Digital Integrated Circuits

Designing Combinational Logic Circuits

Fuyuzhuo

Static CMOS logic

- CMOS static characteristic
- CMOS propagate delay
- Large fan-in technology
- ***Logic effort***
- CMOS power analysis

Sizing Logic Paths for Speed

- Example:

  ALU load in an Intel's microprocessor is 0.5pF
- How to size the ALU datapath to achieve maximum speed?
- We have already solved this for the inverter chain

Can we generalize it for any type of logic?

Modified formula

$$t_p = t_{par} + t_{ext} = \ln 2.(C_{par} + C_{ext})R_{par}$$

$$= \ln 2.R_{inv}C_{int}\left(\frac{C_{par}}{C_{int}} + \frac{C_g}{C_{int}}\frac{C_{ext}}{C_g}\right)$$

$$= t_{p0}\left(\frac{C_{par}}{C_{int}} + \frac{C_g}{C_{int}}\frac{C_{ext}}{C_g}\right)$$

$$= t_{p0}\left(\frac{C_{par}}{C_{int}} + \frac{C_g}{\gamma C_{ginv}}\frac{C_{ext}}{C_g}\right)$$

$$= t_{p0}\left(p + \frac{gf}{\gamma}\right)$$

$$C_{ginv}\Big|_{R_{par}=R_{inv}}$$

Modified formula

$$t_p = t_{p0}\left(1+\frac{C_{ext}}{\gamma C_g}\right) = t_{p0}\left(1+\frac{f}{\gamma}\right)$$

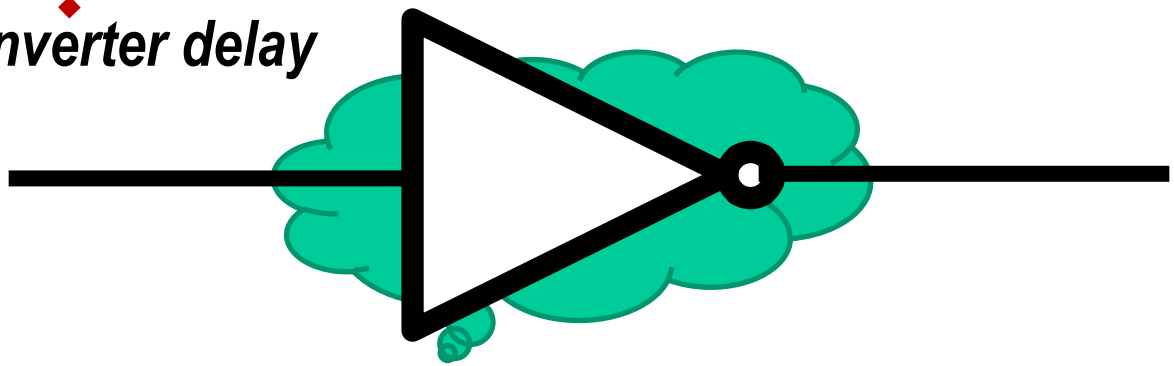
Electric effort

$$t_p = t_{p0}\left(p+g\frac{f}{\gamma}\right)$$

Intrinsic ratio ***logic effort***

$R_{par}=R_{inv}$

Unit inverter delay



Logic effort is related with… …

- ***Transistor's size?***
- ***the special Path?***
- ***the special edge?***

Outline about logic effort

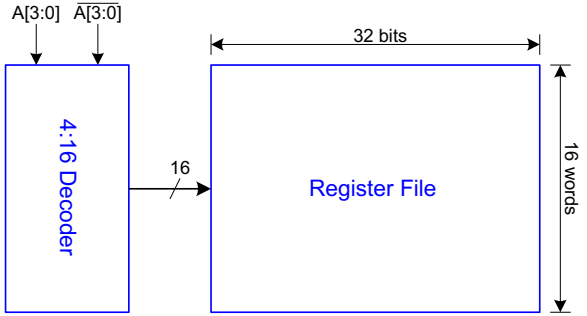
- Introduction
- Delay in a Logic Gate
- Multistage Logic Networks
- Choosing the Best Number of Stages
- Example
- Summary

Introduction

- Chip designers face a bewildering array of choices
  - What is the best circuit topology for a function?
  - How many stages of logic give least delay?
  - How wide should the transistors be?
- Logical effort is a method to make these decisions
  - Uses a simple model of delay
  - Allows **back-of-the-envelope** calculations
  - Helps make rapid comparisons between alternatives
  - Emphasizes remarkable symmetries

Example

- Ben Bitdiddle is the memory designer for the Motoroil 68W86, an embedded automotive processor. Help Ben design the decoder for a register file.
- Decoder specifications:
  - 16 word register file
  - Each word is 32 bits wide
  - Each bit presents load of 3 unit-sized transistors
  - True and complementary address inputs A[3:0]
  - Each input may drive 10 unit-sized transistors
- Ben needs to decide:
  - How many stages to use?
  - How large should each gate be?
  - How fast can decoder operate?



Delay in a Logic Gate

- Express delays in process-independent unit

$$d = \frac{d_{abs}}{\tau}$$

$\tau = 3RC$

$\approx$ 12 ps in 180 nm process

40 ps in 0.6 μm process

Delay in a Logic Gate

- Express delays in process-independent unit

$$d = \frac{d_{abs}}{\tau}$$

- Delay has two components

$$d = p + h$$

Delay in a Logic Gate

- Express delays in process-independent unit

$$d = \frac{d_{abs}}{\tau}$$

- Delay has two components

$$d = p + h$$

- **Effort delay** h= gf (a.k.a. stage effort)
  - Again has two components

Delay in a Logic Gate

- Express delays in process-independent unit

$$d = \frac{d_{abs}}{\tau}$$

- Delay has two components

$$d = p + h$$

- ***g*: logical effort**
  - Measures ***relative ability*** of gate to deliver currer
  - g ≡ 1 for inverter

Delay in a Logic Gate

- Express delays in process-independent unit

$$d = \frac{d_{abs}}{\tau}$$

- Delay has two components

$$d = p + h$$

Delay in a Logic Gate

- Express delays in process-independent unit

$$d = \frac{d_{abs}}{\tau}$$

- Delay has two components

$$d = p + h$$

- Effort delay $h = gf$ (a.k.a. stage effort)
  - Again has two components
- ***f***: electrical effort (or called fanout)= C_{out} / C_{in}
  - Ratio of output to input capacitance

Delay in a Logic Gate

- Express delays in process-independent unit

$$d = \frac{d_{abs}}{\tau}$$

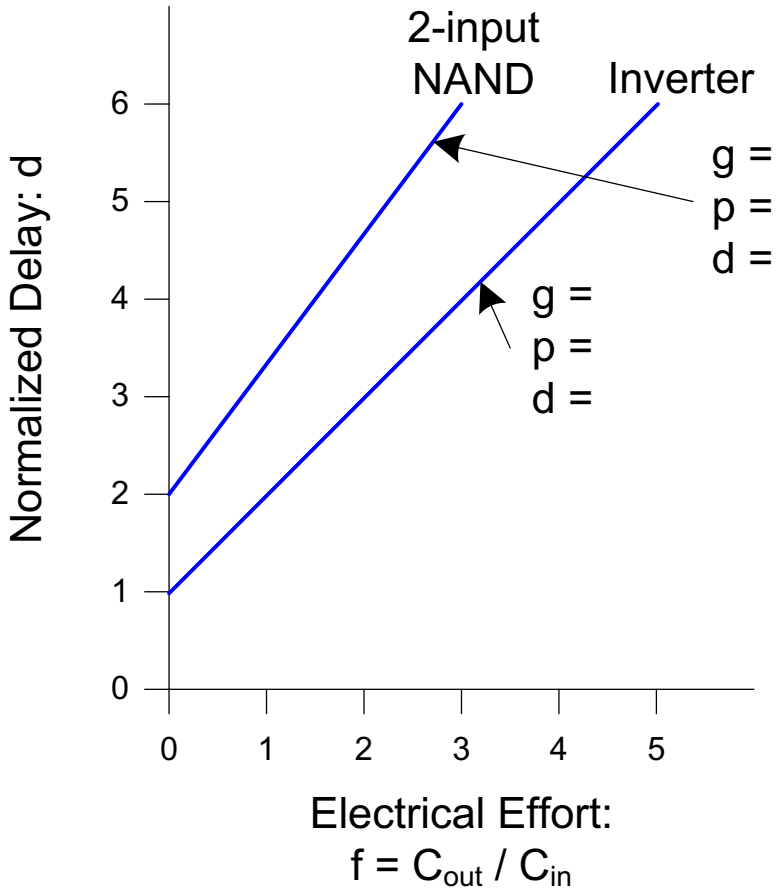
- Delay has two components

$$d = p + h$$

- Parasitic delay ***p***
  - Represents delay of gate driving no load
  - Set by internal parasitic capacitance

Delay Plots

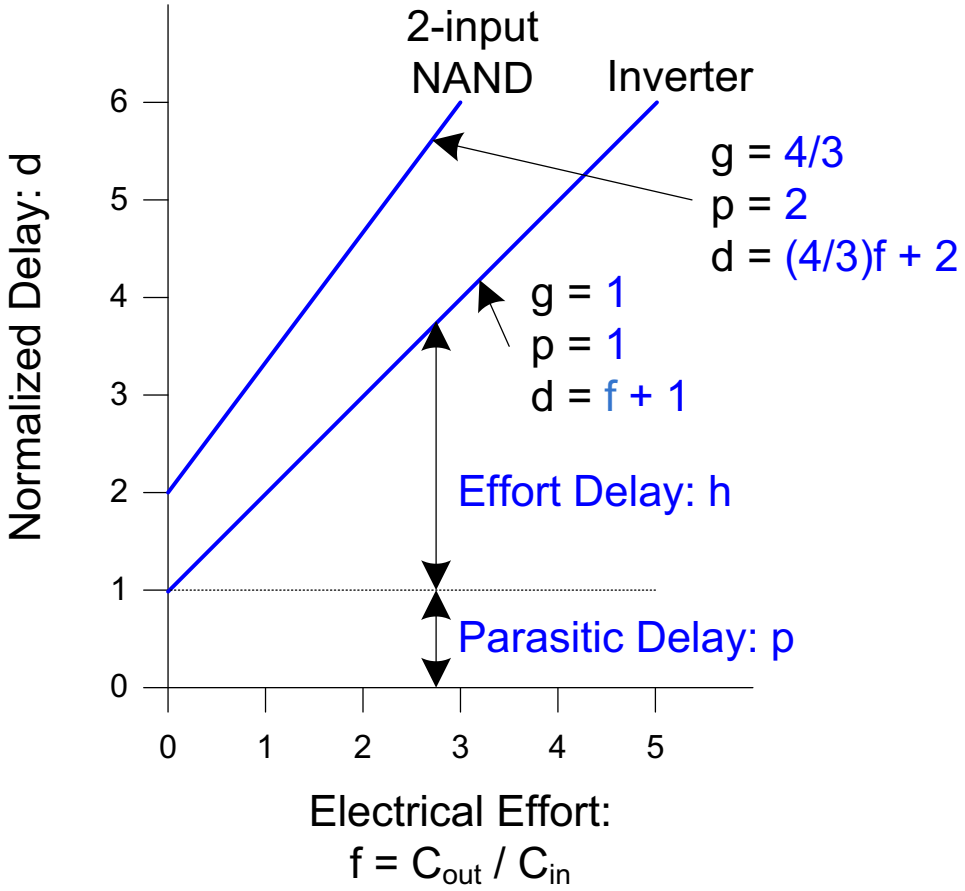
$$d = p + h$$
$$= p + fg$$



Delay Plots

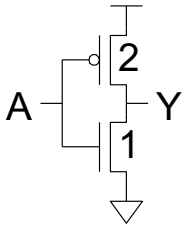
$$d = h + p$$
$$= gf + p$$

What about NOR2?

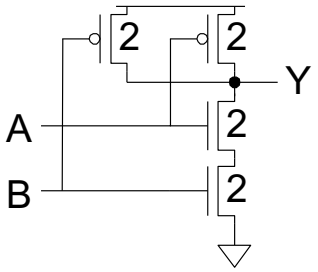


Computing Logical Effort

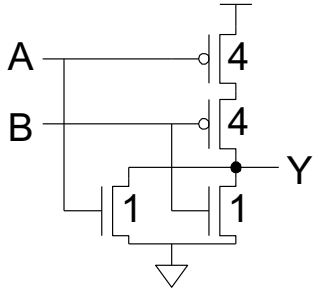
- **DEF:** ***Logical effort is the ratio of the input capacitance of a gate to the input capacitance of an inverter delivering the same output current.***
- Measure from delay vs. fanout plots
- Or estimate by counting transistor widths



C_{in} = 3
g = 3/3



C_{in} = 4
g = 4/3



C_{in} = 5
g = 5/3

Catalog of Gates

Gate type	Number of inputs				
	1	2	3	4	n
Inverter	1				
NAND		4/3	5/3	6/3	(n+2)/3
NOR		5/3	7/3	9/3	(2n+1)/3
Tristate / mux	2	2	2	2	2

Catalog of Gates

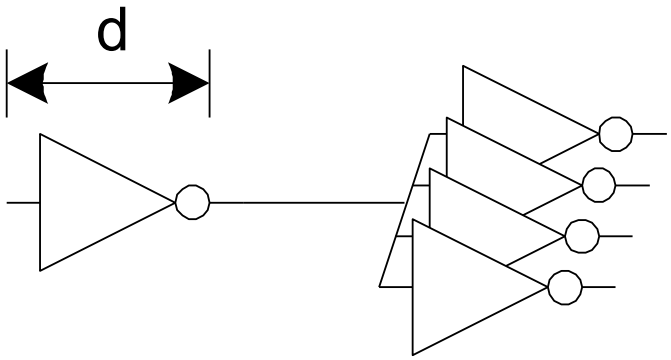
- Parasitic delay of common gates
  - In multiples of p_{inv} (≈ 1)

Gate type	Number of inputs				
	1	2	3	4	n
Inverter	1				
NAND		2	3	4	n
NOR		2	3	4	n
Tristate / mux	2	4	6	8	2n

Example: FO4 Inverter

Estimate the delay of a fanout-of-4 (FO4) inverter

- Logical Effort: g = 1
  Electrical Effort: f = 4
  Parasitic Delay: p = 1
  Stage Delay: d = 5



the FO4 delay is about
200 ps in 600nm process
60 ps in a 180nm process
f/3 ps in an *f* μm process

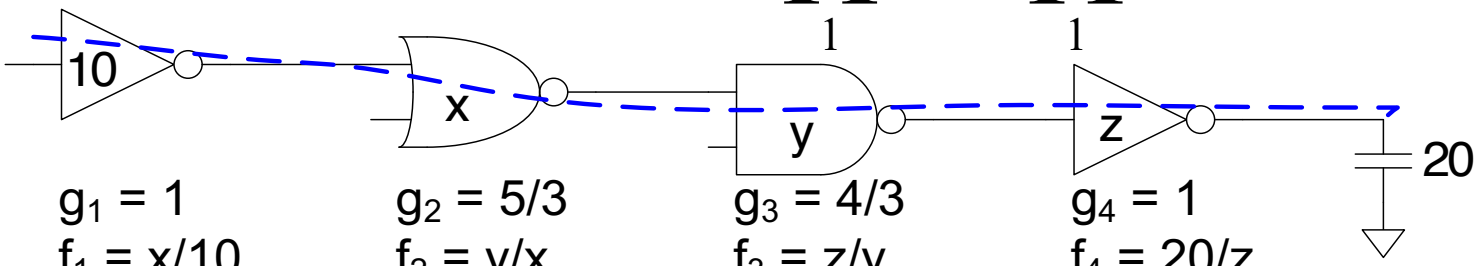
Multistage Logic Networks

Logical effort generalizes to multistage networks

Path Logical Effort $G = \prod g_i$

Path Electrical Effort $F = \frac{C_{out-path}}{C_{in-path}}$

Path Effort $H = \prod_{1}^{N} h_i = \prod_{1}^{N} f_i g_i$



Multistage Logic Networks

Logical effort generalizes to multistage networks

Path Logical Effort $G = \prod g_i$

Path Electrical Effort $F = \frac{C_{out-path}}{C_{in-path}}$

Path Effort $H = \prod_1^N h_i = \prod_1^N f_i g_i$

Can we write H = GF?

F counts once and f not.

Paths that Branch

G =

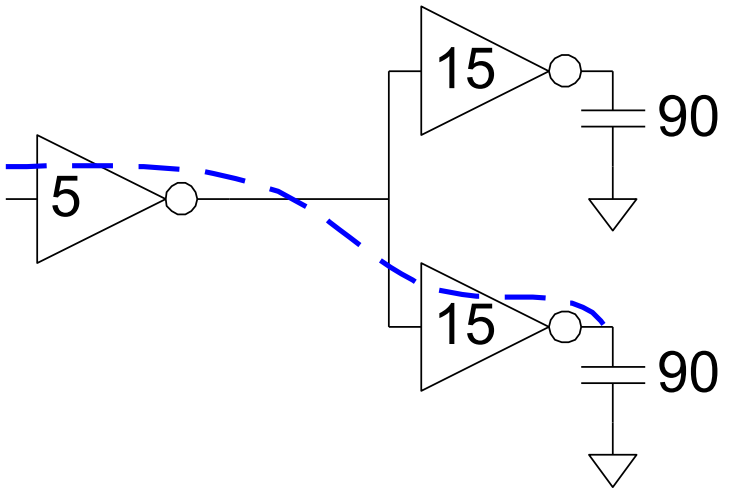
F =

GF =

f_1 =

f_2 =

H =



Paths that Branch

G = 1

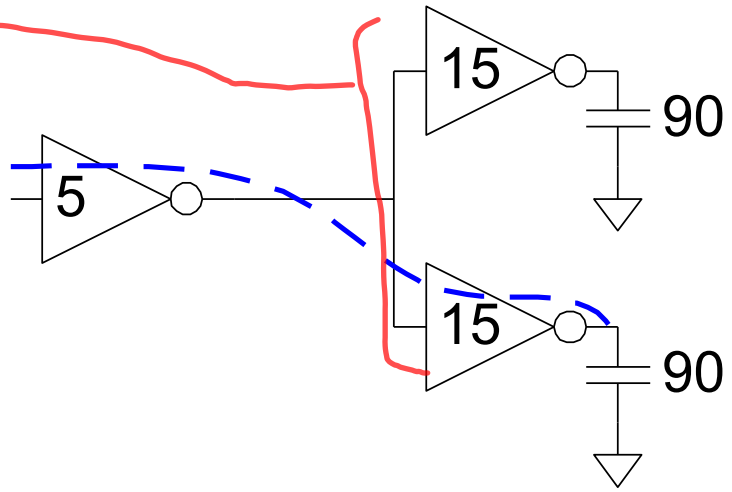
F = 90 / 5 = 18

GF = 18

f_1 = (15 +15) / 5 = 6

f_2 = 90 / 15 = 6

H = $g_1g_2f_1f_2$ = 36 = 2GF



No! Consider paths that branch

Multistage Logic Networks

- Can we write H = GF?
  - **F** count once and **f** not.

$$H = \prod_{1}^{N} h_i = \prod_{1}^{N} f_i g_i = h^N$$

$$F = \frac{C_{out-path}}{C_{in-path}} = \frac{C_2}{C_1}\frac{C_3}{C_2}\cdots\frac{C_L}{C_N} \qquad G = \prod g_i$$

- H >=GF

$$f_i = \frac{C_{i+1} + C'_{i+1}}{C_i}$$

Branching Effort

- Introduce *branching effort*
  - Accounts for branching between stages in path

$$b = \frac{C_{\text{on path}} + C_{\text{off path}}}{C_{\text{on path}}}$$

$$f_i = \frac{C_{i+1}+C'_{i+1}}{C_i} = \frac{C_{i+1}}{C_i}\frac{C_{i+1}+C'_{i+1}}{C_{i+1}} = \frac{C_{i+1}}{C_i} b_i \qquad B = \prod b_i$$

$$H = \prod_1^N h_i = \prod_1^N f_i g_i = \prod_1^N \frac{C_{i+1}}{C_i} b_i g_i = FBG$$

- Now we compute the path effort

H = GBF

It is just like inverter chain!

$$t_p = \ln 2.R_{inv}C_{ginv}\left(\frac{C_{par}}{C_{ginv}} + \frac{C_g}{C_{ginv}}\frac{C_{ext}}{C_g}\right)$$

$$= t_{p0}\left(p + \frac{gf}{\gamma}\right) = t_{p0}\left(p_i + \frac{g_i}{\gamma}\frac{C_{i+1} + C'_{i+1}}{C_i}\right)$$

$$\frac{g_{i-1}}{\gamma}\frac{1+u}{C_{i-1}} - \frac{g_i}{\gamma}\frac{C_{i+1}(1+v)}{C_i^2}\Bigg|_{\substack{C'_i = uC_i \\ C'_{i+1} = vC_{i+1}}} = 0$$

The only size related par.

$$\frac{g_{i-1}}{\gamma}\frac{C_i + C'_i}{C_{i-1}} = \frac{g_i}{\gamma}\frac{C_{i+1} + C'_{i+1}}{C_i}$$

$$g_{i-1}f_{i-1} = g_i f_i = h_{i-1} = h_i$$

Designing Fast Circuits

- Delay is smallest when each stage bears same effort

$$h = g_i f_i = H^{\frac{1}{N}}$$

- Thus minimum delay of N stage path is

$$D = t_{p_0}\left(\sum_{j=1}^{N} p_j + \frac{Nh}{\gamma}\right)$$

- ***This is a key result of logical effort***
  - ***Find fastest possible delay***
  - ***Doesn't require calculating gate sizes***

Gate Sizes

- How wide should the gates be for least delay?

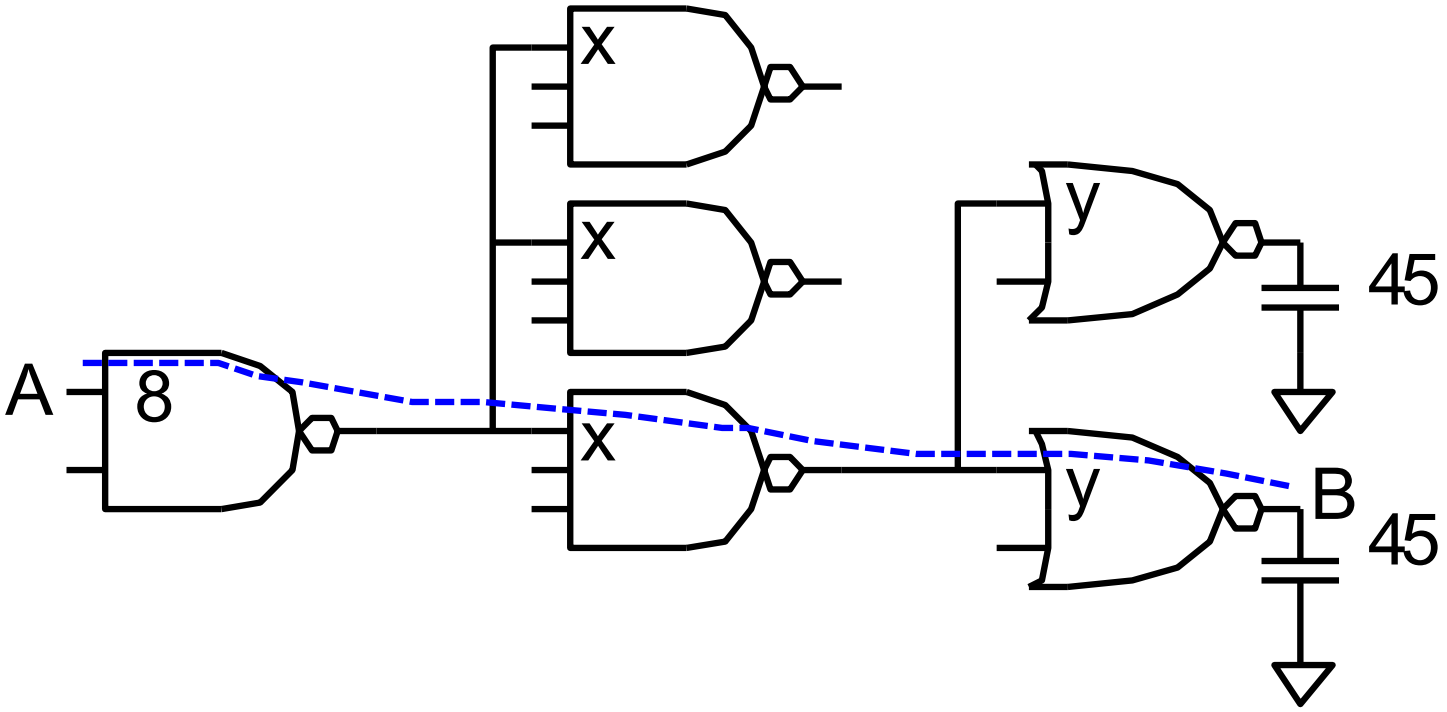
$$h = gf = g\frac{c_{out}}{c_{in}}$$

$$\Rightarrow C_{in_i} = \frac{g_i c_{out_i}}{h}$$

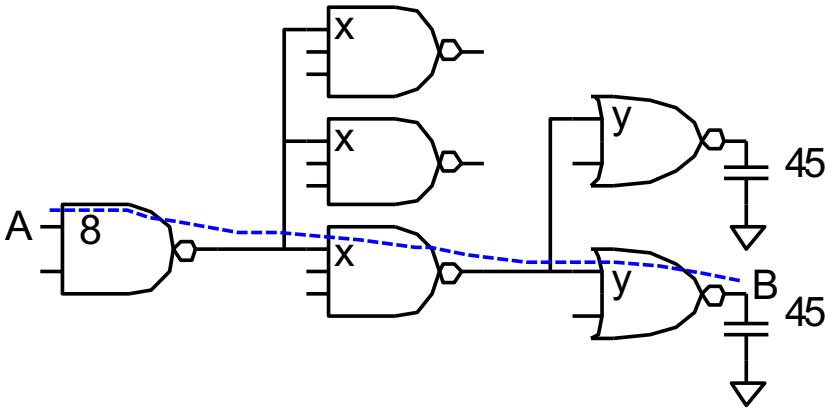
- Working backward/forward, apply capacitance transformation to find input capacitance of each gate given load it drives.

Example: 3-stage path

- Select gate sizes x and y for least delay from A to B



Example: 3-stage path



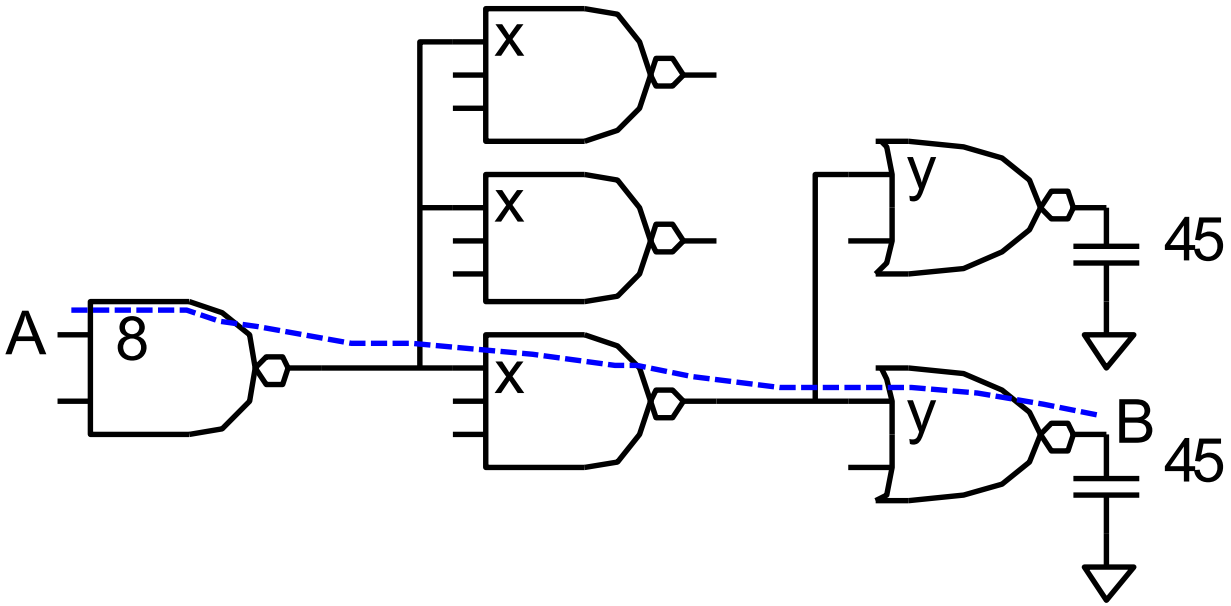
Logical Effort	G =
Electrical Effort	F =
Branching Effort	B =
Path Effort	H =
Best Stage Effort	h=
Parasitic Delay	P =
Delay	D =

Example: 3-stage path

- Work backward for sizes

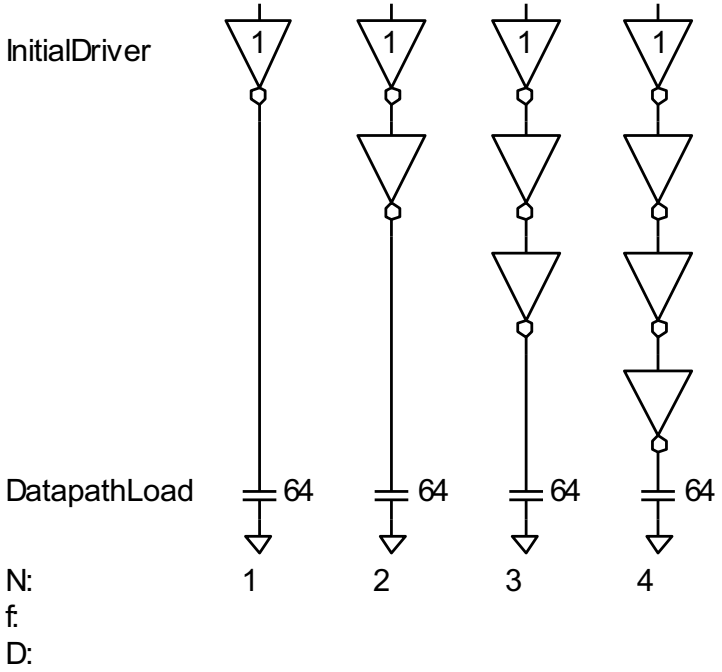
  y =

  x =



Best Number of Stages

- How many stages should a path use?
  - Minimizing number of stages is not always fastest
- Example: drive 64-bit datapath with unit inverter
- D =



Derivation

- Consider adding inverters to end of path
  - How many give least delay?

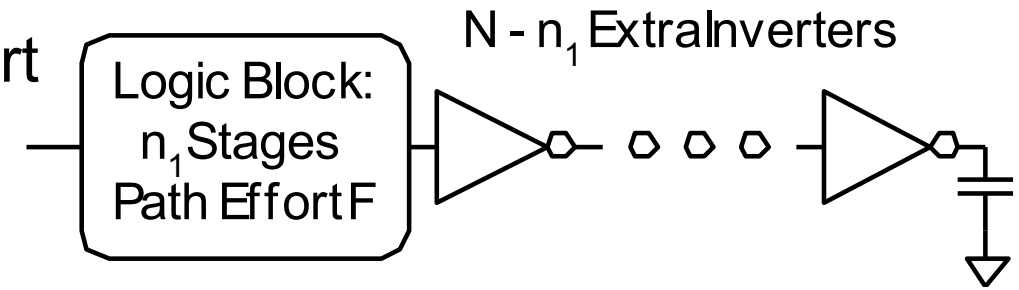
$$D = NH^{\frac{1}{N}} + \sum_{i=1}^{n_1} p_i + (N - n_1)p_{inv}$$

$$\frac{\partial D}{\partial N} = -H^{\frac{1}{N}} \ln H^{\frac{1}{N}} + H^{\frac{1}{N}} + p_{inv} = 0$$

- Define best stage effort

$$h = H^{\frac{1}{N}}$$

$$h(1 - \ln h) + p_{inv} = 0$$



Best Stage Effort

- $p_{inv} + h(1 - \ln h) = 0$ has no closed-form solution
- Neglecting parasitics (p_{inv} = 0), find h = 2.718 (e)
- For p_{inv} = 1, solve numerically for h= 3.59

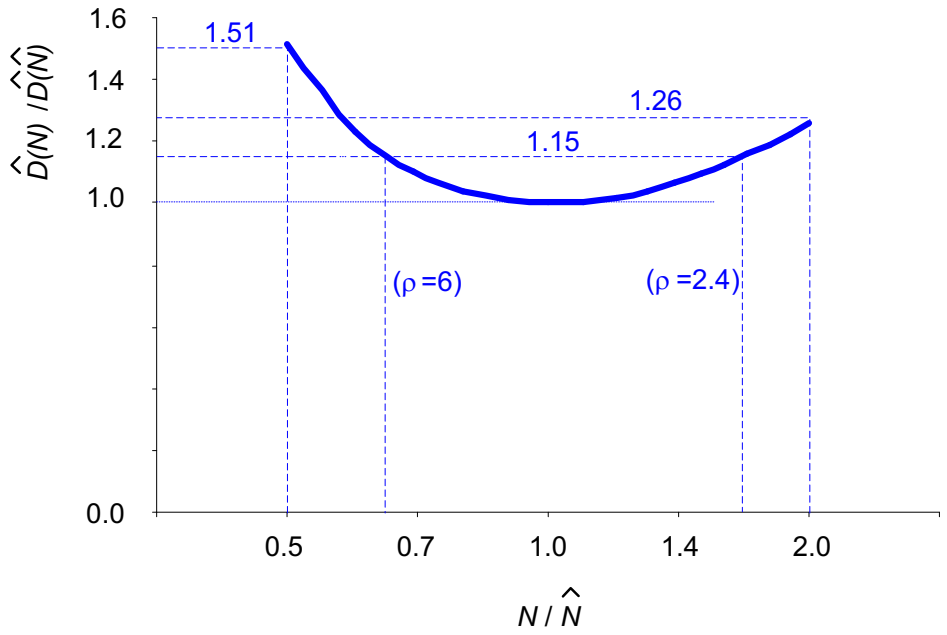
$$N = \log_h H$$

$$D = Nh + \sum_{i=1}^{n_1} p_i + (N - n_1)$$

$$= h \log_h H + \sum_{i=1}^{n_1} p_i + (\log_h H - n_1)$$

Sensitivity Analysis

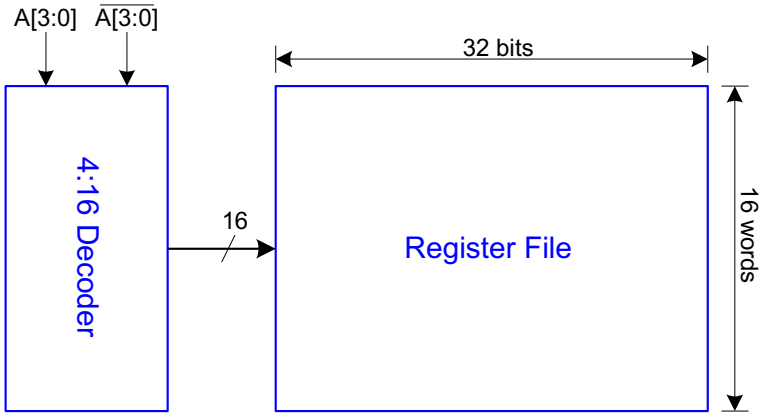
- How sensitive is delay to using exactly the best number of stages?



- 2.4 < *h*< 6 gives delay within 15% of optimal
  - We can be sloppy!
  - I like *h*= 4

Example, Revisited

- Ben Bitdiddle is the memory designer for the Motoroil 68W86, an embedded automotive processor. Help Ben design the decoder for a register file.
- Decoder specifications:
  - 16 word register file
  - Each word is 32 bits wide
  - Each bit presents load of 3 unit-sized transistors
  - True and complementary address inputs A[3:0]
  - Each input may drive 10 unit-sized transistors
- Ben needs to decide:
  - How many stages to use?
  - How large should each gate be?
  - How fast can decoder operate?



Number of Stages

- Decoder effort is mainly electrical and branching

  Electrical Effort: F =

  Branching Effort: B =

- If we neglect logical effort (assume G = 1)

  Path Effort: H =

  Number of Stages: N =

- Try a 3-stage design

Gate Sizes & Delay

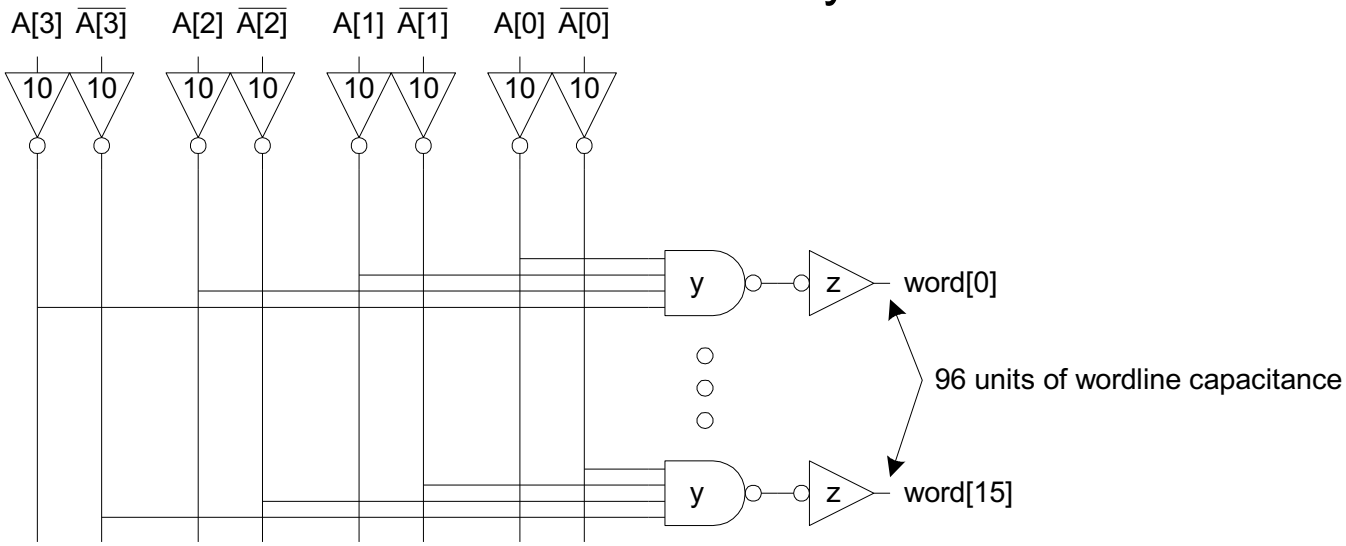
Logical Effort: G =

Path Effort: H =

Stage Effort: h=

Path Delay: D =

Gate sizes: z = y =



Number of Stages

Decoder effort is mainly electrical and branching

Electrical Effort: F= (32*3) / 10 = 9.6

Branching Effort: B = 8

If we neglect logical effort (assume G = 2)

Path Effort: **H = GBF = 2*8*9.6=153.6**

Number of Stages: $N = \log_4 H = 3.6$

Try a **3/4-stage design**

Gate Sizes & Delay

Logical Effort: G = 1 * 6/3 * 1 = 2

Path Effort: H = GBF = 154

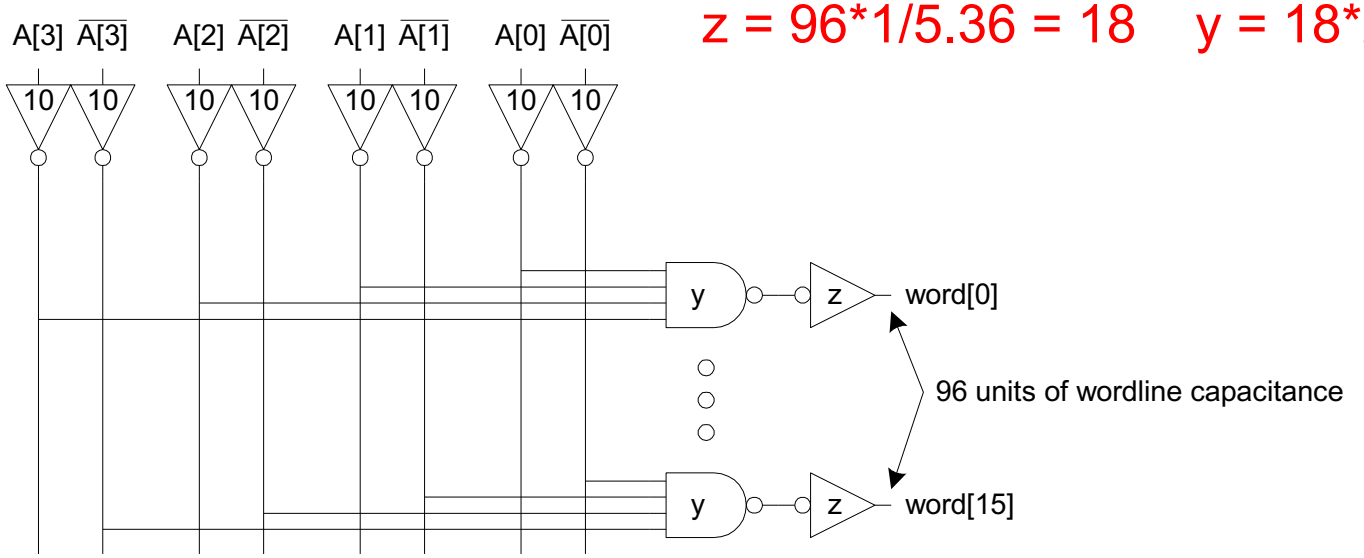
Stage Effort: **$h = H^{1/4} = 3.52$** $h = H^{1/3} = 5.36$

Path Delay: **D =4h+1+4+1+1=7+14.08=21.1**

D =3h+1+4+1=7+14.08=22.1

Gate sizes: **s=96*1/3.52=27; z=27*1/3.52=8; y=8*2/3.52=5**

z = 96*1/5.36 = 18 y = 18*2/5.36 = 6.7



Comparison

- Compare many alternatives with a spreadsheet

Design	N	G	P	D
NAND4-INV	2	2	5	29.8
NAND2-NOR2	2	20/9	4	30.1
INV-NAND4-INV	3	2	6	22.1
NAND4-INV-INV-INV	4	2	7	21.1
NAND2-NOR2-INV-INV	4	20/9	6	20.5
NAND2-INV-NAND2-INV	4	16/9	6	19.7
INV-NAND2-INV-NAND2-INV	5	16/9	7	20.4
NAND2-INV-NAND2-INV-INV-INV	6	16/9	8	21.6

Review of Definitions

Term	Stage	Path
number of stages	N	
logical effort	g	$G = \prod g_i$
electrical effort	$f = C_{out}/C_{in}$	$F = C_{out\text{-}path}/C_{in\text{-}path}$
branching effort	$b = \frac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{on-path}}}$	$B = \prod b_i$
effort	$h = gf$	H=GBF
effort delay	h	$D_F = \sum h_i$
parasitic delay	p	$P = \sum p_i$
delay	$d = p+f$	$D = D_F + P$

Method of Logical Effort

1) Compute path effort H=GBF(G=1)
2) Estimate best number of stages $N=\log_4 H$
3) Sketch path with N stages
4) Re-compute G
5) Determine best stage effort $h=GBF^{1/N}=H^{1/N}$
6) Estimate least delay D=P+h*N
7) Find gate sizes

$$h = gf = g\frac{c_{out}}{c_{in}}$$

$$\Rightarrow C_{in_i} = \frac{g_i c_{out_i}}{h}$$

Limits of Logical Effort

- Chicken and egg problem
  - Need path to compute G
  - But don’t know number of stages without G
- Simplistic delay model
  - Neglects input rise time effects
- Interconnect
  - Iteration required in designs with wire
- Maximum speed only
  - Not minimum area/power for constrained delay

Summary

- Numeric logical effort characterizes gates
- NANDs are faster than NORs in CMOS
- Paths are fastest when effort delays are ~4
- Path delay is weakly sensitive to stages, sizes
- Using fewer stages doesn't mean faster paths
- Delay of path is about $\log_4 H$ stage

Example 1

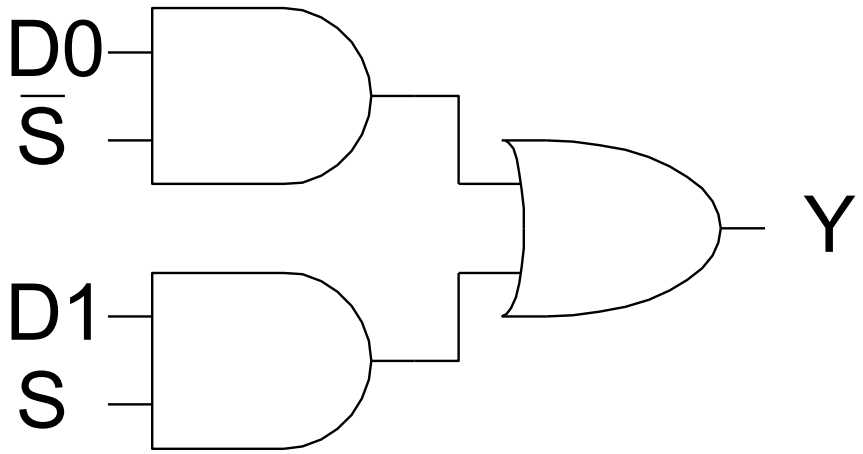
```
module mux(input  s, d0, d1,
           output y);
   assign y = s ? d1 : d0;
endmodule
```

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.

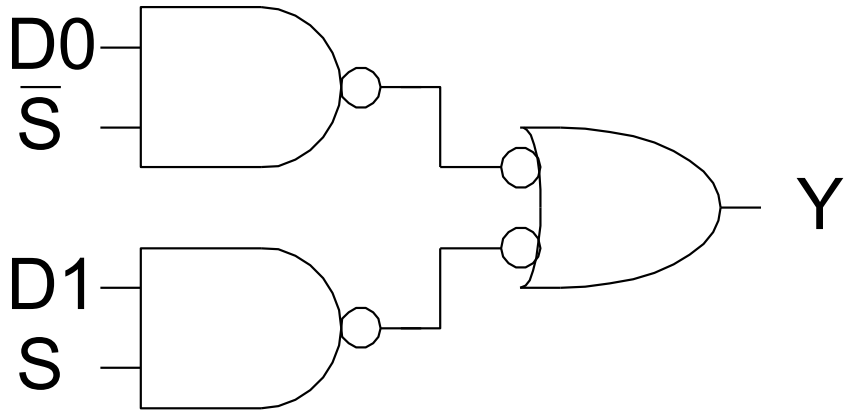


Example 2

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

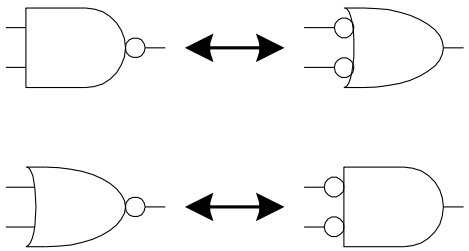
Example 2

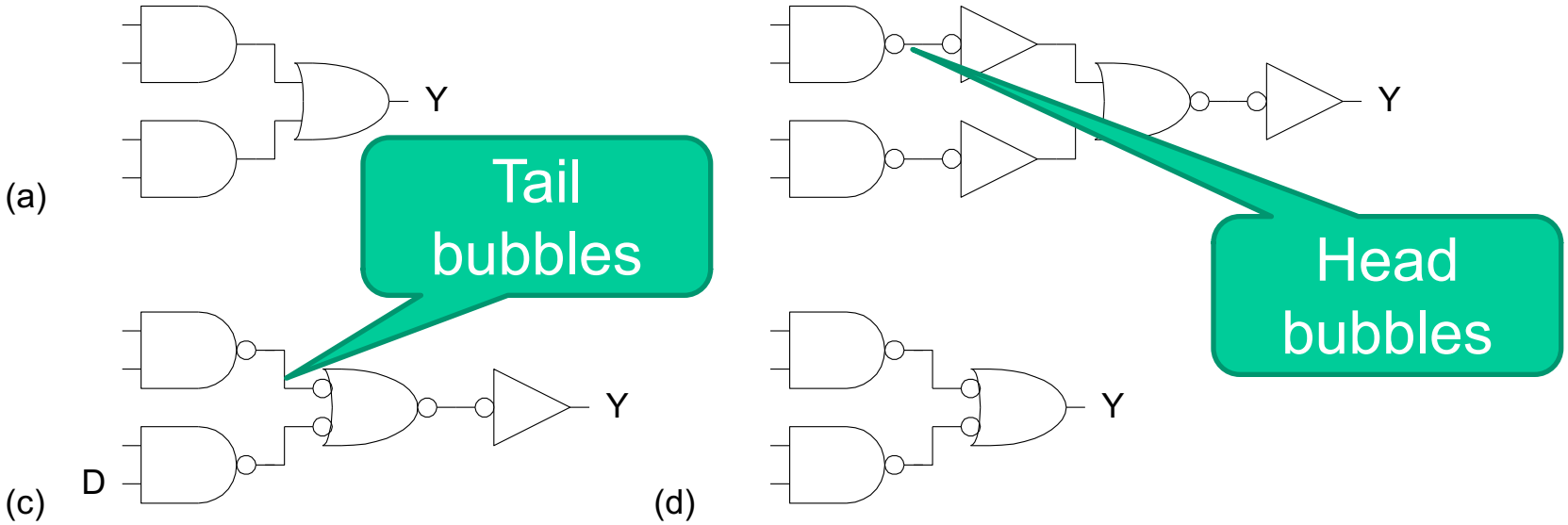
2) Sketch a design using NAND, NOR, and NOT gates.
Assume ~S is available.



Bubble Pushing

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



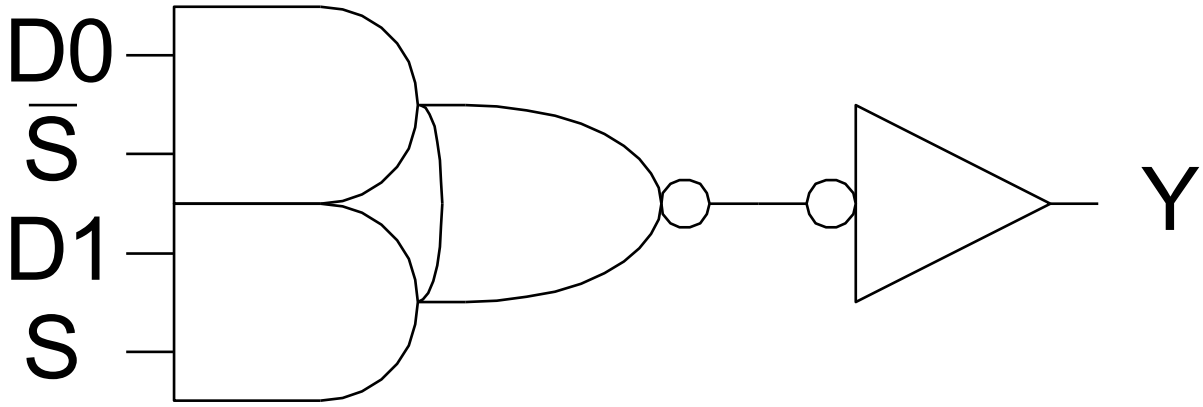


Example 3

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.



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.

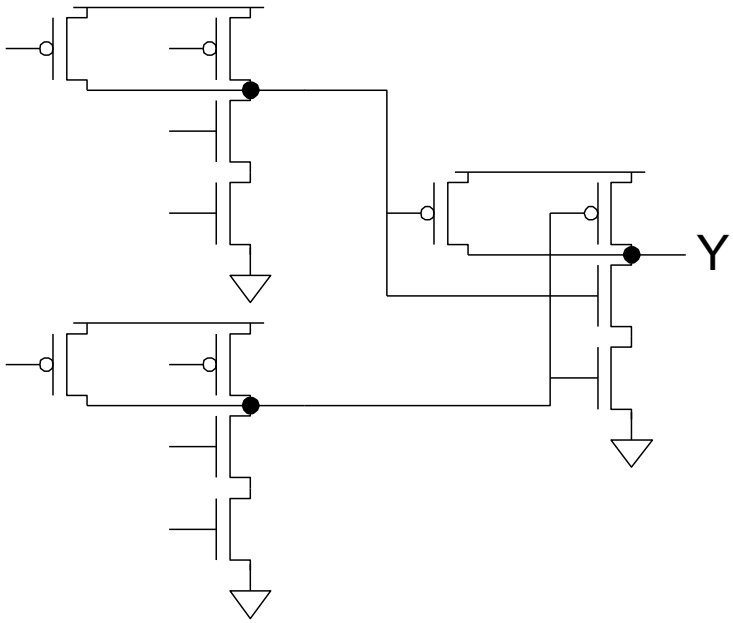
Example 4

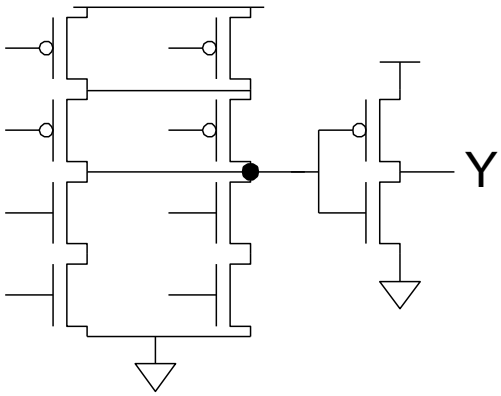
- The multiplexer has a maximum input capacitance of 16 units on each input. It must drive a load of 160 units. Estimate the delay of the NAND and compound gate designs.

F = 160 / 16 = 10
B = 1
N = 2

Example 5

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





Another application

- Example 4.15

The circuit in Figure 4.37 has nonuniform branching, reconvergent fanout, and a wire load in the middle of the path, all of which stymie back-of-the-envelope application of Logical Effort. The wire load is given in the same units as the gate capacitances (i.e., multiples of the capacitance of a unit inverter). Assume the inputs arrive at time 0. Write an expression for the arrival time of the output as a function of the gate drives. Determine the sizes to achieve minimum delay.

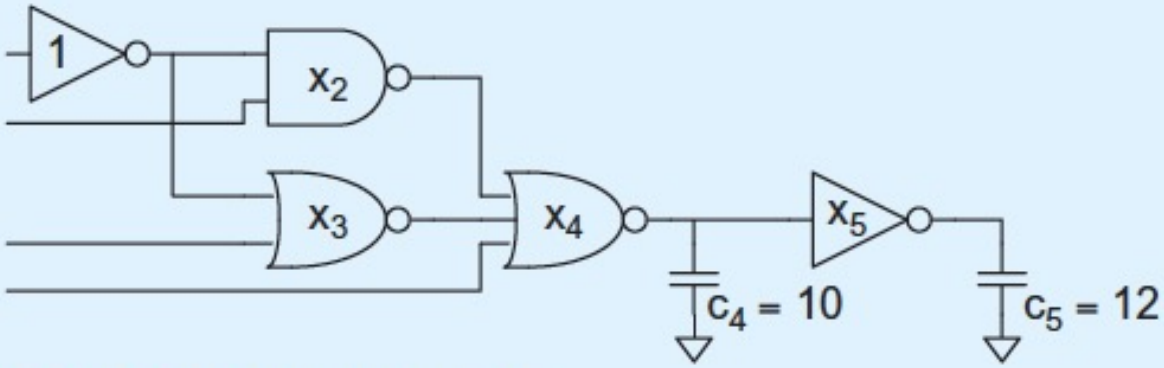


FIGURE 4.37 Example path

Another application

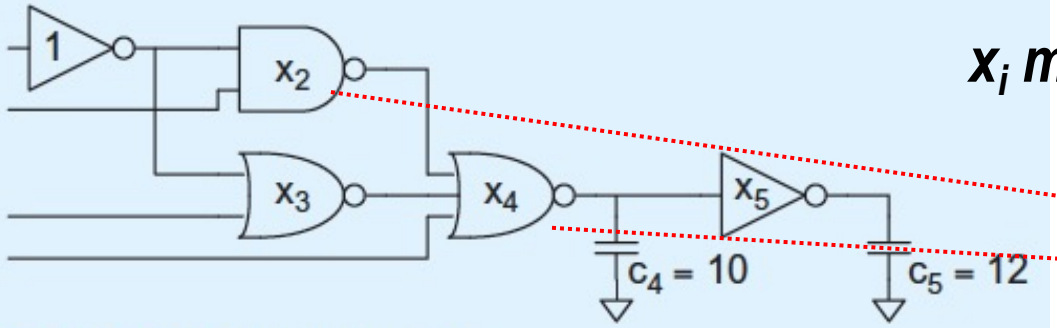


FIGURE 4.37 Example path

x_i means x_i times of unified logic

$g_i.x_i.C_{ref\text{-}inverter}$

$$d_i = p_i + f_i g_i = p_i + \frac{C_{i+1}}{C_i} g_i = p_i + \frac{g_{i+1} x_{i+1} C_{ref-inv}}{g_i x_i C_{ref-inv}} g_i = p_i + \frac{g_{i+1} x_{i+1}}{x_i} \qquad g_{N+1}=1$$

$$d_i = p_i + \frac{C_{i+1} + C'_{i+1}}{C_i} g_i = p_i + \frac{g_{i+1} x_{i+1} C_{ref-inv} + x'_{i+1} C_{ref-inv}}{g_i x_i C_{ref-inv}} g_i = p_i + \frac{g_{i+1} x_{i+1} + x'_{i+1}}{x_i}$$

$$d_1 = 1 + \tfrac{4}{3} x_2 + \tfrac{5}{3} x_3$$

$$d_2 = 2 + \tfrac{7}{3} \frac{x_4}{x_2}$$

$$d_3 = 2 + \tfrac{7}{3} \frac{x_4}{x_3}$$

$$d_4 = 3 + \frac{x_5}{x_4} + \frac{10}{x_4}$$

$$d_5 = 1 + \frac{12}{x_5}$$

$$a_1 = d_1$$

$$a_2 = a_1 + d_2$$

$$a_3 = a_1 + d_3$$

$$a_4 = \max\{a_2, a_3\} + d_4$$

$$a_5 = a_4 + d_5 = d_1 + \max\{d_2, d_3\} + d_4 + d_5$$

Another example of Rabaey.Textbook

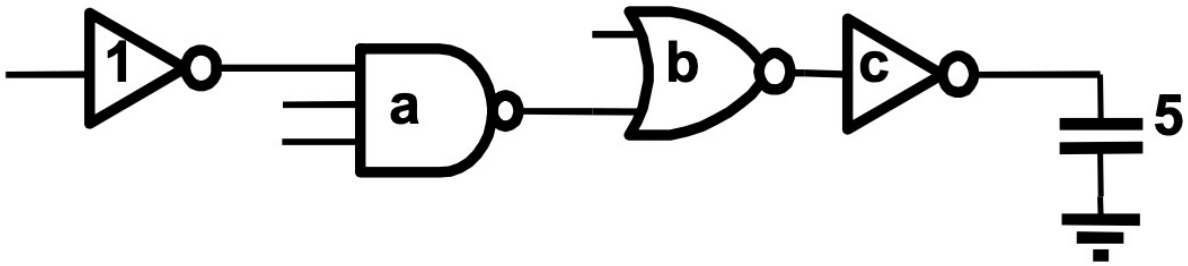


Figure 6.19 Critical path of combinational network.

$H = FG = 125/9$, and the optimal stage effort h is $\sqrt[4]{H} = 1.93$. Taking into account the gate types, we derive the fanout factors: $f_1 = 1.93$; $f_2 = 1.93\times(3/5) = 1.16$; $f_3 = 1.16$; $f_4 = 1.93$. Notice that the inverters are assigned larger electrical efforts than the more complex gates because they are better at driving loads. From this, we can derive the sizes of the gates (with respect to their minimum-sized versions): $a = f_1/g_2 = 1.16$; $b = f_1 f_2/g_3 = 1.34$; $c = f_1 f_2 f_3/g_4 = 2.60$.

F=5,G=1*5/3*5/3*1,B=1

H=GBF=125/9, h=H1/4=1.93

$h=g_if_i=g_i\ ^*g_{i+1}x_{i+1}/g_ix_i=g_{i+1}x_{i+1}/x_i$

$x_i= g_{i+1}x_{i+1}/h$

c=5/1.93=2.59

b=1*2.59/1.93=1.34

a=5/3*1.34/1.93=1.16

$h=g_if_i=g_i\ ^*g_{i+1}x_{i+1}/g_ix_i=g_{i+1}x_{i+1}/x_i$

$x_{i+1} = x_i\, h/g_{i+1} = x_i\, g_if_i\, /g_{i+1}$

$=x_{i-1}\, g_if_i\, g_{i-1}f_{i-1}\, /\, g_i\, g_{i+1}$

$= x_1\, f_1 \ldots f_i\, /\, g_{i+1}$

a=1.93/5/3=1.16

b=1.16*1.93/5/3=1.34

c=1.34*1.93/1=2.59

Another application

- *In paths that branch, each fork should contribute equal delay. If one fork were faster than the other, it could be downsized to reduce the capacitance it presents to the stage before the branch.*
- *The stage efforts, f, are equal for each gate in paths with no fixed capacitive loads,but may change after a load.*
- *To minimize delay, upsize gates on nodes with large fixed capacitances to reduce the effort borne by the gate, while only slightly increasing the effort borne by the predecessor.*

TABLE 4.6 Path design for minimum delays

Stage (i)	x_i	f_i	c_{in}	d_i	a_i
1: INV	1	4.85	1	5.85	5.85
2: NAND2	1.62	4.85	2.16	6.85	12.70
3: NOR2	1.62	4.85	2.70	6.85	12.70
4: NOR3	3.37	4.85	7.86	7.85	20.55
5: INV	6.35	1.89	6.35	2.89	23.44

Logic effort is related with… …

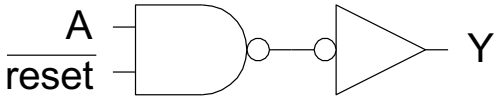
① *Transistor's size?*

② ***the special Path?***

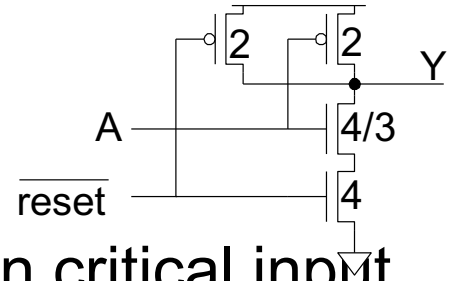
③ ***the special edge?***

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



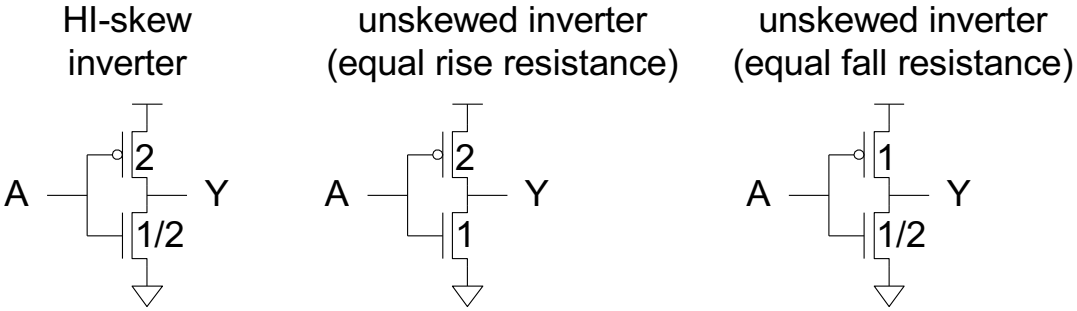
- g_A =
- g_B =
- g_{total} =



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

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.
  - gu =
  - gd =

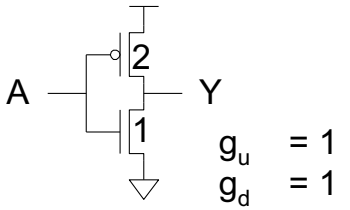
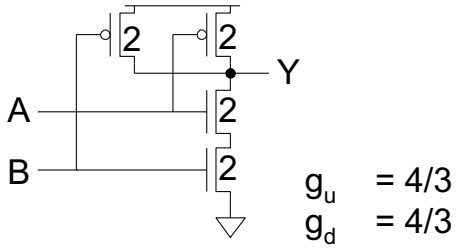
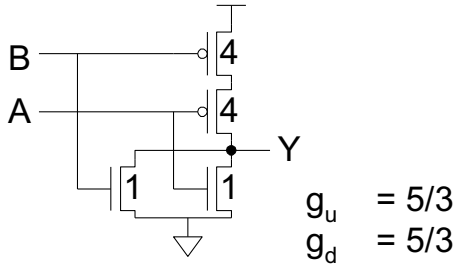
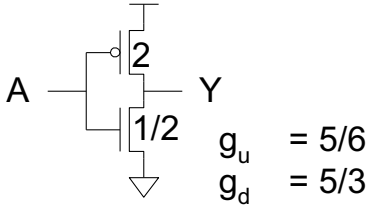
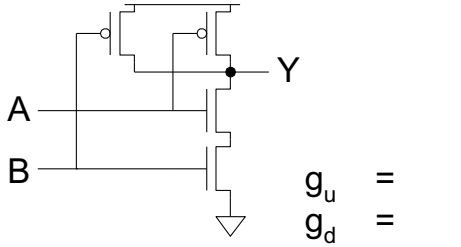
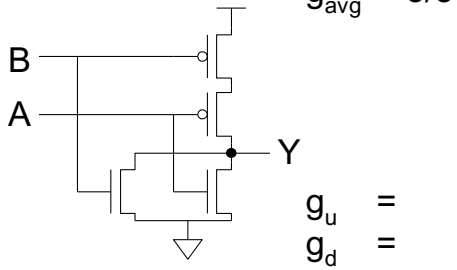
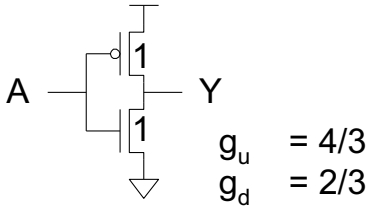
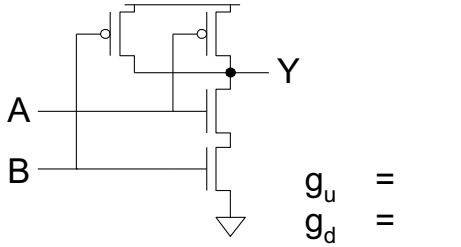
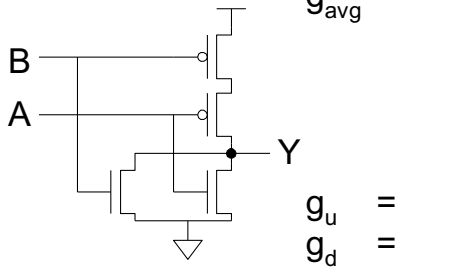
HI- and LO-Skew

- 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)

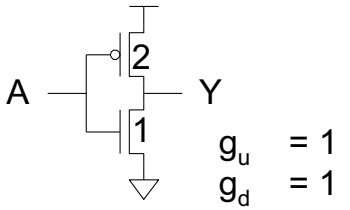
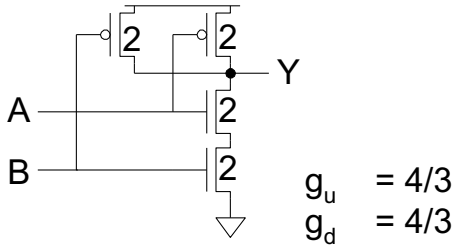
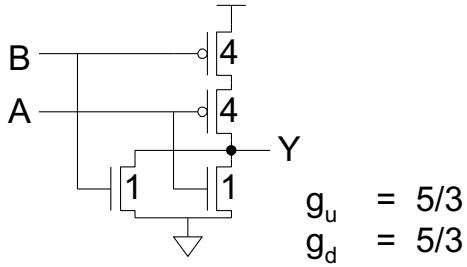
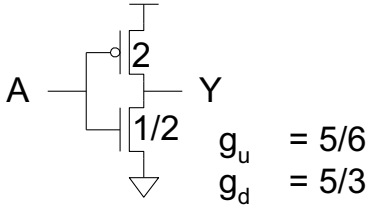
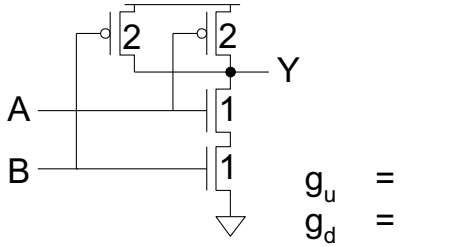
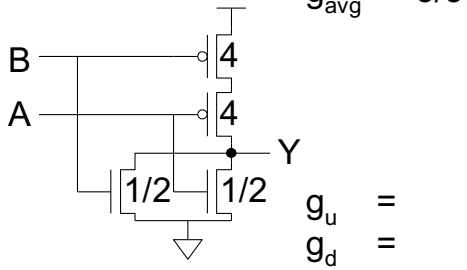
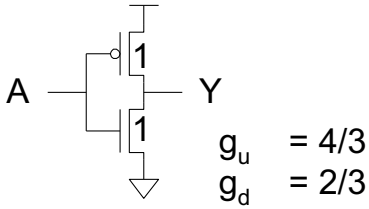
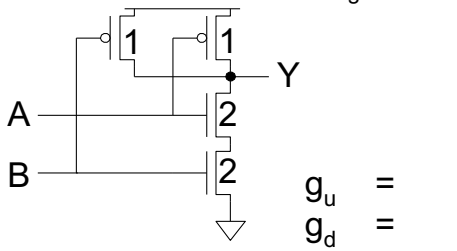
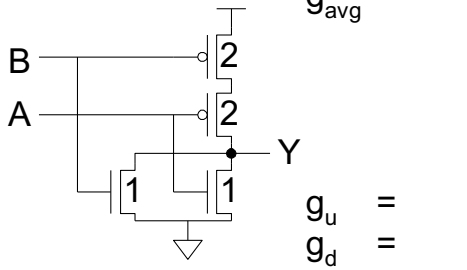
Logical effort is smaller for favored direction

But larger for the other direction

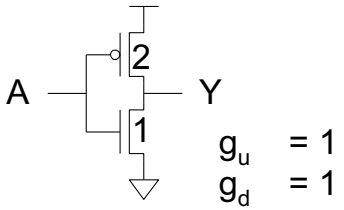
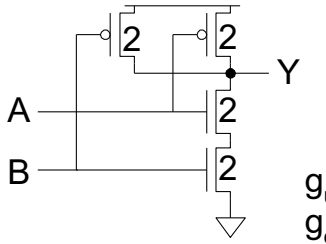
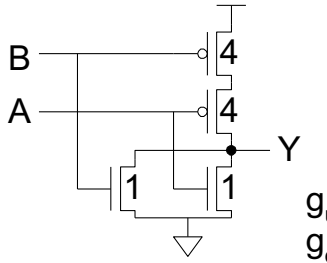
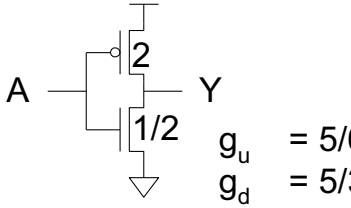
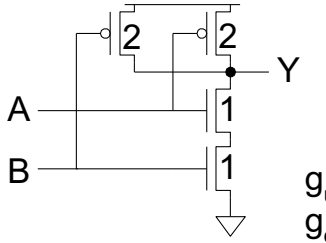
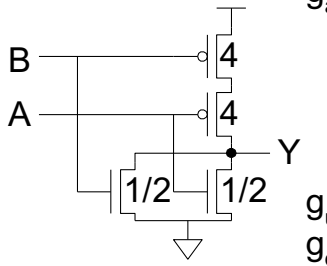
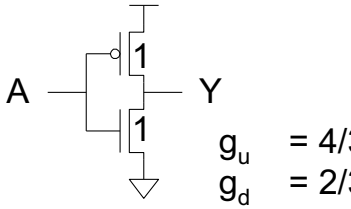
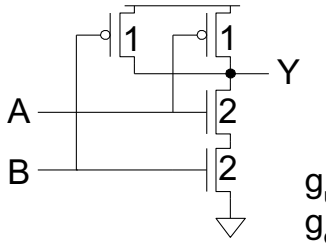
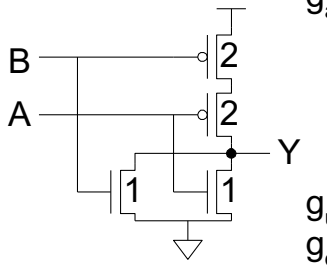
Catalog of Skewed Gates

	Inverter	NAND2	NOR2
unskewed	 $g_u = 1$ $g_d = 1$ $g_{avg} = 1$	 $g_u = 4/3$ $g_d = 4/3$ $g_{avg} = 4/3$	 $g_u = 5/3$ $g_d = 5/3$ $g_{avg} = 5/3$
HI-skew	 $g_u = 5/6$ $g_d = 5/3$ $g_{avg} = 5/4$	 $g_u =$ $g_d =$ $g_{avg} =$	 $g_u =$ $g_d =$ $g_{avg} =$
LO-skew	 $g_u = 4/3$ $g_d = 2/3$ $g_{avg} = 1$	 $g_u =$ $g_d =$ $g_{avg} =$	 $g_u =$ $g_d =$ $g_{avg} =$

Catalog of Skewed Gates

	Inverter	NAND2	NOR2
unskewed	 $g_u = 1$ $g_d = 1$ $g_{avg} = 1$	 $g_u = 4/3$ $g_d = 4/3$ $g_{avg} = 4/3$	 $g_u = 5/3$ $g_d = 5/3$ $g_{avg} = 5/3$
HI-skew	 $g_u = 5/6$ $g_d = 5/3$ $g_{avg} = 5/4$	 $g_u =$ $g_d =$ $g_{avg} =$	 $g_u =$ $g_d =$ $g_{avg} =$
LO-skew	 $g_u = 4/3$ $g_d = 2/3$ $g_{avg} = 1$	 $g_u =$ $g_d =$ $g_{avg} =$	 $g_u =$ $g_d =$ $g_{avg} =$

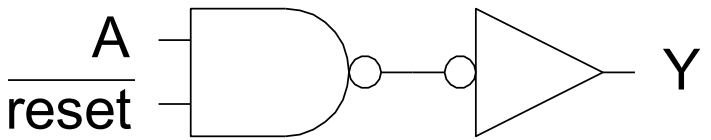
Catalog of Skewed Gates

	Inverter	NAND2	NOR2
unskewed	 $g_u = 1$ $g_d = 1$ $g_{avg} = 1$	 $g_u = 4/3$ $g_d = 4/3$ $g_{avg} = 4/3$	 $g_u = 5/3$ $g_d = 5/3$ $g_{avg} = 5/3$
HI-skew	 $g_u = 5/6$ $g_d = 5/3$ $g_{avg} = 5/4$	 $g_u = 1$ $g_d = 2$ $g_{avg} = 3/2$	 $g_u = 3/2$ $g_d = 3$ $g_{avg} = 9/4$
LO-skew	 $g_u = 4/3$ $g_d = 2/3$ $g_{avg} = 1$	 $g_u = 2$ $g_d = 1$ $g_{avg} = 3/2$	 $g_u = 2$ $g_d = 1$ $g_{avg} = 3/2$

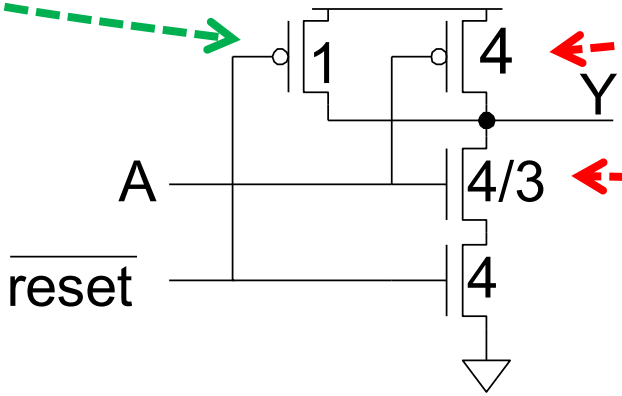
Combine asymmetric and skewed gates

Downsize noncritical transistor on unimportant input

Reduces parasitic delay for critical input



Hiskew for the special path



$$g_{reset\_r} = {}^{5}/_{1.5} = {}^{10}/_{3}$$

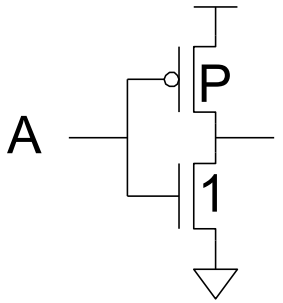
$$g_{a\_r} = \frac{16}{3}\Big/_{6} = {}^{8}/_{9}$$

$$p_{a\_r} = \frac{19}{3}\Big/_{6} = {}^{19}/_{18}$$

critical transistor on important input

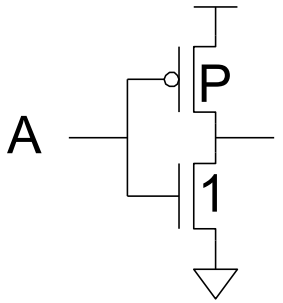
Best P/N Ratio

- 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 =



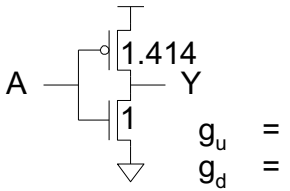
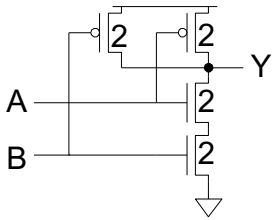
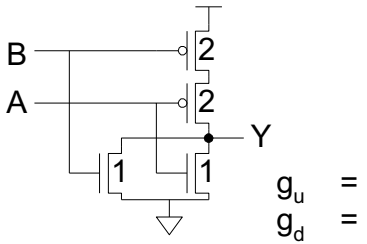
Best P/N Ratio

- 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)(μ/P)
  - t_{pd} = (P+1)(1+μ/P)/2 = (P + 1 + μ + μ/P)/2
  - Differentiate t_{pd} w.r.t. P
  - Least delay for P = $\sqrt{\mu}$



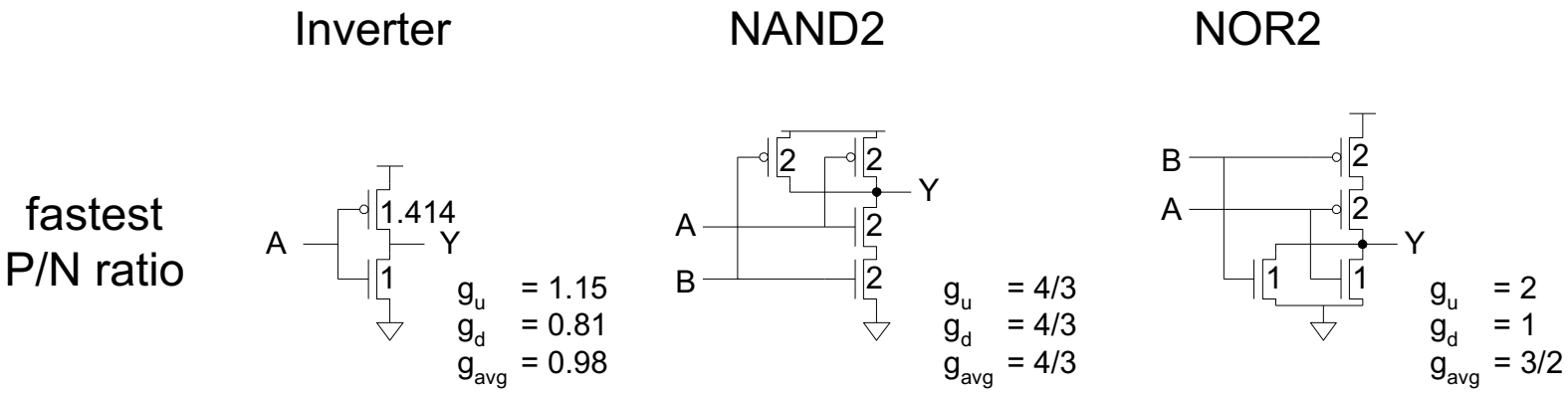
P/N Ratios

- 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_u =	g_u =
	g_d =	g_d =	g_d =
	g_{avg} =	g_{avg} =	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



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