CSci370 Computer Architecture: Homework 5

Due date: On or before Thursday, May 07, 2020
Absolutely no copying others' works **Name: _____**

- The purpose of homeworks is for students to practice for the exams without others' help, so the penalty of mistakes will be minor.
- Without practicing for the exams properly, students would not be able to do well on the exams.

I. In this exercise, we examine how pipelining affects the clock cycle time of the processor. Problems in this exercise assume that individual stages of the datapath have the following latencies (time needed to do their work):

IF	ID	EX	MEM	WB
220 ps	300 ps	140 ps	240 ps	150 ps

Also, assume that instructions executed by the processor are broken down as follows:

ALU	`beq`	`lw`	`sw`
40%	25%	15%	20%

1. (09%) What is the clock cycle time in a pipelined and non-pipelined processor?
   Ans>
   - Non-pipelined:
   - Pipelined:

2. (09%) What is the total latency of an `lw` instruction in a pipelined and non-pipelined processor?
   Ans>
   - Non-pipelined:
   - Pipelined:

3. (09%) If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor?
   Ans>
   - Stage to be split:
   - The new clock cycle time:

4. (09%) Assuming there are no stalls or hazards, what is the utilization (% of cycles used) of the data memory?
   Ans>

5. (09%) Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit?
   Ans>

6. Instead of a single-cycle organization, we can use a multi-cycle organization where each instruction takes multiple cycles but one instruction finishes before another is fetched. In this organization, an instruction only goes through stages it actually needs (e.g., `sw` only takes 4 cycles because it does not need the WB stage). Compare clock cycle times and execution times with single-cycle, multi-cycle, and pipelined organization (i.e., find the speedup for the following two questions).
   a. (09%) How many times is the pipelined execution time as fast as the multi-cycle execution time?
      †Hint:
      - speedup of pipeline
        = execution time$_{\text{(multi-cycle processor)}}$ ÷ execution time$_{\text{(pipelined processor)}}$
      - Instruction counts and clock cycle times of both processors are the same.

      Ans>

   b. (09%) How many times is the pipelined execution time as fast as the single-cycle execution time?
      †Hint:
      - speedup of pipeline
        = execution time$_{\text{(single-cycle processor)}}$ ÷ execution time$_{\text{(pipelined processor)}}$
      - Instruction counts and CPIs of both processors are the same.

      Ans>

II. Consider the following loop.

```
loop: lw   $2, 0($2)          # I1
      and  $2, $2, $3         # I2
      lw   $2, 0($2)          # I3
      lw   $2, 0($2)          # I4
      beq  $2, $0, loop       # I5
```

Assume that perfect branch prediction is used (no stalls due to control hazards), that there are no delay slots, and that the pipeline has full forwarding support. Also assume that many iterations of this loop are executed before the loop exits.

1. (09%) Indicate dependences and their types (i.e., RAR, RAW, WAR, or WAW).
   †Comments:
   - Each dependence includes a type, a register, and two different instructions; e.g., RAW on $2 for I1 and I2.
   - The last instruction to be considered is I5; i.e., there is no need to consider the dependencies from I5 to I1.

   Ans>

2. (19%) Show a pipeline execution diagram for the third iteration of this loop, from the cycle in which we fetch the first instruction of that iteration up to (but not including) the cycle in which we can fetch the first instruction of the next iteration. Show all instructions that are in the pipeline during these cycles (not just those from the third iteration).
   †Comment: Use the symbol "—" for a stall.
   Ans>

Executed Instructions	Pipeline Cycles							
	n+1	n+2	n+3	n+4	n+5	n+6	n+7	n+8

3. (09%) How often (as a percentage of all cycles) do we have a cycle in which all five pipeline stages are doing useful works?
   Ans>

Cycles per loop iteration	Cycles in which all stages do useful work	% of cycles in which all stages do useful work