3. RISC-V中存在许多伪指令，请写出以下伪指令[illegible]

解. 1) nop.

即 addi x0, x0, 0.

2) ret.

即 jalr x0, x1, 0.

3) call offset.

即 auipc x1, offset[31:11]; jalr x1, offset[11:0](x1)

~~3) call offset.~~

~~即 jalr ra, ra, offset.~~

4) mv, rd, rs

即 addi rd, rs, 0

5) rdcycle rd.

即 ~~rdcycle. rd, x0,~~ csrrs rd, cycle, x0

6) sext.w rd rs.

即 addiw rd, rs, 0.

7. 使用额外的指令检查溢出，若t1, t2为有符号数，填入指令使加法溢出时正确跳转.

1). add t0, t1, t2.

slti t3, t1, 0.

slt t4, t0, t2.

bne t3, t4, overflow.

2) 对于t1, t2均为无符号数, 检查溢出

add t0, t1, t2.

bltu t0, t1, overflow.

3) 调研其他指令集架构

在硬件中判断加法进行C_i与C_{out}是否相异的硬件检查机制,

软件使用溢出标志位, 判断当前是否溢出

8. 对整形除法，除数为0时是否引起异常，试分析该设计.

解：

指令	rs1	rs2	Op=DIVU rd	OP=REMU rd	OP=DIV rd	OP=REM rd
Op rd,rs1,rs2	x	0	2^x-1	x	-1	x

在溢出题时不报错，但是返回整形值 这样的设计是为降低硬件复杂性，使使用编译器代替电路处理问题

b) 对除法 对浮点除法，除数为0将引起标志位置位，请说明fflags各位的含义

解：fflags中的NV表示非法操作；DZ表示除数为0；OF表示上溢；UF表示下溢；NX表示不精确

一般系统不处理并继续计算.

c) 其他指令集如何处理.

解. 在x86中，除数为0将引起除法溢出，导致处理器中断.

12. 写出以下程序的特权等级

解. 1) Linux Kernel 处于监督模式(S)

2) BootROM 处于机器模式(M)

3) BootLoader 处于机器模式

4) USB Driver. 处于用户模式(U)

5) vim 处于用户模式(U).

13. 写出实现该程序的汇编代码.

解.

```
vecMul:  addi a5,x0,0.
  loop:  add a6,t1,a5
         lw   a7,a6
         lw   a8,t2,0
         ~~mul  a7,a7,~~
         mul  a7,a8,a7
         add  a4,t0,a5.
         sw   a4,a7
         addi a5,a5.1
         bne  a5,100,loop
```

14. 写出下列代码的汇编.

解. add_b:

slt a5, a1, a0

beqz a5, sub_b

add a2, a0, a1

j end

sub_b:

sub a2, a0, a1

end:

15. 写出实现以下C程序的代码.

1) p[0]=p;

sw t0, 0(t0)

2) int a=3;

addi a4, x0, 3

sw a4, 0(t1)

3) p[1]=a;

lw a4, 0(t1)

sw a4, 1(t0)

4) p[a]=a,

lw a4, 0(t1)

add t5, t0, a4

sw a4, 0(a5)

16. 写出C程序的汇编代码.

解: addi a5, x0, 0

mv a5, t0

mv t0, t1

mv t1, a5

ret

17. 解释以下汇编的功能.

```
    addi a0,x0,0        初始化寄存器.
    addi a1,x0,1.
    addi a2,x0,30
loop: beq a0,a2,done.   循环执行移位
    slli a1,a1,1
    addi a0,a0,1        综上,本程序用于执行循环-逻辑左移30次.
    j loop.
done: exit code.
```