NAME _____
First Last

Student ID# _____

STONY BROOK UNIVERSITY

COMPUTER SCIENCE DEPARTMENT

FINAL EXAMINATION
VERSION A

CSE 320
Spring Semester 2014
May 20, 2014

This is a closed-book exam.
A basic calculator is allowed.
The exam is 150 minutes.

The exam has 9 problems.
It is crucial to show all work done on the provided paper.

#1 _____	(15 pts)	#5 _____	(12 pts)
#2 _____	(10 pts)	#6 _____	(12 pts)
#3 _____	(8 pts)	#7 _____	(12 pts)
#4 _____	(10 pts)	#8 _____	(13 pts)

#9 _____ (8 pts)

TOTAL _____ (90 max + 10 EC)

1 [15 pts] Short Answer

(a) [10pts] True or False:

T / F Program execution time increases when clock rate increases

T / F Program execution time increases when CPI increase

T / F More powerful instructions lead to higher performance since the total number of instructions executed is smaller for a given task with more powerful instructions.

T / F Splitting the shortest stage of the five-stage RISC pipeline will result in a higher clock rate.

T / F In the MIPS five-stage pipeline **with** forwarding, data hazards never cause stalls in the pipeline.

T / F Pipelining improves performance by decreasing the latency of each instruction.

T / F A cache miss occurs when the requested memory location data is found.

T / F Load/store architectures access data memory with each instruction.

T / F Hybrid instruction set architectures result in higher processor throughput than fixed length.

T / F Virtual Memory: A page fault occurs when the virtual to physical address translation (ie. page table entry) can not be found in the TLB

(b) [2 pts] Using 1-bit ALUs we make an 8-bit ALU (operates on 8 bit numbers) and a 16-bit ALU (operates on 16-bit numbers). Consider A AND B, A + B (addition) and A OR B performed by these ALUs. Fill in the blank with "less", "same", or "more".

(i) An 8-bit ALU requires _____ time for A + B as a 16-bit ALU.
(ii) A 16-bit ALU requires _____ time for A AND B than an 8-bit ALU.
(iii) An 8-bit ALU requires _____ time for A + B than a 16-bit ALU doing A AND B.
(iv) A 16-bit ALU requires _____ time for A AND B than an 8-bit ALU doing A OR B

(c) [3pts] Suppose that we can improve the floating point instruction performance of machine by a factor of 15 (the same floating point instructions run 15 times faster on this new machine). What percent of the instructions must be floating point to achieve a Speedup of at least 4 ?

2 [10 pts] Short Answer

(a) [2 pts] What is the advantage of the multicycle datapath over the singlecycle datapath?

(b) [2 pts] A wise-man proposes to modify the memory of the multicycle datapath to read 2 addresses simultaneously . Will this improve execution (ie. reduce the number of cycles required)? Why or Why not?

(c) [2 pts] Consider a new MIPS processor which uses variable length instructions. Describe the high-level changes to the multi-cycle datapath which would be required.

(d) [2 pts] Define pipelining. What is the main advantage?

(e) [2 pts] What are structural hazards in a pipeline?

3 [8 pts] Short Answer

(a) [2pt] What is the formula for the total execution time of a program?

(b) [2 pts] Does it make sense to have a 5-way set associative cache? Why or why not? (No credit without explanation.)

(c) [2 pts] Which is a better metric for evaluating cache performance: miss rate or average access time? Why?

(d) [2 pts] Suppose a computer using set associative cache has 2^{21} words of main memory and a cache of 64 blocks, where each cache block contains 4 words. If the cache is 4-way set associative, what is the format of a memory address as seen by the cache?

4 [10 pts] Boolean Logic

Consider the boolean expression $f = x_2' + x_3' x_2 x_1'$

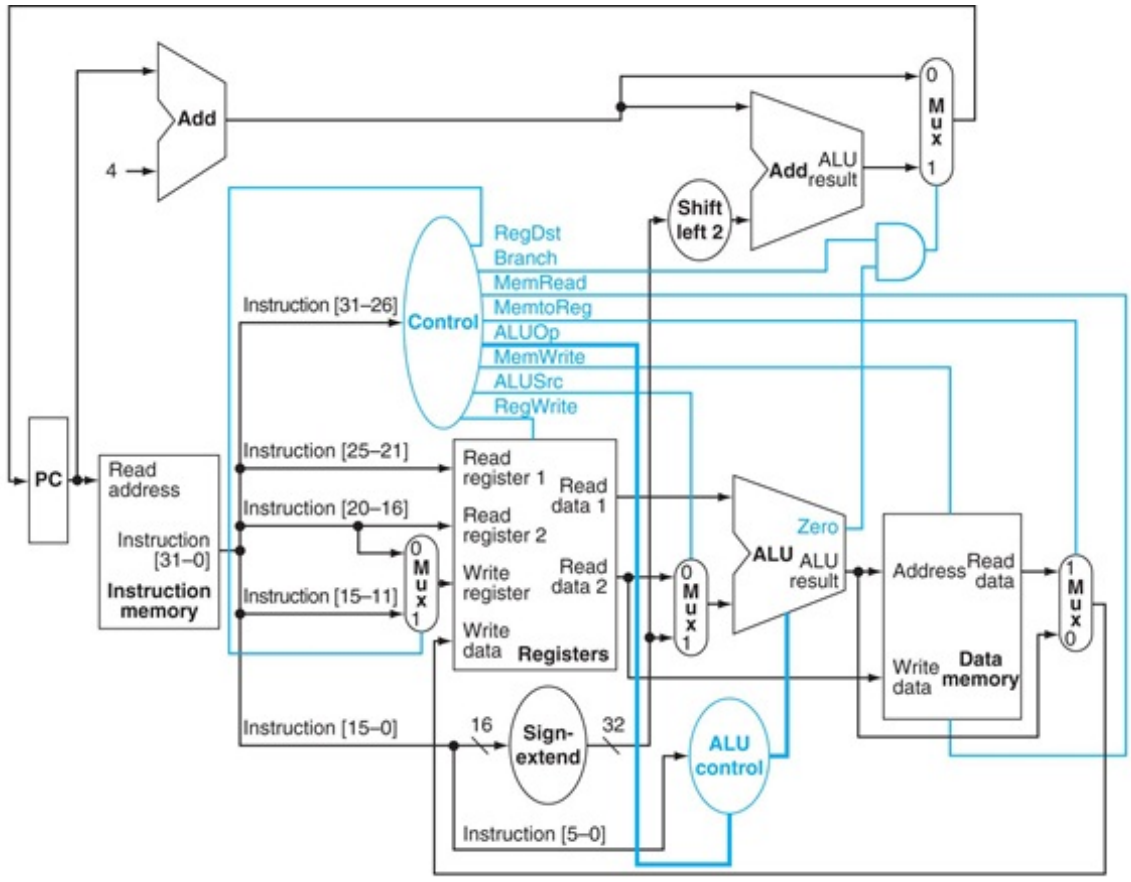
(a) [2 pts] How many literals in f? _____ How many terms in f? _____

(b) [2 pts] If f was written in POS form, how many MAXTERMS are there? _____

(c) [2 pts] Implement f using a 2-level NAND-NAND network.

(d) [4 pts] Implement f' (the inverse of f) using a 2-input decoder and 2-input (1-bit selector) multiplexors

5 [12 pts] Single-cycle Datapath Timing.
Assume the following timing for the components of the single cycle datapath.



Unit	Delay (ns)
Adder(s)	8
Instr Memory (read)	15
Register file read	7
ALU	10
Data Memory (read/write)	15
MUX/AND (each)	1
Register file write	5

(a) [4 pts] Mark the critical path of the store word instruction in the above figure. Only mark the components which contribute to the total length of the instruction (ie Do no include units which happen in parallel with other units).

(b) [4 pts] What is the instruction timing of the store word instruction? What is the timing for the load word instruction and branch on equal instruction?

(c) [2 pts] Based on only the instruction types in part (b). What is the length of the clock cycle for the single cycle datapath? Why?

(d) [2 pts] What negative consequences would happen if ALUSrc control value was stuck at value 1? Which instructions would be impacted?

6 [12 pts] Multi-cycle Datapath Timing

Assume we are using the unmodified multi-cycle datapath and finite state machine. Consider the following MIPS code:

```
        add $s4, $s1, $s0
        sub $s9, $s3, $s4
        add $s4, $s5, $s6
Label1: lw  $s2, 100($s3)
        lw  $s2, 0  ($s2)
        sw  $s2, 100 ($s4)
        and $s2, $s2, $s1
        beq $r9, $r1, Label1
        and $s9, $s9, $s1
```

(a) [2 pts] What clock cycle is the *beq* instruction fetched?

(b) [3 pts] Which instruction is executing during clock cycle 26? What stage is this instruction in (IF, ID, ALU, DATA, WB)? Which main units of the datapath are being used?

(c) [3 pts] Which values (eg. MEM[$s0], branch address, Instruction[15:0], etc) are stored in the A, B, ALUOut, and MDR registers during cycle 15 ?

```
        add $s4, $s1, $s0
        sub $s9, $s3, $s4
        add $s4, $s5, $s6
Label1: lw  $s2, 100($s3)
        lw  $s2, 0  ($s2)
        sw  $s2, 100 ($s4)
        and $s2, $s2, $s1
        beq $r9, $r1, Label1
        and $s9, $s9, $s1
```

(d) [5 pts] Assume the following delays for components in the multi-cycle datapath. What is the minimum clock cycle period for the datapath? (Multicycle diagram on next page & in back)

Memory = 200ps

Register File Read or Write = 50ps

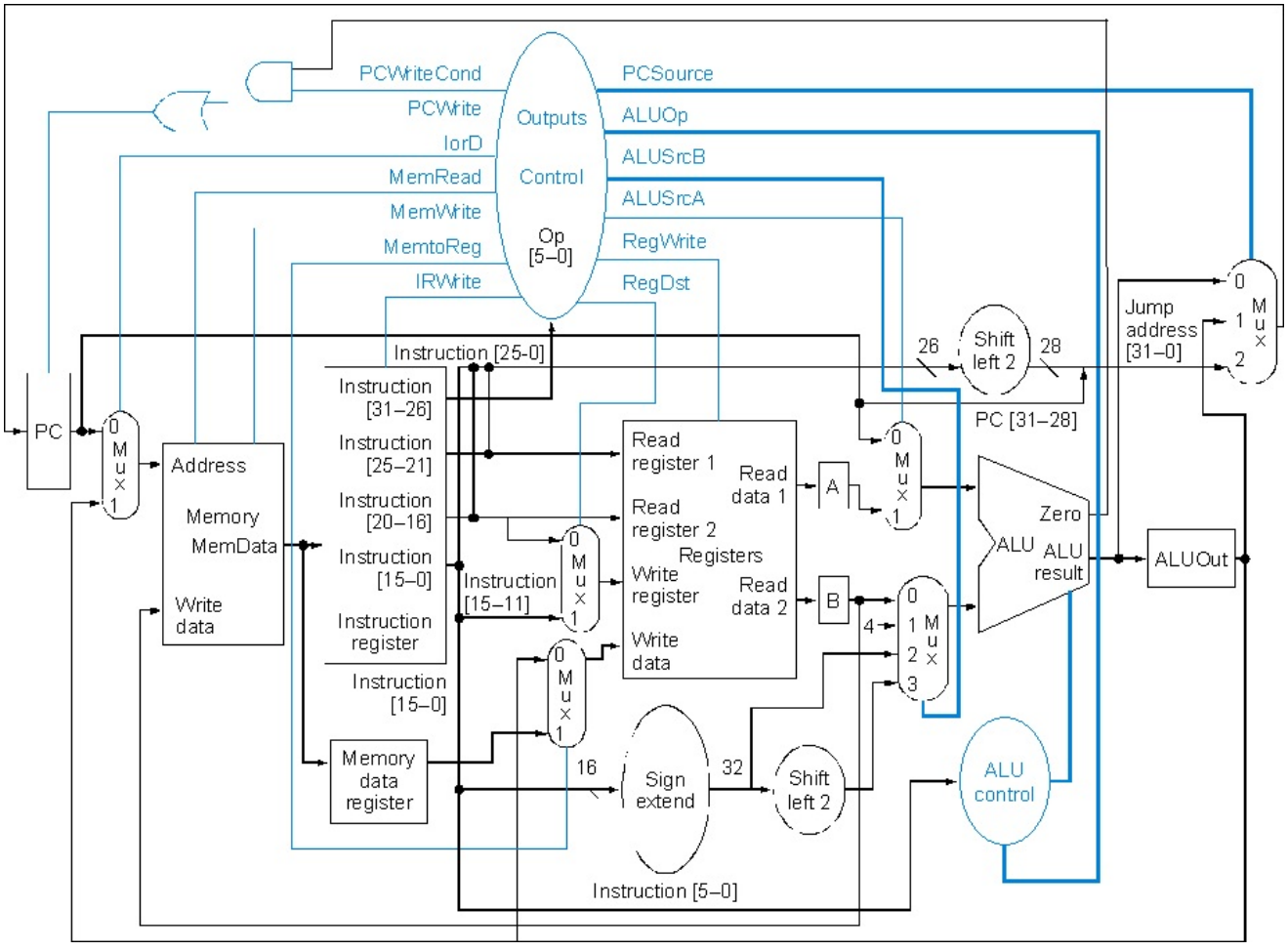
ALU = 100ps

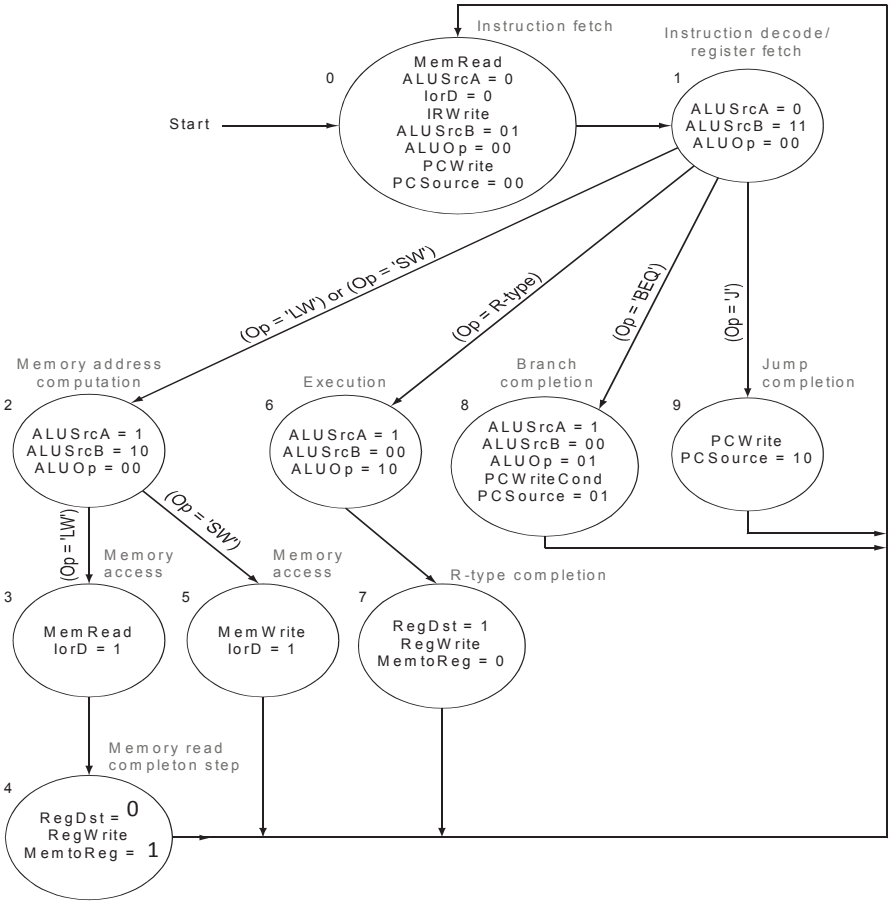
Adders, sign extension, shifters = 50ps

Logic gates (MUX, AND, OR, etc) = 2ps

7 [12 pts] Multi-cycle Datapath Implement the set on greater than zero (SGTEZ) instruction into the multi-cycle MIPS datapath. Modify any and all required changes to the datapath and the control diagram to support the instruction using the minimal amount of additional hardware and clock cycles.

```
SGTEZ $Rt, $Rs
# If $Rs >= 0
# then  $Rt = 1, else  $Rt = 0.
# Note, The values '0' and '1' are must be zero-extended to 32-bits.
```

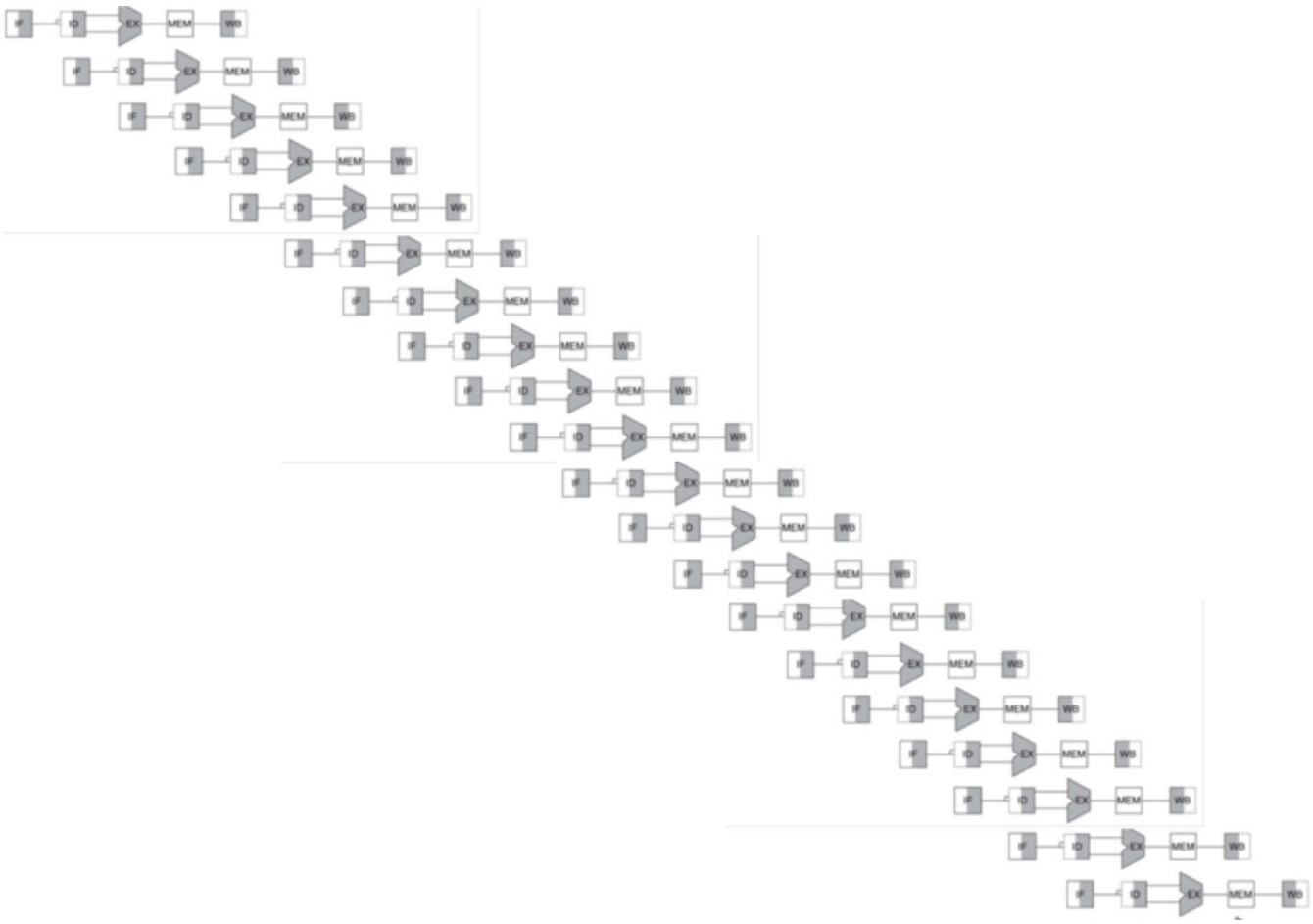



Instruction fetch
Instruction decode/
register fetch
0
MemRead
ALUSrcA = 0
IorD = 0
IRWrite
ALUSrcB = 01
ALUOp = 00
PCWrite
PCSource = 00
1
ALUSrcA = 0
ALUSrcB = 11
ALUOp = 00
Start
(Op = 'LW') or (Op = 'SW')
(Op = R-type)
(Op = 'BEQ')
(Op = 'J')
Memory address
computation
Execution
Branch
completion
Jump
completion
2
ALUSrcA = 1
ALUSrcB = 10
ALUOp = 00
6
ALUSrcA = 1
ALUSrcB = 00
ALUOp = 10
8
ALUSrcA = 1
ALUSrcB = 00
ALUOp = 01
PCWriteCond
PCSource = 01
9
PCWrite
PCSource = 10
(Op = 'LW')
(Op = 'SW')
Memory
access
Memory
access
R-type completion
3
MemRead
IorD = 1
5
MemWrite
IorD = 1
7
RegDst = 1
RegWrite
MemtoReg = 0
Memory read
completon step
4
RegDst = 0
RegWrite
MemtoReg = 1

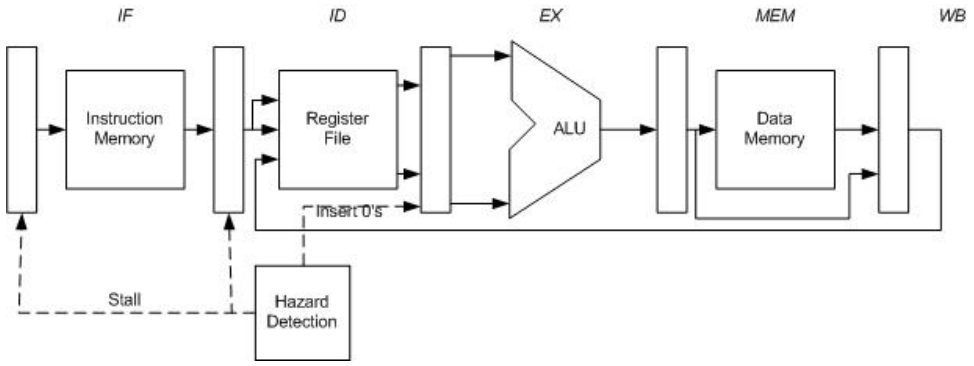
8 [13 pts] Pipelining.

```
      li $t5, 100              # LI
loop: lw $t1, 0($a1)            # LW1
      lw $t2, 4($a2)            # LW2
      add $t3, $t1, $t2         # ADD
      sw $t3, 4($a1)            # SW
      addi $a1, $a1, 4          # ADDI1
      addi $a2, $a2, 4          # ADDI2
      addi $t0, $t0, 1          # ADDI3
      bne $t0, $t5, loop      # BNE
      nor $t6, $t1, $a2          # NOR
      sub $t9, $t7, $t0         # SUB
```

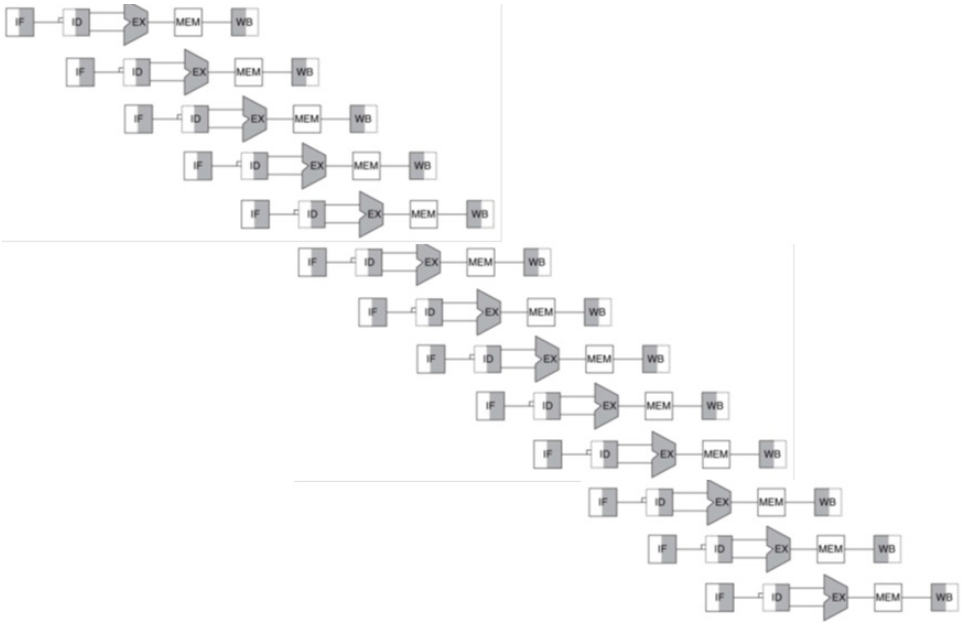
(a) [5 pts] Assume data forwarding is NOT implemented, only hazard detection is performed. For the above instruction sequence, insert stalls/"bubbles" where required. Assume the branch is not taken. Label each instruction. (Draw extra boxes as needed)



(b) [3 pts] We want to eliminate all stalls from Part (a) by adding data forwarding path(s) and reordering the code sequence. Draw in ONLY the REQUIRED forwarding paths. Specify the reordered instruction sequence. Assume the branch is not taken.



(c) [5 pts] Assume all data forwarding paths are added and that the branch is taken. Show the pipeline stalls and forwarding required starting with ADDI1 through the next occurrence of ADDI1.



9 [8 pts] Performance Metrics

Two CSE320 student implemented multi-cycle datapaths for the same MIPS instruction set. Student A and Student B's implementations each execute the following instruction types in different number of cycles.

Type	A (cycles)	B (cycles)	Instruction Mix
jump	3	3	25%
R-type	3	4	25%
beq	4	3	10%
sw	5	5	15%
lw	7	6	25%

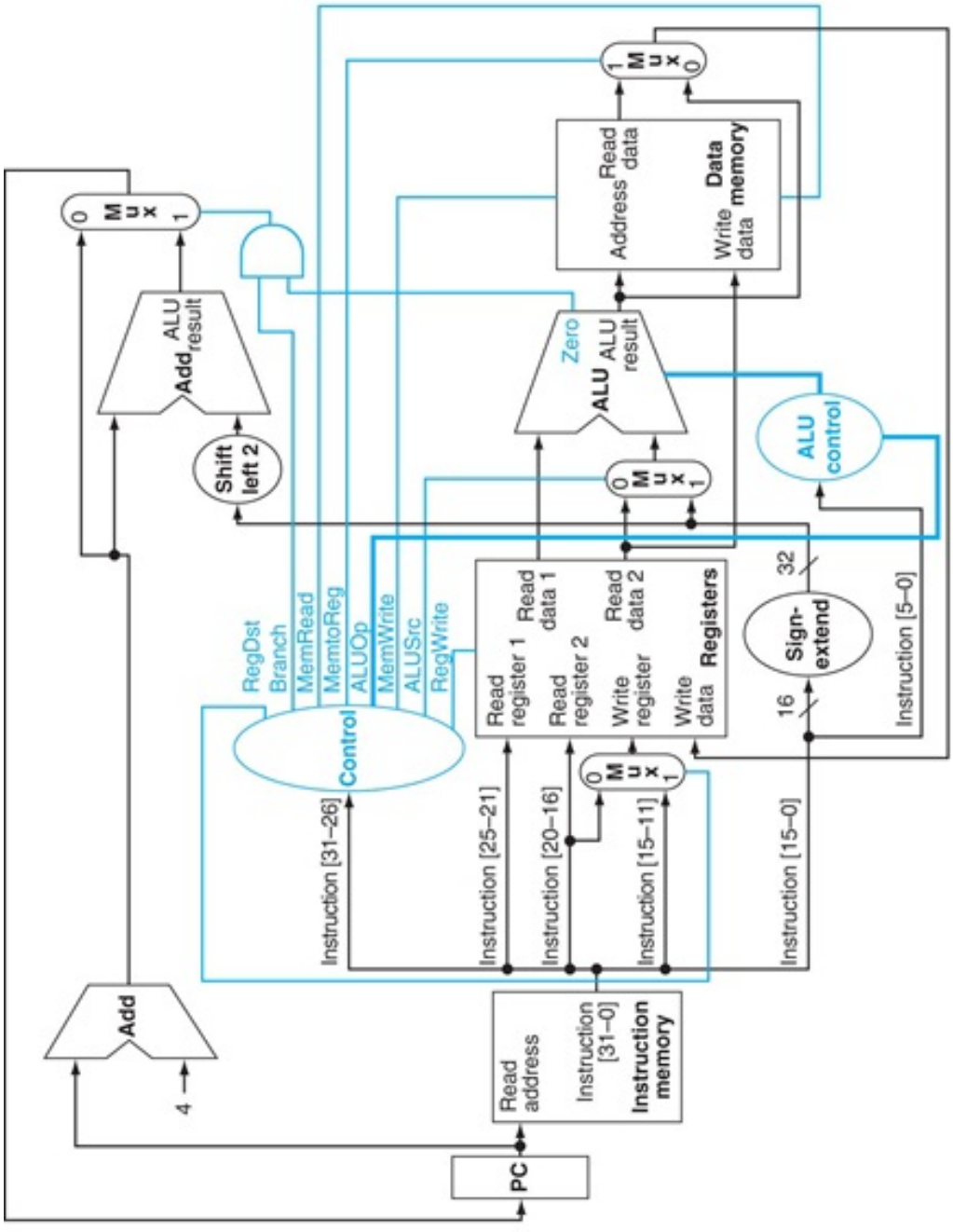
(a) [2 pts] What is the CPI for Student A's and B's implementations?

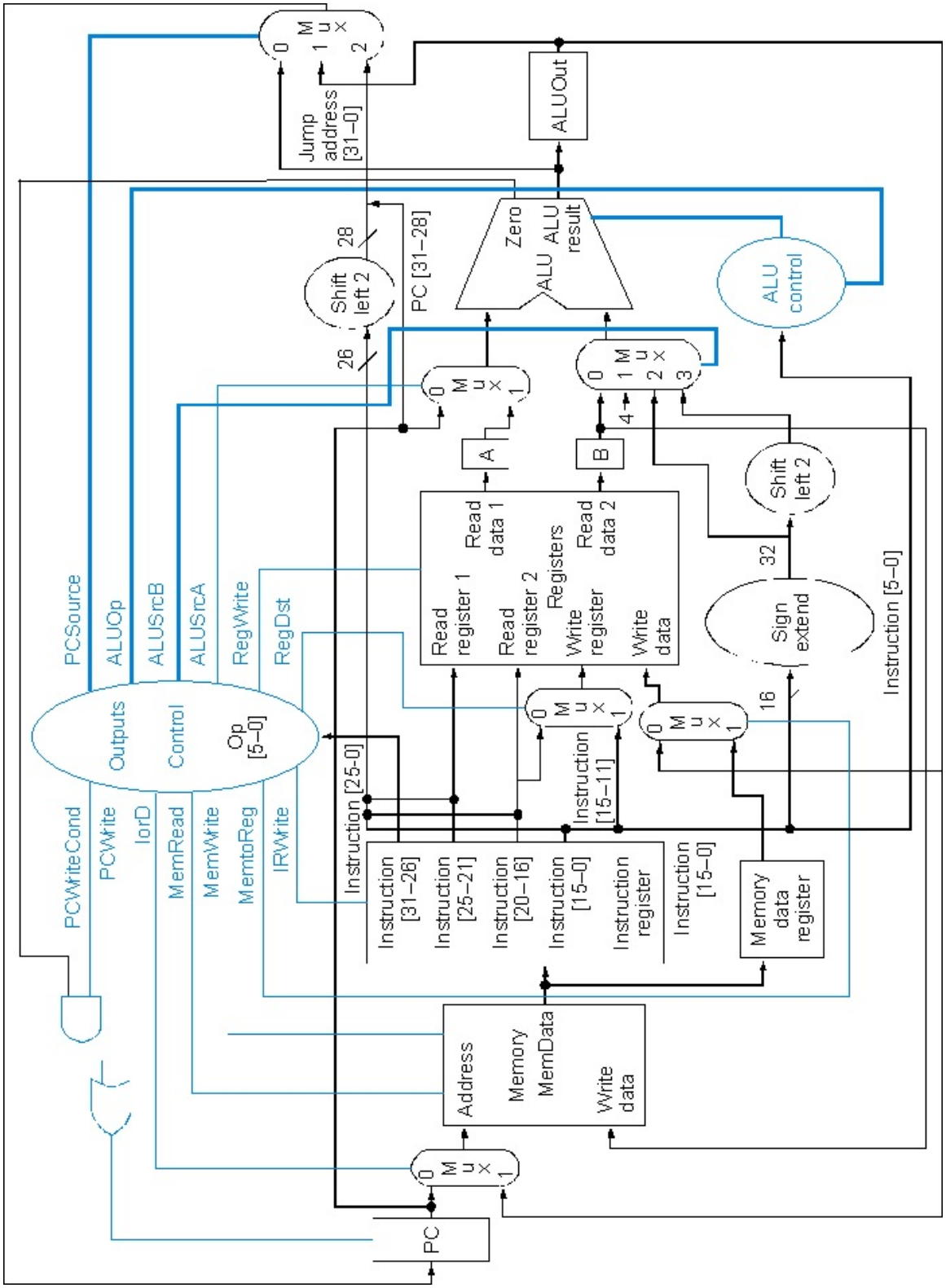
(b) [2 pts] What is the clock rate (Machine cycles per second) required for processor A to be a 1000 MIPS processor?

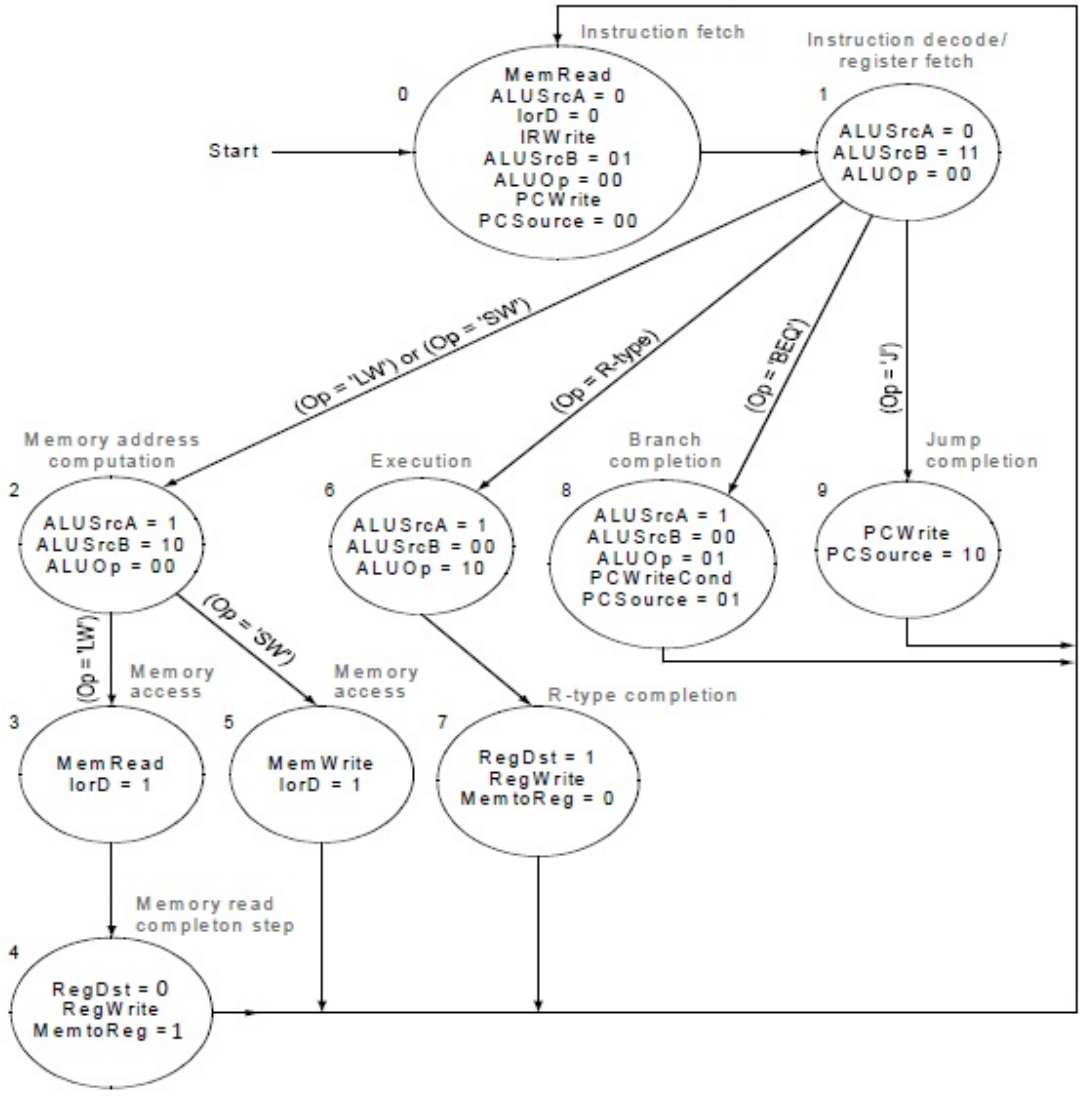
(c) [2 pts] If Student A's implementation has a 2 GHz clock, at what clock speed with Student B's implementation be faster (2 decimal places)?

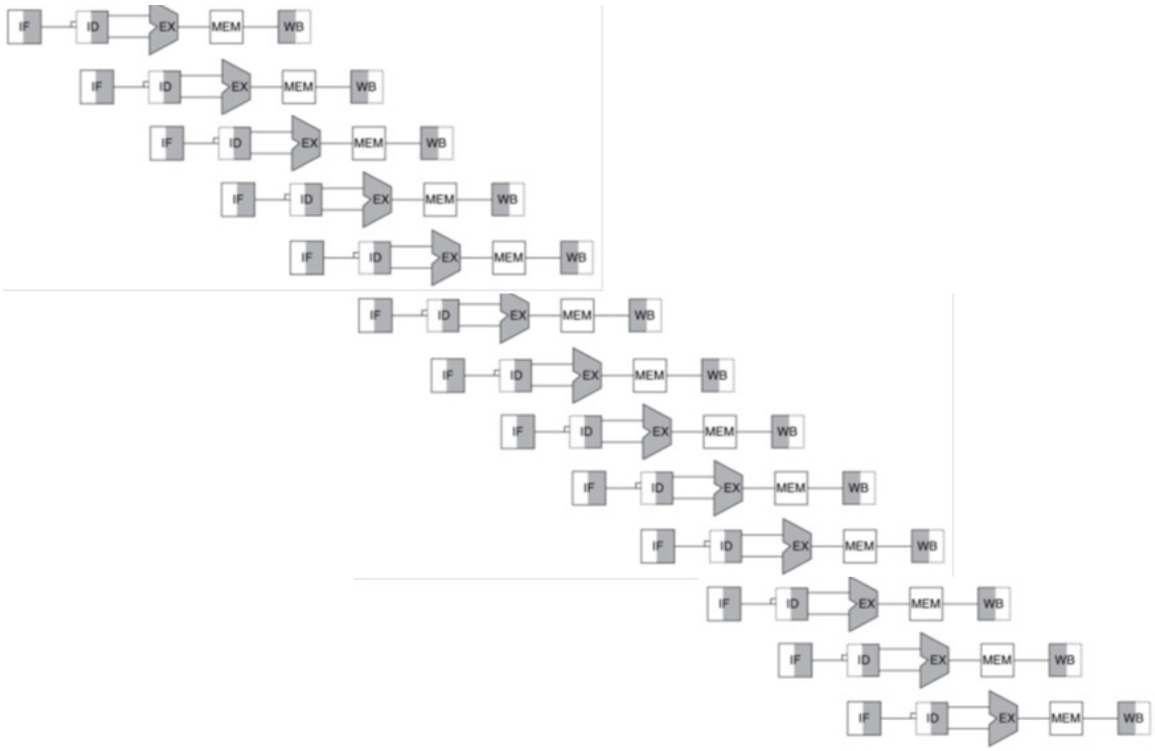
(d) [2 pts] If Suppose Student A's implementation requires an extra 20 machine cycles to retrieve/store data operands from memory. What is the effective CPI of Student A's implementation?

Extra page provided as additional work space.


M u x
ALU result
Add
Shift left 2
Zero
ALU
ALU result
Address
Read data
Write data
Data memory
ALU control
RegDst
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Control
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
32
16
Sign-extend
Instruction [5–0]
Instruction [31–26]
Instruction [25–21]
Instruction [20–16]
Instruction [15–11]
Instruction [15–0]
Add
4
Read address
Instruction [31–0]
Instruction memory
PC


M u x
0
1
2
Jump address [31–0]
ALUOut
Zero
ALU
ALU result
ALU control
Shift left 2
28
26
PC [31–28]
3
4
A
B
Shift left 2
32
Sign extend
16
Instruction [5–0]
Registers
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
PCSource
ALUOp
ALUSrcB
ALUSrcA
RegWrite
RegDst
Outputs
Control
Op [5–0]
PCWriteCond
PCWrite
IorD
MemRead
MemWrite
MemtoReg
IRWrite
Instruction [25-0]
Instruction [15–11]
Instruction [31–26]
Instruction [25–21]
Instruction [20–16]
Instruction [15–0]
Instruction register
Instruction [15–0]
Memory data register
Memory
Address
MemData
Write data
PC


Instruction fetch
Instruction decode/
register fetch
0
MemRead
ALUSrcA = 0
IorD = 0
IRWrite
ALUSrcB = 01
ALUOp = 00
PCWrite
PCSource = 00
Start
1
ALUSrcA = 0
ALUSrcB = 11
ALUOp = 00
(Op = 'LW') or (Op = 'SW')
(Op = R-type)
(Op = 'BEQ')
(Op = 'J')
Memory address
computation
Execution
Branch
completion
Jump
completion
2
ALUSrcA = 1
ALUSrcB = 10
ALUOp = 00
6
ALUSrcA = 1
ALUSrcB = 00
ALUOp = 10
8
ALUSrcA = 1
ALUSrcB = 00
ALUOp = 01
PCWriteCond
PCSource = 01
9
PCWrite
PCSource = 10
(Op = 'LW')
(Op = 'SW')
Memory
access
Memory
access
R-type completion
3
MemRead
IorD = 1
5
MemWrite
IorD = 1
7
RegDst = 1
RegWrite
MemtoReg = 0
Memory read
completon step
4
RegDst = 0
RegWrite
MemtoReg =1



Boolean Identities

In a few instances, the AND operation is represented by a dot ($\cdot$) for clarity.

$x+0$	$=$	x	identity	$x \cdot 1$	$=$	x
$x+1$	$=$	1	null	$x \cdot 0$	$=$	0
$x+x$	$=$	x	idempotence	$x \cdot x$	$=$	x
$x+\bar{x}$	$=$	1	complementarity	$x \cdot \bar{x}$	$=$	0
$\overline{(\bar{x})}$	$=$	x	involution			
$x+y$	$=$	$y+x$	commutative	$x \cdot y$	$=$	$y \cdot x$
$x+(y+z)$	$=$	$(x+y)+z$	associative	$x(yz)$	$=$	$(xy)z$
$x(y+z)$	$=$	$xy+xz$	distributive	$x+yz$	$=$	$(x+y)(x+z)$
$\overline{(x+y)}$	$=$	$\bar{x} \cdot \bar{y}$	deMorgan	$\overline{(xy)}$	$=$	$\bar{x}+\bar{y}$
$x+xy$	$=$	x	absorption	$x(x+y)$	$=$	x
$x+\bar{x}y$	$=$	$x+y$	no-name	$x(\bar{x}+y)$	$=$	xy
$xy+yz+\bar{x}z$	$=$	$xy+\bar{x}z$	consensus	$(x+y)(y+z)(\bar{x}+z)$	$=$	$(x+y)(\bar{x}+z)$