EE334 Computer Architecture Name: SOL.

Homework Assignment 5 **Due: (9:10AM) March 21, 2012**

Problem 1. You are to examine how pipelining affects the clock cycle time of the processor. Please assume that individual stages of the datapath have following latencies:

IF	ID	EX	MEM	WB
250ps	350ps	150ps	300ps	200ps

a) What is clock cycle time of the pipeline?

cct = 350ps

b) What is the total latency of an LW instruction?

350ps x 5 = 1750 ps

c) If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original state; which stage would you split and that is the new clock cycle time of the processor?

Split ID. → $ID_1 = ID_2 = 175ps$

cct = 300ps

Problem 2. Please examine the following instruction: `LW R16,-100(R6)`

a) Which registers need to be read, and which registers are actually read?

R6 needs to be read. R16 and R6 are read

b) What does this instruction do in the EX and MEM stages?

EX: computes mem address R6 + (-100)

MEM: reads memory

Problem 3. Please analyze the following loop. Assume that perfect branch prediction is used (no stalls due to control hazards). The pipeline has full forwarding support. Assume that many iterations of this look are executed before the loop exits.

```
Loop: ADD R1, R2, R1
      LW  R2, 0(R1)
      LW  R2, 16(R2)
      STL R1, R2, R4
      BEQ R1, R9, loop
```

a) Show a pipeline execution diagram for a complete iteration of this loop, from the cycle in which the first instruction of that iteration is fetched up to (but not including) the cycle in which the first instruction of the next instruction is fetched. Please use the pipeline execution diagram below.

Instructions↓ clock→	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Loop: ADD R1, R2, R1	F	D	X	M	W													
LW R2, 0(R1)		F	D	X	M	W												
LW R2, 16(R2)			F	D	S	X	M	W										
STL R1, R2, R4				F	S	D	S	X										
BEQ R1, R9, loop						F	S	D										
ADD R1, R2, R1								F										

↑

b) Please determine the number of cycles that an iteration would take.

7 cycles

Problem 4. (This is problem 4.17 from the textbook)

Problems in this exercise assume that instructions executed by a pipelined processor are broken down as follows:

ADD	BEQ	LW	SW
40%	30%	25%	5%

4.17.1 [5] <4.6> Assuming there are no stalls and that 60% of all conditional branches are taken, in what percentage of clock cycles does the branch adder in the EX stage generate a value that is actually used?

30% × 60% = 18%

4.17.2 [5] <4.6> Assuming there are no stalls, how often (percentage of all cycles) do we actually need to use all three register ports (two reads and a write) in the same cycle?

40% only when ADD

4.17.3 [5] <4.6> Assuming there are no stalls, how often (percentage of all cycles) do we use the data memory?

25% + 5% = 30%

Each pipeline stage in Figure 4.33 has some latency. Additionally, pipelining introduces registers between stages (Figure 4.35), and each of these adds an additional latency. The remaining problems in this exercise assume the following latencies for logic within each pipeline stage and for each register between two stages:

IF	ID	EX	MEM	WB	Pipeline Register
200ps	120ps	150ps	190ps	100ps	15ps

$Clock_{single} = 760ps$

$Clock_{pipe} = 215ps$

4.17.4 [5] <4.6> Assuming there are no stalls, what is the speedup achieved by pipelining a single-cycle datapath?

$Speedup = \frac{Clock_{single}}{Clock_{pipe}} = \frac{760}{215} = 3.53$

4.17.5 [10] <4.6> We can convert all load/store instructions into register-based (no offset) and put the memory access in parallel with the ALU. What is the clock cycle time if this is done in the single-cycle and in the pipelined datapath? Assume that the latency of the new EX/MEM stage is equal to the longer of their latencies.

4.17.6 [10] <4.6> The change in 4.17.5 requires many existing LW/SW instructions to be converted into two-instruction sequences. If this is needed for 50% of these instructions, what is the overall speedup achieved by changing from the 5-stage pipeline to the 4-stage pipeline where EX and MEM are done in parallel?

Number of instructions is increased (i.e. Inst. count - IC is up)

$IC_{4\text{-stage}} = IC_{old} + 50\%(25\% + 5\%) \times IC_{old} = 1.15\, IC_{old}$

$Clock_{single} = 200 + 120 + 190 + 100 = 610ps$

$Clock_{pipe} = 215ps$

$Speedup = \frac{CPU\,time_{5\text{-stage}}}{CPU\,time_{4\text{-stage}}}$

$= \frac{1}{1.15} = 0.87$

No speed up w/ 4-stage