Solution for written examination for

MCC092 Introduction to Integrated Circuit Design

Thursday December 22, 2016, at 8.30-13.30 at SB building

---

1. Logical functions, layout, transistor sizing.

a) The logical functions are inverter (Y1) and nand (Y2-Y4). The expressions for Y1-Y4 are shown in Figure 1a) below. (2 p)

b) The layout is shown in Figure 1b) below. (5 p)

c) There are several solutions. One possibility is to leave the n-net unaltered so that all four nMOS transistors have the same resistance, let's call it R, and the corresponding width **W**, for nMOS transistors 121- 124. Then the n-net will have these resistances for the four logic functions, Y1: R, Y2: 2R, Y3: 3R and Y4: 4R. The worst-case situations for the p-nets are when only one of the pMOS transistors is conducting for that logic function. Thus, transistor 161 should have resistance R (that is width **2W),** transistors 151-152 should have resistances 2R (that is width **W**), transistors 141-143 should have resistances 3R (that is width **2/3 W**) and transistors 131-134 should have resistances 4R (that is width W/2) (3 p)

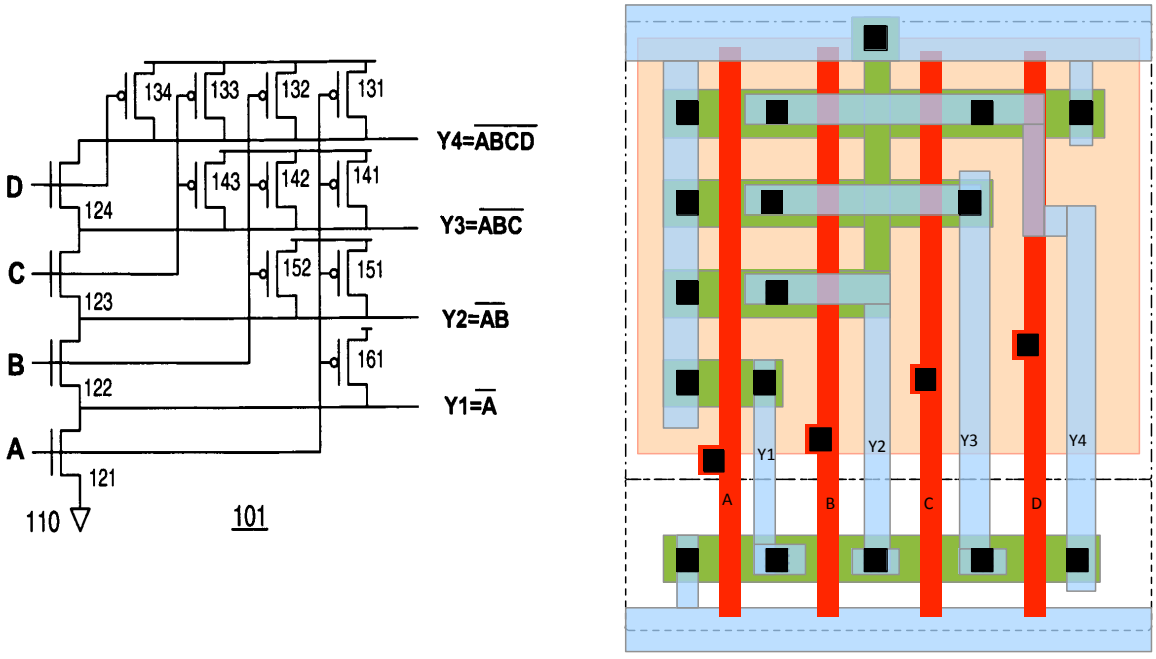


Figure 1 *a) A multiple-input, multiple output static CMOS gate with inverter and nand logic. b) The layout for the gate in Figure 1 a.*

2. Voltage transfer curve, noise margin

a) X (red) is Y4, Y(green) is Y3, and W(purple) is Y2. Motivation: With all inputs connected together we have four inverters with different p-to-n resistance ratios. Assuming that nMOS transistors have twice the current of pMOS transistors (which we know is very close to reality for the 65-nm process), the p-to-n resistance ratios for Y1–Y4 are 2R/R, R/2R, 0.67R/3R 0.5R/4R (Note that this case is not the same the as the one in task1 c) since here all the pMOS transistors are conducting simultaneously). To know exactly where the switching voltages fall for the four inverters, we would need to know the threshold voltages. However, from lab 1 we (hopefully) remember that the 2-to-1 scaled inverter in the 65-nm process has a switching voltage very close to $V_{DD}/2$. So then the VTC that is **not** drawn has to be the one where the p-to-n resistance ratio is larger than 1, that is Y1. The ordering of the three drawn VTCs has to be according to the p-to-n resistance ratios. In Figure 2 below is a diagram of all four VTCs is shown.
Figure 2: ***(To be added later)*** (3 p)

b) The definition of noise margin is how large a margin there is for the signal to be degraded (for example by noise) between the output of a gate and the input of the next (similar) gate. Margins are always positive so one has to consider the signs in the definitions. For a HIGH signal the margin is the difference between the minimum output voltage that counts as HIGH level, and the minimum input voltage that counts as a HIGH level: $\mathbf{V_{OHmin} - V_{IHmin}}$. For a LOW signal the noise margin is the difference between the maximum input voltage that counts as a LOW level and the maximum output voltage that counts as a LOW level: $\mathbf{V_{ILmax} - V_{OLmax}}$. One usually uses the point where the gain is -1 in VTC to define these voltages. It is increasingly important to make designs with high noise margins because the supply voltages are shrinking, making logic more susceptible to noise, and there are many more noise sources on modern chips. Well-designed static CMOS gates have high noise margins and that is one reason why static CMOS is dominating in logic design today (the other reason is that other types of CMOS logic are much harder to use in logic synthesis) (3 p)

c) In Figure 2 below the two points that define the four voltages are shown. The resulting values are: For high level $M_{NH} = V_{OHmin} - V_{IHmin} = 1.19\ V - 0.832\ V = 0.358\ V$. For low level $M_{NL} = V_{ILmax} - V_{OLmax} = 0.735\ V - 0.05\ V = 0.73\ V$. As could be expected from the VTC the noise margins are not very equal so this inverter is not so well designed. (4 p)

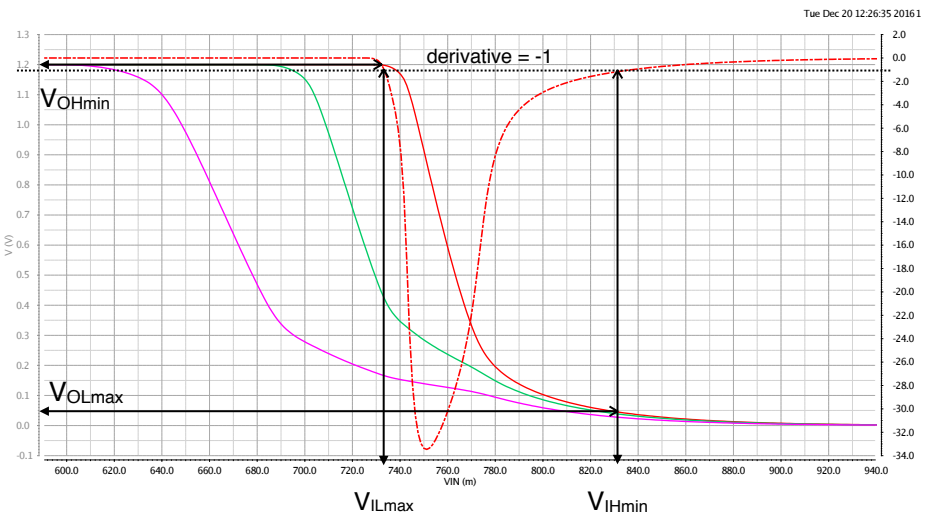


Figure 3: *VTC with voltages necessary to calculated noise margins for output X indicated.*

3. Logical effort, path effort

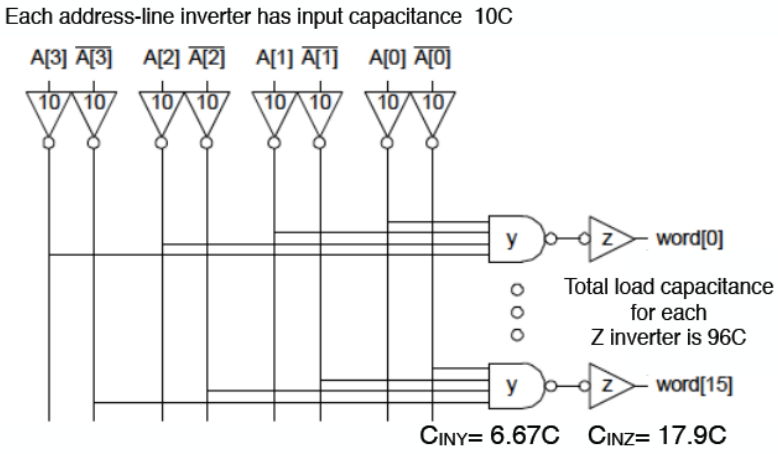


Figure 4*: The detailed schematic of the decoder circuitry with the resulting sizes for task a).*

a) This is a problem where path delay is applicable. The driver inverters each drive eight wires connected to eight of the nand gates (to half of the 16 words). The load capacitance for the z inverter is $16 \cdot 3C = 96C$. The logical effort for the 4-input nand-gate is 2 (which one may have to arrive at from the circuit schematics, though not included here). So for the path we have $G = 1 \cdot 2 \cdot 1$, $B = 8 \cdot 1 \cdot 1$ and $H = \frac{96C}{10C} = 9.6$. All in all the path effort, $F = GBH$ is $2 \cdot 8 \cdot 9.6 = 153.6$. The optimal stage effort is then $f_{opt} = \sqrt[3]{153.6} = 5.36$. We can find the sizing by starting from the output or the input, but usually it is easier to start from the output. We use the relation $f_{opt} = g \cdot h$ for each gate and determine the input capacitance that makes this relation true. Inverter z should accordingly have an input capacitance $C_{inz} = \frac{96C}{5.36} = 17.9C$. The 4-input nand gate should have an input capacitance of $C_{iny} = \frac{2 \cdot 17.9C}{5.36} = 6.67C$.

We should also check that the relationship holds for the first inverter: $C_{inx} = \frac{8 \cdot 6.67C}{5.36} = 9.95C$. It is not exactly $10C$ because there has been some rounding in the calculations, but it is close enough to convince us that we did not make any calculation mistake. (5 p)

b) The normalized delay $D = 3 \cdot f_{opt} + \sum p$. The inverter has a parasitic delay of $p_{inv} = 1$. But the part we do not know is the parasitic delay of the 4-input nand gate. From the schematic we deduce that its parasitic delay for a scaled 4-input nand gate is 4 (this calculation is not included here). Thus we arrive at $D = 3 \cdot 5.36 + 1 + 4 + 1 = 22.08$. In the 65-nm process which has $\tau = 5\text{ps}$ we would thus have a delay of 110ps. (2 p)

c) The difference from the case in task a) is that the word capacitance is doubled which doubles *H*. In this case we have $F = GBH$, $2 \cdot 8 \cdot 19.2 = 307.2$. With four stages we get $f_{opt} = \sqrt[4]{307.2} = 4.18$. The resulting delay with parasitics is then $= 4 \cdot 4.18 + 1 + 4 + 1 + 1 = 23.72$. With three stages we instead have $f_{opt} = \sqrt[3]{307.2} = 6.74$ and a total delay of $3 \cdot 6.74 + 1 + 4 + 1 = 26.22$. So the answer is yes, the delay will be shorter with an extra inverter. (3 p)

4. Power, energy consumption

Preliminaries

Application: The digital video in the video-rendering application has 25 frames per second. Thus the maximum time for the calculations for one frame is 40 ms. One frame is 640x480 pixels, that is 307 200 pixels.

PP processor: The PP processor requires 3 072 000 clock cycles for the calculations for one frame.

a) The first task is to fill in the empty cells in Table 1. To find the maximum clock frequency for lower V_{DD} we need to know how the delay (that is τ) in the process scales with V_{DD}. We know that $\tau = 0.7RC$. In this expression only R changes with V_{DD}. We know that $R = V_{DD}/I_{DSAT}$ and that $I_{DSAT} \sim (V_{DD} - V_T)^2$ if we assume that the quadratic current equations hold as stated in the problem. Thus, the ratio of the max clock frequencies at two supply voltages can be written as:

$$\frac{f_{clk2}}{f_{clk1}} \sim \frac{\tau_1}{\tau_2} \sim \frac{\frac{V_{DD1}}{I_{DSAT1}}}{\frac{V_{DD2}}{I_{DSAT2}}} \sim \frac{\frac{V_{DD1}}{(V_{DD1} - V_T)^2}}{\frac{V_{DD2}}{(V_{DD2} - V_T)^2}}$$

For our three supply voltages $\frac{V_{DD}}{(V_{DD}-V_T)^2}$ evaluates to: 1.2V: 1.48, 1.0V: 2.04, 0.8V: 3.2.

Thus, we have $\frac{f_{clk1.0}}{f_{clk1.2}} = \frac{1.48}{2.04} = 0.73$ and $\frac{f_{clk0.8}}{f_{clk1.2}} = \frac{1.48}{3.2} = 0.46$.

The current corresponding to the dynamic power consumption at the maximum clock frequency is $f_{clkmax}\alpha C V_{DD}$ (it has to be multiplied by the voltage to get the dynamic power). So it scales with the maximum clock frequency (calculated above) and the supply voltage (the activity factor and capacitance does not change) as

$$\frac{I_{dyn2}}{I_{dyn1}} \sim \frac{f_{clk2}V_{DD2}}{f_{clk1}V_{DD1}}$$

Thus we have $\frac{I_{dyn1.0}}{I_{dyn1.2}} = 0.73\frac{1.0}{1.2} = 0.608$ and $\frac{I_{dyn0.8}}{I_{dyn1.2}} = 0.46\frac{0.8}{1.2} = 0.306$. The resulting current values are in Table 1 below. (4 p)

Table 1: Data for the PP processor

Supply voltage V_{DD} [V]	Maximum clock frequency [GHz]	**Current** due to dynamic power consumption @ max clock frequency and a realistic activity factor [mA]	Idle current @ room temperature @ max clock frequency and a low activity factor [mA]	Static current in **sleep mode** @ room temperature (clock signal turned off for logic, but clock generation maintained) [mA]	Static current in **hibernation mode** (clock generation stopped and internal supply voltages turned off) [µA]
1.2	1.0	600	100	60	60
1.0	**0.73**	**438**	80	37.5	60
0.8	**0.46**	**184**	64	28	60

Sleep mode: The time it takes to enter sleep mode is 10 µs and it takes 20 µs for the processor to wake up from sleep mode. The energy required to switch the clocks off is 10 µJ. **Hibernation mode**: The time it takes to enter hibernation mode is 1 ms and it takes 19 ms to wake up from hibernation mode. The energy required to turn off V_{DD} is 500 µJ.

b) It takes the same number of cycles to do the calculations at both voltages, but the cycle time differs. At 1 GHz the cycle time is 1ns. At 460 MHz the cycle time is 2.17 ns. The energy is power times time. Thus, we arrive at:

$$E_{dyn} = I_{dyn} \cdot V_{DD} \cdot t_{cycle} \cdot N_{cycles}$$

And we have for 1.2 V and 0.8 V

$$E_{dyn1.2} = 600[\text{mA}] \cdot 1.2[\text{V}] \cdot 1[\text{ns}] \cdot 3.07\ [\text{M}] = \ 2210\ [\mu\text{J}]$$

$$E_{dyn0.8} = 184[\text{mA}] \cdot 0.8[\text{V}] \cdot 2.17[\text{ns}] \cdot 3.07\ [\text{M}] = \ 918\ [\mu\text{J}]$$

so a large reduction of the dynamic energy at the price of a more than doubled computations time.

c) The time in sleep mode is the time left after the computations for a frame are done. At 1.2 V the necessary computations take 3 ms and at 0.8 V they take 8 ms (we neglect the time it takes to switch to sleep mode since it is so short in comparison). The remaining times for a frame are 40 – 3 = 37 ms and 40-8 = 32 ms respectively. Thus, we have:

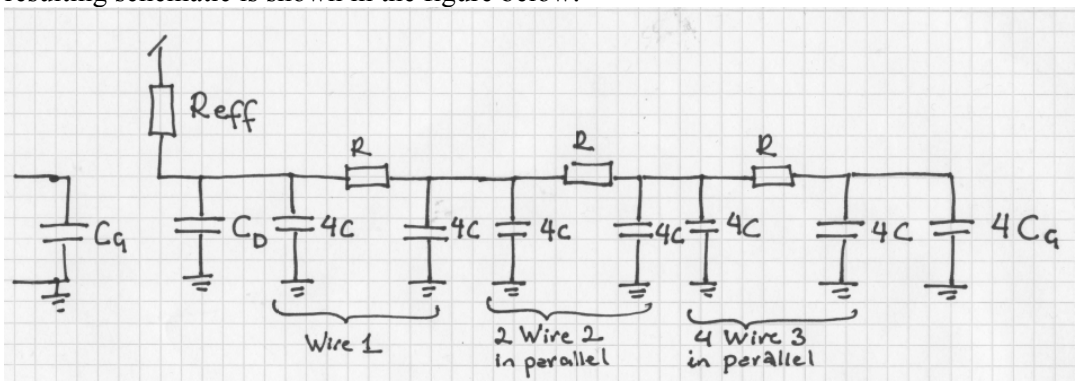
$$E_{sleep1.2} = 60[\text{mA}] \cdot 1.2[\text{V}] \cdot 37[\text{ms}] \cdot = \ 2664[\mu\text{J}]$$

$$E_{sleep0.8} = 28[\text{mA}] \cdot 0.8[\text{V}] \cdot 32[\text{ms}] \cdot = 717\ [\mu\text{J}]$$

d) Since we have so much time available for the computations for one frame we should run at the lowest possible supply voltage, in this case 0.8 V, to save energy. At that supply voltage the energy required to switch off the power supply is larger than what we could save during the 12 ms when the power supply would be off. So the answer is NO. We should not use hibernation mode for this application. (2 p)

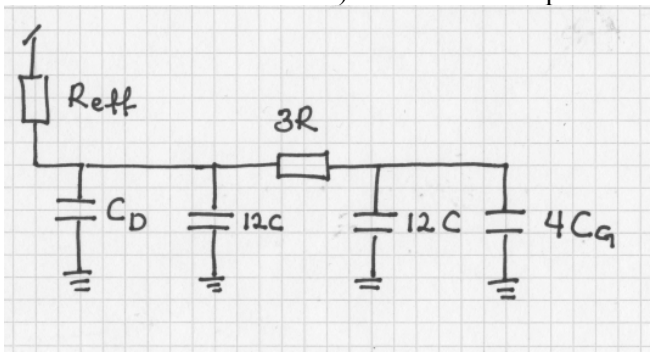
5. Wire delay

a) This problem can be solved by collapsing the H-tree the way we did in lab 4; it is also possible apply Elmore's formula directly and handle the branches explicitly. Here we use the collaped tree. The resulting schematic is shown in the figure below:



Figire

Since the three resulting wire segments are identical we can merge them into one segment to further simplify the calculations (You may remember from lab 4, that in the Cadence simulations the number of segments a wire is divided into makes a difference in the delay calculations, but in the hand calculations it does not). The further simplified schematic is shown below:



We use Elmore's formula to find this expression for t'$_{dFO4}$ (the delay without the 0.7 factor):

$t'_{dFO4} = R_{eff} \cdot C_D + R_{eff} \cdot 24C + R_{eff} \cdot 4C_G + 3R \cdot 12C + 3R \cdot 4C_G$

This expression we can simplify further using $C_D = C_G$ and rearranging:

$t'_{dFO4} = 5\,R_{eff}C_G + 24\,R_{eff}C + 36RC + 12RC_G$

We have that the FO4 delay is $0.7t'_{dFO4}$. We identify $\tau' = R_{eff}C_G$, which we know is a process constant; we then see that the first term in corresponds to the usual FO4 delay. (5 p)

b) To find the optimal R_{eff} we need to set the derivative of t'_{dFO4} with respect to R_{eff} equal to 0 and solve for R_{eff}. But before we do so, we must eliminate C_G from t'_{dFO4} since C_G and R_{eff} are connected through the relation $\tau' = R_{eff}C_G$, where $\tau'$is a constant. So we rewrite t'_{dFO4} as:

$t'_{dFO4} = 5\,\tau' + 24\,R_{eff}C + 36RC + 12R \cdot \frac{\tau'}{R_{eff}}$

It is now clear that there are two terms that depend on R_{eff}. The derivate is:

$$\frac{dt'_{dFO4}}{dR_{eff}} = 24C - \frac{12R\tau'}{{R_{eff}}^2}$$

When we set the derivative equal to 0 and solve for R_{eff} we find:

$$R_{eff} = \sqrt{\frac{R\tau'}{2C}}$$ (5 p)

6. **Prefix Adders** The solution to both task a) and b) is shown in the figure below. Ignore the buffers that are drawn in the figure.