计算机原理第二次实验报告

张蔚桐 2015011493 自55

2017 年 4 月 20 日

1 实验目的

1. 巩固DEBUG和宏汇编的使用
2. 加深对运算指令的理解
3. 注意标志寄存器的变化

2 实验内容

2.1 16位二进制数的加减

在数据区内定义两个16位二进制数，用8位运算指令进行加减运算，首先程序将两个数读入到AX和DX中，之后进行AL和DL的相加，不考虑进位，之后对AH和DH进行相加，考虑进位，得到运算结果

程序如下所示，运行结果如图1所示，我们可以验证，FFFE+FFFF=FFFD，FFFE-FFFF=FFFF结果是正确的。

```
01 DATA SEGMENT
02 X DW 0EEFFH ;THE VALUE 0F X
03 Y DW 0FFEEH  ;THE VALUE OF Y
04 P DW 0FFFFH ;X+Y
05 M DW 0FFFFH ;X-Y
06 ;Z=X+Y
07 DATA ENDS
08
09 STACK SEGMENT PARA STACK
10 DB 10 DUP (?)
11 STACK ENDS
12
```

```
13 CODE SEGMENT
14 ASSUME DS:DATA,ES:DATA,SS:STACK,CS:CODE
15 START:
16 MOV AX,DATA
17 MOV DS,AX
18 MOV ES,AX
19 MOV AX,STACK
20 MOV SS,AX
21
22 ;DATA READ IN
23 MOV AX,X
24 MOV DX,Y
25 ;ADD METHOD
26 ADD AL,DL
27 ADC AH,DH
28 MOV P,AX
29 ;SUB METHOD
30 MOV AX,X
31 SUB AL,DL
32 SBB AH,DH
33 MOV M,AX
34
35 MOV AH,04CH
36 INT 021H
37 CODE ENDS
38 END START
```

```
C:\>debug 2_1.exe
-u
0772:0000 B86A07        MOV     AX,076A
0772:0003 8ED8          MOV     DS,AX
-e 076A:0000
076A:0000  34.FE   12.FF   67.FF   05.FF

-g

Program terminated normally
-d 076A:0000
076A:0000  FE FF FF FF FD FF FF FF-00 00 00 00 00 00 00 00   ................
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

图 1: 两个16位二进制数加减

2.2 压缩BCD码的加减

在数据区定义两个压缩BCD码，程序首先将这两个数据读入到AX，DX中，之后对AL和DL进行ADD（不考虑进位）操作，之后使用DAA指令对AL和DL上的数据进行压缩BCD码调整，因为DAA指令只能对AL寄存器进行调整，因此将AL和AH寄存器数值进行交换XCHG，将AL和DH进行ADC操作（考虑进位），之后再次对AL上的值进行压缩BCD码调整，使用XCHG命令回到原始顺序下，输出结果

对于减法操作基本相同，只是用SUB命令代替ADD命令，用SBB命令代替ADC命令，而采用DAS命令代替DAA命令

程序如下所示，运行结果如图2所示，我们可以验证，98+99=99，98-99=99结果是正确的。

```
01
02
03 DATA SEGMENT
04 X DW 00001H ;THE VALUE 0F X
05 Y DW 09999H  ;THE VALUE OF Y
06 P DW 0FFFFH ;X+Y
07 M DW 0FFFFH ;X-Y
08 ;Z=X+Y
09 DATA ENDS
10
11 STACK SEGMENT PARA STACK
12 DB 10 DUP (?)
13 STACK ENDS
14
15 CODE SEGMENT
16 ASSUME DS:DATA,ES:DATA,SS:STACK,CS:CODE
17 START:
18 MOV AX,DATA
19 MOV DS,AX
20 MOV ES,AX
21 MOV AX,STACK
22 MOV SS,AX
23
24 ;DATA READ IN
25 MOV AX,X
```

```
26 MOV DX,Y
27 ;ADD METHOD
28 ADD AL,DL
29 DAA
30 XCHG AL,AH
31 ADC AL,DH
32 DAA
33 XCHG AL,AH
34 MOV P,AX
35 ;SUB METHOD
36 MOV AX,X
37 SUB AL,DL
38 DAS
39 XCHG AL,AH
40 SBB AL,DH
41 DAS
42 XCHG AL,AH
43 MOV M,AX
44
45 MOV AH,04CH
46 INT 021H
47 CODE ENDS
48 END START
```

```
C:\>debug 2_2.exe
-u
0772:0000 B86A07        MOV     AX,076A
0772:0003 8ED8          MOV     DS,AX
-e 076A:0000
076A:0000  34.98   12.99   78.99   56.99

-g

Program terminated normally
-d 076A:0000
076A:0000  98 99 99 99 97 99 99 99-00 00 00 00 00 00 00 00   ................
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

图 2: 两个压缩BCD码的加减

2.3 压缩BCD码的乘法

程序执行的策略是不断累加代替乘法，首先程序完成对数据的读入，存入DL和CL中，AX置零，其次程序进行循环，以下是循环的过程。

在一次循环中，程序将DL加入到AL中（ADD，不考虑进位），其次完成AL的BCD码调整DAA。之后，程序处理进位问题，将AH和0相加（ADC）处理进位，之后调换AL，AH的值，对AL（原AH）进行压缩BCD码调整，在将其换回原来位置。之后程序递减CL，并将其与AL交换进行压缩BCD码调整（DAS），再将其换回原位置

当CL最终被减至0的时候完成程序退出循环，同时完成数据的存储操作

程序如下所示，运行结果如图3所示，我们可以验证，77*7=539结果是正确的。

```
01
02
03 DATA SEGMENT
04 X DB 038H
05 Y DB 023H
06 XY DW 0FFFFH;X*Y
07 DATA ENDS
08
09 STACK SEGMENT PARA STACK
10 DB 10 DUP (?)
11 STACK ENDS
12
13 CODE SEGMENT
14 ASSUME DS:DATA,ES:DATA,SS:STACK,CS:CODE
15 START:
16 MOV AX,DATA
17 MOV DS,AX
18 MOV ES,AX
19 MOV AX,STACK
20 MOV SS,AX
21
22 ;DATA READ IN
23 MOV DL,X
24 MOV DH,000H
```

```
25 MOV CL,Y
26 MOV AX,00000H
27 L1:
28 ADD AL,DL
29 DAA
30 XCHG AL,AH
31 ADC AL,0
32 DAA
33 XCHG AH,AL
34 DEC CL
35 XCHG AL,CL
36 DAS
37 XCHG AL,CL
38 JNZ L1
39 MOV XY,AX
40
41 MOV AH,04CH
42 INT 021H
43 CODE ENDS
44 END START
```

```
C:\>debug 2_3.exe
-u
0772:0000 B86A07        MOV     AX,076A
0772:0003 8ED8          MOV     DS,AX
-e 076A:0000
076A:0000  12.77   34.07

-g

Program terminated normally
-d 076A:0000
076A:0000  77 07 39 05 00 00 00 00-00 00 00 00 00 00 00 00   w.9.............
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

图 3: 两个压缩BCD码的乘法

2.4 ASCII码的乘法

程序执行的策略是将一个数的每一位分别和另外一个数相乘处理。

程序首先将个位数Y读入CH中，并将CL赋值为X的位数4，之后将DI指向X的首地址（最低位），将SI指向XY的首地址（最低位）开始进行四次循环，DH置零，代表低位的进位。

每次循环过程中，首先程序将X的当前位读入到AL中，并和00FH做与处理从ASCII码得到数字，之后和CH进行乘法运算，对AX进行非压缩BCD码调整处理AAM,将低位AL和上一次的进位DH相加，进行非压缩BCD码调整AAA命令。在之后将DH重新幅值为AH，即为本位向上的进位信息，将AL加030H转换为ASCII码，存入SI所指向的位置，两个指针分别递增，CL递减完成一次循环

最后循环执行完成后即完成了乘法操作，如果此时DH仍存在进位则视为溢出处理。

程序如下所示，运行结果如图4所示，我们可以验证，2754*3=8262结果是正确的。

```
01
02
03 DATA SEGMENT
04 X DB '4572'
05 Y DB '3'
06 XY DW 0FFFFH;X*Y
07 DATA ENDS
08
09 STACK SEGMENT PARA STACK
10 DB 10 DUP (?)
11 STACK ENDS
12
13 CODE SEGMENT
14 ASSUME DS:DATA,ES:DATA,SS:STACK,CS:CODE
15 START:
16 MOV AX,DATA
17 MOV DS,AX
18 MOV ES,AX
19 MOV AX,STACK
20 MOV SS,AX
21
```

```
;DATA READ IN
MOV CH,Y
AND CH,00FH
MOV CL,4
MOV DH,0
LEA DI,X
LEA SI,XY
L1:
MOV AL,[DI]
AND AL,00FH
MUL CH
AAM
ADD AL,DH
AAA
MOV DH,AH
ADD AL,030H
MOV [SI],AL
INC SI
INC DI
DEC CL
JNZ L1

MOV AH,04CH
INT 021H
CODE ENDS
END START
```

```
C:\>debug 2_4.exe
-u
0772:0000 B86A07        MOV     AX,076A
0772:0003 8ED8          MOV     DS,AX
-g

Program terminated normally
-d 076A:0
076A:0000  34 35 37 32 33 32 36 32-38 30 00 00 00 00 00 00   4572326280......
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

图 4: 两个压缩BCD码的乘法