6. REM、REMU给出余数　KEM rd, rs1, rs2

DIV、DIVU指令分别执行有符号、无符号的X位整数除以X位整数除法操作

REM、REMU给出相应的余数

16= 0 000 0000 0000 0000 0000 0000 0001 0000

5= 0 000 0000 0000 0000 0000 0000 0000 0101

-5= 1 111 1111 1111 1111 1111 1111 1111 1011

指令集手册要求结果向0趋近，所以应该为商-3，余1。余数与被除数符号相同

11. 寻址 (1)偏移量寻址 (2)直接寻址 (3)立即数寻址 (4)寄存器寻址 (5)偏移量寻址

第5周作业　3、7、8、12、13、14、15、16、17

3. (1) nop = addi x0, x0, 0　　(2) ret = jalr x0, x1, 0

(3) call offset = auipc x1, offset[31:12]　　(4) mv rd, rs = addi, rd, rs, 0

jalr x1, x1, offset[11:0]　　(5) rdcycle rd = csrrs rd, cycle, x0

(6) sext.w rd, rs = addiw rd, rs, 0

7. (1) $t0=t1+t2$: SLT $t3, t2, 0$　如果t_2小于0，则t_3值为1

SLT $t4, t0, t1$　再比较t_0与t_1，如果t_0小于t_1，则t_4值为1

如果t_0不小于t_1，则t_4为0，溢出

如果t_2大于等于0，则t_3值为0；此时若t_0小于t_1，则t_4为1，溢出

(2) add t0, t1, t2　　都是无符号数，则和t0应大于任意一个加数

Bltu t0, t1, overflow　　现在若t0<t1，则跳转

(3) ARM通过CPSR的状态寄存器反映当前指令的溢出状态

x86：ALU会产生溢出信号

8. (1)

指令	rs1	rs2	op=DIVU时 rd值	op=REMU	op=DIV	op=REM
op. rd, rs1, rs2	x	0	$2^{XLEN}-1$	x	-1	x

整形除法中，除数为0不会引起RISC-V抛出异常。如果除以0时触发异常，这个异常在绝大多数执行环境里，会导致一个自陷，这样会成为指令集标准中唯一的算术自陷在此情形下，需要语言实现者与执行环境的自陷处理函数交互

8(2) NV-非法操作　DZ-除以0　OF-上溢　UF-下溢　NX-不精确
基本指令集中不支持在浮点异常上产生自陷

(3) ARM会产生异常与中断，X86若除数为0，则会引起0号中断

12 (1) Linux Kernel - User　(2) BootRoM: Machine　(3) BootLoader: Machine
(4) USB_Driver - Supervisor　(5) vim - User

13. 暂时不会

14.
```
BGE  a0, a1, part1
sub  a2, a0, a1
part1: add  a2, a0, a1
```

17. a0相当于寄存器里 C语言中的i
应为 for (i=0; i<30; i++)
a1 = a1<<2
i循环 2^{29}

15. 不会

16. 也不太会，就这样的题，一碰到指针我就直接傻眼

抄写编译器

```
addi sp,sp,-32
sw   ra,(28)sp
sw   s0,(24)sp
addi s0,sp,32
sw   a0,-12(s0)
sw   a1,-16(s0)
sw   a2,-20(s0)
li   a0,0
sw   a0,-24(s0)
j    part1
part1: lw   a1,-24(s0)
li   a0,99
blt  a0,a1,part4
j    part2
part2: lw   a0,-16(s0)
lw   a1,-24(s0)
slli a2,a1,2
add  a0,a0,a2
lw   a0,0(a0)
lw   a1,-20(s0)
mul  a0,a0,a1
lw   a1,-12(s0)
add  a1,a1,a2
sw   a0,0(a1)
j    part3
part3: lw   a0,-24(s0)
addi a0,a0,1
sw   a0,-24(s0)
j    part1
part4: lw   a0,-12(s0)
lw   a0,0(a0)
lw   ra,28(sp)
lw   s0,24(sp)
addi sp,sp,32
ret
```