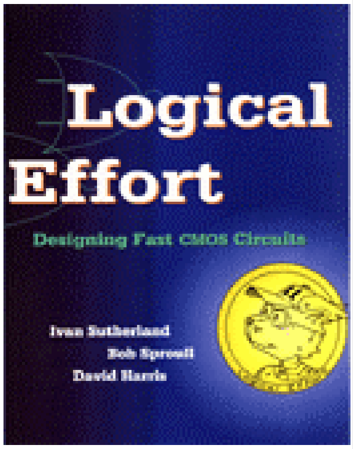
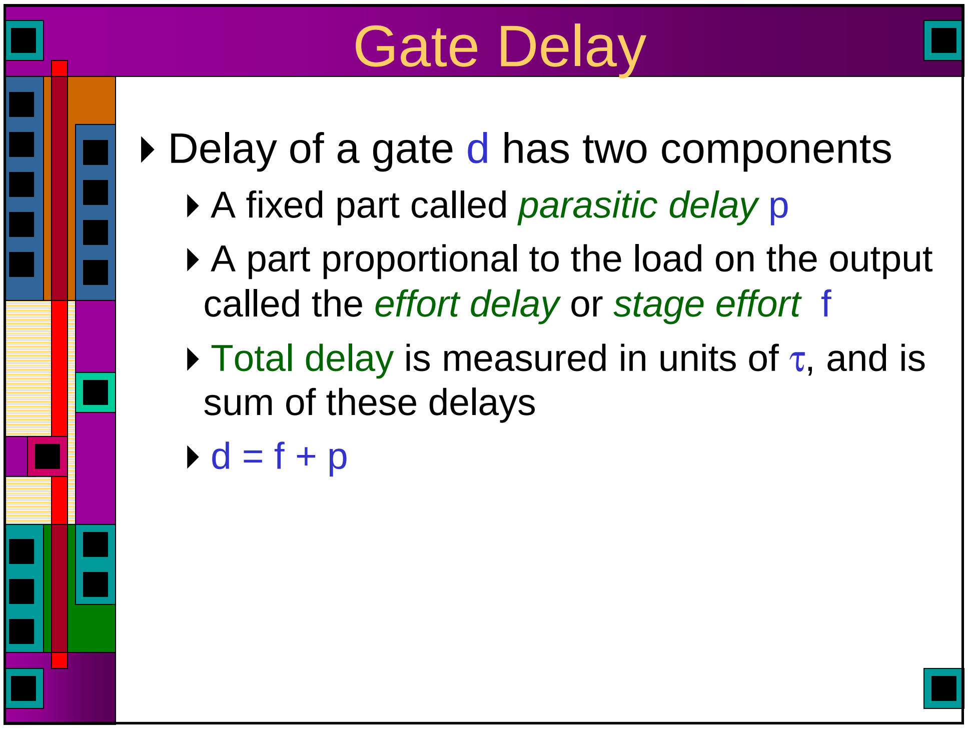
Estimating Delays

- Would be nice to have a “back of the envelope” method for sizing gates for speed
- Logical Effort
  - Book by Sutherland, Sproull, Harris
  - Chapter 1 is on our web page
  - See also section 4.4.3 in your text



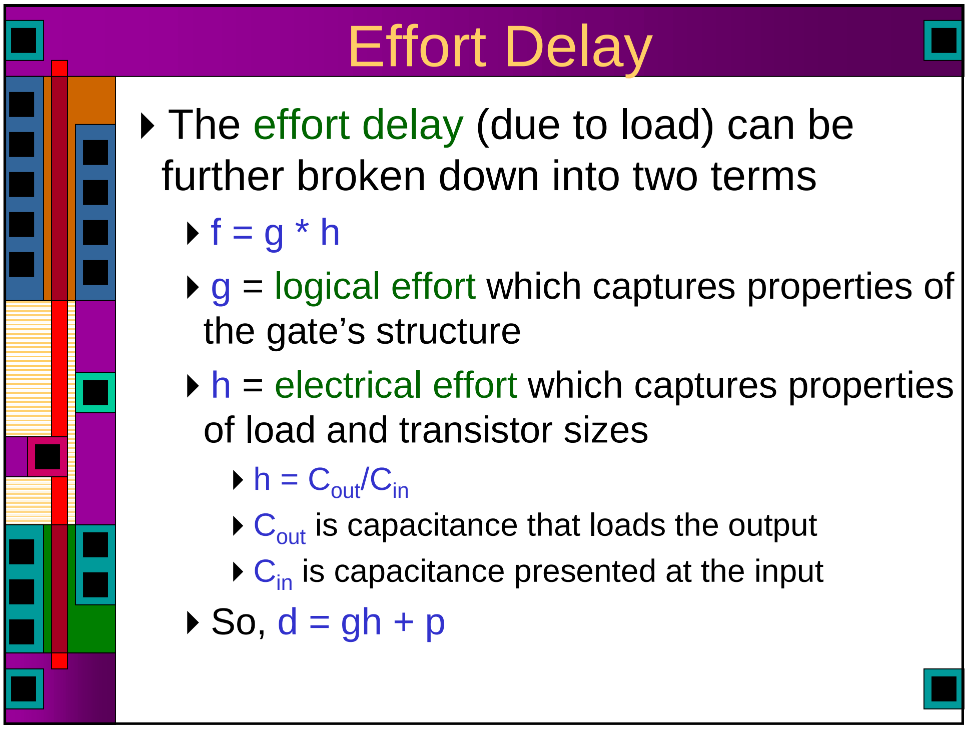
Gate Delay Model

- First, normalize a model of delay to dimensionless units to isolate fabrication effects
  - $d_{abs} = d\ \tau$
  - τ is the delay of a minimum inverter driving another minimum inverter in some process
  - In a 0.6u process, this is approx 50ps
  - Now we can think about delay in terms of d and scale it to whatever process we’re building the circuit in



Gate Delay

- Delay of a gate d has two components
  - A fixed part called *parasitic delay* p
  - A part proportional to the load on the output called the *effort delay* or *stage effort* f
  - Total delay is measured in units of τ, and is sum of these delays
  - $d = f + p$

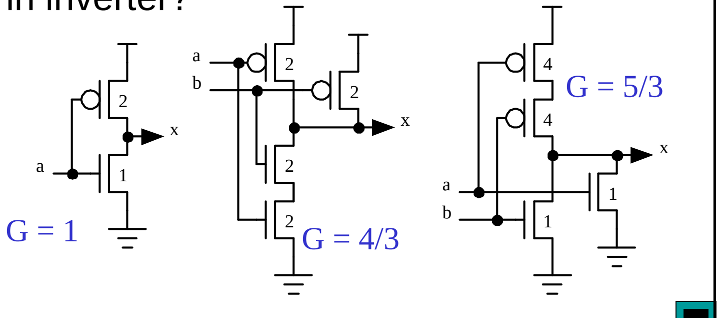


Effort Delay

- The effort delay (due to load) can be further broken down into two terms
  - $f = g * h$
  - g = logical effort which captures properties of the gate's structure
  - h = electrical effort which captures properties of load and transistor sizes
    - $h = C_{out}/C_{in}$
    - C_{out} is capacitance that loads the output
    - C_{in} is capacitance presented at the input
  - So, $d = gh + p$

Logical Effort

- Logical effort normalizes the output drive capability of a gate to match a unit inverter
  - How much more input capacitance does a gate need to present to offer the same drive as in inverter?



(a) (b) (c)

Logical Effort of Other Gates

- Logical effort of common gates assuming that P/N size ratio is 2

Number of inputs

Gate Type	*1*	*2*	*3*	*4*	*5*	*n*
Inverter	1					
NAND		4/3	5/3	6/3	7/3	(n+2)/3
NOR		5/3	7/3	9/3	11/3	(2n+1)/3
MUX		2	2	2	2	2
XOR		4	12	32		

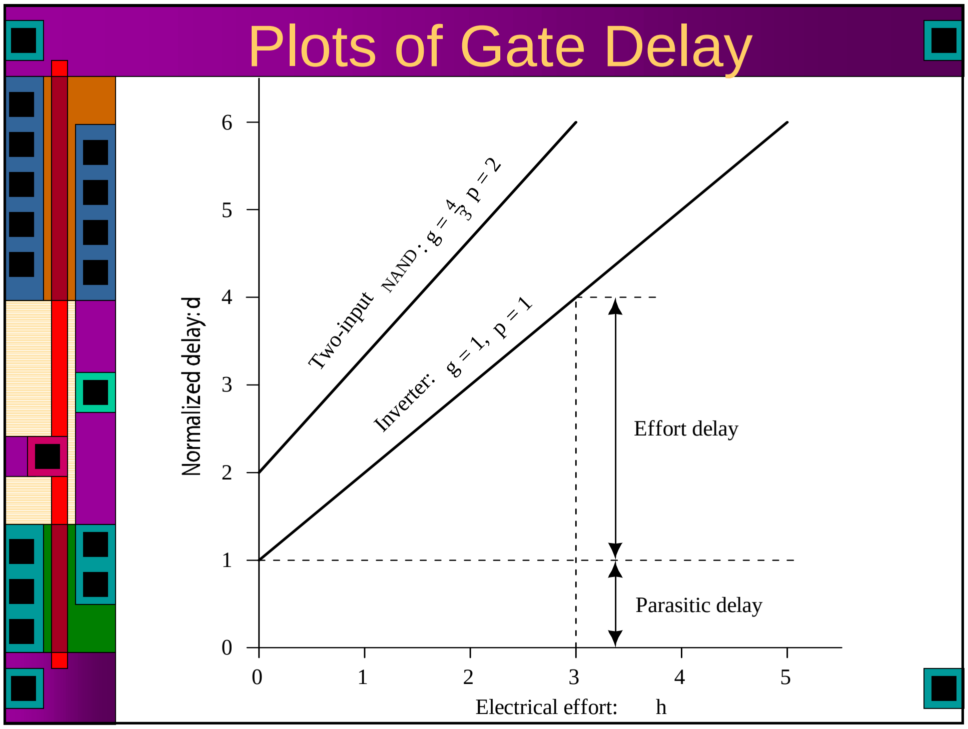
Electrical Effort

- Value of logical effort g is independent of transistor size
  - It’s related to the ratios and the topology
- Electrical effort h captures the drive capability of the transistors via sizing
  - Electrical effort $h = C_{out}/C_{in}$
  - Note that as transistor sizes for a gate increase, h decreases because C_{in} goes up

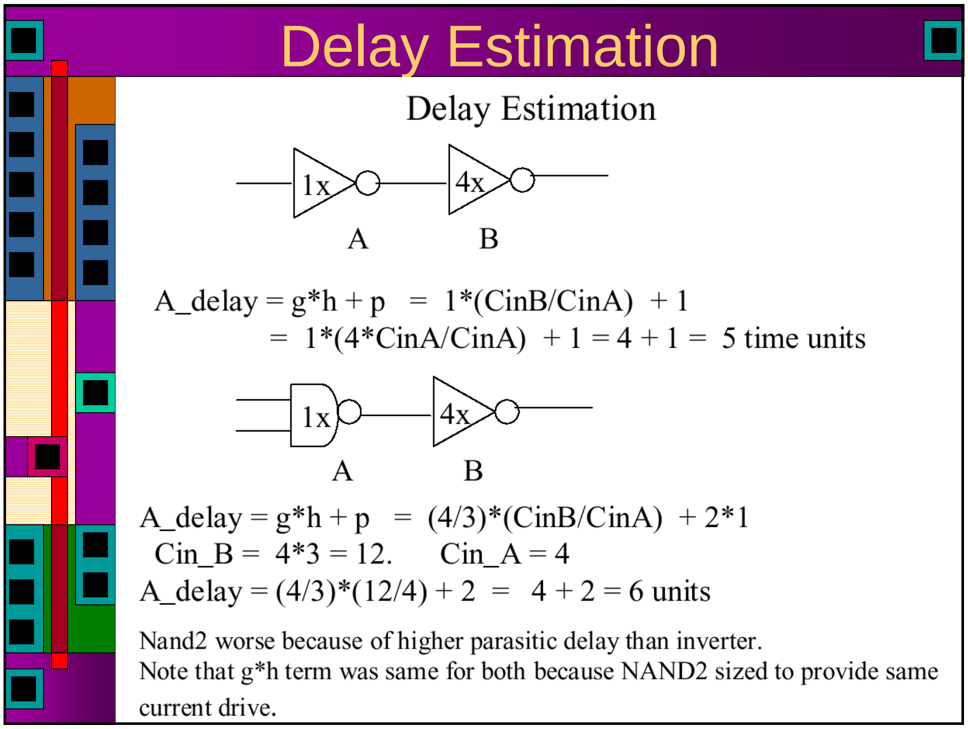
Parasitic Delay

- Parasitic delay p is caused by the internal capacitance of the gate
  - It’s constant and independent of transistor size
  - As you increase the transistor size, you also increase the cap of the gate/source/drain areas which keeps it constant
  - For our purposes, normalize p_{inv} to 1
    - N-input NAND = $n*p_{inv}$
    - N-input NOR = $n*p_{inv}$
    - N-way mux = $2n*p_{inv}$
    - XOR = $4*\ p_{inv}$

Plots of Gate Delay



Delay Estimation

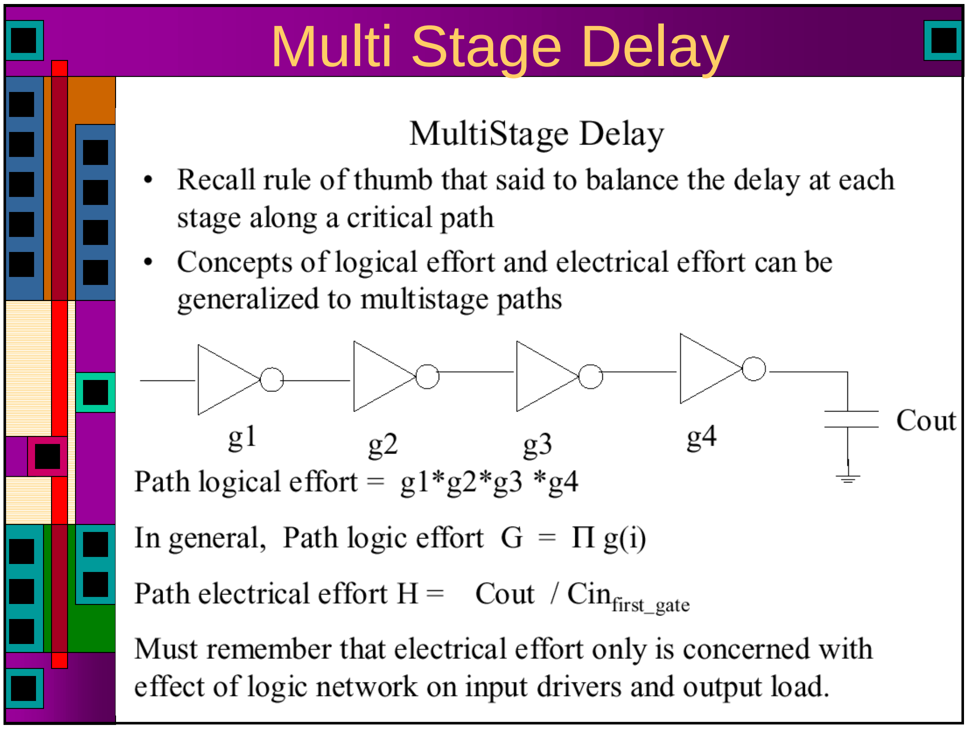


Delay Estimation

A_delay = g*h + p = 1*(CinB/CinA) + 1
= 1*(4*CinA/CinA) + 1 = 4 + 1 = 5 time units

A_delay = g*h + p = (4/3)*(CinB/CinA) + 2*1
Cin_B = 4*3 = 12. Cin_A = 4
A_delay = (4/3)*(12/4) + 2 = 4 + 2 = 6 units

Nand2 worse because of higher parasitic delay than inverter.
Note that g*h term was same for both because NAND2 sized to provide same current drive.



Multi Stage Delay

MultiStage Delay

- Recall rule of thumb that said to balance the delay at each stage along a critical path
- Concepts of logical effort and electrical effort can be generalized to multistage paths

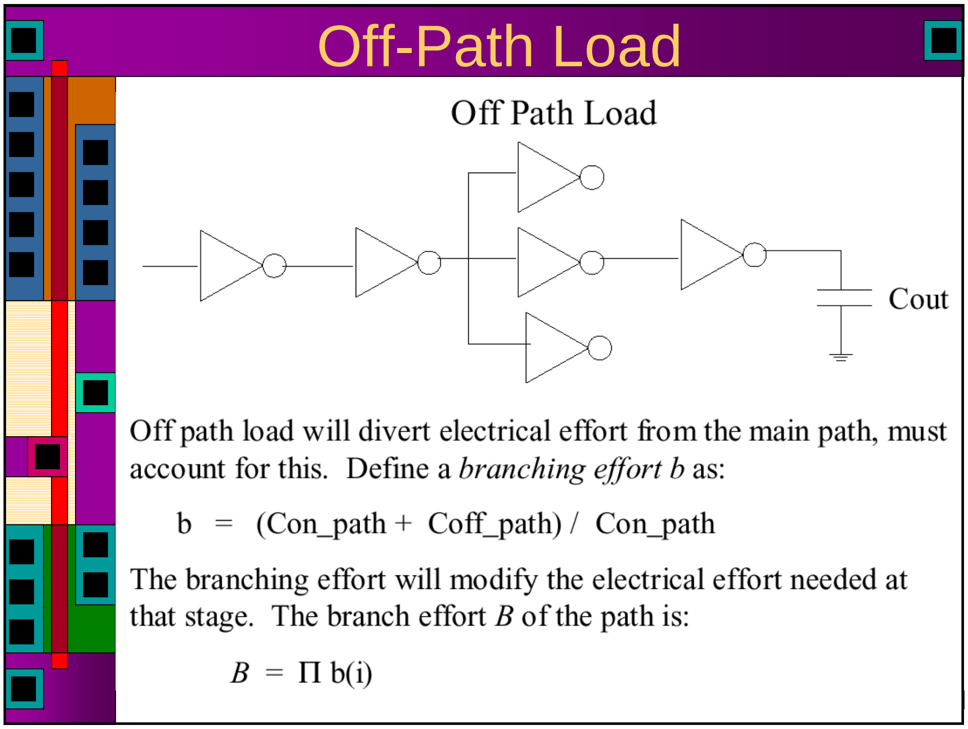
g1
g2
g3
g4
Cout

Path logical effort = g1*g2*g3 *g4

In general, Path logic effort $G = \Pi\, g(i)$

Path electrical effort H = Cout / $\text{Cin}_{\text{first\_gate}}$

Must remember that electrical effort only is concerned with effect of logic network on input drivers and output load.



Off-Path Load

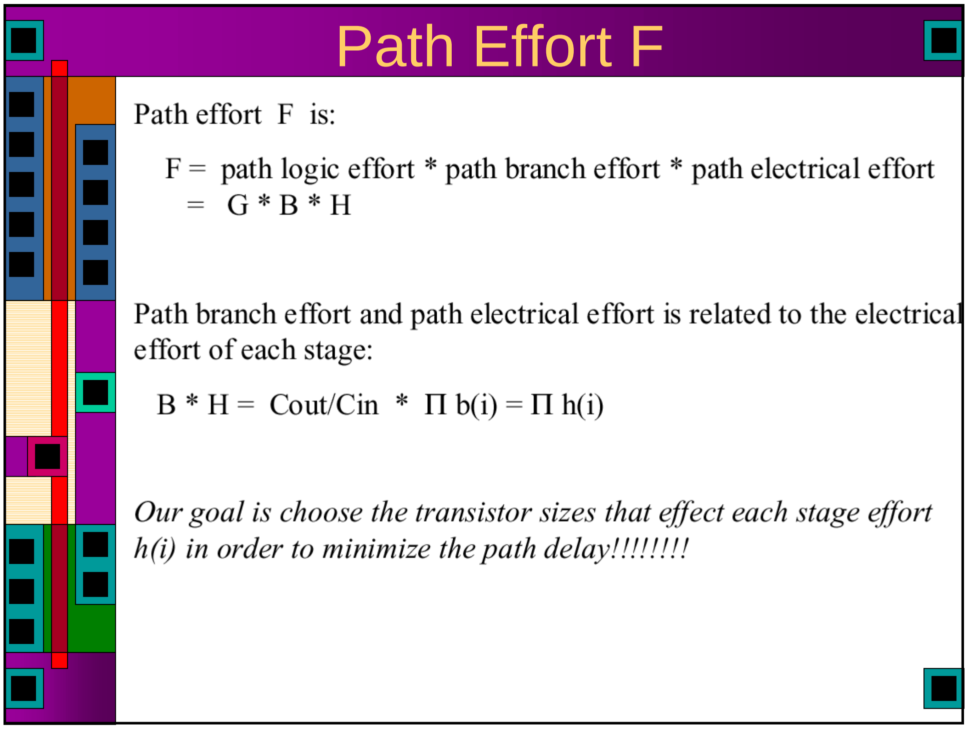
Off Path Load

Off path load will divert electrical effort from the main path, must account for this. Define a *branching effort b* as:

$$b = (\text{Con\_path} + \text{Coff\_path}) / \text{Con\_path}$$

The branching effort will modify the electrical effort needed at that stage. The branch effort *B* of the path is:

$$B = \Pi\, b(i)$$



Path Effort F

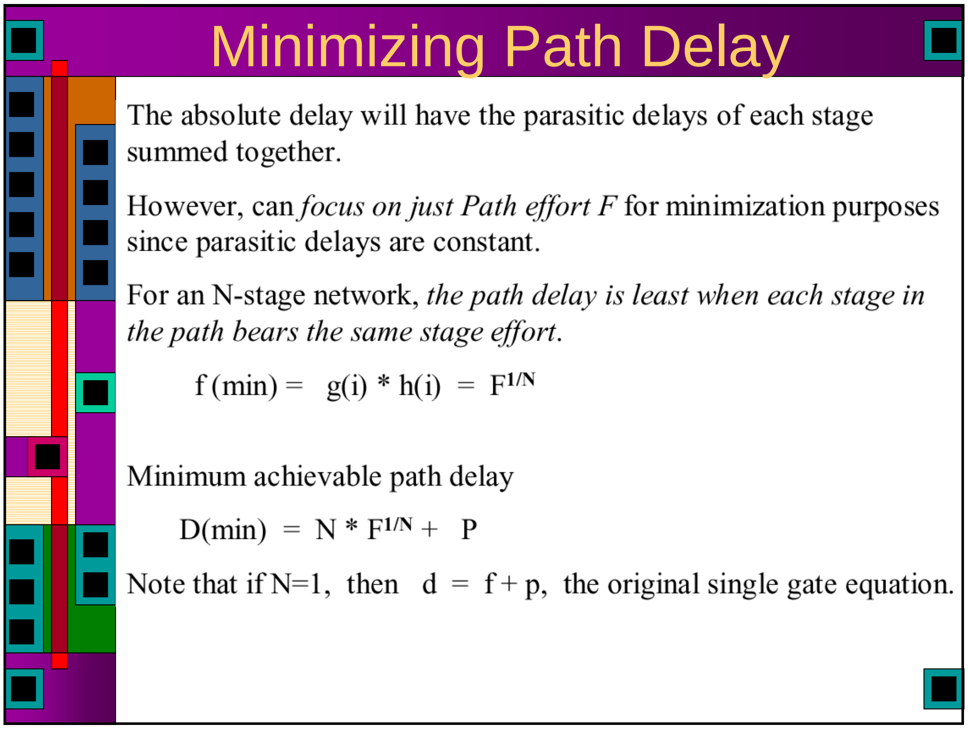
Path effort F is:

F = path logic effort * path branch effort * path electrical effort
= G * B * H

Path branch effort and path electrical effort is related to the electrical effort of each stage:

$B * H = Cout/Cin * \Pi\, b(i) = \Pi\, h(i)$

Our goal is choose the transistor sizes that effect each stage effort h(i) in order to minimize the path delay!!!!!!!!



Minimizing Path Delay

The absolute delay will have the parasitic delays of each stage summed together.

However, can *focus on just Path effort F* for minimization purposes since parasitic delays are constant.

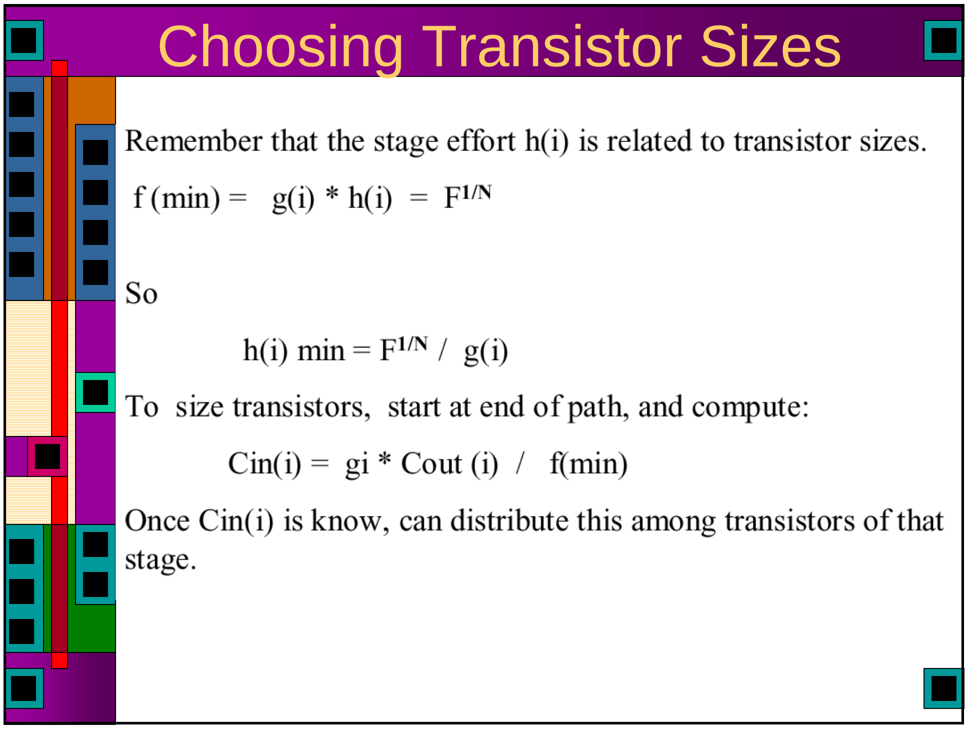
For an N-stage network, *the path delay is least when each stage in the path bears the same stage effort.*

$f\,(min) = g(i) * h(i) = F^{1/N}$

Minimum achievable path delay

$D(min) = N * F^{1/N} + P$

Note that if N=1, then d = f + p, the original single gate equation.



Choosing Transistor Sizes

Remember that the stage effort h(i) is related to transistor sizes.

$f(\min) = g(i) * h(i) = F^{1/N}$

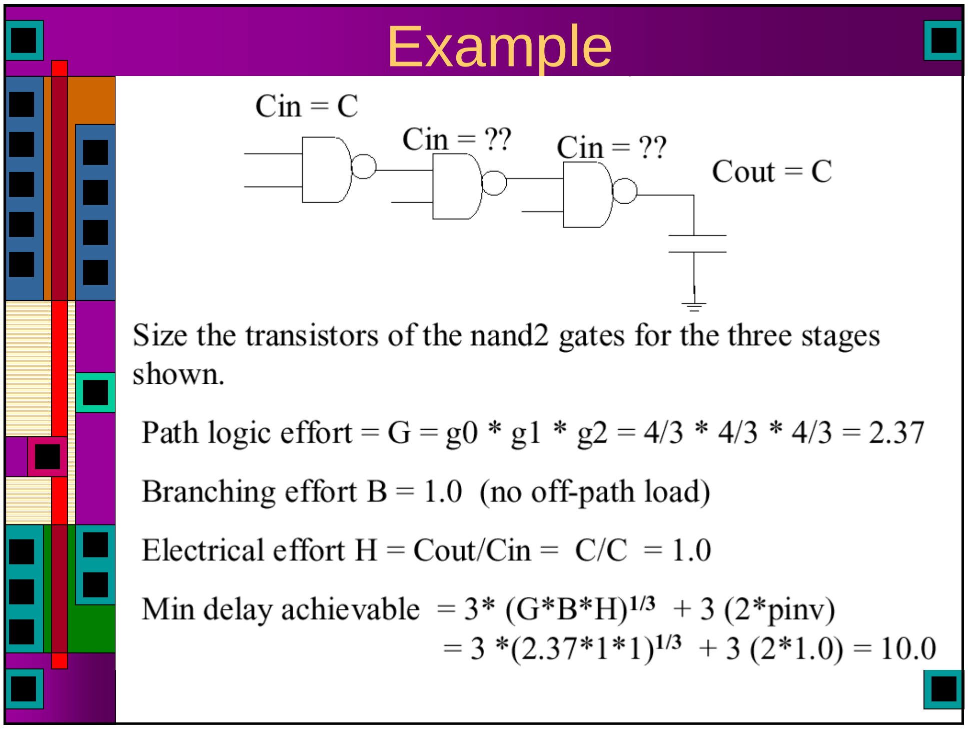
So

$h(i) \min = F^{1/N} / g(i)$

To size transistors, start at end of path, and compute:

$Cin(i) = gi * Cout(i) / f(\min)$

Once Cin(i) is know, can distribute this among transistors of that stage.



Example

Size the transistors of the nand2 gates for the three stages shown.

Path logic effort = G = g0 * g1 * g2 = 4/3 * 4/3 * 4/3 = 2.37

Branching effort B = 1.0 (no off-path load)

Electrical effort H = Cout/Cin = C/C = 1.0

Min delay achievable $= 3* (G*B*H)^{1/3} + 3\,(2*pinv)$
$= 3 *(2.37*1*1)^{1/3} + 3\,(2*1.0) = 10.0$

Example, continued

The effort of each stage will be:

$$f\ min = (G*B*H)^{1/3} = (2.37*1.0*1.0)^{1/3} = 1.33 = 4/3$$

Cin of last gate should equal:

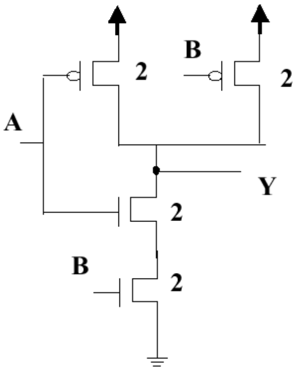
$$\text{Cin last gate (min)} = gi * \text{Cout (i)} \ / \ f(min)$$
$$= 4/3 * C / (4/3) = C$$

Cin of middle gate should equal:

$$\text{Cin middle gate} = gi * \text{Cin last gate} / f(min)$$
$$= 4/3 * C/ (4/3) = C$$

All gates have same input capacitance, distribute it among transistors.

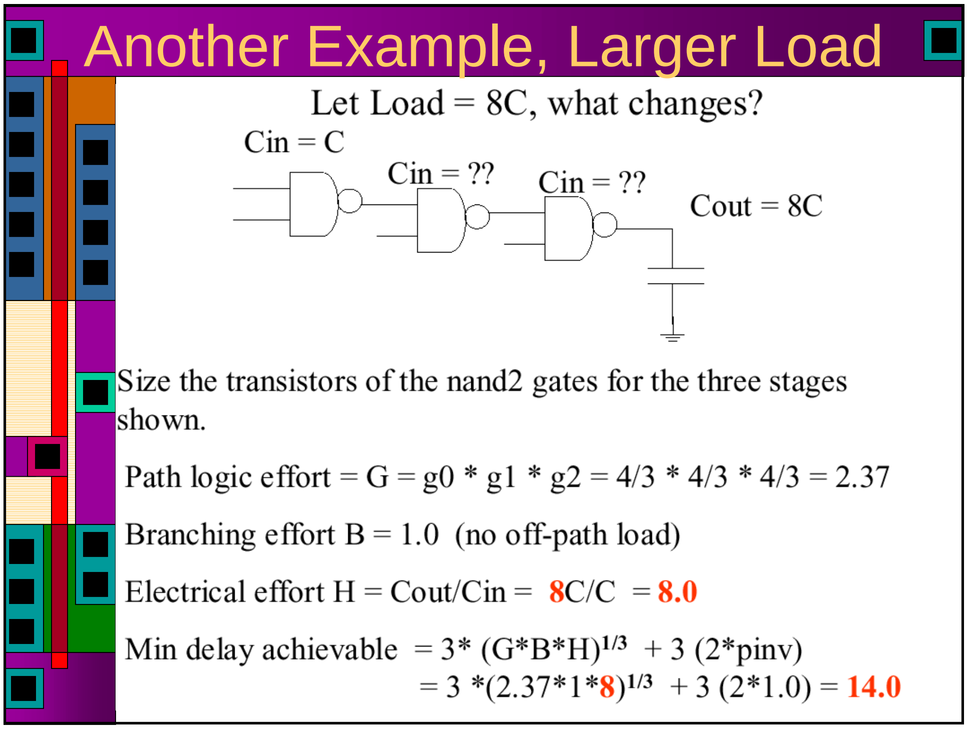
Transistor Sizes for Example



Where gate capacitance of

2 *W *L Mosfet = C/2

Choose W accordingly.



Another Example, Larger Load

Let Load = 8C, what changes?

Cin = C

Cin = ??

Cin = ??

Cout = 8C

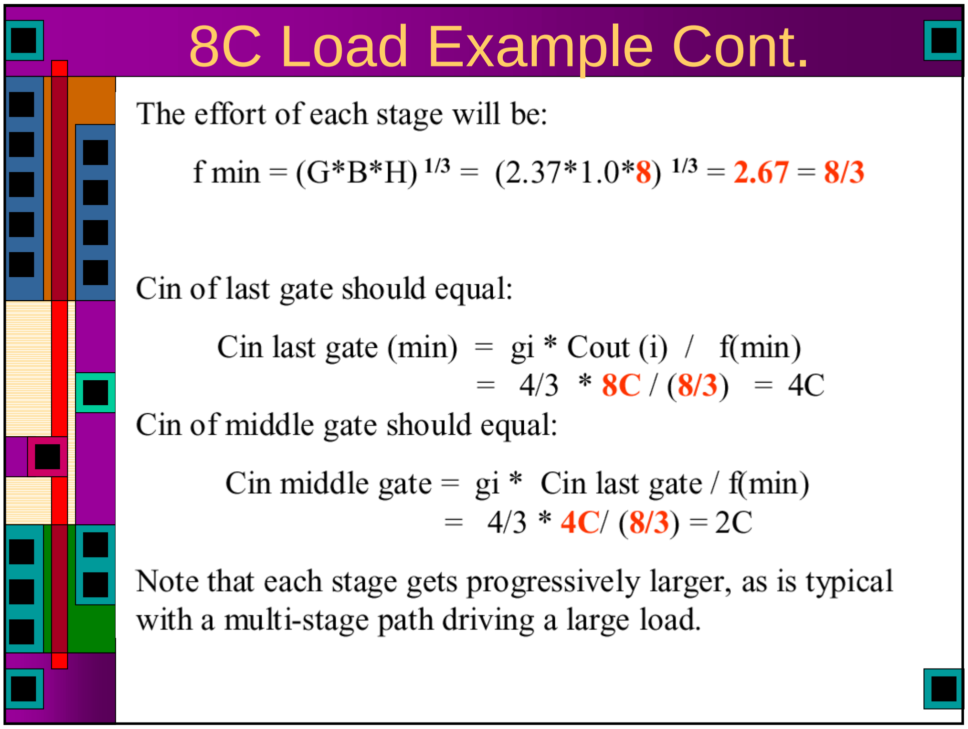
Size the transistors of the nand2 gates for the three stages shown.

Path logic effort = G = g0 * g1 * g2 = 4/3 * 4/3 * 4/3 = 2.37

Branching effort B = 1.0 (no off-path load)

Electrical effort H = Cout/Cin = **8**C/C = **8.0**

Min delay achievable = 3* $(G*B*H)^{1/3}$ + 3 (2*pinv)
= 3 *$(2.37*1*\mathbf{8})^{1/3}$ + 3 (2*1.0) = **14.0**



8C Load Example Cont.

The effort of each stage will be:

f min = $(G*B*H)^{1/3}$ = $(2.37*1.0*\mathbf{8})^{1/3}$ = **2.67** = **8/3**

Cin of last gate should equal:

Cin last gate (min) = gi * Cout (i) / f(min)
= 4/3 * **8C** / (**8/3**) = 4C

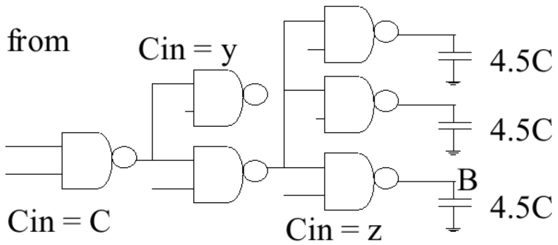
Cin of middle gate should equal:

Cin middle gate = gi * Cin last gate / f(min)
= 4/3 * **4C**/ (**8/3**) = 2C

Note that each stage gets progressively larger, as is typical with a multi-stage path driving a large load.

Example 1.6 from Chap 1

Size path from A to B



Path logic effort G = g0 * g1 * g2 = 4/3 * 4/3 * 4/3 = 2.37

Branch effort, 1st stage = (y+y)/y = 2.

Branch effort, 2nd stage = (z+z+z)/z = 3

Path Branch effort B = 2 * 3 = 6.

Path electrical effort H = Cout/Cin = 4.5C/C = 4.5

Path stage effort = F = G*B*H = 2.37*6*4.5 = 64.

Min delay = $N(F)^{1/N} + P = 3*(64)^{1/3} + 3(2pinv) = 18.0$ units

Example 1.6 Continued

Stage effort of each stage should be:

$f(min) = (F)^{1/N} = (GBH)^{1/N} = (64)^{1/3} = 4$

Determine Cin of last stage:

Cin(z) = g * Cout / f(min) = 4/3 * 4.5C / 4 = 1.5 C

Determine Cin of middle stage:

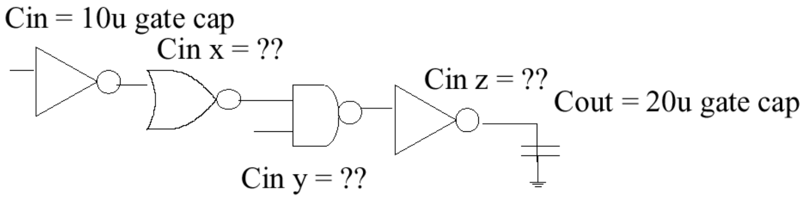
Cin(y) = g * (3*Cin(z))/ f(min) = 4/3 * (3*1.5C) / 4 = 1.5C

Is first stage correct?

Cin(A) = g * (2*Cin(y))/f(min) = 4/3 *(2*1.5C)/4 = C.

Yes, self-consistent.

Example 1.7 from Chap 1



Path logic effort G = g0 * g1 * g2 * g3 = 1*5/3 * 4/3 * 1 = 20/9

Path Branch effort B = 1

Path electrical effort H = Cout/Cin = 20/10 = 2

Path stage effort = F = G*B*H = (20/9)*1*2 = 40/9

For Min delay, each stage has effort $(F)^{1/N} = (40/9)^{1/4} = 1.45$

z = g * Cout/f(min) = 1*20/1.45 = 14

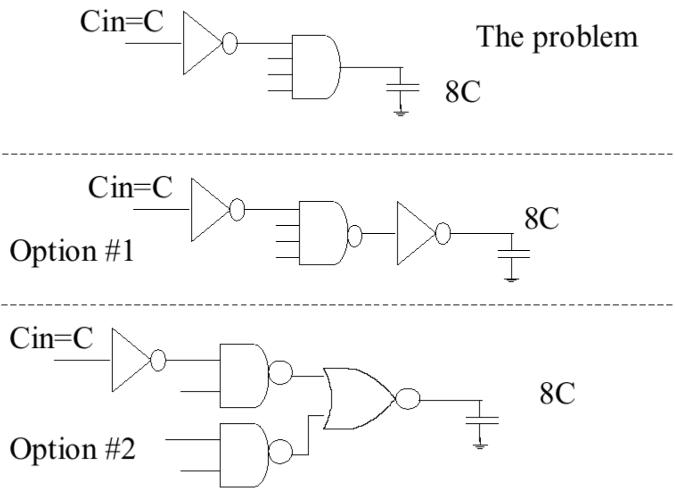
y = g * Cin(z)/f(min) = 4/3 * 14 / 1.45 = 13

x = g * Cin(y)/f(min) = 5/3 * 13 / 1.45 = 15

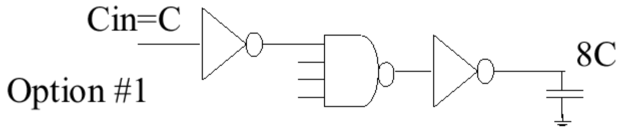
Misc. Comments

- Note that you never size the first gate
  - This gate is assumed to be fixed
  - If you were allowed to size it, the algorithm would try to make it as large as possible
- This is an estimation algorithm
  - Authors claim that sizing a gate by 1.5x too big or small still results in a path delay within 5% of minimum
- In general, the best stage effort is between 3 and 4 (not e as often stated)
  - The e value doesn't use parasitics…

Evaluating Different Options



Option #1



Path logic effort G = g0 * g1 * g2 = 1*6/3 * 1 = 2

Path Branch effort B = 1

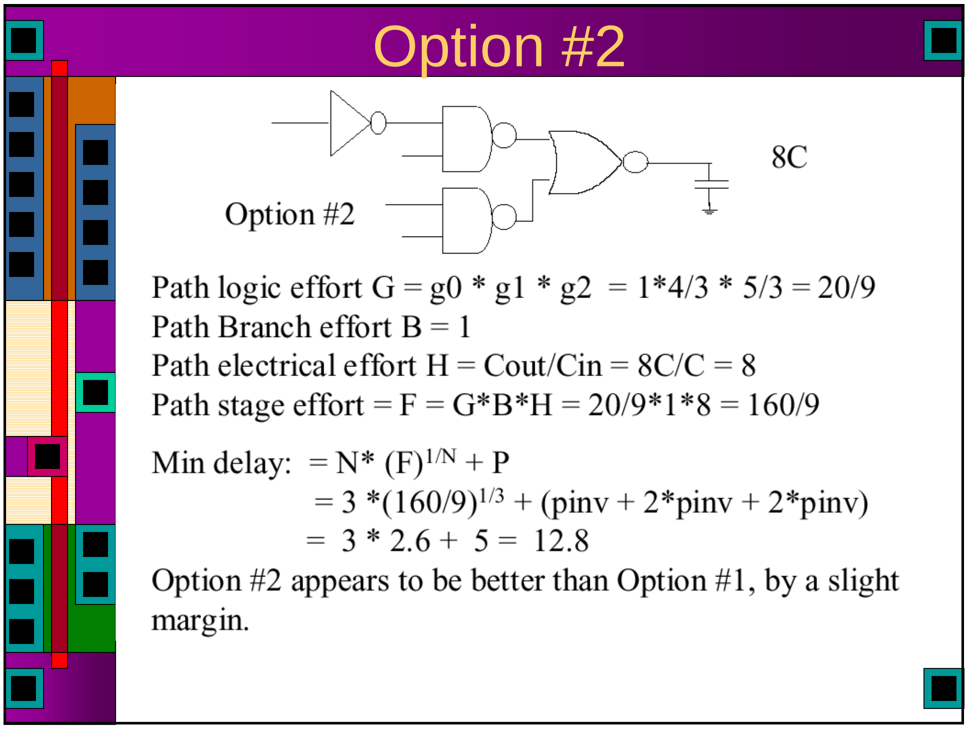
Path electrical effort H = Cout/Cin = 8C/C = 8

Path stage effort = F = G*B*H = 2*1*8 = 16

Min delay: $= N^* (F)^{1/N} + P$

$= 3 * (16)^{1/3} + (\text{pinv} + 4^*\text{pinv} + \text{pinv})$

$= 3 * (2.5) + 6 = 13.5$

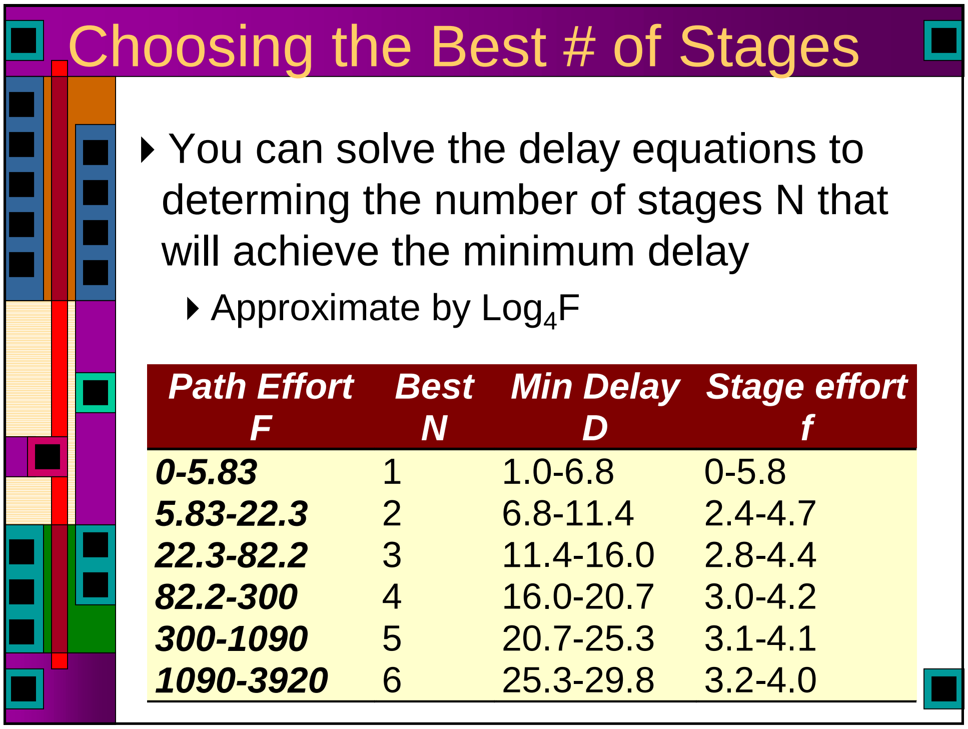


Option #2

Path logic effort G = g0 * g1 * g2 = 1*4/3 * 5/3 = 20/9
Path Branch effort B = 1
Path electrical effort H = Cout/Cin = 8C/C = 8
Path stage effort = F = G*B*H = 20/9*1*8 = 160/9

Min delay: $= N^* (F)^{1/N} + P$
$= 3 *(160/9)^{1/3} + (\text{pinv} + 2^*\text{pinv} + 2^*\text{pinv})$
$= 3 * 2.6 + 5 = 12.8$

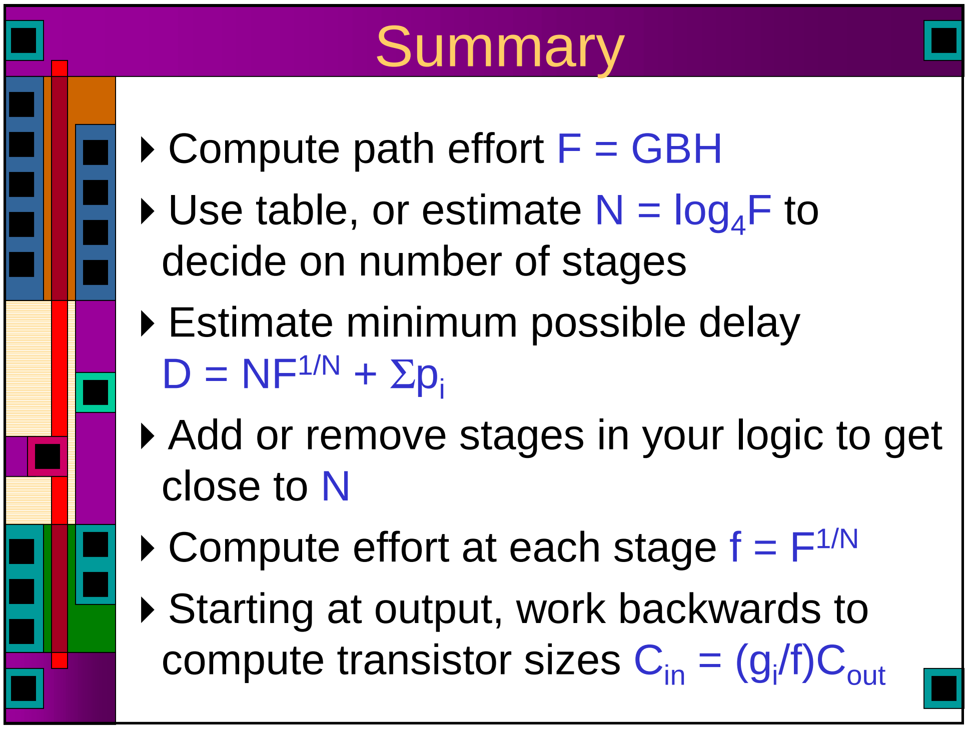
Option #2 appears to be better than Option #1, by a slight margin.



Choosing the Best # of Stages

- You can solve the delay equations to determing the number of stages N that will achieve the minimum delay
  - Approximate by Log_4F

Path Effort F	***Best N***	***Min Delay D***	***Stage effort f***
0-5.83	1	1.0-6.8	0-5.8
5.83-22.3	2	6.8-11.4	2.4-4.7
22.3-82.2	3	11.4-16.0	2.8-4.4
82.2-300	4	16.0-20.7	3.0-4.2
300-1090	5	20.7-25.3	3.1-4.1
1090-3920	6	25.3-29.8	3.2-4.0



Summary

- Compute path effort $F = GBH$
- Use table, or estimate $N = \log_4 F$ to decide on number of stages
- Estimate minimum possible delay $D = NF^{1/N} + \Sigma p_i$
- Add or remove stages in your logic to get close to N
- Compute effort at each stage $f = F^{1/N}$
- Starting at output, work backwards to compute transistor sizes $C_{in} = (g_i/f)C_{out}$