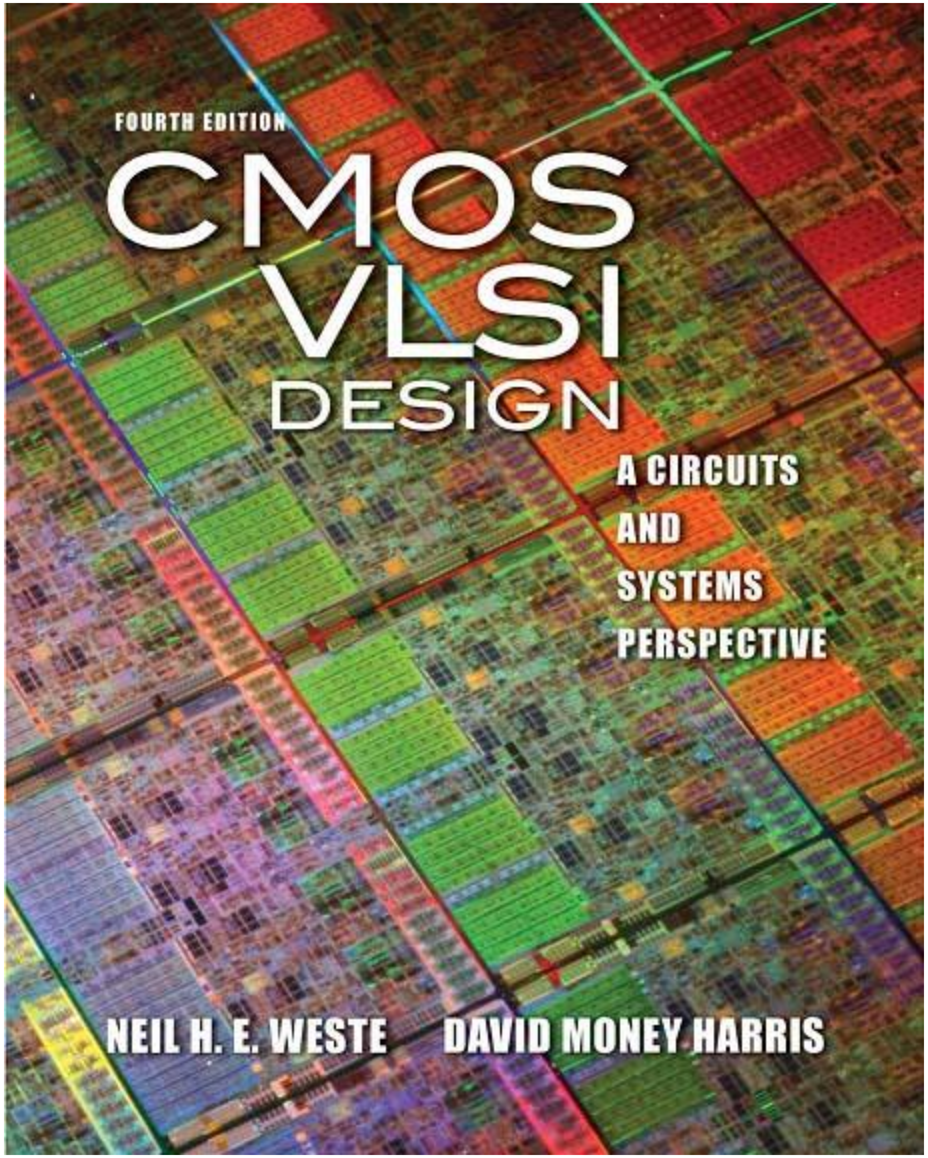


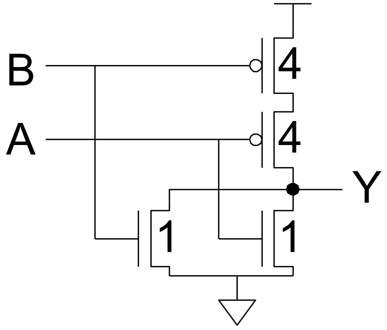
Lecture 10: Circuit Families

Outline

- ❑ Pseudo-nMOS Logic
- ❑ Dynamic Logic
- ❑ Pass Transistor Logic

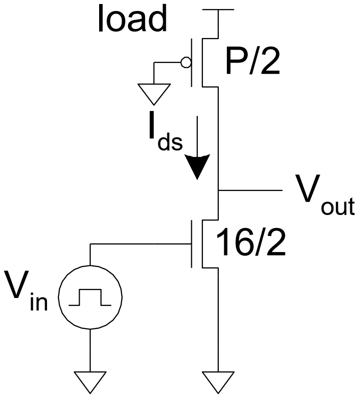
Introduction

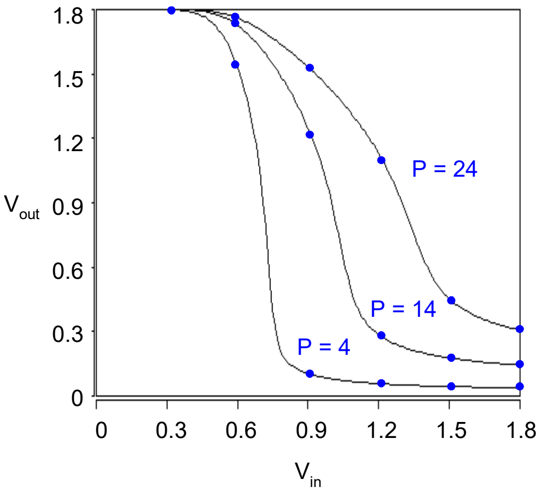
- ❑ What makes a circuit fast?
  - – $I = C\, dV/dt$ -> $t_{pd} \propto (C/I)\, \Delta V$
  - – low capacitance
  - – high current
  - – small swing
- ❑ Logical effort is proportional to C/I
- ❑ pMOS are the enemy!
  - – High capacitance for a given current
- ❑ Can we take the pMOS capacitance off the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



Pseudo-nMOS

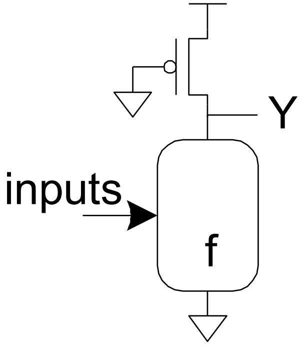
- □ In the old days, nMOS processes had no pMOS
  - – Instead, use pull-up transistor that is always ON
- □ In CMOS, use a pMOS that is always ON
  - – *Ratio* issue
  - – Make pMOS about ¼ effective strength of pulldown network



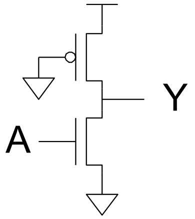


Pseudo-nMOS Gates

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

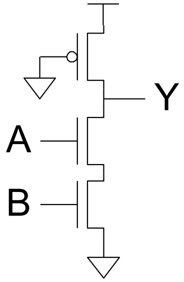


Inverter



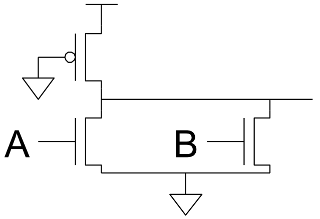
g_u =
g_d =
g_{avg} =
p_u =
p_d =
p_{avg} =

NAND2



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

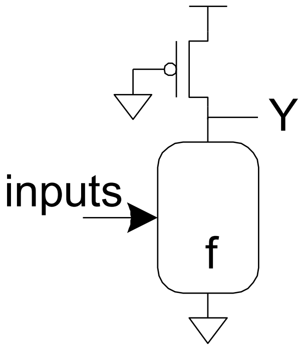
NOR2



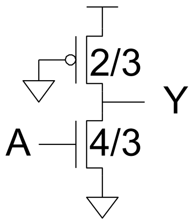
g_u =
g_d =
g_{avg} =
p_u =
p_d =
p_{avg} =

Pseudo-nMOS Gates

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

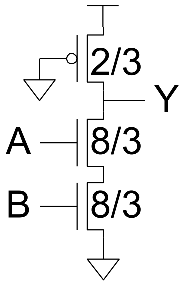


Inverter



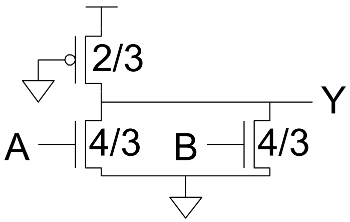
$g_u = 4/3$
$g_d = 4/9$
$g_{avg} = 8/9$
$p_u = 6/3$
$p_d = 6/9$
$p_{avg} = 12/9$

NAND2



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

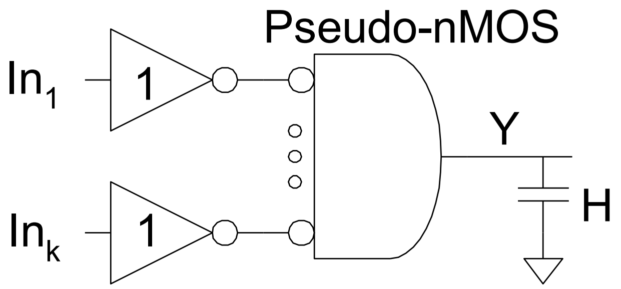
NOR2



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

Pseudo-nMOS Design

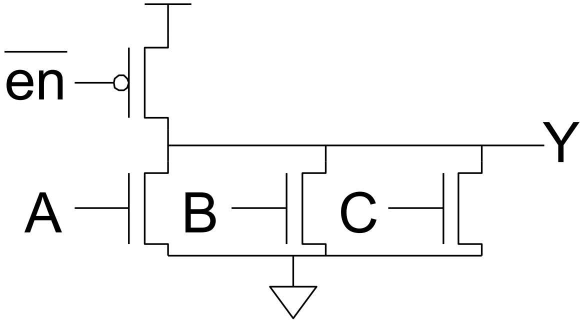
- ❑ Ex: Design a k-input AND gate using pseudo-nMOS. Estimate the delay driving a fanout of H



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

Pseudo-nMOS Power

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



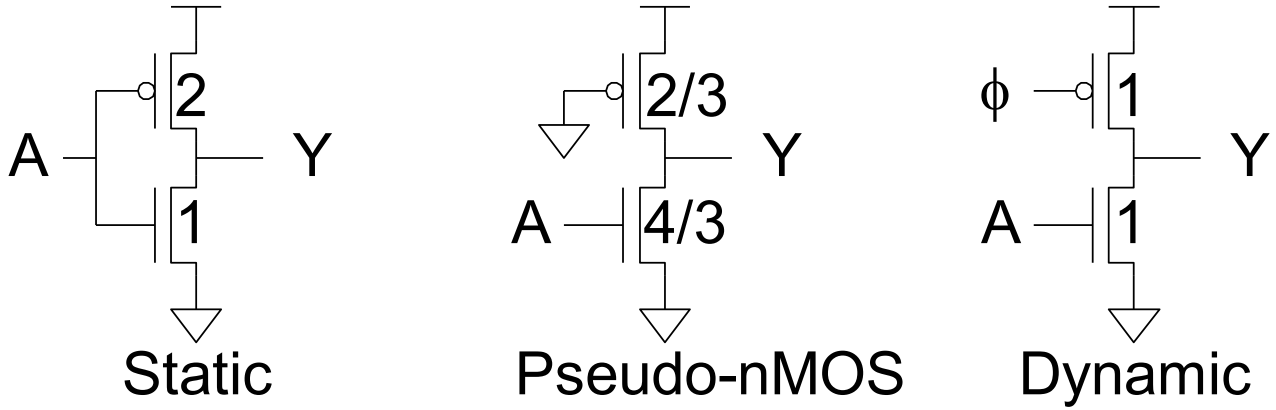
Ratio Example

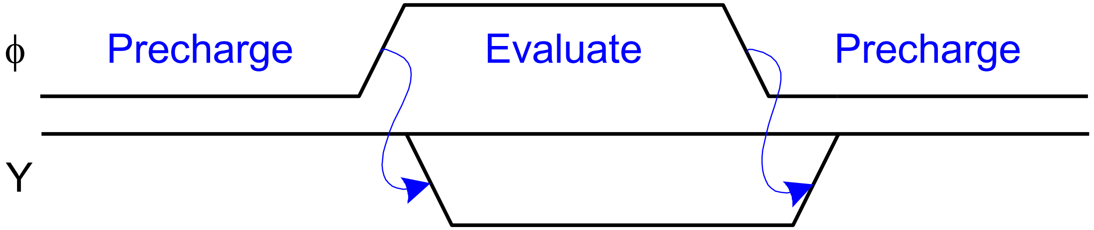
- The chip contains a 32 word x 48 bit ROM
  - Uses pseudo-nMOS decoder and bitline pullups
  - On average, one wordline and 24 bitlines are high
- Find static power drawn by the ROM
  - $I_{on\text{-}p} = 36\ \mu A$, $V_{DD} = 1.0\ V$
- Solution:

$$P_{\text{pull-up}} =$$
$$P_{\text{static}} =$$

Dynamic Logic

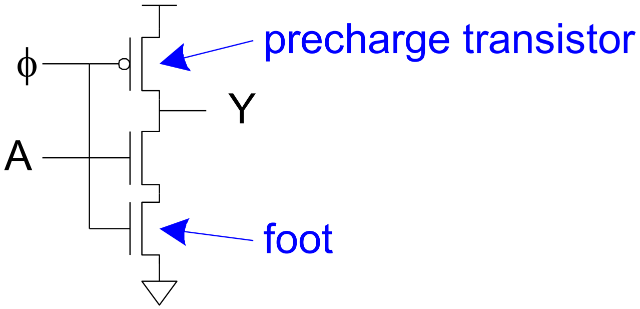
- ❑ *Dynamic* gates uses a clocked pMOS pullup
- ❑ Two modes: *precharge* and *evaluate*

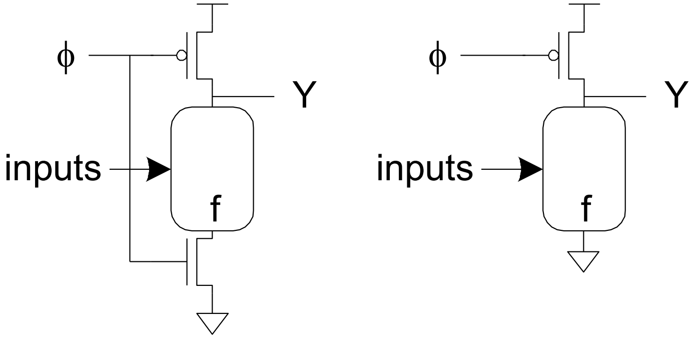




The Foot

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





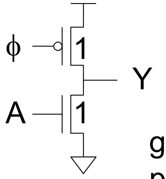
footed

unfooted

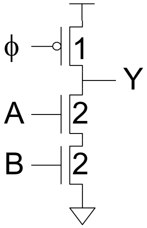
Logical Effort

Inverter NAND2 NOR2

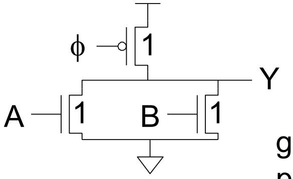
unfooted



g_d =
p_d =

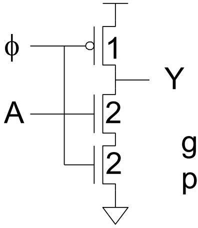


g_d =
p_d =

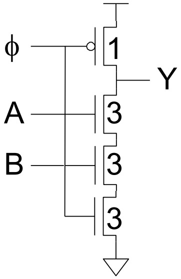


g_d =
p_d =

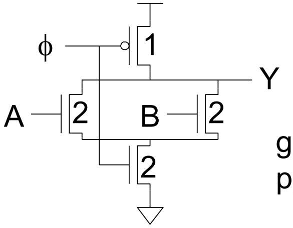
footed



g_d =
p_d =



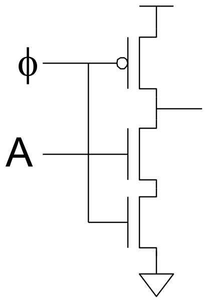
g_d =
p_d =

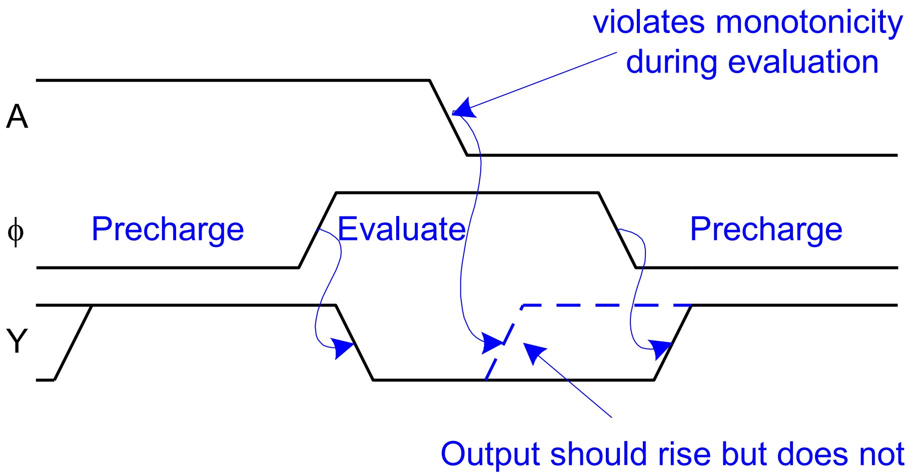


g_d =
p_d =

Monotonicity

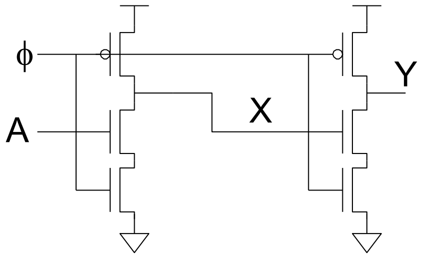
- ❑ Dynamic gates require *monotonically rising* inputs during evaluation
  - 0 -> 0
  - 0 -> 1
  - 1 -> 1
  - But not 1 -> 0

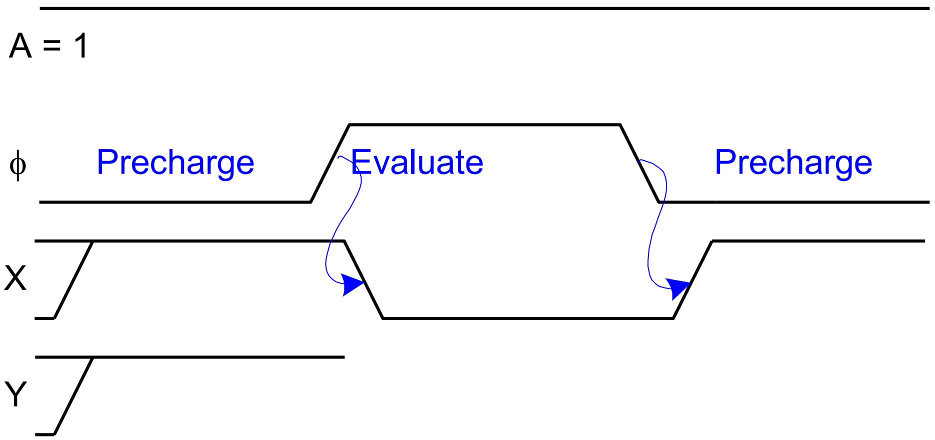



violates monotonicity during evaluation
A
ϕ
Precharge
Evaluate
Precharge
Y
Output should rise but does not

Monotonicity Woes

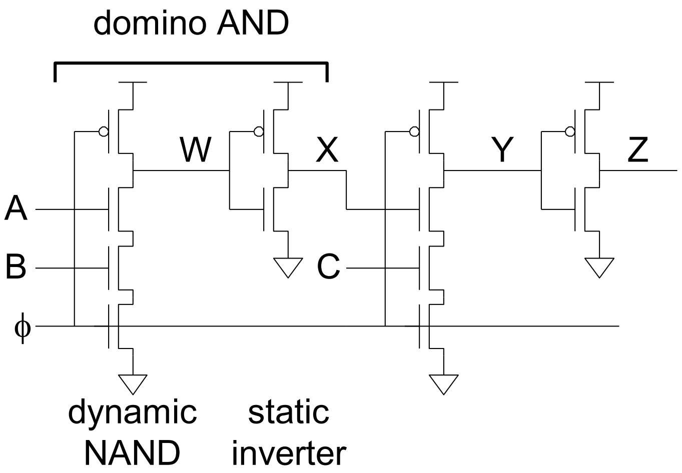
- □ But dynamic gates produce monotonically falling outputs during evaluation
- □ Illegal for one dynamic gate to drive another!

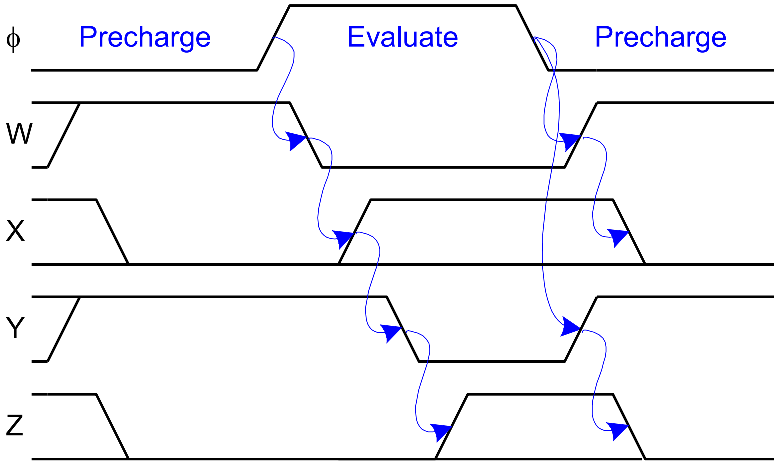


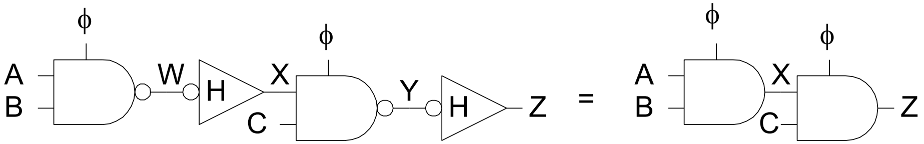


Domino Gates

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

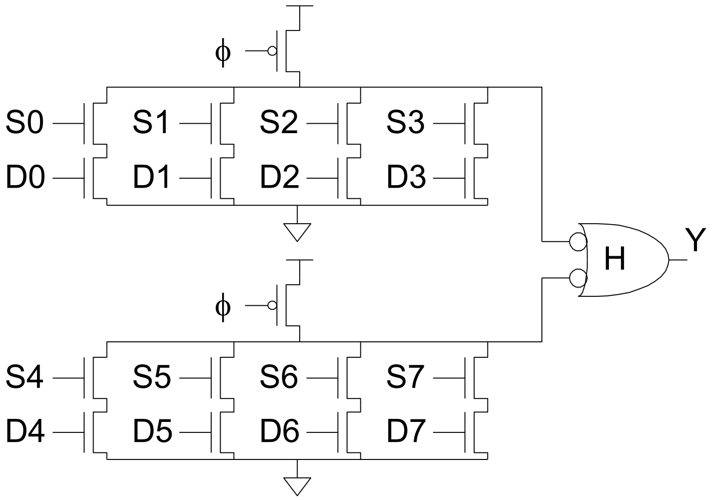






Domino Optimizations

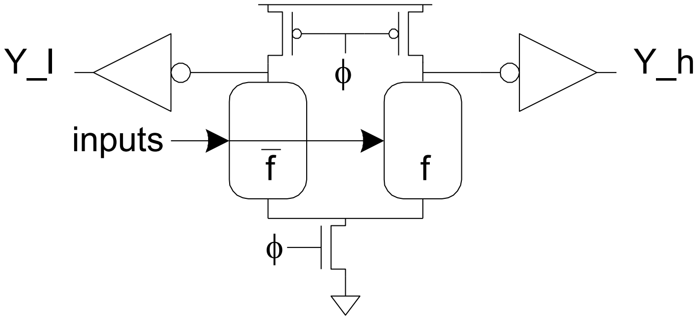
- ❑ Each domino gate triggers next one, like a string of dominos toppling over
- ❑ Gates evaluate sequentially but precharge in parallel
- ❑ Thus evaluation is more critical than precharge
- ❑ HI-skewed static stages can perform logic



Dual-Rail Domino

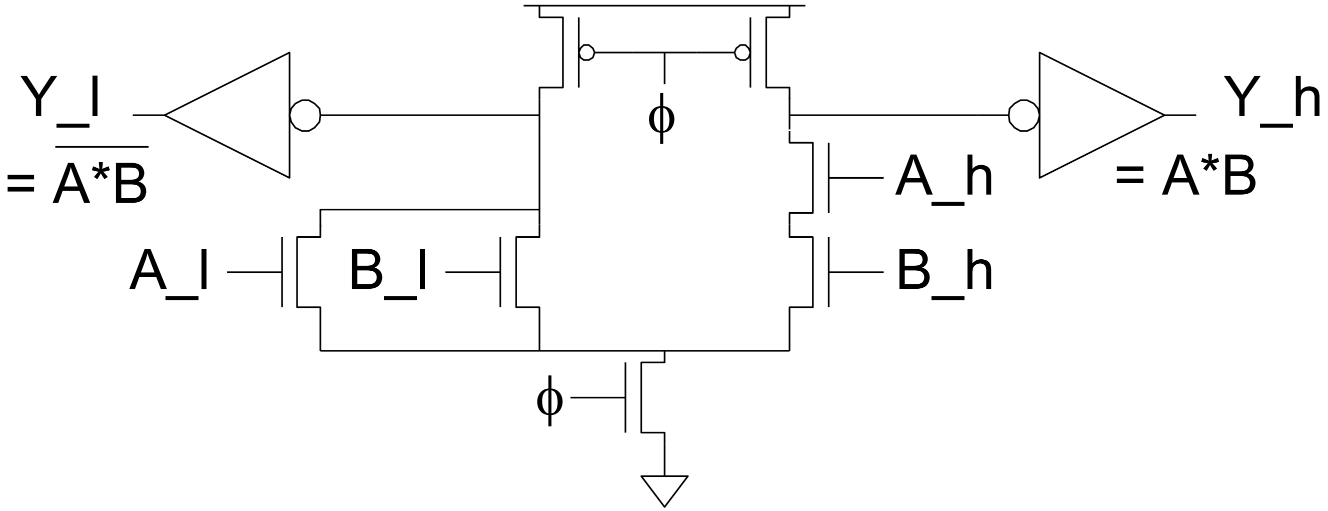
- Domino only performs noninverting functions:
  - AND, OR but not NAND, NOR, or XOR
- Dual-rail domino solves this problem
  - Takes true and complementary inputs
  - Produces true and complementary outputs

sig_h	sig_l	Meaning
0	0	Precharged
0	1	‘0’
1	0	‘1’
1	1	invalid



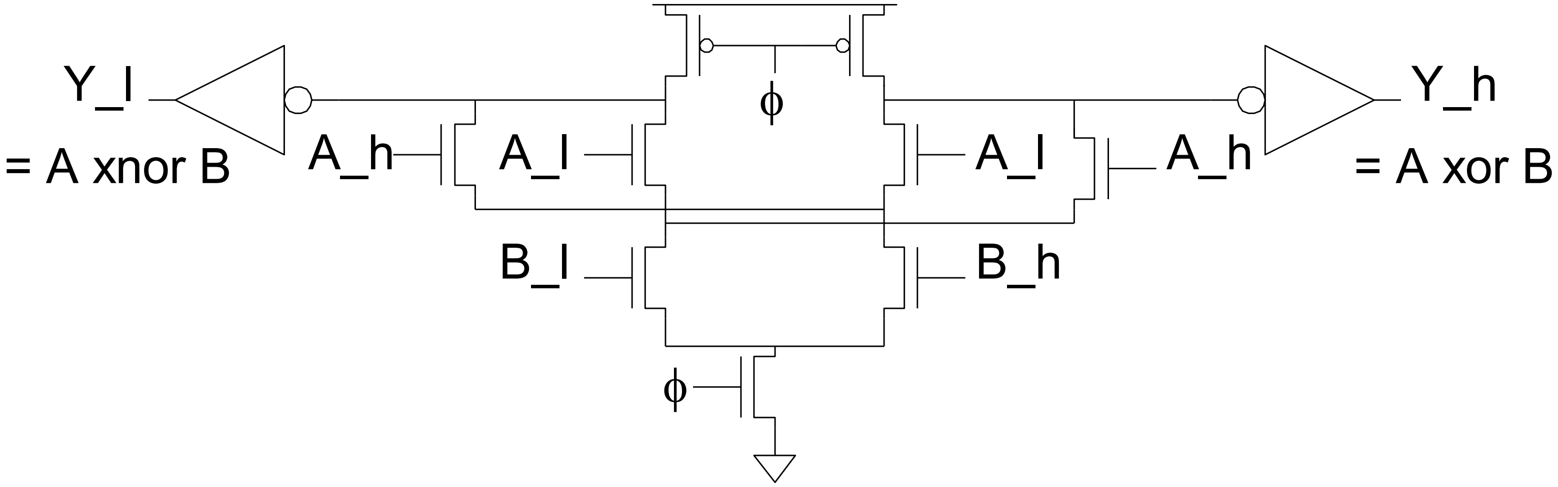
Example: AND/NAND

- ❑ Given A_h, A_l, B_h, B_l
- ❑ Compute Y_h = AB, Y_l = $\overline{AB}$
- ❑ Pulldown networks are conduction complements



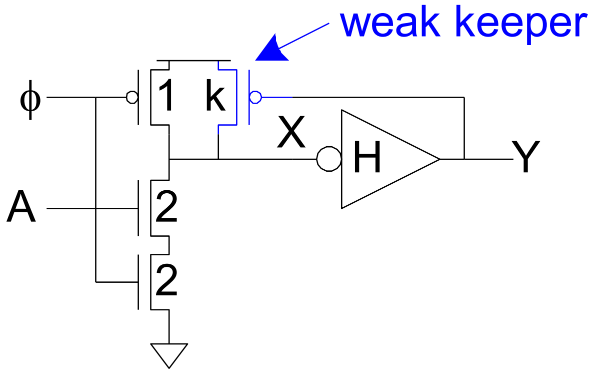
Example: XOR/XNOR

- ❑ Sometimes possible to share transistors



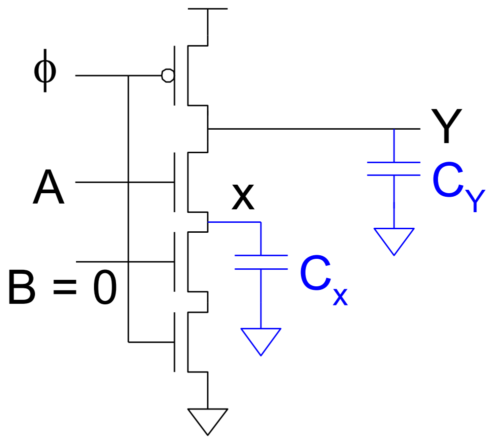
Leakage

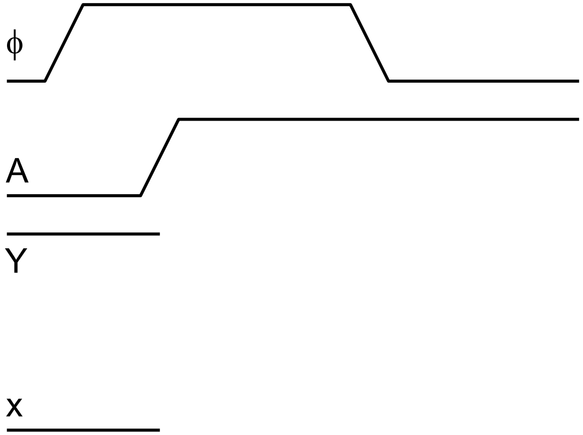
- Dynamic node floats high during evaluation
  - Transistors are leaky ($I_{OFF} \neq 0$)
  - Dynamic value will leak away over time
  - Formerly miliseconds, now nanoseconds
- Use keeper to hold dynamic node
  - Must be weak enough not to fight evaluation


weak keeper

Charge Sharing

- Dynamic gates suffer from charge sharing

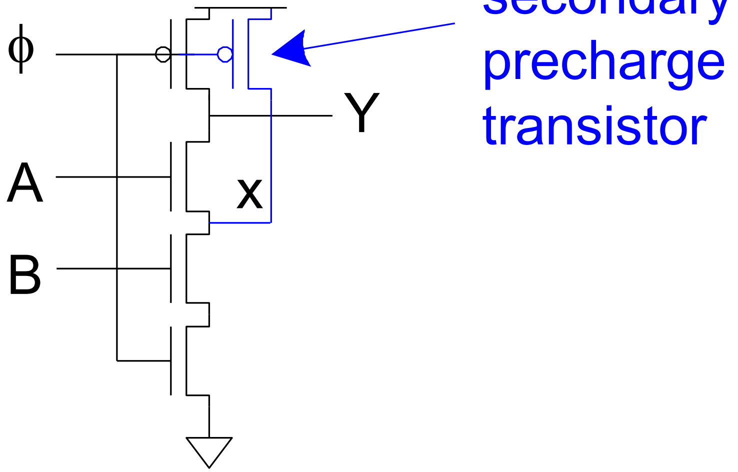




$$V_x = V_Y =$$

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!

Power

- □ Domino gates have high activity factors
  - – Output evaluates and precharges
    - If output probability = 0.5, $\alpha = 0.5$
      - – Output rises and falls on half the cycles
  - – Clocked transistors have $\alpha = 1$
- □ Leads to very high power consumption

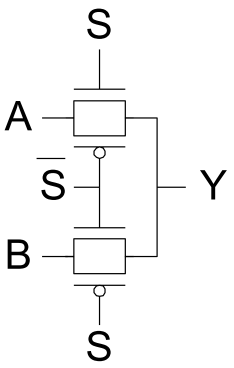
Domino Summary

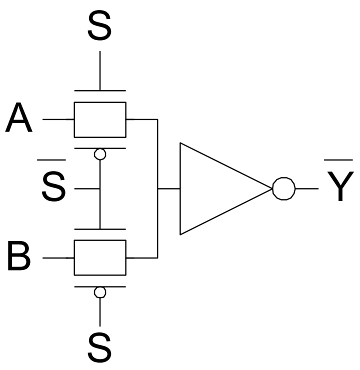
- ❑ Domino logic is attractive for high-speed circuits
  - – 1.3 – 2x faster than static CMOS
  - – But many challenges:
    - • Monotonicity, leakage, charge sharing, noise
- ❑ Widely used in high-performance microprocessors in 1990s when speed was king
- ❑ Largely displaced by static CMOS now that power is the limiter
- ❑ Still used in memories for area efficiency

Pass Transistor Circuits

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

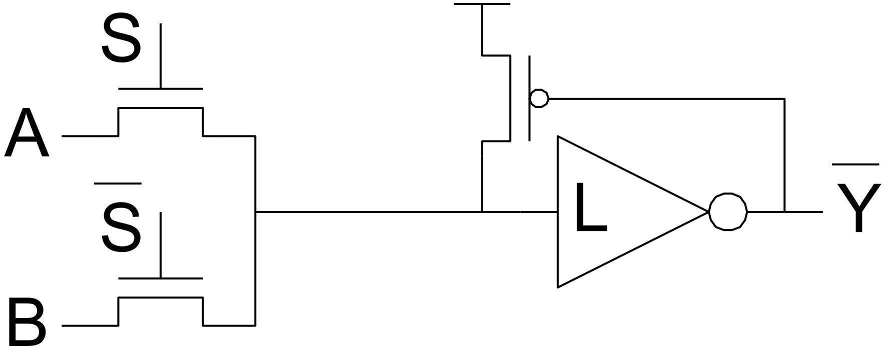
- CMOS + Transmission Gates:
  - 2-input multiplexer
  - Gates should be restoring





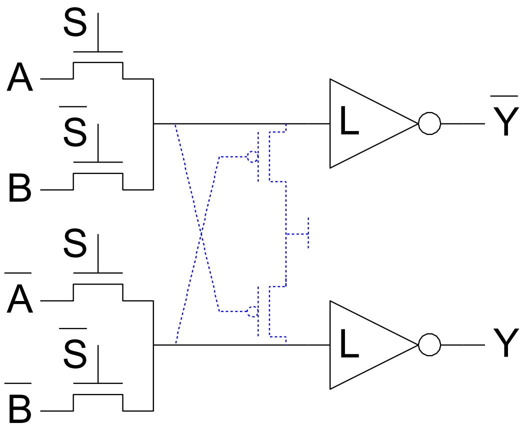
LEAP

- **LEA**n integration with **P**ass transistors
- Get rid of pMOS transistors
  - Use weak pMOS feedback to pull fully high
  - Ratio constraint



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



Pass Transistor Summary

- Researchers investigated pass transistor logic for general purpose applications in the 1990’s
  - Benefits over static CMOS were small or negative
  - No longer generally used
- However, pass transistors still have a niche in special circuits such as memories where they offer small size and the threshold drops can be managed