CAL POLY
SAN LUIS OBISPO

EE 431: COMPUTER-AIDED DESIGN OF VLSI DEVICES

Logic Circuit Families

Nishith N. Chakraborty

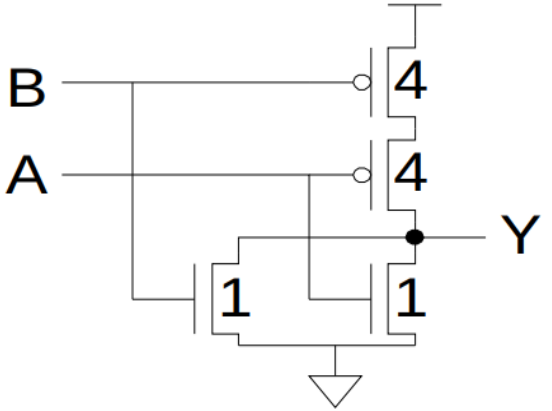
October, 2024

LOGIC FAMILIES OUTLINE

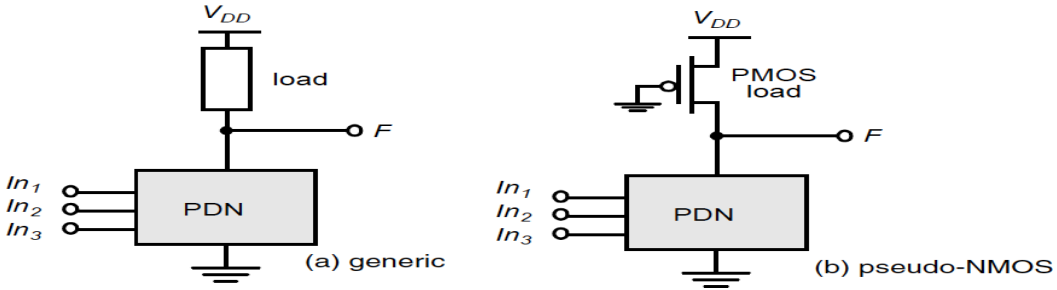
- Ratioed Logic (Pseudo-nMOS Logic)
- Dynamic Logic
- Pass Transistor Logic

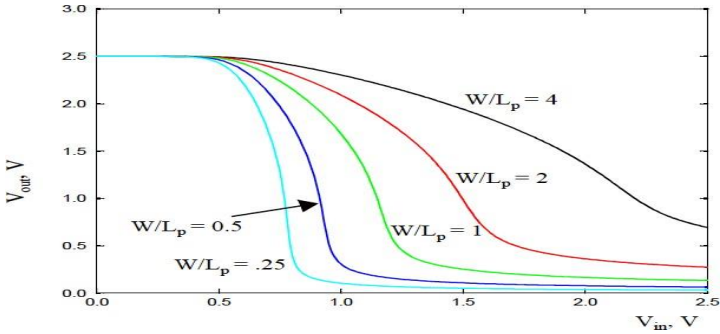
INTRODUCTION

- What makes a circuit fast?
  - $I = C\,dV/dt \rightarrow t_{pd} \propto (C/I)\,\Delta V$
  - low capacitance
  - high current
  - small swing
- Logical effort is proportional to C/I
- pMOS are the enemy!
  - High capacitance for a given current
- Can we take the pMOS capacitance off the input?
- Various circuit families try to do this…



RATIOED LOGIC



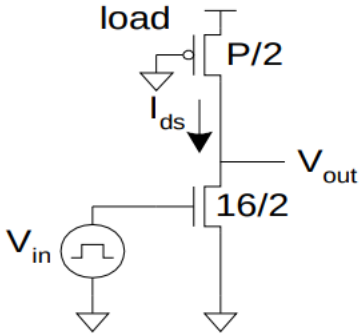


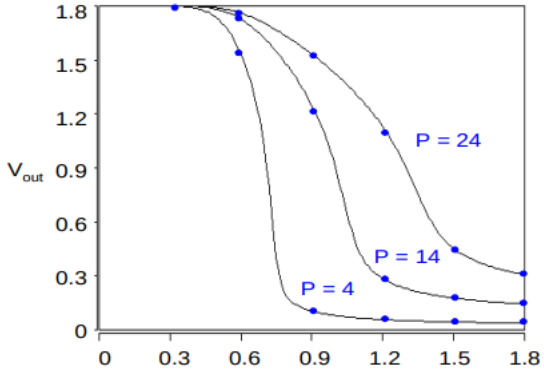
- Ratioed logic is an attempt to reduce the number of transistors required to implement a given logic function, at the cost of reduced robustness and extra power dissipation.
- In ratioed logic, the entire PUN is replaced with a single unconditional load device that pulls up the output.
- The voltage swing on the output and the functionality of the gate depends upon the ratio between the NMOS and PMOS sizes.

- **In *Ratio-less* logic such as complementary CMOS, low and high output levels do not depend on the transistors sizes.**

PSEUDO-NMOS

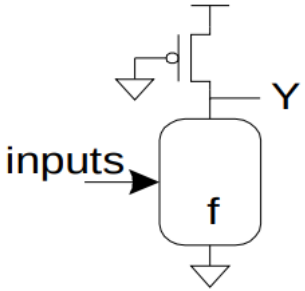
- In the old days, nMOS processes had no pMOS
  - Instead, use pull-up transistor that is always ON
- In CMOS, a pMOS can be used that is always ON
  - Ratio issue – must be careful about sizing
  - Rule of thumb: Make pMOS about ¼ effective strength of pulldown network



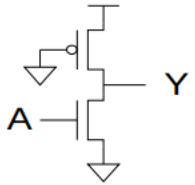


PSEUDO-NMOS GATES

- Design for unit current on output to compare with unit inverter.
- pMOS fights nMOS

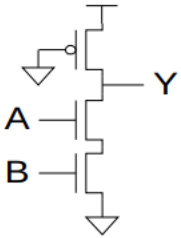


Inverter



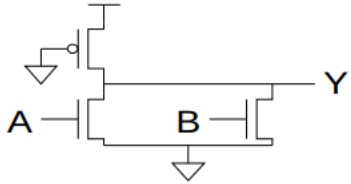
g_u =
g_d =
g_{avg} =
p_u =
p_d =
p_{avg} =

NAND2



g_u =
g_d =
g_{avg} =
p_u =
p_d =
p_{avg} =

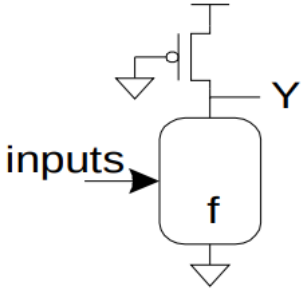
NOR2



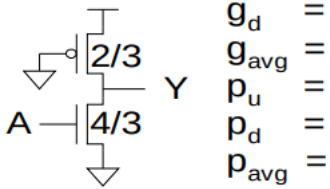
g_u =
g_d =
g_{avg} =
p_u =
p_d =
p_{avg} =

PSEUDO-NMOS GATES

- To calculate the logical effort of pseudo-nMOS gates, suppose a complementary CMOS unit inverter delivers current I in both rising and falling transitions.
- For the widths shown, the pMOS transistors produce I/3 and the nMOS networks produce 4I/3.
- The logical effort for each transition is computed as the ratio of the input capacitance to that of a complementary CMOS inverter with equal current for that transition.

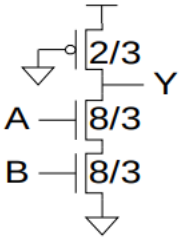


Inverter



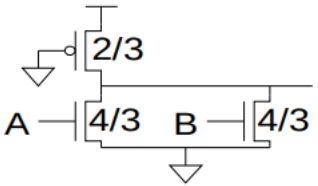
g_u =
g_d =
g_{avg} =
p_u =
p_d =
p_{avg} =

NAND2



g_u =
g_d =
g_{avg} =
p_u =
p_d =
p_{avg} =

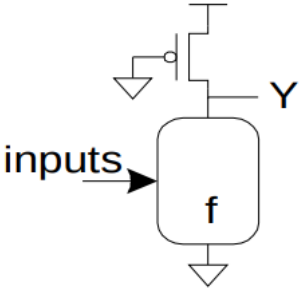
NOR2



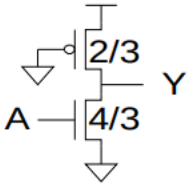
g_u =
g_d =
g_{avg} =
p_u =
p_d =
p_{avg} =

PSEUDO-NMOS GATES

- For the falling transition, the pMOS transistor effectively fights the nMOS pulldown. The output current is estimated as the pulldown current minus the pullup current, (4I/3 – I/3) = I.
- Therefore, we will compare each gate to a unit inverter to calculate g_d.
- For example, the logical effort for a falling transition of the pseudo-nMOS inverter is the ratio of its input capacitance (4/3) to that of a unit complementary CMOS inverter (3), i.e., 4/9. g_u is three times as great because the current is 1/3 as much.

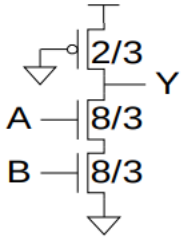


Inverter



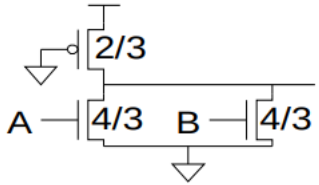
g_u = 4/3
g_d = 4/9
g_{avg} = 8/9
p_u =
p_d =
p_{avg} =

NAND2



g_u = 8/3
g_d = 8/9
g_{avg} = 16/9
p_u =
p_d =
p_{avg} =

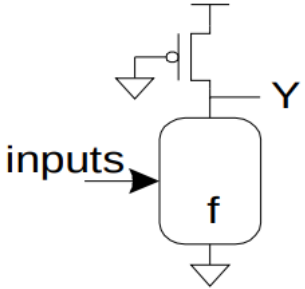
NOR2



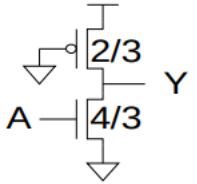
g_u = 4/3
g_d = 4/9
g_{avg} = 8/9
p_u =
p_d =
p_{avg} =

PSEUDO-NMOS GATES

- The parasitic delay is also found by counting output capacitance and comparing it to an inverter with equal current. For example, the pseudo-nMOS NOR has 10/3 units of diffusion capacitance as compared to 3 for a unit-sized complementary CMOS inverter, so its parasitic delay pulling down is 10/9. The pullup current is 1/3 as great, so the parasitic delay pulling up is 10/3.

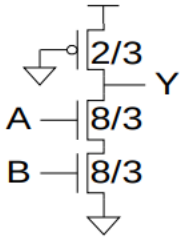


Inverter



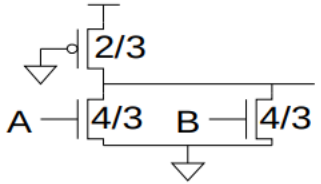
$g_u = 4/3$
$g_d = 4/9$
$g_{avg} = 8/9$
$p_u = 6/3$
$p_d = 6/9$
$p_{avg} = 12/9$

NAND2



$g_u = 8/3$
$g_d = 8/9$
$g_{avg} = 16/9$
$p_u = 10/3$
$p_d = 10/9$
$p_{avg} = 20/9$

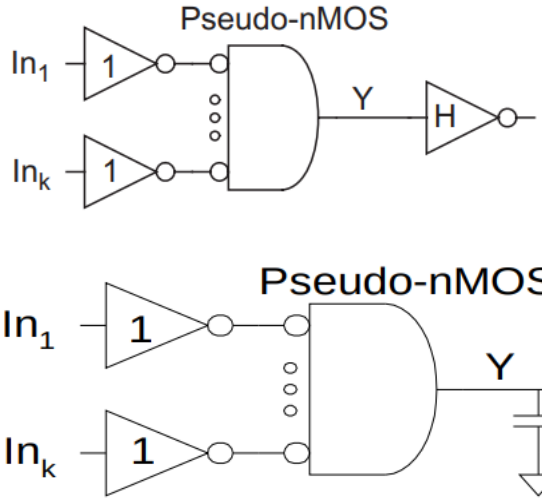
NOR2



$g_u = 4/3$
$g_d = 4/9$
$g_{avg} = 8/9$
$p_u = 10/3$
$p_d = 10/9$
$p_{avg} = 20/9$

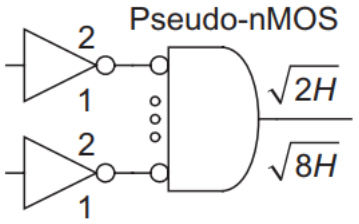
PSEUDO-NMOS DESIGN

- Ex: Design a k-input AND gate using pseudo-nMOS. Use DeMorgan's law using static CMOS inverters followed by a k-input pseudo-nMOS NOR. Estimate the delay driving a fanout of H (an inverter of size H times unit inverter)
- G =
- F =
- P =
- N =
- D =



PSEUDO-NMOS DESIGN

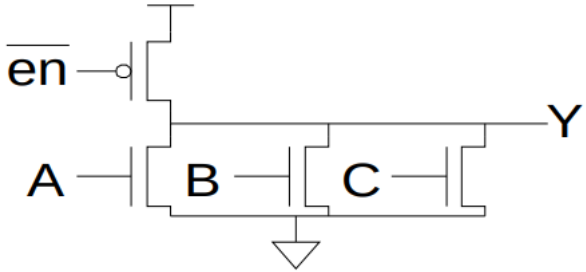
- Ex: Design a k-input AND gate using pseudo-nMOS. Estimate the delay driving a fanout of H
- G = 1 * 8/9 = 8/9
- F = GBH = 8H/9
- N = 2
- $\hat{f} = F^{1/N} = \frac{\sqrt{8H}}{3}$ → $C_{in} = \frac{gC_{out}}{\hat{f}} = \frac{(8/9)H}{\sqrt{8H/9}} = \frac{\sqrt{8H}}{3}$ →



- P = 1 + (4+8k)/9 = (8k+13)/9
- $D = NF^{1/N} + P = \frac{4\sqrt{2H}}{3} + \frac{8k+13}{9}$

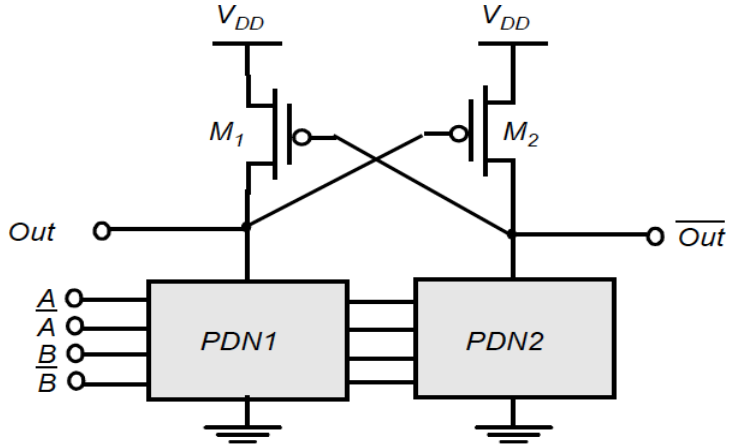
PSEUDO-NMOS POWER

- Pseudo-nMOS draws power whenever Y = 0
  - Called static power P = I•VDD
  - A few mA / gate * 1M gates would be a problem
  - This is why only nMOS designs went extinct!
- Use pseudo-nMOS sparingly for wide NORs
- Turn off pMOS when not in use



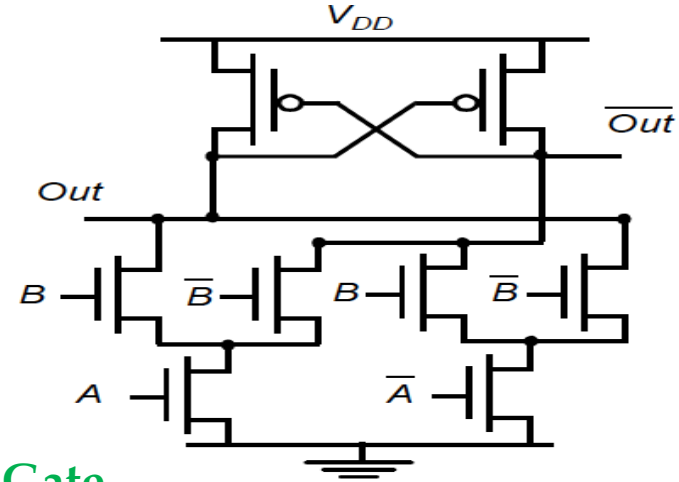
DCVSL

- It is possible to create a ratioed logic style that eliminates static currents and provides rail-to-rail swing
- An example of such a logic family is called Differential Cascode Voltage Switch Logic (or DCVSL)
- This circuit exhibits a rail-to-rail swing, and the static power dissipation is eliminated



(a) Basic principle

DCVSL Logic Gate

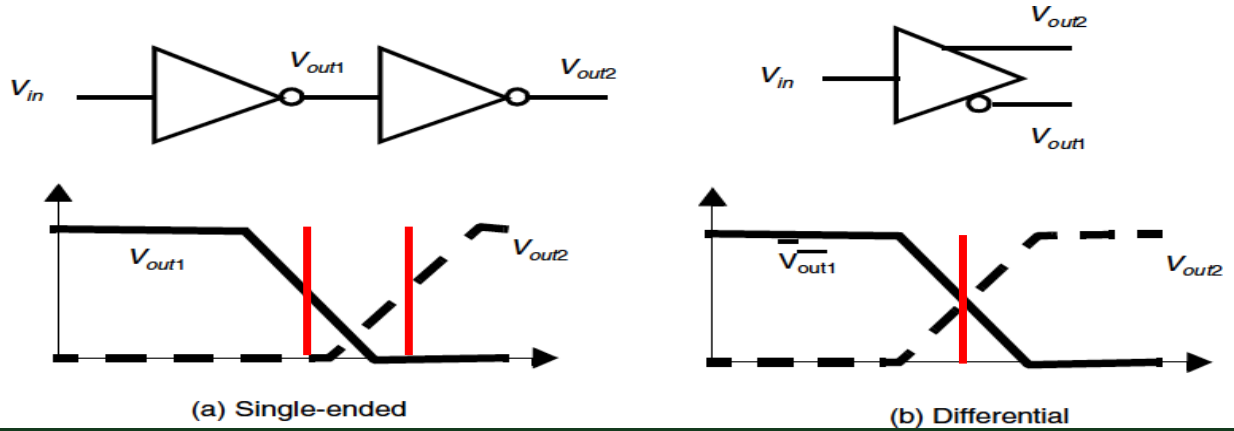


(b) XOR-XNOR gate

DCVSL: PROS & CONS

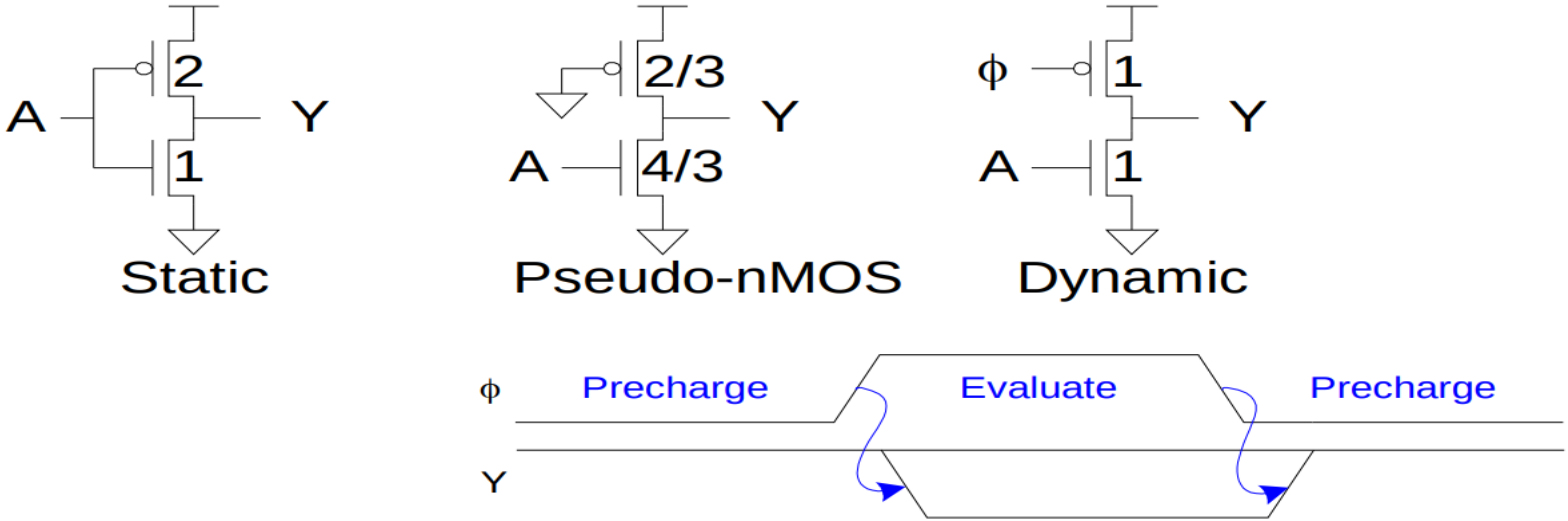
- The DCVSL gate provides differential (or complementary) outputs. Both the output signal and its inverted value are simultaneously available. This is a distinct advantage, as it eliminates the need for an extra inverter to produce the complementary signal.
- In logic design it often happens that both a signal and its complement are needed simultaneously. When the complementary signal is generated using an inverter, the inverted signal is delayed with respect to the original. This causes timing problems, especially in very high-speed designs. The differential output capability avoids this problem.

- **The differential nature virtually doubles the number of wires that has to be routed**
- **The dynamic power dissipation is high**



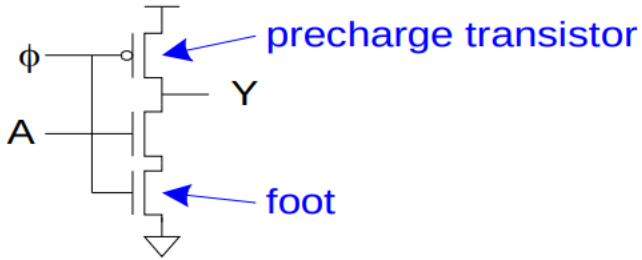
DYNAMIC LOGIC

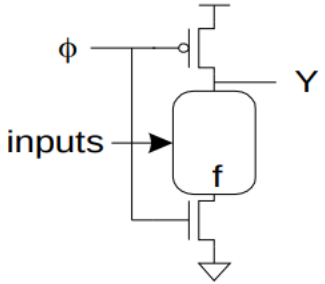
- *Dynamic* gates uses a clocked pMOS pullup
- Two modes: *precharge* and *evaluate*



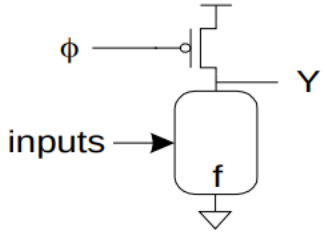
THE FOOT

- What if pulldown network is ON during precharge?
- Use series evaluation transistor to prevent fight.



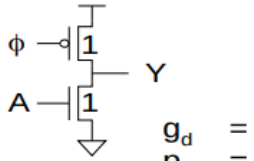
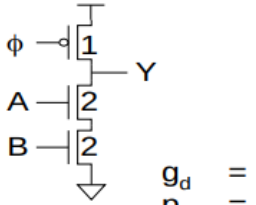
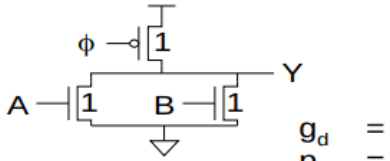
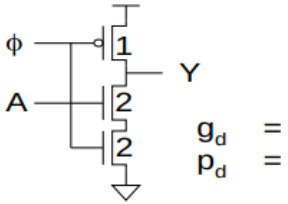
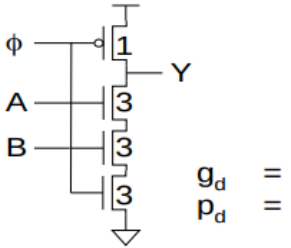
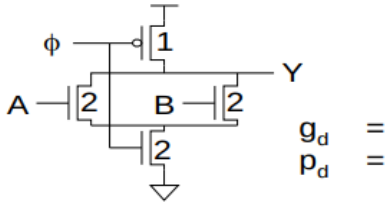


footed

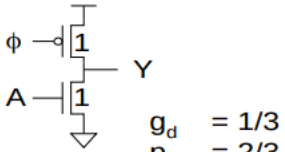
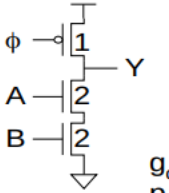
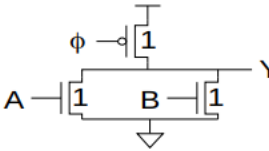
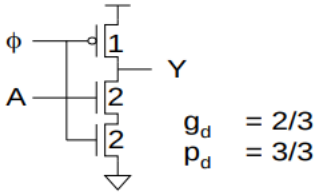
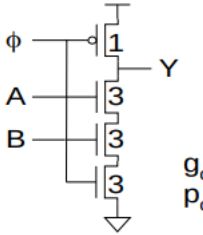
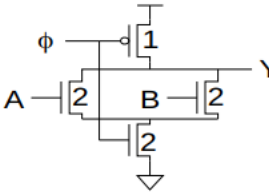


unfooted

LOGICAL EFFORT

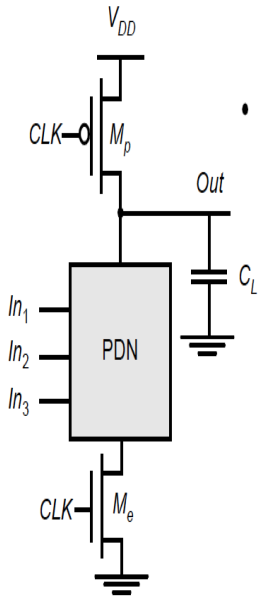
	Inverter	NAND2	NOR2
unfooted	 g_d = p_d =	 g_d = p_d =	 g_d = p_d =
footed	 g_d = p_d =	 g_d = p_d =	 g_d = p_d =

LOGICAL EFFORT

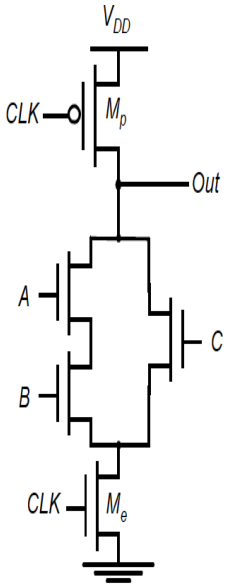
	Inverter	NAND2	NOR2
unfooted	 $g_d = 1/3$ $p_d = 2/3$	 $g_d = 2/3$ $p_d = 3/3$	 $g_d = 1/3$ $p_d = 3/3$
footed	 $g_d = 2/3$ $p_d = 3/3$	 $g_d = 3/3$ $p_d = 4/3$	 $g_d = 2/3$ $p_d = 5/3$

DYNAMIC LOGIC PROPERTIES

- It only consumes dynamic power. No short circuit power.
- Non-ratioed.
- Faster switching speed due to having less capacitance.
- The logic function is implemented by the NMOS pull-down network. The construction of the PDN proceeds just as it does for static CMOS.
- Transistor count is reduced substantially (compared with CMOS).



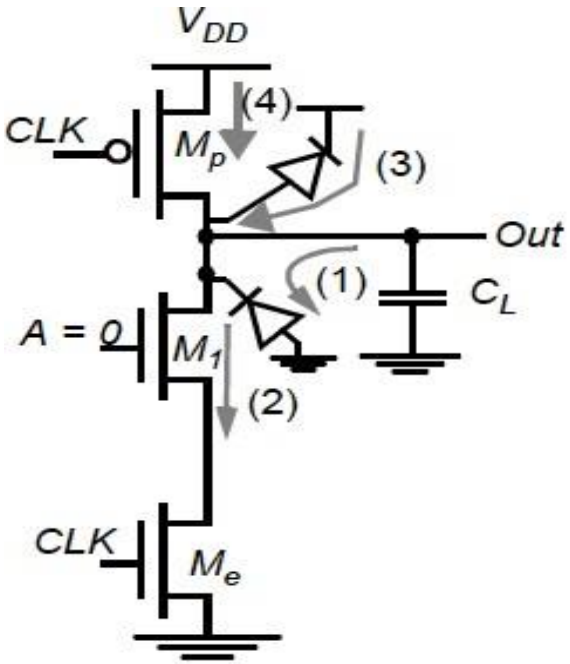
(a) n-type network



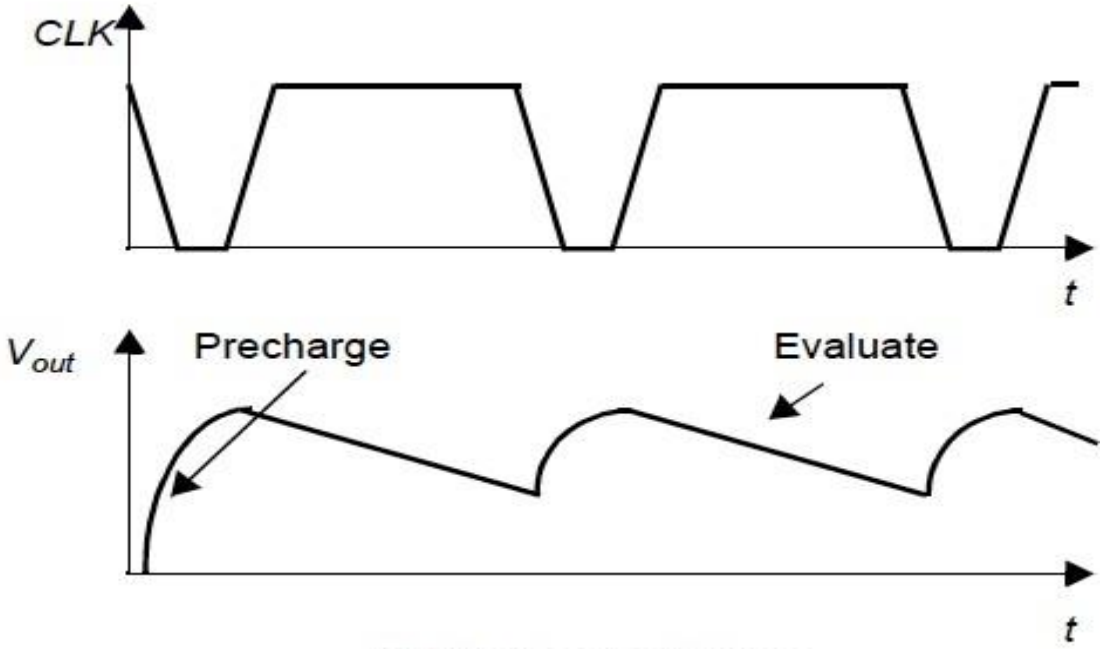
(b) Example

$$Out = \overline{CLK} + \overline{(A \cdot B + C)} \cdot CLK$$

CHARGE LEAKAGE IN DYNAMIC GATES



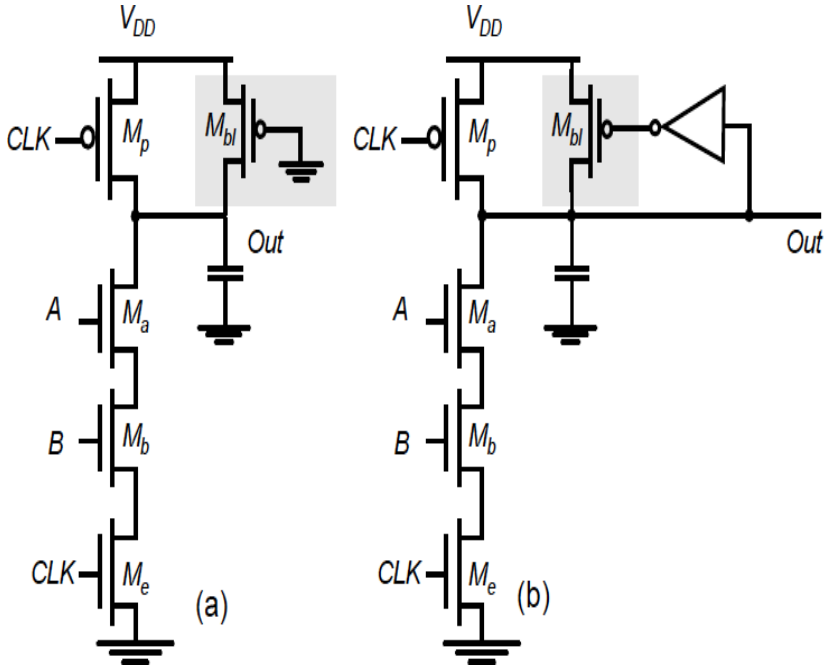
(a) Leakage sources



(b) Effect on waveforms

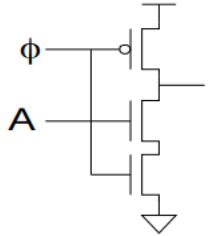
COMPENSATION FOR THE CHARGE-LEAKAGE

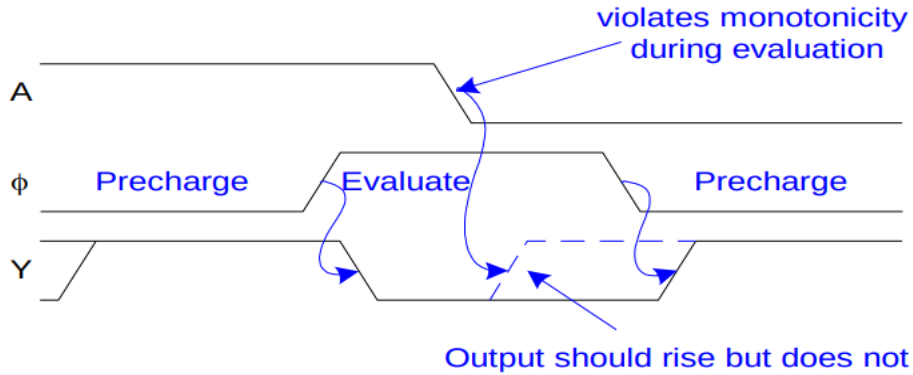
- Leakage is caused by the high impedance state of the output node during the evaluate mode, when the pull down path is turned off.
- The leakage problem can be counteracted by reducing the output impedance on the output node during evaluation. This is often done by adding a bleeder transistor.
- To avoid the ratio problems associated with this style of circuit and the associated static power consumption, the bleeder resistance is made high, or, in other words, the device is kept small.



MONOTONICITY

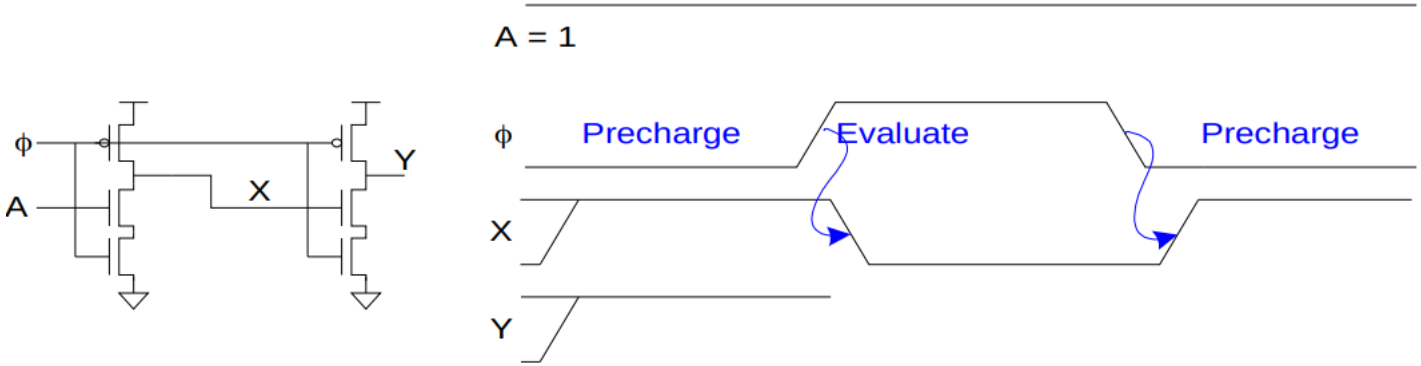
- Dynamic gates require monotonically rising inputs during evaluation
  - 0 -> 0
  - 0 -> 1
  - 1 -> 1
  - But not 1 -> 0





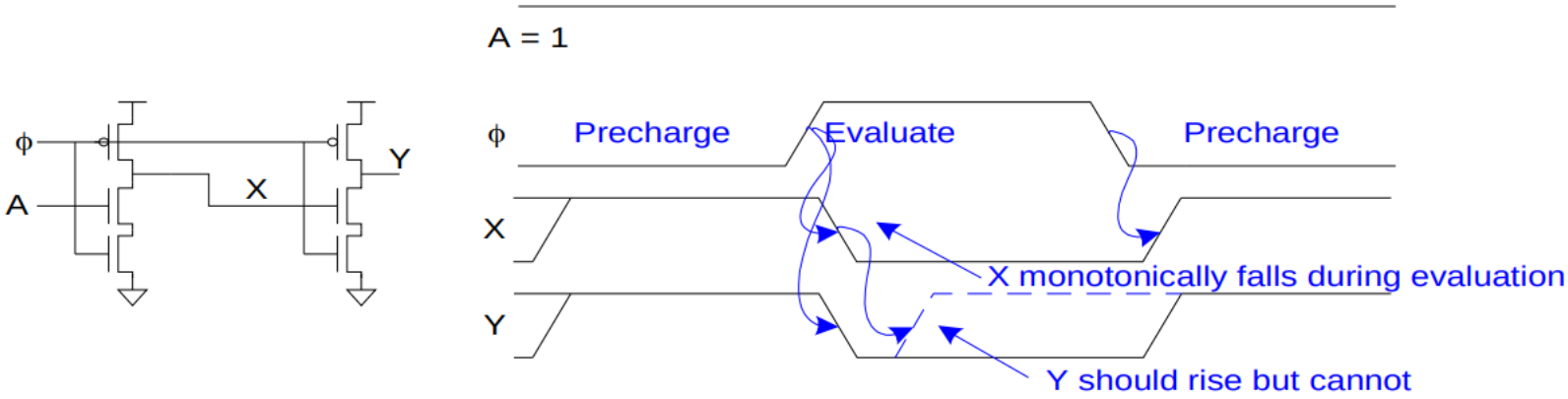
MONOTONICITY WOES

- But dynamic gates produce monotonically falling outputs during evaluation
- Illegal for one dynamic gate to drive another!



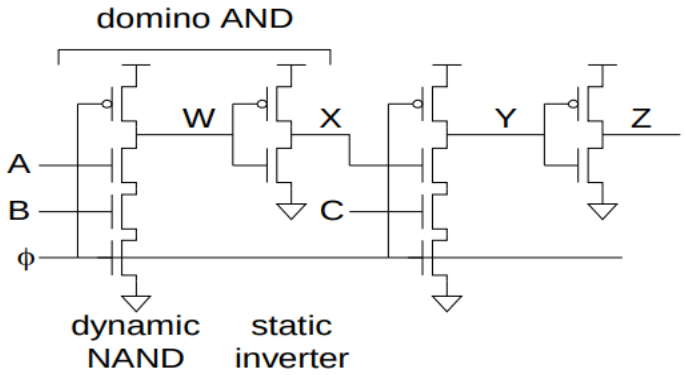
MONOTONICITY WOES

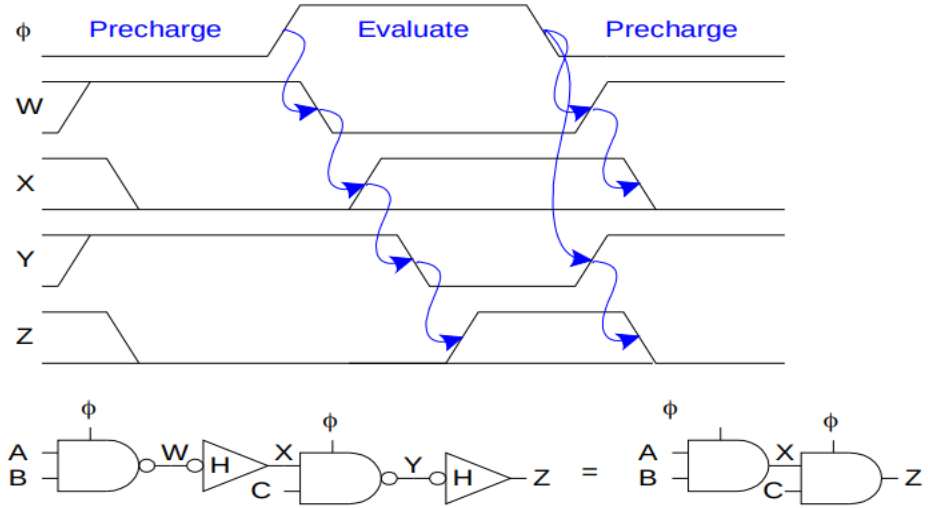
- But dynamic gates produce monotonically falling outputs during evaluation
- Illegal for one dynamic gate to drive another!



DOMINO GATES

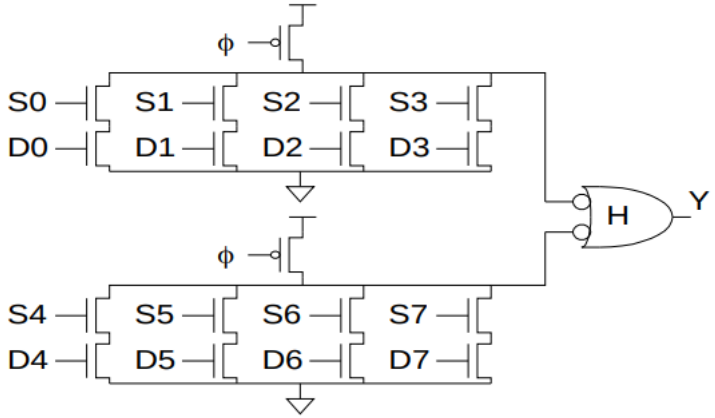
- Follow dynamic stage with inverting static gate
  - Dynamic / static pair is called domino gate
  - Produces monotonic outputs





DOMINO OPTIMIZATIONS

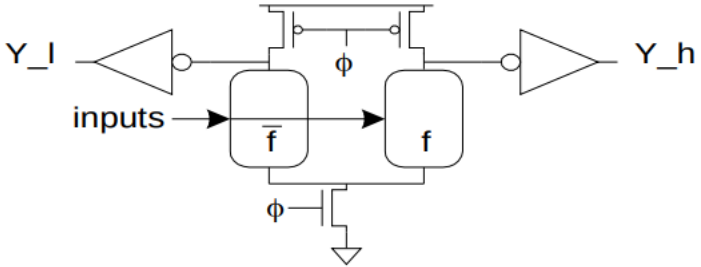
- Each domino gate triggers next one, like a string of dominos toppling over
- Gates evaluate sequentially but precharge in parallel
- Thus evaluation is more critical than precharge
- HI-skewed static stages can perform logic



DUAL-RAIL DOMINO

- Domino only performs noninverting functions:
  - AND, OR but not NAND, NOR, or XOR
- Dual-rail domino solves this problem
  - Takes true and complementary inputs
  - Produces true and complementary outputs

sig_h	sig_l	Meaning
0	0	Precharged
0	1	'0'
1	0	'1'
1	1	invalid

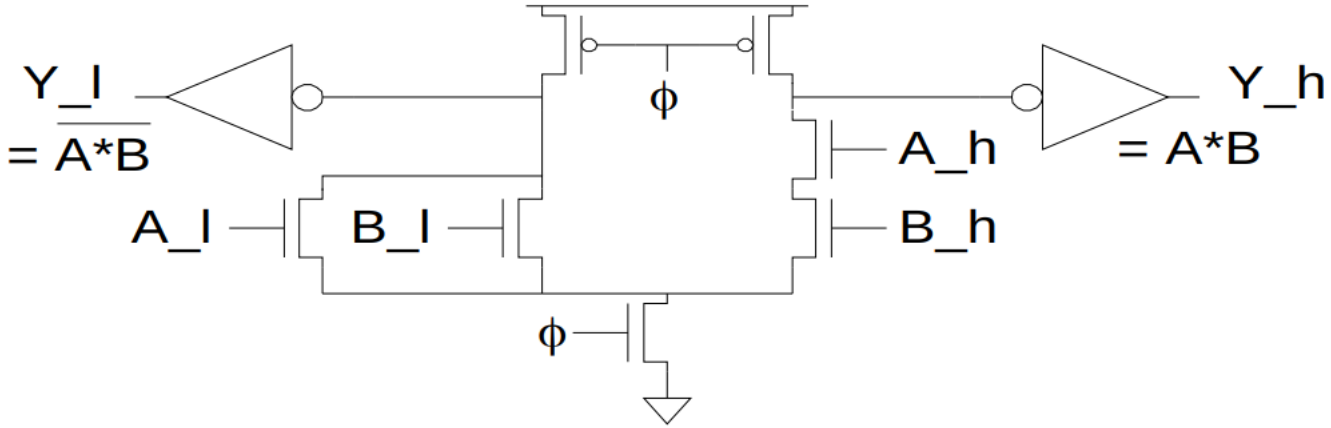


EXAMPLE: AND/NAND

- Given A_h, A_l, B_h, B_l
- Compute Y_h = A * B, Y_l = ~(A * B)

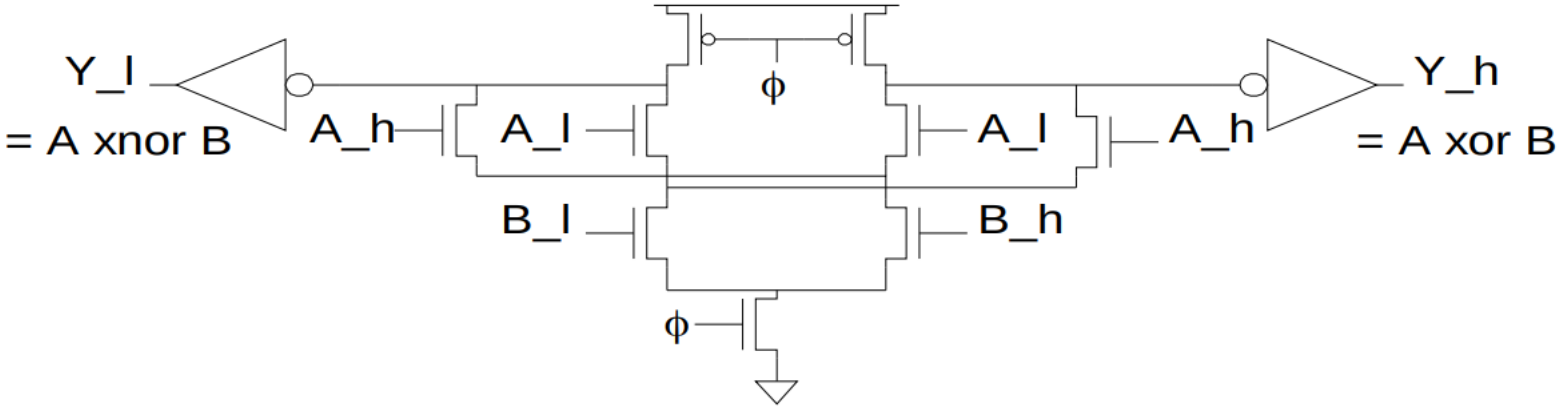
EXAMPLE: AND/NAND

- Given A_h, A_l, B_h, B_l
- Compute Y_h = A * B, Y_l = ~(A * B)
- Pulldown networks are conduction complements



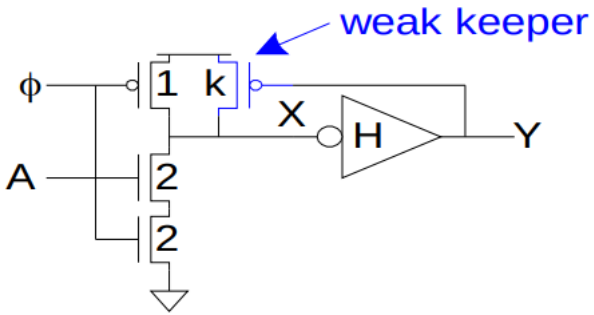
EXAMPLE: XOR/XNOR

- Sometimes possible to share transistors



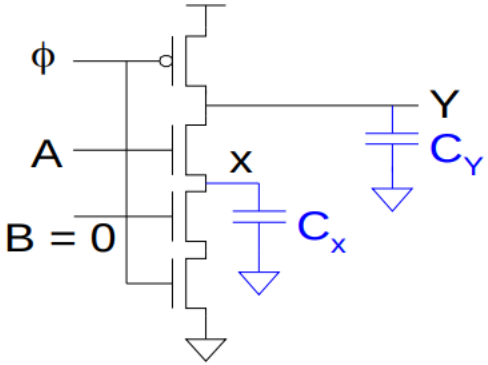
LEAKAGE

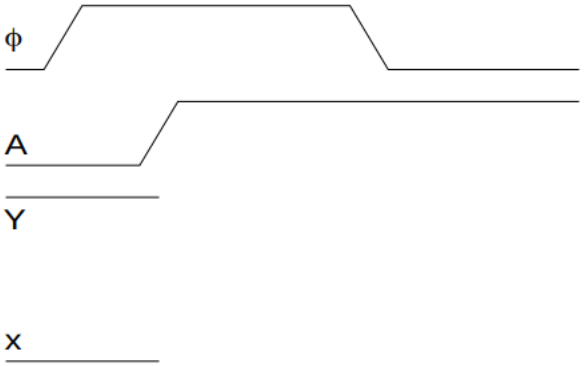
- Dynamic node floats high during evaluation
  - Transistors are leaky ($I_{OFF} \neq 0$)
  - Dynamic value will leak away over time
  - Used to be milliseconds, now nanoseconds!
- Use keeper/bleeder to hold dynamic node
  - Must be weak enough not to fight evaluation



CHARGE SHARING

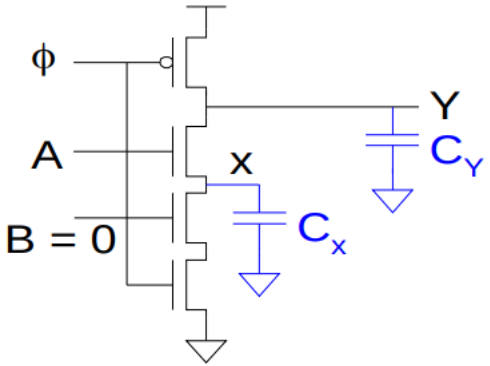
- Dynamic gates suffer from charge sharing

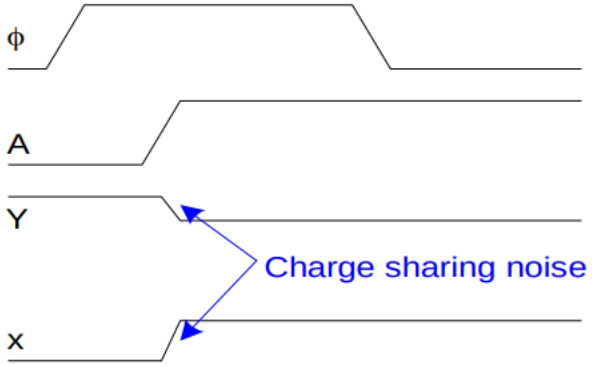




CHARGE SHARING

- Dynamic gates suffer from charge sharing

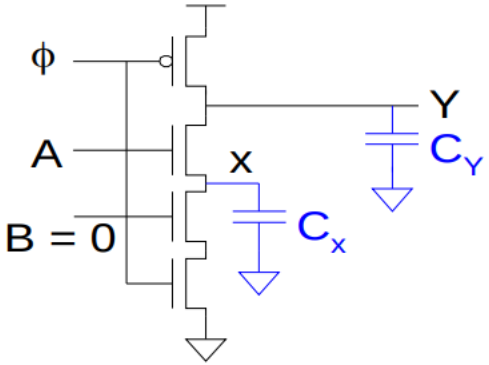


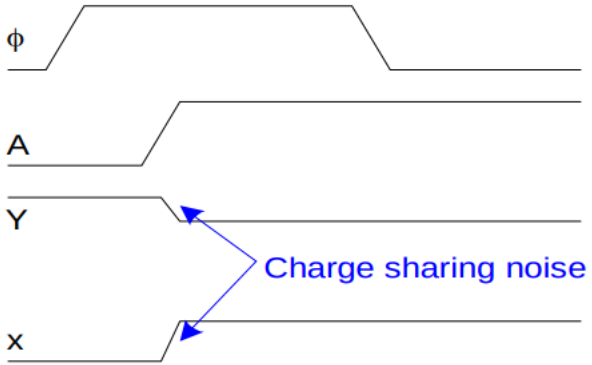


$$V_x = V_Y =$$

CHARGE SHARING

- Dynamic gates suffer from charge sharing

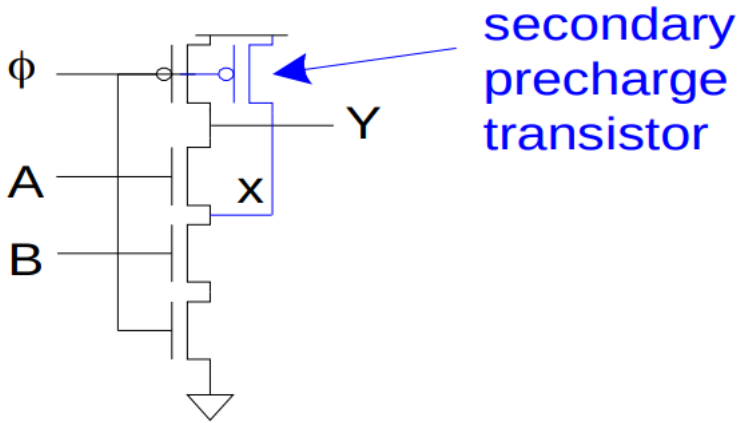




$$V_x = V_Y = \frac{C_Y}{C_x + C_Y} V_{DD}$$

SECONDARY PRECHARGE

- Solution: add secondary precharge transistors
  - Typically need to precharge every other node
- Big load capacitance C_Y helps as well



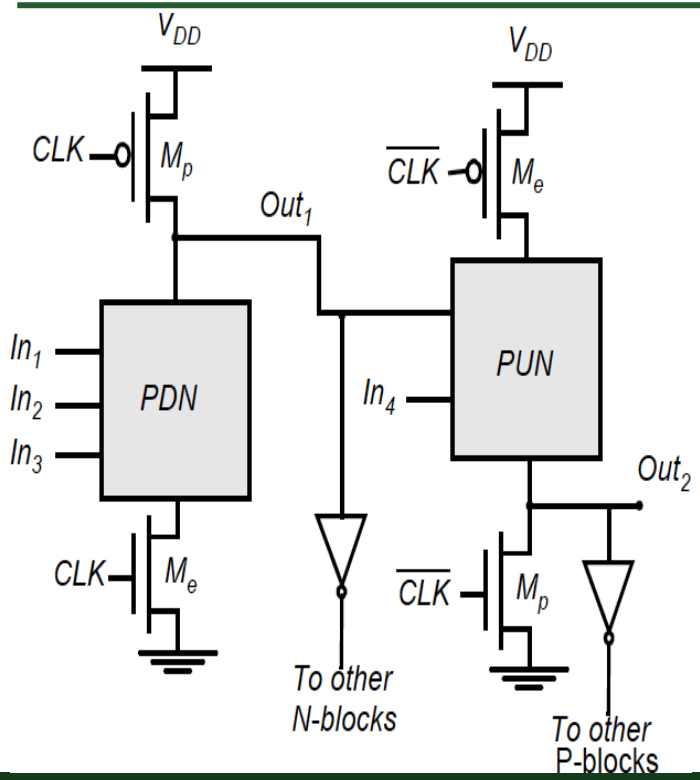
NOISE SENSITIVITY

- Dynamic gates are very sensitive to noise
  - Inputs: When $V_{IH} \approx V_{tn}$, keeper can pull output high
  - Outputs: floating output susceptible to noise
- Noise sources
  - Capacitive crosstalk
  - Charge sharing
  - Power supply noise
  - Feedthrough noise
  - And more!

DOMINO SUMMARY

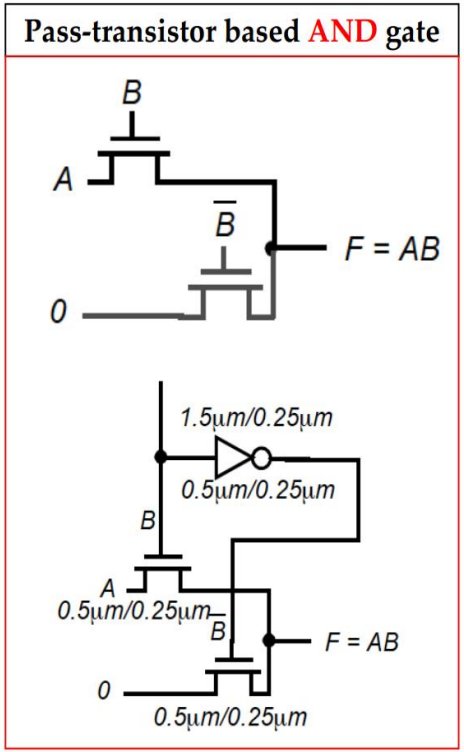
- Domino logic is attractive for high-speed circuits
  - 1.5 – 2x faster than static CMOS
  - But many challenges:
    - Monotonicity
    - Leakage
    - Charge sharing
    - Noise
- Widely used in high-performance microprocessors

NP- CMOS

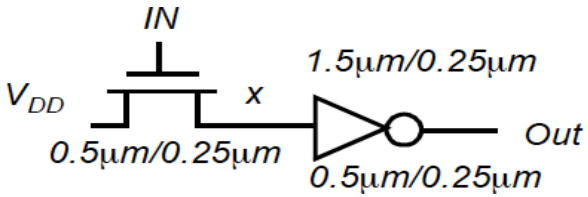


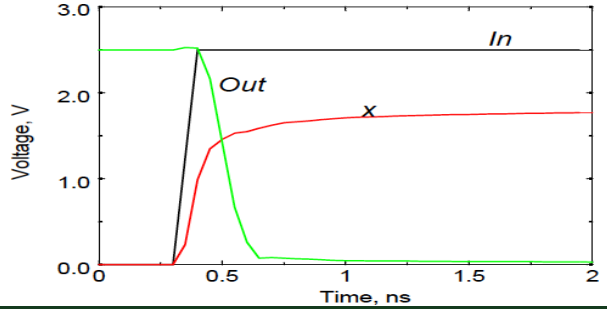
- The Domino logic presented in the previous section has the disadvantage that each dynamic gate requires an extra static inverter in the critical path to make the circuit functional. np-CMOS, provides an alternate approach to cascading dynamic logic by using two flavors (n-tree and p- tree) of dynamic logic.
- In a p-tree logic gate, PMOS devices are used to build a pull-up logic network, including a PMOS evaluation transistor.
- The NMOS pre-discharge transistor drives the output low during pre-charge. The output conditionally makes a 0 → 1 transition during evaluation depending on its inputs.

PASS TRANSISTOR CIRCUITS

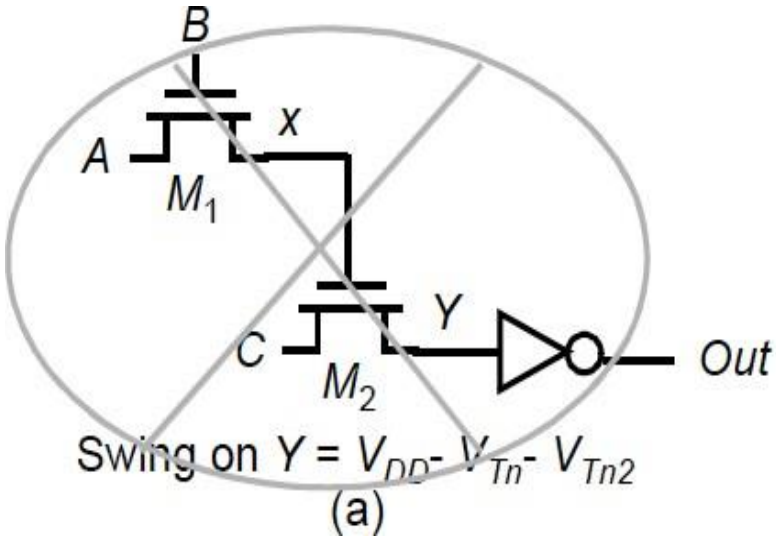


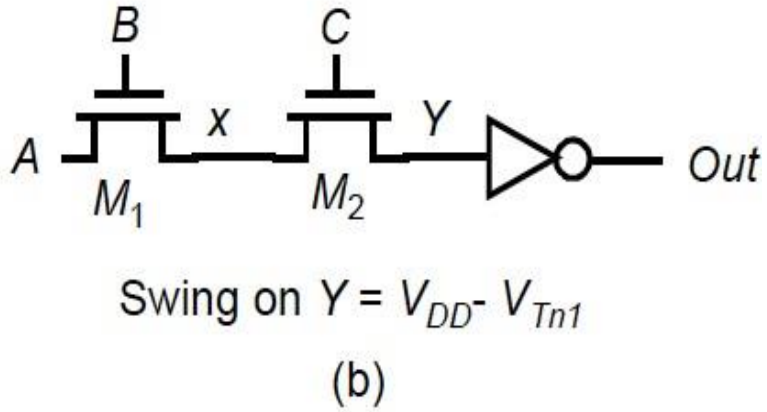
- Allows the primary inputs to drive gate terminals as well as source/drain terminals
- Fewer transistors are required to implement a given function
- Unfortunately, an NMOS is effective at passing a 0 but is poor at pulling a node to V_{DD}. When the pass transistor pulls a node high, the output only charges up to V_{DD}-V_{Tn}. The situation is worsened by the fact that the devices experience body effect, as there exists a significant source-to-body voltage when pulling high.





CASCADING PASS TRANSISTOR LOGIC

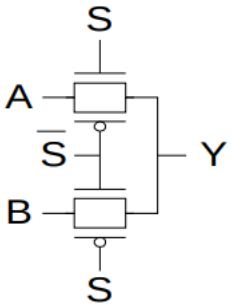


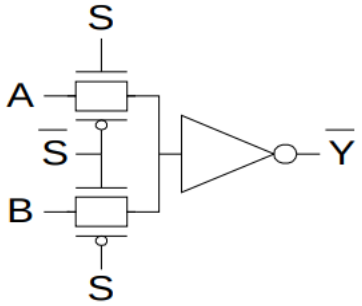


Pass transistor output (Drain/Source) terminal should not drive other gate terminals to avoid multiple threshold drops.

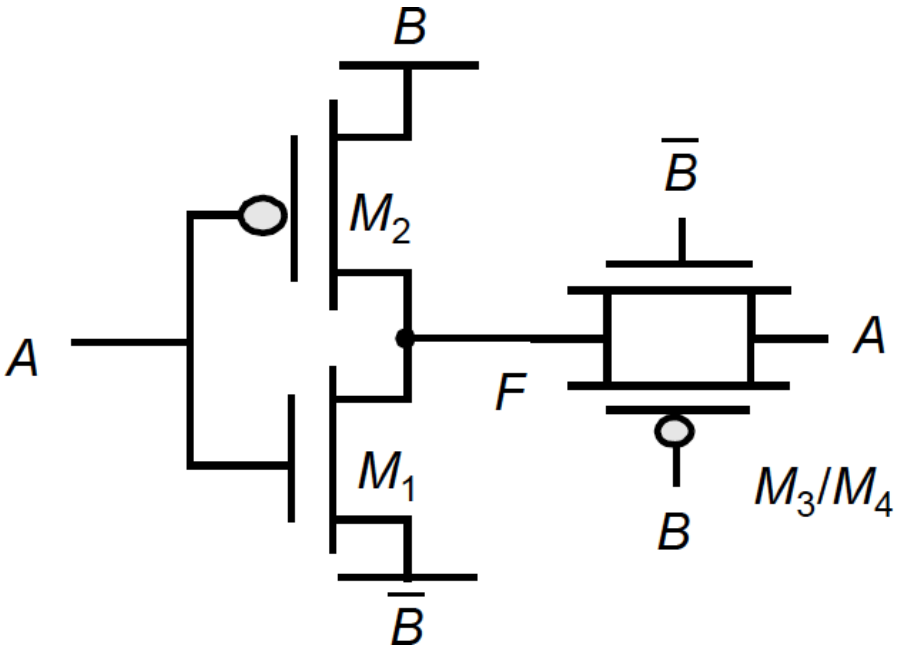
CMOS PASS TRANSISTOR CIRCUITS

- CMOS + Transmission Gates:
  - 2-input multiplexer
  - Gates should be restoring





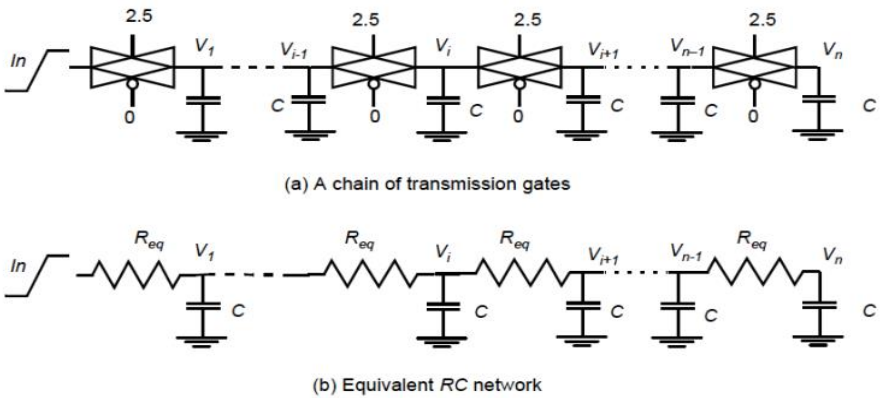
TRANSMISSION GATE XOR

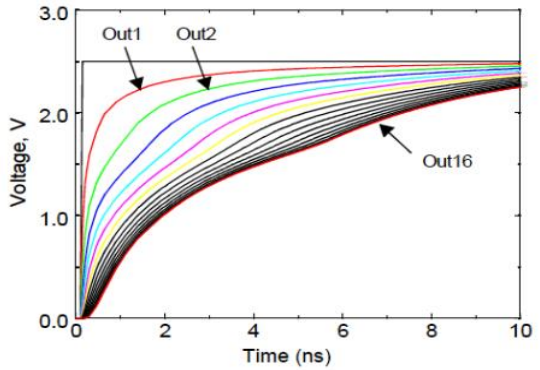


A	B	A **XOR** B
0	0	0
0	1	1
1	0	1
1	1	0

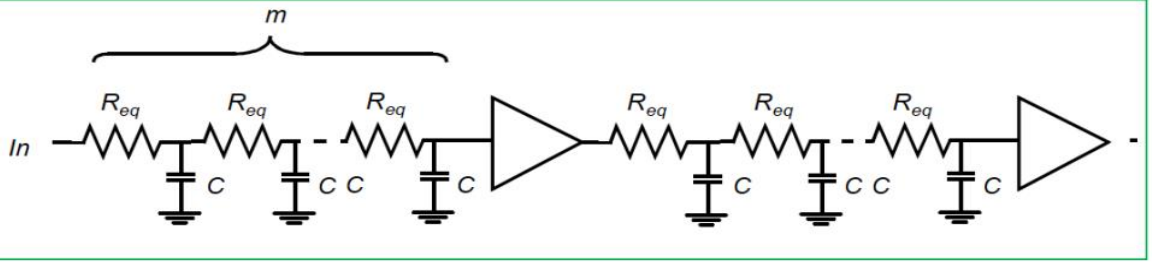
SPEED OPTIMIZATION

Problem



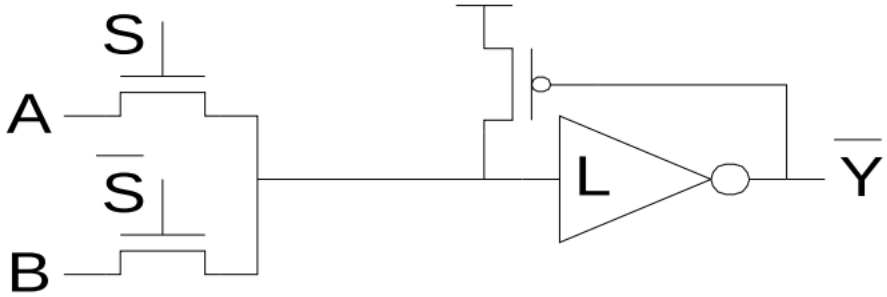


Solution

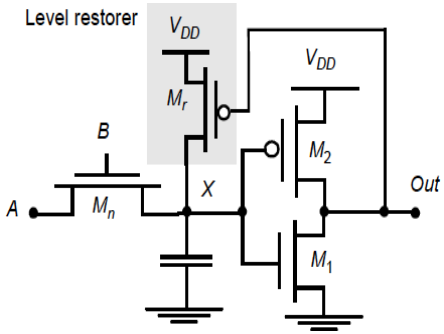


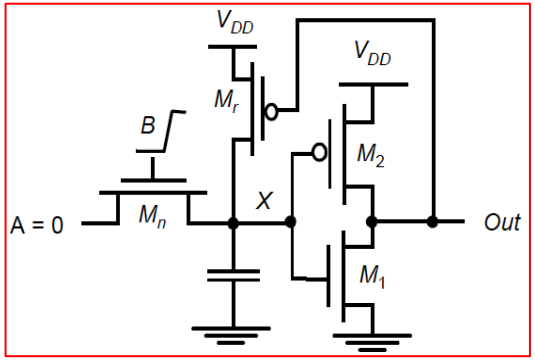
LEAP

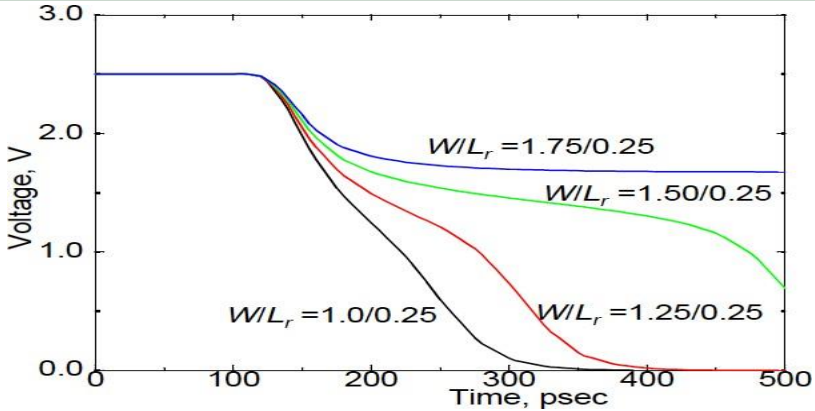
- LEAn integration with Pass transistors
- Get rid of pMOS transistors
  - Use weak pMOS feedback to pull fully high
  - Ratio constraint



LEVEL RESTORER (KEEPER PMOS)

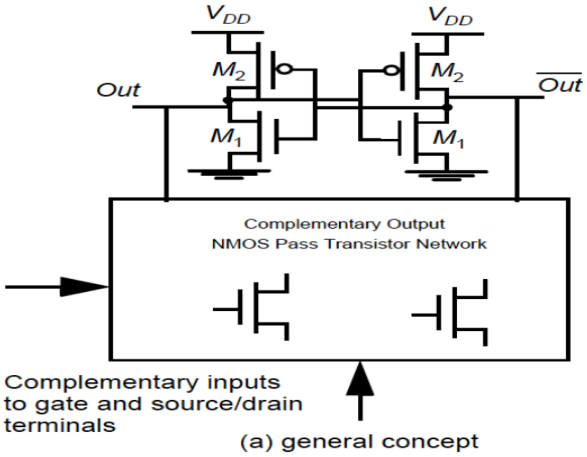




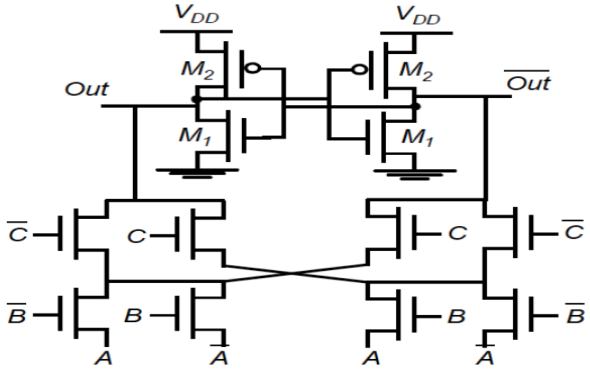


- While this solution is appealing in terms of eliminating static power dissipation, it adds complexity since the circuit is now ratioed.
- The problem arises during the transition of node X from high-to-low. The pass transistor network attempts to pull-down node X while the level restorer pulls X to VDD. Therefore, the pull-down device must be stronger than the pull-up device in order to switch node X and the output.

SWING-RESTORED PASS TRANSISTOR LOGIC



(a) general concept



(b) XOR/XNOR gate

Truth Table

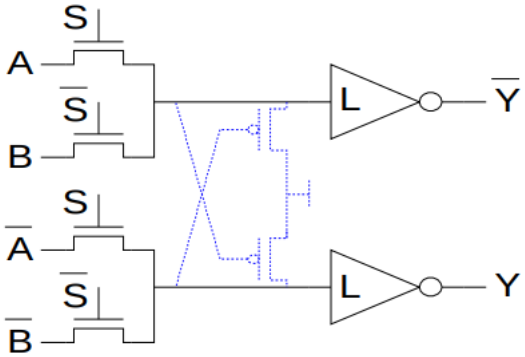
C	B	A	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

$$Y = \bar{C}\bar{B}A + \bar{C}B\bar{A} + C\bar{B}\bar{A} + CBA$$

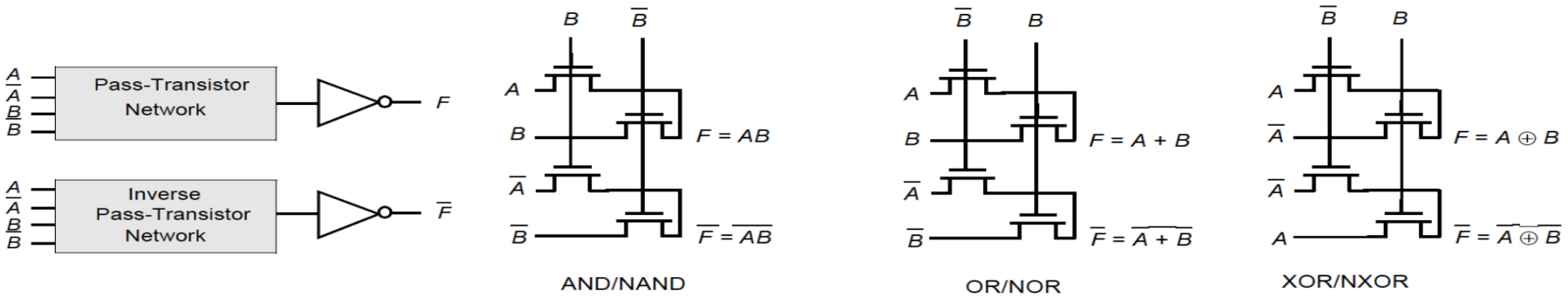
- A modification of the level-restorer.
- Instead of a simple inverter/PMOS combo at the output of the pass transistor network, two back-to-back inverters, configured in a cross-coupled fashion, are used for level restoration and performance improvement.

CPL

- Complementary Pass-transistor Logic
  - Dual-rail form of pass transistor logic
  - Avoids need for ratioed feedback
  - Optional cross-coupling for rail-to-rail swing



CPL



- For high performance design, a differential pass-transistor logic family, called DPL (or CPL), is commonly used.
- Since the circuits are differential, complementary data inputs and outputs are always available. Although generating the differential signals requires extra circuitry, the differential style has the advantage that some complex gates such as XORs and adders can be realized efficiently with a small number of transistors.
- CPL belongs to the class of static gates, because the output-defining nodes are always connected to either VDD or GND through a low resistance path. This is advantageous for the noise resilience.
- The design is very modular. In effect, all gates use exactly the same topology. Only the inputs are permutated. This makes the design of a library of gates very simple.

Thank you!