西南大学

**实 验 报 告**

《计算机组成与汇编语言程序设计》课程

2023-2024学年度第1学期

专业年级 软工中外 22级

姓 名 吴孜远

学 号 222022321062009

任课教师 李玉州

实验教师 李玉州

上机地点 25教 803

**西南大学计算机与信息科学学院**

2023年9月

**《汇编语言程序设计》课程实验报告**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实验题目 | | 计算算术表达式的值 | | |
| **实验时间** | | 2023年 10月17日 | | |
| 1. **实验目的及要求**   编写程序计算 (X+Y-M\*N)/(D+6) ,结果G存商 R存余数，所有变量均为16位无符号数.  要求:使用汇编语言编写程序,并单步运行每条指令,记录每一步的中间结果和标志位。  公式中的X,Y,M,N和D,应该在自定义的存储区空间内进行赋值,值自己取。在实验报告中应该有注释进行对应描述。 | | | | |
| 二、**实验内容**  **源代码：**  DATAS SEGMENT  X DW 20  Y DW 24  M DW 5  N DW 4  D DW 6  G DW 0  R DW 0  DATAS ENDS  STACKS SEGMENT  DB 128 DUP (?)  STACKS ENDS  CODES SEGMENT  ASSUME CS:CODES, DS:DATAS, SS:STACKS  START:  MOV AX, DATAS  MOV DS, AX  ; M\*N  MOV AX, M ;5，这个方括号加不加好像都可以，加上就是指从内存中读取 M 的值，不加就是将 M 当作立即数。  MOV DX, N ;4  MUL DX ;用DX的值乘AX的值，结果存在AX中存低位，DX中存高位  MOV BX, AX ;将AX和DX中存的乘法结果转移到BX和CX中  MOV CX, DX  MOV AX, X ;14  ADD AX, Y ;18 ，计算X+Y，  ADC DX, 0 ;为了手动加上借位的值到DX，毕竟无符号数扩位不能用CWD  SUB AX, BX  SBB DX, CX ;对高低位分别做减法，记得在高位减去借位  MOV CX, D ;计算D+6  ADD CX, 6  DIV CX ;CX为除数，AX为被除数  MOV G, AX ;这里要加方括号，表示将AX的值存入内存 G 中  MOV R, DX  MOV AH, 02  MOV G, AX ;将AX中的商存入G中  INT 21H  MOV AH, 02  MOV R, DX ;将DX中的余数存入G中  INT 21H  MOV AH, 4CH  INT 21H  CODES ENDS  END START  **DEBUG：**  MOV AX, M  MOV DX, N  MUL DX  MOV BX, AX  MOV CX, DX  MOV AX, X  ADD AX, Y    ADC DX, 0  SUB AX, BX  SBB DX, CX  MOV CX, D  ADD CX, 6  DIV CX  **最后的结果是2.0，可以看到AX为2，DX为0** | | | | |
| **三、实验总结与收获**  通过这次实验，我学会了汇编语言基本的加减乘除运算，了解了运算过程中各个寄存器的使用方式，以及标志位代表的含义和使用方法。一个简单的加减乘除运算，要考虑的细节太多，时刻要注意各个寄存器所储存的值到底合不合适，比如乘法除法默认在AX上执行，所以要保证乘数或者被除数在AX上储存。  实验中要求是无符号数，这时候我发现如果单纯像是处理有符号数那样使用CWD指令将字扩为双字会有错误，因为扩展时会将高位寄存器部分全部置为1，而无符号数扩展应该全部置为0才对，所以这时候不应该用CWD。这也算是一个细节。  另外学习求补指令时我发现我并不完全理解补码的含义，所以花了些时间去理解求补和补码，以及计算机中为什么需要补码，算是对第一章内容的一个补充吧：  以下是我的总结：  补码是一种机器数，用于表示负数，将减法操作转换为加法操作。正数的补码是原码  求补是一种操作，正数求补后变成对应的负数补码  例如：0000 0011（3）求补变为1111 1101（原码：1000 0011，-3）  NEG(X)实际上是在取相反数，-5求补是0-（-5）=5    实际上处理减法我们可以采用两种方式一种是补码相加，一种就是求补的方式。  例如： 12-8  12的三码合一都为 0000,1100  -8的原码：1000,1000          反码：1111,0111          补码：1111,1000（这是计算机中表示-8的方法，用256-8，也就是对8求补）  首先以补码相加的形式：        12       0000,1100      -  8     + 1111,1000  ——————————————          4     1,0000,0100  最高位越界丢弃。  另一种方式是求补X-Y=X+NEG(Y),NEG(8)=1111,1000          12       0000,1100      -  8     +  1111,1000  ——————————————           4      10000,0100    其实这两种方式都是一样的，因为NEG(X) = -X的补码。 | | | | |
| **成绩** |  | | **评阅老师** |  |