微机原理与系统 B　第九周作业　11 月 10 日　周二

PB18151866 龚小航

4.11 如果 MOV ESI, [EAX] 指令出现在工作于 16 位指令模式的 Core2 微处理器中，求它对应的机器语言。

解:【课本 81-85 页】当 80386 及更高型号的微处理器按 16 位指令模式工作时，若使用了 32 位寻址模式，则需要加入地址前缀 67H；而若使用了 32 位寄存器，则需要加入寄存器长度前缀 66H。因此在 32 位微机中，这条指令高十六位一定是 67 66 H

对于低 16 位，按按 16 位指令格式:

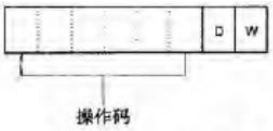

图 4-2　多数机器语言指令的字节 1，显示 D 位和 W 位的位置。

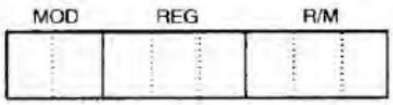

图 4-3　多数机器语言指令的字节 2，显示 MOD，REG 和 R/M 字段的位置

表 4-1　16 位指令模式中的 MOD 字段

MOD	功　能
00	没有位移量
01	8 位符号扩展的位移量
10	16 位有符号的位移量
11	R/M 是寄存器

表 4-5　由 R/M 选择的 32 位寻址方式

R/M 代码	功　能
000	DS:[EAX]
001	DS:[ECX]
010	DS:[EDX]
011	DS:[EBX]
100	使用比例变址字节
101	SS:[EBP]①
110	DS:[ESI]
111	DS:[ESI]

① 参见本书中“特殊寻址方式”一节。

MOV 指令对应 16 位操作码 100010; D 位表示数据流向，此处应有 D = 1，表示数据从 R/M 字段流向 REG 字段; W 段表示传输的是字节(0)或者是字或是双字(1)，此处需要传输的不是字节，因此需要选择 W = 1；MOD 字段规定指令寻址方式，此处使用寄存器间接寻址，没有偏移量，因此 MOD = 00,如左图所示；再查表找到这条指令使用到的 32 位寄存器，可知 ESI 为 110，EAX 为 000

至此，所有需要的信息均已集齐，这条指令在 32 位微机中表示的形式为 67 66 H，100010 11 00 110 000

即为 67 66 8B 30 H

4.21 【课本 88 页】说明 PUSH BX 指令执行时会发生什么操作。假设 SP = 0100H, SS = 0200H，确切指出 BH 和 BL 分别存储在哪个存储单元中。

解:　这条指令将 BX 压入栈，这是两个字节的数据。最高有效位（此处为 BH）存入 SS:SP − 1 指向的存储单元中，此处为 020FF H; 最低有效位(此处为 BL)存入 SS:SP − 2 指向的存储单元中，此处为 020FE H; 完成以上操作后 SP = SP − 2，即 SS:SP 始终指向栈顶字节，此处为 BL

4.25 比较 MOV DI, NUMB 指令和 LEA DI, NUMB 指令的操作。

解：MOV 指令是数据传输指令，它将 NUMB 的内容复制给 DI 寄存器；而 LEA 是取偏移地址的指令，它将 NUMB 这个元素在存储它的段（默认 DS）中的偏移地址存入 DI。

4.43 写一个短程序，用 XLAT 指令将 BCD 码数字 0~9 转换为 ASCII 数字 30H~39H，ASCII 代码存入数据段中的 TABLE 表中。【课本 100 页】

解：参考 101 页的例程，代码如下：

```
ASSUME   CS:CODE, DS:DATA   ;声名代码段和数据段使用的段名

DATA       SEGMENT
TABLE   DB   30H,31H,32H,33H,34H,35H,36H,37H,38H,39H   ;初始化表
BUFFER       DB    ?                                   ;一个字节的写缓冲区
DATA         ENDS

CODE    SEGMENT
START: MOV    AX, DATA
       MOV    DS, AX        ;初始化 DS
       MOV    AH,1          ;INT 21H 的 1 号功能,接受输入并回显
       INT    21H           ;从键盘接收一个字符
       MOV    BUFFER,AL     ;保存到写缓冲区 BUFFER 中

       XOR    AX,AX         ;将 AX 清零，直接对 AL 赋键盘输入的值
       MOV    AL,BUFFER
       SUB    AL, 30H       ;数字的 ASCII 转换为值
       MOV    BX, OFFSET TABLE      ;利用 BX 寻址
       XLAT                         ;此时数据存在 AL 中

       MOV    DL,AL         ;将数据存入 DL，利用 02 号功能打印回屏幕上
       MOV    AH,02H
       INT    21H           ;若输出和输入一样的数字形状，说明成功

CODE   ENDS
END    START
```

【补充题 1】指令 AND AX，7315H AND 0FFH 中，两个 AND 有什么区别？这两个 AND 操作分别在什么时候执行？

解： 这条指令是一条逻辑运算指令，执行寄存器 AX 中的内容与一个立即数进行按位与操作，并将结果存放在 AX 中，其中立即数部分是一个 AND 运算表达式。因此这条指令中第一个 AND 是逻辑运算指令的标识符，它将在 CPU 执行到这条指令的时候运行；第二个 AND 是表达式的运算符，由于两边都是立即数，这个 AND 将在编译阶段由编译器预处理执行。

【补充题 2】 设计指令序列，将字符 $ 送入附加段中偏移地址为 0100H 的连续 10 个单元中。

解： 代码段如下所示：

```
ASSUME   CS:CODE, ES:EDATA   ;声名代码段和附加段使用的段名

EDATA       SEGMENT
BUF     DB 1000 DUP (0)          ;开辟 1000 个字节的附加段空间，初始化 0
EDATA          ENDS

CODE    SEGMENT
START: MOV     AX, EDATA
       MOV     ES, AX            ;初始化 ES
       MOV     BX, 0100H         ;利用 BX 寻址
       MOV     CX, 10            ;存放循环次数
  LOP:
       MOV BYTE PTR ES:[BX], '$'   ;将字节长度的数据存入附加段中
       INC   BX                    ; BX 自增
  LOOP LOP
CODE   ENDS
END    START
```

【补充题 3】设 VAR 是一个 DATE 类型的结构变量。 DATE 结构有 3 个成员：字节变量 MONTH 用于保存月、字节变量 DAY 保存日，字变量 YEAR 用于保存年。要求用伪指令 STRUC 定义该结构体，并给出将 2000 年 1 月 1 日赋值为变量 VAR 的代码段。

解： 利用 STRUC 伪指令定义该结构体：

```
ASSUME  CS:CODE

CODE    SEGMENT
START:
DATE    STRUC
MONTH   DB  ?   ;字节变量
DAY     DB  ?   ;字节变量
YEAR    DW  ?   ;字变量
DATE    ENDS

VAR   DATE  <>  ; 定义结构体 DATE 类型的变量 VAR
MOV BYTE PTR VAR.MONTH, 1
MOV BYTE PTR VAR.DAY, 1
MOV WORD PTR VAR.YEAR, 2000

CODE   ENDS
END    START
```

【补充题 4 】对于指令“MOV AX, DATA”，设 DATA 是段名，代表段基值，属于立即寻址方式。为什么？

解： 由于数据段是在程序编译阶段就确定了其起始地址，用户在汇编程序中引用段名实际上在编译过程中已经被替换成了确定的段基址。比如分配的 DATA 段基址为 0300 H，那么这条指令相当于 MOV AX, 0300 H，当然属于立即寻址方式。