ECE1388 VLSI Design Methodology

Lecture 6: Logical Effort

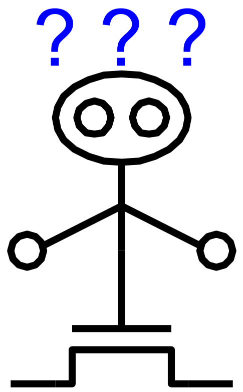
Adapted from Weste, Harris CMOS VLSI Design

Outline

- 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
  - A valuable academic method that helps develop intuition
  - In practice (ie. Industry) - used only for inverters, not other gates

Delay in a Logic Gate

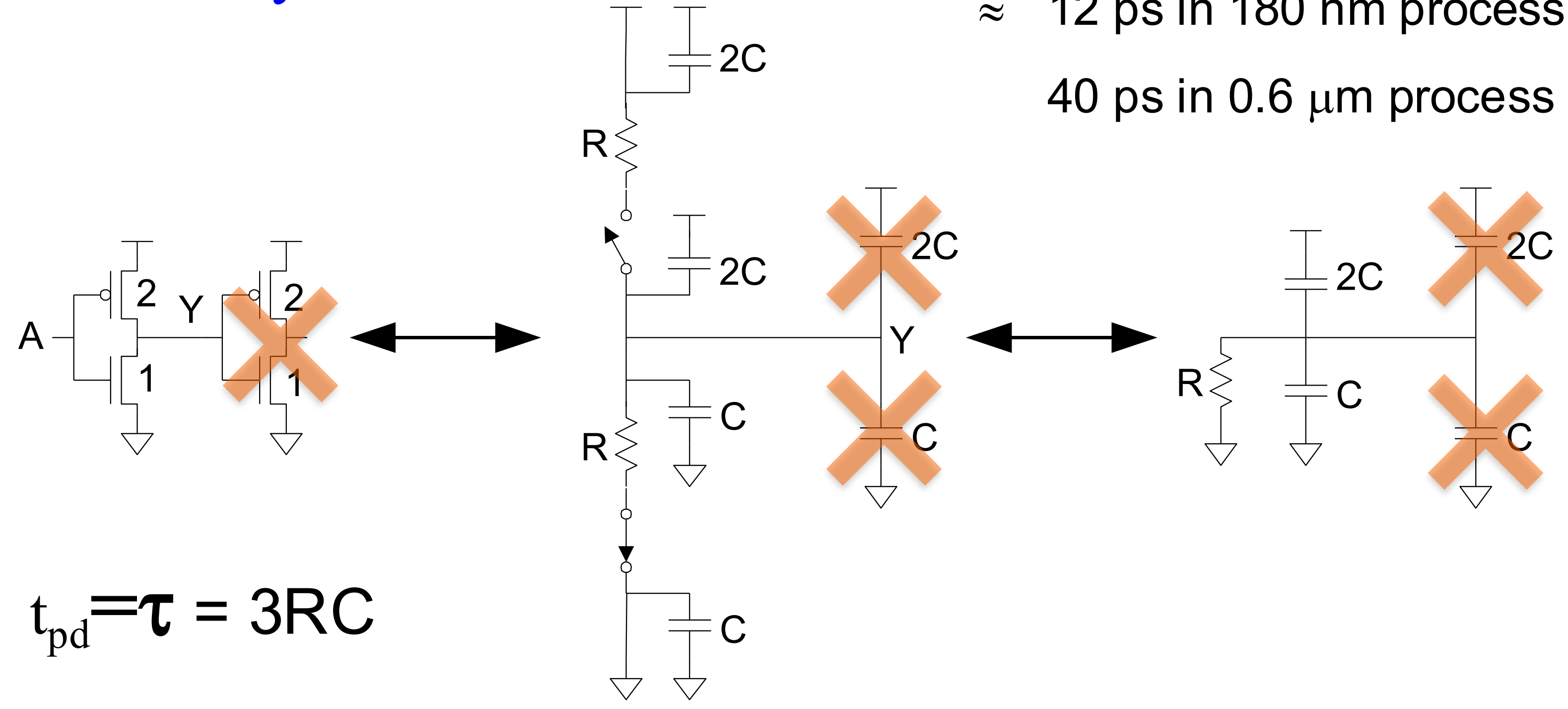
- Express delays in **process-independent unit**

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

$\tau =$ 3RC - unit inverter delay **without load**

$\approx$ 12 ps in 180 nm process

40 ps in 0.6 μm process



$t_{pd} = \tau = 3RC$

Delay in a Logic Gate

- Delay has two components

$$d = f + p$$

- **Effort delay** $f = gh$ (*a.k.a.* stage effort)
  - g: ***logical effort***
    - Measures relative ability of gate to deliver current
    - $g \equiv 1$ for inverter
  - h: ***electrical effort*** = C_{out} / C_{in}
    - Ratio of output to input capacitance (*a.k.a.* fanout)

Delay in a Logic Gate

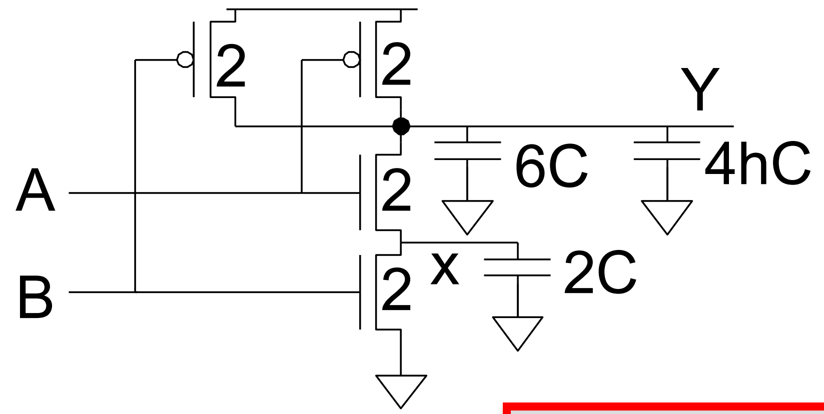
- Delay has two components

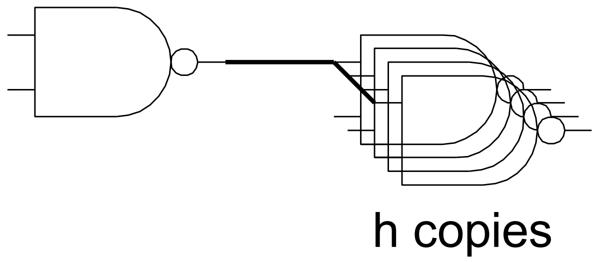
$$d = f + p$$

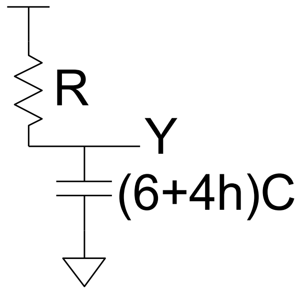
- **Effort delay** $f = gh$ (*a.k.a.* stage effort)
  - g: ***logical effort***
    - Measures *relative* ability of gate to deliver current
    - $g \equiv 1$ for inverter
  - h: ***electrical effort*** $= C_{out} / C_{in}$
    - Ratio of output to input capacitance (*a.k.a.* fanout)
- **Parasitic delay** p
  - Represents delay of gate driving no load
  - Set by internal parasitic capacitance

Recall: 2-input NAND t_{pdr}

- Estimate rising and falling worst-case (propagation) delays of a 2-input NAND driving *h* identical gates.





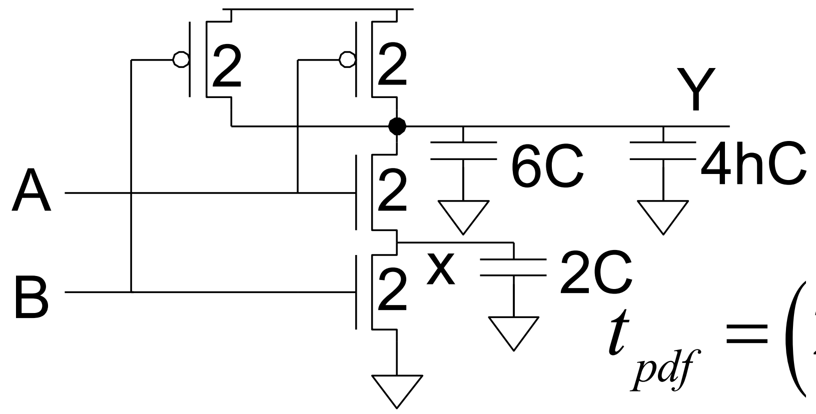


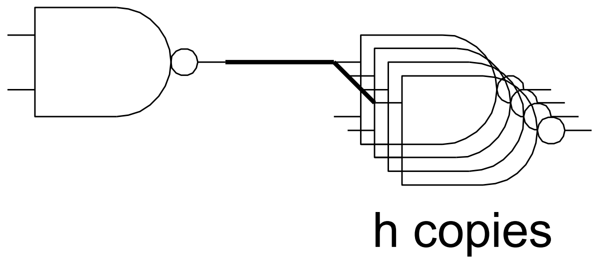
$$\boxed{t_{pdr} = \left(4h+6\right)RC}$$

This is valid for signal A, ie A switches off first.
For signal B, ie if B switches off first, node X contributes an extra 2RC delay (2C*R – resistance of in-path pMOS) which we ignore as critical signals should be connected to node A.

Recall: 2-input NAND t_{pdf}

- Estimate rising and falling propagation delays of a 2-input NAND driving *h* identical gates.

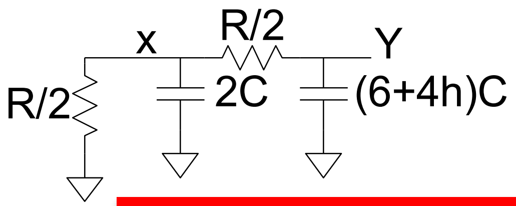




$$t_{pdf} = (2C)\left(\tfrac{R}{2}\right) + \left[(6+4h)C\right]\left(\tfrac{R}{2}+\tfrac{R}{2}\right)$$

$$= (7+4h)RC$$

Valid if signal B turns on last.

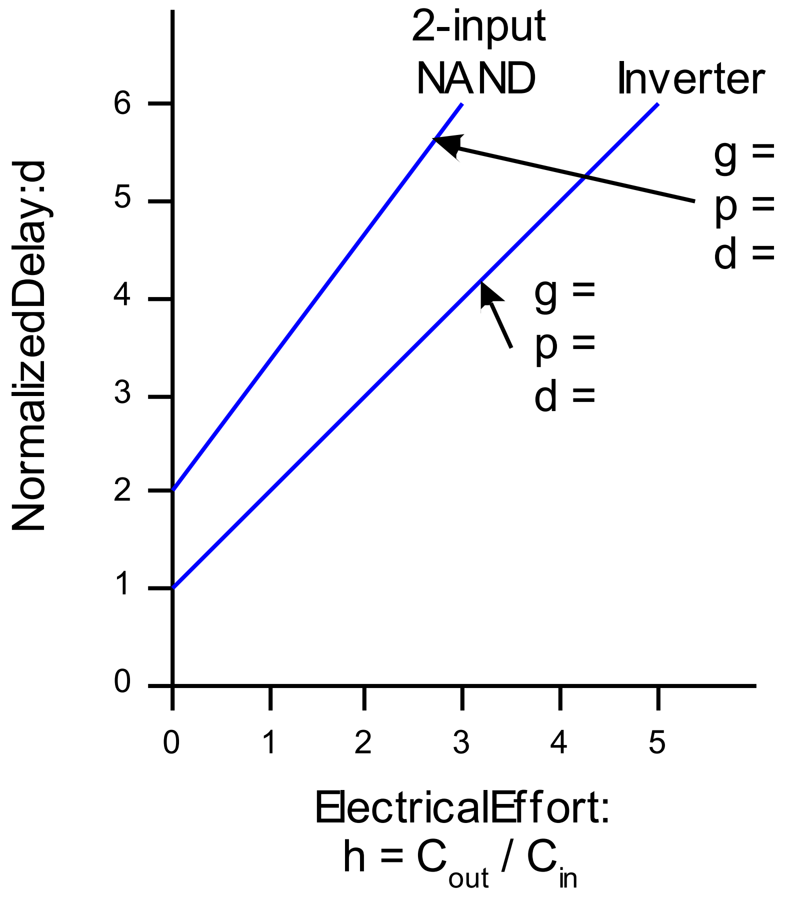


If A turns on last, node X is precharged and we ignore it: $\boxed{t_{pdf} = (4h+6)RC}$

- $\boxed{d = \left((t_{pdr} + t_{pdf})/2\right)/3RC = 4/3h + 2}$

Delay Plots

$$\begin{aligned} d &= f + p \\ &= gh + p \end{aligned}$$



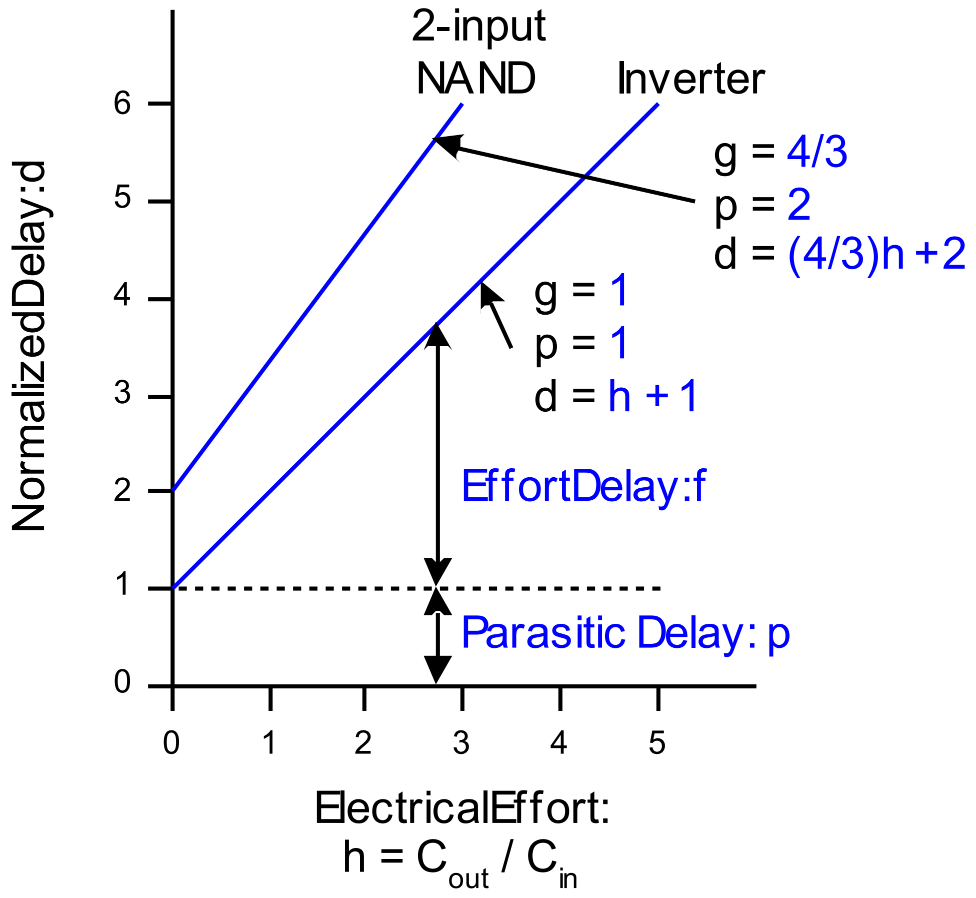
Delay Plots

$$d = f + p$$
$$= gh + p$$

- NAND2 (from past lecture & review): $d = (4h+6)RC / 3RC$
  $= (4/3)h+2$

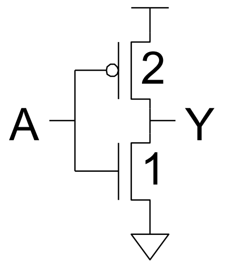
  τ

- What about NOR2?



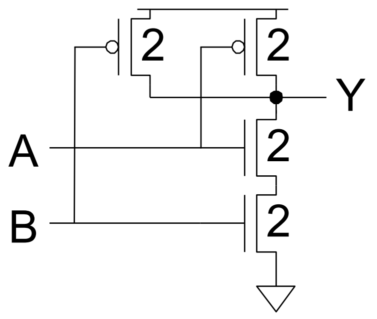
Computing Logical Effort

- 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



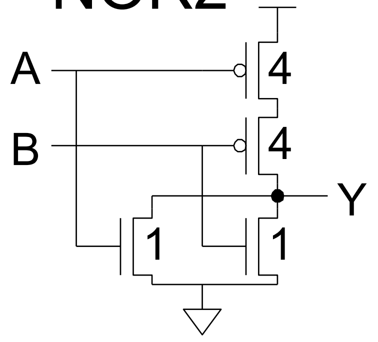
$C_{in} = 3$
$g = 3/3$

NAND2



$C_{in} = 4$
$g = 4/3$

NOR2



$C_{in} = 5$
$g = 5/3$

Catalog of Gates

- **Logical effort** of common 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
XOR, XNOR		4, 4	6, 12, 6	8, 16, 16, 8	

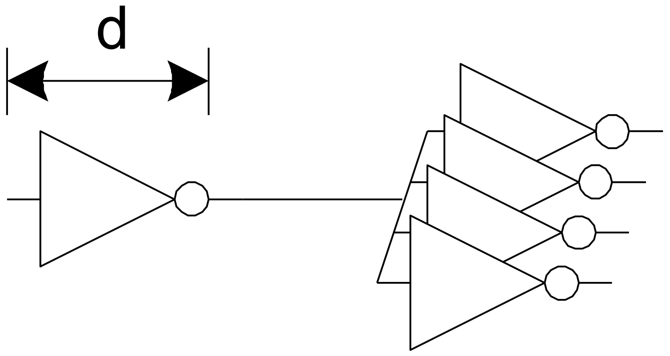
Catalog of Gates

- **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
XOR, XNOR		4	6	8	

Example: FO4 Inverter

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



Logical Effort: g =

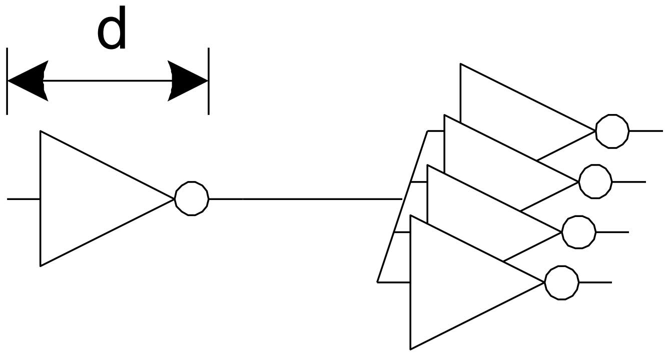
Electrical Effort: h =

Parasitic Delay: p =

Stage Delay: d =

Example: FO4 Inverter

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



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

The FO4 delay is about

200 ps in 0.6 μm process

60 ps in a 180 nm process

f/3 ns in an *f* μm process

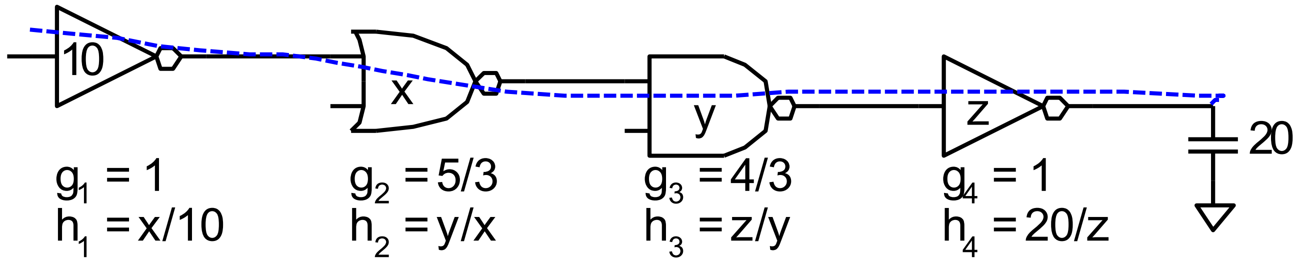
Multistage Logic Networks

- Logical effort generalizes to multistage networks
- *Path Logical Effort* $$G = \prod g_i$$
- *Path Electrical Effort* $$H = \frac{C_{out-path}}{C_{in-path}}$$
- *Path Effort* $$F = \prod f_i = \prod g_i h_i$$
- Can we write F = GH? Only if there are no branches!

Recall:

$d = f + p$

$= gh + p$



Paths that Branch

- Consider paths that branch:

$G = 1$

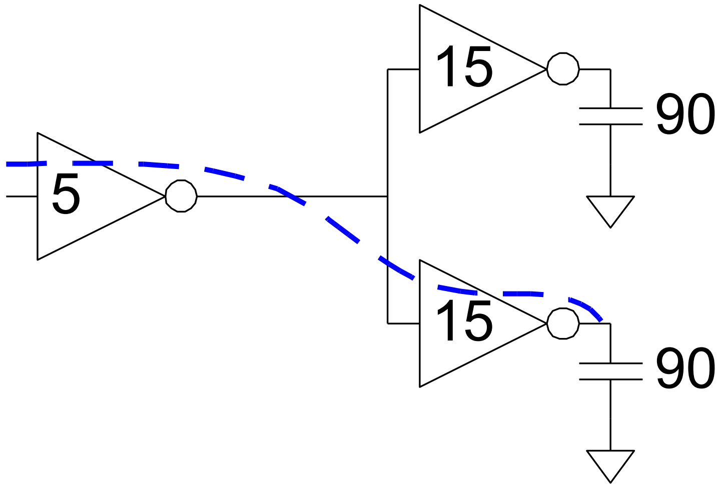
$H = 90 / 5 = 18$

$GH = 18$

$h_1 = (15 + 15) / 5 = 6$

$h_2 = 90 / 15 = 6$

$F = g_1 g_2 h_1 h_2 = 36 = 2GH$



Branching Effort

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

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

$$B = \prod b_i$$

Note:

$$\prod h_i = BH$$

- Now we compute the path effort
  - F = GBH

Multistage Delays

- Path Effort Delay $D_F = \sum f_i$
- Path Parasitic Delay $P = \sum p_i$
- Path Delay $D = \sum d_i = D_F + P$

Designing Fast Circuits

$$D = \sum d_i = D_F + P$$

- Delay is smallest when each stage bears same effort

$$\boxed{\hat{f} = g_i h_i = F^{\frac{1}{N}}}$$

- Thus minimum delay of *N*-stage path is

$$\boxed{D = NF^{\frac{1}{N}} + P}$$

- 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?

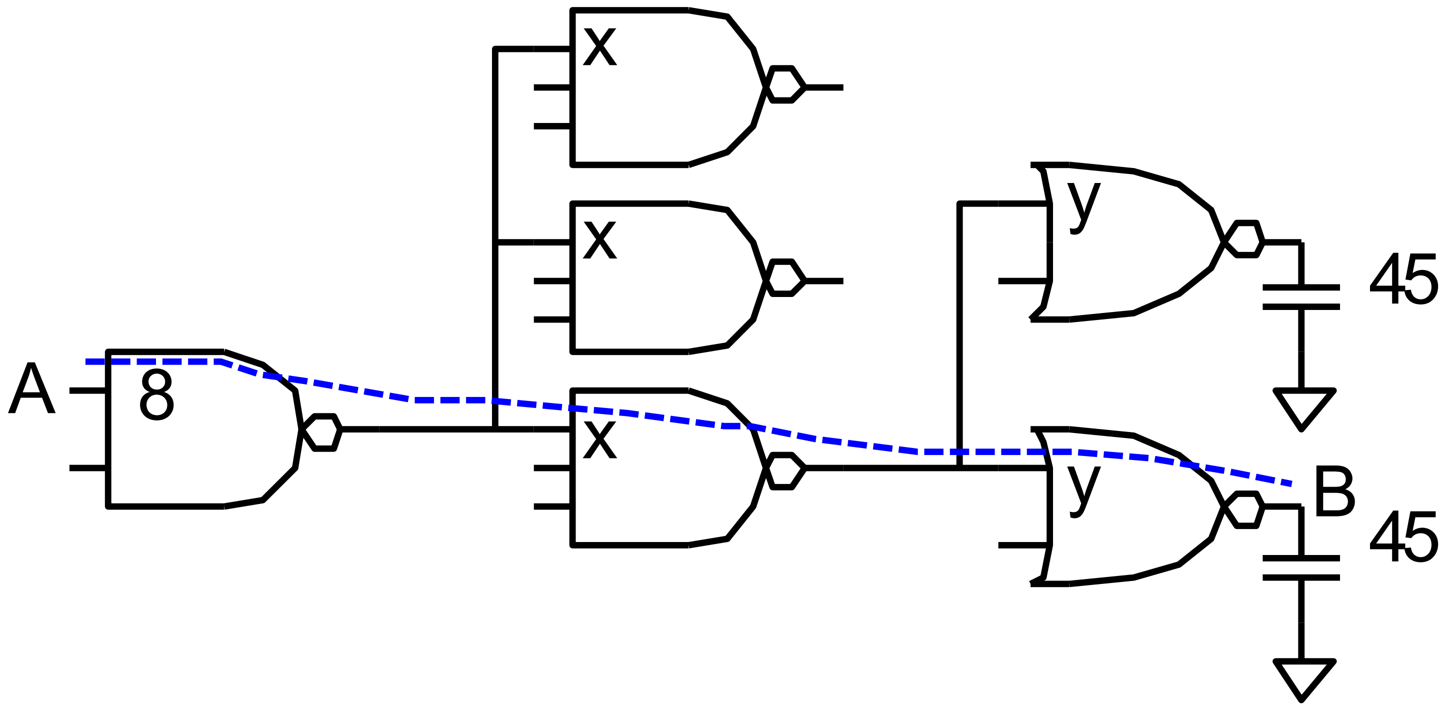
$$\hat{f} = gh = g\frac{C_{out}}{C_{in}}$$

$$\Rightarrow C_{in_i} = \frac{g_i C_{out_i}}{\hat{f}}$$

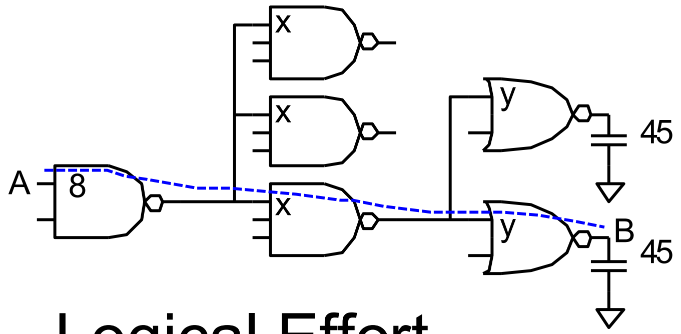
- Working backward, apply capacitance transformation to find input capacitance of each gate given the load it drives.
- Check work by verifying input cap spec is met.

Example: 3-stage path

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

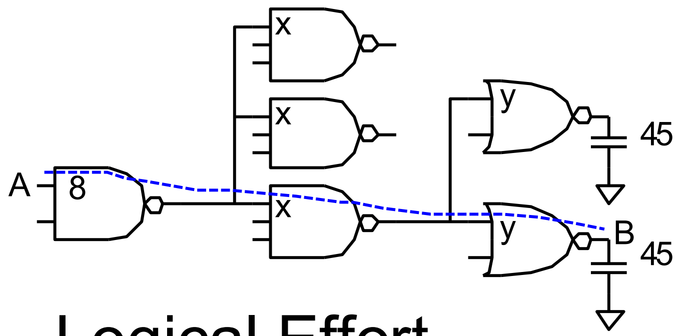


Example: 3-stage path



Logical Effort	G =
Electrical Effort	H =
Branching Effort	B =
Path Effort	F =
Best Stage Effort	$\hat{f} =$
Parasitic Delay	P =
Delay	D =

Example: 3-stage path



Logical Effort	G = (4/3)*(5/3)*(5/3) = 100/27
Electrical Effort	H = 45/8
Branching Effort	B = 3 * 2 = 6
Path Effort	F = GBH = 125
Best Stage Effort	$\hat{f} = \sqrt[3]{F} = 5$
Parasitic Delay	P = 2 + 3 + 2 = 7
Delay	D = 3*5 + 7 = 22 = 4.4 FO4

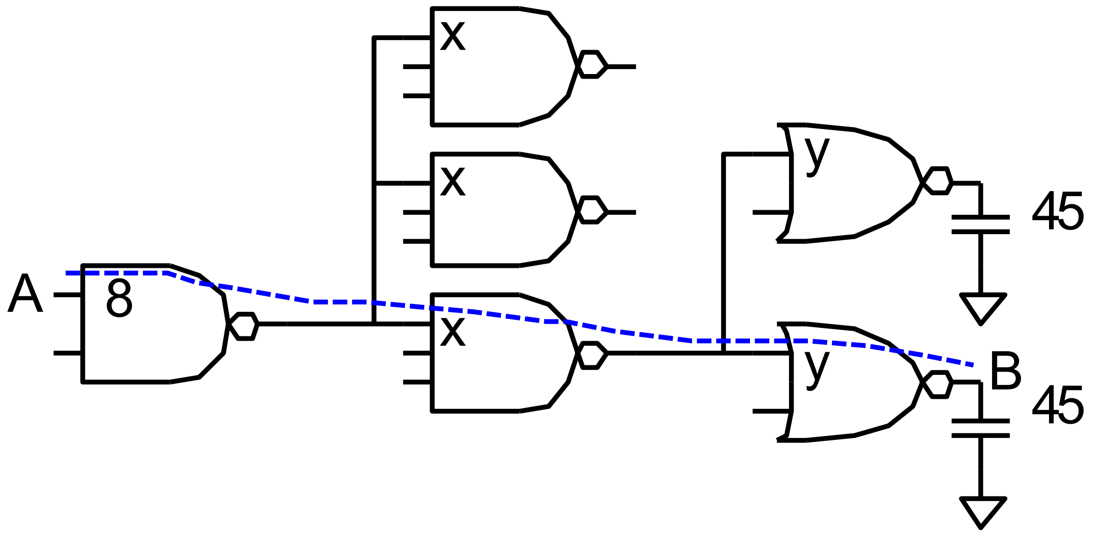
Example: 3-stage path

- Work backward for sizes, use $\hat{f} = gh = g\frac{C_{out}}{C_{in}}$

$$\Rightarrow C_{in_i} = \frac{g_i C_{out_i}}{\hat{f}}$$

y =

x =



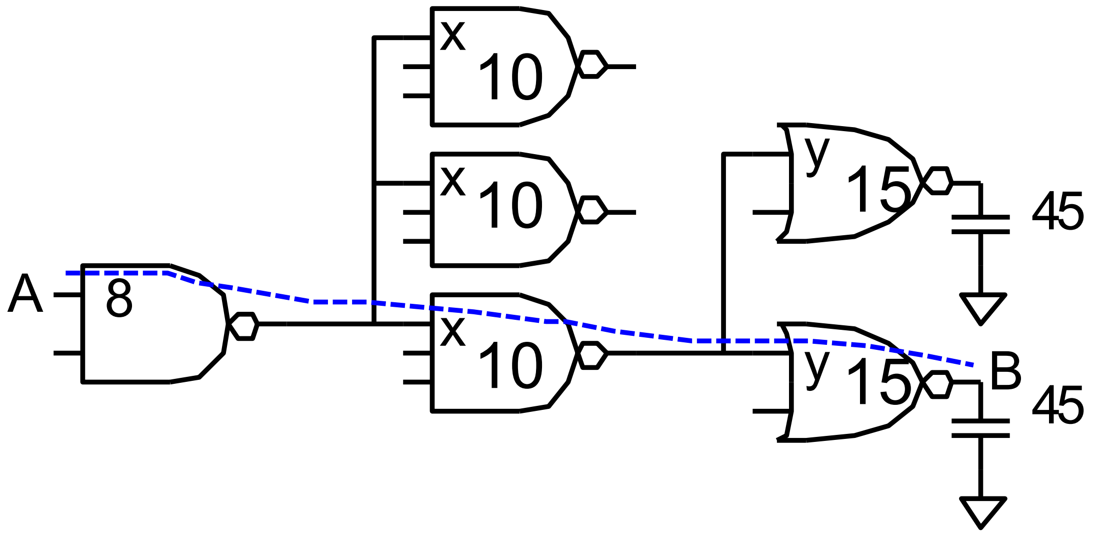
Example: 3-stage path

- Work backward for sizes, use $\hat{f} = gh = g\frac{C_{out}}{C_{in}}$

y = (5/3) * 45 / 5 = 15

x = (5/3) * (15*2) / 5 = 10

$$\Rightarrow C_{in_i} = \frac{g_i C_{out_i}}{\hat{f}}$$

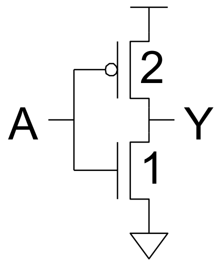


Example: 3-stage path

- Work backward for sizes

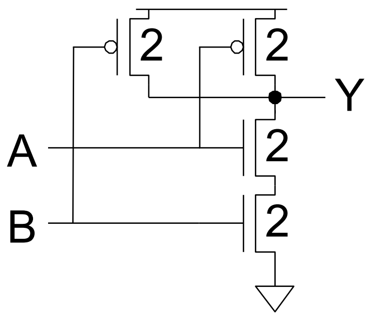
  y = (5/3) * 45 / 5 = 15

  x = (5/3) * (15*2) / 5 = 10



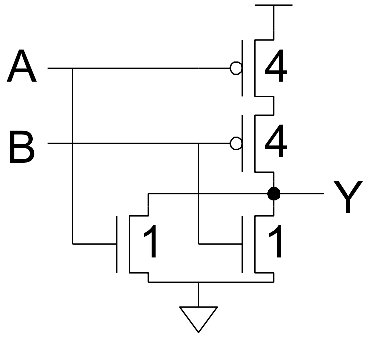
$C_{in} = 3$
g = 3/3

Wp/Wn=1



$C_{in} = 4$
g = 4/3

Wp/Wn=4



$C_{in} = 5$
g = 5/3

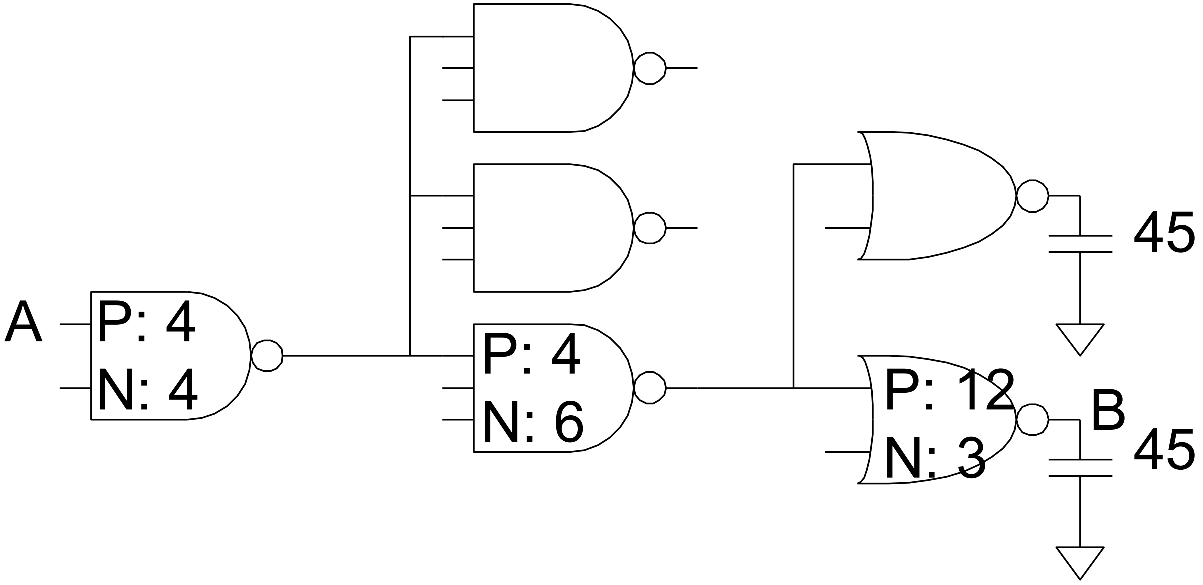
Two equations and two unknowns – solve for Wp and Wn

Example: 3-stage path

- Work backward for sizes

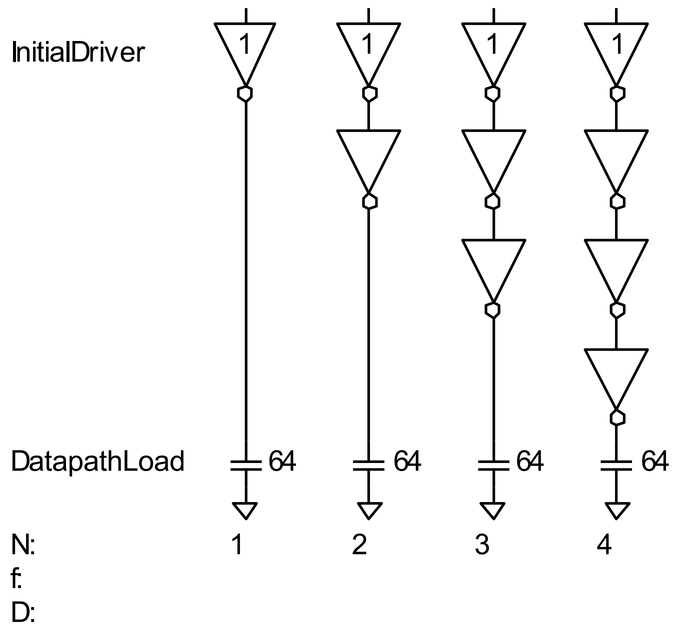
  $y = (5/3) * 45 / 5 = 15$

  $x = (5/3) * (15*2) / 5 = 10$



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 (F=64/1=64)

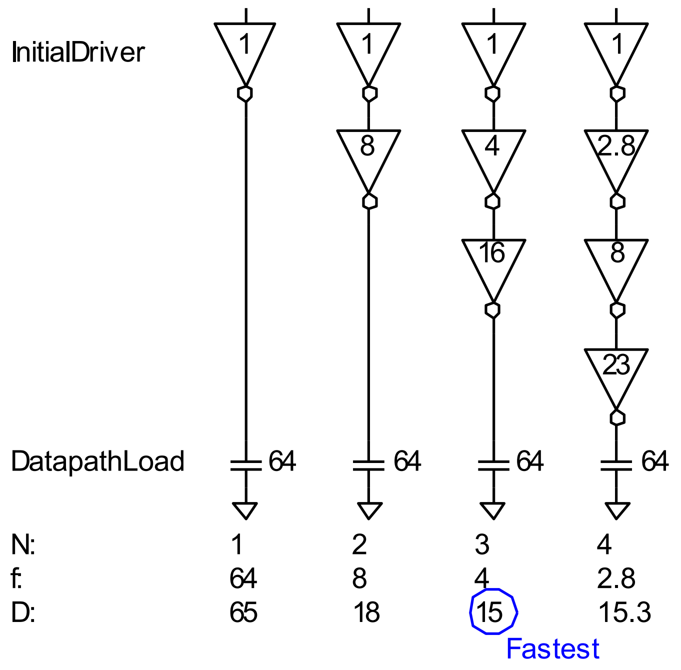


D =

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 (F=64/1=64)

$$D = NF^{1/N} + P$$
$$= N(64)^{1/N} + N$$



Derivation

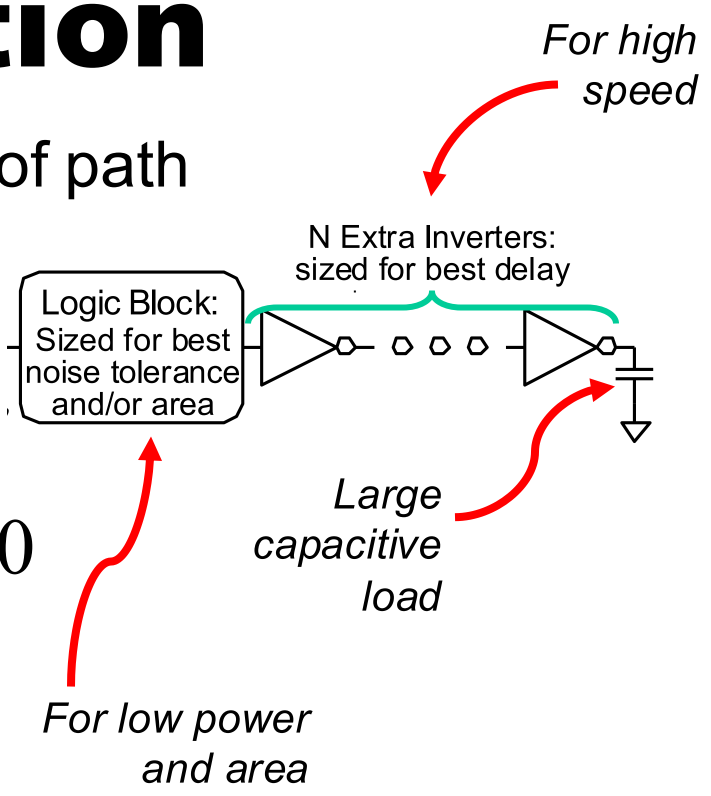
- Consider adding inverters to end of path
  - How many give least delay?

$$D = NF^{\frac{1}{N}} + Np_{inv}$$

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

- Define best stage effort $\rho = F^{\frac{1}{N}}$

$$p_{inv} + \rho\left(1 - \ln \rho\right) = 0$$


For high speed
N Extra Inverters: sized for best delay
Logic Block: Sized for best noise tolerance and/or area
Large capacitive load
For low power and area

- This is the approach preferred in practice (to save power):
  - Logic: sized for best noise tolerance, area (min-size NMOS)
  - Followed by inverter chain: sized for minimal delay

Best Stage Effort

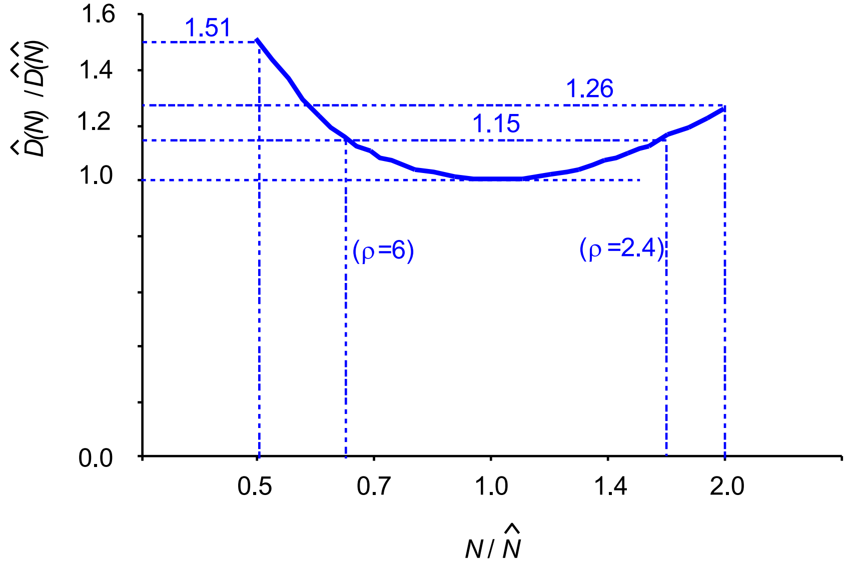
- $p_{inv} + \rho\left(1 - \ln \rho\right) = 0$ has no closed-form solution

- Neglecting parasitics (p_{inv} = 0), we find ρ = 2.718 (e)

- For p_{inv} = 1, solve numerically for ρ = 3.59

Sensitivity Analysis

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



- $2.4 < \rho < 6$ gives delay within 15% of optimal
  - We can round up from $\rho = 3.59$ to $\rho = 4$
    - accounts for wire parasitics etc.

Method of Logical Effort

1) Compute path effort $F = GBH$
2) Estimate best number of stages $N = \log_4 F$
3) Sketch path with N stages
4) Estimate least delay $D = NF^{\frac{1}{N}} + P$
5) Determine best stage effort $\hat{f} = F^{\frac{1}{N}}$

6) Find gate sizes $C_{in_i} = \dfrac{g_i C_{out_i}}{\hat{f}}$

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

- Logical effort is useful for thinking of delay in circuits
  - 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
  - But using fewer stages doesn't mean faster paths
  - Delay of path is about $\log_4 F$ FO4 inverter delays
  - Inverters (and NAND2) best for driving large caps
- Provides language for discussing fast circuits
  - An academic method that helps develop intuition
  - In industry - used only for inverters, not other gates