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: J. Rose and B. Wang
December 10, 2003

Duration: 2.5 Hours

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

1. No aids of any kind are allowed, including calculators.

2. The amount of marks available for each question is given in square brackets [].

3. Answer all questions.

LAST NAME: _____

FIRST NAME: _____

STUDENT NUMBER: _____

CHECK LECTURE SECTION: 01 Rose [ ]
02 Wang [ ]

Question	1	2	3	4	5	6	7	Total
Maximum Mark	15	15	10	10	10	15	10	85
Mark Obtained								

[15] Q1.

[5] (a) A master-slave D-type flip-flop and a single bit of static random access memory (SRAM) are capable of storing 1 bit of information. You are to determine the number of transistors in each, assuming they are built as shown in class. For the flip-flop, assume that there are no set or set reset signals. For the SRAM bit, only count the transistors for a single bit in the array, **not** the circuitry on the periphery of the array.
Show how you arrive at your answer by giving the transistor counts of the appropriate sub-units and adding them all up.

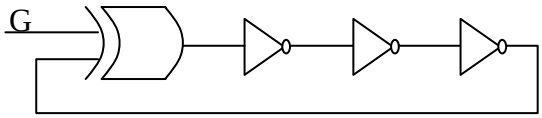
Answer:

Number of Transistors in CMOS Master-Slave D-type Flip-Flop:

Number of Transistors in CMOS SRAM Bit:

Q1, cont'd

[5] (b) Consider the following circuit, which has a exclusive OR gate feeding a chain of three inverters and an input signal G:

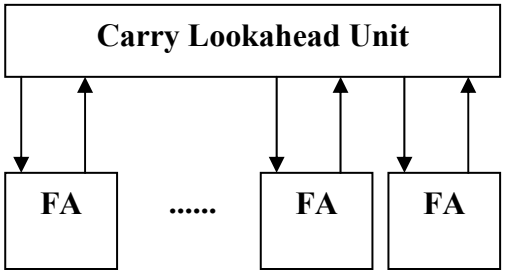


(i) Under what condition does this circuit oscillate?

(ii) If the propagation delay of all the inverters is 2ns, and the propagation delay of the exclusive OR gate is 3ns, what is the *period* of oscillation of the circuit when it oscillates?

Q1, cont'd

[5] (c) Consider a basic carry-lookahead adder, roughly illustrated below:



Assuming that the logic is built using CMOS logic family, what prevents an adder designer from usefully employing a single level of carry lookahead, (as illustrated above) across a large number (more than 4, say) of full adder units? Your answer should consist of just a few sentences.

[15] Q2.

[5] (a) A children's riddle concerns a Farmer who is traveling with a sack of Rye, a Goose and a mischievous Dog. The farmer comes to a river that he must cross from east to west. A boat is available, but it only has room for the farmer and one of his possessions. **If** the farmer is not present on the same side of river,

1) As the goose and the rye, the goose will eat the rye, which is bad.
2) As the dog and the goose, the dog will eat the goose which is also bad.

You are to create a truth table for a logic circuit that will indicate when one of the bad events will happen. A separate switch is provided for the farmer (F), the rye (R), the goose (G) and the dog (D). Each switch will indicate a logic "high" if the corresponding object is on the *east* bank and a logic "low" if that object is on the *west* bank. The output of the circuit, an alarm signal (A) should be set "high" any time the rye or the goose is in danger of being eaten.

Give the truth table for the function A:

F	R	G	D	A
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	

Q2, cont'd

[5] (b) Determine the minimal Product-of-Sums (POS) expression for the following logic function:

$$\mathbf{F\ (P,Q,R,S) = \Sigma\ m(3,6,7,9,12,13,14,15)}$$

Make use of the following Karnaugh Map:

PQ / RS	00	01	11	10
00				
01				
11				
10				

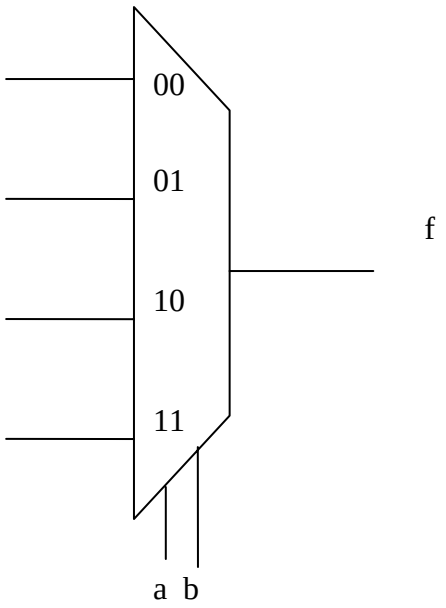
Answer, in **Product of Sum** form:

Q2, cont'd

[5] (c) Give the circuit implementation of the following truth table using **ONE** 4-to-1 multiplexer and as few other logic gates as possible. You must use **a** and **b** as the multiplexor selector inputs, as shown on the 4-to-1 multiplexor below.

a	b	c	d	f
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

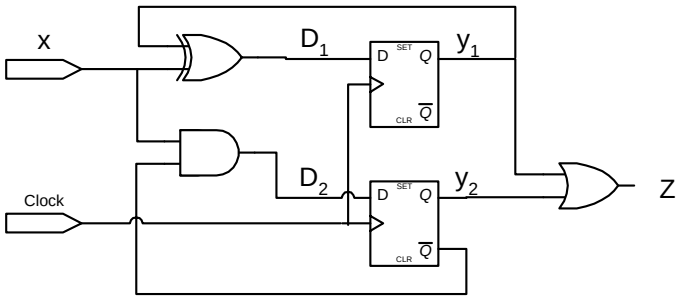
Use this multiplexor circuit as part of your answer:



[10] Q3. Implement the logic function $f(a,b,c,d) = \sum m(1,2,6,8,9,10,11,14)$ using no more than **three** 3-LUTs (3-input look up tables). Indicate inside each LUT the Boolean expression (**not** the truth table) for logic function being implemented in the LUT, in its **minimal sum-of-products form**.

[10] Q4. You are to analyze the finite state machine circuit below, as specified in parts (a), (b) and (c) below. Assume that the D flip-flops are initially reset to zero before the machine begins.

The circuit



[3] (a) Give the logic expressions for next-state variables (D_1 and D_2) and the output (Z);

[4] (b) Give the state transition table. You **must represent** each state by the specific code used by the circuit. Fill in the following table appropriately:

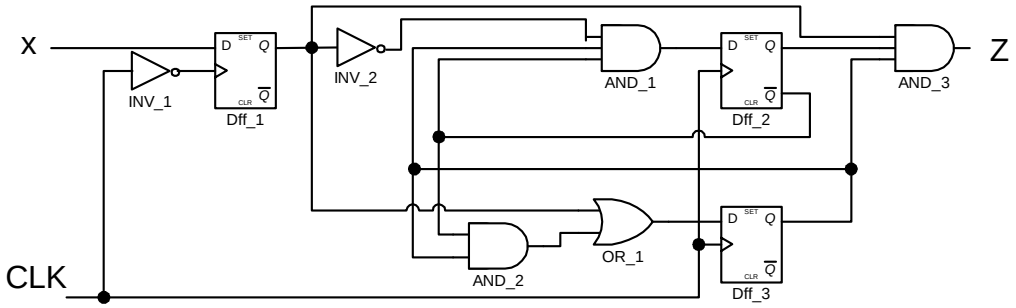
Present State	**Next State**		**Output**

Q4, cont'd

[3] (c) The state diagram, again using the code to represent each state.

[10] Q5. Consider the circuit below and the table giving various maximum and minimum propagation delays for the circuit elements.

Propagation Delay	Maximum (ns)	Minimum (ns)
T_{INV}	2.0	0.5
T_{AND}	5.0	1.5
T_{OR}	5.0	1.5
$T_{Clock\text{-}to\text{-}Q}$	3.0	1.0
$T_{set\text{-}up}$	2.0	
T_{hold}	1.0	



[7] (a) Determine the minimum clock period for which this circuit is guaranteed to operate correctly. Be sure to show how you make the calculation

Q5, cont'd

[3] (b) Will the circuit have a hold-time violation? Explain why or why through a hold-time analysis.

[15] Q6. You are to design a one-input, one-output serial 2's complementer state machine. The input is a series of bits (one per clock cycle) beginning with the least-significant bit of the number to be negated and the output is the 2's complement of the input. The circuit is reset asynchronously to start and end each operation.

[6] (a) Give the state diagram and the state transition table. Use a fully-encoded state coding.

Q6, cont'd

[4] (b) Derive the logic expressions for all next-state variables and the output.

Q6, cont'd

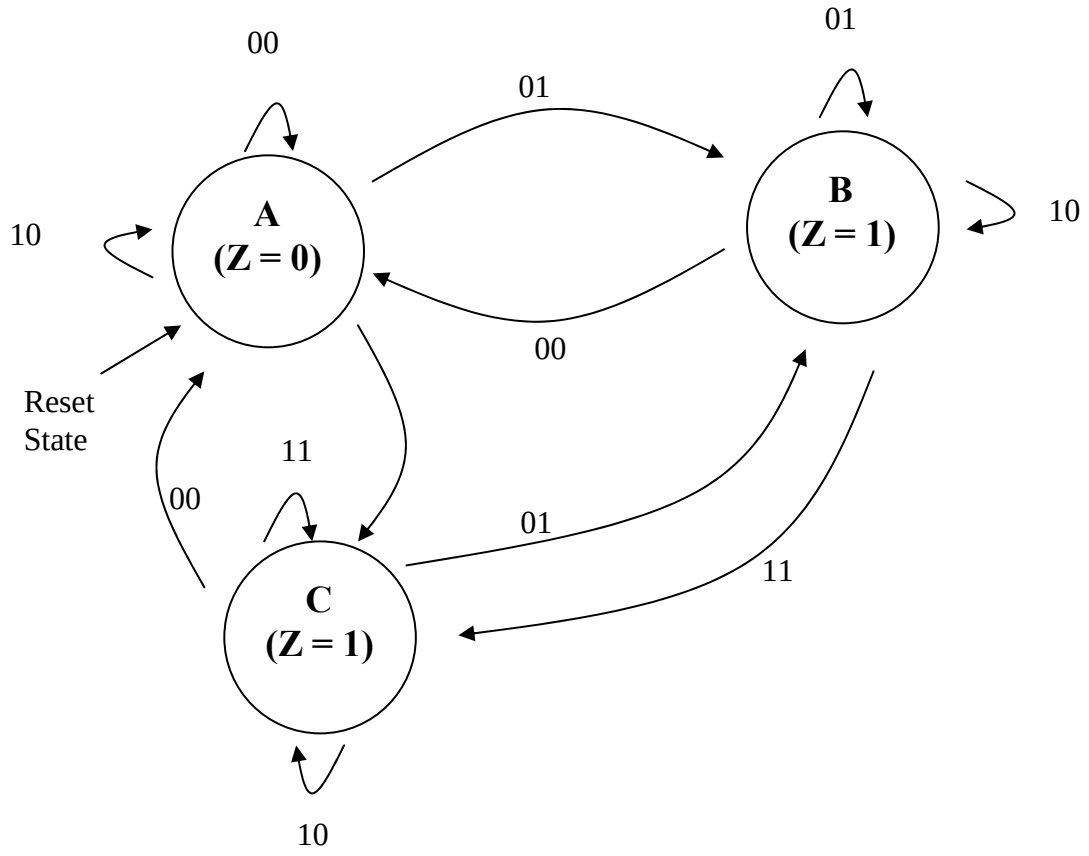
[5] (c) Give the logic circuit for the machine - clearly label all signals.

[10] Q7.

[7] (a) Give the Verilog code that implements the state machine described by the following state diagram, using the same signal and state names and state encoding as given. In your code, you must separate the code that describes the next state computation from the code that describes the state registers.

Code for State A: 00
Code for State B: 01
Code for State C: 11

Inputs X Y
Output: Z



Give Verilog code for part (a) here:

Q7, cont'd

[3] (b) What is the likely cause of the following problem: this above circuit simulates correctly using functional simulation, but incorrectly using timing simulation?