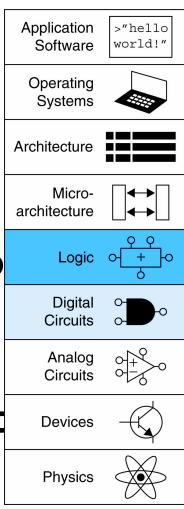
# Chapter 2

Digital Design and CArchitecture, 2nd Ed. L. H Digital Design and Computer Architecture, 2nd Editiononey Harris and Sarah L. Harris



# Chapter 2 :: Topics

- Introduction
- Boolean Equations
- Boolean Algebra
- From Logic to Gates
- Multilevel Combinational Lo
- X's and Z's, Oh My
- Karnaugh Maps
- Combinational Building Bloc
- Timing

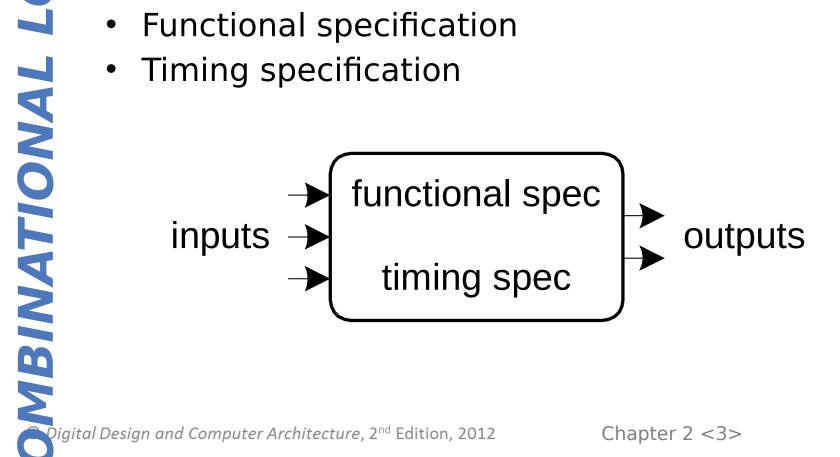




### Introduction

A logic circuit is composed of:

- Inputs
- **Outputs**
- Functional specification





### Circuits

n1 E1 **E**3 E2

- Circuit elements



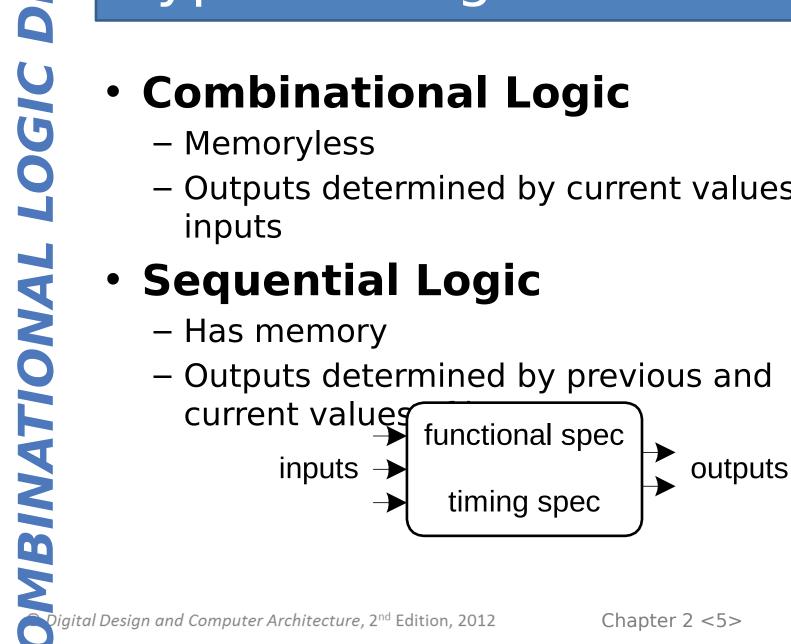
# Types of Logic Circuits

#### Combinational Logic

- Outputs determined by current values of

#### Sequential Logic

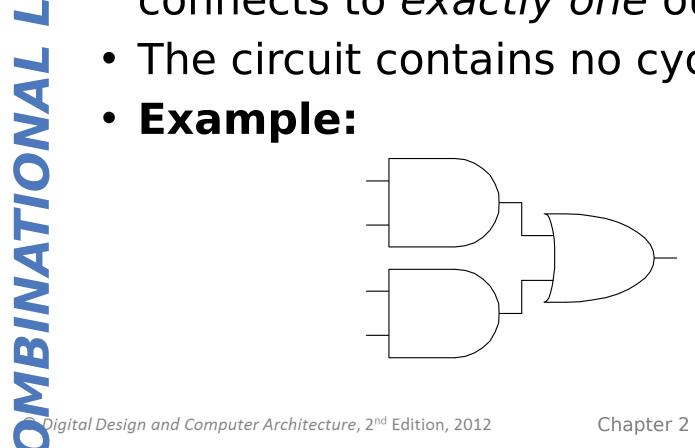
- Outputs determined by previous and





### Rules of Combinational

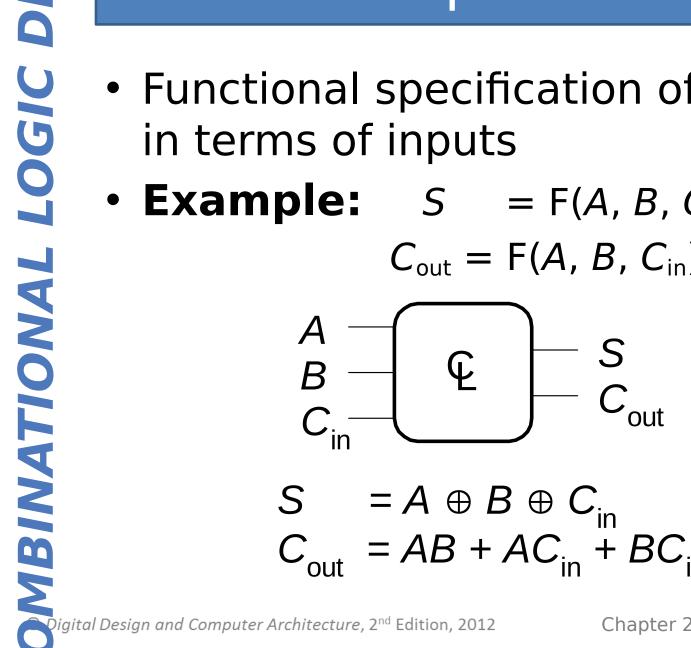
- Every element is combinational
- Every node is either an input or connects to exactly one output
- The circuit contains no cyclic paths





# Boolean Equations

- Functional specification of outputs in terms of inputs
- Example:  $S = F(A, B, C_{in})$  $C_{\text{out}} = F(A, B, C_{\text{in}})$



$$S = A \oplus B \oplus C_{in}$$
  
 $C_{out} = AB + AC_{in} + BC_{in}$ 



#### Some Definitions

- Complement: variable

  A, B, C
   Literal: variable or its

  A, A, B, B, C, C
   Implicant: product of I

  ABC, AC, BC
   Minterm: product that variables

  ABC, ABC, ABC
   Maxterm: sum that inc variables

  (A+B+C), (A+B+C), Complement: variable with a bar over it
  - Literal: variable or its complement
  - Implicant: product of literals
  - Minterm: product that includes all input
  - Maxterm: sum that includes all input

$$(A+B+C), (A+B+C), (A+B+C)$$



# Sum-of-Products (SOP)

- All equations can be written in SOP form
- Each row has a **minterm**
- A minterm is a product (AND) of literals
- Each minterm is TRUE for that row (and only that row)
- Form function by ORing minterms where the output is TRUE
- Thus, a sum (OR) of products (AND terms)

			-		` '	
	<ul> <li>Each mint</li> </ul>	erm	is TI	RUE	for that ro	ow (and o
7	• Form func	tion	by C	ORin	g minterm	s where t
MBINATIONAL	• Thus, a su	m (C	OR) o	of pro	oducts (Al	ND terms
0						minterm
	_	A	В	Y	minterm	name
		0	0	0	$\overline{A} \; \overline{B}$	$m_0$
<b>d</b>		0	1	1	$\overline{A} \; B$	$m_1$
		1	0	0	$A\overline{B}$	$m_2$
		1	1	1	АВ	$m_3^-$
2	]	$Y = \mathbf{I}$	F(A,	<b>B</b> ) =		
	al Design and Computer A	Architec	ture, 2 <sup>nd</sup>	d Edition	n, 2012	Chapter 2

$$Y = F(A, B) =$$



# Sum-of-Products (SOP) Sum-of-Products Form

- All equations can be written in SOP form
- Each row has a **minterm**
- A minterm is a product (AND) of literals
- Each minterm is TRUE for that row (and only that row)
- Form function by ORing minterms where the output is TRUE
- Thus, a sum (OR) of products (AND terms)

			-		` '	
7	<ul> <li>Each min</li> </ul>	term	is TI	RUE	for that ro	ow (and o
7	• Form fun	ction	by C	ORin	g minterm	s where t
MBINATIONAL	• Thus, a si	um (C	OR) o	of pro	oducts (Al	ND terms
						minterm
		_ <b>A</b>	В	Y	minterm	name
		0	0	0	$\overline{A} \; \overline{B}$	$m_0$
A		0	1	1	A B	$m_1$
		1	0	0	$A\overline{B}$	$m_2^-$
		1	1	1	АВ	$m_3$
		Y = I	$\exists (A,$	<b>B</b> ) =		
	al Design and Computer	Architec	ture, 2 <sup>nd</sup>	d Edition	1, 2012	Chapter 2

$$Y = F(A, B) =$$



# Sum-of-Products (SOP) Sum-of-Products Form

- All equations can be written in SOP form
- Each row has a **minterm**
- A minterm is a product (AND) of literals
- Each minterm is TRUE for that row (and only that row)
- Form function by ORing minterms where the output is TRUE
- Thus, a sum (OR) of products (AND terms)

			-		` '	
	<ul> <li>Each min</li> </ul>	ıterm	is TI	RUE	for that ro	ow (and o
7	• Form fun	ction	by C	ORin	g minterm	s where t
MBINATIONAL	• Thus, a s	um (C	OR) o	of pro	oducts (Al	ND terms
						minterm
		_A	В	Y	minterm	name
		0	0	0	$\overline{A} \ \overline{B}$	$m_0$
A		0	1	1	A B	$m_1$
		1	0	0	ΑB	$m_2$
		1	1	1	АВ	$m_3$
M		Y = I	F(A,	<b>B</b> ) =	$\overline{\mathbf{A}}\mathbf{B} + \mathbf{A}\mathbf{B}$	$= \Sigma(1,3)$
	al Design and Compute	r Architec	ture, 2 <sup>nd</sup>	d Edition	1, 2012	Chapter 2

$$Y = F(A, B) = AB + AB = \Sigma(1, 3)$$



## Product-of-Sums (POS)

- All Boolean equations can be written in POS form
- Each row has a **maxterm**
- A maxterm is a sum (OR) of literals
- Each maxterm is FALSE for that row (and only that row)
- Form function by ANDing the maxterms for which the
- Thus, a product (AND) of sums (OR terms)

7	•	Each max	term i	is FA	LSE for th	at row (and o
AL	•	Form function output is F			NDing the	maxterms fo
NC	•	Thus, a pr	oduct	(AN	(D) of sum	s (OR terms)
					_	maxterm
			В	Y	maxterm	name
		<b>(</b>	0	0	A + B	$M_{\circ}$
		0	1	1	$A + \overline{B}$	$M_1$
IN			1 0	1	A + B A + B	$M_1$ $M_2$
BIN		0	1 0 1			$M_1$ $M_2$ $M_3$
MBINATIONAL	ral Dasi	0 1 1	1 (A, B	0 1 3) = (	$\frac{\overline{A} + B}{\overline{A} + B}$ $A + B$ )(A +	$ M_{1} $ $ M_{2} $ $ M_{3} $ $ B) = \Pi(0, 2) $ Chapter 2

$$Y = F(A, B) = (A + B)(A + B) = \Pi(0, 2)$$



# **Boolean Equations**

- You are going to the cafeteria for
  - You won't eat lunch (E)
  - If it's not open (O) or
  - If they only serve corndogs (C)
- You are going to lunch

   You won't eat lur

   If it's not open (C)

   If they only serve

   Write a truth tab if you will eat lur Write a truth table for determining if you will eat lunch (E)C



# Boolean Equations

- You are going to the cafeteria for
  - You won't eat lunch (E)
  - If it's not open (O) or
  - If they only serve corndogs (C)
- You are going to lunch

   You won't eat lunt

   If it's not open (County only serve)

   Write a truth tab if you will eat lunt Write a truth table for determining if you will eat lunch (E)C



### SOP & POS Form

produ	ıcts	Ε	minterm
0	0		<u> 0</u> <u>C</u>
0	1		<u> </u>
1	0		0 <u>C</u>
1	1		0 C



### SOP & POS Form

7				
)	• SOP -	sun	n-of-	
15	produ	cţs	E	minterm
0	0	0	0	<u>0</u> <u>C</u>
7	0	1	0	<u>o</u> c
7	1	0	1	0 <u>C</u>
4	1	1	0	0 C
0				_
MBINATIONAL LOGIC	• POS –	pro	duct E	r-of- maxterm
D	0	0	0	0 + C
	0	1	0	$0 + \overline{C}$
	1	0	1	<u>0</u> + C
	(1	1	0	$\overline{O} + \overline{C}$
Digital	Design and Comp	uter Arcı	hitecture	, 2 <sup>nd</sup> Edition, 2012

$$E = OC$$
$$= \Sigma(2)$$

$$E = (O + C)(O + \overline{C})(\overline{O} + \overline{C})$$
  
=  $\Pi(0, 1, 3)$ 



# Boolean Algebra

- Axioms and theorems to simplify Boolean equations
- Like regular algebra, but simpler: variables have o
  0)
  • Duality in axiom
  – ANDs and ORs, 0's variables have only two values (1 or
  - Duality in axioms and theorems:
    - ANDs and ORs, 0's and 1's interchanged



### **Boolean Axioms**

	Axiom		Dual	Name
A1	$B = 0 \text{ if } B \neq 1$	A1′	$B = 1 \text{ if } B \neq 0$	Binary field
A2	$\overline{0} = 1$	A2′	$\overline{1} = 0$	NOT
A3	$0 \bullet 0 = 0$	A3′	1 + 1 = 1	AND/OR
A4	1 • 1 = 1	A4′	0 + 0 = 0	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	A5'	1 + 0 = 0 + 1 = 1	AND/OR

Во	olean	Axi	oms	
	Axiom		Dual	Name
A1	$B = 0 \text{ if } B \neq 1$	A1′	$B = 1 \text{ if } B \neq 0$	Binary field
A2	$\overline{0} = 1$	A2′	$\overline{1} = 0$	NOT
A3	$0 \bullet 0 = 0$	A3′	1 + 1 = 1	AND/OR
A4	1 • 1 = 1	A4′	0 + 0 = 0	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	A5'	1 + 0 = 0 + 1 = 1	AND/OR
	Theorem		Dual	Name
T1	B • 1 = B	T1'	B+0=B	Identity
T1 T2		T1'	B + 0 = B $B + 1 = 1$	Identity Null Element
	B • 1 = B			· · · · · · · · · · · · · · · · · · ·
T2	$B \bullet 1 = B$ $B \bullet 0 = 0$	T2'	B + 1 = 1	Null Element



# T1: Identity Theorem



Chapter 2 < 19 >

# T1: Identity Theorem

$$\begin{bmatrix} B \\ 0 \end{bmatrix}$$
  $=$   $B$ 



### T2: Null Element Theorem



### T2: Null Element Theorem

$$\begin{bmatrix} B \\ 0 \end{bmatrix} = 0$$

$$\begin{bmatrix} B \\ 1 \end{bmatrix}$$
  $=$  1



# T3: Idempotency Theorem



# T3: Idempotency Theorem

$$B - = B - = B$$

$$\begin{bmatrix} B \\ B \end{bmatrix} = B$$



# T4: Identity Theorem



# T4: Identity Theorem

$$B \longrightarrow B \longrightarrow$$



# T5: Complement Theorem

• 
$$B \cdot B = 0$$

• 
$$B + \overline{B} = 1$$



# T5: Complement Theorem

$$\frac{B}{B}$$
  $\bigcirc$   $\bigcirc$   $\bigcirc$  0

$$\frac{B}{B} \longrightarrow = 1$$



### **Boolean Theorems**

	Theorem		Dual	Name
T1	$B \bullet 1 = B$	T1'	B+0=B	Identity
T2	$B \bullet 0 = 0$	T2'	B + 1 = 1	Null Element
Т3	$B \bullet B = B$	T3′	B + B = B	Idempotency
T4		$\bar{\bar{B}} = B$		Involution
T5	$B \bullet \overline{B} = 0$	T5'	$B + \overline{B} = 1$	Complements
al Design and	Computer Architecture, 2	2 <sup>nd</sup> Edition, 2012	Chapter 2	<29> Ë



### Boolean Theorems of

			Dual	Name
Т6	$R \bullet C = C \bullet R$	T6'	B + C = C + B	Commutati
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	T7'	(B+C)+D=B+(C+D)	Associativit
T8	$(B \bullet C) + (B \bullet D) = B \bullet (C + D)$	T8'	$(B+C) \bullet (B+D) = B + (C \bullet D)$	Distributivi
T9	$B \bullet (B + C) = B$	T9'	$B + (B \cdot C) = B$	Covering
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	T10'	$(B + C) \bullet (B + \overline{C}) = B$	Combining
T11	$(B \bullet C) + (\overline{B} \bullet D) + (C \bullet D)$ = $B \bullet C + \overline{B} \bullet D$	T11'	$(B + C) \bullet (\overline{B} + D) \bullet (C + D)$ = $(B + C) \bullet (\overline{B} + D)$	Consensus
T12	$ \overline{B_0 \bullet B_1 \bullet B_2 \dots} \\ = (\overline{B_0} + \overline{B_1} + \overline{B_2} \dots) $	T12′	$ \overline{B_0} + \overline{B_1} + \overline{B_2} $ $ = (\overline{B_0} \bullet \overline{B_1} \bullet \overline{B_2}) $	De Morgan Theorem
	B • $C = C • B$ (B • C) • D = B • (C • D) (B • C) + (B • D) = B • (C + D) B • (B + C) = B (B • C) + (B • C) = B (B • C) + (B • D) + (C • D) = B • C + B • D $B_0 • B_1 • B_2$ $= (B_0 + B_1 + B_2)$ e: T8' differs from traditional computer Architecture, 2nd Editional Computer Arc	ional a	lgebra: OR (+) distributes c	over AND
Note				



$$Y = AB + AB$$



Simplifying Boo

Example 1:

$$Y = AB + AB$$
 $= B(\overline{A} + A)$  T8

 $= B(1)$  T5'

 $= B$  T1



$$Y = A(AB + ABC)$$



Example 2:  

$$Y = A(AB + ABC)$$

$$= A(AB(1 + C)) \qquad T8$$

$$= A(AB(1)) \qquad T2'$$

$$= A(AB) \qquad T1$$

$$= (AA)B \qquad T7$$

$$= AB \qquad T3$$
Sigital Design and Computer Architecture, 2<sup>nd</sup> Edition, 2012 Chapter



# DeMorgan's Theorem

• 
$$Y = \overline{A}B = \overline{A} + B_{B}$$
•  $Y = \overline{A} + \overline{B} = A$ 

B

Pigital Design and Computer Architecture, 2<sup>nd</sup> Edition, 2012

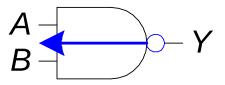
•  $Y = \overline{A}B = \overline{A}B$ 

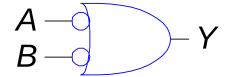
Characteristics of the second sec

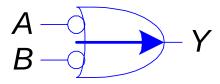
• 
$$Y = \overline{A} + \overline{B} = A$$
 $A = A$ 
 $A =$ 



# **Bubble Pushing**



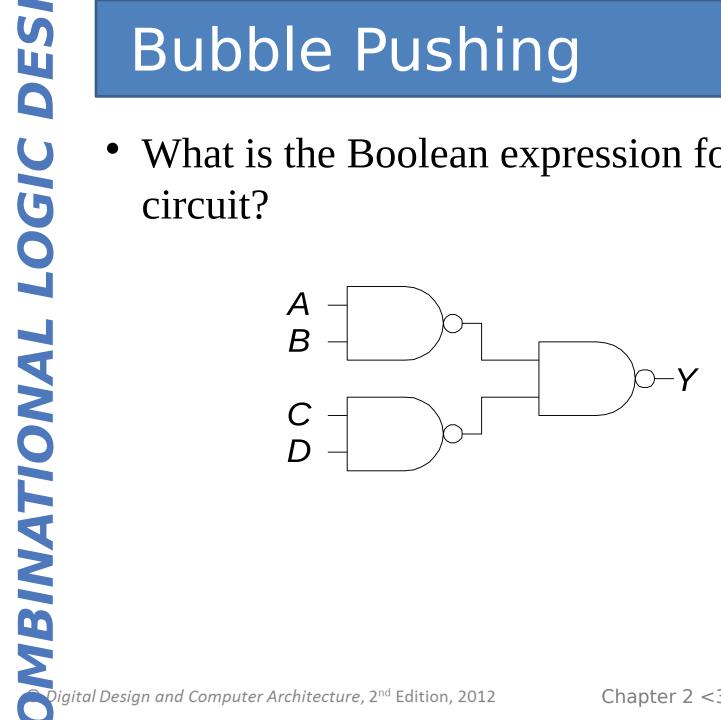






#### **Bubble Pushing**

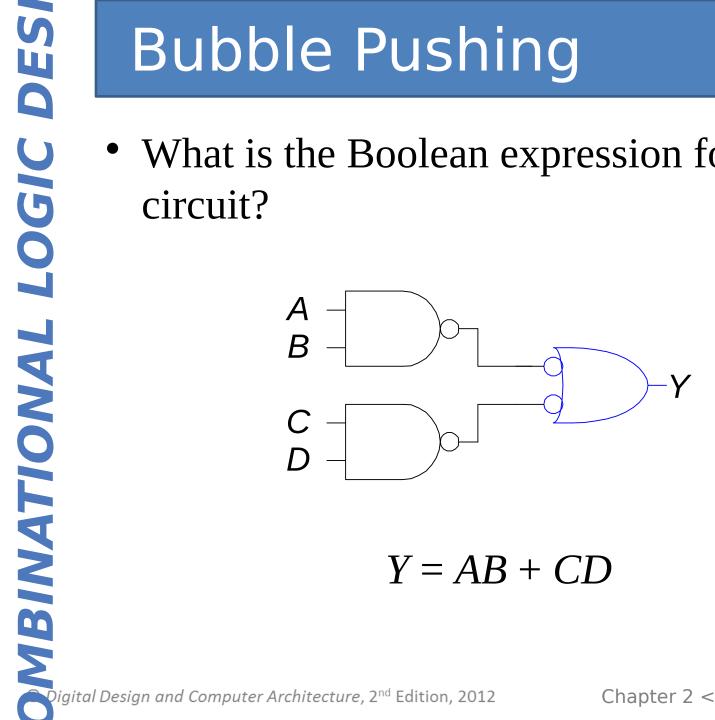
What is the Boolean expression for this





#### **Bubble Pushing**

What is the Boolean expression for this

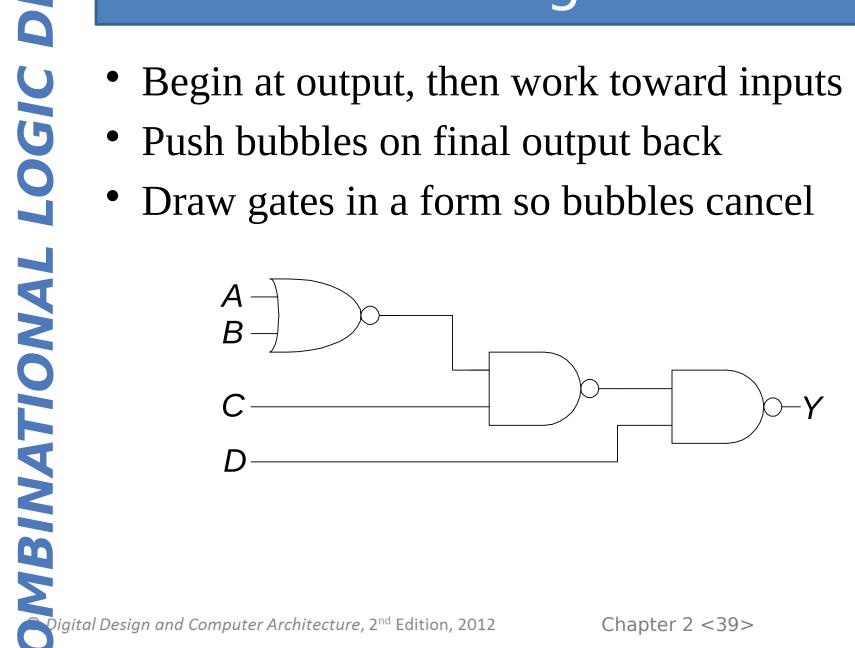


$$Y = AB + CD$$



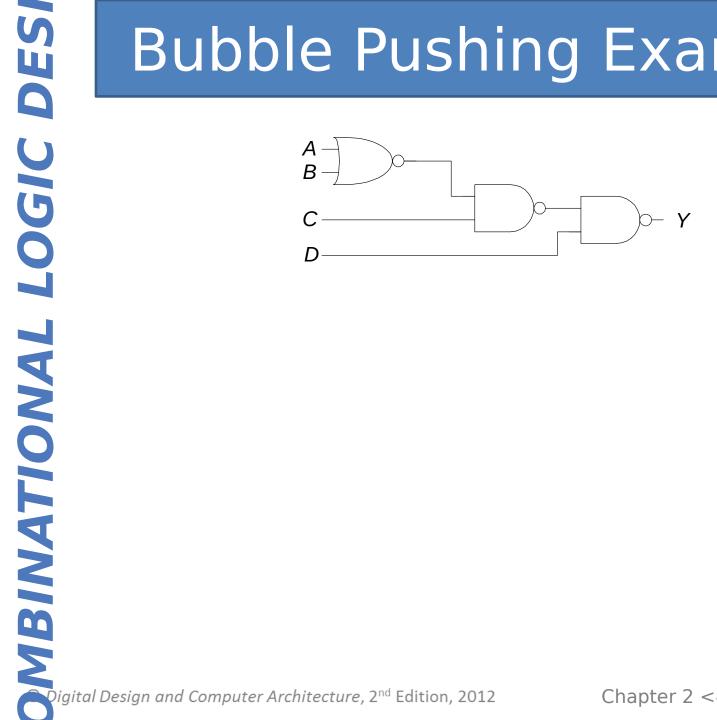
#### **Bubble Pushing Rules**

- Begin at output, then work toward inputs
- Push bubbles on final output back
- Draw gates in a form so bubbles cancel

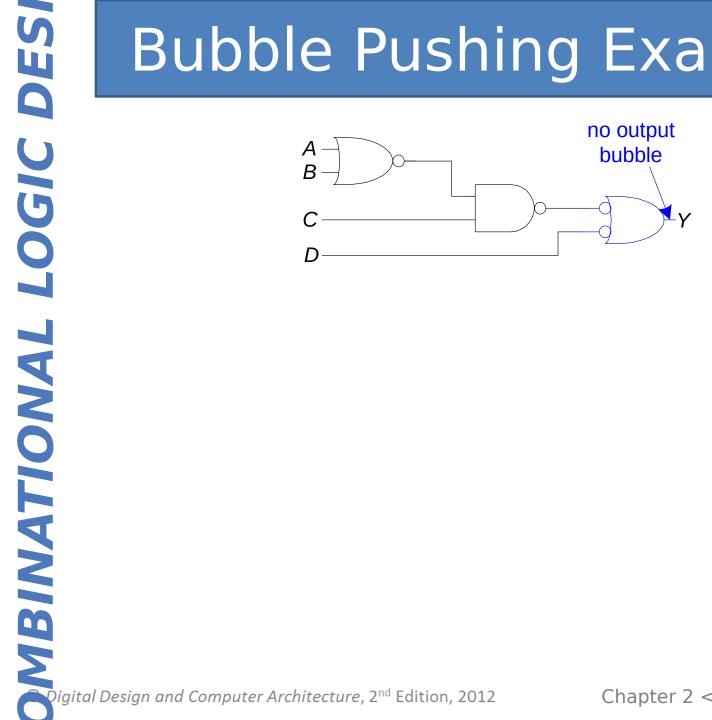




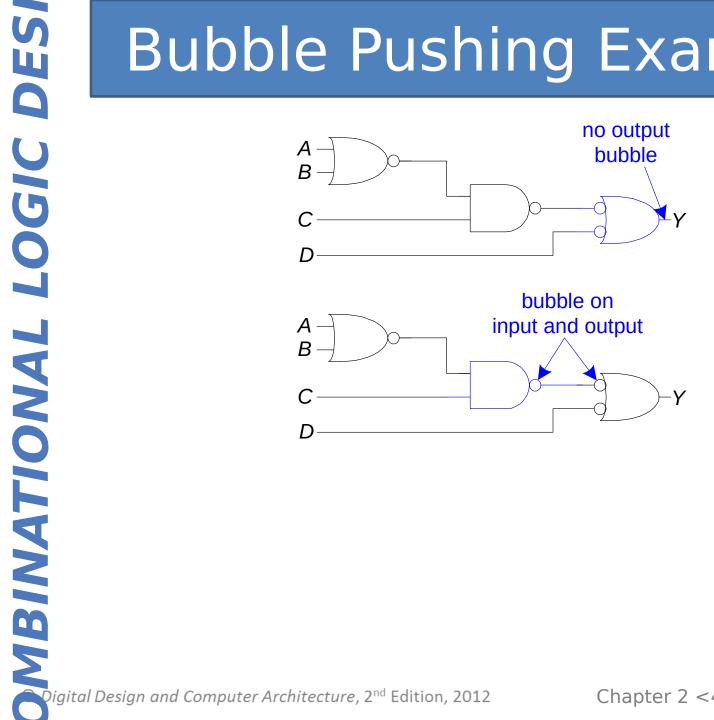
Chapter 2 <39>



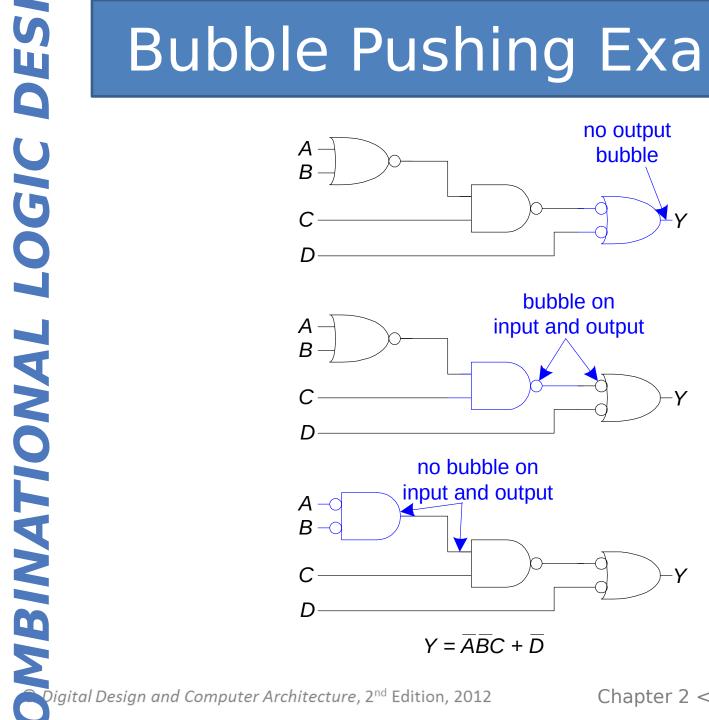








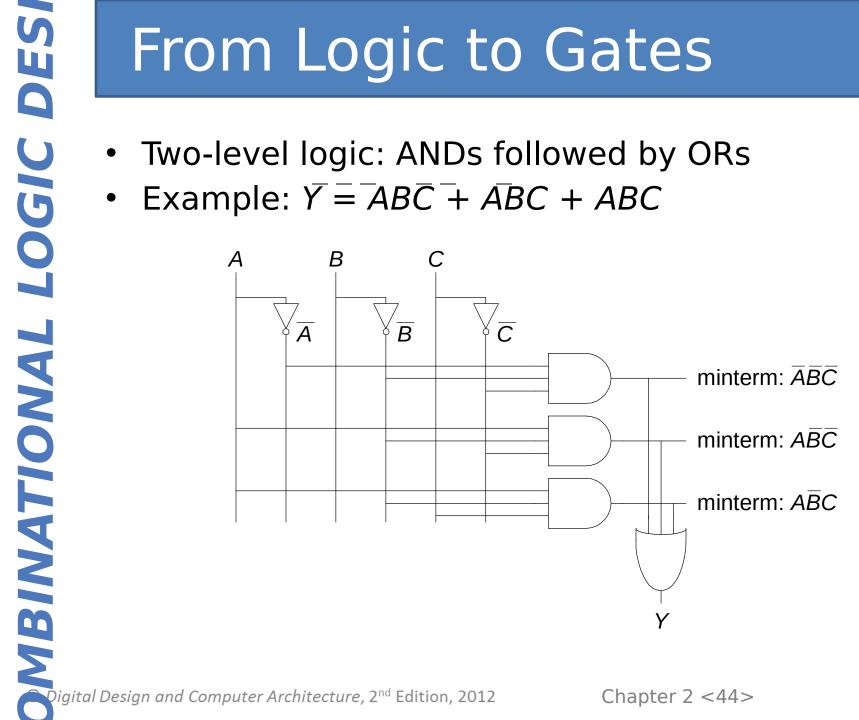






#### From Logic to Gates

- Two-level logic: ANDs followed by ORs
- Example:  $Y = \overline{ABC} + \overline{ABC} + ABC$





#### Circuit Schematics Rules

- Inputs on the left (or top)
- Outputs on right (or bottom)
- Gates flow from left to right
- Straight wires are best



#### Circuit Schematic Rules

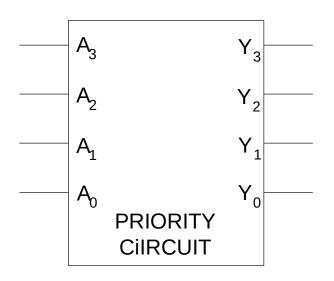
- Wires always connect at a T
- Wires always conjunction
   A dot where wire indicates a connection the wires
   Wires crossing wires connect make mo connect A dot where wires cross indicates a connection between
  - Wires connect ossing without a dot do without a dot do makemo connection not connect



#### Multiple-Output Circuits

#### Example: Priority Circuit

Output asserted corresponding to most significant TRUE input



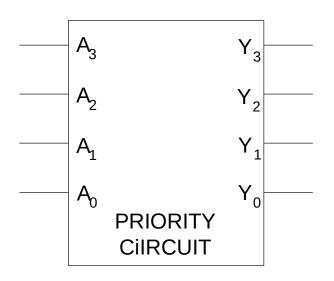
	416						
$A_3$	$A_2$	$A_{1}$	$A_{o}$	Y <sub>3</sub>	$Y_2$	$Y_1$	$Y_{o}$
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	1 0				
1	1	1	1				
			,	•			



#### Multiple-Output Circuits

#### Example: Priority Circuit

Output asserted corresponding to most significant TRUE input

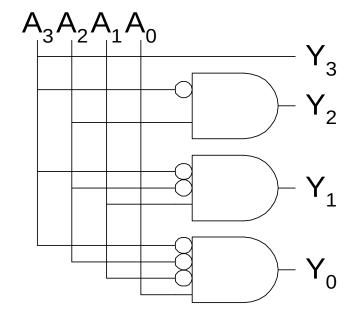


$A_3$	$A_2$	$A_{1}$	$A_{0}$	$Y_3$	$Y_2$	$Y_1$	$Y_{o}$
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0



#### Priority Circuit Hardware

MBINATIONAL LOGIC DE									
U					ı	1			
		$A_{2}$	$A_2$	$A_{\scriptscriptstyle 1}$	$A_{0}$	Y	Υ,	Υ,	Y <sub>0</sub>
5	_	0	0	0	0	0	0	0	0
Õ		0	0	0	1	0	0	0	1
		0 0	0 0	1 1	0 1	0 0	0 0	1	0 0
		0	1	0	0	0	1	0	0
_1		0	1	0	1	0	1	0	0
		0	1	1	0	0	1	0	0
		⊍ 1	U T	U T	Θ T	0 1	9 T	0	0 0
2		i	0	0	1	î	0	0	0
		1	0	1	0	1	0	0	0
		1	0	1	1	1	0	0	0
		⊥ 1	⊥ 1	0	0 1	1	0	0	0
		i	i	1	0	ī	0	0	0
		1	1	1	1	1	0	0	0
2									
	_						- nd - *-		
Digital	Design	and	Comp	uter Ai	rchited	cture, i	2 <sup>na</sup> Edi	tion, 2	2012





$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	SESI		)c	n	't	C	a	re	25	
O	BINATIONAL LOGIC DES	0 0 0 0 1 1 1 1 1	0 0 1 1 1 0 0 0 1 1 1	0 0 1 0 0 1 0 0 1 0 0 1	0 1	0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1	0 0 0 0 1 1 1 1 0 0 0 0 0	0 0 1 0 0 0 0 0 0 0 0	010000000000000000000000000000000000000	_

$A_3$	$A_2$	$A_1$	$A_o$	Y <sub>3</sub> 0 0 0 0	$Y_2$	$Y_{\underline{1}}$	$Y_0$
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	Χ	0	0	1	0
0	1	X	Χ	0	1	0	0
1	Χ	Χ	X	1	0	0	0



#### Contention: X

- Contention: circuit tries to drive output to 1
  - Actual value somewhere in between
  - Could be 0, 1, or in forbidden zone
  - Might change with voltage, temperature, time,
- Contention: circuit and 0

   Actual value somew

   Could be 0, 1, or in 1

   Might change with v noise

   Often causes exte

   Warnings:

   Contention usually in

   X is used for "don' look at the context to - Often causes =xte ≥ive power dissipation -Y = X
  - - Contention usually indicates a bug.
    - X is used for "don't care" and contention look at the context to tell them apart



# Floating: Z

- Floating, high im high Z
   Floating output I somewhere in be
   A voltmeter won't in is floating

   E A Y
   O O Z
   O 1 Z
   1 O O
   1 1 1
   O O
   I 1 1
   O O
   I 1 1
   O O
   I 1 1
   O O
   I 1 1
   O O
   I 1 1
   O O
   I 1 1
   O O
   I 1 1
   O O
   I 1 1
   O O
   I I I
   O O
   I I I
   O O
   I I I
   O O
   I I I
   O O
   I I I
   O O
   I I I
   O O
   I I I
   O O
   I I I
   O O
   I I I
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   O O
   Floating, high impedance, open,
  - Floating output might be 0, 1, or somewhere in between
    - A voltmeter won't indicate whether a node

state Buffer



#### Tristate Busses

• Floating nodes a tristate busses

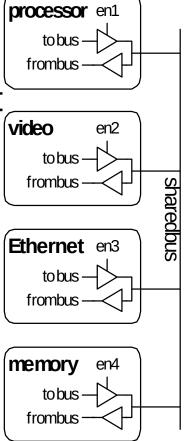
- Many different dri

- Exactly one is action

once Floating nodes are used in

Many different drivers

Exactly one is active at





## Karnaugh Maps (K-Maps)

- Boolean expressions can be minimized by combining terms
- K-maps minimize equations

Α	В	С	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Y AB							
c	00	01	11	10			
0	1	0	0	0			
1	1	0	0	0			

<i>Y</i> ∧ <i>A</i>	В			
C	00	01	11	10
0	ĀĒĈ	ĀBĒ	ABĈ	AĒĈ
1	ĀĒC	ĀBC	ABC	AĒC



## K-Map

- Circle 1's in adjacent squares
- Circle 1's in adjacen
   In Boolean expressic literals whose true an are **not** in the circle

  | A | B | C | Y | Y |
  | 0 | 0 | 0 | 1 |
  | 0 | 0 | 0 | 1 |
  | 0 | 0 | 0 |
  | 1 | 0 | 0 |
  | 1 | 0 | 0 |
  | 1 | 0 | 0 |
  | 1 | 1 | 0 |
  | 1 | 1 | 0 |
  | 1 | 1 | 0 |
  | 1 | 1 | 0 |
  | 1 | 1 | 1 | 0 |
  | 1 | 1 | 1 | 0 |
  | 2 | Y |

  | Y |
  | Y |
  | Y |
  | Y |
  | Y |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | D |
  | O | In Boolean expression, include only literals whose true and complement form

Α	В	С	Y
0	0	0	1
Θ	0	1	1
Θ	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

YA	В			
C	00	01	11	10
0	1	0	0	0
1	1	0	0	0

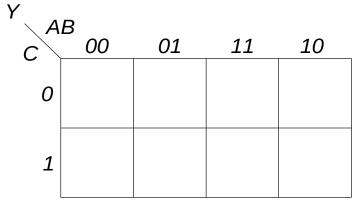
$$Y = AB$$



ESI	3-	In	рι	ıt K	-M	ар	
11C E			Y C	B 00	01	11	10
90			0	ĀĒC	ĀBĒ	ABŌ	AĒĈ
7 7			1	ĀĒC	ĀBC	ABC	ABC
NATIONAL LOGIC DES	<i>A</i> ⊙ ⊙	Truth	Tabl  C 0 1 0	e  Y 0 0 1	Y	AB 00	<b>K-M</b> a
	0 1 1 1	1 0 0 1	1 0 1 0	1 0 0 0		1	
Digital	<b>1</b> Design and (	Comput	er Archi	tecture, 2 <sup>n</sup>	d Edition,	2012	C

A	В	C	Y
0	0	0	0
0	0	1	0
Θ	1	0	1
Θ	1	1	1
1	0	0	0
1	0	1	0
1	1	0	Θ
1	1	1	1

#### K-Map

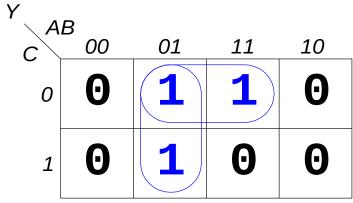




ESI	3-1	n	οl	ıt K	-M	ар	
1 )!!			Y	B 00	01	11	10
90			0	ĀĒC	ĀBĒ	ABŌ	AĒĈ
7 7			1	ĀĒC	ĀBC	ABC	ABC
NATIONAL LOGIC DES	_ A_	В	Tabl	e 	Y	\ AB	K-Ma
ATIC	0 0 0 0 1	0 0 1 1 0	0 1 0 1 0	0 0 1 1 0	C	o <b>O</b>	
BIN	1	0 1	1	0 0		1 0	1
Pigital	1 Design and Co	ompute	er Archi	tecture, 2 <sup>n</sup>	$Y = \overline{A}$	B + B	$\overline{\overline{C}}$

_ <b>A</b>	В	С	Y
0	0	0	0
Θ	Θ	1	0
Θ	1	0	1
Θ	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

#### K-Map



$$Y = \overline{A}B + B\overline{C}$$



#### K-Map Definitions

- Complement: variable with a bar over it
- Literal: variable or its complement
- Implicant: product of literals
- Complement: variable or A, B, C
   Literal: variable or A, A, B, B, C, C
   Implicant: product ABC, AC, BC
   Prime implicant: corresponding to the map Prime implicant: implicant corresponding to the largest circle in a K-



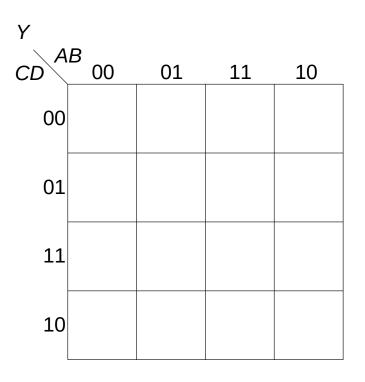
Chapter 2 <58>

#### K-Map Rules

- Every 1 must be circled at least once
- Each circle must span a power of 2 (i.e. 1, 2, 4) squares in each direction
- Each circle must be as large as possible
- A circle may wrap around the edges
- Every 1 must be ci
  Each circle must si
  1, 2, 4) squares in
  Each circle must be
  A circle may wrap at
  A "don't care" (X) it helps minimize the A "don't care" (X) is circled only if it helps minimize the equation



DESI	4-	·lr	ηp	u	t	K-I	VIa
		0 0 0 0 0 0 0 0 1 1 1 1 1	0 0 0 0 1 1 1 0 0 0 1 1 1 1	0 0 1 1 0 0 1 1 0 0 1 1	0 1 0 1 0 1 0 1 0 1 0 1	1 0 1 0 1 1 1 1 1 0 0 0 0	Y



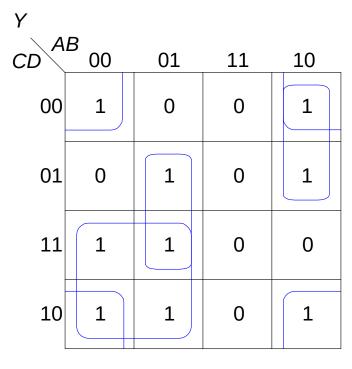


DESI	4-	·lr	ηp	u	t	K-I	VIa
		0 0 0 0 0 0 0 0 1 1 1 1 1	0 0 0 0 1 1 1 0 0 0 1 1 1 1	0 0 1 1 0 0 1 1 0 0 1 1	0 1 0 1 0 1 0 1 0 1 0 1	1 0 1 0 1 1 1 1 1 0 0 0 0	Y

Y				
CDA	B 00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	1



4-Input K-May  A B C D Y  0 0 0 0 0 1  0 0 0 1 0  0 0 1 0 1  0 1 0 0 0  0 1 0 1	DESI	4-	·lr	ηp	u	t	K-I	VIa
			0 0 0 0 0 0 0 0 1 1 1 1 1	0 0 0 0 1 1 1 0 0 0 1 1 1 1	0 0 1 0 0 1 0 0 1 0 0 1 1	0 1 0 1 0 1 0 1 0 1 0 1	1 0 1 1 0 1 1 1 1 0 0 0 0	C

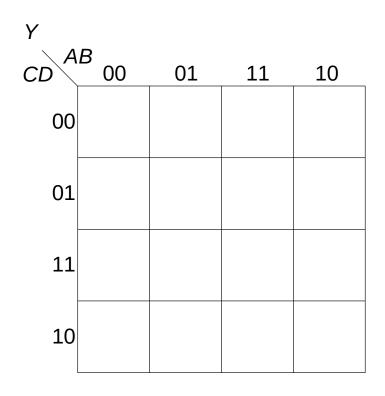


$$Y = \overline{AC} + \overline{ABD} + A\overline{BC} + \overline{BD}$$



#### K-Maps with Don't Cares

)ESI	K-	. <b> </b> V	la	p:	S '	with	
NATIONAL LOGIC DES		A 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1	B 00001111000011111	C 0 0 1 1 0 0 1 1 0 0 1 1	D 0 1 0 1 0 1 0 1 0 1 0 1	Y 1 0 1 1 0 X 1 1 1 X X X	
<b>B</b> Oigital	Design and	d Com	puter	Archite	ecture,	, 2 <sup>nd</sup> Edition, 201	2





#### K-Maps with Don't Cares

7						
0	A 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1					
	Δ	R	C	D	\ \ \	
5	0	<u> </u>	0	0	1	
	0	0	0	1	0	
	Θ	0	1	0	1	
	0	0	1	1	1	
	0	1	0	0	0	
	0	1	0	1	X	
	0 0	⊥ 1	⊥ 1	⊍ 1	1 1	
2	1	0	0	0	1	
	1	0	0	1	1	
	1	0	1	0	X	
	1	0	1	1	X	
	1	1	0	0	X	
	1	1	0	1	X	
2	1	1	1	<b>⊍</b> 1	X	
	т.	<b>T</b>	Т.	Т.	_ ^	
2						
Pigital	Design and Co	mputer	Archit	ecture	, 2 <sup>nd</sup> Edition,	2012

Y				
CDA	B 00	01	11	10
00	1	0	X	1
01	0	X	X	1
11	1	1	Х	X
10	1	1	X	Х



#### K-Maps with Don't Cares

0	A 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1					
	А	В	С	D	Y	
U	0	0	0	0	1	
	Θ	0	0	1	0	
	Θ	0	1	0	1	
	Θ	0	1	1	1	
_1	0	1	0	0	0	
	0	1	0	1	X	
J	0	1	1	0	1	
	U 1	T T	T	T	<u> </u>	
	1	0 0	0	1	1 1	
O	1	0	1	0	X	
	1	0	1	1	X	
	_ 1	1	0	0	X	
<b>T</b>	1	1	0	1	X	
	1	1	1	0	X	
	1	1	1	1	X	
20						
44						
5						
Digital	Design and Co	mnuter	Archi+	ectura	2nd Edition	2012
Jigital	Design and Con	πρατει	AIGIII	ecture	, Z LUILIUI	1, 2012

Y	5			
CDA	00 B	01	11	10
00	1	0	X	1
01	0	Х	Х	1
11	1	1	X	X
10	1	1	X	X

$$Y = A + \overline{B}\overline{D} + C$$



#### Combinational Building

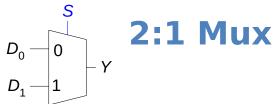


Chapter 2 <66>

### Multiplexer (Mux)

- Selects between one of N inputs to connect to output
- Selects betweer connect to outpu
   log<sub>2</sub>N-bit select i input
   Example:

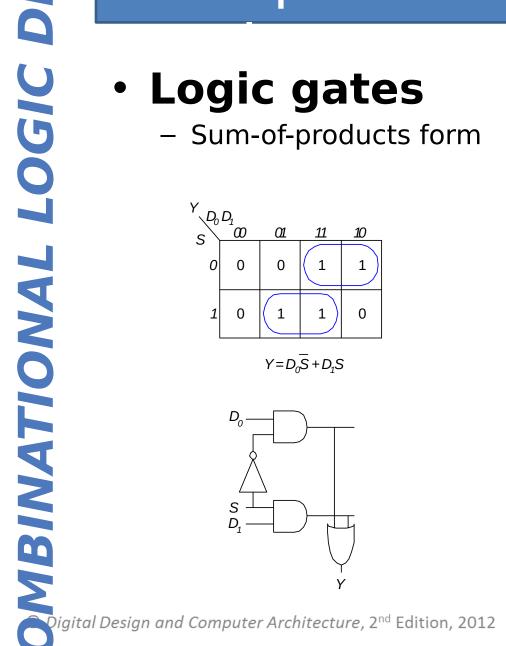
  | S D\_1 D\_0 Y | O D\_1 | 1 | O D\_1 | O log<sub>2</sub>N-bit select input – control

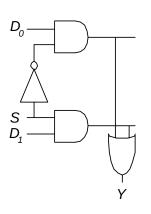


	S	$D_1$	$D_0$	Y		S	Y
•	0	0	0	0	_	0	$D_0$
	0	0	1	1		1	$D_1$
	0	1	0	0			_
	0	1	1	1			
	1	0	0	0			
	1	0	1	0			
	1	1	0	1			
	1	1	1	1			



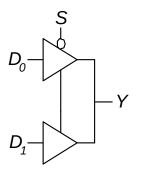
# Multiplexer





#### **Tristates**

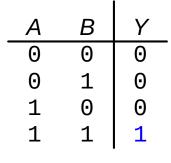
- For an N-input mux, use N tristates
- Turn on exactly one to select the appropriate input



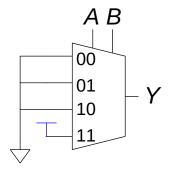


#### Logic using Multiplexers

Using the mux as a lookup table



$$Y = AB$$

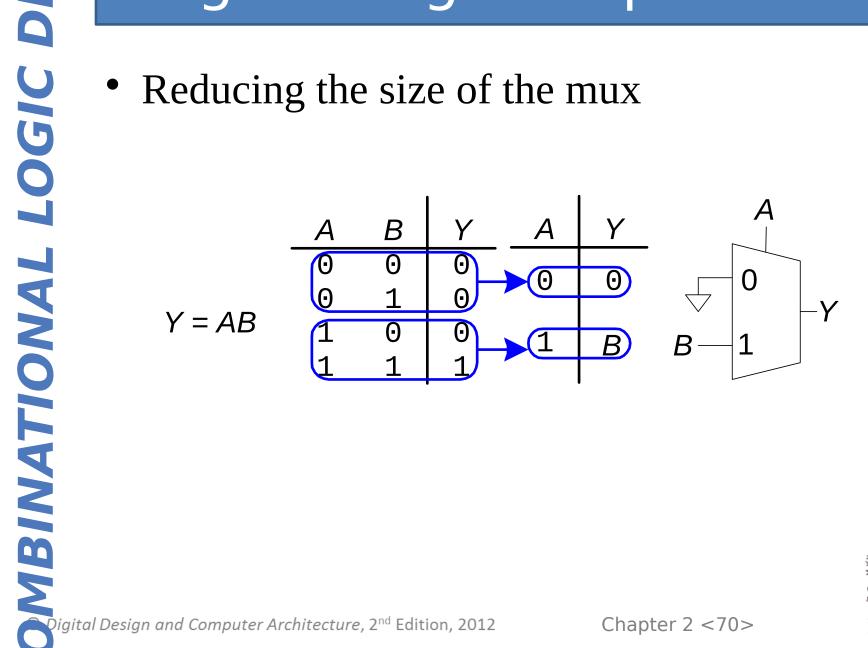




Chapter 2 <69>

#### Logic using Multiplexers

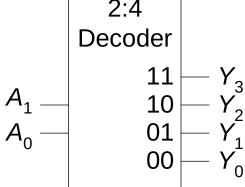
• Reducing the size of the mux





#### Decoders

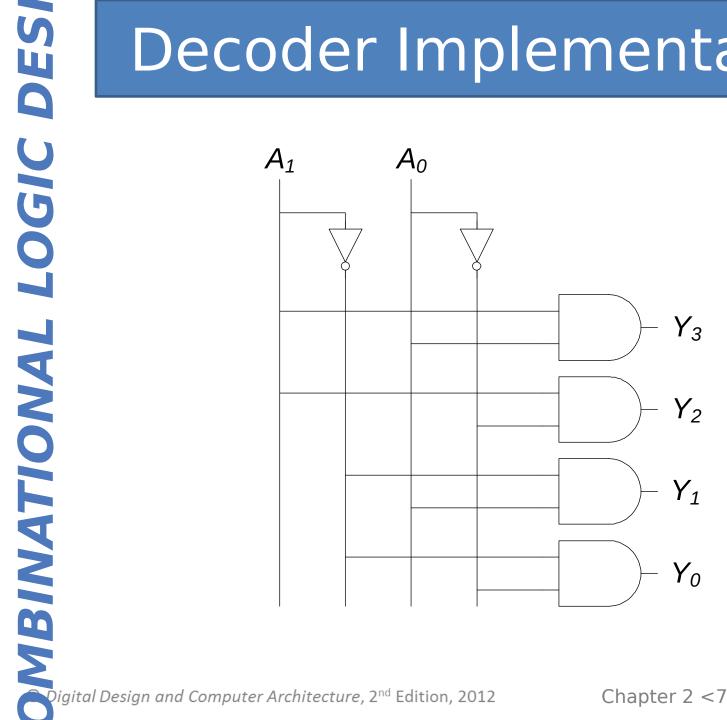
- N inputs,  $2^N$  outputs
   One-hot outputs: only once  $\begin{array}{c|c}
   & 2:4 \\
   & 2:4 \\
   & Decode
  \end{array}$   $\begin{array}{c|c}
   & A_1 & A_0 & Y_3 \\
   & A_0 & 0 \\
   & 0 & 0 \\
   & 0 & 0 \\
   & 0 & 1 \\
   & 0 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 1 \\
   & 1 & 0 \\
   & 1 & 1 \\
   & 1 & 0 \\
   & 1 & 1 \\
   & 1 & 0 \\
   & 1 & 1 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 1 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 1 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   & 1 & 0 \\
   &$ One-hot outputs: only one output HIGH at



$A_1$	$A_0$	$Y_3$	$Y_2$	$Y_1$	$Y_0$
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

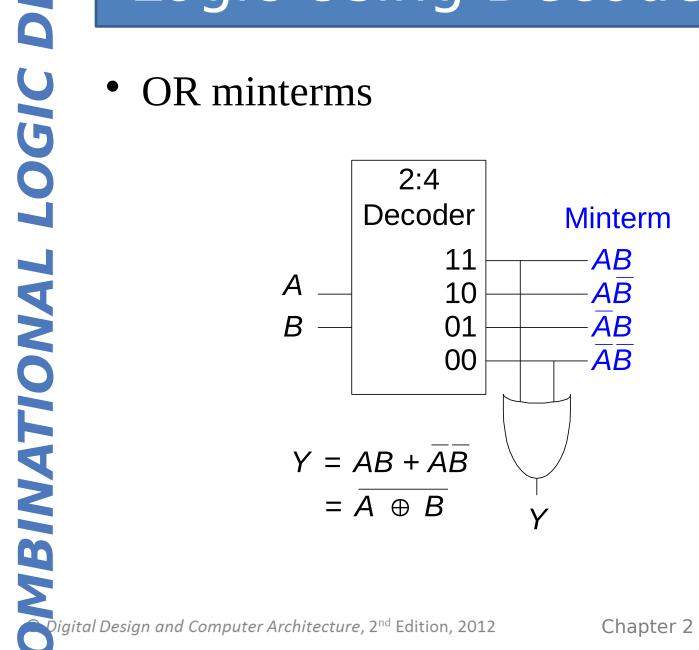


# Decoder Implementation





#### Logic Using Decoders



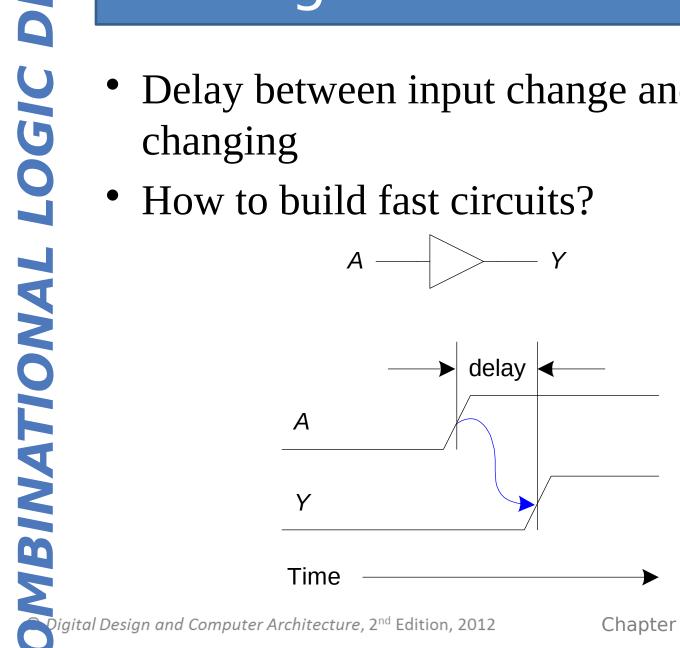


# **ENOUGH FOR**



# Timing

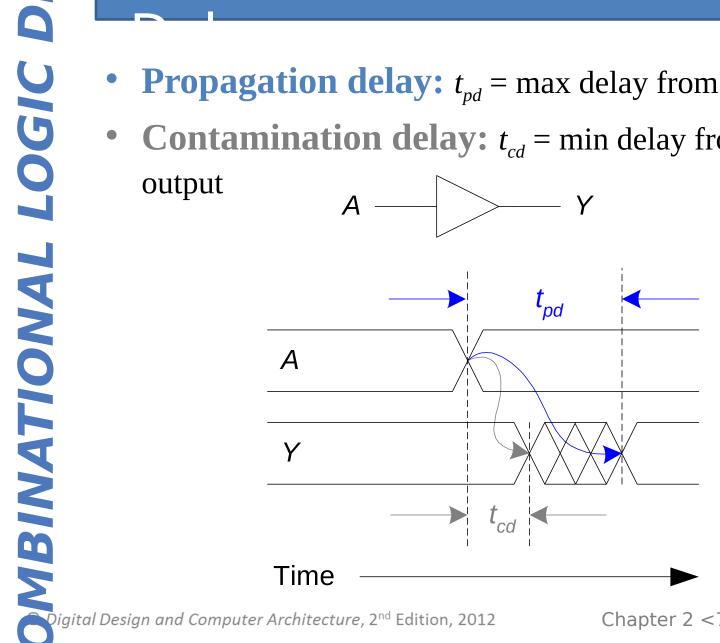
- Delay between input change and output
- How to build fast circuits?





#### Propagation & Contamination

- **Propagation delay:**  $t_{pd}$  = max delay from input to output
- **Contamination delay:**  $t_{cd} = \min \text{ delay from input to}$



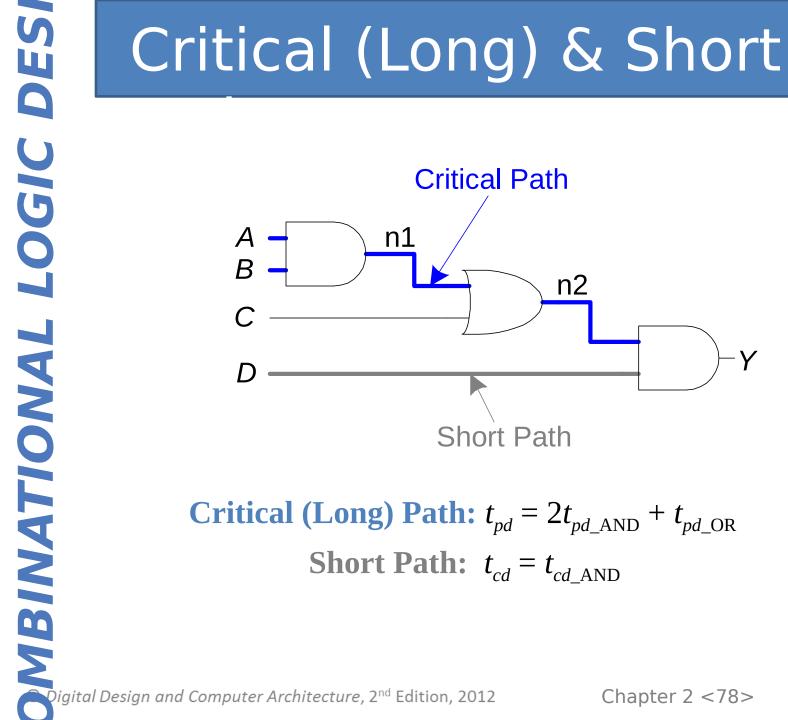


#### Propagation & Contamination

- - Capacitance and resistance in a circuit
  - Speed of light limitation
- Reasons why  $t_{pd}$  and  $t_{cd}$  may be different:
  - Different rising and falling delays
  - Multiple inputs and outputs, some of which are
  - Circuits slow down when hot and speed up when



#### Critical (Long) & Short



Critical (Long) Path: 
$$t_{pd} = 2t_{pd\_AND} + t_{pd\_OR}$$

**Short Path:**  $t_{cd} = t_{cd \text{ AND}}$ 



#### Glitches

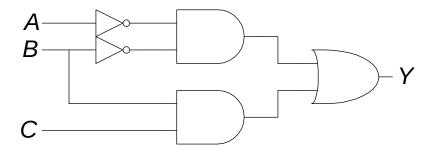
• When a single input to change multiple to change multiple to When a single input change causes an output to change multiple times

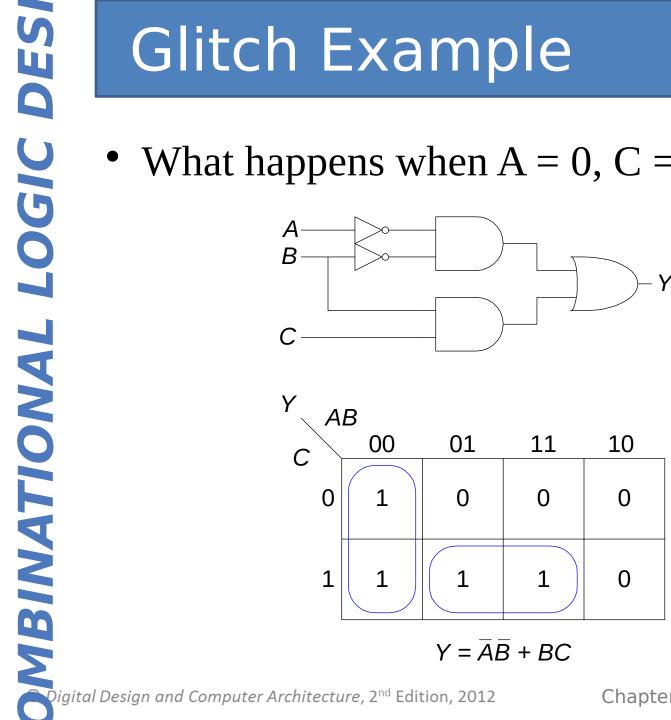


Chapter 2 < 79>

### Glitch Example

• What happens when A = 0, C = 1, B falls?

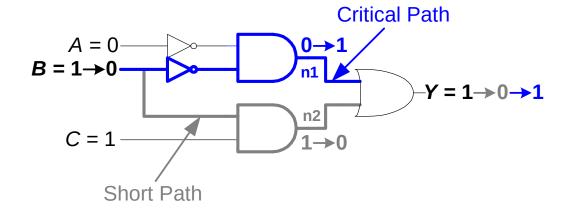


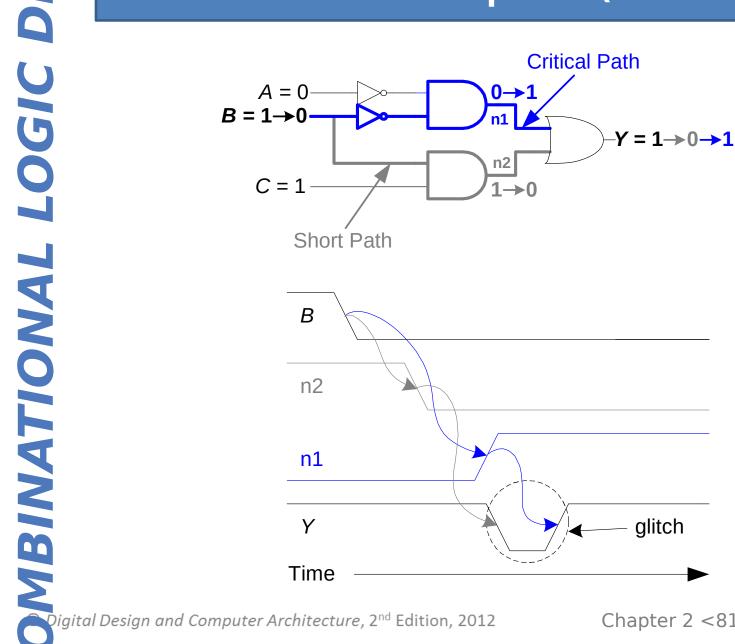






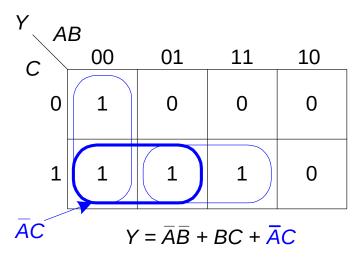
#### Glitch Example (cont.)

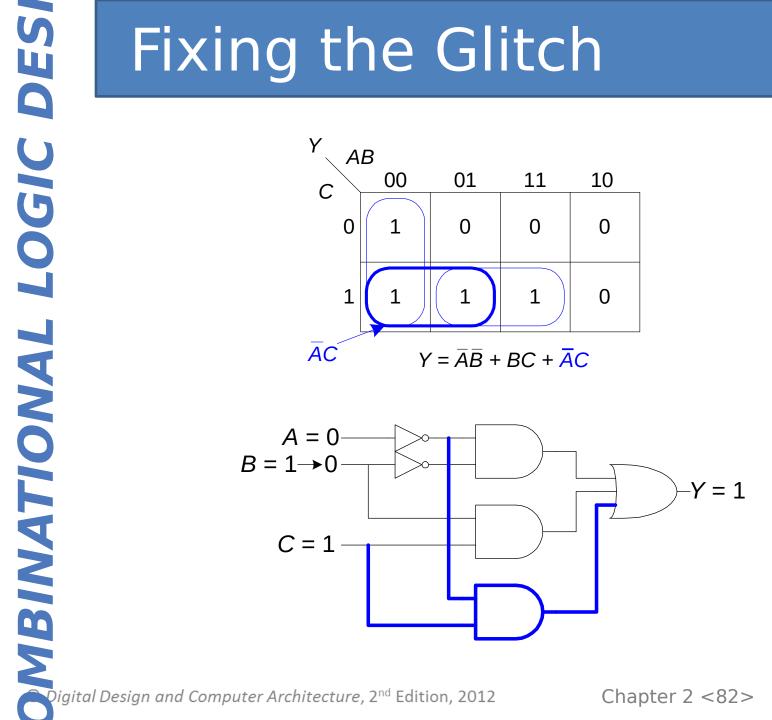






# Fixing the Glitch







#### Why Understand Glitches?

- Glitches don't cause synchronous design Chapter 3)
   It's important to recommulations or on oso
   Can't get rid of all glatransitions on multiple cause glitches Glitches don't cause problems because of synchronous design conventions (see
  - It's important to **recognize** a glitch: in simulations or on oscilloscope
  - Can't get rid of all glitches simultaneous transitions on multiple inputs can also

