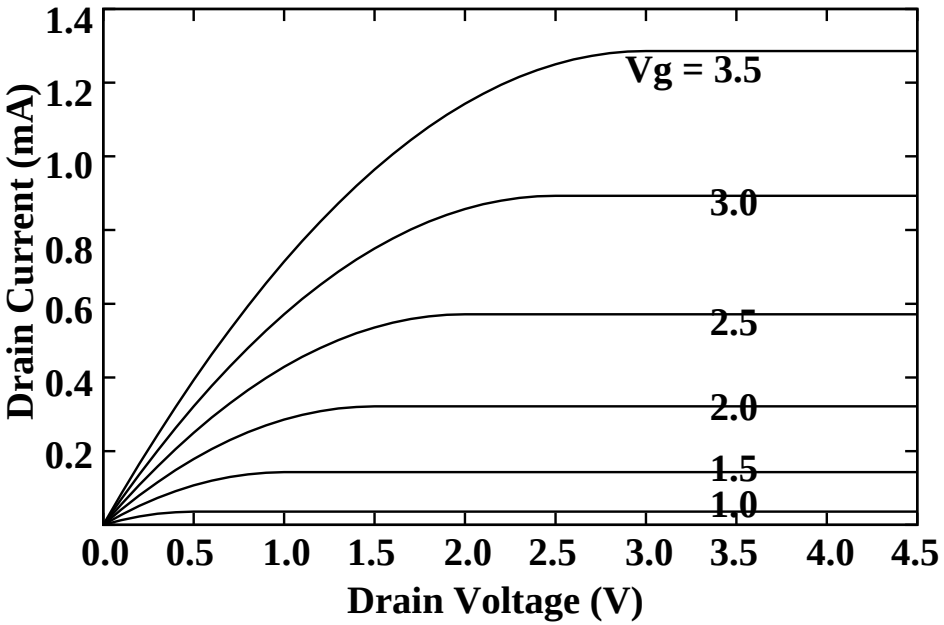
Logic Design Styles

Dinesh Sharma

Microelectronics Group, EE Department
IIT Bombay, Mumbai

June 1,2006

A simple model



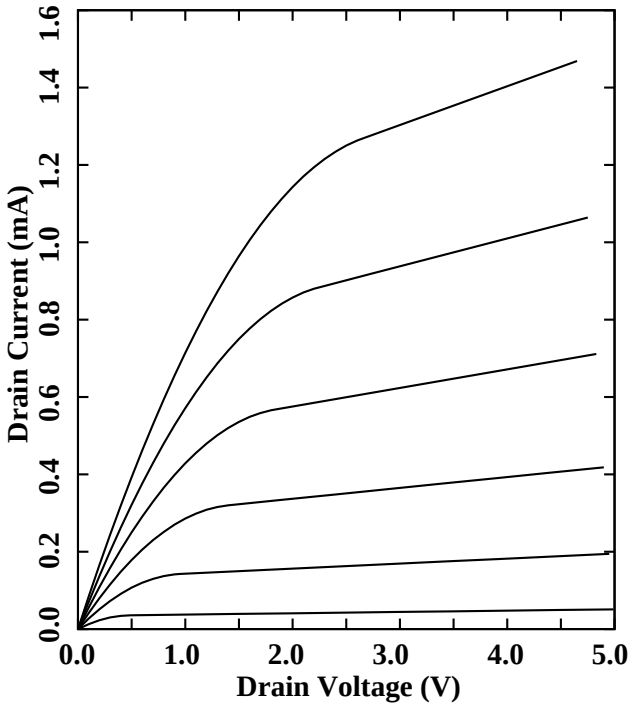
for $V_{gs} \leq V_T$, $I_{ds} = 0$

for $V_{gs} > V_T$ and $V_{ds} \leq V_{gs} - V_T$,
$I_{ds} = K\left[(V_{gs} - V_T)V_{ds} - \frac{1}{2}V_{ds}^2\right]$

for $V_{gs} > V_T$ and $V_{ds} > V_{gs} - V_T$,
$I_{ds} = K\frac{(V_{gs}-V_T)^2}{2}$

This model assumes current to be independent of V_{ds} in the saturation region.
(This is somewhat oversimplified.)

A more realistic model



Let 'Early Voltage' $\equiv V_E$

$$\text{define } V_{dss} \equiv V_E \left(\sqrt{1 + \frac{2(V_{gs} - V_T)}{V_E}} - 1 \right)$$

$$\simeq (V_{gs} - V_T) \left(1 - \frac{V_{gs} - V_T}{2V_E} \right)$$

$$\text{and } I_{dss} \equiv K \left[(V_{gs} - V_T)V_{dss} - \frac{1}{2}V_{dss}^2 \right]$$

$$\text{for } V_{gs} > V_T \text{ and } V_{ds} \leq V_{dss} \quad I_{ds} = K \left[(V_{gs} - V_T)V_{ds} - \frac{1}{2}V_{ds}^2 \right]$$

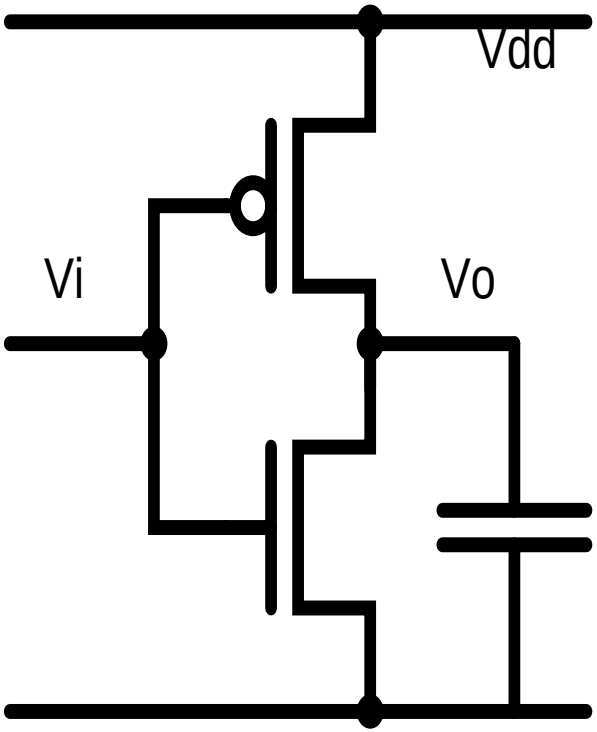
$$\text{for } V_{gs} > V_T \text{ and } V_{ds} > V_{dss} \quad I_{ds} = I_{dss} \frac{V_d + V_E}{V_{dss} + V_E}$$

CMOS Static Logic

- Each logic stage contains pull up and pull down networks controlled by input signals.
- The pull up network contains p channel transistors.
- The pull down network is made of n channel transistors.
- If the pull up network is ‘on’, the pull down network is ‘off’ and *vice versa*.
- Since the pull up and pull down networks are never ‘on’ simultaneously, there is no static power consumption.

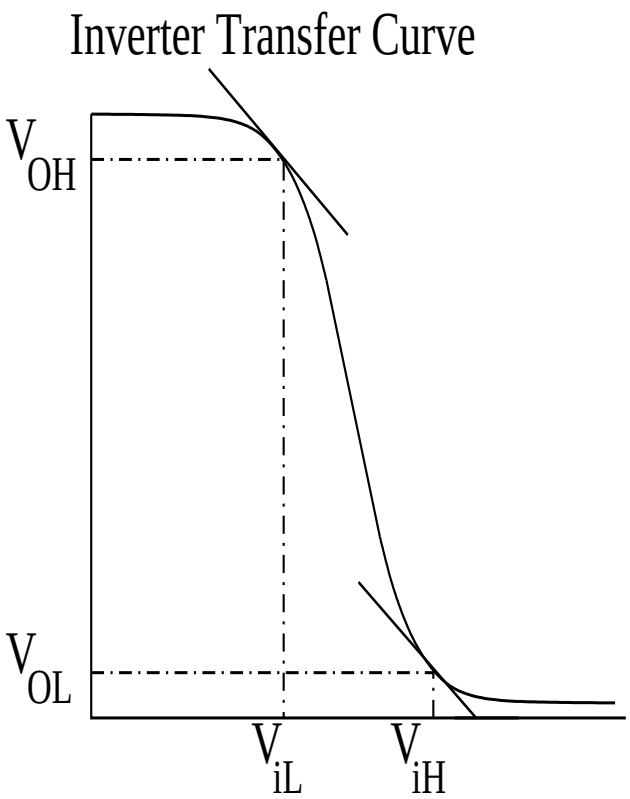
CMOS Inverter

The simplest of CMOS logic structure is the inverter.



- CMOS inverter is the basic gate.
- More complex gates are designed by mapping them to an ‘equivalent’ inverter.
- The pull up network of the logic gate is made equivalent to the pMOS of the inverter.
- The pull down network of the logic gate is made equivalent to the nMOS of the inverter.
- Thumb rules are used to map the geometries of the pull up and pull down networks to single transistors.

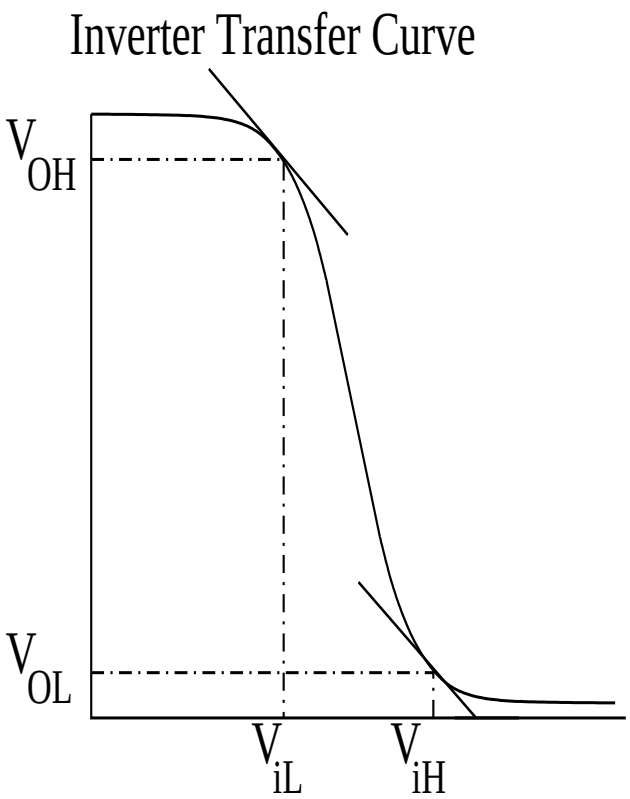
Static Characteristics



The range of input voltages can be divided into several regions.

- nMOS ‘off’, pMOS ‘on’
- nMOS saturated, pMOS linear
- nMOS saturated, pMOS saturated
- nMOS linear, pMOS saturated
- nMOS ‘on’, pMOS ‘off’

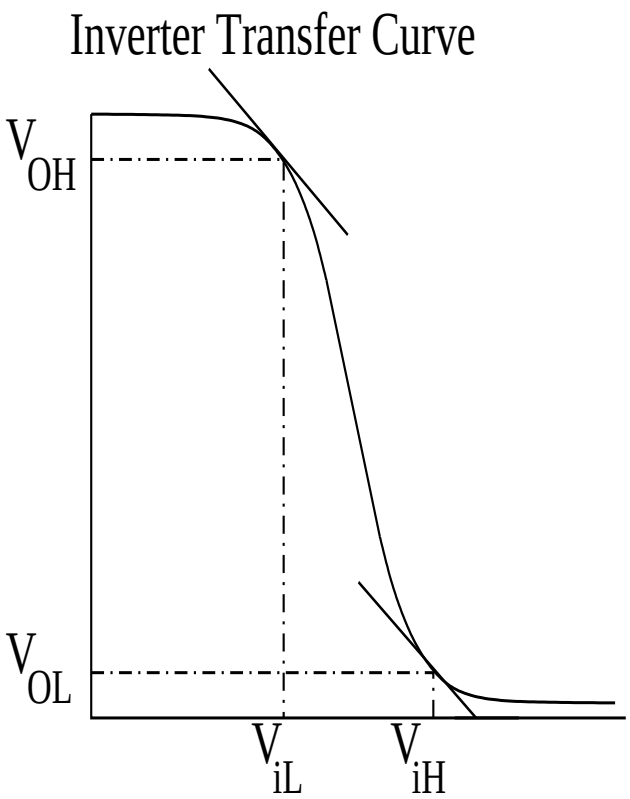
nMOS ‘off’, pMOS ‘on’



For $0 < V_i < V_{Tn}$

- the n channel transistor is ‘off’,
- the p channel transistor is ‘on’ and the output voltage = V_{dd}.
- This is the normal digital operation range with input = ‘0’ and output = ‘1’.

nMOS saturated, pMOS linear



- In this regime, both transistors are ‘on’.
- The input voltage V_i is $> V_{Tn}$, but is small enough so that the n channel transistor is in saturation, and the p channel transistor is in the linear regime.
- In static condition, the output voltage will adjust itself such that the currents through the n and p channel transistors are equal.

nMOS saturated, pMOS linear

- The absolute value of gate-source voltage on the p channel transistor is V_{dd}- V_i, and therefore the "over voltage" on its gate is V_{dd}- V_i- V_{Tp}.
- The drain source voltage of the pMOS has an absolute value V_{dd}-V_o.
- Therefore,

$$\begin{aligned} I_d &= K_p \left[(V_{dd} - V_i - V_{Tp})(V_{dd} - V_o) - \frac{1}{2}(V_{dd} - V_o)^2 \right] \\ &= \frac{K_n}{2}(V_i - V_{Tn})^2 \end{aligned}$$

Where symbols have their usual meanings.

We define $\beta \equiv K_n/K_p$ and $V_{dp} \equiv V_{dd} - V_o$
Then we can solve the quadratic equation:

$$\begin{aligned} I_d &= K_p \left[(V_{dd} - V_i - V_{Tp})(V_{dd} - V_o) - \frac{1}{2}(V_{dd} - V_o)^2 \right] \\ &= \frac{K_n}{2}(V_i - V_{Tn})^2 \end{aligned}$$

$$\text{So } V_o = V_i + V_{Tp} + \sqrt{(V_{dd} - V_i - V_{Tp})^2 - \beta(V_i - V_{Tn})^2}$$

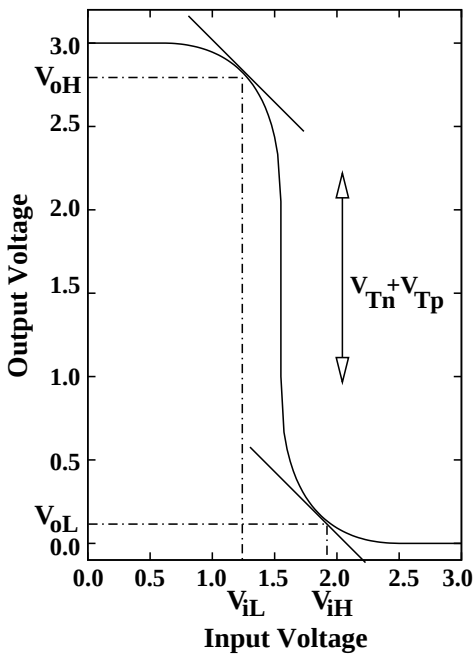
If $K_n = K_p; (\beta = 1)$,

$$V_o = (V_i + V_{Tp}) + \sqrt{(V_{dd} - V_{Tn} - V_{Tp})(V_{dd} - 2V_i + V_{Tn} - V_{Tp})}$$

$$\text{for } V_i \leq \frac{V_{dd} + V_{Tn} - V_{Tp}}{2}$$

nMOS saturated, pMOS saturated

when $V_i = \frac{V_{dd}+\sqrt{\beta}V_{Tn}-V_{Tp}}{1+\sqrt{\beta}}$, both transistors are saturated.



- Currents of both transistors are independent of their drain voltages.
- we do not get a unique solution for V_o by equating drain currents.
- The currents will be equal for *all* values of V_o in the range

$$V_i - V_{Tn} \leq V_o \leq V_i + V_{Tp}$$

Thus the transfer curve of an inverter shows a drop of V_{Tn}+ V_{Tp} at a voltage near $V_{dd}/2$.

nMOS linear, pMOS saturated

As we increase V_i further, so that

$$\frac{V_{dd} + \sqrt{\beta} V_{Tn} - V_{Tp}}{1 + \sqrt{\beta}} < V_i < V_{dd} - V_{Tp}$$

both transistors are still 'on', but nMOS enters the linear regime while pMOS is saturated. Equating currents in this condition,

$$\begin{aligned} I_d &= \frac{K_p}{2}(V_{dd} - V_i - V_{Tp})^2 \\ &= K_n \left[(V_i - V_{Tn}) V_o - \frac{1}{2} V_o^2 \right] \end{aligned}$$

From this, we get the quadratic equation

$$\frac{1}{2} V_o^2 - (V_i - V_{Tn}) V_o + \frac{(V_{dd} - V_i - V_{Tp})^2}{2\beta} = 0$$

$$\frac{1}{2}V_o^2 - (V_i - V_{Tn})V_o + \frac{(V_{dd} - V_i - V_{Tp})^2}{2\beta} = 0$$

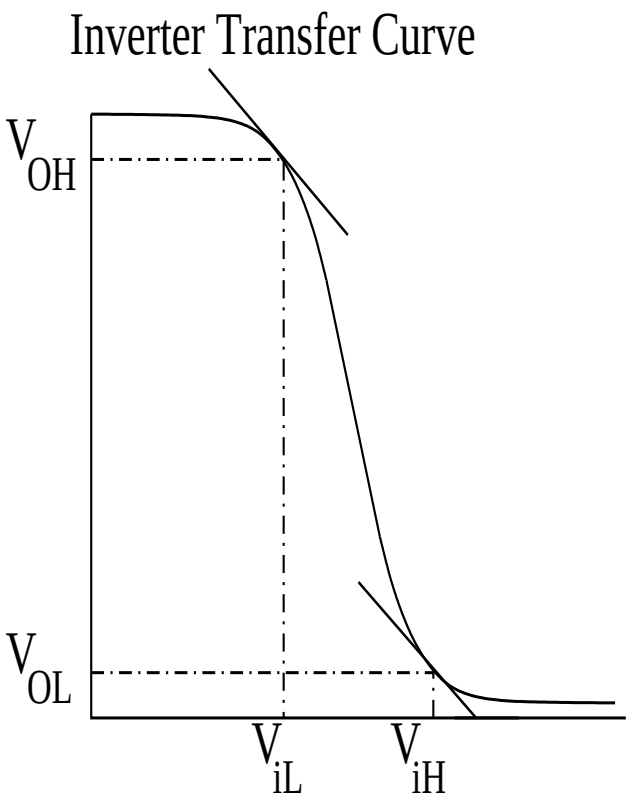
This has solutions

$$V_o = (V_i - V_{Tn}) - \sqrt{(V_i - V_{Tn})^2 - \frac{(V_{dd} - V_i - V_{Tp})^2}{\beta}}$$

In the special case where $\beta = 1$, we have

$$V_o = (V_i - V_{Tn}) - \sqrt{(V_{dd} - V_{Tn} - V_{Tp})(2V_i - V_{dd} - V_{Tn} + V_{Tp})}$$

nMOS ‘on’, pMOS ‘off’



- As we increase the input voltage beyond V_{dd}- V_{Tp}, the p channel transistor turns ‘off’, while the n channel conducts strongly.
- As a result, the output voltage falls to zero.
- This is the normal digital operation range with input = ‘1’ and output = ‘0’.

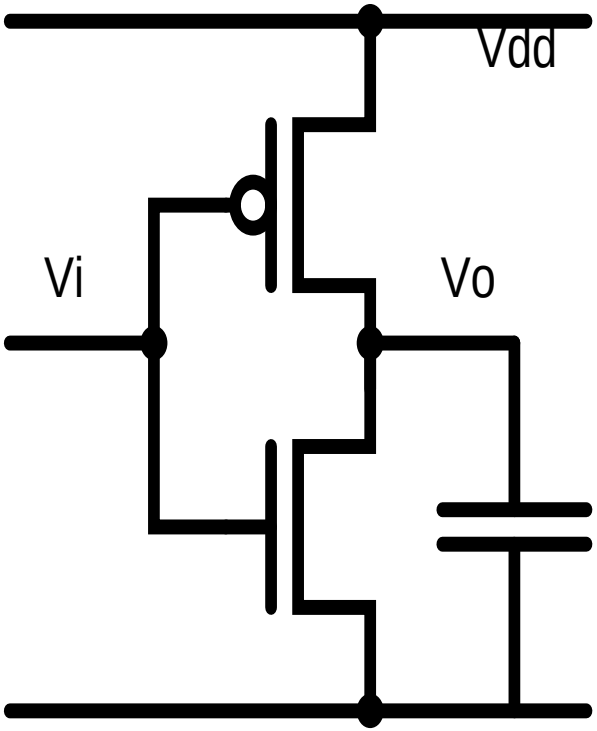
Noise Margins

- For robust design, the output levels must be interpreted correctly at the input of next stage even in the presence of noise.
- For the 'high' level, we require that the output of one stage should still be interpreted as 'high' at the input of the next gate even when pulled down a little due to noise.
- Therefore V_{oH} should be $> V_{iH}$.
- Similarly V_{oL} should be $< V_{iL}$
- The difference, $V_{iL} - V_{oL}$ is the 'low' noise margin. and $V_{oH} - V_{iH}$ is the 'high' noise level.

Logic Levels

- A digital circuit should distinguish logic levels, but be insensitive to the exact analog voltage at the input.
- Therefore flat portions of the transfer curve (where $\frac{\partial V_o}{\partial V_i}$ is small) are suitable for digital logic.
- We select two points on the transfer curve where the slope ($\frac{\partial V_o}{\partial V_i}$) is -1.0.
- The coordinates of these two points define the values of (V_{iL},V_{oH}) and (V_{iH},V_{oL}).
- The region to the left of V_{iL} and to the right of V_{iH} has $|\frac{\partial V_o}{\partial V_i}| < 1$, and is suitable for digital operation.

Calculation of Noise Margins



- To evaluate the values of noise margins, we shall use the expressions derived for $\beta = 1$ to keep the algebra simple.
- When the input is low and output high, the n channel transistor is saturated and the p channel transistor is in its linear regime.
- When the input is high and the output is low, the n channel transistor is in its linear regime, while the p channel transistor is saturated.

Calculation of V_{iL} and V_{oH}

for (V_{iL},V_{oH}), n channel transistor is saturated, while the p channel transistor is in its linear regime.

$$V_o = (V_i + V_{Tp}) + \sqrt{(V_{dd} - V_{Tn} - V_{Tp})(V_{dd} + V_{Tn} - V_{Tp} - 2V_i)}$$

From this, we evaluate $\frac{\partial V_o}{\partial V_i}$ and set it = -1.

$$\frac{\partial V_o}{\partial V_i} = -1 = 1 - \sqrt{\frac{V_{dd} - V_{Tn} - V_{Tp}}{V_{dd} + V_{Tn} - V_{Tp} - 2V_i}}$$

This gives

$$V_{iL} = \frac{3V_{dd} + 5V_{Tn} - 3V_{Tp}}{8}$$

$$V_{oH} = \frac{7V_{dd} + V_{Tn} + V_{Tp}}{8} = V_{dd} - \frac{V_{dd} - V_{Tn} - V_{Tp}}{8}$$

Calculation of V_{iH} and V_{oL}

When the input is 'high', we should use the equation for nMOS linear and pMOS saturated.

$$V_o = (V_i - V_{Tn}) - \sqrt{(V_{dd} - V_{Tn} - V_{Tp})(2V_i - V_{dd} - V_{Tn} + V_{Tp})}$$

Differentiating with respect to V_i gives

$$\frac{\partial V_o}{\partial V_i} = -1 = 1 - \sqrt{\frac{V_{dd} - V_{Tn} - V_{Tp}}{2V_i - V_{dd} - V_{Tn} + V_{Tp}}}$$

From where, we get

$$V_{iH} = \frac{5V_{dd} + 3V_{Tn} - 5V_{Tp}}{8}$$

$$V_{oL} = \frac{V_{dd} - V_{Tn} - V_{Tp}}{8}$$

Calculation of Noise Margins

The 'High' noise margin is given by

$$V_{oH} - V_{iH} = \frac{V_{dd} - V_{Tn} + 3V_{Tp}}{4}$$

Similarly, the 'Low' noise margin is

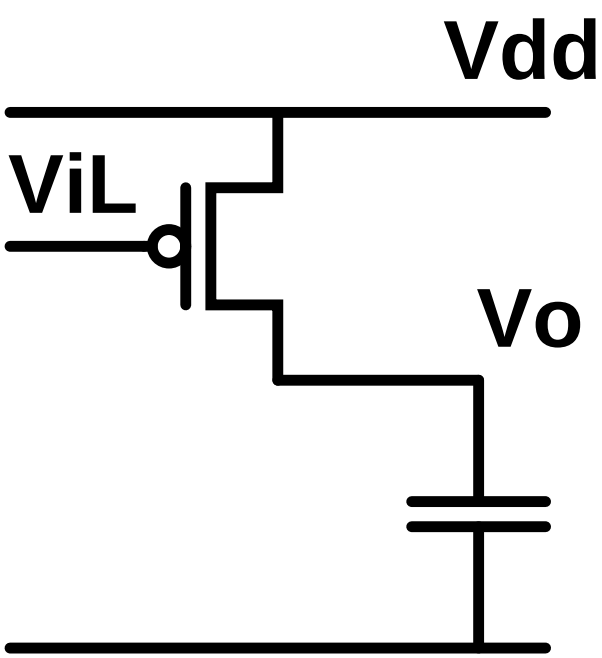
$$V_{iL} - V_{oL} = \frac{V_{dd} + 3V_{Tn} - V_{Tp}}{4}$$

The two noise margins can be made equal by choosing equal values for V_{Tn} and V_{Tp}.

Dynamic Characteristics

- For the calculation of rise and fall times, we shall assume that only one of the two transistors in the inverter is ‘on’.
- This is more conservative than the static logic levels calculated by slope considerations.
- We shall use the simple model described at the beginning of this lecture.

Rise time



When the input is low, the n channel transistor is ‘off’, while the p channel transistor is ‘on’. From Kirchoff’s current law at the output node,

$$I_{dp} = C\frac{dV_o}{dt}$$

so,

$$\frac{dt}{C} = \frac{dV_o}{I_{dp}}$$

Integrating both sides, we get

$$\frac{\tau_{rise}}{C} = \int_0^{V_{oH}} \frac{dV_o}{I_{dp}}$$

$$\frac{\tau_{rise}}{C} = \int_0^{V_{oH}} \frac{dV_o}{I_{dp}}$$

Till the output rises to V_{iL}+ V_{Tp}, the p channel transistor is in saturation.

if $V_{oH} > V_{iL} + V_{Tp}$ (which is normally the case), the integration range can be broken into saturation and linear regimes. Thus

$$\begin{aligned}\frac{\tau_{rise}}{C} &= \int_0^{V_{iL}+V_{Tp}} \frac{dV_o}{\frac{K_p}{2}(V_{dd} - V_{iL} - V_{Tp})^2} \\ &+ \int_{V_{iL}+V_{Tp}}^{V_{oH}} \frac{dV_o}{K_p\left[(V_{dd} - V_{iL} - V_{Tp})(V_{dd} - V_o) - \frac{1}{2}(V_{dd} - V_o)^2\right]}\end{aligned}$$

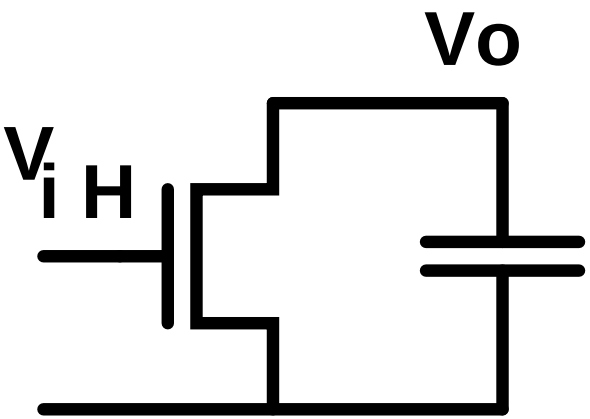
$$\tau_{rise} = \frac{2C(V_{iL} + V_{Tp})}{K_p(V_{dd} - V_{iL} - V_{Tp})^2} + \frac{C}{K_p(V_{dd} - V_{iL} - V_{Tp})} \ln \frac{V_{dd} + V_{oH} - 2V_{iL} - 2V_{Tp}}{V_{dd} - V_{oH}}$$

- The first term is just the constant current charging of the load capacitor.
- The second term represents the charging by the pMOS in its linear range.
- This can be compared with resistive charging, which would have taken a charge time of

$$\tau = RC \ln \frac{V_{dd} - V_{iL} - V_{Tp}}{V_{dd} - V_{oH}}$$

to charge from $V_{iL} + V_{Tp}$ to V_{oH}.

Fall Time



When the input is high, the p channel transistor is ‘off’, while the n channel transistor is ‘on’. From Kirchoff’s current law at the output node,

$$I_{dn} = -C\frac{dV_o}{dt}$$

Separating variables and integrating from the initial voltage (= V_{dd}) to some terminal voltage V_{oL} gives

$$\frac{\tau_{fall}}{C} = -\int_{V_{dd}}^{voL} \frac{dV_o}{I_{dn}}$$

Fall time

$$\frac{\tau_{fall}}{C} = -\int_{V_{dd}}^{voL} \frac{dV_o}{I_{dn}}$$

The n channel transistor will be in saturation till the output falls to V_i- V_{Tn}. Below this, the transistor will be in its linear regime. We can divide the integration range in two parts.

$$\begin{aligned}
\frac{\tau_{fall}}{C} &= -\int_{V_{dd}}^{V_i - V_{Tn}} \frac{dV_o}{I_{dn}} - \int_{V_i - V_{Tn}}^{V_{oL}} \frac{dV_o}{I_{dn}} \\
&= \int_{V_i - V_{Tn}}^{V_{dd}} \frac{dV_o}{\frac{K_n}{2}(V_i - V_{Tn})^2} \\
&+ \int_{V_{oL}}^{V_i - V_{Tn}} \frac{dV_o}{K_n[(V_i - V_{Tn})V_o - \frac{1}{2}V_o^2}
\end{aligned}$$

Fall time

$$\frac{\tau_{fall}}{C} = \frac{V_{dd} - V_i + V_{Tn}}{\frac{K_n}{2}(V_i - V_{Tn})^2} + \frac{1}{K_n(V_i - V_{Tn})} \ln \frac{2(V_i - V_{Tn}) - V_{oL}}{V_{oL}}$$

The first term represents the time taken to discharge at constant current in the saturation regime, whereas the second term is the quasi-resistive discharge in the linear regime.

Trade off between power, speed and robustness

Noise margins are given by

$$V_{oH} - V_{iH} = \frac{V_{dd} - V_{Tn} + 3V_{Tp}}{4}$$

$$V_{iL} - V_{oL} = \frac{V_{dd} + 3V_{Tn} - V_{Tp}}{4}$$

- As we scale technologies, we improve speed and power consumption. However, the noise margin becomes worse.
- We can improve noise margins by choosing relatively higher threshold voltages. However, this will reduce speeds.
- We could also increase V_{dd}- but that would increase power dissipation.

Thus we have a trade off between power, speed and noise margins.

CMOS Inverter Design Flow

- A common design requirement is symmetric charge and discharge behaviour and equal noise margins for high and low logic values.
- This requires matched values of K_n and K_p and equal values of $V_{Tn}$and V_{Tp}.
- Rise and fall times depend linearly on K_n and K_p.
- Thus it is a straightforward calculation to determine transistor geometries if speed requirements and technological parameters are given.
- However, as transistor geometries are made larger, self loading can become significant.

CMOS Inverter Design Flow

- For large self-loading, we have to model the load capacitance as

$$C_{Load} = C_{ext} + \alpha K_n$$

where we have assumed that $\beta = K_n/K_p$ is constant. α is a technological constant.

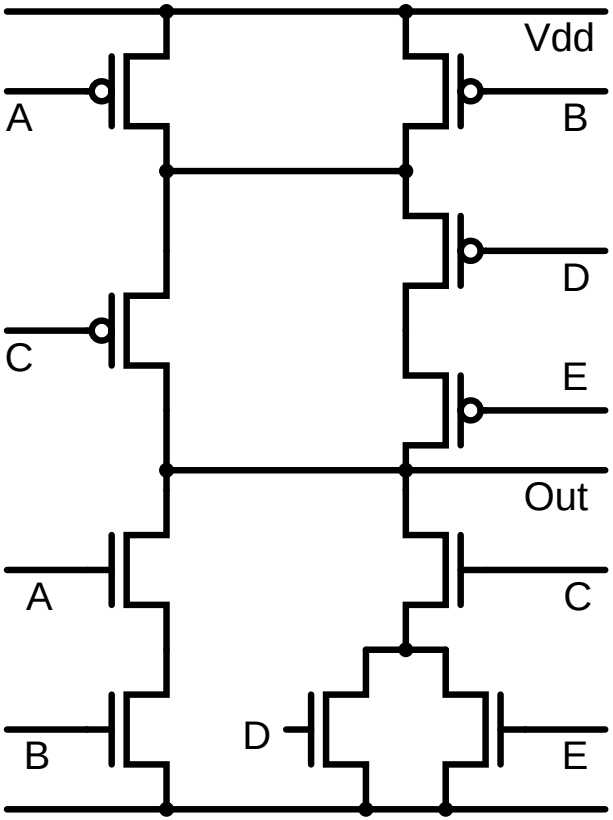
- We use the expressions for $K\tau/C$ which depend only on voltages. Once these values are calculated, the geometry can be determined.
- In the extreme case, when self capacitance dominates the load capacitance, K/C becomes constant and τ becomes geometry independent. There is no advantage in using wider transistors in this regime to increase the speed. It is better to use multi-stage logic with tapered buffers in this regime.

From Inverters to Other Logic

Once the basic CMOS inverter is designed, other logic gates can be derived from it. The logic has to be put in a canonical form which is a sum of products with a bar (inversion) on top.

- For every '.' in the expression, we put the corresponding n channel transistors in series and the corresponding p channel transistors in parallel.
- for every '+', we put the n channel transistors in parallel and the p channel transistors in series.
- We scale the transistor widths up by the number of devices (n or p) put in series.
- The geometries are left untouched for devices put in parallel.

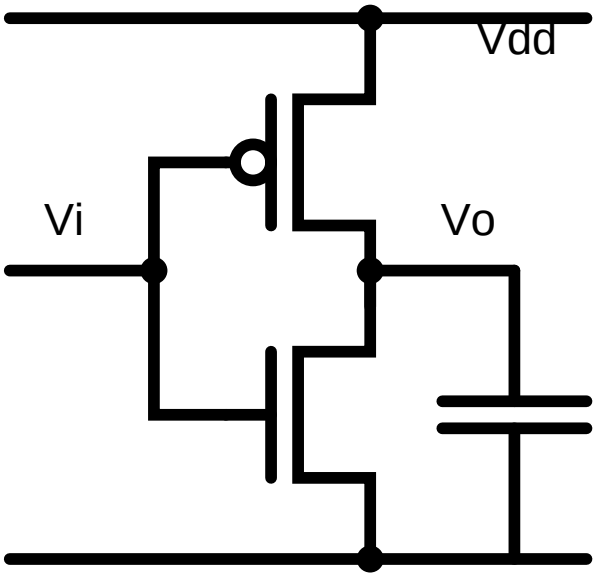
CMOS implementation of $\overline{A.B + C.(D + E)}$



- For n channel, A and B are in series, The pair is in parallel with C which is in series with a parallel combination of D and E.
- For p channel, A is in parallel with B, the pair is in series with C which is in parallel with a series combination of D and E.

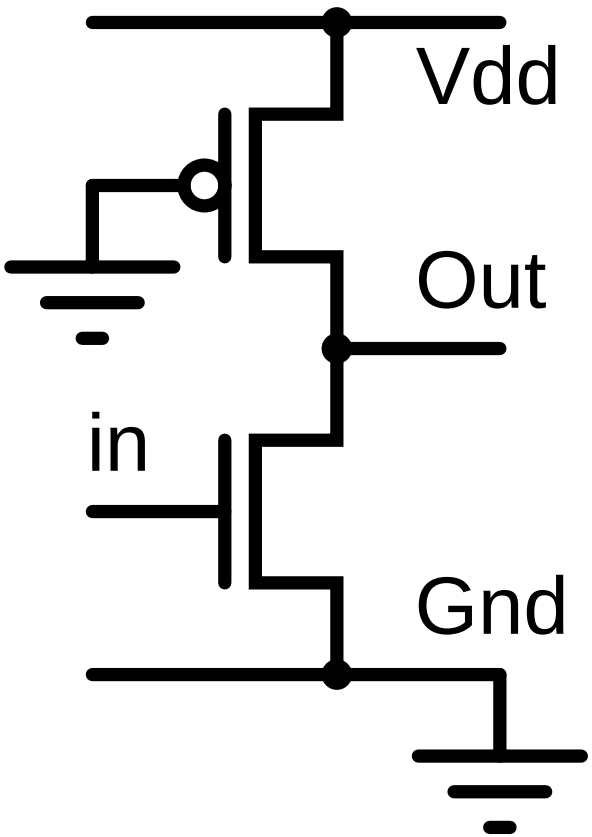
Implementation of $\overline{A.B + C.(D + E)}$ in CMOS logic design style.

CMOS summary



- Logic consumes no static power in CMOS design style.
- However, signals have to be routed to the n pull down network *as well as* to the p pull up network.
- So the load presented to every driver is high.
- This is exacerbated by the fact that n and p channel transistors cannot be placed close together as these are in different wells which have to be kept well separated in order to avoid latchup.

Pseudo nMOS Design Style



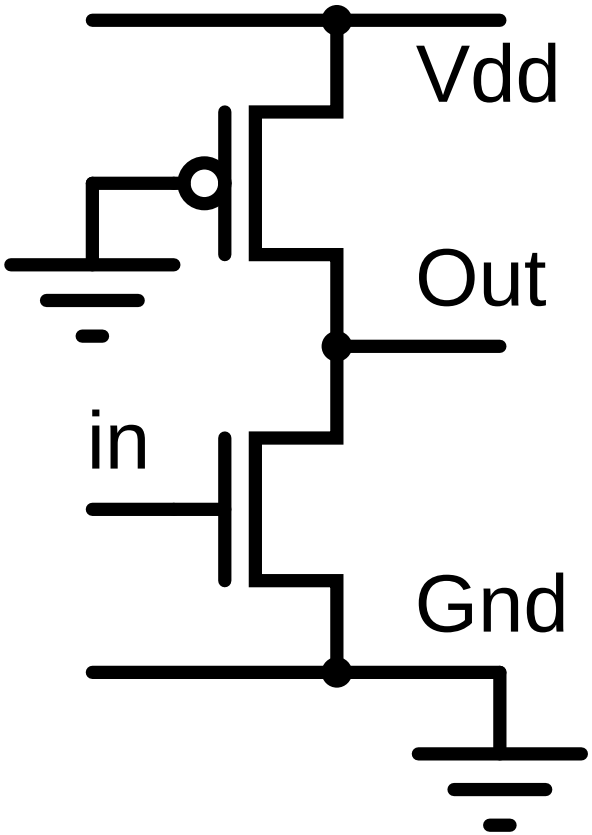
- The CMOS pull up network is replaced by a single pMOS transistor with its gate grounded.
- Since the pMOS is not driven by signals, it is always ‘on’.
- The effective gate voltage seen by the pMOS transistor is V_{dd}. Thus the overvoltage on the p channel gate is always V_{dd} - V_{Tp}.
- When the nMOS is turned ‘on’, a direct path between supply and ground exists and static power will be drawn.
- However, the dynamic power is reduced

Static Characteristics

As we sweep the input voltage from ground to V_{dd}, we encounter the following regimes of operation:

- nMOS 'off'
- nMOS saturated, pMOS linear
- nMOS linear, pMOS linear
- nMOS linear, pMOS saturated

Low input



- When the input voltage is less than V_{Tn}. The output is 'high' and no current is drawn from the supply.
- As we raise the input just above V_{Tn}, the output starts falling.
- In this region the nMOS is saturated, while the pMOS is linear

nMOS saturated, pMOS linear

The input voltage is assumed to be sufficiently low so that the output voltage exceeds the saturation voltage $V_i - V_{Tn}$.
Normally, this voltage will be higher than V_{Tp}, so the p channel transistor is in linear mode of operation.
Equating currents through the n and p channel transistors, we get

$$K_p \left[(V_{dd} - V_{Tp})(V_{dd} - V_o) - \frac{1}{2}(V_{dd} - V_o)^2 \right] = \frac{K_n}{2}(V_i - V_{Tn})^2$$

defining $V_1 \equiv V_{dd} - V_o$ and $V_2 \equiv V_{dd} - V_{Tp}$, we get

$$\frac{1}{2}V_1^2 - V_2 V_1 + \frac{\beta}{2}(V_i - V_{Tn})^2 = 0$$

nMOS saturated, pMOS linear

$$\frac{1}{2}V_1^2 - V_2V_1 + \frac{\beta}{2}(V_i - V_{Tn})^2 = 0$$

The solutions are:

$$V_1 = V_2 \pm \sqrt{V_2^2 - \beta(V_i - V_{Tn})^2}$$

substituting the values of V_1 and V_2 and choosing the sign which puts V_o in the correct range, we get

$$V_o = V_{Tp} + \sqrt{(V_{dd} - V_{Tp})^2 - \beta(V_i - V_{Tn})^2}$$

nMOS linear, pMOS linear

$$V_o = V_{Tp} + \sqrt{(V_{dd} - V_{Tp})^2 - \beta(V_i - V_{Tn})^2}$$

- As the input voltage is increased, the output voltage will decrease.
- The output voltage will fall below $V_i - V_{Tn}$ when

$$V_i > V_{Tn} + \frac{V_{Tp} + \sqrt{V_{Tp}^2 + (\beta + 1)V_{dd}(V_{dd} - 2V_{Tp})}}{\beta + 1}$$

- The nMOS is now in its linear mode of operation. The derived equation does not apply beyond this input voltage.

nMOS linear, pMOS saturated

As the input voltage is raised still further, the output voltage will fall below V_{Tp}. The pMOS transistor is now in saturation regime. Equating currents, we get

$$K_n \left[(V_i - V_{Tn})V_o - \frac{1}{2}V_o^2 \right] = \frac{K_p}{2}(V_{dd} - V_{Tp})^2$$

which gives

$$\frac{1}{2}V_o^2 - (V_o - V_{Tn})V_o + \frac{(V_{dd} - V_{Tp})^2}{2\beta}$$

This can be solved to get

$$V_o = (V_i - V_{Tn}) - \sqrt{(V_i - V_{Tn})^2 - (V_{dd} - V_{Tp})^2/\beta}$$

Noise Margins

We find points on the transfer curve where the slope is -1. When the input is low and output high, we should use

$$V_o = V_{Tp} + \sqrt{(V_{dd} - V_{Tp})^2 - \beta(V_i - V_{Tn})^2}$$

Differentiating this equation with respect to V_i and setting the slope to -1, we get

$$V_{iL} = V_{Tn} + \frac{V_{dd} - V_{Tp}}{\sqrt{\beta(\beta + 1)}}$$

and

$$V_{oH} = V_{Tp} + \sqrt{\frac{\beta}{\beta + 1}}\,(V_{dd} - V_{Tp})$$

When the input is high and the output low, we use

$$V_o = (V_i - V_{Tn}) - \sqrt{(V_i - V_{Tn})^2 - (V_{dd} - V_{Tp})^2/\beta}$$

Differentiating with respect to V_i and setting the slope to -1, we get

$$V_{iH} = V_{Tn} + \frac{2}{\sqrt{3\beta}}(V_{dd} - V_{Tp})$$

and

$$V_{oL} = \frac{(V_{dd} - V_{Tp})}{\sqrt{3\beta}}$$

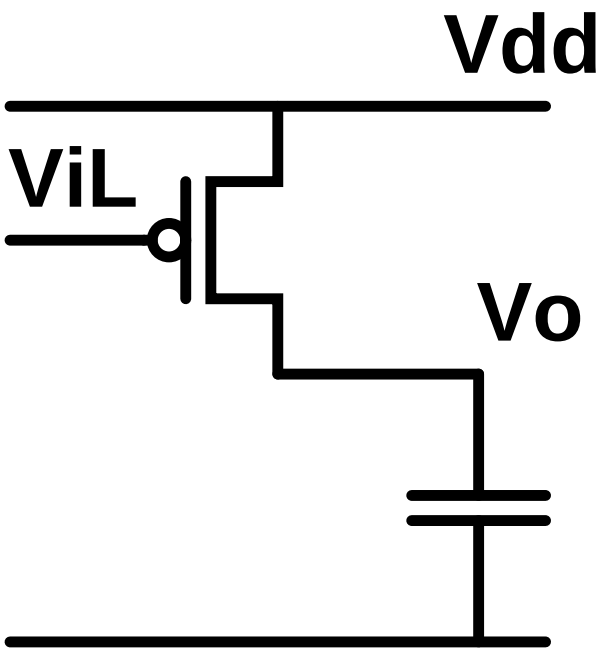
Ratioed Logic

To make the output 'low' value lower than V_{Tn}, we get the condition

$$\beta > \frac{1}{3}\left(\frac{V_{dd} - V_{Tp}}{V_{Tn}}\right)^2$$

- This places a requirement on the ratios of widths of n and p channel transistors. The logic gates work properly only when this equation is satisfied.
- Therefore this kind of logic is also called 'ratioed logic'.
- In contrast, CMOS logic is called ratioless logic because it does not place any restriction on the ratios of widths of n and p channel transistors for static operation.
- The noise margin for pseudo nMOS can be determined easily from the expressions for V_{iL}, V_{oL}, V_{iH}, V_{oH}.

Rise Time

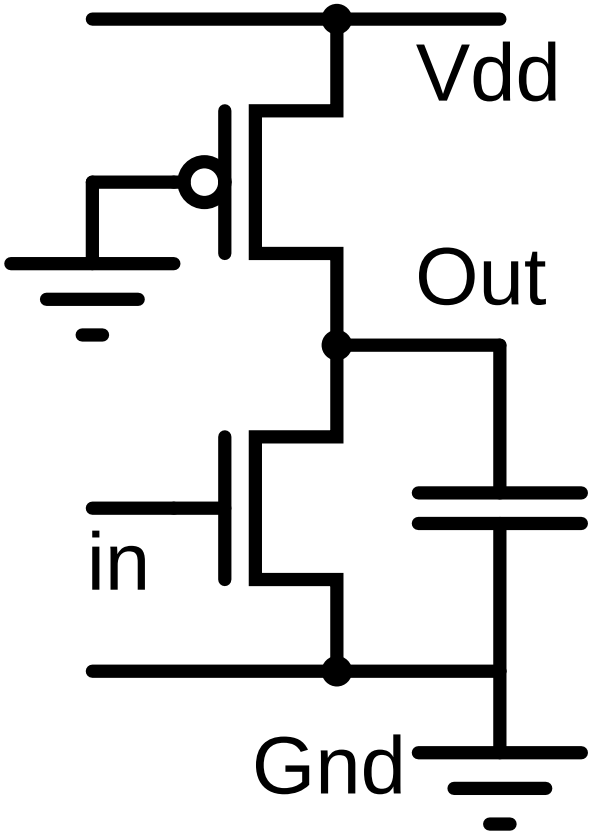


When the input is low, the nMOS is off and the output rises from ‘low’ to ‘high’.
The situation is identical to the charge up condition of a CMOS gate with the pMOS being biased with its gate at 0V.

This gives

$$\tau_{rise} = \frac{C}{K_p(V_{dd} - V_{Tp})}\left[\frac{2V_{Tp}}{V_{dd} - V_{Tp}} + \ln\frac{V_{dd} + V_{oH} - 2V_{Tp}}{V_{dd} - V_{oH}}\right]$$

Fall Time



Calculation of fall time is complicated by the fact that the pMOS load continues to dump current in the output node, even as the nMOS tries to discharge the output capacitor.
The nMOS needs to sink the discharge current as well as the drain current of the pMOS transistor.
Simplifying assumption:
pMOS current remains constant at its saturation value through the entire discharge process.

(This will result in a slightly pessimistic value of discharge time).

Fall Time

If we assume that the pMOS current remains constant at its saturation value,

$$I_p = \frac{K_p}{2}(V_{dd} - V_{Tp})^2$$

. We can write the KCL equation at the output node as:

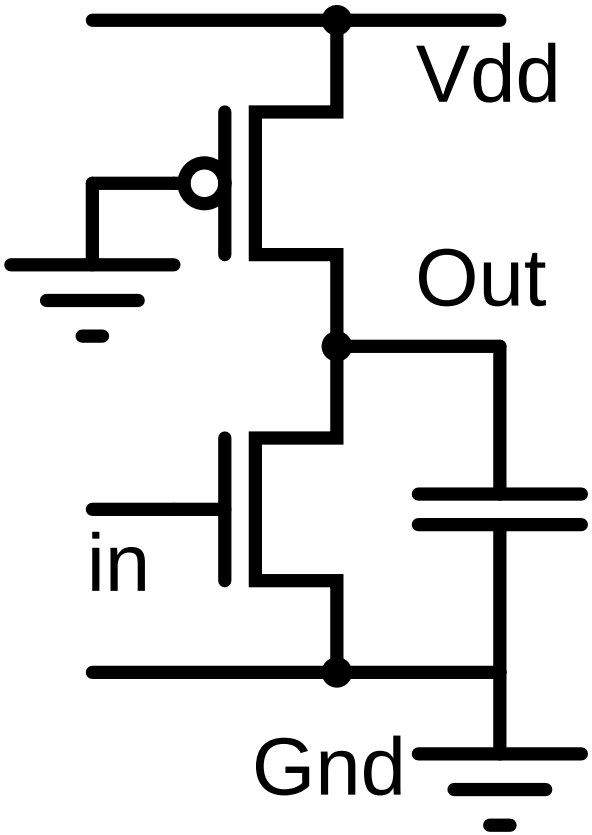
$$I_n - I_p + C\frac{dV_o}{dt} = 0$$

which gives

$$\frac{\tau_{fall}}{C} = -\int_{V_{dd}}^{V_{oL}} \frac{dV_o}{I_n - I_p}$$

We define $V_1 \equiv V_i - V_{Tn}$ and $V_2 \equiv V_{dd} - V_{Tp}$.

Fall Time



The integration range can be divided into two regimes.

- nMOS is saturated when $V_1 \leq V_o < V_{dd}$.
- It is in the linear regime when $V_{oL} < V_o < V_1$.

Fall Time

$$\frac{\tau_{fall}}{C} = -\int_{V_{dd}}^{V_1} \frac{dV_o}{\frac{1}{2}K_n V_1^2 - I_p} - \int_{V_1}^{V_{oL}} \frac{dV_o}{K_n(V_1 V_o - \frac{1}{2}V_o^2) - I_p}$$

so,

$$\frac{\tau_{fall}}{C} = \frac{V_{dd} - V_1}{\frac{1}{2}K_n V_1^2 - I_p} + \int_{V_{oL}}^{V_1} \frac{dV_o}{K_n(V_1 V_o - \frac{1}{2}V_o^2) - I_p}$$

Pseudo nMOS Inverter design

- We design the basic inverter and then scale device sizes based on the logic function being designed.
- The load device size is calculated from the rise time.

$$\tau_{rise} = \frac{C}{K_p(V_{dd} - V_{Tp})}\left[\frac{2V_{Tp}}{V_{dd} - V_{Tp}} + \ln\frac{V_{dd} + V_{oH} - 2V_{Tp}}{V_{dd} - V_{oH}}\right]$$

- Given a value of τ_{rise}, operating voltages and technological constants, K_p and hence, the geometry of the p channel transistor can be determined.

Pseudo nMOS Inverter design

- Geometry of the n channel transistor can be determined from static considerations.

$$V_{oL} = (V_{iH} - V_{Tn}) - \sqrt{(V_{iH} - V_{Tn})^2 - (V_{dd} - V_{Tp})^2/\beta}$$

- We take $V_{oL} = V_{Tn}$, and calculate β.
- But $\beta \equiv K_n/K_p$ and K_p is already known.
- This evaluates K_n and hence, the geometry of the n channel transistor.

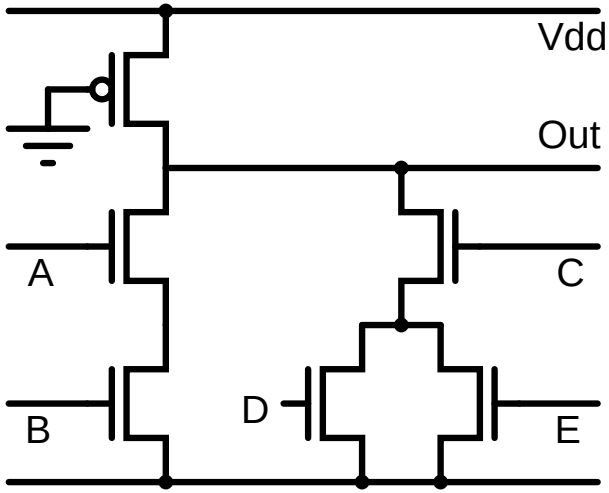
Conversion to other logic

- Once the basic pseudo nMOS inverter is designed, other logic gates can be derived from it.
- The procedure is the same as that for CMOS, except that it is applied only to nMOS transistors.
- The p channel transistor is kept at the same size as that for an inverter.

Conversion to other logic

- The logic is expressed as a sum of products with a bar (inversion) on top.
- For every '.' in the expression, we put the corresponding n channel transistors in series.
- For every '+', we put the n channel transistors in parallel. We scale the transistor widths up by the number of devices put in series.
- The geometries are left untouched for devices put in parallel.

$\overline{A.B + C.(D + E)}$ in pseudo-nMOS



- A and B are in series.
- The pair is in parallel with C which is in series with a parallel combination of D and E.

Implementation of $\overline{A.B + C.(D + E)}$ in pseudo-nMOS logic design style.

Complementary Pass gate Logic

- This logic family is based on multiplexer logic.
- Given a boolean function $F(x_1, x_2, \ldots, x_n)$, we can express it as:

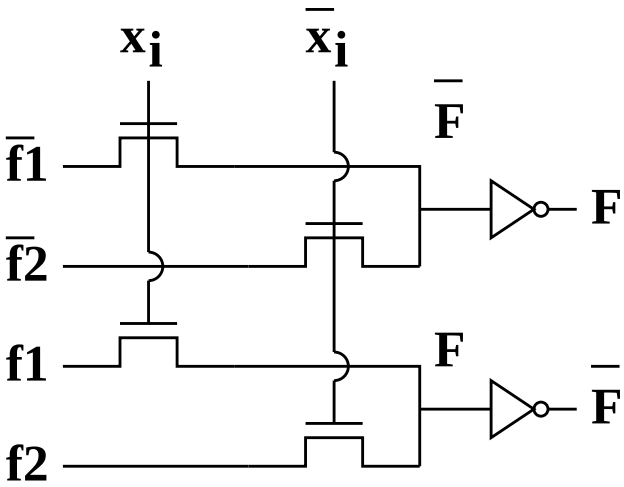
$$F(x_1, x_2, \ldots, x_n) = x_i \cdot f1 + \overline{x_i} \cdot f2$$

  where f1 and f2 are reduced expressions for F with x_i forced to 1 and 0 respectively.
- Thus, F can be implemented with a multiplexer controlled by x_i which selects f1 or f2 depending on x_i.
- f1 and f2 can themselves be decomposed into simpler expressions by the same technique.

Complementary Pass gate Logic

- To implement a multiplexer, we need both x_i and $\overline{x_i}$.
- Therefore, this logic family needs all inputs in true as well as in complement form.
- In order to drive other gates of the same type, it must produce the outputs also in true and complement forms.
- Thus each signal is carried by two wires.
- This logic style is called “Complementary Passgate Logic” or CPL for short.

Basic Multiplexer Structure



Pure passgate logic contains no ‘amplifying’ elements. Therefore, each logic stage degrades the logic level.

Hence, multiple logic stages cannot be cascaded.

We include conventional CMOS inverters to restore the logic level.

Ideally, the multiplexer should be composed of complementary pass gate transistors.

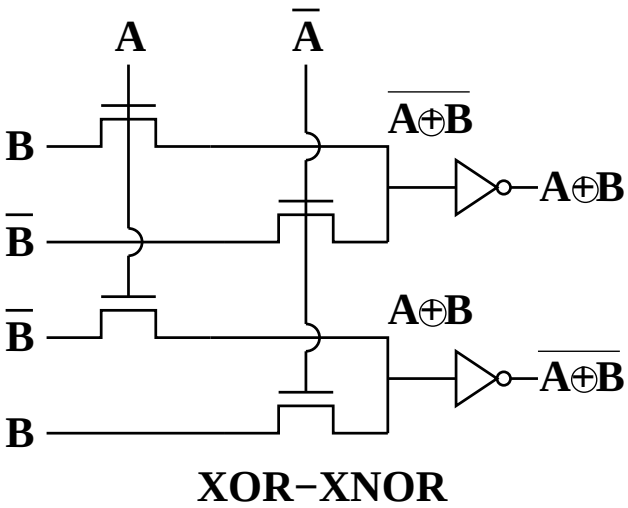
However, we shall use just n channel transistors as switches for simplicity.

Logic Design using CPL

- For any logic function, we pick one input as the control variable.
- Multiplexer inputs are decided by re-evaluating the function, fourcing this variable to 1 and zero respectively.
- Since both true and complement outputs are generated by CPL, we need fewer types of gates.
- For example, we do not need separate gates for AND and NAND functions.
- The same applies to OR-NOR, and XOR-XNOR functions.

Implementation of XOR and XNOR

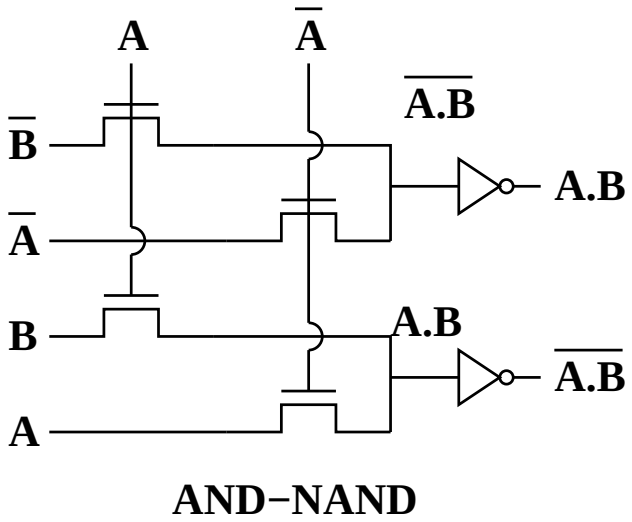
To take an example, let us consider the XOR-XNOR functions.



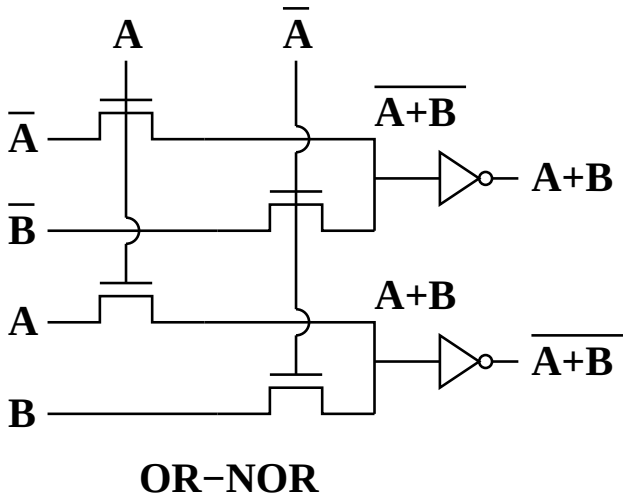
XOR−XNOR

- Because of the inverter, for XOR output, We calculate the XNOR function given by $A.B + \overline{A}.\overline{B}$.
- If we put A = 1, this reduces to B and for A = 0, it reduces to $\overline{B}$.
- For the XNOR output, we generate the XOR expression = $A.\overline{B} + \overline{A}.B$
- The expression reduces to $\overline{B}$ for A = 1 and to B for A = 0.

Implementation of AND-NAND and OR-NOR



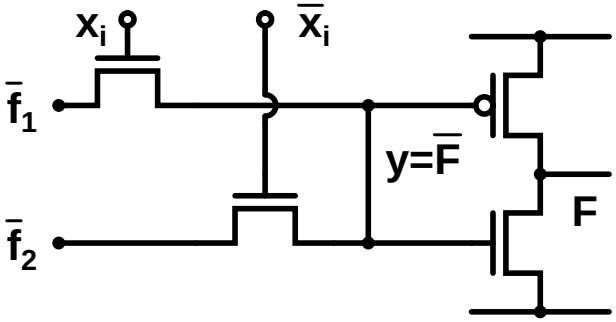
AND−NAND



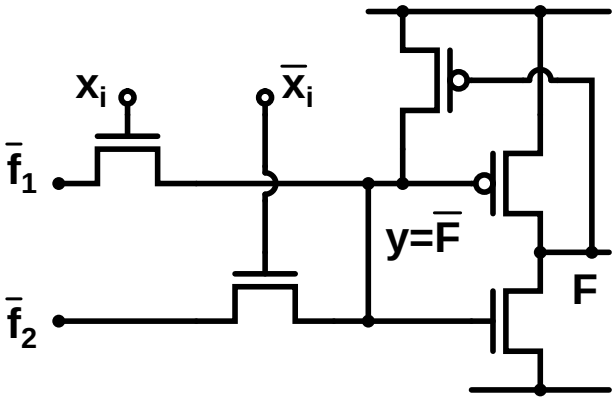
OR−NOR

- For AND, the mux should output $\overline{A.B}$ to be inverted by the buffer. This reduces to $\overline{B}$ when A = 1 and to 1 (= $\overline{A}$) when A = 0.
- Implementation of NAND, OR and NOR functions follows along the same lines.

Buffer Leakage Current

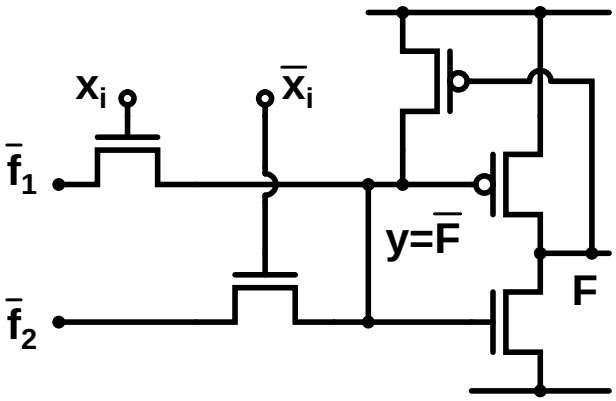


- The high output of the multiplexer (y) cannot rise above V_{dd} - V_{Tn}because we use nMOS multiplexers.
- Consequently, the pMOS transistor in the buffer inverter never quite turns off.
- This results in static power consumption in the inverter.



This can be avoided by adding a pull up pMOS with the inverter.

Use of Pullup PMOS



- When the multiplexer output (y) is ‘low’, the inverter output (F) is high. The pMOS is off and has no effect.
- When the multiplexer output (y) goes ‘high’, the inverter output falls and turns the pMOS on.

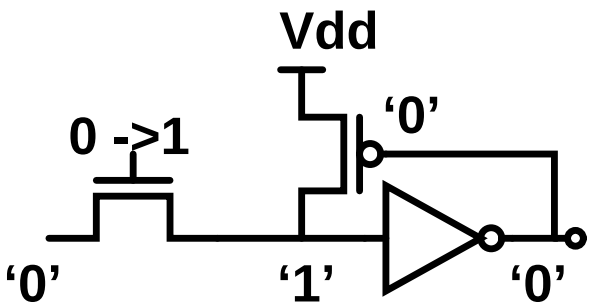
Now, even though the multiplexer nMOS turns ‘off’ as y approaches V_{dd} - V_{Tn}, the pMOS remains ‘on’ and takes the inverter input (y) all the way to V_{dd}.

This avoids leakage in the inverter.

Need for ratioing

The use of pMOS pullup brings up another problem.

Consider the equivalent circuit when the inverter output is ‘low’ and the pMOS is ‘on’.

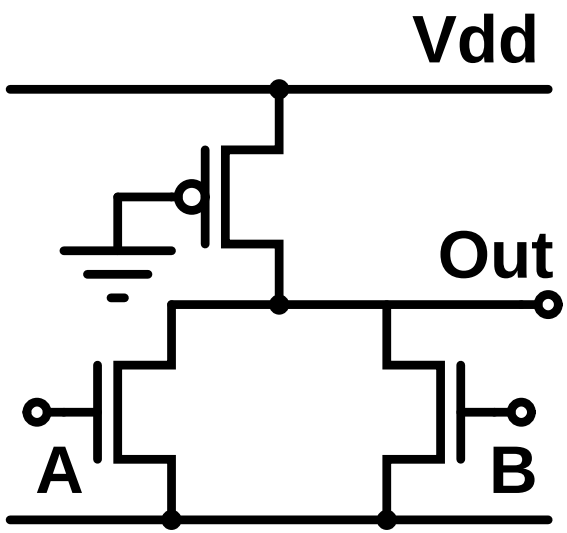


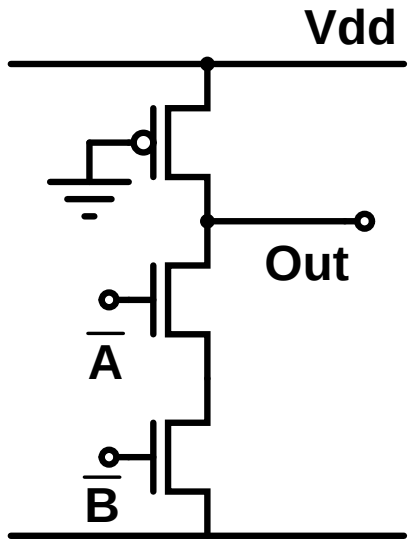
If the final output is ‘low’, the pMOS pullup is ‘on’. Now if the multiplexer output wants to go ‘low’, it has to fight the pMOS pullup - which is trying to keep this node ‘high’.

In fact, the multiplexer n transistor and the pull up p transistor constitute a pseudo nMOS inverter.

Therefore, the multiplexer output cannot be pulled low unless the transistor geometries are appropriately ratioed.

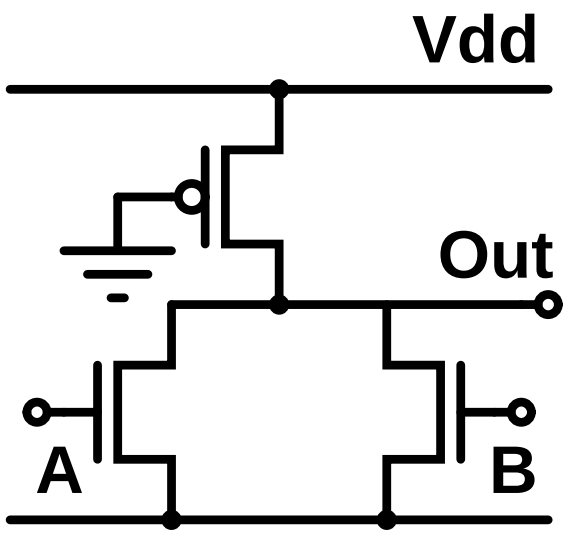
Improving Pseudo nMOS

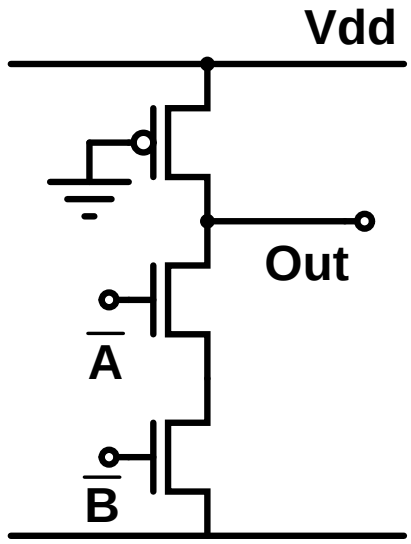




- In the pseudo-nMOS NOR circuit on the left, static power is consumed when the output is 'LOW'
- We would like to turn the pMOS off when A OR B is TRUE.
- The OR logic can be constructed by using a Pseudo-nMOS NAND of $\overline{A}$ and $\overline{B}$ as in the circuit on the right.
- But then what about the pMOS drive of this circuit?

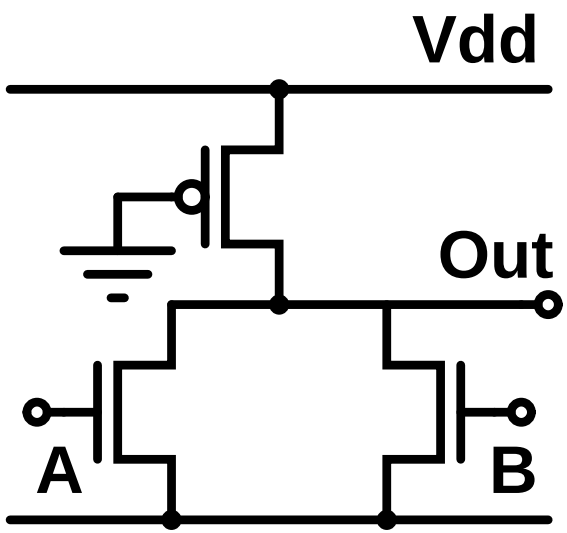
Improving Pseudo nMOS

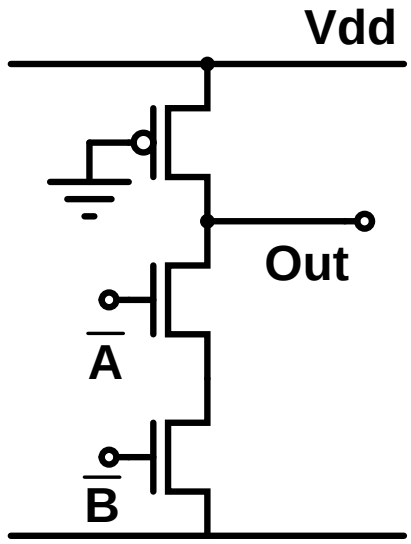




- In the pseudo-nMOS NOR circuit on the left, static power is consumed when the output is 'LOW'
- We would like to turn the pMOS off when A OR B is TRUE.
- The OR logic can be constructed by using a Pseudo-nMOS NAND of $\overline{A}$ and $\overline{B}$ as in the circuit on the right.
- But then what about the pMOS drive of this circuit?

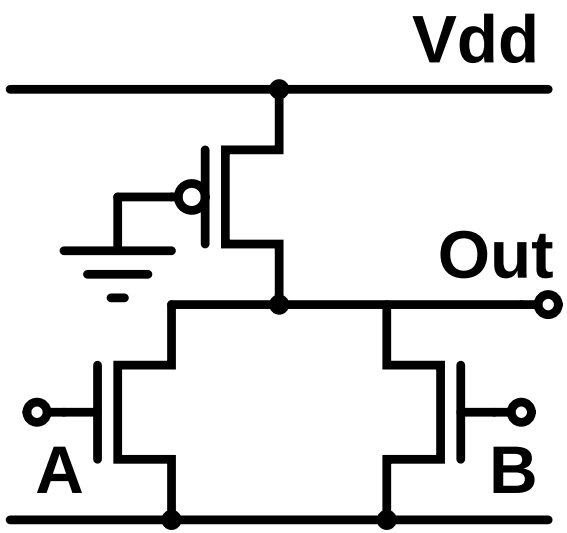
Pseudo nMOS without Static Power

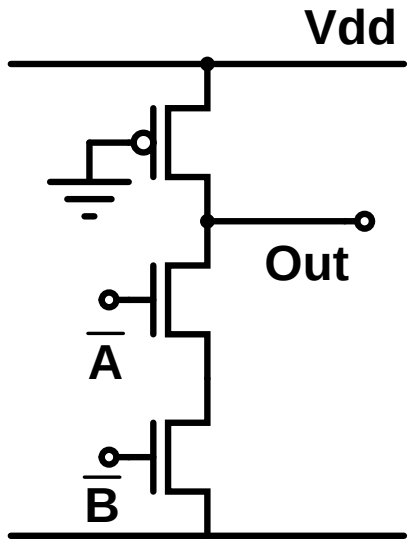




- The output of the circuit on the right is 'LOW' when both $\overline{A}$ and $\overline{B}$ are 'HIGH' (A = B = 0).
- We would like to turn *its* pMOS off when NOR of A and B is 'TRUE'
- But this can be provided by the circuit on the left!
- So the two circuits can drive each other's pMOS transistors and avoid static power consumption.

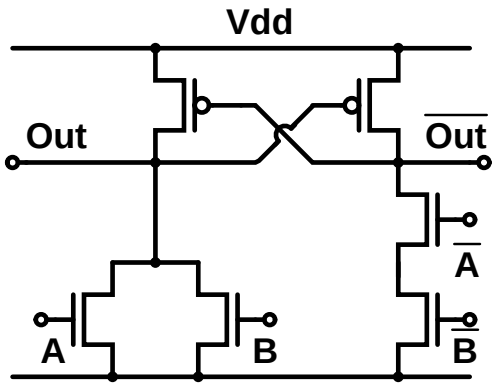
Pseudo nMOS without Static Power





- The output of the circuit on the right is 'LOW' when both $\overline{A}$ and $\overline{B}$ are 'HIGH' (A = B = 0).
- We would like to turn *its* pMOS off when NOR of A and B is 'TRUE'
- But this can be provided by the circuit on the left!
- So the two circuits can drive each other's pMOS transistors and avoid static power consumption.

Cascade Voltage Switch Logic



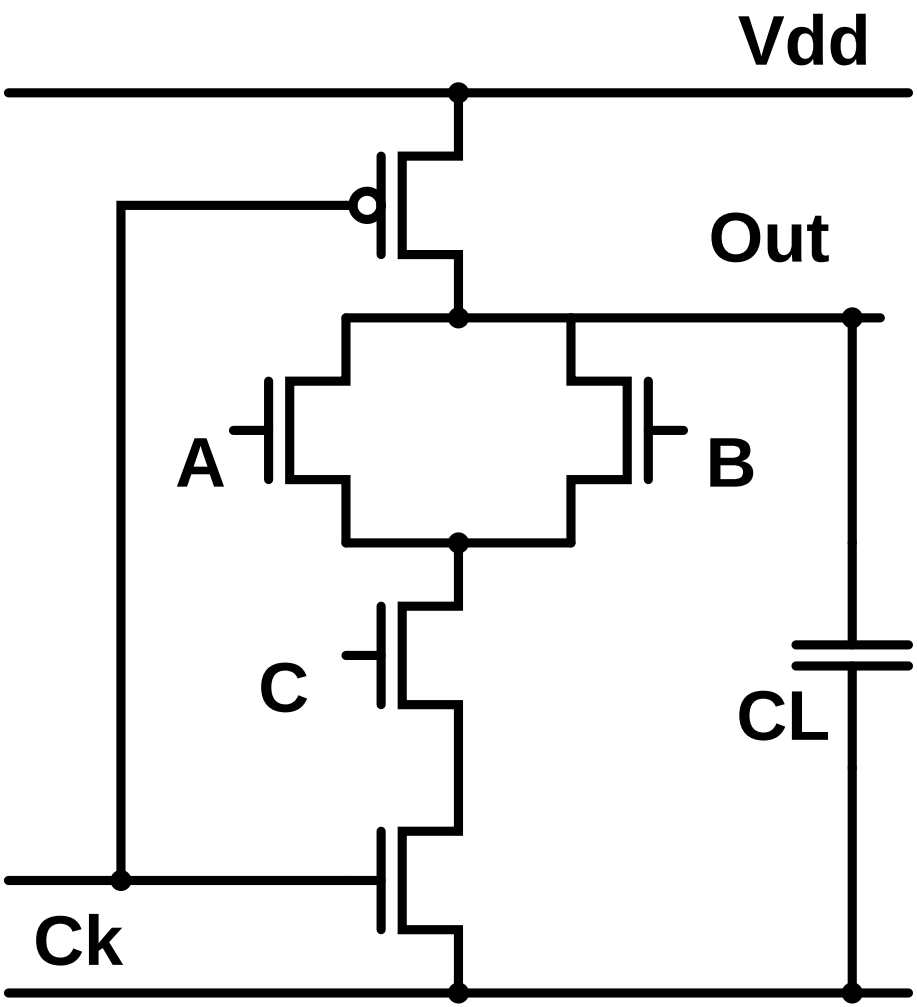
This kind of logic is called Cascade Voltage Switch Logic (CVSL).
It can use any network f and its complementary network $\overline{f}$ in the two cross-coupled branches.

- Like CMOS static logic, there is no static power consumption.
- Like CPL, this logic requires both True and Complement signals. It also provides both True and complement outputs. (Dual Rail Logic).
- Like pseudo nMOS, the inputs present a single transistor load to the driving stage.
- The circuit is self latching. This reduces ratioing requirements.

Dynamic logic

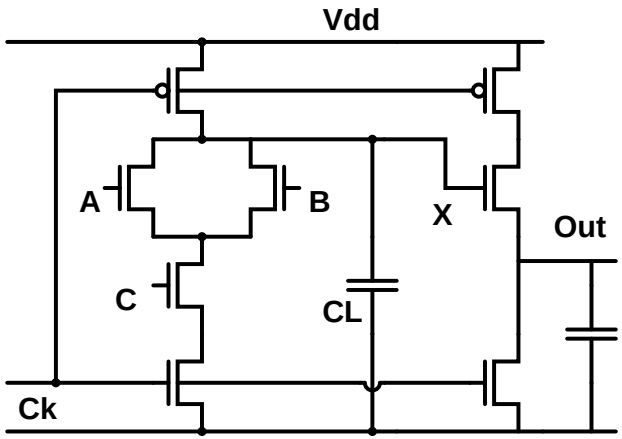
- In this style of logic, some nodes are required to hold their logic value as a charge stored on a capacitor.
- These nodes are not connected to their ‘drivers’ permanently.
- The ‘driver’ places the logic value on them, and is then disconnected from the node.
- Due to leakage etc., the logic value cannot be held indefinitely.
- Dynamic circuits therefore require a *minimum* clock frequency to operate correctly.
- Use of dynamic circuits can reduce circuit complexity and power consumption substantially.

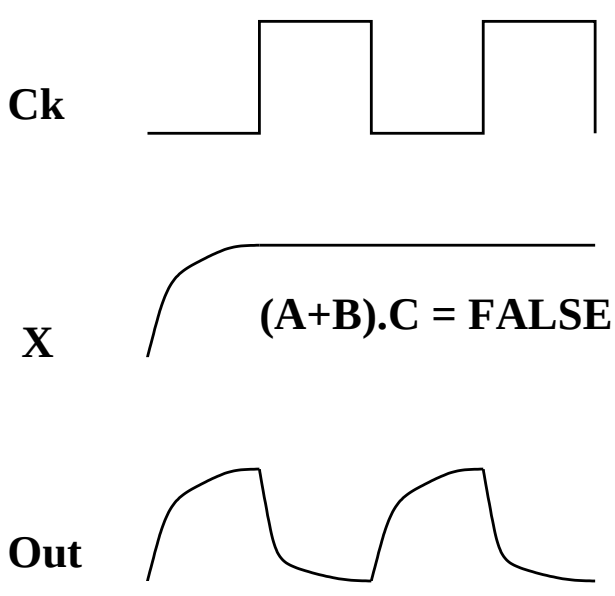
A CMOS dynamic logic circuit

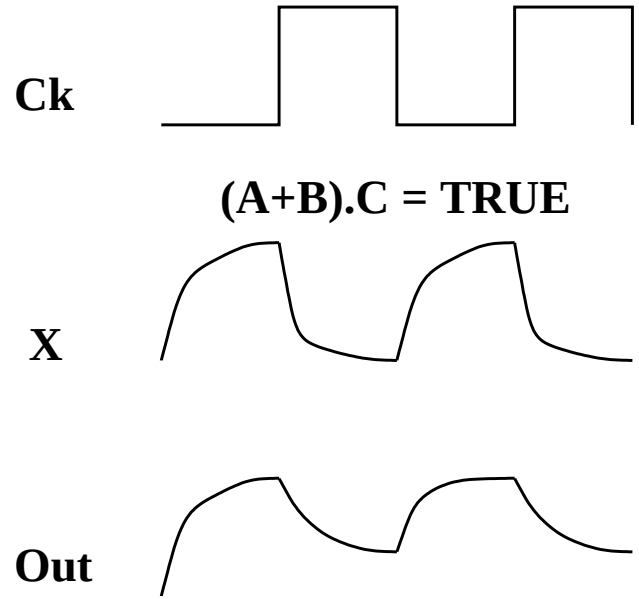


- When the clock is low, pMOS is on and the bottom nMOS is off.
- The output is 'pre-charged' to 1 unconditionally.
- When the clock goes high, the pMOS turns off and the bottom nMOS comes on.
- The circuit then conditionally discharges the output node, if (A+B).C is TRUE.
- This implements the function $\overline{(A+B).C}$.

Problem with Cascading





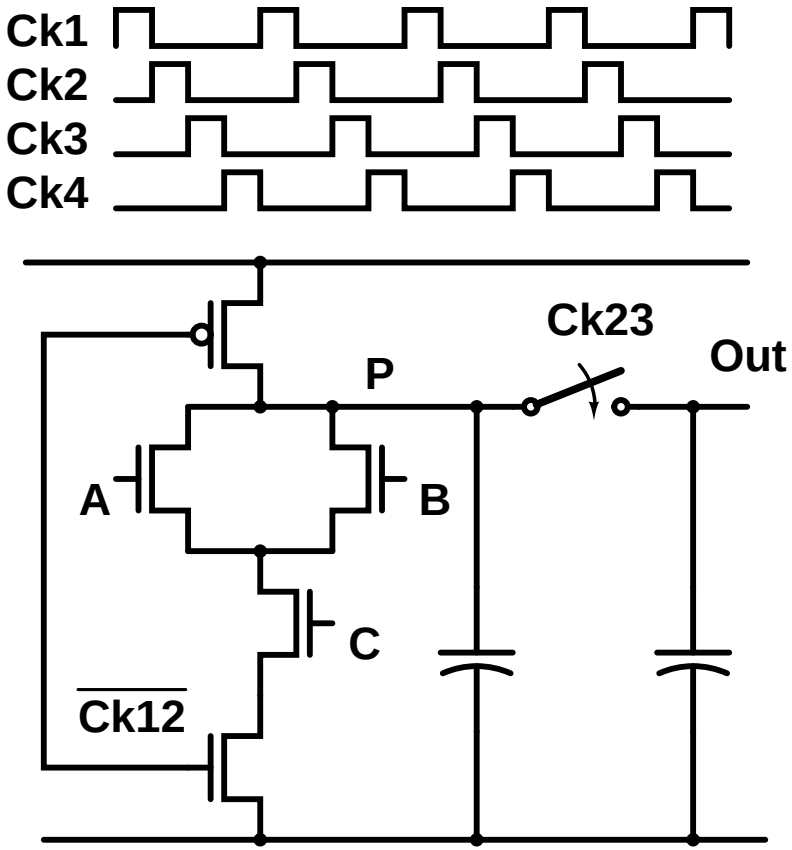


There is no problem when (A+B).C is false. X pre-charges to 1 and remains at 1.

When (A+B).C is TRUE, X takes some time to discharge.

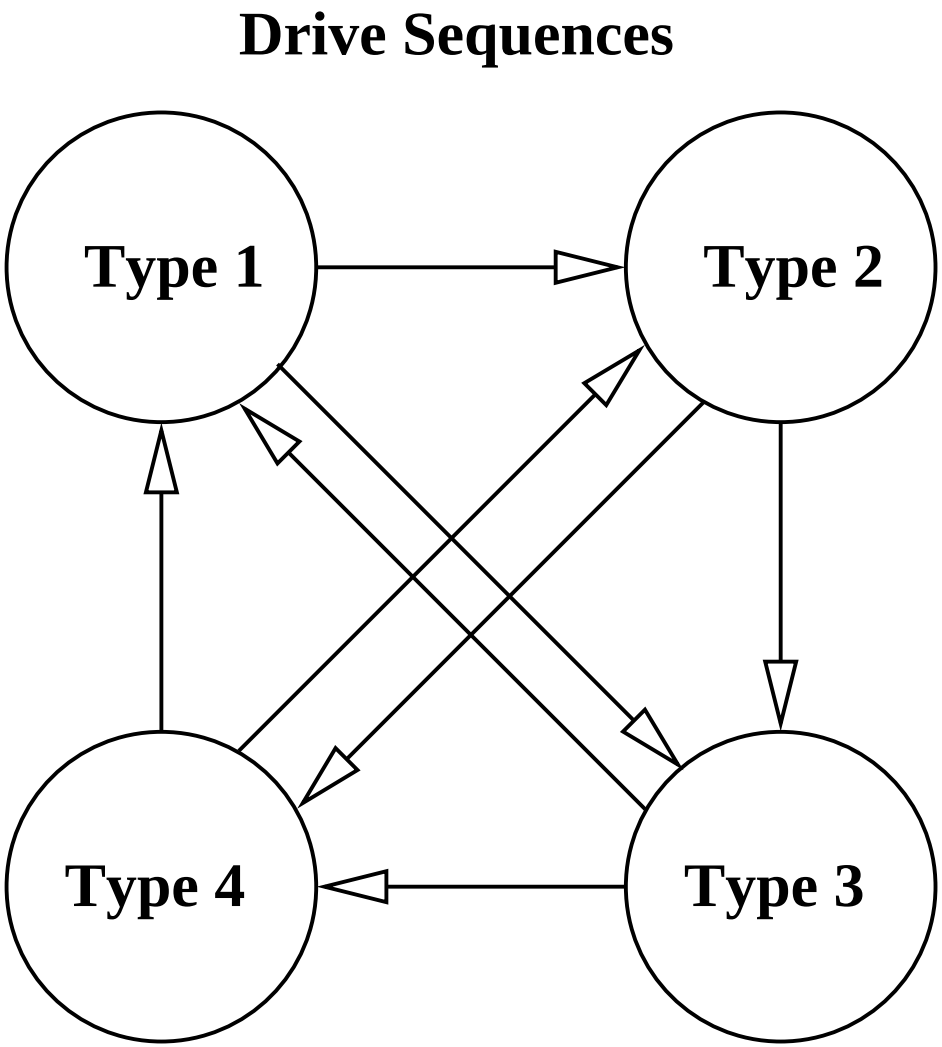
During this time, charge placed on the output leaks away as the input to nMOS of the inverter is not 0.

4 Phase Dynamic Logic



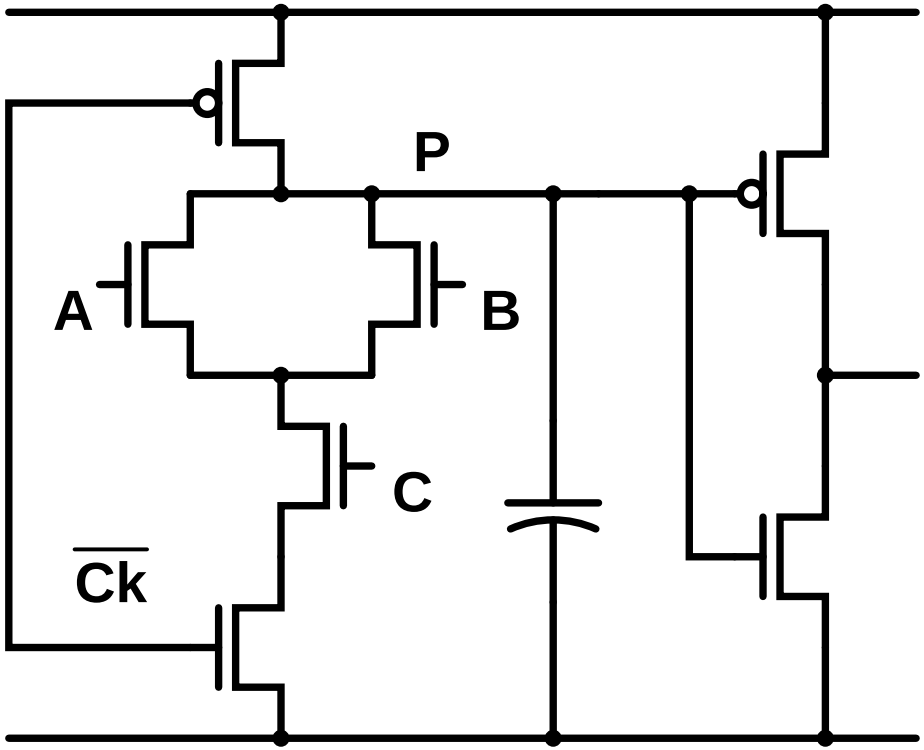
- The problem can be solved by using a 4 phase clock.
- In phase 1 node P is pre-charged.
- In phase 2 P and output are pre-charged.
- In phase 3 The gate evaluates.
- In phases 4 and 1, the output is isolated from the driver and remains valid.
- This is called a type 3 gate. It evaluates in phase 3 and is valid in phases 4 and 1.

Drive cycles



- A type 3 gate can drive a type 4 or a type 1 gate.
- similarly, type 4 will drive types 1 and 2; type 1 will drive types 2 and 3; and type 2 will drive types 3 and 4.
- We can use a 2 phase clock if we stick to type 1 and type 3 gates (or type 2 and type 4 gates) as these can drive each other.

Domino Logic

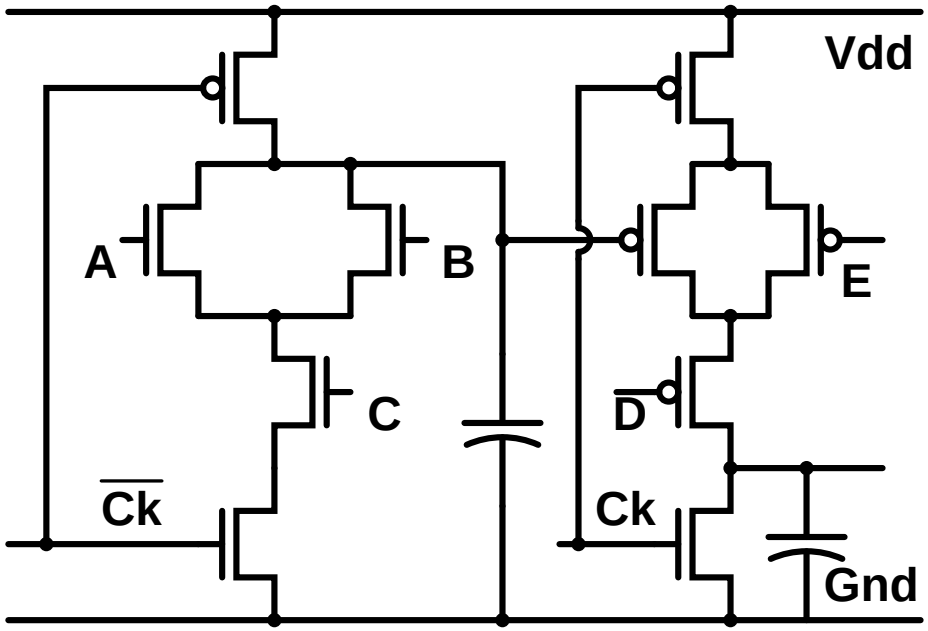


Another way to eliminate the problem with cascading logic stages is to use a static inverter after the CMOS dynamic gate. The output is '0' when it is not valid. Therefore, it does not affect the evaluation of the next gate.

However, the logic is non-inverting. Therefore, it cannot be used to implement any arbitrary logic function.

Zipper Logic

Instead of using an inverter, we can alternate n and p evaluation stages.



A, B, C must be from p stages.
D and E must be from n stages.

- The n stage is pre-charged high, but it drives a p stage.
- A high pre-charged stage will keep the p evaluation stage off, which will not cause any malfunction.
- The p stage will be pre-discharged to ‘low’, which is safe for driving n stages.

This kind of logic is called zipper logic.