CS425: Computer Systems Architecture

Homework Problem Set 2

Assignment: October 22, 2012

Due: October 31, 2012 – 23:59:59

Instructions: Solve all problems in a **.pdf** file and send them via e-mail to Vassilis Papaefstathiou (papaef@csd.uoc.gr). Use the 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)	**Latencies beyond single cycle (dispatch)**	
I0:	DIVD F8,F2,F0	Memory LD	+3
I1:	MULTD F2,F6,F2	Memory SD	+2
I2:	LD F4,0(Ry)	Integer ADD, SUB	+0
I3:	ADDD F4,F0,F4	Branches	+1
I4:	ADDD F10,F8,F2	ADDD	+3
I5:	ADDI Rx,Rx,#8	MULTD	+6
I6:	ADDI Ry,Ry,#8	DIVD	+14
I7:	SD F4,0(Ry)		
I8:	SD F2,0(Rx)		
I9:	SUB R20,R4,Rx		
I10:	BNZ R20,Loop		

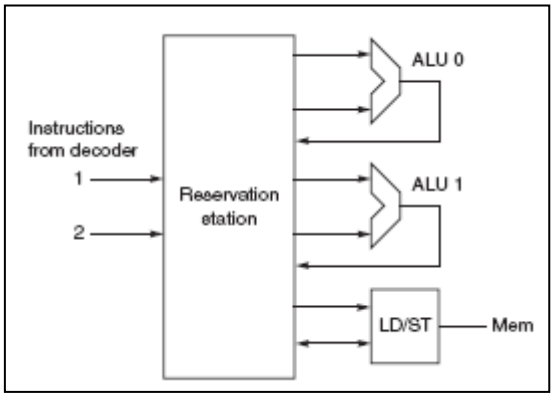
ii. How many cycles would the loop body require if the pipeline detected true data dependences and only stalled on those (in-order), rather than blindly stalling on every instruction? Show the code with <stall> inserted where necessary to accommodate stated latencies.

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.
Bonus: The best solution in the class gets an extra 20% for Problem 1!

Problem 2 (50 points)

Let's consider the out-of-order microarchitecture shown in the figure below. Assume that the ALUs can do all arithmetic ops (MULTD, DIVD, ADDD, ADDI, SUB) and branches, and that the Reservation Station (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. Suppose all of the instructions from the sequence in Problem 1 are present in the RS, with no renaming having been done. Highlight any instructions in the code where register renaming would improve performance. Hint: Look for RAW and WAW hazards. Assume the same functional unit latencies as in Problem 1.

ii. Suppose the register-renamed version of the code from part (i) is resident in the RS in clock cycle N, with latencies as given in Problem 1. Show how the RS should dispatch these instructions out-of-order, clock by clock, to obtain optimal performance on this code. (Assume the same RS restrictions as in part (i). Also assume that results must be written into the RS before they're available for use; i.e., no bypassing). How many clock cycles does the code sequence take?

iii. Part (ii) lets the RS try 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 functional unit latencies are as they were for 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?