

理论课作业: 3, 7, 8, 12, 13, 14, 15, 16, 17

3. 1. nop: 等价于 `addi x0, x0, 0`

2. ret: 等价于 `jalr x0, x1, 0`

3. call offset: 等价于 `auipc x1, offset [31:12] + 1; jalr x0, x1, offset`

4. mv rd, rs: 等价于 ~~`rdrtime rd, csrr rd, time`~~ `add rd, x0, rs`

5. rdcycle rd: 等价于 `rdrtime rd, csrr rd, time`

6. sext.w rd, rs: 等价于 `addiw rd, rs, 0`

7. (1) `add t0, t1, t2`

`sub t3, t0, t1`

`xor t4, t1, t2`

`and t4, t4, t1`

`bne t4, x0, overflow`

(2) `sltu t3, t0, t1`

`sltu t4, t1, t2`

`or t3, t3, t4`

`bne t3, x0, overflow`

(3) 在 x86 架构中, 有专门的指令用于检测加法溢出, 如 `j0 (jump on overflow)` 指令, 它会在发生溢出时跳转到指定

位置。在ARM架构中,有类似的指令,如bvs (branch on overflow) 指令。同时,这些指令也可以用于检测其他运算的溢出。

8.13) x86架构处理除数为0的情况时,会触发一个异常,即“除法错误异常 (HPE)”,CPU会将异常码放入异常向量表中的相应入口,并跳转到该入口处理异常。处理程序可以采取不同的措施,如中止程序、输出错误信息等。ARM架构处理除数为0的情况时,会触发一个异常,即“未定义指令异常 (Undefined Instruction exception)”,CPU会将异常码放入异常向量表中的相应入口,并跳转到该入口处理异常。处理程序可以采取不同的措施,如中止程序、输出错误信息等。

12. 1. Linux kernel - 特权等级 1 或 3 (取决于系统配置)

2. BootROM - 特权等级 3

3. BootLoader - 特权等级 3

4. USB Driver - 特权等级 1 或 3 (取决于系统配置)

5.  $VIM = M / 100$

13: globl

recMul

recMul: addi sp, sp, -16

swra, 0(sp)

sw b0, 4(sp)

sw t1, 8(sp)

sw t2, 12(sp)

li t3, 0

loop: bge t3, 100, end

lw t4, 0(t1)

lw t5, 0(t2)

multo, t4, t5

lw t7, 0(t0)

sw t6, 0(t7)

addi t0, t0, 4

addi t1, t1, 4

addi t3, t3, 1

j loop

end:

lw t0, 4(sp)

lw ra, 0(sp)

lw t1, 8(sp)

lw t2, 12(sp)

lw t3, 0(t0)

addi sp, sp, 16

jr ra

14.

lw a0, 0(sp)

lw a1, 4(sp)

bgt a0, a1, greater-than

sub a2, a0, a1

j end

greater-than

sub a2, a0, a1

end:

sw a2, 8(sp)

15

16. swap:

```
lw t2, 0(t0)    # load tmp = *a
lw t3, 0(t1)    # load *a = *b
sw t3, 0(t0)    # store *b to *a
sw t2, 0(t1)    # store tmp to *b
jr ra           # return
```

17. 这段 RISC-V 汇编代码实现的功能是将寄存器 a1 的值不断左移一位, 直到寄存器 a0 的值等于 30 为止。具体实现过程为先将寄存器 a0 的值设为 0, 寄存器 a1 的值设为 1, 然后进入循环。循环中, 判断寄存器 a0 的值是否等于 30。如果是则跳到 done 标签处结束循环, 否则将寄存器 a1 的值左移一位并将寄存器 a0 的值加 1, 然后继续循环。最终, 寄存器 a1 中存储的就是 2 的 30 次方。