

3. 1> addi x0, x0, 0

2> jalr x0, x1, 0

3> auipc x1, offset[31:12] jalr x1, x1, offset[11:0]

4> add rd, rs, x0

5> csrwr rd, cycle[h], x0.

6> addiw rd, rs, 0

7. 1> 解: subi t3, t2, 0

sub t4, t0, t1

2> 解: add t0, t1, t2

bltu t0, t1, overflow.

3> 解: x86 利用硬件电路检测溢出

ARM 通过 CPSR 的状态寄存器反映当前指令的溢出状态

8. 1> 解: 表格: 2XLEN-1 X -1 X

会. 因为这些异常在大多数执行环境中会导致陷阱.

(2) 解: NV: 无效操作

不会.

DZ: 除以 0

OF: 溢出

UF: 下溢

NX: 不精确的

3) 解: x86 中断并调用软件处理中断.

12 1) 管理员模式

2) 管理员模式

3) 机器模式

4) 管理员模式

5) 用户模式

13 解: vecMul: addi sp, sp, -32
sd ra, 24, sp
sd s0, 16, sp
addi s0, sp, 32
add a5, x0, x0
addi a6, x0, 100
Loop: bge a5, a6, end.

sl, a7, a5, 2
add a3, a7, t0
add a4, a7, t1
lw a3, 0(a3)
lw a4, 0(a4)
mul a3, a4, t2
addi a5, a5, 1
j loop
end: mv a0, a3
ld ra, 24(sp)
ldl s0, 16(sp)
addi sp, sp, 32
ret

14. 解: part1: bge a1, a0, part2.
add a2, a1, a0.
j end.

part2: sub a2, a0, a1.

15 解: lw a3, 0(t0) $\left\{ \begin{array}{l} \text{add a3, x0, t0} \\ \text{addi t1, x0, 3} \\ \text{addi a4, x0, 1} \\ \text{sll a4, a4, 2} \\ \text{sll a5, a5, t1} \\ \text{lw a6, (a4+t0)} \\ \text{lw a7, (a5+t0)} \\ \text{add a6, x0, t1} \\ \text{add a7, x0, t1} \end{array} \right.$

16. 解: swap: addi sp, sp, -32
sd ra, 24, sp.
sd s0, 16, sp
addi s0, sp, 32
lw a3, 0(t0)
mv a4, a3.
lw a5, 0(t1)
mv a3, a5.
mv a5, a4

ret

17. 解: 计算 2^{30} 并保存在寄存器 al 中.