EE434

ASIC and Digital Systems

Midterm Exam 2

Apr. 8, 2020. (2:10pm – 3pm)

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

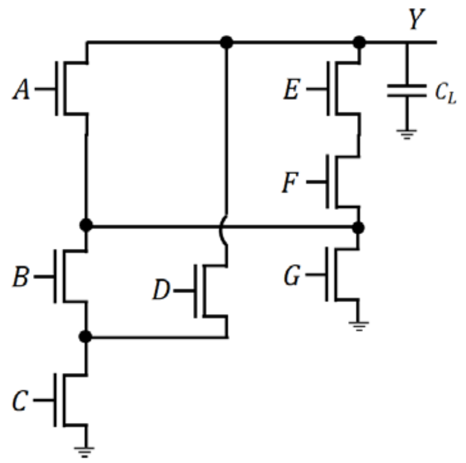
Name:

WSU ID:

Problem	Points	
1	10	
2	10	
3	10	
4	20	
5	10	
6	10	
7	10	
Total	80	

Problem #1 (Transistor Sizing, 10 points)

Size the transistors in the following NFET network. Timing constraint: $\tau \leq R_n C_L$. $\frac{\mu_n}{\mu_p} = 2$. R_n is the resistance of a 1X NFET. Try to minimize the total TR width heuristically.



A:

B:

C:

D:

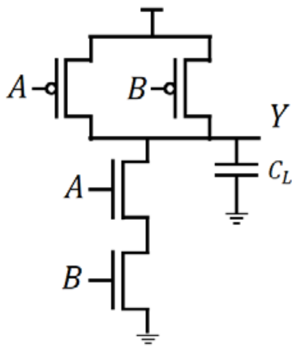
E:

F:

G:

Problem #2 (Transistor Sizing + Timing Analysis, 10 points)

In this problem, we will size the transistors of a gate for more complex timing constraints. $\frac{\mu_n}{\mu_p} = 2$. R_n is the resistance of a 1X NFET. Timing constraint: $\tau \le R_n C_L$.



We usually assume that A and B are available at time 0, so the output Y should be ready by time $R_n C_L$. In this problem, A is available at time $t = 0$ (i.e., the arrival time of signal A is 0), but B is available at time $t = \frac{R_n C_L}{2}$ (i.e., the arrival time of signal B is $\frac{R_n C_L}{2}$). The output Y should be ready by time $t = R_n C_L$. Size the transistors to satisfy the timing constraint (and you should try to minimize the total TR width).

NFETs

A:

B:

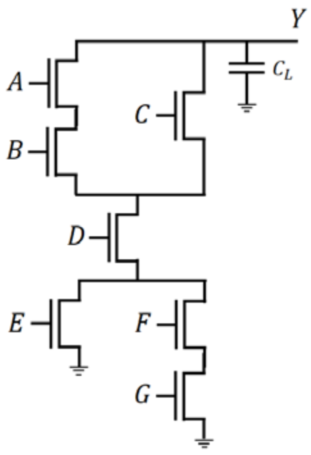
PFETs

A:

B:

Problem #3 (Transistor Sizing + Timing Analysis, 20 points)

This problem is similar to Problem #2. $\frac{\mu_n}{\mu_p} = 2$. R_n is the resistance of a 1X NFET. Timing constraint: $\tau \le R_n C_L$. The following shows the NFET network of $Y = \overline{(AB + C)D(E + FG)}$.



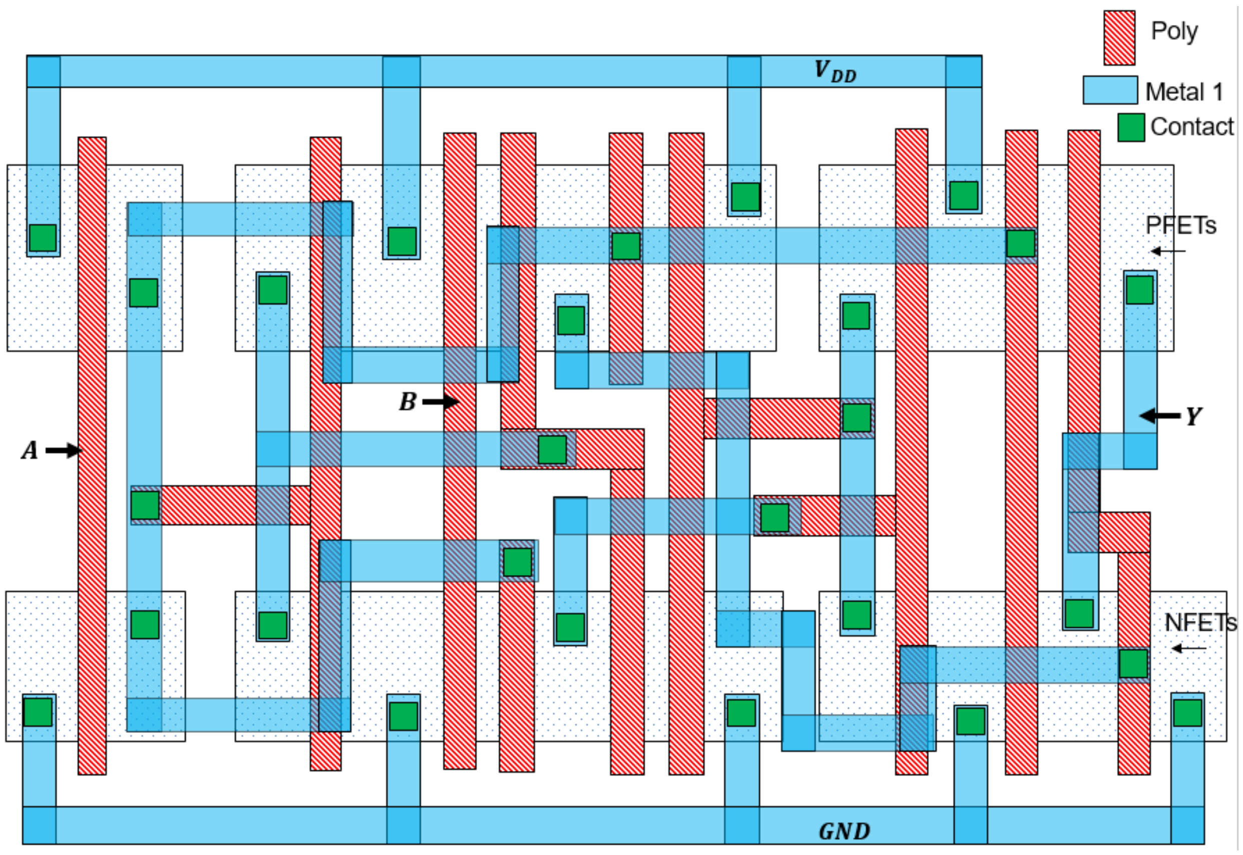
The following shows the arrival times of the input signals. Size the transistors properly to satisfy the timing constraint (and you should try to minimize the total TR width).

- $A: t = \frac{R_n C_L}{6}$
- $B, D, E, G: t = \frac{R_n C_L}{8}$
- $C: t = \frac{R_n C_L}{4}$
- $F: t = \frac{R_n C_L}{16}$

A:
B:
C:
D:
E:
F:
G:

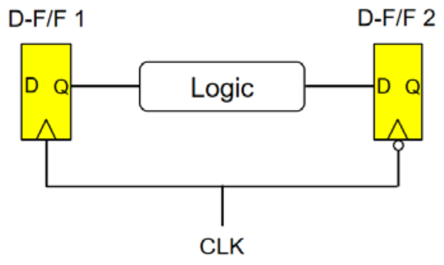
Problem #4 (Layout, 20 points)

Draw a transistor-level schematic for the following layout.



Problem #5 (Static Timing Analysis, 10 points)

If the clock skew for the following logic is too negative or too positive (i.e., too large), it won't work correctly (the signals won't be captured correctly). Derive two inequalities that the clock skew should satisfy.



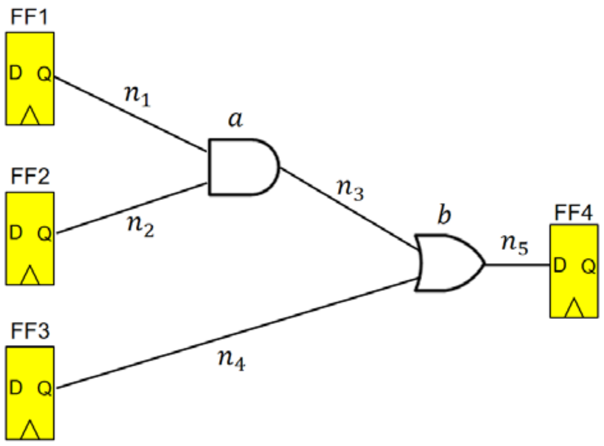
Use the following constants:

- Setup time of D-FF 1, 2: s_1, s_2
- Hold time of D-FF 1, 2: h_1, h_2
- Clock period: T_{CLK}
- Logic delay: T_{logic}
- C-Q delay of D-FF 1, 2: c_1, c_2
- Delay from the CLK source to D-FF 1, 2: d_1, d_2
- The clock skew is defined by $T_{skew} = d_2 - d_1$

Answer: $\boxed{} \leq T_{skew} \leq \boxed{}$

Problem #6 (Static Timing Analysis, 10 points)

Find setup and hold time inequalities that the following logic has to satisfy.



Use the following constants:

- Setup time of D-FF 1, 2, 3, 4: s_1, s_2, s_3, s_4
- Hold time of D-FF 1, 2, 3, 4: h_1, h_2, h_3, h_4
- Clock period: T_{CLK}
- C-Q delay of D-FF 1, 2, 3, 4: c_1, c_2, c_3, c_4
- Delay from the CLK source to D-FF 1, 2, 3, 4: d_1, d_2, d_3, d_4
- Net and gate delays: n_1, n_2, n_3, n_4, a, b

You can also use MAX and MIN operators.

Setup:

Hold:

Problem #7 (Static Timing Analysis + Pipelining, 10 points)

Suppose an (ideally-partitionable) logic is given. Its delay is d. If we partition it into p equally-distributed pipeline stages, the delay of the sub-logic in each pipeline stage becomes $\frac{d}{p}$ (Notice that $p \geq 1$).

If we run N instructions sequentially in the pipeline, it takes total $\#(N + p - 1)$ clock cycles to execute all the instructions. The total execution time is $(N + p - 1) \cdot T_p$ where T_p is the clock period for the p-pipelined system.

All the flip-flops have the same characteristics:

- C-Q delay: c
- Setup time: s
- Hold time: h

Answer the following questions.

(1) Assume $c > 0, s > 0, h > 0, h > c$, and the clock skew is zero. Find the maximum value of p that does not lead to hold-time violations.

(2) Assume $c > 0, s > 0, h > 0$, and the clock skew is zero. Find the minimum value of T_p that does not lead to setup-time violations.

(3) If $c = 0, s = 0, h = 0$, we should always increase p to reduce the execution time (True / False).

(4) If $c > 0, s = 0, h = 0$, we should always increase p to reduce the execution time (True / False).

(5) If $c = 0, s > 0, h = 0$, we should always increase p to reduce the execution time (True / False).

(6) If $c = 0, s = 0, h > 0$, we should always increase p to reduce the execution time (True / False).