成 绩 _____

北京航空航天大學
BEIHANG UNIVERSITY

微机原理及接口技术
实验报告

院（系）名称	自动化科学与电气工程学院
专 业 名 称	自动化
学 生 学 号	16711094
学 生 姓 名	李翰韬
指 导 教 师	林新

2018 年 11 月

实验一　　字符串排序及 BCD 码加法

实验时间 11.18　　实验编号　无　　同组同学　无

一、实验背景

1. 利用 DOS 功能调用的 INT 21H 的 1 号功能使程序能够从键盘每次输入单个字符到寄存器或内存单元，并在屏幕上显示出来。

2. 利用冒泡法，对输入的字符串按字符对应的 ASCII 码的大小从小到大排序。

3. 将字符串利用 DOS 功能调用的 INT 21H 的 9 号功能显示在微机屏幕上。

4. 从微机的键盘输入两个 4 位的十进制数数字，将它们相加的结果，以十进制数的形式显示在微机的屏幕上（如 1234+5678=06912）。

5. 编写子程序实现某项功能，并在主程序中调用。

二、实验原理

1. 分析实验的工作原理。

①. 字符串排序

冒泡算法

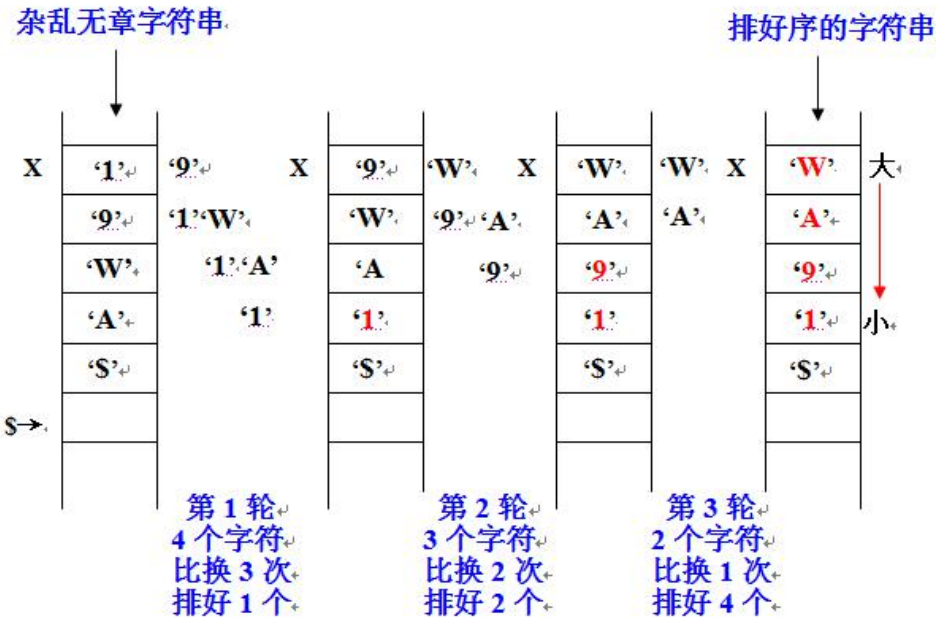


图 2.1.3　冒泡算法原理图

②. 四位 BCD 码相加

1）数据段中需要设置两个变量分别来存放输入数据，每个数据 4 位，每位对应一个字节；相加的结果可能为 5 位数，故还需设置一个 5 字节的变量；

2）数据输入子程序需要使用 INT21H 的 8 号功能。

3）算术等式可以实现一种或者多种方法显示。如：

1234 + 5678 = 06912　　（不要显示 06912）

1234 + 5678 = 6912　　（不要显示 06912）

0051 + 0052 = 103　　（不要显示 0103）

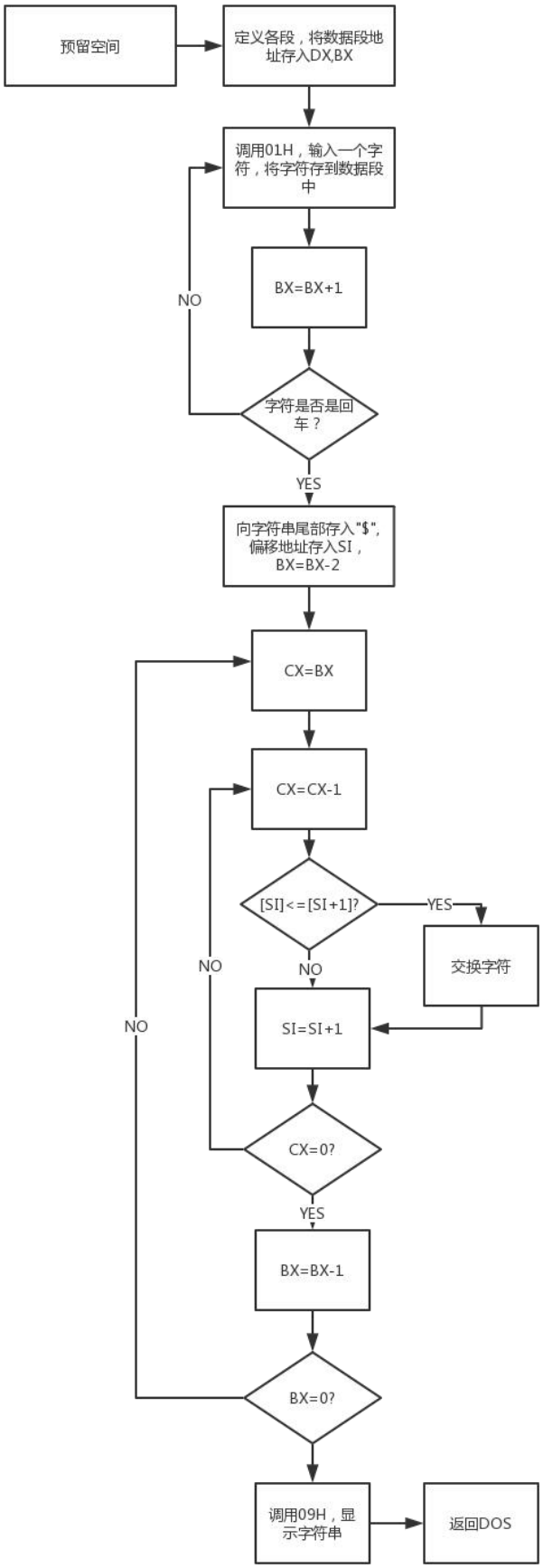
0000 + 0000 = 0　　（结果为 0，不能一个 0 也不显示）

有兴趣的同学，也可将输入的数字及运算结果以竖式的形式显示，如：

```
   1 2 3 4
 + 5 6 7 8
   6 9 1 2
```

2. 流程图

图 1. 字符串排序流程图


预留空间
定义各段，将数据段地址存入DX,BX
调用01H，输入一个字符，将字符存到数据段中
BX=BX+1
NO
字符是否是回车？
YES
向字符串尾部存入"$"，偏移地址存入SI，BX=BX-2
CX=BX
CX=CX-1
[SI]<=[SI+1]?
YES
交换字符
NO
NO
SI=SI+1
NO
CX=0?
YES
BX=BX-1
BX=0?
调用09H，显示字符串
返回DOS

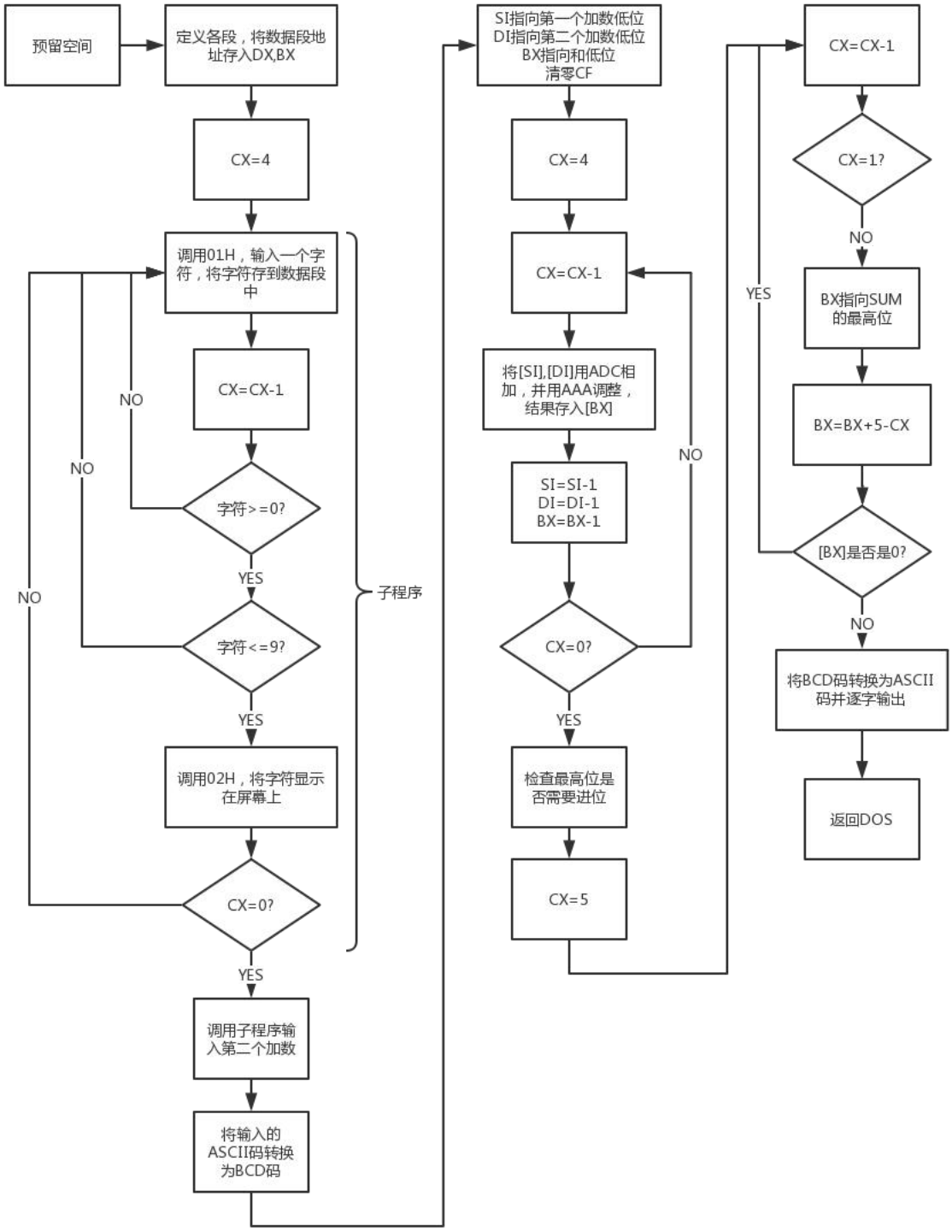


图 2. 四位数加法流程图

三、预习思考题及实验验证

1. 下面指令是否正确

MOV　[SI], AL;　　正确

MOV　[SI] , 12H;　不正确，存储空间和立即数不能直接交换数据

MOV　AL , 12H ;　正确

MOV　DS , 1002H;　不正确，立即数和段存储器不能直接交换数据

2. 分别用 DOS 功能调用的 INT 21H 的 2 号功能和 9 号功能，编程实现在计算机屏幕上输出显示字符串“Welcome to TPC！”。

①2 号功能

```
DATAS SEGMENT
    OUTPUT DB 'Welcome to TPC!'
    COUNT EQU $-OUTPUT
DATAS ENDS

STACKS SEGMENT 'STACK'
     DB 100 DUP(?)
STACKS ENDS

CODES SEGMENT
        ASSUME
CS:CODES,DS:DATAS,SS:STACKS
START: MOV AX,DATAS
       MOV DS,AX
       MOV BX,0
       MOV CX,COUNT
LOOP1: MOV DL,OUTPUT[BX]
       MOV AH,02H
        INT 21H
        INC BX
        LOOP LOOP1
        MOV AX,4C00H
        INT 21H
CODES ENDS
     END START
```

②9 号功能

```
DATAS SEGMENT
    OUTPUT DB 'Welcome to TPC!$'
DATAS ENDS

STACKS SEGMENT 'STACK'
     DB 100 DUP(?)
STACKS ENDS

CODES SEGMENT
     ASSUME
CS:CODES,DS:DATAS,SS:STACKS
START: MOV AX,DATAS
       MOV DS,AX
       LEA DX,OUTPUT
```

```
        MOV AH,09H
        INT 21H
        MOV AX,4C00H
        INT 21H
CODES ENDS
    END START
```

3. 在数据段中存放字符串“ASM2015”，将该字符串按 ASCII 码从大到小排序后输出显示。

```
DATAS SEGMENT
    BUF     DB 'ASM2015'
    COUNT EQU $-BUF
          DB '$'
DATAS ENDS

STACKS SEGMENT 'STACK'
    DB 100 DUP(?)
STACKS ENDS

CODES SEGMENT
    ASSUME
CS:CODES,DS:DATAS,SS:STACKS
START: MOV AX,DATAS
       MOV DS,AX
       MOV CX,COUNT
       DEC CX
       MOV DX,CX
L1:    MOV CX,DX
       MOV BX,0
L2:    MOV AL,BUF[BX]
       CMP AL,BUF[BX+1]
       JAE COU
       XCHG AL,BUF[BX+1]
       MOV BUF[BX],AL
COU:   INC BX
       LOOP L2
       DEC DX
       JNZ L1
       MOV AH,09H
       LEA DX,BUF
       INT 21H
       MOV AX,4C00H
       INT 21H
CODES ENDS
    END START
```

四、实验源程序

```
1.字符串排序
DATAS SEGMENT
    X1 DB 100 DUP('0')
DATAS ENDS
```

```
STACKS SEGMENT
    X DB 50 DUP('0')
STACKS ENDS

CODES SEGMENT
    ASSUME CS:CODES,DS:DATAS,SS:STACKS
MAIN PROC
    MOV AX,DATAS
    MOV DS,AX;将数据段段地址存入 DS
    MOV BX,OFFSET X1;将偏移地址存入 BX
KEYIN: MOV AH,1
    INT 21H;调用 01H 功能输入
    MOV DL,AL;将此输入字节存在 DL 中
    MOV [BX],AL;将此输入存在数据段中
    INC BX
    CMP DL,0DH
    JNZ KEYIN;若不是回车，则重复输入功能，是回车则输入结尾字符
    MOV AL,'$'
    MOV [BX],AL
    SUB BX,2;将 BX-2，除去结尾字符，指向真正的字符串长度
NEXT:  MOV CX,BX;将 BX 中总字符长度存入 CX 中冒泡的次数
    MOV SI,OFFSET X1;将 X1 的偏移地址存入 SI
CHANGE:  MOV AL,[SI];读取 SI 偏移地址所指向的字符
    CMP AL,[SI+1];将 SI 偏移地址所指向的字符与 SI+1 偏移地址所指向的字符比较
    JBE NEXT1;从小至大递增排序，若[SI]比[SI+1]，则交换，否则跳过
    XCHG AL,[SI+1]
    MOV [SI],AL
NEXT1:  INC SI
    LOOP  CHANGE
    DEC BX
    JNE NEXT
    MOV DX,OFFSET X1
    MOV AH,9;调用 09H 功能显示字符串
    INT 21H
  MOV AH,4CH;调用 4CH 功能中断返回 DOS
    INT 21H
MAIN ENDP
CODES ENDS
END MAIN
```

2.4 位 BCD 加法

```
DATAS SEGMENT
        NUM1    DB    4 DUP('0')
        NUM2    DB    4 DUP('0')
        SUM     DB    5 DUP('0')
DATAS ENDS

STACKS SEGMENT
        DW    100 DUP('0')
STACKS ENDS

CODES SEGMENT
        ASSUME
CS:CODES,DS:DATAS,SS:STACKS
MAIN     PROC
        MOV AX,DATAS
        MOV DS,AX;将数据段段地址存入 DS
         MOV     BX,OFFSET NUM1;将 NUM1 偏移地址存入 BX
        MOV    CX,4;CX=4
IN1:CALL KEYIN;输入第一个四位数
        AND    AL,0FH;使高四位清零，将字符转换为 BCD
        MOV    [BX],AL
        INC    BX
        LOOP IN1;将输入的四位数存入 X1
        MOV    AH,2
        MOV    DL,'+'
        INT    21H;调用 02H 功能显示加号
        MOV     BX,OFFSET NUM2;将 NUM2 偏移地址存入 BX
        MOV    CX,4;将 CX 重新置为 4，准备输入第二个数
    IN2:CALL KEYIN;输入第二个四位数，与上同理
        AND    AL,0FH
        MOV    [BX],AL
        INC    BX
        LOOP IN2
        MOV    AH,2
        MOV    DL,'='
        INT    21H;调用 02H 功能显示等于号
    MOV SI,(OFFSET NUM1)+3;将 SI 指向第一个加数的低位
    MOV DI,(OFFSET NUM2)+3;将 DI 指向第二个加数的低位
    MOV BX,(OFFSET SUM)+4;将 BX 指向结果的低位
        MOV    CX,4
        OR     CX,CX;清零
    NEXT:  MOV    AL,[SI]
        ADC    AL,[DI];将俩个数相加
```

```
        AAA;加法 ASCII 调整
        MOV   [BX],AL;将结果存入
SUM
        DEC   SI
        DEC   DI
        DEC   BX
        LOOP NEXT;一位一位加，循
环 4 次
        MOV   AL,0
        ADC   AL,0;看最高位是否需
要进位
        MOV   [BX],AL
        MOV   CX,5
    NEXT1: MOV BX,OFFSET SUM
        MOV AX,5
        SUB AX,CX
        ADD BX,AX;使 BX 指向当前
需要判断的高位
        MOV AL,[BX]
        CMP AL,0
        JNZ OUTP;看当前最高位是
否是 0，若不是 0 则输出显示，是 0 则循
环，直到不是 0
        CMP CX,1
        JNA OUTP
        LOOP NEXT1
    OUTP:   MOV   DL,[BX];回显最终
的加和结果
        ADD   DL,30H;将结果转换
为 ASCII
        MOV   AH,2
        INT   21H;调用 02H 功能输
出
        INC   BX
        LOOP OUTP
    MOV AH,4CH;调用 4CH 功能中
断返回 DOS
        INT 21H
        MAIN    ENDP
        KEYIN   PROC;定义子函数，每
次输入时比较是否是一个 0-9 的数
    AGAIN:MOV AH,8
        INT 21H
        CMP AL,30H
        JB AGAIN
        CMP AL,39H
        JA AGAIN
        MOV DL,AL
        MOV AH,2
        INT 21H
        RET
    KEYIN ENDP

    CODES ENDS
        END MAIN
```

五、实验过程与结果

1. 字符串排序

1)打开运行 TPC-ZK-II 集成开发环境；

2)建立新的.asm 空白文件，将编写好的程序拷贝到空白文件内，或者直接在 TPC-ZK-II 集成开发环境运行界面下打开已编写好并且保存的汇编语言源程序文件，保存该文件；

3)对程序进行编译、连接、运行。从键盘输入一定长度的字符串，最后按回车结束输入并开始排序；

4)排序结果显示正确。

2. 四位 BCD 码相加

1)打开运行 TPC-ZK-II 集成开发环境；

2)建立新的.asm 空白文件，将编写好的程序拷贝到空白文件内，或者直接在 TPC-ZK-II 集成开发环境运行界面下打开已编写好并且保存的汇编语言源程序文件，保存该文件；

3)对程序进行编译、链接、调试，生成可执行文件。；

4)运行，从键盘输入两个 4 位数，显示实验结果，结果验证程序正确。

六、结果分析与实验结论

1. 两个实验结果均同预期结果一样。

2. 第二个实验中，判断高位循环时，要注意在最低位时跳出。否则将无法跳出循环，程序出现乱码。

七、实验后思考题

1. 从键盘输入一个字符，并在屏幕上显示，检查 Ctrl-Break。

2. 用到，作用是条件转移。

3. 注意在字符串结尾添加”$”。

4. 01H 功能输入后在屏幕上显示，08H 功能不显示；简单输入时使用 01H 功能，若需要先判断输入是否合法，则使用 08H 功能。

5. 字符型可用 09H 进行显示，数值型用 02H 进行显示。

6. 从最高位开始，依次是否为 0，若为 0 则继续判断，不为 0 则此位为最高位。

八、收获、体会及建议

1. 本次试验中，学习了 TPC-ZK-II 集成开发环境的使用方法。自己第一次从头至尾编写程序并成功编译，可以得到书本上内容所不能带给我们的编程经验和调试经验。加深了对 8086 程序的理解并能熟练使用。