

1.简要分析 CISC 和 RISC 架构各自的优势和劣势。

CISC:

优点是实现相同操作所需的指令数少,指令类型丰富,操作灵活(可以实现多种复杂的操作和功能,比如乘法和除法等),对编译器和程序存储空间的要求较低,同时也易于编程:

缺点是指令的利用率很不均衡;译码困难,不利于处理器流水线的分割,增加了高性能处理器的设计难度,测试验证难度较高,易出错;CISC 处理器需要更多的功耗来支持繁多的指令集,能耗大。

RISC

优点是指令格式统一,类型简单,硬件开发周期可以更短,适合利用流水线提升性能

缺点是指令灵活度弱于 CISC,实现相同操作所需的指令数可能更多,对编译器设计要求较高,程序的代码密度较低

2.RISC-V 中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和应用范围。

RISC-V 中的基本指令集: R 类型指令、I 类型指令、S 类型指令、B 类型指令、U 类型指令、J 类型指令

RISC-V 标准扩展指令集有 RV64I-基本整数指令集,适用于大多数人常见的计算机操作、RV64F-单精度浮点指令集适用于需要处理浮点数的计算机系统如图像处理、科学计算等、RV64D-双精度浮点指令集,适用于需要高精度的计算、RV64M-乘、除和取模指令集,适用于需要高效处理大量数据的应用,如编解码、加密解密、RV64Zicsr-控制和状态寄存器指令集扩展,适用于系统开发和调试

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 位整型加法指令,它们是否具有相同的指令操作数(opcode)?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(opcode)?试分析为什么采取这样的设计。

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额外的符号扩展才能用于后续 64 位计算?请说明理由。

(1) RV32I 中的 add 指令和 RV64I 中的 addw 指令不具有相同的指令操作数 (opcode)：因为它们是对不同的整数类型执行加法操作。RV32I 中的 add 指令用于对 32 位整数执行加法操作（一个 32 位的操作数和一个 32 位的操作数）；而 RV64I 中的 addw 指令则会将其两个 64 位操作数的低 32 位扩展为 64 位后相加，然后将结果截取回 32 位。因此，addw 指令可以被用来执行 32 位有符号或无符号整数的加法操作。尽管两条指令执行的操作都是加法，但它们操作的数据类型是不同的，因此它们的操作码也是不同的。

RV32I 中的 add 指令和 RV64I 中的 add 指令也不共享相同的指令操作数 (opcode)，因为它们在执行中需要不同的处理：RV32I 中的 add 指令将两个 32 位整数相加，结果需保存在 32 位寄存器中，需要将符号扩展为 64 位；RV64I 中的 add 指令将两个 64 位整数相加，结果也需保存在 64 位寄存器中。

采取这样设计的原因：①为了避免数据的溢出。如果操作数是相同的话，可能会导致 RV32I 在执行 64 位的加法指令时出现数据溢出，而 RV64I 在执行 32 位加法指令时出现数据截断的问题。②这样的设计利于实现特定架构的设备的高效性能运行，同时也可以更加灵活地满足不同架构的习惯和需求。③RV32I 和 RV64I 将操作数设计为不同的指令操作数，可以保证它们的正确性和高效性，降低架构的开发难度，增强可编程的灵活度。

(2) 在 RV64I 中，addw 和 addiw 指令的目标寄存器中存放的 32 位计算结果并不需要进行额外的符号扩展就可以用于后续 64 位计算。因为 RV64I 中定义了 Zero-extension 型和 Sign-extension 型两种类型的指令，其中，addw 和 addiw 属于 Zero-extension 型指令，它们的结果将直接被零扩展，在赋值给 64 位目标寄存器时，高 32 位将被填充为零。而对于 Sign-extension 型指令，则将使用符号位进行符号扩展，以便在 64 位计算中保留完整的有符号值。故 addw 和 addiw 指令的结果已经进行了必要的扩展，能够直接在后续的 64 位计算中使用，而不需要进一步的符号扩展处理。

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？

这里的 hint 指令空间是一种微调指令，它用于提醒处理器，告诉处理器在执行程序的过程中，如何优化指令执行顺序，以提升程序的执行效率。提示的内容通常是关于指令的顺序和执行时间的，并不具有严格的约束性。

通常，hint 指令的作用是给出一些关于程序执行的建议，以便处理器能够最大化的执行效率，以提高整体执行效率。虽然 hint 指令不是必须的指令，但如果程序员知道如何使用它，就可以通过优化程序来提高处理器的性能。

6. 求指令结果，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。

a2 的值为-3，a3 的值为 1

RISC-V N 标准指令集中有两种不同的指令可以用于整数除法和余数计算：①DIV 和 REM 指令对符号的规定是带符号整数和余数计算；②DIVU 和 REMU 指令对符号的要求是无符号整数的乘法和余数计算。

11. 写出以下指令使用的寻址模式。

- 1) 内存直接寻址
- 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