CS61c – Homework 45
EECS - University of California, Berkeley

TA in Charge: Zach

Submitting Your Solution

Bring your solution to class on Monday, July 18. The second half of this assignment involves some sketches that would be inconvenient to try to submit online. Feel free to turn in hand-written solutions to those problems, but make sure they are legible. I suggest typing and printing your solutions to those problems that do not require any sketches.

This is not a partnership assignment; hand in your own work.

1.

The following code fragment processes two arrays and produces an important value in register $v0. Assume that each array consists of 2500 words indexed 0 through 2499, that the base addresses of the arrays are stored in $a0 and $a1 respectively, and their sizes (2500) are stored in $a2 and $a3, respectively. Add comments to the code and describe in one sentence what this code does. Specifically, what will be returned in $v0?

```
                sll         $a2, $a2, 2
                sll         $a3, $a3, 2
                add         $v0, $zero, $zero
                add         $t0, $zero, $zero
outer:          add         $t4, $a0, $t0
                lw          $t4, 0($t4)
                add         $t1, $zero, $zero
inner:          add         $t3, $a1, $t1
                lw          $t3, 0($t3)
                bne         $t3, $t4, skip
                addi        $v0, $v0, 1
skip:           addi        $t1, $t1, 4
                bne         $t1, $a3, inner
                addi        $t0, $t0, 4
                bne         $t0, $a2, outer
```

2.

Assume that the code from problem 1 is run on a machine with a 2 GHz clock that requires the following number of cycles for each instruction:

add, addi, sll: 1 Cycle
lw, bne: 2 Cycles

In the worst case, how many seconds will it take to execute this code?

3.

Show the single MIPS instruction or minimal sequence of instructions for this C statement:

b = 25 | a;

4.

Given your understanding of PC-relative addressing, explain why an assembler might have problems directly implementing the branch instruction in the following code sequence:

```
here:          beq        $s0, $s2, there
...
there:         add        $s0, $s0, $s0
```

Show how the assembler might rewrite this code sequence to solve these problems.

5. You’ve seen this problem before. Give it another try now that you’re not until any particular time pressure. Try to really understand it and how it works.

Examine the following section of MIPS code and use it to answer the questions that follow. You may assume that the first instruction is at memory address 00001000_{hex} and the rest follow in order.

```
        j         magic
tech:   addiu     $t1, $t1, 1
        sw        $t1, 0($t0)
magic: and        $s0, $s0, $s1
        la        $t0, magic
        lw        $t1, 0($t0)
        andi      $t2, $t1, 0x3f
        addi      $t3, $0, 0x26
        bne       $t2, $t3, tech
```

a) Given that x ➔ $s0 and y ➔ $s1: with regard to the contents of $s0 and $s1, what three lines of C have the same result as the above MIPS?

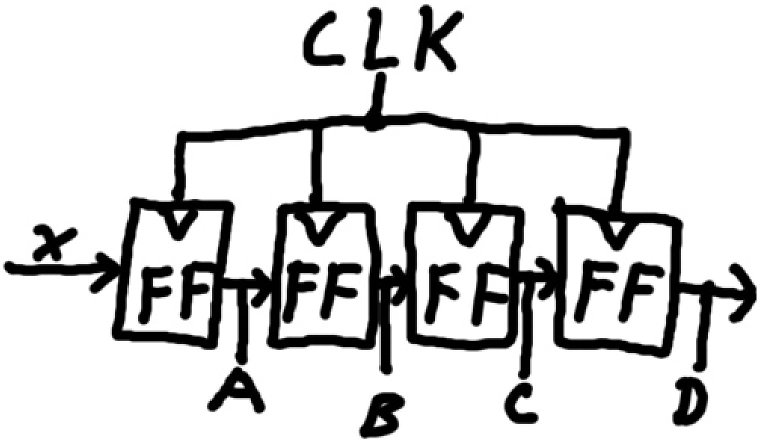
b) Come up with a single MIPS instruction which puts the same result into $s0 as the provided code.

c) Translate the first instruction (`j magic`) into machine code. Express your answer in hexadecimal.

d) Translate the final instruction (`bne $t2, $t3, tech`) into machine code.
Express your answer in hexadecimal.

6.
What is Moore's law (define it)? The Intel Pentium 4 Prescott (3.6 Ghz version), released in 2004, has roughly 125,000,000 transistors on the chip. This chip can perform about 7,000 MIPS (Millions of Instruction Per Second). Based on this information and your knowledge of Moore's law, how many transistors do you think the Intel 8088, which was released in 1979, had on the chip? What would you estimate the MIPS rating of this chip to be?

7.
Consider the circuit of Flip-Flops (FF) below. Assume that input X alternates between 1 and 0, changing half way through each **30 ns clock period** and initializing as 0 (that is, it first becomes 1 at time 15ns). Draw the detailed wave for the clock signal, input X, and the signals at points A, B, C, and D in the circuit for the first 5 clock cycles after startup. Assume that the clk-to-q delay is 7ns.



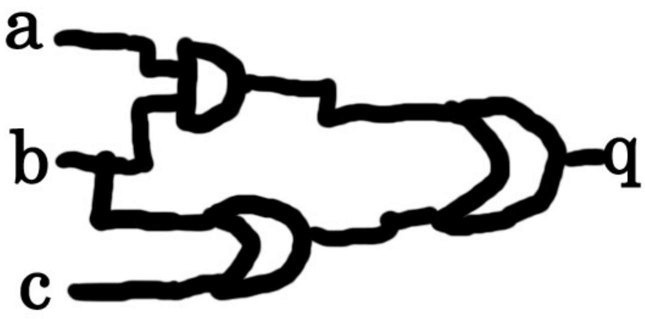
8.
Consider the accumulator discussed in the readings and presented in class. Given the following: The adder propagation delay is 2ns, the register setup time is 1 ns, the register clk-to-q is 1ns, and the clock frequency is 500MHz. Will the accumulator function correctly? If not, what would you suggest changing to fix the problem?

9.
Design a finite state machine (FSM) with the following behavior: Inputs arrive one bit at a time, one bit per clock cycle. The FSM outputs a 1 if the pattern '000' has been recognized and continues to output a 1 as long as 0s continue to be input. The FSM should output 0 at all other times. Your FSM should include a start state, but you need not worry about initialization other than that.

Do your design in three steps. First, draw the state diagram, second specify the truth table for next state and output based on present state and input, finally devise the circuit-level implementation.

10.

Derive the truth-table for the CL circuit shown below. (Remember, you do this by applying all possible input combinations, one at a time). What is the most simplified version of the expression that this circuit/truth-table generates?



11.

Write the canonical sum-of-products form of a Boolean expressions for a 4 input function whose output is 1 if and only if the number of 1's in the input is odd.

12.

Write the most simplified Boolean expression for the function represented by the truth-table below. *The solution is the OR of three AND terms, each with 2 variables.*

abc	***y***
000	0
001	0
010	0
011	1
100	0
101	1
110	1
111	1