Disclaimer:

1. The solution is just for your reference. They may contain some mistakes. DO TRY to solve the problems by yourself. Please also pay attentions to the course website for the updates.

2. Try not to use pseudoinstructions for any exercises that ask you to produce MIPS code. Your goal should be to learn the real MIPS instruction set, and if you are asked to count instructions, your count should reflect the actual instructions that will be executed and not the pseudoinstructions.

Selected exercise for Chapter 4: 4.12, 4.15.1~4.15.3

4.12.1 Dependences to the 1st next instruction result in 2 stall cycles, and the stall is also 2 cycles if the dependence is to both 1st and 2nd next instruction. Dependences to only the 2nd next instruction result in one stall cycle. We have:

CPI	Stall Cycle
1+0.35*2+0.15*1=1.85 (EX to 1, MEM to 1, EX to 1) *2 + (EX to 2, MEM to 2)*2	46%(0.85/1.85)

4.12.2 With full forwarding, the only RAW data dependences that cause stalls are those from the MEM stage of one instruction to the 1st next instruction. Even this dependences causes only one stall cycle, so we have:

CPI	Stall Cycle
1+0.20=1.20	17%(0.20/1.20)

4.12.3

EX/MEM only	EX to 1st dependences : no stall (use forwarding) MEM to 1st dependences: two stalls (wait for data to be written back EX to 2nd dependences: one stall (wait for data to be written back) MEM to 2nd dependences: one stall (wait for data to be written back EX to 1st and MEM to 2nd : one stall	
MEM/WB only	EX to 1st dependences: one stall (because we must wait for the instruction to complete the MEM stage to be able to forward to the next instruction) MEM to1st dependences: one-cycle stall EX to 2nd dependences: no stalls. MEM to 2nd dependences: no stall (use forwarding) EX to 1st and MEM to 2nd : one stall	

We compute stall cycles per instructions for each case as follows:

EX/MEM	MEM/WB	Fewer stall cycles with
0.2*2+0.05+0.1+0.1=0.65	0.05+0.2+0.1=0.35	MEM/WB

4.12.4. In 4.12.1 and 4.12.2, we have already computed the CPI without forwarding and with full forwarding. Now we compute time per instruction by taking into account the clock cycle time:

Without forwarding	With forwarding	Speedup
1.85*150 ps=277.5 ps	1.20*150 ps=180 ps	1.54

4.12.5 We already computed the time per instruction for full forwarding in 4.12.4. Now we compute time-per instruction with time-travel forwarding and the speedup over full forwarding:

With full forwarding	Time-travel forwarding	Speedup
1.20*150 ps=180 ps	1*250 ps=250 ps	0.72

4.12.6

EX/MEM	MEM/WB	Shorter time per instruction with
1.65*150 ps=247.5	1.35*150 ps = 202.5 ps	MEM/WB

4.15.1

Each branch that is not correctly predicted by the always-taken predictor will cause 3 stall cycles, so we have: Extra CPI =3*(1-0.45)*0.25=0.41

4.15.2

Each branch that is not correctly predicted by the always-not-taken predictor will cause 3 stall cycles, so we have: Extra CPI= 3*(1-0.55)*0.25=0.34

4.15.3 Each branch that is not correctly predicted by the 2-bit predictor will cause 3 stall cycles, so we have: Extra CPI= 3*(1-0.85)*0.25=0.113