University of California at Berkeley
College of Engineering
Department of Electrical Engineering and Computer Sciences

EECS150 J. Wawrzynek
Spring 2005 2/22/05

Exam I

Name:___Solution Set_____

ID
number:____150_____

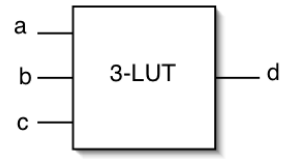
This is a *closed-book, closed-note* exam. No calculators please. You have until 2pm. Each question is marked with its number of points (one point per expected minute of time).

Put your name and SID on each page. You can tear off the spare pages at the end of the booklet and/or use the backs of the pages to work our your answers. Then neatly copy your answers in to the places allocated for them.

Show your work. **Write neatly** and be well organized.
Good luck!

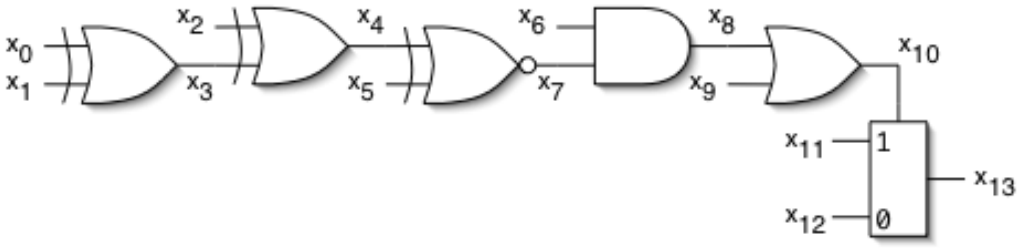
problem	maximum	score
1	10pts	
2	10pts	
3	10pts	
4	5pts	
5	5pts	
6	8pts	
7	8pts	
8	8pts	
total	64pts	

1. [10pts] Consider how to map the circuit shown below to a network of 3-input lookup tables (3-LUTs). A single 3-LUT is illustrated to the right.



"Allocate" LUTs by filling in the table below, one column per LUT. Use the top part of the table to indicate the input and output connections of each used LUT by filling in the table with node names from the circuit diagram. Use the bottom half of the table to fill in the LUT contents.

Connect unused inputs on allocated LUTs to "0". Fill unused locations in allocated LUTs with "0". Try to complete the mapping with as few LUTs as possible.

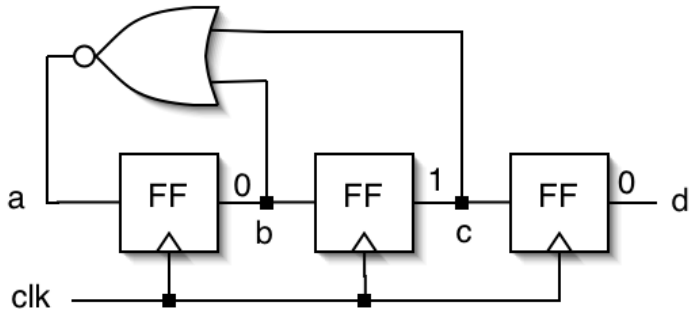


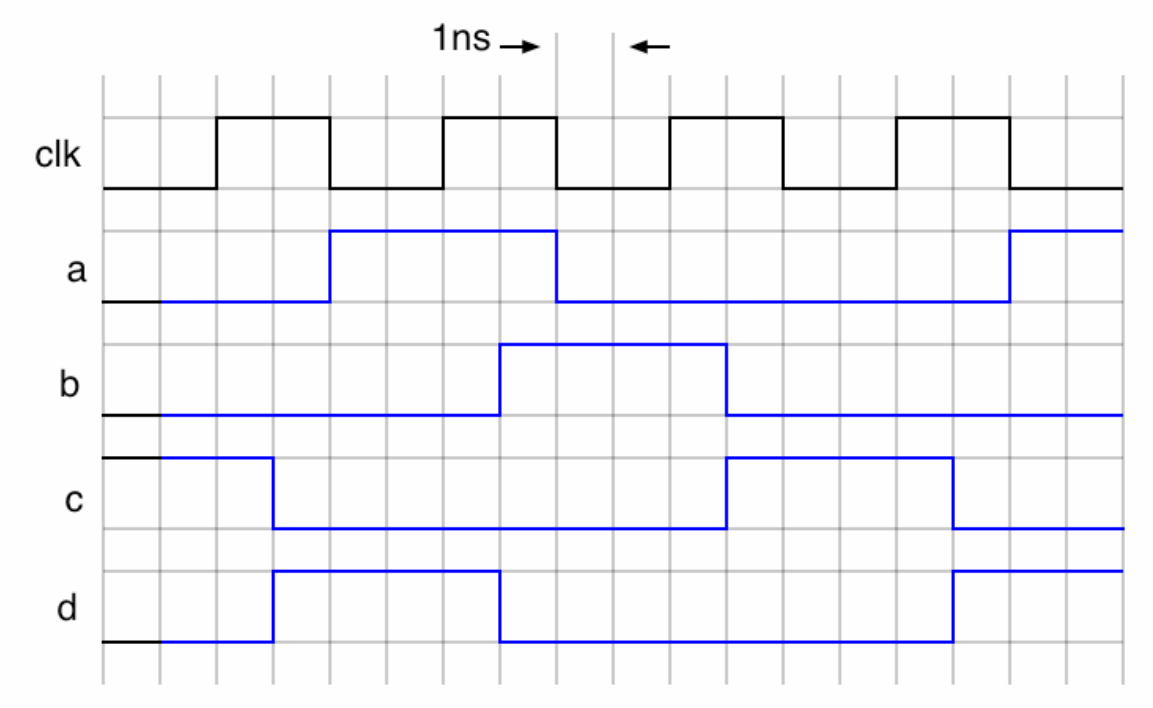
a	x_0	x_4	x_8	x_{10}				
b	x_1	x_5	x_9	x_{11}				
c	x_2	x_6	0	x_{12}				
d	x_4	x_8	x_{10}	x_{13}				
abc								
000	0	0	0	0				
001	1	1	0	1				
010	1	0	1	0				
011	0	0	0	1				
100	1	0	1	0				
101	0	0	0	0				
110	0	0	1	1				
111	1	1	0	1				

-2 For each extra LUT (over 4)
-1 For using 1'bx or NC in the port connections
-1 For using 1'bx or NC in the LUT contents
-1 For each LUT with the wrong function

Notice there are at least 72 different correct answers to this problem, almost one per student.

2. [10pts] Draw out the waveforms at point a, b, c, and d, for the operation of the circuit shown below. Assume that the flip-flop *clock-to-q delay* is 1ns, that the flip-flop *setup time* is 1ns, and the *gate delay* is 1ns. The initial values held by the flip-flops are shown a the output of each.



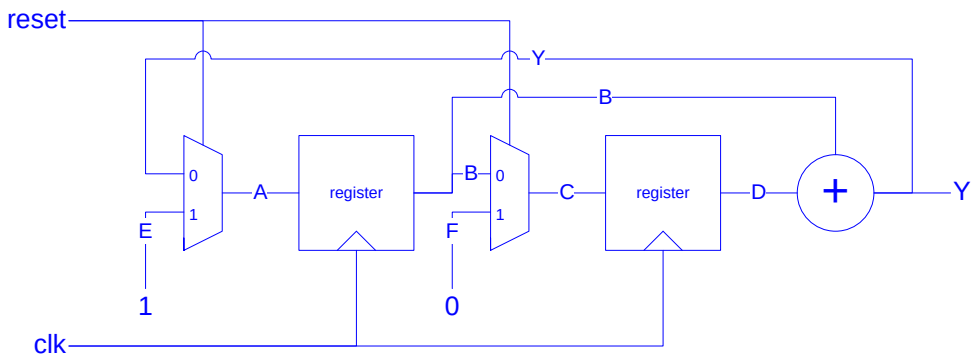


-1 Minor mistakes (1pt per wrong edge)
-3 Adding clock->q to setup
all edges shift right by 1 cycle
-5 Making 'a' a register or flipflop
'a' only changes after a rising edge in this case
-2 Mistaking NOR for NAND (correct timing for NAND)
-2 Per clock cycle with no waveforms drawn

3. [10pts] Listed below are the headers from a set of Verilog modules followed by the structural description of a module called fib:

```
module mux (IN0, IN1, select, OUT) ;  // standard 32-bit wide multiplexor
   input [31:0] IN0, IN1;
   input select;
   output [31:0] OUT;
         .
         .
         .
endmodule

module register (D, clk, Q);  // positive edge triggered register
   input [31:0] D;
   input clk;
   output [31:0] Q;
         .
         .
         .
endmodule


module adder (A, B, R);  // unsigned adder
   input [31:0] A, B;
   output [31:0]  R;
         .
         .
         .
endmodule


module fib (reset, clk, Y);  //structural description
   input reset, clk;
   output [31:0] Y;

   wire  [31:0] A, B, C, D, E, F;

   assign E = 1;
   assign F = 0;

   mux m1 (Y, E, reset, A);
   mux m2 (B, F, reset, C);
   register r1 (A, clk, B);
   register r2 (C, clk, D);
   adder add (B, D, Y);
endmodule
```

3.a) Based on the structural description of fib draw its circuit diagram. Do not show the internals of the submodules:



5points Total

3.b) In the space below write a behavioral description for just the fib module. You solution should have no instantiations of submodules.

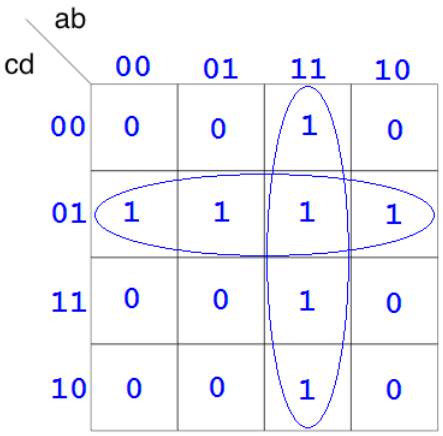
```
module fib (reset, clk, Y);
        input reset, clk;
        output [31:0] Y;

        reg [31:0] B, D;

        assign  Y = D + B;

        always @ (posedge clk) begin
                if (reset) B <= 1;
                else B <= Y;
        end

        always @ (posedge clk) begin
                if (reset) D <= 0;
                else D <= B;
        end
endmodule
```

5points Total

4. [5pts] In the space below, draw a K-map to represent the following function and use it to write the minimal sum-of-products form:

F = abc'd' + b'c'd + abcd' + abd + a'bc'd

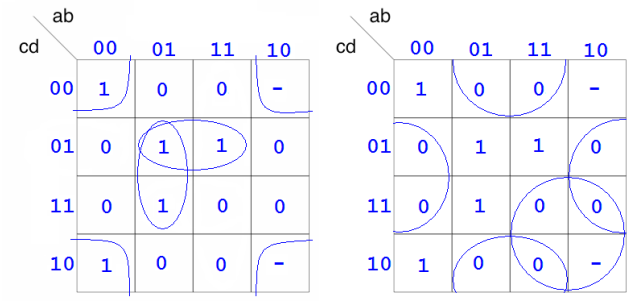
-1	Wrong 1's
-2	Non-max groups
-2	Wrong expression
-1	Inverted something
-2	SOP/POS mixup



F = ab + c'd

5. [5pts] In the space below draw a K-map to represent the following function and use it to write both the minimal sum-of-products form and the minimal product-of-sums form. The dash ("-") in the truth-table represent "don't cares" in the function.

abcd	F
0000	1
0001	0
0010	1
0011	0
0100	0
0101	1
0110	0
0111	1
1000	-
1001	0
1010	-
1011	0
1100	0
1101	1
1110	0
1111	0



sum-of-products F = b'd' + a'bd + bc'd

product-of-sums F = (b'+d)(b+d')(a'+c')

The key to this problem is recognizing and using the Don't Cares in the truth table.
Same grading as problem 4.

6.

a) [5pts] For the following function, using DeMorgan's theorem, derive a NAND only expression for the following function. Show your steps, and put your final result in the form of nested NANDs, i.e, F = NAND(..., NAND(...) ...).

$$F = (a + b)c + d$$

F = ac + bc + d
F = NAND((ac)', (bc)', d')
F = NAND(NAND(a,c), NAND(b,c), NAND(d,d))

-3 If the only knowledge shown is DeMorgan's Theorem
-1, -2 For small errors (only if DeMorgan's is correctly done)

b) [3pts] Using **Boolean algebra**, derive an expression for the following function in sum-of-products form. Show your steps.

$$F = a \oplus b \oplus c$$

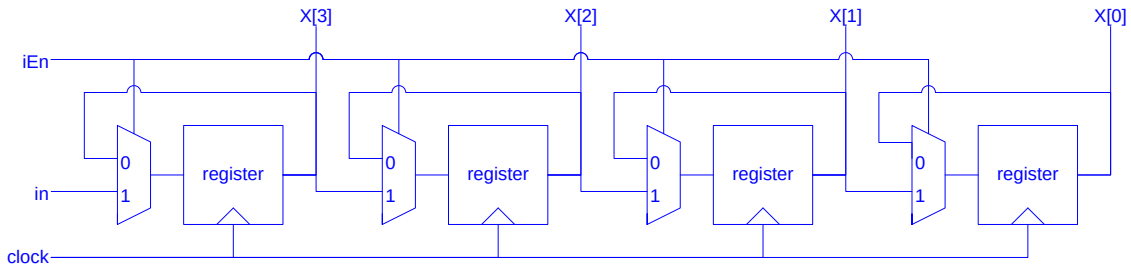
F = a(b ⊕ c)'+a'(b ⊕ c)
F = a(bc+b'c')+a'(bc'+b'c)
F = abc+ab'c'+a'bc'+a'b'c

-2 If the only component of the solution is x ⊕ y = x'y + xy'
-1, -2 For small errors (inversions etc)

7.

a) [5pts] In class we discussed a *parallel-to-serial* converter. Draw the circuit for a 4-bit *serial-to-parallel* converter using only 2-input multiplexors and flip-flops. The circuit has three inputs: the serial input, **in**; the **clock** input; and an input enable, **iEn**. The output is a 4-bit bus, **x[3:0]**.

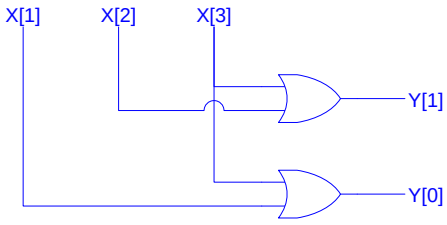
On the rising edge of the clock, if **iEn** is asserted, then the bit value at **in** is stored by the converter, otherwise the converter retains its previous value. Assume that the output is always available, and it is up to external circuitry to read it at the proper time. Also, assume that bits enter the converter, least significant bit first.



Notice there is a second solution to this problem were x[3] = in, and there are only 3 registers. We will accept this solution as well.

-1 LSB vs MSB errors
-3 Incorrect number of FlipFlops (3 or 4 are valid)
-2 Missing enables (either mux or CE on the FlipFlops)
-3 No shifting behavior

b) [3pts] Draw the circuit for a 4-to-2 encoder (one-hot code to binary translator). Inputs are x[3:0] and outputs are y[1:0]. *Assume that this circuit will always have exactly one 1 at its input.*



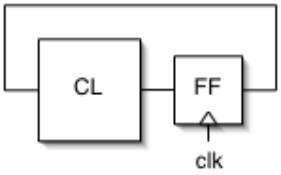
+3 For any functionally correct solution (need not be minimal)
-1 Minor errors
-1 For any use of 1' bx (which is not a valid value in hardware)
-1,-2 For missing or labeling errors

8. Short Answer.

a) [1pt] What is the frequency of a clock signal with a period of 4ns?

1/4ns = 250MHz

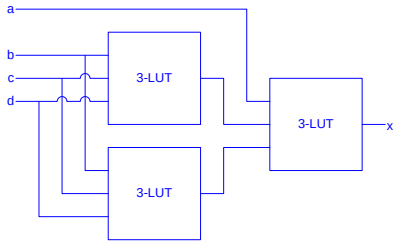
b) [1pt] In this circuit, the flip-flop setup time is 1ns, its hold time is 1ns, and its clock-to-q delay is 1ns. With a clock period of 4ns, what is the maximum delay through the combinational logic block that would permit proper circuit operation?



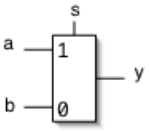
$t_{period} \geq t_{Setup} + t_{CLK->Q} + t_{CL}$

$t_{period} - (t_{Setup} + t_{CLK->Q}) = 4ns - (1ns + 1ns) = 2ns$

c) [2pts] Using nothing other than 3-LUTs, draw a circuit that implements a 4-LUT. Do not fill in the contents of the LUTs.



d) [1pt] Write a Boolean expression to represent the function of a two-input multiplexor:



y=sa + s'b

e) [2pt] Imagine you are designing and manufacturing a product and must choose between using an FPGA or ASIC (custom IC) for the internal electronics of the product.

Which would give the highest performance? (circle one) [FPGA, **ASIC**]

" " " " highest NRE cost? [FPGA, **ASIC**]

" " " " highest unit cost cost? [**FPGA**, ASIC]

" " " " longest time to market? [FPGA, **ASIC**]

-1 Point for each wrong answer to a minimum of 0 points.

[1pt] How many different Boolean functions can be implemented with a 3-LUT? (leave your answer as a decimal number)

2^3 = 8 Configuration Bits
2^8 = 256 Functions