值。实际上，定义 EXPR(x)为 fabs(x)就能得到这段代码，fabs 是在< math.h> 中定义的。

B. 可以看到 vxorpd 指令将整个寄存器设置为 0，所以这是一种产生浮点常数 0.0 的方法。

C. 可以看到从地址.LC2 开始的 16 个字节是一个掩码，它只有一个 1 位，位于 XMM 寄存器中低位数值的符号位。计算这个掩码与%xmm0 的 EXCLUSIVE－OR 值时，会改变 x 符号的值，计算出表达式-x。

3.57　同样地，为代码加注释，包括处理条件分支：

```
    double funct3(int *ap, double b, long c, float *dp)
    ap in %rdi, b in %xmm0, c in %rsi, dp in %rdx
1   funct3:
2     vmovss  (%rdx), %xmm1                    Get d = *dp
3     vcvtsi2sd       (%rdi), %xmm2, %xmm2     Get a = *ap and convert to double
4     vucomisd        %xmm2, %xmm0             Compare b:a
5     jbe     .L8                              If <=, goto lesseq
6     vcvtsi2ssq      %rsi, %xmm0, %xmm0       Convert c to float
7     vmulss  %xmm1, %xmm0, %xmm1              Multiply by d
8     vunpcklps       %xmm1, %xmm1, %xmm1
9     vcvtps2pd       %xmm1, %xmm0             Convert to double
10    ret                                      Return
11  .L8:                                     lesseq:
12    vaddss  %xmm1, %xmm1, %xmm1              Compute d+d = 2.0 * d
13    vcvtsi2ssq      %rsi, %xmm0, %xmm0       Convert c to float
14    vaddss  %xmm1, %xmm0, %xmm0              Compute c + 2*d
15    vunpcklps       %xmm0, %xmm0, %xmm0
16    vcvtps2pd       %xmm0, %xmm0             Convert to double
17    ret                                      Return
```

由此，可以写出 funct3 的代码如下：

```
double funct3(int *ap, double b, long c, float *dp) {
    int a = *ap;
    float d = *dp;
    if (a < b)
        return c*d;
    else
        return c+2*d;
}
```