Ejercicios 3.1 y 5.1 del libro de Hennessy

3.1. What is the baseline performance (in cycles, per loop iteration) of the code sequence in Figure 3.47 if no new instruction's execution could be initiated until the previous instruction's execution had completed? Ignore front-end fetch and decode. Assume for now that execution does not stall for lack of the next instruction, but only one instruction/cycle can be issued. Assume the branch is taken, and that there is a one-cycle branch delay slot.

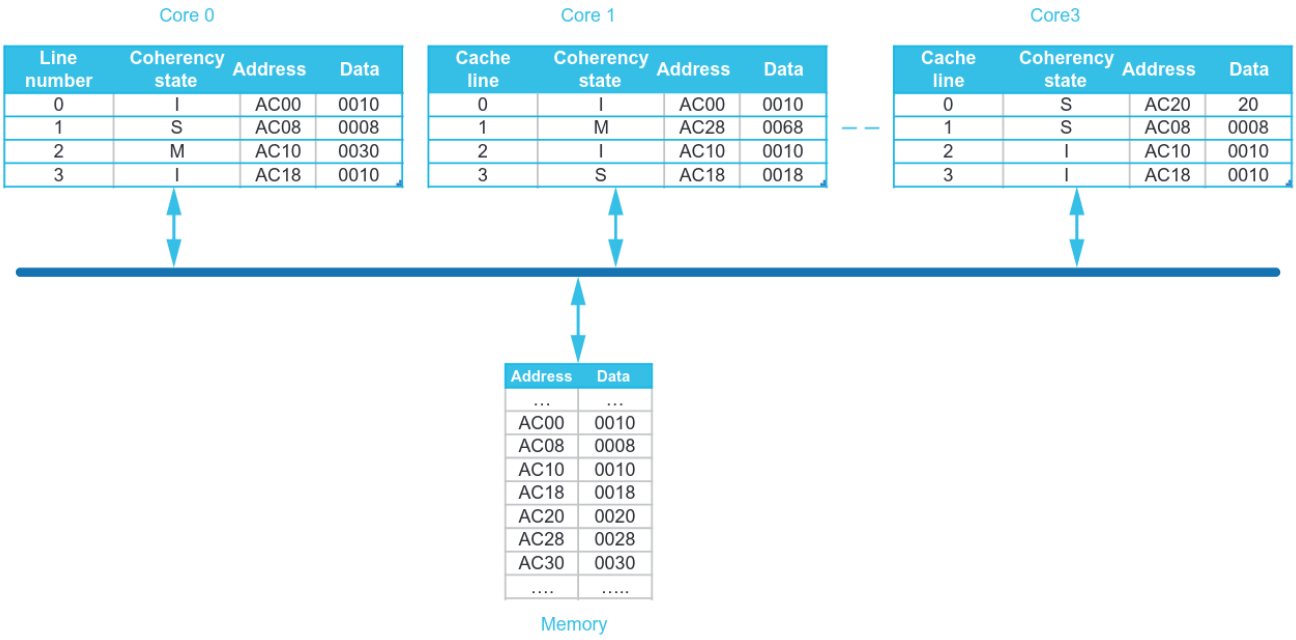
Latencies beyond single cycle	
Memory LD	+3
Memory SD	+1
Integer ADD, SUB	+0
Branches	+1
fadd.d	+2
fmul.d	+4
fdiv.d	+10

```
Loop:   fld     f2,0(Rx)
I0:     fmul.d  f2,f0,f2
I1:     fdiv.d  f8,f2,f0
I2:     fld     f4,0(Ry)
I3:     fadd.d  f4,f0,f4
I4:     fadd.d  f10,f8,f2
I5:     fsd     f4,0(Ry)
I6:     addi    Rx,Rx,8
I7:     addi    Ry,Ry,8
I8:     sub     x20,x4,Rx
I9:     bnz     x20,Loop
```

Figure 3.47 Code and latencies for Exercises 3.1 through 3.6.

De acuerdo al enunciado, cada instrucción requiere un ciclo de reloj para ejecución, es decir, solo una instrucción a la vez puede ocupar la unidad de ejecución. Esto nos da sumado un total de 11 ciclos de reloj. A ese resultado se le suman los ciclos de latencia, lo cual sería un total de 26. Finalmente, hay que sumar el ciclo extra de delay que ocurre por el branch. Tras sumar todo, el total de ciclos de reloj es de 38 (por cada iteración en el loop).

5.1.



a. C0: R, AC20

C0.L0: (S, AC20,0020)

b. C0: W, AC20 <-- 80

C0.L0: (M, AC20, 0080)
C3.L0: (I, AC20, 0020)

c. C3: W, AC20 <-- 80

C3.L0: (M, AC20, 0080)

d. C1: R, AC10

C1.L2: (S,AC10, 0010)

e. C0: W, AC08 <-- 48

C0.L1: (M, AC08, 0048)
C3.L1: (I, AC08, 0008)

f. C0: W, AC30 <-- 78

C0.L2: (M, AC30, 0078)

M: AC10 <-- 0030

g. C3: W, AC30 <-- 78

C3.L2: (M, AC30, 0078)