Introduction to CMOS VLSI Design

Chapter 4
Delay

thực tế trên mạch, về mặt tín hiệu, tín hiệu ngõ vào sẽ có tín hiệu ngõ ra tương ứng sau 1 khoảng thời gian nào đó nhất định, tức là có độ trễ
xét module logic hay module inverter có độ trễ rất nhỏ, vài chục ns, nhưng khi xem xét 1 con chip lớn có số lượng transistor tích hợp trên đó là cực kì lớn hoặc là con chip gồm nhiều module nhỏ ở trong, thì số lượng delay tích lũy cực kì lớn
delay có nhiều dạng, ảnh hưởng nhiều nhất giữa ngõ vào và ngõ ra, ngta sẽ tính toán và gọi là propagation delay, thời gian delay này sẽ ảnh hưởng đến cái tần số hoạt động tối đa con chip của mình
f max: tần số hoạt động tối đa mà con chip có thể đáp ứng được

timing diagram: dạng sóng giản đồ thời gian thể hiện tín hiệu ngõ vào và tín hiệu ngõ ra, mối quan hệ như thế nào

Delay Definitions

độ trễ truyền sóng xét theo cạnh lên

- t_{pdr}: *rising propagation delay*
  - From input to rising output crossing $V_{DD}/2$

độ trễ truyền sóng xét theo cạnh xuống

- t_{pdf}: *falling propagation delay*
  - From input to falling output crossing $V_{DD}/2$

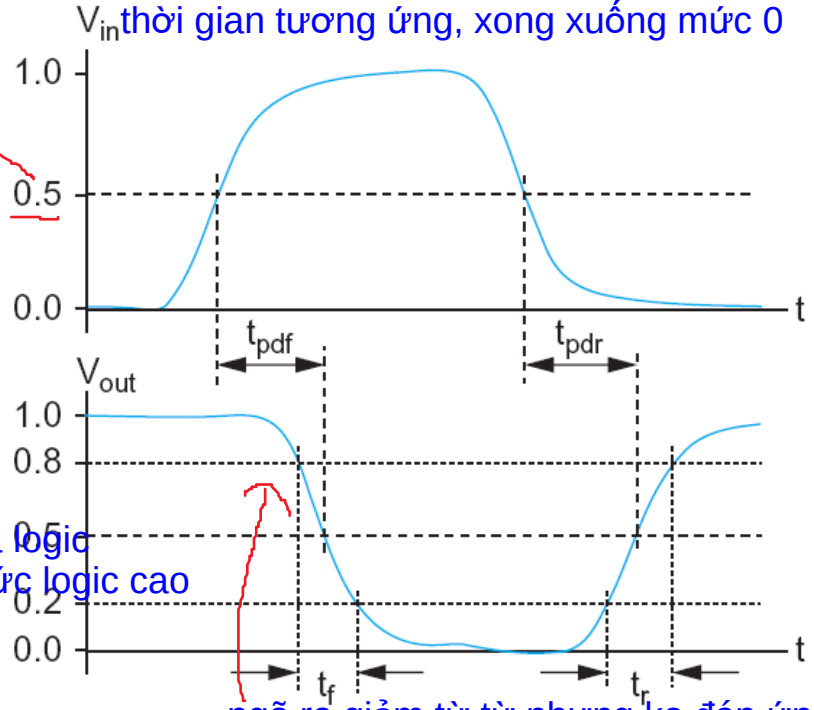
độ trễ truyền sóng trung bình

- t_{pd}: *average propagation delay*
  - $t_{pd} = (t_{pdr} + t_{pdf})/2$

để chính xác lấy tb

- t_r: *rise time* khoảng thời gian từ lúc nó là logic mức thấp đến khi xác lập mức logic cao
  - From output crossing 0.2 V_{DD} to 0.8 V_{DD}
- t_f: *fall time* khoảng thời gian từ lúc nó là logic mức cao đến khi xác lập mức logic thấp
  - From output crossing 0.8 V_{DD} to 0.2 V_{DD}

điện áp mức 0, muốn cung cấp điện áp mức logic 1, quá trình tăng lên từ từ, trong khoảng thời gian tương ứng, xong xuống mức 0



ngõ ra giảm từ từ nhưng ko đáp ứng ngay lập tức, có độ trễ truyền sóng nhất định Inverter

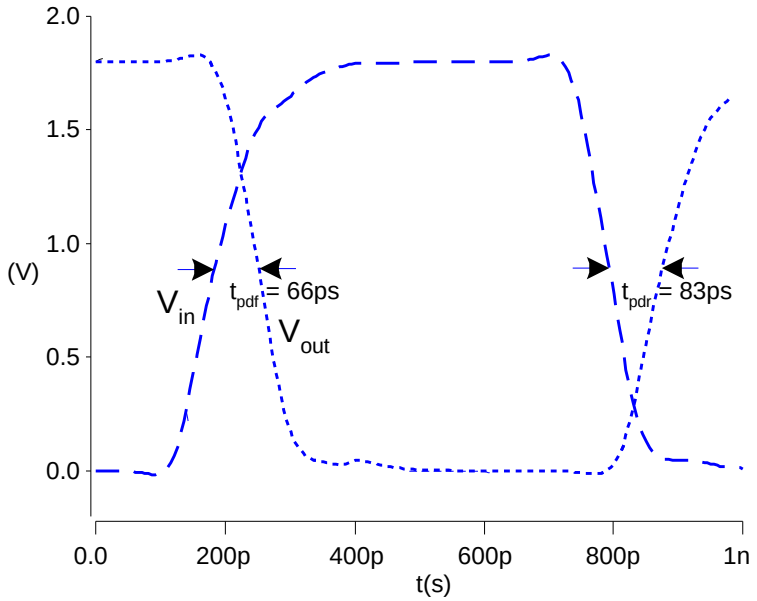
để sau

Delay Definitions

- **t_{cdr}**: *rising contamination delay*
  - From input to rising output crossing $V_{DD}/2$
- **t_{cdf}**: *falling contamination delay*
  - From input to falling output crossing $V_{DD}/2$
- **t_{cd}**: *contamination delay*
  - $t_{cd} = (t_{cdr} + t_{cdf})/2$??
  - $t_{cd} = \min(t_{cdr}, t_{cdf})$

Simulated Inverter Delay

- Solving differential equations by hand is too hard
- SPICE simulator solves the equations numerically
  - Uses more accurate I-V models too!
- But simulations take time to write, may hide insight



minh họa timing lệch như thế nào

Delay Estimation

- We would like to be able to easily estimate delay
  - Not as accurate as simulation
  - But easier to ask “What if?”
- The step response usually looks like a 1st order RC response with a decaying exponential.
- Use RC delay models to estimate delay
  - C = total capacitance on output node
  - Use *effective resistance* R
  - So that t_{pd} = RC
- Characterize transistors by finding their effective R
  - Depends on average current as gate switches

Effective Resistance

- Shockley models have limited value
  - Not accurate enough for modern transistors
  - Too complicated for much hand analysis
- Simplification: treat transistor as resistor coi cmos như những cái điện trở
  - Replace $I_{ds}(V_{ds}, V_{gs})$ with effective resistance R
    - $I_{ds} = V_{ds}/R$
  - R averaged across switching of digital gate
- Too inaccurate to predict current at any given time
  - But good enough to predict RC delay

RC Delay Model

dùng mạch tương đương để thể hiện transistor MOS

- Use equivalent circuits for MOS transistors
  - Ideal switch + capacitance and ON resistance

switch lí tưởng, on hoặc off điện dung điện trở

nếu 2 cái này đều xem xét ở kích thước giống nhau

  - Unit nMOS has resistance R, capacitance C

độ linh động thấp hơn 2 lần, độ dẫn điện thấp hơn 2 lần, suy ra điện trở cao hơn 2 lần

  - Unit pMOS has resistance 2R (lower carrier mobility), capacitance C

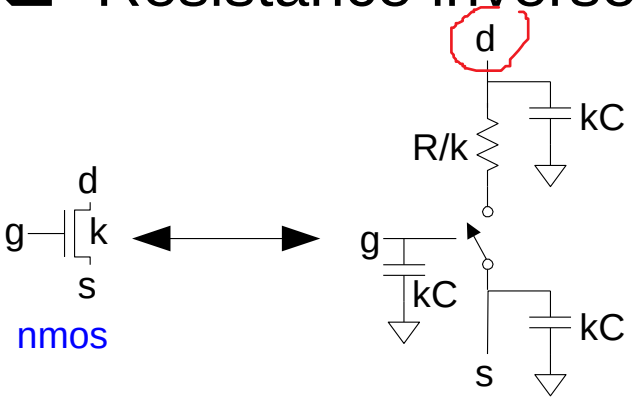
slide 17 ch02

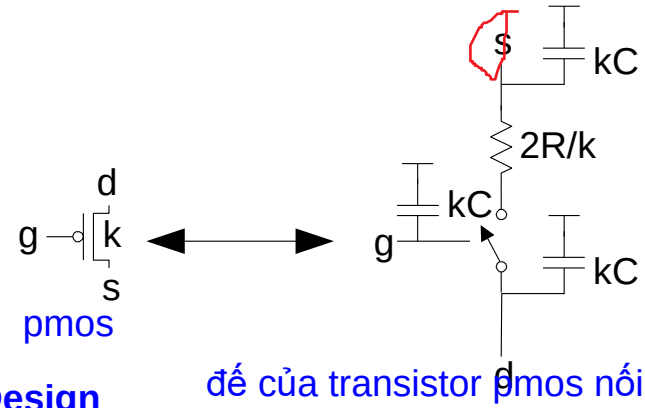
- Capacitance proportional to width (k)

điện dung tương đương tỉ lệ thuận với kích thước transistor, kích thước tăng lên k lần, điện dung tăng lên k lần

- Resistance inversely proportional to width

điện trở tỉ lệ nghịch với kích thước, khi kích thước tăng lên, số hạt mang điện tăng lên, dòng điện cũng sẽ tăng lên tương ứng, suy ra điện trở giảm đi





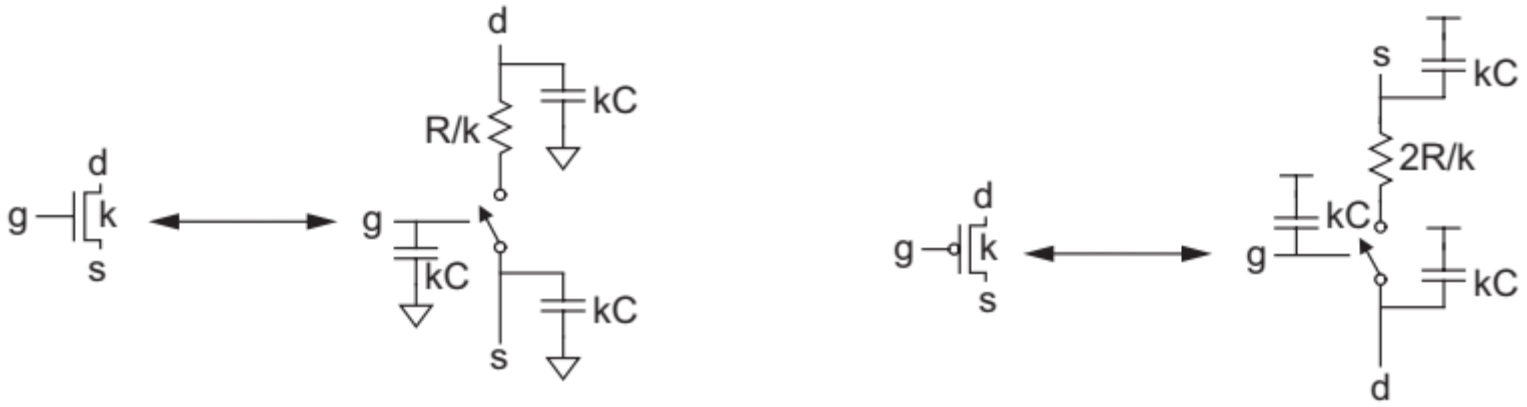
3 cái tụ kí sinh nối từ g xuống cái body, body nối GND

đế của transistor pmos nối với Vdd, điện dung kí sinh tính từ cái cực đến Vdd

RC Values

- Capacitance
  - $C = C_g = C_s = C_d = 2$ fF/μm of gate width in 0.6 μm
  - Gradually decline to 1 fF/μm in nanometer techs. (
  (1 fF = 1.0E-15 F)
- Resistance
  - $R \approx 6$ KΩ*μm in 0.6 μm process
  - Improves with shorter channel lengths
- Unit transistors
  - May refer to minimum contacted device (4/2 λ)
  - Or maybe 1 μm wide device
  - Doesn't matter as long as you are consistent

Equivalent RC circuits



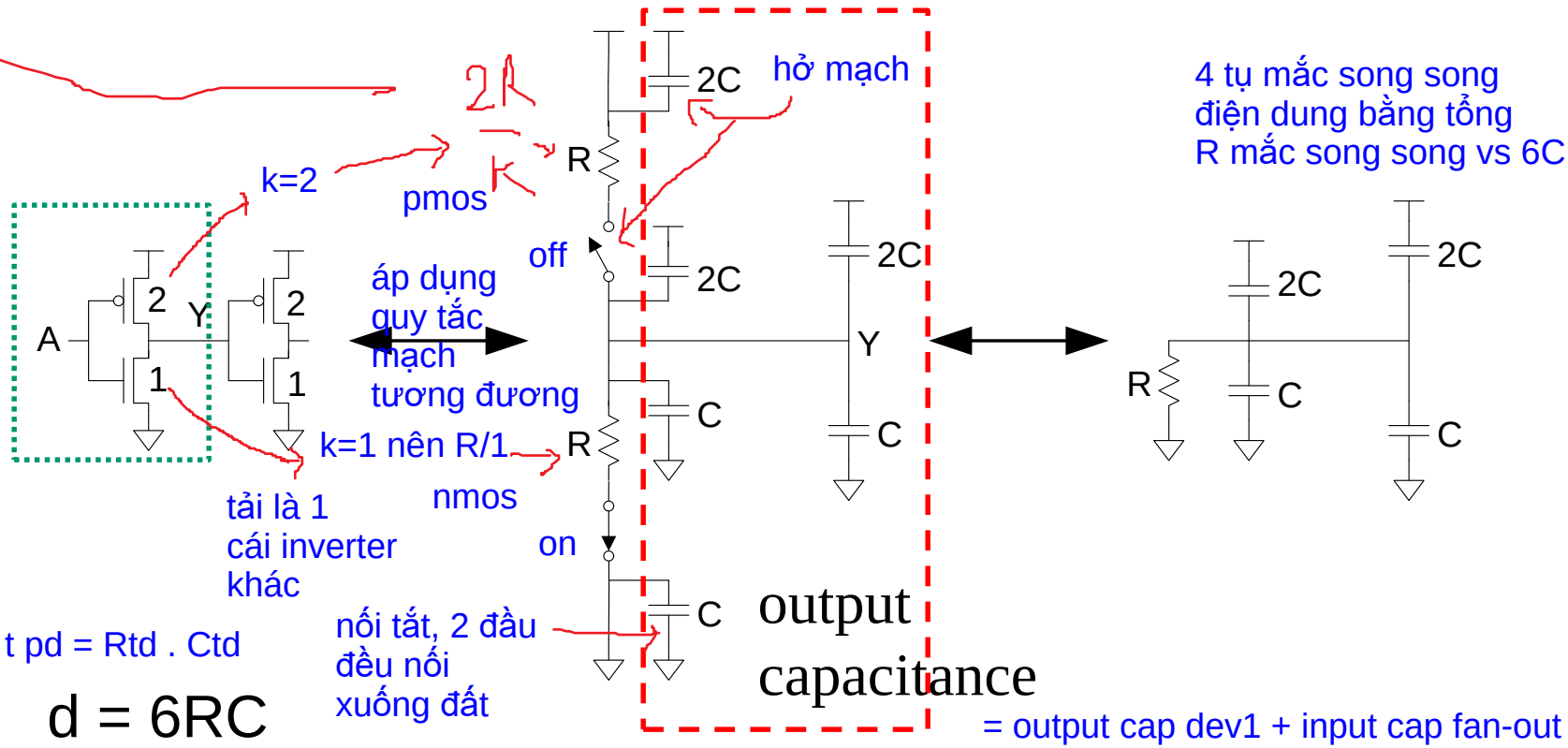
do Y ko phụ thuộc vào điện dung Cg nên bỏ đi cho gọn, do Y ngõ ra nằm ở cực g của mạch kế tiếp, do đó xem xét Cg ở tầng fanout

fanout 2 inverter and or ...

Inverter Delay Estimate

khi tính toán delay, delay phụ thuộc vào cái tải

- Estimate the delay of a fanout-of-1 inverter
- A=1, consider output capacitance

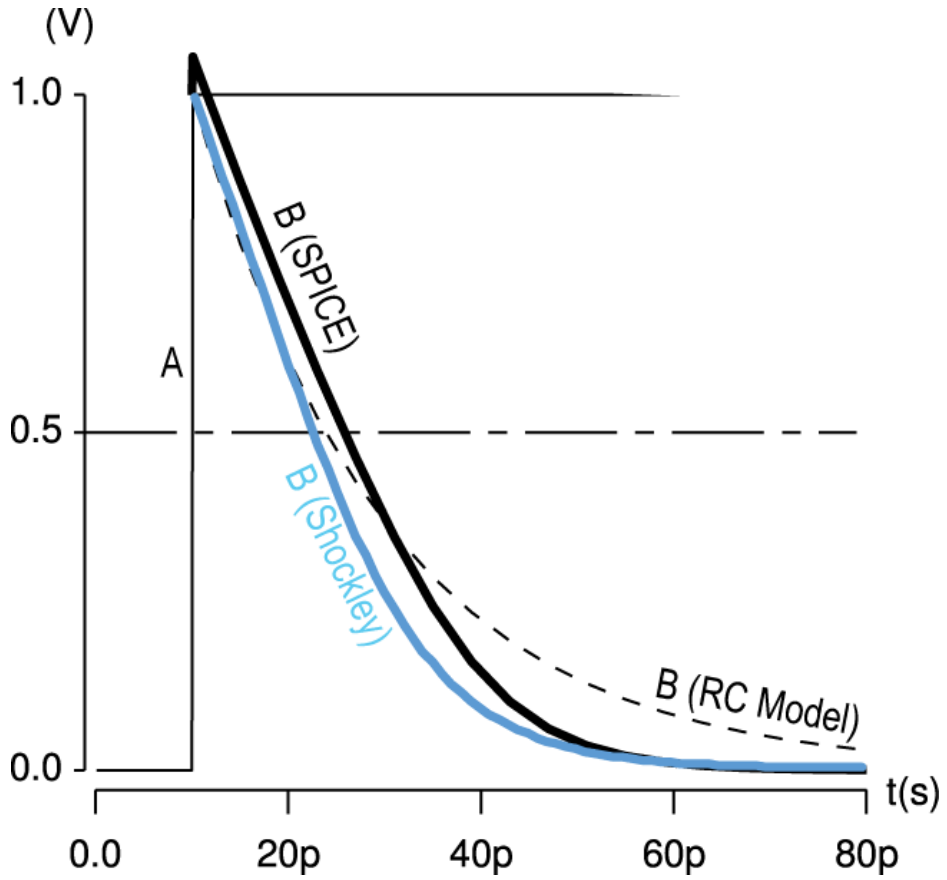


t pd = Rtd . Ctd

d = 6RC

= output cap dev1 + input cap fan-out dev

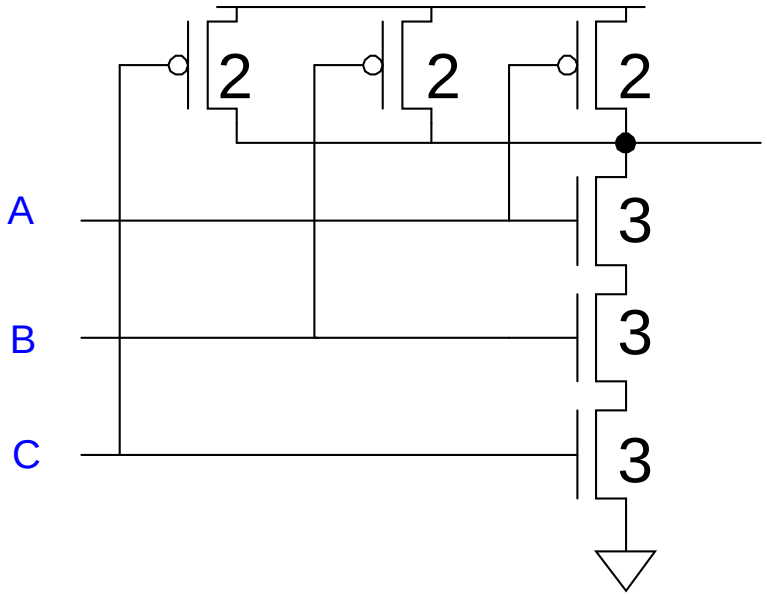
Delay Model Comparison (Example 4.1, p.145)



Example: 3-input NAND

phác họa

- Sketch a 3-input NAND with transistor widths chosen to achieve effective rise and fall resistances equal to a unit inverter (R).

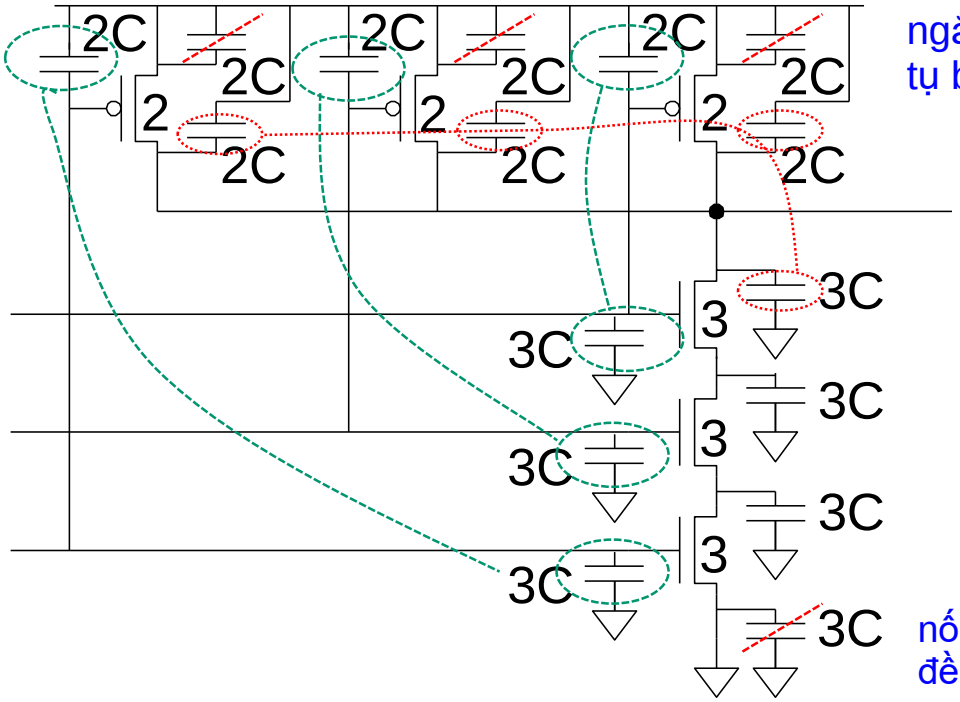


3-input NAND Caps

chú thích

- Annotate the 3-input NAND gate with gate and diffusion capacitance.

Các tụ điện mắc song song thì có điện dung tương đương bằng tổng điện dung của các tụ cộng lại

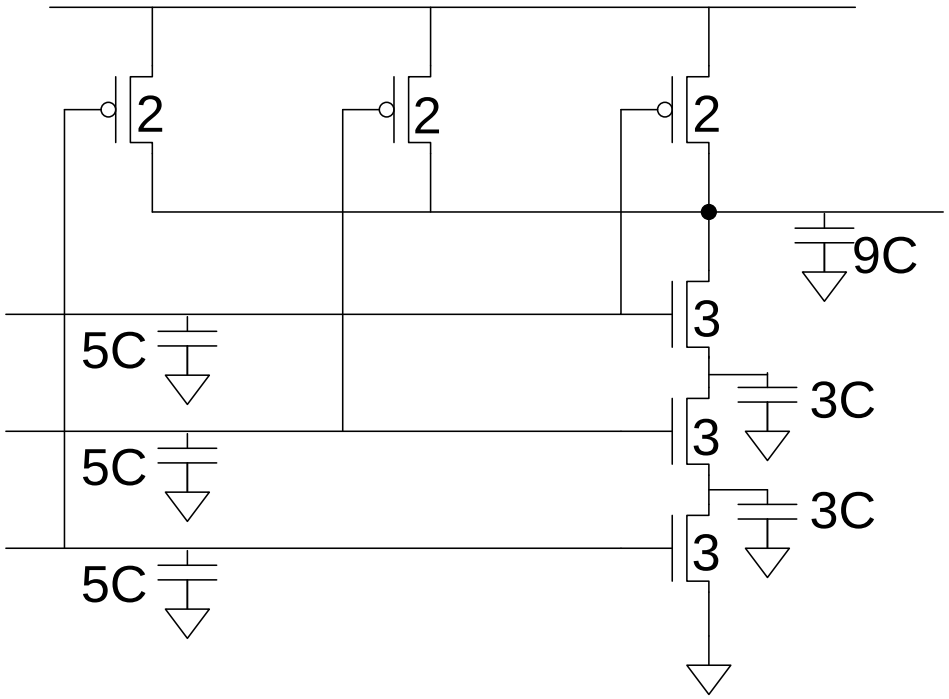


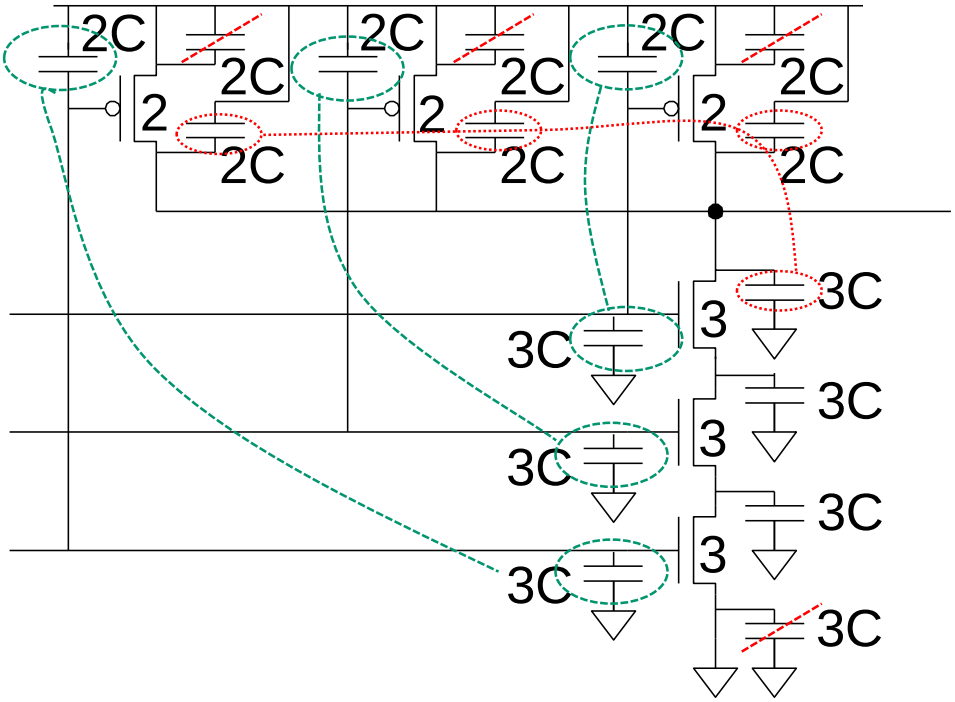
ngắn mạch do điện áp 2 đầu tụ bằng nhau

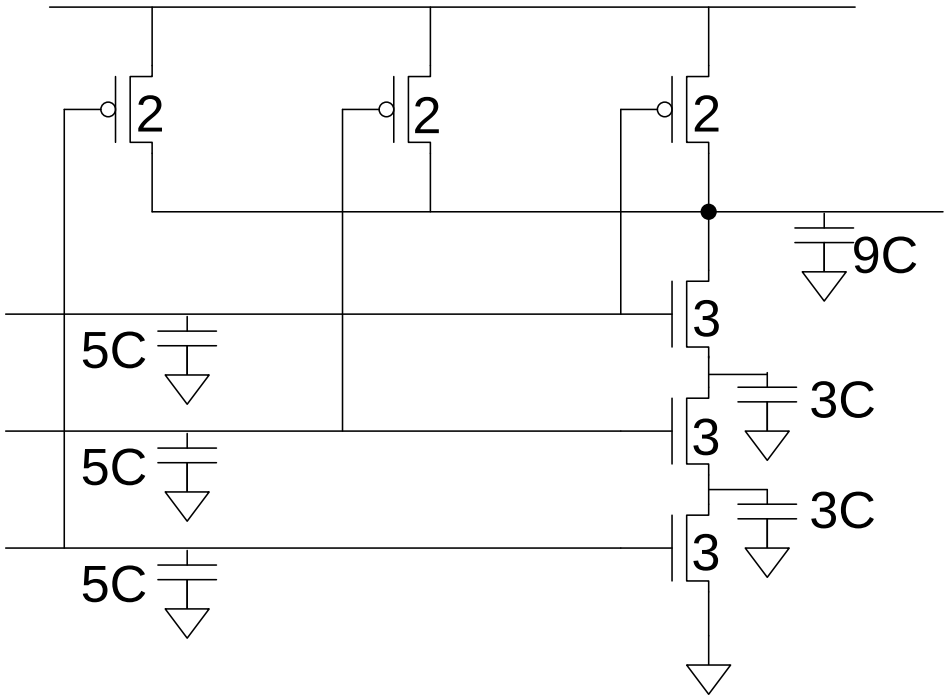
nối tắt, 2 đầu đều nối xuống đất

3-input NAND Caps

- Annotate the 3-input NAND gate with gate and diffusion capacitance.




2C
2C
2C
2C
2C
2C
2
2
2
2C
2C
2C
3C
3C
3C
3C
3C
3C
3
3
3


2
2
2
9C
5C
5C
5C
3C
3C
3
3
3

Elmore Delay

tụ điện đầu ra

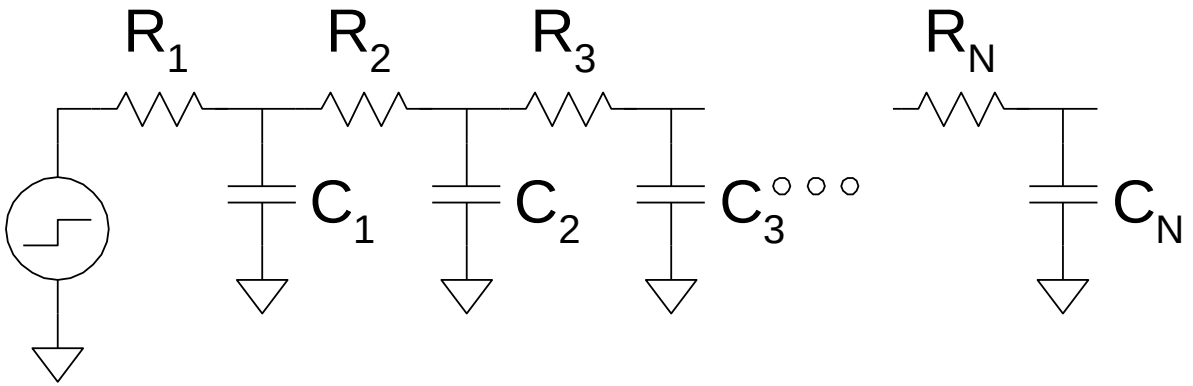
- ON transistors look like resistors (output capacitor)

kéo lên kéo xuống

- Pullup or pulldown network modeled as *RC ladder*
- Elmore delay of RC ladder

$$t_{pd} \approx \sum_{\text{nodes } i} R_{i-to-source} C_i$$

$$= R_1 C_1 + (R_1 + R_2) C_2 + \ldots + (R_1 + R_2 + \ldots + R_N) C_N$$

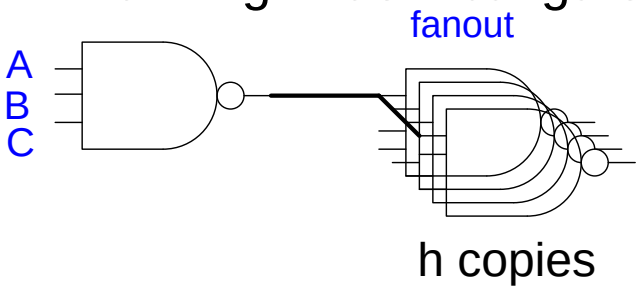


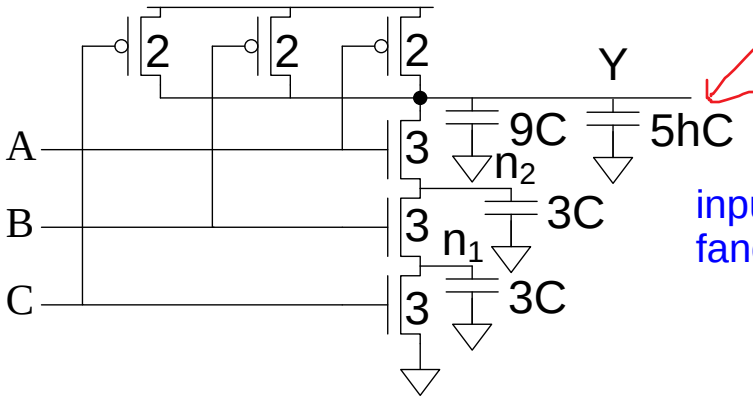
khi tính toán điện dung để ra cái điện dung tương đương, ngta bỏ đi Cg , do Y ngõ ra nằm ở cực g của mạch kế tiếp, do đó xem xét Cg ở tầng fanout

Example: 3-input NAND

ngõ ra NAND 3-input nối với h cổng giống nhau

- Estimate worst-case rising and falling delay of 3-input NAND driving *h* identical gates.

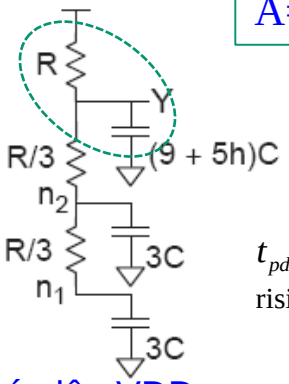




input cap of 1 fanout x h

bởi vì đầu ra chỉ được nạp điện qua R

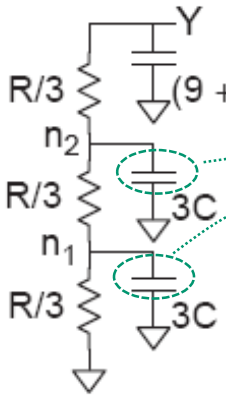
A=1, B=1, C=0



$t_{pdr} = (9+5h)RC$

rising output (nạp tụ)

A=1, B=1, C=1



$t_{pdf} = (3C)\left(\frac{R}{3}\right)+(3C)\left(\frac{R}{3}+\frac{R}{3}\right)+\left[(9+5h)C\right]\left(\frac{R}{3}+\frac{R}{3}+\frac{R}{3}\right)$

$= (11+5h)RC$

12

falling output (xả tụ)

Y được kéo lên VDD thông qua một transistor pMOS

Đầu ra kéo xuống thông qua ba transistor nMOS nối tiếp

Delay Components

- Delay has two parts
  - *Parasitic delay* (ký sinh)
    - 9 or 11 RC (15 / 12)
    - Independent of load (fanout)
  - *Effort delay*
    - 5h RC
    - Proportional to load capacitance

delay ký sinh là thời gian để một cổng điều khiển điện dung khuếch tán bên trong của chính nó. Tăng chiều rộng của transistor làm giảm điện trở nhưng tăng điện dung nên delay ký sinh là độc lập với kích thước cổng

delay nỗ lực phụ thuộc vào tỷ lệ h của điện dung tải bên ngoài đến điện dung đầu vào và do đó thay đổi theo độ rộng của transistor. Nó cũng phụ thuộc vào độ phức tạp của cổng

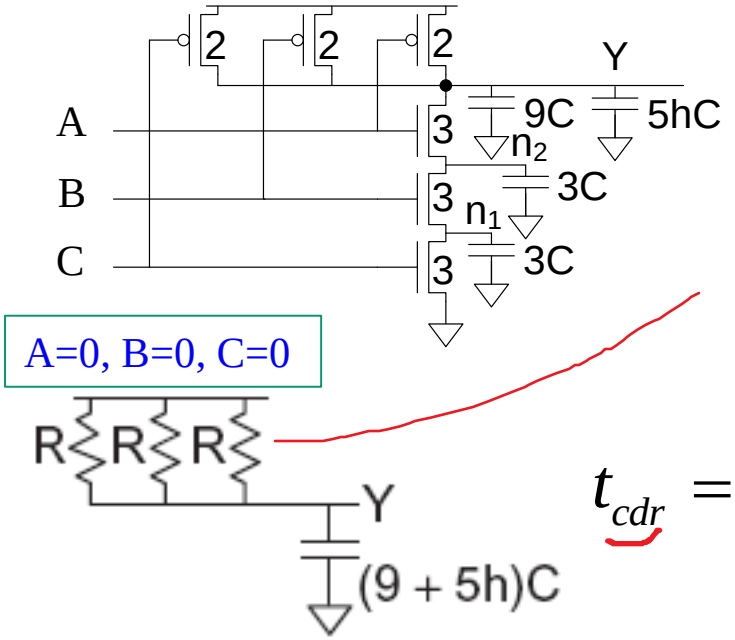
Tỷ lệ với điện dung tải

ô nhiễm

Contamination Delay

delay ô nhiễm là delay nhanh nhất mà cổng có thể chuyển

- Best-case (contamination) delay (minimum delay) can be substantially less than propagation delay.
  đáng kể truyền sóng
- Ex: If all three inputs fall simultaneously (rising output)



rising output (nạp tụ)

$$\frac{1}{R_{td}} = \frac{1}{R} + \frac{1}{R} + \frac{1}{R} = \frac{3}{R} \Leftrightarrow R_{td} = \frac{R}{3}$$

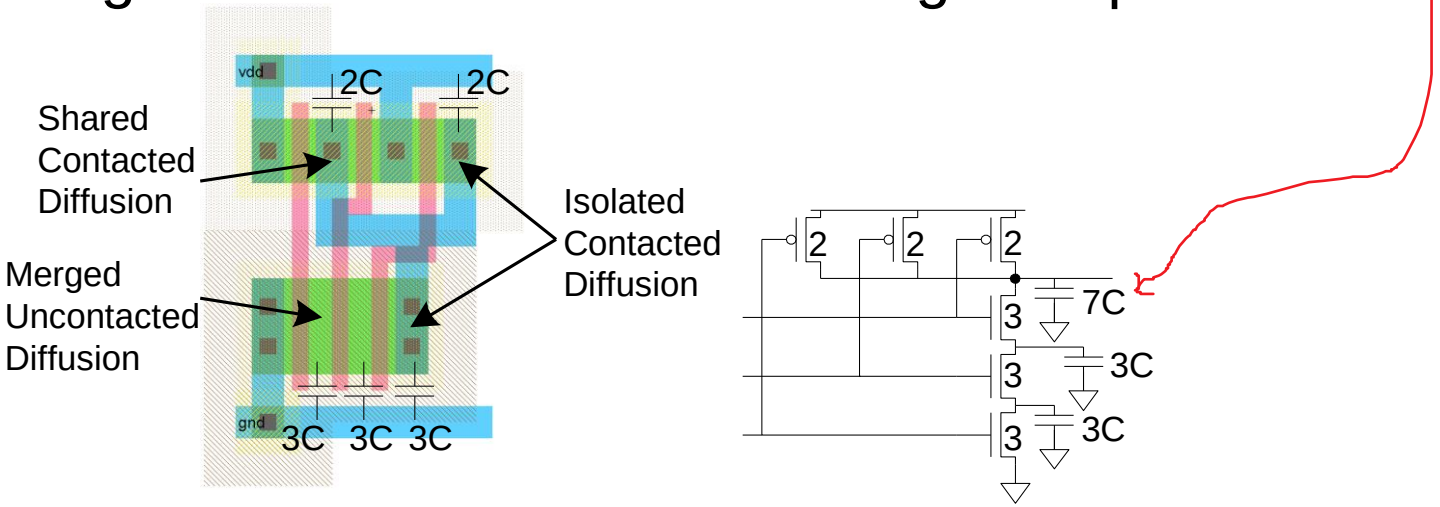
$$t_{cdr} = \left[(9+5h)C\right]\left(\frac{R}{3}\right) = \left(3+\frac{5}{3}h\right)RC$$

Đối với quá trình chuyển đổi gia tăng, trường hợp tốt nhất là cả ba transistor pMOS đều bật đồng thời. Các transistor nMOS tắt, vì vậy n1 và n2 không được kết nối với ngõ ra và không góp phần làm delay

Diffusion Capacitance

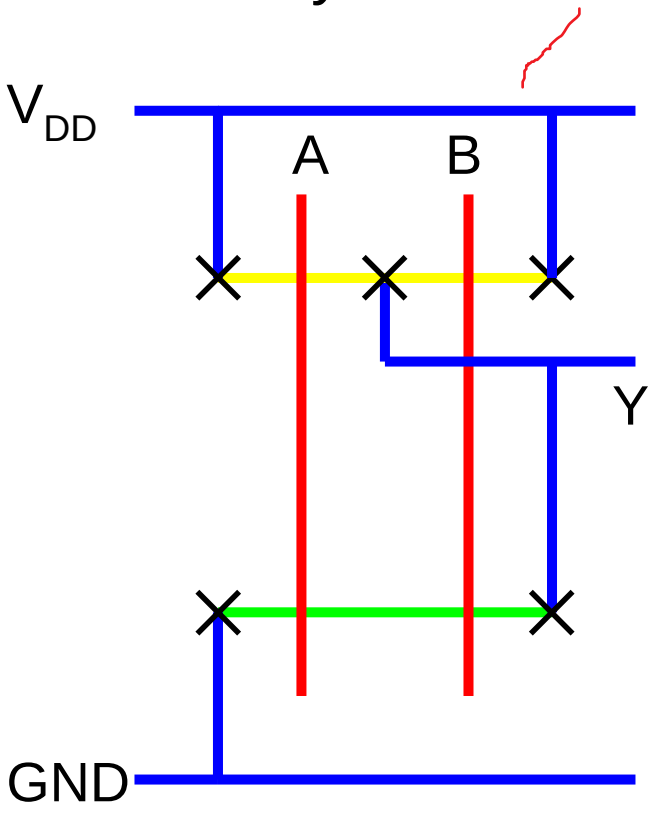
source/ drain

- We assumed contacted diffusion on every s / d.
- Good layout minimizes diffusion area
- Ex: NAND3 layout shares one diffusion contact
  - Reduces output capacitance by 2C giảm điện dung ngõ ra 2C
  - Merged uncontacted diffusion might help too

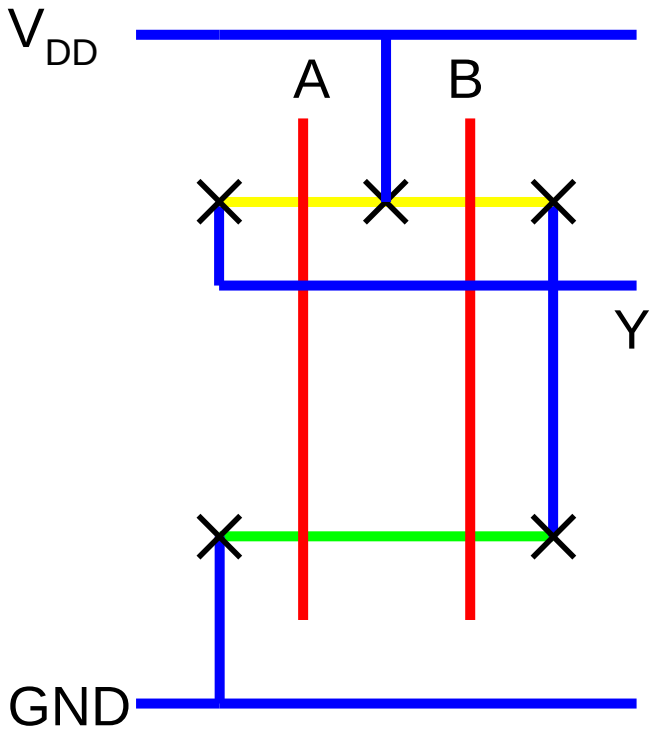


Layout Comparison

- Which layout is better?



điểm kết nối ít hơn



Logical Effort Review

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

Phép tính back-of-the-envelope là một phép tính thô, thường được ghi nhanh trên bất kỳ mẩu giấy vụn nào có sẵn như phong bì. Nó hơn một phỏng đoán nhưng ít chuẩn hơn một phép tính chính xác hoặc bằng chứng toán học. Đặc điểm xác định của các phép tính back-of-the-envelope bao gồm việc sử dụng các giả định đã được đơn giản hóa

Introduction

một loạt các lựa chọn hoang mang

- 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?

cấu trúc liên kết

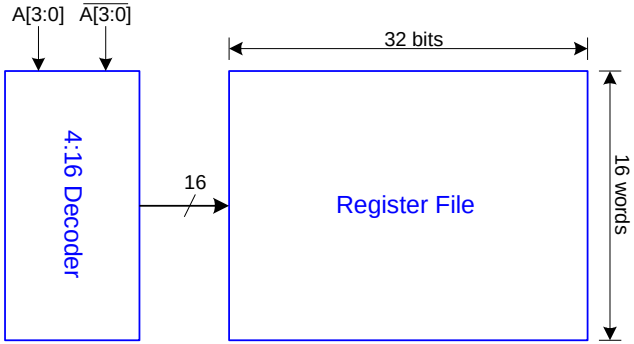
- Logical effort is a method to make these decisions
  - Uses a simple model of delay
  - Allows back-of-the-envelope calculations
  - Helps make rapid comparisons between alternatives
  - Emphasizes remarkable symmetries

lựa chọn thay thế

sự đối xứng

Example

- A memory designer for an embedded automotive processor. Help 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] bổ sung
  - Each input may drive 10 unit-sized transistors
- needs to decide:
  - How many stages to use?
  - How large should each gate be?
  - How fast can decoder operate?

Delay in a Logic Gate

độc lập với quy trình

- Express delays in process-independent unit $d = \frac{d_{abs}}{\tau}$
- Delay has two components: d = f + p
- *f*: *effort delay* = *gh* (a.k.a. stage effort)
  - Again has two components
- *g*: *logical effort*

tau
τ = 3RC
≈ 3 ps in 65 nm process
60 ps in 0.6 μm process

Đo khả năng cung cấp dòng điện tương đối của cổng

  - Measures relative ability of gate to deliver current
  - $g \equiv 1$ for inverter
- *h*: *electrical effort (or fanout)* = C_{out} / C_{in}
  - Ratio of output to input capacitance
  - Sometimes called fanout
- *p*: parasitic delay (normally ~1)

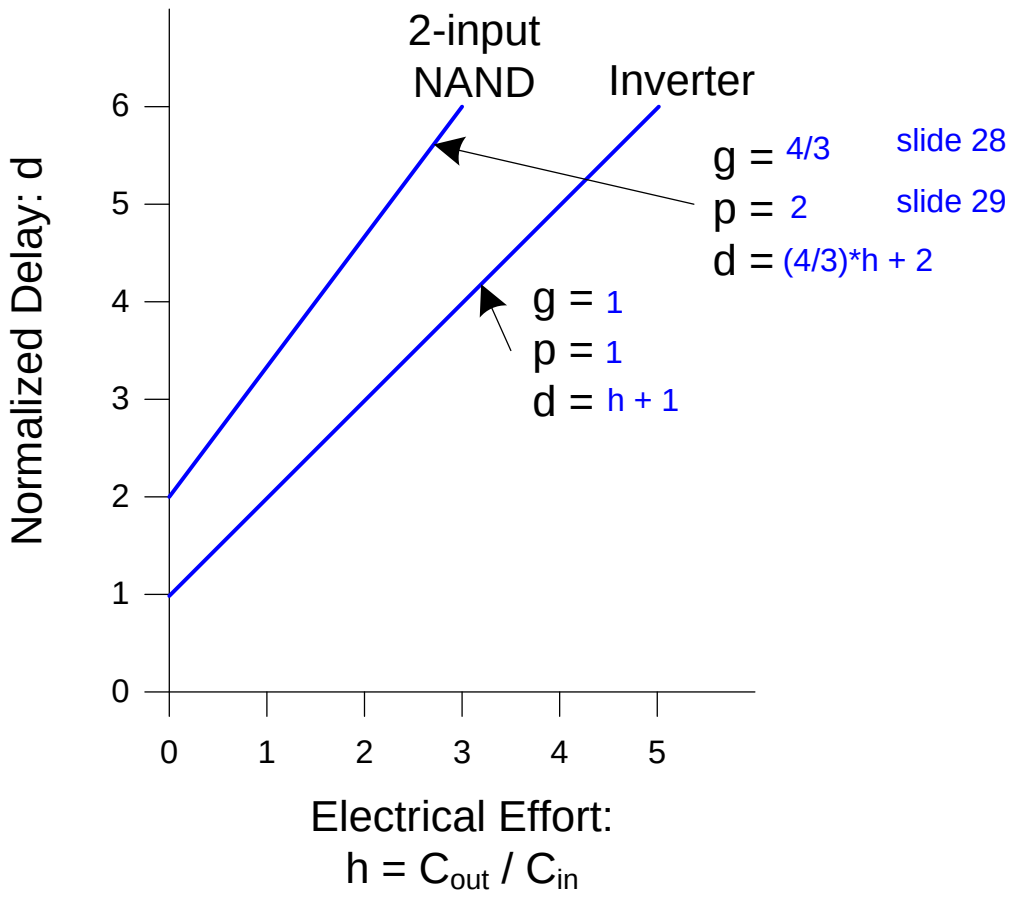
truyền động

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

Delay Plots

$d = f + p$

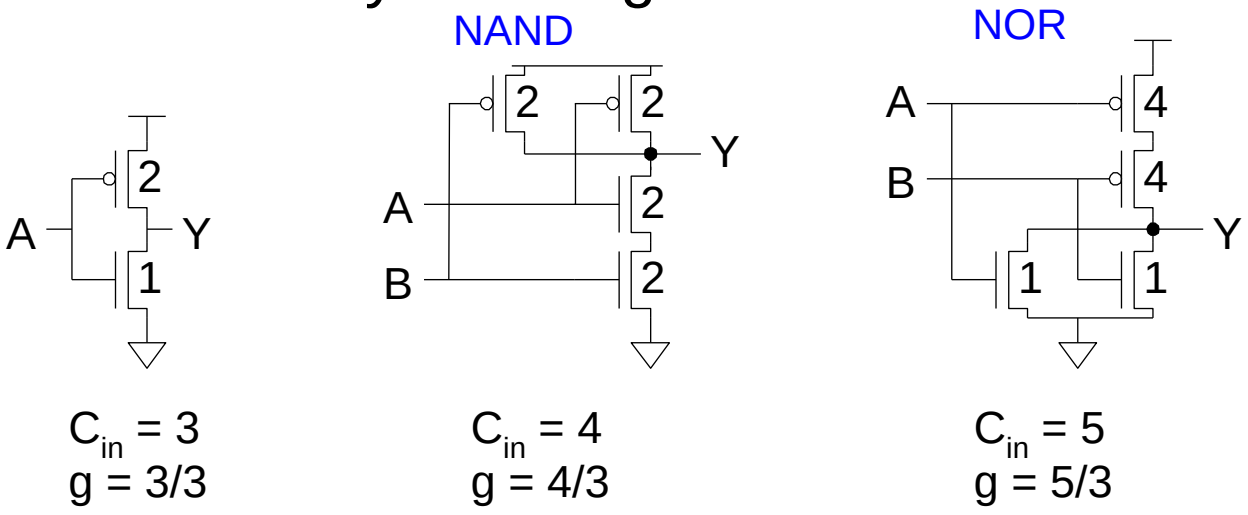
$= gh + p$



Computing Logical Effort

- DEF: *Logical effort (g) 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

$$g = \frac{C_{in}}{C_{in\_inv}}$$



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

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

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

Catalog of Gates

- Logical effort of common gates

g

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

p

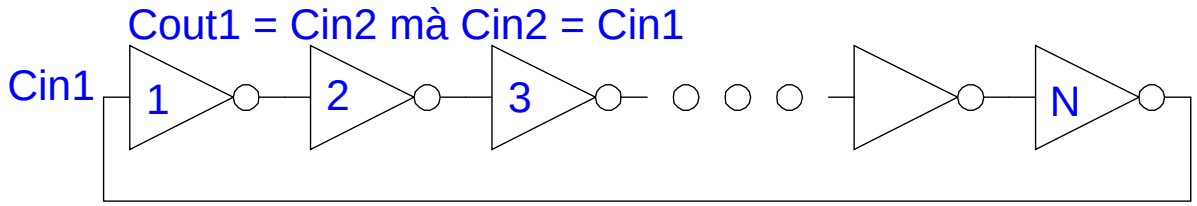
- Parasitic delay of common gates
  - In multiples of p_{inv} (≈1)

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	

vòng

Example: Ring Oscillator

- Estimate the frequency of an N-stage ring oscillator



Logical Effort: $g = 1$

Electrical Effort: $h =$ Cout / Cin = 1 vì nó truyền động một tải giống hệt nhau

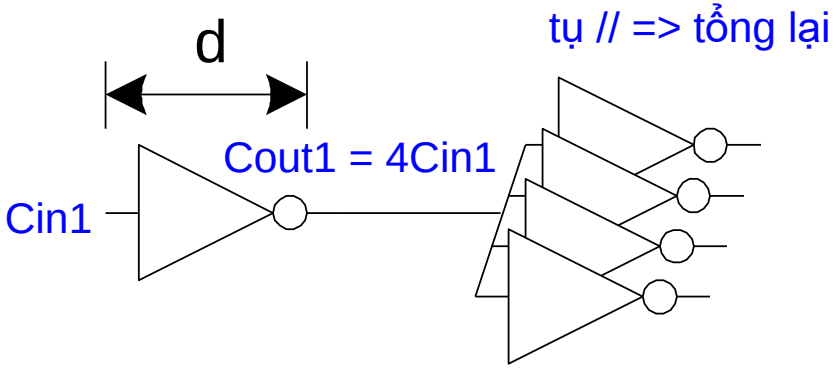
Parasitic Delay: $p = 1$

Stage Delay: $d =$ gh + p = 1*1+1 = 2

Frequency: $f_{osc} =$ 1/(2*N*d) = 1/(2*2*N) = 1/(4N)

Example: FO4 Inverter

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



Logical Effort: g = 1

Electrical Effort: h = Cout1 / Cin1 = 4

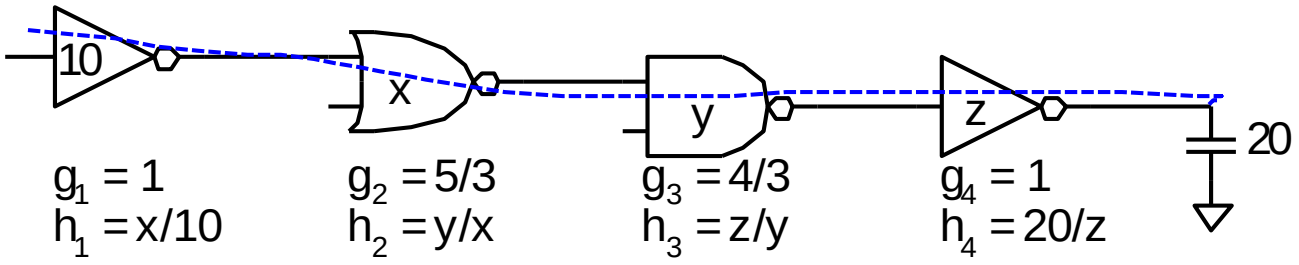
Parasitic Delay: p = 1

Stage Delay: d = gh + p = 1*4 + 1 = 5

Multistage Logic Networks

khái quát thành nhiều tầng

- Logical effort generalizes to multistage networks
- *Path Logical Effort* $G = \prod g_i$
- *Path Electrical Effort* $H = \frac{C_{\text{out-path}}}{C_{\text{in-path}}}$
- *Path Effort* $F = \prod f_i = \prod g_i h_i$



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?

vấn đáp

Paths that Branch

- No! Consider paths that branch:

$G = 1*1=1$

$H = 90/5 = 18$

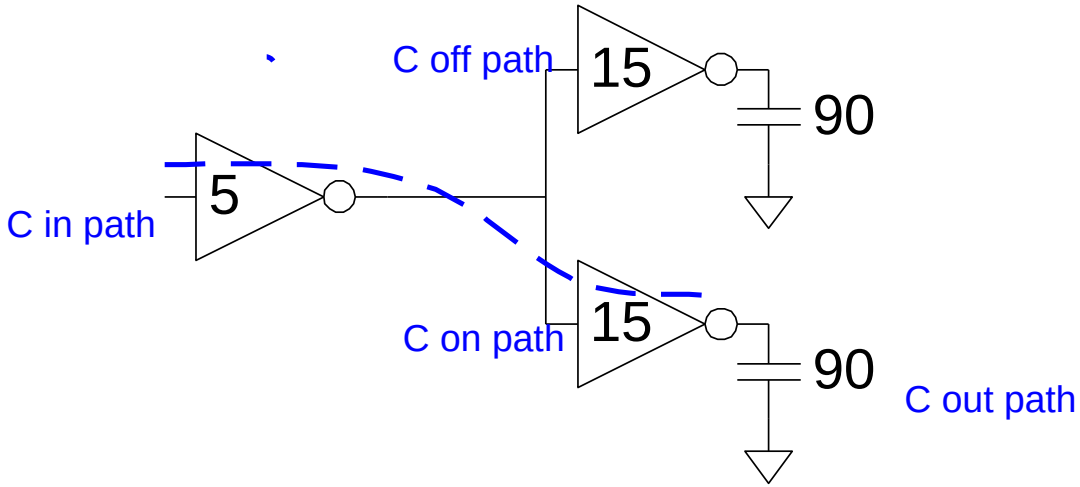
$GH = 18$

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

$h_2 = 90/15 = 6$

$F = f1*f2 = g1*h1*g2*h2 = 1*6*1*6 = 36$

$= 2GH = 2*18 = 36$



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

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

$\sqrt[N]{F}$

- Thus minimum delay of N stage path is

$D = N \cdot F^{1/N} + P$

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

Gate Sizes (~ width k)

- How wide should the gates be for least delay?

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

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

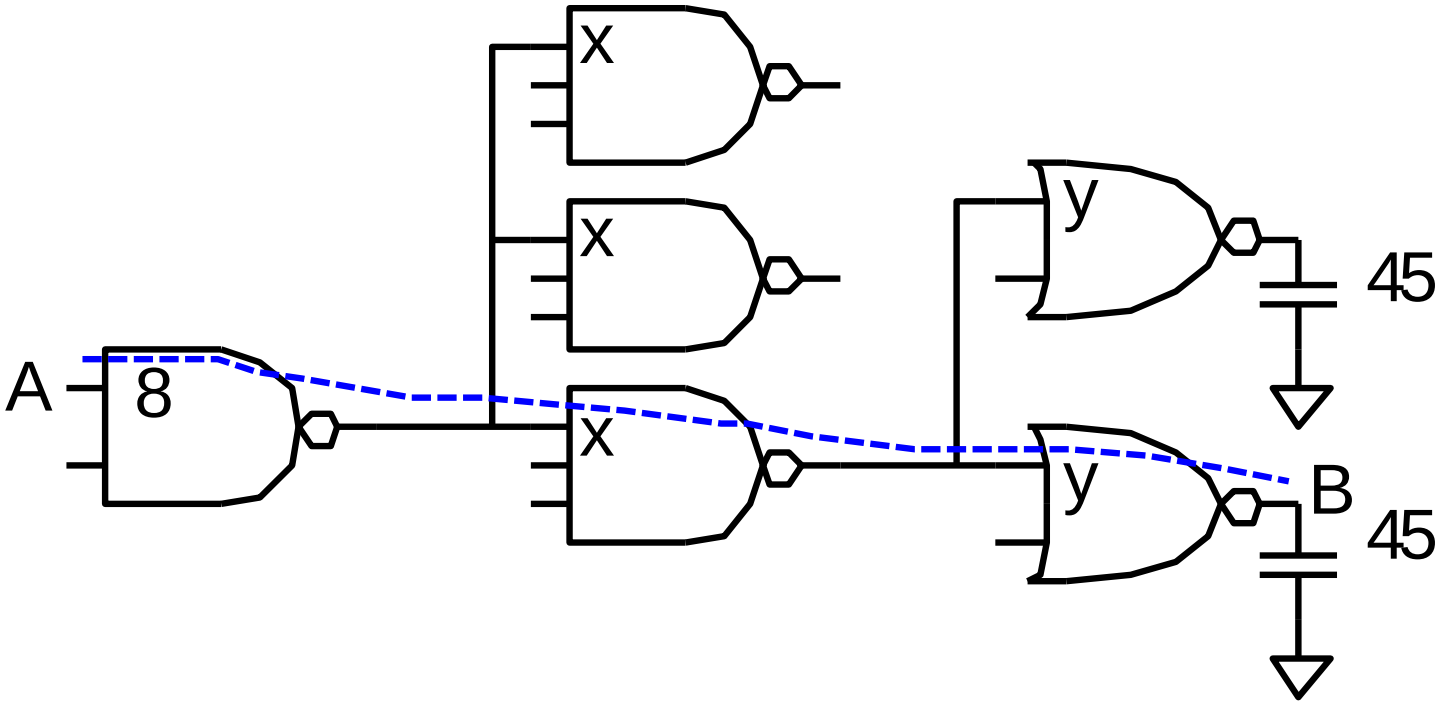
kích thước cổng

x/y/z

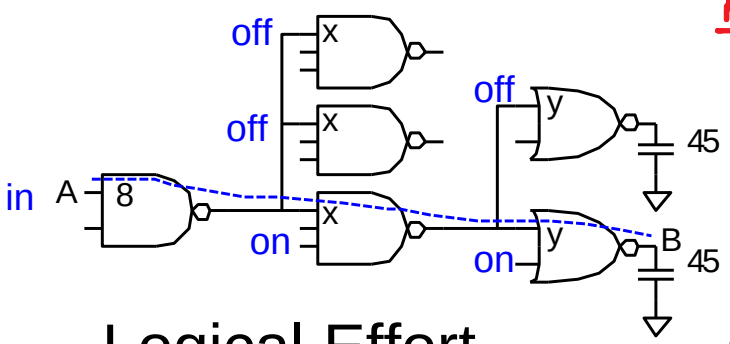
- Working backward, apply capacitance transformation to find input capacitance of each gate given load it drives.
- Check work by verifying input cap spec is met.

Example: 3-stage path

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



Example: 3-stage path



$b_1 = \frac{8}{8}$ $b_2 = \frac{3x}{x}$ $b_3 = \frac{2y}{y}$

tổng các C lại chia cho chính nó

Logical Effort	G =	tích g i = (4/3) × (5/3) × (5/3) = 100/27 dò catalog
Electrical Effort	H =	C out-path / C in-path = 45/8
Branching Effort	B =	tích b i = 1*3*2 = 6
Path Effort	F =	GBH = 100/27 * 45/8 * 6 = 125
Best Stage Effort	$\hat{f}$ =	F^(1/N) = 125^(1/3) = 5
Parasitic Delay	P =	sigma p i = 2 + 3 + 2 = 7 dò catalog
Delay	D =	Df + P = N * F^(1/N) + P = 3*5 + 7 = 22 = 4.4 FO4

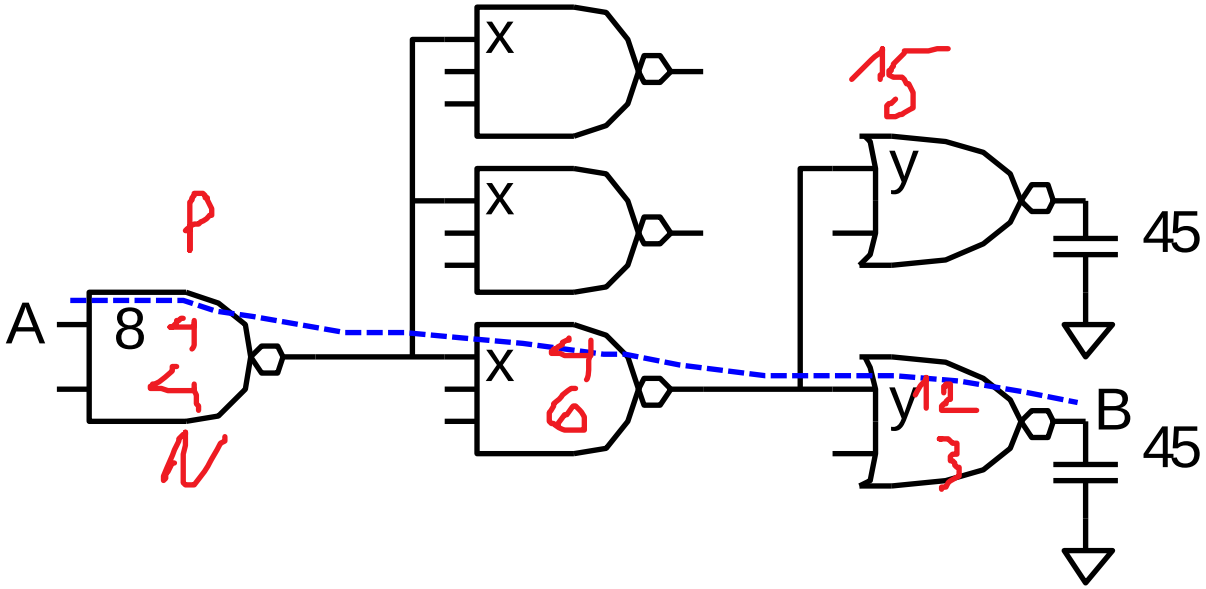
3 tầng

Example: 3-stage path

- Work backward for sizes

  y = g i * Cout i / f^ = (5/3) * 45 / 5 = 15

  x = g i * Cout i / f^ = (5/3)*(15 + 15) / 5 = 10



Best Number of Stages

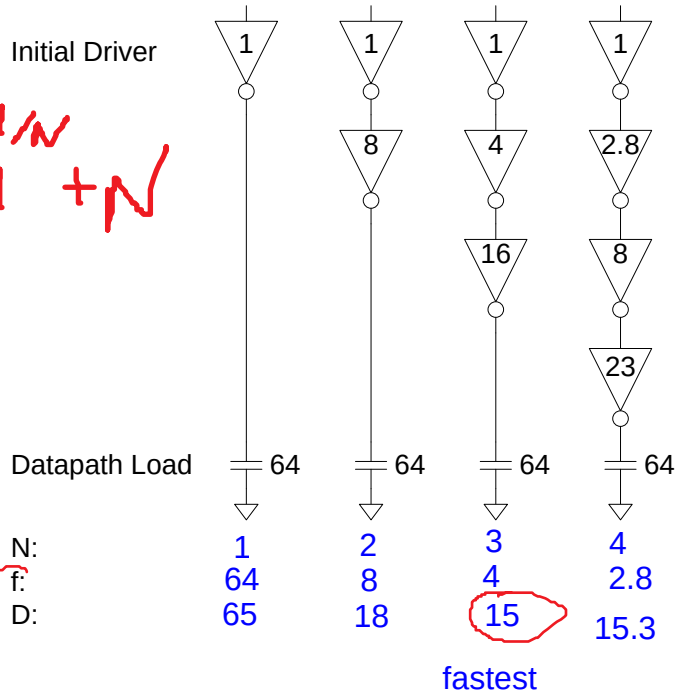
- How many stages should a path use?
  - Minimizing number of stages is not always fastest
- Example: drive 64-bit datapath with unit inverter

do cùng 1 loại cổng inverter nên P = N

$$D = NF^{1/N} + P = N \cdot 64^{1/N} + N$$

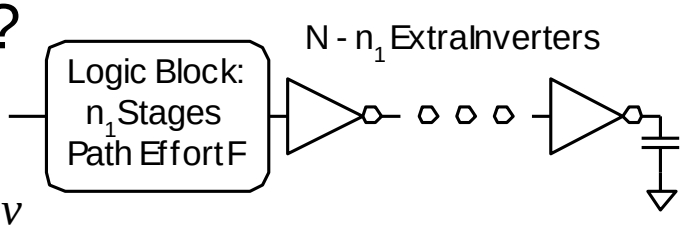
1*64 + 1 = 65
1*64 + 1*(64/8) + 2 = 18
1*4 + 1*(16/4) + 1*(64/16) + 3 = 15
1*2.8 + 1*(8/2.8) + 1*(23/8) + 1*(64/23) + 4 = 15.3

F^(1/N) = (G*H)^(1/N) = (1*64)^(1/N)
Df + P = sigma f i + sigma p i



Derivation

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



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

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

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

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

Best Stage Effort

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

bỏ qua

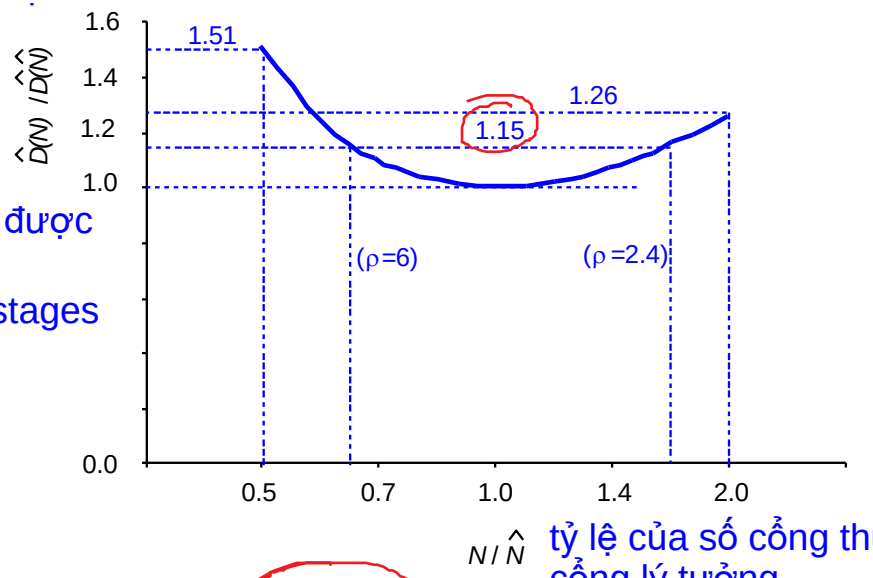
- Neglecting parasitics ($p_{inv} = 0$), we find $\rho = 2.718$ (e)
- For $p_{inv} = 1$, solve numerically for $\rho = 3.59$

Sensitivity Analysis

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

tỷ lệ của delay thực tế
với delay tốt nhất có thể đạt được

A path achieves least delay by using N^ = log p F stages



tỷ lệ của số cổng thực tế với số cổng lý tưởng tối ưu

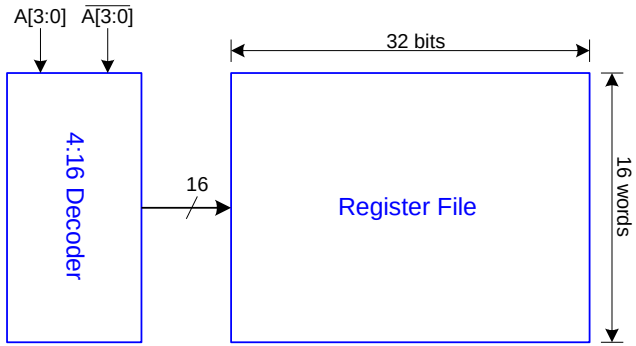
- 2.4 < ρ < 6 gives delay within 15% of optimal
  - We can be sloppy!
  - Harris uses ρ = 4, exact ρ is process dependent

Optimal Power?

- Switching vs. Crowbar

Example, Revisited

- A memory designer for an embedded automotive processor. Help 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
- 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: H = 96/10 = 9.6

  Branching Effort: B = (10 + 10*7)/10 = 8

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

  Path Effort: F = GBH = 1*8*9.6 = 76.8

  Number of Stages: N = log p F = log 4 76.8 = 3

- Try a -stage design *(But G≠1 and ρ≠4)*

Gate Sizes & Delay

H = 96/10 = 9.6
B = 1*8*1 = 8

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

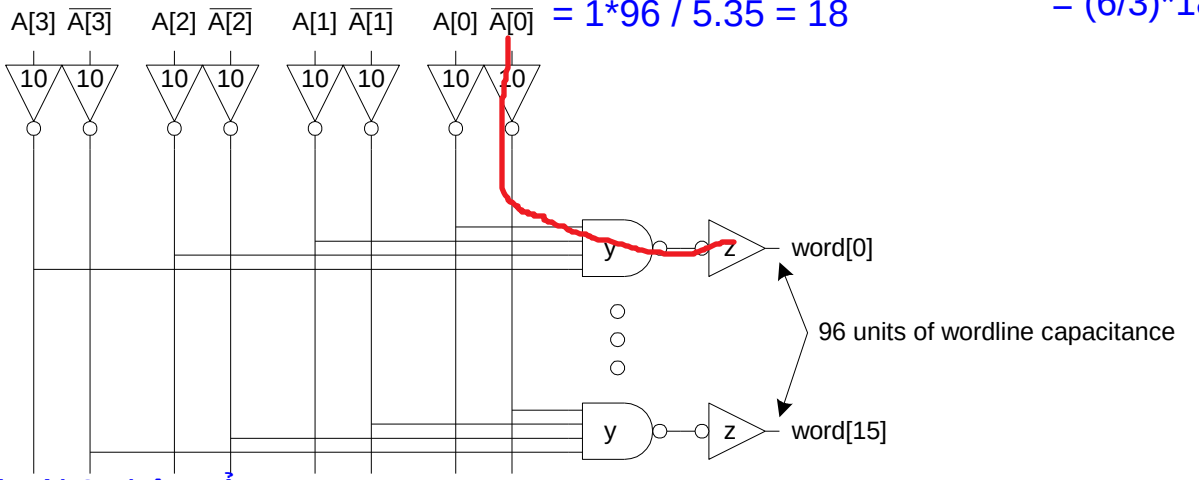
Path Effort: F = GBH = 2*8*9.6 = 153.6

Stage Effort: $\hat{f}$ = F^(1/N) = 153.6^(1/3) = 5.35

Path Delay: D = N * F^(1/N) + P = 3*5.35 + (1+4+1) = 22.06

Gate sizes: z = g i * Cout i / f^ = 1*96 / 5.35 = 18

y = g i * Cout i / f^ = (6/3)*18 / 5.35 = 6.7



1 cổng A nối với 8 chân cổng NAND

0000 -> 1111

Comparison

bảng tính

- Compare many alternatives with a spreadsheet
- $D = N(76.8\ G)^{1/N} + P$

Design	N	G	P	D
NOR4	1	3	4	234
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

vấn đáp

Review of Definitions

Term	Stage	Path
number of stages	1	N
logical effort	g	$G = \prod g_i$
electrical effort	$h = \frac{C_{out}}{C_{in}}$	$H = \frac{C_{out\text{-}path}}{C_{in\text{-}path}}$
branching effort	$b = \frac{C_{on\text{-}path} + C_{off\text{-}path}}{C_{on\text{-}path}}$	$B = \prod b_i$
effort	$f = gh$	$F = GBH$
effort delay	f	$D_F = \sum f_i$
parasitic delay	p	$P = \sum p_i$
delay	$d = f + p$	$D = \sum d_i = D_F + P$

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

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

sự lặp lại

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
  - But requires practice to master