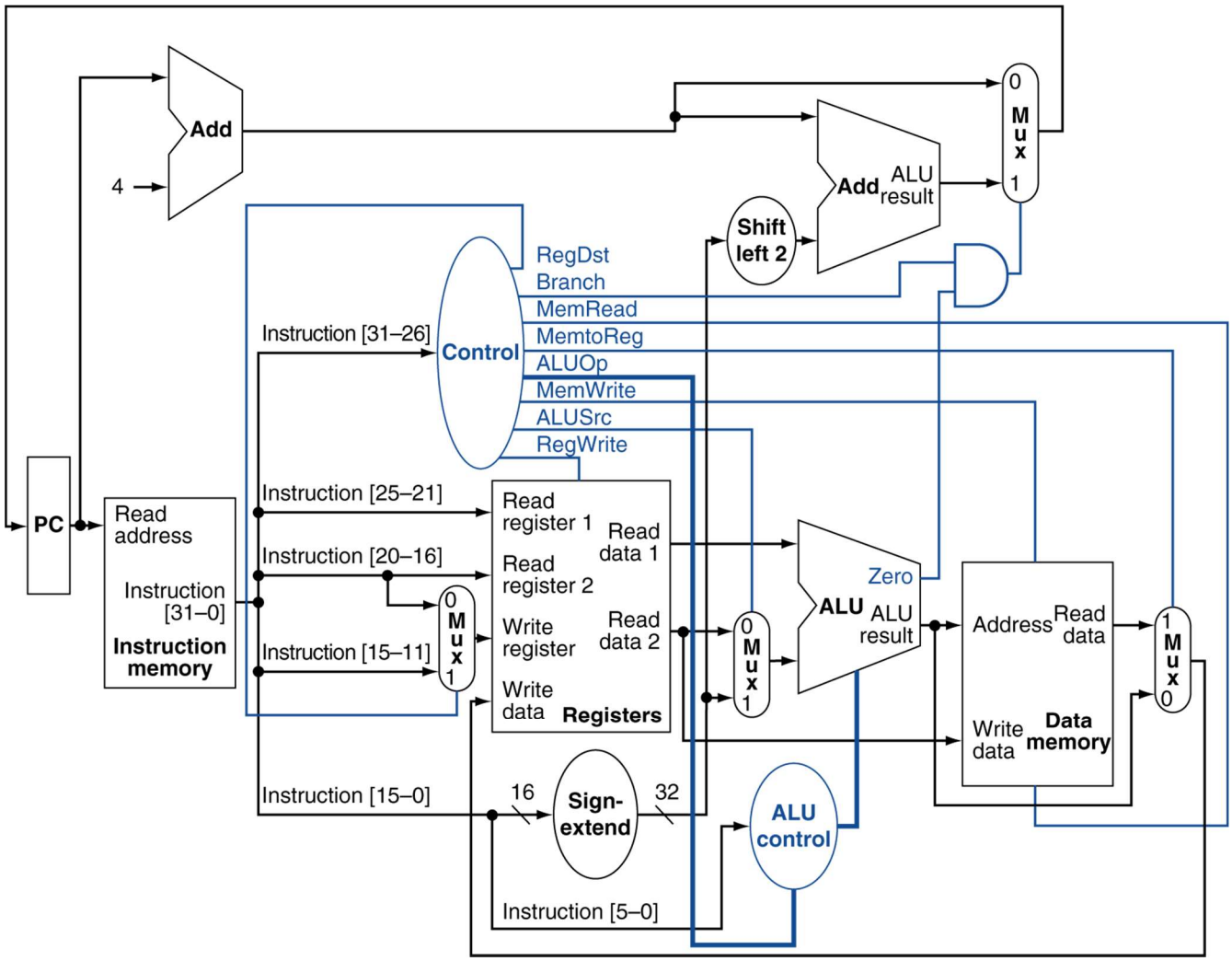
CSE 2312: Computer Organization & Assembly Language Programming Spring 2018

Homework #3

Student Name: _____

Student ID: _____

Directions: Answer the questions on the following pages. Show all applicable steps for any problems requiring the use of formulas or calculations. Submit your completed assignment electronically as a single PDF document with this completed coversheet as the first page and your name written at the top of all additional pages. You may also submit the document in person before the deadline, in which case this coversheet must be completed and stapled to your solution pages.



1. Consider the single stage CPU represented in the diagram. Complete the control line table below for the given instructions by entering 0, 1, or x for “don't care”.

```
Instruction #1:     sltu $t0,$t1,$t2
Instruction #2:     10101100011000100000000000010100
Instruction #3:     sh $t0,$t1,100
Instruction #4:     beq  $t0,$t1,L1
Instruction #5:     0x0B090001
```

	RegDst	Branch	MemRead	MemtoReg	MemWrite	ALUSrc	RegWrite
#1							
#2							
#3							
#4							
#5							

2. For each instruction in part 1, what would be the the values of the *ALUop* and *ALU control input* lines? Be sure to include only the proper number of bits for each.

3. Suppose the logic blocks in a processor have the following latencies...

Instruction fetch	Register read	ALU operation	Data access	Register write
350ps	150ps	200ps	450ps	200ps

a) In a single cycle, non-pipelined processor, what is the minimum time between instructions for an application executing only R-type instructions?

b) In a single cycle, non-pipelined processor, what is the minimum time between instructions for an application executing R, I, and J-type instructions?

c) If the logic blocks above are each implemented as individual pipeline stages, what would be the minimum time between instructions for this pipelined CPU if hazards are ignored?

4. Consider the following sequence of instructions executed on the basic 5 stage pipeline...

```
add  $t3, $t2, $t1
or   $t5, $t3, $t4
and  $t6, $t7, $t4
add  $t7, $t2, $t1
```

a) Assuming our processor has no forwarding or hazard detection, insert a minimal amount of pipeline stalls to ensure correct execution (you may stall the pipeline with the instruction `add $zero, $zero, $zero`).

b) Assuming our processor has no forwarding or hazard detection, rearrange the instructions and add stalls only if necessary to ensure proper execution. The values contained in the registers after executing your modified code should be equivalent to the unmodified execution.

5. Consider the following simple program executed on a pipelined MIPS CPU

```
        addi $s0, $zero, 3
LOOP:   lw $t0, 0($s1)
        add $t1, $t0, $t2
        add $t3, $t1, $tt4
        addi $s0, $s0, -1
        bne $s0, $zero, LOOP
        addi $s1, $zero, $zero
```

a) How many cycles will be required for the program to fully execute on a 5 stage pipeline with forwarding and perfect branch prediction?

b) How many cycles will be required for the program to fully execute on a 5 stage pipeline that has no forwarding or branch prediction, but automatically inserts a minimal number of stalls?

1. sltu $t0, $t1, $t2 | 0 | $t1 | $t2 | $t0 | 0 | 0x2b | R-t.

sw $v0, 20($v1) | 0x2b | $v1 | $v0 | 20 | I-type

sh $t0, $t1, 100 I-type with address. Basically a store

beq B-type

slti I-type

20pts
-1 per blank
20pts off max

	RegDst	Branch	MemRead	MemReg	MemWrite	ALUSrc	RegWrite
1	1	0	0	0	0	0	1
2	X	0	0	X	1	1	0
3	X	0	0	X	1	1	0
4	X	1	0	X	0	0	0
5	0	0	0	0	0	1	1

2.

	ALU op	ALU control	
1	10	0110	5pts
2	00	0010	5pts
3	00	0010	5pts
4	01	0110	5pts
5	10	0001	0pts

3a) 350 + 150 + 200 + 200 = 900 ps 7pts
IF ID EX WB

b) 900 + 450 = 1350 ps 7pts
MEM

c) Slowest stage = 450 ps 6pts

4. a)

	CC1	CC2	3	4	5	6	7	8	9	10	11	12
add $t3,$t2,$t1	IF	ID	EX	MEM	WB							
or $t5,$t3,$t4		Nop	Nop	IF	ID	EX	MEM	WB				
and $t6,$t7,$t4					IF	ID	EX	MEM	WB			
add $t7,$t2,$t1						IF	ID	EX	MEM	WB		

2 NOPs (assuming WB and ID can be done at once]

10pts

```
add $t3, $t2, $t1
add $zero, $zero, $zero
add $zero, $zero, $zero
or $t5, $t3, $t4
and $t6, $t7, $t4
add $t7, $t2, $t1
```

3 Nops can be used if they assume WB and ID can not be done together.
Correct answers must have 2 or 3 NOPs

b) Move $t3 as far away from assignment as possible. Can move to the end because $t5 isnt used

10pts

```
add $t3, $t2, $t1
and $t6, $t1, $t4
add $t7, $t2, $t1
or $t5, $t7, $t4
```

	CC1	CC2	3	4	5	6	7	8	9
add $t3,$t2,$t1	IF	ID	EX	MEM	WB				
and $t6,$t1,$t4		IF	ID	EX	MEM	WB			
add $t7,$t2,$t1			IF	ID	EX	MEM	WB		
or $t5,$t7,$t4				IF	ID	EX	MEM	WB	

can be 1 or 0 nops. See part a explanation

5a) each loop takes 7 cycles before next IF of next loop

	cc1	cc2	3	4	5	6	7	8	9	10	11	
lw	IF	ID	EX	MEM	WB							
add		NOP	IF	ID	EX	MEM	WB					
add				IF	ID	EX	MEM	WB				loop 1
addi					IF	ID	EX	MEM	WB			
bne						IF	ID	EX	MEM	WB		
lw							IF	ID	EX	MEM	WB	3

↘ = forwarding

loop goes 3 times

First instruction (addi $s0, $zero, 3) adds 1 to pipeline

Last instruction adds 5 cycles, full pipeline process

cc1 + cc2-cc7 + cc8-cc13 + cc14+cc19 + cc20-cc24

addi + loop1 + loop2 + loop3 + additional

1 + 6 + 6 + 6 + 5 = 24 cycles

10 pts

b) Must add stalls for all hazards

```
addi $s0, $zero, 3
lw   $t0, 0($s1)
add  $t1, $t0, $t2   1 hazard
add  $t3, $t1, $t4   2 hazard
andi $s0, $s0, -1
bne  $s0, $zero, loop   3 hazard
addi $s1, $zero, $zero   4 hazard
```

1 = data hazard = 1 stall

2 = data hazard = 2 stalls

3 = data hazard = 3 stalls

4 = branch hazard = 2 stalls

(branch signal available after EX)

1 + 2 + 2 + 2 = 7 extra cycles

1 + (6+7) + (6+7) + (6+7) + 5 = 45 cycles

10 pts

5 also ok. See #4