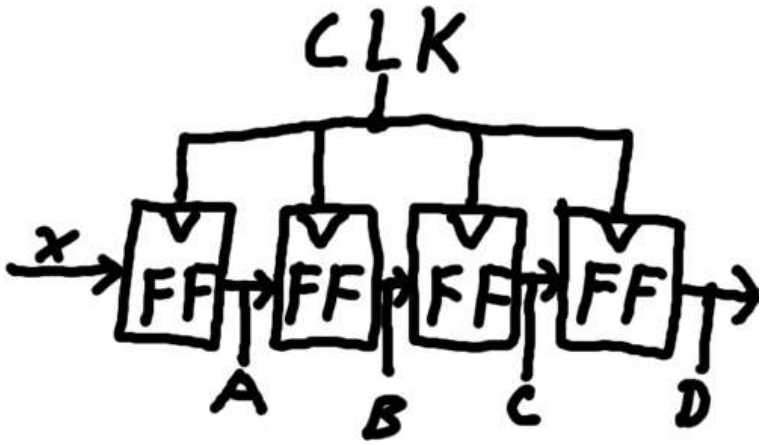
University of California at Berkeley
College of Engineering
Department of Electrical Engineering and Computer Science

EECS 61c – Fall 2004
HW 05 – Due Before Lecture Friday, 10/29
TA In Charge: Andy

Due before lecture on Friday, October 29. Turn in a paper copy at the front of the lecture hall before lecture or to a TA or Prof. Garcia anytime beforehand. Any submissions received after the start of lecture will be penalized one slip day. Make sure to clearly mark your submission with your name, login, and Lab TA's name.

1. What is Moore's law? What are two possible implications of Moore's law on the design of digital systems? (ie. What are two things that the effects of Moore's law allow that would not otherwise be possible?)

2. Consider the circuit of Flip-Flops (FF) below. Assume that input X alternates between 1 and 0, changing at the start of each clock period. Draw the detailed wave for the clock signal, input X, and the signals at points A, B, C, and D in the circuit for 5 clock cycles. Assume that the clock period is 6ns and the clk-to-q delay is 2ns.

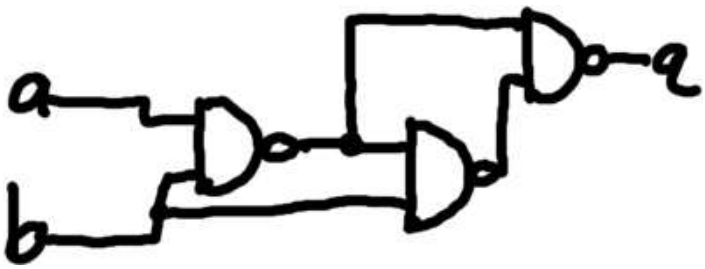


3. Consider the accumulator discussed in the readings and presented in class. Given the following: The adder propagation delay is 7ns, the register setup time is 1 ns, the register clk-to-q is 1ns, and the clock frequency is 100MHz. Will the accumulator function correctly? If not, what would you suggest changing to fix the problem?

4. Design a finite state machine (FSM) with the following behavior: Inputs arrive one bit at a time, one bit per clock cycle. The FSM outputs a 1 if the pattern '01' has been recognized and continues to output a 1 as long as bits matching that pattern continue to be input. The FSM should output 0 at all other times. (ie. '010101...' will continue to output 1 after it starts to do so, but '01000...' will not. '010' will output a 1, "trusting" that the next input will be a '1'.) Your FSM should include a start state, but you need not worry about initialization other than that.

   Do your design in three steps. First, draw the state diagram, second specify the truth table for next state and output based on present state and input, finally devise the circuit-level implementation.

5. Derive the truth-table for the CL circuit shown below. (Remember, you do this by applying all possible input combinations, one at a time). What is the most simplified version of the expression that this circuit/truth-table generates?



6. Write the canonical sum-of-products form of a Boolean expressions for a 5 input function whose output is 1 if and only if the number of 1's in the input is exactly 3.

7. Write the most simplified Boolean expression for the function represented by the truth-table below. *The solution is the OR of three AND terms, each with 2 variables.*

abc	***y***
000	0
001	0
010	0
011	1
100	0
101	1
110	1
111	1

8. Draw the complete circuit diagram for a 1-bit wide, 2-to-1 mux using only NAND gates. You must show all the gates used (ie. You can not just draw a mux).