COM 205 - Digital Logic Design
Boolean Algebra and Logic Gates -II

Assist. Prof. Özge ÖZTİMUR KARADAĞ

ALKÜ

Last Week

3 terms			Minterms		Maxterms	
x	y	z	term	Designation	term	designation
0	0	0	x’y’z’	m0	x+y+z	M0
0	0	1	x’y’z	m1	x+y+z’	M1
0	1	0	x’yz’	m2	x+y’+z	M2
0	1	1	x’yz	m3	x+y’+z’	M3
1	0	0	xy’z’	m4	x’+y+z	M4
1	0	1	xy’z	m5	x’+y+z’	M5
1	1	0	xyz’	m6	x’+y’+z	M6
1	1	1	xyz	m7	x’+y’+z’	M7

Each maxterm is the complement of its corresponding minterm.

Last Week

- Standard forms:
  - Standard product: product of sums (product of maxterms)
  - Standard sum: sum of products (sum of minterms)
- Obtain the standard form by
  - Truth table
  - Algebraic manipulation

Last Week

- Ex: F = A+B'C

- F=$m_1+m_4+m_5+m_6+m_7$
- F(A,B,C)=$\sum(1,4,5,6,7)$

A	B	C	F
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

Product of Sums (Product of Maxterms)

- In order to represent a Boole function in product of maxterms form, the terms of function should consist of ORs. For this purpose,
  - we apply the distributive law $x+yz = (x+y)(x+z)$
  - After that, the term with a missing literal x is ORed with xx’

Product of Sums (Product of Maxterms)

- Ex: $F=xy+x'z$ Represent the function in product of maxterms (sums) form.
  - $F=(xy+x')(xy+z)$ distributive law
  - $=(x+x')(y+x')(x+z)(y+z)$ distributive law
  - $=(x'+y)(x+z)(y+z)$ one literal is missing in each term
  - $x'+y = x'+y +zz' = (x'+y+z)(x'+y+z')$
  - $x+z=x+z+yy'=(x+y+z)(x+y'+z)$
  - $y+z=xx'+y+z=(x+y+z)(x'+y+z)$
  - $F=(x+y+z)(x'+y+z)(x'+y+z')(x+y'+z)$
  - $=M_0M_4M_5M_2$
  - $=\Pi\ (0,2,4,5)$

Conversion Among Canonical Forms

- How can you represent the complement of a function in sum of minterms form given the truth table of the function?
- The complement of a function represented in sum of minterms form is the sum of minterms which do not appear in the original function. Because the complement of function is equal to the sum of minterms which makes the function 0.
- Ex: F(A,B,C)=$\sum(1,4,5,6,7)$
  - F'(A,B,C)=$\sum(0,2,3) = m_0 + m_2 + m_3$
  - F' → take complement by DeMorgan as follows:
  - F=$(m_0+m_2+m_3)'=m_o'm_2'm_3'=M_0M_2M_3=\prod(0,2,3)$
  - Remember the relation $m_j'=M_j$

Conversion Among Canonical Forms

- While converting a canonic form to another canonic form:
  - $\sum$ and $\prod$ are interchanged
  - the terms which do not appear in the original function appear in the new form.

  - Ex: F(A,B,C)=$\sum(1,4,5,6,7)$
    - F(A,B,C) = $\prod(0,2,3)$

- Boole functions can be converted to the product of maxterms form using truth tables and canonical conversion methods.

3 terms			Minterms		Maxterms	
x	y	z	term	Designation	term	designation
0	0	0	x’y’z’	m0	x+y+z	M0
0	0	1	x’y’z	m1	x+y+z’	M1
0	1	0	x’yz’	m2	x+y’+z	M2
0	1	1	x’yz	m3	x+y’+z’	M3
1	0	0	xy’z’	m4	x’+y+z	M4
1	0	1	xy’z	m5	x’+y+z’	M5
1	1	0	xyz’	m6	x’+y’+z	M6
1	1	1	xyz	m7	x’+y’+z’	M7

Conversion Among Canonical For

- Ex: f=xy+x’z
  - Truth table:

x	y	z	F
0	0	0	0
0	0	1	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

xy = 11 and xz=01 combinations make F=1.

Sum of products:

$$F(x,y,z)=\sum(1,3,6,7)$$

Product of sums:

$$F(x,y,z)=\prod(0,2,4,5)$$

Conversion Among Canonical Forms

- Boole Functions can be represented in non-standard forms;
  - F1=y’+xy+x’yz’
  - F2=x(y’+z)(x’+y+z’+w)

  - Ex: F3=(AB+CD)(A’B’+C’D’) convert the function into a standard forms
    - F3=A’B’CD+ABC’D’ Distributive law
    - F3 = $\sum(m_3, m_{12})$

OTHER LOGIC OPERATIONS

- For n Binary variables there are 2^{2n} functions:
  - Truth table for 16 functions for 2 variables:

x	Y	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

OTHER LOGIC OPERATIONS

- What are these functions?

x	Y	F0	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
		0	.	x/y	x	y/x	y	⊕	+	↓ NOR	⊙	y’	⊂	x’	⊃	↑	1

OTHER LOGIC OPERATIONS

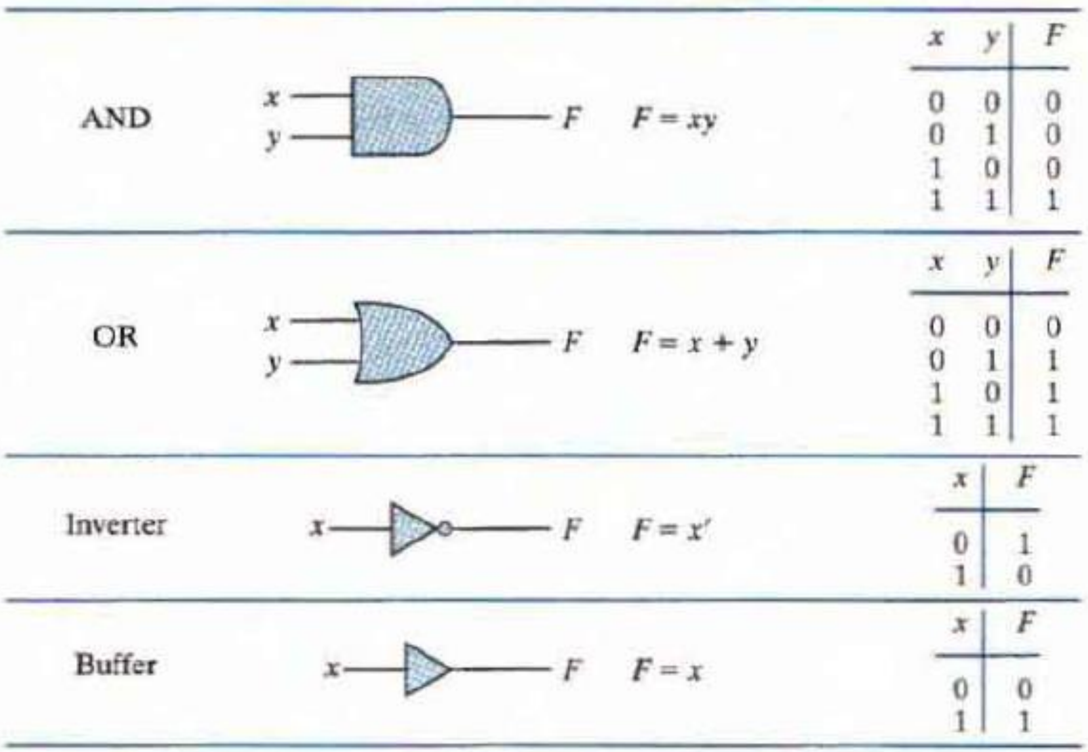
- Functions can be grouped into three:
  - 0 and 1 constants
  - Four functions with unary operations: complement and transfer
  - Ten functions with Binary opeators that define eight different operations; AND, OR, NAND, NOR, exclusive OR, equivalance, inhibation and implication.

Boolean Expressions for the 16 Functions of 2 variables

Boolean Functions	Operator Symbol	Name	Comments
$F_0 = 0$		Null	Binary constant 0
$F_1 = xy$	$x \cdot y$	AND	x and y
$F_2 = xy'$	x/y	Inhibition	x, but not y
$F_3 = x$		Transfer	x
$F_4 = x'y$	y/x	Inhibition	y, but not x
$F_5 = y$		Transfer	y
$F_6 = xy' + x'y$	$x \oplus y$	Exclusive-OR	x or y, but not both
$F_7 = x + y$	$x + y$	OR	x or y
$F_8 = (x + y)'$	$x \downarrow y$	NOR	Not-OR
$F_9 = xy + x'y'$	$(x \oplus y)'$	Equivalence	x equals y
$F_{10} = y'$	y'	Complement	Not y
$F_{11} = x + y'$	$x \subset y$	Implication	If y, then x
$F_{12} = x'$	x'	Complement	Not x
$F_{13} = x' + y$	$x \supset y$	Implication	If x, then y
$F_{14} = (xy)'$	$x \uparrow y$	NAND	Not-AND
$F_{15} = 1$		Identity	Binary constant 1

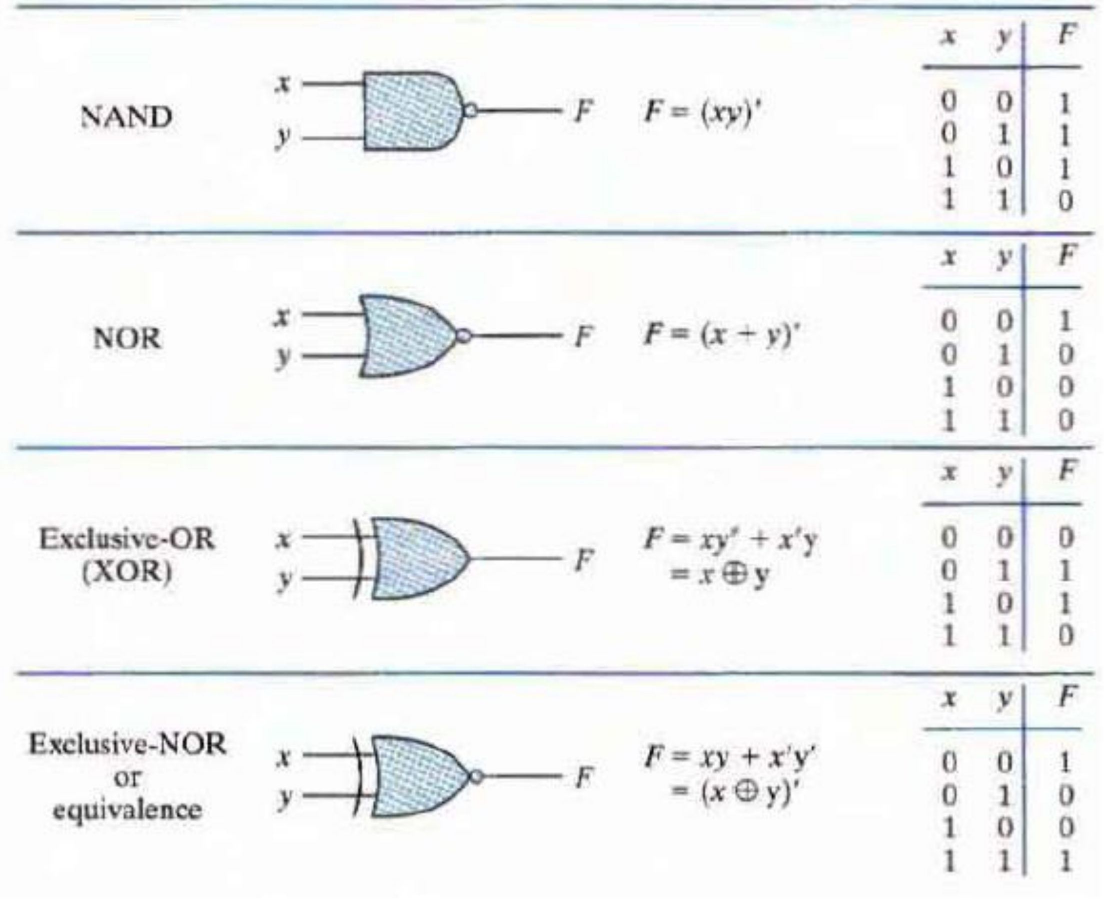
DIGITAL LOGIC GATES

- Inhibation and implication don’t have commutativity and distributivity properties. For that reason, they are not used as standard logic gates.
- Logic Gates:



DIGITAL LOGIC GATES

- Logic Gates:



		x	y	F
NAND	$F = (xy)'$	0	0	1
		0	1	1
		1	0	1
		1	1	0
NOR	$F = (x + y)'$	0	0	1
		0	1	0
		1	0	0
		1	1	0
Exclusive-OR (XOR)	$F = xy' + x'y = x \oplus y$	0	0	0
		0	1	1
		1	0	1
		1	1	0
Exclusive-NOR or equivalence	$F = xy + x'y' = (x \oplus y)'$	0	0	1
		0	1	0
		1	0	0
		1	1	1

DIGITAL LOGIC GATES

- Extension to Multiple Inputs:
  - Gates other than buffer and inverter can be extended to multiple inputs.
  - OR and AND operations have the following properties in Boolean Algebra:
    - Commutative $x + y = y + x$
    - Associative $(x+y)+z = x + (y+z) = x + y + z$

    - Gate inputs can be interchanged and the number of inputs can be increased.

DIGITAL LOGIC GATES

- Extension to Multiple Inputs:
  - NAND and NOR are not associative:
    - $(x \downarrow y) \downarrow z \neq x \downarrow (y \downarrow z)$
    - $(x \downarrow y) \downarrow z$
      - $= [(x+y)' + z]'$
      - $= (x+y) . z'$
      - $= xz'+yz'$

$x \downarrow (y \downarrow z)$

$= [x+(y+z)']'$

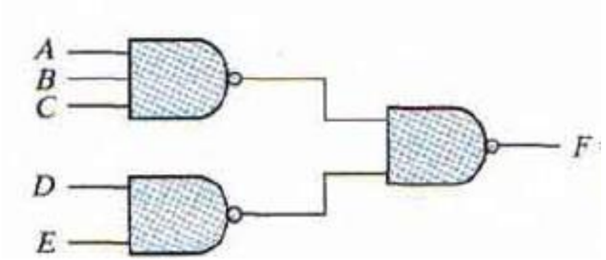
$= x'. (y+z)$

$= x'y + x'z$

  - To overcome this difficulty, we define the multiple NOR (NAND) gate as a complemented OR (AND) gate:
    - $X \downarrow y \downarrow z = (x+y+z)'$
    - $X \uparrow y \uparrow z = (xyz)'$

DIGITAL LOGIC GATES

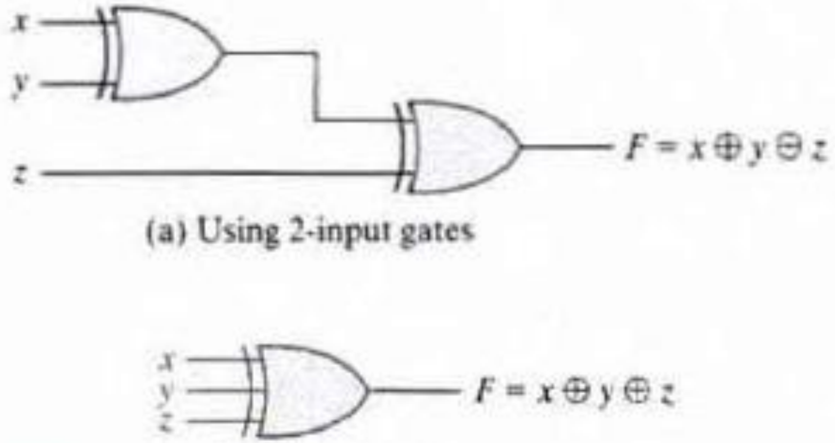
- Ex: F= [(ABC)’(DE)’]’=ABC+DE (DeMorgan)



An expression in sum-of-products form can be implemented with NAND gates.

DIGITAL LOGIC GATES

- Exclusive OR and Equivalance gates are both commutative and associative and can be extended to more than two inputs:



(a) Using 2-input gates

(b) 3-input gate

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

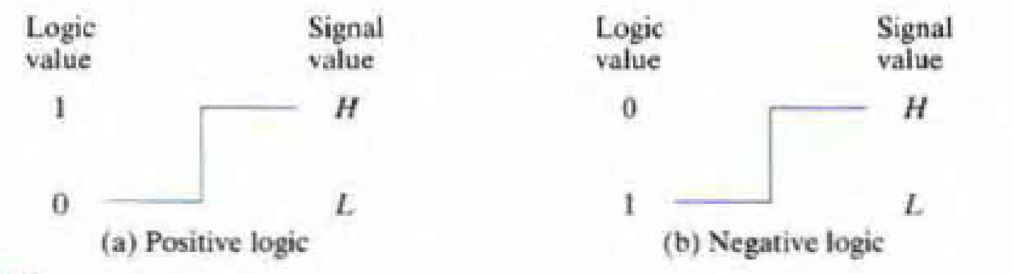
(c) Truth table

F =1 when there is an odd number of 1 in inputs.

Positive and Negative Logic

- Binary logic has two signals
  - 0 and 1

- Assignment of signal level to logic value?
  - Positive logic
  - Negative Logic



(a) Positive logic (b) Negative logic