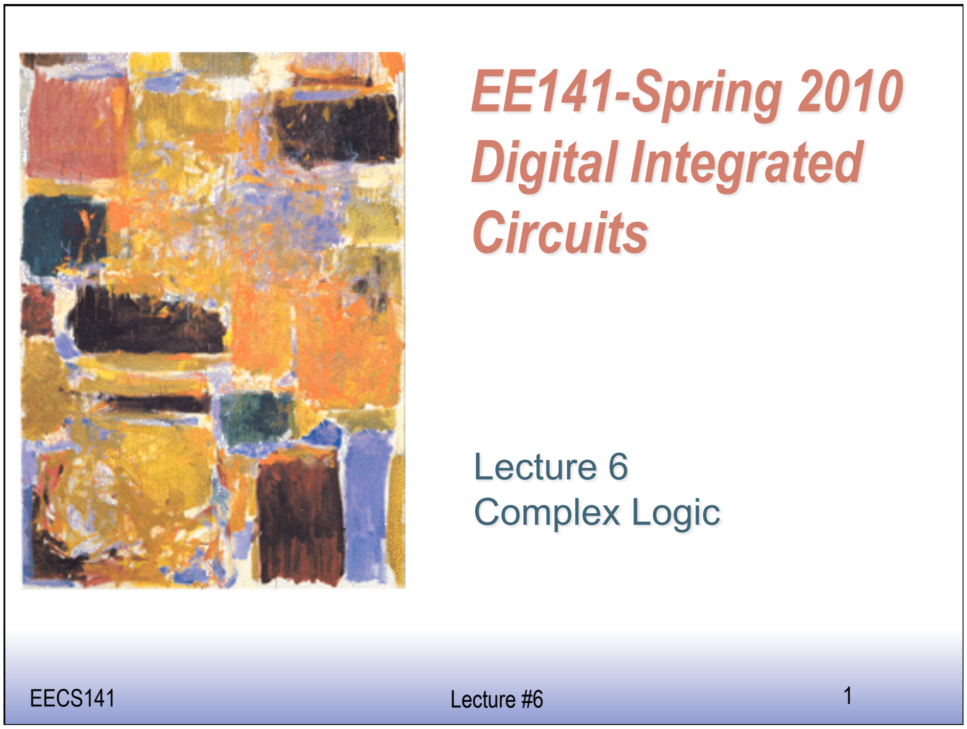
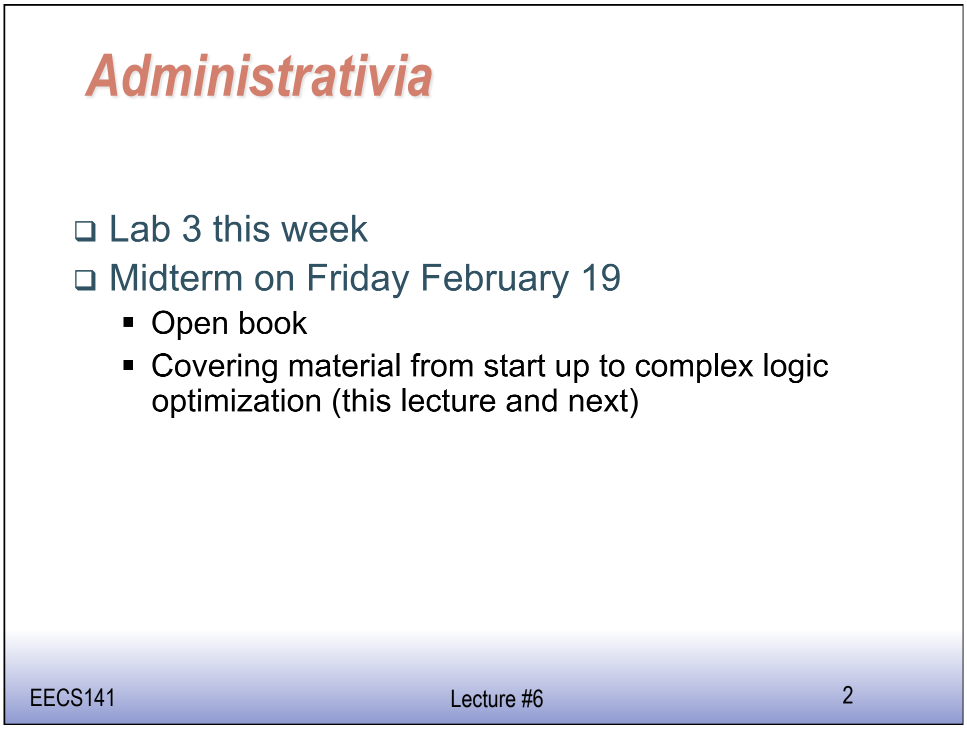


EE141-Spring 2010 Digital Integrated Circuits

Lecture 6
Complex Logic



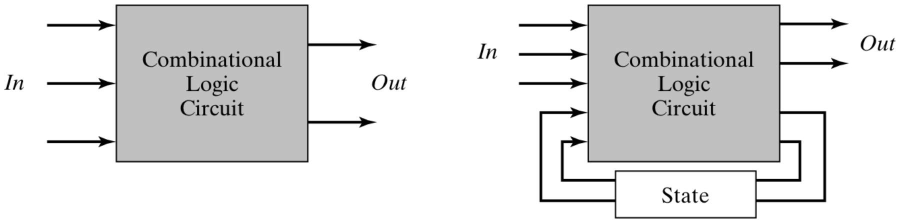
Administrativia

- ❑ Lab 3 this week
- ❑ Midterm on Friday February 19
  - Open book
  - Covering material from start up to complex logic optimization (this lecture and next)

Class Material

- ❑ Last lecture
  - ▪ Sizing inverters
- ❑ Today’s lecture
  - ▪ Complex logic
  - ▪ Optimizing complex logic
- ❑ Reading (2.3, 3.3.1-3.3.2)

Combinational vs. Sequential Logic

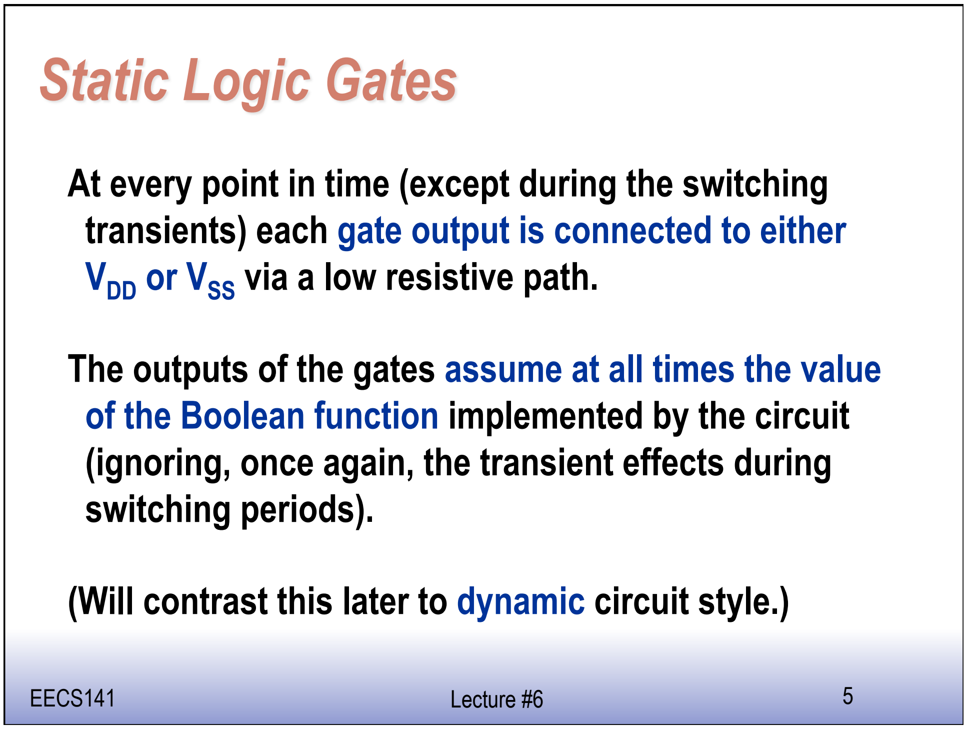


(a) Combinational

(b) Sequential

Output = ***f*****(*****In*****)**

Output = ***f*****(*****In, Previous In*****)**

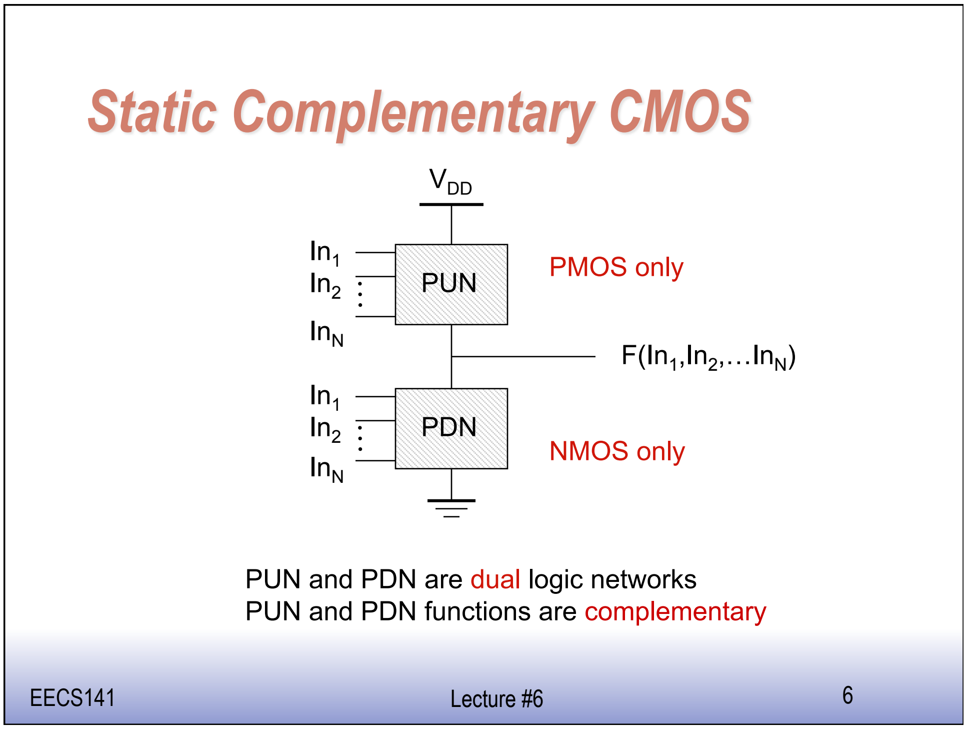


Static Logic Gates

At every point in time (except during the switching transients) each gate output is connected to either V_{DD} or V_{SS} via a low resistive path.

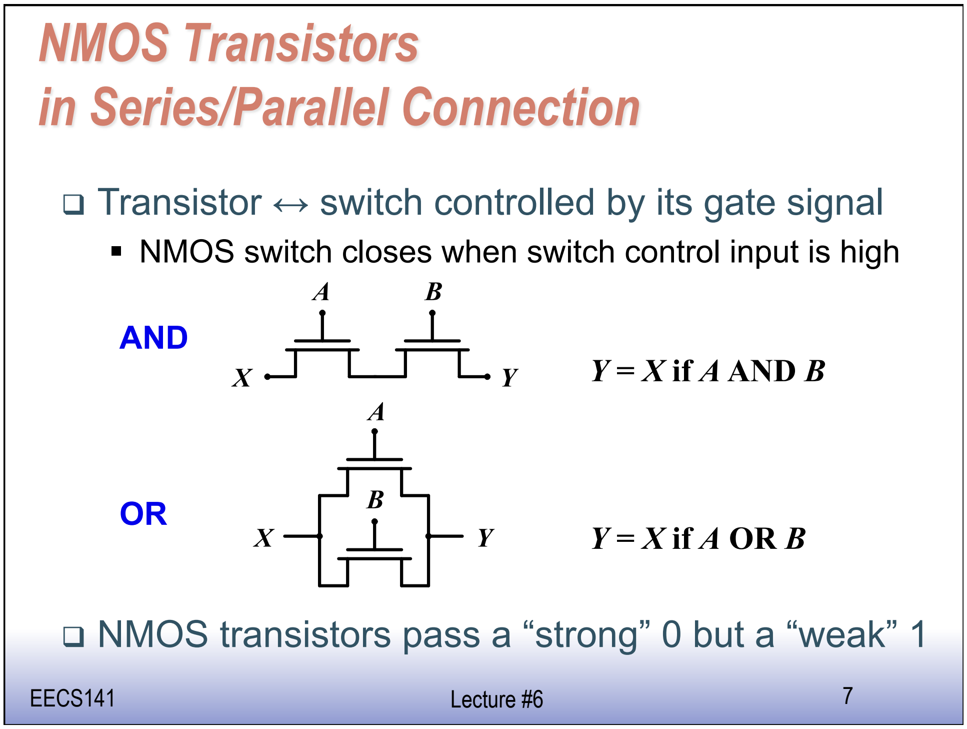
The outputs of the gates assume at all times the value of the Boolean function implemented by the circuit (ignoring, once again, the transient effects during switching periods).

(Will contrast this later to dynamic circuit style.)



Static Complementary CMOS

PUN and PDN are dual logic networks
PUN and PDN functions are complementary



NMOS Transistors in Series/Parallel Connection

- ❑ Transistor ↔ switch controlled by its gate signal
  - NMOS switch closes when switch control input is high

AND

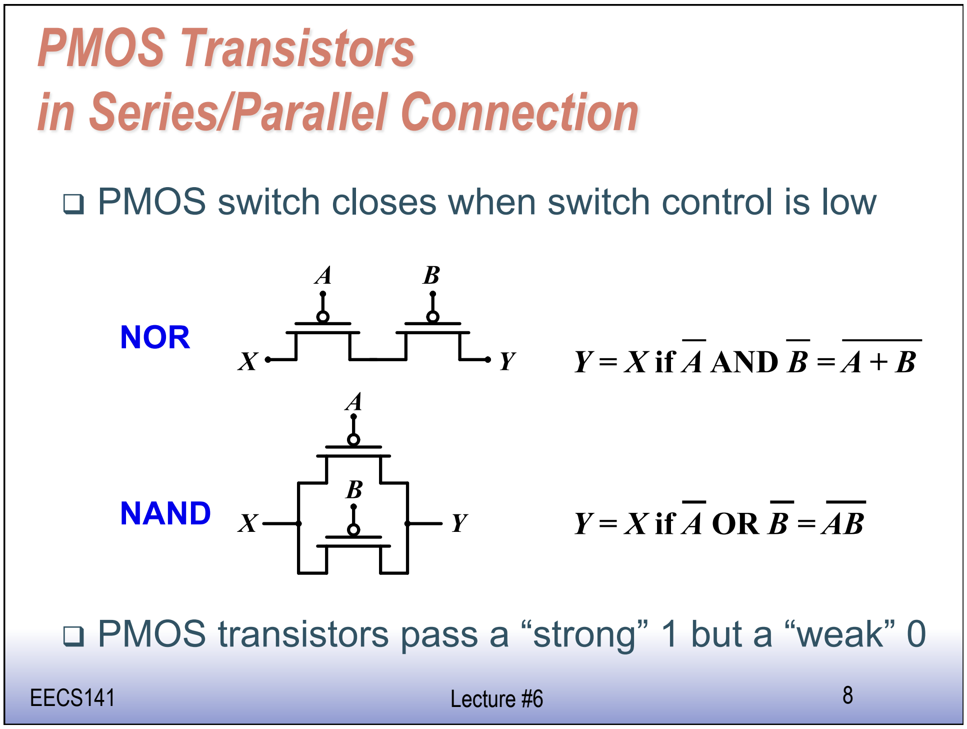
$Y = X$ if A AND B

OR

$Y = X$ if A OR B

- ❑ NMOS transistors pass a "strong" 0 but a "weak" 1

EECS141 Lecture #6 7



PMOS Transistors in Series/Parallel Connection

- ❑ PMOS switch closes when switch control is low

NOR

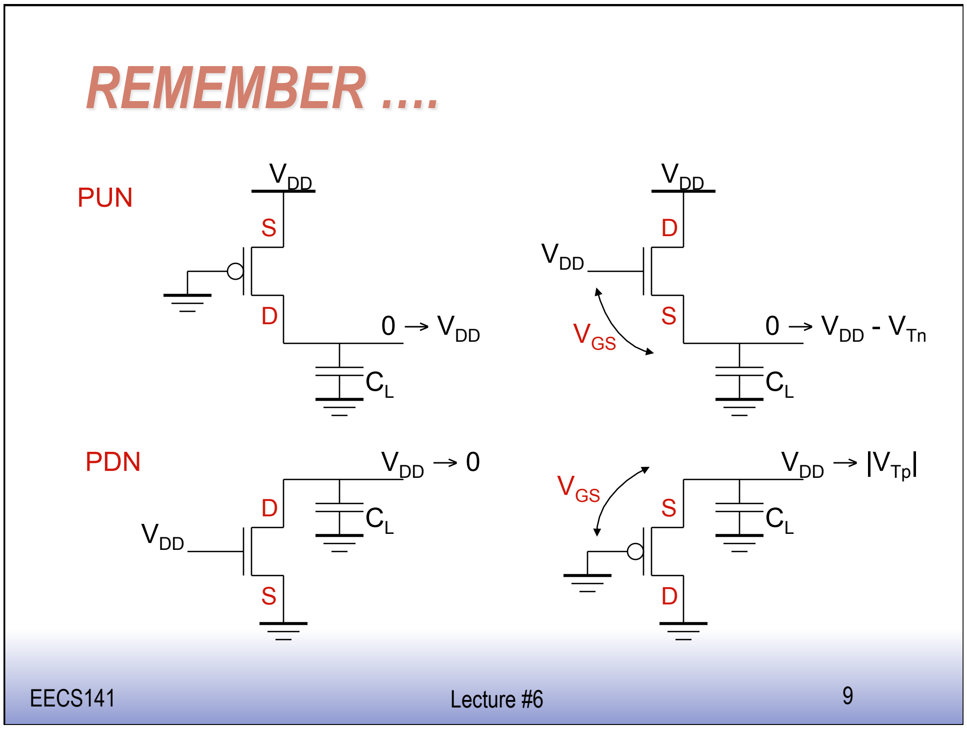
$Y = X \text{ if } \overline{A} \text{ AND } \overline{B} = \overline{A + B}$

NAND

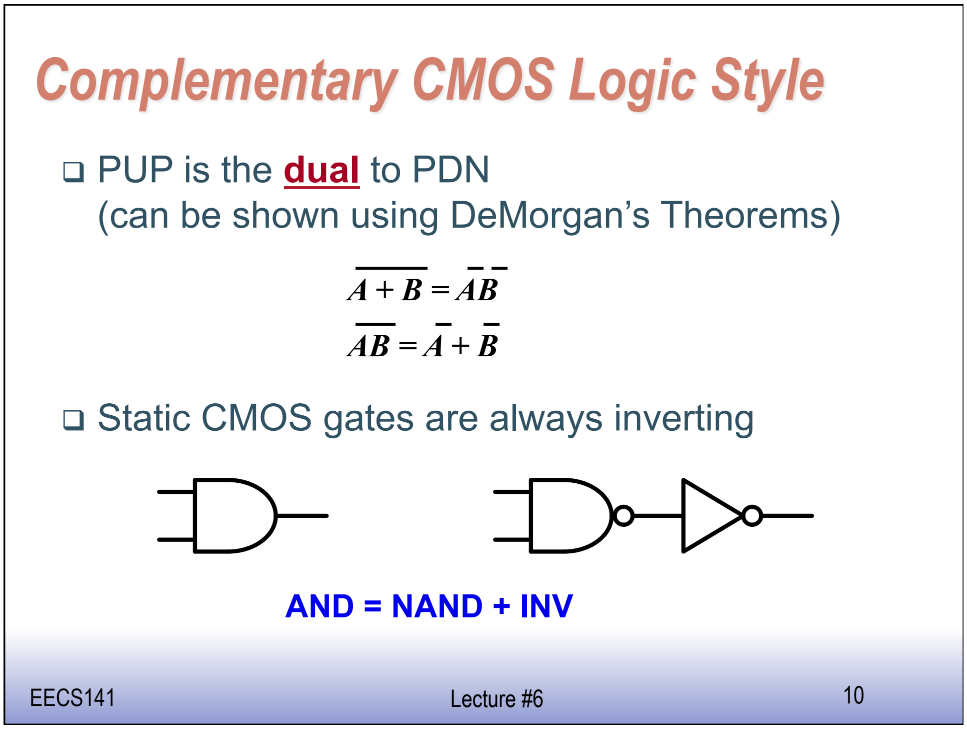
$Y = X \text{ if } \overline{A} \text{ OR } \overline{B} = \overline{AB}$

- ❑ PMOS transistors pass a "strong" 1 but a "weak" 0

EECS141 Lecture #6 8



REMEMBER ....



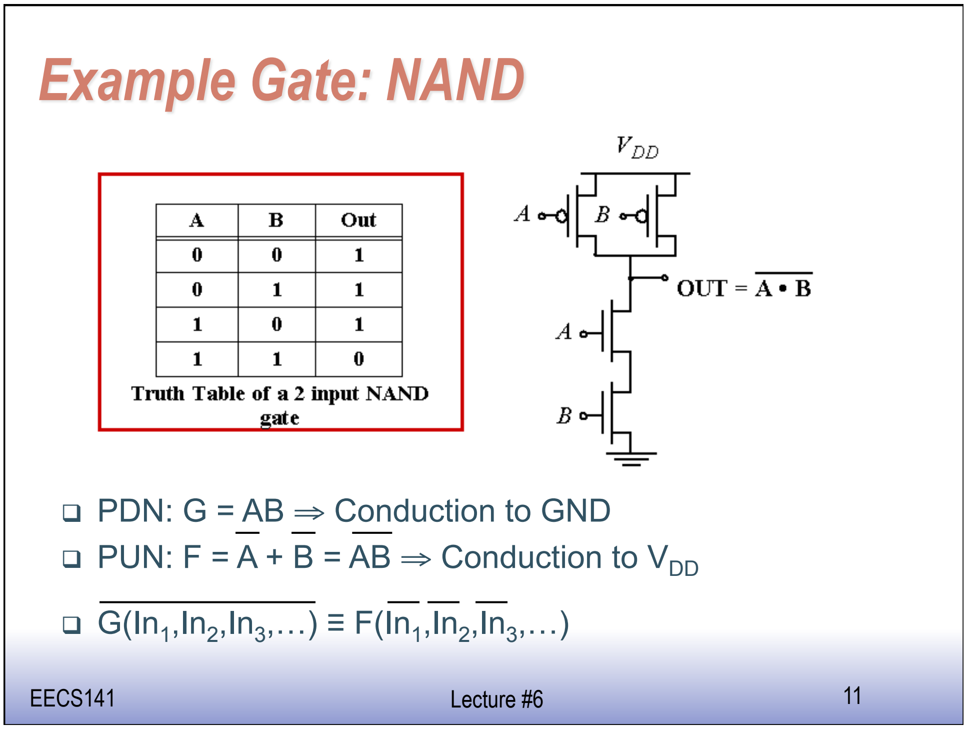
Complementary CMOS Logic Style

- PUP is the **dual** to PDN
  (can be shown using DeMorgan's Theorems)

$$\overline{A+B} = \bar{A}\bar{B}$$
$$\overline{AB} = \bar{A}+\bar{B}$$

- Static CMOS gates are always inverting

AND = NAND + INV



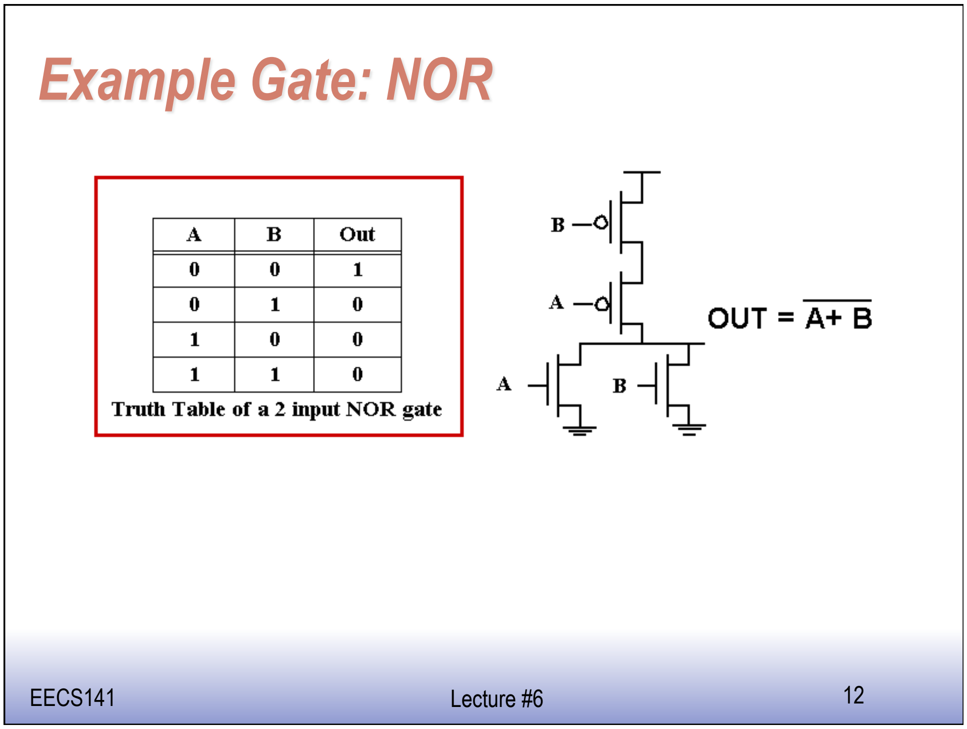
Example Gate: NAND

A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0

Truth Table of a 2 input NAND gate

- ❑ PDN: G = AB ⇒ Conduction to GND
- ❑ PUN: F = $\overline{A} + \overline{B} = \overline{AB}$ ⇒ Conduction to V_{DD}
- ❑ $\overline{G(In_1,In_2,In_3,\ldots)} \equiv F(\overline{In_1},\overline{In_2},\overline{In_3},\ldots)$

EECS141 Lecture #6 11



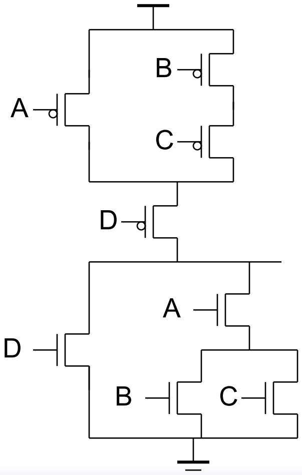
Example Gate: NOR

A	B	Out
0	0	1
0	1	0
1	0	0
1	1	0

Truth Table of a 2 input NOR gate

EECS141 Lecture #6 12

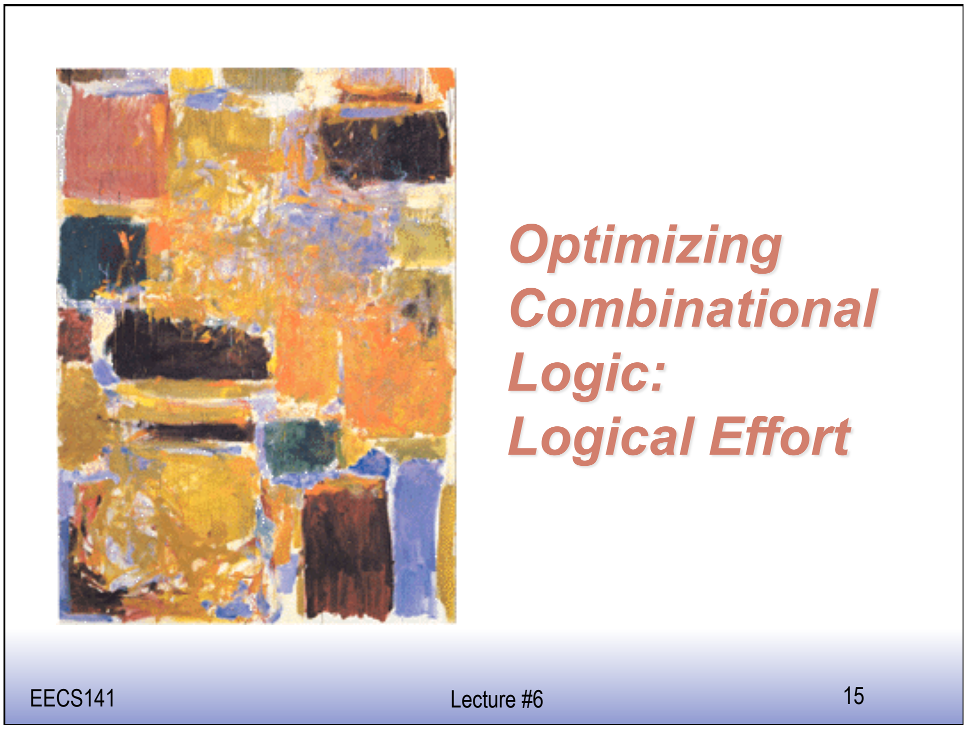
Complex CMOS Gate



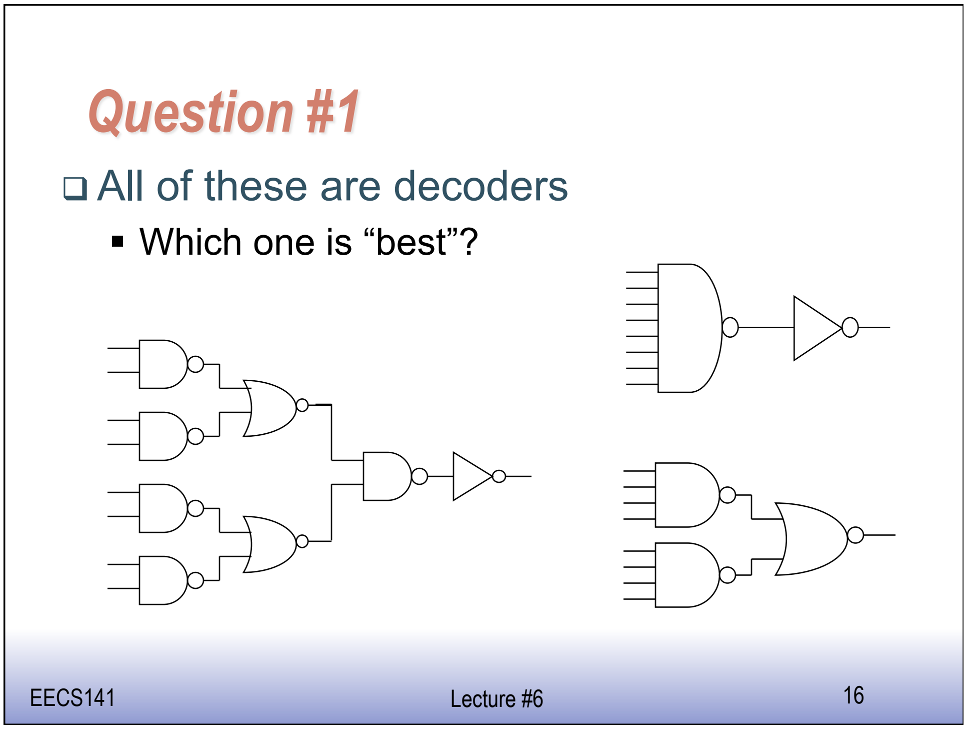
OUT = $\overline{D + A \cdot (B + C)}$

CMOS Properties

- ❑ Full rail-to-rail swing
- ❑ Symmetrical VTC
- ❑ Propagation delay function of load capacitance and resistance of transistors
- ❑ No static power dissipation
- ❑ Direct path current during switching



Optimizing Combinational Logic: Logical Effort

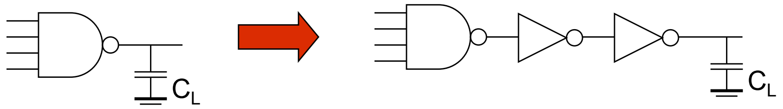


Question #1

- ❑ All of these are decoders
  - Which one is “best”?

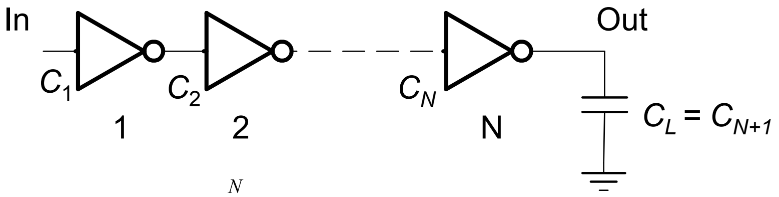
Question #2

- Is it better to drive a big capacitive load directly with the NAND gate, or after some buffering?



- Method to answer both of these questions:
  - Logical effort
  - Extension of buffer sizing problem

Recap: Buffer Sizing



$$Delay = t_{inv} \sum_{i=1}^{N} (\gamma + f_i)$$

$\boxed{f_i = C_{i+1}/C_i}$

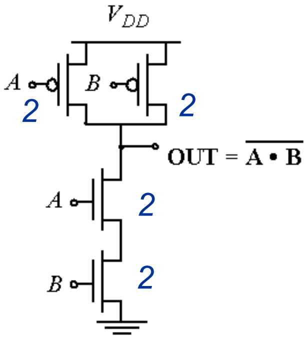
For given N: $C_{i+1}/C_i = C_i/C_{i-1}$
To find N: $C_{i+1}/C_i \sim 4$
How to generalize this to any logic path?

Delay Of NAND Gate

$C_{dnand} = 6C_D$

$C_{gnand} = 4C_G = (4/3)\ C_{ginv}$

$C_D/C_G = \gamma$



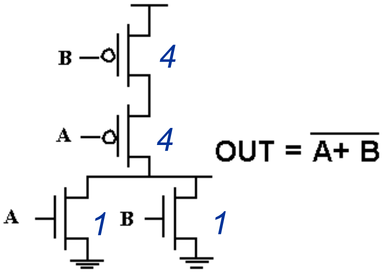
$$\begin{aligned} t_{pNAND} &= kR(C_{int} + C_L) \\ &= k(R_{min}/W)(WC_{dnand} + C_L) \\ &= k(R_{min}C_{gnand})(C_{dnand}/C_{gnand} + C_L/(WC_{gnand})) \\ &= k(R_{min}C_{gnand})(\gamma 3/2 + C_L/(WC_{gnand})) \\ &= t_{inv}\ (2\gamma + (4/3)f) \end{aligned}$$

Delay Of NOR Gate

$C_{dnor} = 6C_D$

$C_{gnor} = 5C_G = (5/3)\ C_{ginv}$

$C_D/C_G = \gamma$



$$\begin{aligned} t_{pNAND} &= kR(C_{int} + C_L) \\ &= k(R_{min}/W)(WC_{dnor} + C_L) \\ &= k(R_{min}C_{gnor})(C_{dnor}/C_{gnor} + C_L/(WC_{gnor})) \\ &= k(R_{min}C_{gnor})(\gamma 6/5 + C_L/(WC_{gnor})) \\ &= t_{inv}\ (2\gamma + (5/3)f) \end{aligned}$$

Logical Effort

$$\boxed{t_{pgate} = t_{inv}\,(p\gamma + LE \times f)}$$

Measure everything in units of t_{inv} (divide by t_{inv}):

p – intrinsic delay ($k\gamma_g$) - gate parameter $\neq$ f(W)
LE – logical effort (k) – gate parameter $\neq$ f(W)
f – electrical effort (effective fanout)

Normalize everything to an inverter:
$LE_{inv} = 1$, $p_{inv} = \gamma$

Delay in a Logic Gate

Gate delay:

Delay* = *EF* + *p **(measured in units of t_{inv})**

effective fanout **intrinsic delay**

Effective fanout:

EF = LE f

logical effort **electrical fanout = C_{out}/C_{in}**

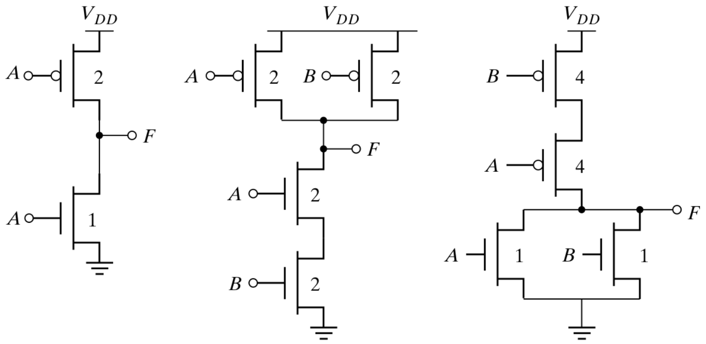
Logical effort is a function of topology, independent of sizing
Effective fanout is a function of load/gate size

Logical Effort

- Inverter has the smallest logical effort and intrinsic delay of all static CMOS gates
- Logical effort LE is defined as:
  - $(R_{eq,gate}C_{in,gate})/(R_{eq,inv}C_{in,inv})$
  - Easiest way to calculate (usually):
    - Size gate to deliver same current as an inverter, take ratio of gate input capacitance to inverter capacitance
- LE increases with gate complexity

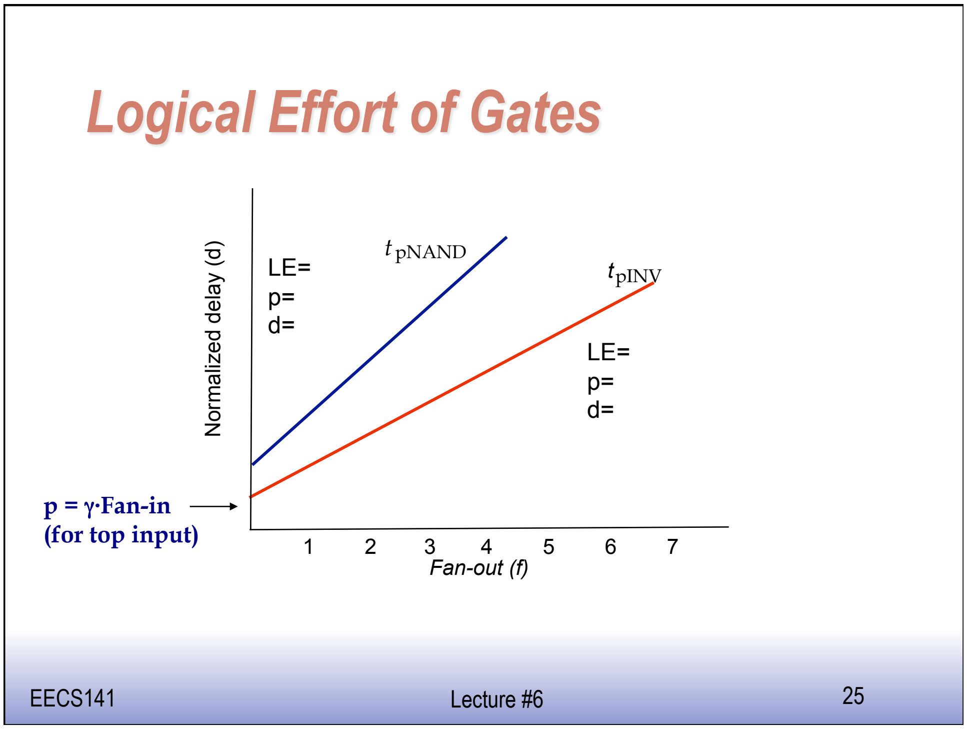
Logical Effort

Calculating LE by sizing for same drive strength:



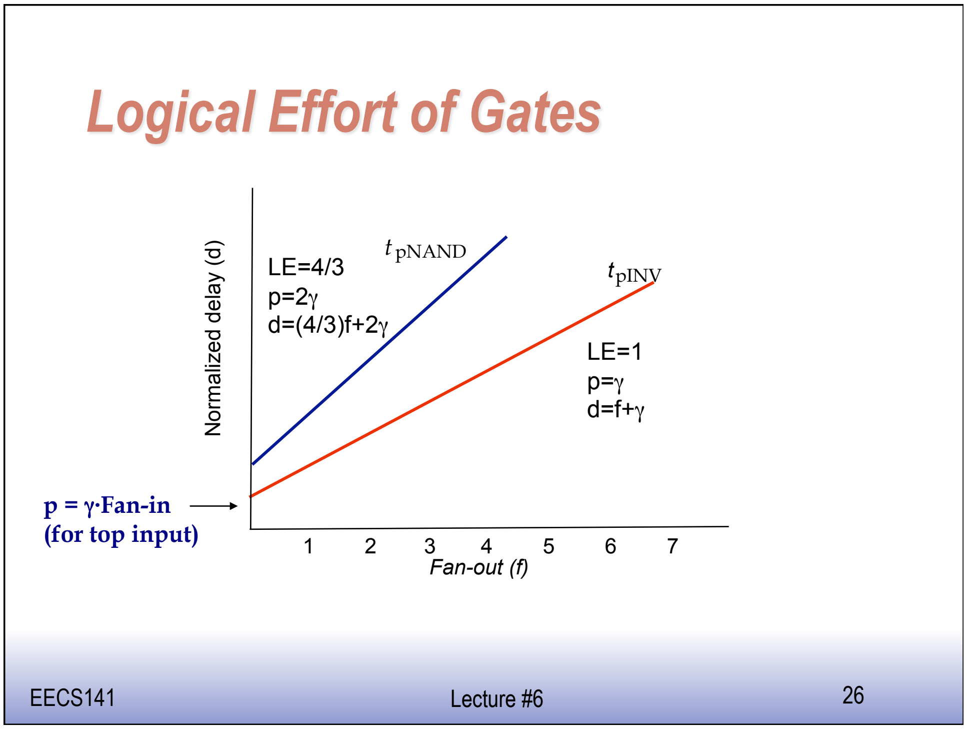
Inverter Two-input NAND Two-input NOR

LE **= 1** ***LE*** **= 4/3** ***LE*** **= 5/3**



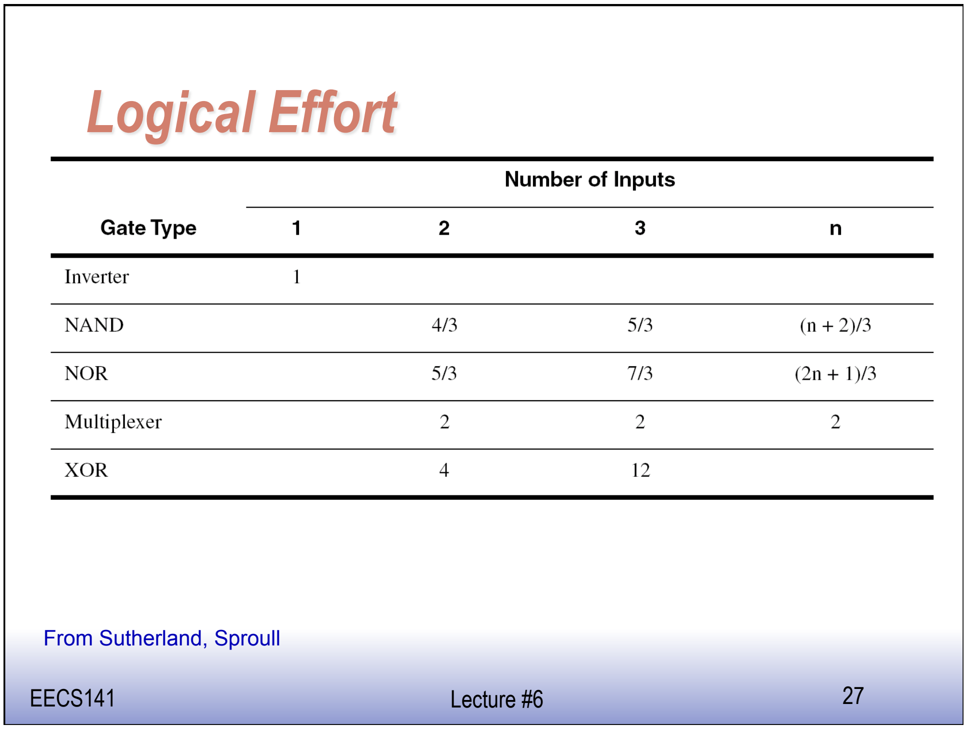
Logical Effort of Gates

p = γ·Fan-in
(for top input)



Logical Effort of Gates

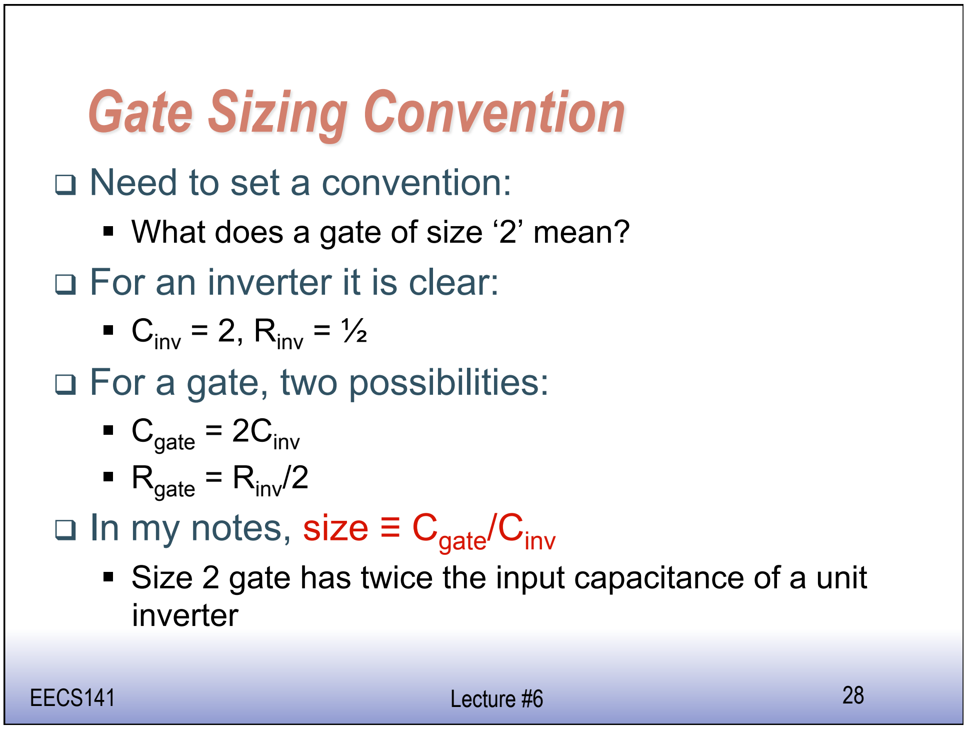
p = γ·Fan-in
(for top input)



Logical Effort

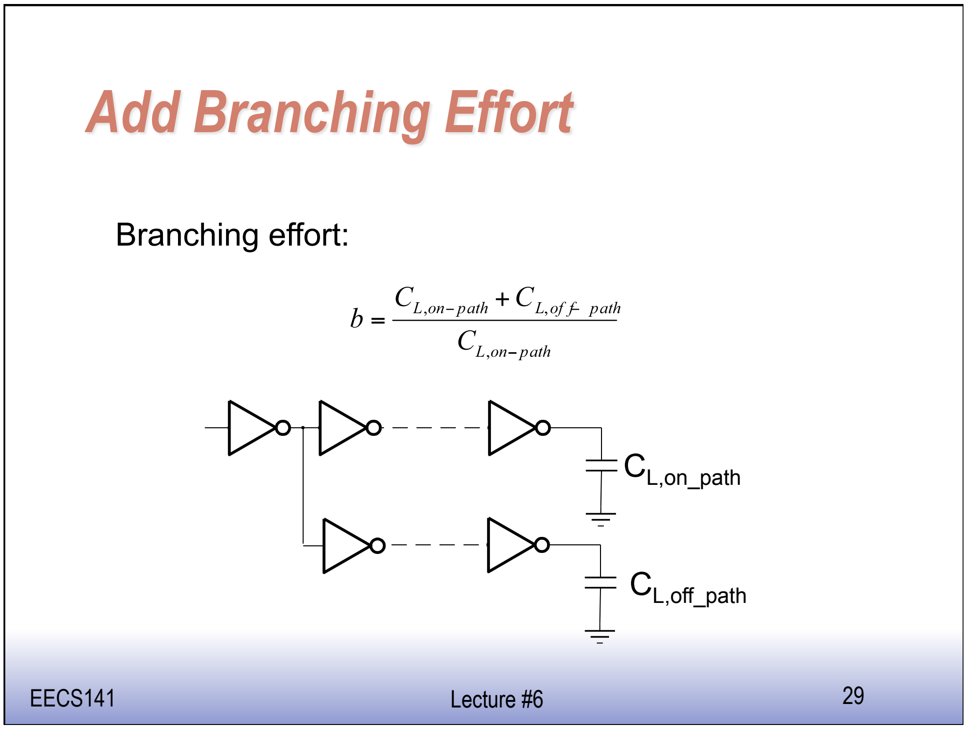
	Number of Inputs			
Gate Type	**1**	**2**	**3**	**n**
Inverter	1			
NAND		4/3	5/3	(n + 2)/3
NOR		5/3	7/3	(2n + 1)/3
Multiplexer		2	2	2
XOR		4	12	

From Sutherland, Sproull



Gate Sizing Convention

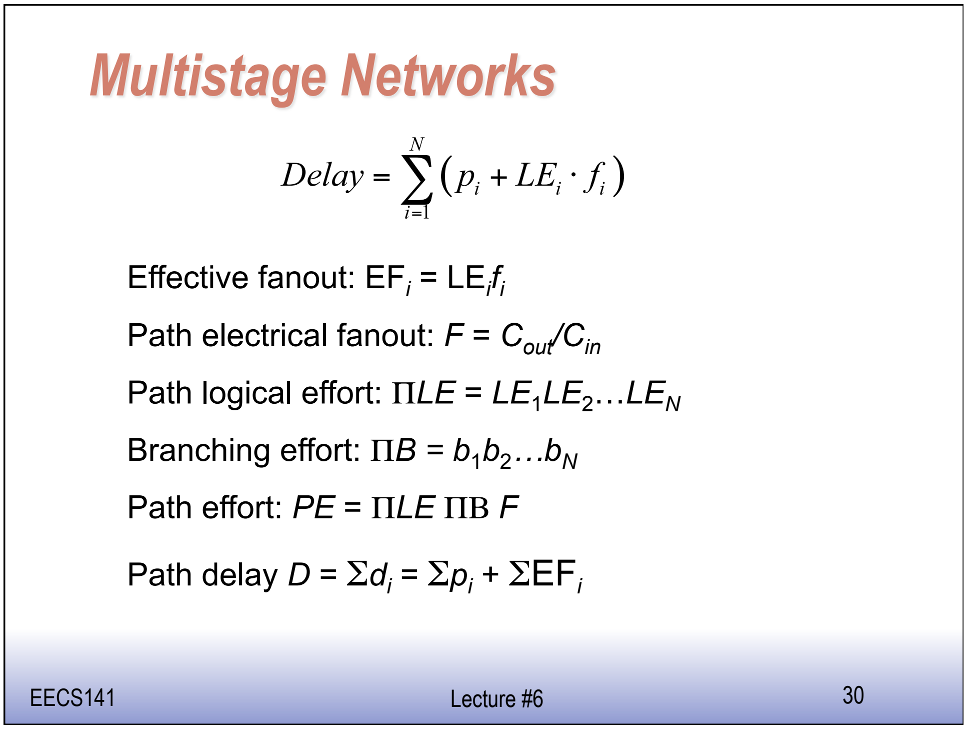
- ❑ Need to set a convention:
  - What does a gate of size ‘2’ mean?
- ❑ For an inverter it is clear:
  - $C_{inv} = 2$, $R_{inv} = ½$
- ❑ For a gate, two possibilities:
  - $C_{gate} = 2C_{inv}$
  - $R_{gate} = R_{inv}/2$
- ❑ In my notes, size ≡ C_{gate}/C_{inv}
  - Size 2 gate has twice the input capacitance of a unit inverter



Add Branching Effort

Branching effort:

$$b = \frac{C_{L,on-path} + C_{L,off-path}}{C_{L,on-path}}$$



Multistage Networks

$$Delay = \sum_{i=1}^{N} \left(p_i + LE_i \cdot f_i \right)$$

Effective fanout: $EF_i = LE_i f_i$

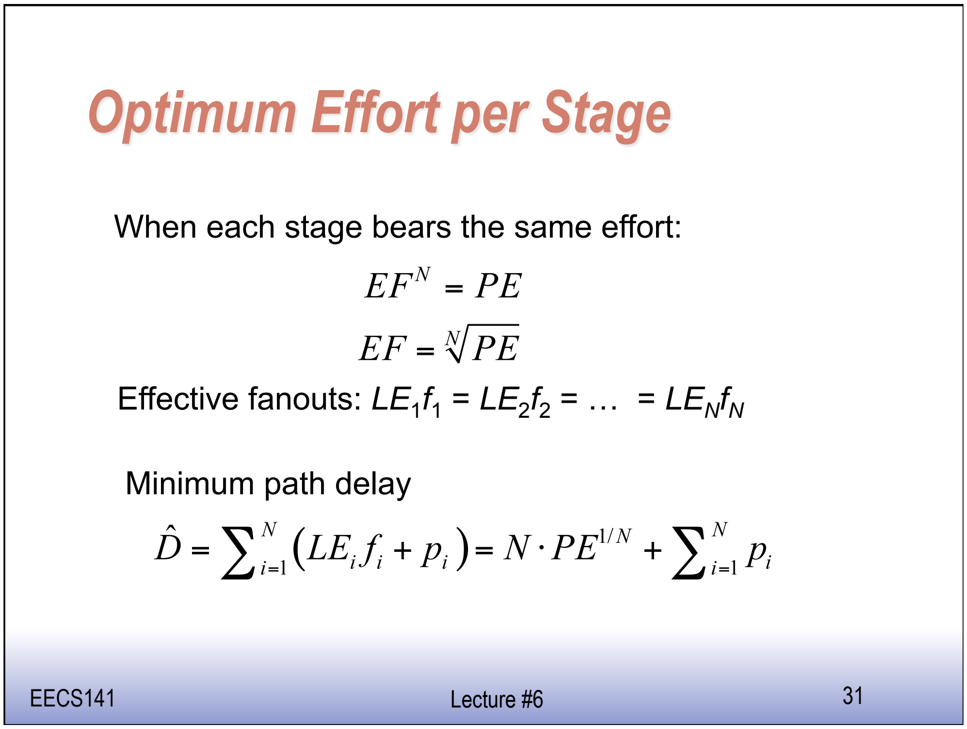
Path electrical fanout: $F = C_{out}/C_{in}$

Path logical effort: $\Pi LE = LE_1 LE_2 \ldots LE_N$

Branching effort: $\Pi B = b_1 b_2 \ldots b_N$

Path effort: $PE = \Pi LE \; \Pi B \; F$

Path delay $D = \Sigma d_i = \Sigma p_i + \Sigma EF_i$



Optimum Effort per Stage

When each stage bears the same effort:

$$EF^N = PE$$

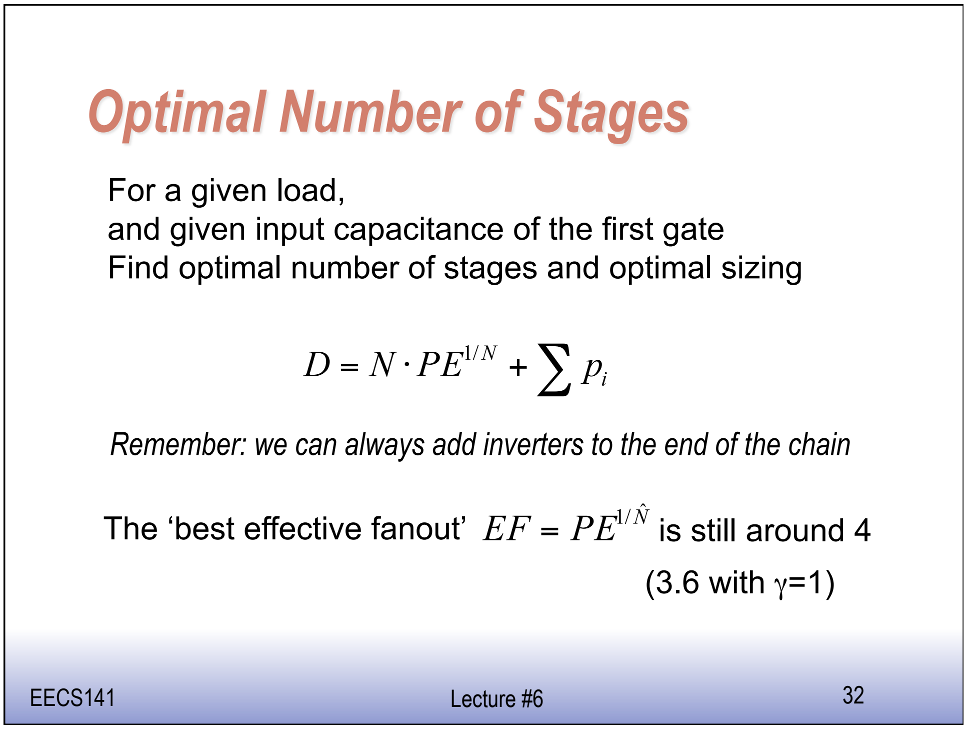
$$EF = \sqrt[N]{PE}$$

Effective fanouts: $LE_1f_1 = LE_2f_2 = \ldots = LE_Nf_N$

Minimum path delay

$$\hat{D} = \sum_{i=1}^{N}\left(LE_i f_i + p_i\right) = N \cdot PE^{1/N} + \sum_{i=1}^{N} p_i$$

EECS141 Lecture #6 31



Optimal Number of Stages

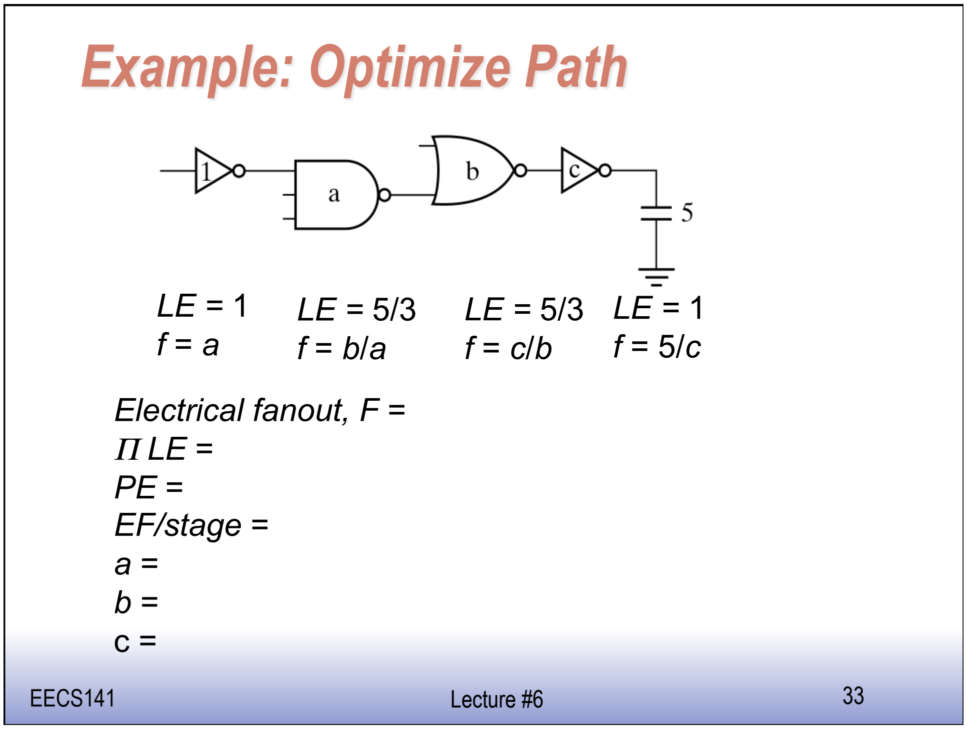
For a given load,
and given input capacitance of the first gate
Find optimal number of stages and optimal sizing

$$D = N \cdot PE^{1/N} + \sum p_i$$

Remember: we can always add inverters to the end of the chain

The 'best effective fanout' $EF = PE^{1/\hat{N}}$ is still around 4
(3.6 with γ=1)

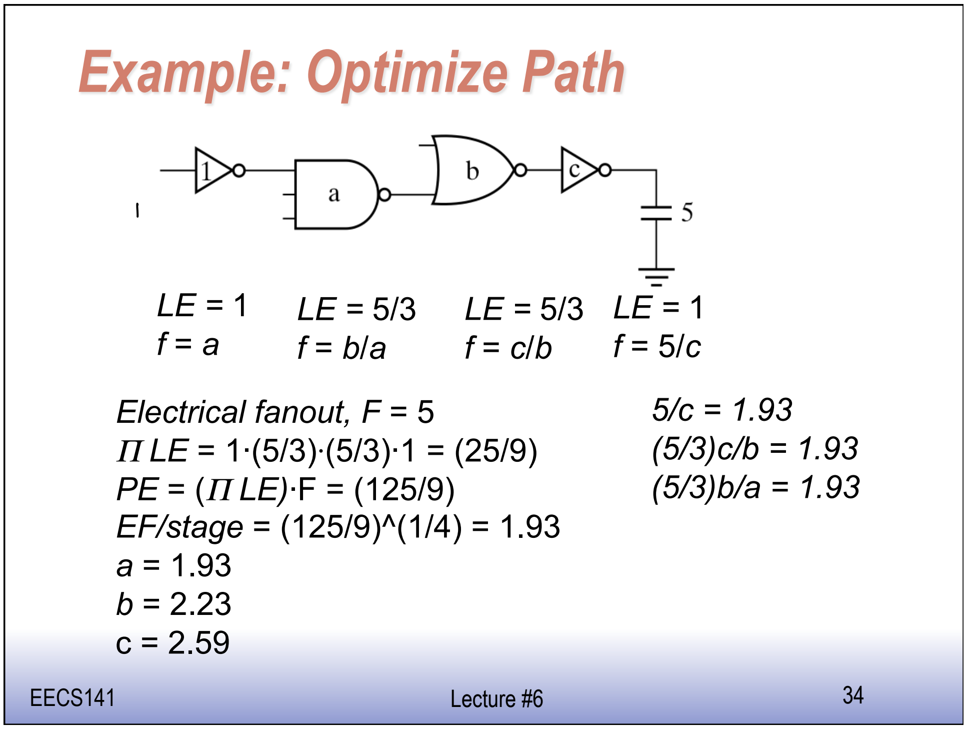
EECS141 Lecture #6 32



Example: Optimize Path

LE = 1	*LE* = 5/3	*LE* = 5/3	*LE* = 1
f = *a*	*f* = *b*/*a*	*f* = *c*/*b*	*f* = 5/*c*

Electrical fanout, F =
Π *LE* =
PE =
EF/stage =
a =
b =
c =

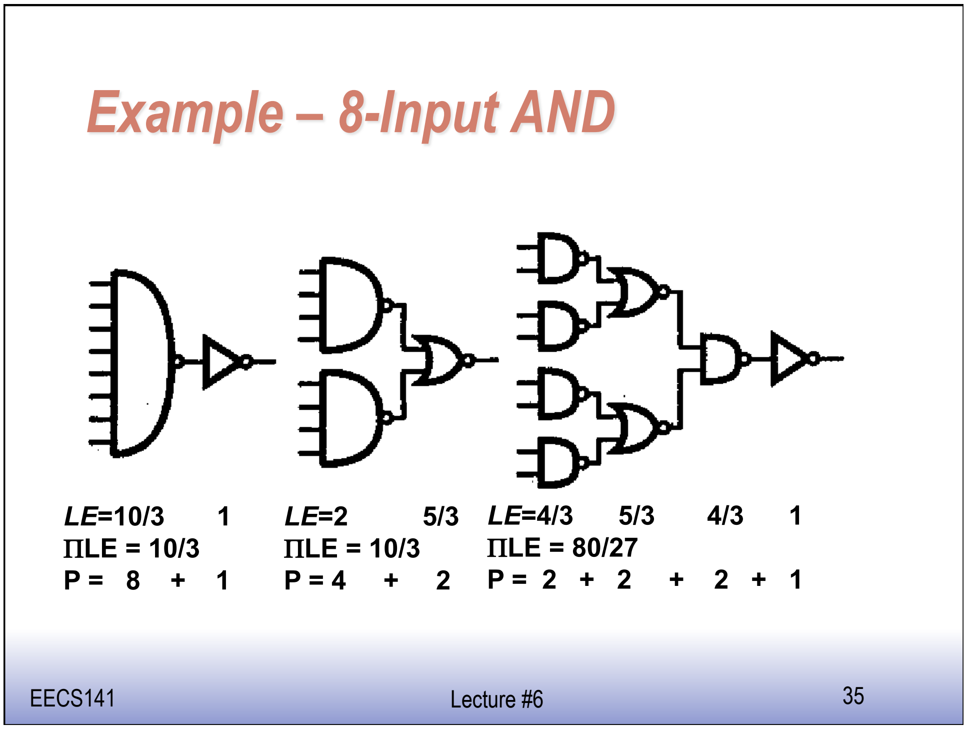


Example: Optimize Path

LE = 1	*LE* = 5/3	*LE* = 5/3	*LE* = 1
f = *a*	*f* = *b*/*a*	*f* = *c*/*b*	*f* = 5/*c*

Electrical fanout, F = 5
Π *LE* = 1·(5/3)·(5/3)·1 = (25/9)
PE = (Π *LE*)·F = (125/9)
EF/stage = (125/9)^(1/4) = 1.93
a = 1.93
b = 2.23
c = 2.59

5/c = 1.93
(5/3)c/b = 1.93
(5/3)b/a = 1.93

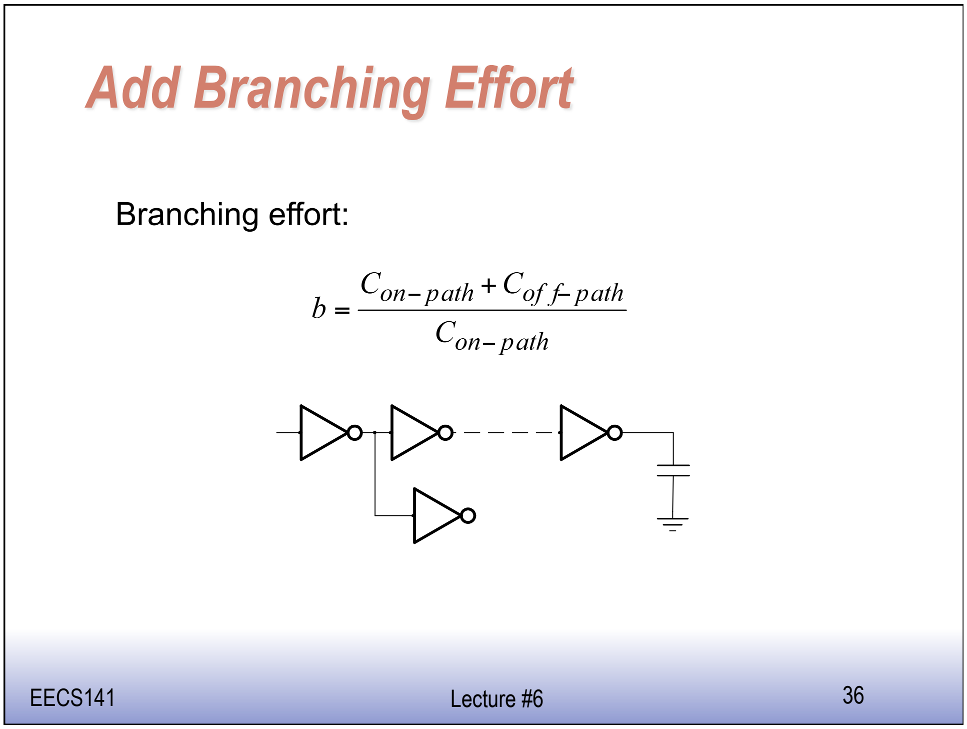


Example – 8-Input AND

LE=10/3 1 *LE*=2 5/3 *LE*=4/3 5/3 4/3 1

ΠLE = 10/3 ΠLE = 10/3 ΠLE = 80/27

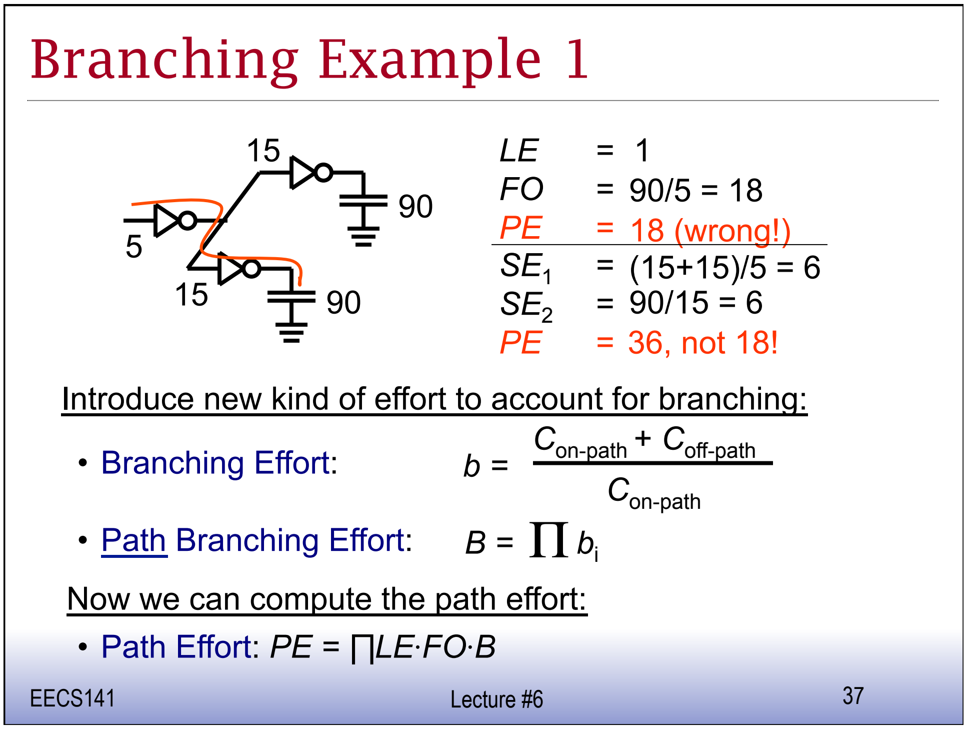
P = 8 + 1 P = 4 + 2 P = 2 + 2 + 2 + 1



Add Branching Effort

Branching effort:

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



Branching Example 1

$LE = 1$

$FO = 90/5 = 18$

$PE = 18$ (wrong!)

$SE_1 = (15+15)/5 = 6$

$SE_2 = 90/15 = 6$

$PE = 36$, not 18!

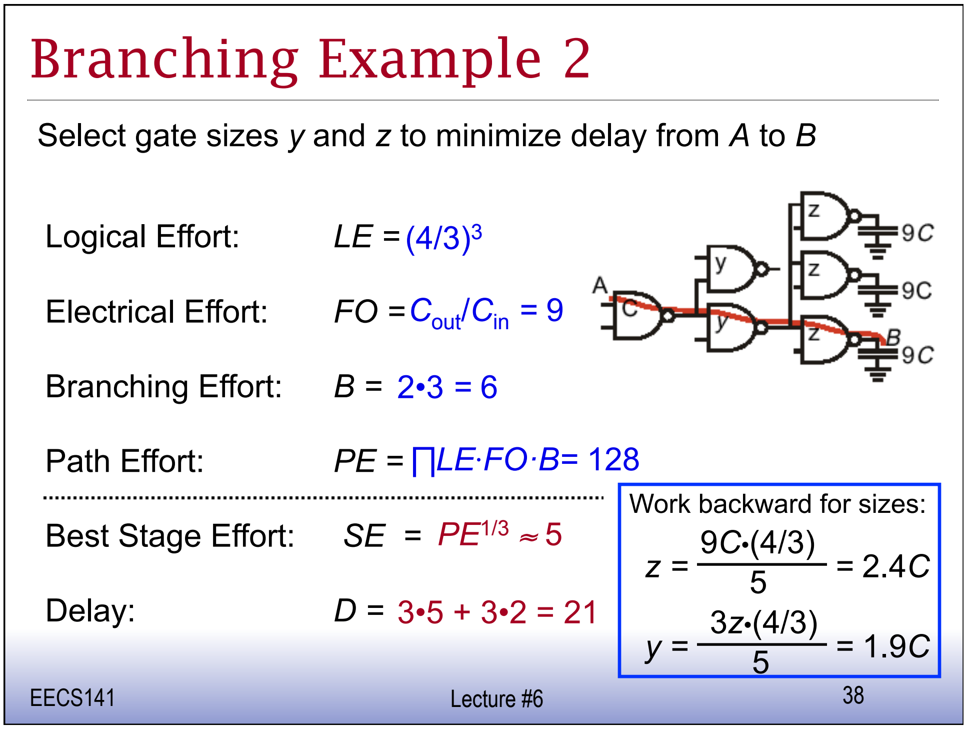
Introduce new kind of effort to account for branching:

- Branching Effort: $b = \dfrac{C_{\text{on-path}} + C_{\text{off-path}}}{C_{\text{on-path}}}$
- Path Branching Effort: $B = \prod b_i$

Now we can compute the path effort:

- Path Effort: $PE = \prod LE \cdot FO \cdot B$

EECS141 Lecture #6 37



Branching Example 2

Select gate sizes *y* and *z* to minimize delay from *A* to *B*

Logical Effort: $LE = (4/3)^3$

Electrical Effort: $FO = C_{out}/C_{in} = 9$

Branching Effort: $B = 2 \cdot 3 = 6$

Path Effort: $PE = \prod LE \cdot FO \cdot B = 128$

Best Stage Effort: $SE = PE^{1/3} \approx 5$

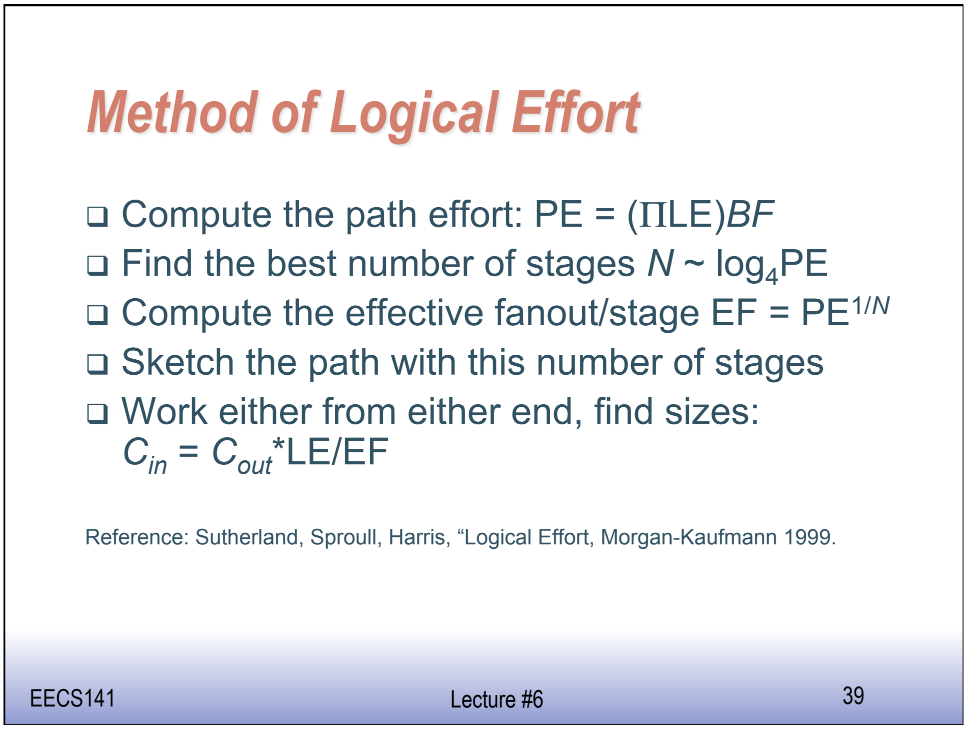
Delay: $D = 3 \cdot 5 + 3 \cdot 2 = 21$

Work backward for sizes:

$$z = \frac{9C \cdot (4/3)}{5} = 2.4C$$

$$y = \frac{3z \cdot (4/3)}{5} = 1.9C$$

EECS141 Lecture #6 38



Method of Logical Effort

- ❑ Compute the path effort: PE = (ΠLE)BF
- ❑ Find the best number of stages $N \sim \log_4$PE
- ❑ Compute the effective fanout/stage EF = PE$^{1/N}$
- ❑ Sketch the path with this number of stages
- ❑ Work either from either end, find sizes:
  $C_{in} = C_{out}$*LE/EF

Reference: Sutherland, Sproull, Harris, “Logical Effort, Morgan-Kaufmann 1999.