Name:
Login Name:
Preceptor Name:
Precept Number:

Computer Science 126 — Second Midterm Exam
11/22/1999 — 7pm - 9pm

This exam has 10 questions. The weight of each question is printed in the table below and next to each question. Do all of your work on these pages (use the back for scratch space), giving the answer in the space provided. Put your name on each page (now). Sign the Honor Code pledge.

"I pledge my honor that I have not violated the Honor Code during this examination."

Question		Worth	Earned
1	Number Representation	4	
2	TOY Programming	5	
3	FSA	4	
4	Gates and MUXes	3	
5	Turing Machine	5	
6	Combinational Circuits	6	
7	Circuit Debugging	5	
8	Circuit Timing Diagram	6	
9	TOY Architecture	6	
10	Cumulative	6	
Total		50	

Name:

1. Number Representation [4]

Assume 8-bit signed two's complement for the hexadecimal representations in the following questions.

a) Convert 25 from decimal to hexadecimal.

b) Convert 0x25 from hexadecimal to decimal.

c) Convert -25 (note the negative sign) from decimal to hexadecimal.

d) Convert 0xE9 from hexadecimal to decimal.

Name:

2. TOY Machine Language Programming [5]

Assume that the following TOY program is loaded at 0x10 and it starts execution from there. Give the contents of the registers R1 and R2 in hexadecimal when the machine halts. (The TOY instruction set definition is provided in the appendix at the back of this exam.)

```
0x10 B101
0x11 B20A
0X12 B301
0X13 1111
0X14 2223
0X15 7213
0X16 0000
```

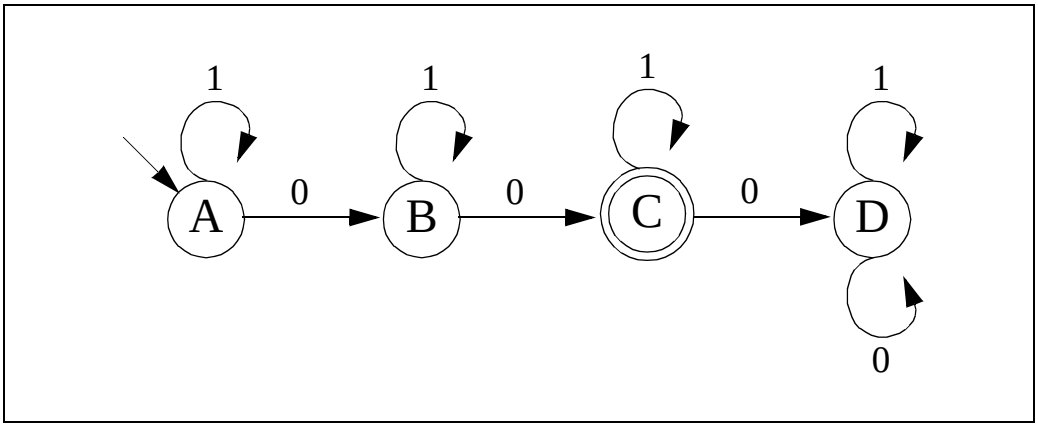
R1 =

R2 =

Name:

3. Finite State Automata [4]

The machine shown below is started in state A. State C is the accept state.



a) Is the machine deterministic?

b) What state does the machine stop in when given the string 010010?

c) What state does the machine stop in when given the string 110110?

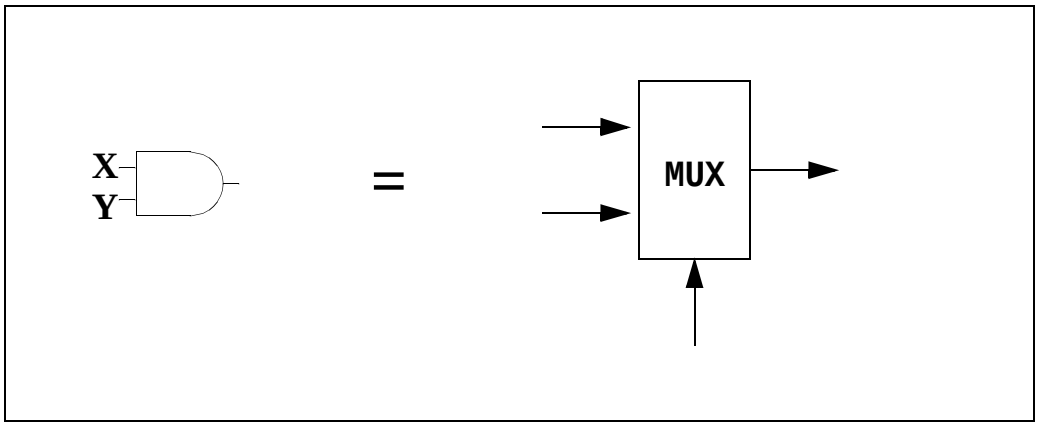
d) Give the regular expression accepted by this machine.

Name:

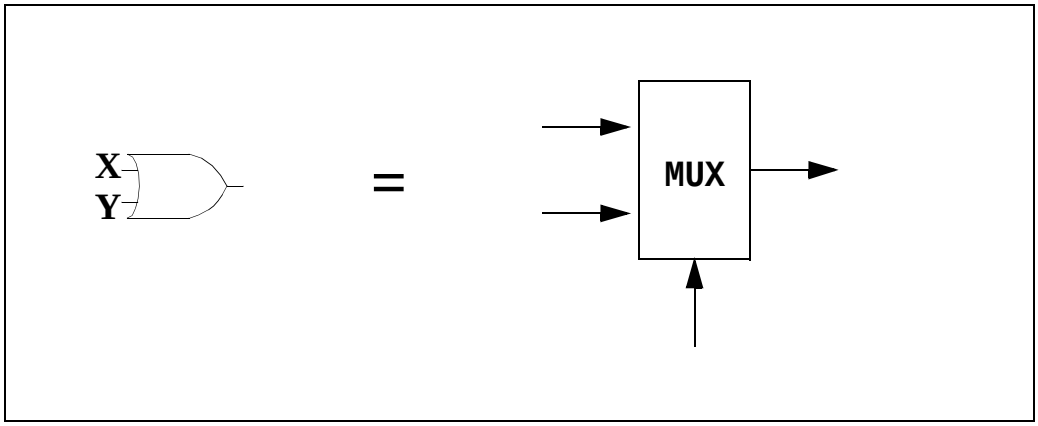
4. Logic Gates and Multiplexers [3]

Assume the availability of constant signals 0 and 1.

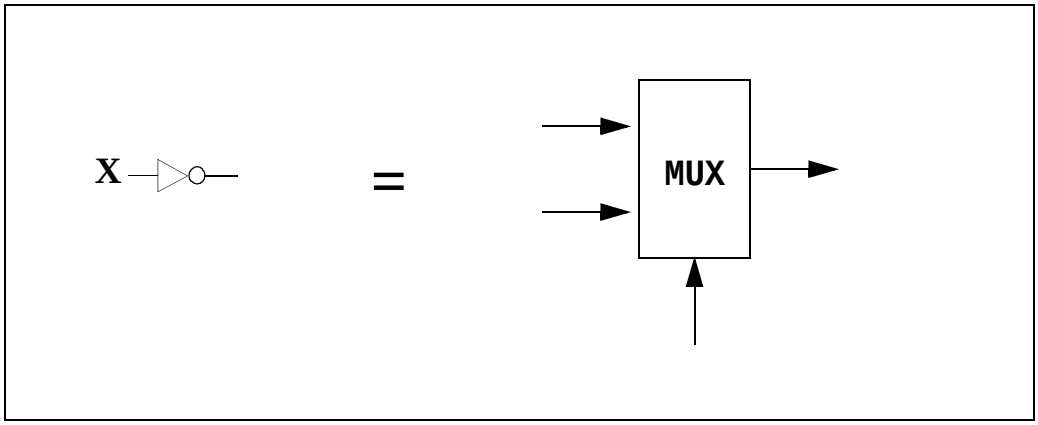
a) Use a single 2-to-1 MUX to construct a 2-input AND-gate (label the MUX input signals with X, Y, 0, or 1 in such a way that it behaves like the AND-gate to the left):



b) Use a single 2-to-1 MUX to construct a 2-input OR-gate:



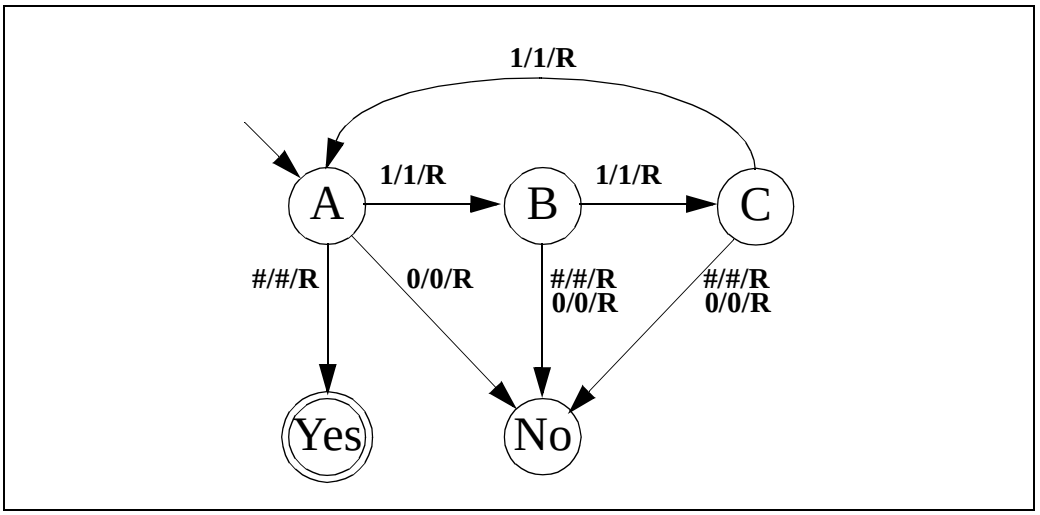
c) Use a single 2-to-1 MUX to construct a NOT-gate:



Name:

5. Turing Machines [5]

Consider the Turing machine below. The alphabet consists of three characters: “0”, “1”, and “#”. (The symbol “#” acts as a blank character and marks the two ends of a string.) State “A” is the start state.



a) What state does the machine stop in when given the following string and initial read head position?

#	#	#	#	#	#	#	#	#	#
			↑						

b) Answer the same question for the following string.

#	#	1	1	0	1	0	1	#	#
		↑							

c) Answer the same question for the following string.

#	#	1	1	1	1	1	1	#	#
		↑							

d) Does there exist a finite state automaton that can recognize the language accepted by this Turing Machine?

e) Does there exist a non-deterministic pushdown automaton that can recognize the language accepted by this Turing Machine?

Name:

6. Combinational Circuits and Two's Complement Arithmetic [6]

Build a combinational circuit that has two bits of input (X_1 and X_0) and three bits of output (Y_1, Y_0, and V).

• The two input bits (X_1X_0) are interpreted as a 2-bit two's complement integer.

• The first two output bits (Y_1Y_0) represent the result of subtracting 1 from the input integer (X_1X_0).

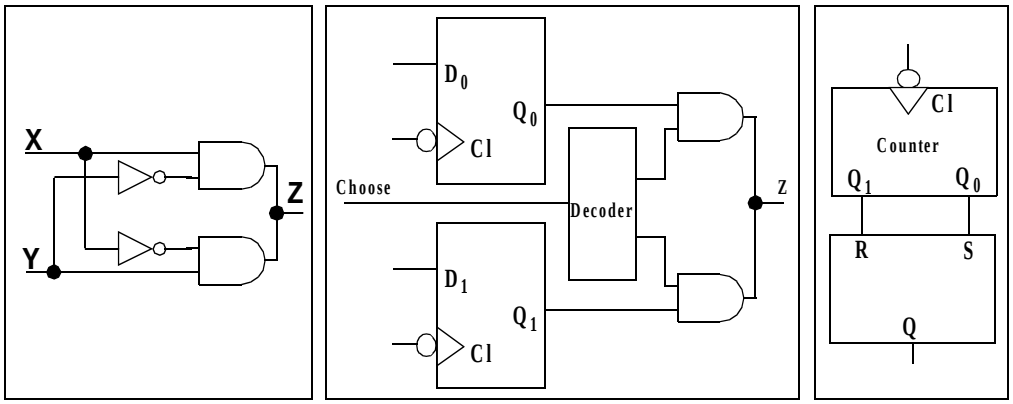
• The last output bit (V) is the overflow/underflow bit so it is 1 iff the addition results in an overflow or underflow condition. (Recall an overflow or underflow occurs when the answer does not fit in the given number of bits.)

a) Derive all the necessary truth tables.

b) Derive all the output boolean expressions.

Name:

7. Debugging Circuits [5]



(a) (b) (c)

a) The circuit in Figure (a) is intended to implement a two-input parity function.

a.1) Explain in one sentence what is wrong with this circuit.

a.2) Sketch how you would correct it.

b) In Figure (b), the intended role of the signal labeled with "Choose" is to choose the content of one of the two D flip-flops as the output "Z".

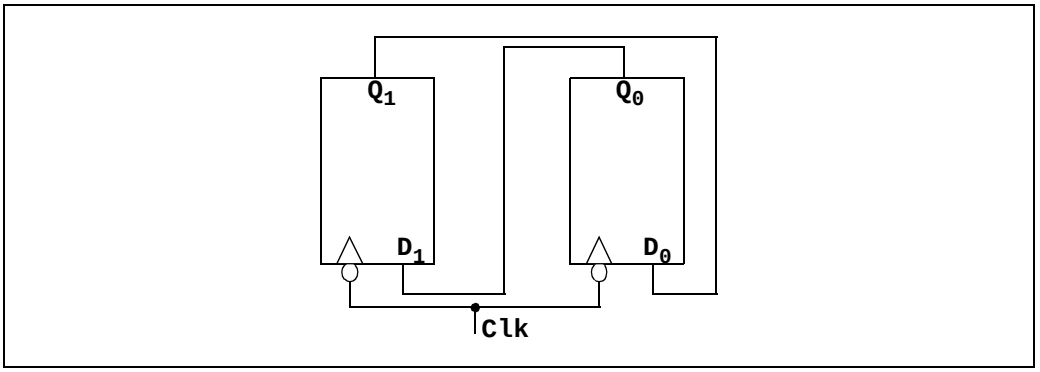
a.1) Explain in a one sentence why the circuit does not work.

a.2) Sketch how you would correct it.

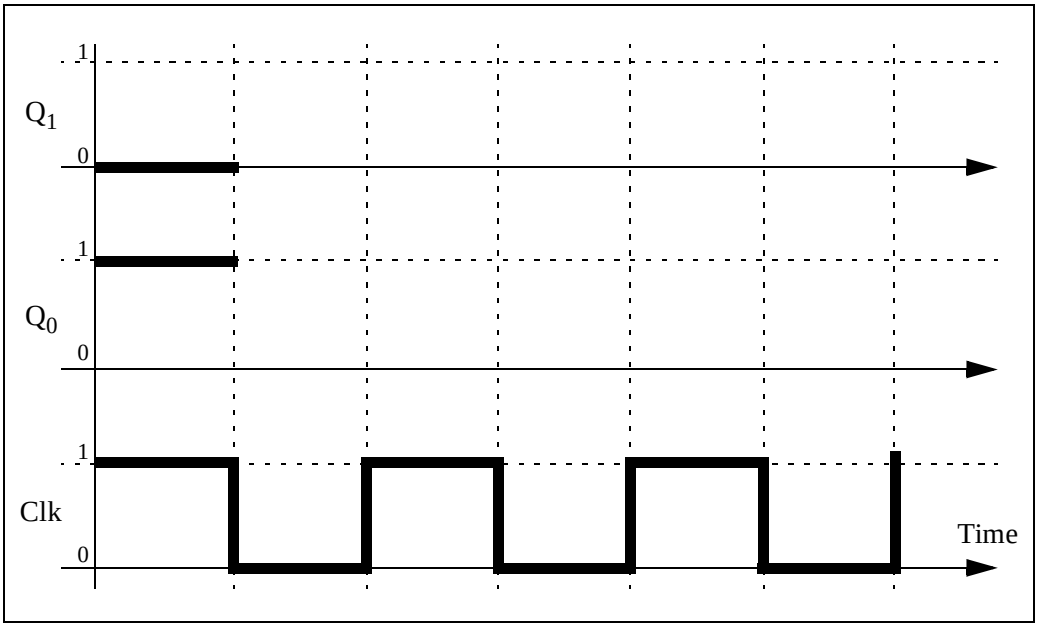
c) In Figure (c), the counter at the top generates a regular 2-bit pattern (Q_1Q_0) which is fed into an SR flip-flop at the bottom to generate a regular 1-bit pattern (Q). Explain in one sentence what problem this circuit has. (You do not need to correct it.)

Name:

8. Sequential Circuits and Timing Diagrams [6]



Two negative edge-triggered D flip-flops are connected as shown in the figure above. Q_1 is initialized to 0 and Q_0 is initialized to 1. Fill in the timing diagram below for these two signals. (You may assume that the delays introduced by the flip-flops and wires are negligible.)



Name:

9. TOY Architecture [6]

Assume that each of the following operations in the TOY datapath takes the stated amount of time (in nanoseconds):

	Operation	Time (ns)
1	**fetching an instruction from memory**	**4**
2	**reading from the register file**	**2**
3	**executing any ALU operation**	**2**
4	**loading data from memory**	**4**
5	**storing data into memory**	**5**
6	**writing the result back into the register file**	**2**
7	**all other delays (those of MUXes, controls, wires, etc.)**	**0**

(Each of these operations must finish within a single clock cycle.)

a) How much time (in nanoseconds) does each of the following instructions need to complete?

a.1) load (from memory) using indexed addressing;

a.2) store (to memory) using indexed addressing.

b)

b.1) What is the minimum cycle time for a single cycle design?

b.2) What is the minimum cycle time for a multicycle design?

c) Suppose we speed up the 4th operation in the table (loading data from memory). Instead of the 4 ns, it now takes 2 ns. (All other times remain the same, including the time required to fetch an instruction from memory.)

c.1) What is the minimum cycle time for a single cycle design?

c.2) What is the minimum cycle time for a multicycle design?

Name:

10. A Cumulative Question: Circuits, ADT, and Regular Expressions [6]

You have learned postfix notation; we put operators *after* operands using this notation. We can also use postfix notation to write boolean expressions. Suppose the AND operator is represented by "*", and the OR operator is represented by "+".

a) Draw the circuit represented by the postfix expression "`0 1 + 0 1 1 + * +`".

b) The following program evaluates postfix boolean expressions using a stack.

```
#include <stdio.h>
#include "stack.h"

main (int argc, char *argv[]) {
    char *a = argv[1];
    int i, n = strlen(a);

    stackInit();
    for (i = 0; i < n; i++) {
        if (a[i] == '+') stackPush(stackPop() | stackPop());
        if (a[i] == '*') stackPush(stackPop() & stackPop());
        if (a[i] == '0') stackPush(0);
        if (a[i] == '1') stackPush(1);
    }
    printf ("%d\n", stackPop());
}
```

If we evaluate "`0 1 + 0 1 1 + * +`" using this code, at the moment when the stack is fullest, what is the stack content? (Pictorially point out the top of the stack.)

c) Are boolean expressions regular expressions? Justify your conclusion with one sentence. (Hint: think about expressions like "1 1 1 1 + + +".)

Appendix. TOY Instruction Set

INSTRUCTION FORMATS	
Format 1: opcode, r0, r1, and r2 Format 2: opcode, r0, and 8-bit addr Indexed addressing (for format 2): if leading bit of r0 digit is 1, then addr = r1 + r2	
TRANSFER between registers and memory	
9: load A: store B: load address	r0 <- mem[addr] mem[addr] <- r0 r0 <- addr
ARITHMETIC operations	
1: add 2: subtract 3: multiply	r0 <- r1 + r2 r0 <- r1 - r2 r0 <- r1 * r2
LOGICAL operations	
C: xor D: and E: shift right F: shift left	r0 <- r1 ^ r2 r0 <- r1 & r2 r0 <- r0 >> addr r0 <- r0 << addr
CONTROL	

0: halt
4: system call
5: jump
6: jump if positive
7: jump and count
 	
8: jump and link
	halt
	print r0 on tty
	pc <- addr
	if (r0 > 0) pc <- addr
	r0--
	if (r0 != 0) pc <- addr
	r0 <- pc
	pc <- addr

Feel free to tear out this sheet.