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: 4.16, 4.19.1~4.19.4

4.16.1 Solution:

Always-taken predictor

Actual Pattern	T	N	T	T	N
Predicted	T	T	T	T	T

Accuracy of always-taken predictor : 3/5=60%

Always-not-taken predictor

Actual Pattern	T	N	T	T	N
Predicted	N	N	N	N	N

Accuracy of always-not-taken predictor : 2/5=40%

4.16.2 First 4 branches are T, N, T, T. (N: not taken, T: taken, I:incorrect, C:correct)

Actual Pattern	T	N	T	T
Predictor value at time of prediction	0	1	0	1
Predicted action	N	N	N	N
Prediction result in steady state	I	C	I	I

Accuracy = 1/4=25%

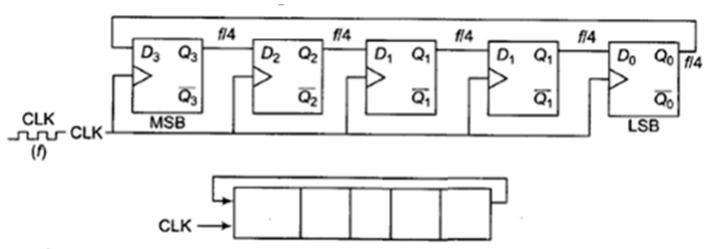
4.16.3 Solution

The first few recurrences of this pattern do not have the same accuracy as the later ones because the predictor is still warming up. To determine the accuracy in the “steady state”, we must work through the branch predictions until the predictor values start repeating (i.e., until the predictor has the same value at the start of the current and the next recurrence of the pattern. N: not taken, T: taken, I:incorrect, C:correct)

Actual Pattern	T	N	T	T	N	T	N	T	T	N	T	N	T	T	N	T	N	T	T	N
Predictor value at time of prediction	0	1	0	1	2	1	2	1	2	3	2	3	2	3	3	2	3	2	3	3
Predicted action	N	N	N	N	T	N	T	N	T	T	T	T	T	T	T	T	T	T	T	T
Prediction result in steady state	I	C	I	I	I	I	I	I	C	I	C	I	C	C	I	C	I	C	C	I

Accuracy = 3/5= 60%

4.16.4 Solution: The predictor should be an N-bit shift register, where N is the number of branch outcomes in the target pattern. The shift register should be initialized with the pattern itself (0 for NT, 1 for T), and the prediction is always the value in the left most bit of the shift register. The register should be shifted after each predicted branch.



Load patterns into the shift register

4.16.5 Solution: Since the predictor's output is always the opposite of the actual outcome of the branch instruction, the accuracy is zero.

4.16.6 The predictor is the same as in Exercise 4.16.4, except that it should compare its prediction to the actual outcome and invert (logical NOT) all the bits in the shift register if the prediction is incorrect. This predictor still always perfectly predicts the given pattern. For the opposite pattern, the first prediction will be incorrect, so the predictor's state is inverted and after that the predictions are always correct. Overall, there is no warm-up period for the given pattern, and the warm-up period for the opposite pattern is only one branch.

4.19.1 The energy for the two designs is the same: I-Mem is read, two registers are read, and a register is written. We have: 140 pJ+2*70 ps +60 pJ =340 pJ

4.19.2 The instruction memory is read for all instructions. Every instruction also results in two register reads (even if only one of those values is actually used). A load instruction results in a memory read and a register write, a store instruction results in a memory write, and all other instructions result in either no register write (e.g., BEQ) or a register write. Because the sum of memory read and register write energy is larger than memory write energy, the worst-case instruction is a load instruction. For the energy spent by a load, we have: 140 pJ+2*70 pJ +60 pJ + 140 pJ =480 pJ

4.19.3 Instruction memory must be read for every instruction. However, we can avoid reading registers whose values are not going to be used. To do this, we must add RegRead1 and RegRead2 control inputs to the Registers unit to enable or disable each register read. We must generate these control signals quickly to

avoid lengthening the clock cycle time. With these new control signals, a LW instruction results in only one register read (we still must read the register used to generate the address), so we have:

Energy before change	Energy saved by change	% Savings
140 pJ+2*70 pJ +60 pJ +140 pJ =480 pJ	70 pJ	14.6%

4.19.4 Before the change, the Control unit decodes the instruction while register reads are happening. After the change, the latencies of Control and Register Read cannot be overlapped. This increases the latency of the ID stage and could affect the processor's clock cycle time if the ID stage becomes the longest-latency stage. We have:

Clock cycle time before change	Clock cycle time after change
250 ps (D-Mem in MEM stage)	No change (150 ps + 90 ps< 250 ps)