Homework Assignment 2 of Computer Architecture

College of Computer Science, Zhejiang University

Total 200 points

Submission deadline: Nov.10, 11:59pm

For C.1 and C.3 Let us assume that the Branch is resolved at ID stage.

C.1 [15/15/15/15/25/10/15] <A.2> Use the following code fragment:

```
Loop:   LD      R1,0(R2)     ;load R1 from address 0+R2
        DADDI   R1,R1,#1     ;R1=R1+1
        SD      R1,0,(R2)    ;store R1 at address 0+R2
        DADDI   R2,R2,#4     ;R2=R2+4
        DSUB    R4,R3,R2     ;R4=R3-R2
        BNEZ    R4,Loop      ;branch to Loop if R4!=0
```

Assume that the initial value of R3 is R2 + 396.

a. [15] <C.2> Data hazards are caused by data dependences in the code. Whether a dependency causes a hazard depends on the machine implementation (i.e., number of pipeline stages). List all of the data dependences in the code above. Record the register, source instruction, and destination instruction; for example, there is a data dependency for register R1 from the LD to the DADDI.

b. [15] <C.2> Show the timing of this instruction sequence for the 5-stage RISC pipeline without any forwarding or bypassing hardware but assuming that a register read and a write in the same clock cycle "forwards" through the register file, as shown in Figure C.6. Use a pipeline timing chart like that in Figure C.5. Assume that the branch is handled by flushing the pipeline. If all memory references take 1 cycle, how many cycles does this loop take to execute?

a. 数据冲突来自数据依赖。是否发生冲突取决于 CPU 的实现(也就是流水线的极数)。列出所有上述代码所有的数据依赖。记录 寄存器，源指令和目标指令.

解决：

R1	LD	DADDI
R1	DADDI	SD
R2	LD	DADDI
R2	SD	DADDU
R2	DSUB	DADDI
R4	BNEZ	DSUB

这里的 source 和 destination 我还不是很清楚。再描述一下我认为的依赖。第一行 LD 和第二行 DADDI 中 R1 有数据依赖，因为 LD 到 R1 的数据影响了 R2。第二行的 DADDI 和第三行的 SD 中 R1 有数据依赖，因为第二行的运算改变了 R1 的值，而第三行要写入。第四行的 DADDI 和第一行的 LD 中 R2 有数据依赖，因为第四行改变了 R2 的值，在循环回去的过程中，影响了 LD 的地址。同理第三行的 SD 的地址也受 R2 影响。第四行中的 ADDI 和第五行的 DSUB 中 R2 有数据依赖，第四行中的 DSUB 和第五行的 BNEZ 中 R4 有数据依赖。

注：写完后明白了，题目中想表述的 source 和 destination 其实是指的发生数据依赖的寄存器本身在指

令中的身份。

b. 显示 5-极 RISC 流水线的指令序列时序图。这里没有 forwarding 和 bypassing 硬件，但是假设有 double bump（当然我们显然是要解决数据冲突的，否则这些指令执行都不正确，因此要加入 stall）。假设 branch 指令的解决办法是 flushing-------也就是后面增加停顿或者说情况为 nop。如果所有的内存访问指令花费 1 个 cycle，这个 loop 需要多少 cycyles。

Inst	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
LD	F	D	X	M	W																
DADDI		F	S	S	D	X	M	W													
SD					F	S	S	D	X	M											
DADDI								F	D	X	M	W									
DSUB									F	S	S	D	X	M	W						
BNEZ												F	S	S	D	X	M	W			
BNEZ+1															F	S	S	S	S		
LD(2)																F	D				

注意，题目说明 branch 指令的写在 ID 阶段，因此只有在 BNEZ 的 ID 后的下一个阶段取的地址有效。前面的那个 flush 了.

对于第二个问题。题目中已知 R3 = R2+396。程序中 R2 每一次循环+4，R4 每次都等于 R3 的值减去 R2，而 R3 本身不变。循环终止的条件是 R4 = 0，也即 R3=R2.

由分析可知，循环一共持续了 396 / 4 = 99 轮。

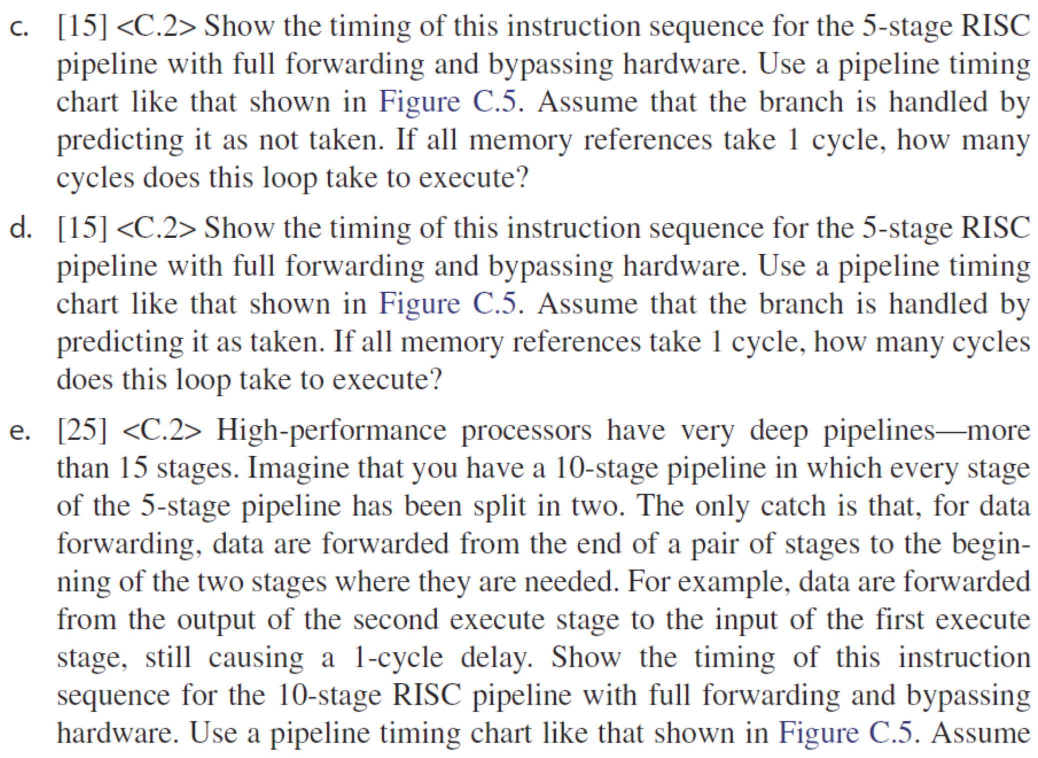
每一轮循环中，LD 到 BNEZ 以及他后面的一个 stall，或者说是一条指令的 idle。（浪费了一个周期）

一共的周期数是 15.

因此最终结果是 15 * 98 + 18 = 1488。

最后一个周期要花费 18 个周期，因为 beq 完全执行完需要到 18 个周期，他本身是这个 loop 的一部分。而下一条指令不是 LD 指令了，不是 loop 的一部分。

有人说这个 BNEZ 后面的 flush 要浪费两个周期，我是不认同的，因为是在 ID 阶段判断 beq 和计算 beq 地址的。


c. [15] <C.2> Show the timing of this instruction sequence for the 5-stage RISC pipeline with full forwarding and bypassing hardware. Use a pipeline timing chart like that shown in Figure C.5. Assume that the branch is handled by predicting it as not taken. If all memory references take 1 cycle, how many cycles does this loop take to execute?

d. [15] <C.2> Show the timing of this instruction sequence for the 5-stage RISC pipeline with full forwarding and bypassing hardware. Use a pipeline timing chart like that shown in Figure C.5. Assume that the branch is handled by predicting it as taken. If all memory references take 1 cycle, how many cycles does this loop take to execute?

e. [25] <C.2> High-performance processors have very deep pipelines—more than 15 stages. Imagine that you have a 10-stage pipeline in which every stage of the 5-stage pipeline has been split in two. The only catch is that, for data forwarding, data are forwarded from the end of a pair of stages to the beginning of the two stages where they are needed. For example, data are forwarded from the output of the second execute stage to the input of the first execute stage, still causing a 1-cycle delay. Show the timing of this instruction sequence for the 10-stage RISC pipeline with full forwarding and bypassing hardware. Use a pipeline timing chart like that shown in Figure C.5. Assume

c. 显示 5-极 RISC 流水线的指令序列时序图。包含完全的 forwarding 硬件。Branch 指令采取 predict not taken 策略。如果所有的访存指令花费一个周期，这个 loop 需要花费多少个周期。

Inst	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
LD	F	D	X	M	W																
DADDI		F	D	S	X	M	W														
SD			F	S	D	X	M	W													
DADDI					F	D	X	M	W												
DSUB						F	D	X	M	W											
BNEZ							F	S	D	X	M	W									
BNEZ+1									F	S	S	S	S								
LD(2)										F	D	X	M	W							

这里由于采用 predict not taken，因此在题目的循环中，始终是预测错误的，要浪费一个周期。

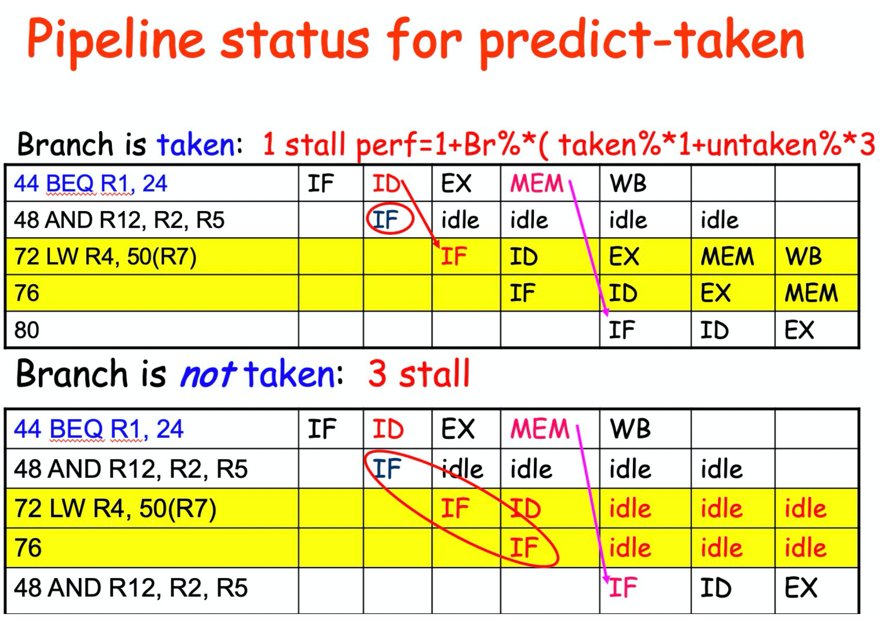
对于第二个问题

类似上面的，同样的 98 * 9 + 12 = 894。最后的 12 是 beq 完成执行的周期。

d. 显示 5-极 RISC 流水线的指令序列时序图。包含完全的 forwarding 硬件。Branch 指令采取 predict taken 策略。如果所有的访存指令花费一个周期，这个 loop 需要花费多少个周期。

Inst	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
LD	F	D	X	M	W																
DADDI		F	D	S	X	M	W														
SD			F	S	D	X	M	W													
DADDI					F	D	X	M	W												
DSUB						F	D	X	M	W											
BNEZ							F	S	D	X	M	W									
									F	S	S	S	S								
LD(2)										F	D	X	M	W							

事实上，尽管采用的是 branch taken，但是 branch 地址的计算还是要到 ID 之后。
根据课本结果是这样的。因此和 c 中的表格应该是没有差别的。
所以最终结果仍然是 98 * 9 + 12 = 894。



Pipeline status for predict-taken

Branch is taken: 1 stall perf=1+Br%*(taken%*1+untaken%*3

44 BEQ R1, 24	IF	ID	EX	MEM	WB		
48 AND R12, R2, R5		IF	idle	idle	idle	idle	
72 LW R4, 50(R7)			IF	ID	EX	MEM	WB
76				IF	ID	EX	MEM
80					IF	ID	EX

Branch is *not* taken: 3 stall

44 BEQ R1, 24	IF	ID	EX	MEM	WB		
48 AND R12, R2, R5		IF	idle	idle	idle	idle	
72 LW R4, 50(R7)			IF	ID	idle	idle	idle
76				IF	idle	idle	idle
48 AND R12, R2, R5					IF	ID	EX

但似乎 有人 认为这里最后的 LD(2)的 F 上和 BNEZ 的 D 同时发生的，所以结果是 8 * 98 + 12 = 796，如果是在 LD 阶段完成 Branch 地址计算的话，这是不可能的，因为寄存器每次读的都是上一次线上的数据.

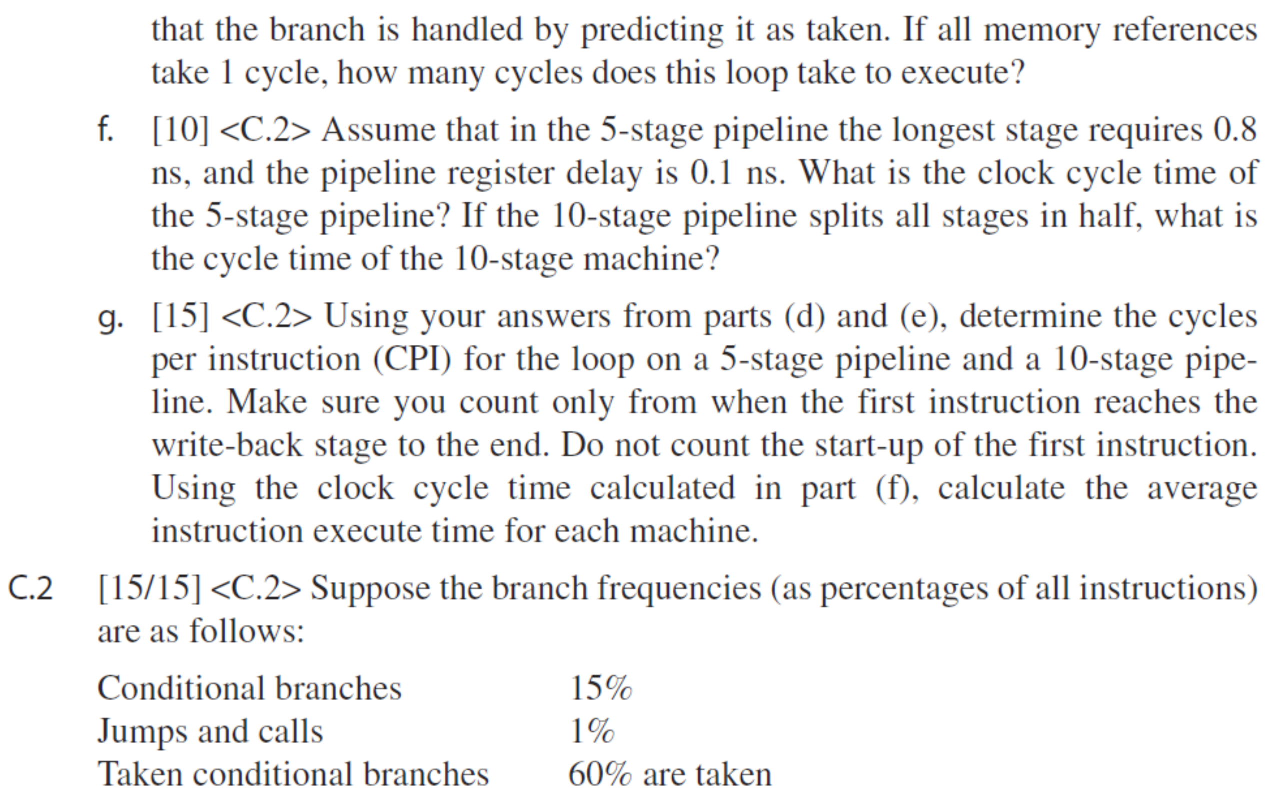
因此此处我猜测，应该是这种方式的解法，认为 Branch 的地址计算是在 IF 阶段，Branch 的判断在 ID 阶段。因为课程 slides 也一般都认为，beq 的判断比地址计算多至少一个周期。////但我还是按自己的实现和理解解决的这一大题。所以后面也都认为 Branch 的计算和判断写入都在 ID 阶段。不过同时也给出不同的地方。

e. 高性能流水线处理器有很深的流水线极数，超过了 15 级。假设有一个 10 级流水线，他将 5 级流水线的每一个步骤都分为了两部分。唯一的要求是，数据 forwarding 仍然是按照 5 级的形式，也就是他在对应的 5 级上分为两部分的最后一部分到另一个对应五级分为两部分的前一部分。显示对应的指令顺序时序图，包含完全 forwarding 硬件。计算 loop 花费的周期数。假设采取 predict taken 策略。

Inst	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
LD	F1	F2	D1	D2	X1	X2	M1	M2	W1	W2										
DADDI		F1	F2	D1	D2	S	S	S	X1	X2	M1	M2	W1	W2						
SD			F1	F2	D1	S	S	S	D2	X1	X2	M1	M2	W1	W2					
DADDI				F1	F2	S	S	S	D1	D2	X1	X2	M1	M2	W1	W2				
DSUB					F1	S	S	S	F2	D1	D2	S	X1	X2	M1	M2	W1	W2		
BNEZ									F1	F2	S	S	S	S	D1	D2	X1	X2	M1	…
LD(2)											S	S	S	S	F1	F2	…			

总的周期数为 10 * 98 + 24 = 1004

有答案认为是 10 * 98 + 19 = 999；区别主要在于 bnez 指令的时序图。我认为 BNEZ 的 ID 是应该在上面 SUB 的 EX 的后面的，这样才能 forward 过来，并且正确判断。


that the branch is handled by predicting it as taken. If all memory references take 1 cycle, how many cycles does this loop take to execute?

f. [10] <C.2> Assume that in the 5-stage pipeline the longest stage requires 0.8 ns, and the pipeline register delay is 0.1 ns. What is the clock cycle time of the 5-stage pipeline? If the 10-stage pipeline splits all stages in half, what is the cycle time of the 10-stage machine?

g. [15] <C.2> Using your answers from parts (d) and (e), determine the cycles per instruction (CPI) for the loop on a 5-stage pipeline and a 10-stage pipeline. Make sure you count only from when the first instruction reaches the write-back stage to the end. Do not count the start-up of the first instruction. Using the clock cycle time calculated in part (f), calculate the average instruction execute time for each machine.

C.2 [15/15] <C.2> Suppose the branch frequencies (as percentages of all instructions) are as follows:

Conditional branches	15%
Jumps and calls	1%
Taken conditional branches	60% are taken

f. 假设 5 级流水线，最长的级需要 0.8ns，流水线寄存器的延迟为 0.1ns，五级流水线的时钟周期应该为多少？如果是 5 级每个阶段分为两部分的 10 级流水线，他的周期是多少？

比较容易理解。

0.8+0.1 = 0.9ns

0.4+0.1 = 0.5ns

g. 根据 d 和 e，确定 5 级和 10 级流水线 CPU 的 CPI。使用 f 中计算的时钟周期，计算两个 CPU 的平均执行时间。

根据我的运算，5 级的周期为 894，10 级的周期为 1004
因此 CPI 分别为 894 / (99*6) = 1.50 和 1004 / (99*6) = 1.69
平均周期分别为 1.5 * 0.9 = 1.35 和 1.69 * 0.5 = 0.84

根据另一种计算结果，尽管我并不认为正确
周期分别为 796 和 999
因此 CPI 分别为 796 / (99*6) = 1.34 和 999 / (99*6) = 1.68
平均周期分别为 1.5 * 0.9 = 1.21 和 1.69 * 0.5 = 0.84

15%的条件跳转
1%的无条件跳转
条件跳转中，有 60%实际发生跳转。

a. [15] <C.2> We are examining a four-deep pipeline where the branch is resolved at the end of the second cycle for unconditional branches and at the end of the third cycle for conditional branches. Assuming that only the first pipe stage can always be done independent of whether the branch goes and ignoring other pipeline stalls, how much faster would the machine be without any branch hazards?

b. [15] <C.2> Now assume a high-performance processor in which we have a 15-deep pipeline where the branch is resolved at the end of the fifth cycle for unconditional branches and at the end of the tenth cycle for conditional branches. Assuming that only the first pipe stage can always be done independent of whether the branch goes and ignoring other pipeline stalls, how much faster would the machine be without any branch hazards?

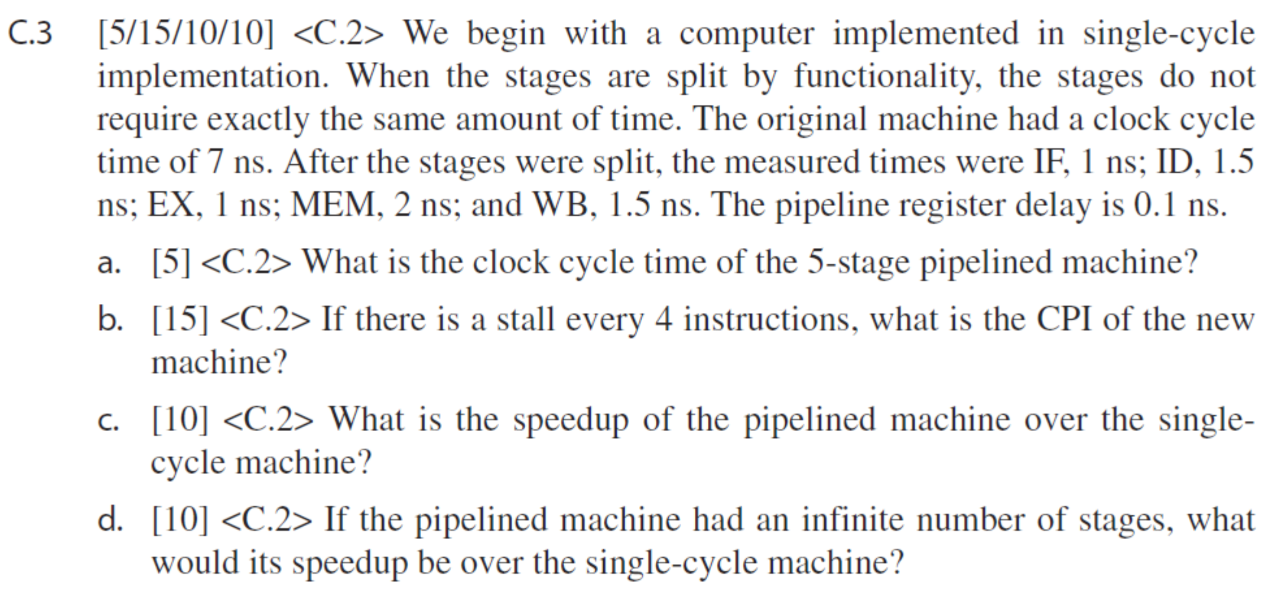
a. 我们考虑 4 级流水线，无条件跳转解析发生在第二个阶段结束。条件跳转解析发生在第三个阶段结束。假设只有第一个 pipe stage 可以独立于 beq 是否发生，并且他能忽略其他的流水线 stall。计算加速比。

应该只考虑控制冲突产生的 stall
解决：
对于无条件跳转，发生 1 个额外的 stall。对于条件跳转，如果采用 predict branch-taken 的方式，如果预测失败，有 2 个 stall，如果预测成功 1 个 stall(这里我认为，branch 的地址计算是在第二个阶段，解析是否跳转在第三个阶段)。如果采用 predict branch-not-taken，如果预测失败，有 2 个 stall，如果预测成功，0 个 stall。由此计算得到。
predict branch-taken，　　CPI = 1 + 0.01*1 + 0.15*0.6*1 + 0.15*0.4*2 = 1.22, speedup. = 4 / 1.22 = 3.28
predict branch-not-taken，　CPI = 1 + 0.01*1 + 0.15*0.6*2 + 0.15*0.4*0 = 1.19, speedup = 4 / 1.19 = 3.36

考虑到题目中到描述不是完全理解，一般情况，CPI = 1 + (1∗1%)+(2∗9%)+(1∗6%)=1.24
Speedup = 3.23

b. 假设有一个高性能处理器，15 级深度流水线。无条件跳转在第五季结束解析，有条件跳转在第十级结束解析。计算 speedup

类似的
一般情况 CPI = 1+(1% * 4) + (9 * 9%) + (8*6%) = 1.516
Speedup = 4/1.516 = 2.64



C.3 [5/15/10/10] <C.2> We begin with a computer implemented in single-cycle implementation. When the stages are split by functionality, the stages do not require exactly the same amount of time. The original machine had a clock cycle time of 7 ns. After the stages were split, the measured times were IF, 1 ns; ID, 1.5 ns; EX, 1 ns; MEM, 2 ns; and WB, 1.5 ns. The pipeline register delay is 0.1 ns.

a. [5] <C.2> What is the clock cycle time of the 5-stage pipelined machine?

b. [15] <C.2> If there is a stall every 4 instructions, what is the CPI of the new machine?

c. [10] <C.2> What is the speedup of the pipelined machine over the single-cycle machine?

d. [10] <C.2> If the pipelined machine had an infinite number of stages, what would its speedup be over the single-cycle machine?

先考虑单周期的 CPU，如果根据功能划分为不同的阶段，不同的阶段花费不同的时间。原来机器的周期是 7ns。变为 5 个阶段后，各自的周期为 1ns，1.5ns，1ns，2ns，1.5ns。流水线寄存器延迟为 0.1ns

a. 5 级流水线的时钟周期

解决：

选择最大的，2+0.1 = 2.1ns

b. 如果每 4 条指令一次 stall，CPI 是多少

CPI = 1 + 1/4 = 1.25

c. 计算加速比

加速比 = 7 / (2.1 * 1.25) = 2.67

d. 如果流水线有无穷的阶段，加速比会怎么样？

加速比就是约等于流水线的级数。但如果过于大，会导致寄存器开销以及其他的开销占比大，反而性能下降。

C.7 [10/10] <C.3> In this problem, we will explore how deepening the pipeline affects performance in two ways: faster clock cycle and increased stalls due to data and control hazards. Assume that the original machine is a 5-stage pipeline with a 1 ns clock cycle. The second machine is a 12-stage pipeline with a 0.6 ns clock cycle. The 5-stage pipeline experiences a stall due to a data hazard every 5 instructions, whereas the 12-stage pipeline experiences 3 stalls every 8 instructions. In addition, branches constitute 20% of the instructions, and the misprediction rate for both machines is 5%.

a. [10] <C.3> What is the speedup of the 12-stage pipeline over the 5-stage pipeline, taking into account only data hazards?

b. [10] <C.3> If the branch mispredict penalty for the first machine is 2 cycles but the second machine is 5 cycles, what are the CPIs of each, taking into account the stalls due to branch mispredictions?

我们将要研究流水线的深度是如何通过更快的时钟周期和更多由于数据冲突产生的 stall 影响性能的。假设原始的机器是 5 级流水线，时钟周期为 1ns。第二个机器是 12 级流水线，时钟周期为 0.6ns。5 级流水线每 5 条指令遇到一次由于数据冲突造成的 stall。12 级流水线每 8 条指令遇到 3 个 stall。除此之外，branch 指令包含了 20%，错误预测的比率为 5%。

a. 计算 speedup，只考虑数据冲突，不考虑控制冲突。

解决：

5 级， CPI = 1 + 1/5 = 1.2

12 级，CPI = 1 + 3/8 = 1.375

加速比 = 1.2 * 1 / (1.375 * 0.6) = 1.45

这里是 12 级更优。

b. 如果 5 级流水线的 branch 预测出错的惩罚是 2 个周期，12 级流水线的 branch 预测出错的惩罚是 5 个周期。他们各自的 CPI 是多少。考虑数据冲突和控制冲突造成的 stall。

5 级， CPI = 1.2 + 0.2 * 0.05 * 2 = 1.22

12 级，CPI = 1.375 + 0.2 * 0.05 * 5 = 1.425

加速比 = 1.22*1 / (1.425 * 0.6).= 1.426