Date.

嵌入式作业第5周

3. 写出伪指令等价的基本指令或指令组合.

(1) nop （空操作指令）

addi x0, x0, 0

(2) ret (3) call offset (4) mv, rd, rs

(5) rdcycle rd (6) sext.w rd, rs

(2) jalr x0, 0(x1)，用于从函数中返回. 等价于 将x0（零寄存器）与x1（链接寄存器）中存储的地址相加，并将结果存回x0，同时将程序计数器设置为x1中存储的地址.

(3) 等价于 jal ra, offset，用于调用一个子程序，它将ra（返回地址寄存器）与当前程序计数器中的地址相加，并将结果存回ra，同时将程序计数器设置为 offset. 它将导致程序转到偏移量处的地址，继续执行子程序.

(4) 等价于 addi rd, rs, 0.

(5) 等价于 csrrs rd, cycle. cycle是一个特殊寄存器的名称，包含CPU周期计数器的当前值，存入指定的目标寄存器中. 访问CSR（控制和状态寄存器）.

(6) 等价于 slli rd, rs, 0; srai, rd, rd, 0. 将rs的值左移0位，再右移0位，将导致rd中的值等于rs的值，但是是符号扩展的.

7. 解：(1) t_1 和 t_2 都是有符号数：

```
add t0, t1, t2
slti t3, t2, 0      # 溢出情况：t2>0，但 t0<t1；t2<0，但 t0>t1
slt t4, t0, t1
bne t3, t4, overflow
```

(2)

```
add t0, t1, t2
bltu t0, t2, overflow      # 溢出情况：t0<t1，且 t0<t2
                           正常：t0>t1且 t0>t2
```

(3) x86架构中，有符号数加法可以使用add和adc指令来实现，adc指令将进位标志CF（Carry Flag）也加入到操作数中，然后使用jno指令来检测溢出。无符号数加法可以直接使用add指令，然后用jc指令检测进位标志CF.

ARM架构中，有符号数可使用add和sbc指令来实现，sbc指令也会将进位标志C（carry）加入到操作数中，然后用bvs指令检测溢出. 无符号数加法可直接用add指令，然后用bcs指令来检测进位标志C.

是具体的数十，是异常吗？

8. 1) op = DIVU时，rd = 0xfff… 即抛出异常
   op = DIV时，rd = 0xff…
   op = REM时，rd = x
   op = REMU时，rd = x.
   分析：提示程序员或调试人员程序中存在的错误.

2) fcsr控制寄存器中的fflags字段是用来表示浮点运算中的异常情况的标志位.
   NV：表示无效操作，比如除零、平方根负数等. (Invalid operation)
   DZ：表示操作数为非规格化数，即指数位为全0的浮点数。(Denormalized operand)
   OF (overflow)：操作结果溢出了，
   UF (underflow)：下溢，即小于浮点数所能表示的最小值.
   NX (Inexact)：表示操作结果无法精确表示，即最终结果无能保证精度.
   当浮点除法的除数为0时，会导致无效操作NV被置位，fflags被置位，处理器不会陷入系统调用，而是将控制权交给应用程序，由应用程序来处理异常情况：可中断应用程序，或通过一些异常处理机制.

3) 在x86：设置异常标志，去引发一个异常，将异常状态保存在处理器状态寄存器的特定位中，即置位零除错误标志. 此外，还将除数为0的指令地址放入异常返回指令指针寄存器中，以便程序在异常处理程序中返回正确的位置. 接后一页→

12. 写出以下程序在RISC-V中应当处于的特权等级
解：1) Linux kernel：
处于特权等级1或更高级别，因为它需要访问操作系统内核的各种功能和资源.
2) BootROM：特权等级0. 它是系统启动时运行的，是上电后最先运行的程序，是系统启动的第一阶段，只能访问部分寄存器和内存空间.
3) BootLoader：特权等级1或更高，它也需要访问操作系统内核的各种功能和资源. 是引导加载程序，硬件会从硬盘等读取Bootloader，将其加载到内存中，并将控制权转移给它. 负责加载操作系统内核并启动操作系统.
4) USB Driver：特权等级1或更高：需要访问硬件设备，需要在特权模式下进行. USB驱动程序需要与内核交互，以便将数据传递给应用程序.
5) Vim：用户级别，是一个文本编辑器，没有必要访问特权模式下的资源.
USB Driver是一种软件程序，允许计算机操作系统与USB设备进行通信。每个USB设备都需要特定的驱动程序才能在计算机上正确运行。

13. 解：(返回值在于a0寄存器中)
(t0, t1, t2分别为B[0], B[0], C的地址)

```
add  t3, x0, x0  # i(初始值为0)
addi t4, x0, 100 # 100
lw   t5, 0(t2)   # C的值
loop:
    bge  t3, t4, exit
    slli t0, t3, 2 # 4i
    add  t0, t1, t0 # B[i]的地址
    lw   t0, 0(t0) # B[i]的值
    slli t1, t3, 2 # 4i
    add  t1, t0, t1 # A[i]的地址
    mul  t0, t0, t5 # B[i]*C值
    sw   t0, 0(t1) # B[i]放地址里去
exit:
    lw a0, 0(t0)
part1:
14. blt  a1, a0, part2 # a>b跳
    sub a2, a0, a1
    j exit
part 2:
    add a2, a0, a1 # a>b, t0
    j exit
exit:
    ret
```

p地址(指针)在t0, a值在t1

15.
```
sw   t0, 0(t0)  # p[0]=p
addi t1, x0, 3  # a值3
sw   t1, 4(t0)  # p[1]=a
sw   t1, 12(t0) # p[3]=a
```

16.
```
lw t2, 0(t0) # t2为a值
lw t3, 0(t1) # t3为b值
sw t3, 0(t0)
sw t2, 0(t1) # 值互换
```

17. 功能：计算2^{30}的值，并将结果保存在寄存器a1中.

18. 同除数为0时引发的异常可被应用程序或操作系统处理.

19. ARM：类似，处理器产生异常并设置特定的标志位，不同的是，ARM中的异常处理程序是通过向量表来实现的，当异常发生时，处理器会跳转到对应的异常处理程序入口地址。