HW2：

第一题

(1) 0xFFFFFFF0+0x00000080=0x00000070未溢出，但有进位, 因此EDX寄存器中的值会变为0x00000070, 同时OF=SF=ZF=0, CF=1
(2) 执行的运算为0x00000010-0x80000008=0x80000008, 有溢出，有借位，因此ECX寄存器中的值保持不变，OF=CF=SF=1, ZF=0
(3) 执行的运算为0xFF00 | 0x0100 = 0xFF00, 因此EBX寄存器中的值会变为0x0000FF10，OF=CF=ZF=0, SF=1
(4) 执行的实际运算为0x80 & 0x80 = 0x80, 因此内存和寄存器中的值均不会发生变化，OF=CF=ZF=0, SF=1
(5) 执行的实际运算为0x908F12A8 * HEX(32) = 0x11E25500（截断为32位）同时由布斯乘法知出现溢出，因此ECX寄存器中的值被设置为0x11E25500，OF=CF=1。SF和ZF的取值在该指令中是未定义的。
(6) 执行的指令实际上是将BX寄存器和AX寄存器相乘，得到的32位乘积的高16为放在DX中，低16位放在AX中。由于0x9300*0x0100=0x00930000（32位结果），因此AX寄存器被修改为0x0000，DX寄存器被修改为0x0093，同时由于高16位不是全0，因此最终结果溢出，OF=CF=1，SF和ZF的取值在该指令中是未定义的。
(7) 执行的实际运算为0x0010--=0x000F，同时未出现溢出和进位，因此ECX寄存器中的值被设置为0x0000000F，OF=SF=ZF=0。由于指令不影响CF，因此CF保持旧值不变。

第二题

- 注释如下

```
movb    8(%ebp), %dl          // 将x的值移动到dl寄存器中
movl    12(%ebp), %eax        // 将指针p的值移动到eax寄存器中
testl   %eax, %eax            // 判断p的值是否为0，并设置标志位ZF
je      .L1                   // 如果ZF=1(p为0)，则跳转到.L1，退出函数
testb   $ox80, %dl            // 判断x的符号位是否为1（即判断x的正负性），并设置标志位ZF
je      .L1                   // 如果ZF=1(x为负数)，则跳转到.L1，退出函数
addb    %dl, (%eax)           // 否则，将eax中的地址所指向的数加上x
.L1:
```

因为if的条件判断中有两条子条件，必须在两条子条件都满足时才能执行addb %dl, (%eax)

- 代码如下

```
void comp(char x, int* p) {
    if (p==0)
        goto done;
    if (x<0)
        goto done;
    *p += x;
done: ;
    return;
}
```

第三题