Solutions

Solutions for *CMOS VLSI Design* 4th Edition. Last updated 26 March 2010.

Chapter 1

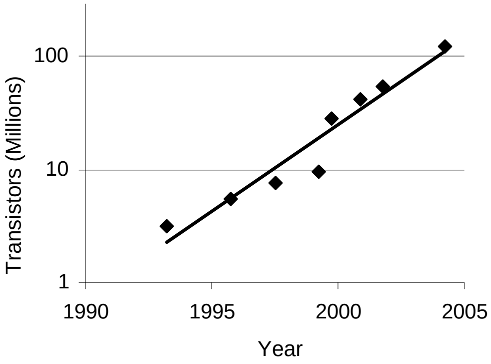
1.1 Starting with 100,000,000 transistors in 2004 and doubling every 26 months for 12 years gives $10^8 \bullet 2^{\left(\frac{12 \cdot 12}{26}\right)} \approx 4.6\text{B}$ transistors.

1.2 See Figure 1.4 for data through 2009. Some data includes:

Table 1: Microprocessor transistor counts

Date	CPU	Transistors (millions)
3/22/93	Pentium	3.1
10/1/95	Pentium Pro	5.5
5/7/97	Pentium II	7.5
2/26/99	Pentium III	9.5
10/25/99	Pentium III	28
11/20/00	Pentium 4	42
8/27/01	Pentium 4	55
2/2/04	Pentium 4 HT	125

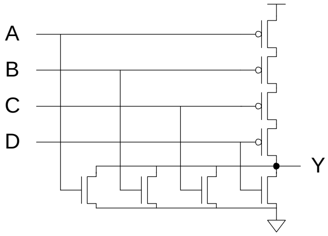
The transistor counts double approximately every 24 months.



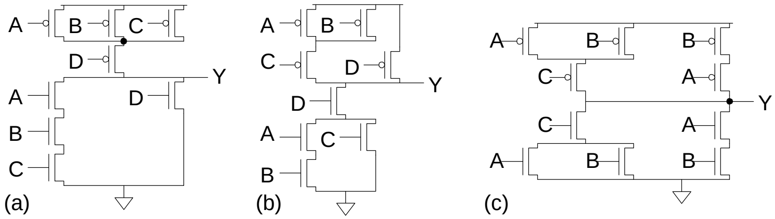
1.3 Let your imagination soar!

1.4 Don’t be a jerk! Your answers should vary.

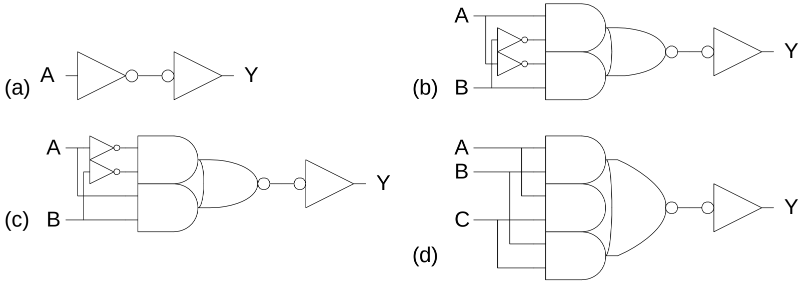
1.5



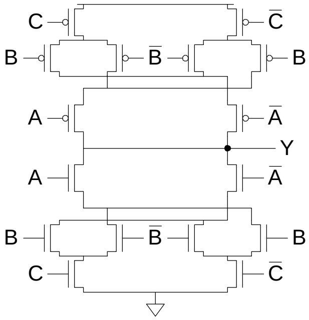
1.6



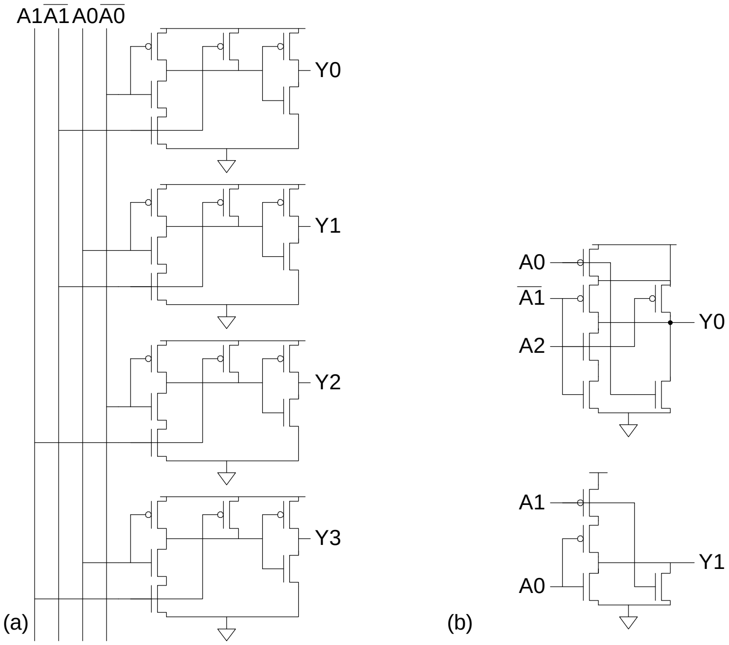
1.7


(a)
A
Y
(b)
A
B
Y
(c)
A
B
Y
(d)
A
B
C
Y

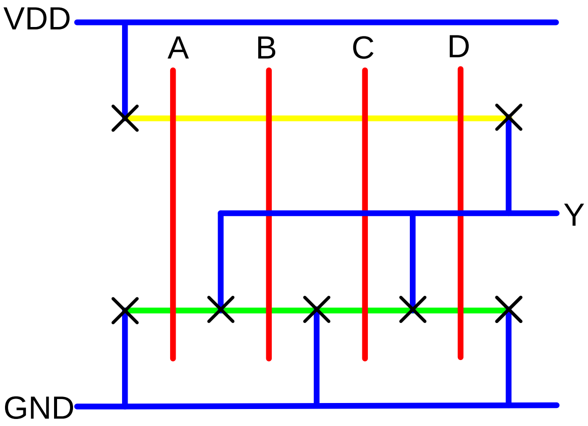
1.8


C
$\overline{C}$
B
$\overline{B}$
B
A
$\overline{A}$
Y
A
$\overline{A}$
B
$\overline{B}$
B
C
$\overline{C}$

1.9

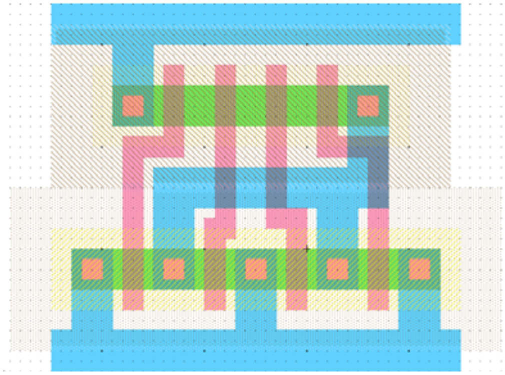


1.10

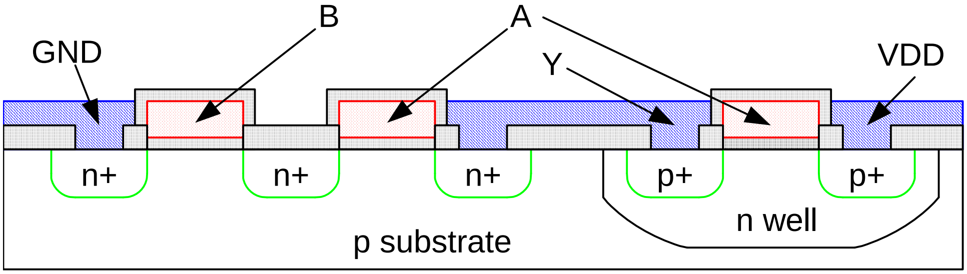


1.11 The minimum area is 5 tracks by 5 tracks (40 λ x 40 λ = 1600 λ^2).

1.12 The layout is 40 λ x 40 λ if minimum separation to adjacent metal is considered, exactly as the track count estimated.

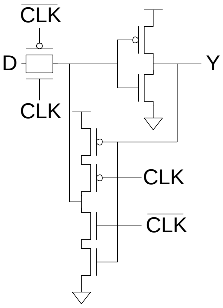


1.13

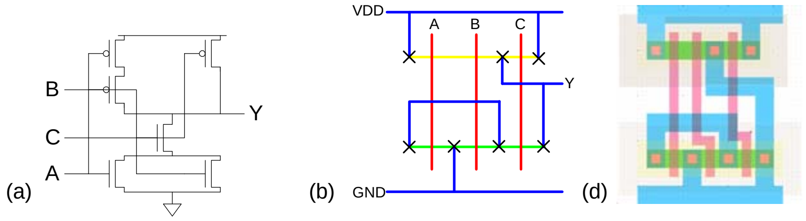


1.14 6 tracks wide by 6 tracks tall, or 2304 λ^2.

1.15 This latch is nearly identical save that the inverter and transmission gate feedback has been replaced by a tristate feedaback gate.



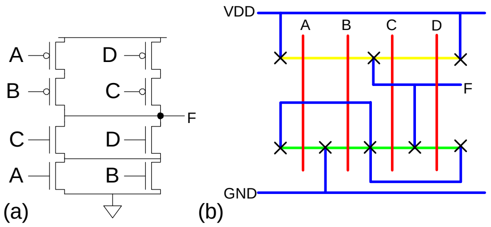
1.16



(c) 4 x 6 tracks = 32 λ x 48 λ = 1536 λ^2.

(e) The layout size matches the stick diagram.

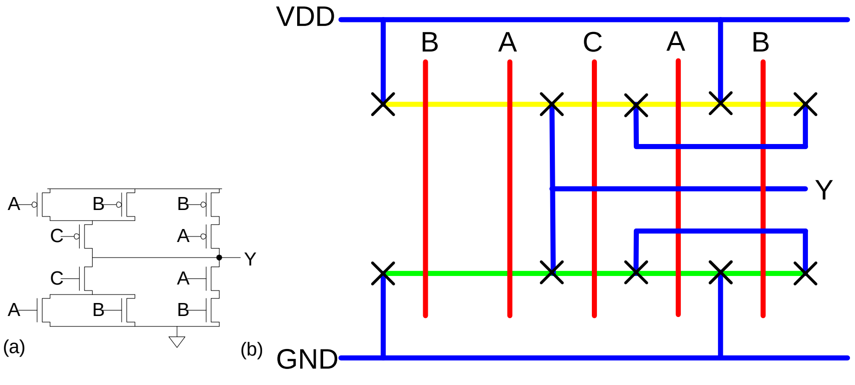
1.17



(c) 5 x 6 tracks = 40 λ x 48 λ = 1920 λ^2. (with a bit of care)

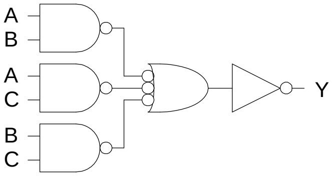
(d-e) The layout should be similar to the stick diagram.

1.18

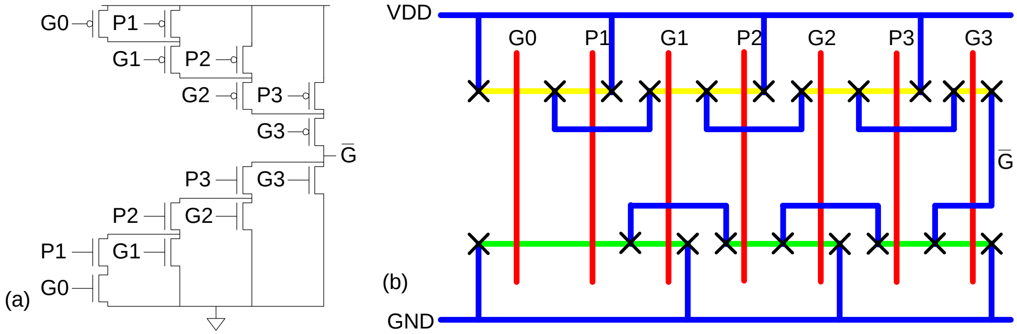


(c) 6 tracks wide x 7 tracks high = (48 x 56) = 2688 λ^2.

1.19 20 transistors, vs. 10 in 1.16(a).



1.20



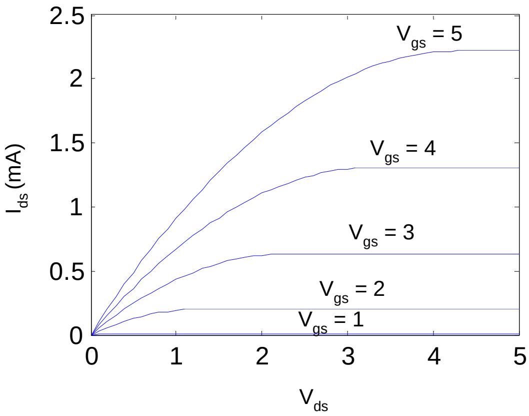
(c) The area of this stick diagram is 11 x 6 tracks = 4224 λ^2 if the polysilicon can be bent.

1.21 The Electric lab solutions are available to instructors on the web. The Cadence labs include walking you through the steps.

Chapter 2

2.1

$$\beta = \mu C_{ox}\frac{W}{L} = (350)\left(\frac{3.9 \bullet 8.85\cdot 10^{-14}}{100\cdot 10^{-8}}\right)\left(\frac{W}{L}\right) = 120\frac{W}{L}\mu A/V^{2}$$



2.2 In (a), the transistor sees V_{gs} = V_{DD} and V_{ds} = V_{DS}. The current is

$$I_{DS1} = \frac{\beta}{2}\left(V_{DD} - V_t - \frac{V_{DS}}{2}\right)V_{DS}$$

In (b), the bottom transistor sees V_{gs} = V_{DD} and V_{ds} = V_1. The top transistor sees V_{gs} = V_{DD} - V_1 and V_{ds} = V_{DS} - V_1. The currents are

$$I_{DS2} = \beta\left(V_{DD} - V_t - \frac{V_1}{2}\right)V_1 = \beta\left((V_{DD} - V_1) - V_t - \frac{(V_{DS} - V_1)}{2}\right)(V_{DS} - V_1)$$

Solving for V_1, we find

$$V_1 = (V_{DD} - V_t) - \sqrt{(V_{DD} - V_t)^2 - \left(V_{DD} - V_t - \frac{V_{DS}}{2}\right)V_{DS}}$$

Substituting V_1 indo the I_{DS2} equation and simplifying gives I_{DS1} = I_{DS2}.

2.3 The body effect does not change (a) because V_{sb} = 0. The body effect raises the threshold of the top transistor in (b) because V_{sb} > 0. This lowers the current through the series transistors, so I_{DS1} > I_{DS2}.

2.4 $C_{\text{permicron}}$ = $\varepsilon L/t_{ox}$ = 3.9 * 8.85e-14 F/cm * 90e-7 cm / 16e-4 µm= 1.94 fF/µm.

2.5 The minimum size diffusion contact is 4 x 5 λ, or 1.2 x 1.5 μm. The area is 1.8 μm^2 and perimeter is 5.4 μm. Hence the total capacitance is

$$C_{db}(0V) = (1.8)(0.42) + (5.4)(0.33) = 2.54\text{fF}$$

At a drain voltage of VDD, the capacitance reduces to

$$C_{db}(5V) = (1.8)(0.42)\left(1 + \frac{5}{0.98}\right)^{-0.44} + (5.4)(0.33)\left(1 + \frac{5}{0.98}\right)^{-0.12} = 1.78\text{fF}$$

2.6 Set the two parts of EQ (2.26) equal at $V_{ds} = V_{dsat.}$ Assume that EQ (2.27) is true and substitute it into (2.26) for $V_{dsat.}$, then simplify.

2.7 The new threshold voltage is found as

$$\phi_s = 2(0.026)\ln\frac{2\bullet 10^{17}}{1.45\bullet 10^{10}} = 0.85V$$

$$\gamma = \frac{100\bullet 10^{-8}}{3.9\bullet 8.85\bullet 10^{-14}}\sqrt{2\left(1.6\bullet 10^{-19}\right)\left(11.7\bullet 8.85\bullet 10^{-14}\right)\left(2\bullet 10^{17}\right)} = 0.75V^{1/2}$$

$$V_t = 0.7 + \gamma\left(\sqrt{\phi_s + 4} - \sqrt{\phi_s}\right) = 1.66V$$

The threshold increases by 0.96 V.

2.8 No. Any number of transistors may be placed in series, although the delay increases with the square of the number of series transistors.

2.9 The threshold is increased by applying a negative body voltage so $V_{sb} > 0$.

2.10 (a) $(1.2 - 0.3)^2 / (1.2 - 0.4)^2 = 1.26$ (26%)

(b) $$\frac{e^{\frac{-0.3}{1.4\,\bullet\,0.026}}}{e^{\frac{-0.4}{1.4\,\bullet\,0.026}}} = 15.6$$

(c) $v_T = kT/q = 34$ mV; $$\frac{e^{\frac{-0.3}{1.4\,\bullet\,0.034}}}{e^{\frac{-0.4}{1.4\,\bullet\,0.034}}} = 8.2$$; note, however, that the total leakage will normally be higher for both threshold voltages at high temperature.

2.11 The nMOS will be OFF and will see $V_{ds} = V_{DD}$, so its leakage is

$$I_{leak} = I_{dsn} = \beta v_T^2 e^{1.8} e^{\frac{-V_t}{nv_T}} = 69\,pA$$

2.12 If the voltage at the intermediate node is *x*, by KCL:

$$I_{leak} = \beta v_T^2 e^{1.8} e^{\frac{-V_t}{nv_T}} \left(1 - e^{\frac{-x}{v_T}}\right) = \beta v_T^2 e^{1.8} e^{\frac{-(x + V_t)}{nv_T}}$$

Now, solve for *x* using *n* = 1:

$$\left(1 - e^{\frac{-x}{v_T}}\right) = e^{\frac{-x}{nv_T}} \rightarrow x = v_T \ln 2$$

Substituting, the current is exactly half that of the inverter.

2.13 Assume V_{DD} = 1.8 V. For a single transistor with *n* = 1.4,

$$I_{leak} = I_{dsn} = \beta v_T^2 e^{1.8} e^{\frac{-V_t + \eta V_{DD}}{nv_T}} = 499\,pA$$

For two transistors in series, the intermediate voltage *x* and leakage current are found as:

$$I_{leak} = \beta v_T^2 e^{1.8} e^{\frac{-V_t + \eta x}{nv_T}} \left(1 - e^{\frac{-x}{v_T}}\right) = \beta v_T^2 e^{1.8} e^{\frac{\eta(V_{DD} - x) - V_t - x}{nv_T}}$$

$$e^{\frac{-V_t + \eta x}{nv_T}} \left(1 - e^{\frac{-x}{v_T}}\right) = e^{\frac{\eta(V_{DD} - x) - V_t - x}{nv_T}}$$

$$x = 69 \text{ mV};\ I_{leak} = 69 \text{ pA}$$

In summary, accounting for DIBL leads to more overall leakage in both cases. However, the leakage through series transistors is much less than half of that through a single transistor because the bottom transistor sees a small Vds and much less DIBL. This is called the *stack effect*.

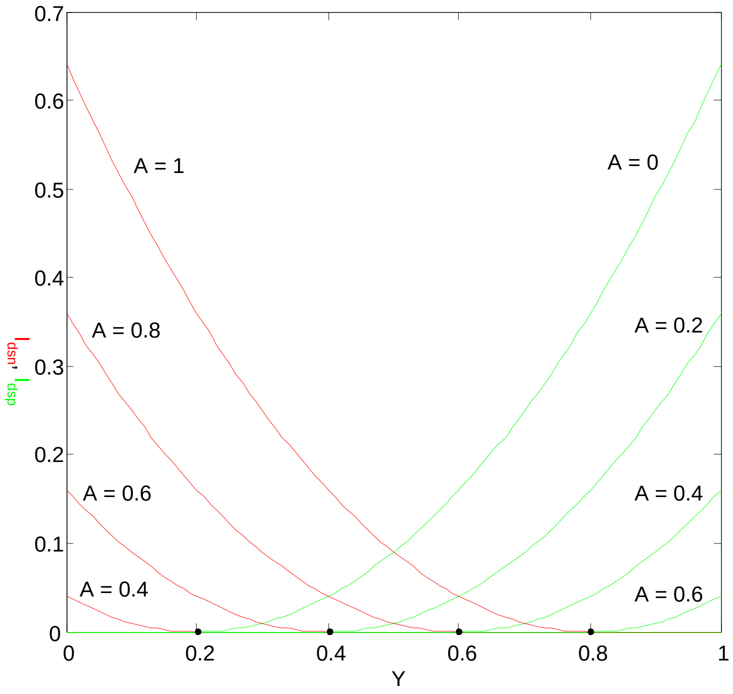
For *n* = 1.0, the leakage currents through a single transistor and pair of transistors are 13.5 pA and 0.9 pA, respectively.

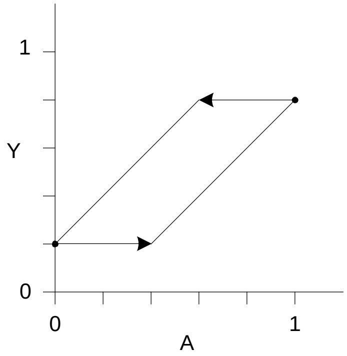
2.14

$$I_{dsn} = \begin{cases} 0 & (A-Y) < V_t \\ \frac{\beta}{2}\left(A-Y-V_t\right)^2 & (A-Y) > V_t \end{cases}$$

$$I_{dsp} = \begin{cases} 0 & (A-Y) > -V_t \\ \frac{\beta}{2}\left(A-Y+V_t\right)^2 & (A-Y) < -V_t \end{cases}$$

The graph below is normalized to $V_{DD} = 1$, $\beta = 2$, and $V_t = V_{DD}/5$. This is a bad buffer because the output does not swing rail-to-rail and because it exhibits hysteresis.





2.15 $V_{IL} = 0.3$; $V_{IH} = 1.05$; $V_{OL} = 0.15$; $V_{OH} = 1.2$; $NM_H = 0.15$; $NM_L = 0.15$

2.16 Set the currents through the transistors equal and solve the nasty quadratic for V_{out}.

In region B, the nMOS is saturated and pMOS is linear:

$$\frac{\beta}{2}\left(V_{\text{in}}-V_t\right)^2=\beta\left(\left(V_{\text{in}}-V_{DD}\right)-\frac{\left(V_{\text{out}}-V_{DD}\right)}{2}+V_t\right)\left(V_{\text{out}}-V_{DD}\right)$$

$$V_{\text{out}}=\left(V_{\text{in}}+V_t\right)+\sqrt{\left(V_{\text{in}}+V_t\right)^2-\left(V_{\text{in}}-V_t\right)^2+V_{DD}\left(V_{DD}-2V_{\text{in}}-2V_t\right)}$$

In region D, the nMOS is linear and the pMOS is saturated:

$$\frac{\beta}{2}\left(V_{\text{in}}-V_{DD}+V_t\right)^2=\beta\left(V_{\text{in}}-V_t-\frac{V_{\text{out}}}{2}\right)\left(V_{\text{out}}\right)$$

$$V_{\text{out}}=\left(V_{\text{in}}-V_t\right)-\sqrt{\left(V_{\text{in}}-V_t\right)^2-\left(V_{DD}-V_{\text{in}}-V_t\right)^2}$$

2.17 Either take the grungy derivative for the unity gain point or solve numerically for V_{IL} = 0.46 V, V_{IH} = 0.54 V, V_{OL} = 0.04 V, V_{OH} = 0.96 V, NM_H = NM_L = 0.42 V.

2.18 The switching point where both transistors are saturated (region C) is found by solving for equal currents:

$$\frac{\beta_n}{2}\left(V_{\text{in}}-V_{tn}\right)^2=\frac{\beta_p}{2}\left(V_{\text{in}}-V_{DD}-V_{tp}\right)^2$$

$$V_{\text{in}}{}^2\left(\beta_n-\beta_p\right)+V_{\text{in}}\left(-2\beta_nV_{tn}+2\beta_p\left(V_{DD}+V_{tp}\right)\right)+\left(\beta_nV_{tn}{}^2-\beta_p\left(V_{DD}+V_{tp}\right)^2\right)=0$$

$$V_{\text{in}}=\frac{\beta_nV_{tn}-\beta_p\left(V_{DD}+V_{tp}\right)+\left(V_{DD}+V_{tp}-V_{tn}\right)\sqrt{\beta_n\beta_p}}{\beta_n-\beta_p}$$

$$=\frac{V_{DD}+V_{tp}+\sqrt{\frac{\beta_n}{\beta_p}}V_{tn}}{1+\sqrt{\frac{\beta_n}{\beta_p}}}$$

The output voltage in region B is found by solving

$$\frac{\beta_n}{2}\left(V_{\text{in}}-V_{tn}\right)^2=\beta_p\left(\left(V_{\text{in}}-V_{DD}\right)-\frac{\left(V_{\text{out}}-V_{DD}\right)}{2}-V_{tp}\right)\left(V_{\text{out}}-V_{DD}\right)$$

$$V_{\text{out}}=\left(V_{\text{in}}-V_{tp}\right)+\sqrt{\left(V_{\text{in}}-V_{tp}\right)^2-\tfrac{\beta_n}{\beta_p}\left(V_{\text{in}}-V_{tn}\right)^2+V_{DD}\left(V_{DD}-2V_{\text{in}}+2V_{tp}\right)}$$

and the output voltage in region D is

$$\frac{\beta_p}{2}\left(V_{\text{in}} - V_{DD} - V_{tp}\right)^2 = \beta_n\left(V_{\text{in}} - V_{tn} - \frac{V_{\text{out}}}{2}\right)\left(V_{\text{out}}\right)$$

$$V_{\text{out}} = \left(V_{\text{in}} - V_{tn}\right) - \sqrt{\left(V_{\text{in}} - V_{tn}\right)^2 - \frac{\beta_p}{\beta_n}\left(V_{DD} - V_{\text{in}} + V_{tp}\right)^2}$$

2.19 Take derivatives or solve numerically for the unity gain points: V_{IL} = 0.43 V, V_{IH} = 0.50 V, V_{OL} = 0.04 V, V_{OH} = 0.97 V, NM_H = 0.39, NM_L = 0.47 V.

2.20 (a) 0; (b) $2|V_{tp}|$; (c) $|V_{tp}|$; (d) V_{DD} - V_{tn}

2.21 (a) 0; (b) 0.6; (c) 0.8; (d) 0.8

Chapter 3

3.1 First, the cost per wafer for each step and scan. 248nm – number of wafers for four years = 4*365*24*80 = 2,803,200. 193nm = 4*365*24*20 = 700,800. The cost per wafer is the (equipment cost)/(number of wafers) which is for 248nm $10M/2,803,200 = $3.56 and for 193nm is $40M/700,800 = $57.08. For a run through the equipment 10 times per completed wafer is $35.60 and $570.77 respectively.

Now for gross die per wafer. For a 300mm diameter wafer the area is roughly 70,650 mm^2 (π*(r^2/A – r/(sqrt(2*A))). For a 50mm^2 die in 90nm, there are 1366 gross die per wafer. Now for the tricky part (which was unspecified in the question and could cause confusion). What is the area of the 50nm chip? The area of the core will shrink by (90/50)^2 = .3086. The best case is if the whole die shrinks by this factor. The shrunk die size is 50*.3086 = 15.43mm^2. This yields 4495 gross die per wafer.

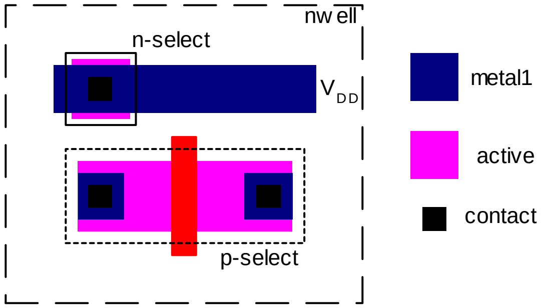
The cost per chip is $35.60/1413 = $0.026 and $570.77/4578 = $0.127 respectively for 90nm and 50nm. So roughly speaking, it costs $0.10 per chip more at the 50nm node.

Obviously, there can be variations here. Another way of estimating the reduced die size is to estimate the pad area (if it's not specified as in this exercise) and take that out or the equation for the shrunk die size. A 50mm^2 chip is roughly 7mm on a side (assuming a square die). The I/O pad ring can be (approximately) between 0.5 and 1 mm per side. So the core area might range from 25mm^2 to 36mm^2. When shrunk, this core area might vary from 7.7 to 11.1mm^2 (2.77 and 3.33mm on a side respectively). Adding the pads back in (they don't scale very much), we get die sizes of 4.77 and 4.33 mm on a side. This yield possible areas of 18.7 to 22.8 mm^2, which in turn yields a cost of processing on the stepper of between $0.155 and $0.189. This is a rather more pessimistic (but realistic) value.

3.2 The answer to this question is based on the difference in dielectric constant between SiO_2 (k=3.9) and HfO_2 (k=20). The oxide thickness would be 2 nm*(20/3.9) = 10.26 nm.

3.3 Polycide – only gate electrode treated with a refractory metal. Salicide – gate and source and drain are treated. The salicide should have higher performance as the resistance of source and drain regions should be lower. (Especially true at RF and for analog functions).

3.4 The pMOS transistor and well contact will be surrounded by the n-well. The pMOS transistor will have active surrounded by p-select while the well contact will have active surrounded by n-select. Contact and metal will be located in the well contact and at the source and drain of the pMOS transistor (and possibly the gate connection).



3.5) Siliver has better conductivity than copper, but it can migrate into the silicon and wreck the transistors.

3.6 Metal1 has a 2x2 contact with an overlap of 1 on each side, requiring a 4-λ width. The metal spacing is 3 λ, so the pitch is 7 λ if contacts can meet head to head.

A denser wiring strategy is to offset contacts. The pitch is reduced by half the distance that the contact extends beyond minimum metal width, thus giving a 6.5 λ pitch.

3.7 The uncontacted transistor pitch is = 2*half the minimum poly width + the poly space over active = 2*0.5*2 + 3 = 5 λ. The contacted pitch is = 2*half the minimum poly width + 2 * poly to contact spacing + contact width = 2*0.5*2 + 2*2 + 2 = 8 λ.

The reason for this problem is to show that there is an appreciable difference in gate spacing (and therefore source/drain parasitics) between contacted source and drains and the case where you can eliminate the contact (e.g. in NAND structures). In the main this may not be important but if you were trying too eke out the maximum performance you might pay attention to this. In some advanced processes, the spacing between polysilicon increases to the point that the uncontacted pitch may be the same as the contacted pitch.

3.8 The vertical pitch is divided into three basic segments. First, we have to determine the spacing of the n-transistor to the GND bus. The next segment is defined by the n-transistor to p-transistor spacing. Finally, the p-transistor to V_{DD} bus spacing needs to be determined. (all spacings are center to center).

N-transistor to GND bus

First let us assume minimum metal1 widths. Next, the width of a metal contact is equal to the contact width plus twice the overlap of the metal over the contact = 2 + 2*1 = 4 λ. The minimum width of a transistor is the contact width plus 2*active overlap of contact = 2 + 2*1 = 4 λ (actually the same as a metal1 contact). So the spacing of the n-transistor to the GND bus will be half the GND bus width plus metal spacing plus half of the metal contact width = 0.5*3 + 3 + 0.5*4 = 6.5 λ .

N-transistor to P-transistor spacing

There are two cases: with a polysilicon contact to the gate and without. With the metal-to-polysilicon contact, the spacing will probably be half of the n-transistor width plus the metal space plus the polysilicon contact width plus the metal space plus half the p-transistor width = 0.5*4 + 3 + 4 + 3 + 0.5*4 = 14 λ. The spacing without a contact is half the n-transistor width plus n-active to p-active spacing plus half the p-transistor width = 0.5*4 + 4 + 0.5*4 = 8 λ. However, the n-well must surround the pMOS transistor by 6 and be 6 away from the nMOS. This sets a minimum pitch of 0.5*4 + 6 + 6 + 0.5*4 = 16 for both cases.

P-transistor to V_{DD} bus

By symmetry, this is also 6.5 λ.

Summary

The total pitch is 2*6.5 + 16 = 29 λ. The total height of the inverter is 35 λ including the complete supply lines and spacing to an adjacent cell. In the case where the V_{DD} and GND busses are not minimum pitch, the vertical pitch and cell height increase appropriately.

In this inverter the substrate connections have not been included. They could be included in the horizontal plane so that the vertical pitch is not affected. If they are included under the metal power busses, the spacing on the transistors to the power busses may be altered. Normally, this is what is done the power bus can be sized up to account for the spacing. This helps power distribution and does not affect the pitch much.

In an SOI process, if the n to p spacing is 2 λ rather than 12 λ, the pitches are 2*6.5 + 14 = 27 λ and 2 * 6.5 + 6 = 19 λ respectively for interior poly connection and not.

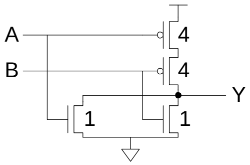
In older standard cell families (two and three level metal processes), the polysilicon contact was often eliminated and the contact to the gate was made above or below the cell in the routing channels. With modern standard cells, all connections to the cells are normally completed within the cell (up to metal2).

3.9 A fuse is a necked down segment of metal (Figure 3.24) that is designed to blow at a certain current density. We would normally set the width of the fuse to the minimum metal width – is this case 0.5 µm. At this width, the maximum current density is 500 µA. At a programming current of 10 times this – 5mA, the fuse should blow reliably. The “fat” conductor connecting to the fuse has to be at least 2.5 µm to carry the fuse current. Actually, the complete resistance from the programming source to the fuse has to be calculated to ensure that the fuse is the where the maximum voltage drop occurs.

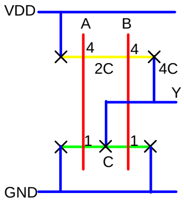
The length of the fuse segment should be between 1 and 2 µm. Why? It’s a guess – in a real design, this would be prototyped at various lengths and the reliability of blowing the fuse could be determined for different lengths and different fuse currents. The fabrication vendor may be able to provide process-specific guidelines. One needs enough length to prevent any sputtered metal from bridging the thicker conductors.

Chapter 4

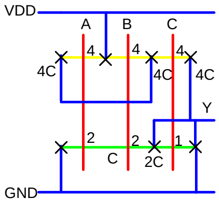
4.1 The rising delay is (R/2)*8C + R*(6C+5hC) = (10+5h)RC if both of the series pMOS transistors have their own contacted diffusion at the intermediate node. More realisitically, the diffusion will be shared, reducing the delay to (R/2)*4C + R*(6C+5hC) = (8+5h)RC. Neglecting the diffusion capacitance not on the path from Y to GND, the falling delay is R*(6C+5hC) = (6+5h)RC.



4.2 The rising delay is (R/2)*2C + R*(5C+5hC) = (6+5h)RC and the falling delay is R*(5C+5hC) = (5+5h)RC.



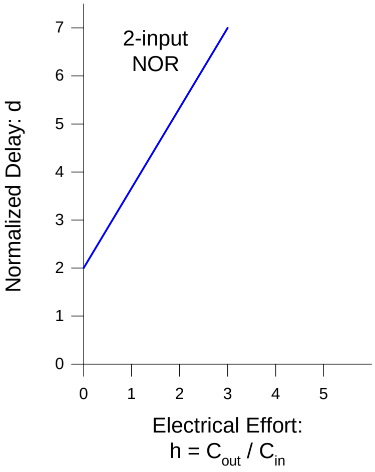
4.3 The rising delay is (R/2)*(8C) + (R)*(4C + 2C) = 10 RC and the falling delay is (R/2)*(C) + R(2C + 4C) = 6.5 RC. Note that these are only the parasitic delays; a real gate would have additional effort delay.



4.4 The output node has 3*n*C. Each internal node has 2*n*C. The resistance through each pMOS is R/*n*. Hence, the propagation delay is

$$t_{pd} = R(3nC) + \sum_{i=1}^{n-1}\left(\frac{iR}{n}\right)(2nC) = \left(n^2 + 2n\right)RC$$

4.5 The slope (logical effort) is 5/3 rather than 4/3. The y-intercept (parasitic delay) is identical, at 2.

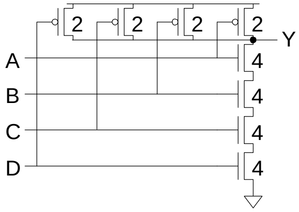


4.6 C_{in} = 12 units. $g = 1$. $p = p_{\text{inv}}$. Changing the size affects the capacitance but not logical effort or parasitic delay.

4.7 The delay can be improved because each stage should have equal effort and that effort should be about 4. This design has imbalanced delays and excessive efforts. The path effort is $F = 12 * 6 * 9 = 648$. The best number of stages is 4 or 5. One way to speed the circuit up is to add a buffer (two inverters) at the end. The gates should be resized to bear efforts of $f = 648^{1/5} = 3.65$ each. Now the effort delay is only $D_F = 5f = 18.25$, as compared to 12 + 6 + 9 = 27. The parasitic delay increases by $2p_{\text{inv}}$, but this is still a substantial speedup.

4.8 (a) 4 units. (b) (3/4 units).

4.9 g = 6/3 is the ratio of the input capacitance (4+2) to that of a unit inverter (2 + 1).



4.10 (a) should be faster than (b) because the NAND has the same parasitic delay but lower logical effort than the NOR. In each design, H = 6, B = 1, P = 1 + 2 = 3. For (a), G = (4/3) * 1 = (4/3). $F = GBH$ = 8. $f = 8^{1/2}$ = 2.8. $D = 2f + P$ = 8.6 τ. x = 6C * 1 / f = 2.14C. For (b), G = 1 *(5/3). $F = GBH$ = 10. $f = 10^{1/2}$ = 3.2. $D = 2f + P$ = 9.3 τ. x = 6C * (5/3) / f = 3.16C.

4.11 $D = N(GH)^{1/N} + P$. Compare in a spreadsheet. Design (b) is fastest for H = 1 or 5. Design (d) is fastest for H = 20 because it has a lower logical effort and more stages to drive the large path effort. (c) is always worse than (b) because it has greater logical effort, all else being equal.

Comparison of 6-input AND gates

Design	*G*	*P*	*N*	*D* (*H*=1)	*D* (*H*=5)	*D* (*H*=20)
(a)	**8/3 * 1**	**6 + 1**	**2**	**10.3**	**14.3**	**21.6**
(b)	**5/3 * 5/3**	**3 + 2**	**2**	**8.3**	**12.5**	**19.9**
(c)	**4/3 * 7/3**	**2 + 3**	**2**	**8.5**	**12.9**	**20.8**
(d)	**5/3 * 1 * 4/3 * 1**	**3 + 1 + 2 + 1**	**4**	**11.8**	**14.3**	**17.3**

4.12 H = (64 * 3) / 10 = 19.2. B = 32 / 2 = 16. Compare several designs in a spreadsheet. The five-stage design is fastest, with a path effort of $F = GBH$ = 683 and stage effort of $f = F^{1/5}$ = 3.69. The gate sizes from end to start are: 192 * 1 / 3.69 = 52; 52 * (4/3) / 3.69 = 18.8; 18.8 * 1 / 3.69 = 5.1; 5.1 * (5/3) / 3.69 = 2.3; 2.3 * 1 / 3.69 = 0.625.

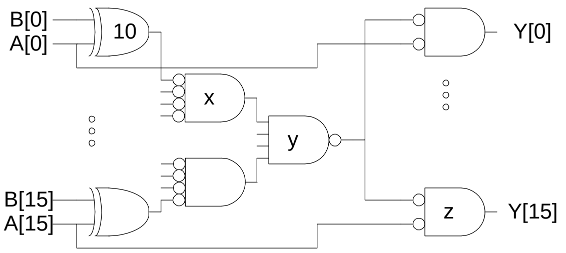
Comparison of decoders

Design	G	P	N	D
NAND5 + INV	**7/3**	**6**	**2**	**59.5**

Comparison of decoders

Design	G	P	N	D
INV + NAND5 + INV	**7/3**	**7**	**3**	**33.8**
NAND3 + INV + NAND2 + INV	**20/9**	**7**	**4**	**27.4**
INV + NAND3 + INV + NAND2 + INV	**20/9**	**8**	**5**	**26.4**
NAND3 + INV + NAND2 + INV + INV + INV	**20/9**	**9**	**6**	**26.8**
NAND2 + INV + NAND2 + INV + NAND2 + INV	**64/27**	**9**	**6**	**27.0**

4.13 One reasonable design consists of XNOR functions to check bitwise equality, a 16-input AND to check equality of the input words, and an AND gate to choose Y or 0. Assuming an XOR gate has $g = p = 4$, the circuit has $G = 4 * (9/3) * (6/3) * (5/3) = 40$. Neglecting the branch on A that could be buffered if necessary, the path has $B = 16$ driving the final ANDs. $H = 10/10 = 1$. $F = GBH = 640$. $N = 4$. $f = 5.03$, high but not unreasonable (perhaps a five stage design would be better). $P = 4 + 4 + 4 + 2 = 14$. $D = Nf + P = 34.12\ \tau = 6.8$ FO4 delays. $z = 10 * (5/3) / 5.03 = 3.3$; $y = 16 * z * (6/3) / 5.03 = 21.1$; $x = y * (9/3) / 5.03 = 12.6$.



4.14 t_{pd} = 76 ps, 72 ps, 67 ps, 70 ps for the XL, X1, X2, and X4 NAND2 gates, respectively. The XL gate has slightly higher parasitic delay because the wiring capacitance is a greater fraction of the total. It also has a slightly higher logical effort, possibly because stray wire capacitance is counted in the input capacitance and forms a greater fraction of the total C_{in}.

4.15 Using average values of the intrinsic delay and K_{load}, we find $d_{abs} = (0.029 + 4.55*C_{load})$ ns. Substituting $h = C_{load}/C_{in}$, this becomes $d_{abs} = (0.029 + 0.020h)$ ns. Normalizing by τ, $d = 1.65h + 2.42$. Thus the average logical effort is 1.65 and parasitic delay is 2.42.

4.16 X2: $g = 1.55$; $p = 2.14$. X4: $g = 1.56$; $p = 2.17$. The logical efforts are about 6% lower and parasitic delay 12% lower than in the X1 gate. Because of differing layout parasitics and slightly nonlinear I vs. W behavior, the parasitic delay and logical effort are not entirely independent of transistor size. However, the variations with size are small enough that the model is still useful.

4.17 $g = 1.47$, $p = 3.08$. The parasitic delay is substantially higher for the outer input (B) because it must discharge the internal parasitic capacitance. The logical effort is slightly lower for reasons discussed in Section 6.2.1.3.

4.18 Parasitic delay decreases relative to the estimates when sharing is considered, but increases when the internal diffusion and wire capacitance are considered.

4.19 NAND2: g = 5/4; NOR2: g = 7/4. The inverter has a 3:1 P/N ratio and 4 units of capacitance. The NAND has a 3:2 ratio and 5 units of capacitance, while the NOR has a 6:1 ratio and 7 units of capacitance.

4.20 NAND: $g = (\mu + k) / (\mu + 1)$; NOR: $g = (\mu k + 1) / (\mu + 1)$. As μ increases, NOR gates get worse compared to NAND gates because the series pMOS devices become more expensive.

4.21 $d = (4/3) * 3 + 2 = 6\ \tau$ = 1.2 FO4 inverter delays.

4.22 $d = (4/3) * 3 + 2 * 0.75 = 5.5\ \tau$ = for $p_{\text{inv}} = 0.75$; $d = 6.5\ \tau$ for $p_{\text{inv}} = 1.25$. The FO4 inverter delays are 4.75 and 5.25 τ, respectively. Hence, the delays are 1.16 and 1.24 FO4 delays, respectively, only a 4% change in normalized delay for a 25% change in parasitics. Hence, delay measured in FO4 delays is relatively insensitive to variations in parasitics from one process to another.

4.23 The adder delay is 6.6 FO4 inverter delays, or about 133 ps in the 65 nm process.

4.24 $F = (10\text{ pF} / 20\text{ fF}) = 500$. $N = \log_4 F = 4.5$. Use a chain of four inverters with a stage (5 would also work, but would produce the opposite polarity). $D = 4F^{1/4} + 4 = 22.9\ \tau$ = 4.58 FO4 delays.

4.25 If the first upper inverter has size x and the lower 100-x and the second upper inverter has the same stage effort as the first (to achieve least delay), the least delays are: $D = 2(300/x)^{1/2} + 2 = 300/(100-x) + 1$. Hence $x = 49.4$, $D = 6.9\ \tau$, and the sizes are 49.4 and 121.7 for the upper inverters and 50.6 for the lower inverter. Such circuits are called *forks* and are discussed in depth in [Sutherland99].

4.26 Let the sizes be x and y in the 2-stage path and C_1 - x in the 1-stage path.

$$D = 2\sqrt{\frac{C_2}{x}} + 2p_{\text{inv}} = \frac{C_2}{C_1 - x} + p_{\text{inv}}$$

This has a nasty third-order closed form solution, but can be solved numerically for x and thus D. $y = \sqrt{xC_2}$.

Chapter 5

5.1 $P = aCV^2f$ = 0.1 * (450e^{-12} * 70) * (0.9)^2 * 450e^6 = 1.08 W.

5.2 Dynamic power consumption will go down because it is quadratically dependent on V_{DD}. Static power will go up because subthreshold leakage is exponentially dependent on V_t.

5.3 Simplify using $V_{DD} >> v_T$:

$$I_1 = I_{ds0} e^{\frac{-V_t}{v_T}} \left[1 - e^{\frac{-V_{DD}}{v_T}} \right] \approx I_{ds0} e^{\frac{-V_t}{v_T}}$$

$$I_2 = I_{ds0} e^{\frac{-V_t}{v_T}} \left[1 - e^{\frac{-x}{v_T}} \right] = I_{ds0} e^{\frac{-V_t - x}{v_T}} \left[1 - e^{\frac{-V_{DD} + x}{v_T}} \right]$$

$$I_2 \approx I_1 \left[1 - e^{\frac{-x}{v_T}} \right] = I_1 e^{\frac{-x}{v_T}}$$

$$1 - e^{\frac{-x}{v_T}} = e^{\frac{-x}{v_T}} \Rightarrow e^{\frac{-x}{v_T}} = \tfrac{1}{2} \Rightarrow I_2 / I_1 = 1/2$$

5.4 The signal makes4 transitions in 10 cycles, so the activity factor is (1/2)(4/10) = 0.2.

5.5 A two-stage design will use the least energy because it has the smallest amount of switching hardware. The sizes are 1 and x. The delay is d = x + 64/x + 2. Solving for d = 20 gives x = 4.88.

5.6 Consider designs with 2, 3, and 4 stages. Choose sizes x, y, z to minimize energy (E) such that d = 30. The 3-stage design is best.

2 stage: d = x + 500/x + 2; E = 1 + x:Delay constraint can’t be met.

3 stage: d = x + y/x + 500/y + 3; E = 1 + x + y:

x = 5, y = 32.09, E=38.09

4 stage: d = x + y/x + z/y + 500/z + 4; E = 1 + x + y + z:

x = 2.15; y = 6.23; z = 31.43; E = 39.81

5.7 AND2: Y = 1 when A = 1 and B = 1

AND3: Y =1 when A, B, and C all are 1

OR2: Y = 1 unless A = 0 and B = 0

NAND2: Y = 1 unless A = 1 and B = 1

NOR2: Y = 1 when A = 0 and B = 0

XOR2: Y = 1 when A = 1 and B = 0 or when A = 0 and B = 1

5.8 This problem should state 5 FO4, not 4 FO4.

Consider a 2-stage design with four parallel 2-input NORs with drive x followed by a 4-input NAND with drive y. The inputs have an activity factor of (1/2)*(1/2) = 1/4. The output probability of the NOR2s is P = 1/4, so α = (1/4)*(3/4) = 3/16. The output probability for the NAND4 is 255/256, so α = (255/256)*(1/256) = 255/65536. The input capacitance of the NOR2 is 5/3 x and the input capacitance of the NAND4 is 6/3 y. To meet the input capacitance spec, x <= 3/5. The parasitic delay is 2 + 4 = 6. The total delay is d = 2y/x + 16/y + 6. 5 FO4 corresponds to a delay of 25 tau. The energy (including the output stage, normalized for voltage) is E = (1/4)*(5/3)x*8 + (3/16)*(6/3)y*4 + (255/65536)*16. Choose x and y to minimize E subject to d <= 25 and x <= 3/5. Using the Excel solver gives x = 0.42 and y = 1.21, for a normalized energy of E = 3.27 and delay of d = 25 = 5 FO4.

5.9 Gate leakage through an ON nMOS transistor is 6.3 nA and through an ON pMOS transistor is negligible. Subthreshold leakage through the nMOS transistors is 5.6 nA. Subthreshold leakage through a single pMOS transistor is 9.3 nA.

Table 2: NOR leakage

State (AB)	Isub	Igate	Itotal
00	5.6 * 2 (2 nMOS)	0	11.2
01	9.3 (pMOS)	6.3 (1 nMOS)	15.6
10	< 9.3 (pMOS with intermediate node at |Vt|)	6.3 (1 nMOS)	~ 12
11	<< 9.3 (stack effect with two OFF pMOS)	6.3 * 2 (2 nMOS)	~ 13

5.10 For a 2% delay increase, the supply should droop by less than about 2% of VDD (e.g. 20 mV @ 1.0 V). Thus the effective resistance must be R = 20 mV / 100 mA = 0.2 Ω. This requires a width of W = 2.5 kΩ * μm / 0.2 Ω = 12.5 mm.

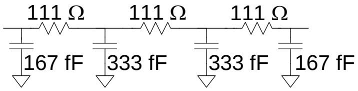
Chapter 6

6.1 The resistance per micron is (22 mΩ*μm)/((t-0.01 μm)*(w-0.02 μm)). Thus, the resistance of each layer is

Table 3:

Layer	t (μm)	w (μm)	R/μm
M9	7	17.5	0.00018
M8	0.720	0.400	0.082
M7	0.504	0.280	0.17
M6	0.324	0.180	0.44
M5	0.252	0.140	0.76
M4	0.216	0.120	1.07
M3/M2/M1	0.144	0.080	2.74

6.2 The wire width is 1.2 μm so the wire is 5000 μm/1.2 μm = 4167 squares in length. The total resistance is (0.08 Ω/sq)•(4167 sq) = 333 Ω. The total capacitance is (0.2 fF/μm)•(5000 μm) = 1 pF.



6.3 (This problem is inconsistent because it refers to a wire in a 0.6 μm process, but gives a transistor resistance characteristic of a 180 nm process. Use λ = 90 nm for transistor dimensions.) A unit inverter has a 4 λ = 0.36 μm wide nMOS transistor and an 8 λ = 0.72 μm wide pMOS transistor. Hence the unit inverter has an effective resistance of (2.5 kΩ•μm)/(0.36 μm) = 6.9 kΩ and a gate capacitance of (0.36 μm + 0.72 μm)•(2 fF/μm) = 2.2 fF. The Elmore delay is t_{pd} = (690 Ω)•(500 fF) + (690 Ω + 330 Ω)•(500 fF + 2.2 fF) = 0.86 ns.

6.4 $R = 0.05*l/W$; $C = l*C(W, S)$; $W + S$ = 1000 nm. $C(W, S)$ is found from Table 4.8. The C_{adj} term is doubled if the adjacent bits might switch in the opposite direciton. If neighbors are not switching, choose S = 320 nm and W = 680 nm. If neighbors are switching, choose S = 500 nm and W = 500 nm. In the first case, resistance dominates so the wide wire is fastest. In the second case, the coupling capacitance is exacerbated by the switching neighbors, so increasing the spacing is most useful.

6.5 Take the partial derivatives of (6.26) with respect to N and W and set them to 0 to minimize delay:

6.6 Write the delay equation using the Elmore delay model and differentiate with respect to W1, N, and k.

6.7 Compute the results with a spreadsheet:

$$D = \left(2+\sqrt{2}\right)\sqrt{R_w C_w \left(2.5k\Omega\right)\left(0.7+1.4fF\right)}$$

Characteristic velocity of repeated wires

Layer	Pitch (µm)	R_w	C_w	Delay (ps/mm)
1	**0.25**	**0.32**	**210**	**64**
1	**0.50**	**0.16**	**167**	**40**
2	**0.32**	**0.16**	**232**	**47**
2	**0.64**	**0.078**	**191**	**30**
4	**0.54**	**0.056**	**232**	**28**
4	**1.08**	**0.028**	**215**	**19**

6.8 See Morgenshtein09.

Chapter 7

7.1 The gate delay component scales as S^{-1} to 250 ps. The delay of a repeated wire of reduced thickness scales as $S^{-1/2}$ to 354 ps. The path delay scales to 604 ps, a 66% speedup.

7.2 An exponential fit to the data gives a life of 1.9 x 10^{10} * $10^{-4.49V}$ hours. 10 years is 87660 hours. Solving for V gives a maximum voltage of 1.2 V.

7.3 Solving for the CDF = 0.99999 gives 4.76 standard deviations.

7.4 Use three inputs and three outputs and three modules and three voters. Put one voter in front of each module to vote on the input to the module.

7.5 Solve $X_m = 3X_m^2 - 2X_m^3$ for $X_m = 0.5$.

7.6 A ring oscillator's period involves two trips around the ring, or 22 inverter delays. It has a mean of 22 * 10 = 220 ps and a standard deviation of sqrt(22) * 1 = 4.7 ps. According to Table 7.9, the slowest of the 100 ring oscillators has a mean delay of 220 + 2.50 * 4.7 = 231.8 ps and a standard deviation of 0.43 * 4.7 = 2.0 ps.

(a) 1 / 231.8 ps = 4.31 GHz.

(b) 97.7% yield corresponds to 2 sigma of variation, or a period of 231.8 + 2 * 2.0 = 235.8 ps. This corresponds to 4.24 GHz operation.

7.7 84% parametric yield corresponds to one standard deviation of systemic variation. The leakage power dominates the variability. If the channel length is 1 standard deviation (4 nm) short, the leakage increases by 4/40 = 10%, or 2 W. The threshold voltage decreases by 10 mV, causing leakage to increase by a factor of $e^{0.01 \ln 10/0.1}$ = 26%, or 5 W. Within-die channel length variation has a 3 * 2.5 = 7.5 mV effect on threshold voltage, so the threshold voltage has an random distribution with a standard deviation of sqrt(7.5^2 + 30^2) = 31 mV. This increases the expected value of leakage by a factor of $e^{(0.031 \ln 10/0.1)\hat{}2/2}$ = 1.29, or 6 W. The total power budget thus increases by 13 W to 73 W.

Chapter 8

8.1 t_{pd} = 107 ps.

```
* 51-fo5.sp
* created by Ted Jiang 9/20/2004
***********************************************************************
* Parameters and models
***********************************************************************
.param SUP=1.8
.param H=5
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post

***********************************************************************
* Subcircuits
***********************************************************************
.global vdd gnd

.subckt inv a y N=4 P=8
M1      y       a       gnd     gnd     NMOS    W='N'L=2
+ AS='N*5' PS='2*N+10' AD='N*5' PD='2*N+10'
M2      y       a       vdd     vdd     PMOS    W='P'L=2
+ AS='P*5' PS='2*P+10' AD='P*5' PD='2*P+10'
.ends

***********************************************************************
* Simulation netlist
***********************************************************************
Vdd     vdd     gnd     'SUPPLY'
```

```
Vin     a       gnd     PULSE   0 'SUPPLY' 0ps 100ps 100ps 500ps 1000ps
X1      a       b       inv              * shape input waveform
X2      b       c       inv     M='H'    * reshape input waveform
X3      c       d       inv     M='H**2' * device under test
X4      d       e       inv     M='H**3' * load
x5      e       f       inv     M='H**4' * load on load

***********************************************************************
* Stimulus
***********************************************************************
.tran 1ps 1000ps
.measure tpdr                        * rising propagation delay
+     TRIG v(c)       VAL='SUPPLY/2' FALL=1
+     TARG v(d)       VAL='SUPPLY/2' RISE=1
.measure tpdf                        * falling propagation delay
+     TRIG v(c)       VAL='SUPPLY/2' RISE=1
+     TARG v(d)       VAL='SUPPLY/2' FALL=1
.measure tpd param='(tpdr+tpdf)/2'   * average propagation delay
.end
```

8.2 t_{pd} = 84 ps, a 21% decrease.

```
* 52-vstep.sp
* created by Ted Jiang 9/20/2004
***********************************************************************
* Parameters and models
***********************************************************************
.param SUP=1.8
.param H=5
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post

***********************************************************************
* Subcircuits
***********************************************************************
.global vdd gnd

.subckt inv a y N=4 P=8
M1      y       a       gnd     gnd     NMOS    W='N'   L=2
+ AS='N*5' PS='2*N+10' AD='N*5' PD='2*N+10'
M2      y       a       vdd     vdd     PMOS    W='P'   L=2
+ AS='P*5' PS='2*P+10' AD='P*5' PD='2*P+10'
.ends

***********************************************************************
* Simulation netlist
***********************************************************************
```

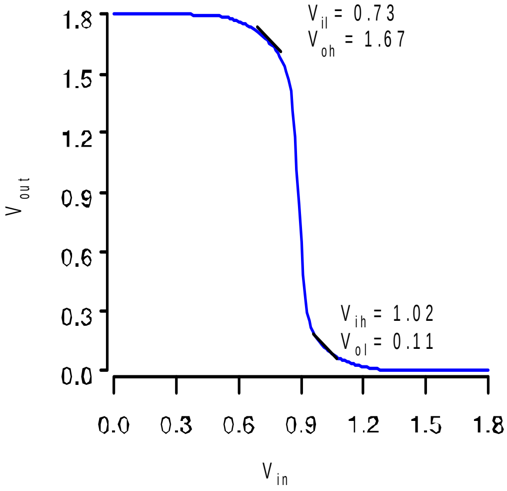
```
Vdd     vdd     gnd'    SUPPLY'
Vin     c       gnd     PULSE0 'SUPPLY' 0ps 100ps 100ps 500ps 1000ps
X3      c       d       inv     M='H**2' * device under test
X4      d       e       inv     M='H**3' * load
x5      e       f       inv     M='H**4' * load on load

************************************************************************
* Stimulus
************************************************************************
.tran 1ps 1000ps
.measure tpdr                         * rising propagation delay
+     TRIG v(c)VAL='SUPPLY/2' FALL=1
+     TARG v(d)  VAL='SUPPLY/2' RISE=1
.measure tpdf                         * falling propagation delay
+     TRIG v(c)  VAL='SUPPLY/2' RISE=1
+     TARG v(d)  VAL='SUPPLY/2' FALL=1
.measure tpd param='(tpdr+tpdf)/2'    * average propagation delay
.end
```

8.3 t_{pd} = 110 ps, a 3% increase.

```
* 53-noX5.sp
* Created by Ted Jiang 9/20/2004
************************************************************************
* Parameters and models
************************************************************************
.param SUP=1.8
.param H=5
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post

************************************************************************
* Subcircuits
************************************************************************
.global vdd gnd

.subckt inv a y N=4 P=8
M1      y       a       gnd     gnd     NMOS    W='N'   L=2
+ AS='N*5' PS='2*N+10' AD='N*5' PD='2*N+10'
M2      y       a       vdd     vdd     PMOS    W='P'   L=2
+ AS='P*5' PS='2*P+10' AD='P*5' PD='2*P+10'
.ends

************************************************************************
* Simulation netlist
************************************************************************
Vdd     vdd     gnd     'SUPPLY'
```

```
Vin     a       gnd     PULSE   0 'SUPPLY' 0ps 100ps 100ps 500ps 1000ps
X1      a       b       inv                   * shape input waveform
X2      b       c       inv     M='H'         * reshape input waveform
X3      c       d       inv     M='H**2'      * device under test
X4      d       e       inv     M='H**3'      * load


************************************************************************
* Stimulus
************************************************************************
.tran 1ps 1000ps
.measure tpdr                        * rising propagation delay
+     TRIG v(c)        VAL='SUPPLY/2' FALL=1
+     TARG v(d)        VAL='SUPPLY/2' RISE=1
.measure tpdf                        * falling propagation delay
+     TRIG v(c)        VAL='SUPPLY/2' RISE=1
+     TARG v(d)        VAL='SUPPLY/2' FALL=1
.measure tpd param='(tpdr+tpdf)/2'   * average propagation delay
.end
```

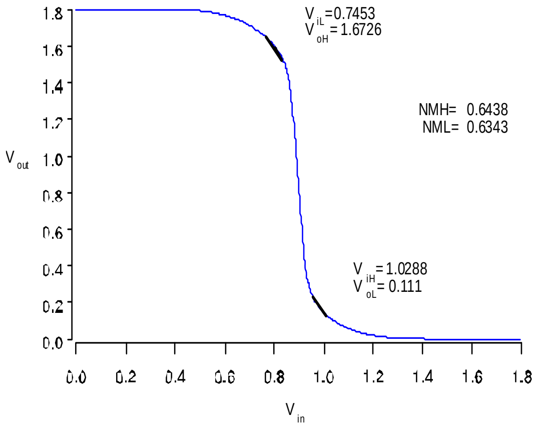
8.4 V_{IH} = 1.02 V; V_{IL} = 0.73 V; V_{OH} = 1.67 V; V_{OL} = 0.11 V. NM_H = 0.65 V (36% of V_{DD}). NM_L = 0.62 V (35% of V_{DD}).



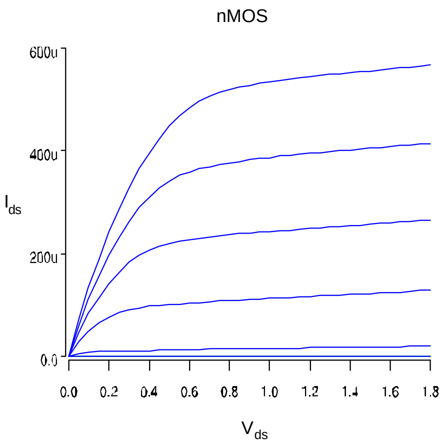
```
* 54-31ratio.sp
* created by Ted Jiang 10/6/2004
************************************************************************
* Parameters and models
************************************************************************
.param SUP=1.8
.option scale=90n
```

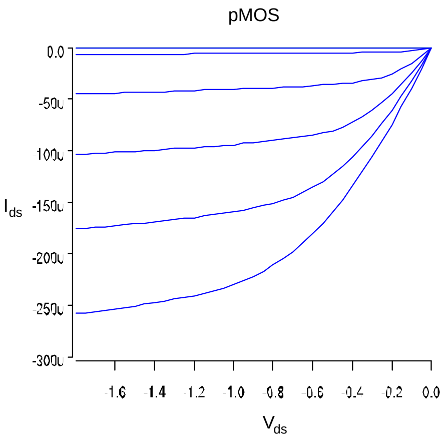
```
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post

************************************************************************
* Simulation netlist
************************************************************************
Vdd      vdd      gnd      'SUPPLY'
Vin      a        gnd      0
M1       y        a        gnd      gnd      NMOS     W=4      L=2
M2       y        a        vdd      vdd      PMOS     W=12     L=2

************************************************************************
* Stimulus
************************************************************************
.dc Vin 0 1.8 0.01
.end
```

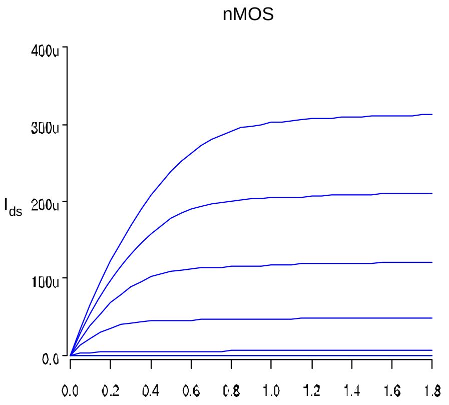
8.5 The best P/N ratio can be found by sweeping the ratio, generating the DC transfer curve, and measuring the input and output voltage levels and noise margins. A ratio of 3.2 / 1 gives maximum noise margin of 0.63 V, as shown below.

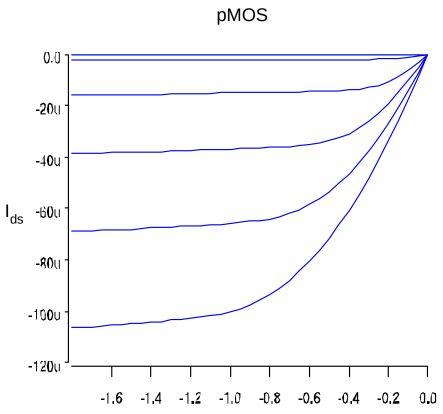


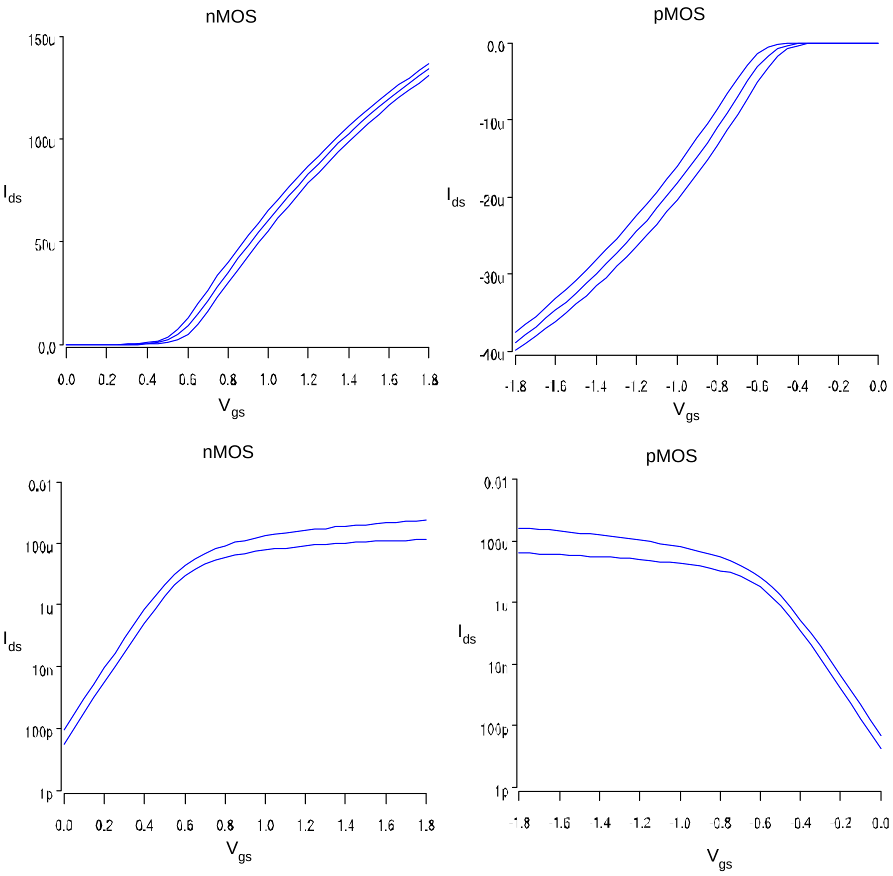
8.6 Use DC sweeps to vary V_{gs} and V_{ds}.


nMOS
600u
400u
200u
0.0
I_{ds}
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
V_{ds}


pMOS
0.0
-50u
-100u
-150u
-200u
-250u
-300u
I_{ds}
-1.6
-1.4
-1.2
-1.0
-0.8
-0.6
-0.4
-0.2
0.0
V_{ds}


nMOS
400u
300u
200u
100u
0.0
I_{ds}
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8


pMOS
0.0
-20u
-40u
-60u
-80u
-100u
-120u
I_{ds}
-1.6
-1.4
-1.2
-1.0
-0.8
-0.6
-0.4
-0.2
0.0



8.7 Your results will vary with your process.

8.8 Your results will vary with your process.

8.9 $g = 1.79$, $p = 6.53$

```
# charlib.lst
# Created by Ted Jiang 10/6/2004
GATE inv
in a
out y
* *
ENDGATE

GATE nand5
```

```
in a
in b
in c
in d
in e
out y
* 1 1 1 1 *
ENDGATE
END
```

8.10 The NAND and NOR designs have propagation delays of 153 and 174 ps using x = 1.9 and y = 2.65. Logical effort would predict best sizes of x = 2.14 and y = 3.16, giving delays of 8.6 τ = 172 ps and 9.3 τ = 186 ps. Simulating with this size give actual delays of 153 and 180 ps, respectively. The delays are slightly better than logical effort predicts because NANDs and NORs have slightly lower logical effort than estimated.

Tuning the sizes through simulation gives little improvement compared to using sizes predicted by logical effort.

```
* 510-And2.sp
* Create by Ted Jiang 10/6/2004
************************************************************************
* Parameters and models
************************************************************************
.param SUP=1.8
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post

************************************************************************
* Subcircuits
************************************************************************
.global vdd gnd

.subckt inv a y N=4 P=8
M1      y      a       gnd    gnd     NMOS   W='N'   L=2
+ AS='N*5' PS='2*N+10' AD='N*5' PD='2*N+10'
M2      y      a       vdd    vdd     PMOS   W='P'   L=2
+ AS='P*5' PS='2*P+10' AD='P*5' PD='2*P+10'
.ends

.subckt        nand2 InOuter InInner Out N='48/2' P='48/2'
m1      Mid InOuter Gnd Gnd   nmos l=2 w=N   as='5*N' ad='1.5*N'
+                                            ps='2*N+10' pd='N+3'
m2      Out InInner Mid Gnd   nmos l=2 w=N   as='1.5*N' ad='5*N'
+                                            ps='N+3' pd='2*N+10'
```

```
m3      Out InOuter Vdd Vdd           pmos l=2 w=P   as='5*P' ad='3*P'
+                                                    ps='2*P+10' pd='P+6'
m4      Out InInner Vdd Vdd           pmos l=2 w=P   as='5*P' ad='3*P'
+                                                    ps='2*P+10' pd='P+6'
.ends

.subckt nor2 InOuter InInner Out N='48/5' P='48*4/5'
m1      Out InOuter Gnd Gnd     nmos l=2 w=N   as='5*N' ad='3*N'
+                                              ps='2*N+10' pd='N+6'
m2      Out InInner Gnd Gnd     nmos l=2 w=N   as='5*N' ad='3*N'
+                                              ps='2*N+10' pd='N+6'
m3      Mid InOuter Vdd Vdd     pmos l=2 w=P   as='5*P' ad='1.5*P'
+                                              ps='2*P+10' pd='P+3'
m4      Out InInner Mid Vdd     pmos l=2 w=P   as='1.5*P' ad='5*P'
+                                              ps='P+3' pd='2*P+10'
.ends

************************************************************************
* Simulation netlist
************************************************************************


Vdd     vdd    gnd     'SUPPLY'
*Vin1   a      gnd     PULSE  0 'SUPPLY' 0ps 100ps 100ps 500ps 1000ps
*X1     a      vdd     b      nand2   P = 55         N= 55
*X2     b      c       inv    P = '2*c1*10/(3*.09)'  N = 'c1*10/(3*0.09)'
*X3     c      g       inv            P = 444        N = 222

Vin2    w      gnd     PULSE  0 'SUPPLY' 0ps 100ps 100ps 500ps 1000ps
X4      w      x       inv            P =74          N = 37
X5      vdd    y       inv            P = 74         N = 37
X6      x      y       z nor2 P = '4*c1*10/(5*0.09)' N = 'c1*10/(5*0.09)'
X7      z      g       inv            P = 444        N = 222

.param c1 = optrange(3,1,5)
.model optmod opt itropt=30
.measure bestratioX param = 'c1'

*.param c2 = optrange(2,1,5)
*.model optmod2 opt itropt=30
*.measure bestratioY param = 'c2'

************************************************************************
* Stimulus
************************************************************************
*.tran 1ps 1000ps SWEEP OPTIMIZE=optrange RESULTS=tpd1 MODEL=optmod
*.measure tpdr1                       * rising propagation delay
*+     TRIG v(a)              VAL='SUPPLY/2' RISE=1
*+     TARG v(c)              VAL='SUPPLY/2' RISE=1
*.measure tpdf1                      * falling propagation delay
*+     TRIG v(a)              VAL='SUPPLY/2' FALL=1
*+     TARG v(c)              VAL='SUPPLY/2' FALL=1
*.measure tpd1 param='(tpdr1+tpdf1)/2' goal=0 * average propagation delay

.tran 1ps 1000ps SWEEP OPTIMIZE=optrange RESULTS=tpd2 MODEL=optmod
.measure tpdr2                       * rising propagation delay
+     TRIG v(w)               VAL='SUPPLY/2' RISE=1
+     TARG v(z)       VAL='SUPPLY/2' RISE=1
.measure tpdf2                       * falling propagation delay
+     TRIG v(w)       VAL='SUPPLY/2' FALL=1
+     TARG v(z)       VAL='SUPPLY/2' FALL=1
.measure tpd2 param='(tpdr2+tpdf2)/2' goal =0 * average propagation delay

.end
```

8.11 Your results will vary with your design.

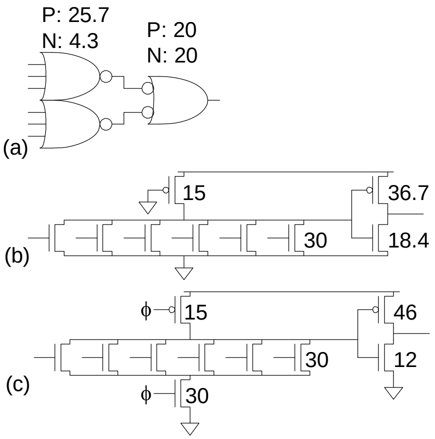
Chapter 9

9.1 In each case, B = 1 and H = (60+30)/30 = 3.

(a) NOR3 (p = 3) + NAND2 (p = 2). G = (7/3)*(4/3) = 28/9. $F = GBH$ = 28/3. $f = F^{1/2}$ = 3.05. Second stage size = 90*(4/3)/f = 39. $D = 2f + P$ = 11.1.

(b) Pseudo-nMOS NOR6 (p = 52/9) + static INV (p = 1). G = (8/9)*(1) = 8/9. $F = GBH$ = 8/3. $f = F^{1/2}$ = 1.63. Second stage size = 90*1/f = 55.1. D = 10.0.

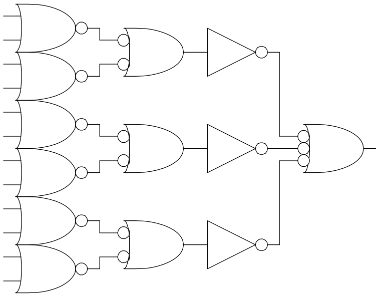
(c) Dynamic NOR6 (p = 13/3) + high-skew INV (p = 5/6). G = (2/3)*(5/6) = 10/18. $F = GBH$ = 5/3. $f = F^{1/2}$ = 1.29. Second stage size = 90*(5/6)/f = 58. D = 7.75.



9.2 Your mileage may vary. τ = 15 ps.

	rising	falling	average
static	200 ps	157 ps	178 ps = 11.8 τ
pseudo-nMOS	93 ps	148 ps	121 ps = 8.1 τ
domino	94 ps	n/a	94 ps = 6.3 τ

9.3 There are many designs such as NOR2 + NAND2 + INV + NAND3.



9.4 $H = 500 / 30 = 16.7$. Consider a two stage design: OR-OR-AND-INVERT + INV. $G = 2 * 1 = 2$. $P = 4 + 1 = 5$. $F = GBH = 33.3$. $f = F^{1/2} = 5.77$. $D = 2f + P = 16.5$

τ. The inverter size is 500 * 1 / 5.77 = 87.

9.5 (a) For $0 \le A \le 1$, $B = 1$, $I(A)$ depends on the region in which the bottom transistor operates. The top transistor is always saturated because $V_{gs} \le V_{ds}$.

$$I(A) = \begin{cases} \left(A - \frac{x}{2}\right)x & x < A \\ \frac{1}{2}A^2 & x \ge A \end{cases} = \tfrac{1}{2}(1-x)^2$$

Thus the bottom transistor is saturated for $A < 1/2$ and linear for $A > 1/2$. Solve for x in each of these two cases:

$$\tfrac{1}{2}A^2 = \tfrac{1}{2}(1-x)^2 \Rightarrow x = 1 - A \qquad A < \tfrac{1}{2}$$

$$\left(A - \tfrac{x}{2}\right)x = \tfrac{1}{2}(1-x)^2 \Rightarrow x = \frac{A + 1 - \sqrt{(A+1)^2 - 2}}{2} \qquad A \ge \tfrac{1}{2}$$

Substituting, we obtain an equation for I vs. A:

$$I(A) = \begin{cases} \frac{1}{2}A^2 & A < \frac{1}{2} \\ \dfrac{A^2 + (1-A)\sqrt{A^2 + 2A - 1}}{4} & A \ge \frac{1}{2} \end{cases}$$

For $0 \le B \le 1$, $A = 1$, the top transistor is always saturated because $V_{gs} = V_{ds}$. The bottom transistor is always linear because $V_{gs} > V_{ds}$. The current is

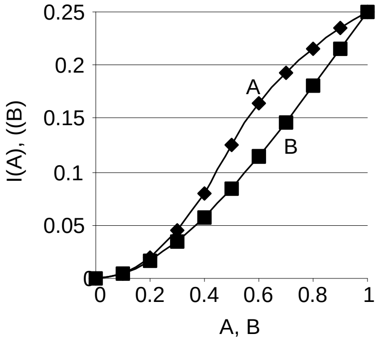
$$I(B) = \tfrac{1}{2}(B - x)^2 = \left(1 - \tfrac{x}{2}\right)x$$

Solve for x and $I(B)$:

$$x = \frac{B+1-\sqrt{(B+1)^2-2B^2}}{2}$$

$$I(B) = \frac{1+(B-1)\sqrt{-B^2+2B+1}}{4}$$

Plotting I vs. A and B, we find that the current is always higher when the lower transistor is switching than when the higher transistor is switching for a given input voltage. This plot may have been found more easily by numerical methods.



(b) The inner input of a NAND gate or any gate with series transistors has grater logical effort than the outer input because the inner transistor provides slightly less current while partially ON. This is because the intermediate node x rises as B rises, providing negative feedback that quadratically reduces the current through the top transistor as it turns ON.

9.6 g = 6/3 at the OR terminals and 4/3 at the AND terminal. p = 8/3.

9.7 Use charlib.pl from exercise 5.8. The average logical efforts and parasitic delays are 1.93, 1.92, and 1.97 and 4.49, 3.80, and 2.44 from the outer, middle, and inner inputs, respectively. The inner input has lower parasitic delay but slightly higher logical effort, as expected.

```
# charlib.lst
# Created by Ted Jiang 10/6/2004
GATE inv
in a
out y
* *
ENDGATE

GATE nor3
```

```
in a
in b
in c
out y
0 0 * *
0 * 0 *
* 0 0 *
ENDGATE

END
```

9.8 t_{pdr} = 0.0313 + 4.5288*0.0042h (in units of ns) = 2.52 + 1.53h (in units of τ)

t_{pdf} = 0.0195 + 2.8429*0.0042h (in units of ns) = 1.57 + 0.96h (in units of τ)

g_u = 1.53; p_u = 2.52; g_d = 0.96; p_d = 1.57

9.9 t_{pdr} = 0.0400 + 4.5253*0.0039h (in units of ns) = 3.22 + 1.42h (in units of τ)

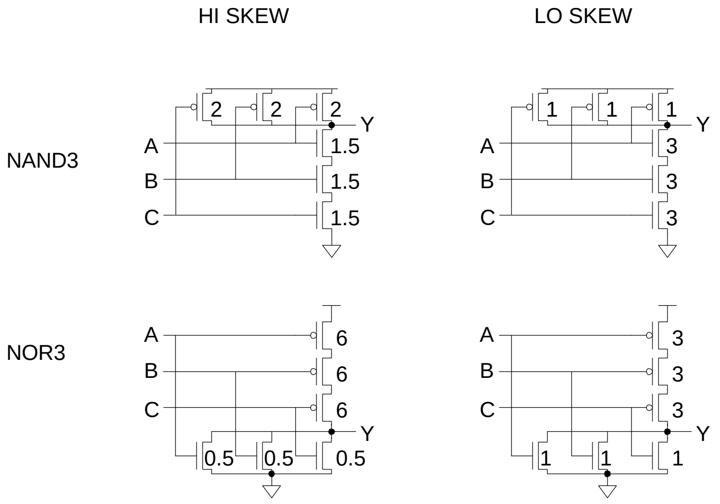
t_{pdf} = 0.0242 + 2.8470*0.0039h (in units of ns) = 1.95 + 0.90h (in units of τ)

g_u = 1.42; p_u = 3.22; g_d = 0.90; p_d = 1.95

As compared to input A, input B has a greater parasitic delay and slightly smaller logical effort. Input B must be the outer input, which must discharge the parasitic capacitance of the internal node, increasing its parasitic delay.

9.10 NAND3: HI-skew: g_u = 7/6; LO-skew: g_d = 4/3

NOR3: HI-skew: g_u = 13/6; LO-skew: g_d = 4/3



9.11 HI-skew: pMOS = 2, nMOS = sk, g_u = (2 + ks)/3, g_d = (2 + ks)/3s, g_{avg} = (2 + k + ks + 2/s)/6

LO-skew: pMOS = 2*s*, nMOS = *k*, g_u = (2*s* + *k*)/3*s*, g_d = (2*s* + *k*)/3, g_{avg} = (2 + *k* + 2*s* + *k*/*s*)/6

9.12 n_{crit} = 1. For g_{crit} = 1.5, C_{in} = 4.5, so p_{crit} = 4.5-1 = 3.5 on the critical input. For unit resistance, *R* = 2/p_{crit} + 2*(2/$p_{noncrit}$) = 1 -> $p_{noncrit}$ = 4/(1-2/p_{crit}) = 28/3. If $n_{noncrit}$ = 1/2, $g_{noncrit}$ = ($p_{noncrit}$ + $n_{noncrit}$) / 3 = 3.28.

9.13 Suppose a P/N ratio of *k* gives equal rise and fall times. If the pMOS device is of width *p* and the nMOS of width 1, then we find ***.

9.14 ρ(1, *p/g*) is the value of ρ satisfying *p/g* + ρ(1 - ln ρ) = 0. Suppose we have a path with n_1 stages, a path effort *F*, and a path parasitic delay *P*. If we add *N*-n_1 buffers of parasitic delay *p* and logical effort *g*, the best path delay is

$$D = N\left(Fg^{N-n_1}\right)^{1/N} + P + \left(N - n_1\right)p$$

Differentiate this path delay with respect to *N* to find the number of stages that minimizes delay. The best stage effort at this number of stages is ρ(*g*, *p*) = $(Fg^{N-n1})^{1/N}$. Substituting this into the derivative and simplify to find

$$\frac{\partial D}{\partial N} = \left(Fg^{N-n_1}\right)^{1/N}\left(1 + \frac{n_1 \ln g}{N} - \frac{\ln F}{N}\right) + p = 0$$

$$\rho(g,p)\left[1 - \ln\frac{\rho(g,p)}{g}\right] + p = 0$$

Assume the equality we are trying to prove is true and substitute it into the equation above to obtain

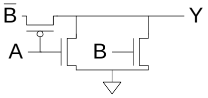
$$\rho(1,\tfrac{p}{g})\left[1 - \ln\rho(1,\tfrac{p}{g})\right] + \tfrac{p}{g} = 0$$

This is just the definition of ρ that we began with, so the substitution must have been valid and the equality is proven.

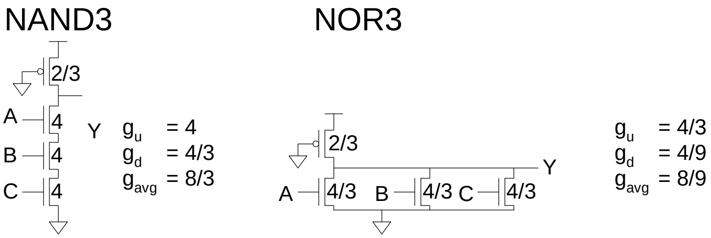
9.15 According to Section 5.2.5 for the TSMC 180 nm process, a P/N ratio of 3.6:1 gives equal rising and falling delays of 84 ps, while a P/N ratio of 1.4:1 gives the minimum average delay of 73 ps, a 13% improvement (not to mention the savings in power and area). Recall that the minima is very flat; a ratio between 1.2:1 and 1.7:1 all produce a 73 ps average delay.

9.16 The ratio for equal rise and fall delays is slower and requires large pMOS transistors. The ratio for minimum average delay results in significantly different rising and falling delays. Some paths are primarily influenced by one of these two delays, so a long rising delay can be problematic. Choosing an intermediate P/N ratio can give average delay nearly equal to the minimum and transistor sizes smaller than those for equal delay while avoiding the very slow rising delay.

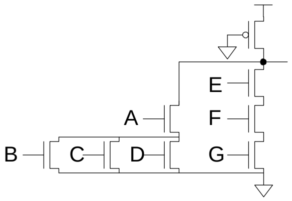
9.17 The 3-transistor NOR is nonrestoring.



9.18



9.19

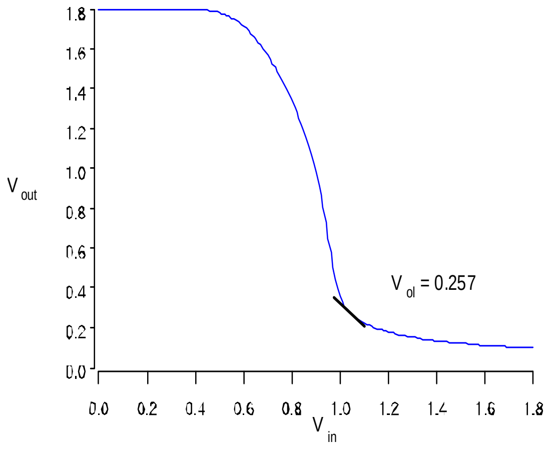


9.20 Use 8 CMOS inverters driving an 8-input psuedo-nMOS NOR. $F = 1 * (8/9) * 6 = 16/3$. $P = 1 + 34/9$. $D = 2*(16/3)^{1/2} + 43/9 = 9.4\ \tau$.

9.21 $g_d = 0.77$, $g_u = 0.76$, $g_{avg} = 0.76$; $p_d = 0.71$, $p_u = 1.13$, $p_{avg} = 0.92$

These delays can be found with charlib.pl.

V_{OL} is 0.26 V, as measured from the DC transfer characteristics.



```
# charlib.lst
# Created by Ted Jiang 10/06/04


GATE inv
in a
out y
* *
ENDGATE

GATE pseudoinv
in a
out y
* *
ENDGATE

END


* 621-Pseudo.sp
*Created by Ted Jiang 10/6/2004
************************************************************************
* Parameters and models
************************************************************************
.param SUP=1.8
.param N=32
.param P=16
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post

************************************************************************
```

```
* Simulation netlist
************************************************************************
Vdd     vdd    gnd     'SUPPLY'
Vin     a      gnd     0
m1      y a Gnd Gnd    nmos   l=2 w=N        as='5*N' ad='5*N'
+                                            ps='2*N+10' pd='2*N+10'
m2      y Gnd Vdd Vdd  pmos   l=2 w=P        as='5*P' ad='5*P'
+                                            ps='2*P+10' pd='2*P+10'
************************************************************************
* Stimulus
************************************************************************
.dc Vin 0 1.8 0.01
.end
```

9.22 No process corners are available in the MOSIS TSMC models. Your mileage may vary with other models. The following code could be used if corners were available.

```
* 622-PseudoFSSF.sp
* Created by Ted Jiang 10/6/2004
************************************************************************
* Parameters and models
************************************************************************
.param SUP=1.8
.param N=32
.param P=16
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' FS
.option post

************************************************************************
* Simulation netlist
************************************************************************
Vdd            vdd     gnd    'SUPPLY'
Vin            a       gnd    0
m1             y a Gnd Gnd    nmos    l=2 w=N       as='5*N' ad='5*N'
+                                                   ps='2*N+10' pd='2*N+10'
m2             y Gnd Vdd Vdd  pmos    l=2 w=P       as='5*P' ad='5*P'
+                                                   ps='2*P+10' pd='2*P+10'
************************************************************************
* Stimulus
************************************************************************
.dc Vin 0 1.8 0.01
.alter
.lib '../models/mosistsmc180/opconditions.lib' SF
.end
```

9.23 The average logical effort is 5/6, substantially better than 7/3 for a static CMOS

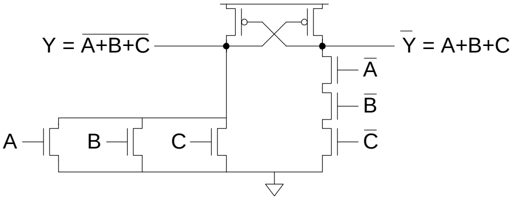
NOR3.

9.24 The average logical effort is 5/4, slightly less than 4/3 for a static CMOS NOR2. The improvement is marginal and comes at the cost of contention.

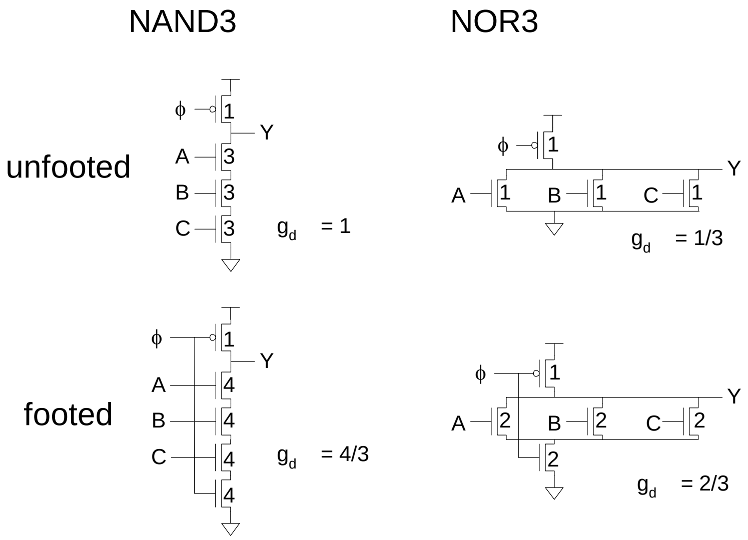
9.25 Simulating the various gates gave the following average propagation delays (in ps). This is a bit surprising and indicates SFPL may be advantageous for wide NORs..

# inputs	Pseudo-nMOS	SFPL
2	67	71
4	83	79
8	116	98
16	182	129

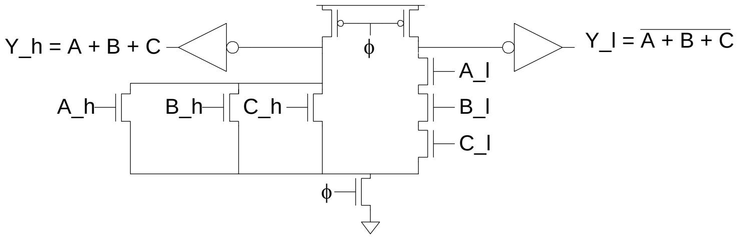
9.26



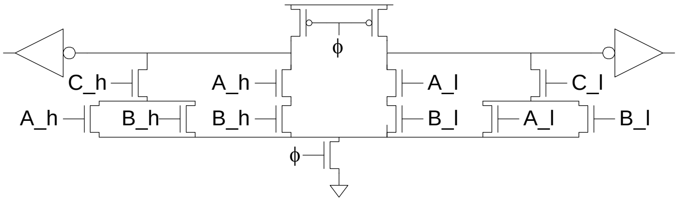
9.27



9.28

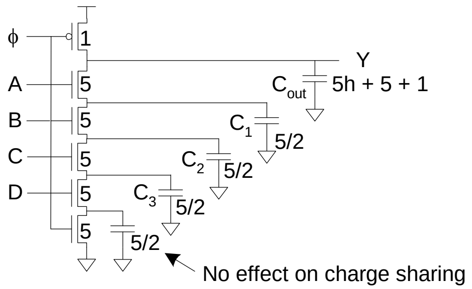


9.29

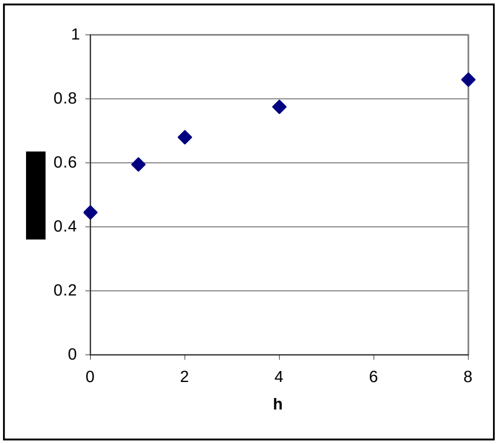


9.30 Your results should be similar to those of Figure 6.37.

9.31 The worst case is when *A* is low on one cycle, *B*, *C*, and *D* are high, and all the internal nodes become predischarged to 0. Then *D* falls low during precharge. Then *A* goes high during evaluation. The NAND has 11 units of capacitance on C_{out} precharged to V_{DD} and 7.5 units of internal capacitance (C_1, C_2, C_3) that will be initially low. The output will thus droop to 11/(11+7.5) V_{DD} = 0.59 V_{DD}.

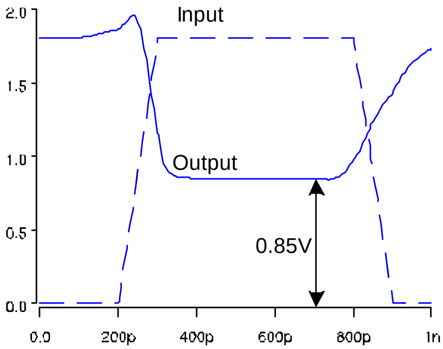


9.32 The droop is (6+5h)/(6+5h+7.5). Charge sharing is less serious at high fanout.



9.33 With a secondary precharge transistor, one of the internal nodes is guaranteed to be high rather than low. Thus 11 + 2.5 = 13.5 units of capcitance are high and 5 units are low, reducing the charge sharing noise to 13.5 / (13.5 + 5) V_{DD} = 0.73 V_{DD}.

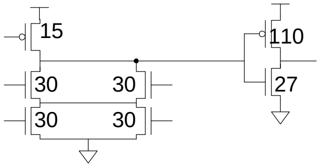
9.34 The simulated output droops to 0.85 V = 0.46 V_{DD}, somewhat lower than 0.59 V_{DD} predicted in Exercise 6.31. The capacitance of the internal nodes is larger than anticipated. According to Table 5.5, the diffusion capacitance on an uncontacted (merged) node is 1.41 / 1.67 times that of gate capaictance. Using this figure, the expected droop is 11/(11+3 * 5 * 1.41/1.67) = 0.465, an excellent match with simulation.



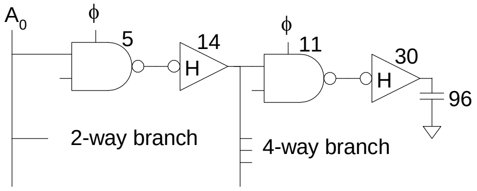
```
*634-631.sp
***********************************************************************
* Parameters and models
***********************************************************************
.param SUP=1.8
.param P = 4
.param N = 20
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post
***********************************************************************
*subcircuit
***********************************************************************
.subckt        inv In Out N=16 P=32
* Assumes 5 lambda of diffusion on the source/drain
m1      Out In Gnd Gnd       nmos    l=2 w=N   as='5*N' ad='5*N'
+                                              ps='2*N+10' pd='2*N+10'
m2      Out In Vdd Vdd       pmos    l=2 w=P   as='5*P' ad='5*P'
+                                              ps='2*P+10' pd='2*P+10'
.ends
***********************************************************************
* Simulation netlist
***********************************************************************
Vdd     vdd    gnd     'SUPPLY'
Vin     A      gnd     PULSE 0 'SUPPLY' 200ps 100ps 100ps 500ps 1000ps
Vclk    clk    gnd     PULSE 0 'SUPPLY' 100ps 100ps 100ps 500ps 1000ps
.global vdd gnd
m1      Y clk Vdd Vdd  pmos  l=2    W=P       as='5*P' ad='5*P'
+                                              ps='2*P+10' pd='2*P+10'
m2      Y A  h  gnd  nmos   l=2     W=N       as='3*N' ad='5*N'
+                                              ps='2*N+6' pd='2*N+10'
m3      h vdd  i  gnd  nmos l=2     w=N       as='3*N' ad='0'
```

```
+                                            ps='2*N+6' pd='0'
m4      i vdd  j  gnd  nmos l=2     w=N      as='3*N' ad='0'
+                                            ps='2*N+6' pd='0'
m5      j gnd  l  gnd  nmos l=2    W=N       as='3*N' ad='0'
+                                            ps='2*N+6' pd='0'
m6      l clk gnd gnd  nmos l=2    W=N       as='5*N' ad='0'
+                                            ps='2*N+10' pd='0'
X2      Y      Z       inv  P='2*N/3'        N='N/3'

***********************************************************************
* Stimulus
***********************************************************************
.ic v(y)='SUPPLY' v(h)=0 v(i)=0 v(j)=0 v(l)=0
.trans 1ps 2000ps
.end
```

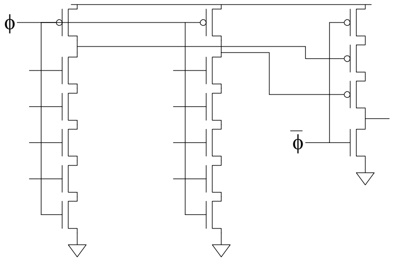
9.35 H = 500 / 30 = 16.7. Consider a two stage design: footless dynamic OR-OR-AND-INVERT + HI-skew INV. G = 2/3 * 5/6 = 10/18. P = 5/3 + 5/6 = 5/2. $F = GBH$ = 9.3. $f = F^{1/2}$ = 3.0. $D = 2f + P$ = 8.6 τ. The inverter size is 500 * (5/6) / 3.0 = 137.



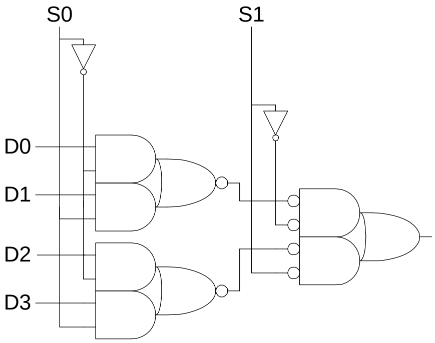
9.36 One design is: Dynamic NAND2 - HI skew INV - Dynamic NAND2 - HI skew INV. G = 1 * (5/6) * 1 * (5/6) = 25/36. F = (25/36) * 8 * 9.6 = 53.3. P = 4/3 + 5/6 + 4/3 + 5/6 = 4.3. $f = F^{1/4}$ = 2.7. $D = 4f + P$ = 15.1 τ.

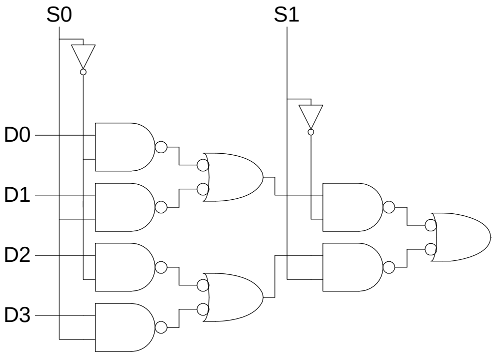


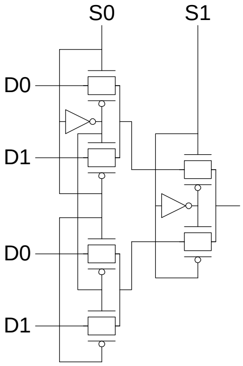
9.37



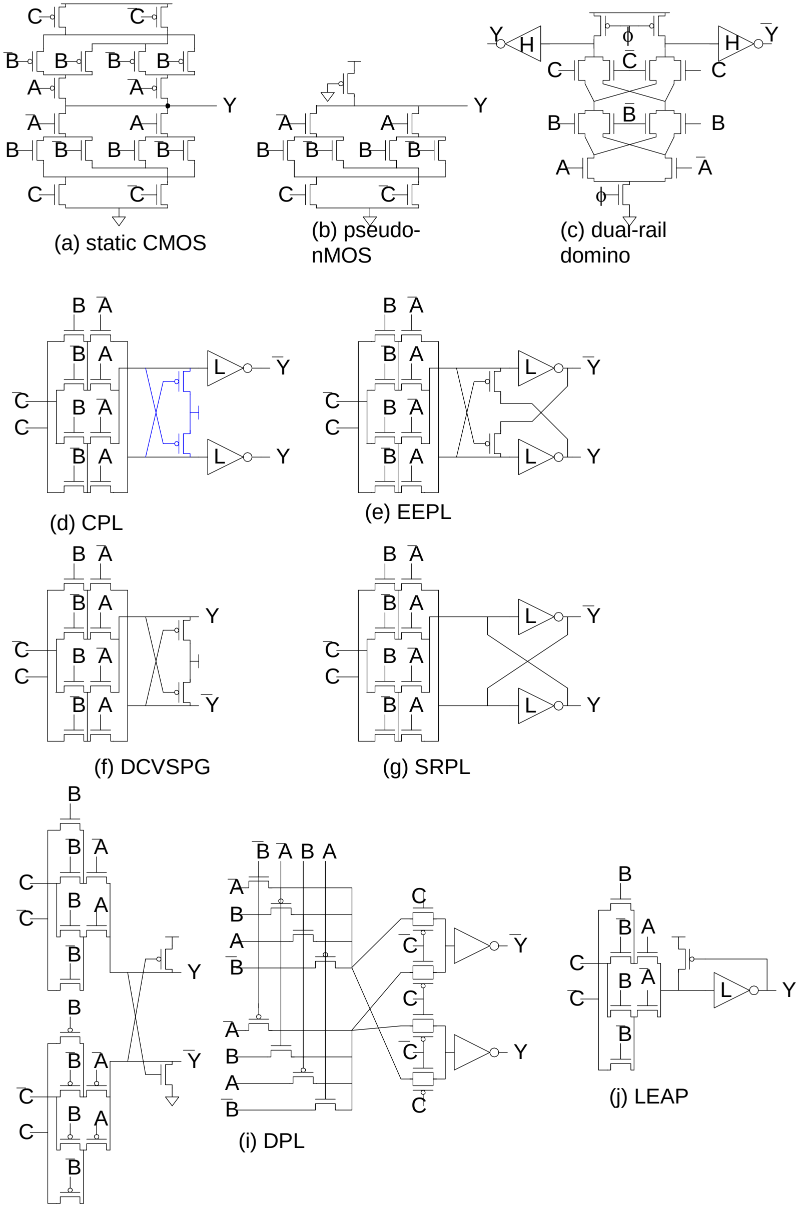
9.38 The static designs with and without complex AAOI gates use 28 and 40 transistors, respectively. The nonrestoring transmission gate design uses 16 transistors, though adding inverters on the inputs would raise that to 24 and make the mux restoring but inverting. Adding an output inverter to make the mux noninverting brings the transistor count to 26.







9.39



(a) static CMOS

(b) pseudo-nMOS

(c) dual-rail domino

(d) CPL

(e) EEPL

(f) DCVSPG

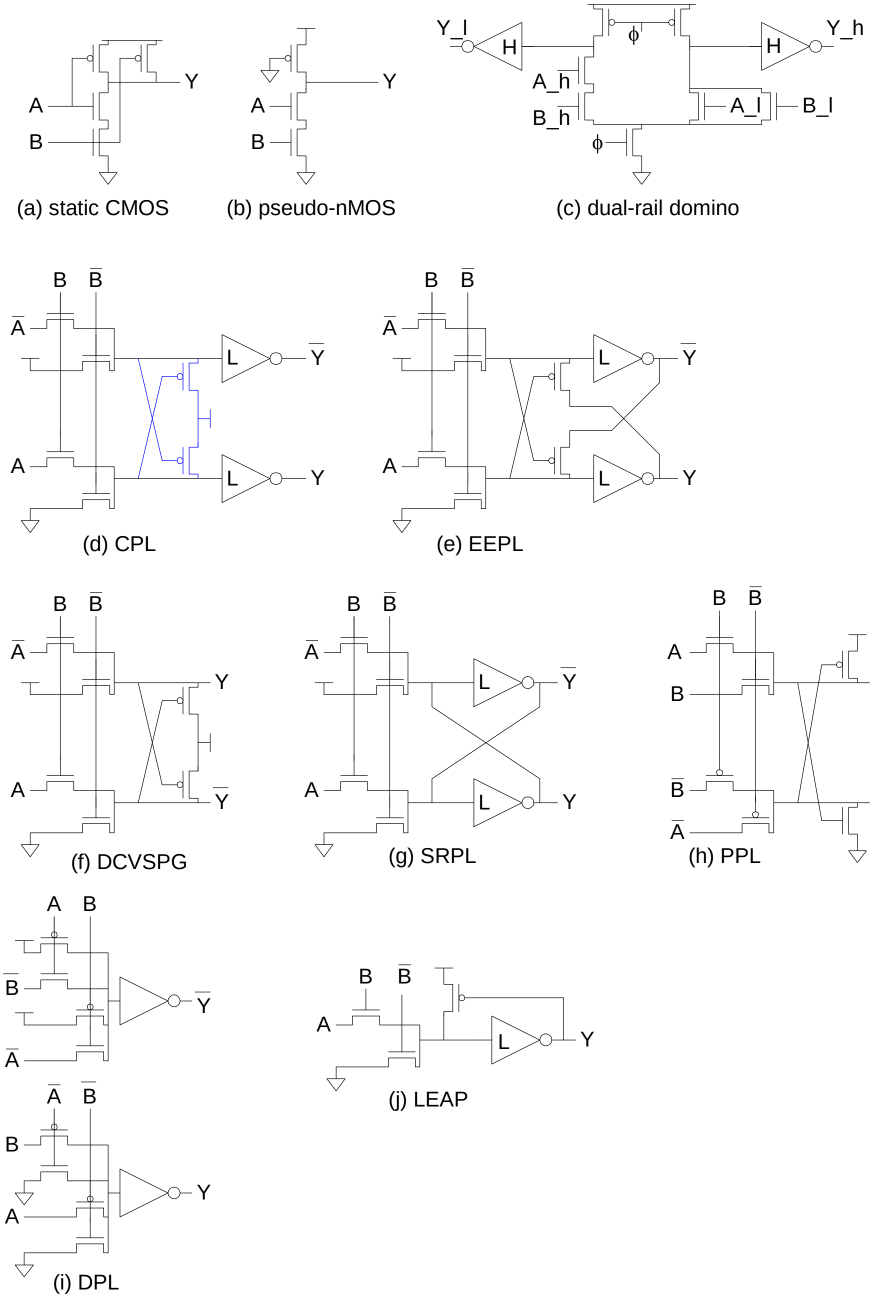
(g) SRPL

(i) DPL

(j) LEAP

(h) PPL

9.40



(a) static CMOS (b) pseudo-nMOS (c) dual-rail domino

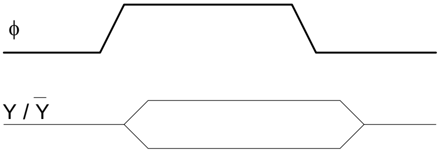
(d) CPL (e) EEPL

(f) DCVSPG (g) SRPL (h) PPL

(i) DPL (j) LEAP

9.41 ### no solution available

9.42 When the clock is low, the two outputs equalize at $V_{DD}/2$. When the clock rises, one side pulls down, fully turning ON the pMOS transistor to pull the other side up. This gate saves precharge power relative to dynamic logic because the precharge equalizes the two outputs rather than drawing power from the rail. The partial swing may lead to faster transitions. However, it consumes extra power early in evaluation because of contention between the partially ON pMOS transistor and the ON pull-down stack.



9.43 n/a

9.44 The nMOS is in the linear region and the pMOS is saturated. By KCL

$$\beta_n\left(V_{\mathrm{DD}}-V_{tn}-\frac{\mathrm{V}_{\mathrm{out}}}{2}\right)V_{\mathrm{out}}=\frac{\beta_p}{2}\left(V_{DD}+V_{tp}\right)^2$$

$$V_{\mathrm{out}}=\left(V_{DD}-V_{tn}\right)-\sqrt{\left(V_{DD}-V_{tn}\right)^2-\tfrac{\beta_p}{\beta_n}\left(V_{DD}+V_{tp}\right)^2}$$

Chapter 10

10.1 (a) t_{pd} = 500 - (50 + 65) = 385 ps; (b) t_{pd} = 500 - 2(40) = 420 ps; (c) t_{pd} = 500 - 40 = 460 ps.

10.2 (a) t_{pd} = 500 - (50 + 65 + 50) = 335 ps; (b) t_{pd} = 500 - 2(40) = 420 ps; (c) t_{pd} = 500 - (50 + 25 - 80 + 50) = 455 ps.

10.3 (a) t_{cd} = 30 - 35 = 0; (b) t_{cd} = 30 - 35 = 0; (c) t_{cd} = 30 - 35 - 60 = 0; (d) t_{cd} = 30 - 35 + 80 = 75 ps.

10.4 (a) t_{cd} = 30 - 35 + 50 = 45 ps; (b) t_{cd} = 30 - 35 + 50 = 45 ps; (c) t_{cd} = 30 - 35 + 50 - 60 = 0; (d) t_{cd} = 30 - 35 + 80 + 50 =125 ps.

10.5 (a) t_{borrow} = 0; (b) t_{borrow} = 250 - 25 = 225 ps; (c) t_{borrow} = 250 - 25 - 60 = 165 ps; (d) t_{borrow} = 80 - 25 = 55 ps.

10.6 (a) t_{borrow} = 0; (b) t_{borrow} = 250 - 25 - 50 = 175 ps; (c) t_{borrow} = 250 - 25 - 60 -50 = 115 ps; (d) t_{borrow} = 80 - 25 - 50 = 5 ps.

10.7 If the pulse is wide and the data arrives while the pulsed latch is transparent, the latch contributes its D-to-Q delay just like a regular transparent latch. If the pulse is narrow, the data will have to setup before the earliest skewed falling edge. This is at time t_{setup} - t_{pw} + t_{skew} before the latest rising edge of the pulse. After the rising edge, the latch contributes a clk-to-Q delay. Hence, the total sequencing overhead is t_{pcq} + t_{setup} - t_{pw} + t_{skew}.

10.8 $t_{setup\text{-}flop} = t_{setup\text{-}latch} + t_{nonoverlap}$; $t_{hold\text{-}flop} = t_{hold} - t_{nonoverlap}$; $t_{pcq\text{-}flop} = t_{pcq}$.

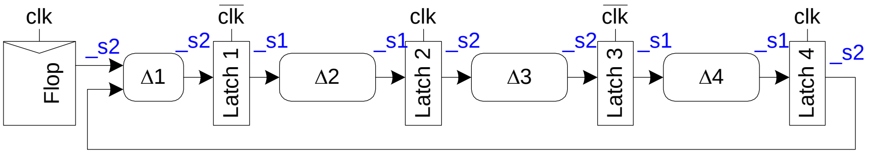
10.9 (a) 1200 ps: no latches borrow time, no setup violations. 1000 ps: 50 ps borrowed through L1, 130 ps through L2, 80 ps through L3. 800 ps: 150 ps borrowed through L1, 330 ps borrowed through L2, L3 misses setup time.

(b) 1200 ps: no latches borrow time, no setup violations. 1000 ps: 100 ps borrowed through L2, 50 ps through L4. 800 ps: 200 ps borrowed through L2, 200 ps borrowed through L3, 350 ps borrowed through L4, 250 ps borrowed through L1, L2 then misses setup time.

10.10 (a) 700 ps; (b) 825 ps; (c) 1200 ps.

10.11 (a) 700 ps; (b) 825 ps; (c) 1200 ps. The transparent latches are skew-tolerant and moderate amounts of skew do not slow the cycle time.

10.12



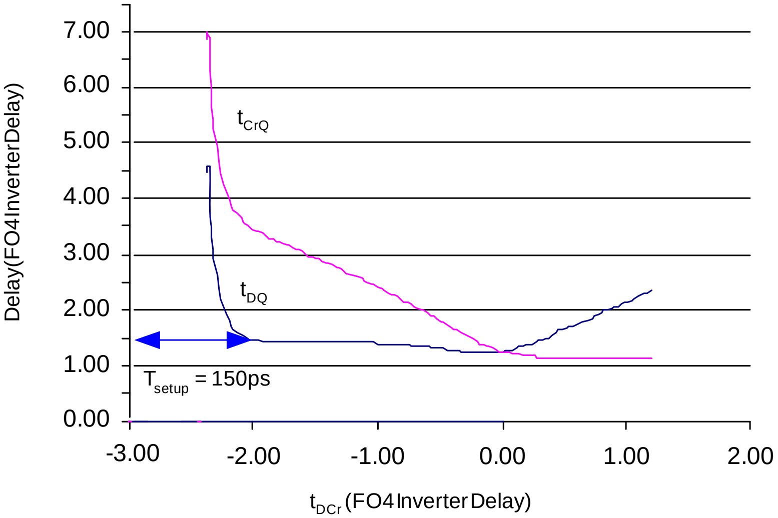
10.13 The t_{pdq} delays are 151 ps for a conventional dynamic latch and 162 ps for a TSPC latch.

```
*713-latch.sp
***********************************************************************
* Parameters and models
***********************************************************************
.param SUP=1.8
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post

***********************************************************************
* Subcircuits
***********************************************************************
.global vdd gnd
.subckt        inv In Out N=4 P=8
* Assumes 5 lambda of diffusion on the source/drain
m1      Out In Gnd Gnd  nmos    l=2 w=N        as='5*N' ad='5*N'
```

```
+                                            ps='2*N+10' pd='2*N+10'
m2     Out In Vdd Vdd pmos   l=2 w=P         as='5*P' ad='5*P'
+                                            ps='2*P+10' pd='2*P+10'
.ends
.subckt latchd c nc D Q N=4 P=4
X1     D       x       inv
m1     Q       c       x      gnd     nmos   l=2 w=N as='5*N' ad='5*N'
+                                            ps='2*N+10' pd='2*N+10'
m2     Q       nc      x      vdd     pmos   l=2 w=P as='5*P' ad='5*P'
+                                            ps='2*P+10' pd='2*P+10'
.ends
************************************************************************
* Simulation netlist
************************************************************************
Vdd    vdd     gnd     'SUPPLY'
Vin    a       gnd     PULSE  0 'SUPPLY' 400ps 100ps 100ps 2000ps 4000ps
Vclk   clk     gnd     PULSE 0 'SUPPLY' 200ps 100ps 100ps 1000ps 2000ps
Vnclk  nclk    gnd     PULSE 'SUPPLY' 0 200ps 100ps 100ps 1000ps 2000ps
X1 clk nclk a D latchd
X2 clk nclk D Q latchd M=4
X3 clk nclk Q Y latchd M=16

************************************************************************
* Stimulus
************************************************************************
.trans 1ps     4000ps
.measure tdqf
+     TRIG v(D)        VAL='SUPPLY/2' RISE=1
+     TARG v(Q)        VAL='SUPPLY/2' FALL=1
.measure tdqr
+     TRIG v(D)        VAL='SUPPLY/2' FALL=1
+     TARG v(Q)        VAL='SUPPLY/2' RISE=1
.measure tdq param='(tdqf+tdqr)/2'
.end

*713-tspc.sp
************************************************************************
* Parameters and models
************************************************************************
.param SUP=1.8
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post
************************************************************************
* Subcircuits
************************************************************************
.global vdd gnd
```

```
.subckt tspclatch c D Q N=4 P=8

m1     x      D       vdd    vdd     pmos   l=2 w=P as='5*P' ad='5*P'
+                                           ps='2*P+10' pd='2*P+10'
m2     x      c       y      gnd     nmos   l=2 w=N as='5*N' ad='5*N'
+                                           ps='2*N+10' pd='2*N+10'
m3     y      D       gnd    gnd     nmos   l=2 w=N as='5*N' ad='5*N'
+                                           ps='2*N+10' pd='2*N+10'
m4     Q      x       vdd    vdd     pmos   l=2 w=P as='5*P' ad='5*P'
+                                           ps='2*P+10' pd='2*P+10'
m5     Q      c       z      gnd     nmos   l=2 w=N as='5*N' ad='5*N'
+                                           ps='2*N+10' pd='2*N+10'
m6     z      x       gnd    gnd     nmos   l=2 w=N as='5*N' ad='5*N'
+                                           ps='2*N+10' pd='2*N+10'
.ends
***********************************************************************
* Simulation netlist
***********************************************************************
Vdd    vdd    gnd     'SUPPLY'
Vin    A      gnd     PULSE   0 'SUPPLY' 400ps 100ps 100ps 2000ps 4000ps
Vclk   clk    gnd     PULSE 0 'SUPPLY' 200ps 100ps 100ps 1000ps 2000ps
X1 clk  A D tspclatch
X2 clk  D Q tspclatch m=4
X3 clk  Q Y tspclatch M=16

***********************************************************************
* Stimulus
***********************************************************************
.trans 1ps    4000ps
.measure tdqf
+     TRIG v(D)       VAL='SUPPLY/2' RISE=1
+     TARG v(Q)       VAL='SUPPLY/2' RISE=1
.measure tdqr
+     TRIG v(D)       VAL='SUPPLY/2' FALL=1
+     TARG v(Q)       VAL='SUPPLY/2' FALL=1
.measure tdq param='(tdqf+tdqr)/2'
.end
```

10.14 For a rising input, t_{setup} = 150 ps. t_{hold} = -43 ps. Note that t_{hold} is defined as the minimum time from *clk* to *D* for which the output never rises



```
*714-holdtime.sp
***********************************************************************
* Parameters and models
***********************************************************************
.param SUP=1.8
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post
.param t = 1
***********************************************************************
* Subcircuits
***********************************************************************
.global vdd gnd

.subckt        inv In Out N=4 P=8
* Assumes 5 lambda of diffusion on the source/drain
m1     Out In Gnd Gnd        nmos    l=2 w=N as='5*N' ad='5*N'
+                                            ps='2*N+10' pd='2*N+10'
m2     Out In Vdd Vdd        pmos    l=2 w=P as='5*P' ad='5*P'
+                                            ps='2*P+10' pd='2*P+10'
.ends

.subckt tspclatch c D Q N=4 P=8

m1     x       D      vdd    vdd     pmos    l=2 w=P as='5*P' ad='5*P'
+                                            ps='2*P+10' pd='2*P+10'
m2     x       c      y      gnd     nmos    l=2 w=N as='5*N' ad='5*N'
```

```
+                                               ps='2*N+10' pd='2*N+10'
m3      y       D       gnd     gnd     nmos    l=2 w=N as='5*N' ad='5*N'
+                                               ps='2*N+10' pd='2*N+10'
m4      Q       x       vdd     vdd     pmos    l=2 w=P as='5*P' ad='5*P'
+                                               ps='2*P+10' pd='2*P+10'
m5      Q       c       z       gnd     nmos    l=2 w=N as='5*N' ad='5*N'
+                                               ps='2*N+10' pd='2*N+10'
m6      z       x       gnd     gnd     nmos    l=2 w=N as='5*N' ad='5*N'
+                                               ps='2*N+10' pd='2*N+10'
.ends

***********************************************************************
* Simulation netlist
***********************************************************************
Vdd     vdd     gnd     'SUPPLY'
Vran    ran     gnd     '0.25*SUPPLY'
Vin     A gnd   PULSE   0 'SUPPLY' '400ps+arrival' 100ps 100ps 1000ps 8000ps
Vclk    clk gnd PULSE 0 'SUPPLY' 400ps 100ps 100ps 400ps 8000ps
X1 clk A D tspclatch
X2 clk  D Q tspclatch m=4
X3 clk  Q Y tspclatch M=16

***********************************************************************
* Stimulus
***********************************************************************
.trans 1ps      8000ps SWEEP arrival 312ps 315ps 2ps

.measure thold
+     TRIG v(clk)             VAL='SUPPLY/2' FALL=1
+     TARG v(D)               VAL='SUPPLY/2' RISE=1
.measure tcq
+       TRIG v(clk)           VAL='SUPPLY/2'  FALL=1
+       TARG v(q)             VAL='SUPPLY/20' RISE=1
.end
```

10.15 t_{pd} = 500 - 2(40) = 420 ps.

10.16 t_{pd} = 500 - 2(40 + 50) = 320 ps.

10.17 t_{pd} = 500 ps. Skew-tolerant domino with no latches has no sequencing overhead.

10.18 t_{pd} = 500 ps. The path can tolerate moderate amounts of skew without degradation.

10.19 t_{borrow} = 125 ps - 50 ps - t_{hold} = 75 ps - t_{hold}.

10.20 t_{borrow} = (0.65-0.25)*500 - 50 ps - t_{hold}= 150 ps - t_{hold}.

10.21 ### no solution available

10.22 In absolute terms, the dynamic NOR falling propagation delay varies by 135.1 / 69.0 = 96%. But in relative terms compared to a FO4 inverter that it is tracking, it only varies by 1.10 / 1.05 = 5%.

T (^oC)	V_{DD}	FO4 Inv Delay (ps)	NOR Delay (ps)	NOR Delay (FO4)
70	1.8	89.6	96.1	1.07
0	1.98	66.0	69.0	1.05
125	1.98	94.1	99.6	1.06
0	1.62	79.3	86.2	1.09
125	1.62	123.3	135.1	1.10

Note that according to Table 5.8, one would expect a 99 ps FO4 delay in the 70 C, 1.8 V corner. However, the table was generated using a P/N ratio of 32 / 16 λ. In this simulation, the inverter used 8 / 4 λ devices and appears to be slightly faster.

```
*722-corners.sp
*created by Ted Jiang 11/3/04
*******************************************
*Parameters
*******************************************
.param SUP=1.8
.option scale=90n
.lib '../models/mosistsmc180/opconditions.lib' TT
.option post

*******************************************
*Subcircuits
*******************************************
.global vdd gnd

.subckt dyNOR8      A B C D E F G H CLK Y  P=8     N =8
m1   Y       CLK     vdd     vdd     pmos    L =2 W = 'P'
+as='5*P' ad='5*P' ps='10+2*P' pd='10+2*P'
m2   Y       A       W       gnd     nmos    L=2 W='N'
+as='5*N' ad='5*N' ps='10+2*N' pd='10+2*N'
m3   Y       B       W       gnd     nmos    L=2 W='N'
+as='5*N' ad='5*N' ps='10+2*N' pd='10+2*N'
m4   Y       C       W       gnd     nmos    L=2 W='N'
+as='5*N' ad='5*N' ps='10+2*N' pd='10+2*N'
m5   Y       D       W       gnd     nmos    L=2 W='N'
+as='5*N' ad='5*N' ps='10+2*N' pd='10+2*N'
```

```
m6  Y       E       W      gnd    nmos    L=2 W='N'
+as='5*N' ad='5*N' ps='10+2*N' pd='10+2*N'
m7  Y       F       W      gnd    nmos    L=2 W='N'
+as='5*N' ad='5*N' ps='10+2*N' pd='10+2*N'
m8  Y       G       W      gnd    nmos    L=2 W='N'
+as='5*N' ad='5*N' ps='10+2*N' pd='10+2*N'
m9  Y       H       W      gnd    nmos    L=2 W='N'
+as='5*N' ad='5*N' ps='10+2*N' pd='10+2*N'
m10 W       CLK     gnd    gnd    nmos    L=2 W='N'
+as='5*N' ad='5*N' ps='10+2*N' pd='10+2*N'

.ends

.subckt inv a y N=4 P=8
M1  y       a       gnd    gnd    NMOS    W='N'   L=2
+ AS='N*5' PS='2*N+10' AD='N*5' PD='2*N+10'
M2  y       a       vdd    vdd    PMOS    W='P'   L=2
+ AS='P*5' PS='2*P+10' AD='P*5' PD='2*P+10'
.ends

********************************************
*Simulation netlist
********************************************
Vdd vdd     gnd     'SUPPLY'
Vin A       gnd     PULSE  0 'SUPPLY'     200ps 100ps    100ps 700ps 2000ps
Vclk CLK    gnd     PULSE  0 'SUPPLY'     0       100ps  100ps   1500ps
2000ps
X1  A       gnd gnd gnd gnd gnd gnd gnd     CLK     B       dyNOR8
X9  B       1       inv    m='32/12'
X2  1       gnd gnd gnd gnd gnd gnd gnd     CLK     2       dyNOR8 m='16'
X10 2       C       inv    m='512/12'
X3  C       gnd gnd gnd gnd gnd gnd gnd     CLK     D       dyNOR8 m='256'

X4  A       X       inv
X5  X       Y       inv    m=4
X6  Y       Z       inv    m=16
X7  Z       H       inv    m=64
X8  H       J       inv    m=256
********************************************
*Stimulus
********************************************
.trans 1ps  2000ps
.measure invtpdr           * rising propagation delay
+     TRIG v(y)     VAL='SUPPLY/2' FALL=1
+     TARG v(z)     VAL='SUPPLY/2' RISE=1
.measure invtpdf           * falling propagation delay
+     TRIG v(y)     VAL='SUPPLY/2' RISE=1
+     TARG v(z)     VAL='SUPPLY/2' FALL=1
.measure invtpd param='(invtpdr+invtpdf)/2' * average propagation delay

.measure nortpdf           * falling propagation delay
+     TRIG v(1)     VAL='SUPPLY/2' RISE=1
+     TARG v(2)     VAL='SUPPLY/2' FALL=1

.alter
.lib '../models/mosistsmc180/opconditions.lib' FF
.alter
.lib '../models/mosistsmc180/opconditions.lib' FS
.alter
.lib '../models/mosistsmc180/opconditions.lib' SF
.alter
.lib '../models/mosistsmc180/opconditions.lib' SS
.end
```

10.23 Solve for T_c:

$$100\text{ years} = \frac{T_c e^{\frac{T_c}{54\text{ ps}}}}{(10^7)(21\text{ ps})} \Rightarrow T_c = 1811\text{ ps}$$

10.24 ### no solution available

10.25 If the flip-flop goes metastable near $V_{DD}/2$, the synchronizer will indeed produce a good high output during metastability. However, the flip-flop may eventually resolve to a low value, causing the synchronizer output to suddenly fall low. Because the resolution time can be unbounded, the clock-to-Q delay of the synchronizer is also unbounded. The problem with synchronizers is not that their output takes on an illegal logic level for a finite period of time (all logic gates do that while switching), but rather that the delay for the output to settle to a correct value cannot be bounded. With high probability it will eventually resolve, but without knowing more about the internal characteristics of the flip-flop, it is dangerous to make assumptions about the probability.

Chapter 11

11.1 Your results will vary.

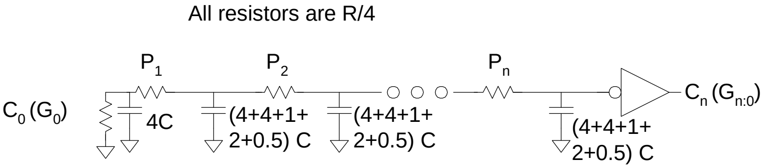
11.2 Overflow for signed numbers only occurs when adding numbers with the same sign (positive or negative). The numbers overflow (V) if the sign of the result Y does not match the sign of the inputs A and B:

$$V = A_{N-1}B_{N-1}\overline{Y}_{N-1} + \overline{A}_{N-1}\overline{B}_{N-1}Y_{N-1}$$

11.3 $V = A_{N-1}(B_{N-1} \oplus \text{SUB})\overline{Y}_{N-1} + \overline{A}_{N-1}(\overline{B}_{N-1} \oplus \text{SUB})Y_{N-1}$

11.4 The dynamic chain drives a known load capacitance, so its delay can be treated entirely as a parasitic delay. Then the output inverter contributes logical effort and additional parasitic delay. The input capacitance is 4. The output resistance of the inverter is *R* for the critical rising output, equal to that of a unit inverter. Hence, the logical effort is *g* = 4/3. The output inverter has a parasitic delay of 5/6. The parasitic delay of the dynamic stage is computed using the Elmore delay model and added on to make:

$$p = \frac{5}{6} + \frac{\left(\frac{R}{4}\right)(4C) + \sum_{i=1}^{n}(n+1)\left(\frac{R}{4}\right)(11.5C)}{3RC} = \frac{5}{6} + \frac{1 + \frac{11.5n}{8}(n+3)}{3} = \frac{11.5}{24}n^2 + \frac{11.5}{8}n + \frac{7}{6}$$



11.5 Assuming the side loads are negligible so that each carry chain drives another identical chain and has *h* = 1, the stage delay is *g* + *p*. The number of stages is inversely proportional to *n*. Hence the delay per bit scales as:

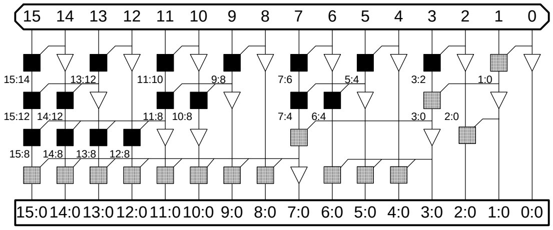
$$d = \frac{1}{n}\left[\frac{11.5}{24}n^2 + \frac{11.5}{8}n + \frac{7}{6} + \frac{4}{3}\right]$$

Taking the derivative of delay with respect to the length of each chain n and setting that equal to zero gives allows us to solve for the best chain length. Because the parasitic capacitance is large, the best delay is achieved with short carry chains (*n* = 2 or 3).

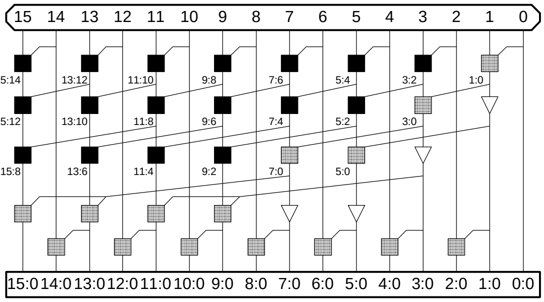
$$\frac{\partial}{\partial n}d = \frac{11.5}{24} - \frac{15}{6n^2} = 0 \Rightarrow n = 2.28$$

11.6 8 stages for 32-bit, 11 stages for 64-bit addition.

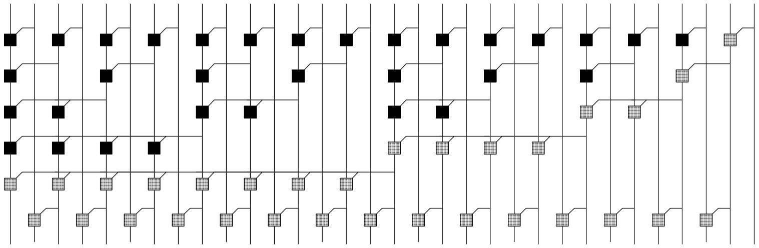
11.7



11.8



11.9



11.10

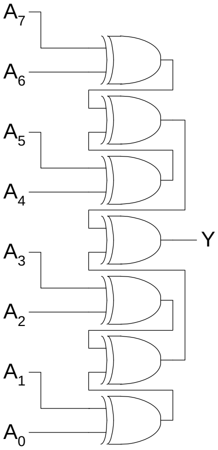
$$
\begin{aligned}
C_{out} &= \overline{(A \oplus B)\bar{C} + \overline{(A \oplus B)}\bar{A}} \\
&= \overline{A\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}\bar{B}} \\
&= \mathrm{MAJ}(A, B, C)
\end{aligned}
$$

11.11

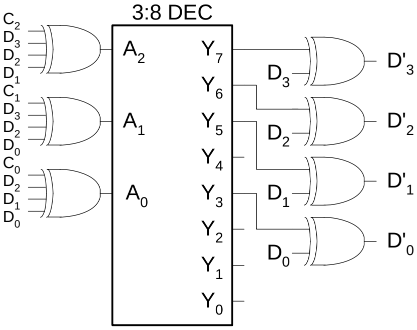
$$
\begin{aligned}
H_{i:j} &= G_{i:k} + G_{i-1:k} + P_{i-1:k-1}H_{k-1:j} \\
&= G_{i:k} + G_{i-1:k} + P_{i-1:k}P_{k-1:k-1}H_{k-1:j} \\
&= G_{i:k} + G_{i-1:k} + P_{i-1:k}G_{k-1:j} \\
&= G_{i:k} + G_{i-1:k} + G_{i-1:j} \\
&= G_{i:j} + G_{i-1:j} \\
I_{i:j} &= P_{i-1:k-1}P_{k-2:j-1} \\
&= P_{i-1:j-1}
\end{aligned}
$$

11.12 $-B = \overline{B} + 1$. Thus, the design of Figure 10.53 can be used if the *B* input is complemented and $c_0 = 1$.

11.13



11.14 Use multiple-input XOR gates to compute the syndrome. Use a decoder to identify which bit needs correcting (000 means none need correcting). Use XOR gates to flip the bit that needs to be corrected to produce the outputs D'.



11.15 4 check bits suffice for up to 2^4-4-1 = 11 data bits.

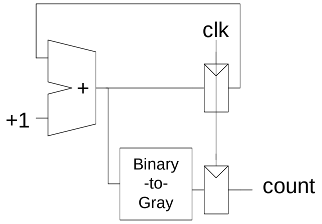
12	11	10	9	8	7	6	5	4	3	2	1
	■		■		■		■		■		▨
	■	■			■	■			■	▨	
■					■	■	■	▨			
■	■	■	■	▨							
D_7	D_6	D_5	D_4	C_3	D_3	D_2	D_1	C_2	D_0	C_1	C_0

$$
\begin{aligned}
C_0 &= D_6 \oplus D_4 \oplus D_3 \oplus D_1 \oplus D_0 \\
C_1 &= D_6 \oplus D_5 \oplus D_3 \oplus D_2 \oplus D_0 \\
C_2 &= D_7 \oplus D_3 \oplus D_2 \oplus D_1 \\
C_3 &= D_7 \oplus D_6 \oplus D_5 \oplus D_4
\end{aligned}
$$

11.16 0: 0000; 1: 0001; 2: 0011; 3: 0010; 4: 0110; 5: 0111; 6: 0101; 7: 0100; 8: 1100; 9: 1101; 10: 1111; 11: 1110; 12: 1010; 13: 1011; 14: 1001; 15: 1000.

11.17 One way to do this is with a finite state machine, in which the state indicates the present count. The FSM could be described in a hardware description language with a case statement indicating the order of states. This technique does not generalize to N-bit counters very easily.

Another approach is to use an ordinary binary counter in conjunction with a binary-to-Gray code converter (N-1 XOR gates). The converter output must also be registered to prevent glitches in the binary counter from appearing as glitches in the Gray code outputs.



11.18

Inputs			Partial Product	Booth Selects		
x_{2i+1}	x_{2i}	x_{2i-1}	PP_i	POS	NEG_i	$DOUBLE$
0	0	0	0	0	0	0
0	0	1	Y	1	0	0
0	1	0	Y	1	0	0
0	1	1	$2Y$	1	0	1

1	0	0	–2*Y*	0	1	1
1	0	1	–*Y*	0	1	0
1	1	0	–*Y*	0	1	0
1	1	1	–0 (= 0)	0	0	0

$\text{POS} = \overline{x}_{2i+1}(x_{2i} + x_{2i-1})$; $\text{NEG} = x_{2i+1}(\overline{x}_{2i} + \overline{x}_{2i-1})$;

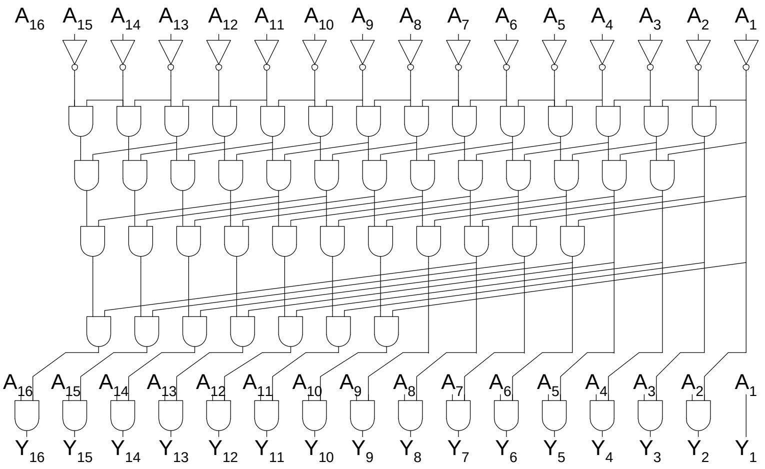
$\text{DOUBLE} = \overline{x}_{2i+1}x_{2i}\, x_{2i-1} + x_{2i+1}\overline{x}_{2i}\overline{x}_{2i-1}$

$\text{PP}_{ij} = (y_j\text{POS} + \overline{y}_j\text{NEG})\overline{\text{DOUBLE}} + (y_{j-1}\text{POS} + \overline{y}_{j-1}\text{NEG})\text{DOUBLE}$

11.19 X0, X1, and X2 indicate exactly zero, one, or two 1’s in a group. Y1, Y2, and Y3 are one-hot vectors indicating the first, second, and third 1.

$$
\begin{aligned}
X0_{i:i} &= \overline{A_i} \\
X1_{i:i} &= A_i && \text{bitwise precomputation} \\
X2_{i:i} &= 0 \\
X0_{i:j} &= X0_{i:k} \bullet X0_{k-1:j} \\
X1_{i:j} &= X1_{i:k} \bullet X0_{k-1:j} + X0_{i:k} \bullet X1_{k-1:j} && \text{group logic} \\
X2_{i:j} &= X1_{i:k} \bullet X1_{k-1:j} + X2_{i:k} \bullet X0_{k-1:j} + X0_{i:k} \bullet X2_{k-1:j} \\
Y1_i &= A_i X0_{i-1:1} \\
Y2_i &= A_i X1_{i-1:1} && \text{output logic} \\
Y3_i &= A_i X2_{i-1:1}
\end{aligned}
$$

11.20


A16 A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1
A16 A15 A14 A13 A12 A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1
Y16 Y15 Y14 Y13 Y12 Y11 Y10 Y9 Y8 Y7 Y6 Y5 Y4 Y3 Y2 Y1

11.21 Assume the branching effort on each A input is approximate 2 because it drives two gates (the initial inverter and the final AND). A path from input to output passes through an inverter and five AND gates, each made from a NAND and an inverter. There are four two-way branches within the network. Hence, $B = 32$. $G = 1^6 * (4/3)^5 = 4.2$. $H = 1$. $P = 1*6 + 2*5 = 16$. $F = GBH = 135$. $N = 11$. $f = F^{1/N} = 1.56$. $D = Nf + P = 33.2\ \tau$. Note that the stage effort is lower than that desirable for a fast circuit. The circuit might be redesigned with NANDs and NORs in place of ANDs to reduce the number of stages and the delay.

11.22 The following equations are a slight modification of EQ 10.50. Use the base case $X_{1:1} = 1$, $W_{1:1} = 0$.

$$\begin{aligned}
X_{i:i} &= \overline{A_i \overline{A}_{i-1}} \\
W_{i:i} &= A_i \overline{A}_{i-1} && \text{bitwise precomputation} \\
X_{i:j} &= X_{i:k} \bullet X_{k-1:j} \\
W_{i:j} &= W_{i:k} \bullet X_{k-1:j} + X_{i:k} \bullet W_{k-1:j} && \text{group logic} \\
Y_i &= W_{i:i} \bullet W_{i-1:1} && \text{output logic}
\end{aligned}$$

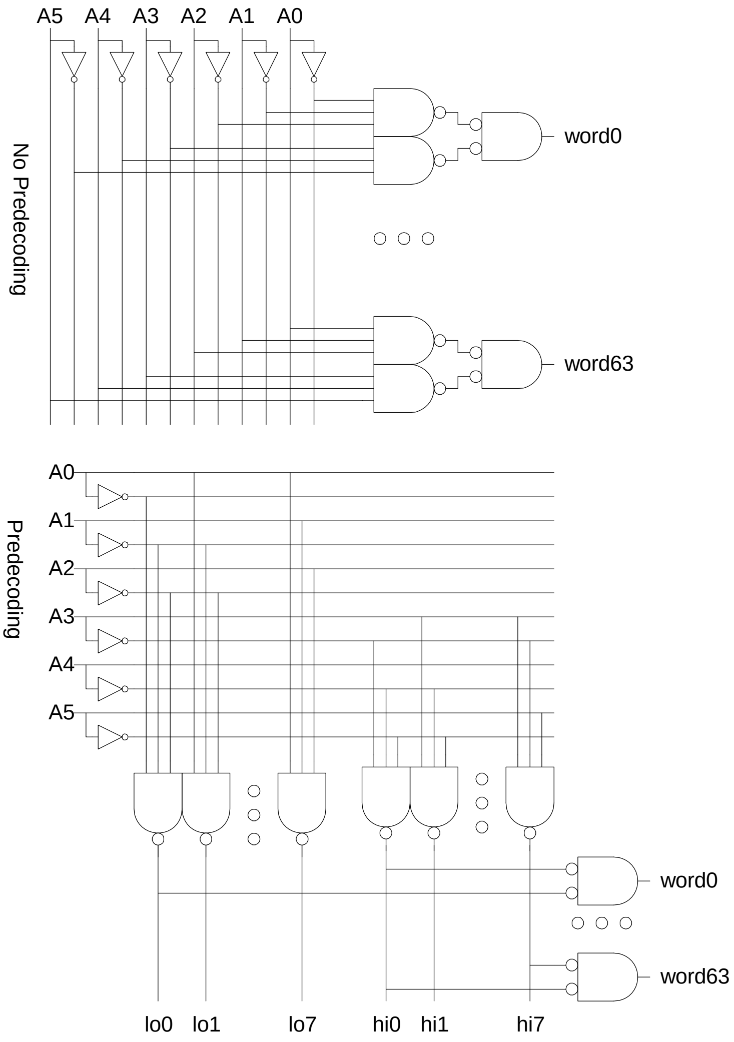
11.23 Open-ended problem. See [Burgess09] for one implementation.

Chapter 12

12.1 If the array is organized as 128 rows by 128 columns, each column multiplexer must choose among (128/8) = 16 inputs.

12.2 The dimensions are (128 columns * 1.3 µm/col* 1.1) x (128 rows * 1.44 µm/row* 1.1) = 183 µm x 203 µm.

12.3 The design with predecoding uses 16 3-input NANDs while the design without uses 128. Both designs have the same path effort. Hence, the layout of the predecoded design tends to be more convenient.



12.4 The logical effort is 1.5 per wordline output, as compared to 7/3 for an ordinary NOR3. However, the parasitic delay is large for a falling transition because the big capacitances between the pMOS transistors must be discharged through the nMOS transistors.

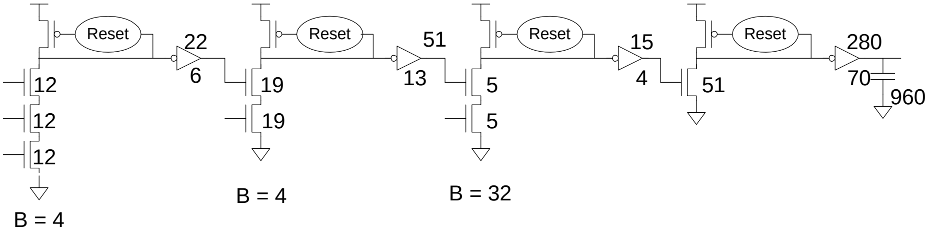
12.5 (a) B = 512. H = 20. A 10-input NAND gate has a logical effort of 12/3, so estimate that the path logical effort is about 4. Hence $F = GBH$ = 40960. The best number of stages is $\log_4 F$ = 7.66, so try an 8-stage design: NAND3-INV-NAND2-INV-NAND2-INV-INV-INV. This design has an actual logical effort of G = (5/3)

* (4/3) * (4/3) = 2.96, so the actual path effort is 30340. The path parasitic delay is $P = 3 + 1 + 2 + 1 + 2 + 1 + 1 + 1 = 12$. $D = NF^{1/N} + P = 41.1\ \tau$.

(b) The best number of stages for a domino path is typically comparable to the best number for a static path because both the best stage effort and the path effort decrease for domino. Using the same design, the footless domino path has a path logical effort of $G = 1 * (5/6) * (2/3) * (5/6) * (2/3) * (5/6) * (1/3) * (5/6) = 0.071$ and a path effort of $F = 732$. The path parasitic delay is $P = 4/3 + 5/6 + 3/3 + 5/6 + 3/3 + 5/6 + 1/3 + 5/6 = 7$. $D = NF^{1/N} + P = 25.2\ \tau$.

12.6 The function consists of 1024 10-input AND gates, each built with four levels of noninverting logic (AND3 - AND2 - AND2 - BUF). Fanning out directly to 1024/2 gates for both true and complementary inputs would exceed any reasonable input capacitance specificaiton. Instead, let's do 1-of-32 predecoding, so each predecoded line drives 32 2-input AND gates. At the first level, predecode two groups of three bits into two 1-of-8 hot bundles and two groups of two bits into two 1-of-4 hot bundles. The slowest predecoder is the 1-of-8 hot, which involves sending each true or complementary input in the group of three to four 3-input AND gates. At the second stage, use 2-input AND gates to predecode the 1-of-8 hot and 1-of-4 hot bundles into 1-of-32 hot bundles. The critical path from the 1-of-8 hot output passes through four 2-input AND gates. Each of the 1-of-32-hot predecoded signals drives 32 2-input ANDs at the 3rd level, and a buffer at the fourth level. The load is 48H = 960 λ.

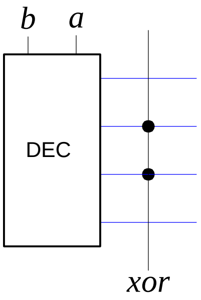
The path can be viewed as AND3 - AND2 - AND2 - BUF with branching efforts of 4, 4, 32, and 1. Assume the reset chain consists of three inverters that present a negligible load to the forward path (this is only true if the transistors on the forward path are rather large). From the previous problem, $F = 610$, so $f = F^{1/8} = 2.23$. Working from the load back, the gate sizes are 960 * (5/6) / f = 350, 350*(1/3) / f = 51, 51 * (5/6) / f = 19, 19 * (2/3) / f = 5.5, 5.5 * 32 * (5/6) / f = 64, 64 * (2/3) / f = 19, 19 * 4 * (5/6) / f = 27, and 27 * 1 / f = 12. Hence, the input capacitance specification is satisfied. These sizes are divided among the transistors as shown below and rounded. The delay should match that predicted in the previous problem, 25.2 τ.



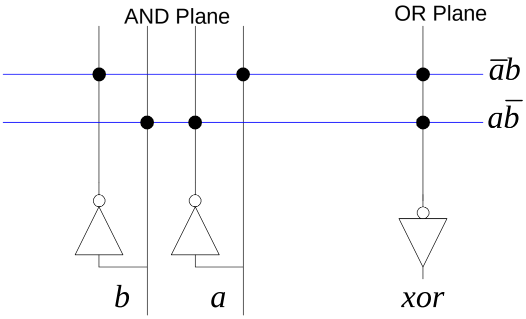
12.7 $H = 2^m$. $B = 2^{n-1}$ because each input affects half the rows. For a conservative estimate, assume that the decoder consists of an n-input NAND gate followed by a string of inverters. The path logical effort is thus G = (n+2)/3, so the path effort is $F = GBH = 2^{n+m}(n+2)/6$. The best number of stages is $N = \log_4 F \sim (n+m)/2$. The parasitic delay of the n-input NAND and N-1 inverters is $P = n + (N\text{-}1)$. Hence, the path delay can be estimated as D = ((n+m)/2) (2^{n+m}(n+2)/6)^(2/(n+m)) + n + (N-1)

12.8 In an open bitline, the sense amplifer compares the voltage on a bitline from the active subarray to the voltage on a bitline from a quiescent subarray. Power supply noise between subarrays makes sensing a small swing impossible. In a closed bitline, the two sense amplifier inputs come from bitlines in the same subarray, but only one of the two is activated. This design requires somewhat more layout area but eliminates most supply noise problems. It is still sensitive to coupling that affects one sense amplifier input more than the other. Twisted bitlines route the folded bitlines in such a way that each one sees exactly the same coupling capacitances, hence making coupling noise common mode as well. This is necessary in modern DRAM designs and costs slightly more area to perform the twists.

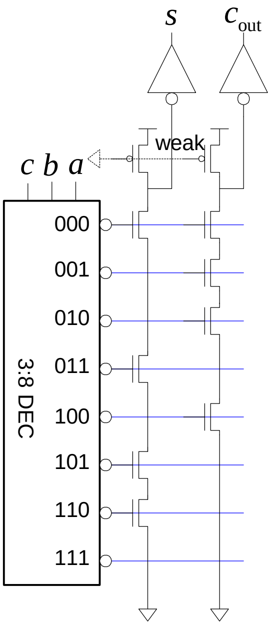
12.9



12.10



12.11



12.12 NAND ROMs use series rather than parallel transistors and one-cold rather than one-hot wordlines. They tend to be smaller than NOR ROMs because they do not require contacts between the series transistors, but they are also slower because of the series transistors.

12.13 The ROM cell is smaller than the SRAM cell. It presents one unit of capacitance for the transistor. It has only a single transistor in the pulldown path on the bitline so the resistance is R. Hence, the logical effort is 1/3, as compared to 2 for the SRAM cell.

The bitline has a capacitance of C/2 from the half contact so the total bitline capacitance is 2^{n-1} C. Because the cell has a resistance R, the delay is 2^{n-1} RC and the parasitic delay is $2^n/6$.

The ROM can use the same decoder as the SRAM, with a logical effort of $(n+2)/3$ and parasitic delay of n. Assume the bitline drives a load equal to that seen by the address so the path electrical effort is $H = 1$.

Putting this all together, the path effort is $F = GBH = 2^N(n+2)/9$. The path parasitic delay is $n + 2^n/6$. The path delay is $D = 2N + 4\log_4[(n+2)/9] + n + 2^n/6$.

Your modeling and loading assumptions may vary somewhat. The assumptions about wire capacitance have a large effect on the model.

Chapter 13

13.1 P_{max} = (110-50) / (10 + 2) = 5 W.

13.2 During an ESD event, large currents may flow into the input protection diodes. The diode to ground is an n-diffusion region in the p-type grounded substrate. Large currents flowing through this diode will locally raise the substrate voltage because of its series resistance to a substate tap. This forward-biases the parasitic npn transistor formed by the n-diffusion, the p-substrate, and any nearby n-well, as discussed in Section 4.8.5. Positive feedback leads to latchup. The problem is prevented with generous substrate contacts forming a guard ring around the protection diode.

13.3 H-trees ideally have zero skew and relatively low metal resource requirements, but in practice see significant skews, even locally, because of mismatches in loading, processing, and environment among the branches. Clock grids have low local skew because they short together nearby points, but can have large global skew and require lots of metal and associated capacitance. The hybrid tree/grid achieves low local skew because of the shorting without using as much metal as a full clock grid.

Chapter 14

14.1 If we summarize the attributes we need for a control RAM cell for an FPGA, we would like it to be small. In addition, as the RAM cells are dispersed across the chip, it probably would be advisable to design a cell with the lowest wiring overhead. Finally, we want a circuit that is robust and easy to use in an FPGA.

A conventional RAM cell has a write line, a read line and data and complement data lines. Data is read or written using the data lines. To read the RAM cell, fairly complicated sense amplifiers are required and there is normally a complicated precharge and timing sequence required (Section 11.2.1). We would prefer a RAM cell that operated with full logic levels.

A single-ended RAM cell that is often used as a register cell is probably the best choice. A typical circuit is shown in Figure 7.17j. This circuit has a single ports for data-in, data-out, write and read. In addition, all signals are full logic levels with the exception of the data-out signal which has to be held high with a pMOS load (or precharged and then read). This is probably OK as the global read operation is only used for testing or to infrequently read out the control RAM contents. It does not have to be fast.

Design starts with the write operation. The switching point of the “input” inverter is a balance between the write zero and one operations. This is achieved by using a single nMOS pass transistor to overwrite a pair of asymmetric inverters. When trying to write a zero, the driving inverter n-transistor and the memory cell write n-transistor have to overcome the p-transistor pullup of the feedback inverter in the memory cell. The circuit is shown below. We can arbitrarily size the weak-feedback inverter so that the pull down circuit triggers the input inverter.

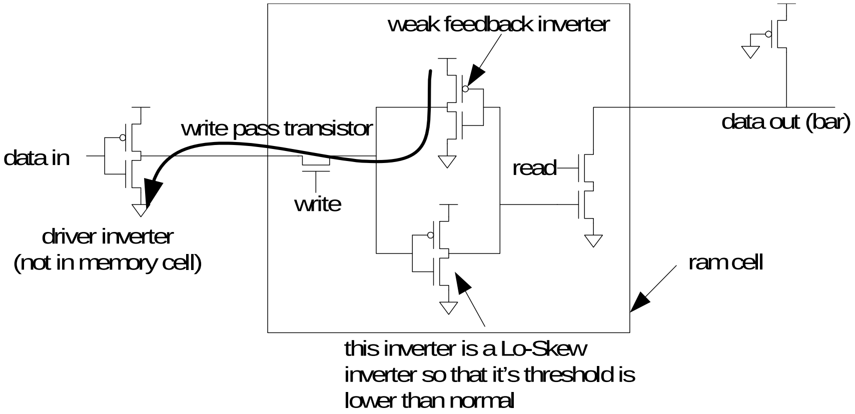


Figure E8.2 – Write Zero operation for single-ended RAM cell

Writing a one is somewhat constrained by the fact that the write n-transistor can only pull up to a threshold below V_{DD} (V_{DD}-V_{tn}). This means that the trip point for the RAM inverter has to be set well below this. This is achieved by having a LO-skewed inverter (Section 2.5.2). This involves sizing the n-transistor in the inverter up until the input switch point is comfortably below the V_{DD}-V_{tn} voltage.

Once the cell can be written, the read operation may be considered. If we use a pMOS load in what is effectively a two input pseudo-nMOS NAND gate or one leg of a multiplexer, the n-transistor pull-downs have to be able to pull the output to near zero when both transistors are turned on. Assuming the pulldown n-transistors are minimum size, this involves lengthening the pMOS pullup until acceptable operation over voltage, temperature, and process is achieved.

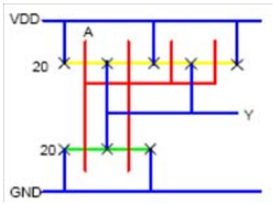
14.2 FPGA routing blocks cascade the switches used in routing blocks. Therefore every routing block that a "wire" passes through adds the delay of the switch (Figure 8.23a and Figure 8.24). Transmission gates can reduce the delay for small numbers of cascaded transmission gates but the delay builds as a square law (The square law increase in delay mentioned in Section 4.6.4 also applies here.) A tristate inverter costs a load dependent delay per stage (rather like the repeaters mentioned in Section 4.6.4). So the tradeoff involves determining which on average one uses. Or perhaps a mix of styles might be used. [Footnote: An FPGA company (now defunct) entered the FPGA market based on the fact that they had a patent on using a tristate inverter as a routing switch. It was reasoned that as processors got faster and routing (real wires) got slower, this would be a market edge.]

14.3 Using Equation 8.7, the (yielded) gross die per wafer for the first process is 1500 (1914*.8*.98) and the die cost is $1.47. For the scaled process there are 2227 yielded die (2841*.8*.98) which cost $1.35. So it is probably worth moving considering that the yield probably improves as well (smaller die).

14.4 An XOR gate is a 1-bit multiplier. A basic FIR filter is the sum of products of the delayed samples of the input and the coefficients. So for a 288 tap FIR we would have Y(t) = h_1*X(t) + h_2*X(t+1) + … +h_{288}*X(t+287).

One way to buil this is to use a 288 by 1-bit serial shift register. The output of each register would be connected to one input of a 2 input XOR gate. The other XOR input would be connected to the coefficient. The outputs of the 288 XOR gates then have to be summed. This can be done with a tree of adders (see Chapter 10). We can implement the adder on terms of 3:2 compressors (just a full adder). So 288 bits compress with the first rank of adders to 192 to 128 to 86 to 58 to 39 to 26 to 18, at which point a parallel carry add would be completed.

14.5 The order of contacts affects the parasitic delay of the gate. For example, if the GND wire were contacted to the middle of the nMOS pair and the Y wires to the outside, there would be twice as much n-diffusion capacitance.



14.6 No solution available.

14.7 No solution available. This problem seems to be missing the defect density or yield.

Chapter 15

15.1 Cooling a circuit improves the mobility of the transistors which in turn improves the speed. Raising V_{DD} has the same effect. These two tests together probably point to a path that is too slow at normal temperature and voltage. Re-simulating the path ensuring to include all parasitics (at especially the slow process corner), should reveal the problem.

15.2 If we take the parameters of Exercise 8.13, a 10 mm * 10 mm die costs $10.05 assuming an 8" wafer, 80% die yield (high!!) and 98% package yield. The package cost is significant at $5. So it is probably best to test at the wafer level.

15.3 Absolutely not! Any discrepancy between a golden model and the design should be tracked down and explained and eliminated. Often small deviations hide much larger problems.

15.4 This is straight from the text – Section 9.5.1.1. A Stuck-at-1 means that a node is shorted to V_{DD}. A Stuck-at-0 means a node is shorted to GND.

15.5 Again straight from the text (pp 590). Figure 9.10 is an example.

15.6 Wires can touch (especially if there is a dust particle over two wires or separation rules are broken) – this leads to shorts.

Wires can neck down (say due to overetching) – this can lead to opens.

Contacts or vias can be faulty leading to opens.

Gate oxide can have pin holes that leads to shorts.

15.7 Right out of the text. Controllability – Section 9.5.3. Observability – Section 9.5.2. Fault Coverage – Section 9.5.4

15.8 A high fault coverage means that the set of test vectors is effectively capable of finding as many faults as possible. Testing costs money, so the smaller a test set is while being effective, the lower the cost of the chip.

15.9 Another question straight out of the book (these are too easy…). Section 9.6.2. Basically, a scan design is implemented by turning all D flip-flops into scannable D flip-flops. This usually involves adding a two input multiplexer to the existing D flip-flop designs that are used (this isn't done manually, but using library elements).

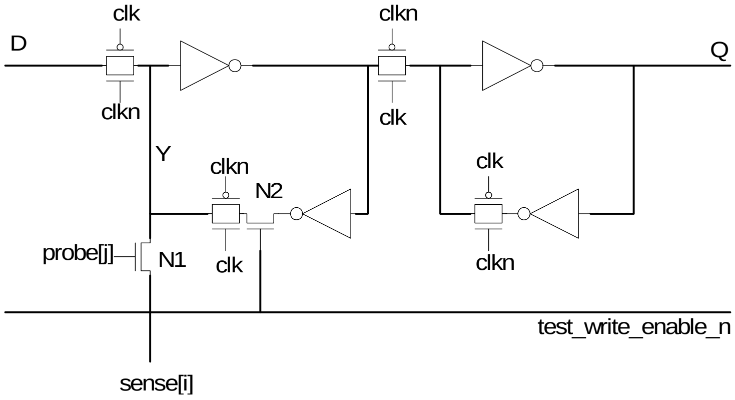
Once scan flip flops are inserted, the task remains to divide the flip-flops into scan chains.

15.10 BIST builds on scan by surrounding logic blocks with a pseudo random sequence generator on the logic inputs and a scan chain on the output of the logic. These two functions (in addition to the normal operation of the flip flops) may be combined into the one structure (Figure 9.23)). BIST can reduce the number of external test vectors needed if it is compatible with the system test methodology. It can also perform high-speed testing with a low-speed tester. It costs area on chip.

15.11 The point that is trying to be illustrated here is that there are some areas where we do not want to encumber a flip-flop with extra circuitry. This is the case for high speed flip-flips used in dividers (irregardless of circuit design). So no scan elements. Just test by observing the frequency of the MSB of the counter (lowest frequency) with a frequency counter. This is more classed as an analog block.

15.12 This register was featured in the second edition.

Transistors N1 and N2 are added to a regular static D flip-flop. Transistor N2 is used to prevent the master stage of the D flip-flop from writing. Setting signal probe[j] allows node Y to be read or written via signal sense[i]. If test_write_enable_n is true, the cell is read. If test_write_enable_n is false, the cell may be written (providing the D flip-flop master inverter is LO-skewed). Be careful of the single nMOS pass-gates because they do not pass rail-to-rail outputs



15.13 Essentially, this is a slice through Figure 9.24. The 16-bit datapath has a 16-bit LFSR on the input and a 16 bit signature analyser on the output. The sequence to test is as follows:

Initialize LFSR (i.e. set flip flops to all ones)

Place signature analyzer in “analyze” mode

Cycle LFSR through a “large” number of vectors – can be exhaustive.

Shift signature analyzer out and observe syndrome – check whether it matches the simulated value. If it does your circuit is OK, if not, it’s faulty.

15.14 The data input, address and control (read/write controls and clocks) are muxed with test generators. The test structure for the address can be a counter. The data generator can be a simple logic structure that generates “all 0’s”, “all 1’s” and “alternating 1’s and 0’s”. The control generator generate a simple control sequence. A comparator compares the RAM data with what is expected. Typical operation might be as follows:

Stage 1: Write Data

Set data generator to “all 0’s”

Loop Counter through address range and write data to RAM

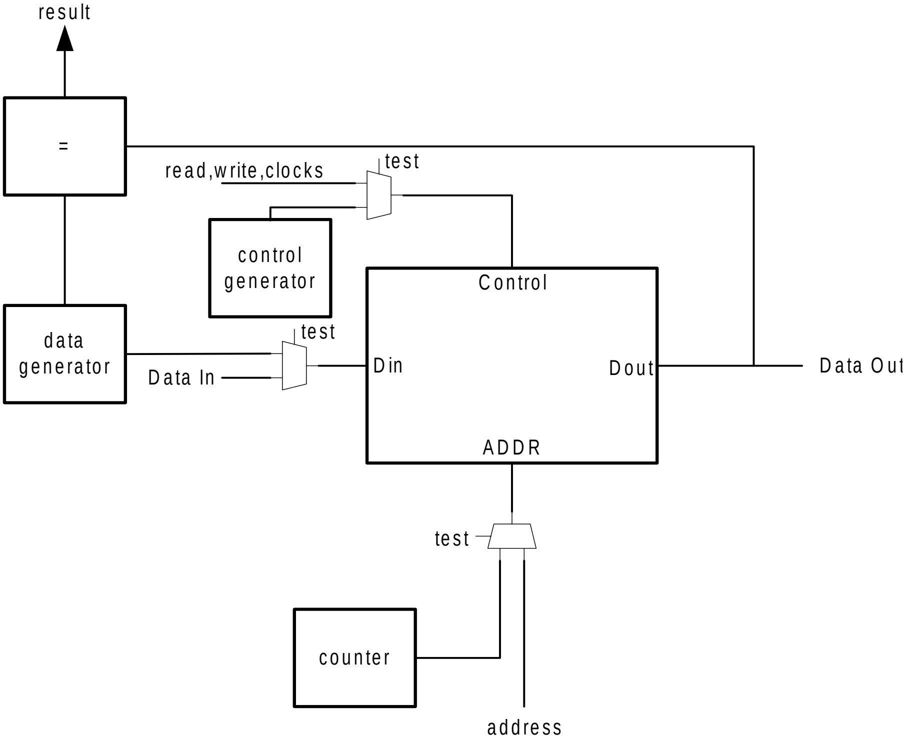
Stage 2: Check RAM

Set data generator to “all 0’s”

Loop Counter through address range and read RAM

Check RAM output at each step

The same would be done for “all 1’s” and “alternating 1’s and 0’s”.



15.15 According to Wikipedia, a shmoo is a fictional cartoon caracter created by Al Capp in a 1948 issue of Li’l Abner. The test plots may have received their name because they resembled shmoos. See Baker and van Beers, “Shmoo Plotting: The Black Art of IC Testing,” *IEEE Design and Test of Computers*, vol. 14, no. 3, 1997, p. 90-97.

Appendix A

No solutions presently available.