4.3

(1) load and store 25%+10%=35%

(2) all the fractions 100%

(3) I-type, load, store, branch, jump 28%+25%+10%+11%+2%=76%

(4) 正常工作，但是输出会陪忽略

4.7

(1) 30+250+150+25+200+25+20=700ps

(2) 30+250+150+25+200+250+25+20=950ps

(3) 30+250+150+200+25+250=905ps

(4) 30+250+150+25+200+5+25+20=705ps

(5) 30+250+150+20+200+25+20=700ps

(6) minimum clock = clock period of ld = 950ps

4.10

(1) 12%\*(25%+11%)=4.32%

950/((950+10)\*(1-4.32%))=1.03

(2) 5+1000+200+100+2000+4\*10+500\*2+30\*2+2\*1+100=4507

4507+200=4707

(4707-4507)/4507=4.44%

(3) 在需要高速计算的情形下（如股市开盘等），一般而言，时间是非常有限的，在这段节省的时间中，可以创造出远超4.4%成本的价值，此时这种优化是有价值的；相反，在对速度没有需求的项目中，多余的成本就是没必要的。

4.24

第二个是正确的。第一二语句中存在冒险，第二句的add操作需要在x11赋值之后运行，因此bubble需要在EX之前，ID之后，保证x11值的正确性，所以第二种正确。

4.27

(1)

add x15, x12, x11

nop

nop

ld x13, 4(x15)

ld x12, 0(x2)

nop

or x13, x15, x13

nop

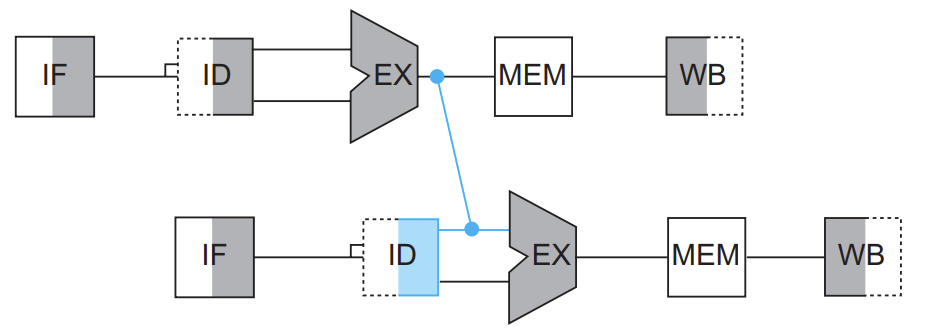
nop

sd x13, 0(x15)

(2)

这题不太会，改不出来感觉

(3)



x15的值能直接从上一级EX赋到下一级的EX中，不会出现x15值的错误，x13的值也会直接到执行or x13, x15, x13中。所以代码可以正常执行。