

Chapter 2 Homework (3)

1. 简要分析CISC和RISC架构各自的优势和劣势。

答: CISC: 优点: 对编译器和程序存储空间的要求较低。
缺点: 硬件设计复杂, 测试验证难度较高。

RISC: 优点: 硬件设计较为简单, 适合流水线提升性能。
缺点: 对编译器设计的要求较高, 程序的代码密度较低。

2. RISC-V的基本指令集是什么? 列举五个常见的RISC-V标准扩展指令集并简要说明它们的作用; 应用范围。

答: 其基本指令集为(根据寄存器大小)的RV32I, RV32E 和 RV64I 整数指令。

列举: (1) M 指令集, 扩展了整数乘法和除法指令 (2) F 指令集, 扩展了 IEEE 标准单精度浮点数运算指令, 增加了 32 个 32 位浮点寄存器。(3) C 指令集, 定义了部分指令的 16 位版本, 用于小内存的嵌入式应用 (4) B 指令集, 扩展了位操作指令。(5) T 指令集, 扩展了事务性内存指令。

3. 4. 阅读 RISC-V 规范以回答以下问题:

1) RV32I 中的 add 指令和 RV64I 中的 addw 指令均为 32 位整形加法指令, 它们是否具有相同的指令操作数? 此外, RV32I 中的 add 指令和 RV64I 中的 add 指令是否具有相同的指令操作数? 试分析为什么采取这样的设计。

答: add 的指令操作数为 0110011 而 addw 为 0111011, 两者不相同。而 add 指令有相同操作数。

分析: RV64I 是由 RV32I 扩展而来, 只是增大了数据容量, 但操作相同, 故 add 操作码同, 但 addw 是运算后截断 32 位, 而非在计算时对源操作数, 从结果上 add 和 addw 都可能存在差异, 操作码当然会设计的不同。

5. 什么是 RISC-V 的 I 标准指令集中存在的 HINT 指令空间? 它有什么作用?

答: RV32I 为 HINT 指令保留了很大一片编码空间, HINT 指令通常用于向微架构传达性能提示, 并且就像 NOP 一样, 除推动 PC 以及往回可用性能计数器外, 并不改变体系结构可见状态。

6. 考虑如下指令序列：

div a2.a0.a1

rem a3.a0.a1

假设寄存器 a2 和 a1 的初始值分别为 16 和 -5，问上述指令序列执行完成后 a2 和 a3 寄存器中的值分别是多少？简要说明 RISC-V 的 M 标准指令集中对除法和余数指令的符号规定。

答： $a_2 = -3$, $a_3 = 1$; 除法规定对结果进行四舍五入, 余数则保持与被除数符号相同。

(For both signed and unsigned division, it holds that dividend = divisor \times quotient + remainder).

11. 写出以下指令的使用的寻址模式。

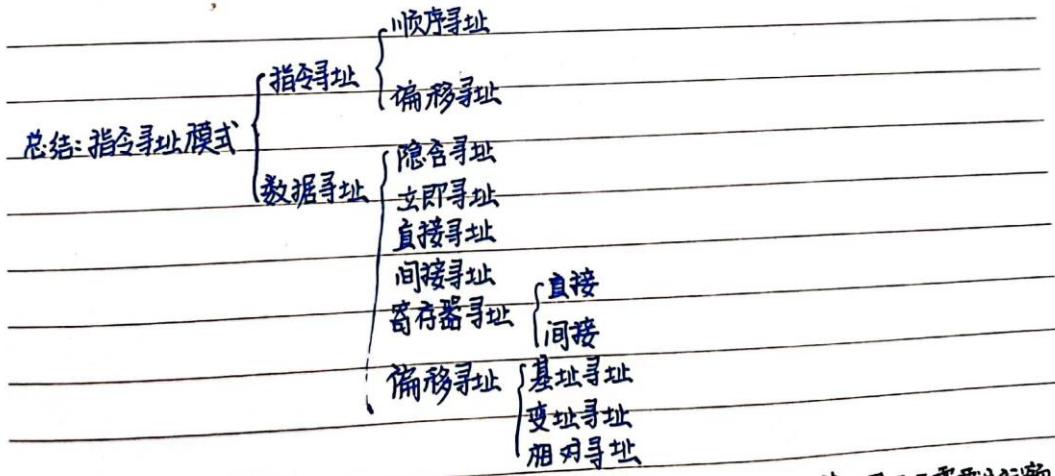
1) jal ra, 0x88 偏移量寻址

2) jalr x0, ra, 0 寄存器间接寻址 × 偏移量寻址

3) addi a0, a1, 4 立即数寻址

4) mul a0, a1, a2 寄存器寻址

5) ld a4, 16(SP) 偏移量寻址



补充 4.2) 在 RV64I 中, addw 和 addiw 指令的目标寄存器中存放的 32 位计算结果是否需要进行额外的符号扩展才能用于后续 64 位计算？请说明理由。

无需，理由 ADDW 和 ADDIW 指令带有符号位无端额外扩展。