1、

	CISC	RISC
优	1、指令丰富，功能强大 2、寻址方式灵活 3、指令、数据共享一个物理空间，性能强大。	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	1、指令使用率不均衡 2、不利用采用先进结构提高性能。 3、结构复杂不利用大规模集成，难于成本大。	1、指令数少，功能不及CISC强大。 2、寻址方式不够灵活。

2、

RISC-V基本指令集
- 32+1个寄存器
- R、I、S、B、U、J六种基本指令格式。

基础指令集为RV32I、RV32E、RV64I、RV128I等整型指令集，

标准扩展指令集：整型乘除M指令集、单精度浮点数操作F指令集
双精度浮点数操作D指令集、原子操作A指令集、
压缩C指令集

M：主要为乘除取余等操作，在各种场景下应用都很多。
F：加入了32个浮点寄存器和一个浮点控制器，来对浮点数进行运算，在需要浮点数小数计算、图形处理等场景广泛应用。
D：扩展了双精度浮点寄存器，相比于F，精度更高。
A：增加对储存器的原子读、写、修改，可以实现更多的算法。
C：增加压缩指令，主要用于改善程序大小。

4. (1) RV32I中的add指令有7位操作码，RV64I中的addw有7位操作码。是，而名相同，而64的add是和32的add一样。

(2) 这样做的好处是可以规范化机器码的结构，使结构更加精简，在硬件设计上方便很多。

(3) 需要。因为此时的寄存器位宽是64位，必须进行位拓展。

5. 一些指令仅在某些操作码时是有效的，当无效时，有可能被标记为HINT，意味着这个操作码必须被保留给未来的微体系结构提示(hint)。在提示被没有效果的实现上，标记为HINT的指令必须被当作空操作指令执行。

6. div a_2, a_0, a_1 ($a_0=16$, $a_1=-7$)

rem a_3, a_0, a_1

① div为有符号除法指令，$a_2=-3$

rem为有符号取余，$a_3=-1$

② 除法和余数都有无符号、有符号不同的版本.

如果是无符号数除余，divu，remu，机器将操作数当作无符号数处理；如果是有符号数除、余，div，rem，机器会根据有符号数的除余运算规则进行运算.

11.

(1) jal ra, 0x88　直接寻址

(2) jalr x0, ra, 0　立即数寻址.

(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.