

中山大學



微机实验

题 目：	作业2
上课时间：	第14-15周
授课教师：	何涛
姓 名：	周德峰
学 号：	21312210
日 期：	2023-5-28

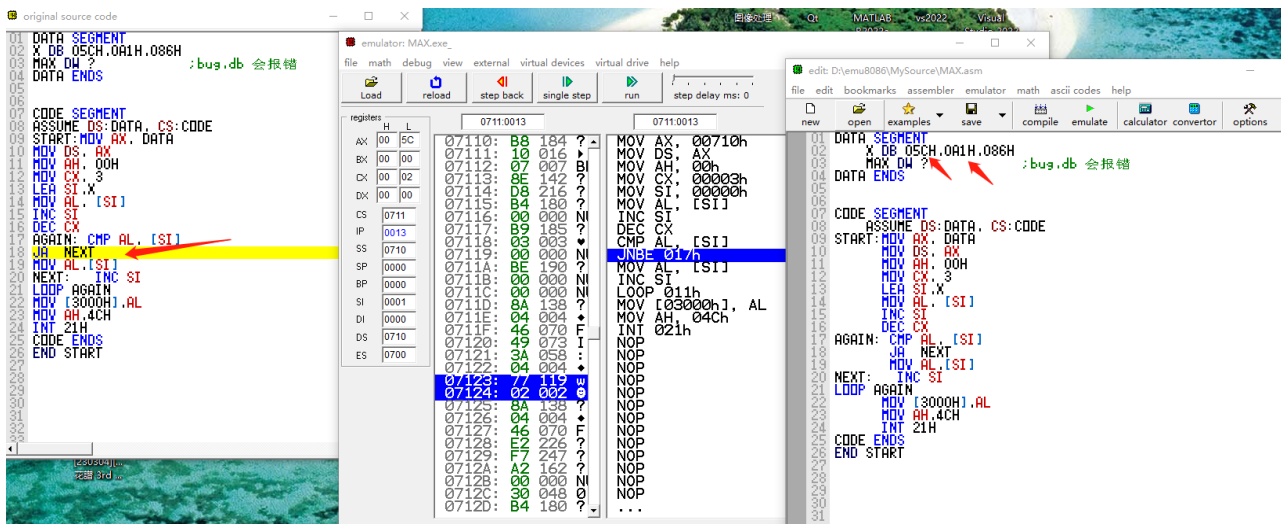
问题1

已知数据段中初始存放了 3 个字节的无符号数，编写程序找出当 中最大的数，将结果置于 3000H 的地址单元中。(样例: 05CH, 0A1H, 86H)

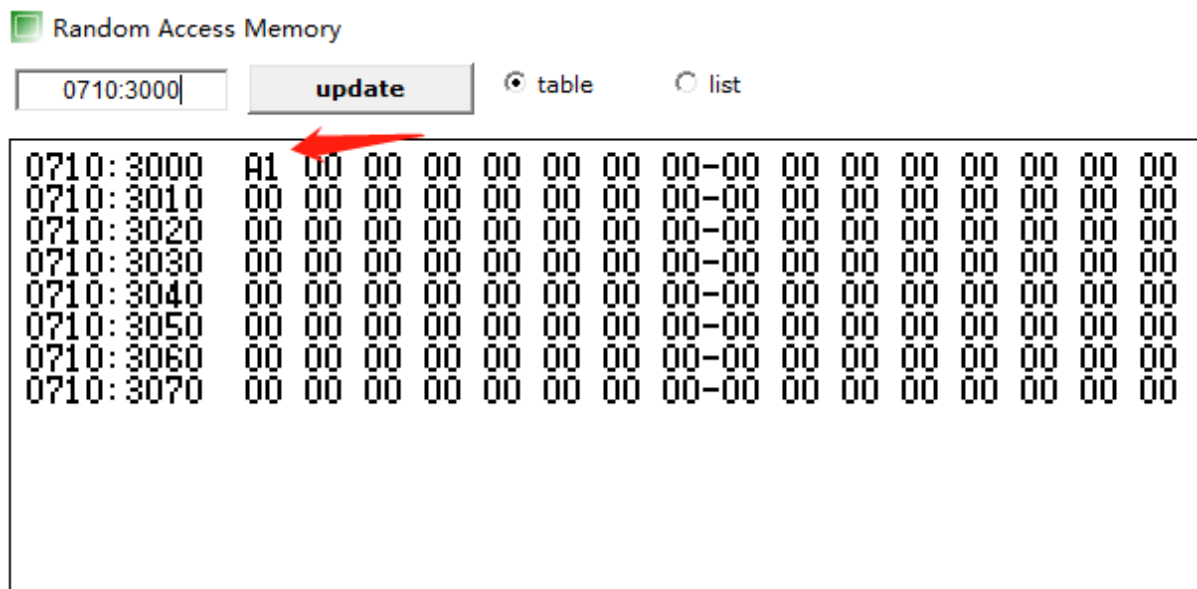
1 实验思路

将第一个数放入AL中，设置成MAX值，对后面的数逐个与AL比较，若比MAX大，则更新Max值，否则继续循环，最后储存在AL中的即为最大值

2 实验结果



可以看到，第一次循环时，程序判断A1大于5C，因此不进入NEXT段，而是顺序进行，更新max值



当程序运行完后，可以看到对应地址填上了最大值

同时该代码实现了任意量数据的比较，只需在x处添加数据，以及修改对应CX（循环次数）的值即可

源码：MAX.asm

```
1  DATA SEGMENT
2      X DB 05CH,0A1H,086H
3      MAX DW ?           ;bug,db 会报错
4  DATA ENDS
5
6
7  CODE SEGMENT
8      ASSUME DS:DATA, CS:CODE
9  START:MOV AX, DATA
10     MOV DS, AX
11     MOV AH, 00H
12     MOV CX, 3
13     LEA SI,X
14     MOV AL, [SI]
15     INC SI
16     DEC CX
17 AGAIN: CMP AL, [SI]
18     JA  NEXT
19     MOV AL,[SI]
20 NEXT:  INC SI
21 LOOP AGAIN
22     MOV [3000H],AL
23     MOV AH,4CH
24     INT 21H
25 CODE ENDS
26 END START
```

心得：注意跳转指令是否考虑符号，此处JA为无符号大于跳转，JG则为有符号大于，所以用JG的话，5C为最大值

第二题

设数据段中初始存放了1个字节的无符号数，编写程序判断这个数(字节)的最高位和最低位是否全为1，如果全为1，将储存单元100H设为1；否则设为0。（样例: 8BH\47H\64H

8BH=1000 1011（对应为1

47H=0100 0111（对应为0

64H=0110 0100（对应为0

1 实验思路

先将字节与81H进行与操作，再判断操作完的数是否与81H相等，若相等，说明最高位与最低位都为1，否则将对应位置置0

此处为了避免重复进行更换数值，因此将所有数值一次输入，在代码中进行循环，对应判断的结果也在100H后的每一位

为了实现上述的效果，只需维护两个指针即可，对应每次循环，两个指针均加1

2 实验结果

可以看出，当第一个数为8BH时，程序进行全为1的程序段，同时也将对应的100H赋为1

The screenshot displays a debugger interface with three main panels. The left panel shows the 'original source code' window with the following assembly code:

```
DATA SEGMENT
X DB 8BH,47H,64H
DATA ENDS

CODE SEGMENT
ASSUME DS:DATA,CS:CODE
START: MOV AX,DATA
MOV DS,AX
MOV CX,3
LEA SI,X
MOV DI,100H
MOV BL,81H
JUDGE: MOV AL,[SI]
AND AL,BL
CMP AL,BL
JNE M1      ;不全为1
MOV [DI],1  ;全为1
M1: INC DI
INC SI
LOOP JUDGE
CODE ENDS
END START
```

The middle panel shows the 'registers' window with the following values:

Register	H	L
AX	07	81
BX	00	81
CX	00	03
DX	00	00
CS	0711	
IP	0018	
SS	0710	
SP	0000	
BP	0000	
SI	0000	
DI	0100	
DS	0710	
ES	0700	

The right panel shows the 'Random Access Memory' window with the following data:

Address	Value
0710:0100	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0710:0110	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0710:0120	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0710:0130	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0710:0140	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0710:0150	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0710:0160	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0710:0170	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

最后实验结果：

Random Access Memory

0710:0100 **update** ☒ table ☐ list

0710:0100	01	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	ff...
0710:0110	00	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:0120	00	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:0130	00	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:0140	00	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:0150	00	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:0160	00	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:0170	00	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00

与最开始的直接计算答案一致

源码: ALL1.asm

```

1  DATA SEGMENT
2      X DB 8BH,47H,64H
3  DATA ENDS
4
5
6  CODE SEGMENT
7      ASSUME DS:DATA,CS:CODE
8  START:MOV AX,DATA
9          MOV DS,AX
10         MOV CX,3
11         LEA SI,X
12         MOV DI, 100H
13         MOV BL,81H
14  JUDGE:  MOV AL, [SI]
15          AND AL, BL
16          CMP AL,BL
17          JNE N1          ;不全为1
18          MOV [DI],1      ;全为1
19  N1:     INC DI
20          INC SI
21  LOOP JUDGE
22  CODE ENDS
23  END START

```

问题三

设数据段中初始存放了分数在 1~100 的 10 个成绩，将这些成绩放入 初始地址为 3000H 的单元当中，3000H+I 表示第 I 位同学的成绩。编写程序。找出最高分的同学，将其编号 I 与成绩分别按字节放入 3100H 为起始单元的地址中。找出最低分的同学将其编号 I 与成绩分别按字节放入 3200H 为起始单元的地址中。 本题编号 I 的范围是 0~9。（样例: 56H, 4DH, 5DH, 52H, 64H, 47H, 51H, 5BH, 4FH,

1 实验思路

原计划采用双指针维护数据结构，实现数据迁移，但是由于寄存器数量不够，因此只好在对应的单元前补0

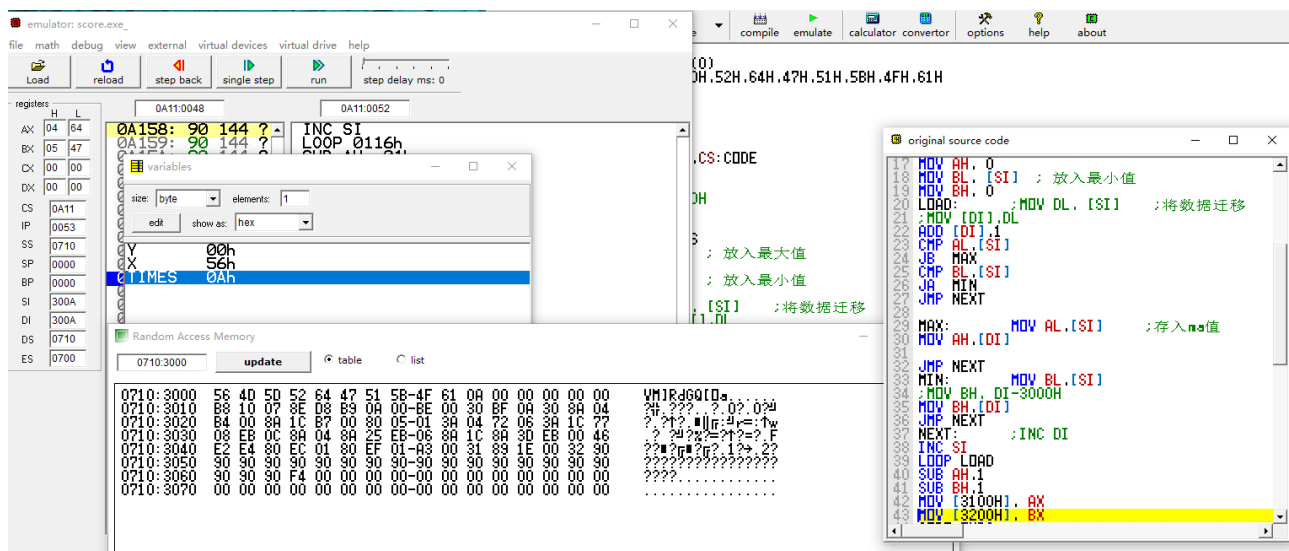
AX中，AH记录最大值编号，AL记录最大值

同理，BX记录最小值的信息

另外设置变量 `TIMES`，记录当前循环次数

2 实验结果

由下图可以看到，当程序运行完后



通过查询对应的偏移地址，可以发现对应最小值和最大值的信息已经存储在内存里

Random Access Memory

0710:3100

update

☒ table

☐ list

0710:3100	64	04	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:3110	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:3120	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:3130	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:3140	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:3150	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:3160	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00
0710:3170	00	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00

Random Access Memory

0710:3200

update

☒ table

☐ list

0710:3200	47	05	00	00	00	00	00	00-00	00	00	00	00	00	00	00	00	611
0710:3210	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	00	...
0710:3220	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	00	...
0710:3230	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	00	...
0710:3240	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	00	...
0710:3250	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	00	...
0710:3260	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	00	...
0710:3270	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	00	...

er

源码：

```

1  DATA SEGMENT
2      Y DB 3000H DUP(0)
3      X DB 56H,4DH,5DH,52H,64H,47H,51H,5BH,4FH,61H
4      TIMES DB 0
5  DATA ENDS
6
7
8  CODE SEGMENT
9      ASSUME DS:DATA,CS:CODE
10 START:MOV AX, DATA
11         MOV DS, AX
12         ;MOV DI, 3000H
13         MOV CX, 10
14         LEA SI, X
15         LEA DI, TIMES
16         MOV AL, [SI] ; 放入最大值
    
```

```
17      MOV AH, 0
18      MOV BL, [SI] ; 放入最小值
19      MOV BH, 0
20  LOAD:      ;MOV DL, [SI] ;将数据迁移
21             ;MOV [DI],DL
22             ADD [DI],1
23             CMP AL,[SI]
24             JB  MAX
25             CMP BL,[SI]
26             JA  MIN
27             JMP NEXT
28
29  MAX:      MOV AL,[SI] ;存入ma值
30             MOV AH,[DI]
31
32             JMP NEXT
33  MIN:      MOV BL,[SI]
34             ;MOV BH, DI-3000H
35             MOV BH,[DI]
36             JMP NEXT
37  NEXT:      ;INC DI
38             INC SI
39  LOOP LOAD
40  SUB AH,1
41  SUB BH,1
42  MOV [3100H], AX
43  MOV [3200H], BX
44  CODE ENDS
45  END START
```