COMPUTER ARCHITECTURE & ASSEMBLY LANGUAGE
14:332:331
Rutgers University
Fall 2016

Homework 4
Due: Nov. 23, 2016

1. Please determine the control signals for the following instructions in the un-pipelined MIPS.

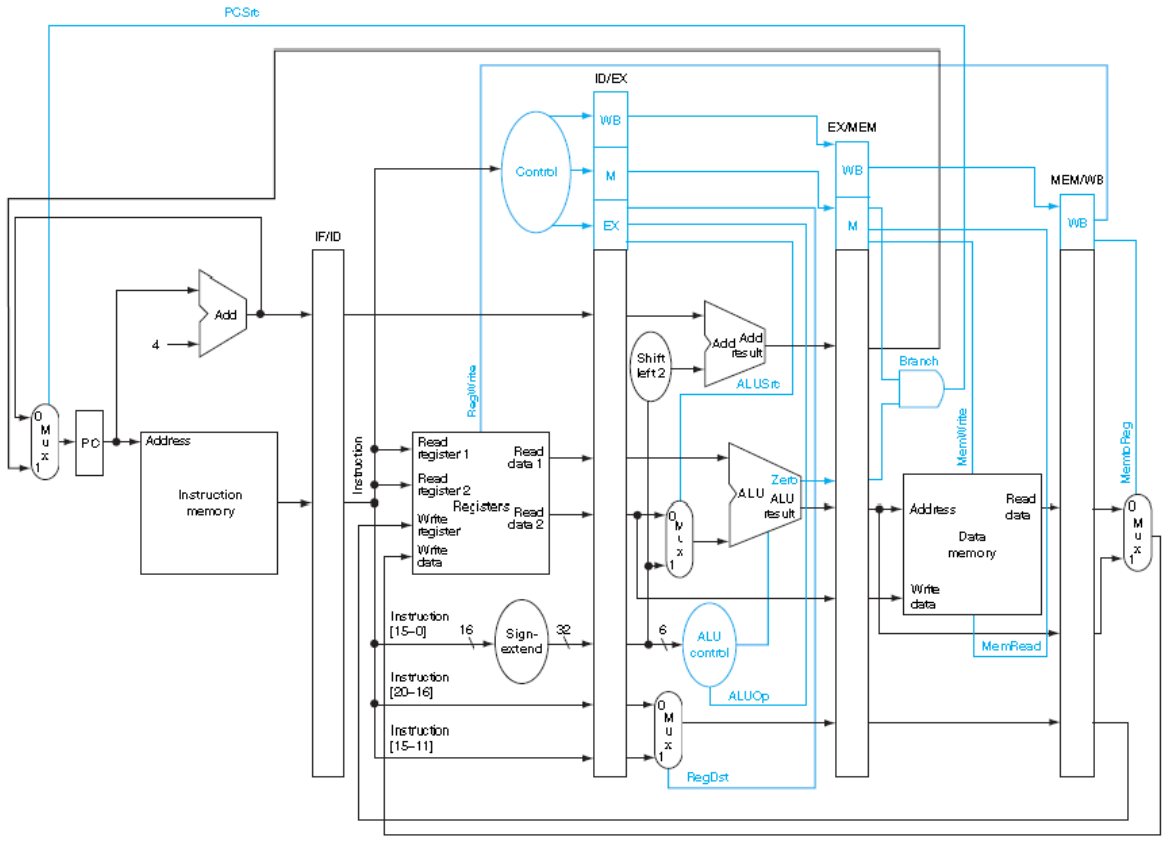
   a) Andi $t0, $t1, #12
   b) lw $t0,12($t2)
   c) sw $t4,12($t7)

Control Signal	Value	Control Signal	Value
RegDst		**ALUOp1**	
Jump		**AlUOp0**	
Branch		**MemWrite**	
MemRead		**ALUSrc**	
MemtoReg		**RegWrite**	

2. Please apply required changes (if any) to the datapath and controller of singly cycle MIPS to support the following instruction:

   swi $t1, $t2($t4) #memory address is ($t2) + ($t4)

3. We would like to run the following code on the pipeline MIPS. Please show the values of all the control signals **only** in clock cycle 6 (you can find the figure in course lectures too). Note that the first instruction (add) is fetched in clock cycle 1.



add $1, $3, $5
and $10, $8, $3
lw $4, 16($3)
sub $11, $2, $7
sw $2,100($6)

4. Consider the following instruction sequence. We want to run this code on pipelined MIPs.

add R3, R2,R7
lw R4, 20(R3)
and R6, R5,R4
sw R3, -40(R5)
add R4, R6, R3
sub R5, R6, R4

(a) Show all the data hazards (assuming that there is no forwarding).
(b) Add no-operations (stall) between these instructions as required to resolve the data hazards. Assume that there is no forwarding mechanism. How many clock cycles needed to run this code?
(c) Repeat part b but assume that there is a full forwarding mechanism.

For part b and part c show the pipeline stages as IF,ID,EX,MEM,WB for each instruction as below:

INST1: IF ID EX MEM WB
INST2: IF ……..

(d) To reduce clock cycle time, we are considering a split of the MEM stage into two stages. Please find all hazards in this 6-stage pipeline for the same instruction sequence with and without forwarding.

5. Consider a 5-stage pipelined machine (M1) which employs **data forwarding**. Assuming that the frequency of Load, Store, and ALU instructions are 30%, 20%, 50%.
Suppose the following:

40% of all load instructions are followed immediately by an instruction that uses the value just loaded in.

30% of all store instructions are immediately preceded by an R-type instruction that computes the value that is written to memory.

20% of all ALU instructions are immediately followed by another ALU instruction that uses the value just computed.

What is the throughput (**in instructions per cycle**) while executing this program? Assume all instructions go through all five stages and there is no additional pipelining overhead, except stalls.