

3. 1) `nop` \Leftrightarrow `addi x0, x0, 0` 空指令
 2) `ret` \Leftrightarrow `jalr x0, x1, 0`
 3) `call offset` \Leftrightarrow `auipc x6, offset[31:12]`
 `jalr x1, x6, offset[11:0]`
 4) `mv rd, rs` \Leftrightarrow `addi rd, rs, 0`
 5) `rdcycle rd` \Leftrightarrow `csrrs rd, cycle, x0`
 6) `sxt.w rd, rs` \Leftrightarrow `addiw rd, rs, 0`

7. 1) `sub t3, t0, t1`
 `mv t4, t2`
 2) `add. t0, t1, t2`
 `slt t3, t0, t1`
 `bne t3, x0, overflow.`

- 3) x86中采用OF标志位, 如果执行加法指令后OF为1, 则溢出.
 ARM中采用C和V标志位检测. C标志指示无符号加法是否溢出,
 V标志有符号加法是否溢出.

- 8 (1). `OP = DIV` `rd = 0xFFFFFFFFFFFFFFFF`
 `OP = REMV` `rd = 0`
 `OP = DIV` `rd = 0xFFFFFFFFFFFFFFFF`
 `OP = REM` `rd = 0`

为避免程序错误执行除以0操作, 通过检测引发异常来终止程序.

(2). NV \rightarrow 发生不支持操作.

DZ \rightarrow 发生某种浮点非数值情况.

OF \rightarrow 计算结果溢出

当 $fflags$ 被置位, 指令跳转后

VF \rightarrow 计算结果下溢.

使处理器进入系统调用状态.

NX \rightarrow 执行无效操作

(3). x86中, 整数除法使用 $idiv$ 指令, 被除数有符号将相应寄存器第
标志位置 1, ARM中, 也置 1. 由 $VDIV$ 和 $SDIV$ 指令执行.

12. (1). 管理员 Mode

(2). 机器模式

(3). 管理员 Mode

(4). 管理员模式

(5). 用户模式

13. $addi\ a2, x_0, 1$

$addi\ a1, x_0, 100$

loop: $bge\ a2, a3, exit.$

$slli\ a4, a2, 2$

$add\ a5, a4, t_0.$

$add\ a6, a4, t_1$

$lw\ a5, 0(a5)$

$lw\ a6, 0(a6).$

$mul\ a8, a6, t_2.$

$sw\ a8, 0(a5)$

$addi\ a2, a2, 1$

$j\ loop.$

exit: $lw\ t_0, 0(t_0)$

ret

14. $bgt\ a_0, a_1, big.$

$j\ small$

$big: add\ a2, a_0, a_1$

$small: sub\ a2, a_0, a_1$

15. $sw\ t_0, 0(t_0)$

$addi\ t_1, x_0, 3$

$sw\ t_1, 4(t_0).$

$sw\ t_1, 12(t_0).$

16. $mv\ t_2, t_0$

$mv\ t_0, t_1$

$mv\ t_1, t_2$

17. 每次 loop 将 a_1 左移 1 位, 共移 30 位 则 $a_1 = 2^{30}.$