

1. CISC 优点：1. 指令丰富，功能强大

2. 寻址方式灵活

性能强大

3. 以微程序控制器为核心，指令存储器与数据存储器共享同一个物理存储空间。

缺点：1. 指令使用率不均衡

2. 不利于采用先进结构 提高性能

3. 结构复杂 不利于 VLS (超大规模集成电路) 实现

RISC 优点：1. 结构简单，易于设计

2. 指令精简，使用率均衡

3. 程序执行效率高

缺点：1. 指令数较少，功能不及 CISC 强大

2. 寻址方式不够灵活。

2. RISC-V 是 RISC 指令集，寄存器-寄存器型

R型：[funct7] [rs2] [rs1] [funct3] [rd] [opcode]，用于寄存器-寄存器操作

I型：[imm[11:0]] [rs1] [funct3] [rd] [opcode]，用于立即数和访存 load 操作

S型：[imm[11:5]] [rs2] [rs1] [funct3] [imm[4:0]] [opcode]，用于访存 store 操作

B型：[imm[2] imm[10:2]] [rs2] [rs1] [funct3] [imm[4:1]] [imm[11]] [opcode]，用于条件跳转操作

U型：[imm[31:12]] [rd] [opcode]，用于卡立即数

J型 [imm[20]] [imm[10:1]] [imm[1]] [imm[9:12]] [rd] [opcode]，用于无条件跳转

4. 1) RV32I 中的 add 指令与 RV64I 的 addw 指令的操作数不同，为了区别指令
⇒ RV32I 中 add 指令与 RV64I 中 add 指令 操作数相同，方便移植、扩展

2) 需要，addw 和 addiw 的结果是 32 位，若要将 32 位结果扩展为 64 位，需要进行行位扩展

5. HINT 表示为 将来标准微架构的 hints 而保存的操作数。

设计 HINT 指令是为了 支持 将来 添加(微架构加入时)的那些可能影响性能、但不会影响状态的微架构的 hints，HINT 编码 被选中后，简单的执行就可以忽略 HINT encoding 并将 HINT 执行为不改变体系结构的常规状态

6. a2 = -3

a3 = 1

符号规定：

	被除数	除数	DIVU	REMU	DIV	REN
除数为 0	x	0	$2^{XLEN}-1$	x	-1	x
溢出(仅对符号)	$-2^{XLEN}-1$	-1	-	-	$-2^{XLEN}-1$	0

II. 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