EE 447
VLSI Design

Lecture 7:
Combinational Circuits

Outline

- Bubble Pushing
- Compound Gates
- Logical Effort Example
- Input Ordering
- Asymmetric Gates
- Skewed Gates
- Best P/N ratio

Example 1

```
module mux(input  s, d0, d1,
           output y);

   assign y = s ? d1 : d0;
endmodule
```

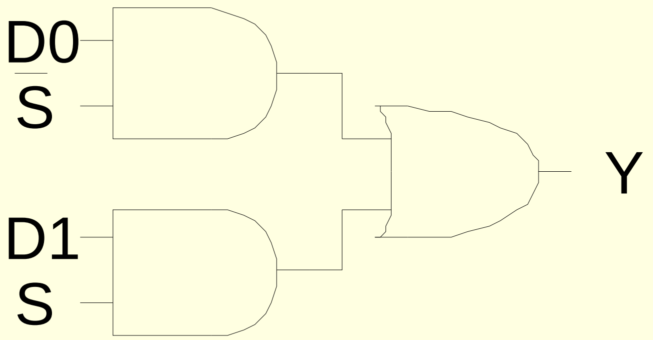
1) Sketch a design using AND, OR, and NOT gates.

Example 1

```
module mux(input  s, d0, d1,
           output y);

   assign y = s ? d1 : d0;
endmodule
```

1) Sketch a design using AND, OR, and NOT gates.

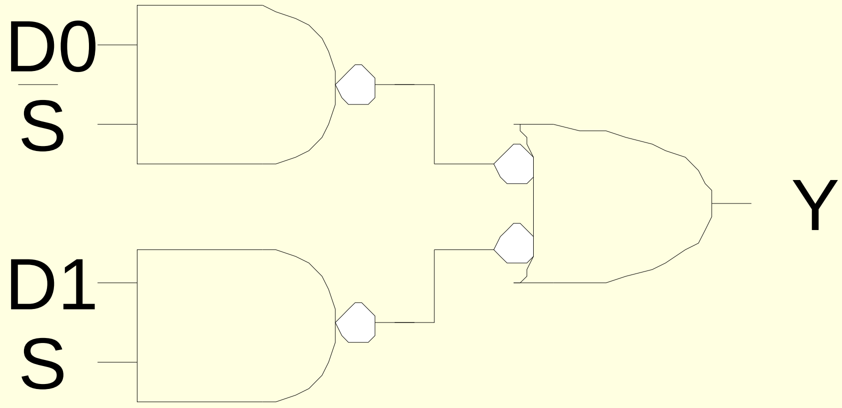


Example 2

2) Sketch a design using NAND, NOR, and NOT gates. Assume ~S is available.

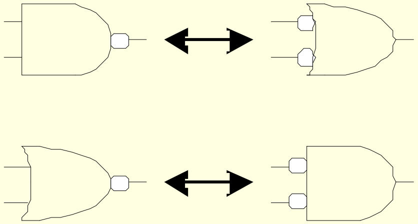
Example 2

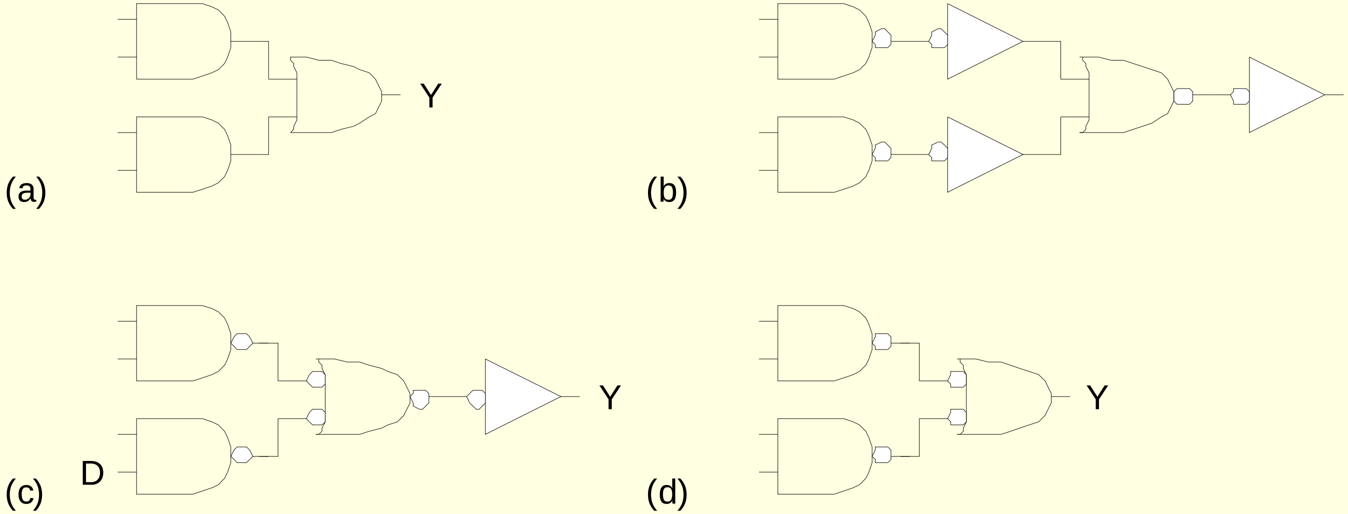
2) Sketch a design using NAND, NOR, and NOT gates. Assume ~S is available.



Bubble Pushing

- Start with network of AND / OR gates
- Convert to NAND / NOR + inverters
- Push bubbles around to simplify logic
  - Remember DeMorgan's Law



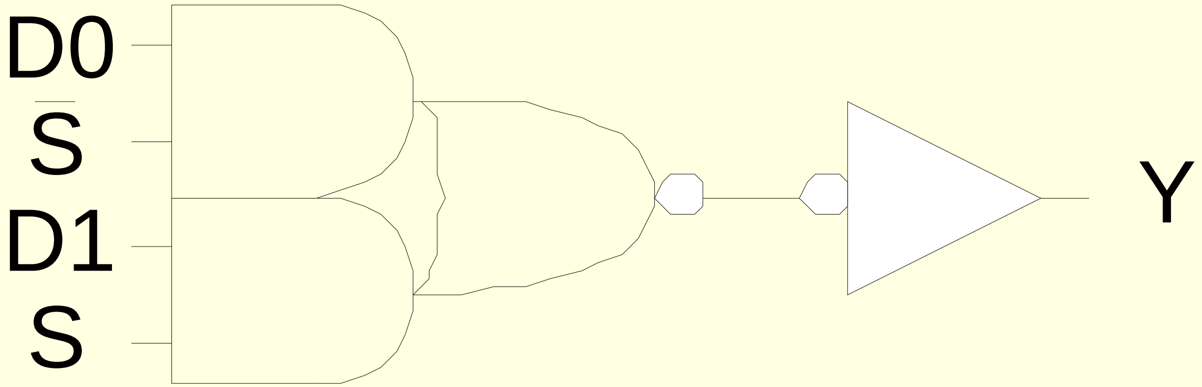


Example 3

3) Sketch a design using one compound gate and one NOT gate. Assume ~S is available.

Example 3

3) Sketch a design using one compound gate and one NOT gate. Assume ~S is available.

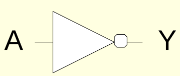


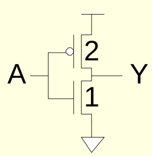
Compound Gates

- Logical Effort of compound gates

unit inverter

$$Y = \overline{A}$$



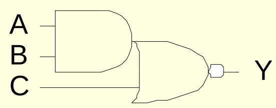


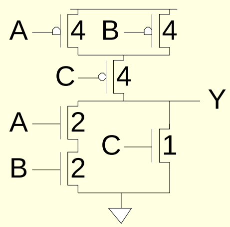
$g_A = 3/3$

$p = 3/3$

AOI21

$$Y = \overline{A \cdot B + C}$$





$g_A = 6/3$

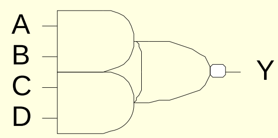
$g_B = 6/3$

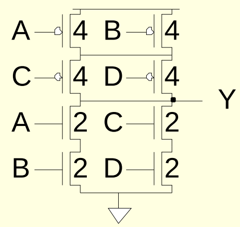
$g_C = 5/3$

$p = 7/3$

AOI22

$$Y = \overline{A \cdot B + C \cdot D}$$





$g_A =$

$g_B =$

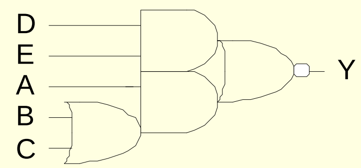
$g_C =$

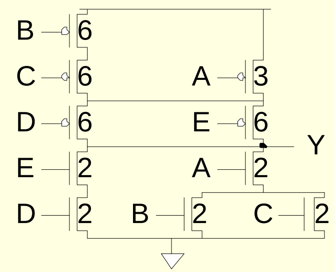
$g_D =$

$p =$

Complex AOI

$$Y = \overline{A \cdot (B + C) + D \cdot E}$$





$g_A =$

$g_B =$

$g_C =$

$g_D =$

$g_E =$

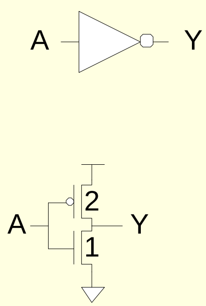
$p =$

Compound Gates

- Logical Effort of compound gates

unit inverter

$$Y = \overline{A}$$

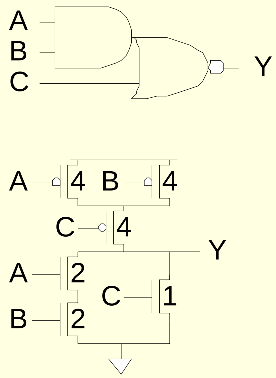


$g_A = 3/3$

$p = 3/3$

AOI21

$$Y = \overline{A \cdot B + C}$$



$g_A = 6/3$

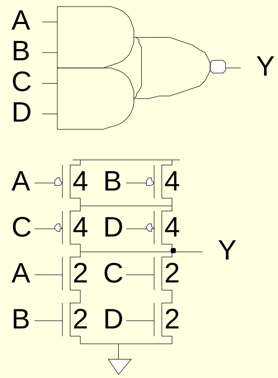
$g_B = 6/3$

$g_C = 5/3$

$p = 7/3$

AOI22

$$Y = \overline{A \cdot B + C \cdot D}$$



$g_A = 6/3$

$g_B = 6/3$

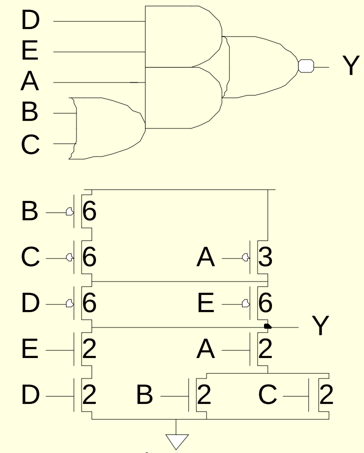
$g_C = 6/3$

$g_D = 6/3$

$p = 12/3$

Complex AOI

$$Y = \overline{A \cdot (B + C) + D \cdot E}$$



$g_A = 5/3$

$g_B = 8/3$

$g_C = 8/3$

$g_D = 8/3$

$g_E = 8/3$

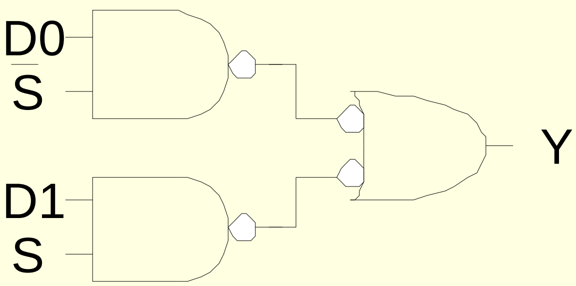
$p = 16/3$

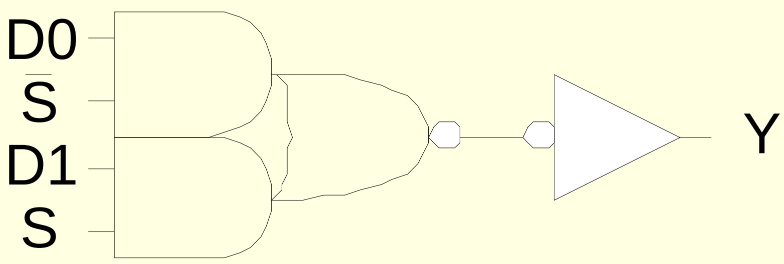
Example 4

- The multiplexer has a maximum input capacitance of 16 units on each input. It must drive a load of 160 units. Estimate the delay of the NAND and compound gate designs.

Example 4

- The multiplexer has a maximum input capacitance of 16 units on each input. It must drive a load of 160 units. Estimate the delay of the NAND and compound gate designs.



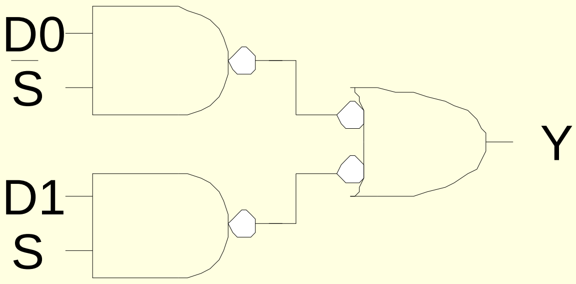


H = 160 / 16 = 10

B = 1

N = 2

NAND Solution



NAND Solution

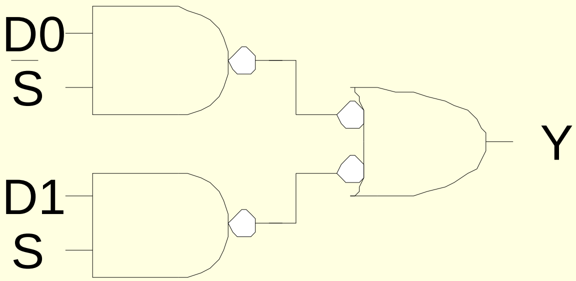
$$P = 2 + 2 = 4$$

$$G = (4/3)\cdot(4/3) = 16/9$$

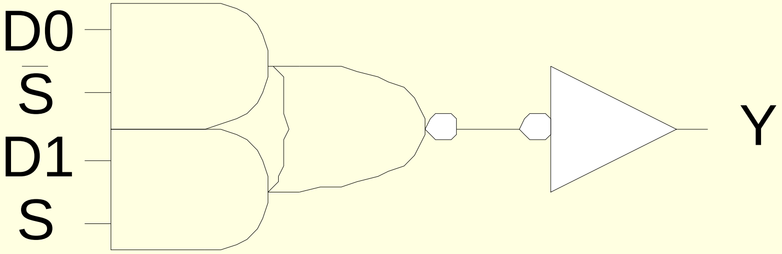
$$F = GBH = 160/9$$

$$\hat{f} = \sqrt[N]{F} = 4.2$$

$$D = N\hat{f} + P = 12.4\tau$$



Compound Solution



Compound Solution

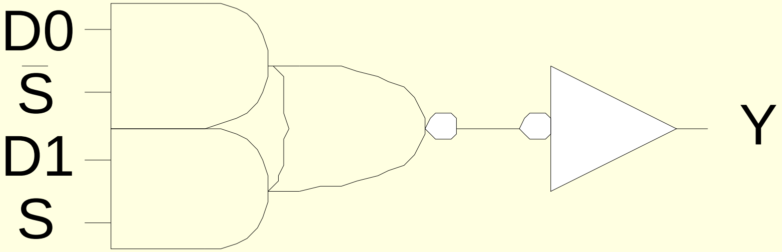
$$P = 4 + 1 = 5$$

$$G = (6/3)g(1) = 2$$

$$F = GBH = 20$$

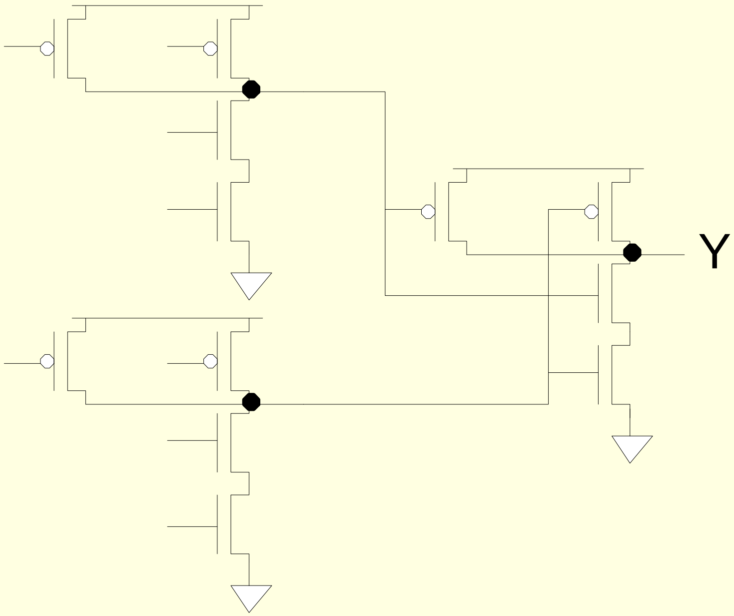
$$\hat{f} = \sqrt[N]{F} = 4.5$$

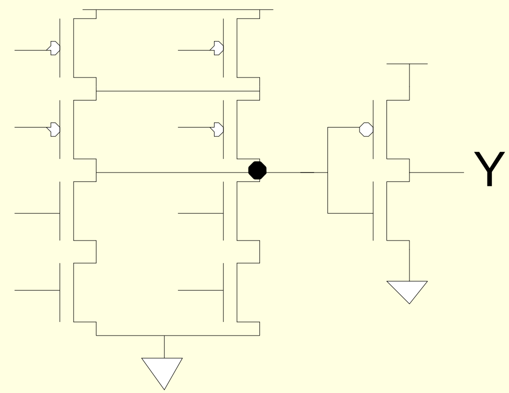
$$D = N\hat{f} + P = 14\tau$$



Example 5

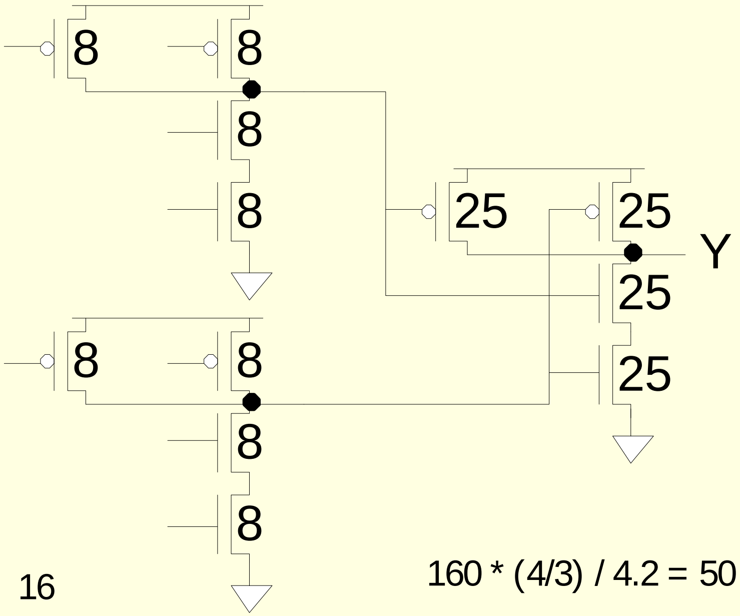
- Annotate your designs with transistor sizes that achieve this delay.



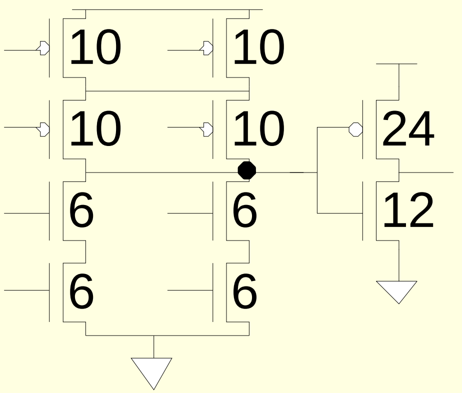


Example 5

- Annotate your designs with transistor sizes that achieve this delay.



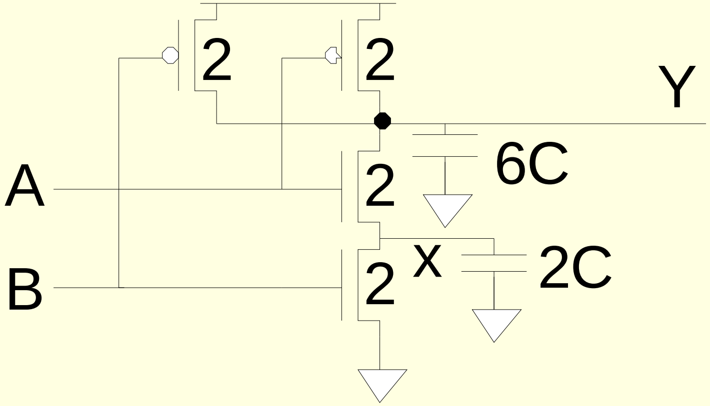
16 160 * (4/3) / 4.2 = 50



16 160 * 1 / 4.5 = 36

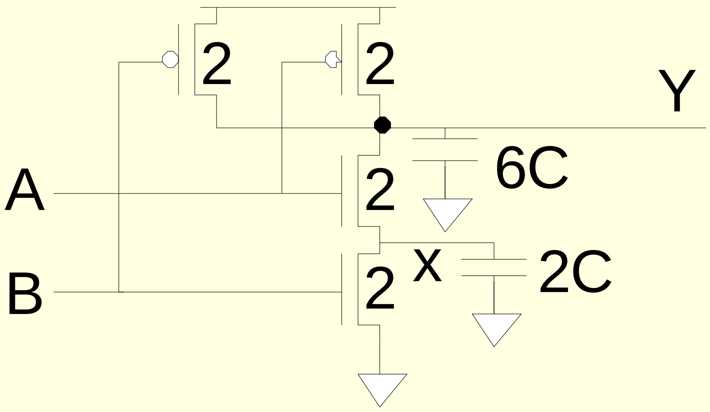
Input Order

- Our parasitic delay model was too simple
  - Calculate parasitic delay for Y falling
    - If A arrives latest?
    - If B arrives latest?



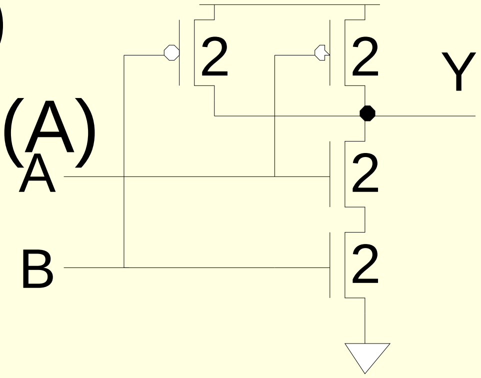
Input Order

- Our parasitic delay model was too simple
  - Calculate parasitic delay for Y falling
    - If A arrives latest? 2τ
    - If B arrives latest? 2.33τ



Inner & Outer Inputs

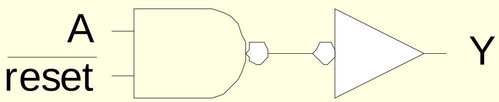
- *Outer* input is closest to rail (B)
- *Inner* input is closest to output (A)



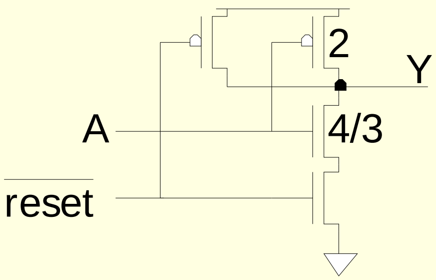
- If input arrival time is known
  - Connect latest input to inner terminal

Asymmetric Gates

- Asymmetric gates favor one input over another
- Ex: suppose input A of a NAND gate is most critical
  - Use smaller transistor on A (less capacitance)
  - Boost size of noncritical input
  - So total resistance is same



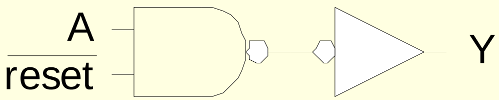
- $g_A =$
- $g_B =$
- $g_{total} = g_A + g_B =$

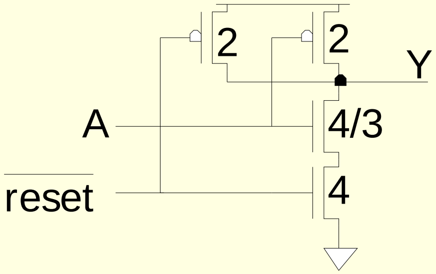


- Asymmetric gate approaches g = 1 on critical input
- But total logical effort goes up

Asymmetric Gates

- Asymmetric gates favor one input over another
- Ex: suppose input A of a NAND gate is most critical
  - Use smaller transistor on A (less capacitance)
  - Boost size of noncritical input
  - So total resistance is same

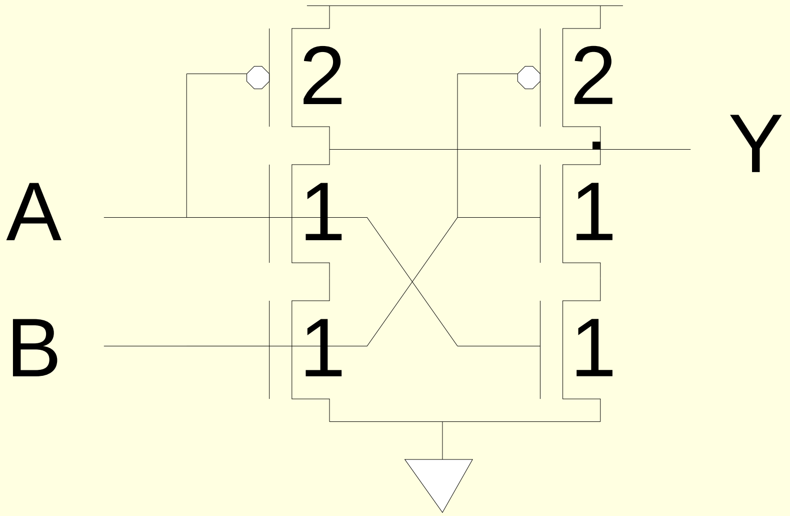




- $g_A = 10/9$
- $g_B = 2$
- $g_{total} = g_A + g_B = 28/9$
- Asymmetric gate approaches g = 1 on critical input
- But total logical effort goes up

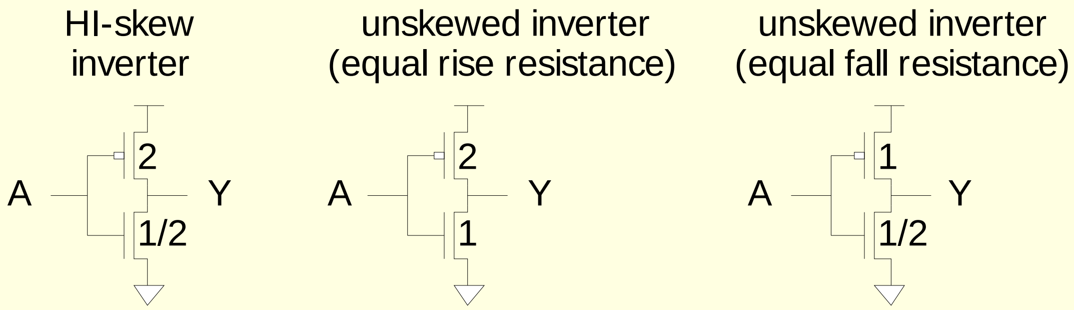
Symmetric Gates

- Inputs can be made perfectly symmetric



Skewed Gates

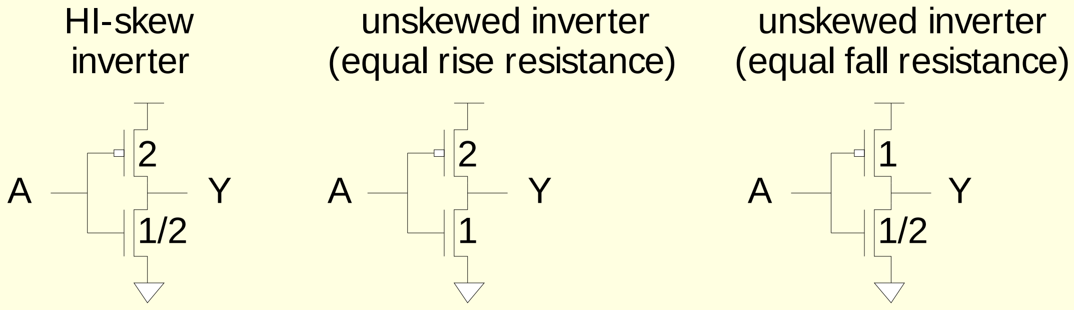
- Skewed gates favor one edge over another
- Ex: suppose rising output of inverter is most critical
  - Downsize noncritical nMOS transistor



- Calculate logical effort by comparing to unskewed inverter with same effective resistance on that edge.
  - g_u =
  - g_d =

Skewed Gates

- Skewed gates favor one edge over another
- Ex: suppose rising output of inverter is most critical
  - Downsize noncritical nMOS transistor



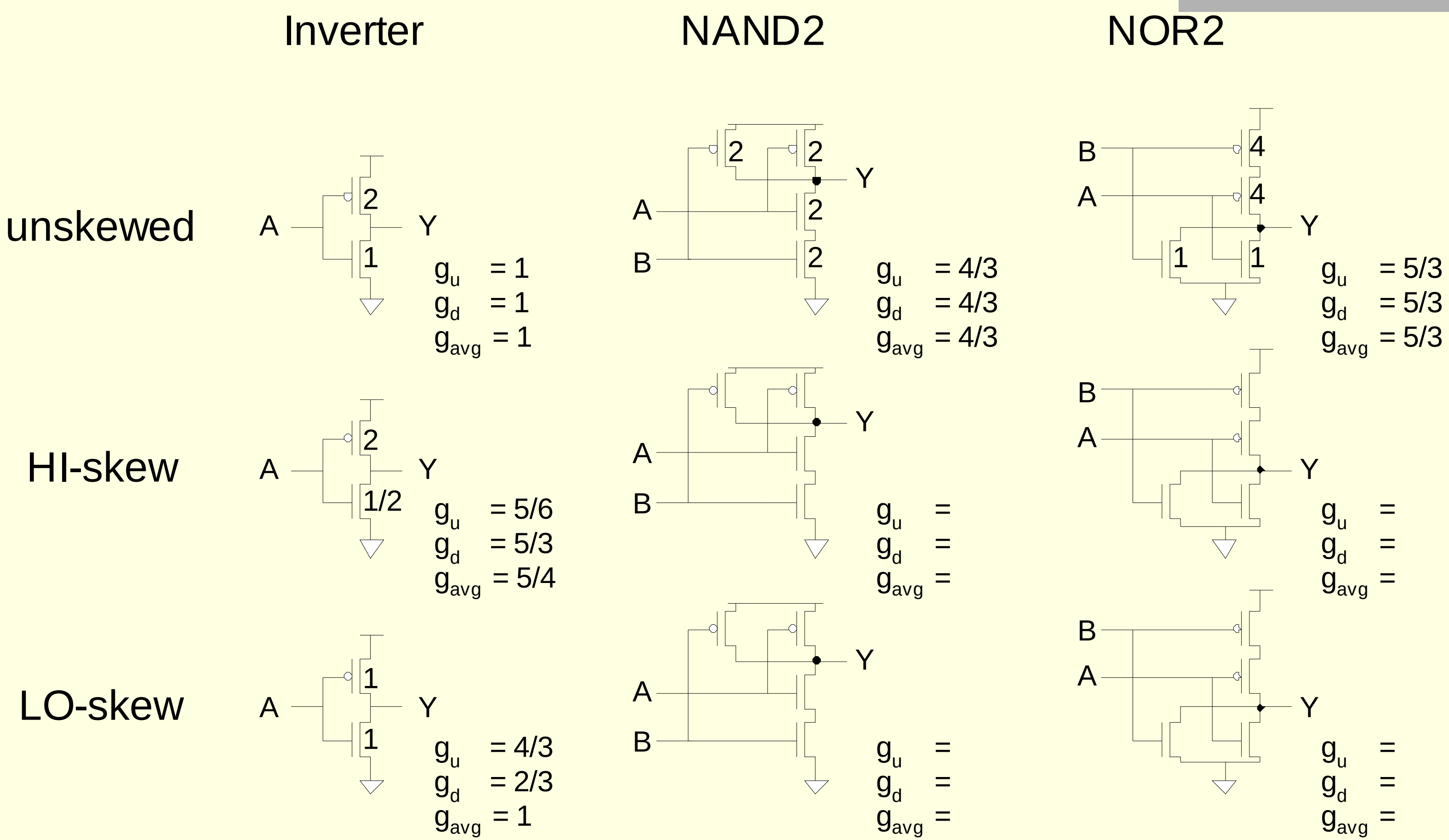
- Calculate logical effort by comparing to unskewed inverter with same effective resistance on that edge.
  - $g_u = 2.5 / 3 = 5/6$
  - $g_d = 2.5 / 1.5 = 5/3$

HI- and LO-Skew

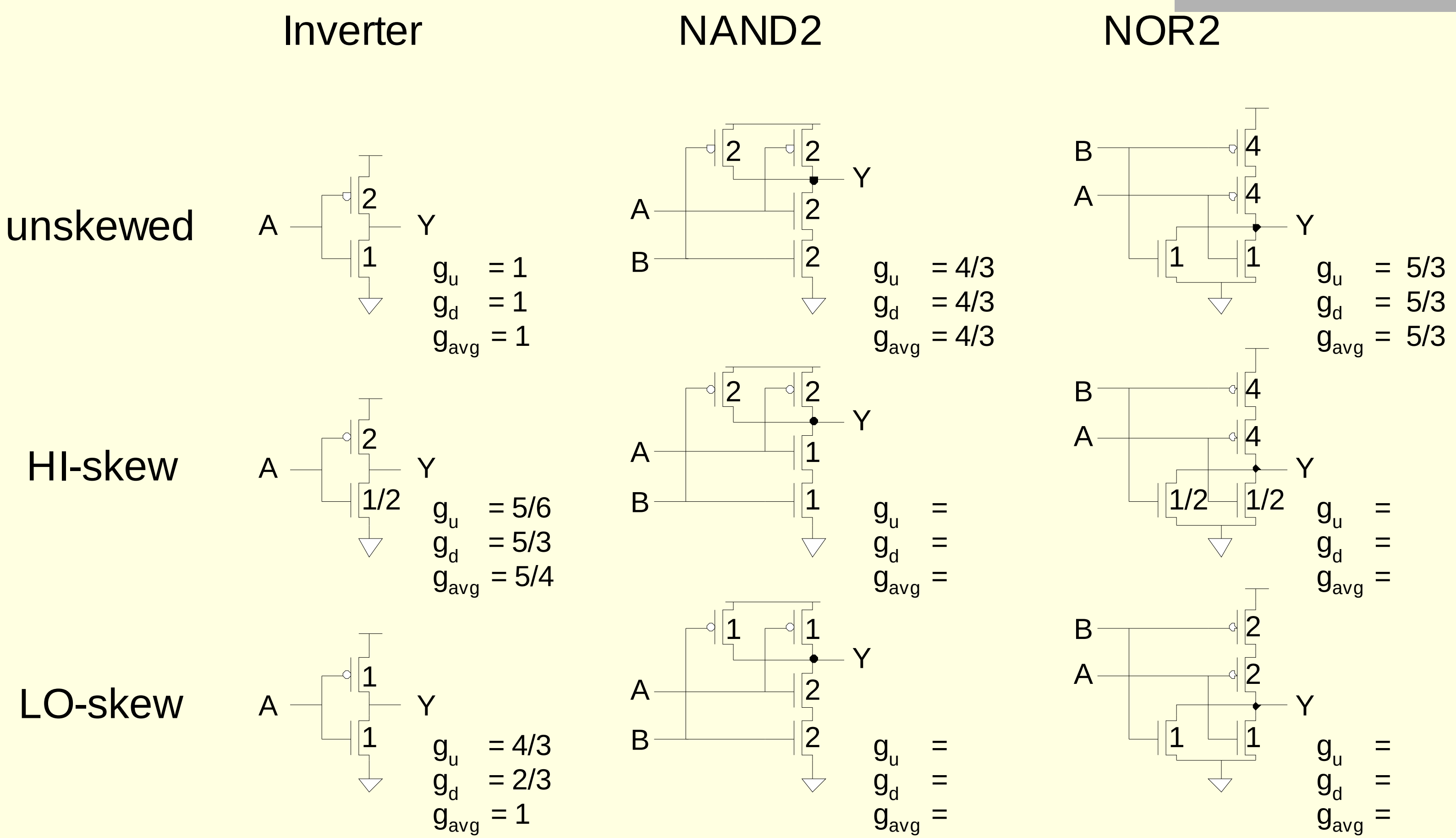
- Def: Logical effort of a skewed gate for a particular transition is the ratio of the input capacitance of that gate to the input capacitance of an unskewed inverter delivering the same output current for the same transition.

- Skewed gates reduce size of noncritical transistors
  - HI-skew gates favor rising output (small nMOS)
  - LO-skew gates favor falling output (small pMOS)
- Logical effort is smaller for favored direction
- But larger for the other direction

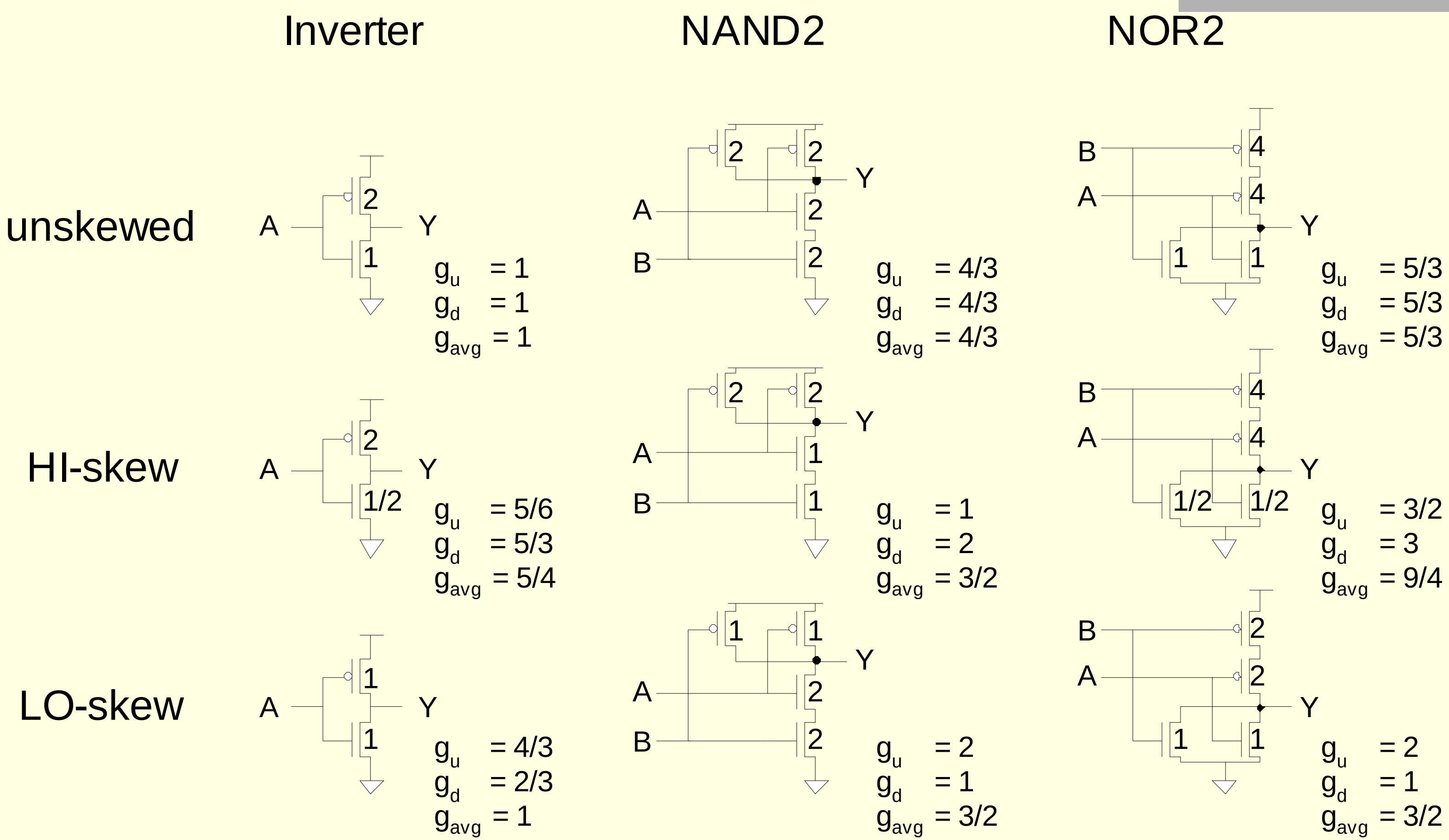
Catalog of Skewed Gates



Catalog of Skewed Gates

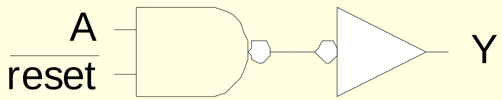


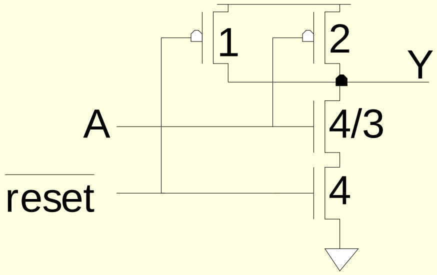
Catalog of Skewed Gates



Asymmetric Skew

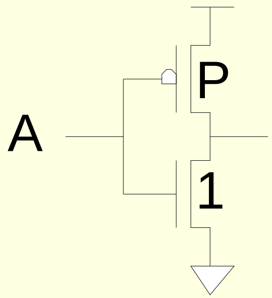
- Combine asymmetric and skewed gates
  - Downsize noncritical transistor on unimportant input
  - Reduces parasitic delay for critical input





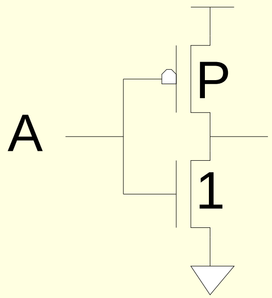
Best P/N Ratio

- We have selected P/N ratio for unit rise and fall resistance (μ = 2-3 for an inverter).
- Alternative: choose ratio for least average delay
- Ex: inverter
  - Delay driving identical inverter
  - t_{pdf} =
  - t_{pdr} =
  - t_{pd} =
  - Differentiate t_{pd} w.r.t. P
  - Least delay for P =



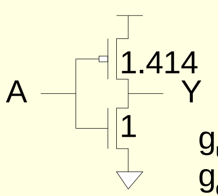
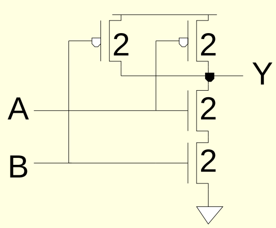
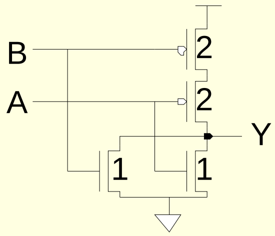
Best P/N Ratio

- We have selected P/N ratio for unit rise and fall resistance (μ = 2-3 for an inverter).
- Alternative: choose ratio for least average delay
- Ex: inverter
  - Delay driving identical inverter
  - $t_{pdf} = (P+1)$
  - $t_{pdr} = (P+1)(\mu/P)$
  - $t_{pd} = (P+1)(1+\mu/P)/2 = (P + 1 + \mu + \mu/P)/2$
  - Differentiate t_{pd} w.r.t. P
  - Least delay for P = $\sqrt{\mu}$



P/N Ratios

- In general, best P/N ratio is sqrt of equal delay ratio.
  - Only improves average delay slightly for inverters
  - But significantly decreases area and power

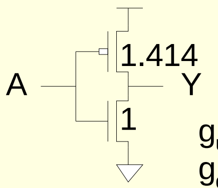
	Inverter	NAND2	NOR2
fastest P/N ratio			
	g_u = g_d = g_{avg} =	g_u = g_d = g_{avg} =	g_u = g_d = g_{avg} =

P/N Ratios

- In general, best P/N ratio is sqrt of that giving equal delay.
  - Only improves average delay slightly for inverters
  - But significantly decreases area and power

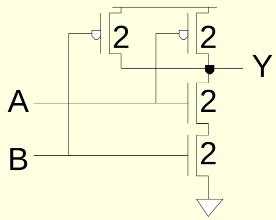
fastest P/N ratio

Inverter



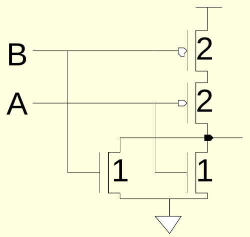
$g_u = 1.15$
$g_d = 0.81$
$g_{avg} = 0.98$

NAND2



$g_u = 4/3$
$g_d = 4/3$
$g_{avg} = 4/3$

NOR2



$g_u = 2$
$g_d = 1$
$g_{avg} = 3/2$

Observations

- For speed:
  - NAND vs. NOR
  - Many simple stages vs. fewer high fan-in stages
  - Latest-arriving input
- For area and power:
  - Many simple stages vs. fewer high fan-in stages