Student Number _____

University of Toronto
Faculty of Applied Science and Engineering
Edward S. Rogers Sr. Department of Electrical and Computer Engineering

Final Examination
ECE 241F - Digital Systems

Examiners: S. Brown, J. Rose, K. Truong and B. Wang
December 9, 2004

Duration: 2.5 Hours

ANSWER ALL QUESTIONS ON THESE SHEETS, USING THE BACKS IF NECESSARY.

- Exam Type D, these specific aids allowed:
  i. Original Versions (no photocopies) of the course text, **Fundamentals of Digital Logic with Verilog Design,** by Brown & Vranesic, ISBN 0-07-282315-1.
  ii. One 8.5 x 11" two-sided aid sheet.

- The amount of marks available for each question is given in square brackets [].
- No calculators of any type are allowed. Cellular phones are also prohibited.

LAST NAME: _____

FIRST NAME: _____

STUDENT NUMBER: _____

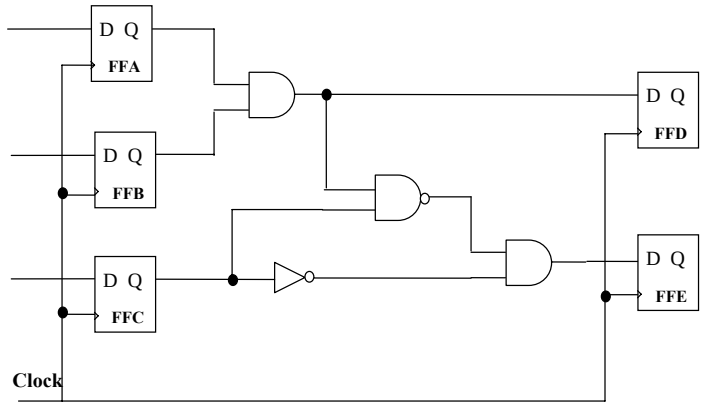
Lecture Section: Section 01 (Rose) [ ]
Section 02 (Wang) [ ]
Section 03 (Brown) [ ]
Section 04 (Truong) [ ]

Question	1	2	3	4	5	6	7	8	9	Total
Maximum Mark	8	10	6	9	6	6	8	8	10	71
Mark Obtained										

Student Number _____

[8] Q1. Consider the circuit below and the table giving various maximum propagation delays and other timing parameters for the circuit elements.

Timing Parameter	Maximum (ns)
T_{INV} – Delay of Inverter	2.0
T_{AND} – Delay of AND	5.0
T_{NAND}– Delay of NAND	4.0
Flip-Flop $T_{Clock\text{-}to\text{-}Q}$	3.0
Flip-Flop $T_{Set\text{-}up}$	2.0
Flip-Flop T_{Hold}	1.5



(a) [4] Determine the minimum clock period for which this circuit is guaranteed to operate correctly. Show how you arrived at your answer.

Student Number _____

Question 1, continued

(b) [4] Assume that the *minimum* propagation delay for all gates and the clock-to-Q of the flip-flops is 0.5ns. Will the hold time for the Flip-Flops labeled **FFD** and **FFE** be violated? Give an answer for *both* flip-flops and explain each answer.

Student Number _____

[10] Q2. Consider the one-input (**X**) one-output (**Out**) 3-state Moore-type Finite State Machine described in the State Transition and Assignment Table given below. Notice that this is a fully-encoded state machine for which the codes for States A and C have already been assigned, but the code for State B has not yet been determined. The code for state B is listed as $\mathbf{b_1\ b_0}$ in the table.

You are to choose the code for State B (without changing the given code for A and C) such that the total cost of the sum-of-products form of the resulting **Next State** logic and the **Output** logic is minimized. To measure cost, use the (# gates) + (# inputs) metric, in which you **should include** the cost of inverters.

You must show all of your work in the area provided, including how you decided which encoding resulted in the best cost. At the end, you must give the circuit schematic (diagram) of the state machine.

In the state transition and assignment table below, the *current state* (the flip-flop outputs) of the corresponding finite state machine are denoted Q_1Q_0, and the next state (flip-flop inputs) are denoted as D_1D_0.

Current State		**Next State**				**Out**
		X = 0		**X = 1**		
	Q_1Q_0	**D_1**	**D_0**	**D_1**	**D_0**	
A	00	0	0	b_1	b_0	1
B	b_1b_0	0	1	b_1	b_0	0
C	01	0	0	b_1	b_0	1

Below are some extra state tables that will help you determine your answer; be sure to fill them out as part of your answer.

Current State		**Next State**				**Out**
		X = 0		**X = 1**		
	Q_1Q_0	**D_1**	**D_0**	**D_1**	**D_0**	
A	00	0	0			1
B		0	1			0
C	01	0	0			1

Q2, continued.

Current State		Next State				Out
		X = 0		X = 1		
	Q_1Q_0	D_1	D_0	D_1	D_0	
A	00	0	0			1
B		0	1			0
C	01	0	0			1

Work area:

(a) [7] Answer: The Best Code for state B is:_____
Be sure to show all of your work above, including how you determined the cost of all of the alternative codes for B.

Q2, continued.

(b) [3] Give the full state machine design, (using D-type flip-flops, AND, OR and NOT gates,) of the state machine that uses the encoding that results in the minimum-cost logic.

Student Number _____

[6] Q3. Consider the following logic function:

$$\mathbf{f(x_3,x_2,x_1,x_0) = \Sigma\, m\,(0, 4, 7, 8, 9, 12, 13, 15)}$$

(a) [3] Determine the minimal sum-of-products form for the logic function. Make use of the following Karnaugh map:

x_1x_0 \ x_3x_2	00	01	11	10
00				
01				
11				
10				

Answer:

Student Number _____

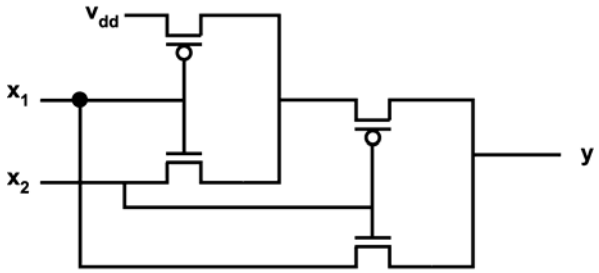
Q3, continued

(b) [3] Implement the above function using only one 4:1 multiplexor and as many basic gates (NOT, AND, OR) as necessary. Note: full marks will be given to a solution that uses no basic gates.

Student Number _____

[9] Q4.

(a) [3] Consider the following transistor-level design of a 2-input logic gate. Derive the truth table of the output **y** in terms of the inputs x_1 and x_2. Place your answer in the truth table beside the circuit



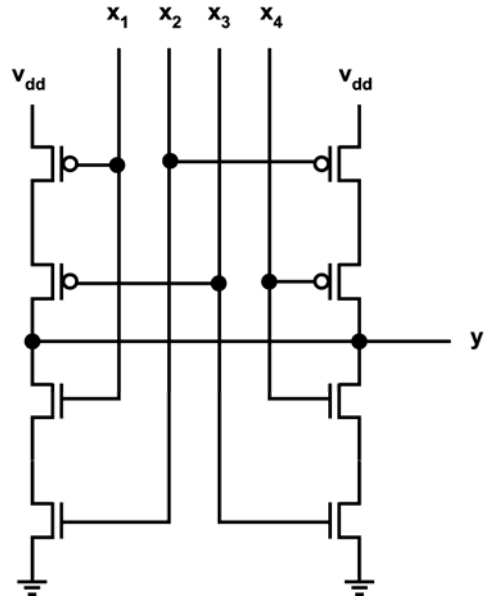
x_1	x_2	y
0	0	
0	1	
1	0	
1	1	

Put your rough work here:

Student Number _____

Q4, continued

(b) [6] Consider the transistor level design of the 4-input gate below. Determine the truth table of the output **y** in terms of the inputs x_1, x_2, x_3 and x_4. **For the cases where y is not driven to 0 or 1, give the answer 'Z'.** Place your answer in the truth table give at the right.



x_1	x_2	x_3	x_4	y
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

Student Number _____

[6] Q5. Consider the two-input, two-output State Machine described by the state transition table below.

Present State	Next State				Outputs	
	x_1x_2=00	x_1x_2=01	x_1x_2=10	x_1x_2=11	z_1	z_2
S_0	S_0	S_6	S_2	S_4	1	0
S_1	S_0	S_3	S_1	S_5	0	1
S_2	S_1	S_6	S_2	S_4	1	1
S_3	S_1	S_0	S_4	S_6	0	1
S_4	S_0	S_3	S_2	S_0	1	0
S_5	S_1	S_4	S_0	S_1	0	1
S_6	S_1	S_0	S_4	S_3	0	1

(a) [4] Determine the minimum number of states that could be used to implement this state machine, showing your work

Student Number _____

Q5, continued

(b) [2] Give the state transition table for the minimized state machine:

Present State	Next State				Outputs	
	x_1x_2=00	x_1x_2=01	x_1x_2=10	x_1x_2=11	z_1	z_2

Student Number _____

[6] Q6. Consider the following two logic functions:

$$f = x_1 \cdot x_2 \cdot x_3 \cdot x_4 \cdot x_5 \cdot x_6$$
$$g = \overline{x}_1 \cdot \overline{x}_2 \cdot \overline{x}_3 \cdot \overline{x}_4 \cdot \overline{x}_5 \cdot \overline{x}_6$$

You are to implement these two functions using a total of three 4-input lookup tables. Show what logic function has to be implemented in each of the three lookup tables by giving the logic expression as a function of its labeled inputs (in the case where there isn't a label, you should create one).

Use the simplest logic expressions possible for each lookup table, using AND, OR, and NOT operators. Show your logic expression inside each lookup table, and show how the lookup tables are connected together.

Note that some part marks will be given if you use four or more lookup tables, but to get full marks you have to use exactly three lookup tables.

ANSWER:

Student Number _____

[8] Q7. Consider the Verilog code for a finite state machine shown below:

```
module whatsthis (Clock, Resetn, In3, Out);
    input Clock, Resetn, In3;
    output [1:3] Out;
    reg [1:3] y, Y;
    parameter [1:3] A = 3'b000, B = 3'b001, C = 3'b010,
        D = 3'b011, E = 3'b100, F = 3'b101, G = 3'b110, H = 3'b111;

    // Define the next state combinational circuit
    always @(In3 or y)
        case (y)
                A:    if (In3) Y = B;
                      else     Y = A;
                B:    if (In3) Y = D;
                      else     Y = C;
                C:    if (In3) Y = F;
                      else     Y = E;
                D:    if (In3) Y = H;
                      else     Y = G;
                E:    if (In3) Y = B;
                      else     Y = A;
                F:    if (In3) Y = D;
                      else     Y = C;
                G:    if (In3) Y = F;
                      else     Y = E;
                H:    if (In3) Y = H;
                      else     Y = G;
        endcase

    // Define the sequential block
    always @(negedge Resetn or posedge Clock)
        if (Resetn == 0) y <= A;
        else y <= Y;

    // Define output
    assign Out = y;

endmodule
```

(a) [4] At the top of the next page give the state-assigned table (the table that shows the state codes and not the letters A, B, etc for the state names) for this FSM.

Student Number _____

Q7, continued
ANSWER - State-Assigned Table:

(b) [4] State in words, as simply as possible, what function is performed by this FSM.

ANSWER:

_____

_____

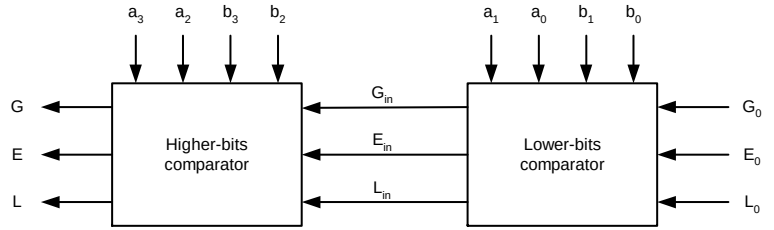
_____

_____

Student Number _____

[8] Q8. A digital comparator is a logic circuit that takes two **n**-bit numbers **A** and **B** as input and compares the magnitude of A and B. It has three outputs, **G, E, and L**, where G=1 only if A>B, E=1 only if A=B, and L =1 only if A<B.

A 4-bit magnitude comparator (for unsigned numbers) is to be constructed from two 2-bit comparators as shown in the diagram below. One of these will compare the high-order 2-bits (a_3a_2 and b_3b_2) of each input and the other will compare the lower 2-bits (a_1a_0 and b_1b_0). The outputs, greater (G_{in}), equal (E_{in}), and less (L_{in}) from the lower-bits comparator become additional inputs to the higher-bits comparator. The inputs to the lower-bits comparator are labeled G_0, E_0, and L_0.



(a) [2] For what values of the inputs A and B will the result of the lower-bits comparator (G_{in} , E_{in} and L_{in}) influence the overall outputs *G*, *E* and *L*? Explain. Your answer *should not* give every possible input combination – you must express your answer by stating relationships between the input values of the a_i, b_i, G_{in} , E_{in} and L_{in}.

Student Number _____

Q8, continued

(b) [2] If both 2-bit comparator units are to contain identical logic circuits, specify, for the lower-bits comparator, what values the inputs G_0, E_0, and L_0 should have in order for the complete 4-bit comparator to operate properly.

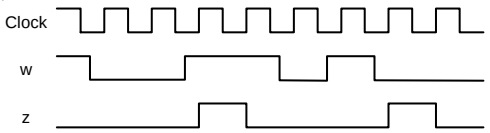
(c) [4] Give a logic expression for *G* (recall that G = 1 if A > B) in terms of a_3, a_2, b_3, b_2 and G_{in}. You are allowed to use the AND, OR, NOT and XOR logic operators as needed.

Student Number _____

[10] Q9. A finite-state machine has one input (**w**) and one output (**z**). The input **w** is a serial input synchronized to a clock in a similar manner to the sequence detectors you built in lab #6.

The state machine is part of a communication system that is using the following rules for transmission of data through the input w: If a *0* occurs in the input stream, then there should be an odd numbers of 0's; if a *1* occurs, then there should be an even numbers of 1's.

The purpose of the state machine is to detect errors in the sequence that doesn't obey these rules. The output, z, should be set to 1 for one clock cycle whenever an even number of *zeroes* occur in the input sequence or an odd number of *ones*. The following pattern shows the corresponding values of w and z.



Give a Moore-type state diagram for this FSM. Use as few states as possible: