2.3 (1) nop = addi x0, x0, 0

(2) ret = jalr x0, x1, 0

(3) call offset = auipc x1, offset[31:12]
jalr x1, x1, offset[11:0]

(4) mv rd, rs = addi rd, rs, 0

(5) rdcycle rd = csrrs rd, cycle[h], x0

(6) sext.w rd, rs = addiw rd, rs, 0

2.7 (1) SLTi t3, t2, 0
SLT t4, t0, t1

(2) add t0, t1 t2
blt t0, t1, overflow

(3) x86提供了JO和JNO进行跳转
arm有CV标志位提供检测

2.8 (1)没有异常. rd值如下.

$2^{XLEN}-1$ X -1 X

XLEN为寄存器位数

(2) NV: 非法操作 DZ: 除以0 OF: 上溢 UF: 下溢 NX: 不精确用
RISCV不会陷入系统级调用

(3) x86会产生中断类型码为0的中断
arm当CCR中DIV_0_TRP寄存器控制是否触发异常

2.12 (1) S-Mode

(2) M-Mode

(3) M-Mode

(4) U-Mode

(5) U-Mode

2.13

```
mv a4, a0        // 将a地址存到a4
addi a3, a1, 404 // 循环终止地址
1:
    lw a5, 0(a1)
    addi, a4, a4, 4
    addi a1, a1, 4
    mul a5, a5, a2
    sw a5, -4(a4)
    bne a1 a3 1b
    lw a0, 0(a6)
    ret
```

2.14

```
    blt a1, a0 1f
    sub a2 a1 a0
    j 2f
1:
    add a2 a1 a0
2:
```

nop

2.15 sw t0, 0(t0)

li t1, 3

sw t1, 4(t0)

slli t2, t1, 2 (乘4得到偏移量)

add t2, t2, t0 (偏移加绝对地址)

sw t1, 0(t2)

2.16 lw t2, t0

lw t3, t1

sw t3, t0

sw t2, t1

2.17 addi a0, x0, 0 (将a0置0)

addi a1, x0, 1 (将a1置1)

addi a2, x0, 30 (将a2置30)

loop: beq a0, a2 done (当a0=a2时退出循环)

slli a1, a1, 1 (将a1乘2)

addi a0, a0, 1 (将a0加1)

j loop (跳转)

done (程序退出)

这个等价将a1左移30次

等价 slli a1, a1, 30