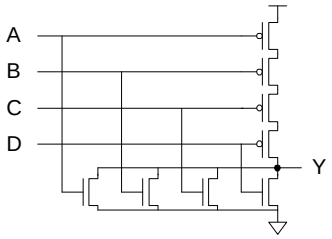
Solutions

Note: Version 2.0 completed 12/23/04. Includes solutions to nearly all problems. Thanks to Ted Jiang for generating many of the SPICE simulations.

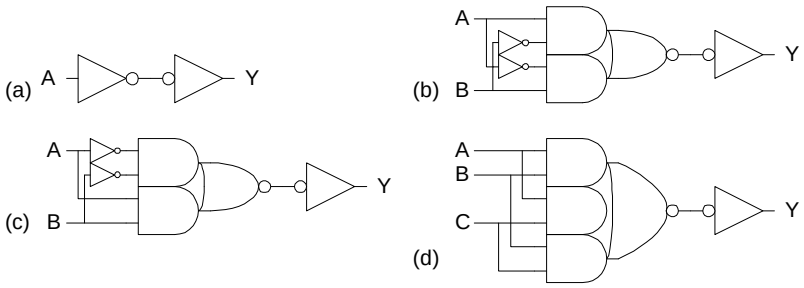
Chapter 1

1.1 Starting with 42,000,000 transistors in 2000 and doubling every 26 months for 10 years gives $42\text{M} \bullet 2^{\left(\frac{10 \cdot 12}{26}\right)} \approx 1\text{B}$ transistors.

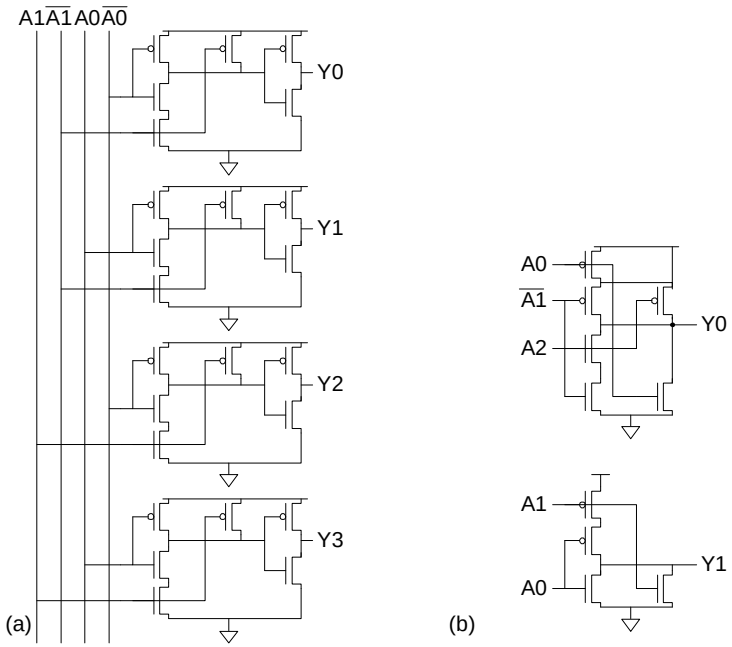
1.3



1.5

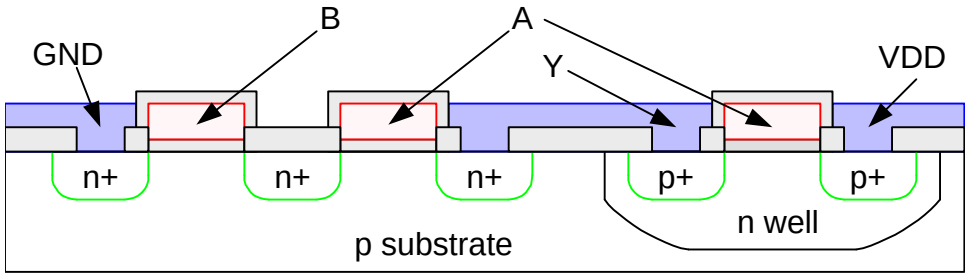


1.7



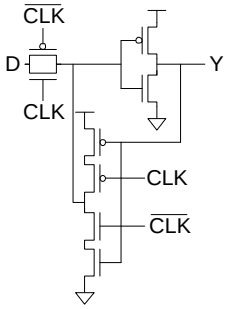
1.9 The minimum area is 5 tracks by 5 tracks (40 λ x 40 λ = 1600 λ^2).

1.11

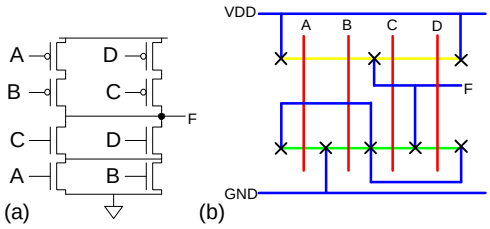


1.13 This latch is nearly identical save that the inverter and transmission gate feedback

has been replaced by a tristate feedaback gate.



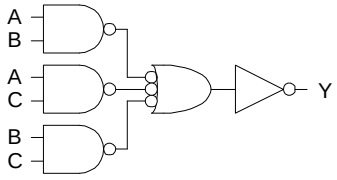
1.15



(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.

1.17 20 transistors, vs. 10 in 1.16(a).

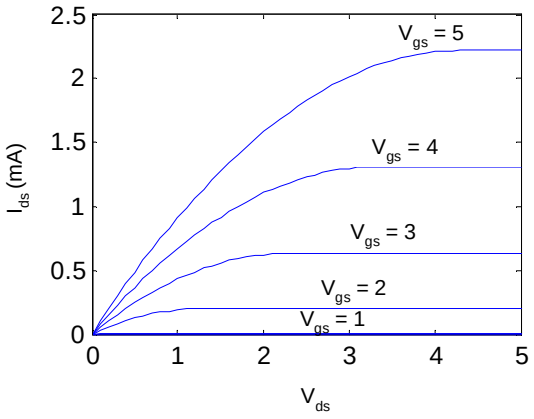


1.19 The lab solutions are available to instructors on the web.

Chapter 2

2.1

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



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 $I_{DS1} > I_{DS2}$.

2.5 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

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

At a drain voltage of VDD, the capacitance reduces to

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

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

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

(c) $v_T = kT/q = 34$ mV; $\dfrac{e^{\frac{-0.3}{1.4 \bullet 0.034}}}{e^{\frac{-0.4}{1.4 \bullet 0.034}}} = 8.2$; 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

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

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

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

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

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

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

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

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.

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

2.17 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.

2.19 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.

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

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. 157nm = 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 147nm is $40M/700,800 = $57.08. For a run through the equipment 10 times per completed wafer is $35.60 and $570.77 respectively.

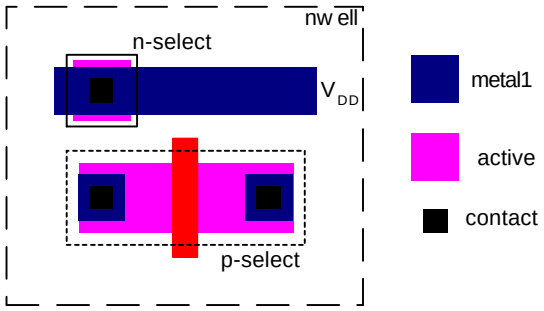
Now for gross die per wafer. For a 300mm diameter wafer the area is roughly 70,650 mm^2 ($\pi*(r^2/A - r/(\text{sqrt}(2*A))$). For a $50mm^2$ die in 90nm, there are 1366 gross die per wafer. Now for the tricky part (which was unspecified in the question and could cause confusion). What is the area of 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.

3.3 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).

3.5 This question is poorly worded. The metals that were intended were silver and gold.



(This information isn't in the book. The student would have to do a bit of web searching.) Silver has better conductivity than copper and gold while having poorer conductivity than copper, has good immunity to oxidization. The reason for not using gold or silver is that they both have the property that they can migrate and enter the silicon. This alters CMOS device characteristics in undesirable ways. This question should probably be reworded in any new printing.

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.

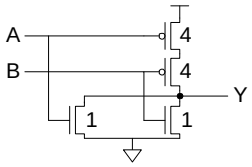
The reason for this problem is to show that there is an appreciable difference in gate spacing (and therefore source/drain parasitics) between contacted source and drains and the case where you can eliminate the contact (e.g. in NAND structures). In the main this may not be important but if you were 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.

3.9 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.

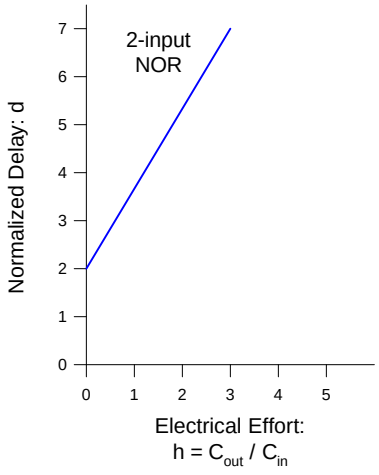
Chapter 4

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.



4.3 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.

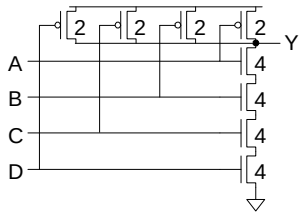
4.5 The slope (logical effort) is 5/3 rather than 4/3. The y-intercept (parasitic delay) is identical, at 2.



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 $2p_{inv}$, but this is still a substantial speedup.

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



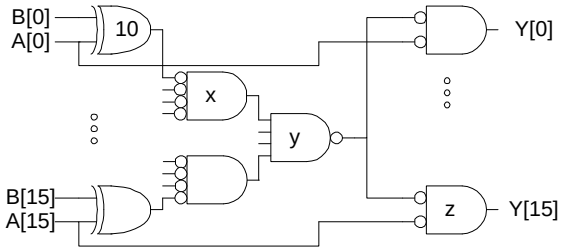
4.11 $D = N(GH)^{1/N} + P$. Compare in a spreadsheet. Design (b) is fastest for $H = 1$ or 5. Design (d) is fastest for $H = 20$ because it has a lower logical effort and more stages to drive the large path effort. (c) is always worse than (b) because it has greater logical effort, all else being equal.

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

4.13 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; $x = y * (9/3) / 5.03 = 12.6$.



4.15 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.

4.17 $g = 1.47$, $p = 3.08$. 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.

4.19 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.

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

4.23 The adder delay is 6.6 FO4 inverter delays, or about 133 ps in the 70 nm process.

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].

4.27 $P = aCV^2f = 0.1 * (150e^{-12} * 70) * (0.9)^2 * 450e^6 = 0.38$ W.

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

(a)

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

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

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

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

(b) Increasing η increases I_1 because the threshold is effectively reduced. The current change is 31.9-fold. DIBL is very important for subthreshold leakage

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

$$\frac{I_1(h = 0.05)}{I_1(h = 0)} = e^{\frac{hV_{DD}}{v_T}} = 31.9$$

(c) Increasing η increases I_2 because the threshold is effectively reduced. However, the relative increase is less than that of I_1. Solve numerically for the change in I_2 to be a factor of 2.25, with x = 83 mV. DIBL has much less effect on stacked devices, so the relative leakage current of two series devices is much less than half of a single one when DIBL is pronounced.

(d) First solve for x

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

(The last term is approximately unity)

$$= I_1 e^{\frac{hx}{v_T}} \left[1 - e^{\frac{-x}{v_T}}\right] \approx I_1 e^{\frac{-x}{v_T}} e^{\frac{h(V_{DD} - x)}{v_T}}$$

$$1 = e^{\frac{-x}{v_T}} \left[e^{\frac{h(V_{DD} - 2x)}{v_T}} + 1\right]$$

(Assume $x << V_{DD}$)

$$x = v_T \ln\left(1 + e^{\Delta}\right) \approx \Delta v_T$$

Then substitute x to find the relative currents in stacked vs. unstacked transistors:

$$I_2' = I_1 e^{\frac{hx}{v_T}} \left[1 - e^{\frac{-x}{v_T}}\right] = I_1 e^{h\Delta} \left[1 - e^{-\Delta}\right]$$

$$\frac{I_2'}{I_2} \approx 2e^{h\Delta}$$

$$\frac{I_2'}{I_1'} \approx e^{(h-1)\Delta}$$

(e) When DIBL is significant, we see from (d) that the current in stacked transistors is exponentially smaller because the bottom transistor sees a small drain voltage and thus much less DIBL than a single transistor.

4.31 (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.

4.33 The Elmore delay of each segment is

$$t_{pd-seg} = \frac{R}{W}\left(\frac{C_w l}{N} + C'W\right) + \left(\frac{R_w l}{N}\right)\left(\frac{C_w l}{2N} + C'W\right)$$

The total delay is N times greater:

$$t_{pd} = NRC' + l\left(R_w C'W + \frac{RC_w}{W}\right) + l^2 \frac{R_w C_w}{2N}$$

Take the partial derivatives with respect to N and W and set them to 0 to minimize delay:

$$\frac{\partial t_{pd}}{\partial N} = RC' - l^2 \frac{R_w C_w}{2N^2} = 0 \Rightarrow N = l\sqrt{\frac{R_w C_w}{2RC'}}$$

$$\frac{\partial t_{pd}}{\partial W} = l\left(R_w C' - \frac{RC_w}{W^2}\right) = 0 \Rightarrow W = \sqrt{\frac{RC_w}{R_w C'}}$$

Using these gives a delay per unit length of

$$\frac{t_{pd}}{l} = \left(2 + \sqrt{2}\right)\sqrt{RC'R_w C_w}$$

4.35 Compute the results with a spreadsheet:

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

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**

4.37 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.

Chapter 5

5.1 t_{pd} = 107 ps. Note that according to Table 5.8, one would expect a FO5 delay of 120 ps. However, Table 5.8 was generated using P/N = 32/16 λ. In this process, the smaller 8/4 devices appear to have a slightly shorter delay on account of second-order effects.

```
* 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
```

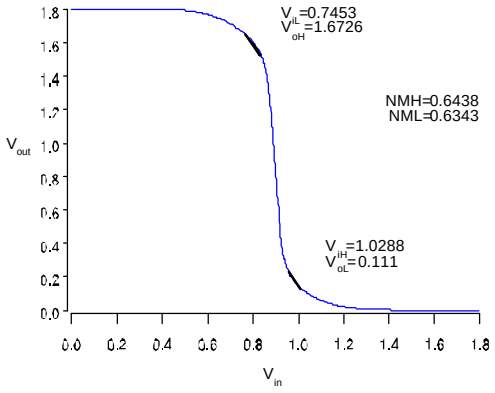
```
*********************************************************************
* Stimulus
*********************************************************************
.tran 1ps 1000ps
.measure tpdr                        * rising propagation delay
+     TRIG v(c)       VAL='SUPPLY/2' FALL=1
+     TARG v(d)     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

5.3 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
```

5.5 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.



5.7 Your results will vary with your process.

5.9 $g = 1.79$, $p = 6.53$

```
# 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
```

5.11 Your results will vary with your design.

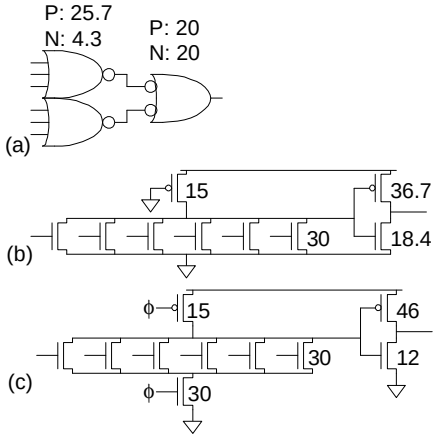
Chapter 6

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

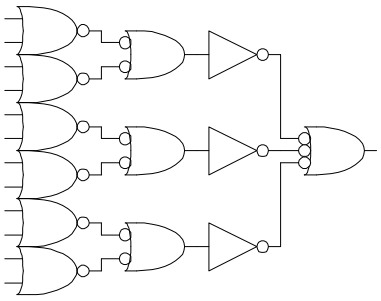
(a) NOR3 ($p = 3$) + NAND2 ($p = 2$). $G = (7/3)*(4/3) = 28/9$. $F = GBH = 28/3$. $f = F^{1/2} = 3.05$. Second stage size = $90*(4/3)/f = 39$. $D = 2f + P = 11.1$.

(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 = $90*(5/6)/f = 58$. $D = 7.75$.



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



6.5 (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

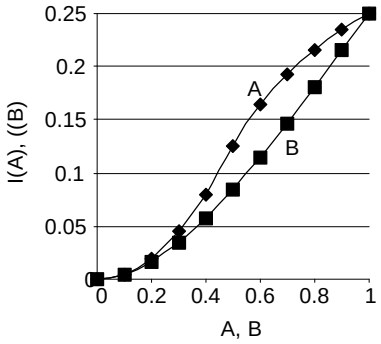
$$I(B) = \tfrac{1}{2}(B - x)^2 = \left(1 - \tfrac{x}{2}\right)x$$

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.

6.7 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
```

6.9 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.

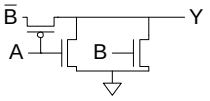
6.11 HI-skew: p*MOS* = 2, nMOS = sk, $g_u = (2 + ks)/3$, $g_d = (2 + ks)/3s$, $g_{avg} = (2 + k + ks + 2/s)/6$

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

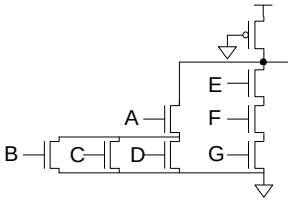
6.13 Suppose a P/N ratio of k gives equal rise and fall times. If the pMOS device is of width p and the nMOS of width 1, then we find

6.15 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.

6.17 The 3-transistor NOR is nonrestoring.



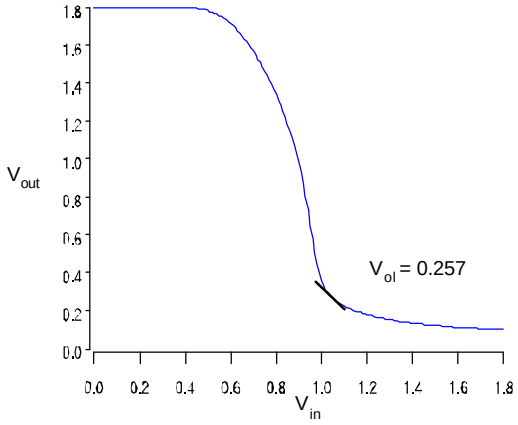
6.19



6.21 $g_d = 0.77$, $g_u = 0.76$, $g_{avg} = 0.76$; $p_d = 0.71$, $p_u = 1.13$, $p_{avg} = 0.92$

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
```

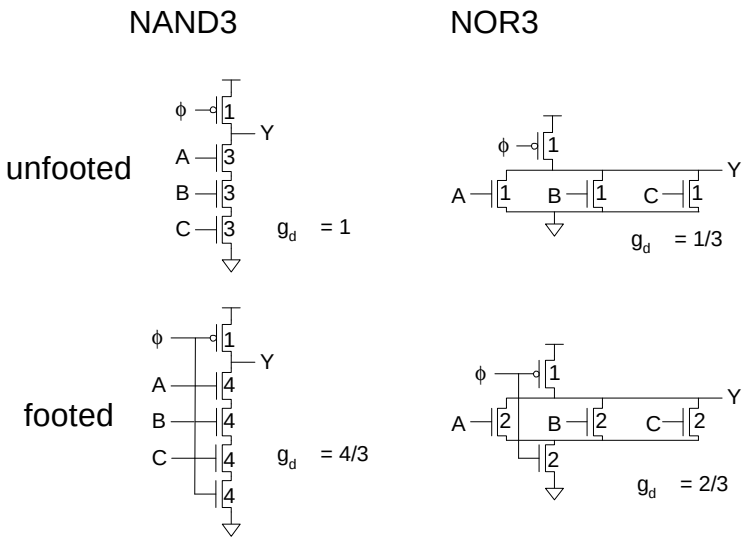
6.23 The average logical effort is 5/6, substantially better than 7/3 for a static CMOS NOR3.

6.25 Simulating the various gates gave the following average propagation delays (in ps).

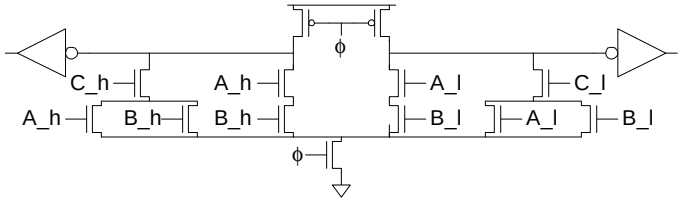
This is a bit surprising and indicates SFPL may be advantageous for wide NORs..

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

6.27

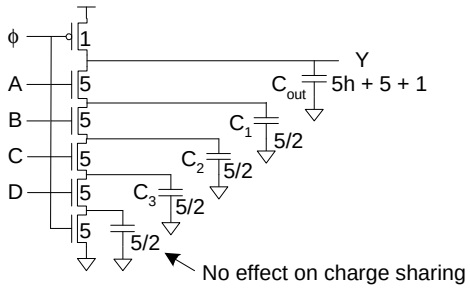


6.29



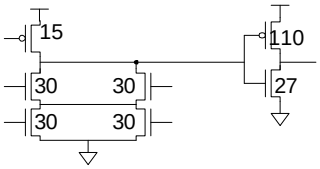
6.31 The worst case is when *A* is low on one cycle, *B*, *C*, and *D* are high, and all the internal nodes become predischarged to 0. Then *D* falls low during precharge. Then *A*

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}.

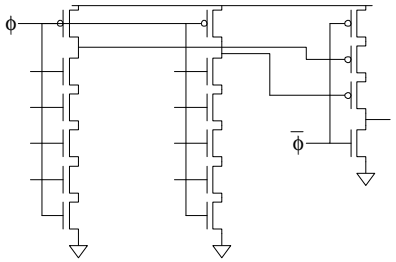


6.33 With a secondary precharge transistor, one of the internal nodes is guaranteed to be high rather than low. Thus 11 + 2.5 = 13.5 units of capcitance are high and 5 units are low, reducing the charge sharing noise to 13.5 / (13.5 + 5) V_{DD} = 0.73 V_{DD}.

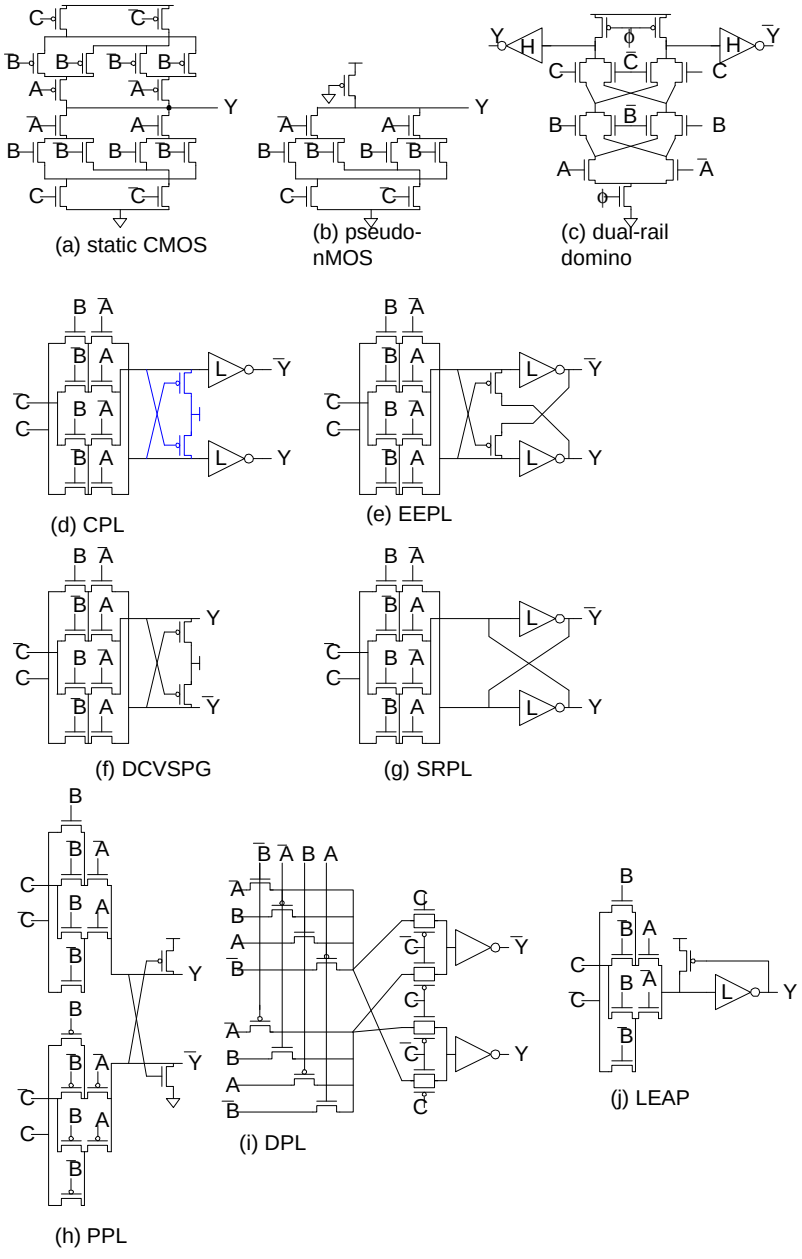
6.35 H = 500 / 30 = 16.7. Consider a two stage design: footless dynamic OR-OR-AND-INVERT + HI-skew INV. G = 2/3 * 5/6 = 10/18. P = 5/3 + 5/6 = 5/2. F = GBH = 9.3. f = $F^{1/2}$ = 3.0. D = 2f + P = 8.6 τ. The inverter size is 500 * (5/6) / 3.0 = 137.



6.37



6.39



6.41 ### no solution available

6.43 n/a

Chapter 7

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

7.3 (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.

7.5 (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.

7.7 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 $t_{\text{setup}} - t_{\text{pw}} + t_{\text{skew}}$ 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 $t_{pcq} + t_{\text{setup}} - t_{\text{pw}} + t_{\text{skew}}$.

7.9 (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.

7.11 (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.

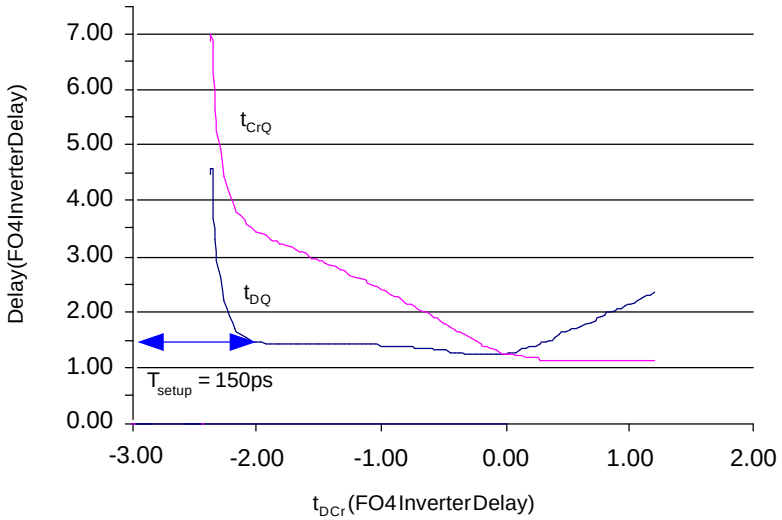
7.13 The t_{pdq} delays are 151 ps for a conventional dynamic latch and 162 ps for a TSPC latch.

```
*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
```

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



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

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

7.21 ### no solution available

7.23 Solve for T_c:

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

7.25 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 8

8.1 Selection of a gate array cell comes down to selecting the number of transistors to place

in series in a cell, remembering that if a cell uses less transistors, then the extra transistors are not utilized. We can categorize individual gates by the number of series transistors they require (i.e. an n-strip below a p-strip, as in Figure 8.28). The following table summarizes the usage in particular chip in the exercise:

Cell Type	Series transistors	Circuit	Percentage
D-latch	5	Figure 7.17f with clock inverter	
D-flipflop	10	Two D-latches	
Scannable D-flipflop	15-16	Allowing for input multiplexer	30
4 input gate	4		10
3 input gate	3		10
2 input gate	2		40
buffers	various		10

Clearly if we were to center on a D-flop, and use a five series transistor cell, at least 60% of the gates would waste transistors (2,3,4 input gates). As an aside, a scannable D-register would need three blocks if they were 5 transistor series blocks.

While we can guess, a little bit of analysis might help. The pitch of a contacted transistor is 8λ (Exercise 3.7). (However see Exercise 8.5 where this gets blown out to 14λ if interior poly-contacts are required. For this exercise we'll stick with 8λ.) A break in the active area adds 3λ (Table 3.2 Rule 2.2). So the pitch of various gate arrays is as follows:

Series Transistors	Pitch	
2	2*8+3	19
3	3*8+3	27
4	4*8+3	35
5	5*8+3	43

We can construct a table as follows that charts usage per 1000 gates. A scannable D flipflip is assumed to use 15 series transistors. We ignore buffers.

5 series transistors

Cell	Total Cells	Area Used	Area Wasted
DFF	3*300=900	900*43	0
4 input gates	100	100*43	100*8
3 input gates	100	100*43	100*16
2 input gates	400	400*43	400*24
Total Area		64500	12000
Percentage Wastage			18.6%

4 series transistors

Cell	Total Cells	Area Used	Area Wasted
DFF	4*300=1200	1200*35	1200*8
4 input gates	100	100*35	0
3 input gates	100	100*35	100*8
2 input gates	400	400*35	400*16
Total Area		63000	16800
Percentage Wastage			26.7%

3 series transistors

Cell	Total Cells	Area Used	Area Wasted
DFF	5*300=1500	1500*27	0

4 input gates	100*2	200*27	200*8
3 input gates	100	100*27	0
2 input gates	400	400*27	400*8
Total Area		59400	4800
Percentage Wastage			8.1%

2 series transistors

Cell	Total Cells	Area Used	Area Wasted
DFF	8*300=2400	2400*19	300*8
4 input gates	100*2	200*19	0
3 input gates	100*2	200*19	200*8
2 input gates	400	400*19	0
Total Area		60800	4000
Percentage Wastage			6.6%

Based on this analysis, a three series transistor gate array looks the lowest area.Note that it does not have the best utilization (lowest area wasted), but three series transistors are denser than two because there is less space between transistors.

If we use a Sea-of-gates structure, the pitch is 8 λ but each gate has an extra series transistor (in general) to isolate the gate. This the table looks as follows:

Sea-of-gates

Cell	Total Cells	Area Used	Area Wasted
DFF	300*16 = 4800	4800*8 = 38400	300*8=2400
4 input gates	100*5	500*8=4000	100*8 = 800
3 input gates	100*4	400*8=3200	100*8 = 800

2 input gates	400*3	1200*8=9600	400*8 = 3200
Total Area		55200	7200
Percentage Wastage			13%

This is close to the 3-input case, but takes the guesswork out of estimating gate mixes. This is the reason SOG is widely used. In the end, this problem is looking for some reasoning why one cell size is better than another. Any well reasoned argument is probably acceptable.

8.3 [Admittedly, this exercise may require some knowledge of Chapter 11 although the design we use has been presented in Chapter 7. A detailed design is possible with a simulator and process files, but it's OK to just capture the basic principles.]

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.

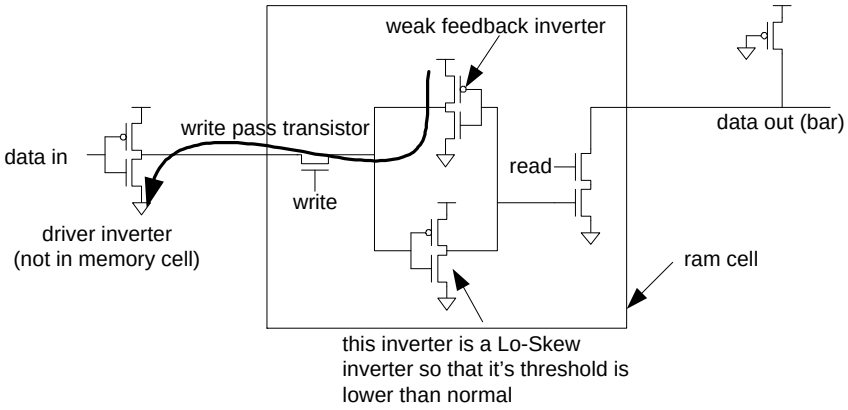


Figure E8.2 – Write Zero operation for single-ended RAM cell

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.

8.5 [Yikes, a term project!! I think I meant “design in principle”…. Also SUBM rules assumed.]

Exercise 3.8 calculated the vertical pitch for minimum sized n and p-transistors (4λ). Here the pMOS is 6λ, so the vertical pitch (without substrate contacts) would be 29 λ.

Adding a substrate contact alters the n-transistor to VSS and p-transistor to VDD spacing. Taking the n to VSS spacing first, we have a 4λ VSS contact, a 4λ transistor and a 4λ spacing, so the center to center separation is 8λ compared to 6.5 λ without contacts. The p to VDD spacing will be 9λ . So the pitch can be 8+8+8+9 = 33λ.

The horizontal pitch is determined by the figure below.

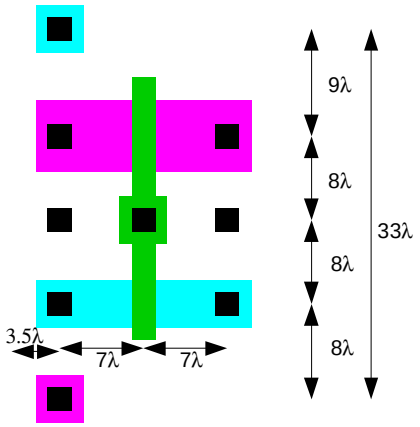


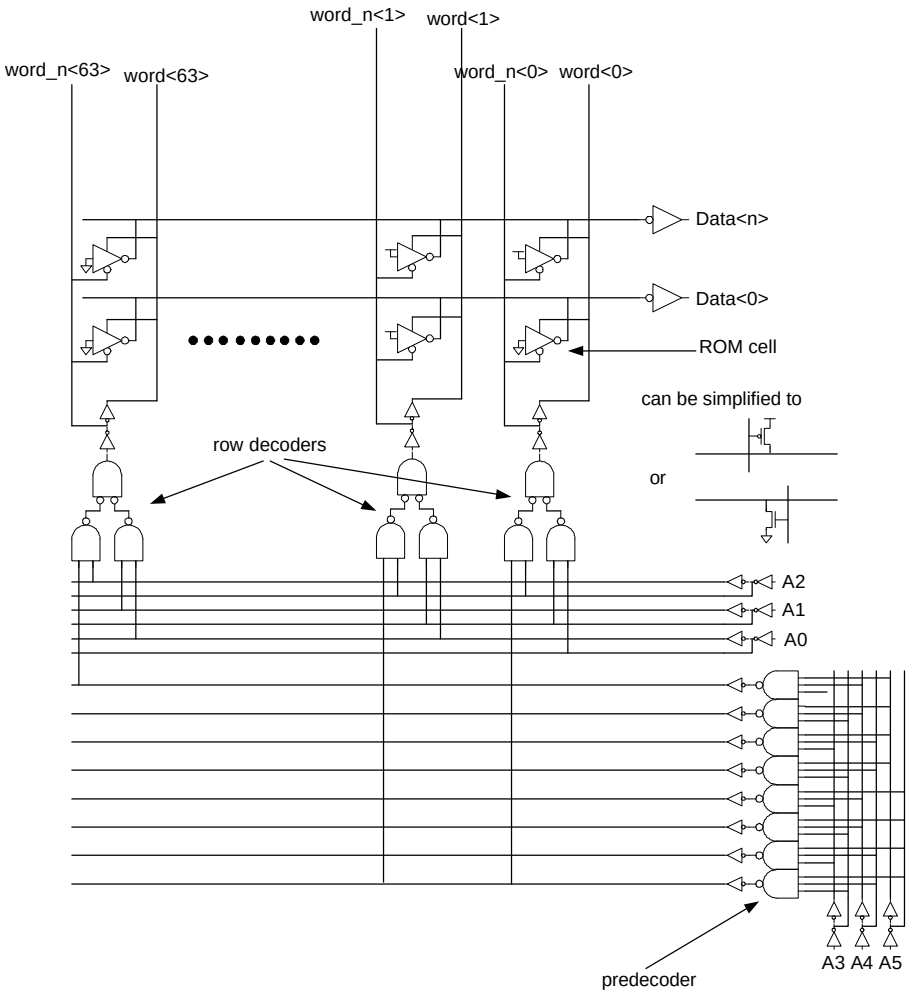
Figure E8.3 Horizontal Standard Cell Pitch

We need to contact the gate and be able to run a vertical metal pitch over the adjacent source and drain. It's likely that the source and drain require a metal2/metal1 contact as well, so the pitch has to be the contacted pitch. This is 1/2*4 + 1/2*4 + 3 = 7 λ per half-pitch or 14 λ between transistors. At the end of each cell we also leave this space to allow easy abutment (3.5 λ to centre of space from centre of source/drain contact) .

• • •

8.7 This is a little more complicated (than Exercise 8.6). Approaches here will vary widely, so it is fairly pointless to specify code (even though we did do it in the previous example). The main thing would be to look for a credible layout. My approach would be similar to the previous example. Write code for the primitives and then

hierarchically build these up to the ROM.



8.9 Using a standard cell library usually means that only normal logic gates, inverters and tristate buffers are available. The sense amp would just be an inverter. The column decode address buffers remain the same (inverters). The row decoder has to be implemented in terms of normal logic gates. The horizontal pitch depends on what is used for the ROM cell itself (see next). So we will delay a decision on the row decoder until the ROM cell is decide upon.

The simplest cell that may be used for the ROM cell is a tristate buffer. The tristate buffer has the input connected to either VDD or GND to program the cell. The output is connected to the bit line. The enable and enable_bar are routed from the row decoder (word and word_bar). If you have control over the standard cell library

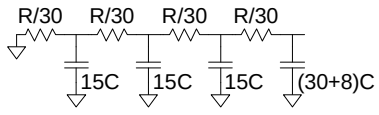
one can eliminate the extraneous transistors. Connection to VDD or GND can be left to the automatic router or completed inside the cell. Remember if this is done, the connection will usually have to be via a diffusion wire so that the gate is not directly to the supply rail (for gate breakdown reasons). For either the full tristate buffer or depleted cell, the horizontal pitch is two transistors, meaning that any cell with two series (or paralleled) transistors will "pitch match" in the vertical plane (assuming transistors are running horizontal).

Now that the ROM cell is decided, the row decoder may be designed. Is essence one "builds down" from the ROM cell. First we need a buffers (inverters) for the word and word_bar signals. These take the first two rows. Then a stack comprised of a two input NOR gate and two, two input NAND gates. The NAND gates decode address bits A0-A2 and one of eight predecoded lines of A3-A5. The predecode lines are located with the address buffers in the lower right.

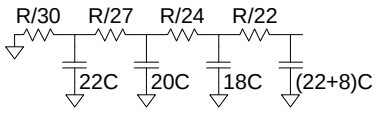
8.11 Another largish exercise. The main portions of the code appear in the text (sans mistakes). The gross test of a working solution at the end of this exercise is a simulation showing a sine wave. The good thing about this is that artefacts are easily spotted by eye. A nice clean sine wave means the design was probably done correctly. The latter part of the problem (comparing with the ROM based design) depends obviously on the student having done that problem.

8.13 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).

8.15 RC models of the two circuits are shown below. The Elmore delay of the untapered stack is [15*(1/30) + 15*(2/30) + 15*(3/30) + 38*(4/30)]RC = 8.07 RC. The Elmore delay of the tapered stack is [22*(1/30) + 20*(1/27 + 1/30) + 18*(1/24 + 1/27 + 1/30) + 30*(1/22 + 1/24 + 1/27 + 1/30)]RC = 8.88 RC. The untapered design is faster.



Untapered



Tapered

Chapter 9

9.1 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.

9.3 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.

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

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

9.9 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.

9.11 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.

9.13 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.

9.15 The software radio consists of an IQ conversion unit, four microprocessors with multipliers and four memories. At the SOC level, we would start by adding the required Wrapper Serial Port (WSP) to each block. The decision then may be made as to whether a Wrapper Parallel Port (WPP) is required. This would depend on whether the intelligence for the block could be implemented internally or externally. On a case by case basis, let us look at each module.

The IQ conversion unit consists of an NCO and IQ multipliers. The inputs are an I and Q signal and control values for an internal NCO. The output is the sum of the products of the NCO and IQ inputs. The NCO (Figure 9.25) can be tested autonomously using a signature analyzer. It would be possible to extend this to the full

module by placing LFSRs on the I and Q inputs. A fault analysis would indicate how many vectors would have to be run to achieve an acceptable fault coverage. So we probably do not need a WPP here.

The microprocessor has a sequencer that can be used to set up tests autonomously. So it probably does not need a WPP port.

The memories do not have any innate intelligence, so a WPP port may be used here to test the memories in parallel from a central RAM test unit (not unlike the design in the previous example). So one test unit tests four RAMs. Including the test unit in each RAM would mean that no WPP would be required.

Overall no WPPs are required at all – the time to serially shift data in and out just affects testing time – so they probably would go in for the RAMs.

In terms of TAM design, one could select the Daisy-chained TAM. But this is likely to impact test time (but good if you want to minimize pin count). The local TAM controller option is likely to be good as it minimizes pins and the local controllers default to very simple circuits for the processor and IQ converter. The basic thing here is that any of the designs work – we just want some good reasons such as reducing test time, complexity or pin count.

Chapter 10

10.1 ### no solution available

10.3 $V = A_{N-1}(B_{N-1} \oplus SUB)\overline{Y}_{N-1} + \overline{A}_{N-1}(\overline{B}_{N-1} \oplus SUB)Y_{N-1}$

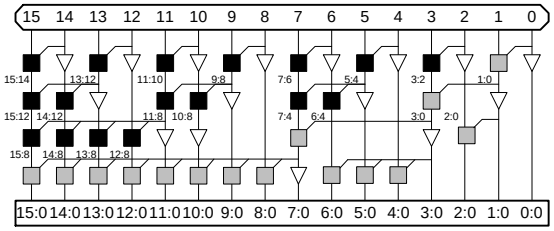
10.5 Assuming the side loads are negligible so that each carry chain drives another identical chain and has $h = 1$, the stage delay is $g + p$. The number of stages is inversely proportional to n. Hence the delay per bit scales as:

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

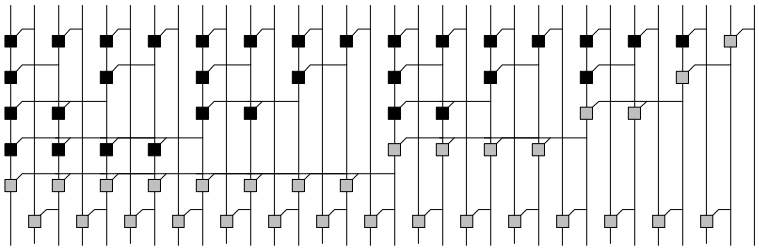
Taking the derivative of delay with respect to the length of each chain n and setting that equal to zero gives allows us to solve for the best chain length. Because the parasitic capacitance is large, the best delay is achieved with short carry chains ($n = 2$ or 3).

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

10.7



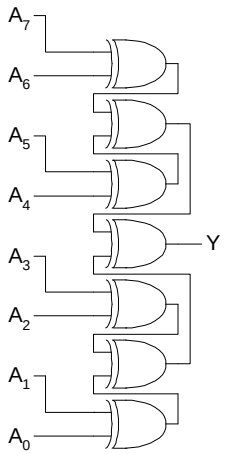
10.9



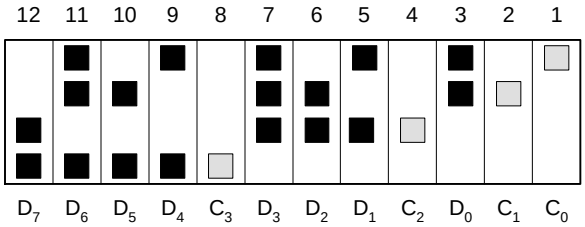
10.11

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

10.13



10.15 4 check bits suffice for up to 2^4-4-1 = 11 data bits.

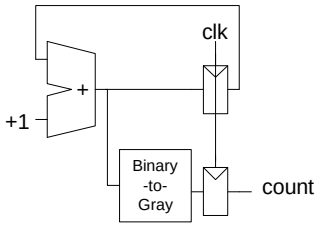


$$C_0 = D_6 \oplus D_4 \oplus D_3 \oplus D_1 \oplus D_0$$
$$C_1 = D_6 \oplus D_5 \oplus D_3 \oplus D_2 \oplus D_0$$
$$C_2 = D_7 \oplus D_3 \oplus D_2 \oplus D_1$$
$$C_3 = D_7 \oplus D_6 \oplus D_5 \oplus D_4$$

10.17 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.



10.19 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.

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

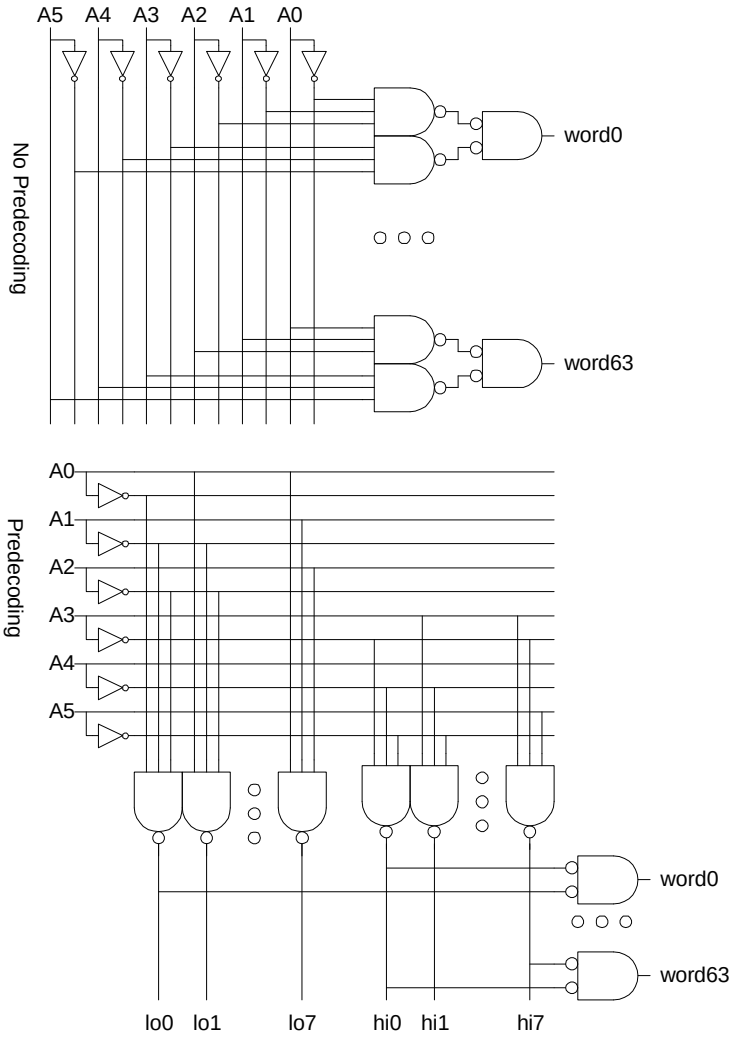
10.21 Assume the branching effort on each A input is approximate 2 because it drives two gates (the initial inverter and the final AND). A path from input to output passes through an inverter and five AND gates, each made from a NAND and an inverter. There are four two-way branches within the network. 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.

Chapter 11

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

11.3 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 prede-

coded design tends to be more convenient.



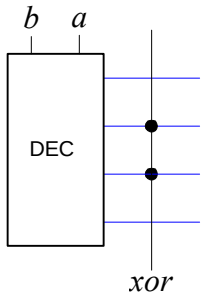
11.5 (a) B = 512. H = 20. A 10-input NAND gate has a logical effort of 12/3, so estimate that the path logical effort is about 4. Hence $F = GBH$ = 40960. The best number of stages is $\log_4 F$ = 7.66, so try an 8-stage design: NAND3-INV-NAND2-INV-NAND2-INV-INV-INV. This design has an actual logical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

(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

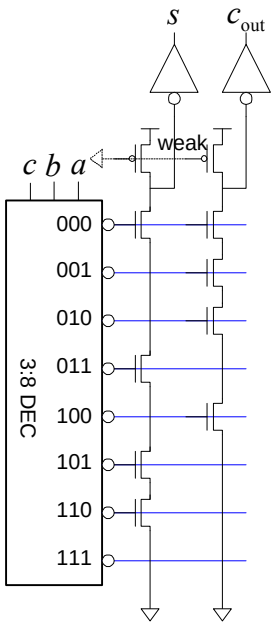
decrease for domino. Using the same design, the footless domino path has a path logical effort of $G = 1 * (5/6) * (2/3) * (5/6) * (2/3) * (5/6) * (1/3) * (5/6) = 0.071$ and a path effort of $F = 732$. The path parasitic delay is $P = 4/3 + 5/6 + 3/3 + 5/6 + 3/3 + 5/6 + 1/3 + 5/6 = 7$. $D = NF^{1/N}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

11.7 $H = 2^m$. $B = 2^{n-1}$ 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 $P = n + (N-1)$. Hence, the path delay can be estimated as D = $((n+m)/2)\ (2^{n+m}(n+2)/6)^{(2/(n+m))} + n + (N-1)$

11.9



11.11



11.13 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 $D = 2N + 4\log_4[(n+2)/9] + n + 2^n/6$.

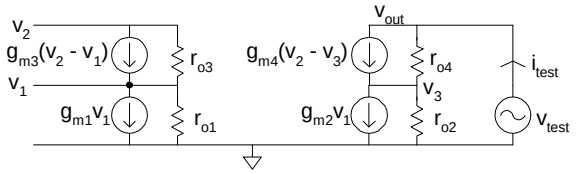
Your modeling and loading assumptions may vary somewhat. The assumptions about wire capacitance have a large effect on the model.

Chapter 12

12.1 $P_{max} = (110\text{-}50) / (10 + 2) = 5$ W.

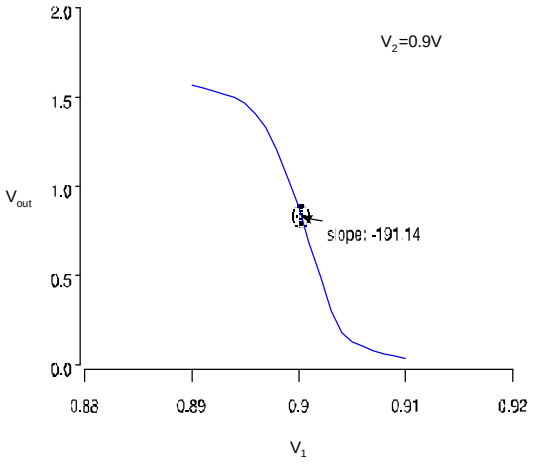
12.3 H-trees ideally have zero skew and relatively low metal resource requirements, but in practice see significant skews, even locally, because of mismatches in loading, processing, and environment among the branches. Clock grids have low local skew because they short together nearby points, but can 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.

12.5 The small signal model is shown below. The input is open-circuited and a test voltage is applied to the output. Because no input current flows, $v_1 = v_2 = 0$. $v_3 = i_{test} r_{o2}$. Applying KCL at v_{test} gives $i_{test} = g_{m4}(-v_3) + (v_{test}-v_3)/r_{o4}$. Substituting v_3 and solving for $R_{out} = v_{test} / i_{test}$ gives EQ (12.24). The approximation holds because $g_m >> 1/r_o$ because transistors have high output impedance.



12.7 Solve EQ (12.23) numerically for $R = 4.46\ k\Omega$ given $I_1 = 200\ \mu A$, assuming $V_{DD} = 1.8$ V. $V_1 = 1.8 - I_1 R = 0.98$ V. V_{out} must be greater than $V_1 - V_t = 0.58$ V to keep the output transistor in saturation.

12.9 To get satisfactory results, the resistor needs to be increased to 100 kW and the P1 and P2 channel lengths must be doubled. With these changes, the gain is 191.




```
*129-opamp.sp
*created by Ted Jiang 11/8/04
*
*******************************************
```

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

*********************************************
*Simulation Netlist
*********************************************
Vdd     vdd     gnd     'SUPPLY'
V1      1       gnd     0
V2      2       gnd     0.9
R1      vdd     B       R=100k
MN3     B       B       gnd     gnd     NMOS    W=4     L=2
MP1     C       C       vdd     vdd     PMOS    W=4     L=4
MN1     C       1       X       gnd     NMOS    W=4     L=2
MN4     X       B       gnd     gnd     NMOS    W=4     L=2
MP2     Y       C       vdd     vdd     PMOS    W=4     L=4
MN2     Y       2       X       gnd     NMOS    W=4     L=2
MP3     Out     Y       vdd     vdd     PMOS    W=4     L=2
MN5     Out     B       gnd     gnd     NMOS    W=4     L=2

*********************************************
*Stimulus
*********************************************
.dc v1 0.89 0.91 0.001
.end
```

12.11

$$\left.\frac{\partial \Delta V}{\partial x}\right|_{x=0.5} = \left[\frac{2}{a}(2x)^{\frac{1}{a}-1} + \frac{2}{a}\left(2(1-x)\right)^{\frac{1}{a}-1}\right] V_{go}\Big|_{x=0.5} = \frac{4}{a} V_{go}$$

$$I_1 = xI_{\text{ref}}$$

$$g_m = \left.\frac{\partial I_{ds}}{\partial V_{gs}}\right|_{V_{gs}=V_{go}+V_t} = \frac{\partial}{\partial V_{gs}} k\left(V_{gs}-V_t\right)^a = a\,k\left(V_{gs}-V_t\right)^{a-1} = a\,\frac{I_{ds}}{V_{go}} = \frac{a}{2}\frac{I_{\text{ref}}}{V_{go}}$$

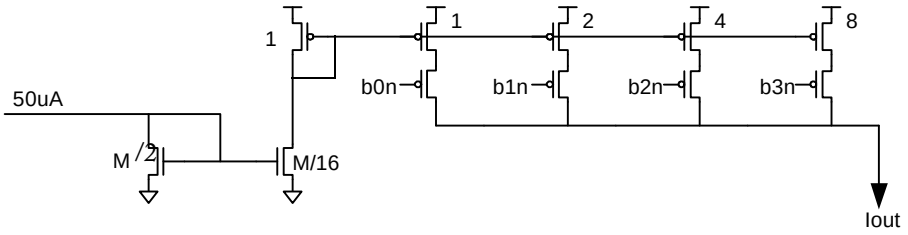
$$\frac{\partial I_1}{\partial \Delta V} = \frac{\partial I_1}{\partial x}\frac{\partial x}{\partial \Delta V} = \frac{a}{4}\frac{I_{\text{ref}}}{V_{go}} = \frac{g_m}{2}$$

12.13 g_m = 7.2 * 10^{-5}; r_o = 7.0 MΩ; $g_m r_o$ = 50.3

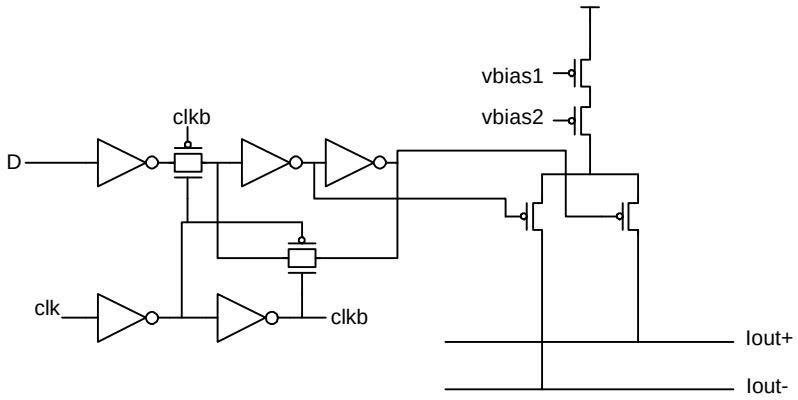
Use the same SPICE deck as 12.12 but change channel length to 4.

12.15 A possible circuit is shown below. An nMOS current mirror scales the input refer-

ence down by a factor of 8. This is then mirrored to a pMOS transistor which drives four scaled pMOS current sources. Any variation of mirrors and polarity is acceptable as none was specified in the question. The main thing to look for is the right scaling of the current mirrors.



12.17 The main idea in inserting a latch in the DAC cell is to reduce the uncertainty in decode times and to some extent time of flight to the switched cells. So the latch is placed in the current cell itself. This is a slave latch. The master latch may be placed before the row or column decode gates (or after). It really depends on the timing of signals in the rest of the system. The following diagram shows a latch inserted in the conventional pMOS cascade DAC current cell.



Note that the pMOS switches are driven by the cascaded latch inverters. This means that there is a slight asymmetry in the rising and falling edges applied to the switches which is not ideal. We would like the switches to be driven with perfectly complimentary signals. One can play with this by inserting a pass gate between the output of the first inverter and the leftmost pMOS switch. (e.g. see Section 12.5.5).

Although this appears to markedly increase the complexity of the current cell, transistors are so small in current processes that compact DACs are still possible even with all of these transistors present.

With an internal latch and well matched pMOS switch drivers, the only systematic timing deviation that we now need to worry about is clock skew between current

cells. This is achieved by paying close attention to the clock routing and keeping the RC time constants low. The clock network may be run in a mesh on a single upper metal layer with shield layers below to shield the analog signals. The other signal to pay attention to is V_{DD}. Again a mesh connection on one or more layers should be used to keep the resistance to the cells low and relatively equal.