

## 第4周作业

1. CISC: 优势: 实现相同操作所需的指令数少, 指令类型丰富, 操作灵活, 对编译器和程序存储空间的要求较低

劣势: 硬件设计复杂, 测试验证难度较高

RISC: 优势: 硬件设计较为简单, 适合利用流水线提升性能, 硬件开发周期更短

劣势: 对编译器设计的要求较高, 程序的代码密度较低, 在指令灵活性上受到一些限制

## 2 RISC-V 属于寄存器-寄存器型指令集

1. 寄存器寻址: 在操作数已经加载到寄存器中使用, 读取源操作数寄存器中的数, 操作后写到目的寄存器中

2. 立即数寻址: 当某个操作数为常数时使用, 读取立即数及源操作数寄存器中的数, 操作后写到目的寄存器中

3. 偏移量寻址: 访问局部变量时使用, 寄存器内存放的操作数的地址, 读取此地址, 加上一个偏移量得到新地址, 读取此地址指向的操作数, 操作后写到目的寄存器中

4. 寄存器间接寻址: 访问指针内容或计算的地址时使用, 操作与偏移量寻址类似, 不同点在于偏移量此时取0

5. 直接寻址: 可以在访问静态数据时使用, 直接通过地址读取其指向的静态数据, 操作后写到目的寄存器中

4 1) 不相同, 不相同. `addw`的操作具体是将2个64位寄存器中的数相加后, 将结果截断为32位, 把符号位扩展的结果写入并忽略算术溢出, `RV64I`中`add`的操作数为64位, 而`RV32I`中`add`操作数为32位, 因此不同. 由于32位字仍然是程序中的有效





数据类型,因此 RV64I 需要支持字,存在截断操作。

(2) 不需要, RV64I 在将数据截断为 32 位后,又将结果符号扩展后再写入目标寄存器,寄存器为 64 位宽,其宽度决定了指令运算操作后的结果长度。

5 是指 RISC-V 为 HINT 指令保留的编码空间。HINT 指令指提示指令,通常用于向微架构传达性能提示,并且与 NOP 相似,除推动 PC 以及任何可用性能计数器外,并不改变任何体系结构可见的状态。具体实现可以选择忽略这些提示编码。这些 HINT 的编码在选择上是为了方便让简单的实现能够完全忽略 HINT: 将 HINT 作为一条普通且不会改变体系结构状态的指令来执行。

6  $a_2$  为 -3,  $a_3$  为 1, 除法指令为 mul, 默认为有符号数-有符号数运算, 包括 s 为有符号数-无符号数运算, 包括 u 为无符号数-无符号数运算, 实现低 32 位运算和对输出结果进行高位截取(包括 h), 取余指令为 rem, 默认为有符号数-有符号数取余, 结果与被除数符号相同, 包括 u 表示无符号数-无符号数取余, 包括 w 为实现 32 位取余。

11 (1) 直接寻址 (2) 寄存器间接寻址 (3) 立即数寻址 (4) 寄存器寻址  
(5) 偏移量寻址

