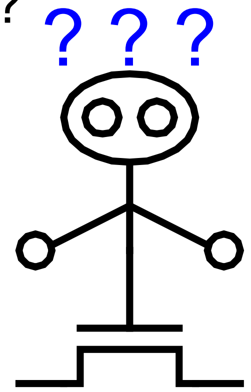
Logical Effort Outline

Introduction

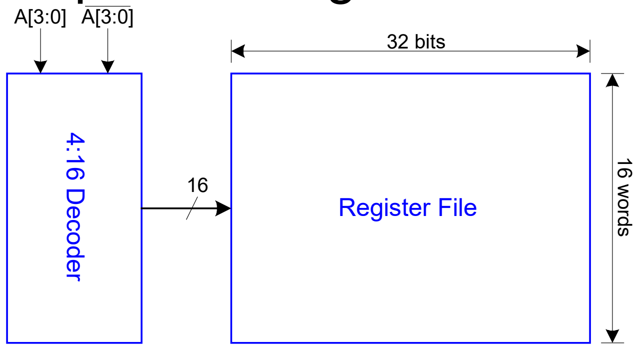
- ❑ Chip designers face a bewildering array of choices
  - (x) What is the best circuit topology for a function?
  - How many stages of logic give least delay?
  - How wide should the transistors be?



- ❑ 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 Motorola 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 (normalized) delays in process-independent unit

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

$\tau =$ 3RC is the delay of an ideal fanout of 1 inverter with no parasitic capacitance

$\approx$ 12 ps in 180 nm process

40 ps in 0.6 μm process

Delay in a Logic Gate (cont1)

- Express(normalized) delays in process-independent unit

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

- Delay has two components

$$d = f + p$$

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

Delay in a Logic Gate (cont2)

- Express(normalized) delays in process-independent unit

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

- Delay has two components

$$d = f + p$$

- Effort delay *f* = *gh* (also known as stage effort)
  - Again f has two components
    - *g*: *logical effort*
      - Measures relative ability of gate to deliver current
      - g $\equiv$ 1 for inverter

Delay in a Logic Gate (cont3)

- Express(normalized) delays in process-independent unit

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

- Delay has two components

$$d = f + p$$

- Effort delay *f* = *gh* (a.k.a. stage effort)
  - *g*: *logical effort*
  - *h*: *electrical effort* = C_{load} / C_{in}
    - Ratio of load capacitance to input capacitance
    - Sometimes called fanout

Delay in a Logic Gate (cont4)

- Express(normalized) delays in process-independent unit

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

- Delay has two components

$$d = f + p$$

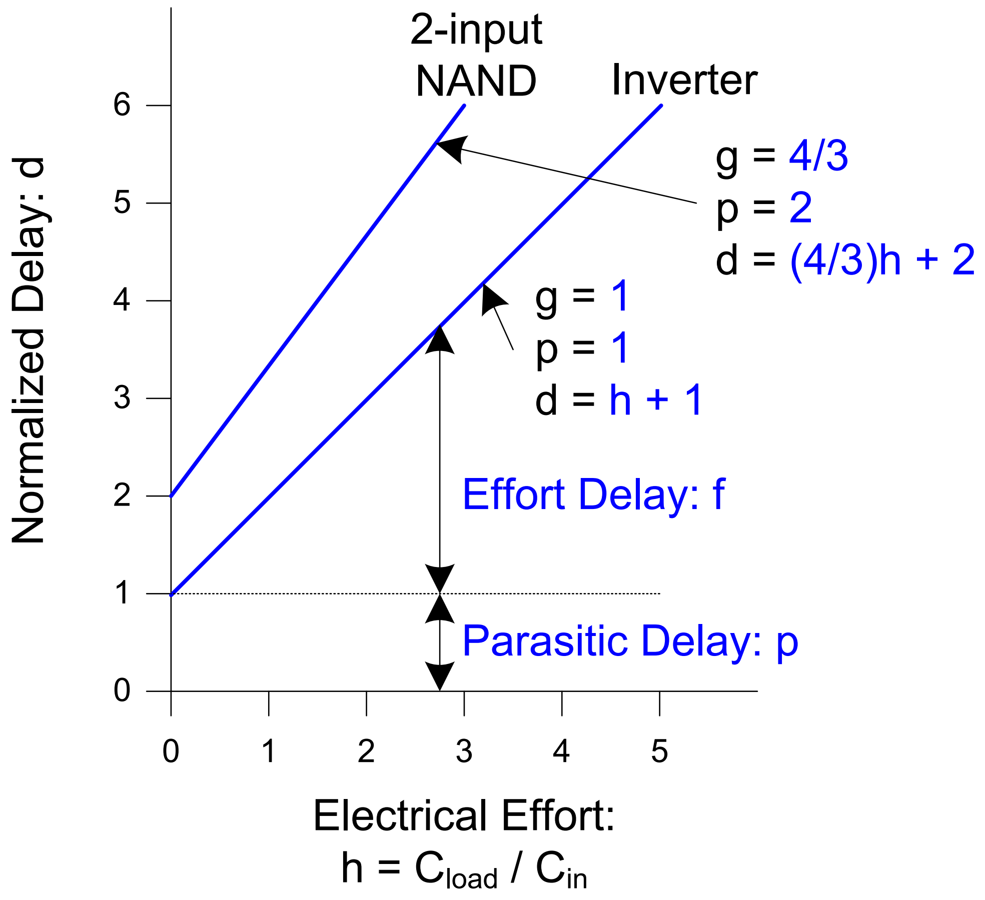
- Parasitic delay *p*
  - Represents delay of gate driving itself (no load)
  - Set by internal parasitic capacitance (C_{db}, C_{sb})

Delay Plots

$d = f + p = gh + p$

As recalled, the rising delay of 2-input NAND gate with h fanout is $d_{abs}=(4h+6)RC$

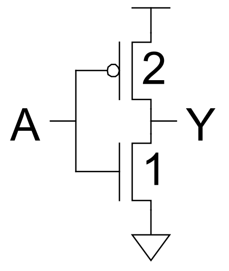
$$d = \frac{d_{abs}}{\tau} = \frac{(4h+6)RC}{3RC} = \left(4/3\right)h + 2$$



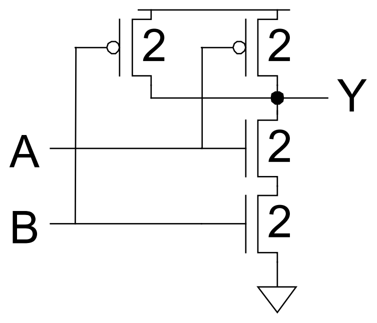
- ❑ What about $\overline{\text{NOR2}}$?
- ❑ What about (ab+c)?

Computing Logical Effort

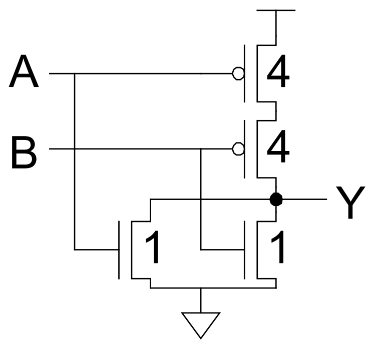
- Def: *Logical effort is the ratio of input capacitance of the analyzed gate to input capacitance of an inverter delivering the same output current (same effective resistance).*
- Measure from delay vs. fanout plots; g=slope
- Or estimate by counting transistor widths



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



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



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

Catalog of Gates

- Logical effort (g) 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	

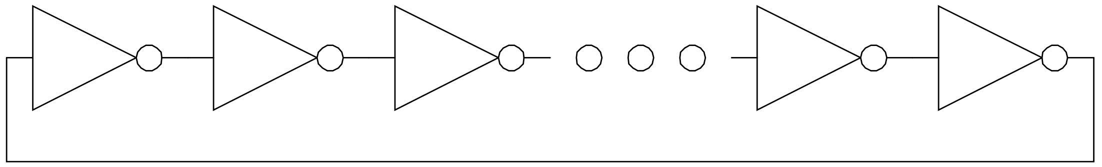
Catalog of Gates

- 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: Ring Oscillator

- Estimate the frequency of an N-stage ring oscillator



Logical Effort: $g =$

Electrical Effort: $h =$

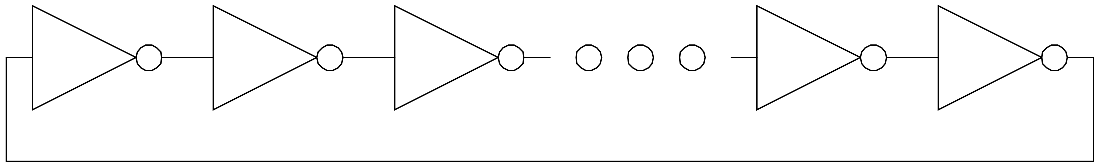
Parasitic Delay: $p =$

Stage Delay: $d =$

Frequency: $f_{osc} =$

Example: Ring Oscillator

- Estimate the frequency of an N-stage ring oscillator

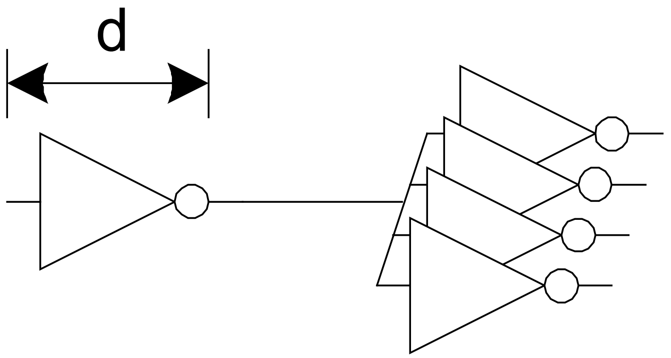


Logical Effort:	$g = 1$
Electrical Effort:	$h = 1$
Parasitic Delay:	$p = 1$
Stage Delay:	$d = 2$
Frequency:	$f_{osc} = 1/(2*N*d) = 1/(4N)$

31 stage ring oscillator in 0.6 μm process has frequency of ~ 200 MHz

Example: FO4 Inverter

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



Logical Effort: g =

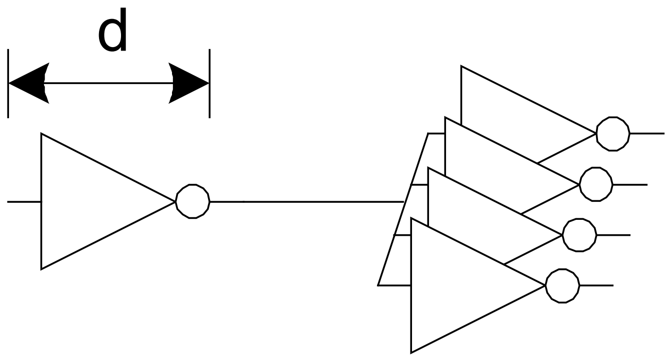
Electrical Effort: h =

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

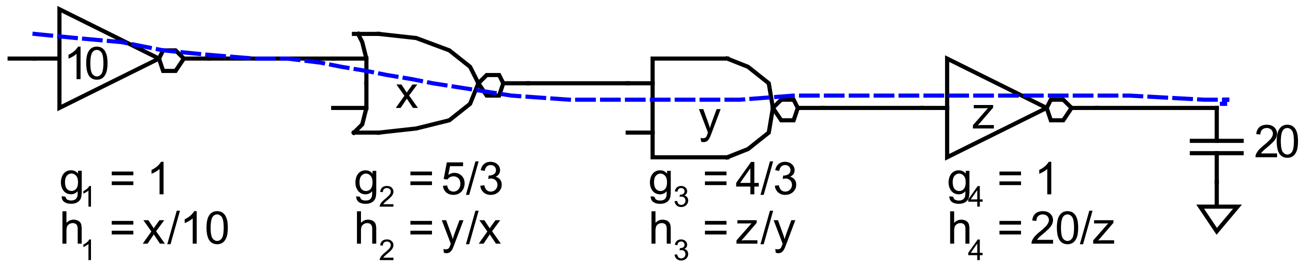
200 ps in 0.6 μm process

60 ps in a 180 nm process

f(1/3—1/2) ps in an *f* nm process

Multistage Logic Networks

- Logical effort generalizes to multistage networks
- *Path Logical Effort* $G = \prod g_i$
- *Path Electrical Effort* $H = \dfrac{C_{load-path}}{C_{in-path}}$
- *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* $H = \frac{C_{load-path}}{C_{in-path}}$
- *Path Effort* $F = \prod f_i = \prod g_i h_i$
- Can we write F = GH?

Paths that Branch

- No! Consider paths that branch:

$G =$

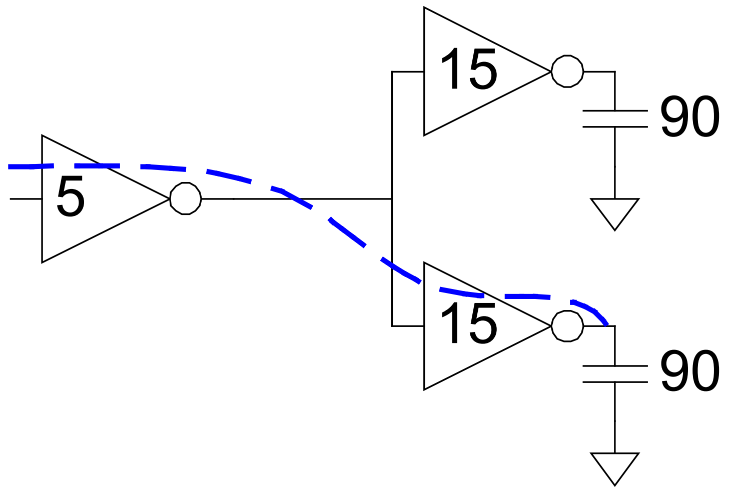
$H =$

$GH =$

$h_1 =$

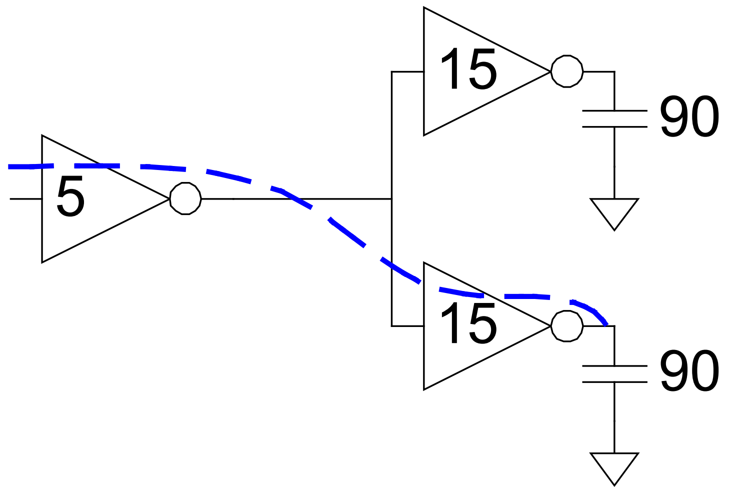
$h_2 =$

$F = GH?$



Paths that Branch

- No! Consider paths that branch:



$G = 1(1) = 1$

$H = 90 / 5 = 18$

$GH = 18$

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

$h_2 = 90 / 15 = 6$

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

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 the smallest when each stage bears same effort

$$\hat{f} = g_i h_i = F^{\frac{1}{N}} \qquad \text{best stage effort}$$

- Thus minimum delay of N stage path is

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

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

Gate Sizes

- How wide should the gates be for least delay?

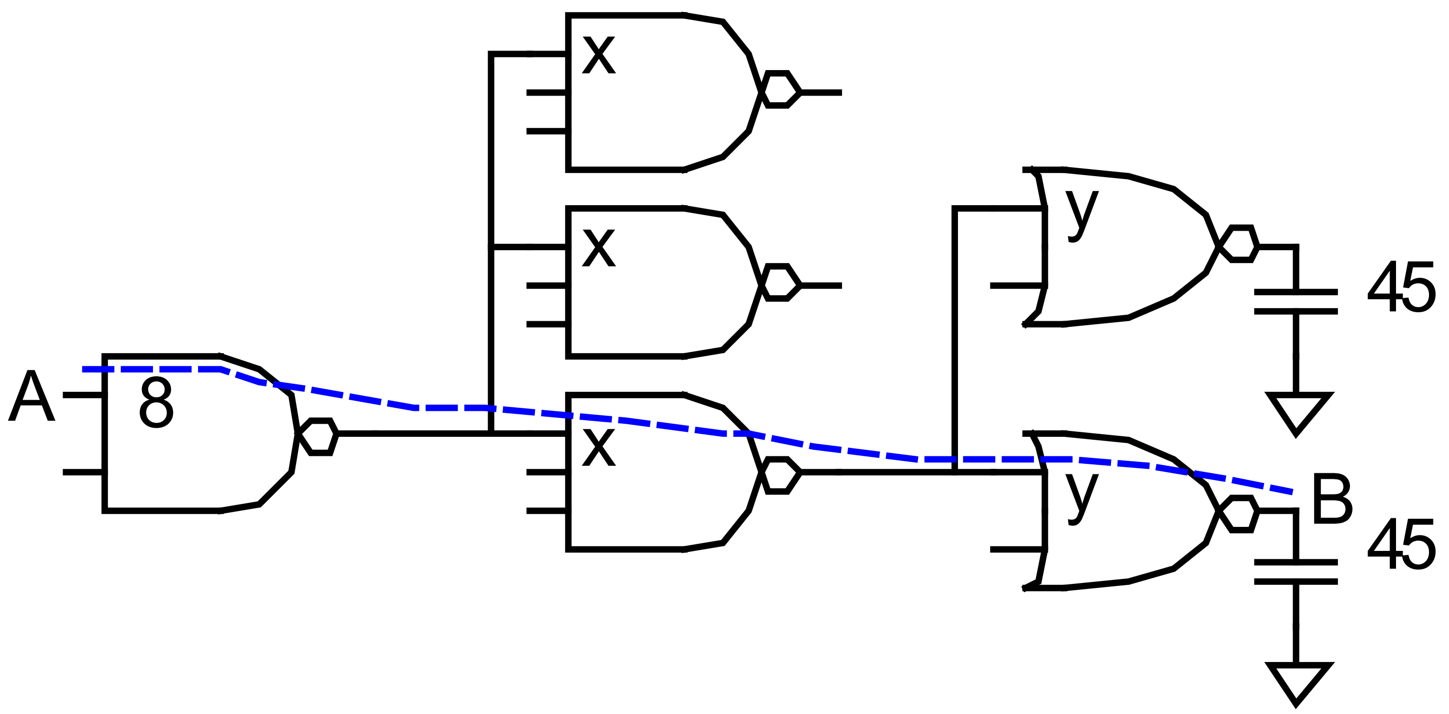
$$\hat{f} = gh = f_i = g_i \frac{C_{load}(i)}{C_{in}(i)}$$

$$\Rightarrow C_{in}(i) = \frac{g_i C_{load}(i)}{\hat{f}}$$

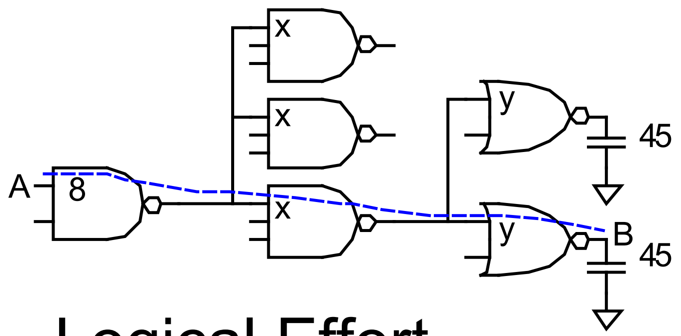
- Working backward, apply capacitance transformation to find input capacitance of each gate given load it drives.
- Check work by verifying input cap spec is met.

Example: 3-stage path

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

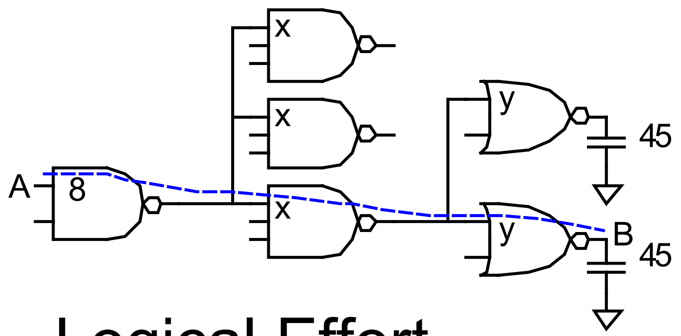


Example: 3-stage path



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

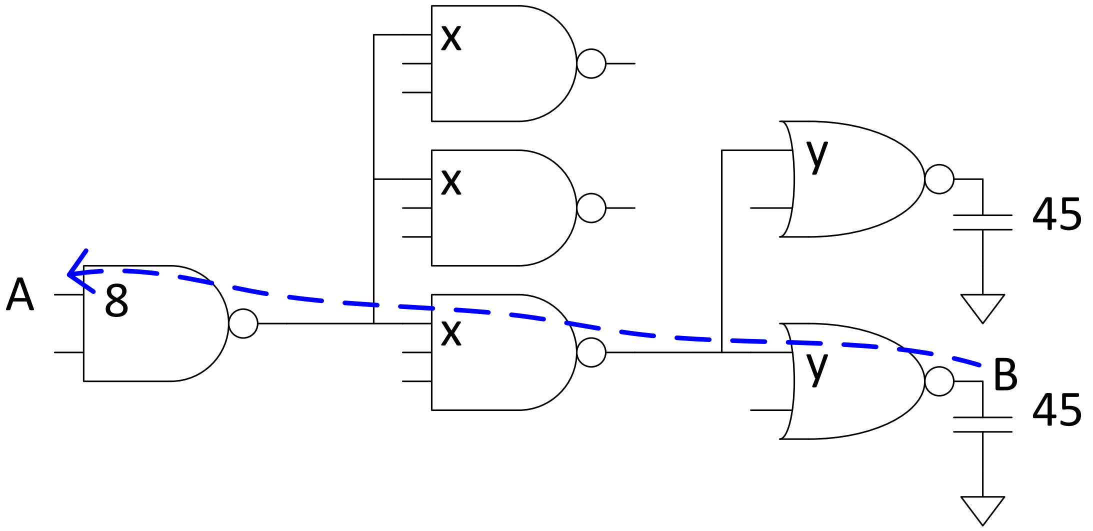
Example: 3-stage path

- Work backward for sizes

  y =

  x =

$$Cin_i = (W_p + W_n) = \frac{g_i C_{load\,i}}{\hat{f}}$$



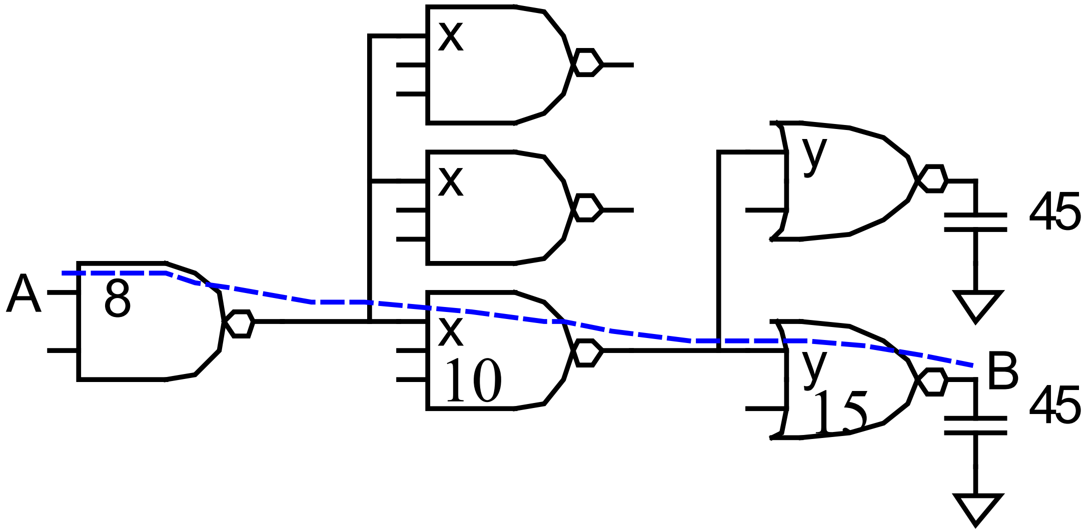
Example: 3-stage path

- Work backward for sizes

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

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

$$Cin_i = (W_p + W_n)C = \frac{g_i C_{load\,i}}{\hat{f}}$$



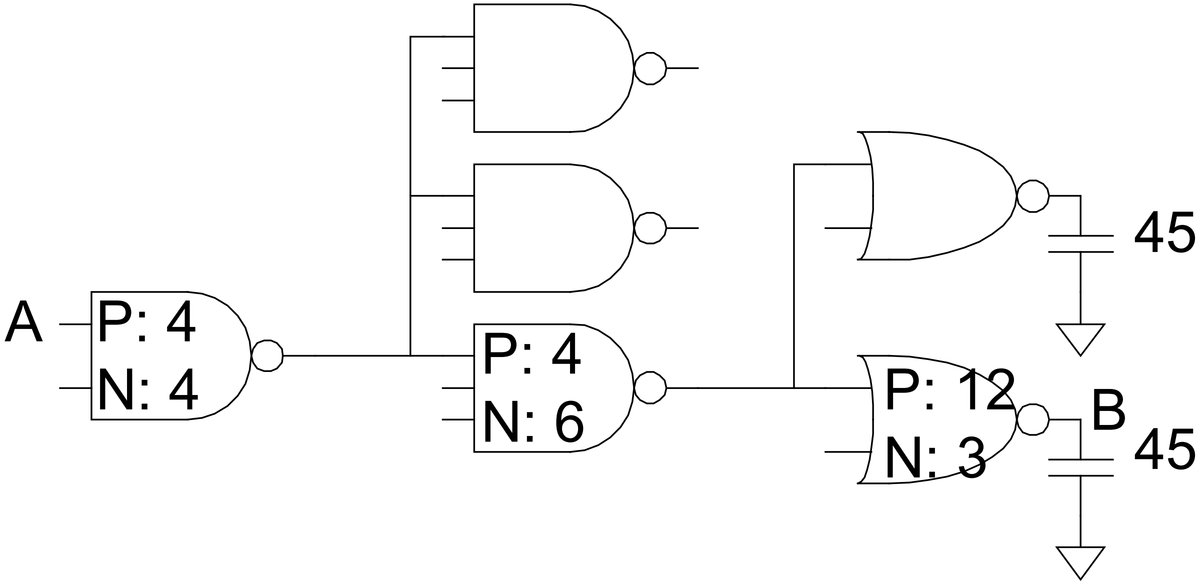
Example: 3-stage path

- Work backward for sizes

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

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

$$Cin_i = (W_p + W_n) = \frac{g_i C_{load\,i}}{\hat{f}}$$



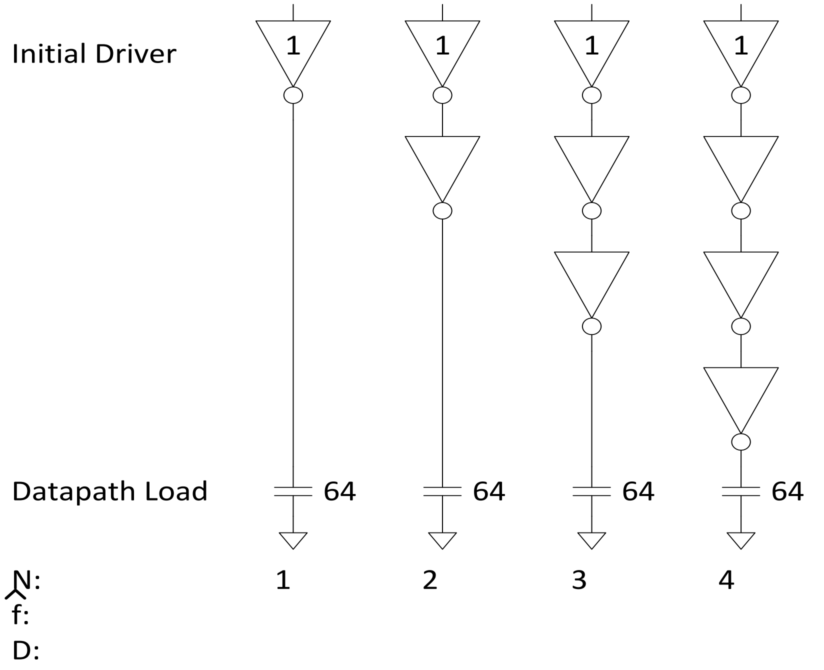
Best Number of Stages

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

- G=1

$$H = \frac{64}{1} = 64$$

F=GBH=64



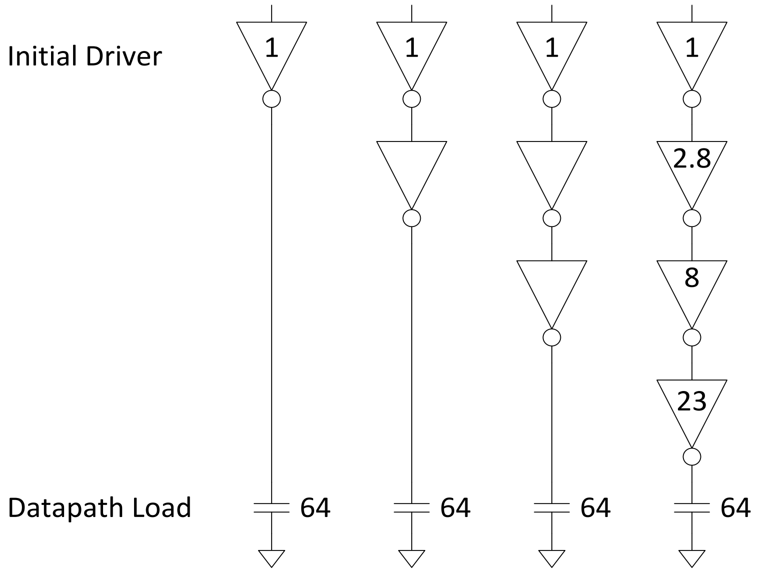
Best Number of Stages

- 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

Best stage effort: $\hat{f} = F^{1/N}$

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



N :	1	2	3	4
$\hat{f}$:	64	8	4	2.8
D :	65	18	15	15.2
			Fastest	

Best Number of Stages

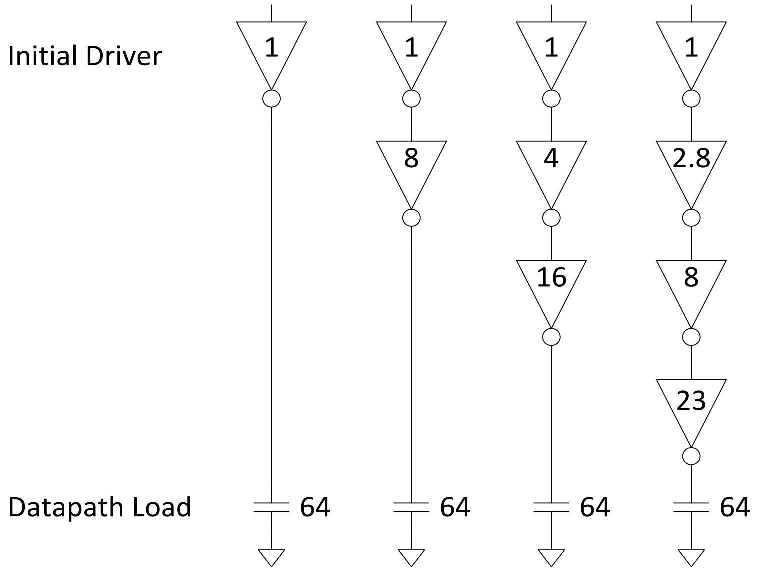
- ❑ How many stages should a path use?
  - – Minimizing number of stages is not always fastest
- ❑ Example: drive 64-bit datapath with unit inverter

F=64

Best stage effort: $\hat{f} = F^{1/N}$

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

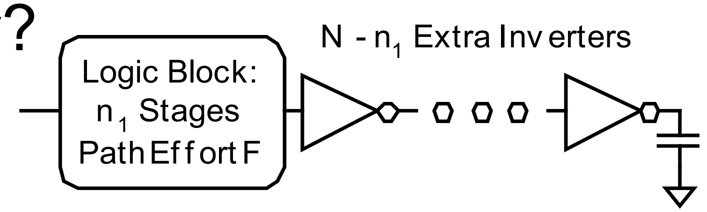
$$= N(64)^{1/N} + N$$



N :	1	2	3	4
$\hat{f}$:	64	8	4	2.8
D :	65	18	15	15.2
			Fastest	

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

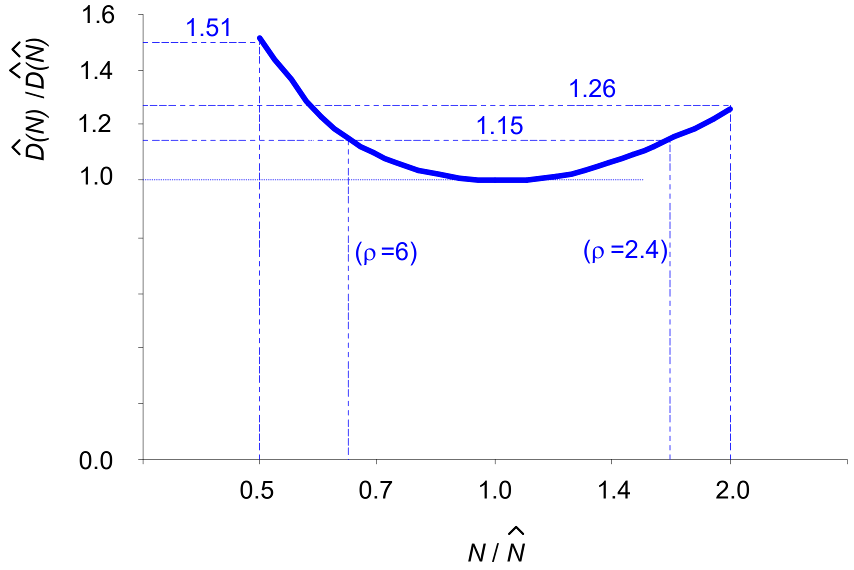
- $p_{inv} + \rho\left(1 - \ln\rho\right) = 0$ has no closed-form solution

- For $p_{inv} = 1$, solve numerically for $\rho = 3.59$
- A path achieves the least delay by using the number of stages

$$\hat{N} = \log_{\rho} F$$

Sensitivity Analysis

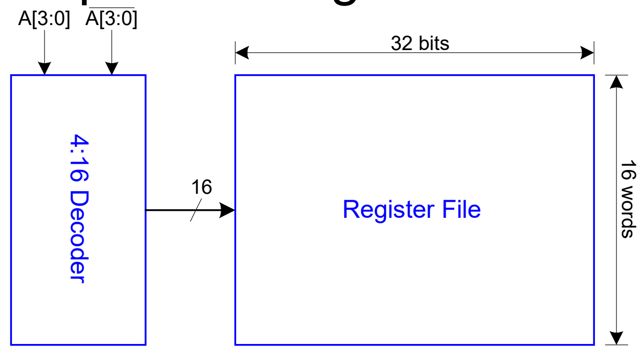
- How sensitive is delay to using exactly the best number of stages?



- $2.4 < \rho < 6$ gives delay within 15% of optimal
  - $\rho = 4$ is a reasonable number

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:	$H = (32*3)/10 = 9.6$
Branching Effort:	$B = 16/2 = 8$

  - 16 bit decoder will be wired to either the address (A3-A0) or the address not (A3'-A0'), two options.

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

Path Effort:	$F = GBH = 76.8$
Number of Stages:	$N = \log_4 F = 3.1$

- ❑ Try a 3-stage design

Gate Sizes & Delay

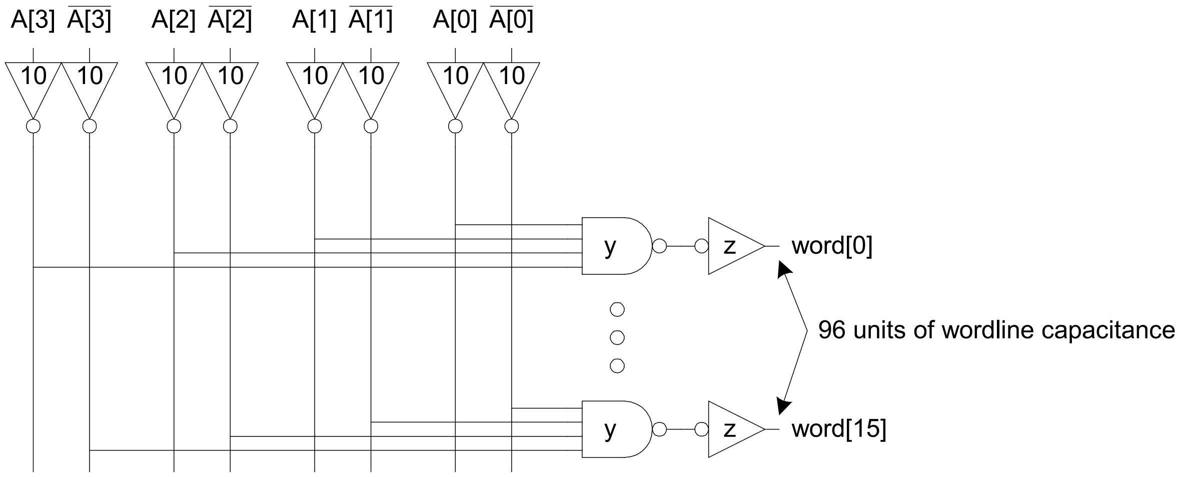
Logical Effort: G =

Path Effort: F =

Stage Effort: $\hat{f} =$

Path Delay: $D =$

Gate sizes: z = y =



Gate Sizes & Delay

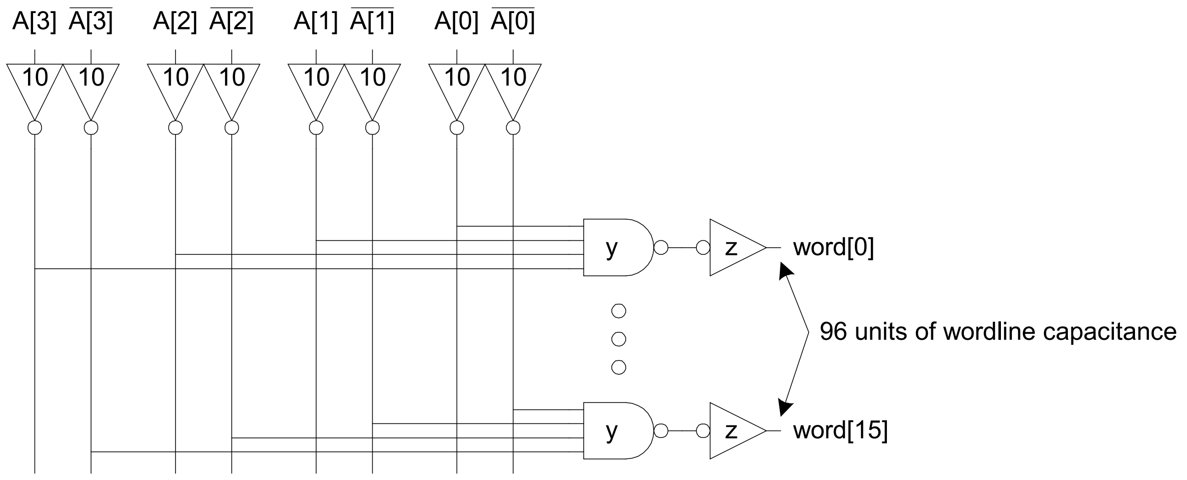
Logical Effort: $G = 1 * 6/3 * 1 = 2$

Path Effort: $F = GBH = 2*8*9.6 = 154$

Stage Effort: $\hat{f} = F^{1/3} = 5.36$

Path Delay: $D = 3\hat{f} + 1 + 4 + 1 = 22.1$

Gate sizes: $z = 96*1/5.36 = 18$ $y = 18*2/5.36 = 6.7$



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	1	N
logical effort	g	$G = \prod g_i$
electrical effort	$h = \frac{C_{load}}{C_{in}}$	$H = \frac{C_{load-path}}{C_{in-path}}$
branching effort	$b = \frac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{on-path}}}$	$B = \prod b_i$
effort	$f = gh$	$F = GBH = \prod f_i$
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 $C_{in_i} = \frac{g_i C_{L_i}}{\hat{f}}$
7) Using rising effective resistance equals falling effective resistance to find N and P transistors sizes

Limits of Logical Effort

- Linear delay model fails to capture the effort of input slope
- Logical Effort does not account for Interconnect
  - Iteration required in designs with wire
- Logical effort is most applicable to high speed circuit with regular layouts where routing delay does not dominate
- Maximum speed only
  - Not minimum area/power for constrained delay
- Paths with complex branching are difficult to analyze by hand.

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 stage effort is ~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
  - Practical limit of about *4 transistors in series and 4 inputs to multiplexers*
- Provides language for discussing fast circuits

HW

- 4.3, 4.4, 4.6, 4.9, 4.10, 4.11