Read and fill in this page now.
Do NOT turn the page until you are told to do so.

Your name: _____

Your login name: _____

Your lab section day and time: _____

Your lab t.a.: _____

Name of the person sitting to your left: _____

Name of the person sitting to your right: _____

Problem 0	____			Total:		____	/20
Problem 1	____	____	____				
Problem 2	____	____	____	Problem 3	____	____	____

This is an open-book test. You have approximately 80 minutes to complete it; the time estimates indicate a pace sufficient to finish in 50 minutes. You may consult any books, notes, or other paper-based inanimate objects available to you. To avoid confusion, read the problems carefully. If you find it hard to understand a problem, ask us to explain it. If you have a question during the test, please come to the front or the side of the room to ask it.

This exam comprises 10% of the points on which your final grade will be based. Partial credit may be given for wrong answers. Your exam should contain four problems (numbered 0 through 3) on seven pages. Please write your answers in the spaces provided in the test; in particular, we will not grade anything on the back of an exam page unless we are clearly told on the front of the page to look there.

Relax—this exam is not worth having heart failure about.

Problem 0 (1 point, 1 minute)

Put your login name on each page. Also make sure you have provided the information requested on the first page.

Problem 1 (5 points, 12 minutes)

Figure A.19 in P&H contains a table of MIPS machine operations and their corresponding opcodes. Some printings of P&H have misprints in the table, specifically the following.

operation	*printed (incorrect) op code*	*actual op code*
sb	011001	011000
sh	011010	011001
swl	011011	011010
sw	011100	011011
swr	011111	011110

This problem concerns only the *misprinted* (i.e. incorrect) op codes.

Part a

Consider now only the rightmost three bits of the incorrect opcodes, as represented in the following truth table:

x_2	x_1	x_0	*some kind of store operation?*	x_2	x_1	x_0	*some kind of store operation?*
0	0	0	0	1	0	0	1
0	0	1	1	1	0	1	0
0	1	0	1	1	1	0	0
0	1	1	1	1	1	1	1

Give a sum-of-products expression for this truth table that contains at most four terms. Show your work for full credit.

Part b

Code your solution to part a as a Verilog module named storeCheck. Use only not, and, and or gates. You may provide any number of inputs to and and or.

```
module storeCheck (isStore, x2, x1, x0);
   output isStore;
   input x2, x1, x0;
```

Part c

Fill in the blank with the name of a gate to produce another expression that represents the truth table on the preceding page.

_____ $(x_2, x_1, x_0) + \overline{x_2} \cdot x_1 \cdot x_0$

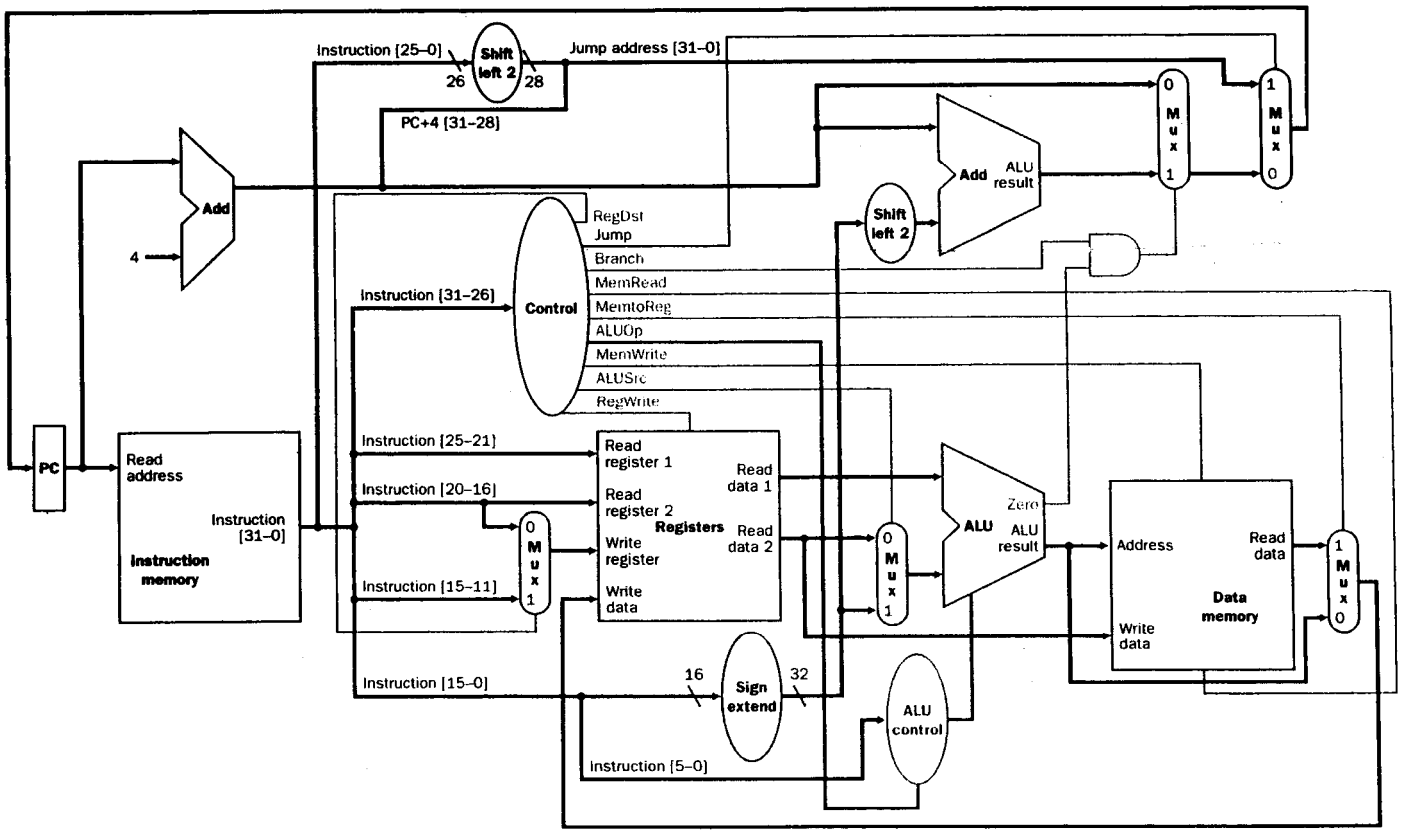
Your login name: cs61c-_____

Problem 2 (7 points, 15 minutes)

Part a

A copy of Figure 5.23 (which displays the addition of jumps to the organization of Figure 5.19) from Patterson and Hennessy appears below. Indicate how to add the jr (jump register) instruction to this implementation, making clear what components you add, what they are connected to, and how they are used. Also provide the values of all the other control signals.

Assume there is a new signal output by the instruction decoder named jumpreg. The argument register appears in bits 20-16 of the jr instruction.



Part b

Describe the effect of the signal ALUOp[1] being "stuck at 0". Also describe which instructions would fail to work because of this error.

Problem 3 (7 points, 22 minutes)

Consider the following function, which when given as arguments a pointer to the start of an integer array values and an integer n, returns the sum of the first n elements of values.

```
# $a0 contains array address, $a1 contains # elements
firstNsum:
   move $t0,$a0  # current element address
   move $t1,$0   # current element index
   move $v0,$0   # sum
loop:
   beq $t1,$a1,gotsum
   lw $t2,0($t0) # get element
   add $v0,$v0,$t2
   addi $t0,$t0,4
   addi $t1,$t1,1
   j loop
gotsum:
   jr $ra
```

Part a

Assume that there is *no* forwarding, and branch and load delays are required as described in P&H section 6.1. (Branches are resolved in stage 2.) Indicate clearly in the code above where to add nop instructions to remove *all* data and control hazards in the function. Provide only as many nops as are necessary to avoid hazards.

Part b

Indicate which of the nop instructions you added become unnecessary with the addition of forwarding circuitry from the ALU output and the memory unit to the ALU inputs.

Part c

Rewrite the loop so that each iteration takes only five clock cycles once the pipeline is full. Assume that forwarding, branch, and load delays are as described in P&H section 6.1, with branches resolved in stage 2.