1. CISC 优点：指令复杂、代码密度高，实现相同操作所需的指令数少，对编译器和程序存储空间要求低，具有更大性能优势；指令类型丰富、操作灵活。

缺点：①"二八定理"，程序中80%的指令只占所有指令类型的20%，指令的利用率不均

②逻辑设计复杂、译码困难、不利于流水线分割，处理器设计复杂，测试与验证难度高。

RISC 优点：指令格式单一、类型简单，硬件设计较为简单，适合流水线提升性能，硬件开发周期更短。

缺点：对编译器设计的要求较高，程序的代码密度较低；指令灵活性弱于CISC，实现相同操作所需指令数更多。

2. 基本指令集

RV32I 32位整数指令集 { RISC(B)V(I)

RV32E RV32I子集，用于小型嵌入式场景

RV64I 64位整数指令集，兼容RV32I

RV128I 128位整数指令集，兼容RV64I和RV32I

标准扩展指令集

M：乘除法、取模求余指令

F：单精度浮点指令

D：双精度浮点指令

Q：四倍浮点指令

A：原子操作指令，例如常见的cas (compare and swap)指令

4. 1) ①不同　RV32I add指令操作数为0110011

RV64I addw指令操作数为0111011

②相同　RV32I和RV64I中，add指令操作数均为0110011

分析：RV32I和RV64I中，add指令功能相同，故操作码相同；addw相比add，在实现加法运算后，还要将结果截断为32位，故操作码与add不同.

2)不需要，因为RV64I中的addw和addiw指令的目标寄存器中存放的32位计算结果已经是经过符号扩展后的结果。

5. 一些指令仅在某些操作数时是有效的，当无效时，可能被标记为RES、NSE或HINT. 被标记为HINT，意味着该操作码被保留给未来的微体系结构提示。在提示没有效果时，标记为HINT的指令必须作为空操作指令执行。

6. a2中值为-3，a3中值为1

符号规定：先将两数作无符号除法，设商为a，余数为b (a,b>0)。若除数和被除数符号相同，则商取a，若相反，则商取-a。余数符号跟随被除数，绝对值为b。原则：始终满足"被除数÷除数=商+余数"，且不论被除数和除数符号，商和余数绝对值不变。

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。

11. 1)偏移量寻址

2)寄存器间接寻址

3)立即数寻址

4)寄存器直接寻址

5)偏移量寻址