学号： WA2224013

实验日期：2024.12.13

专业： 机器人工程

指导教师： 鲍华

姓名： 郭义月

实验成绩：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程目标 1 (权重 ) | 课程目标 2 (权重 ) | 课程目标 3 (权重 ) | 课程目标 4 (权重 ) | 课程目标 5 (权重 ) | 课程目标 6 (权重 ) | 综合成绩  (目标数可增删) |
|  |  |  |  |  |  |  |

安徽大学人工智能学院本科实验报告

【课程名称】 微型计算机原理及接口技术

【课程目标】

提升对内存操作、条件判断、循环控制与子程序的理解与应用能力。

【实验名称】 实验三 子程序调用

【实验目的】

1.学习子程序的定义和调试方法。

2.掌握子程序、子程序嵌套、递归子程序的结构:

3.掌握子程序的程序设计及调试方法。

【实验原理及方法】

本实验的原理基于汇编语言中对无符号字节序列的处理，通过逐字节比较找出序列中的最大值和最小值。实验中，数据存储在内存特定地址（3000H），利用子程序结构将求最大值和最小值的功能模块化，提高了程序的可维护性与调试效率。通过条件判断与循环控制，每次比较当前字节与已有的最大值或最小值，更新相应的值。同时，采用字符串偏移量输出16进制数据，避免了直接输出时可能出现的乱码问题。最终，程序通过寄存器（如AX）存储结果，其中AH表示最大值，AL表示最小值，完成了对字节序列的有效处理与计算。

【实验内容及过程】

求无符号字节序列中的最大值和最小值，其存储首地址为3000H，字节数为08H。利用子程序的方法编程求出该序列中的最大值和最小值。程序流程如所示。

流程图：

|  |  |
| --- | --- |
|  |  |

实验步骤:

(1)根据程序流程图编写实验程序;

(2)经编译、链接无误后装入系统:

(3)在 3000H 为存储首地址处输入 8 个字节的数据，如: D9 07 8B C5 EB 04 9D F9

(4)运行实验程序;

(5)点击停止按钮，停止程序运行，观察寄存器窗口中AX 的值，AX 应为 F9 04，其中 AH 中为最大值，AL 中为最小值;

(6)反复测试几组数据，检验程序的正确性。

本实验共写了三个子程序，分别为：

MAX\_MIN：找到最大值及最小值放在BX中，其中BH存放最大值，BL存放最小值；

PRINTLOWER：输出低八位的值，配合TEMP DB '\*\*\*\*\*\*\*MIN\*\*\*\*\*\* $'字符串可输出；

PRINTHIGHER：输出高八位的值；配合TEMP1 DB '\*\*\*\*\*\*\*MAX\*\*\*\*\*\* $'字符串可输出

主程序：

DATAS SEGMENT

;此处输入数据段代码

ORG 3000H

s1 db 0D9h, 05h, 8Bh, 0C5h, 0EBh, 47h, 9Dh, 0FEh

TEMP DB '\*\*\*\*\*\*\*MIN\*\*\*\*\*\* $'

TEMP1 DB '\*\*\*\*\*\*\*MAX\*\*\*\*\*\* $'

CHANGELINE DB 13,10,'$'

; 输入的字节序列

DATAS ENDS

STACKS SEGMENT

;此处输入堆栈段代码

STACKS ENDS

CODES SEGMENT

ASSUME CS:CODES,DS:DATAS,SS:STACKS

START:

MOV AX,DATAS

MOV DS,AX

;此处输入代码段代码

;MOV SI,3000H

MOV CX,8

CALL MAX\_MIN

MOV AX,BX

CALL PRINTLOWER

MOV DX,OFFSET CHANGELINE

MOV AH,09H

INT 21H

CALL PRINTHIGHER

MOV AH,4CH

INT 21H

按照流程图的逻辑编写MAX\_MIN程序：

MAX\_MIN PROC NEAR

MOV SI,OFFSET S1

MOV CX,8

MOV BH,00H

MOV BL,0FFH

AGAIN:

MOV AL,[SI]

CMP AL,BH

JAE GREATER

CMP AL,BL

JBE LOWER

JMP NEXT

GREATER:

MOV BH,AL

JMP NEXT

LOWER:

MOV BL,AL

NEXT:

INC SI

DEC CX

JNZ AGAIN

RET

MAX\_MIN ENDP

PRINTLOWER：输出bx的低8位，即最小值（在主程序中要提前实现MOV AX,BX）因为在本子程序中，默认的数是存储在ax中的：

PRINTLOWER PROC NEAR

MOV SI,OFFSET TEMP+1;输出的偏移地址

XOR CX,CX

MOV CL,02H

PRINT:

MOV DH,AL

SHR AX,1

SHR AX,1

SHR AX,1

SHR AX,1

AND DH,0FH

ADD DH,30H

CMP DH,':'

JA IS

JB NO

IS:

ADD DH,07H

NO:

MOV [SI],DH

DEC SI

LOOP PRINT

MOV DX,OFFSET TEMP

MOV AH,09H

INT 21H

RET

PRINTLOWER ENDP

PRINTHIGHER：输出bx的高8位，即最大值

PRINTHIGHER PROC NEAR

MOV SI,OFFSET TEMP1+1

XOR CX,CX

MOV CL,02H

PRINT:

MOV DH,AL

SHR AX,1

SHR AX,1

SHR AX,1

SHR AX,1

AND DH,0FH

ADD DH,30H

CMP DH,':'

JA IS

JB NO

IS:

ADD DH,07H

NO:

MOV [SI],DH

DEC SI

LOOP PRINT

MOV DX,OFFSET TEMP1

MOV AH,09H

INT 21H

RET

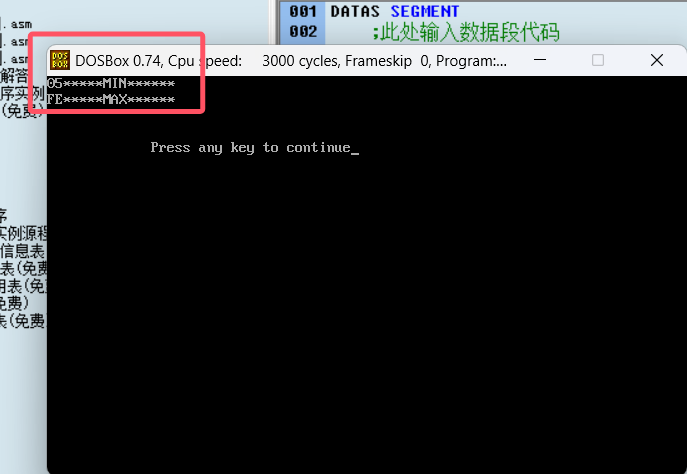
PRINTHIGHER ENDP

【实验结果】

当输入s为：

s1 db 0D9h, 05h, 8Bh, 0C5h, 0EBh, 47h, 9Dh, 0FEh

此时可输出最大值和最小值，其中05h是最大值，FEH是最小值



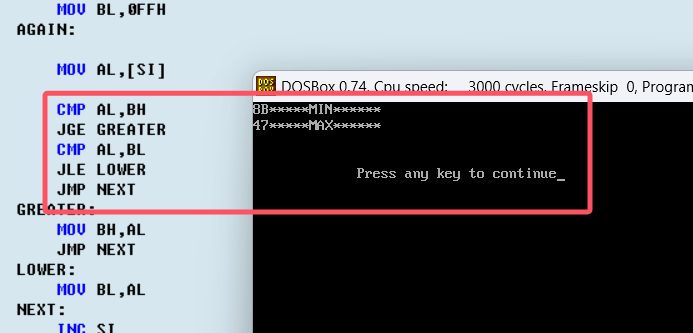
若更改代码逻辑为有符号数比较，此时负数以补码形式存放，可输出有符号的最大值最小值，更改跳转函数如下：

CMP AL,BH

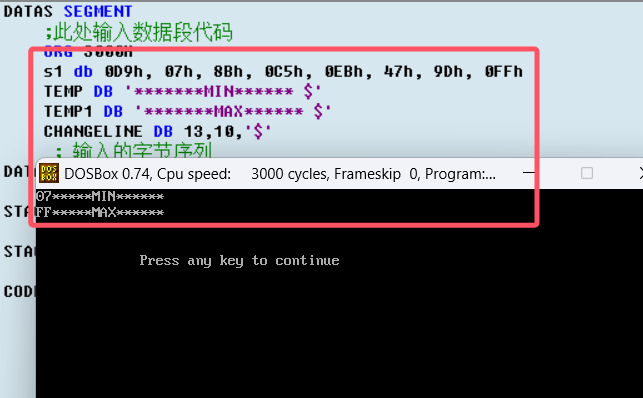
JGE GREATER

CMP AL,BL

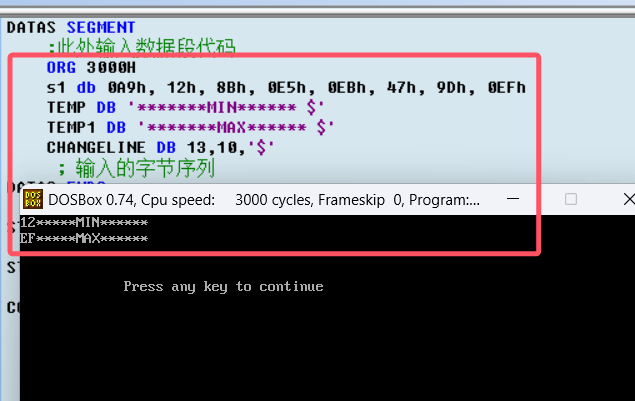
JLE LOWER



可更改s1的值，多次运行，可验证代码的正确性：



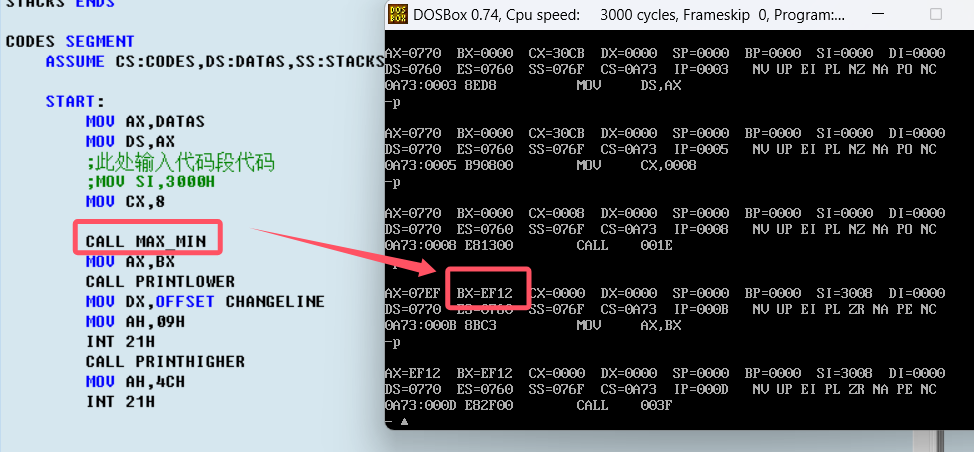
多次更改s1的值，可验证代码能实现输出最大值最小值的功能



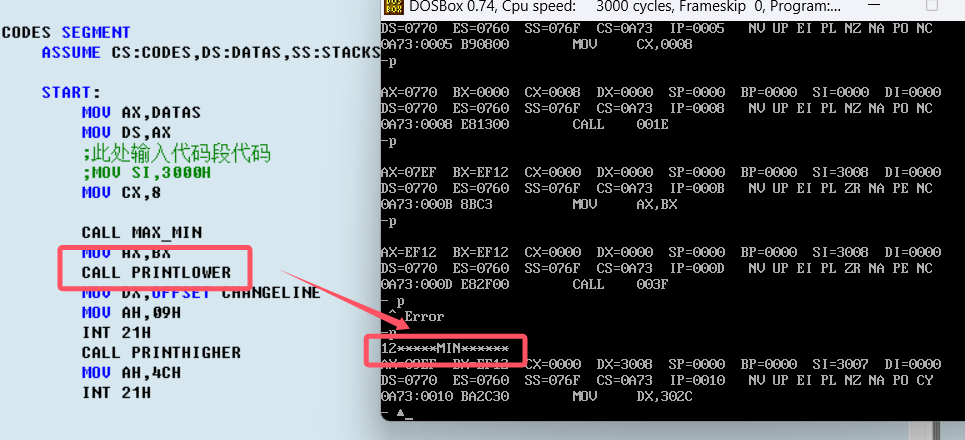
【数据分析及处理】

比较大小求最小值的逻辑并不复杂，按照流程图里面走就可以，本部分主要展示将16进制数输出的逻辑。

由调试结果可得，当调用max\_min函数后，bx中已经存放了最大值与最小值，其中，bh存放最大值，bl存放最小值：

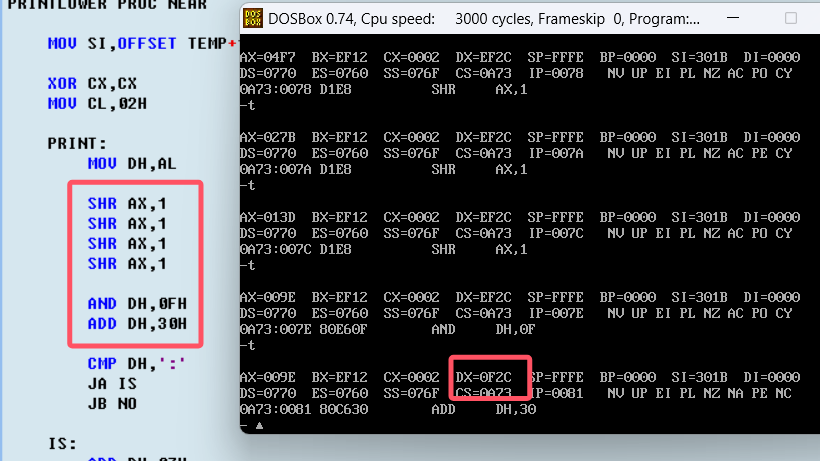


调用输出低位的函数，可实现输出低位与temp字符串：

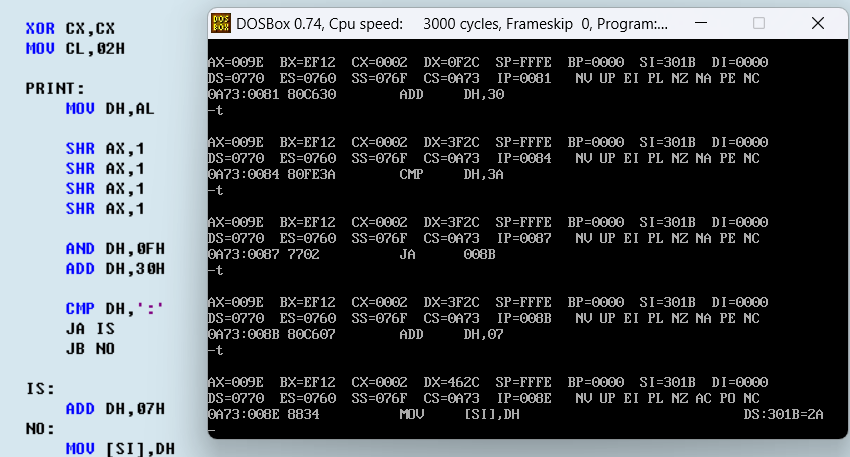


由于PRINTERLOWER与PRINTERHIGHER的函数逻辑是类似的，现只展示PRINTERHIGHER单步进入的逻辑：

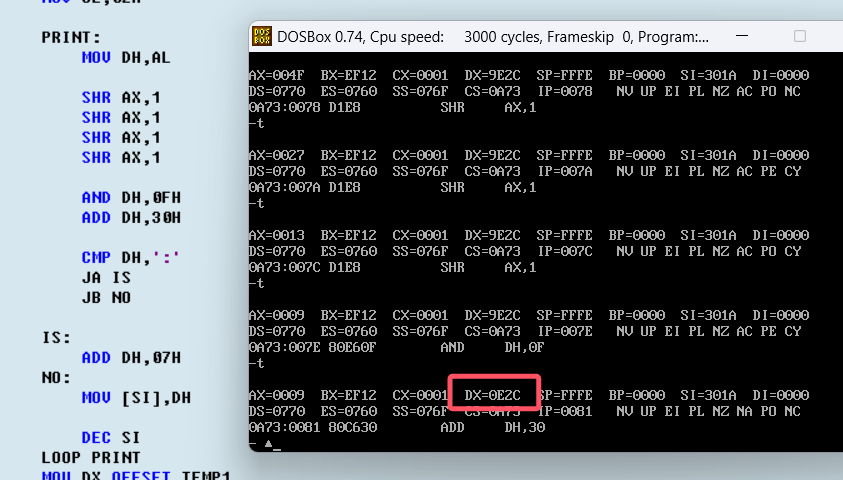
因为要取2位16进制数，所以cx=2，循环两次。完成第一次循环后，已经把最大值的低位取出，存放在dh的低位中



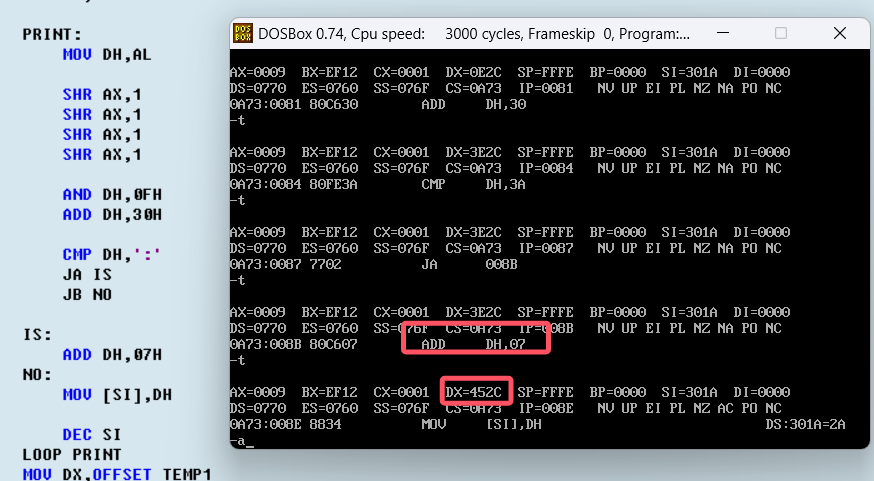
‘:’是数字与字母的ascii码界限，f是字母，需要加07h变成6h



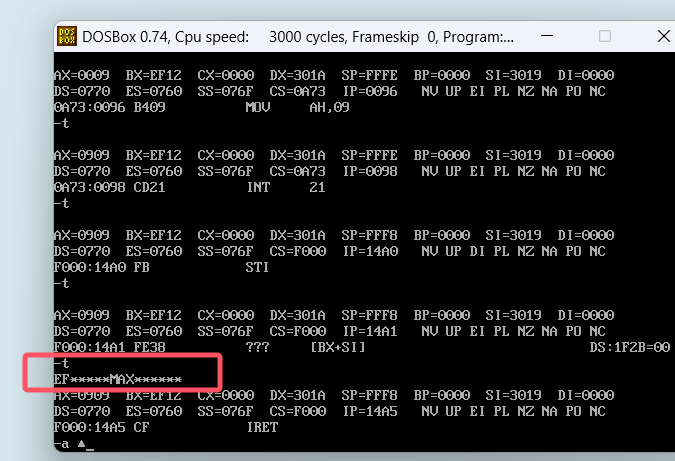
第二次循环取出bh的高位，放在dh的低位中



E是字母，需要加上07h变成5



最后存放在si中的最大值就是ef



【总结或讨论】

这次实验深刻体会到了子程序结构的重要性。通过将不同功能模块独立成子程序，代码变得更加清晰，调试也更加高效。之前的编程经验中，代码常常堆在一起，导致程序结构混乱，修改时也比较困难。而通过子程序的设计，程序被分解成多个小块，每个子程序负责一个具体任务，提升了代码的可维护性与可读性。当某个功能出现问题时，直接定位到对应子程序，避免了在整个程序中查找错误的繁琐过程。

绘制流程图的步骤也提供了很大帮助。虽然一开始觉得直接开始写代码就好，但实际上，提前画出流程图能帮助理清程序的结构和逻辑，避免了后期出现思路混乱或逻辑不清的问题。流程图不仅是一个规划工具，也能有效地减少编写过程中的错误，节省调试时间。在规划和设计阶段就做好充分准备，整个编程过程会更加顺利。

在DOS环境中，直接输出16进制数值经常出现乱码，导致难以查看结果。最终采用了通过字符串偏移量来实现输出的方案，例如将数据嵌入字符串中，使用偏移量控制显示内容。虽然这种方法较为曲折，但解决了问题，也提醒了在低级编程中，有时需要通过间接的方式绕过系统的限制。

本次实验不仅加深了对汇编语言中子程序设计的理解，也提升了调试能力与编程思维。在汇编这种低级语言中，合理的程序结构和细致的调试尤为重要，通过这次实践，不仅技术得到了提升，编程习惯也变得更加严谨和规范。