《计算机系统设计》平时作业 3

计算机科学与技术 2110939 李颖

1. 假设 R[ax]=FFFAH, R[bx]=FFF0H, 则执行指令”addw %bx, %ax”后，AX、BX 中的内容是什么？标志 CF、OF、ZF、SF 各是什么？要求分别将操作数作为无符号数和带符号整数解释并验证指令执行结果。

add 指令由四条加法指令组成，即 addb(8 位)，addw(16 位)，addl(32 位)，addq(64 位)，指令格式如下所示。这些指令根据需要处理的数据类型的不同，选择不同的指令来执行加法操作，以确保正确的数据处理和结果。

指令	描述	效果
add S, D	D = S + D	加法

【无符号数】：AX 中的内容为 R[bx]+R[ax]=FFF0H+FFFAH=FFEAH，BX 中的内容不变，仍为 FFF0H。由于无符号数真正运算过程为 FFF0H+FFFAH=1FFEAH，产生了进位，同时溢出。因此，进位标志 CF=1，溢出标志 OF=1，零标志 ZF=0，符号标志 SF=1。

验证：16 位无符号整数能表示的范围为 0-65535，而原 AX 中的数 FFFAH 真值为 65530，原 BX 中的数 FFF0H 真值为 65520；而现 AX 中的数 FFEAH 的真值为 65514，并不等于 65520+65530，说明结果发生溢出。

【有符号数】：AX 中的内容为 R[bx]+R[ax]=FFF0H+FFFAH=FFEAH，BX 中的内容不变，仍为 FFF0H。十六进制真正运算过程为 FFF0H+FFFAH=1FFEAH，产生了进位。进位标志 CF=1，溢出标志 OF=0，零标志 ZF=0，符号标志 SF=1。

验证：AX 与 BX 中的数都为负数，将补码转换为源码，得 AX 中原码的真值为-4；BX 中原码的真值为-16。因此，AX 中的新值为-20，转换成补码得 FFEAH。由于 16 位有符号整数的表示范围为-32768~+32767，因此，结果未溢出。

2. 假设 R[eax]=000000B4H，R[ebx]=00000011H, M[000000F8H]=000000A0H，请问：

(1) 执行指令”mulb %bl”后，哪些寄存器的内容会发生变化？是否与执行”imulb %bl”指令所发生的变化一样？为什么？请用该例给出的数据验证你的结论。

在 x86 汇编语言中，mulb 指令用于执行无符号字节乘法操作。其格式和功能描述如下：

指令	描述
mulb source	将 AL 寄存器中的无符号字节与 source 中无符号字节相乘，结果存储在 AX 寄存器中。 由于 mulb 指令执行的是无符号字节乘法，因此结果是无符号数。如果乘积超过了 AX 寄存器的范围（255），则会导致溢出，高于 AX 寄存器的部分会存储在 DX 寄存器中。 如果源操作数是内存地址，mulb 指令会将源操作数加载到寄存器中进行乘法操作。

因此，执行指令后，AX=AL x BL=B4H x 11H=0BF4H。即真值为 180x17=3060。因此，执行”mulb %bl”指令后，EAX 寄存器中的 AX 部分发生了变化。

对于”imulb %bl”指令，由于”imulb”指令用于执行有符号字节乘法操作，因此，将 AL、BL 的值转化为原码，得 AL=CCH，真值为-76；BL 不变，真值为 17。在有符号乘法中，结果的最高位会被符号位填充，AX=FBF4H，真值为(-76)x17=-1297。因此，”mulb”和”imulb”指令的执行结果的 AH 部分不同，”mulb”为 0BH，”imulb”为 FBH。

(2) 执行指令”imull $-16, (%eax, %ebx, 4), %eax ”后哪些寄存器和存储器单元发生了变化？乘积的机器数和真值各是多少？

imull 是 x86 汇编语言中的一条指令，用于执行有符号整数乘法操作。下面是 imull 指令的格式和功能描述：

格式	描述
imull S1, S2, D	将 S1 和 S2 进行有符号整数乘法，结果保存在 D 中。

因此，执行该指令后，首先计算-16 和(%eax, %ebx, 4)的乘积。先对

(%eax, %ebx, 4)进行计算：该表达式的含义为取地址为 eax + ebx*4 的内存单元的值。该地址为 R[eax]+R[ebx]*4=000000B4H + 00000011H*4 = 000000B4H + 00000011H<<2 = 000000F8H。由题意得，000000F8H 地址中存储的数值为 M[000000F8H]=000000A0H。

由此，可以进行有符号整数乘法运算：(-16)*000000A0H。由于该值乘上了一个负数，相当于进行符号扩展后再左移 4 位，因此(-16)*000000A0H = FFFFFFA0H<<4 = FFFFF600H。

综上可知，最终 R[eax]=FFFFF600H。将补码变为原码，得原码为 80000A00H，转换为真值得-2560。