3/14

1. CISC

优点：指令功能强，代码密度高，对编译和存储要求低。

缺点：硬件设计复杂，调试难度大。

RISC

优点：硬件设计简单，适合流水化以提升性能。

缺点：指令功能单一，对编译器要求高。

2. 基本指令集为RV32I、RV32E和RV64I。

扩展 M：扩展了整数乘除法指令。

F：扩展了IEEE单精度浮点数运算指令。

V：扩展了向量操作指令。

B：扩展了位操作指令。

RV64G：扩展了乘除法、原子操作，单双精度浮点数操作。

4. (1)不同，add是0110011，addw是0111011

两个add相同，均为0110011

两个add功能相同，位数不同，而addw要对加法结果进行符号扩展。两个add，操作码相同可简化机器指令，而addw操作码不同用以区分不同的操作。

扫描全能王 创建

(2) 不需要. 在addwi和addw指令中已包含符号扩展操作.

5. 为HINT指令保留的大片编码空间. 可用于向微架构传达性能提示. 但不会改变体系结构的可见状态.

6. a_2的值为-3, a_3的值为1. 余数符号与被除数相同.

11. (1) 偏移量寻址　　(2) 偏移量寻址

(3) 立即数寻址　　(4) 寄存器直接寻址

(5) 偏移量寻址

3. (1) addi x0,x0,0　　(2) jalr x0,x1,0

(3) auipc x6, offset[31:12]　　(4) addi rd, rs, 0

jalr x1, x6, offset[11:0]

(5) csrrs rd, cycle, x0　　(6) addiw rd, rs, x0

7. (1) sub t3, t0, t2　　(2) add t0, t1, t2

mv t4, t1　　bltu t0, t1, overflow

(3) MIPS 在指令执行时检查溢出. 若溢出将触发指令中断.

ARM 采用CPSR寄存器的V位表示加法溢出状态, V=1表示溢出.

8. (1) $2^{XLEN}-1$; X; -1; X

不会异常. 若触发异常, 需用户处理系统运行异常. 若不包括, 设计者可自行定义异常处理, 更加自由.

(2) NV: 非法操作　DZ: 除数为0.　OF: 上溢出.　UF: 下溢出　NX: 不精确.

不会陷入系统调用

扫描全能王 创建

(3) x86在除数为0时引起0号中断

ARM没有除法指令

MIPS触发break 0x07中断

12. (1)管理员模式

(2)机器模式

(3)机器模式

(4)管理员模式

(5)用户模式

13. vecMul:

```
start: addi sp,sp,-16      //开辟保存调用信息
       sw   ra,12(sp)
       sw   s0,8(sp)
       addi s0,sp,16
prepare: lw  t2,0(t2)      //加载c
       li   t4,99          //计数器
       mv   t5,t0          //保存A[0]地址
muloop: lw   t3,0(t1)      //加载B[i]
       mul  t3,t3,t2       //乘法
       sw   t3,0(t0)       //存至A[i]
       addi t0,t0,4        //A指针后移一位
       addi t1,t1,4        //B后移一位
       addi t4,t4,-1       //计数器-1
       bnez t4,muloop      //t4不为0就循环
end:   lw   t0,0(t5)       //加载返回值A[0]
       lw   ra,12(sp)      //从栈中恢复调用信息
       lw   s0,8(sp)
       addi sp,sp,16
       ret
```

14.

```
    blt  a0,a1,else
    add  a2,a0,a1
    j    endif
else: sub a2,a0,a1
endif:
```

15.

```
    sw   t0,0(t0)
    li   t1,3
    sw   t1,16(t0)
    sw   t1,48(t0)
```

16. swap:

```
start: addi sp,sp,-16
       sw   ra,12(sp)
       sw   s0,8(sp)
       addi s0,sp,16
main:  lw   t2,0(t0)       //取a的值到t2
```


扫描全能王 创建

```
    lw  t3, 0(t1)    //取b的值至t3
    sw  t3, 0(t0)    //b的值赋给a
    sw  t2, 0(t1)    //a的值赋给b
end: lw  ra, 12(sp)
     lw  s0, 8(sp)
     addi  sp, sp, 16
     ret
```

17. 开头：a0=0, a1=1, a2=30

loop：先判断再循环. a0从0~29循环30次.

循环体：每次a1左移一位.

功能：给a1写入2^{30}

扫描全能王 创建