Department of Electrical and Computer Engineering
The University of Texas at Austin

EE 306, Fall 2009
Yale Patt, Instructor
Aater Suleman, Chang Joo Lee, Ameya Chaudhari, Antonius Keddis, Arvind Chandrababu, Bhargavi Narayanasetty, Eshar Ben-dor, Faruk Guvenilir, Marc Kellermann, RJ Harden, TAs
Exam 1, October 14, 2009

Name: SOLUTION

Problem 1 (30 points):_____

Problem 2 (10 points):_____

Problem 3 (10 points):_____

Problem 4 (10 points):_____

Problem 5 (20 points):_____

Problem 6 (20 points):_____

Total (100 points):_____

Note: Please be sure that your answers to all questions (and all supporting work that is required) are contained in the space provided.

Note: Please be sure your name is recorded on each sheet of the exam.

I will not cheat on this exam.

_____
Signature

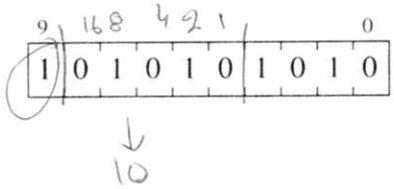
GOOD LUCK!

Name:_____

Problem 1. (30 points):

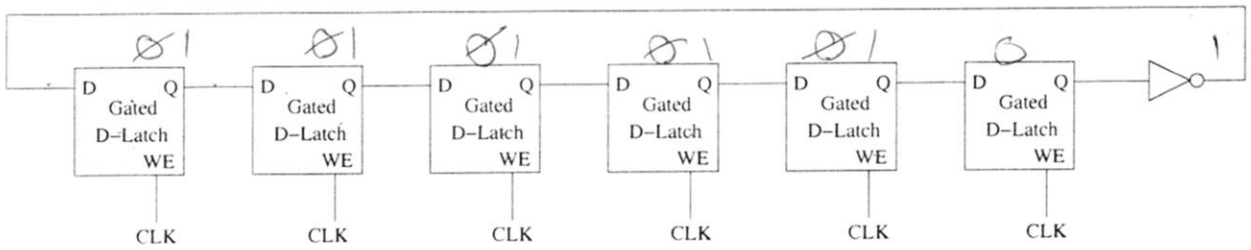
Part a. (5 points): Recall that the instruction cycle consists of six phases, not all of which are needed by all instructions. However, all instructions do need the Fetch phase and the Decode phase.

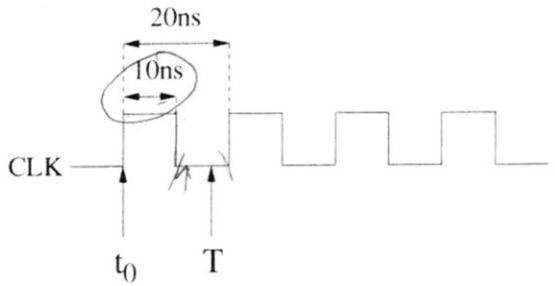
Part b. (5 points): In the spirit of the IEEE Floating Point standard, we have specified a 10-bit floating point data type. Five bits are allocated to the exponent, using an excess-12 code. Four bits are allocated to the fraction. Convert the Floating Point data-type number shown below to a decimal fraction:



Answer: $-13/32$

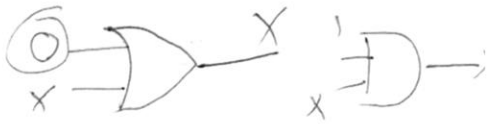
Part c. (5 points): Recall the logic circuit of problem 1 of Problem Set 3, shown below, EXCEPT the clock signals to all latches is CLK (NONE are $\overline{CLK}$). Assume as in the problem set that the initial state of the six latches just before t_0 are all 0. Assume the clock cycle is 20 nanoseconds. Assume the propagation time for a signal to get from the input of a latch to the output of that latch is 2 nanoseconds. What will be the state of the six latches at time T shown on the timing diagram?



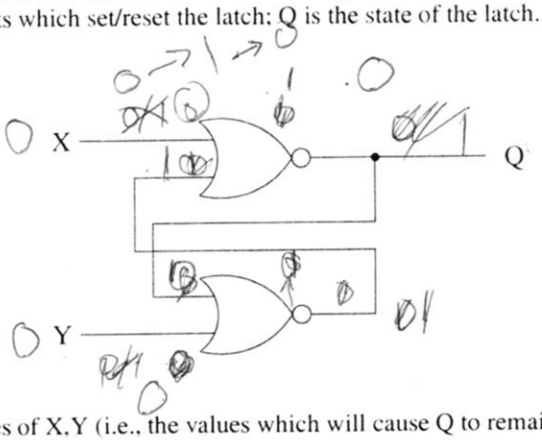


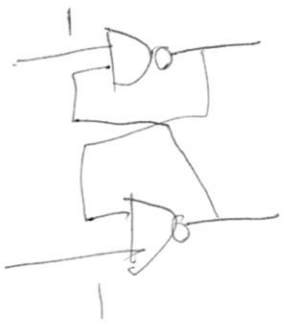
Answer: 11110

Name:_____



Part d. (5 points): In class we discussed latches made of NAND gates. Latches can also be made of NOR gates as shown below. X and Y are logical inputs which set/reset the latch; Q is the state of the latch.





(1) What are the quiescent values of X,Y (i.e., the values which will cause Q to remain unchanged)?

X: 0 Y: 0

(2) Suppose the value of X is changed from its quiescent value to its other logical value for a long enough period of time that this change can propagate through all the gates, and then the value of X returns to its quiescent value. What will be the value of Q after X returns to its quiescent value?

Q: 0

Part e. (10 points): Assume the contents of memory locations x3000 and x3001 are as shown below.

```
x3000: 0010 001 000000000
x3001: 0011 001 000000000
```

Using the LC-3 Simulator, you load PC with x3000, set a breakpoint at xF000, and push the RUN button. What happens? (in 20 words or fewer, please)

x3002 — xF000 ← x3200

Memory location x3002 - xF000 get the value x3200

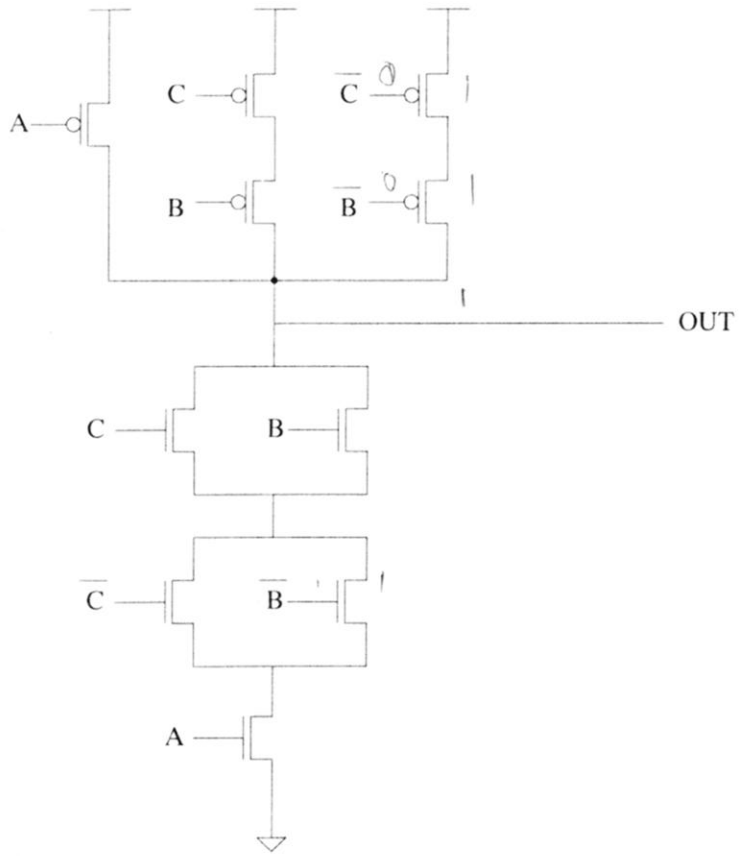
x3003 : x3200

x3004 : x3200

x3005 :

Name: _____

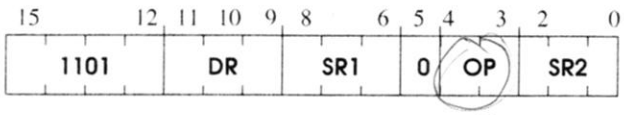
Problem 2. (10 points): The transistor circuit shown below has three input variables A, B, C. The output is labeled OUT. Complete the truth table.



A	B	C	OUT
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Name:_____

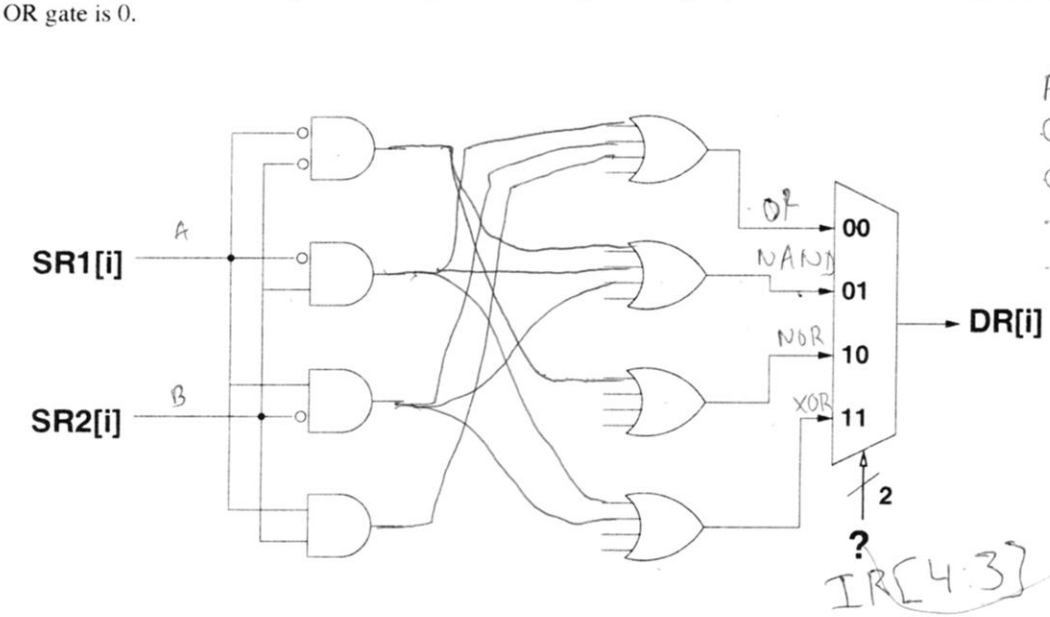
Problem 3. (10 points): A digital logic fanatic begged us to use the reserved opcode to implement four bit-wise logic functions OR, NAND, NOR, and XOR. He suggested the following instruction format:



Note that bits[4:3] specifies which of the four logical functions is to be performed as follows:

bits[4:3]	Operation
→ 00	OR
01	NAND
10	NOR
11	XOR

Part a. (8 points): Complete the logic circuit shown below to implement one bit (bit i) of the bit-wise operation by connecting the outputs of the AND gates to the inputs of the OR gates as appropriate. Assume that the unconnected inputs to each OR gate is 0.



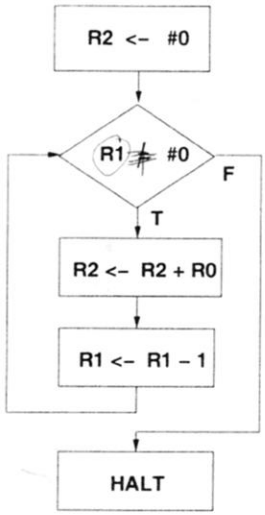
A	B	OR	NAND
0	0	0	1
0	1	1	1
1	0	1	1
1	1	1	0

Part b. (2 points): What two bits control the select lines of the MUX?

IR[4:3]

Name:_____

Problem 4. (10 points): Professor Patt's favorite example of a program that uses simpler instructions to perform a more complicated operation is the multiply routine shown below. It multiplies the positive numbers in R0 and R1, and puts the result in R2, even though the LC-3 has no multiply instruction. The flow chart below illustrates the algorithm.



The code was produced by an Aggie, and true to form, there are **two** errors in the program. Find the errors and, in each case, write the correct instruction in the corresponding slot to the right. Please leave BLANK the slots corresponding to the instructions that are already correct.

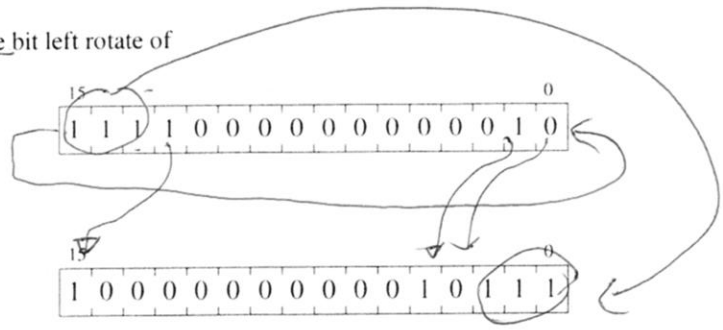
Address	Aggie's code (comment)	Corrected instruction (ONLY)
x3000	0001 0100 1010 0000 ; ADD	
x3001	0001 0010 0110 0000 ; ADD	
x3002	0000 1010 0000 0011 ; BR	
x3003	0001 0100 1000 0000 ; ADD	
x3004	0001 0010 0111 1111 ; ADD	
x3005	0000 1111 1111 1100 ; BR	
x3006	1111 0000 0010 0101 ; HALT	

Name:_____

Problem 5. (20 points): Once you graduate past the LC-3, you will encounter ISAs that have a LEFT ROTATE instruction. Since the LC-3 does not have one, we wish to write a program to perform the operation.

A one-bit LEFT ROTATE operation is nothing more than a LEFT SHIFT operation with the added feature of taking the bit shifted out of bit[15] and putting it back into bit[0], the place vacated by shifting bit[0] into position bit[1]. An n-bit LEFT ROTATE is the same as n successive one-bit left rotate operations.

Example: The result of a three bit left rotate of



produces

The program below will take the 16 bits contained in x3100 and left rotate it by the number of bit positions specified by the value in x3101, and put the result in x3102. Note that three of the instructions have been omitted. Your job: write in the missing three instructions.

```
        0011 0000 0000 0000          ; Starts at x3000

x3000   0010 000 011111111           ; Loads the value from x3100 into R0
3001    0010 001 011111111           ; Loads the value from x3101 into R1
3002    0000 010 000000111           ; If zero, branches to x300A
3003    0001 010 000 000 000         ; Adds R0 and R0 and stores result in R2
                                     R2 ← shifted R0
3004    ADD R0, R0, #0
3005    0000 011 000000001           -> ; If zero or positive, branches to x3007
3006    ADD R2, R2, #1
3007    0001 000 010 1 00000         ; Adds R2 and 0 and stores result in R0
                                     R2 + 0 -> R0
3008    ADD R1, R1, #-1
3009    0000 111 111111000           ; Branches to x3002
300A    0011 000 011110111           ; Stores value in R0 into x3102
300B    1111 0000 00100101           ; Halt
```

Name:_____

Problem 6. (20 points): Shown below are the contents of all the relevant (and some irrelevant) registers at the completion of three SUCCESSIVE clock cycles which are part of the instruction cycle for processing the instruction α.

Note: The start of the FETCH phase is before clock cycle N and memory is NOT accessed during cycles N, N+1, and N+2.

Clock Cycle	IR	MAR	MDR	R0	R1	R2	R3	R4	R5	R6	R7
N	α	x4010	α	xC89A	xA1B1	x1234	x2222	x2345	x8333	x5678	x00FF
N+1	α	x833A	α	xC89A	xA1B1	x1234	x2222	x2345	x8333	x5678	x00FF
N+2	α	x833A	xA1B1	xC89A	xA1B1	x1234	x2222	x2345	x8333	x5678	x00FF

R1 → MDR

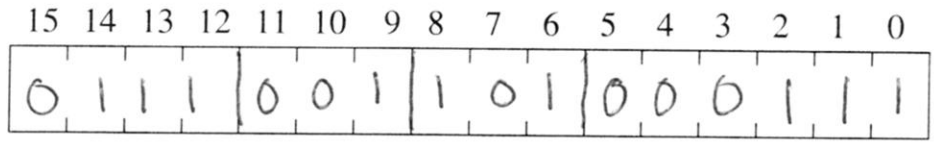
Part a. (4 points): Which states in the LC-3 state machine correspond to the processing in cycles N, N+1, and N+2? (Write down ONLY the state numbers).

N: 32 N+1: 7 N+2: 23

Part b. (4 points): What is the address of the instruction being processed? Please answer in hexadecimal notation.

Address: x4010

Part c. (12 points): Fill in the bits of the instruction being processed.



STR R1, R5, #7

3A
33
7