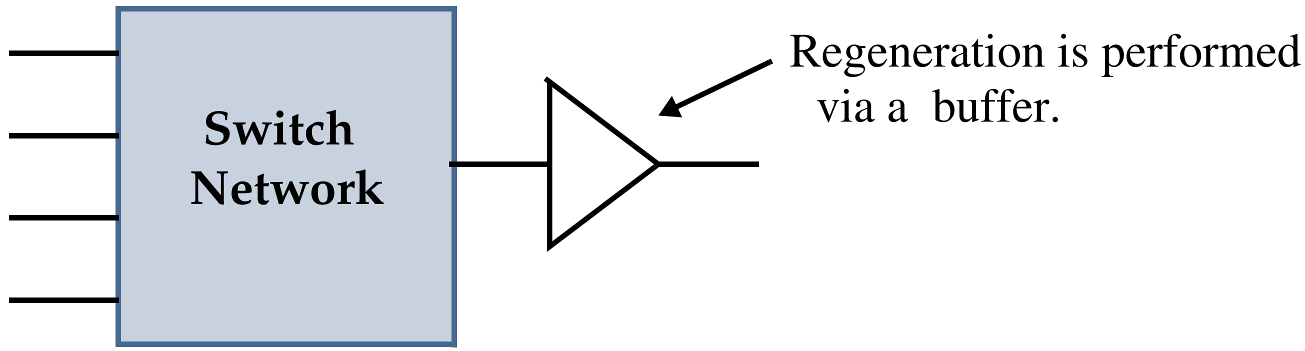
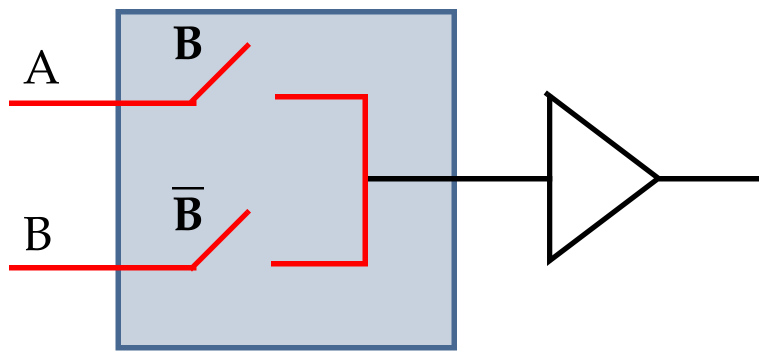
Pass Gate Logic

An alternative to implementing complex logic is to realize it using a logic network of pass transistors (switches).



We have already observed a series connection of two switches implements AND while a parallel connection implements OR.



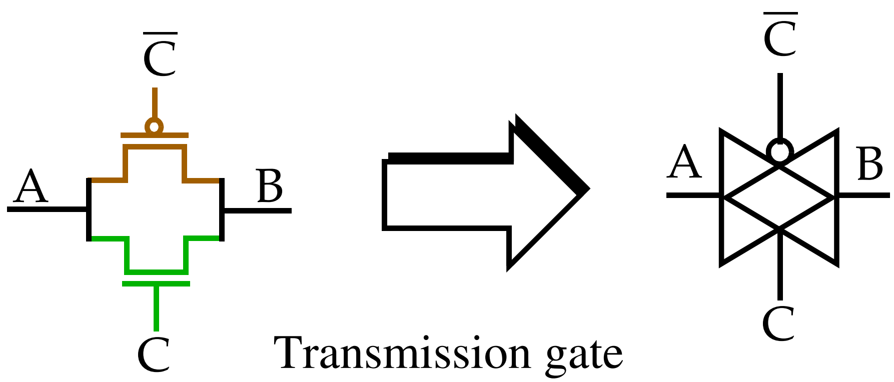
$\overline{\mathrm{B}}$ is not redundant, it ensures a low impedance path exists when B is low.

Pass Gate Logic

Advantage: fast and simple.

Complex gates can be implemented using minimum number of transistors, which also reduces parasitics.

Static and dynamic performance depends on a switch with low parasitic resistance and capacitance.

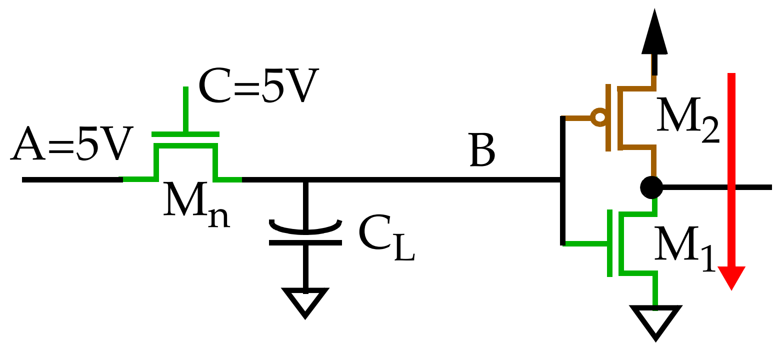


Transmission gate

Therefore, pass gate networks are often constructed from bi-directional transmission gates.

Pass Gate Logic

Both transistors are important:



Here, M_n turns off when V_B reaches (5 - V_{Tn}) or approximately 3.5V!

Note, the V_{Tn} is increased due to the **body effect**.

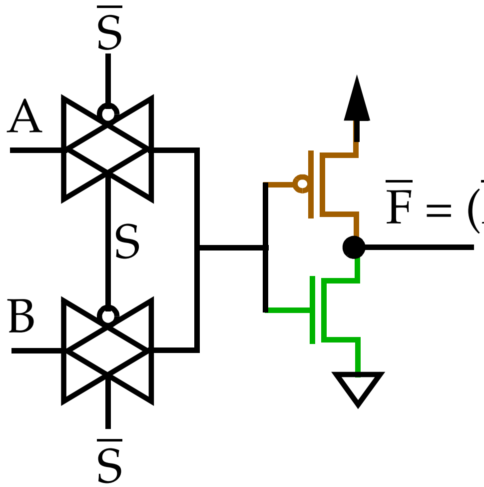
This reduces the *noise margin* and increases *static power* dissipation.

Also, the **resistance** of the switch increases dramatically when the output voltage reaches V_{in} -V_{Tn} (linear mode).

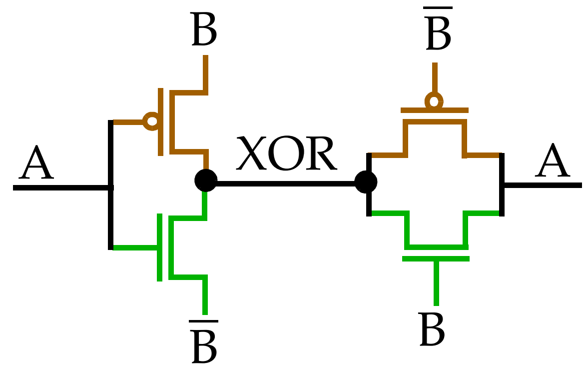
The combination of both an PMOS and NMOS avoids this problem but requires that the control and its complement be available.

Pass Gate Logic

Transmission gates can implement complex gates very efficiently



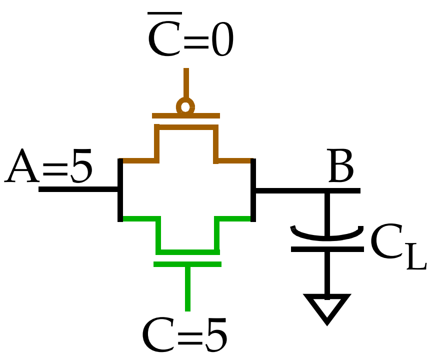
2-to-1 MUX requires 6 transistors



XOR requires 6 transistors

Design Issues

- Resistance



Parallel connection of resistances R_n and R_p

$$R_n = (V_{DD} - V_{out})/I_n$$

$$R_p = (V_{DD} - V_{out})/I_p$$

Currents are dependent on V_{out} and operation region

Pass Gate Logic Design Issues

- Resistance (cont).

  During the *low-to-high* transition, the pass transistors pass through several operation modes.

  As V_{GS} is always equal to V_{DS}, the NMOS is either in saturation or off.

  The V_{GS} of the PMOS is V_{DD}, and the device changes from saturation to linear.

  - $V_{out} < |V_{Tn}|$: NMOS and PMOS saturated.
  - $|V_{Tp}| < V_{out} < V_{DD} - V_{Tn}$: NMOS saturated, PMOS linear.
  - $V_{DD} - V_{Tn} < V_{out}$: NMOS cutoff, PMOS linear.

  It is important to incorporate the *body effect* when computing I_p and I_n.

  The expression for the resistance of a pass gate *without* the body effect.

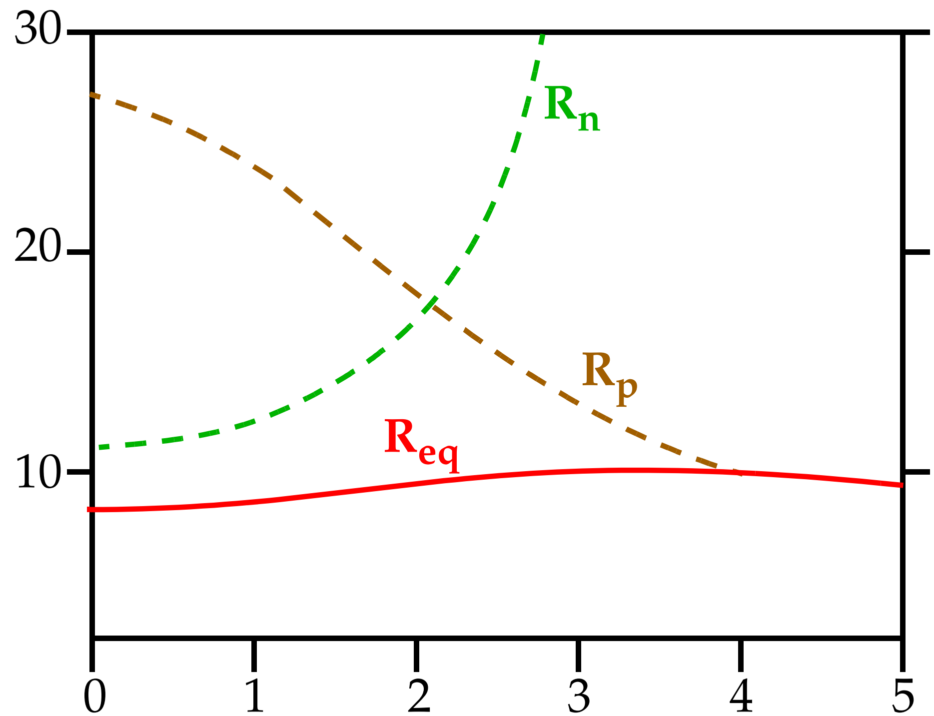
$$R_{eq} \approx \frac{1}{k_n(V_{DD} - V_{Tn}) + k_p(V_{DD} - |V_{Tp}|)}$$

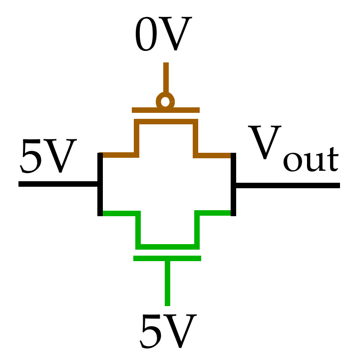
Pass Gate Logic Design Issues

- Resistance (cont).

  Simulated values of :

$$R_{eq} = R_p /\!/ R_n$$

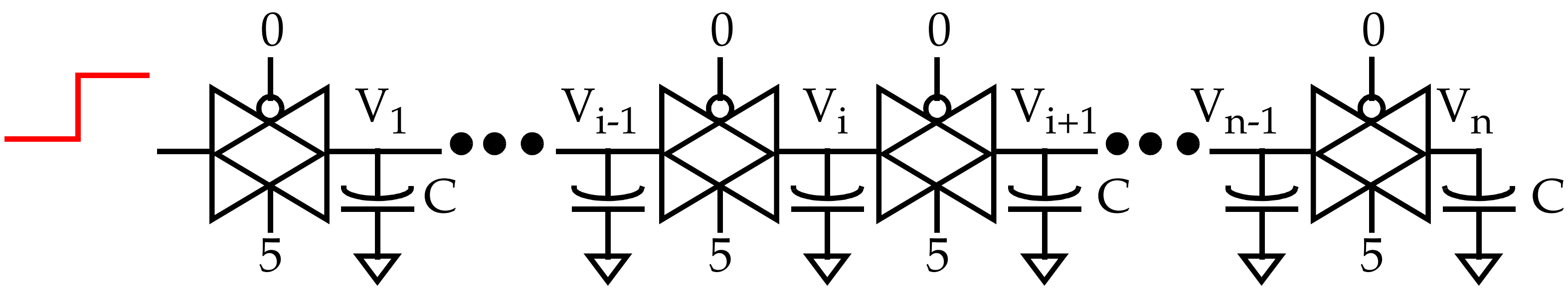




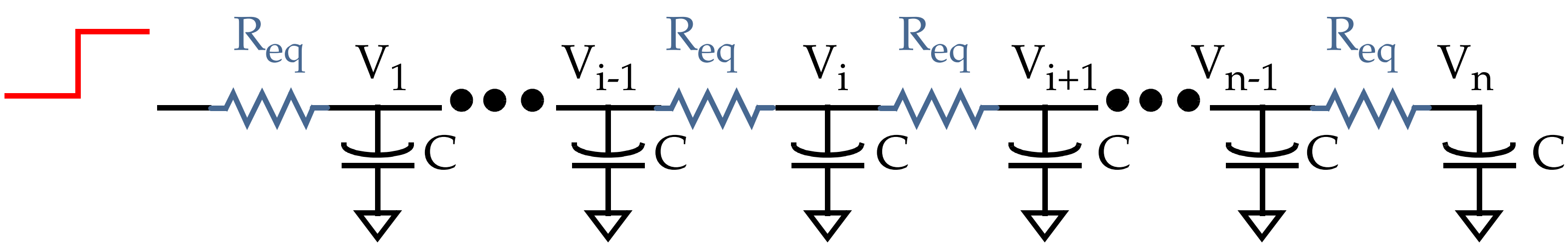
R_{eq} is relatively constant at 10 kΩ so a **constant resistance** switch model is reasonable.

Pass Gate Logic Design Issues

- Delay



In order to analyze the response, let's replace the pass gates with R_{eq}s.



Delay is found by solving a set of differential equations of the form:

$$\frac{\partial V_i}{\partial t} = \frac{1}{R_{eq}C}(V_{i+1} + V_{i-1} - 2V_i)$$

Pass Gate Logic Design Issues

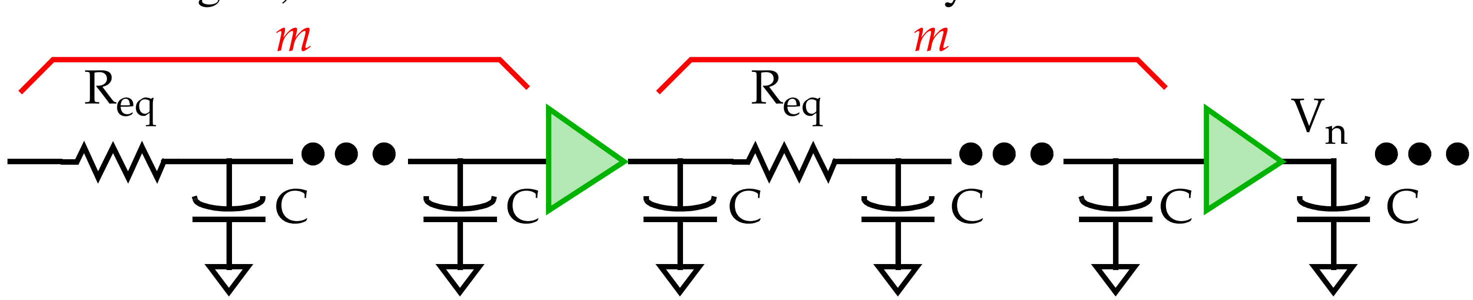
- Delay (cont).

An estimate of the dominant time constant at the output of n pass gates:

$$\tau(V_n) = \sum_{k=0}^{n} CR_{eq}k = CR_{eq}\frac{n(n+1)}{2}$$

Propagation delay is proportional to n^2!

For large n, it is better to break the chain every m switches and insert buffers:



Total delay assuming buffer delay is t_{buf} is:

$$t_p = 0.69\left[\frac{n}{m}CR_{eq}\frac{m(m+1)}{2}\right] + \left(\frac{n}{m} - 1\right)t_{buf} = 0.69\left[CR_{eq}\frac{n(m+1)}{2}\right] + \left(\frac{n}{m} - 1\right)t_{buf}$$

Pass Gate Logic Design Issues

- Delay (cont).

  Here, delay exhibits only a linear dependence on the # of switches *n*.

  The optimal number of switches, m_{opt}, between buffers is found:

$$\frac{\partial t_p}{\partial m} = 0 \longrightarrow m_{opt} = 1.7\sqrt{\frac{t_{pbuf}}{CR_{eq}}}$$

  As t_{buf} increases, the number of switches grows.

  In current technologies, m_{opt} is typically 3 or 4.

  For example, assume $R_{eq} = 10k\Omega$, $C = 10fF$, and $t_{pbuf} = 500ps$.

  This yields an optimal value of *m* equal to **3.8**.

  Therefore, a buffer every 4 transmission gates is suggested.

Pass Gate Logic Design Issues

- Transistor sizing

    Pass gate logic family is a member of the *ratioless* logic class.

    The dc characteristics are not affected by the sizes.

    Performance, to the first order, is **not impacted** by changing the W/L.

    - Increasing the size reduces the resistance, but this is offset by the increase in diffusion capacitance.

    Therefore, minimum sized devices should ALWAYS be used, unless the chain drives a significant external load capacitance.

    In this case, ordering transistors from largest to smallest in the pass gate chain will help reduce delay.

    - This is analogous to the argument given earlier for logic gate transistors close to the output.

NMOS-Only Transmission Gate

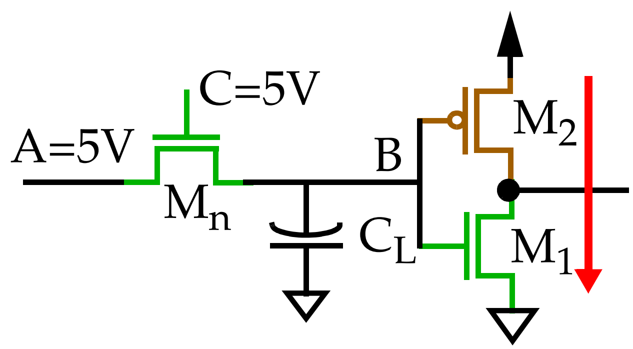
Disadvantages of pass gate:

- Requires both NMOS and PMOS, in different wells.
- Both true and complemented polarities of the control signal needed.
- Parallel connection of both transistors increases node capacitance.

Therefore, an *NMOS-only* version is advantageous.

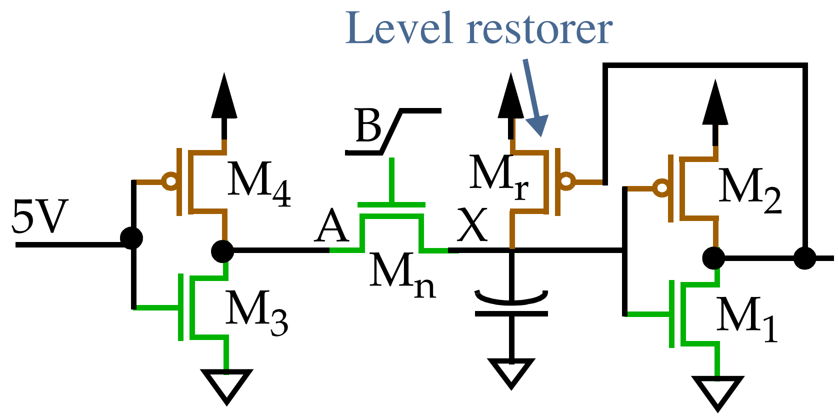
Problems:

- Reduced noise margins due to the threshold voltage drop.
- Static power consumption.



NMOS-Only Transmission Gate

One solution is to add a PMOS device, called a **level restorer**.



The output of the inverter is "feedback" as a control signal.

- It turns on when the inverter output goes low ($V_{out} < V_{DD} - |V_{tp}|$) and restores node *X* to V_{DD}.
- This eliminates the static power consumed.

However, the size of the PMOS transistor is important, since a conflict is created during switching.

- For example, assume node *A*=0, storage node *X*=V_{DD} and *B*=0->1.
- A conducting path exists from V_{DD}-M_r-M_n-M_3-GND.

NMOS-Only Transmission Gate

Let R_r, R_n and R_3 represent the resistances of transistors M_r, M_n and M_3.

If R_r is too small, it will be *impossible* to bring node *X* below V_M.

This is called the **writability problem**, used in reference to feedback circuits.

Let's simplify the analysis of finding the switching point by grounding M_r's input (open the feedback loop).

Assume M_r is in *linear* mode, M_n is in *saturation* and V_A is close to GND.

$$I = k_3(V_{DD} - V_{Tn})V_A \qquad \text{(linear)} \qquad (1)$$

$$I = \frac{k_n}{2}(V_B - V_A - V_{Tn})^2 \qquad \text{(for } V_X = V_M\text{)} \qquad (2)$$

$$I = k_r\left[(V_{DD} - |V_{Tp}|)\left((V_{DD} - V_M) - \frac{(V_{DD} - V_M)^2}{2}\right)\right] \qquad (3)$$

I is set by (3), which allows V_A to be found via (1) and then V_B as a function of the k-parameters (the objective).

NMOS-Only Transmission Gate

Let's set the condition that $V_B < V_{DD}$ -- in other words, some value of V_B less than V_{DD} will set $V_X < V_M$ (which allows the inverter to switch).

Assume the sizes of M_3 and M_n are identical and V_{DD}=5V, $V_{Tn}=|V_{Tp}|$=0.75V and V_M=2.5V:

$$V_B = 3.87\sqrt{\frac{k_r}{k_n}} + 1.76\frac{k_r}{k_n} + 0.75 \le 5V$$

The boundry condition for this constraint to be valid is $m = k_n/k_p > 1.55$.

Smaller values do not allow the inverter to switch.

Using a value of 3 is reasonable, which amounts to making the NMOS pass gate transistor equal to PMOS restoring device.

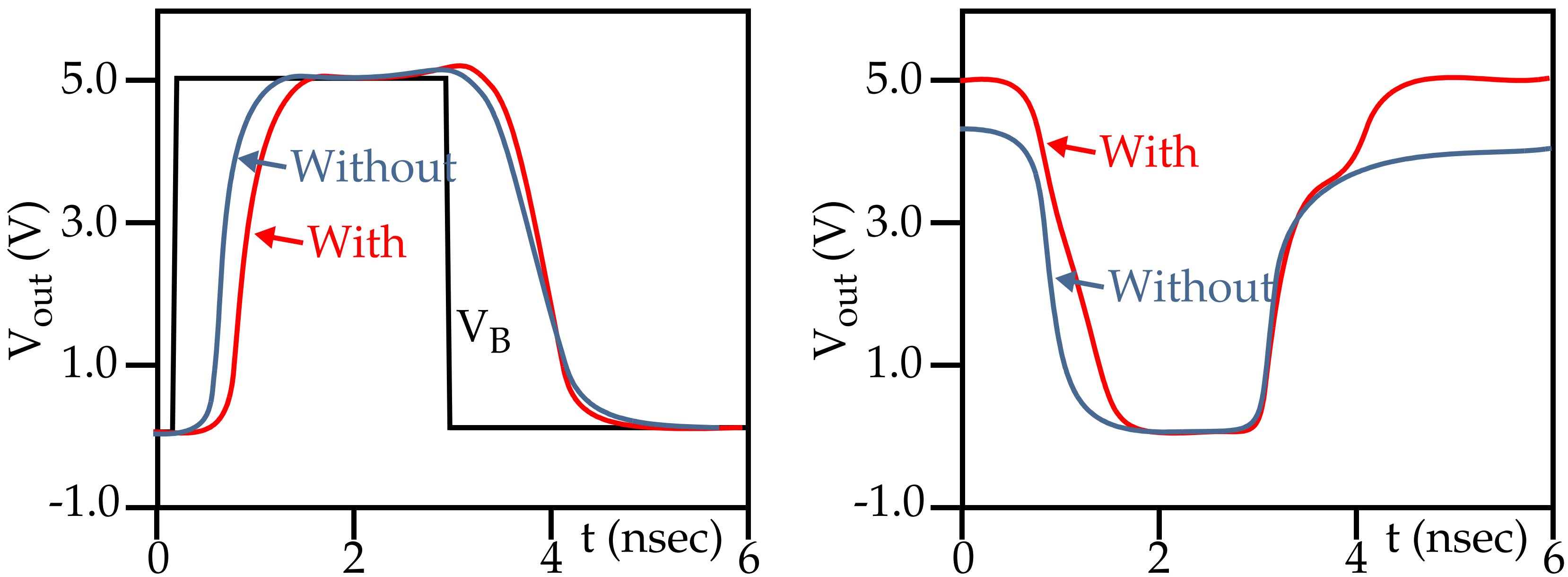
What about performance?

Adding the level restorer increases the capacitance at V_X.

Also, the rise time of the inverter is slowed due to the fight.

NMOS-Only Transmission Gate

However, the fall time is improved slightly.



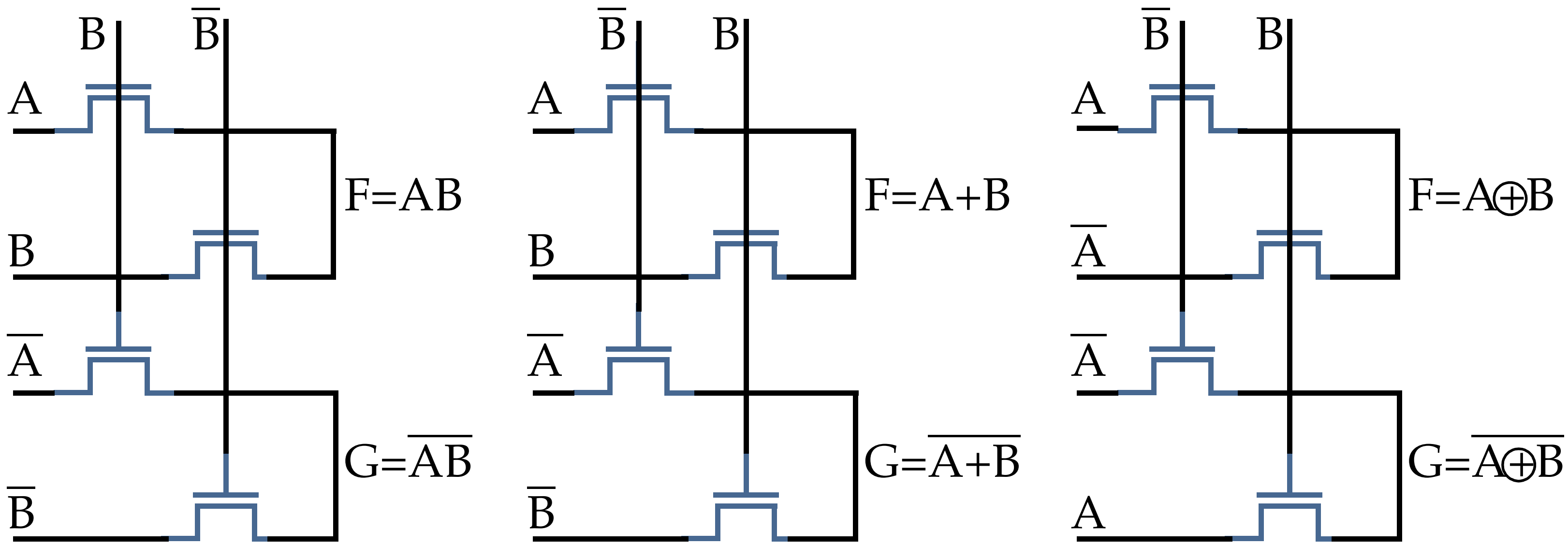
A second method of implementing NMOS-only pass gate networks is to change V_T (if your manufacturer supports it).

A zero V_T transistor for M_n (a natural device) is one possibility.

This logic style is called **Complementary Pass-Transistor Logic** (CPL).

CPL

Examples:



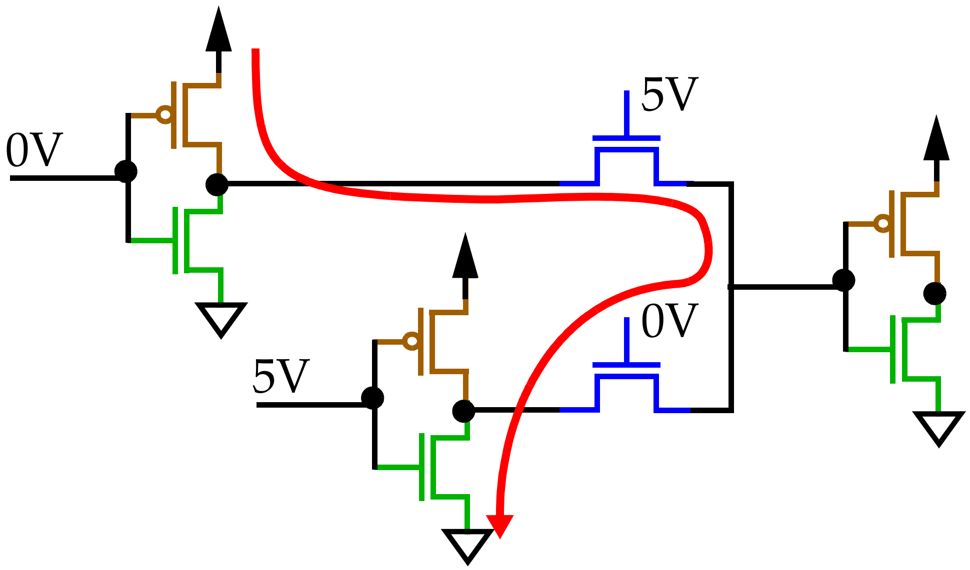
Properties:

- They are *differential* circuits.

  Eliminates inverters and allows minimal implementations, e.g., XOR.
- CPL is *static* (low impedance connection to V_{DD} and GND).
- V_T (including body effect) is reduced to below $|V_{Tp}|$, eliminating *static power* in successor gates.
- The design is *modular* -- all gates use exactly the same topology.

CPL

The main disadvantages is that turning off a zero-V_T device is hard (plus it has a reduced noise margin).



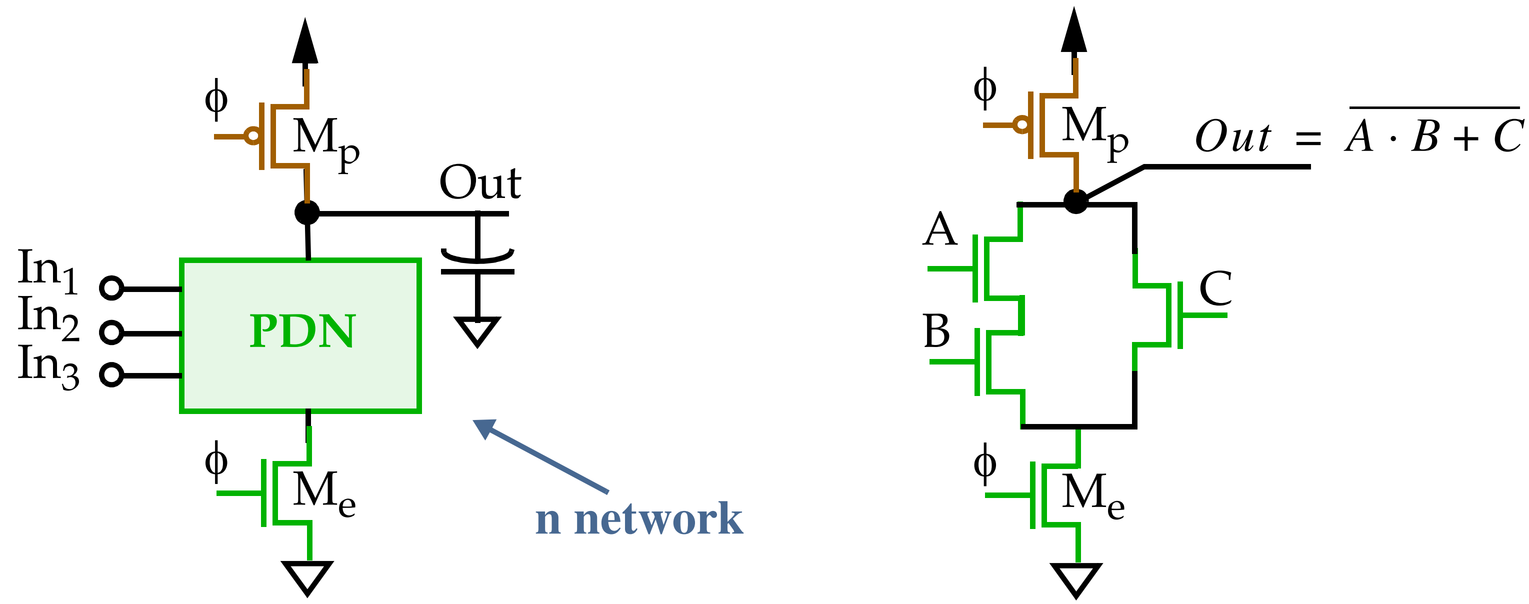
Note that a 4-input NAND requires three 2-input NANDs + buffer for **14** transistors, which is > **8** for the full complementary version!

The applicability of CPL is strongly dependent on the logic function to be implemented, e.g. 2-transistor XOR good for multipliers and adders.

CPL is extremely fast and efficient. Routing overhead is significant however.

Dynamic Logic

Dynamic logic reduces the fan-in, similar to pseudo-NMOS, **without** the static power consumption.



Precharge

When $\phi=0$, the output node *Out* is precharged to V_{DD} by M_p.

Evaluation:

When $\phi=1$, M_e is on and node *Out* discharges conditionally, depending on the value of the input signals.

Dynamic Logic

If no path exists during evaluate, then *Out* remains high via C_L (diffusion, wiring and gate capacitance).

Note that once *Out* is discharged, it cannot be recharged.

Therefore, the inputs can make *at most* **one transition** during evaluation.

Properties:

- The logic function is implemented in the NMOS pull-down network.
- The # of transistors is **N+2** instead of 2N
- It is non-ratioed (noise margin does not depend on transistor ratios).
- It only consumes dynamic power.
- Faster switching due to reduced internal and downsteam capacitance.

Steady-state behavior

V_{OL} and V_{OH} are GND and V_{DD}.

Our standard definitions of noise margins and switching thresholds *do not include time*, which is required in this case.

Dynamic Logic

Steady-state behavior (cont):

For example, noise margins depend on the length of the evaluate.

If *clk* is too long, leakage affects the high output level significantly.

Since the pull down network starts to conduct when the input signal exceeds V_{Tn}, it is reasonable to set V_M, V_{IH}, $V_{IL} = V_{Tn}$.

Therefore, NM_L is very low.

Note that this is a conservative estimate since *subthreshold leakage* occurs for inputs below V_{Tn}.

Also note that the high output level is sensitive to noise and coupling disturbances because of its **high** output impedance.

The high value of NM_H compensates for this increased sensitivity.

Dynamic Logic

Dynamic behavior

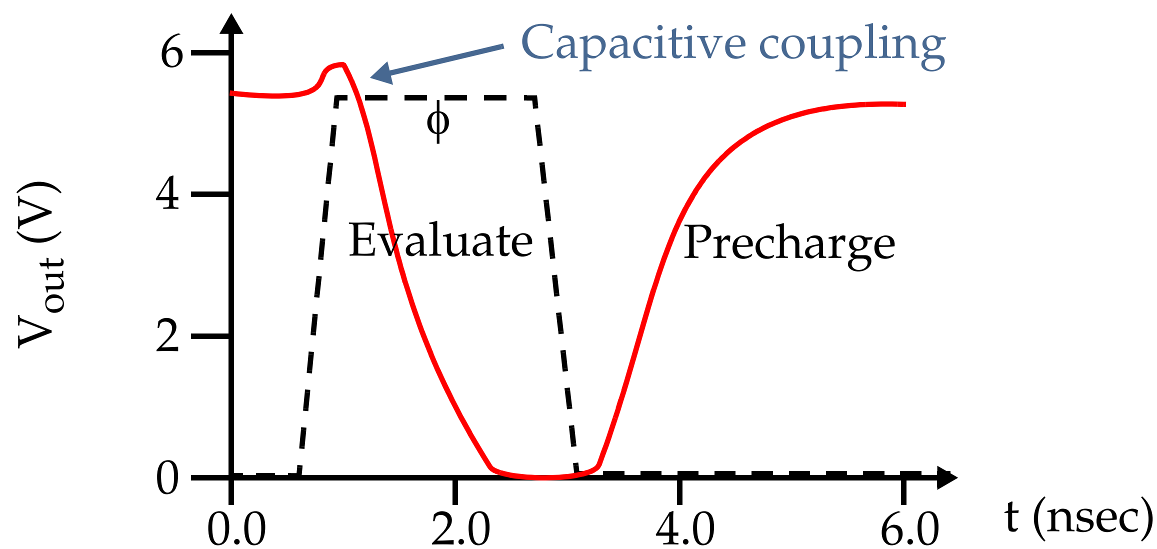
Also, after precharge, the output is high. Therefore, $t_{pLH} = 0$!

This is somewhat unfair since it ignores the precharge time.

The designer is free to choose the size of the PMOS device, smaller is faster but increases load and t_{pHL}.

The t_{pHL} is proportional to C_L and current-sinking capabilities of PDN.

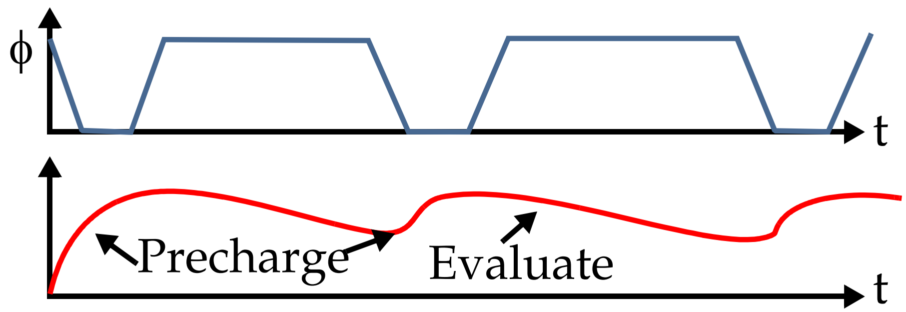
M_e slows down the gate a little.



Dynamic Logic

There are three sources of noise

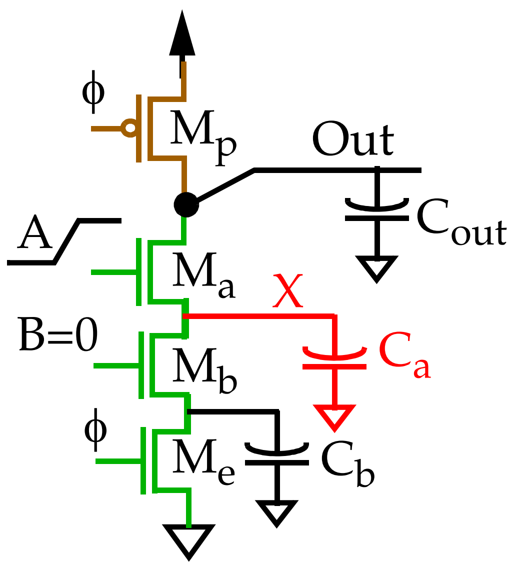
- *Charge Leakage*



Via reversed-biased diffusion diodes and subthreshold leakage

Sets the minimum clock to 250Hz to 1kHz (testing difficulties)

- *Charge Sharing*



If $\Delta V_{out} > V_{Tn}$

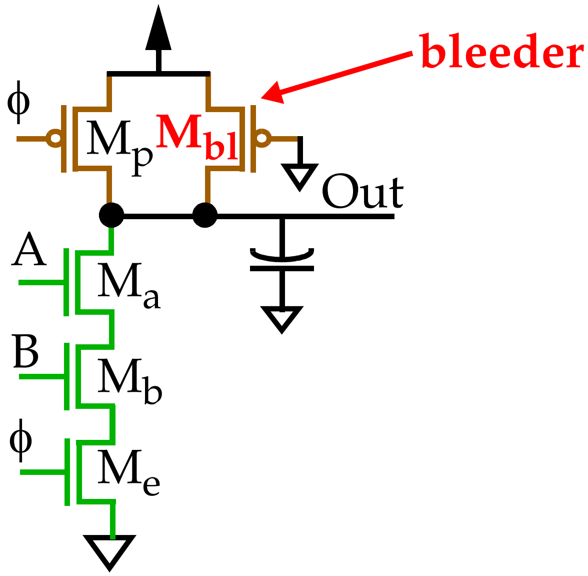
then V_{out} and V_x reach the same value.

$$\Delta V_{out} = -V_{DD}\left(\frac{C_a}{C_a + C_L}\right)$$

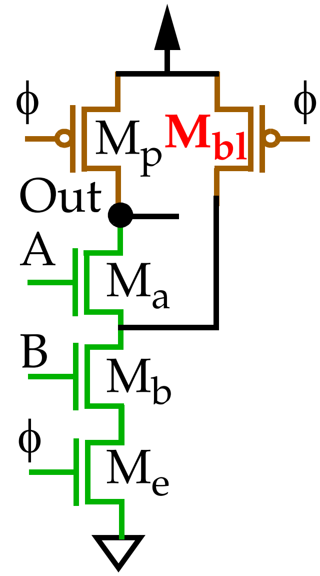
Target is to keep $\Delta V_{out} < |V_{Tp}|$ since output may drive a static gate. $C_a/C_L < 0.2$.

Dynamic Logic

One way to combat both of these:



Static bleeder



Precharge Internal nodes

Pseudo-static: M_{bl} is a highly resistive (long and narrow) PMOS transistor.

Alternatively, *precharge* internal nodes using a clock driven PMOS.

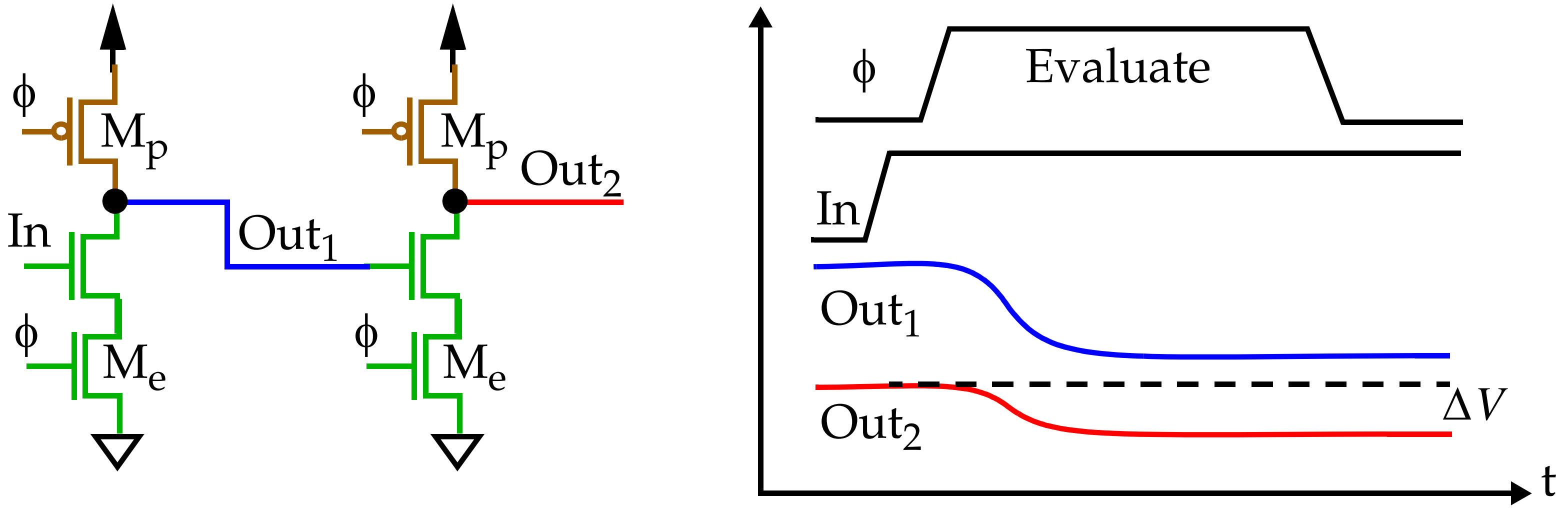
- ***Clock Feedthrough***

  The clock is coupled to the storage node via C_{gs} and gate-overlap caps.

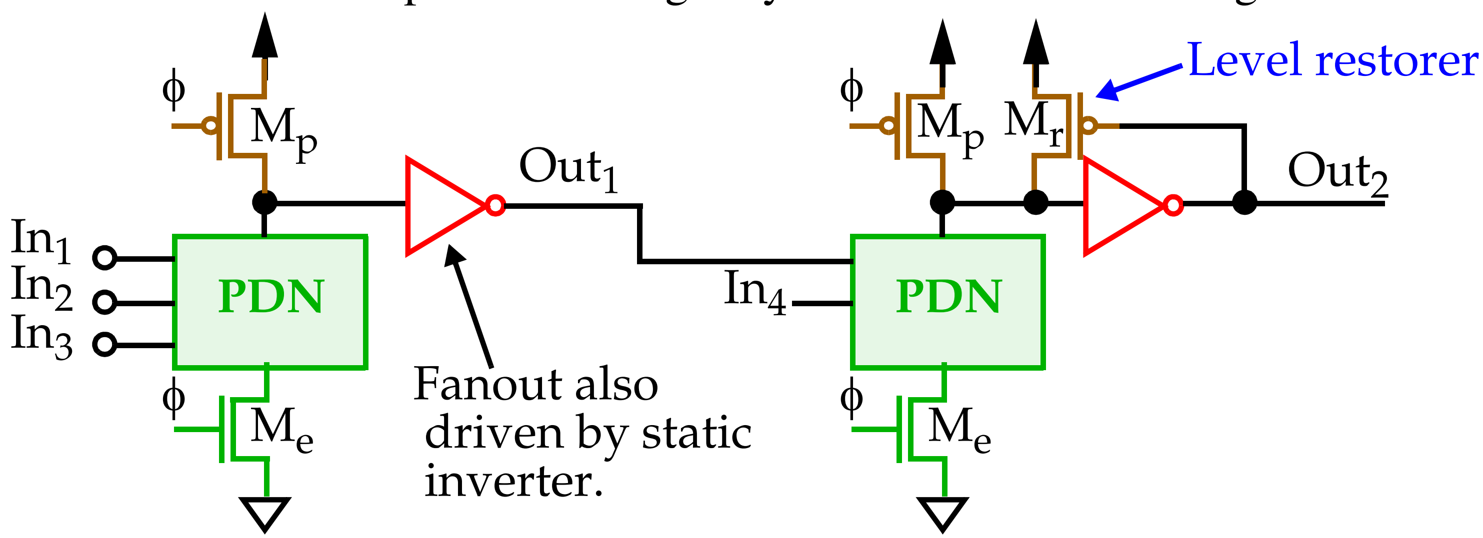
  May forward bias the junction and inject electrons into substrate.

DOMINO Logic

Cascading Dynamic gates



Fix is to restrict the inputs to making only a *0->1* transition during eval.



DOMINO Logic

During evaluation, either the output of the first DOMINO stays at 0 (no delay!) or makes a *0->1* transition.

The transition may ripple all the way down the chain.

Properties:

- Only **non-inverting** logic can be implemented.
- Appropriate for complex, large fan-out circuits such as ALUs or control circuits.
- Very high speeds can be achieved, $t_{pHL} = 0$.

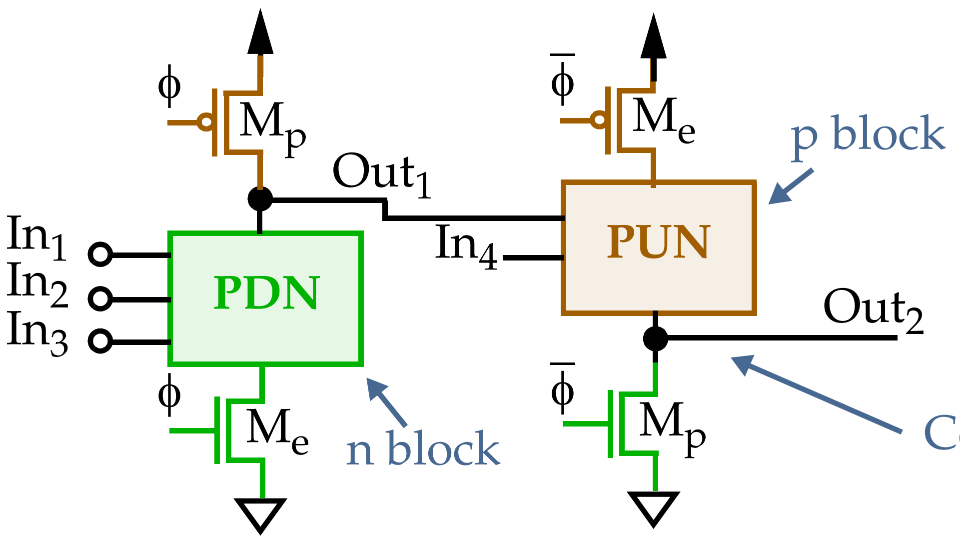
In the past, DOMINO was used in the design of a number of high speed ICs.

The first 32-bit microprocessor (BellMAC 32) used it.

Recently, pure DOMINO circuits are rare, mainly due to the non-inverting logic property.

np-CMOS Logic

PUN networks replace the static inverters.



Also called **ZIPPER** logic

Note that the ϕp blocks are driven with the *Clk_bar* so that the precharge and evaluate periods coincide.

np-CMOS logic style is **20%** faster than DOMINO, despite the slower PMOS pull-up devices.

The DEC alpha-processor (first at 250MHz) used this logic extensively.

Disadv: $NM_L = V_{Tn}$ and $NM_H = |V_{Tp}|$.

Power Consumption

We've already discussed sources of power consumption in CMOS inverter.

$$P_{dyn} = C_L V_{DD}^2 f_{0->1}$$

We now discuss the effects of *switching activity*, *glitching* and *direct-path* current.

Note that the factor $f_{0->1}$ complicates the analysis for complex gates.

Factors affecting the **switching activity** include the *statistics of the input signals*, the *circuit style* (dynamic/static), *the function*, and *network topology*.

These are incorporated by:

$$P_{dyn} = C_L V_{DD}^2 P_{0->1} f$$

where f is the average event rate, and $P_{0->1}$ is the **probability** an input transition results in a *0->1* power-consuming event.

Complex Static Gate Power Consumption

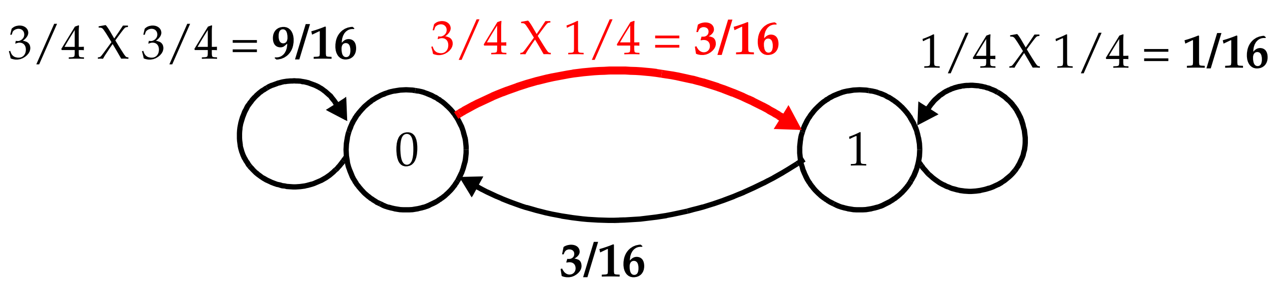
Consider a *2-input* NOR gate, assume the input signals have a uniform distribution of high and low values.

e.g., the 4 input combinations, *AB* = *00*, *01*, *10*, *11*, are equally likely.

Therefore, the probability the output is low or high is **3/4** and **1/4**, respectively.

The probability of an energy consuming transition is the probability that the output is initially low, 3/4, times the probability it will become high, 1/4.

$$P_{0\text{->}1} = P_0 P_1 = (1 - P_1) P_1 = \frac{3}{4} \times \frac{1}{4} = \frac{3}{16}$$



Complex Static Gate Power Consumption

Note that the output probabilities are **no longer uniform**.

This suggests that the input signals are not uniform, since gates are typically cascaded.

The probability that the output is 1 (P_1) is a function of the *input distributions*, P_A and P_B (the probabilities the inputs are 1).

$$P_1 = (1 - P_A)(1 - P_B)$$ for the NOR gate.

The transition probability is then:

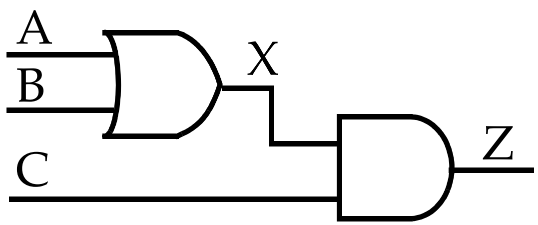
$$P_{0\text{->}1} = (1 - P_1)P_1 = [1 - (1 - P_A)(1 - P_B)][(1 - P_A)(1 - P_B)]$$

3-D graph shown in text.

Derive these expressions for AND, OR and XOR.

Complex Static Gate Power Consumption

For example:



No reconvergent fan-out

With no reconvergent fan-out, the probability that *X* undergoes a power consuming transition is **3/16**.

X = *1*, 3 out of 4 times. Therefore, *X* has an uneven distribution yielding a transition probability on *Z* as:

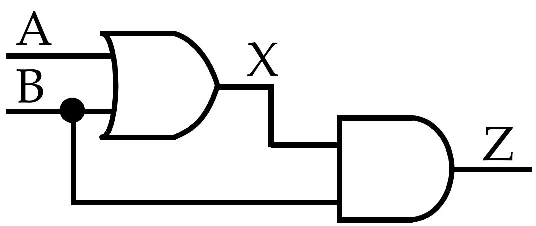
$$Z = (1 - P_X P_C) P_X P_C = \left(1 - \frac{3}{4} \times \frac{1}{2}\right)\left(\frac{3}{4} \times \frac{1}{2}\right) = \frac{15}{16}$$

The orderly calculations from input to output is not possible for

- Circuits with **feedback** (sequential circuits).
- Circuits with **reconvergent fanout**.

Complex Static Gate Power Consumption

In the latter case, the input signals are **not** independent.



Reconvergent fan-out

The procedure above yeilds **15/64** for the transition probability.

However, reduction yields $Z = B$, and the $P_{0\text{->}1}$ transition probability on Z is (1/2 X 1/2) = 1/4.

Conditional probabilities take signal inter-dependencies into account.

For example, $Z = 1$ iff B and $X = 1$.

$$P_Z = P(Z=1) = P(B=1, X=1)$$

This expresses the probability that B and X are 1 simultaneously.

If a dependency exists, a *conditional probability* is required for expansion:

$$P_Z = P(X=1|B=1) \cdot P(B=1|X=1) = P(X=1|B=1) \cdot P(B=1)$$

Dynamic Gate Power Consumption

What about **dynamic circuits**?

During precharge, the output node is charged to 1.

Therefore, power is consumed every time the PDN is on (output is 0), independent of the preceding or following values!

Power consumption is determined solely by signal value probabilities, and **not** by transition probabilities.

These is always *larger* than the transition probability, since the latter is the product of two signal probabilities both of which is smaller than 1.

For example, the *0-probability* of a *2-input* NOR is

$$P_0 = (P_A + P_B - P_A P_B)$$

If the inputs are equally probably, there is a 75% chance of a *1->0*.

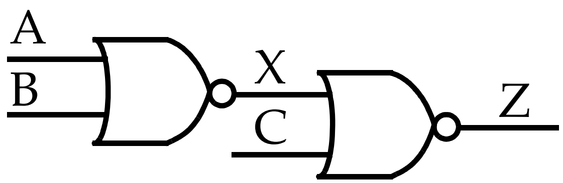
$$P_{NOR} = 0.75 C_L V_{DD}^2 f_{clk}$$

Note C_L is smaller than a static gate but the clock load must be considered.

Glitches in Static CMOS Circuits

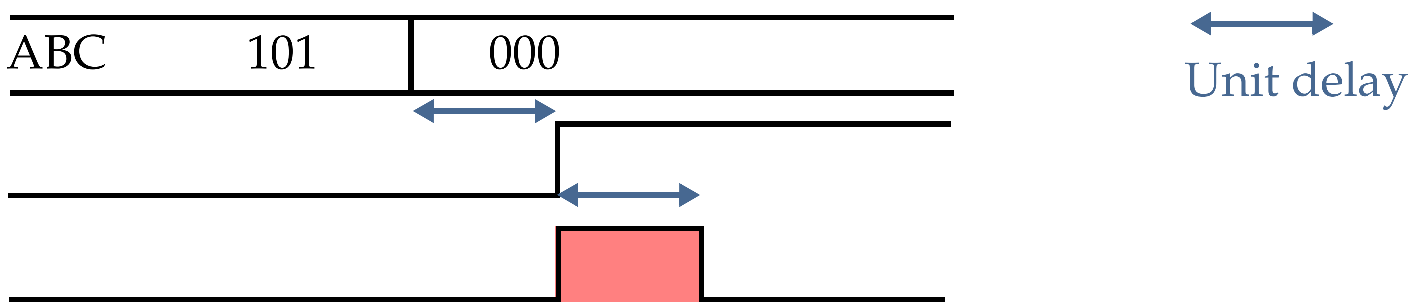
The finite propagation delay through gates in a network can cause spurious transitions called **glitches**, **critical races** or **dynamic hazards**.

These are multiple transitions during a single clock cycle.



Assume a unit delay and all inputs arrive at the same time.

The second NOR evaluates **twice**, the first one with the previous value of *X*. This consumes unnecessary power.



Redesign can eliminate glitches by matching delays along signal paths.

Summary

Choosing a logic style depends on Ease of design, Robustness, System clocking requirements, Fan-out, Functionality and Testing.

Static is robust and easy to design (ameanable to design automation).

- Complementary complex gates are expensive in area and performance.
- Pseudo-NMOS is simple and fast but reduces noise margins and increases power consumption.
- Pass-transistor logic is good for certain classes of circuits (MUX/adders).

Dynamic logic gives fast and small circuits but complicates the design process and restricts the minimum clock rate.

For a *4-input* NAND gate:

Style	*Ratioed*	*Static power*	*# of trans.*	*Area (um^2)*	*delay (ns)*
Complementary	No	No	8	533	0.61
Pseudo-NMOS	Yes	Yes	5	288	1.49
CPL	No	No	14	800	0.75
Dynamic (np)	No	No	6	212	0.37