

计算机工程学院

**汇编语言与接口技术**

**实验报告**

学年学期：  **2022 - 2023 学年 第1学期**

实验名称：  **实验一 算术运算程序设计**

班级： 计算2114

学号： 202121331104

姓名： 庄佳强

日期： 2022.10.16

成绩：

**一、 实验要求**

1. 编程实现两个数：#998877H 和 #778899H 的加法运算。
2. 编程实现两个数：#998877H 和 #778899H 的减法运算。

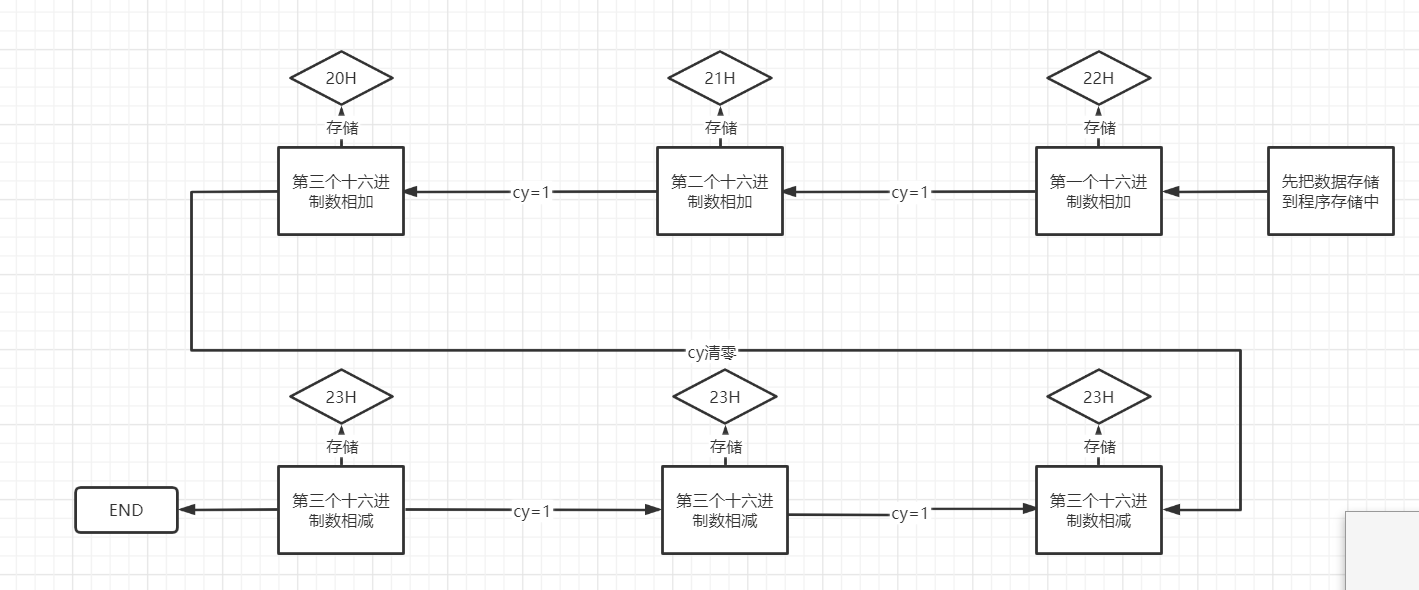
**二、 实验设计**

**1.整体思路**

通过对程序存储字节访问，把存储的数据读入到累加器中和寄存器中暂时存储，再通过ADDC和SUBB计算 最后存储到memory中 。

注意A寄存器和cy的清零就行。

**2.流程图**



**3.主要模块设计思路及分析**

1. 加法模块

加法中首先DPTR定位到NUM1的位置，再用MOVC取出第一个数的第一位十六进制数 存放到R1寄存器中，同理再取出第二个数的第一位十六进制数 存放在A中 ，在用ADDC 计算加法存放在memory中。 其他同理。

在进行完加法模块后应将cy和A清空，防止出现错误。

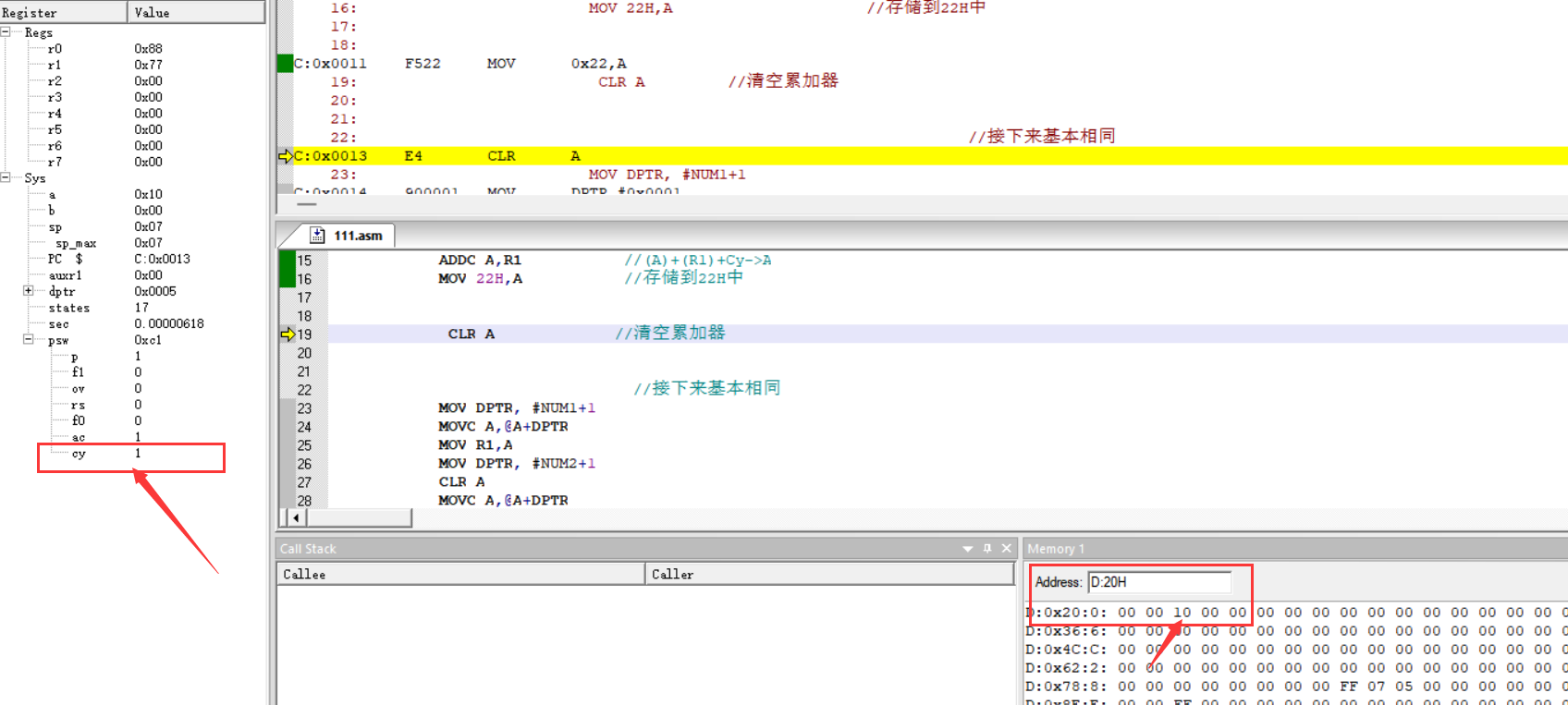
1. 减法模块

加法中首先DPTR定位到NUM2的位置，再用MOVC取出第一个数的第一位十六进制数 存放到R1寄存器中，同理再取出第二个数的第一位十六进制数 存放在A中 ，在用SUBB 计算加法存放在memory中。 其他同理。

因为减法是 A-B 但 SUBB只能 A寄存器减XXX 所有要先取出NUM2 的值。

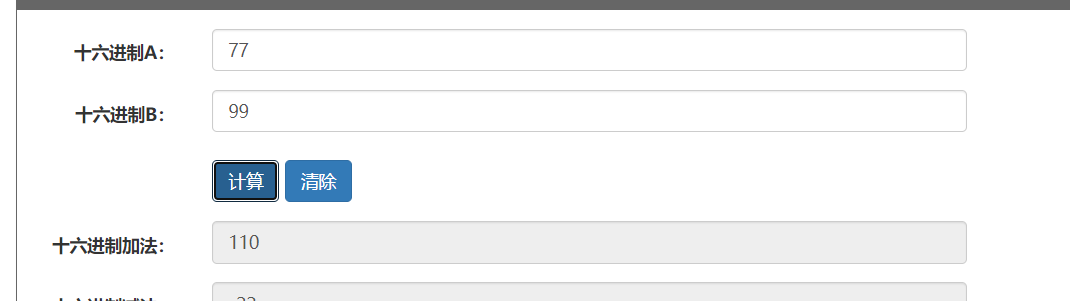
1. **实现效果**

第一位的十六进制数相加后

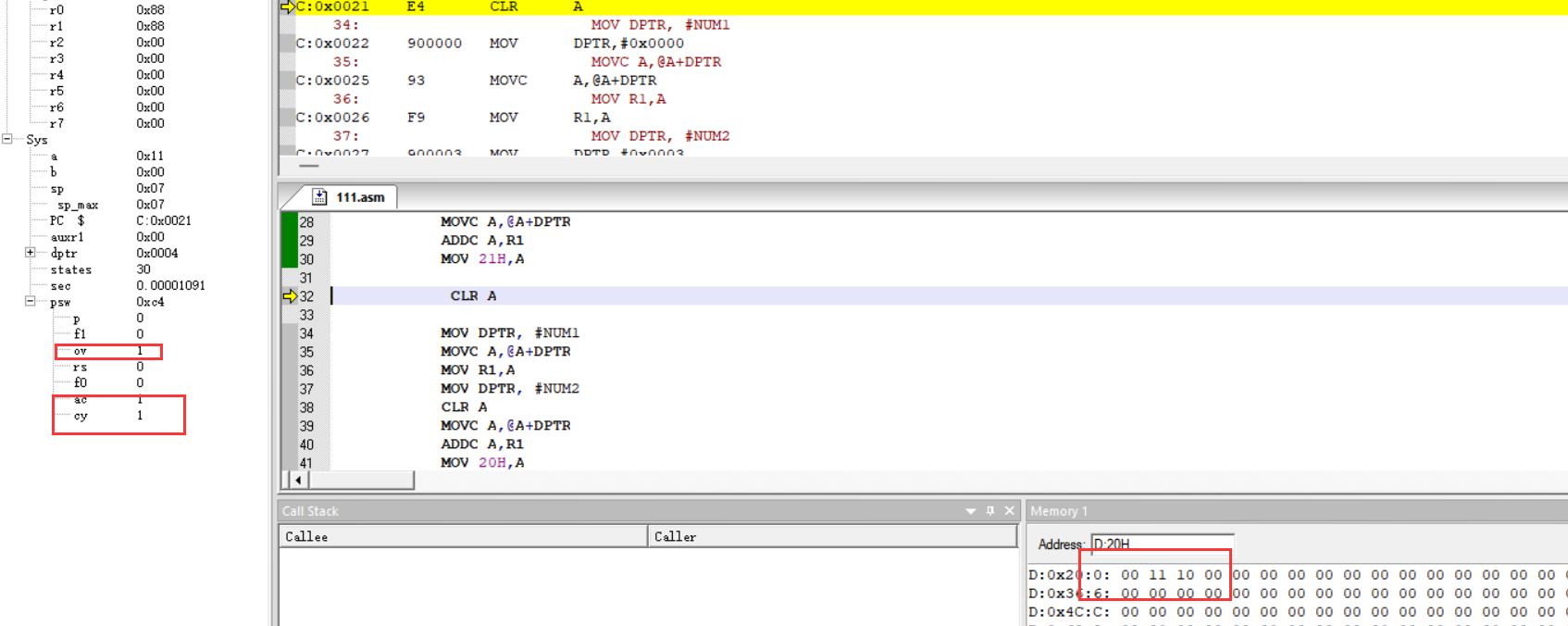


77H+99H=110H

位7有进位，位3也有进位 位6有进位 所有cy=1 ac=1 ov=0



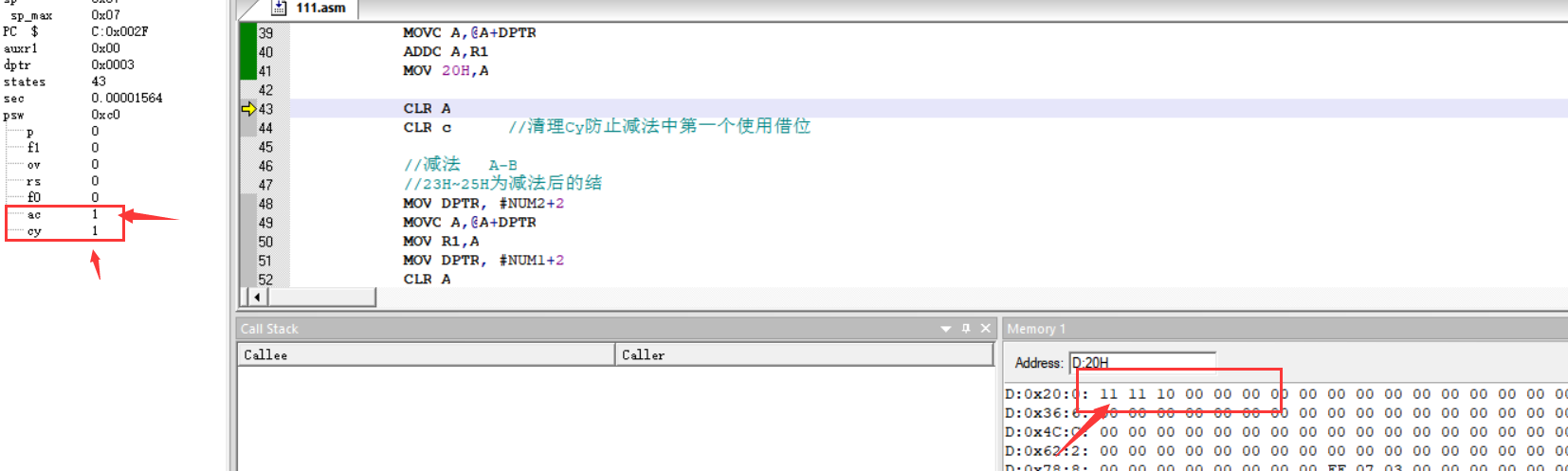
第二位的十六进制数相加后



88+88+1=111

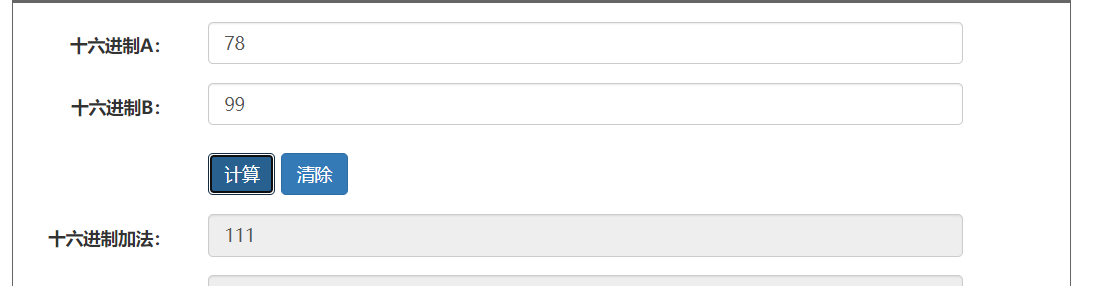
位7有进位，位3也有进位位6没有进位 所有cy=1 ac=1 ov=1



第三位的十六进制数相加后

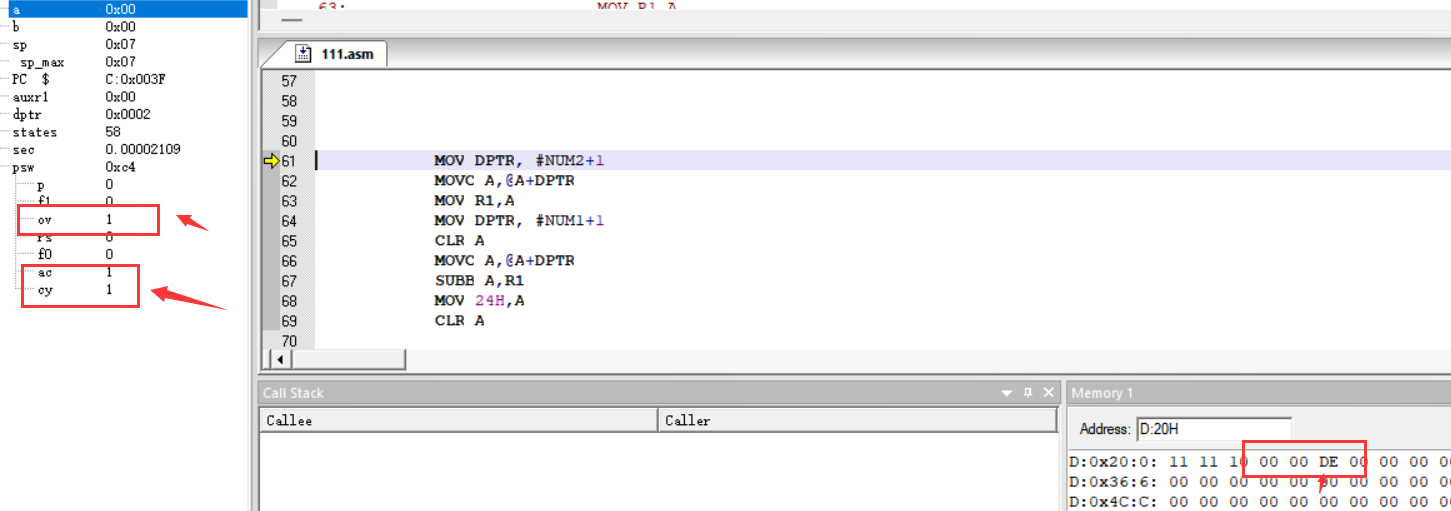
99+77+1=111

位7有进位，位3也有进位 位 6有进位 所有cy=1 ac=1 ov=0



减法中

第一位的十六进制数相减后



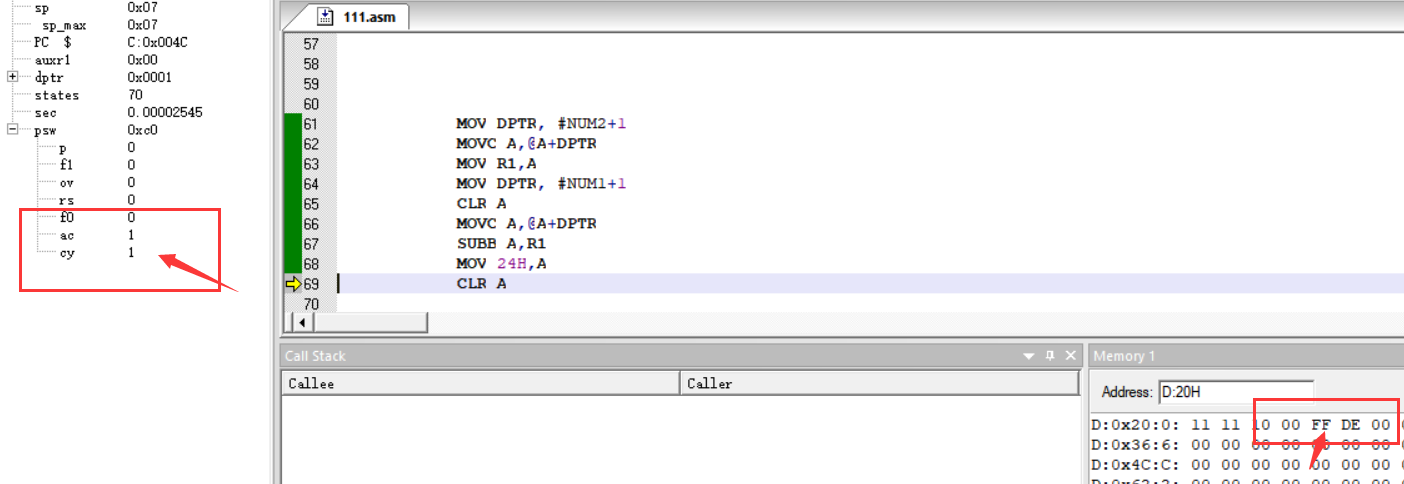
77-99 不够 向下一个借一位 cy =1

位3也借位 ac=1 位6没有借位 ov=1

177-99=DE



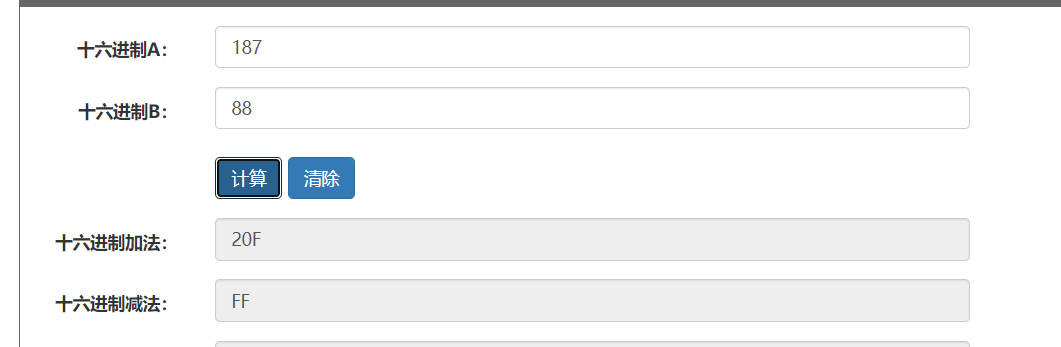
第二位的十六进制数相减后



88-1 —> 87 -88 不够 向下一个借一位 cy =1

位3也借位 ac=1 位6有借位 ov=0

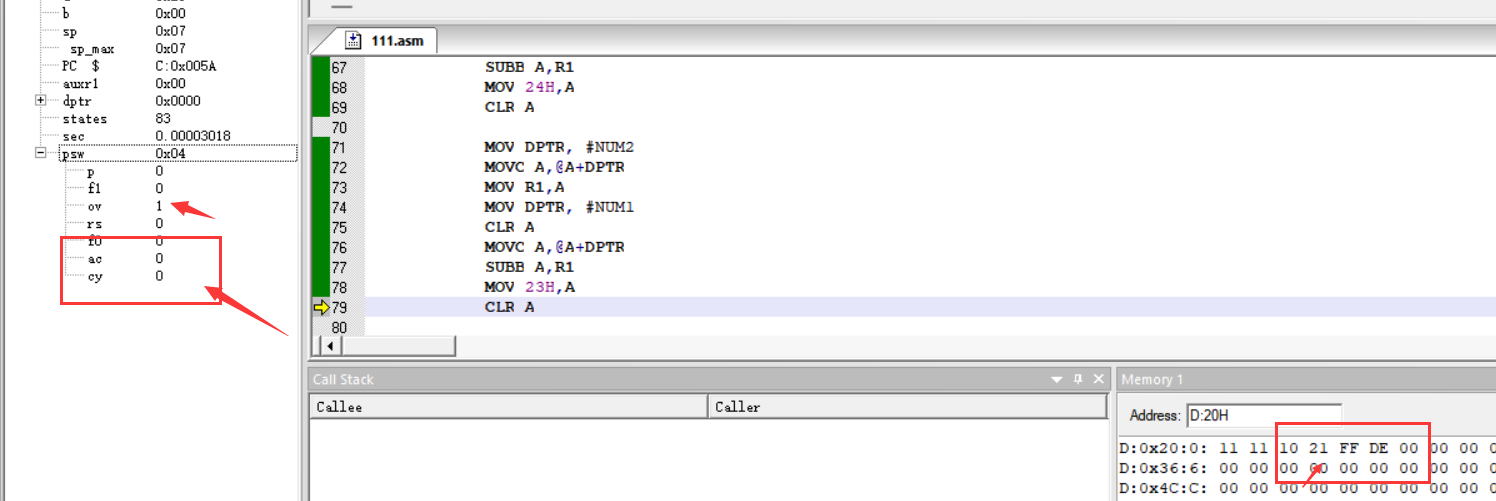
187-88=FF

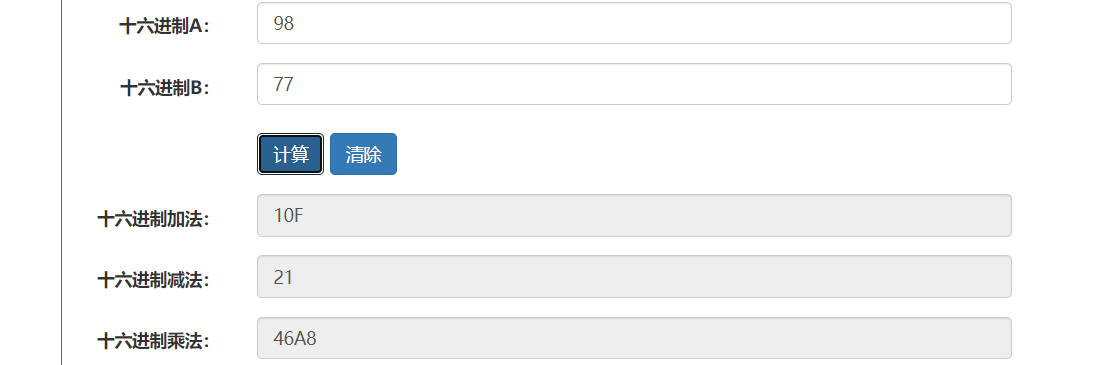


99-1-77=21

没有向下一个借一位 cy =0

位3也借位 ac=0 位6有借位 ov=1





1. **总结**

1.通过这个实验，我学会了基本的数据处理和存储。

2.掌握了ADDC 和SUBB，认识了cy 进位和借位。

实验的心得体会，包括遇到了哪些困难，怎么一一克服的，有什么收获，以及对实验的改进建议。

**附录：**

ORG 0000H

NUM1: DW 9988H

DB 77H

NUM2: DB 77H, 88H, 99H

//第一个十六进制数加法

//20H~22H为 加法后的结果

START: MOV DPTR, #NUM1+2 //寻找到第一个十六进制数的程序存储字节位置

MOVC A,@A+DPTR //把对于位置的字节传送到累加器

MOV R1,A //先把累加器的字节传送到R1备用

MOV DPTR, #NUM2+2 //在寻找另外一个十六进制数的程序存储字节位置

CLR A //清空累加器防止 传送出现错误

MOVC A,@A+DPTR //把对于位置的字节传送到累加器

ADDC A,R1 //(A)+(R1)+Cy->A

MOV 22H,A //存储到22H中

CLR A //清空累加器

//接下来基本相同

MOV DPTR, #NUM1+1

MOVC A,@A+DPTR

MOV R1,A

MOV DPTR, #NUM2+1

CLR A

MOVC A,@A+DPTR

ADDC A,R1

MOV 21H,A

CLR A

MOV DPTR, #NUM1

MOVC A,@A+DPTR

MOV R1,A

MOV DPTR, #NUM2

CLR A

MOVC A,@A+DPTR

ADDC A,R1

MOV 20H,A

CLR A

CLR c //清理Cy防止减法中第一个使用借位

//减法 A-B

//23H~25H为减法后的结果

MOV DPTR, #NUM2+2

MOVC A,@A+DPTR

MOV R1,A

MOV DPTR, #NUM1+2

CLR A

MOVC A,@A+DPTR

SUBB A,R1

MOV 25H,A

CLR A

MOV DPTR, #NUM2+1

MOVC A,@A+DPTR

MOV R1,A

MOV DPTR, #NUM1+1

CLR A

MOVC A,@A+DPTR

SUBB A,R1

MOV 24H,A

CLR A

MOV DPTR, #NUM2

MOVC A,@A+DPTR

MOV R1,A

MOV DPTR, #NUM1

CLR A

MOVC A,@A+DPTR

SUBB A,R1

MOV 23H,A

CLR A

END