東北大學
Northeastern University

汇编语言程序设计

主讲： 刘松冉

单位： 东北大学计算机学院
智慧系统国际联合实验室

联系方式： liusongran@cse.neu.edu.cn

个人主页： http://faculty.neu.edu.cn/liusongran
https://liusongran.github.io/

第十章 算术运算与代码转换

一. 多字节加减运算

二. 多字节整数乘除运算

三. BCD码运算

四. 浮点数的加减法

五. 十进制数的ASCII码串转换为二进制定点数

六. 二进制定点数转换为十进制数的ASCII码串

一. 多字节加减运算

例：内存DATA1和DATA2分别存放一个多字节数据,数据长度在LEGH单元存放。编制程序计算两个数据之和并存入SUM开始的单元。

一. 多字节加减运算

例：内存DATA1和DATA2分别存放一个多字节数据，数据长度在LEGH单元存放。编制程序计算两个数据之和并存入SUM开始的单元。

目前程序中未考虑高字节产生进位的情况，如果考虑最高字节产生进位，应如何修改程序？

```
1   DSEG    SEGMENT
2   DATA1   DB      85H,27H,4AH; (4A2785H)
3   DATA2   DB      93H,87H,65H; (658793H)
4   LEGH    DW      3
5   SUM     DB      3  DUP(0)
6   DSEG    ENDS
7   CSEG    SEGMENT
8           ASSUME  CS:CSEG,DS:DSEG
9   START:  MOV     AX,DSEG
10          MOV     DS,AX
11          LEA     SI,DATA1
12          LEA     BX,DATA2
13          LEA     DI,SUM
14          MOV     CX,LEGH
15          CLC
16  AGAIN:  MOV     AL,[SI]
17          ADC     AL,[BX]
18          MOV     [DI],AL
19          INC     SI
20          INC     BX
21          INC     DI
22          LOOP        AGAIN
23          MOV     AH,4CH
24          INT     21H
25  CSEG    ENDS
26          END     START
```

一. 多字节加减运算

例：内存DATA1和DATA2分别存放一个多字节数据，数据长度在LEGH单元存放。编制程序计算两个数据之和并存入SUM开始的单元。

```
1   DSEG    SEGMENT
2   DATA1   DB      85H,27H,4AH; (4A2785H)
3   DATA2   DB      93H,87H,65H; (658793H)
4   LEGH    DW      3
5   SUM     DB      3 DUP(0)
6   DSEG    ENDS      4
7   CSEG    SEGMENT
8           ASSUME  CS:CSEG,DS:DSEG
9   START:  MOV     AX,DSEG
10          MOV     DS,AX
11          LEA     SI,DATA1
12          LEA     BX,DATA2
13          LEA     DI,SUM
14          MOV     CX,LEGH
15          CLC
16  AGAIN:  MOV     AL,[SI]
17          ADC     AL,[BX]
18          MOV     [DI],AL
19          INC     SI
20          INC     BX
21          INC     DI
22          LOOP        AGAIN
23          MOV     AH,4CH
24          INT     21H
25  CSEG    ENDS
26          END     START
```

ADC BYTE PTR [DI], 0

一. 多字节整数乘除运算

1. 一般整数乘法运算
2. 多字节整数乘法运算
3. 一般整数除法运算
4. 多字节整数除法运算

二. 多字节整数乘除运算

1. 一般整数乘法运算

- 指令：MUL，IMUL
- 数据类型：字节型，字型
- 与符号标志CF、OF的联合使用

二. 多字节整数乘除运算

1. 一般整数乘法运算
2. **多字节整数乘法运算**
   - 多字节乘法是一类通用技术
   - 手算乘法过程与模拟人工乘法过程

手算乘法过程：

```
    0 1 1 0
  × 1 0 1 1
-----------
    0 1 1 0
  0 1 1 0
0 0 0 0
+ 0 1 1 0
-----------
• 1 0 0 0 0 1 0
```

模拟人工乘法过程：

```
    0 1 1 0     被乘数
  × 1 0 1 1     乘数
-------------
    0 0 0 0     中间结果（开始为零）
  + 0 1 1 0     部分积（0110×1）
-------------
    0 1 1 0     中间结果
  + 0 1 1 0     部分积（0110×1）
-------------
  1 0 0 1 0     中间结果
+ 0 0 0 0       部分积（0110×0）
-------------
  0 1 0 0 1 0   中间结果
+ 0 1 1 0       部分积（0110×1）
-------------
  1 0 0 0 0 1 0 结果
```

二进制数相乘，部分积的计算实际上是用“1”乘被乘数或者用“0”乘被乘数；所以中间结果的计算是根据乘数的每一位状态是“1”还是“0”而决定中间结果加被乘数，还是不加。

二. 多字节整数乘除运算

1. 一般整数乘法运算

2. 多字节整数乘法运算 — 算法

1) 取乘数和被乘数；
2) 中间结果单元清零；
3) 若乘数为零则结束乘法；
4) 乘数逻辑右移一位,最低位移入进位标志CF中,
   如果CF为零则转第（6）步；
5) 中间结果加上被乘数；
6) 被乘数左移一位；
7) 重复第3、4、5、6步,直到乘完所有位。

模拟人工乘法过程：

```
    0 1 1 0     被乘数
  × 1 0 1 1     乘数
  -----------
    0 0 0 0     中间结果（开始为零）
  + 0 1 1 0     部分积（0110×1）
  -----------
    0 1 1 0     中间结果
  + 0 1 1 0     部分积（0110×1）
  -----------
  1 0 0 1 0     中间结果
+ 0 0 0 0       部分积（0110×0）
  -----------
  0 1 0 0 1 0   中间结果
+ 0 1 1 0       部分积（0110×1）
  -----------
  1 0 0 0 0 1 0 结果
```

二. 多字节整数乘除运算

1. 一般整数乘法运算

2. 多字节整数乘法运算 – 算法

```
30  MUL8     PROC
31           XOR     AH,AH
32           XOR     DX,DX
33  MUL80:   OR      BL,BL
34           JNZ     MUL81
35           RET
36  MUL81:   SHR     BL,1
37           JNC     MUL82
38           ADD     DX,AX
39  MUL82:   SHL     AX,1
40           JMP     MUL80
41  MUL8     ENDP
```

子程序说明文件如下：
(1)子程序名：MUL8
(2)子程序功能：两个8位数相乘
(3)入口条件：被乘数在AL中，
乘数在BL中
(4)出口条件：乘积在DX中
(5)受影响的寄存器：
F, AX, BL, DX

二. 多字节整数乘除运算

3. 一般整数除法运算

- 指令：DIV，IDIV
- 数据类型：字节型，字型
- 与符号标志CF、OF的联合使用

二. 多字节整数乘除运算

4. 多字节整数除法运算

用程序实现除法常常采用模拟人工笔算的方法，下面举例说明除法过程，为了使问题简单，叙述方便，假设被除数为8位二进制数，除数为4位二进制数，如下所示：

01000011÷0110=1011　　余0001

```
          1011   ……商
0110 ) 01000011
       0110
       ------
       001001
         0110
         ----
         00111
          0110
          ----
          0001 ……余数
```

由此过程可以看出人工笔算除法的步骤：

(1)判断被除数(以后为余数)是否大于除数。若大于除数,则从被除数(后为余数)中减去除数,该位商上1；否则不减除数,商上0；

(2)落下被除数中的下一位,重复第一步,直至得到商的最低位。

用计算机模拟人工运算，结合计算机提供的指令功能，其过程如下所示：

二. 多字节整数乘除运算

4. 多字节整数除法运算

具体算法过程如下：

1）取被除数和除数；

2）设置运算次数（获得商的位数）；

3）被除数和商左移一位；

4）比较；

5）不够减,商上0,转（7）；

6）够减则相减,商上1；

7）运算次数减1,不为0,转（3）；

8）运算次数为0,结束除法运算。

二. 多字节整数乘除运算

4. 多字节整数除法运算

例：设被除数为M个字节,存放在DIVND开始的连续单元；除数为N个字节,存放在DIVOR开始的连续单元；其中M>N,两数均为无符号整数。求其商和余数,并分别存入QUO和REM开始的连续单元。

如果被除数M个字节中的前N个字节（高位部分）小于除数,则商为M–N个字节,余数为N个字节。

我们可以按照前面介绍的多字节整数除法的算法编制多字节整数除法,此时,要解决多字节数据的比较、相减与移位,这些操作都可以用子程序来实现。各子程序说明文件及其清单如下：

二. 多字节整数乘除运算

多字节数据比较子程序MCMP：

1）子程序名：MCMP；

2）子程序功能：多字节数据比较（无符号数）；

3）入口条件：两数最高字节地址在SI和DI中，数据长度在AH中；

4）出口条件：进位标志CF=1,被减数小于减数
进位标志CF=0,被减数大于或等于减数；

5）受影响的寄存器：F。

high 被除数 | 1DH | 68H | DCH | E4H | D4H | 5FH | low

除数 | 4AH | 27H | 85H |

```
1    ;多字节数据比较子程序MCMP
2  MCMP    PROC
3          PUSH    SI
4          PUSH    AX
5          PUSH    DI
6  MCMP1:  MOV     AL,[SI]
7          CMP     AL,[DI]
8          JNZ     MCMPR
9          DEC     SI
10         DEC     DI
11         DEC     AH
12         JNZ     MCMP1
13 MCMPR:  POP     DI
14         POP     AX
15         POP     SI
16         RET
17   MCMP    ENDP
```

二. 多字节整数乘除运算

多字节数据相减子程序MSUB:

1）子程序名：MSUB；

2）子程序功能：多字节数据相减；

3）入口条件：被减数和减数的低字节地址分别在SI和DI中，

字节数在AH中；

4）出口条件：结果值在被减数单元（SI为地址指示器）；

被减数大于或等于减数时,CF=0,否则CF=1；

5）受影响的寄存器：F。

```
19    ;多字节数据相减子程序MSUB
20    MSUB     PROC
21             PUSH     SI
22             PUSH     AX
23             PUSH     DI
24             CLC
25    MSUB1:   MOV      AL,[DI]
26             SBB      [SI],AL
27             INC      SI
28             INC      DI
29             DEC      AH
30             JNZ      MSUB1
31             POP      DI
32             POP      AX
33             POP      SI
34             RET
35    MCMP     ENDP
```

二. 多字节整数乘除运算

多字节数据左移一位子程序MSHL：

1）子程序名：MSHL；

2）子程序功能：多字节数据左移一位；

3）入口条件：数据低字节地址在SI,数据长度在CH中；

4）出口条件：数据低字节地址在SI,移位前数据最高位状态在CF ；

5）受影响的寄存器：F。

```
37    ;多字节数据左移一位子程序MSHL
38    MSHL     PROC
39             PUSH     SI
40             PUSH     CX
41             CLC
42    MSHL1:   RCL      BYTE  PTR  [SI],1
43             INC      SI
44             DEC      CH
45             JNZ      MSHL1
46             POP      CX
47             POP      SI
48             RET
49    MCMP     ENDP
```

二. 多字节整数乘除运算

```
51   DSEG    SEGMENT
52   DIVND   DB      5FH,0D4H,0E4H,0DCH,68H,1DH
53   DIVOR   DB      85H,27H,4AH
54   DIVM    DW      6
55   DIVN    DW      3
56   QUON    DW      0
57   QUO     DB      3  DUP (0)
58   REM     DB      3  DUP (0)
59   DSEG    ENDS
60   SSEG    SEGMENT STACK
61   STK     DB      20  DUP (0)
62   SSEG    ENDS
63   CSEG    SEGMENT
64           ASSUME  CS:CSEG,DS:DSEG
65           ASSUME  ES:DSEG,SS:SSEG
66   START:  MOV     AX,DSEG
67           MOV     DS,AX
68           MOV     ES,AX
69           MOV     AX,SSEG
70           MOV     SS,AX
71           MOV     SP,SIZE  STK
```

high 被除数

1DH	68H	DCH	E4H	D4H	5FH

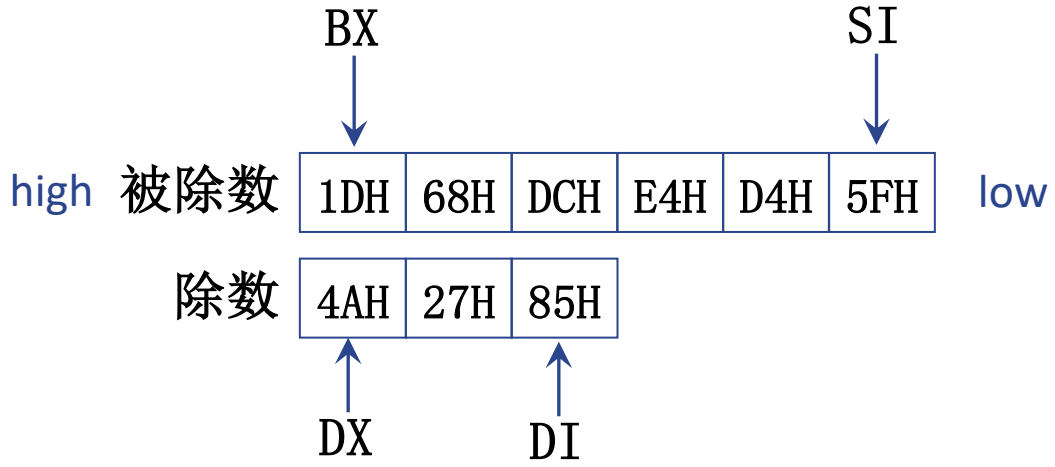
low

除数

4AH	27H	85H

二. 多字节整数乘除运算

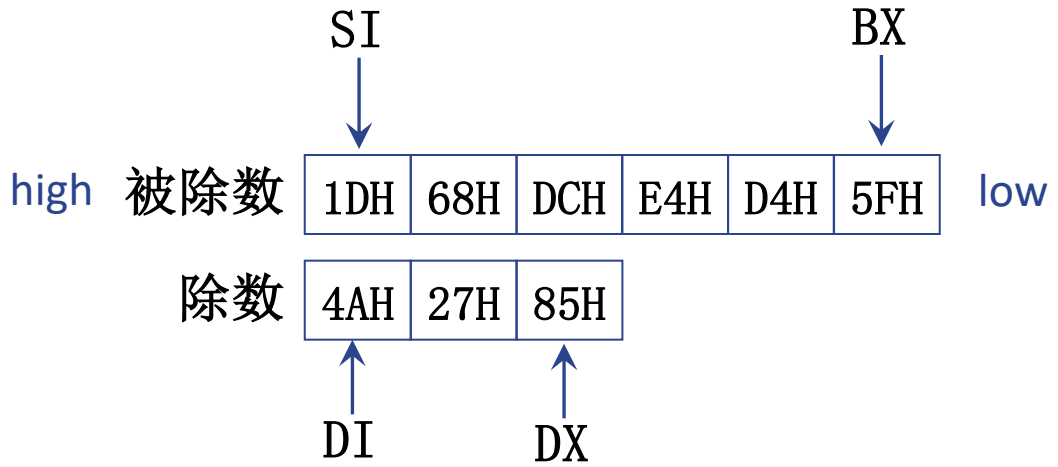
```
72          LEA     SI,DIVND
73          MOV     BX,SI
74          ADD     BX,DIVM
75          DEC     BX
76          LEA     DI,DIVOR
77          MOV     DX,DI
78          ADD     DX,DIVN
79          DEC     DX
80          MOV     CX,DIVM
81          SUB     CX,DIVN
82          MOV     QUON,CX
83          ADD     CX,CX
84          ADD     CX,CX
85          ADD     CX,CX
86          MOV     CH,BYTE  PTR  DIVM
87          MOV     AH,BYTE  PTR  DIVN
88          XCHG    SI,BX
89          XCHG    DI,DX
90          CALL    MCMP
91          XCHG    BX,SI
92          XCHG    DX,DI
93          JNC     OVER
94  MDIV1:  CALL    MSHL
```




```
51  DSEG    SEGMENT
52  DIVND   DB      5FH,0D4H,0E4H,0DCH,68H,1DH
53  DIVOR   DB      85H,27H,4AH
54  DIVM    DW      6
55  DIVN    DW      3
56  QUON    DW      0
```

二. 多字节整数乘除运算

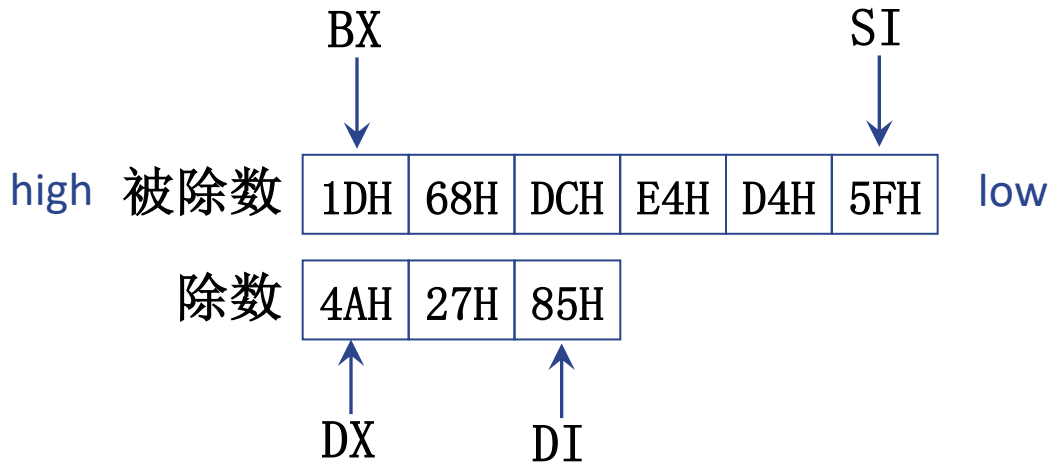
```
72          LEA     SI,DIVND
73          MOV     BX,SI
74          ADD     BX,DIVM
75          DEC     BX
76          LEA     DI,DIVOR
77          MOV     DX,DI
78          ADD     DX,DIVN
79          DEC     DX
80          MOV     CX,DIVM
81          SUB     CX,DIVN
82          MOV     QUON,CX
83          ADD     CX,CX
84          ADD     CX,CX
85          ADD     CX,CX
86          MOV     CH,BYTE  PTR  DIVM
87          MOV     AH,BYTE  PTR  DIVN
88          XCHG    SI,BX
89          XCHG    DI,DX
90          CALL    MCMP
91          XCHG    BX,SI
92          XCHG    DX,DI
93          JNC     OVER
94  MDIV1:  CALL    MSHL
```




```
51  DSEG    SEGMENT
52  DIVND   DB      5FH,0D4H,0E4H,0DCH,68H,1DH
53  DIVOR   DB      85H,27H,4AH
54  DIVM    DW      6
55  DIVN    DW      3
56  QUON    DW      0
```

二. 多字节整数乘除运算

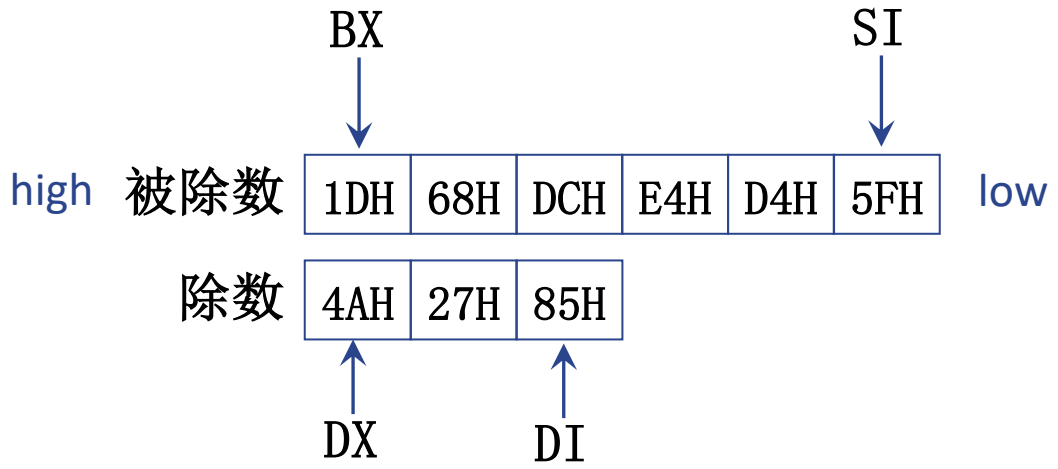
```
72          LEA     SI,DIVND
73          MOV     BX,SI
74          ADD     BX,DIVM
75          DEC     BX
76          LEA     DI,DIVOR
77          MOV     DX,DI
78          ADD     DX,DIVN
79          DEC     DX
80          MOV     CX,DIVM
81          SUB     CX,DIVN
82          MOV     QUON,CX
83          ADD     CX,CX
84          ADD     CX,CX
85          ADD     CX,CX
86          MOV     CH,BYTE  PTR  DIVM
87          MOV     AH,BYTE  PTR  DIVN
88          XCHG    SI,BX
89          XCHG    DI,DX
90          CALL    MCMP
91          XCHG    BX,SI
92          XCHG    DX,DI
93          JNC     OVER
94  MDIV1:  CALL    MSHL
```




```
51  DSEG    SEGMENT
52  DIVND   DB      5FH,0D4H,0E4H,0DCH,68H,1DH
53  DIVOR   DB      85H,27H,4AH
54  DIVM    DW      6
55  DIVN    DW      3
56  QUON    DW      0
```

二. 多字节整数乘除运算

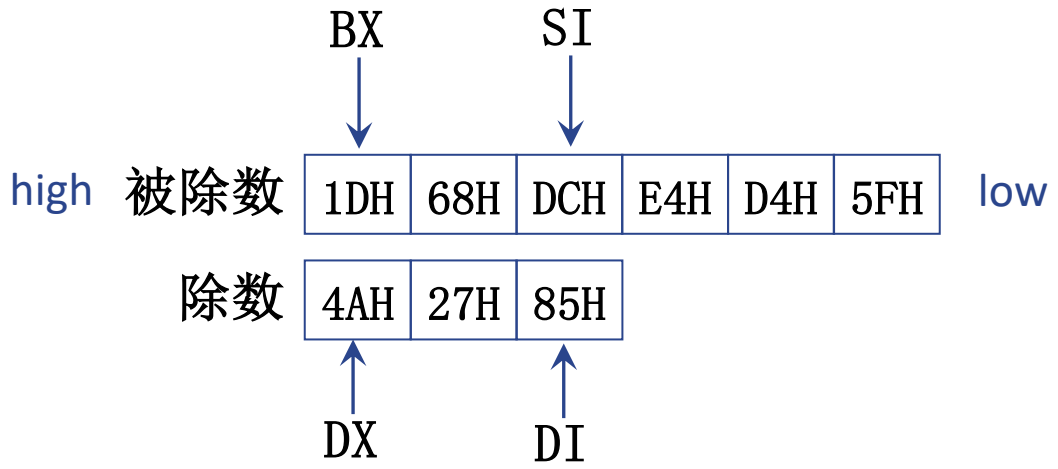
```
95              XCHG    BX,SI
96              XCHG    DX,DI
97              CALL    MCMP
98              XCHG    BX,SI
99              XCHG    DX,DI
100             JC      CHCNT
101             ADD     SI,QUON
102             CALL    MSUB
103             SUB     SI,QUON
104             INC     BYTE  PTR  [SI]
105     CHCNT:  DEC     CL
106             JNZ     MDIV1
107             MOV     CX,DIVM
108             LEA     DI,QUO
109             REP     MOVSB
110             JMP     EXIT
111     OVER:   MOV     AL,0FFH
112             LEA     DI,QUO
113             MOV     CX,DIVM
114             REP     STOSB
115     EXIT:   MOV     AH,4CH
116             INT     21H
117     CSEG    ENDS
118             END     START
```




```
51      DSEG    SEGMENT
52      DIVND   DB      5FH,0D4H,0E4H,0DCH,68H,1DH
53      DIVOR   DB      85H,27H,4AH
54      DIVM    DW      6
55      DIVN    DW      3
56      QUON    DW      0
```

二. 多字节整数乘除运算

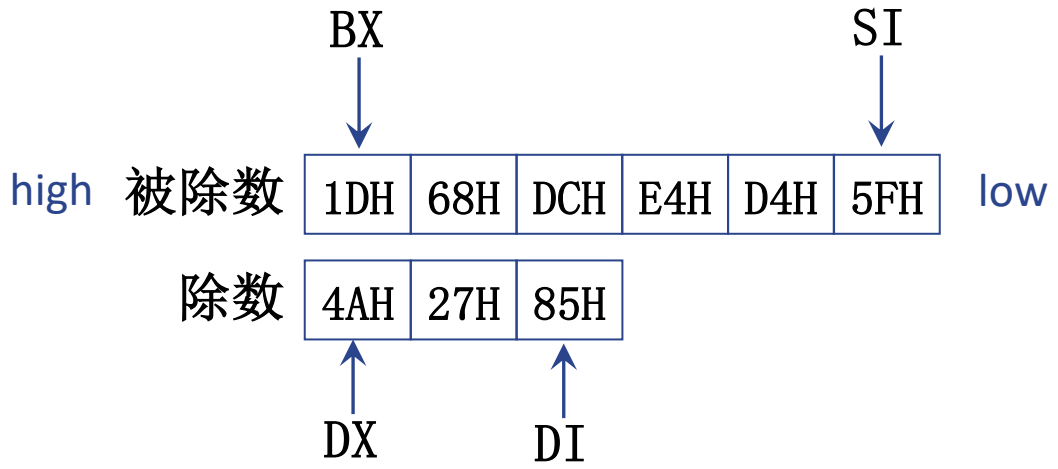
```
95              XCHG    BX,SI
96              XCHG    DX,DI
97              CALL    MCMP
98              XCHG    BX,SI
99              XCHG    DX,DI
100             JC      CHCNT
101             ADD     SI,QUON
102             CALL    MSUB
103             SUB     SI,QUON
104             INC     BYTE  PTR  [SI]
105     CHCNT:  DEC     CL
106             JNZ     MDIV1
107             MOV     CX,DIVM
108             LEA     DI,QUO
109             REP     MOVSB
110             JMP     EXIT
111     OVER:   MOV     AL,0FFH
112             LEA     DI,QUO
113             MOV     CX,DIVM
114             REP     STOSB
115     EXIT:   MOV     AH,4CH
116             INT     21H
117     CSEG    ENDS
118             END     START
```




```
51      DSEG    SEGMENT
52      DIVND   DB      5FH,0D4H,0E4H,0DCH,68H,1DH
53      DIVOR   DB      85H,27H,4AH
54      DIVM    DW      6
55      DIVN    DW      3
56      QUON    DW      0
```

二. 多字节整数乘除运算

```
95              XCHG    BX,SI
96              XCHG    DX,DI
97              CALL    MCMP
98              XCHG    BX,SI
99              XCHG    DX,DI
100             JC      CHCNT
101             ADD     SI,QUON
102             CALL    MSUB
103             SUB     SI,QUON
104             INC     BYTE  PTR  [SI]
105     CHCNT:  DEC     CL
106             JNZ     MDIV1
107             MOV     CX,DIVM
108             LEA     DI,QUO
109             REP     MOVSB
110             JMP     EXIT
111     OVER:   MOV     AL,0FFH
112             LEA     DI,QUO
113             MOV     CX,DIVM
114             REP     STOSB
115     EXIT:   MOV     AH,4CH
116             INT     21H
117     CSEG    ENDS
118             END     START
```




```
51      DSEG    SEGMENT
52      DIVND   DB      5FH,0D4H,0E4H,0DCH,68H,1DH
53      DIVOR   DB      85H,27H,4AH
54      DIVM    DW      6
55      DIVN    DW      3
56      QUON    DW      0
```

一. BCD码运算

1. BCD码的运算规则：

用一组二进制数表示一位十进制数的编码方法,称为二进制编码的十进制数,简称BCD码。

注意：BCD码只使用二进制数中的十种状态，其余六种未使用为非法状态。

十进制数	BCD码	十进制数	BCD码
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

一. BCD码运算

1. BCD码的运算规则

例：BCD码的运算

```
   11+15=26
   00010001
+  00010101
   00100110     26
   结果正确
```

```
   加6调整
   00101100
+  00000110
   00110010     32
   结果正确
```

```
   17+15=32
   00010111
+  00010101
   00101100     2C
出现非法数字，结果不正确
```

小结：

两个BCD码相加，结果出现非法数字时，需要加6调整。

一. BCD码运算

1. BCD码的运算规则

例：BCD码的运算

```
    17+19=36
    00010111
+   00011001
    00110000      30
```

未出现非法数字，结果不正确

```
    加6调整
    00110000
+   00000110
    00110110      36
    结果正确
```

小结：

两个BCD码相加，出现进位时，需要加6调整。

结论：

两个BCD码相加，结果出现非法数字或出现进位时，需要加6调整。

同样：两个BCD码相减，结果出现非法数字或出现借位时，需要减6调整。

一. BCD码运算

2. 组合BCD码与非组合BCD码

- 组合BCD：在一个字节中存放二位BCD码。例如：54表示为：01010100。
- 非组合BCD：在一个字节中存放一位BCD码。例如：54表示为：00000101 00000100。

一. BCD码运算

2. 组合BCD码与非组合BCD码

- 组合BCD码的调整指令

1) DAA　十进制加法调整

指令汇编格式：DAA

操作：如果 AL ∧ 0FH＞9 或 AF=1

则AL←AL+6；AF←1

如果 AL ∧ 0F0H＞90H 或 CF=1

则AL←AL+60H；CF←1

受影响的标志位： OF,SF,ZF,AF,PF,CF

举例：

```
MOV   AL, 45H          ;十进制45
ADD   AL, 47H          ;与47相加出现非法数字
DAA                    ;调整，得到正确结果
```

一. BCD码运算

2. 组合BCD码与非组合BCD码

- 组合BCD码的调整指令

2）DAS　十进制减法调整

指令汇编格式：DAS

操作：如果 AL ∧ 0FH＞9 或 AF=1

则AL←AL−6；AF←1

如果 AL ∧ 0F0H＞90H 或 CF=1

则AL←AL−60H；CF←1

受影响的标志位： OF,SF,ZF,AF,PF,CF

举例：

```
MOV   AL, 45H          ;十进制45
SUB   AL, 17H          ;与17相减出现非法数字
DAS                    ;调整，得到正确结果
```

一. BCD码运算

2. 组合BCD码与非组合BCD码

- 非组合BCD码的调整指令

1) AAA　ASCII码加法调整

指令汇编格式：AAA

操作：如果 AL ∧ 0FH＞9 或 AF=1

则AL←AL+6；AH←AH+1；AF←1 ；CF←1

则AL←AL ∧ 0FH

受影响的标志位： OF,SF,ZF,AF,PF,CF

举例：

```
MOV   AL, 05          ;十进制5
ADD   AL, 07          ;与7相加出现非法数字
AAA                   ;调整，得到正确结果
```

一. BCD码运算

2. 组合BCD码与非组合BCD码

- 非组合BCD码的调整指令

2）AAS　ASCII码减法调整

指令汇编格式：AAS

操作：如果 AL ∧ 0FH＞9 或 AF=1

则AL←AL−6；AH←AH−1；AF←1 ；CF←1

则AL←AL ∧ 0FH

受影响的标志位： OF,SF,ZF,AF,PF,CF

举例：

```
MOV   AL, 0105H        ;十进制15
SUB   AL, 07           ;与7相减出现非法数字
AAS                    ;调整，得到正确结果
```

一. BCD码运算

2. 组合BCD码与非组合BCD码

- 非组合BCD码的调整指令

3）AAM　ASCII码乘法调整

指令汇编格式：AAS

操作：将AL中的内容（乘法运算后）除以10，商送入AH中，余数送入AL中

受影响的标志位： OF,SF,ZF,AF,PF,CF

说明：将两个有效的非组合十进制数相乘后得到的结果调整为一个有效的非组合十进制数

举例：

```
MOV   AX, 0905        ;十进制9和5
MUL   AH              ;9*5，结果AX=002DH
AAM                   ;调整，得到正确结果
```

一. BCD码运算

2. 组合BCD码与非组合BCD码

- 非组合BCD码的调整指令

4）AAD　ASCII码除法调整

指令汇编格式：AAD

操作：将AH中的内容乘以10后与AL相加，结果送入AL中，然后将AH清零

受影响的标志位： OF,SF,ZF,AF,PF,CF

说明：AAD指令在两个有效的非组合十进制数做除法之前调整AL中的内容，以使除法得到的商为一个有效的非组合十进制数，除法执行后商在AL中，余数在AH中。

举例：

```
MOV   AX, 0405        ;十进制45
MOV   BL, 08
AAD                   ;将AX内容调整为2DH
DIV   BL              ;相除，得到正确结果
```

二. 浮点数的加减法

浮点数据的格式：

1.fff…fff

尾数23位, 阶码8位,过余量127(7FH)

加减运算时，先对阶(使阶码相同)，然后，再对尾数进行加减运算即可。运算完成后，将结果规格化。

二. 浮点数的加减法

浮点数据加法的计算步骤：

1) 被加数为0? 是,加数为运算结果,送结果寄存器,转16;

2) 加数为0? 是,被加数为运算结果,送结果寄存器,转16;

3) 两数都不为0,求两数阶差;

4) 阶差为0,则阶码相等,恢复小数高位1,转10;

5) 阶差的绝对值大于24吗?

6) 大于24,则阶码大的数据为结果,送结果寄存器,转16;

7) 小于24,则先保存大数符号作为结果符号;

8) 恢复小数高位1;

9) 对阶;

10) 恢复大数高位1;

11) 两数同号? 异号转13;

12) 同号,两数绝对值相加,转14;

13) 异号,用绝对值大的数减去绝对值小的数;

14) 规格化浮点数;

15) 为结果配置符号位;

16) 返回主程序。

二. 浮点数的加减法

- 浮点数据的另一种表示方法：

二. 浮点数的加减法

浮点数的加法子程序：

1) 子程序名: FADDI

2) 子程序功能: 两个浮点数相加;

3) 入口条件: 被加数在 AX,BX中,加数在 CX,DX中;

4) 出口条件: 运算结果在 AX,BX中;

5) 受影响的寄存器: AX,BX 和标志寄存器 F。

二. 浮点数的加减法

浮点数的加法子程序：

```
1:;*****FLOAT ADD*****
2:FADDI         PROC
3:              PUSH    CX          ;保存加数
4:              PUSH    DX
5:              PUSH    AX          ;保存被加数阶码和尾数高字节
6:              OR      AX,BX       ;测试被加数是否为0
7:              POP     AX          ;恢复被加数阶码和尾数的高字节
8:              JZ      FADD10      ;被加数为0,加数为结果
9:              PUSH    CX          ;保存加数阶码和尾数的高字节
10:             OR      CX,DX       ;测试加数是否为0
11:             POP     CX          ;恢复加数阶码和尾数的高字节
12:             JZ      FADD08      ;加数为0,被加数为结果
13:             SHL     AX,1        ;将被加数的阶码与尾数分开
14:             RCR     AL,1
15:             SHL     CX,1        ;将加数的阶码与尾数分开
16:             RCR     CL,1
17:             CMP     AH,CH       ;比较两数的阶码
18:             JZ      FADD11      ;两数阶码相同转
19:             JNC     FADD01      ;被加数阶码大转
20:             XCHG    AX,CX       ;被加数阶码小,则将被加数与加数互换
21:             XCHG    BX,DX
```

二. 浮点数的加减法

浮点数的加法子程序：

```
22:FADD01:    SUB     AH,CH     ;大数阶码减小数阶码,求得阶差
23:           CMP     AH,24     ;阶差大小24?
24:           JC      FADD02    ;小于24,对阶
25:           ADD     AH,CH     ;大于等于24,恢复大数阶码
26:           JMP     FADD15    ;AX,BX的内容(大数)为结果
27:FADD02:    AND     AL,AL     ;设置符号标志
28:           PUSHF             ;大数的符号压入堆栈,作为结果的符号
29:           XOR     AL,CL     ;确认两数同号还是异号
30:           PUSHF             ;结果压栈
31:           XOR     AL,CL     ;恢复AL的内容
32:           OR      CL,80H    ;恢复小数的整数1
33:FADD03:    SHR     CL,1      ;尾数右移一位
34:           RCR     DX,1
35:           INC     CH        ;阶码加1
36:           DEC     AH        ;对阶结束?
37:           JNZ     FADD03    ;未结束,继续
38:           MOV     AH,CH     ;和的阶码送AH
39:           OR      AL,80H    ;恢复大数的整数1
40:           POPF              ;取数两数同异号标志
41:           JS      FADD09    ;两数异号转
42:FADD04:    ADD     BX,DX     ;两数同号,尾数相加
```

二. 浮点数的加减法

浮点数的加法子程序：

```
43:                ADC     AL,CL      ;
44:                JNC     FADD05     ;相加之后,无进位转
45:                RCR     AL,1       ;有进位,将进位移入尾数最高位
46:                RCR     BX,1
47:                INC     AH         ;阶码加1
48:FADD05:         TEST    AL,80H     ;测试是否已是规格化数
49:                JNZ     FADD06     ;已是,转
50:                SHL     BX,1       ;进行规格化
51:                RCL     AL,1       ;尾数左移一位
52:                DEC     AH         ;阶码减1
53:                JMP     FADD05     ;
54:FADD06:         AND     AL,7FH     ;将结果调整成原来格式
55:                SHR     AH,1       ;
56:                JNC     FADD07
57:                OR      AL,80H
58:FADD07:         POPF               ;取出结果符号
59:                JNS     FADD08     ;为运行结果配符号位
60:                OR      AH,80H
61:FADD08:         CLC
62:                POP     DX         ;恢复加数
63:                POP     CX
64:                RET
65:        ;
```

二. 浮点数的加减法

浮点数的加法子程序：

```
66:FADD09:    SUB     BX,DX     ;两数异号,大数减小数
67:           SBB     AL,CL
68:           JMP     FADD05    ;转规格化
69:;
70:FADD10:    XCHG    AX,CX     ;被加数为0,加数为结果
71:           XCHG    BX,DX
72:           JMP     FADD08
73:;
74:FADD11:    AND     AL,AL     ;阶码相同,保存被加数符号
75:           PUSHF
76:           XOR     AL,CL     ;测试两数是否同号
77:           PUSHF             ;测试结果压栈
78:           XOR     AL,CL     ;恢复AL的内容
79:           POPF              ;取出测试结果
80:           JS      FADD12    ;两数异号,转
81:           OR      AL,80H    ;两数同号,恢复被加数的整数1
82:           OR      CL,80H    ;恢复加数的整数1
83:           JMP     FADD04    ;转两数相加
84:FADD12:    AND     CL,CL     ;两数异号,求加数符号
85:           PUSHF             ;加数符号压栈
86:           OR      CL,80H    ;恢复加数的整数1
```

二. 浮点数的加减法

浮点数的加法子程序：

```
87:             OR      AL,80H   ;恢复被加数的整数1
88:             CMP     AL,CL    ;两数绝对值进行比较
89:             JNZ     FADD13   ;尾数高字节不等,转
90:             CMP     BX,DX    ;比较尾数中低字节
91:             JNZ     FADD13   ;中低字节不同,转
92:             ADD     SP,4     ;两数相同符号相反,恢复栈指针
93:             XOR     AX,AX    ;结果为0
94:             XOR     BX,BX
95:             POP     DX       ;恢复原加数
96:             POP     CX
97:             RET              ;返回
98:;
99:FADD13:      JNC     FADD14   ;被加数大,转
100:            XCHG    AX,CX    ;被加数与加数交换,大数送AX,BX
101:            XCHG    BX,DX
102:            ADD     SP,4     ;恢复栈指针
103:            AND     AL,AL    ;测试加数符号
104:            PUSHF            ;把加数符号作为结果符号
105:            JMP     FADD09   ;转两数相减
```

二. 浮点数的加减法

浮点数的加法子程序：

```
106:FADD14:    POPF              ;修改栈指针
107:           JMP     FADD09    ;
108:FADD15:    SHL     AL,1      ;两数阶差大于等于24时,大数为结果
109:           RCR     AX,1      ;结果调整原格式
110:           STC
111:           POP     DX        ;恢复原加数
112:           POP     CX
113:           RET               ;退出
114:FADDI      ENDP
```

三. 十进制数的ASCII码串转换为二进制定点数

- 从键盘上输入一个数据,例如365,计算机内部得到的是组成该数据的各个数字的ASCII码：33H,36H,35H。在对数据处理之前,必须将其ASCII码串转换成二进制定点数。

首先将其转换为 03H,06H,05H

然后计算：03H*100+06H*10+05H*1

或者使用：$N=N*10+N_i$

其中N_i为一位十进制数的ASCII码代表的数值，N为转换的结果值，其初值为0。

三. 十进制数的ASCII码串转换为二进制定点数

从键盘上输入一个数据,例如365,计算机内部得到的是组成该数据的各个数字的ASCII码：33H,36H,35H。在对数据处理之前,必须将其ASCII码串转换成二进制定点数。

算法：

1) 初始化N,N←0；
2) 取一位字符的ASCII码；
3) 判断是否为十进制数字的ASCII码？不是,转（6）；
4) 是,将其转换为对应的数字N_i；
5) 计算$N=N*10+N_i$，转（2）；
6) 出口。

按以上算法编制一个子程序：
（1）子程序名：DATBIN
（2）功能：十进制数的ASCII码串转换为二进制数（小于65535）
（3）入口条件：ASCII码串首址在SI中；且以非十进制数字结束
（4）出口条件：CX中为转换结果值；AL中为字符串终止字符；SI指向终止字符
（5）受影响的寄存器：AX,CX,SI,F

三. 十进制数的ASCII码串转换为二进制定点数

从键盘上输入一个数据,例如365,计算机内部得到的是组成该数据的各个数字的ASCII码：33H,36H,35H。在对数据处理之前,必须将其ASCII码串转换成二进制定点数。

算法：

1) 初始化N,N←0；
2) 取一位字符的ASCII码；
3) 判断是否为十进制数字的ASCII码？不是,转（6）；
4) 是,将其转换为对应的数字N_i；
5) 计算$N=N*10+N_i$，转（2）；
6) 出口。

```
43  DATBIN  PROC
44          PUSH    BX
45          XOR     CX,CX
46  GETA:   MOV     AL,[SI]
47          CMP     AL,'0'
48          JB      RETURN
49          CMP     AL,'9'
50          JA      RETURN
51          SUB     AL,'0'
52          XOR     AH,AH
53          MOV     BX,CX
54          SHL     CX,1
55          SHL     CX,1
56          ADD     CX,BX
57          SHL     CX,1
58          ADD     CX,AX
59          INC     SI
60          JMP     GETA
61  RETURN: POP     BX
62          RET
63  DATBIN  ENDP
```

三. 十进制数的ASCII码串转换为二进制定点数

例子：将从键盘上接收的十进制的ASCII数据转换为二进制，并存入BIN单元。

```
66  DSEG    SEGMENT
67  BUFF    DB      10H DUP (0)
68  BIN     DW      ?
69  DSEG    ENDS
70  CSEG    SEGMENT
71          ASSUME  CS:CSEG,DS:DSEG
72  DTB:    MOV     AX,DSEG
73          MOV     DS,AX
74          MOV     ES,AX
75          LEA     DI,BUFF
76  AG:     MOV     AH,1
77          INT     21H
78          STOSB
79          CMP     AL,0DH
80          JNE     AG
81          CALL    DATBIN
82          MOV     BIN,CX
83          MOV     AH,4CH
84          INT     21H
85  CSEG    ENDS
86  END     DTB
```

```
43  DATBIN  PROC
44          PUSH    BX
45          XOR     CX,CX
46  GETA:   MOV     AL,[SI]
47          CMP     AL,'0'
48          JB      RETURN
49          CMP     AL,'9'
50          JA      RETURN
51          SUB     AL,'0'
52          XOR     AH,AH
53          MOV     BX,CX
54          SHL     CX,1
55          SHL     CX,1
56          ADD     CX,BX
57          SHL     CX,1
58          ADD     CX,AX
59          INC     SI
60          JMP     GETA
61  RETURN: POP     BX
62          RET
63  DATBIN  ENDP
```

四. 二进制定点数转换为十进制数的ASCII码串

- 要把运算的结果以十进制形式在显示器上输出,则需要将其转换成十进制数的ASCII码串,然后逐个字符的输出。

设X为要转换的16位的二进制数据(对应的十进制的范围为0~65535), 可采用如下方法：

X/10000 → { N （商）　N<6, 万位
　　R （余数）　R → X

X/1000 → { N （商）　N<10, 千位
　　R （余数）　R → X

X/100 → { N （商）　N<10, 百位
　　R （余数）　R → X

X/10 → { N （商）　N<10, 十位
　　R （余数）　R<10, 个位

四. 二进制定点数转换为十进制数的ASCII码串

算法：

(1) 设置常数表首址, 常数依次为10000,1000,100,10；
(2) 取一常数C_i ；
(3) 将二进制数X扩展为双字数据N；
(4) 计算N/C_i ,商为N_i ,余数为R；
(5) 将N_i转换为对应数字的ASCII码,并保存；
(6) 余数R作为N；
(7) 修改常数表地址指针和存放ASCII码值的地址指针；
(8) C_i =10？不是,转（2）；
(9) 是,将余数R作为个位数值,转换成ASCII码,保存起来；
(10) 结束。

按上述算法编制一个子程序,说明文件如下：

（1）子程序名：BTODA
（2）子程序功能：AX中的二进制整数（无符号）转换为十进制数并显示；
（3）入口条件：要转换的二进制数在AX中；ASCII码串存放首址在ES:DI中；
（4）出口条件：无
（5）受影响的寄存器：AX,F

四. 二进制定点数转换为十进制数的ASCII码串

算法：

```
BTODA   PROC
        PUSH    DS
        PUSH    CS
        POP     DS
        PUSH    DI
        PUSH    CX
        PUSH    BX
        PUSH    DX
        CLD
        LEA     BX,CTAB
BTDA1:  MOV     CX,[BX]
        XOR     DX,DX
        DIV     CX
        PUSH    DX
        ADD     AL,30H
        MOV     AH,2
        MOV     DL,AL
        INT     21H
        POP     DX
        MOV     AX,DX
        ADD     BX,2
        CMP     CX,10
        JNZ     BTDA1
        ADD     AL,30H
        MOV     DL,AL
        MOV     AH,2
        INT     21H
        POP     DX
        POP     BX
        POP     CX
        POP     DI
        POP     DS
        RET
CTAB    DW      10000,1000,100,10
BTODA   ENDP
```

四. 二进制定点数转换为十进制数的ASCII码串

例题：利用前面的DATBIN子程序和BTODA子程序，编写实现简单计算器功能（整数加、减、乘、除）的程序。

实现步骤如下：

1. 循环调用INT 21H的1号功能，从键盘读取第一个数据，直到遇到非十进制数字字符为止；
2. 调用DATBIN 完成转换；
3. 循环调用INT 21H的1号功能，从键盘读取第二个数据，直到遇到非十进制数字字符为止；
4. 调用DATBIN 完成转换；
5. 根据两个数据之间的按键（+–*/)完成相应的运算；
6. 调用BTODA完成转换并显示。

四. 二进制定点数转换为十进制数的ASCII码串

```
PSEG    SEGMENT
BUFF    DB      10 DUP (0)
OP      DB      ?
        ASSUME  CS:PSEG,DS:PSEG
MAIN:   MOV     AX,CS
        MOV     DS,AX
        MOV     ES,AX
        CLD
AG0:    LEA     DI,BUFF
AG1:    MOV     AH,1
        INT     21H
        STOSB
        CMP     AL,'0'
        JB      OK1
        CMP     AL,'9'
        JA      OK1
        JMP     AG1
OK1:    LEA     SI,BUFF
        CALL    DATBIN
        PUSH    CX
        MOV     OP,AL
        LEA     DI,BUFF
AG2:    MOV     AH,1
        INT     21H
        STOSB
        CMP     AL,'0‘
        JB      OK2
        CMP     AL,'9'
        JA      OK2
        JMP     AG2
OK2:    LEA     SI,BUFF
        CALL    DATBIN
        POP     AX
        CMP     OP,'+'
        JE      ADDI
        CMP     OP,'–'
        JE      SUBT
        CMP     OP,'*'
        JE      MUTI
        XOR     DX,DX
        DIV     CX
        JMP     DISP
ADDI:   ADD     AX,CX
        JMP     DISP
SUBT:   SUB     AX,CX
        JMP     DISP
MUTI:   MUL     CX
DISP:   CALL    BTODA
        MOV     AH,2
        MOV     DL,0DH
        INT     21H
        MOV     DL,0AH
        INT     21H
        JMP     AG0
PSEG    ENDS
        END     MAIN
```