EE434

ASIC and Digital Systems

Final Exam

May 5, 2016. (1pm – 3pm)

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

Name:

WSU ID:

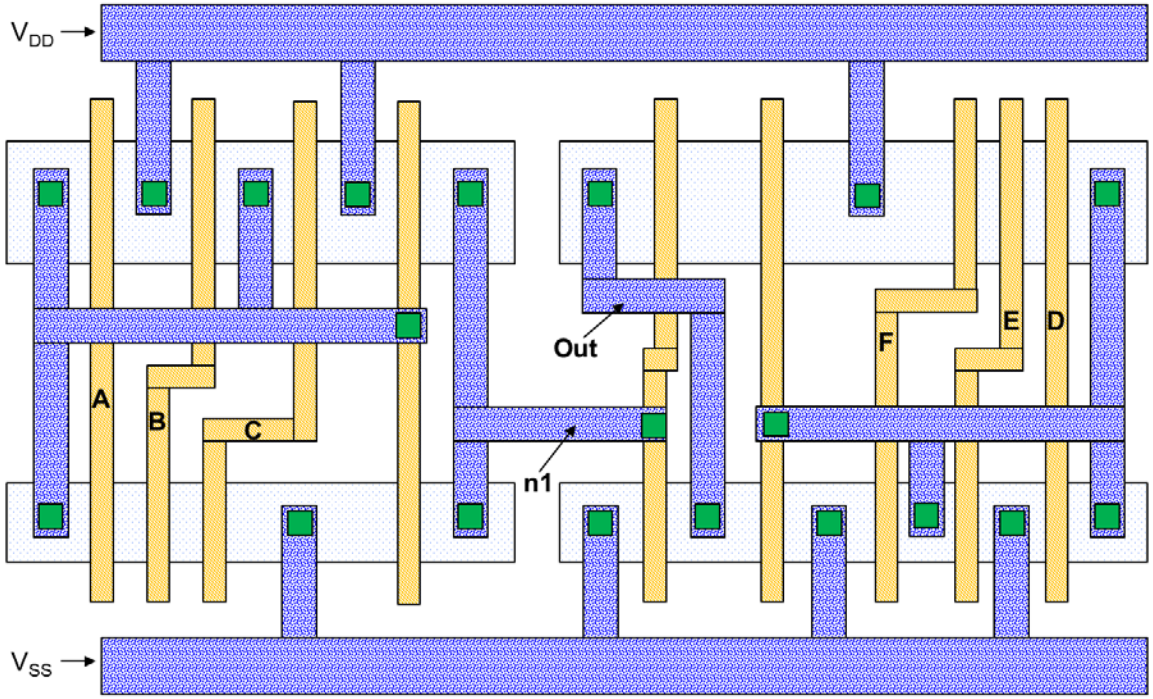
Problem	Points	
1	10	
2	10	
3	10	
4	10	
5	10	
6	10	
7	10	
8	10	
Total	80	

* 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 + Testing, 10 points)

The following layout consists of six primary inputs (A, B, C, D, E, F) and a primary output (Out). n1 is an internal node. Find all input vectors that can detect a stuck-at-0 fault at node n1.

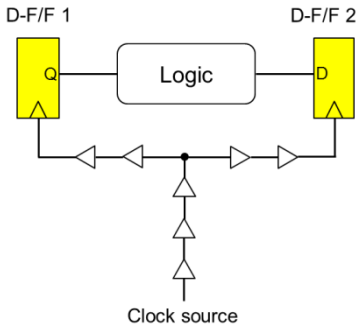


Problem #2 (Testing, 10 points)

A combinational logic has n inputs ($x_1, x_2, \dots, x_n$) and an output (Z). Z is a Boolean function of the inputs, $Z = g(x_1, \dots, x_n, \overline{x_1}, \dots, \overline{x_n}, AND, OR)$. To find input vectors that can detect a stuck-at-v fault f at a node, we compute Z_f by setting the value of the node to v and solving $Z \oplus Z_f = 1$. Let T_1 be a set of all input vectors that detect fault f_1 and T_2 be a set of all input vectors that detect fault f_2 ($f_1 \neq f_2$). Prove that if $Z_{f1} \neq Z_{f2}$, T_1 cannot be equal to T_2.

(Example: A three-input AND gate has three inputs a, b, c and an output Z. Let f_1 be a stuck-at-0 fault at input a and f_2 be a stuck-at-1 fault at input b. Then, $Z = a \cdot b \cdot c, Z_{f1} = 0, Z_{f2} = a \cdot c$ ($Z_{f1} \neq Z_{f2}$). In this case, $T_1 = \{111\}$ and $T_2 = \{101\}$, so $T_1 \neq T_2$.)

Problem #3 (Timing Analysis, 10 points).



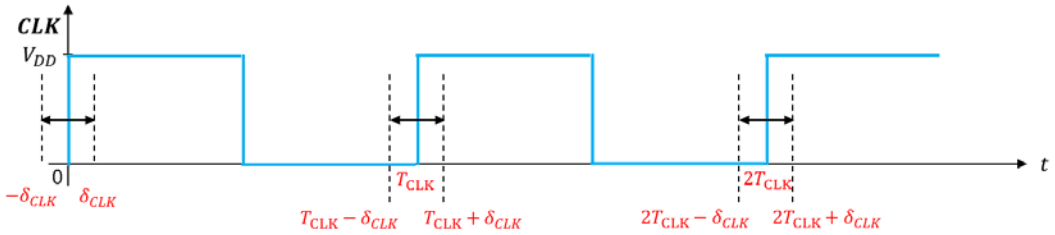
- Setup time of the F/Fs: T_{s}
- Hold time of the F/Fs: T_{h}
- D-F/F internal delay: T_{CQ}
- Clock skew: T_{skew} = delay from the clock source to D-FF2 – delay from the clock source to D-FF1
- Logic delay: T_{logic}
- Clock period: T_{CLK}
- Buffer delay: T_{b}

There are five buffers between the clock source and the clock pin of D-FF1 (and D-FF2) as shown in the figure. Ideally, the following inequalities should be satisfied:

- Setup time: $T_{\mathrm{s}} \leq T_{\mathrm{CLK}} + T_{\mathrm{skew}} - T_{\mathrm{logic}} - T_{\mathrm{CQ}}$
- Hold time: $T_{\mathrm{h}} \leq T_{\mathrm{CQ}} + T_{\mathrm{logic}} - T_{\mathrm{skew}}$

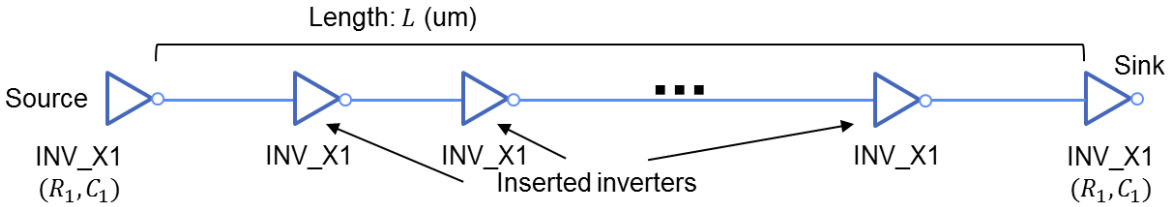
However, there exist uncertainties such as delay variations due to temperature, so we should incorporate those uncertainties (variations) into the setup and hold time inequalities. The followings show the variation sources we are going to consider:

- $T_{\mathrm{b}} \rightarrow T_{\mathrm{b}} \pm \delta_b$
- $T_{\mathrm{CQ}} \rightarrow T_{\mathrm{CQ}} \pm \delta_{CQ}$
- $T_{\mathrm{logic}} \rightarrow T_{\mathrm{logic}} \pm \delta_{logic}$
- Clock jitter: $\pm\delta_{CLK}$



Derive a new setup time and a new hold time constraints (inequalities) that should be satisfied under the variations.

Problem #4 (Interconnect Optimization, 10 points)

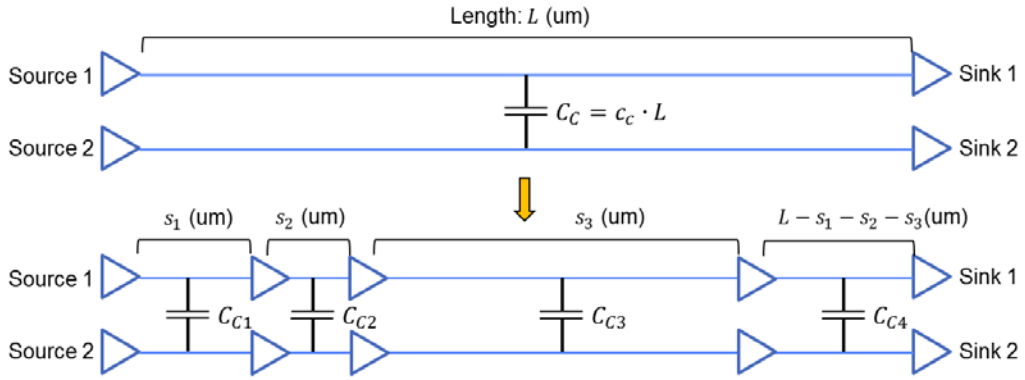


A buffer is composed of two inverters, so it consumes more power than an inverter. Thus, we can insert inverters instead of buffers to optimize a net while minimizing power consumption. However, only an even number of inverters can be inserted (if the inverter count is odd, there will be signal inversion).

In the above figure, the driver, the sink, and the inserted inverters have the same input capacitance (C_1) and output resistance (R_1), so the inserted inverters should be evenly distributed between the source and the sink. Now, suppose the optimal number of inverters we find is k where k is odd. Since k is odd, we have to insert either $k-1$ or $k+1$ inverters. Which will lead to a better result (shorter delay)? $k-1$ or $k+1$? Show all the details about your answer.

- Net length: L (um)
- Inverter output resistance and input capacitance: R_1, C_1
- Unit wire resistance and capacitance: r, c
- Inverter delay: d

Problem #5 (Interconnect Optimization, 10 points).

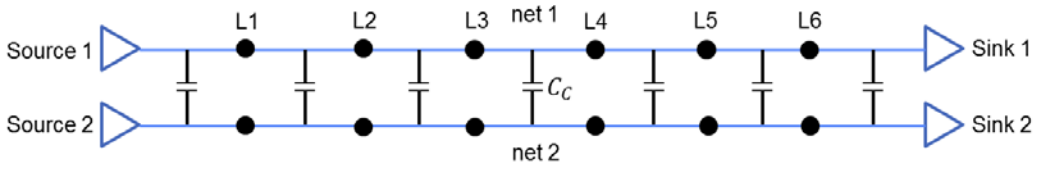


Two nets are routed as shown above. The coupling capacitance between them is C_c. You are supposed to insert buffers into the nets identically as shown above. All the drivers, sinks, and buffers are of the same type. The following list shows all the parameters and variables you should use:

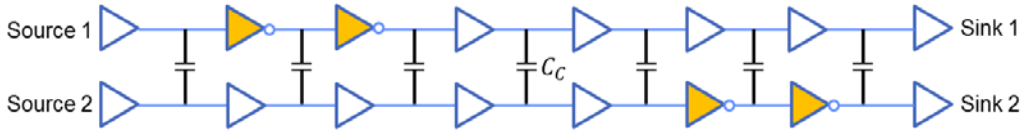
- Output resistance: R
- Input capacitance: C
- Total length: L (um)
- Unit wire resistance: r_w/um
- Unit wire capacitance: c_w/um
- Unit coupling capacitance: c_c/um
- Buffer delay: d

Insert buffers into the nets optimally, i.e., find the number of buffers to insert and their locations to minimize the delay of the nets.

Problem #6 (Interconnect Optimization, 10 points).



Two nets (net 1 and net 2) are coupled as shown above and you are supposed to insert a repeater (inverter or buffer) into each of the designated locations (L1 ~ L6). For instance, the following figure shows a repeater insertion solution (its delay and area are $12d$ and $20S$, respectively):

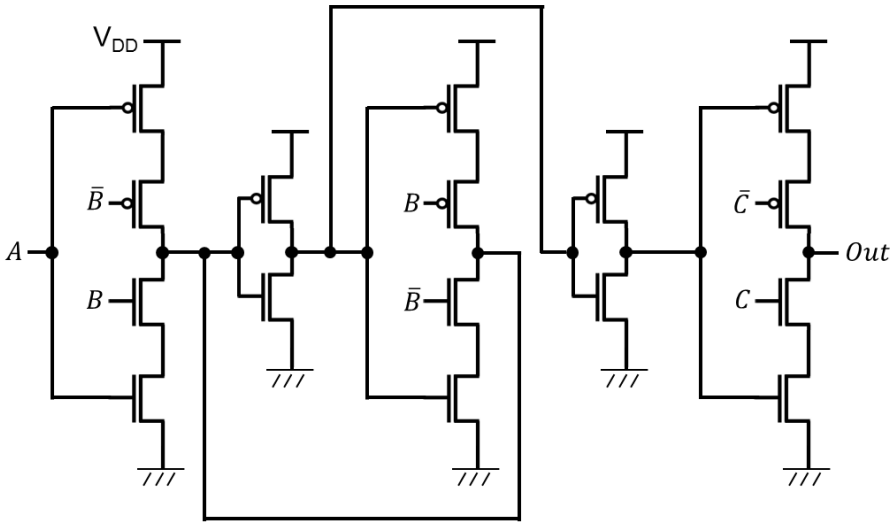


The following list shows the parameters used in this problem:

- Area of an inverter: S
- Area of a buffer: $2S$
- Delay of a net segment whose total capacitance is C_g: d
- Delay of a net segment whose total capacitance is $C_g + C_c$: $1.5d$
- Delay of a net segment whose total capacitance is $C_g + 2C_c$: $2d$
- Input pattern of net 1: 010101…
- Input pattern of net 2: 101010…
- Signal inversion at the sinks is not allowed.

The goal is to optimally insert repeaters to minimize the sum of the delay values of the nets. However, you should also minimize the total area. Find an optimal solution that minimizes the sum of the delay values and the total area. Notice that minimizing the total delay has a higher priority. Thus, if there exists only one solution that minimizes the total delay, find it. If there exist multiple solutions that minimize the total delay, find the smallest-area solution among them.

Problem #7 (CMOS Gates, 10 points).



What does the above circuit do? Describe the function of the circuit in as much detail as possible (A, B, C: input, Out: output).

Problem #8 (Adder, 10 points).

Draw a gate-level schematic of a four-bit conditional sum adder (use full adders and muxes). Input: A[3:0], B[3:0], CI. Output: S[3:0], CO.