

1. CISC: 优势: (1) 单个指令完成的任务量大, 对编译器和程序存储空间要求低

(2) 指令长度灵活, 寻址方式灵活

缺点: (1) 指令的使用率不均衡

(2) 硬件设计复杂, 测试验证的难度高, 不利于采用先进结构改进性能

RISC: 优势: (1) 硬件设计简单, 适合利用流水线提升性能

(2) 指令精简, 且使用率均衡

缺点: (1) 对编译器设计要求高; (2) 程序代码密度较低.

2. RISC-V 中的基本指令集是 整数指令集 (用 I 表示), 根据寄存器位宽的大小分为 32I, 64I, 128I 三种整数指令集. 而每一种中又有六种基本指令类型, 如: R-type, I-type, S-type, B-type, U-type, J-type.

RISC-V 标准扩展指令集: M: 乘除法, 取模, 指令余指令

F: 单精度浮点指令

D: 双精度浮点指令

A: 原子操作指令, 例令 CAS 和 LL/SC 指令等

C: 压缩指令, 主要用于改善程序大小

4. 1) RV32I 中 add 指令的 opcode 为 011001, RV64I 中 addw 指令的 opcode 为 011101, 所以它们的 opcode 不同

RV32I 中 add 指令和 RV64I 中 add 指令的 opcode 都为 011001。

原因: RV64I 是 RV32I 的超集, 包括 RV32I 的所有 40 条指令, 另增加 12 条指令. RV32I add 和 RV64I 的 addw 实现的虽都是 32 位数加法, 但 add 处理 32 位 imm 的全部位数, addw 处理的是 64 位 imm 的低 32 位, 而 RV64I 的 add 也是处理整个 imm 的全部位数, 操作相同. 所以 opcode 相同.

2) 不需要; RV64I 中的 addw 和 addiw 指令的目标寄存器中存放的 32 位

计算结果已经是加法计算后经过符号扩展后的结果。

5. RISC-V的I标准指令集中存在的HINT指令又称为提示指令,通常向微架构传达性能提示,除推动PC及任何可用性能计数器外,并不改变任何系统结构可见的状态,如X86的NDP指令一样,仅为空操作。

多数RV32I HINTs都被编码为 $rd=x0$ 的整数计算指令,如ANDI指令,令 $rd=x0$,则为HINT指令,32位减去7位opcode,5位 $rd=x0$ 和3位funct3,剩下17位,编码位点为 2^{17} ,加上其余HINT指令的编码位点,就构成了RV32I的HINT空间。

6. $a2$ 寄存器中的值为-3, $a3$ 寄存器中的值为1

div除法指令为有符号数间的除法,在执行操作前要将 $rs1$, $rs2$ 中的数进行有符号扩展;

rem余数指令为有符号数间的取余,执行前要对 $rs1$, $rs2$ 进行有符号扩展,一般来说余数为正值。

11. 1) 偏移量寻址

2) 立即数寻址

3) 立即数寻址

4) 寄存器直接寻址

5) 偏移量寻址