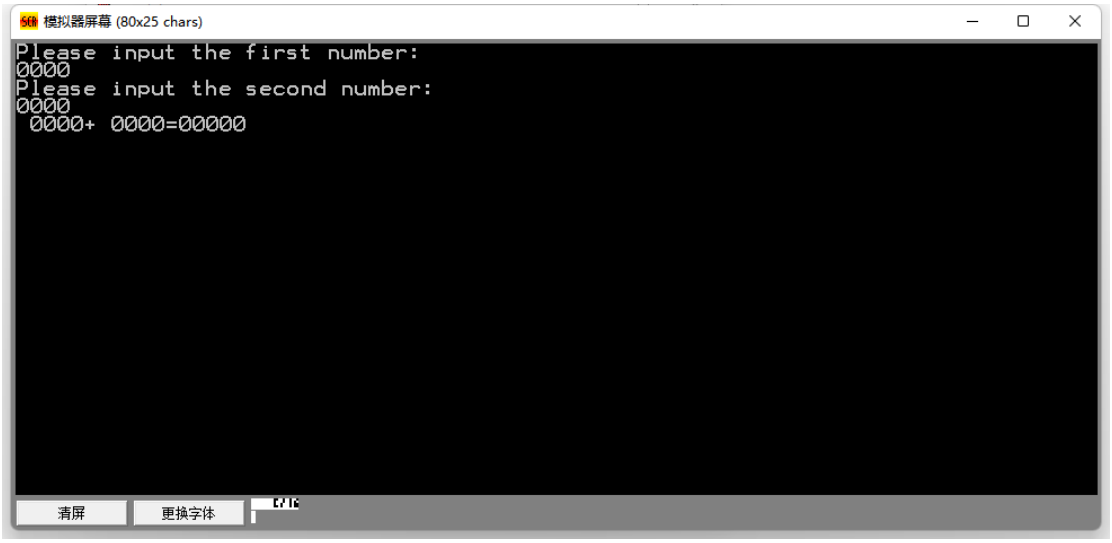
```
    SSUM DB 0,0,0,0,0,'$'
DATA ENDS

STACK SEGMENT
    STA DB 10 DUP(0)
STACK ENDS

CODE SEGMENT
    ASSUME CS:CODE,DS:DATA,SS:STACK
START:
    MOV AX,DATA
    MOV DS,AX
    MOV DX,OFFSET STRING_1;
    MOV AH,09H
    INT 21H;输出第一个数的提示信息
    MOV SI,OFFSET SBCD1
    ADD SI,3;前面有 0AH,ODH,0
    CALL INPUT;调用 input 过程获取输入并送入 SI 指向位置
    MOV DX,OFFSET STRING_2;
    MOV AH,09H
    INT 21H;输出第二个数的提示信息
    MOV SI,OFFSET SBCD2
    INC SI;前面有一个 0
    CALL INPUT;调用 input 过程获取输入并送入 SI 指向位置
    MOV DX,OFFSET SBCD1
    MOV AH,09H
    INT 21H;输出换行符，第一个数和加号
    MOV DX,OFFSET SBCD2
    MOV AH,09H
    INT 21H;输出第二个数和等号
    MOV SI,OFFSET SBCD1
    ADD SI,3;前面有 0AH,ODH,0
    CALL GET;将第一个四位数中的每一位换成 BSD 码
    MOV SI,OFFSET SBCD2
    INC SI;前面有一个 0
    CALL GET;将第二个四位数中的每一位换成 BCD 码
    DEC SI;将 SI 指向第二个四位数的最后一位（GET 过程中多加了一次）
    MOV DI,OFFSET SBCD1
    ADD DI,6;将 DI 指向第一个四位数的最后一位
    MOV CX,5;循环 5 次，进行 5 次加法
    CLC;清除 CF 位
    MOV BX,OFFSET SSUM;
    ADD BX,4;将 BX 指向和的最后一位
LOOP_3:
```

```
    MOV AL,[SI]
    ADC AL,[DI];用最低位进行带进位加法
    AAA;将 AL 中未压缩的十进制进行调整
    MOV [BX],AL;送回
    DEC SI
    DEC DI
    DEC BX
    LOOP LOOP_3
    MOV SI,OFFSET SSUM
    CALL GET_1;将每一位数字转化为 ASCII 码
OUTPUT:
    MOV DX,OFFSET SSUM
    MOV AH,09H
    INT 21H;输出最终结果
    JMP OVER

INPUT PROC
    PUSH CX
    PUSH AX
    MOV CX,4;循环 4 次
LOOP_1:
    MOV AH,01H
    INT 21H
    MOV [SI],AL;获取一位输入
    INC SI;SI+1
    LOOP LOOP_1
    POP AX;恢复使用过的 AX 和 CX
    POP CX;恢复使用过的 AX 和 CX
    RET;返回
INPUT ENDP

GET PROC
    XOR AX,AX
    MOV CX,4;循环 4 次
LOOP_2:
    MOV AL,[SI]
    SUB AL,30H;对每一位减 30H
    MOV [SI],AL
    INC SI
    LOOP LOOP_2
    RET;返回
GET ENDP

GET_1 PROC
```

```
    XOR AX,AX
    MOV CX,5;循环 5 次
LOOP_4:
    MOV AL,[SI]
    ADD AL,30H;对每一位加 30H 将 BCD 转化为 ASCII 码
        MOV [SI],AL
         INC SI
         LOOP LOOP_4
         RET;返回
  GET_1 ENDP

  OVER:

  CODE ENDS
  END START
```

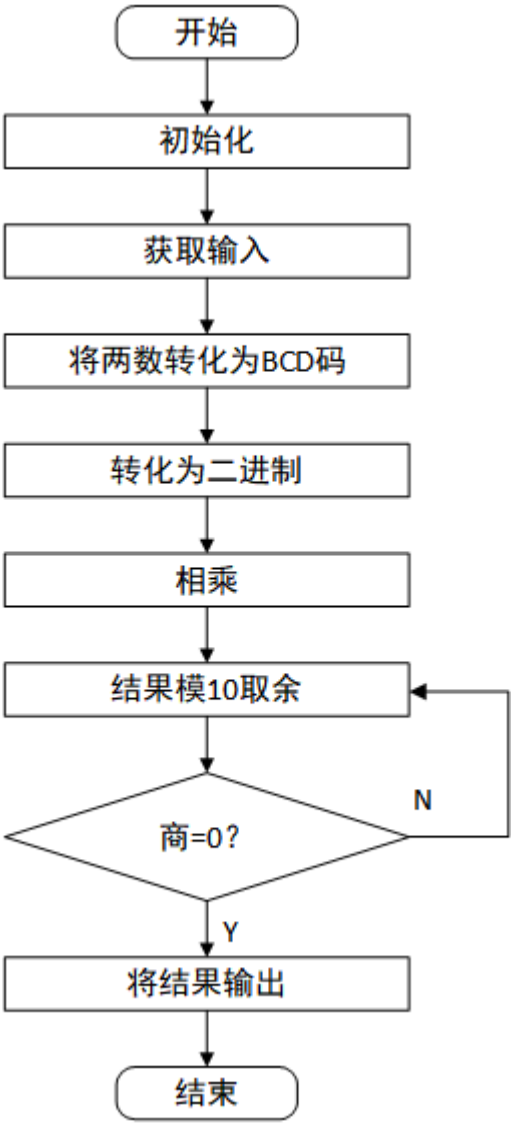
3. 实验结果：





实验四

1. 实验流程图：



2. 实验代码：

```
DATA SEGMENT
    STR1 DB 'PLEASE INPUT THE FIRST NUMBER:$'
    STR2 DB 'PLEASE INPUT THE SECOND NUMBER:$'
    STR3 DB 'RESULT=$'
    A DW ?
    B DW ?
    C DW ?
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE,DS:DATA
    MAIN PROC FAR
    START:
        MOV AX,DATA
        MOV DS,AX
```

```
        LEA DX,STR1
        MOV AH,9H
        INT 21H
        CALL INPUT
        MOV A,BX
        LEA DX,STR2
        MOV AH,9H
        INT 21H
        CALL INPUT
        MOV B,BX
        LEA DX,STR3
        MOV AH,9H
        INT 21H
        MOV AX,A
        AAD
        MOV BX,AX
        MOV AX,B
        AAD
        MUL BX
        MOV WORD PTR C,AX
        MOV DI,0AH
        MOV CX,0
    TOTEN:
        DIV DI
        PUSH DX
        INC CX
        CMP AX,0
        JE PRINT
        CWD
        JMP TOTEN
    PRINT:
        POP DX
        ADD DL,30H
        MOV AH,2H
        INT 21H
    LOOP PRINT
        RET
    MAIN ENDP
    INPUT PROC
        MOV BX,0
        INPUTA:
        MOV AH,1
        INT 21H
        SUB AL,30H
```

```
        MOV BL,AL
        MOV AH,1
        INT 21H
        CMP AL,0DH
        JE EXIT
        SUB AL,30H
        MOV CL,8
        ROL BX,CL
        MOV BL,AL
        JMP EXIT
        EXIT:
            CALL CRLF
        RET
    INPUT ENDP
    CRLF PROC NEAR
        MOV DL,0AH
        MOV AH,2H
        INT 21H
        MOV DL,0DH
        MOV AH,2H
        INT 21H
        RET
    CRLF ENDP

CODE ENDS
END START
```

3. 实验结果：

四、 实验总结

本次实验中，我们学习了 8086 汇编语言的基础知识和一些常用的指令，以

及如何在汇编程序中进行输入输出。在实践过程中，我们也发现了一些需要注意的点：

1.在 8086 汇编中，如果要实现换行输出，需要同时使用“回车”和“换行”两个控制字符。

2.汇编程序中不同寄存器有不同的作用，特别是 DX 和 AX 寄存器经常用于存储运算结果和源数据，因此需要特别小心处理，避免数据丢失。

3.如果需要输出字符串，必须在字符串末尾加上‘$’符号来表示字符串的结束位置。

4.关于栈的使用，需要注意栈的大小，防止溢出，同时在使用 DI 和 SI 指针时也要小心处理，避免越界出现错误。