

3/14. Chapter 2.

1. CISC RISC

- 优势:
- ① 实现相同操作所需的指令数少.
  - ② 指令类型丰富, 操作灵活.
- 劣势:
- ① 指令集复杂, 高性能硬件设计复杂.
  - ② 不易编译优化.
- 优势:
- ① 指令格式统一, 类型简单, 硬件的开发周期短.
  - ② 编译简单.
- 劣势:
- ① 指令灵活性受限制.
  - ② 程序长, 难调试.

2. RISC-V 中的基本指令集包括: RV32I, RV32E.  
下列五个常见的 RISC-V 扩展指令集: (及应用)

- ① M: 扩展了整数乘法和除法指令, 适用于需要高效运算的应用, 如 DSP.
- ② F: 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, 适用于需要高效处理浮点数运算的应用, 如科学计算和图形处理.
- ③ C: 定义了部分指令的 16 位版本, 用于小内存的嵌入式应用.
- ④ A: 扩展了并发操作中的原子指令, 用于构建高性能和高效的多线程编程系统.
- ⑤ V: 扩展了向量操作指令, 用于构建高性能和高效的多媒体和机器学习系统.

4. 1) <sup>中 add</sup> RV32I 和 RV64I 中的 addw 指令的指令操作码 opcode 不同.  
且 RV32I 和 RV64I 中的 add 指令的 opcode 也不同.

0110011      0111011

这种设计是为了在 32 位和 64 位架构之间容易区分, 使指令能正确执行.

## 2) 需要额外的符号扩展

原因: ① RV64I 中 `addw` 操作数的低 32 位相加, 并将结果存在目标寄存器中。但目标寄存器的高 32 位将被清零, 因此若在后继指令中使用该结果进行 64 位计算, 需将其符号扩展为 64 位, 即将其高 32 位复制为符号位。

② RV64I 中 `addiw` 指令也需要进行符号扩展, 它会将其操作数的低 12 位符号扩展为 32 位后相加, 并将结果储存在目标寄存器中。若需要在后继指令中使用这个结果进行 64 位计算, 需将其扩展为 64 位, 将其高 32 位复制为符号位。补码  $-100 = 0b1001100$  符号扩展 (全 1)。

例如: `add a0, x0, -100`  $\Rightarrow a_0[31:0] = 0xfffff9c$

`addi a0, a0, 50`  $\Rightarrow a_0[63:0] = 0xfffffffff9fce$   
32bit 64bit 14个

DSP.

寄存器

理.

的

和

5. HINT 指令空间是一组特殊的指令, 用于在处理器和操作系统之间进行通信和协作, 这些指令非必需, 因此被放在一个“保留”指令空间中, 以便未来对其进行扩展而不破坏向后兼容性。

HINT 的作用是向处理器发出暗示或提示, 以便它可以更好地优化或执行更好的调度。允许处理器和操作系统之间更加紧密的协作和交互, 从而提高计算机系统的性能和效率。

6.  $a_2$  存放 -3,  $a_3$  存放 1。

符号规定: 商和余数符号取决于被除数符号。

行.

若除数和被除数符号相同, 则商和余数均为正数。

若除数和被除数符号不同, 则商和余数中有一个负数。

11. 1) jal ra, 0x88 立即数寻址.

2) jalr x0, ra, 0. ~~偏移量寻址~~ 寄存器间接寻址.

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.