ECE 6473
Revision up to mid-term

Concepts covered in class

1. **Scaling according to Moore's law**
    a. Why does reducing the size of transistors on the chip help to improve the transistor performance?
    b. Goals of scaling: reduce gate delay by 30%, increase operating frequency by 43%, double the transistor density, reduce energy dissipation by 65%, power reduces by 50%.

2. **Equivalent transistor resistance**
    a. Understand that the transistor is a non-linear resistor. The value of the resistance depends on the terminal voltages: V_{gs}, V_{ds}, V_{bs}.
    b. Modes of transistor operation: Cut-off and ON (linear and saturation)
    c. Typically NMOS is used to pull node from V_{DD} to GND. PMOS is used to pull up a node from GND to V_{DD}. Be careful about body effect when the source of NMOS is not grounded and is changing voltage. Body effect for PMOS is important when the source is not connected to V_{DD}. Look at examples covered in HW#2.

3. **Calculation of equivalent transistor resistance**
    a. Identify the large-signal resistance of the transistor at the beginning of the transition and halfway through the transition. Lets say these large-signal resistances are R_1 and R_2. Then $R_{eq} = 0.5(R_1+R_2)$. Typically, the transistor will be in saturation mode to compute R_1 and R_2. Make sure you use the right values of the terminal voltages for calculating R_1 and R_2.
    b. Only large-signal resistances R_1 and R_2 must be used. Do not bother with small-signal resistances. I will NOT quiz you on that.

4. **Scaling trend of R_{eq}**
    a. R_{eq} increases when (i) V_{DD} decreases, (ii) (W/L) decreases, (iii) V_T increases, (iv) (μCox) decreases.

5. **Effect of μ_p and μ_n on R_{eq}**
    a. Typically for a process, $\mu_p < \mu_n$. Therefore, for the same (W/L) of PMOS and NMOS, PMOS has higher equivalent resistance than the NMOS transistor.
    b. This also means that for the same (W/L), PMOS will deliver lower current than the NMOS device.

6. **Charging/discharging time constant**
    a. Always given as $0.69R_{eq}C_L$. So two things you must know are equivalent resistance and load capacitance.

b. R_{eq} is computed using averaging method as explained in point (3a). C_L is computed from the self-loading capacitance PLUS the fan-out and wire capacitance. Understand each component of C_L properly.

7. Self-loading or parasitic capacitances

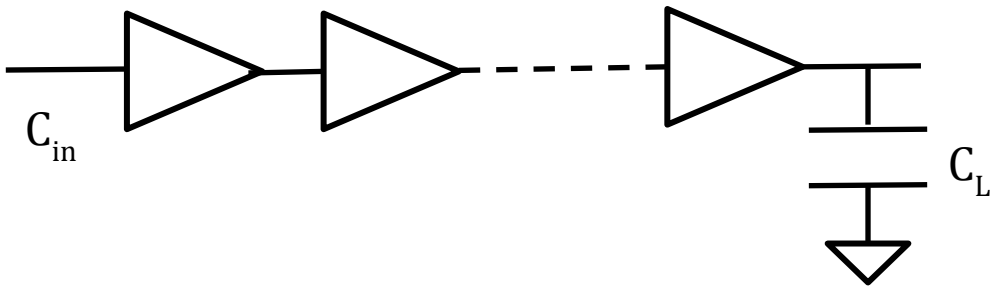
a. Parasitic capacitance consists of overlap (source and drain) and junction (source and drain diode) capacitance.
b. Note that overlap capacitance is a Miller capacitance. Therefore, when you transfer it to the output node, its value must be doubled. All parasitic capacitances scale with the width of the transistor (note that the length of the transistor is not relevant here).

8. Inverter analysis

a. Understand the figures of merit: (i) switching threshold, (ii) low-to-high (PMOS) and high-to-low (NMOS) delay, (iii) gain
b. The formula for sizing the inverter (PMOS and NMOS) for a given switching threshold must be clear. However, qualitatively when PMOS is stronger then V_M moves toward V_{DD}, while when NMOS is stronger then V_M moves toward GND.
c. What does a transistor being "strong" really mean? It means that the transistor has high saturation current. A transistor can be made strong by increasing its size or by increasing the effective voltage drop (V_{GS}-V_T) in saturation.
d. Switching delay: $t_{pLH} = 0.69R_{eqp}C_L$ while t_{pHL}: $069R_{eqn}C_L$.
e. For same switching delay: $t_{pHL} = t_{pLH}$, we want to have Reqp = Reqn. This means, $I_{DSATp} = I_{DSATn}$. For the same threshold voltage, $V_{Tn} = |V_{Tp}|$, this condition means $\frac{W_p/L_p}{W_n/L_n} = \frac{\mu_n}{\mu_p}$. Typically, $\mu_n > \mu_p$. So we must make PMOS bigger.
f. Reference inverter/min.-sized inverter: Usually we fix $L_n = L_p = L_{min}$ given by the technology. For 0.25 micron, we will have Ln = L_p = 0.25 µm. Then we have NMOS width, $W_n = L_n$ = 0.25 µm. We adjust W_p of the PMOS such that we get $W_p/W_n = \mu_n/\mu_p$. This is also a balanced inverter because the pull-up and pull-down have the same resistance.
g. Self-loading of inverter: The self-loading capacitance (C_{FET}) at the output of the inverter comes from junction and overlap of both NMOS and PMOS. That is, $C_{FET} = (C_{db\_FET,n} + 2C_{gd\_FET,n}) + (C_{db\_FET,p} + 2C_{gd\_FET,p})$.
h. So $C_{FET} = (1+\alpha)C_{FET,n}$, where $\alpha = W_p/W_n$ in the inverter.
i. Assuming the inverter of size $\alpha = W_p/W_n$ is driving another inverter of the same size, then the question is how should you choose α to minimize the delay of the driving inverter. It turns out that here you must choose $\alpha = \sqrt{\mu_n/\mu_p}$.

9. **Sizing a chain of inverters to drive a large load**.

a. Consider the scenario shown below

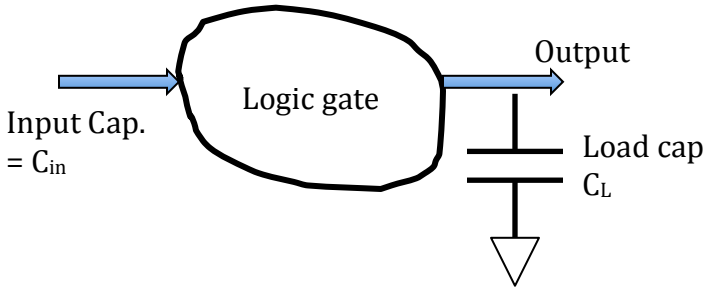


b. If there is a big capacitance C_L, then it makes sense to insert a chain of inverters to drive that big capacitance. The questions are: (i) When do I know that I need to insert a chain of inverters in a logic? (ii) How many inverters do I need in the chain? (iii) What is the size of each inverter?

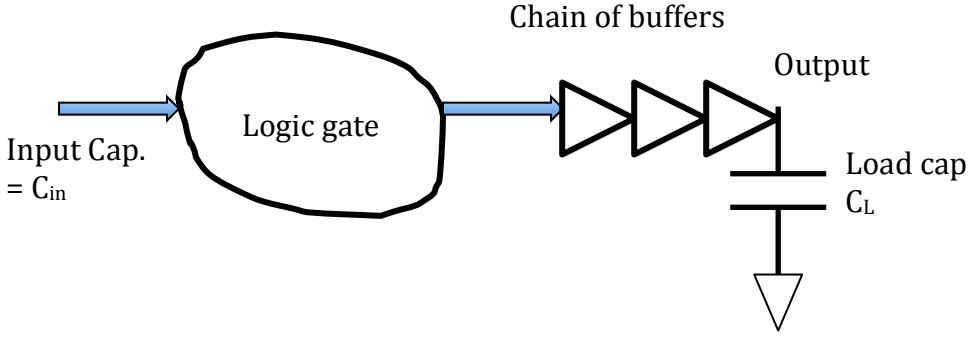
c. **Answers: (i) You need to use a chain of inverters to drive a big load when typically $C_L > 4C_{in}$. (ii) optimal number of stages $N_{opt} = \ln(C_L/C_{in})$ (iii) Each inverter will be up-sized by u = e = 2.712 (for no parasitic load) and u = 3.6 for (parasitic load p = 1). That is make each inverter bigger by a factor of "u" compared to its previous inverter.**

10. Inserting inverter chain after a logic gate

Consider the following scenario.



If C_L is big compared to C_{in}, then it is better to insert a chain of properly sized inverters between the logic gate and the load capacitance. That is, the following circuit is a better option. This is also called isolating fan-in from fan-out.



a. Use inverter chain when $C_L > 4\ C_{in}$, even when there is a complex logic gate driving the load.
b. The inverter sizing is similar to the one mentioned in part (9c). Size each inverter up by a factor of "u" where u = 2.712 (when no parasitic) and u = 3.6 when parasitic delay (normalized) = 1.
c. Total number of stages: $\mathbf{N_{opt} = ln(F)/ln(u)}$, where F = total path effort.
d. In this analysis (when using inverter chain after a complex logic gate), make sure you count the number of stages even in the logic gate. Compute the total F for the structure which is simply F = GBH, where G = product of all g, H = product of all h, and B = product of all b.

11. Energy dissipation in inverter

a. Energy drawn from supply: $E_{VDD} = \int_0^\infty i_{VDD}(t)V_{DD}(t)dt$
b. Energy stored in capacitor: $E_C = \int_0^\infty i_c(t)v_{out}(t)dt$
c. Energy dissipated in resistor (or transistor) $E_R = \int_0^\infty i_R^2(t)dt$
d. Average power dissipation: $P_{avg} = \alpha_{0\to1}C_L V_{DD}^2 f_{CLK} + \alpha_{\to10}\, E_{SC} + V_{DD}I_{static}$, where $\alpha_{0\to1}$ is the activity factor. It is given as $\alpha_{0\to1} = p_0 p_1$, where p_0 is the probability that the output node is initially evaluated as "0 and p_1 is the probability that the node is in logic state "1".

12. Static CMOS gates

a. Consist of pull-up PMOS and pull-down NMOS network
b. Size each network by identifying the worst case delay and make sure that the worst case gives you the same pull-up and pull-down resistance as the reference inverter in that technology
c. Metrics: Switching threshold, low-to-high (PUN) and high-to-low (PDN) delays. These metrics depend on input patterns unlike the inverter case.
d. When finding worst-case delays, make sure you consider internal node capacitance. Refer to HW#3 problem.
e. Impact of fan-in and fan-out on the delay: $Delay = a_1 FI + a_2 FI^2 + a_3 FO$. Here FI = fan-in and FO = fan-out. Having large fan-in is not good as the delay scales quadratically with fan-in.
f. Consider these techniques for reducing the delay of complex logic gates: progressive sizing input-reordering, alternative logic structures, and isolating fan-in from fan-out.
g. *Isolating fan-in from fan-out*: add a chain of buffers after a complex logic gate when the complex gate is driving a large load. Refer to point 10 in these notes.

13. Definition of logical effort and how to compute it for a given gate

a. *Definition: The logical effort of a logic gate is defined as the ratio of its input capacitance to that of an inverter that delivers the same amount of curren.t*
b. First you must know what the min. sized or reference inverter in your technology.
c. Then consider the complex gate for which you are trying to find the logical effort. Size this complex gate such that it delivers the same amount of current as the min. sized inverter. This also means that the worst-case pull-up and pull-down resistances of the complex gate are the same as the reference inverter.
d. Now simply compute g = C_{gate}/C_{ref}, where C_{ref} is the input capacitance of the reference inverter.
e. Note that "*g*" does not change with the sizing of the complex gate, since by definition you're evaluating it for the minimum-sized gate for which the Pull-up and Pull-down resistances match that of the ref. inverter.
f. Very important: for a complex gate with multiple inputs, the logical effort can be different for different inputs.

Note: several students asked me why "g" does not change if I make the NAND gate bigger than what we discussed in class. The answer is because, by definition, you're computing "g" for that NAND gate that is matched with the ref. inverter. Hence, sizing does not play a role here.

14. Logical effort for complex gates

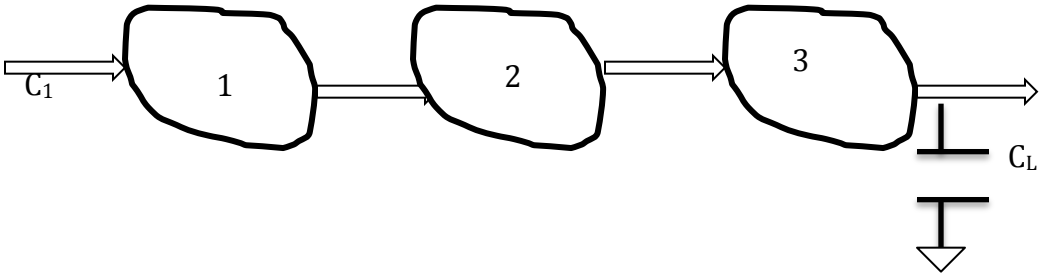
a. For NAND and NOR gates, logical effort does not depend on the specific input. That is, for all the inputs in multiple input NAND and NOR gates, g_{NAND} = (N+r)/(1+r) and g_{NOR} = (1+nr)/(1+r), where N = fan-in of the gate.
b. Always remember that the logical effort must be defined per input. For NAND and NOR, each input has the same logical effort as they are symmetric gates. You can construct asymmetric gates where logical effort will depend on the specific input under consideration.

15. Delay of a logical path

a. Delay of a logic path with several complex gates is given as $D = \sum_{i=1}^{N} p_i + g_i h_i b_i$, where "*i*" denotes the i^{th} stage of the path. p_i is the parasitic delay, g_i = logical effort, h_i = electrical effort and b_i = branch effort.
b. Note that the delay expressed above is the normalized delay with respect to $\tau_{ref} = R_{ref} C_{ref}$, where R_{ref} is the resistance of the reference inverter and C_{ref} is the input capacitance of the reference inverter.
c. Remember τ_{ref} is NOT the delay of the reference inverter. The normalized delay of the reference inverter is $d_{1X} = p + h$ since g = 1.

16. Optimizing a logic path

a. Consider the situation shown in figure below. How should we size each gate in the logic path? *Note: the number of stages in this problem is already known.*



b. Basic idea is to make the stage delay of each stage identical.
c. First compute the total path effort F = (GHB), where G = g1*g2*…gN (product of g's of all stages); H = C_L/C_1; B = b1*b2*…..bN (product of all b's of all stages).
d. Now each stage will have an equal effort $f = F^{1/N}$. Use F calculated from (c) above and N is already known. At this point, you already know the stage effort of each stage.
e. For each stage you know "g" and "b". What you don't know is "h". But you know "f" from (d) above. So compute $h_i = f/(g_i b_i) = F^{1/N}/(g_i b_i)$
f. Once you have optimized the logic chain, the optimal delay is D_{opt} = $(NF^{1/N})$+P, where P is the sum of all parasitic delays.

17. Other logic styles

a. *Pseudo-NMOS*: fewer transistors, but VOL is not zero. Also ratioed design.
b. *Pass transistor logic*:
    i. uses transistors as switches.
    ii. Fewer transistors but has threshold voltage drop.
    iii. Also NMOS will be under body effect and that aggravates the problem of threshold voltage drop.
    iv. If this logic gate feeds into an inverter, then the fan-out inverter will have static power dissipation.
    v. Use level restorer to pull the node back to V_{DD}. However, sizing of level restorer must be considered.
c. *Complementary pass transistor*:
    i. Modular logic style
    ii. threshold voltage drop ☹ leading to static power dissipation in fan-out inverter

iii. Both inverting and non-inverting outputs are available

d. Transmission gate (TG):

i. Contains both PMOS and NMOS so there is no threshold voltage drop
ii. Useful for multiplexers (MUX)
iii. Propagation delay increases as n^2, where n = no. of select signals in the TG MUX.
iv. If implementing a big MUX more than 4 signals, decoding first is a useful thing to do.

e. Dynamic logic

i. Relies on storage of charge at a capacitor node (high impedance node and susceptible to noise and charge leakage)
ii. Faster than static logic but power dissipation is more than static; also loads the clock signal on the chip ☹
iii. Two cycles: precharge and evaluate using a clock signal
iv. Logic is only implemented by the pull-down network; the design is non-ratioed and full 0 to VDD swing at the output
v. Noise margins are poor
vi. Some common issues: charge leakage during evaluate phase when the PDN is not on (use a level restorer or keeper circuit), charge sharing from output node to internal node (imposes a min. limit on clock frequency; can be avoided by pre-charging the internal nodes), cascading dynamic gates.
vii. Design of level restorer must be done carefully.

18. Power dissipation in complex logic

a. Dynamic power dissipation depends on the activity factor at the output node, which is related with the signal probabilities.
b. Evaluate: $\alpha_{0\to 1} = p_0 p_1$, where p_0 = probability OUT = 0 and p_1 = probability OUT becomes 1.
c. Carefully estimate p_0 and p_1 from the input signal probabilities especially when signals are correlated and not equally distributed (i.e. when the probability of the signal being 1 is not equal to 0.5)
d. Techniques to reduce switching activity: (i) logic restructuring, (ii) input reordering, (iii) glitch reduction
e. Pay special attention to activity factors at internal nodes.
f. It is beneficial to introduce the signal with higher transition probability later on in the chain.

19. Sequential logic

a. Synchronized with a clock signal
b. Relies on feedback loop and stores "1" or "0". The feedback loop must be cut open or overcome to change the stored state value.
c. Latches (level sensitive) and registers or flip flops (edge sensitive)
d. Important timings: set-up time, C-Q delay and hold delay

e. Set-up time, C-Q delay and logic delay impose a limit on the maximum clock frequency
f. Hold time depends on contamination delay of the register and the combinational logic.
g. Understand the issue of clock overlap in several implementations of the flip-flop. Clock overlap causes race condition. Two-phase clock can be used to overcome this issue.
h. Final word: Clock loading is a serious issue. Try reducing the clock load. However, the challenge then is the robustness and the design of the flip-flop.
i. Use dynamic latch carefully and only when you want to speed up things.