CPE/EE 427, CPE 527
VLSI Design I
Circuit Families

Department of Electrical and Computer Engineering
University of Alabama in Huntsville

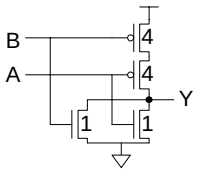
Aleksandar Milenkovic (www.ece.uah.edu/~milenka)

Outline

- Skewed Gates
- Pseudo-nMOS Logic
- Dynamic Logic
- Pass Transistor Logic

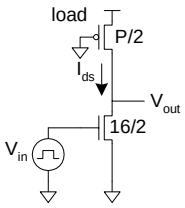
Introduction

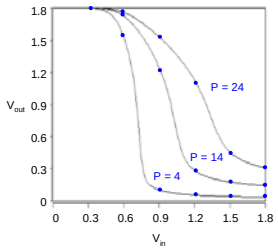
- What makes a circuit fast?
  - I = C dV/dt -> $t_{pd} \propto (C/I)\,\Delta V$
  - low capacitance
  - high current
  - small swing
- Logical effort is proportional to C/I
- pMOS are the enemy!
  - High capacitance for a given current
- Can we take the pMOS capacitance off the input?
- Various circuit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



Pseudo-nMOS

- In the old days, nMOS processes had no pMOS
  - Instead, use pull-up transistor that is always ON
- In CMOS, use a pMOS that is always ON
  - *Ratio* issue
  - Make pMOS about ¼ effective strength of pulldown network





Asymmetric Gates

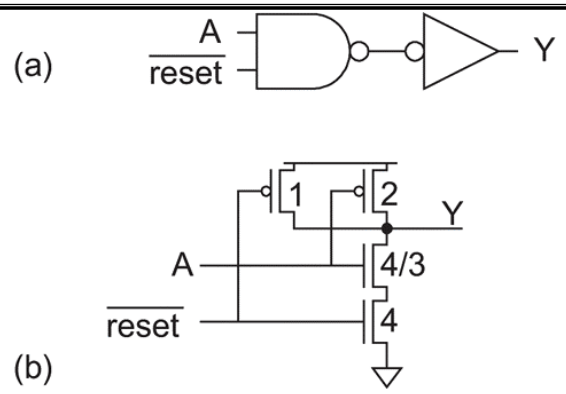


FIG 6.7 Resettable buffer optimized for data input

Skewed Gates

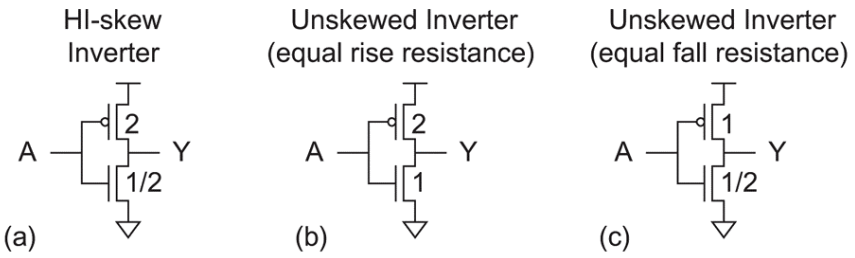


FIG 6.9 Logical effort calculation for HI-skew inverter

Skewed Gates

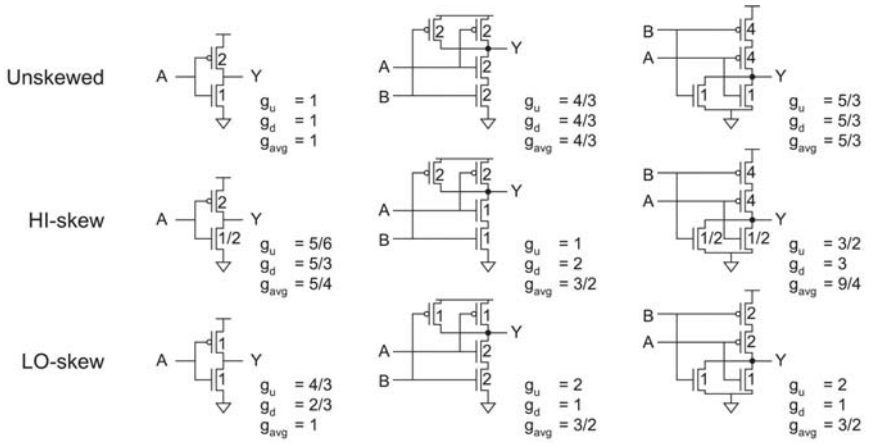
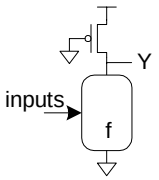


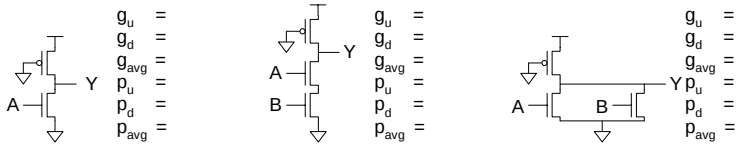
FIG 6.10 Catalog of skewed gates

Pseudo-nMOS Gates

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

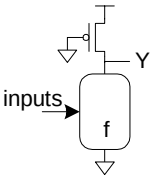


Inverter NAND2 NOR2

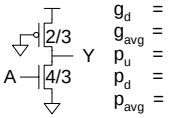


Pseudo-nMOS Gates

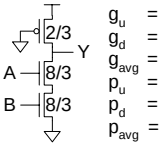
- Design for unit current on output to compare with unit inverter.
- pMOS fights nMOS



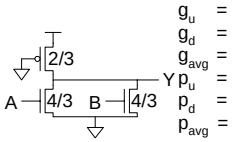
Inverter



NAND2

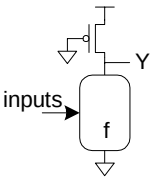


NOR2

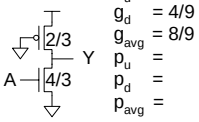


Pseudo-nMOS Gates

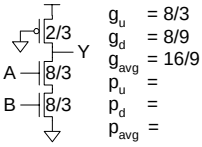
- Design for unit current on output to compare with unit inverter.
- pMOS fights nMOS



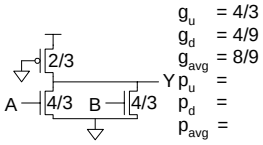
Inverter



NAND2

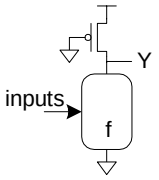


NOR2

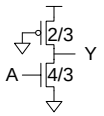


Pseudo-nMOS Gates

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

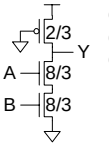


Inverter



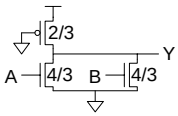
g_u = 4/3
g_d = 4/9
g_{avg} = 8/9
p_u = 6/3
p_d = 6/9
p_{avg} = 12/9

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

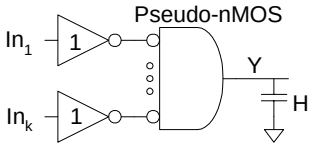


g_u = 4/3
g_d = 4/9
g_{avg} = 8/9
p_u = 10/3
p_d = 10/9
p_{avg} = 20/9

Pseudo-nMOS Design

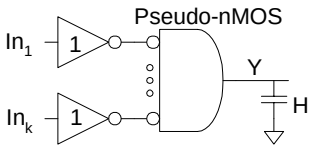
- Ex: Design a k-input AND gate using pseudo-nMOS. Estimate the delay driving a fanout of H

- G =
- F =
- P =
- N =
- D =



Pseudo-nMOS Design

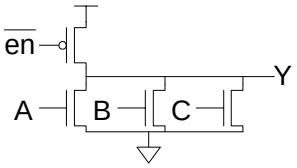
- Ex: Design a k-input AND gate using pseudo-nMOS. Estimate the delay driving a fanout of H



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

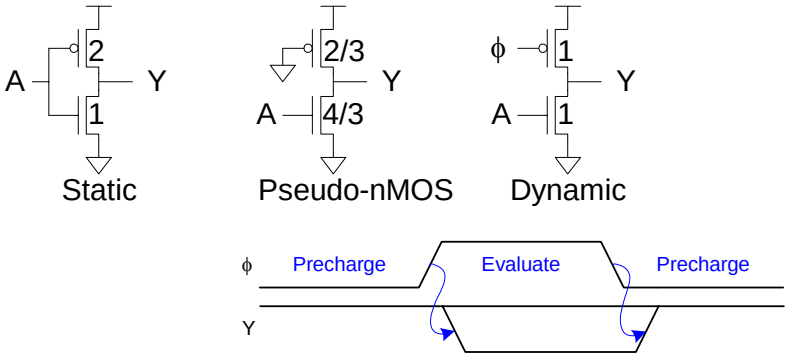
Pseudo-nMOS Power

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



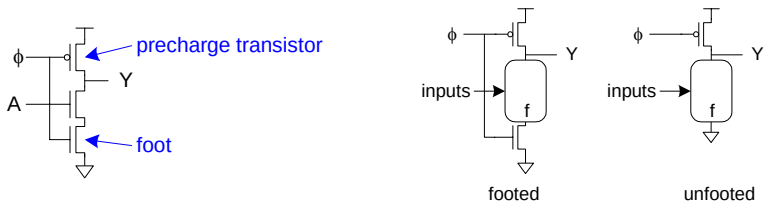
Dynamic Logic

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

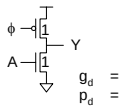
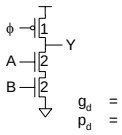
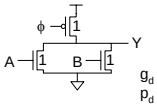
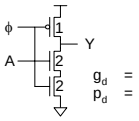
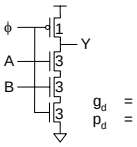
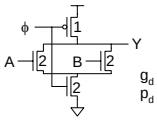


The Foot

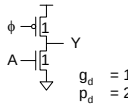
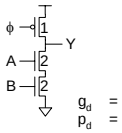
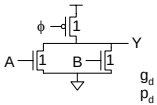
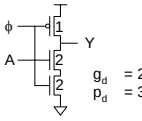
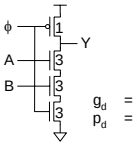
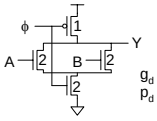
- What if pulldown network is ON during precharge?
- Use series evaluation transistor to prevent fight.



Logical Effort

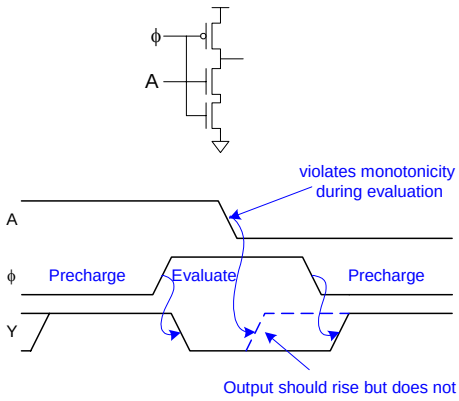
	Inverter	NAND2	NOR2
unfooted	 g_d = p_d =	 g_d = p_d =	 g_d = p_d =
footed	 g_d = p_d =	 g_d = p_d =	 g_d = p_d =

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

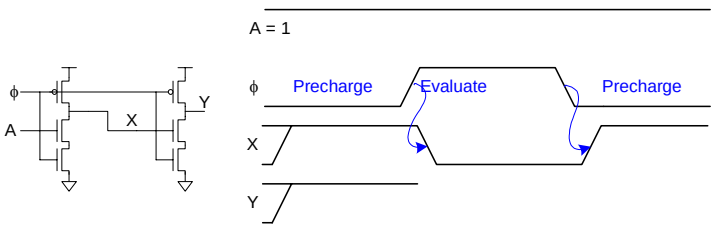
Monotonicity

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



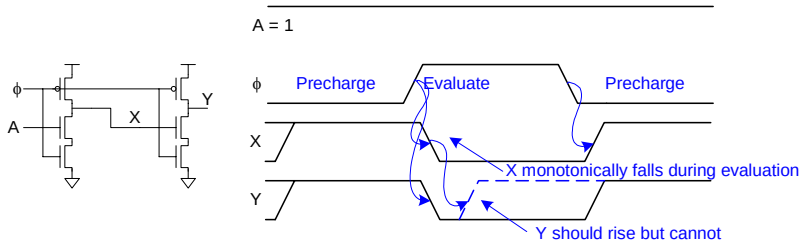
Monotonicity Woes

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



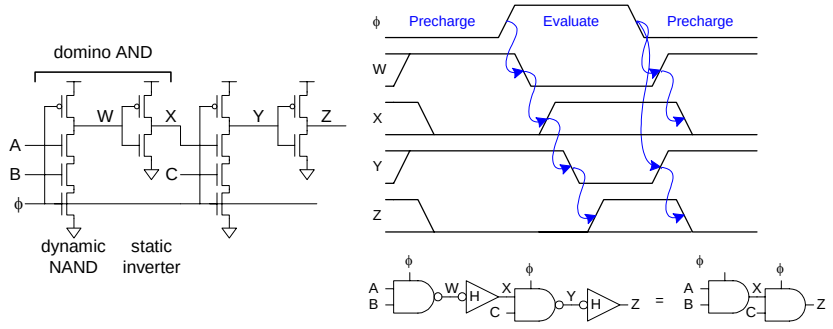
Monotonicity Woes

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



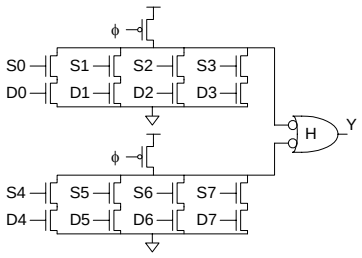
Domino Gates

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



Domino Optimizations

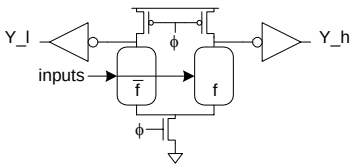
- Each domino gate triggers next one, like a string of dominos toppling over
- Gates evaluate sequentially but precharge in parallel
- Thus evaluation is more critical than precharge
- HI-skewed static stages can perform logic



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

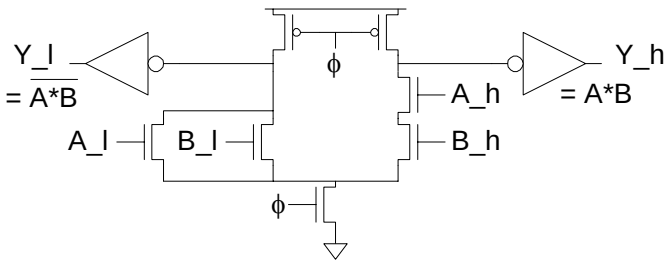


Example: AND/NAND

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

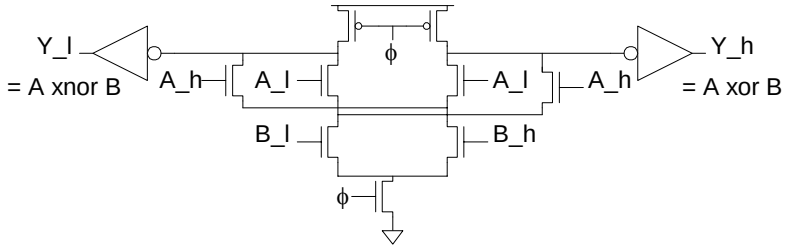
Example: AND/NAND

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



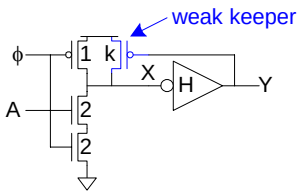
Example: XOR/XNOR

- Sometimes possible to share transistors



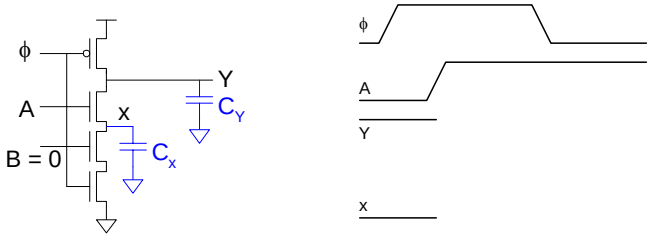
Leakage

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



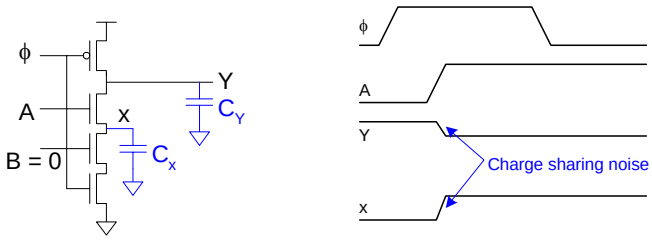
Charge Sharing

- Dynamic gates suffer from charge sharing



Charge Sharing

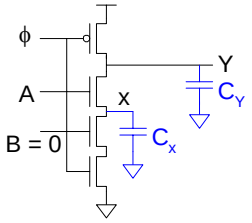
- Dynamic gates suffer from charge sharing

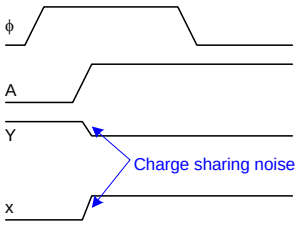


$$V_x = V_Y =$$

Charge Sharing

- Dynamic gates suffer from charge sharing

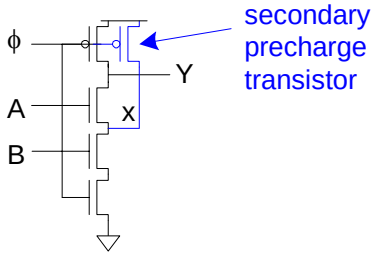




$$V_x = V_Y = \frac{C_Y}{C_x + C_Y} V_{DD}$$

Secondary Precharge

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



Noise Sensitivity

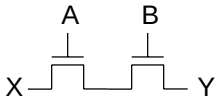
- Dynamic gates are very sensitive to noise
  - Inputs: $V_{IH} \approx V_{tn}$
  - Outputs: floating output susceptible noise
- Noise sources
  - Capacitive crosstalk
  - Charge sharing
  - Power supply noise
  - Feedthrough noise
  - And more!

Domino Summary

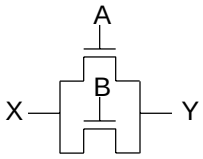
- Domino logic is attractive for high-speed circuits
  - 1.5 – 2x faster than static CMOS
  - But many challenges:
    - Monotonicity
    - Leakage
    - Charge sharing
    - Noise
- Widely used in high-performance microprocessors

NMOS Transistors in Series/Parallel

- Primary inputs drive both gate and source/drain terminals
- NMOS switch closes when the gate input is high



X = Y if A and B

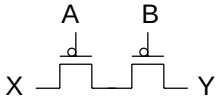


X = Y if A or B

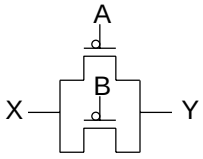
- Remember –
  NMOS transistors pass a strong 0 but a weak 1

PMOS Transistors in Series/Parallel

- Primary inputs drive both gate and source/drain terminals
- PMOS switch closes when the gate input is low



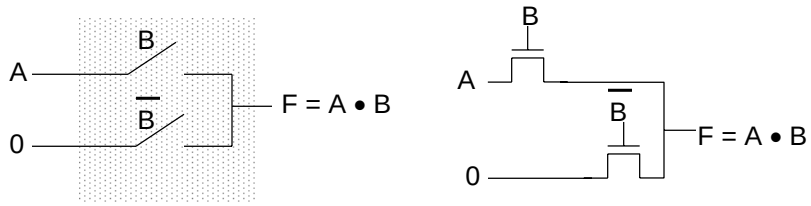
X = Y if $\overline{A}$ and $\overline{B} = \overline{A + B}$



X = Y if $\overline{A}$ or $\overline{B} = \overline{A \bullet B}$

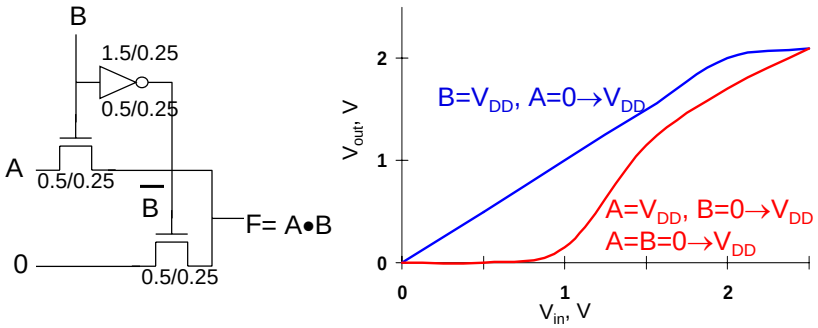
- Remember –
  PMOS transistors pass a strong 1 but a weak 0

Pass Transistor (PT) Logic



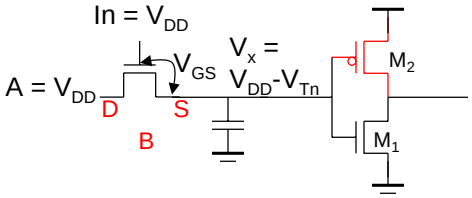
- Gate is static – a low-impedance path exists to both supply rails under all circumstances
- N transistors instead of 2N
- No static power consumption
- Ratioless
- Bidirectional (versus undirectional)

VTC of PT AND Gate



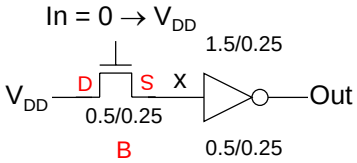
Pure PT logic is not regenerative - the signal gradually degrades after passing through a number of PTs (can fix with static CMOS inverter insertion)

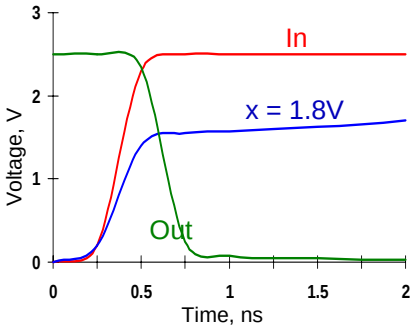
NMOS Only PT Driving an Inverter



- V_x does not pull up to V_{DD}, but $V_{DD} - V_{Tn}$
- Threshold voltage drop causes static power consumption (M_2 may be weakly conducting forming a path from V_{DD} to GND)
- Notice V_{Tn} increases of pass transistor due to body effect (V_{SB})

Voltage Swing of PT Driving an Inverter

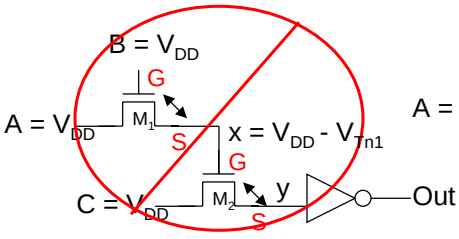


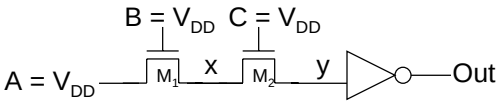


- Body effect – large V_{SB} at x - when pulling high (B is tied to GND and S charged up close to V_{DD})
- So the voltage drop is even worse

$$V_x = V_{DD} - (V_{Tn0} + \gamma(\sqrt{(|2\phi_f| + V_x)} - \sqrt{|2\phi_f|}))$$

Cascaded NMOS Only PTs



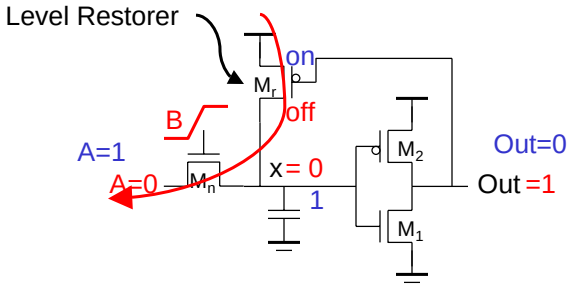


Swing on y = $V_{DD} - V_{Tn1} - V_{Tn2}$

Swing on y = $V_{DD} - V_{Tn1}$

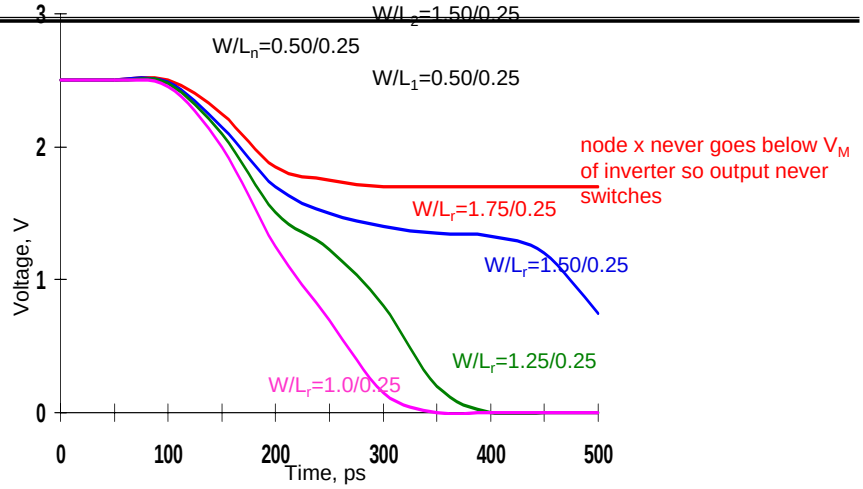
- Pass transistor gates should never be cascaded as on the left
- Logic on the right suffers from static power dissipation and reduced noise margins

Solution 1: Level Restorer



- Full swing on x (due to Level Restorer) so no static power consumption by inverter
- No static backward current path through Level Restorer and PT since Restorer is only active when A is high
- For correct operation M_r must be sized correctly (ratioed)

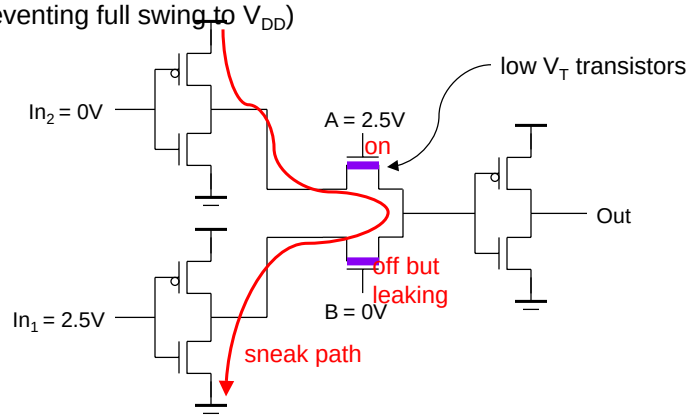
Transient Level Restorer Circuit Response



- Restorer has speed and power impacts: increases the capacitance at x, slowing down the gate; increases t_r (but decreases t_f)

Solution 2: Multiple V_T Transistors

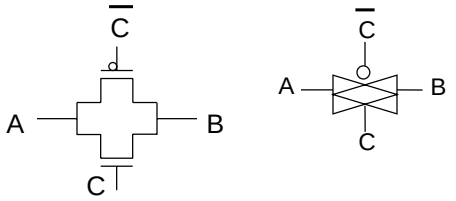
- Technology solution: Use (near) zero V_T devices for the NMOS PTs to eliminate *most* of the threshold drop (body effect still in force preventing full swing to V_{DD})

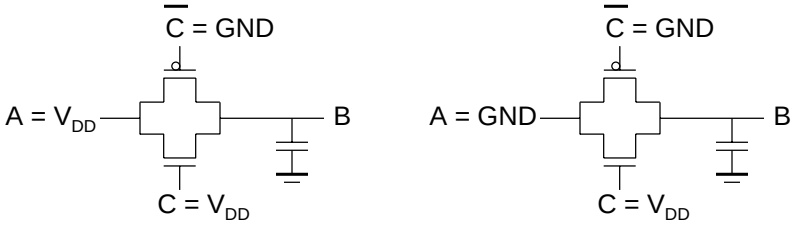


- Impacts static power consumption due to subthreshold currents flowing through the PTs (even if V_{GS} is below V_T)

Solution 3: Transmission Gates (TGs)

- Most widely used solution

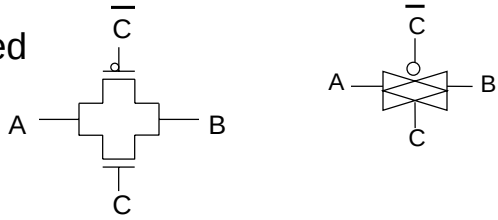


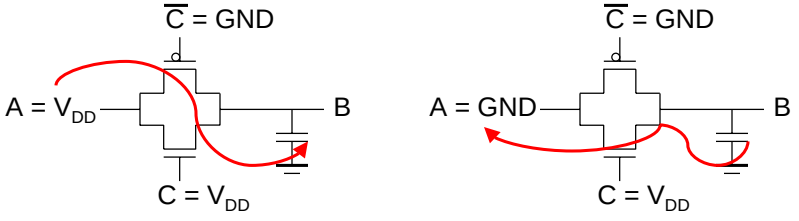


- Full swing bidirectional switch controlled by the gate signal C, A = B if C = 1

Solution 3: Transmission Gates (TGs)

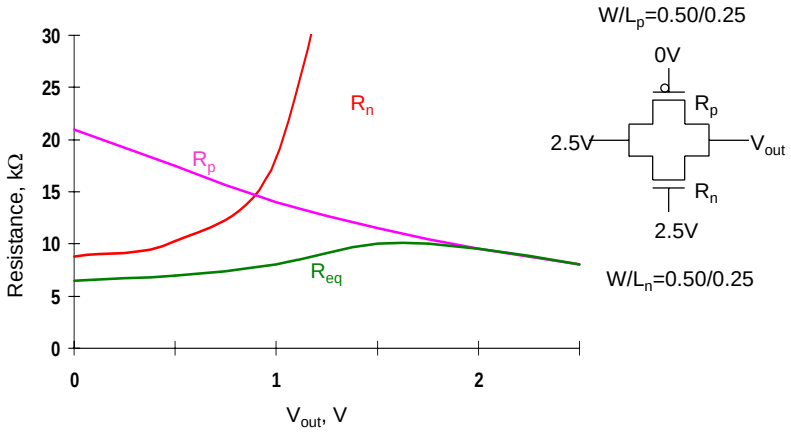
- Most widely used solution





- Full swing *bidirectional* switch controlled by the gate signal C, A = B if C = 1

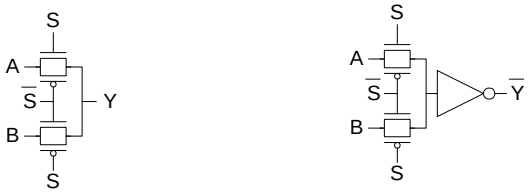
Resistance of TG



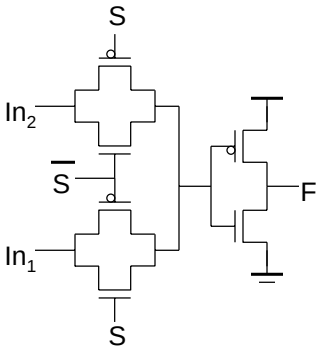
Pass Transistor Circuits

- Use pass transistors like switches to do logic
- Inputs drive diffusion terminals as well as gates

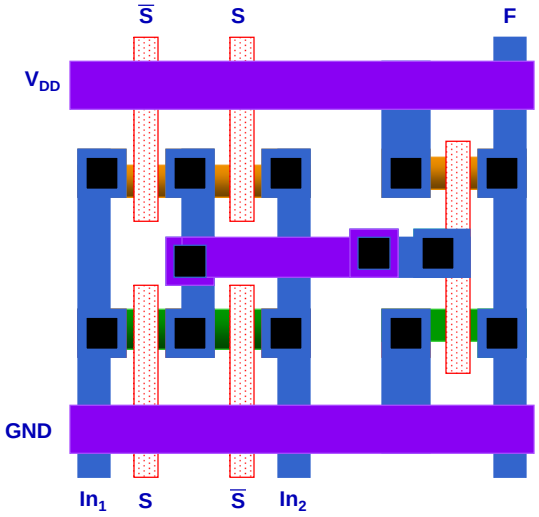
- CMOS + Transmission Gates:
  - 2-input multiplexer
  - Gates should be restoring



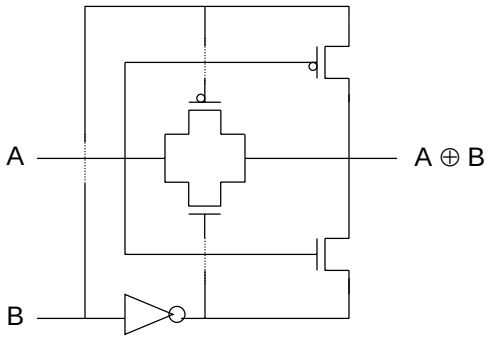
TG Multiplexer



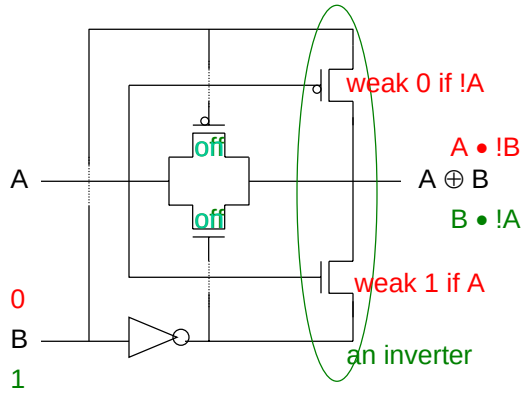
$F = !(In_1 \bullet S + In_2 \bullet \overline{S})$



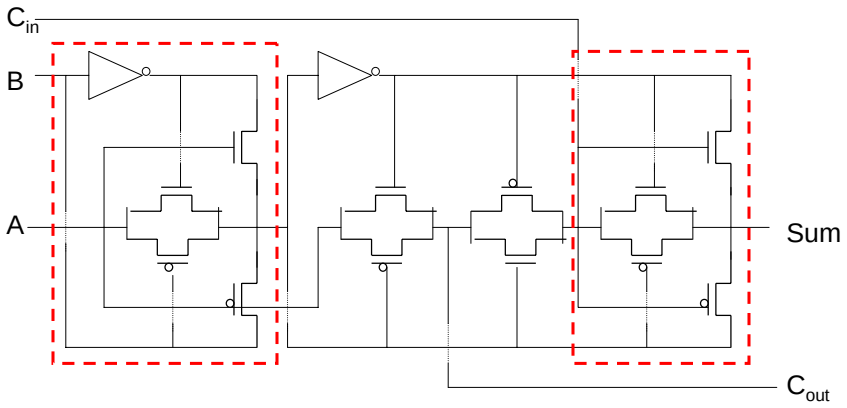
Transmission Gate XOR



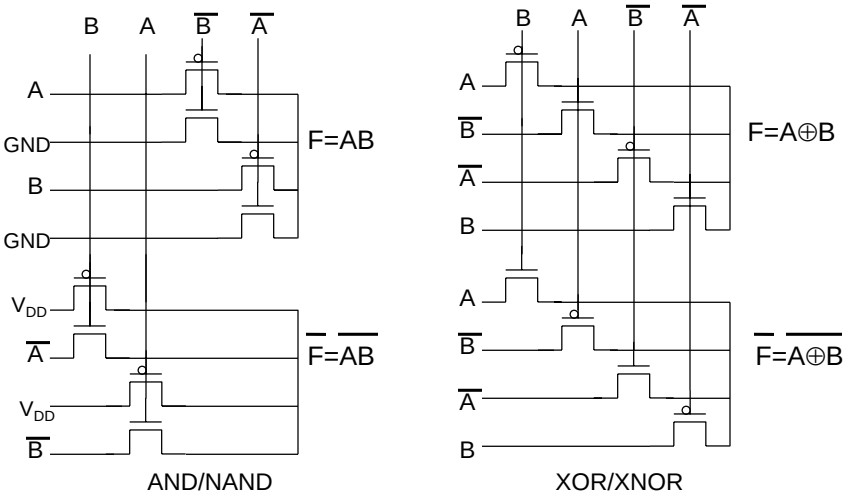
Transmission Gate XOR



TG Full Adder

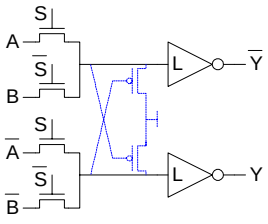


Differential TG Logic (DPL)

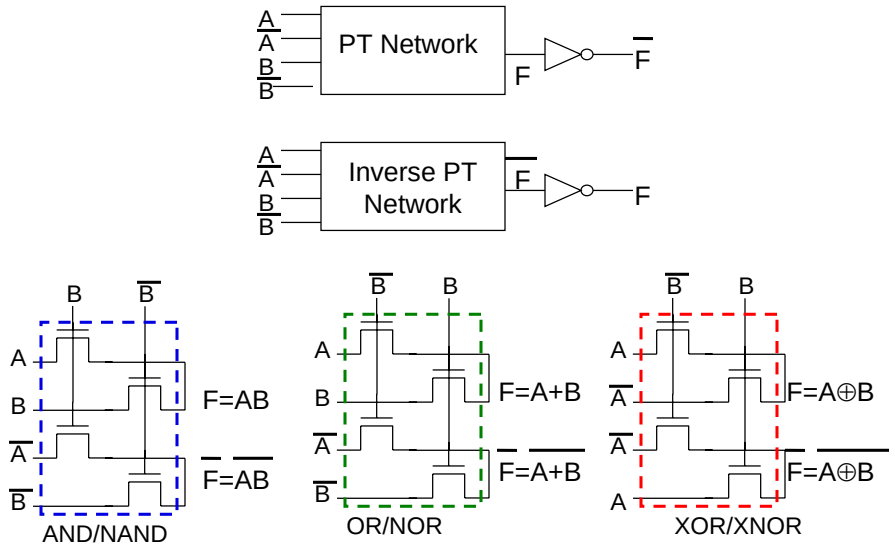


CPL

- **C**omplementary **P**ass-transistor **L**ogic
  - Dual-rail form of pass transistor logic
  - Avoids need for ratioed feedback
  - Optional cross-coupling for rail-to-rail swing



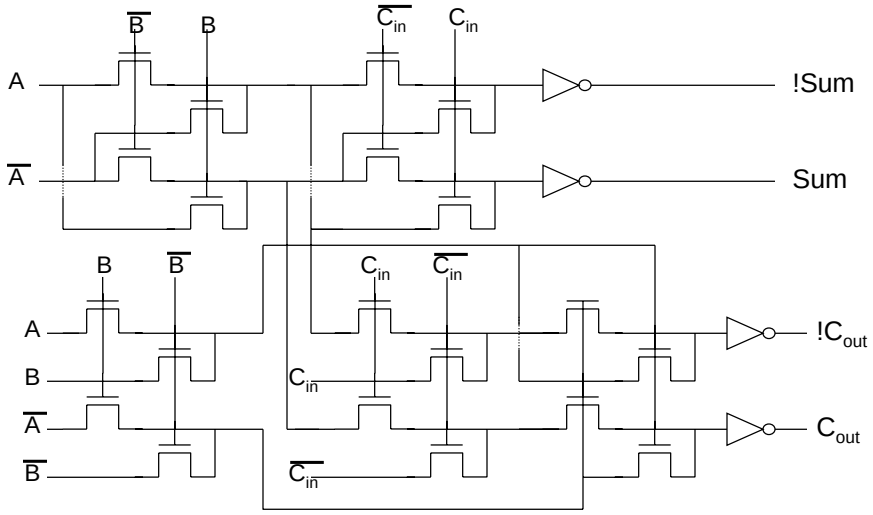
Differential PT Logic (CPL)



CPL Properties

- Differential so complementary data inputs and outputs are always available (so don't need extra inverters)
- Still static, since the output defining nodes are always tied to V_{DD} or GND through a low resistance path
- Design is modular; all gates use the same topology, only the inputs are permuted.
- Simple XOR makes it attractive for structures like adders
- Fast (assuming number of transistors in series is small)
- Additional routing overhead for complementary signals
- Still have static power dissipation problems

CPL Full Adder



CPL Full Adder