This is a closed book, closed notes, closed calculator, closed neighbor test. If you crack a book, consult any written material whatsoever, use a calculator, or talk to anyone but the TA, you will receive an F in the course and your name will be forwarded to the Dean for academic review and possible further sanctions.
Question 1: Multiplying in Software

Instead of using a special hardware multiplier, it is possible to multiply instead using shift and add instructions. Show a minimal sequence of MIPS instructions for putting 5 times the value already in $s0 into $s1. Don’t worry about overflow. If you don’t recall the exact name of MIPS instructions make them up. You will receive most of the credit for a correct, minimal solution even if your instruction mnemonics (names) are slightly off. If you have doubts about the name of an instruction, give it a name and state clearly what it does.
**Question 2a: Single Cycle Datapath**

Refer to the figure on the opposite page. It is the start of a diagram of the single-cycle datapath in the book. Except all the buses, muxes, shifters, sign extenders, and adders are missing. Consult the MIPS instruction formats for R-type and I-type instructions (attached) and complete the diagram so that it is a correct one for R-type (like add) and I-type (like loads, stores, and branches). You *do* need to show the widths of buses and what instruction bits (if any) they carry. You should be able to determine this from the formats. Don’t worry about control in this part. Describe briefly the datapath for add, load, store, and branch.
Question 2b: Single Cycle Control

Show the settings for the control of your datapath schematic above when doing an add. You don’t need to show control logic, just say what the muxes are set to when doing an add. What should the MemWrite signal be set to?
Question 3: Instruction Mix

Program X executes 1.1M dynamic instructions. Of these 30% are loads and stores. Of the loads and stores, 40% are stores. What percentage of total memory references are loads? (I don’t want to hear that UCSD students need a calculator to do this problem! If you have really forgotten arithmetic, set the problem up, describe the calculations you would make—you’ll get most of the credit if correct.)
4 a: Performance

Why is MIPS (Millions of Instructions Per Second) potentially deceptive as a measure of performance. Is MFLOPS (Millions of Floating Point Operations Per Second) any better? Why or why not?
4 b: Performance and Cost Analysis

Our company (8-bit Incorporated) knows that 20% of the instructions (on average) in our customer’s programs are multiplies. Currently we multiply in software (see problem 1). But we are considering adding a hardware multiplier. Multiplies currently take 128 cycles in software. If we did them in hardware they would take only 64 cycles (1/2 the time). How much would this speed up the average customer’s program?

What if adding this hardware increased cycle time by 5%; then how much would the average customer’s program speed up or slowdown?

Let’s say we have to charge 5% more for our processors when they include a hardware multiplier. Let’s also say time = money so therefore customers won’t buy a processor for 5% more $ unless it speeds up their programs 5% or more. Are we going to sell any of our new product?
Question 5: Multiplying in Hardware.

The figure opposite (from the book) gives a multiplication algorithm that works correctly and builds up the product of a 32 bit multiplier and a 32 bit multiplicand in a 64 bit product. Modify it so that it takes one fewer step and saves a 32 bit register.
Question 6: ISA Design

Here's a proposed 8 bit ISA. It has 4 instruction formats A, B, C & D.

A-FORMAT  2 bit opcode|2 bit gp reg|2 bit gp reg|2 bit address
B-FORMAT  4 bit opcode|2 bit gp reg|2 bit unused
C-FORMAT  4 bit opcode|2 bit gp reg|2 bit gp reg
D-FORMAT  4 bit opcode|4 bit immediate value

note that there are two different opcode lengths (2 bit and 4 bit).
an address can have values 00, 01, 10, 11

There are 4 gp registers addressed 00, 01, 10, 11

Here are the instructions and opcodes

be   opcode  00  A-FORMAT
bne  opcode  11  A-FORMAT
add  opcode  1000  C-FORMAT
sub  opcode  1001  C-FORMAT
load opcode  1010  C-FORMAT
store opcode  1011  C-FORMAT
inc  opcode  1100  B-FORMAT
dec  opcode  1101  B-FORMAT
set  opcode  1110  B-FORMAT
addi opcode  1111  D-FORMAT

Is this ISA unambiguous? An ISA is unambiguous if there are not two different interpretations of the same instruction. If this ISA is ambiguous, show an ambiguous case and suggest a fix.