EE434

ASIC and Digital Systems

Final Exam

May 4, 2017. (8am – 10am)

Instructor: Dae Hyun Kim (daehyun@eecs.wsu.edu)

Name:

WSU ID:

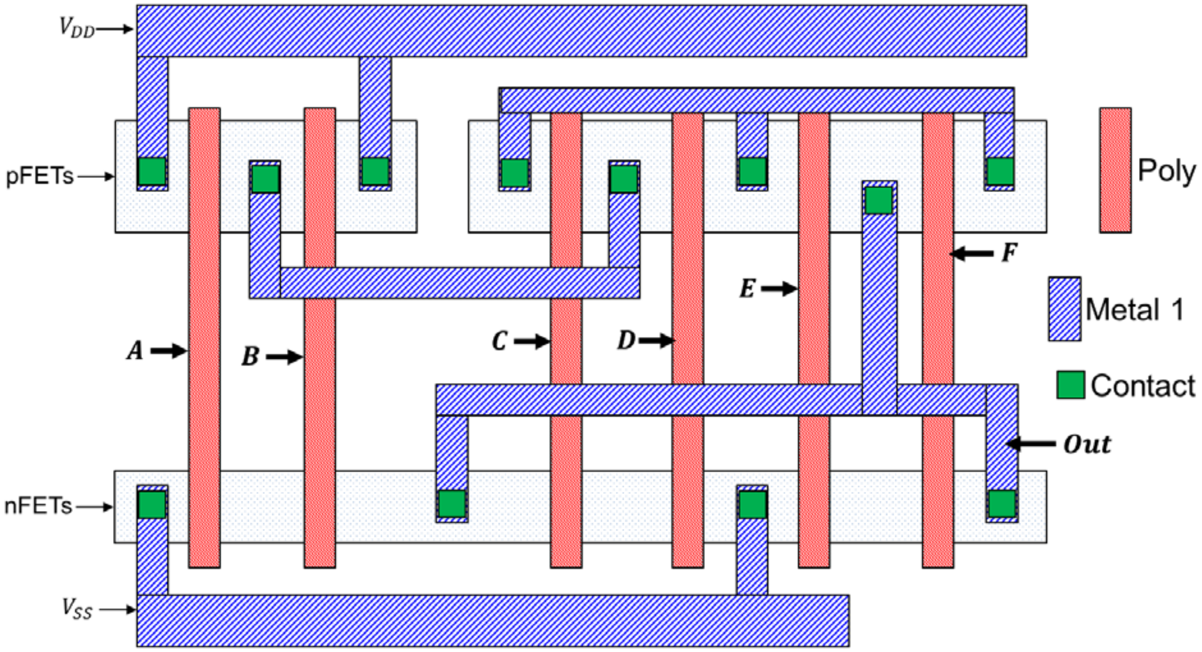
Problem	Points	
1	10	
2	10	
3	10	
4	10	
5	30	
6	10	
7	10	
8	10	
Total	100	

* Allowed: Textbooks, cheat sheets, class notes, notebooks, calculators, watches

* Not allowed: Electronic devices (smart phones, tablet PCs, laptops, etc.) except calculators and watches

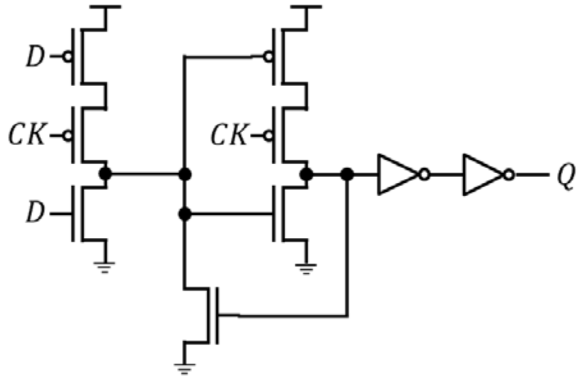
Problem #1 (Layout Analysis, 10 points)

The following combinational logic has six primary inputs (A, B, C, D, E, F) and a primary output (Out). Find all input vectors that can detect a stuck-at-1 fault at input E.



Problem #2 (Static CMOS gates, 10 points)

Describe the function of the following circuit in as much detail as possible (D: data input, CK: clock).



(Partovi, ISSCC'96)

Problem #3 (Timing Analysis, 10 points)

Answer the following questions.

- WNS can be less than TNS (i.e., “WNS<TNS” can happen). (True/False)

- TNS can be less than WNS (i.e., “TNS<WNS” can happen). (True/False)

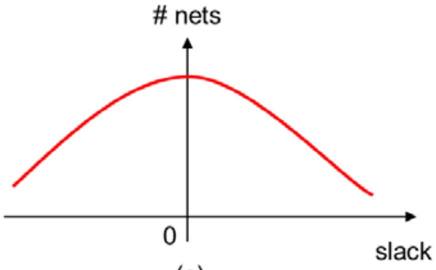
- WNS can be equal to TNS (i.e., “WNS=TNS” can happen). (True/False)

- A design has only two violating paths. In this case, the following can happen. (True/False)
    - “WNS of the design is -2ns and TNS of the design is -4.5ns.

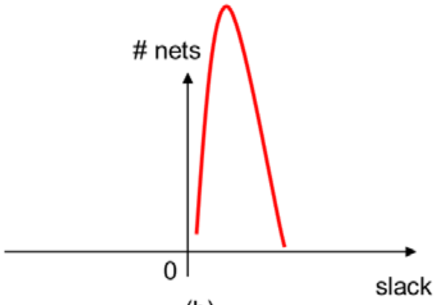
- A design has only four violating paths. In this case, the following can happen. (True/False)
    - “WNS of the design is -2ns and TNS of the design is -8.2ns.

Problem #4 (Timing Analysis, 10 points)

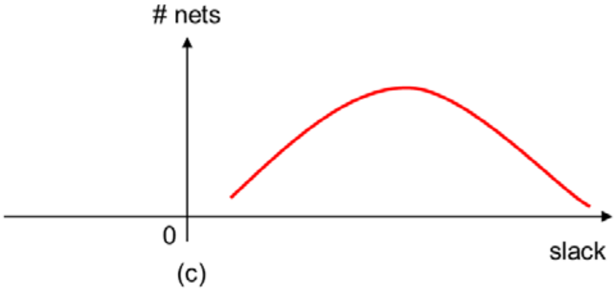
You are given six designs, (a), (b), …, (f). Their timing analysis results are shown below. It is also known that the power consumption and the total layout area of a design are proportional to the total positive slack. You are supposed to choose a design and send it to a foundry for fabrication without any further optimization. Choose one among the six designs and explain why you decided to choose the design for fabrication.



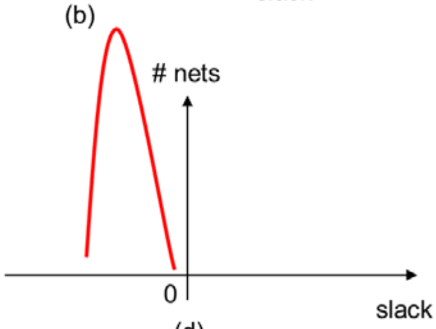
(a)



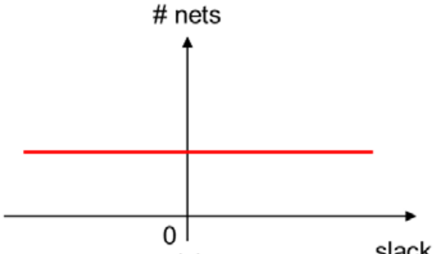
(b)



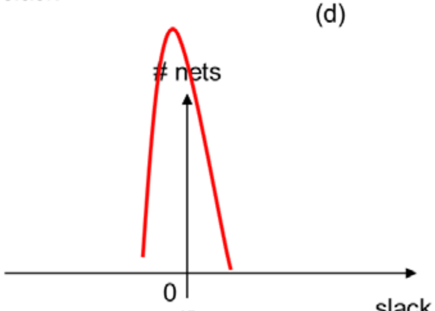
(c)



(d)



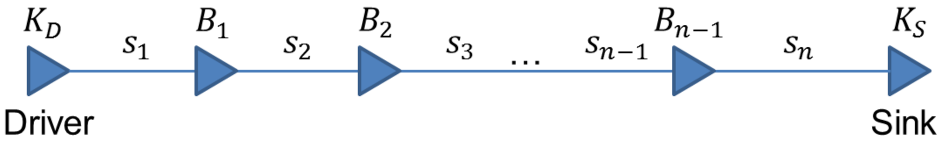
(e)



(f)

Problem #5 (Interconnect Optimization, 30 points)

The following figure shows a net optimized by buffer insertion. The driver and the sink are denoted by K_D and K_S, respectively, and the inserted buffers are denoted by B_i $(1 \leq i \leq n-1)$. $n \geq 2$, i.e., there is at least one buffer between the driver and the sink.



- Output resistance of K_D: R_D
- Output resistance of B_i (1≤ $i \leq n-1$): R_i (e.g., $R_1, R_2, \ldots$)
- Input capacitance of K_S: C_S
- Input capacitance of B_i (1≤ $i \leq n-1$): C_i
- Delay of B_i $(1 \leq i \leq n-1)$: D_i
- Length of the i-th net $(1 \leq i \leq n)$: s_i (um)
- $\sum_{i=1}^{n} s_i = L$ (um)
- Wire unit resistance: r (Ω/um)
- Wire unit capacitance: c (fF/um)

<u>We assume that the net is optimized to minimize the delay from the driver to the sink.</u>

(Hint: Derive s_1 and s_2 as functions of the above parameters when $n = 2$. You can somehow use the result for the following questions).

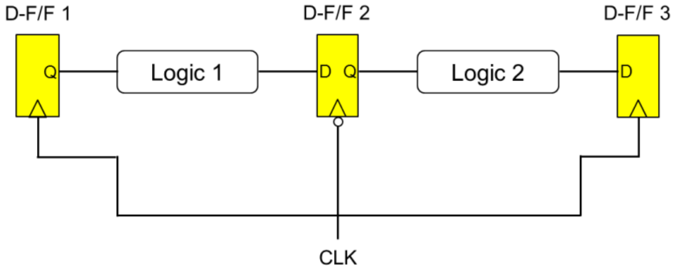
Answer the following questions <u>for $n = 10$</u> (i.e., we insert 9 buffers optimally):

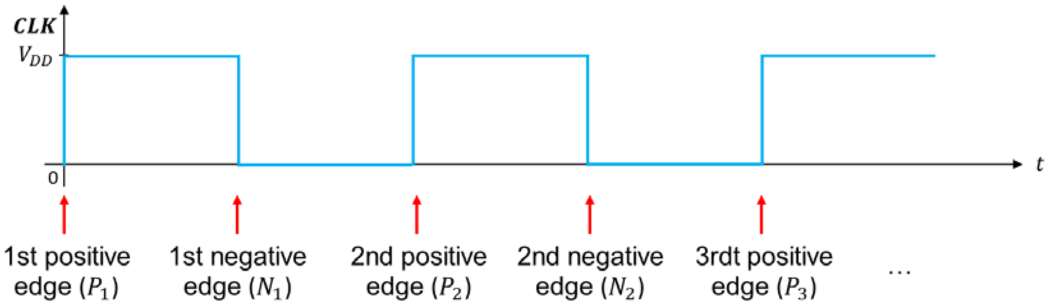
- If C_S increases, we should increase s_1 to minimize the total delay. (True/False)
- If R_9 increases, we should increase s_1 to minimize the total delay. (True/False)
- If D_9 increases, we should increase s_1 to minimize the total delay. (True/False)
- If C_5 increases, we should increase s_8 to minimize the total delay. (True/False)
- If R_5 increases, we should increase s_8 to minimize the total delay. (True/False)
- If D_5 increases, we should increase s_8 to minimize the total delay. (True/False)
- If R_D increases, we should increase s_1 to minimize the total delay. (True/False)
- Suppose $s_1 \approx 0$ because $R_D \gg R_1, \ldots, R_9$. In this case, if r and c increase at the same time, we should increase s_1 to minimize the total delay. (True/False)

Answer the following questions assuming n is to be determined optimally (i.e., we find # buffers $(n-1)$ and $s_1 \sim s_n$ optimally) and $L \gg 0$, so $n \gg 1$:

- If R_D increases, we should increase n to minimize the total delay. (True/False)
- If C_2 increases, we should increase n to minimize the total delay. (True/False)
- If C_S increases, we should increase n to minimize the total delay. (True/False)
- If the delay of each buffer is increased, we should generally increase n to minimize the total delay. (True/False)
- If L increases, we should increase n to minimize the total delay. (True/False)
- If r increases, we should increase n to minimize the total delay. (True/False)
- If c increases, we should increase n to minimize the total delay. (True/False)

Problem #6 (Timing Analysis, 10 points)




CLK
V_{DD}
0
t
1st positive edge (P_1)
1st negative edge (N_1)
2nd positive edge (P_2)
2nd negative edge (N_2)
3rdt positive edge (P_3)
…

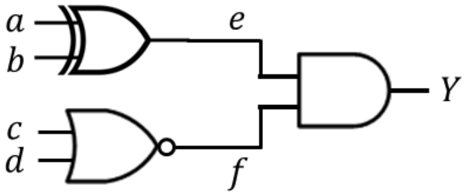
- Setup time of a D-FF: T_s
- Hold time of a D-FF: T_h
- D-F/F internal delay: T_{CQ}
- Logic 1 delay: T_{L1}
- Logic 2 delay: T_{L2}
- Clock period: T_{CK} (duty cycle: 50%, i.e., the clock is high for $T_{CK}/2$ and low for $T_{CK}/2$.)
- Delay from CLK to D-FF 1: D_1
- Delay from CLK to D-FF 2: D_2
- Delay from CLK to D-FF 3: D_3
- D-F/F 2 is a negative-edge FF (i.e., it captures the input signal at falling edges.)

The above figure shows three FFs connects in series. D-FF 2 is a negative-edge FF, whereas D-FF 1 and 3 are positive-edge FFs. The operation of the circuit is as follows. D-FF 1 captures its input signal at k-th positive clock edge P_k. Logic 1 performs computation for the output of D-FF 1. D-FF 2 captures its input signal at k-th negative clock edge N_k. Logic 2 performs computation for the output of D-FF2. D-FF 3 captures its input signal at $(k+1)$-th positive clock edge P_{k+1}.

Derive two setup time inequalities (one for Logic 1 and the other for Logic 2).

Problem #7 (Testing, 10 points)

We want to detect stuck-at-0 and stuck-at-1 faults at all the primary inputs, a, b, c, d, and the two internal nodes, e, f. Computation of Y to detect a stuck-at-0/1 fault at an internal node can be done by setting the value of the node to constant 0 (for stuck-at-0 faults) or 1 (for stuck-at-1 faults). Find a ***minimal*** set of test vectors that can detect all the s-a-0 and s-a-1 faults at $a, b, c, d, e,$ and f for the following logic (Hint: all the minimal sets have five test vectors).



Problem #8 (Testing, 10 points)

A combinational logic G is given. It has n inputs and one output. The inputs are $x_1, x_2, \dots, x_n$ $(n \geq 2)$ and the output is y, i.e., $y = G(x_1, \dots, x_n)$. To find an input vector that can detect a stuck-at-v fault ($v = 0$ or 1) at x_i $(1 \leq i \leq n)$, we solve $G(x_1, \dots, x_n) \oplus G_f(x_1, \dots, x_i = v, \dots, x_n) = 1$. Let S_i be the set of all input vectors that can detect a stuck-at-v_i ($v_i = 0$ or 1) fault at x_i and S_k be the set of all input vectors that can detect a stuck-at-v_k ($v_k = 0$ or 1) fault at x_k $(i \neq k)$.

If we assume that two stuck-at-v faults occur at the same time, we can find an input vector that can detect the faults. For example, we solve $G(x_1, \dots, x_n) \oplus G_f(x_1, \dots, x_i = v_i, \dots, x_k = v_k, \dots, x_n) = 1$ to find an input vector that can detect a stuck-at-v_i fault at x_i and a stuck-at-v_k fault at x_k occurring at the same time $(i \neq k)$. Let $S_{i,k}$ be the set of all input vectors that can detect a stuck-at-v_i ($v_i = 0$ or 1) fault at x_i and a stuck-at-v_k ($v_k = 0$ or 1) fault at x_k $(i \neq k)$ occurring at the same time.

Prove or disprove the following statement:

$$S_{i,k} = S_i \cap S_k$$

(If you want to disprove it, you can just show a counterexample.)