Computer Architecture
Quiz 2

December 3, 2024

Student id: Name:

---

1. (40%) Question 1
Consider a version of the pipeline from Section 4.5 that does not handle data hazards (i.e., the programmer is responsible for addressing data hazards by inserting NOP instructions where necessary). Suppose that (after optimization) a typical n instruction program requires an additional $0.5n$ NOP instructions to correctly handle data hazards. Here NOP stands for the "no operation" instruction. Question 1 ANS:

   (a) (8%) Given that the cycle time of this pipeline CPU without forwarding is 200 ps (1 ps = 10^{-12} seconds), and also given that adding forwarding hardware will reduce the number of NOPs from $0.5n$ to $0.05n$, but increase the cycle time to 250 ps, what is the speedup ratio of this new pipeline CPU compared to the one without forwarding?
   200*1.5/250*1.05 = 1.143... (or 8/7)
   answer : 2, formula: 6 (even with reciprocal)

   (b) (8%) Based on the cycle time in (a), at most how many NOPs (as a ratio of code instructions) can remain in the typical program that runs faster on the pipeline CPU with forwarding? Hint:(no forwarding speed/with forwarding speed) ratio should be no more than 1. If at most 0.5n NOPs can remain, the answer will be 0.5.
   200*1.5/250 = 1.2, 1.2-1 = 0.2
   Answer : 3, Formula : 4, correct time estimate with/without forwarding: 1

   (c) (8%) Repeat (b); Let x represent the number of NOP instructions relative to n without forwarding (that is, an n-instruction program requires an additional $x \cdot n$ NOP instructions). Your answer will be with respect to x. Hint: If at most $((100x - 20)/100)n$ NOPs can remain, the answer will be $(100x - 20)/100$; NOPs ratio with forwarding can be any number no less than zero and no greater than 1.
   $200 * (1 + x) >= 250 * (1 + NOP)$, ans = $(200x - 50)/250$ or $(4x - 1)/5$
   Answer : 3, Formula : 4, correct time estimate with/without forwarding: 1

   (d) (8%) Based on the cycle time in (a), can a program without forwarding with only $0.2n$ NOPs possibly run faster on the pipeline CPU with forwarding? Explain why or why not. Hint: NOPs ratio with forwarding can be any number no less than zero and no greater than 1.
   It's impossible, because time without forwarding: $200 * 1.2 = 240$; time with forwarding is at least $250 * 1.0 = 250$, it's detined to be larger than time without forwarding. ; from c, if x = 0.2, NOP ratio after forwarding should be less than a negative number
   "Impossible" : 3, Explanation is reasonable : 5

   (e) (8%) Based on the cycle time in (a). At minimum how many NOPs (as a ratio of code instructions) must be there in a program **without forwarding** before it can possibly run faster on the pipeline with forwarding? Hint: If at minimum $0.5n$ NOPs can remain, the answer will be 0.5.
   $200 * (1 + x) >= 250$, x = 0.25 (standard answer)
   or $200 * (1 + x) >= 250 * 1.05$,x = 0.3125 (not standard but question's hint may not be enough)
   Answer : 3, Formula : 4, correct time estimate with/without forwarding: 1

2. (40%) Question 2
Consider the following loop. The instructions below are operated under the 5-stage pipelined RISC-V CPU. The pipeline has full forwarding support, and the correct branch results are obtained in the ID stage, while the branch prediction unit is in the IF stage. Assume the initial value in register x10, x11, x12, and x13 is 0. Given the following code:

```
        addi x14, x0, 3
        addi x15, x0, 5
        Loop_1:
            addi x12, x12, 1
            beq  x12, x14, END
            addi x13, x0,  0
        Loop_2:
            addi x13, x13, 1
            add  x11, x12, x13
            add  x10, x10, x11
            beq  x13, x15, Loop_1
            jal Loop_2
        END:
```

Question 2 ANS:

(a) (12%) List all the results when executing a branch instruction. Mark Y for branch taken and N for branch not taken.

#th branch instruction	1	2	3	4	5	6	7	8	9	10
Result(taken or not)	N	N	N	N	N	Y	N	N	N	N

#th branch instruction	11	12	13	14	15	16	17	18	19	20
Result(taken or not)	N	Y	Y							

-1% each error (14th - 20th instruction should be left blank, or will still be counted as error)

(b) (4%) Please give the accuracy (in percentage) of always-taken prediction unit for operating the instructions above.
accuracy = 3/13 = 23.08%
If (a) is wrong but give correct equation (always-taken prediction) here, get 2%

(c) (10%) If a 2-bit branch predictor in the figure is utilized as the prediction unit, please list the state and predict result (taken or not) when executing a branch instruction.(The initial state of the predictor is state 1)

#th branch instruction	1	2	3	4	5	6	7	8	9	10
state	1	2	3	3	3	3	2	3	3	3
Predicted result(taken or not)	Y	N	N	N	N	N	N	N	N	N

#th branch instruction	11	12	13	14	15	16	17	18	19	20
state	3	3	2	1						
Predicted result(taken or not)	N	N	N	Y						

-1% each error (14th (15th) - 20th instruction should be left blank, or will still be counted as error)

(d) (6%) Please give the accuracy (in percentage) of the 2-bit branch prediction unit in (c).
accuracy = 9/13 = 69.23%
If (a) or (b) is wrong but give correct equation (2-bit branch predictor result) here, get 3%
If (a) or (b) is correct but give wrong answer here because of computing error (show process), get 4%

(e) (8%) What is the final value in register x10?

(2+3+4+5+6)+(3+4+5+6+7) = 45

Show reasonable process with wrong answer, get 4%

3. (30%) Question 3
Given the following code:

```
add R3, R1, R2
add R5, R1, R4
mul R6, R1, R2
add R7, R1, R5
mul R8, R3, R5
```

In this question we assume the machine has four stage pipeline (Fetch, Decode, Execute, and Write-back) with data forwarding, the needed cycle for each stage are listed below.

- Fetch (1 cycle)
- Decode (1 cycle)
- Execute (MUL: 6 cycles, ADD: 3 cycles)
- Write-back (1 cycle)

Note: Unlike the ALU we taught in class which can only execute 1 instruction at a time, here we assume the ALU are composed of 1 Adder and 1 Multiplier which are able to execute simultaneously.

Question3 ANS:

(a) (6%) Calculate the number of cycles to execute the code for a non-pipelined machine.
number of cycles = $3\times6+2\times9 = 36$
if your equations are correct but the answer is incorrect -2 points

(b) (12%) Draw the pipeline diagram for a pipelined machine. **Use F, D, E, W, and "-" to represent Fetch, Decode, Execute, Write-back, and Stall, respectively.**

Cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
ADD R3,R1,R2	F	D	E	E	E	W														
ADD R5,R1,R4		F	D	-	-	E	E	E	W											
MUL R6,R1,R2			F	D	E	E	E	E	E	E	W									
ADD R7,R1,R5				F	D	-	-	-	E	E	E	W								
MUL R8,R3,R5					F	D	-	-	-	-	E	E	E	E	E	E	W			

-3 points for the wrong diagram for each instruction

(c) (12%) Draw the pipeline diagram for a pipelined machine with 2 Adder and 2 Multiplier.

Cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
ADD R3,R1,R2	F	D	E	E	E	W														
ADD R5,R1,R4		F	D	E	E	E	W													
MUL R6,R1,R2			F	D	E	E	E	E	E	E	W									
ADD R7,R1,R5				F	D	-	E	E	E	W										
MUL R8,R3,R5					F	D	E	E	E	E	E	E	W							

-3 points for the wrong diagram for each instruction