CS425: Computer Systems Architecture

Homework Problem Set 2
Assignment Date: Wednesday 09/11/2022
Due Date: Monday 21/11/2022 23:59

Instructions: Solve all problems, create a .pdf file and send it via e-mail to HY425 course e-mail (hy425@csd.uoc.gr). Set the e-mail subject: HY425 - Homework 2

Problem 1 (50 points)

i. Calculate the performance (cycles) of each iteration of the loop code given below. Assume that: (a) no new instruction execution could be initiated until the previous instruction execution had completed, (b) the branch is taken, and (c) that there is a 1 cycle branch delay slot.

```
Loop:   LD     F2,0(Rx)
I0:     DIVD   F8,F2,F0
I1:     MULTD  F2,F6,F2
I2:     LD     F4,0(Ry)
I3:     ADDD   F4,F0,F4
I4:     ADDD   F10,F8,F2
I5:     ADDI   Rx,Rx,#8
I6:     ADDI   Ry,Ry,#8
I7:     SD     F4,0(Ry)
I8:     SD     F2,0(Rx)
I9:     SUB    R20,R4,Rx
I10:    BNZ    R20,Loop
```

Latencies beyond single cycle (dispatch)	
Memory LD	+2
Memory SD	+1
Integer ADD, SUB	+0
Branches	+1
ADDD	+2
MULTD	+5
DIVD	+10

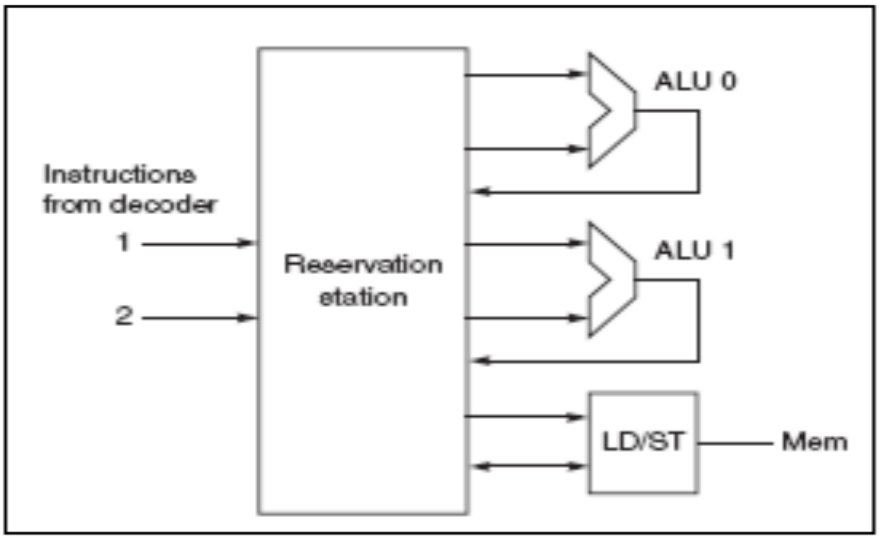
ii. How many cycles would the loop body require if the pipeline detected true data dependences and only stalled on those (in-order), rather than blindly stalling on every instruction? Show the code with <stall> inserted where necessary to accommodate the stated latencies, indicate the reason for each stall and show when each instruction completes.

iii. Reorder the instructions to improve performance of the code. How many cycles does your reordered code take in the pipeline? Take care of possible hazards and do not violate any dependencies!

iv. Hand-unroll two iterations of the loop and try to optimize. You are free to use as many extra registers as you wish. What speedup did you obtain when compared to the results of (ii) and (iii)? Color the N+1 iteration's instructions appropriately to distinguish them from the Nth iteration's.

Problem 2 (50 points)

Let's consider the out-of-order microarchitecture shown in the figure below. Assume that we have a single the Reservation Station (RS) with "many slots" and that the ALUs can do all arithmetic ops (MULTD, DIVD, ADDD, ADDI, SUB) and branches. The RS can dispatch at most one operation to each functional unit per cycle (one op to each ALU plus one memory op to the LD/ST unit) – i.e. all functional units are pipelined and may complete successive instructions out-of-order.



i. Assume that all instructions from the sequence in Problem 1 are already present in the RS (have been issued in-order), with no renaming having been done. Highlight any instructions in the code where register renaming would improve performance. Assume an infinite amount of registers and produce the register-renamed version of the code by using the following notation: the register F2 becomes F2a after the first renaming, F2a becomes F2b after the second renaming, etc. Hint: Look for RAW, WAR and WAW hazards. Assume the functional unit latencies from Problem 1.

ii. Assume that the complete register-renamed version of the code from part (i) is already present in the RS in clock cycle N (have been issued in-order and the values of the "ready" registers have been read) and assume the instruction latencies from Problem 1. Show how the RS should dispatch these instructions out-of-order, cycle-by-cycle, to obtain optimal performance on this code. Also assume that results must be written into the RS before they're available for use, i.e. no bypassing, and this takes 1 clock cycle. How many clock cycles does the code sequence take?

iii. Part (ii) allows the RS to optimally schedule these instructions. But in reality, the whole instruction sequence of interest is not usually present in the RS. Instead, various events clear the RS, and as a new code sequence streams in from the decoder, the RS must choose to dispatch what it has. Suppose that the RS is empty. In cycle 0 the first two register-renamed instructions of this sequence appear in the RS. Assume it takes 1 clock cycle to dispatch any op and assume the functional unit latencies from Problem 1. Further assume that the front end (decoder/register-renamer) will continue to supply two new instructions per clock cycle. Show the cycle-by-cycle order of dispatch of the RS. How many clock cycles does this code sequence require now?