**计算机科学与工程学院实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实验课程名称** | | **汇编语言程序设计** | | | **实验总成绩** |  |
| **专业** | **计算机科学与技术** | | **班级** | **计算机1803** | **指导教师签字** |  |
| **学号** | **20184484** | | **姓名** | **胥卜凡** | **实验报告批改时间** |  |
| **实验报告分项成绩**   |  |  |  | | --- | --- | --- | | **序号** | **实验项目** | **成绩** | | **1** | **子程序设计实验** |  | | **2** |  |  | | **3** |  |  | | **4** |  |  | | **5** |  |  | | | | | | | |
| **实验课程总结**  从以下方面总结：1.实验体现知识应用和初步研究能力；2. 反映基本观察、发现问题和分析问题能力；3. 实验项目内容或者实验课程是否存在问题及下一年度改进意见）4.其他方面  总体来看，本次实验极大地丰富了我的动手实践能力，让我深入了子程序的书写方法、使用方法、调试方法。  具体来说，为完成本次实验，我首先观察了题目，发现需要对多字节数进行取绝对值，这里需要用到循环语句、分支语句的设计，既需要用LOOP、JNS等操作进行循环、分支的实现，也需要使用ADC、NOT等操作进行取补运算。其次，对于子程序的设计，要按标准进行编写使用PROC、ENDP等关键字。  综上所述，本次实验极大地增加了我的观察问题、发现问题、分析问题的能力。 | | | | | | |

**实验一**

**（一）实验内容：**

**题目：**

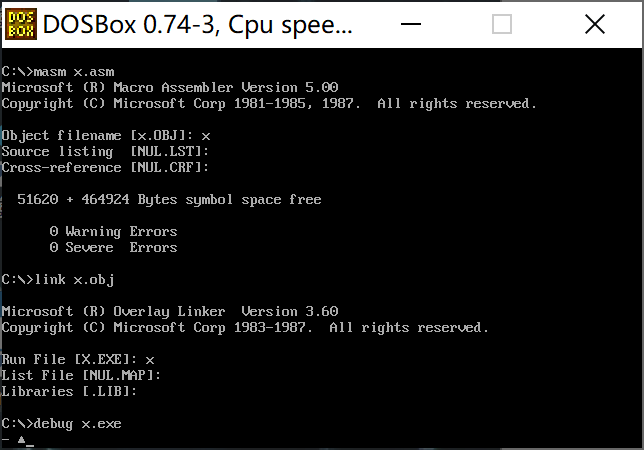
已知两个长度相等的带符号的多字节数据分别存放在内存DATA1和DATA2开始的连续单元中，数据长度存放在LEN单元。编制程序，计算两个数据的绝对值之和，将结果存放在SUM开始的连续单元中。

**要求：**

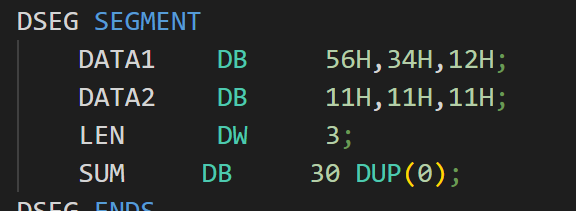
求一个多字节数据的绝对值及求两个多字节数据的和的功能分别用子程序来实现；分别用三组数据调试程序，检验程序的正确性：① 两个多字节数据都是正数；② 两个多字节数据都是负数；③ 两个多字节数据一正一负。

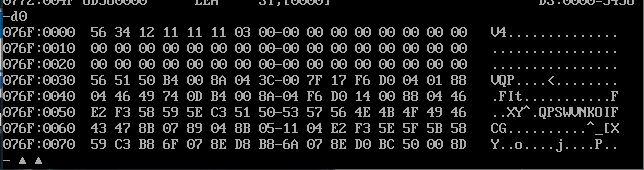
**（二）实验步骤：**

①编写程序，并进行汇编、连接，进入调试模式。。

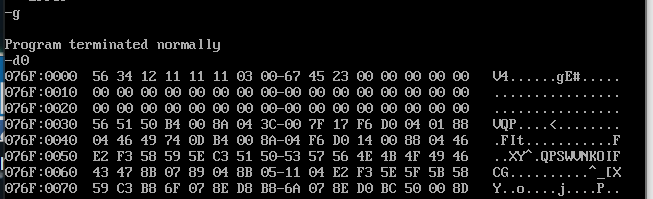


②首先查看第一组数据（56H,34H,12H）(11H,11H,11H),期望输出(234567)H，即（67H,45H,23H）



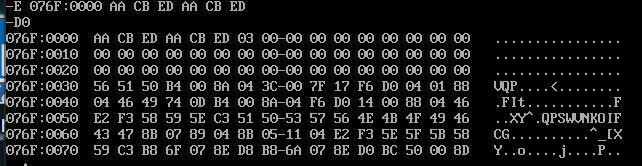


③执行程序，查看结果

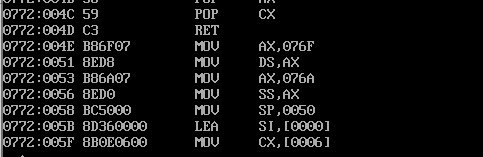


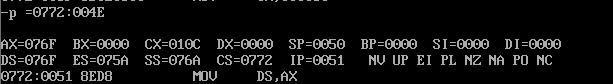
可以看到对应的值已经被取了绝对值，SUM中存储了（67H,45H,23H）

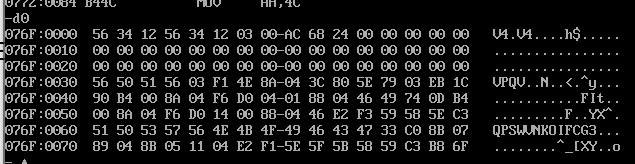
④用E命令更改内存数据为DATA1:（AAH,CBH,EDH）(-123456)H DATA2:(AAH,CBH,EDH)期望输出2468ACH,即内存中为（AC，68，24）H



⑤从头开始重新执行到目标处





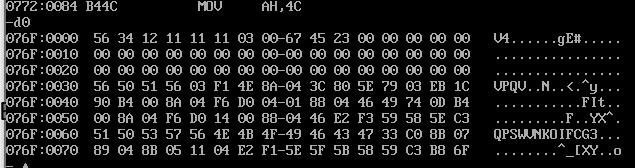


可以看到对应的值已经被取了绝对值，SUM中存储了（AC，68，24）H

⑥用E命令更改内存数据为DATA1:（56H,34H,12H）DATA2:(EFH,EEH,EEH) (-111111)H期望输出(234567)H,即内存中为（67，45，23）H



⑦执行文件并查看结果



可以看到对应的值经被取了绝对值，SUM中存储了（67，45，23）H

**（三）遇到问题及思考：**

①目标输出与期望输出不对应

答：存储器上多字节数的存储时按照高字节在高地址，低字节在低地址的顺序存储

②子程序使用的相关讨论

答：子程序就相当于高级语言的函数，为了不影响原数据，需要用栈来保存现场

③使用p、t重新进行操作时出现bug

答：一开始使用的地址是子程序的地址，故出现bug

④如何队多字节数据进行取绝对值

答：正数不动，负数进行取补运算，判断要根据最高位的状态进行判断，即最高位与80H进行比较后进行分支跳转，对于多字节数据，低位取反后加1，之后的各位采用取反后，与0进行ADC 运算即可。

⑤数据输出错误

答：加法操作时，CF清零位置放错甚至第一次忘记清零了

**附录：**

1. **源代码:**

**实验五:**

SSEG SEGMENT    STACK

 STK DB 80 DUP (0)

SSEG ENDS

DSEG SEGMENT

    DATA1    DB     56H,34H,12H;

    DATA2    DB     11H,11H,11H;

    LEN      DW     3;

    SUM      DB     30 DUP(0);

DSEG ENDS

CSEG SEGMENT

    ASSUME DS:DSEG,CS:CSEG

    ASSUME SS:SSEG

MNEG PROC

    PUSH SI;保存低字节地址

    PUSH AX;由于下面使用了，所以需要保存现场

    PUSH CX

ZHENGFU:

    PUSH SI

    ADD SI,CX

    DEC SI

    MOV AL,BYTE PTR[SI];

    CMP AL,80H

    POP SI

    JNS QUBU

    JMP RETURN            ;判断最高位

QUBU:

    MOV AH,0;

    MOV AL,BYTE PTR [SI];从最低位开始算起，因为需要加1所以单独运算，大概也可以通过STC等简化操作

    NOT AL;取反

    ADD AL,1;加一，得到补码，影响CF

    MOV BYTE PTR[SI],AL;

    INC SI;访问更高的位，不影响CF

    DEC CX;不影响cf

    JZ RETURN;判断CX是否为0

AGAIN:

    MOV AH,0;也可使用LODSB等控制，这里手动进行操作

    MOV AL,BYTE PTR[SI]

    NOT AL;取非，不影响CF

    ADC AL,0;带进位加法

    MOV BYTE PTR[SI],AL

    INC SI

    LOOP AGAIN

RETURN:

    POP CX

    POP AX

    POP SI;返回低字节地址

    RET

MNEG ENDP

JIA     PROC

        PUSH CX

        PUSH AX

        PUSH BX

        PUSH DI

        PUSH SI;保存现场

        DEC SI

        DEC BX

        DEC DI

        DEC CX

        XOR AX,AX;CF位清零

AGAIN2:

        INC SI

        INC BX

        INC DI

        MOV AX,[BX]

        MOV [SI],AX

        MOV AX,[DI]

        ADC [SI],AX

        LOOP AGAIN2

        POP SI

        POP DI

        POP BX

        POP AX

        POP CX

        RET

JIA  ENDP

START:

    MOV AX,DSEG;

    MOV DS,AX;

    MOV AX,SSEG;

    MOV SS,AX

    MOV SP,LENGTH STK

    LEA SI,DATA1;

    MOV CX,LEN

    CALL MNEG;执行子程序

    LEA SI,DATA2;

    MOV CX,LEN;

    CALL MNEG;

    LEA SI,SUM;

    LEA BX,DATA1;

    LEA DI,DATA2;

    MOV CX,LEN;

    CALL JIA

    MOV AH,4CH

    INT 21H

ENDSSS:

        MOV  AH,4CH

        INT  21H

CSEG ENDS

    END START