第五周作业

3. (1) nop　　addi x0, x0, 0

(2) ret　　jalr x0, x1, 0

(3) call offset　　auipc x1, offset[31:12]

jalr x1, x1, offset[11:0]

(4) mv rd, rs　　addi rd, rs, 0

(5) rdcycle rd　　csrrs rd, cycle, x0

(6) sext.w rd rs　　addiw rd, rs, 0

7. (1) slti t3, t2, 0

slt t4, t0, t1

(2) add t0, t1, t2

bltu t0, t1, overflow

(3) 对于x86指令集，其中无符号数运算的溢出称为进位，可以通过带进位加法指令adc解决，而有符号数运算的溢出才叫做溢出，而此时可以通过OF标志位来检测到。而在ARM体系结构中，通过CPSR的状态寄存器反映溢出状态。

8. (1)

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

不一定会引起RISC-V抛出异常。如在REM下除数为0时rd返回值仍为x，这样设计可使得除0的指令返回正常结果，使得后续指令得以继续进行。

(2) fflags中：NV代表 Invalid Operation (无效操作)

DZ代表 Divide by Zero (除数为0)

OF代表 Overflow (溢出)　UF代表 Underflow (错位)

NX代表 Inexact (不精确)

fflags被置位不会使处理器陷入系统调用。

(3) x86指令集中要求除数不能为0，否则将产生错误，导致处理器异常并暂停执行程序，可以用CPU跳转语句避开除数为0的情况。ARM中的除法指令也采取类似的跳转语句避开除数为0的情况。

12.(1) Linux kernel　　处于机器模式(M)

(2) BootROM　　机器模式(M)

(3) BootLoader　　机器模式(M)

(4) USB Driver　　用户模式(U)

(5) vim　　管理员模式(S)

13.
```
# Assign t4=i
mv t3, t0
add t4, x0, x0
addi t5, x0, 100
Loop:
bge t4, t5, exit
sll t6, t4, 2
sll t7, t4, 2
add t7, t7, t1
add t6, t6, t0
lw t6, 0(t6)
lw t7, 0(t7)
mul t6, t7, t2
addi t4, t4, 1
j Loop
exit: mv t0, t3
```

14.
```
blt  a1, a0, 1f
j    2f
1:
add  a2, a0, a1
2:
sub  a2, a0, a1
```

15.
```
addi  t2, x0, 1
add   t3, x0, t1
sll   t4, t2, 2
add   t4, t4, t0
add   t4, x0, t1
Loop:
bge   t2, t3, exit
~~sll   t4, t2, 2~~
sll   t4, t2, 2
addi  t2, t2, 1
exit: add   t4, t4, t0
      add   t4, x0, t1
```

16.
```
add  t2, x0, x0
addi t2, t0, 0
addi t0, t1, 0
addi t1, t2, 0
```

17. 解：每次循环(loop)将a1向左移1位，直至a0=a2=30，左移三十位，此时a1中存储的即是1×2^{30}。代码实现了这一计算。