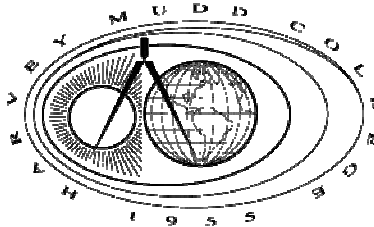# Introduction to CMOS VLSI Design

Lecture 9: Circuit Families

David Harris


Harvey Mudd College

Spring 2004

Outline

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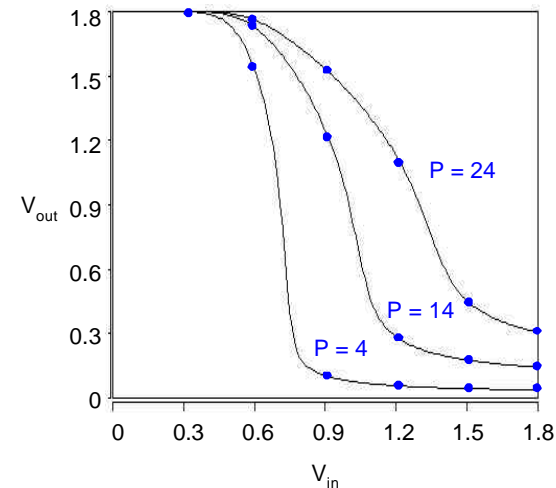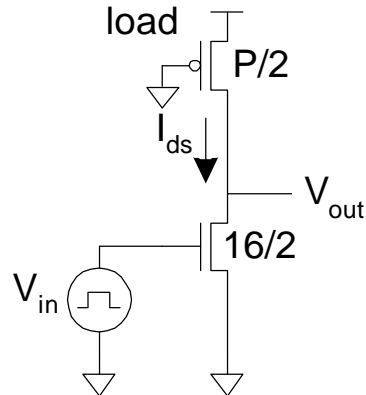- ❑ Dynamic Logic
- ❑ Pass Transistor Logic

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?
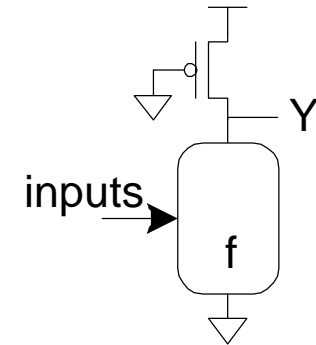
❑ Various circuit families try to do this…
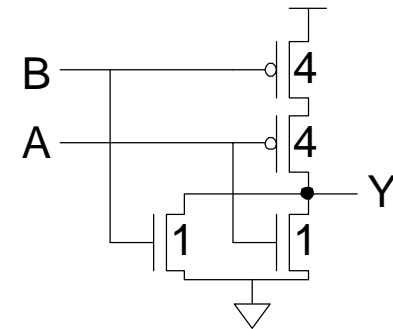
Pseudo-nMOS

❑ 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

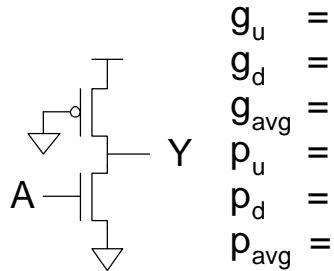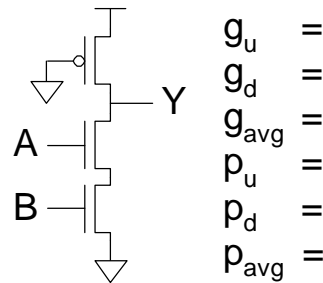   – Make pMOS about ¼ effective strength of pulldown network

Pseudo-nMOS Gates

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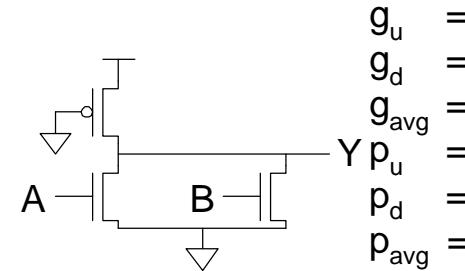   to compare with unit inverter.
❑ pMOS fights nMOS

inputs → Y

f

Inverter   NAND2   NOR2

g_u = g_u = g_u =
g_d = g_d = g_d =
g_{avg} = g_{avg} = g_{avg} =
p_u = p_u = p_u =
p_d = p_d = p_d =
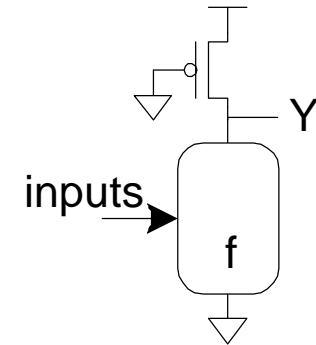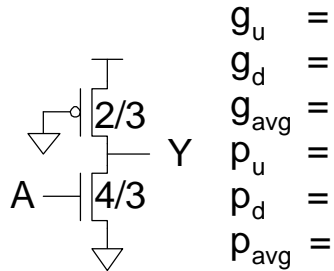p_{avg} = p_{avg} = p_{avg} =
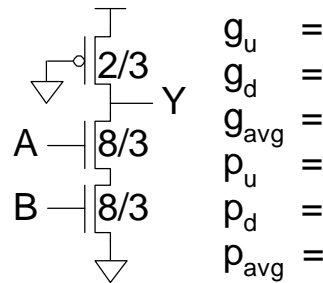
Pseudo-nMOS Gates

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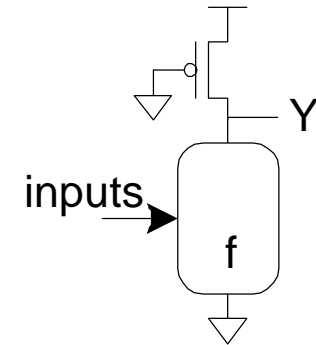to compare with unit inverter.
❑ pMOS fights nMOS

inputs

Y

f

Inverter

A —| 4/3   Y

2/3

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

NAND2

2/3

A —| 8/3   Y

B —| 8/3

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

NOR2

2/3

A —| 4/3   B —| 4/3   Y

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

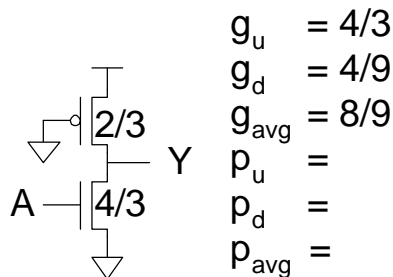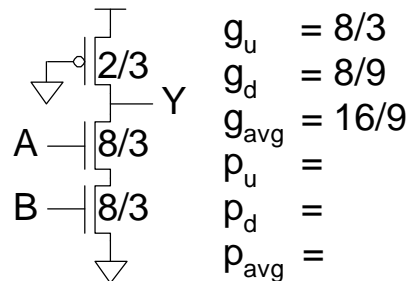# Pseudo-nMOS Gates

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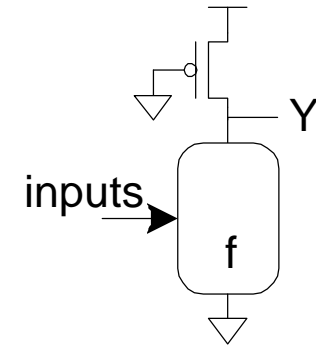   to compare with unit inverter.
❑ pMOS fights nMOS

inputs → Y, f

Inverter

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

(2/3, A — 4/3, Y)

NAND2

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

(2/3, Y, A — 8/3, B — 8/3)

NOR2

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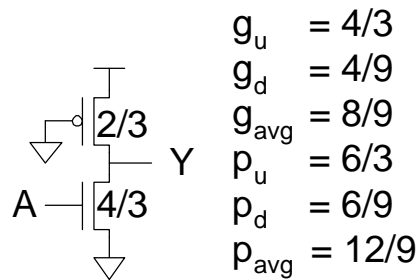g_{avg} = 8/9
p_u =
p_d =
p_{avg} =

(2/3, A — 4/3, B — 4/3, Y)

Pseudo-nMOS Gates

❑ Design for unit current on output
   to compare with unit inverter.

❑ pMOS fights nMOS

inputs → f → Y

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

2/3

A — 4/3 Y

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

2/3

A — 8/3 Y

B — 8/3

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

2/3

A — 4/3   B — 4/3 Y
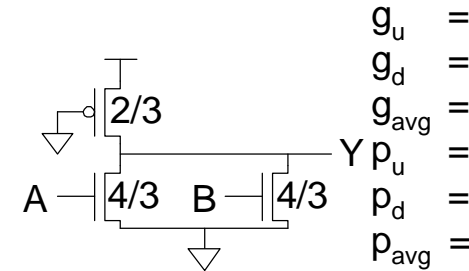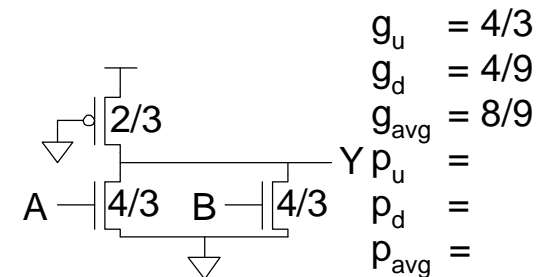
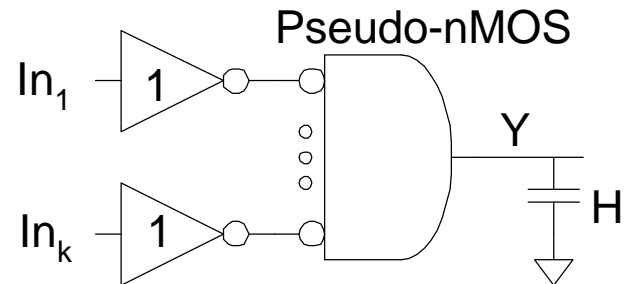# Pseudo-nMOS Design

❑ 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

In_1 — 1
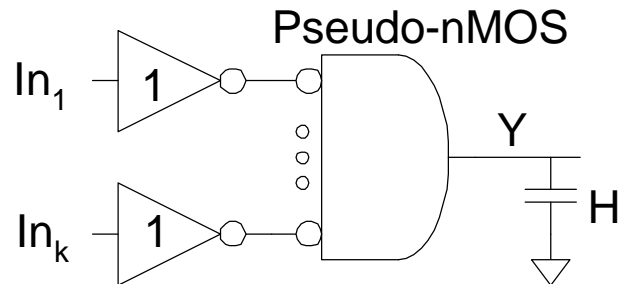
In_k — 1

Y

H

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

❑ $D = NF^{1/N} + P = \dfrac{4\sqrt{2H}}{3} + \dfrac{8k+13}{9}$
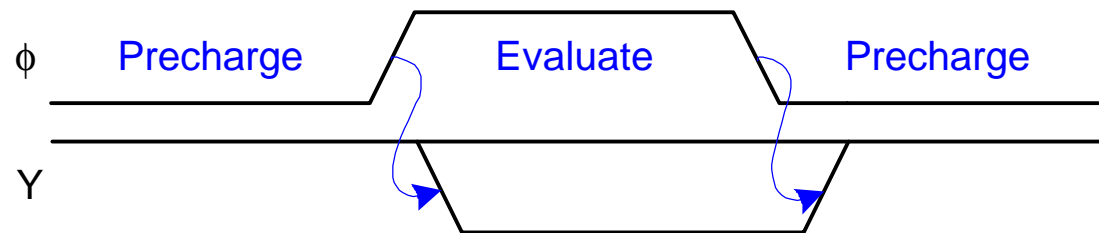
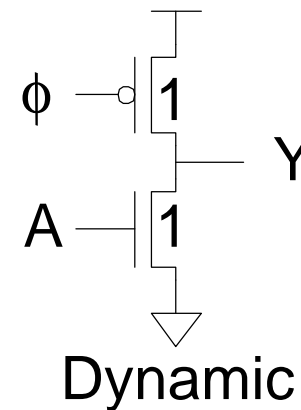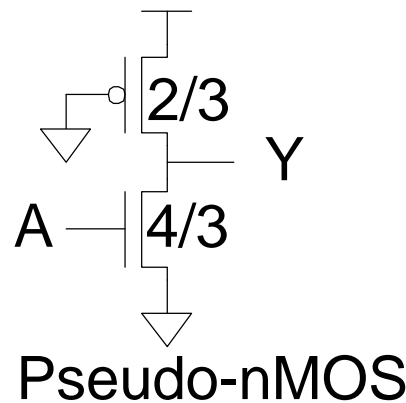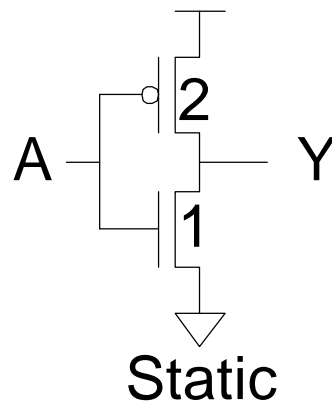Pseudo-nMOS

In_1 — ▷ 1 ◦ ◦

Y

In_k — ▷ 1 ◦ ◦

H

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!

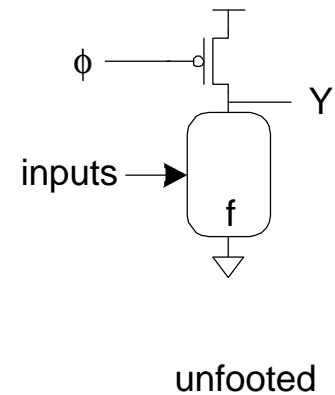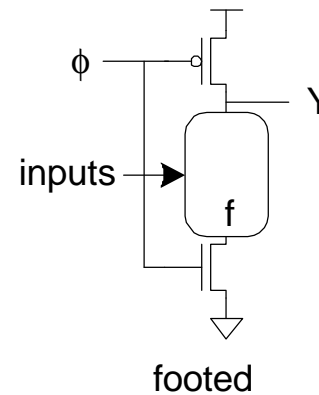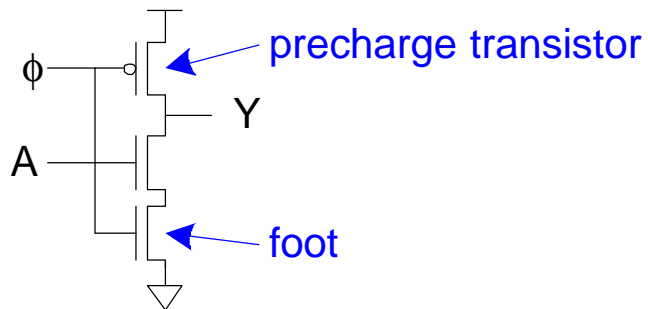❑ Use pseudo-nMOS sparingly for wide NORs

❑ Turn off pMOS when not in use
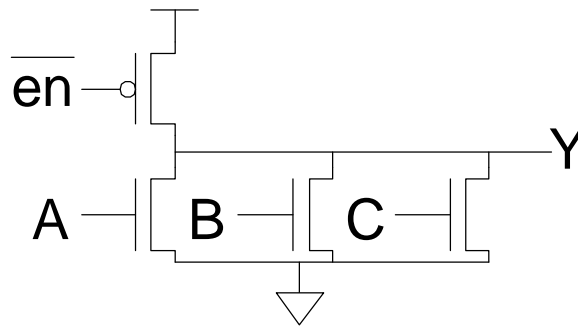
Dynamic Logic

❑ *Dynamic* gates uses a clocked pMOS pullup

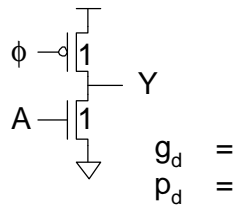❑ Two modes: *precharge* and *evaluate*

Static   Pseudo-nMOS   Dynamic

The Foot

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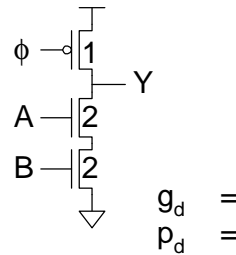
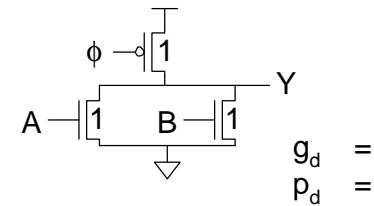❑ Use series evaluation transistor to prevent fight.

precharge transistor

foot

footed   unfooted

Logical Effort

Inverter

NAND2

NOR2

unfooted

footed

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

$g_d = 2/3$
$p_d = 3/3$

$g_d = 3/3$
$p_d = 4/3$

$g_d = 2/3$
$p_d = 5/3$
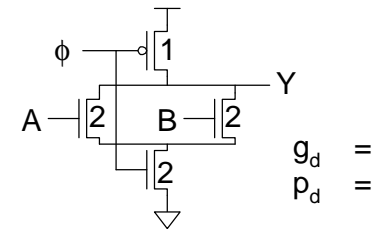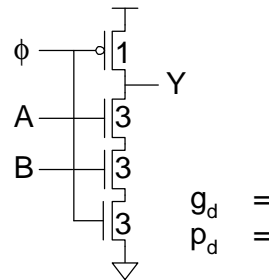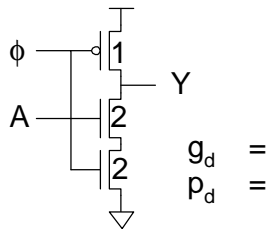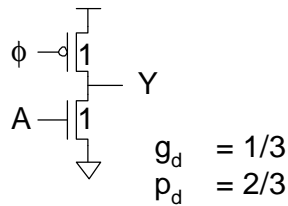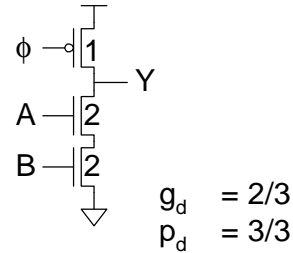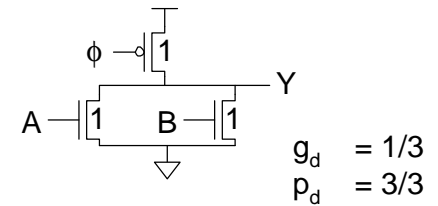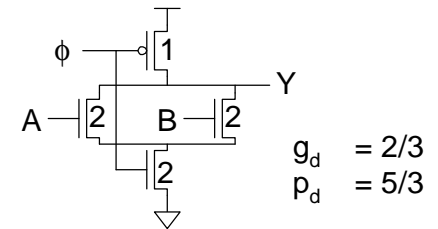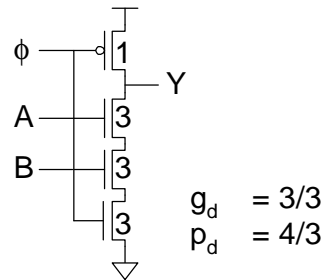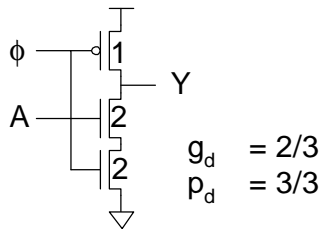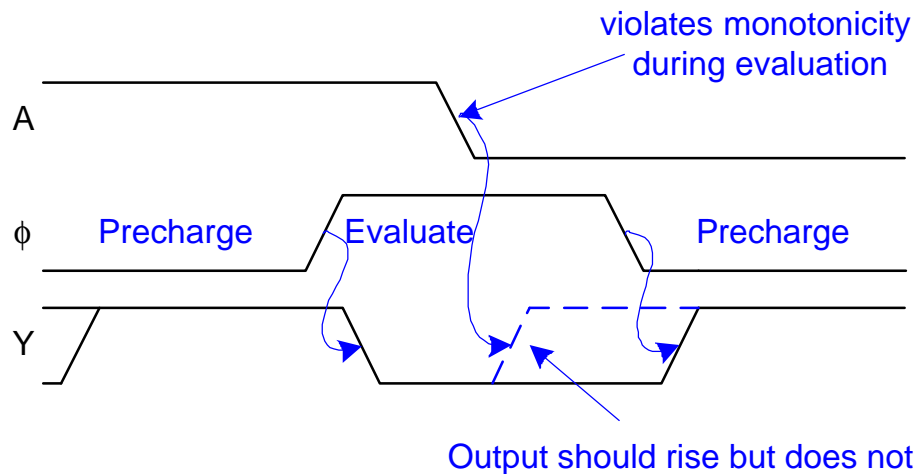
Monotonicity

❑ Dynamic gates require *monotonically rising* inputs during evaluation

  – 0 -> 0

  – 0 -> 1

  – 1 -> 1

  – But not 1 -> 0
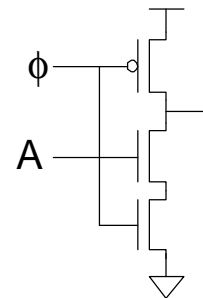
violates monotonicity during evaluation

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!

A = 1

φ   Precharge   Evaluate   Precharge

X

Y

Monotonicity Woes

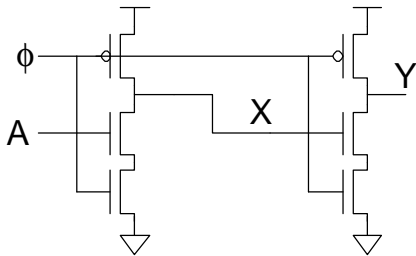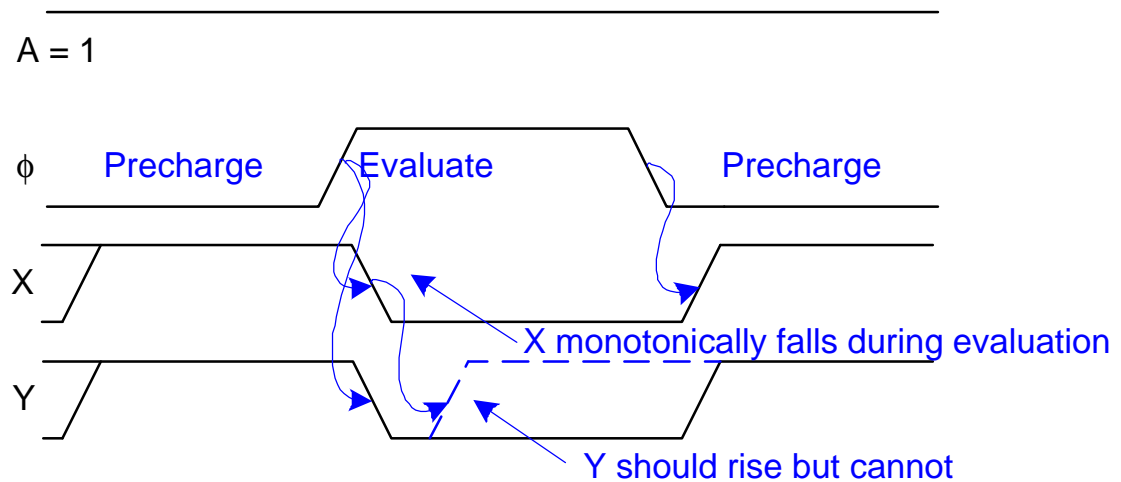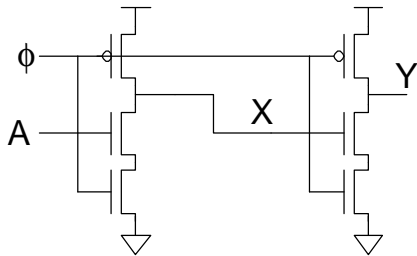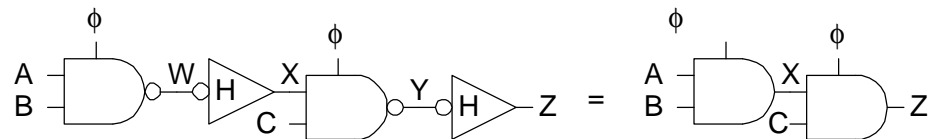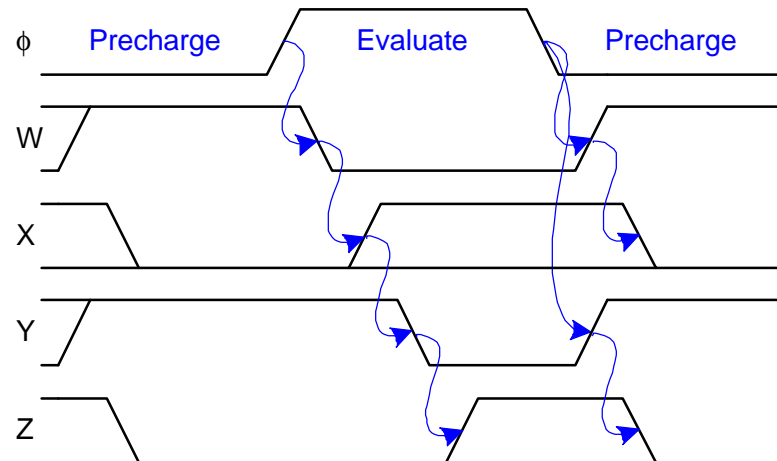❑ But dynamic gates produce monotonically falling outputs during evaluation

❑ Illegal for one dynamic gate to drive another!

A = 1

φ   Precharge   Evaluate   Precharge

X

X monotonically falls during evaluation

Y

Y should rise but cannot
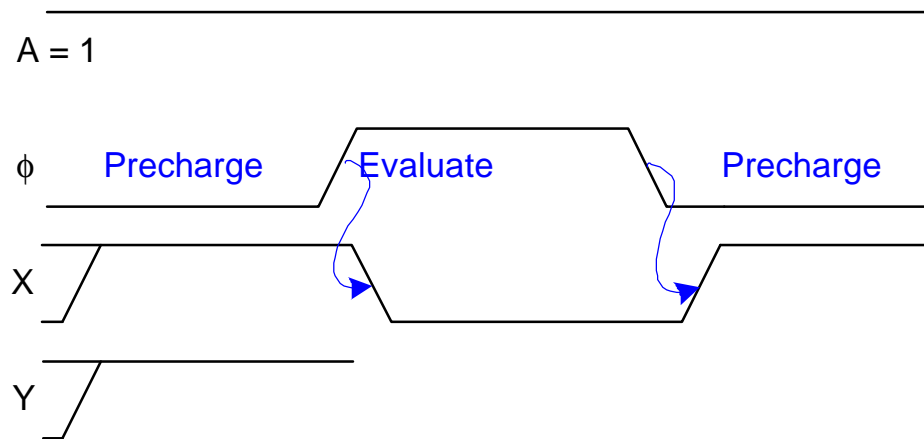
Domino Gates

❑ Follow dynamic stage with inverting static gate
  – Dynamic / static pair is called domino gate
  – Produces monotonic outputs
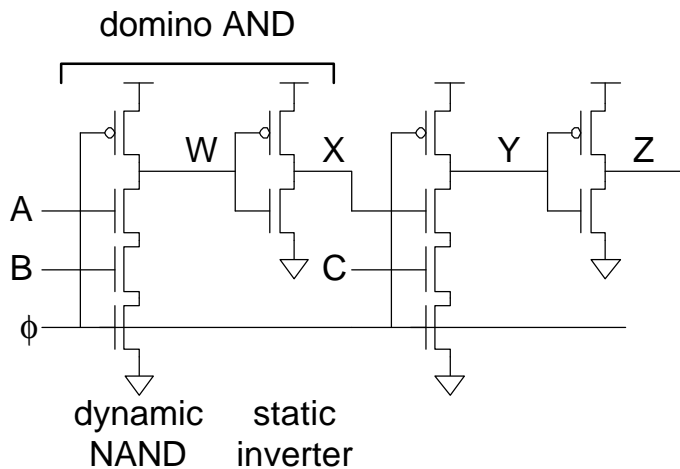
Domino Optimizations

❏ 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

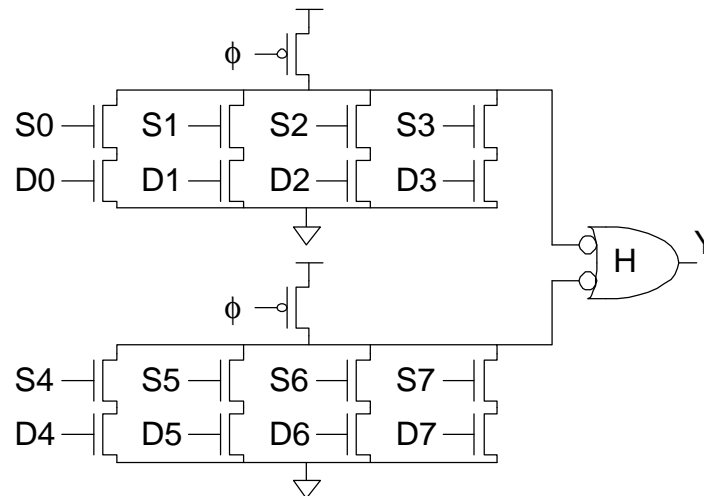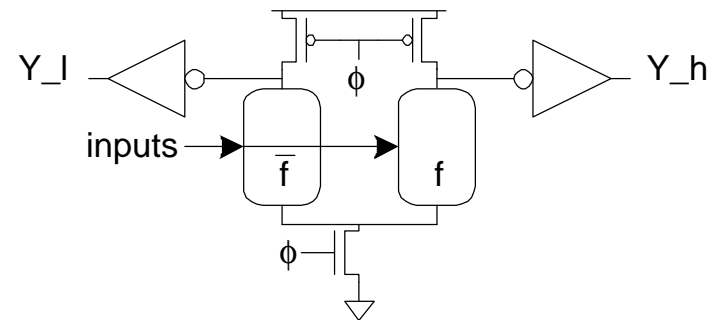❑ Domino only performs noninverting functions:
  – AND, OR but not NAND, NOR, or XOR
❑ Dual-rail domino solves this problem
  – Takes true and complementary inputs
  – Produces true and complementary outputs

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

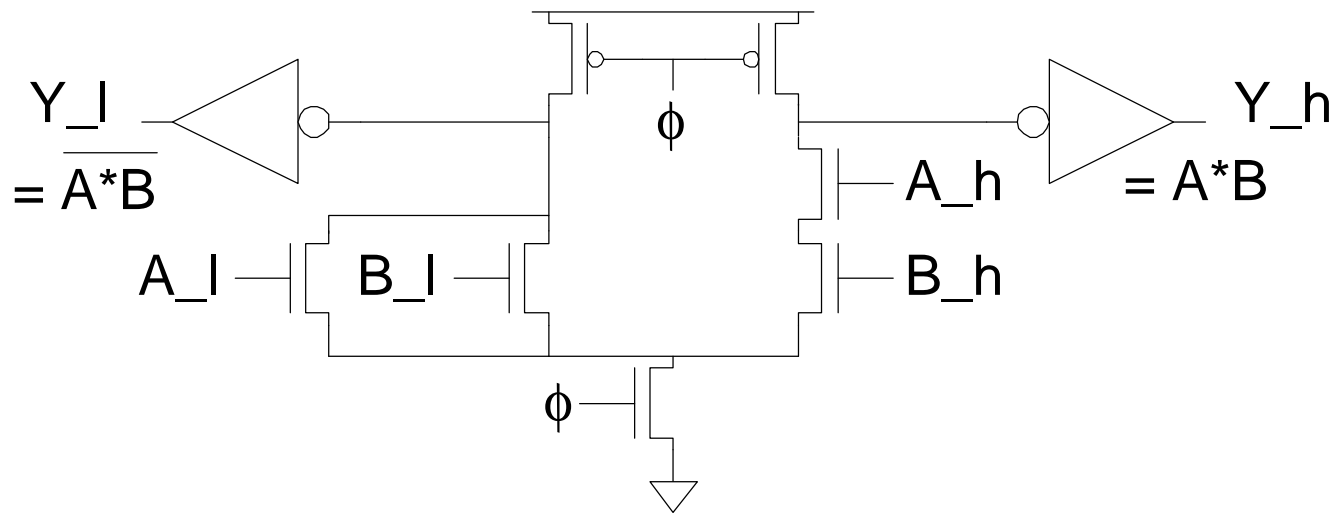# Example: AND/NAND

❑ Given A_h, A_l, B_h, B_l

❑ Compute Y_h = A * B, Y_l = ~(A * B)

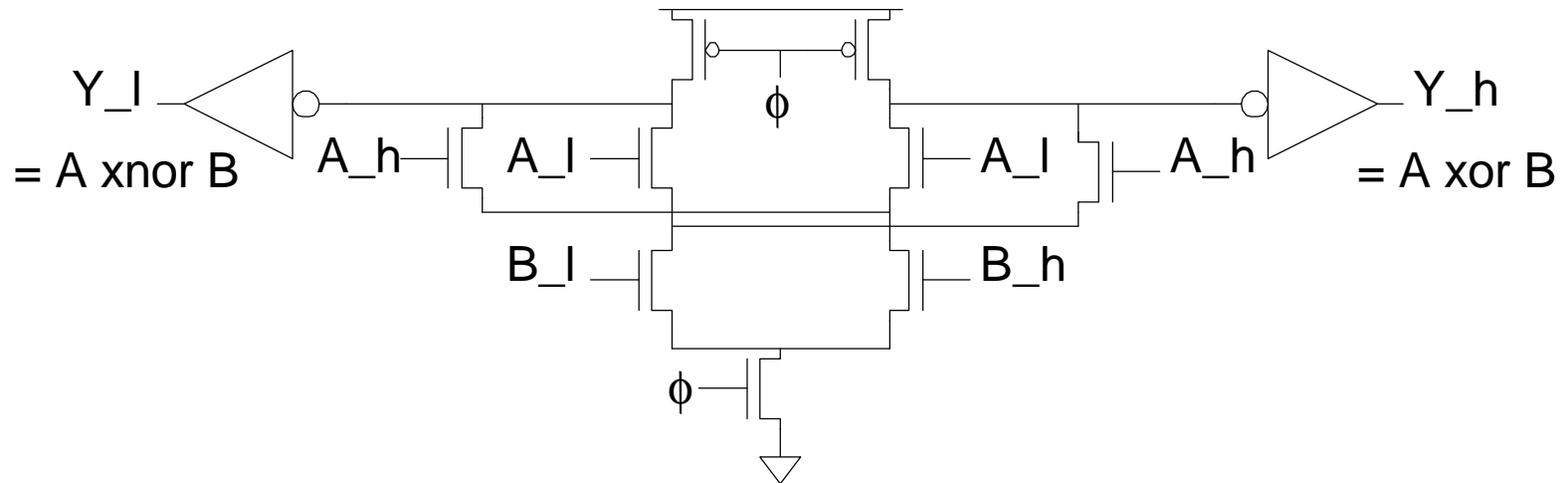# Example: AND/NAND

❑ 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

Y_l
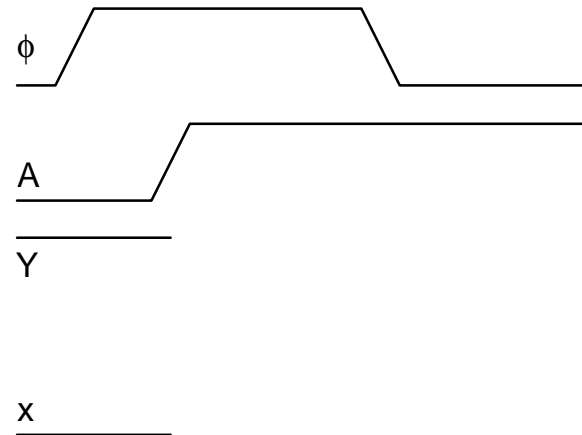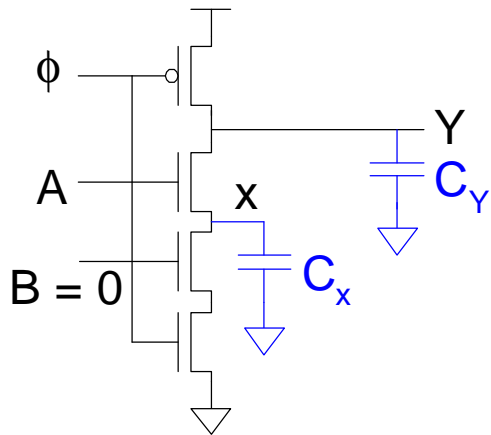= A xnor B   A_h   A_l   φ   A_l   A_h   = A xor B   Y_h

B_l   B_h

φ

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
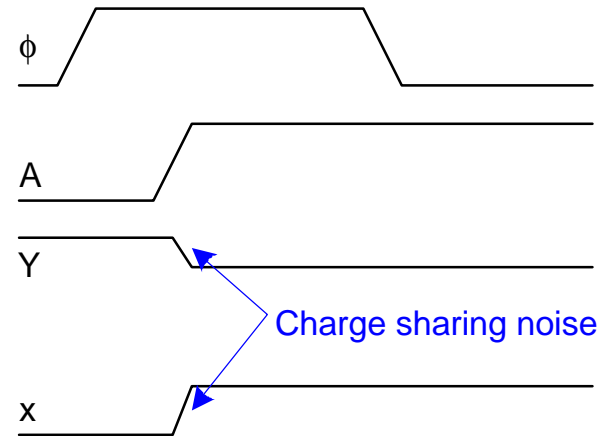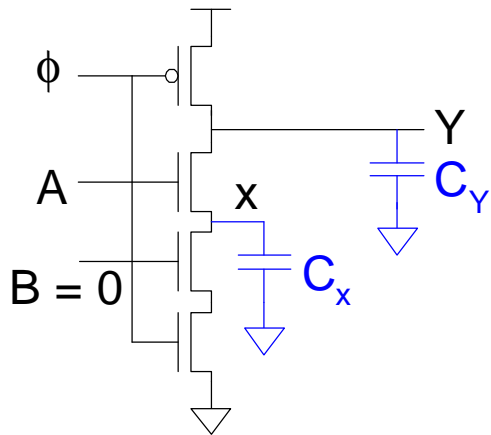
– Must be weak enough not to fight evaluation

weak keeper
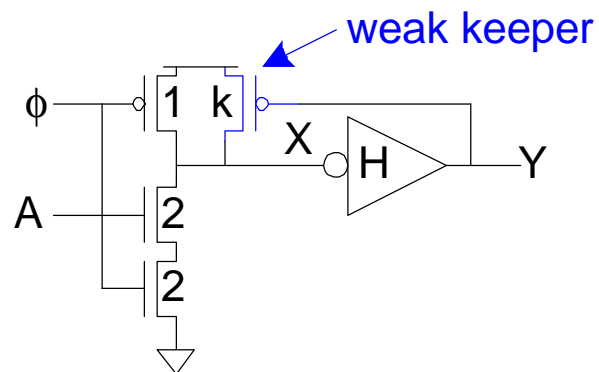
Charge Sharing

❑ Dynamic gates suffer from charge sharing
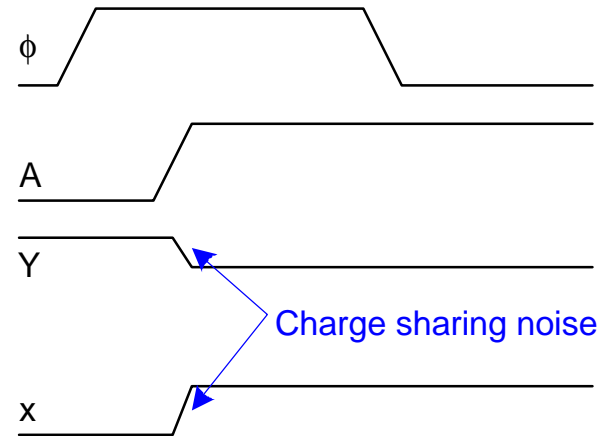
Charge Sharing

❑ Dynamic gates suffer from charge sharing

Charge sharing noise

$$V_x = V_Y =$$

Charge Sharing

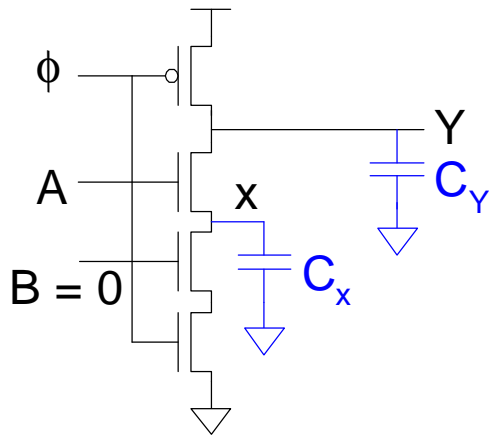❑ Dynamic gates suffer from charge sharing
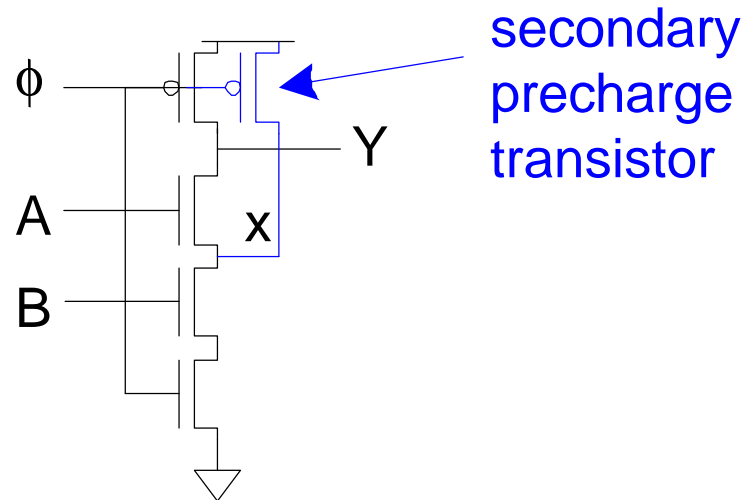

Charge sharing noise

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

Secondary Precharge

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

secondary
precharge
transistor

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

Pass Transistor Circuits
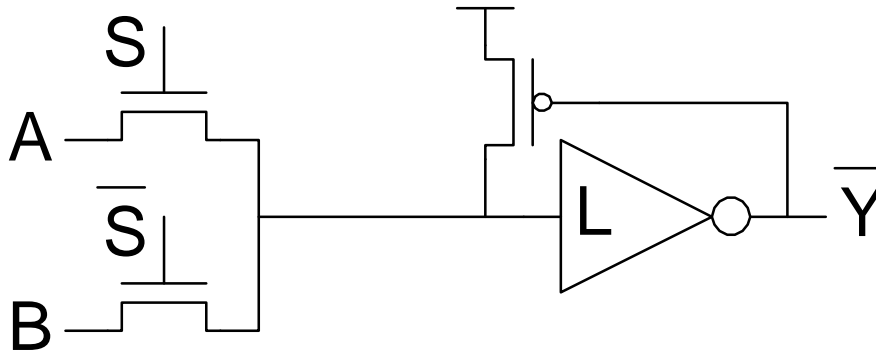
❑ Use pass transistors like switches to do logic

❑ Inputs drive diffusion terminals as well as gates

❑ CMOS + Transmission Gates:

 – 2-input multiplexer

 – Gates should be restoring
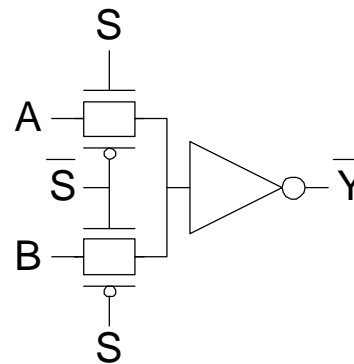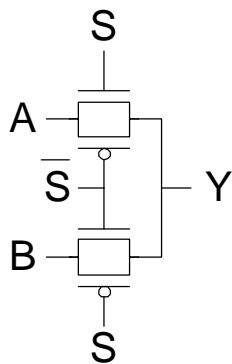
LEAP

❑ **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