Circuit Families

What makes a circuit fast

$I = C\, dV/dt \rightarrow t_{pd} \sim (C/I)\, \Delta V$

Low capacitance

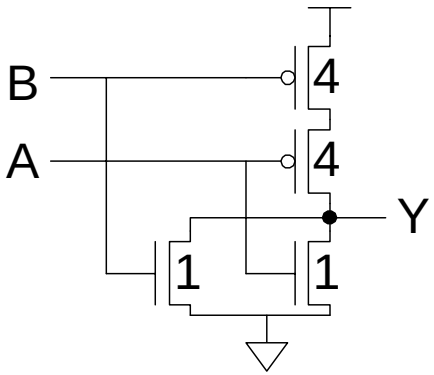
High current

Small swing

Logical effort is proportional to C/I

PMOS transistors cause problems

High capacitance for a given current



Can we take the PMOS capacitance off the input?

Several circuits families have been proposed to replace static CMOS gates to do this

Pseudo NMOS

In old days there were only NMOS transistors in the process

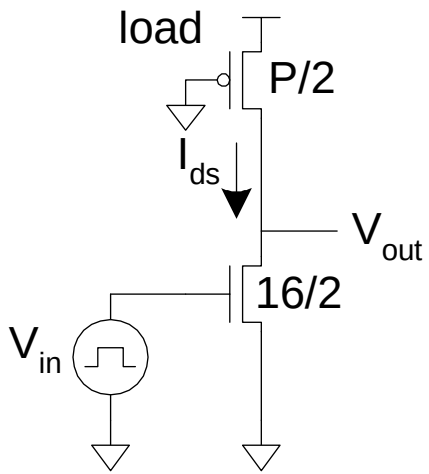
No PMOS can be used to make pseudo-NMOS gates, instead a pull-up transistor is used that is always on

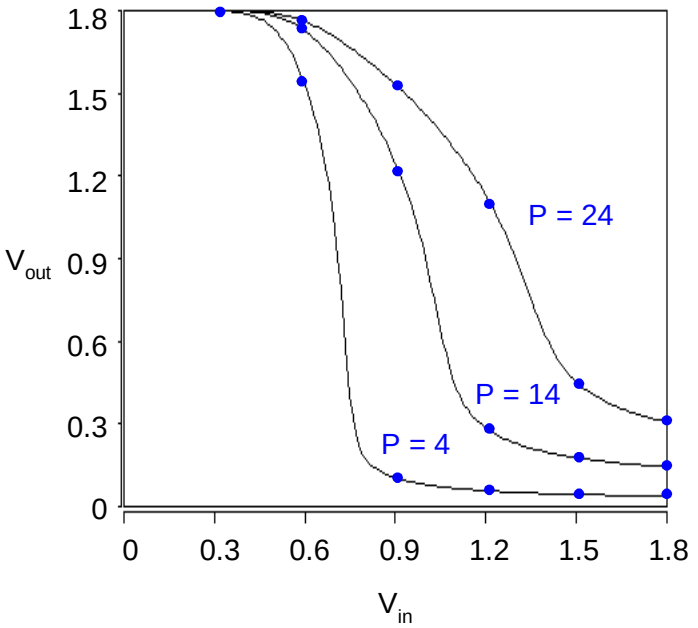
In CMOS, use PMOS that is always ON to make pseudo-NMOS gates

PMOS transistor is always on and thus *fights* the pull-down NMOS network

Ratio of NMOS to PMOS strength very important design parameter

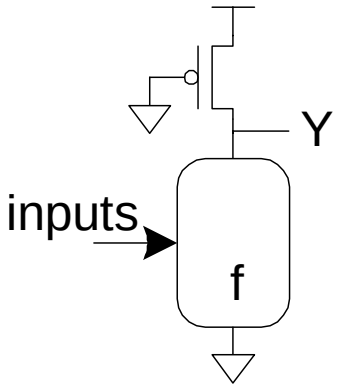
Make PMOS about 1/4 effective strength of pulldown network



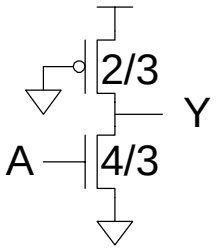


Pseudo NMOS

Design for unit current on output to compare with unit inverter

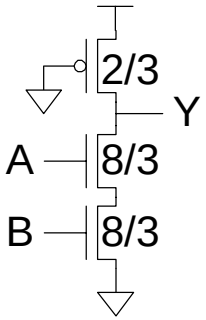


Inverter



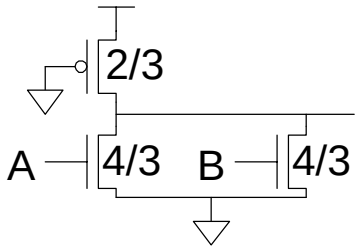
$g_u = 4/3$
$g_d = 4/9$
$g_{avg} = 8/9$
$p_u = 6/3$
$p_d = 6/9$
$p_{avg} = 12/9$

NAND2



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

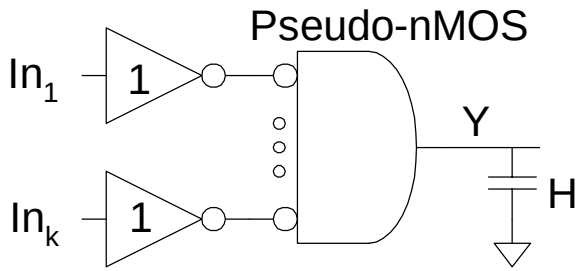
NOR2



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

Pseudo NMOS

Example: Design a k-input AND gate using pseudo NMOS, estimate the delay when driving a fanout of H



$G = 1 * 8/9 = 8/9$

$F = GBH = 8H/9$

$P = 1 + (4 + 8k)/9 = (8k + 13)/9$

$N = 2$

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

Pseudo NMOS

Pseudo NMOS power consumption

Pseudo NMOS draws power whenever Y = 0

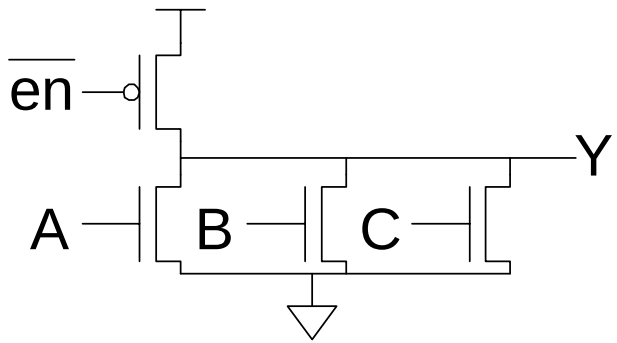
This is static power $P = I * V_{DD}$

A few mA/gate * 1M gates would be a problem

This is why NMOS went extinct !!!

Use pseudo NMOS sparingly for wide NORs

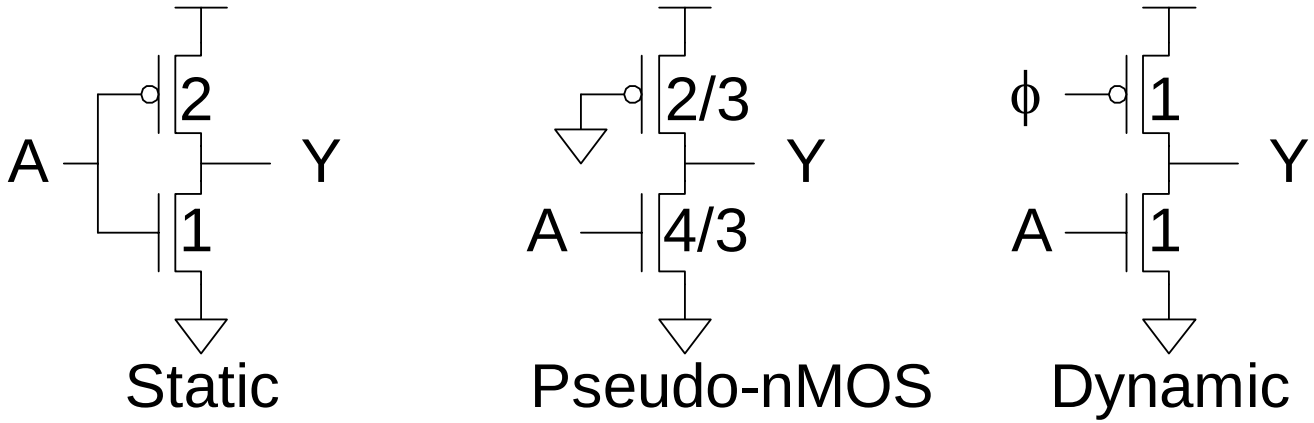
Turn off PMOS when not in use

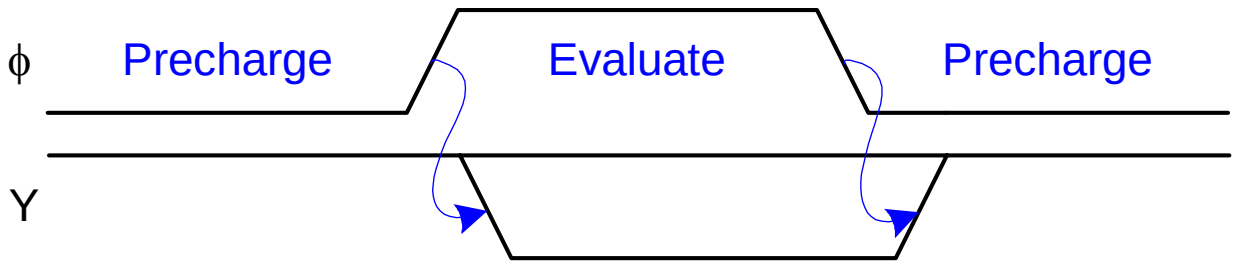


Dynamic Logic

Dynamic gates use a clocked PMOS pullup

Two modes of operation: ***precharge*** and ***evaluate***

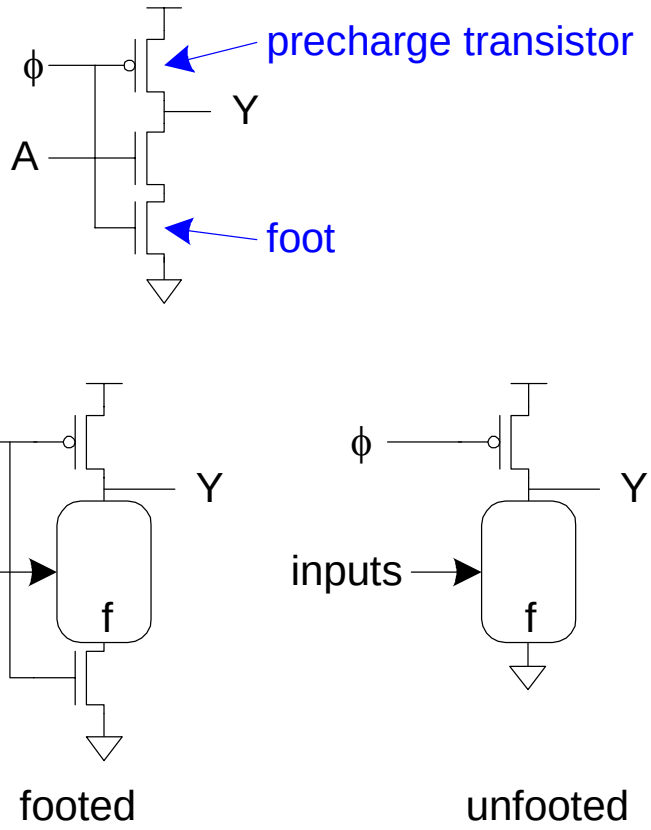




Dynamic Logic

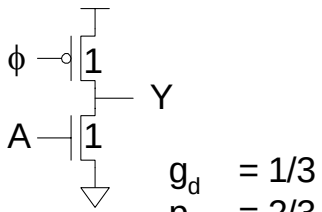
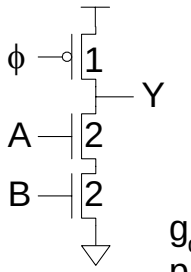
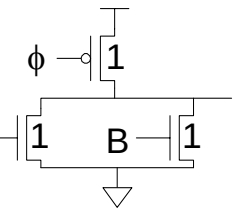
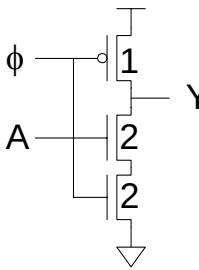
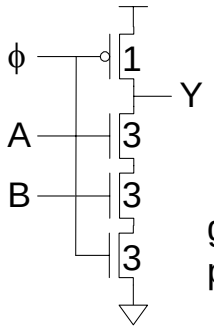
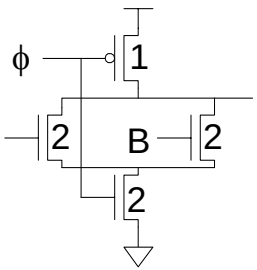
What is the pull down network is ON during the precharge phase?

Use series evaluation transistor to prevent fight



Dynamic Logic

Logical Effort of Dynamic Logic Gates

	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

Monotonicity

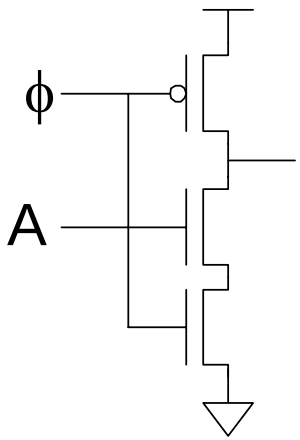
Dynamic gates require monotonically rising inputs during evaluation

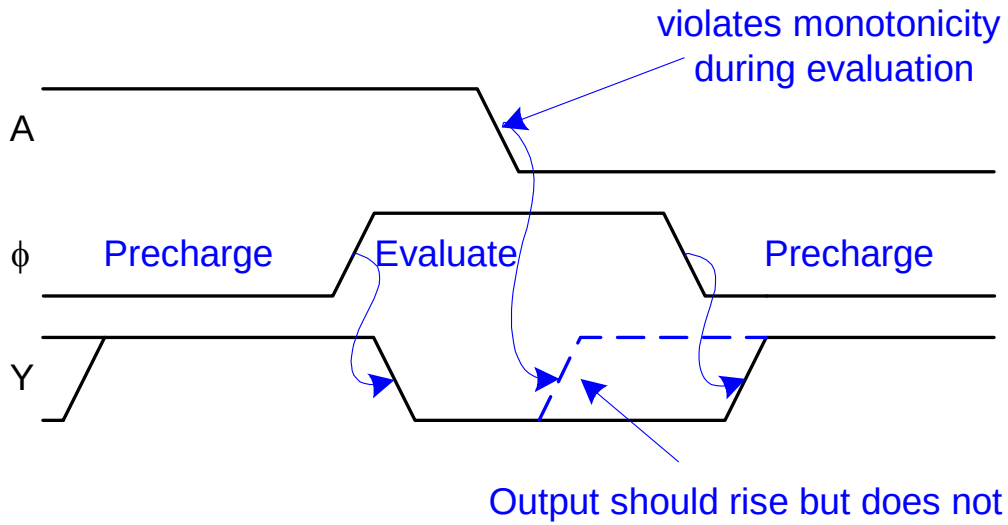
0 --> 0

0 --> 1

1 --> 1

But not 1 --> 0



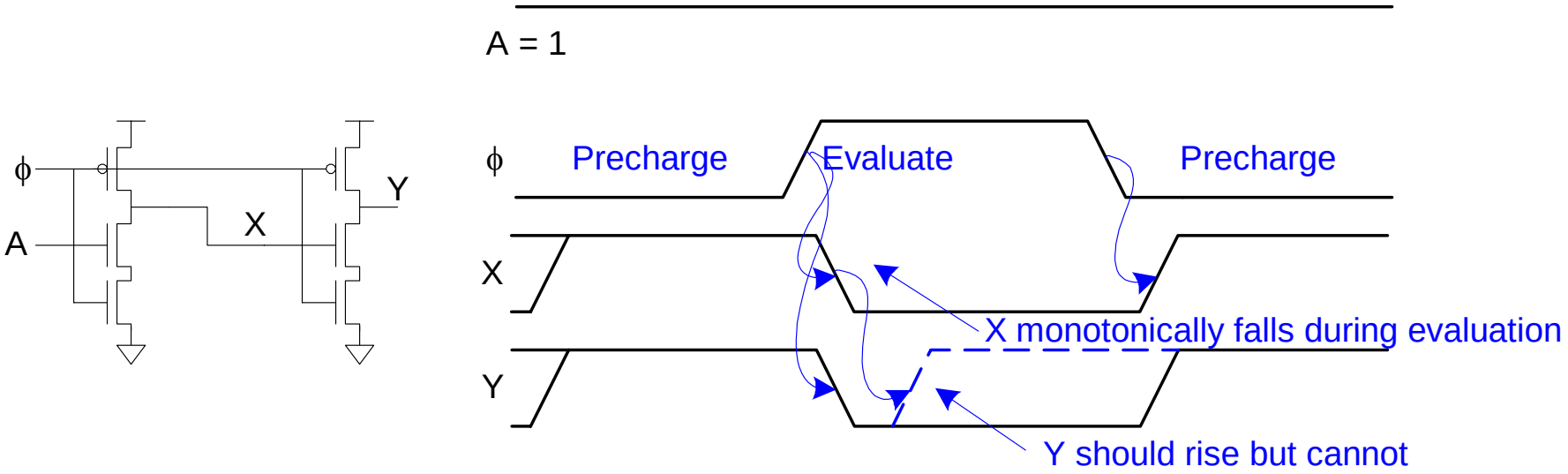


Dynamic Logic

Monotonicity

But dynamic gates produce monotonically falling outputs during evaluation

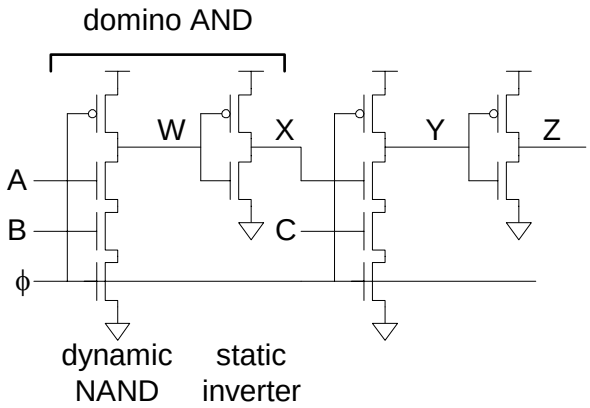
Illegal for one dynamic gate to drive another !!!

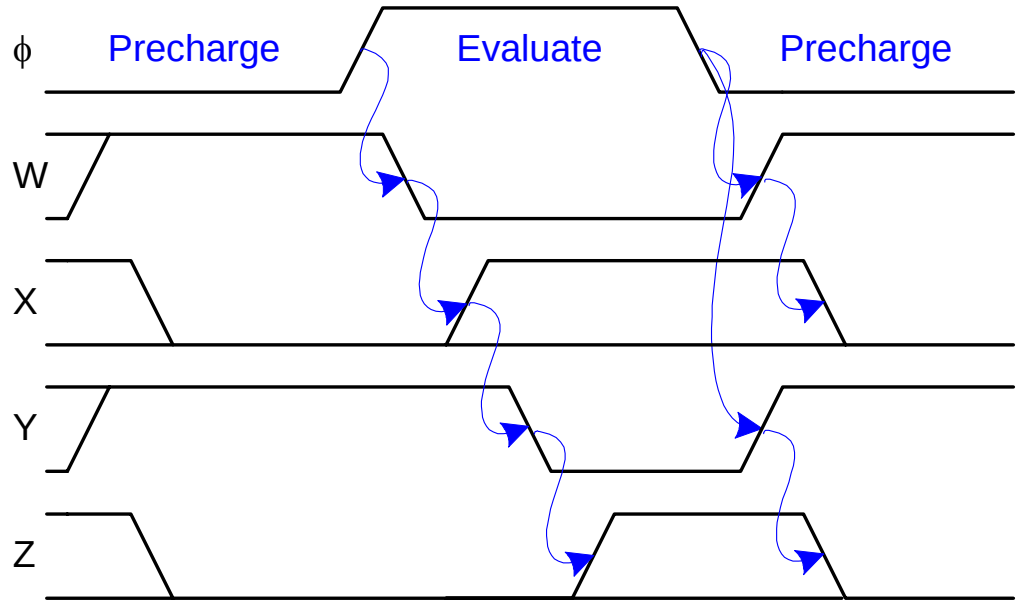


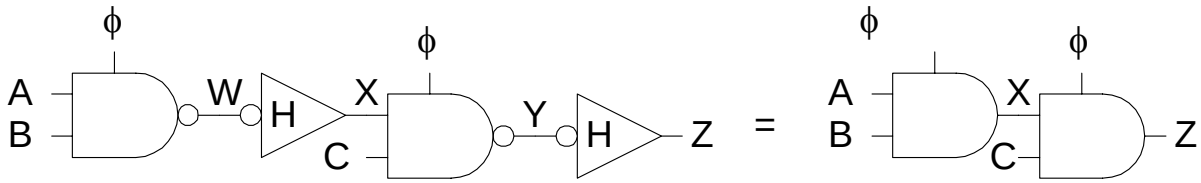
Domino Gates

Follow dynamic stage with inverting static stage

- Dynamic / static pair is called *domino* gate
- Produces monotonic outputs







Domino Gates

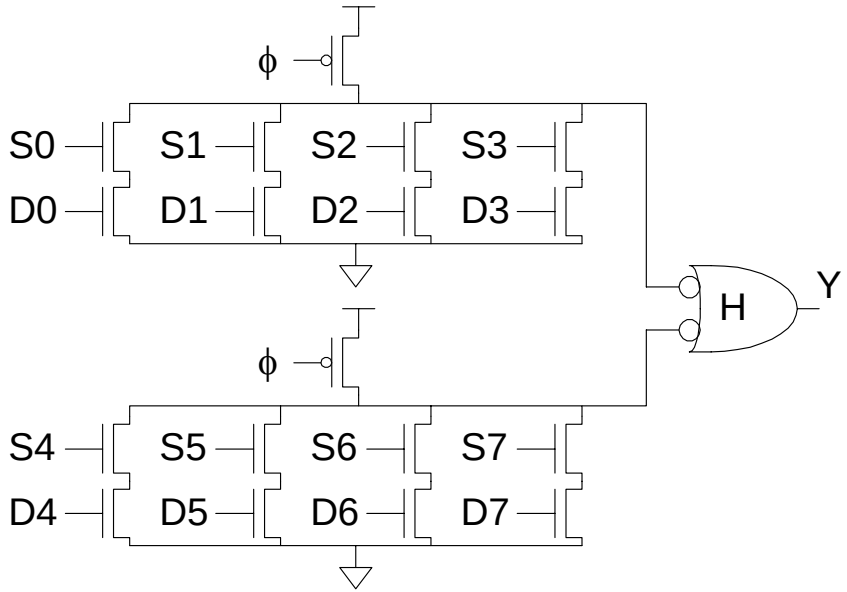
Domino Optimizations

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

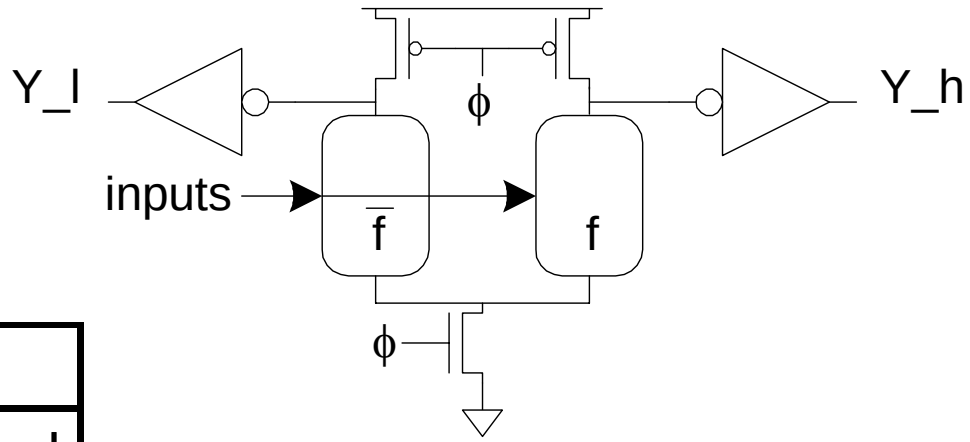
Domino only performs non-inverting functions:

AND, OR but not NAND, NOR, XOR

Dual-rail domino solves this problem

Takes true and complimentary inputs

Produces true and complimentary outputs

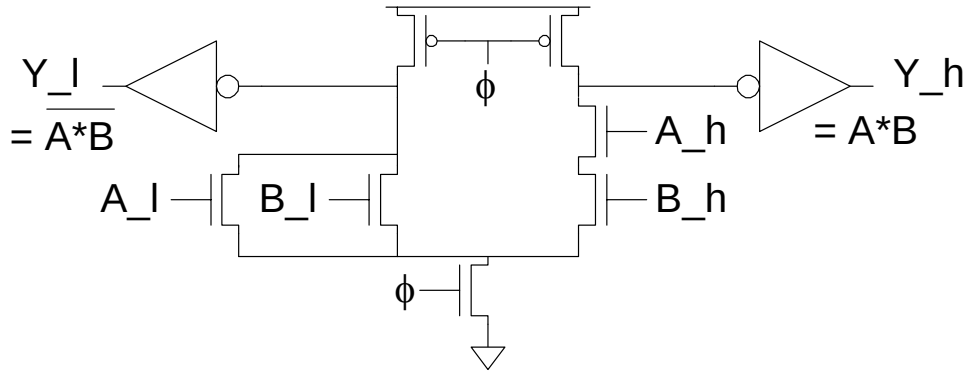


sig_h	sig_l	Meaning
0	0	Precharged
0	1	'0'
1	0	'1'
1	1	invalid

Dual-Rail Domino

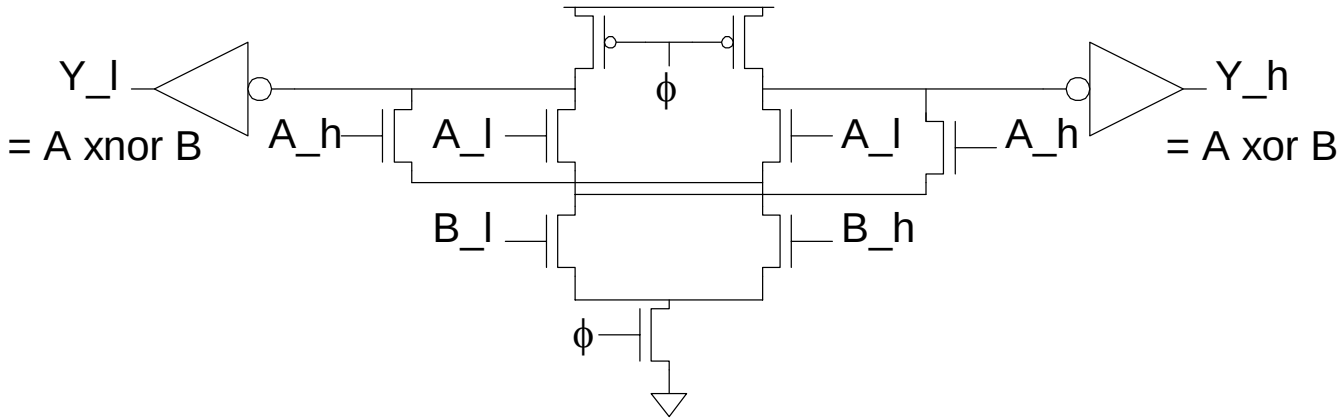
Example: AND/NAND. Given A_l, A_h, B_l, B_h, Compute ***Y_h = A*B, Y_l = ~(A*B)***

Pulldown networks are conduction compliments



Example: XOR/XNOR

Sometimes possible to share transistors, save some of the extra area required



Dynamic Gates Issues

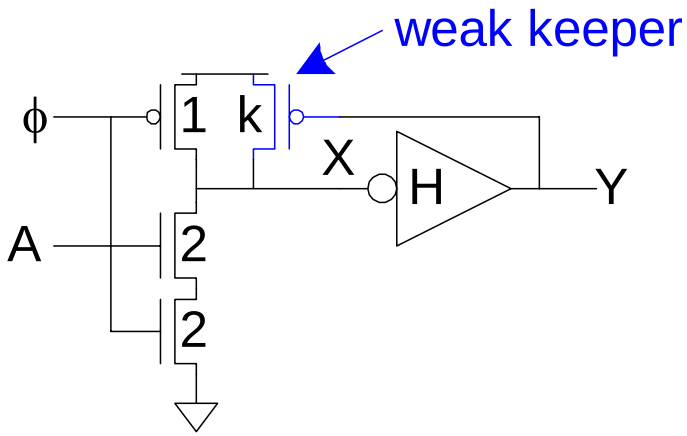
Leakage

Dynamic node floats high during evaluation

- Transistors are leaky (I_{off} is not 0)
- Dynamic value will leak away over time
- Formerly milliseconds, now nanoseconds !!!

Use keeper to hold dynamic node

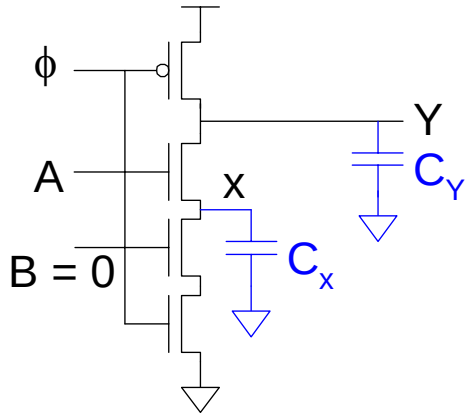
Must be weak enough not to fight evaluation

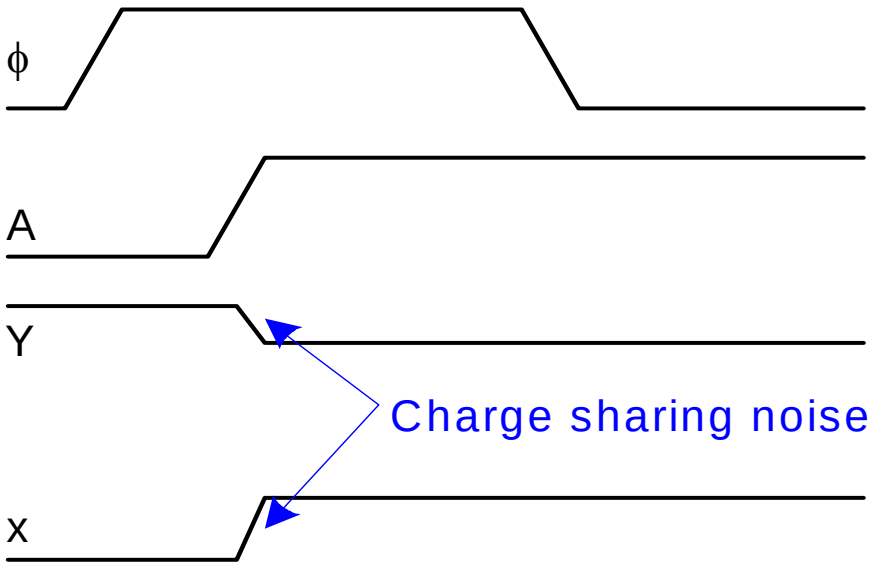


Dynamic Gates Issues

Charge Sharing

Dynamic gates suffer from charge sharing





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

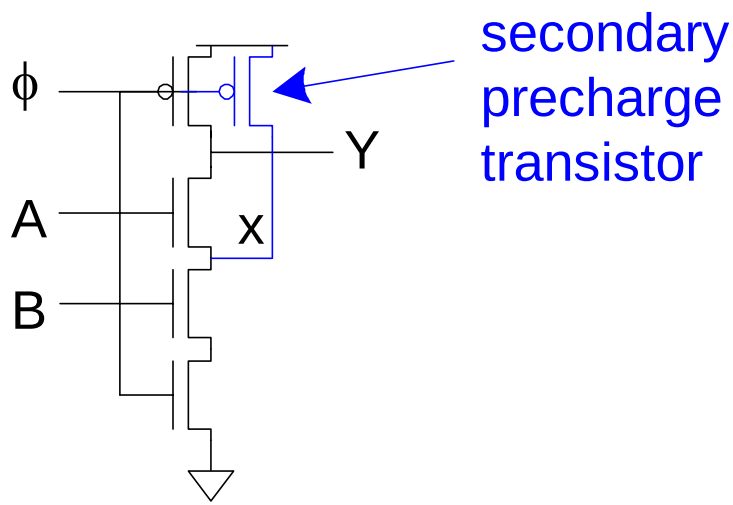
Dynamic Gates Issues

Secondary Precharge

Solution: add secondary precharge transistors

Typically need to precharge every other node

Big load capacitance C_Y helps as well



Dynamic Gates

Dynamic gates are very sensitive to noise

- Inputs: $V_{IH} \sim = V_{tn}$
- Outputs: floating output susceptible noise

Noise sources

- Capacitive crosstalk
- Charge sharing
- Power supply noise
- Feedthrough noise
- And more !!!

Summary

Domino logic is attractive for high-speed circuits

1.5 - 2x faster than static CMOS

But many challenges, discussed before

Widely used in high-performance microprocessors

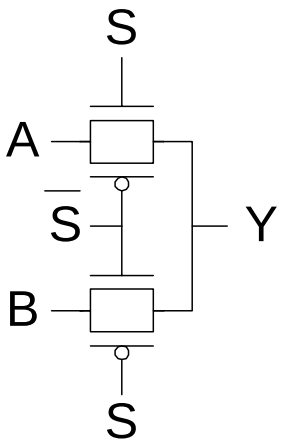
Pass Transistor Circuits

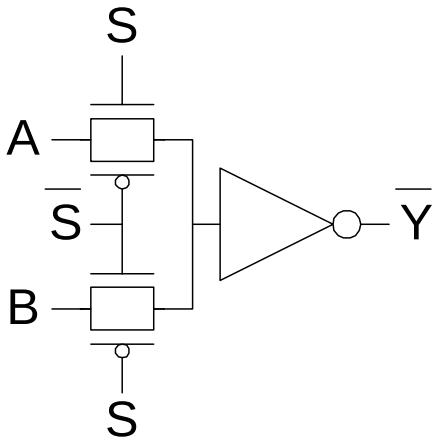
Use pass transistors like switches to do logic

Inputs drive diffusion terminals as well as gates

CMOS + pass transistors

- 2-input multiplexer
- Gates should be restoring

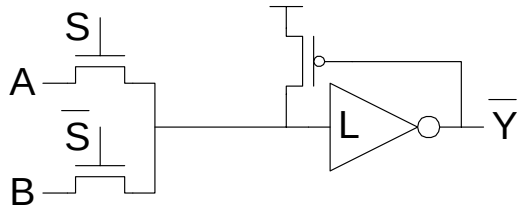




LEAP and CPL

*LEA*n integration with *P*ass transistors

- Get rid of PMOS transistors
- Use weak PMOS feedback to pull fully high, Ratio constraint



*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