

# ECE 223 Digital Circuits and Systems

## Boolean Algebra & Logic Gates



M. Sachdev,  
Dept. of Electrical & Computer Engineering  
University of Waterloo

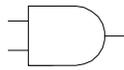
1

## Binary (Boolean) Logic

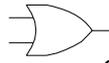
- Deals with binary variables and binary logic functions
- Has two discrete values
  - 0 → False, Open
  - 1 → True, Close
- Three basic logical operations
  - AND (.); OR (+); NOT (')

2

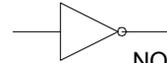
## Logic Gates & Truth Tables



AND



OR



NOT

AND			OR			NOT	
A	B	A.B	A	B	A+B	A	A'
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

- AND; OR gates may have any # of inputs
  - AND  $\leftarrow$  1 if all inputs are 1; 0 other wise
  - OR  $\leftarrow$  1 if any input is 1; 0 other wise

3

## Boolean Algebra

- Branch of Algebra used for describing and designing two valued state variables
  - Introduced by George Boole in 19<sup>th</sup> century
  - Shannon used it to design switching circuits (1938)
- Boolean Algebra – Postulates
  - An algebraic structure defined by a set of elements, B, together with two binary operators + and . that satisfy the following postulates:
    1. Postulate 1:  
Closure with respect to both (.) and (+)
    2. Postulate 2:  
An identity element with respect to +, designated by 0. An identity element with respect to . designated by 1

4

## Boolean Algebra - Postulates

3. Postulate 3:  
Commutative with respect to + and .
4. Postulate 4:  
Distributive over . and +
5. Postulate 5:  
For each element a of B, there exist an element a' such that  
(a)  $a + a' = 1$  and (b)  $a.a' = 0$
6. Postulate 6:  
There exists at least two elements a, b in B, such that  $a \neq b$

5

## Boolean Algebra - Postulates

- Postulates are facts that can be taken as true; they do not require proof
  - We can show logic gates satisfy all the postulates

AND			OR			NOT	
A	B	A.B	A	B	A+B	A	A'
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

6

# Boolean Algebra - Theorems

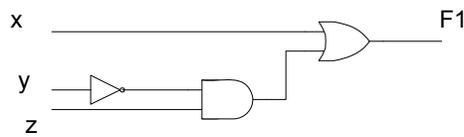
- Theorems help us out in manipulating Boolean expressions
  - They must be proven from the postulates and/or other already proven theorems

Postulate 2	(a) $x + 0 = x$	(b) $x \bullet 1 = x$	
Postulate 5	(a) $x + x' = 1$	(b) $x \bullet x' = 0$	
Theorem 1	(a) $x + x = x$	(b) $x \bullet x = x$	
Theorem 2	(a) $x + 1 = 1$	(b) $x \bullet 0 = 0$	
Theorem 3	$(x')' = x$		involution
Postulate 3	(a) $x + y = y + x$	(b) $x \bullet y = y \bullet x$	commutative
Theorem 4	(a) $x + (y + z) = (x + y) + z$	(b) $x \bullet (y \bullet z) = (x \bullet y) \bullet z$	associative
Postulate 4	(a) $x \bullet (y + z) = x \bullet y + x \bullet z$	(b) $x + y \bullet z = (x + y) \bullet (x + z)$	distributive
Theorem 5	(a) $(x + y)' = x' \bullet y'$	(b) $(x \bullet y)' = x' + y'$	DeMorgan
Theorem 6	(a) $x + x \bullet y = x$	(b) $x \bullet (x + y) = x$	absorption

- **Exercise** – Prove theorems from postulates/other proven theorems

7

# Boolean Functions

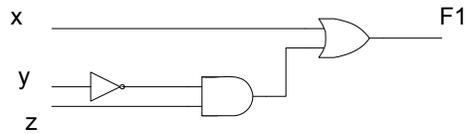


- Are represented as
  - Algebraic expressions;  $F1 = x + y'z$
  - Truth Table
- Synthesis
  - Realization of schematic from the expression/truth table
- Analysis
  - Vice-versa

x	y	z	F1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

8

## Synthesis – F1



- Assume true as well as complement inputs are available
- Cost
  - A 2-input AND gate
  - A 2-input OR gate
  - 4 inputs

x	y	z	F1
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

9

## Canonical and Standard Forms

- Minterms
  - A minterm is an AND term in which every literal (variable) or its complement in a function occurs once
  - For n variable  $\rightarrow 2^n$  minterms
  - Each minterm has a value of 1 for exactly one combination of values of n variables (e.g., n = 3)

10

## Minterms

x	y	z	Corresponding minterm	Designation
0	0	0	$x'y'z'$	$m_0$
0	0	1	$x'y'z$	$m_1$
0	1	0	$x'yz'$	$m_2$
0	1	1	$x'yz$	$m_3$
1	0	0	$xy'z'$	$m_4$
1	0	1	$xy'z$	$m_5$
1	1	0	$xyz'$	$m_6$
1	1	1	$xyz$	$m_7$

- One method of Writing Boolean function is the canonical minterm (sum of products or **SOP**) form
  - $F = x'y'z + xy'z + xyz' = m_1 + m_5 + m_6 = \Sigma(1,5,6)$

11

## Minterms – examples

x	y	z	F2 (Given)	Designation
0	0	0	1	$m_0$
0	0	1	1	$m_1$
0	1	0	1	$m_2$
0	1	1	1	$m_3$
1	0	0	0	
1	0	1	1	$m_5$
1	1	0	0	
1	1	1	0	

$$\begin{aligned}
 F2 &= \Sigma(0,1,2,3,5) \\
 &= x'y'z' + x'y'z + x'yz' + x'yz + xy'z
 \end{aligned}$$

12

## Minterms – examples

x	y	z	F2 (Given)	Designation
0	0	0	1	$m_0$
0	0	1	1	$m_1$
0	1	0	1	$m_2$
0	1	1	1	$m_3$
1	0	0	0	
1	0	1	1	$m_5$
1	1	0	0	
1	1	1	0	

- $(F2)' = \sum(\text{all minterms not in } F2) = \sum(4,6,7)$   
 $= xy'z' + x'yz' + xyz$

13

## Maxterms

x	y	z	Corresponding maxterm	Designation
0	0	0	$x + y + z$	$M_0$
0	0	1	$x + y + z'$	$M_1$
0	1	0	$x + y' + z$	$M_2$
0	1	1	$x + y' + z'$	$M_3$
1	0	0	$x' + y + z$	$M_4$
1	0	1	$x' + y + z'$	$M_5$
1	1	0	$x' + y' + z$	$M_6$
1	1	1	$x' + y' + z'$	$M_7$

- A maxterm is an OR term in which every literal (variable) or its complement in a function occurs once
  - Each maxterm has a value 0 for one combination of values of n variables

14

## Minterms & Maxterms

- Conversion between minterms & maxterms

$$m_0 = x'y'z' = (x+y+z)' = (M_0)'$$

In general,  $m_i = (M_i)'$

- An alternative method of writing a Boolean function is the canonical maxterm (product of sums or **POS**) form
- The canonical product of sums can be written directly from the truth table

15

## Maxterms

x	y	z	F3 (Given)	Designation
0	0	0	0	$M_0$
0	0	1	1	
0	1	0	0	$M_2$
0	1	1	0	$M_3$
1	0	0	0	$M_4$
1	0	1	1	
1	1	0	1	
1	1	1	0	$M_7$

$$\begin{aligned} F3 &= (x+y+z)(x+y'+z)(x+y'+z')(x'+y+z)(x'+y'+z') \\ &= \pi(0,2,3,4,7) \end{aligned}$$

$$(F3)' = \pi(\text{all maxterm not in } F3)$$

16

## Standard Forms

- In canonical forms, each minterm (or maxterm) must contain all variables (or its complements)

The algebraic expressions can further be simplified

- Example

F4 (x,y,z) = xy + y'z (sum of products, standard form)

F5 (x,y,z) = (x+y')(y+z) (product of sums, standard form)

- Conversion

Standard form can be converted into canonical form using identity elements

$$\begin{aligned} F4 &= xy + y'z = xy.1 + 1.y'z = xy(z+z') + (x+x')y'z \\ &= xyz + xyz' + xy'z + x'y'z = m7 + m6 + m5 + m1 \end{aligned}$$

- How about the conversion from canonical forms to standard forms?
- Exercise – convert F5 into maxterms

17

## Non-Standard Forms

- A Boolean function may be written in non-standard form

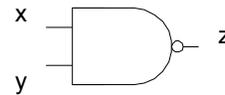
$$\begin{aligned} F6 (x,y,z) &= (xy + z)(xz + y'z) \\ &= xy(xz + y'z) + z(xz + y'z) \\ &= xyz + xy'y'z + xz + y'z \\ &= xyz + xz + y'z \\ &= xz + y'z \text{ (standard form)} \end{aligned}$$

18

## Other Logic Gates – NAND Gate

- So far, we discussed AND, OR, NOT gates
  - 2-input NAND (NOT-AND operation)
  - Can have any # of inputs
  - NAND gate is not associativeAssociative property to be discussed later

x	y	z
0	0	1
0	1	1
1	0	1
1	1	0

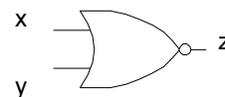


19

## Other Logic Gates – NOR Gate

- 2-input NOR (NOT-OR operation)
  - Can have any # of inputs
  - NOR gate is not associativeAssociative property to be discussed later

x	y	z
0	0	1
0	1	0
1	0	0
1	1	0

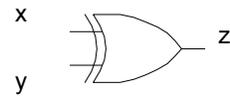


20

## Other Logic Gates – XOR Gate

- 2-input XOR
  - Output is 1 if any input is one and the other input is 0
  - Can have any # of inputs

x	y	z
0	0	0
0	1	1
1	0	1
1	1	0

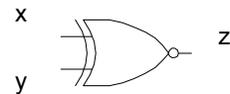


21

## Other Logic Gates – XNOR Gate

- 2-input XNOR
  - Performs the NOT-XOR operation  
Output is 1 if both inputs are 1; or both inputs are 0
  - Can have any # of inputs

x	y	z
0	0	1
0	1	0
1	0	0
1	1	1



22

## Extension to Multiple Inputs

- So far, we restricted ourselves to 1 or 2-input gates
  - A logic gate (except inverter) can have any number of inputs
- AND, OR logic operations have two properties
  - $x + y = y + x$  (commutative)
  - $(x + y) + z = x + (y + z) = x + y + z$  (associative)
- NAND and NOR operations are commutative, but not associative
  - $(x \downarrow y) \downarrow z \neq x \downarrow (y \downarrow z)$        $\downarrow$  = NOR operation
  - $(x \uparrow y) \uparrow z \neq x \uparrow (y \uparrow z)$        $\uparrow$  = NAND operation
- How about XOR

23

## Positive & Negative Logic

- Positive Logic
  - 0 = False (Low Voltage)
  - 1 = True (High Voltage)
- Negative Logic
  - 0 = True (High Voltage)
  - 1 = False (Low Voltage)
- Implement truth table with positive & negative logic
  - Positive logic → AND gate
  - Negative logic → ?

x	y	z
L	L	L
L	H	L
H	L	L
H	H	H

24

## Integrated Circuit - Evolution

- Transistor was invented in 1947/48
- Integrated Circuits were invented in 1959/60
  - Since then, larger # of transistors/chip are integrated
  - 10<sup>1</sup>            Small Scale Integration
  - 10<sup>2-3</sup>        Medium Scale Integration
  - 10<sup>3-6</sup>        Large Scale Integration
  - 10<sup>6-9</sup>        Very Large Scale Integration
- Digital Logic Families (technologies)
  - TTL            Transistor-Transistor Logic
  - ECL            Emitter Coupled Logic
  - MOS            Metal Oxide Semiconductor
  - CMOS        Complementary Metal Oxide Semiconductor

25

## Book Sections – Boolean Algebra & Logic Gates

- Material is covered in Sections 2.1– 2.8

26