1. 指令、寻址方式、寄存器，到机器码的对应

对比

mov ax, 1

和

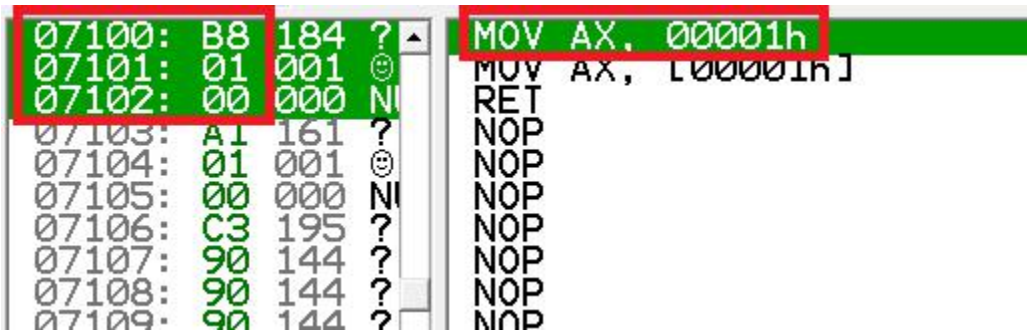
mov ax, [1]

的机器码，会发现二者虽然操作符都是 mov，目标操作数都是 ax，仅源操作数的寻址方式不同，但是机器码的操作码部分却不同。

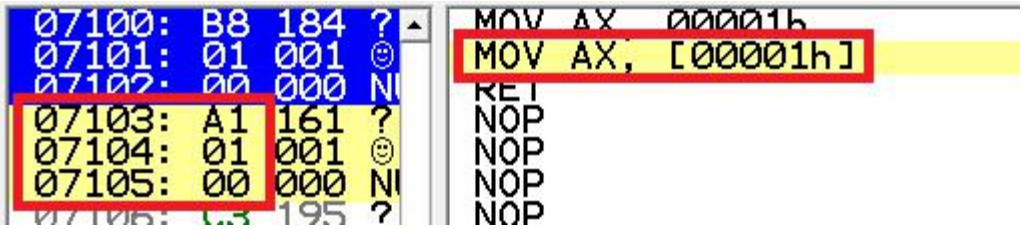
我们使用 emu8086 查看汇编代码对应的机器码。

(1) 寻址方式

下图，红框绿底部分，左侧 B8 是 mov ax, 1 的操作码，01 00 是小端模式的操作数。



下图，红框黄底部分，左侧 A1 是 mov ax,[1]的操作码，01 00 是小端模式的操作数。



同样以 MOV 作为操作符，以 AX 作为目标操作数，机器码的操作码却不同。

为什么会有这种差异呢？因为寻址方式和指令一样，都是影响机器码中操作码的因素。

在 [http://ref.x86asm.net/coder32.html](即本章阅读文献 coder32 edition _ X86 Opcode and Instruction Reference 1.pdf)中，可以查到，

在上述代码中，B8 机器码的含义是把立即数 imm16 移入 AX 寄存器；

	B8+r							MOV	r16/32	imm16/32

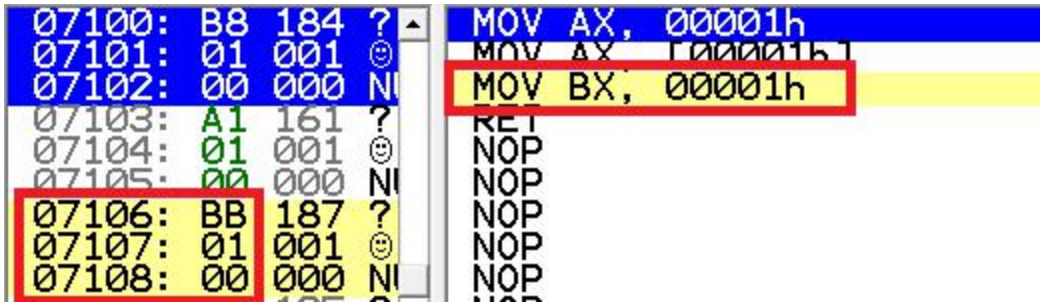
A1 机器码的含义是把存储器偏移量 moffs16 中的值移入 AX 寄存器。

	A1							MOV	eAX	moffs16/32

(2) 目标寄存器

那么，目标寄存器如果不是 AX，而是其他寄存器呢？

在下图中，红框黄底部分，左侧是 mov bx, 1 的操作码。



为什么操作码是 BB(移到 bx)，与 B8(移到 ax)是什么关系呢？
如果没有手册，或者我们通过训练设计实验发现规律，可以连续写出

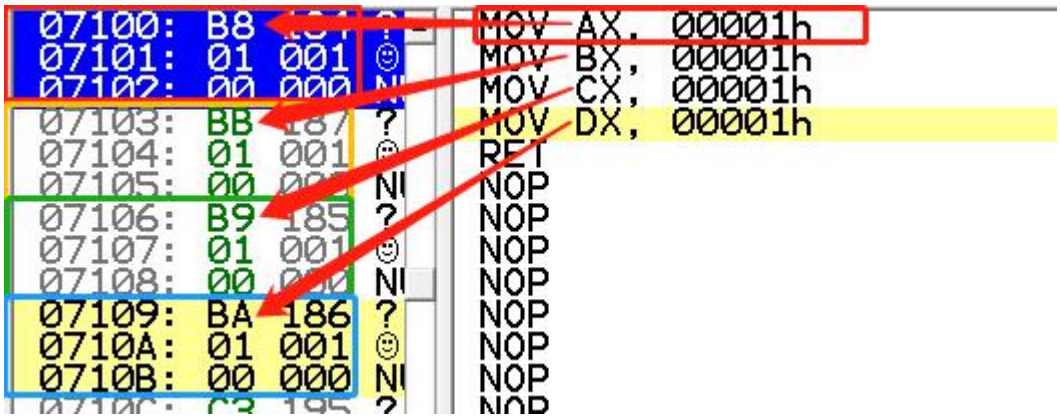
mov ax, 1

mov bx, 1

mov cx, 1

mov dx, 1

仿真运行以后看到机器码。



可以发现规律，是 B8 加上某个值，虽然指令操作符都是 mov，但是不同的机器码数值对应移向不同的目标寄存器。

B8+r	目标操作数，寄存器	机器码
B8+0	AX	B8
B8+1	CX	B9
B8+2	DX	BA
B8+3	BX	BB

(3) 还可以进一步深入底层

以上分析表明，操作数中的目标操作数、操作数的寻址方式都可以影响机器码。这些规则 Intel 有明确的文档说明，作为开始，可以参见 阅读文献 X86_Assembly_Machine_Language_Conversion.pdf 及 Assembly Language & Computer Architecture Lecture (CS 301).pdf。

感兴趣的同学可以自行查阅 x86 汇编和指令系统的资料，进一步深入了解。