3. 1) nop : addi X_0, X_0, 0.
2) ret : jalr X_0, 0(x1)
3) calloffset : auipc x1, offset[31:12] + offset
4) mv rd, rs : addi rd, rs, 0.
5) rdcycle rd : csrrs rd, cycle, x0.

7. 2) blt t0, t1, overflow.
3) 方法一：采用一位符号位，设A的符号为A_s，B的符号为B_s，运算结果的符号为S_s，则溢出逻辑表达式为 $V = A_s B_s \overline{S_s} + \overline{A_s} \overline{B_s} S_s$
若 V=0 无溢出，V=1，有溢出。
方法二：采用一号符号位，根据数据位进位情况判断溢出符号位的进位C_s，最高数值位的进位C_1。

	C_s	C_1
上溢	0	1
下溢	1	0

8. 1) 2^{XLEN-1} , X, -1, X. 不会引起。假如触发异常，会导致一个自陷，需要语言实现者与执行环境的自陷处理函数交互，将增加性能开销。

2) NV 非法操作，DZ 除以0，OF 上溢，UF 下溢，NX 不精确。
不会使得处理器陷入系统调用。
3) x86：CPU 做何于抛出异常中断，例如除以0或取消引用NULL指针。这些中断被捕获，比如硬件中断时，停止当前程序的执行并将控制权返回给操作系统，然后操作系统处理事件。虽然这些操作非常依赖于环境，但通常程序可能会被终止，释放所有的资源。

12~17

12. 1) S 2) M 3) M 4) S 5) V.

17. $a_1 = 1 + 30 \times 2 = 61$