实验一 寻址方式与基本指令实验

一、 实验目的

1、熟悉80x86寻址方式及基本指令的功能，进一步理解和巩固课堂学习内容。

2、掌握汇编语言程序设计上机过程, 掌握汇编语言源程序结构，为后续汇编语言程序设计打好基础。

3、熟悉Microsoft的DEBUG或Borland的Turbo DEBUG调试工具的使用方法

**二、 实验预习要求**

1、认真阅读本书第一部分第1章，熟悉汇编语言程序设计上机步骤。

2、认真阅读本书第三部分，熟悉DEBUG调试工具的使用方法。

3、复习80x86操作数寻址方式及基本指令（数据传送、算术运算和逻辑运算等）。

4、了解实验内容，并在实验前根据课堂所学知识回答有关问题（个别取决于程序实际运行环境的问题除外），以便在实验时进行验证。

**三、 实验内容**

1. 读懂下列源程序，使用EDIT生成名为EX11.ASM的源程序，汇编生成EX11.OBJ文件和EX11.LST文件，连接生成EX11.EXE文件；用EDIT打开EX11.LST文件，了解.LST文件包含的信息；使用DEBUG调试工具单步执行EX11.EXE程序，注意观察IP值的变化，并回答下列问题。
2. 程序装入后，代码段寄存器CS的内容为\_\_\_\_04AF\_\_\_\_H，代码段第一条可执行指令MOV AX, DATA对应的机器代码为B8AE04 H，它是一个\_\_\_\_3\_\_\_\_字节指令(因为是16进制，所以一位可以写成4 个2进制位 ，两个16进制数字便是一个字节)，注意观察执行该指令时IP值的变化情况，该指令源操作数DATA的寻址方式是\_\_\_立即数寻址\_\_\_\_\_，其值为\_\_\_\_\_04AE H ( 就是DATA的值 )\_\_。
3. 执行完MOV DS, AX指令后，数据段寄存器DS的内容为\_\_\_\_04AE\_\_\_\_H，源程序在数据段中定义的数据82H、68H和88H被装入的存储单元的物理地址分别为DS:0000 H、DS:0001 H和\_\_DS:0002\_\_H。( -d ds:0 /\*输入该命令\*/ )
4. 程序中第一条ADD AL, [BX] 指令对应的机器代码为\_ 0207 H，它是一个\_\_\_\_2\_\_\_\_字节指令，注意观察执行该指令时IP值的变化情况；该指令中源操作数的寻址方式为\_\_\_\_\_寄存器间接寻址\_\_，该操作数所在存储单元的逻辑地址(DS): (BX)为\_04AE:0001\_，其物理地址为\_\_\_\_\_04AE1\_\_\_\_\_H；执行完该指令后(AL)=\_\_\_EA\_\_\_H，CF=\_\_\_0\_\_\_，OF\_\_\_\_0\_\_，ZF\_\_0\_\_\_\_，SF=\_1\_\_\_，AF=\_\_\_0\_\_\_，PF\_\_0\_\_\_\_；若两操作数为无符号数（看CF），计算结果是否正确\_\_\_\_正确\_\_\_？若两操作数为带符号数（看OF），计算结果是否正确\_\_\_\_正确\_\_\_\_？若计算结果正确，结果是正数还是负数\_\_\_\_负数（看NG是否为1 ）\_\_
5. 执行完第二条ADD AL, [BX] 指令后(AL)=\_\_72\_\_\_\_H，CF=\_\_1\_\_\_\_，OF\_\_\_\_1\_\_，ZF\_\_0\_\_\_\_，SF\_\_0\_\_\_\_，AF\_\_1\_\_\_\_，PF\_\_\_0\_\_\_；若两操作数为无符号数（CF），计算结果是否正确\_\_\_\_错误\_\_\_？若两操作数为带符号数（OF），计算结果是否正确\_\_\_\_错误\_\_\_\_
6. 指令MOV SUM, AL中目的操作数的寻址方式为\_\_\_\_\_立即数寻址不太清楚哦\_\_\_\_\_。该指令执行完后，注意观察(DS):0003H单元中值的变化，该单元的值变为\_\_\_\_72\_\_\_\_H。（三个数加起来就等于72 ，去掉进位）

DATA SEGMENT

NUM DB 82H, 68H, 88H

SUM DB ?

DATA ENDS

CODE SEGMENT

ASSUME CS:CODE, DS:DATA

START: MOV AX, DATA

MOV DS, AX

MOV BX, OFFSET NUM

MOV AL, [BX]

INC BX

ADD AL, [BX]

INC BX

ADD AL, [BX]

MOV SUM, AL

MOV AH, 4CH

INT 21H

CODE ENDS

END START

1. 读懂下列源程序；编辑、汇编、连接生成EX12.ASM、EX12.OBJ、EX12.LST、EX12.EXE文件；使用DEBUG单步执行EX12.EXE文件，并回答以下问题。
2. 代码段的前三条指令MOV AX, STACK、MOV SS, AX和MOV SP, LENGTH STL的功能为堆栈初始化，试问执行完这三条指令后，堆栈栈底所在单元的逻辑地址(SS): (SP)为\_\_04AE:0100\_；执行PUSH AX指令后堆栈指针寄存器(SP)=\_\_00FE\_\_\_\_H，此时，(SS): (SP+1)和(SS): (SP)单元的值为\_\_\_5B0A\_\_\_\_H；执行完PUSH BX后(SP)=\_\_00FC\_\_\_\_H，此时，(SS): (SP+1)和(SS): (SP)单元的值为\_\_\_4CA3\_\_\_\_H；执行完POP AX后(SP)=\_\_\_00FE\_\_\_H；执行完POP BX后(SP)=\_\_0100\_\_\_\_H；此时(AX)=\_\_\_4CA3\_\_\_H，(BX)=\_\_\_5B0A\_\_\_\_H。请读者通过对堆栈操作指令执行情况的观察，进一步掌握进、出栈操作过程。（其实就是利用堆栈进行了两个数的交换操作而已）
3. 执行ADD AX, BX指令后(AX)=\_\_\_\_A7AD\_\_\_\_H，CF=\_\_\_0\_\_\_，OF\_\_\_\_1\_\_，ZF\_\_0\_\_\_\_，SF\_\_1\_\_\_\_，AF\_\_\_0\_\_\_，PF\_\_0\_\_\_\_（注意考虑PF值与计算结果高8位有无关系）；若两操作数为无符号数（CF），计算结果是否正确\_\_\_正确\_\_\_\_？若两操作数为带符号数（OF），计算结果是否正确\_\_\_\_错误\_\_\_\_？
4. 执行AND AX, BX指令后(AX)=\_\_\_\_4802\_\_\_\_H，CF=\_\_\_0\_\_\_，OF\_\_\_0\_\_\_，ZF\_\_\_0\_\_\_，SF\_\_0\_\_\_\_，AF\_\_0\_\_\_\_，PF\_\_\_0\_\_\_。

STACK SEGMENT stack

STL DW 100H DUP(?)

STACK ENDS

CODE SEGMENT

ASSUME CS:CODE,SS:STACK

START: MOV AX, STACK

MOV SS, AX

MOV SP, LENGTH STL

；表达式“LENGTH STL”的功能为计算STL中定义的变量的个数，

；汇编后其值为100H ，详见教材5.2.2

MOV AX, 0101101100001010B

MOV BX, 0100110010100011B

PUSH AX

PUSH BX

ADD AX, BX

POP AX

POP BX

AND AX, BX

MOV AH,4CH

INT 21H

CODE ENDS

END START

1. 指出下列指令的错误原因，上机运行，观察汇编程序(MASM.EXE)给出的出错信息，改正后再上机验证。

(1) MOV BP, BL (2) MOV [BX],[BP]

(3) MOV [BX],20H (4) INC [SI]

(5) ADD AX,[SI+DI] (6) SHL AX,3

(7) PUSH 2000H (8) POP AL

(9) MOV CS,AX (10) MOV DS,1000H

有关常见汇编错误的详细内容见本书第三部分第3章“汇编语言程序常见汇编错误分析”。

warning Operand types must match

Error Improper operand type

error Operand must have size

Error Operand must have size

Error Multiple index registers

error Improper operand type

Error Immediate mode illegal

waring Illegal size for operand

Error Illegal use of CS register

Error Immediate mode illegal

**四、实验报告要求**

1. 写出任务1和任务2中所有问题的正确答案；对实验前做错或做不出来的问题，说明通过上机操作对这些问题有哪些新的认识。
2. 写出任务3中10条错误指令的错误原因，对可以改正的指令给出其正确形式，并分别记录汇编程序对每条错误指令给出的出错信息。
3. 简要说明汇编语言程序设计的步骤，每一步使用什么工具软件，生成什么类型的文件。
4. 写出实验小结，内容包括实验心得（收获）、实验中遇到的问题及解决办法、不足之处或今后应注意的问题等。

/\*

溢出标志OF(Over flow flag) OV(1) NV(0)

方向标志DF(Direction flag) DN(1) UP(0)

中断标志IF(Interrupt flag) EI(1) DI(0)

符号标志SF(Sign flag) NG(1) PL(0)

零标志ZF(Zero flag) ZR(1) NZ(0)

辅助标志AF(Auxiliary carry flag) AC(1) NA(0)

奇偶标志PF(Parity flag) PE(1) PO(0)

进位标志CF(Carry flag) CY(1) NC(0)

AH AL(低八位) 共同组成了AX

OF DF IF SF ZF AF PF CF

1 OV DN EI NG ZR AC PE CY

0 NV UP DI PL NZ NA PO NC

OF 判断带符号数是否运算溢出 1，则溢出

CF 判断无符号数是否运算溢出 1，则溢出

SF 符号标志位 ，最高位为1 时，SF == 1

\* /