Boolean Algebra and Logic Gates

Dr. Bassem A. Abdullah

Computer and Systems Department

Lectures Prepared by Dr.Mona Safar, Edited and Lectured by Dr.Bassem A. Abdullah

Outline

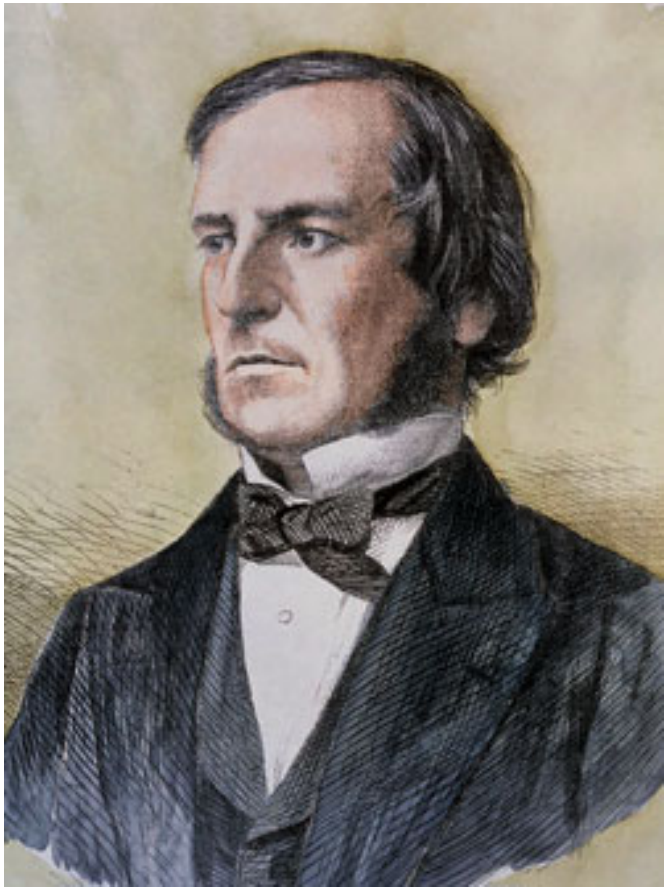
1. Basic Definitions
2. Axiomatic Definition of Boolean Algebra
3. Basic Theorems and Properties of Boolean Algebra
4. Boolean Functions
5. Canonical and Standard Forms
6. Other Logical Operations
7. Digital Logic Gates
8. Integrated Circuits

Boolean Algebra

- In mathematics and mathematical logic, Boolean algebra is the branch of algebra in which:
  - The values of the variables are the truth values true and false, usually denoted 1 and 0 respectively.
  - Instead of elementary algebra where the values of the variables are numbers, and the main operations are addition and multiplication, the main operations of Boolean algebra are the conjunction and, denoted ∧, the disjunction or, denoted ∨, and the negation not, denoted ¬.
- It is thus a formalism for describing logical relations in the same way that ordinary algebra describes numeric relations.

Historical Background

- George Boole (an English mathematics professor 1815-1864) introduced Boolean algebra in his first book The Mathematical Analysis of Logic (1847)
- The term "Boolean algebra" was first suggested by Sheffer in 1913.



George Boole (British)
1815-1864

Historical Background

- In the 1930s, Claude Shannon observed that one could apply the rules of Boole's algebra in switching circuits.
- He introduced **switching algebra** as a way to analyze and design circuits by algebraic means in terms of logic gates.
- Efficient implementation of Boolean functions is a fundamental problem in the design of combinational logic circuits.
- Modern electronic design automation tools for VLSI circuits often rely on an efficient representation of Boolean functions.

Claude Shannon (American)
1916-2001

Features of Boolean Algebra

- Boolean algebra is composed of:
  - **Sets of elements S**
  - **Set of operators like** •(Λ), ′(¬), +(V)
  - **Postulates**

The Postulates for Various Algebra Structure

1. Closure
2. Associative law
3. Commutative law
4. Identity element
5. Inverse
6. Distributive law

Boolean Algebra Postulates

- Closure

A set S is closed with respect to (w.r.t.) an operator if, for operands consisting of elements of S, the operator specifies a rule for obtaining a unique element of S.

- Example:

N = {1, 2, 3, …}, the set of natural number.

  - Operator(+): closed
  - Minus(-):not closed (2-3=-1)

Boolean Algebra Postulates (cont.)

- Closure:
  - Closure w.r.t operator $\bullet$:

    $0 \bullet 0 = 0$ $\quad$ $1 \bullet 0 = 1$

    $0 \bullet 1 = 0$ $\quad$ $1 \bullet 1 = 1$
  - Closure w.r.t operator $+$:

    $0 + 0 = 0$ $\quad$ $1 + 0 = 1$

    $0 + 1 = 1$ $\quad$ $1 + 1 = 1$

Boolean Algebra Postulates (cont.)

- Identity element:
  - Identity element w.r.t operator $\bullet$: 1

    $x \bullet 1 = x$
  - Identity element w.r.t operator $+$: 0

    $x + 0 = x$

Boolean Algebra Postulates (cont.)

- Inverse element:
  - For every element x, there exists element x’ (complement of x) such that

$$x + x' = 1 \qquad x \bullet x' = 0$$

Boolean Algebra Postulates (cont.)

- Commutative:
  - Commutative w.r.t operator ● :

    $x \bullet y = y \bullet x$
  - Commutative w.r.t operator + :

    $x + y = y + x$

Boolean Algebra Postulates (cont.)

- Associative:
  - Associative w.r.t operator ● :

    $x \bullet (y \bullet z) = (x \bullet y) \bullet z$
  - Associative w.r.t operator + :

    $x + (y + z) = (x + y) + z$

Boolean Algebra Postulates (cont.)

- Distributive:
  - $\bullet$ is distributive over $+$:

    $x \bullet (y + z) = (x \bullet y) + (x \bullet z)$
  - $+$ is distributive over $\bullet$:

    $x + (y \bullet z) = (x + y) \bullet (x + z)$

Differences Between Boolean Algebra and Ordinary Algebra

- The distributive law over + holds for Boolean algebra but not ordinary algebra

$$x + (y \bullet z) = (x + y) \bullet (x + z)$$

- Boolean algebra has no additive or multiplicative inverse. Instead Boolean algebra has complement

$$x + x' = 1 \qquad x \bullet x' = 0$$

Duality Principle

- Every algebraic expression deducible from the postulates of Boolean algebra remains valid if the operators and identity elements are interchanged.
- To get dual form:
  - Interchange OR(+) and AND(.)
  - Toggle 0's and 1's

Basic Theorems and Postulates

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

Operator Precedence

1. Parentheses
2. Not
3. AND
4. OR

Outline

1. Basic Definitions
2. Axiomatic Definition of Boolean Algebra
3. Basic Theorems and Properties of Boolean Algebra
4. Boolean Functions
5. Canonical and Standard Forms
6. Other Logical Operations
7. Digital Logic Gates
8. Integrated Circuits

Theorems of Boolean Algebra

- Any of those theorems or postulates can be proofed by truth table or using the other theorems or postulates.
- ***NOTE:*** $x \cdot y$ ***is*** $\equiv$ ***to*** $x\ y$

Example

- Prove Theorem 1(a) : x + x = x

x + x = (x + x) . 1	by postulate	2-b
= (x + x) . (x + x')		5-a
= x + xx'		4-b
= x + 0		5-b
= x		2-a

Example

- Prove Theorem 1(b) : x . x = x

x . x = (x . x) + 0	by postulate 2-a
= (x . x) + (x . x')	5-b
= x (x + x')	4-a
= x . 1	5-a
= x	2-b

- Theorem 1(b) is dual of 1(a), each step of proof for 1(b) is dual for the corresponding step of proof of 1(a).
- Any dual theorem can be derived by proof of its pair.

Example

- Prove Theorem 2(a) : x + 1 = 1

  x + 1 = (x + 1) . 1 by postulate 2-b

  = (x + 1) . (x + x') 5-a

  = x + (x'.1) 4-b

  = x + x' 2-b

  = 1 5-a

- Prove Theorem 2(b) : x . 0 = 0 by duality

Example

- Prove Theorem 6(a) : x + xy = x (absorption)

  x + xy = (x . 1) + (x . y) by postulate 2-b

  = x (1 + y) 4-a

  = x . 1

  = x

Example

- Verify the absorption theorem by Truth Table.

x	y	xy	x+xy
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

$$x + x\,y = x$$

Example

- Verify DeMorgan's Theorem by Truth Table

X	y	x + y	(x + y)'	x'	y'	x'y'
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

$$(x + y)' = x' y'$$

Outline

1. Basic Definitions
2. Axiomatic Definition of Boolean Algebra
3. Basic Theorems and Properties of Boolean Algebra
4. Boolean Functions
5. Canonical and Standard Forms
6. Other Logical Operations
7. Digital Logic Gates
8. Integrated Circuits

Boolean function

- Binary variables (0 or1)
- Binary Operators: OR and AND
- Unary Operator NOT
- Parentheses
- An equal sign

Implementing functions

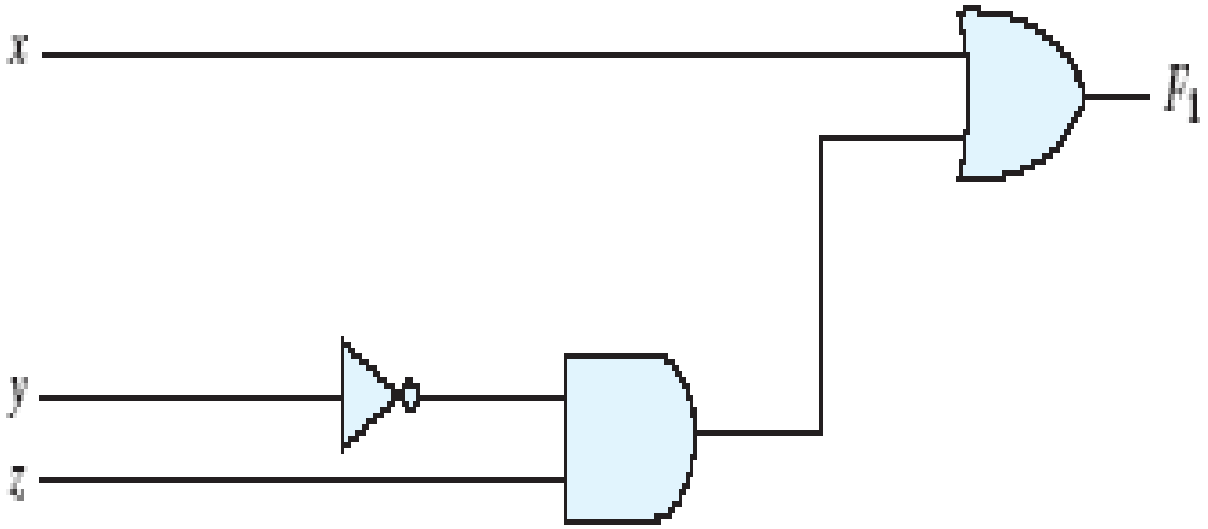
F1 = x + y'z

F2 = x'y'z + x'yz + xy'

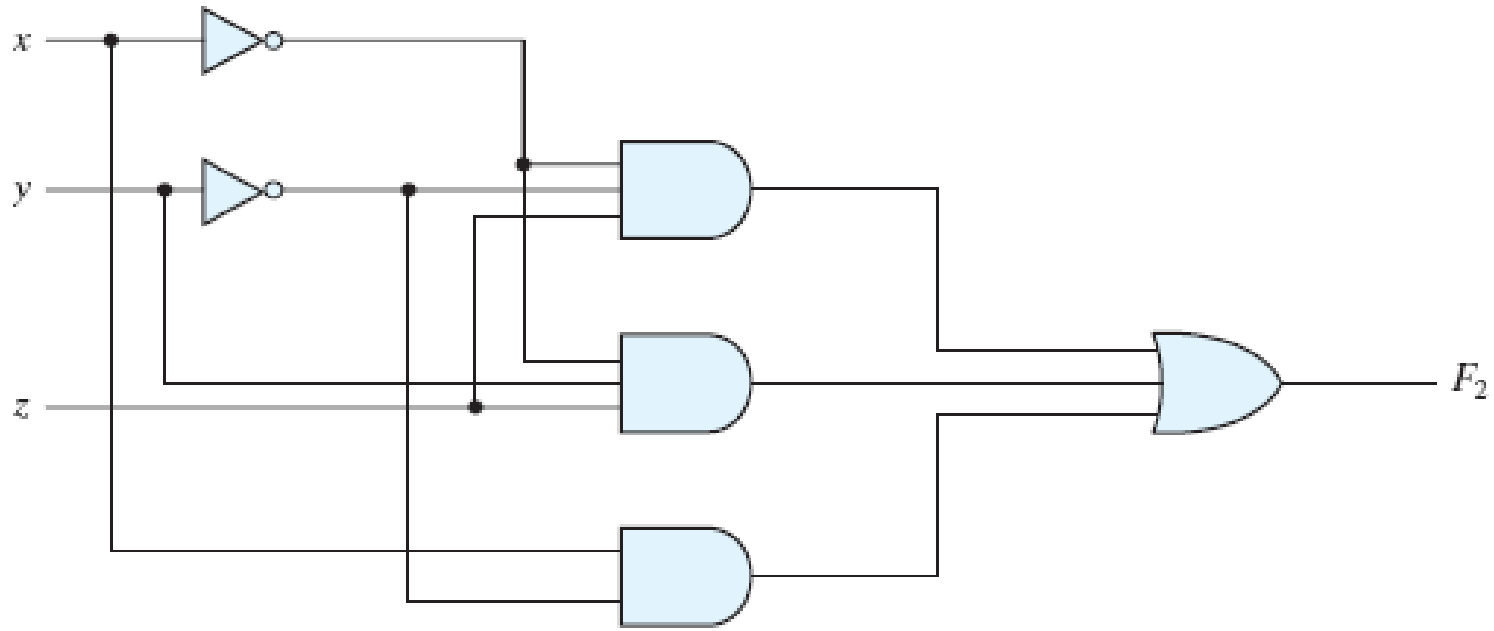
Truth Tables for F_1 and F_2

x	y	z	F_1	F_2
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	0
1	1	1	1	0

Gate Implementation of F1 = x + y'z



Gate Implementation of F2= x'y'z + x'yz + xy'



(a) $F_2 = x'y'z + x'yz + xy'$

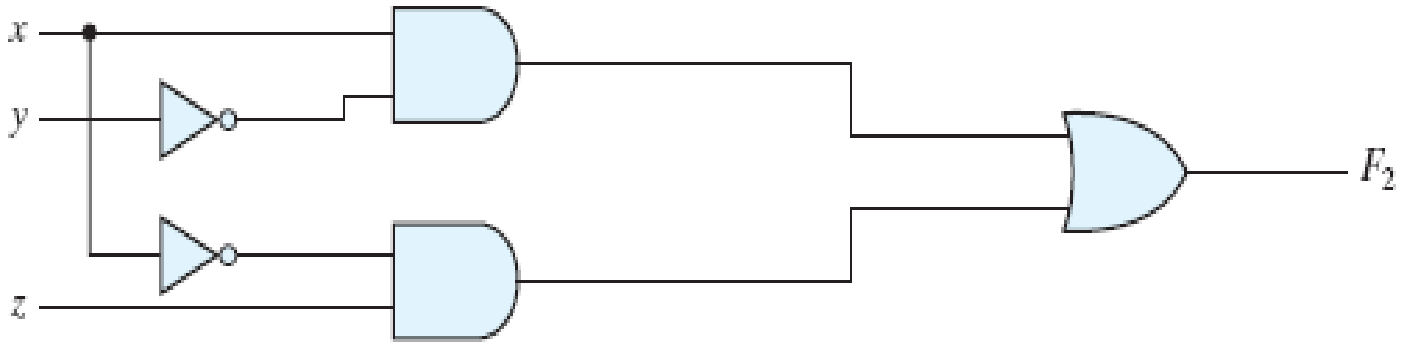
Simplifying F2

- **F2 = x'y'z + x'yz + xy'**

  = x'z(y' + y) + xy'

  = x'z + xy' (reduced form)

Gate Implementation of reduced F2 = x'z + xy'



(b) $F_2 = xy' + x'z$

Simplify the following Boolean Functions

- F = x(x’ + y)

  = xx’ + xy

  = 0 + xy

  = xy

Simplify the following Boolean Functions

- F = x + x'y

  = (x + x')(x + y)

  = 1(x + y)

  = (x + y)

Simplify the following Boolean Functions

- F = (x + y)(x+ y’)

  = x + xy + xy’ +yy’

  = x(1 + y + y’) (yy’= 0)

  = x (y+y’=1)

Complement Function

- Generalized form of DeMorgan's Theorem

(A+ B + C)' = (A + X)'	Let B + C = X
= A' X'	DeMorgan'sTheorem
= A' (B + C)'	B+C = X
= A'(B'C')	DeMorgan's Theorem
= A'B'C'	Associative Law

Example

- Find the Complement Function of:

F1 = x'yz' + x'y'z

F1' = (x'yz' + x'y'z)'

= (x'yz')' . (x'y'z)'

= (x + y' + z) (x + y + z')

Example

- Find the Complement Function of:

F2 = x(y'z' + yz)

F2' = [x(y'z' + yz)]'

= x' + (y'z' + yz)'

= x' + [(y'z')' . (yz)']

= x' + (y + z) (y' + z')

Outline

1. Basic Definitions
2. Axiomatic Definition of Boolean Algebra
3. Basic Theorems and Properties of Boolean Algebra
4. Boolean Functions
5. Canonical and Standard Forms
6. Other Logical Operations
7. Digital Logic Gates
8. Integrated Circuits

Canonical and Standard Forms

- Minterms (Standard Product)
  - Binary variables combined with AND operation
  - All possible products of any n binary variables are called minterms or standard product:
  - Example: For 2 binary variables x and y. Minterms are: x'y',x'y, xy', xy

Canonical and Standard Forms (cont.)

- Maxterms (Standard Sum)
  - Binary variables combined with OR operation
  - All possible sums of any n binary variables are called maxterms or standard sum:
  - Example: For 2 binary variables x and y. Minterms are: x+y, x’+y’, x’+y, x+y'

Minterms and Maxterms

Minterms and Maxterms for Three Binary Variables

			Minterms		Maxterms	
x	y	z	Term	Designation	Term	Designation
0	0	0	$x'y'z'$	m_0	$x + y + z$	M_0
0	0	1	$x'y'z$	m_1	$x + y + z'$	M_1
0	1	0	$x'yz'$	m_2	$x + y' + z$	M_2
0	1	1	$x'yz$	m_3	$x + y' + z'$	M_3
1	0	0	$xy'z'$	m_4	$x' + y + z$	M_4
1	0	1	$xy'z$	m_5	$x' + y + z'$	M_5
1	1	0	xyz'	m_6	$x' + y' + z$	M_6
1	1	1	xyz	m_7	$x' + y' + z'$	M_7

Each maxterm is the complement of its corresponding minterm: **mj' = Mj**

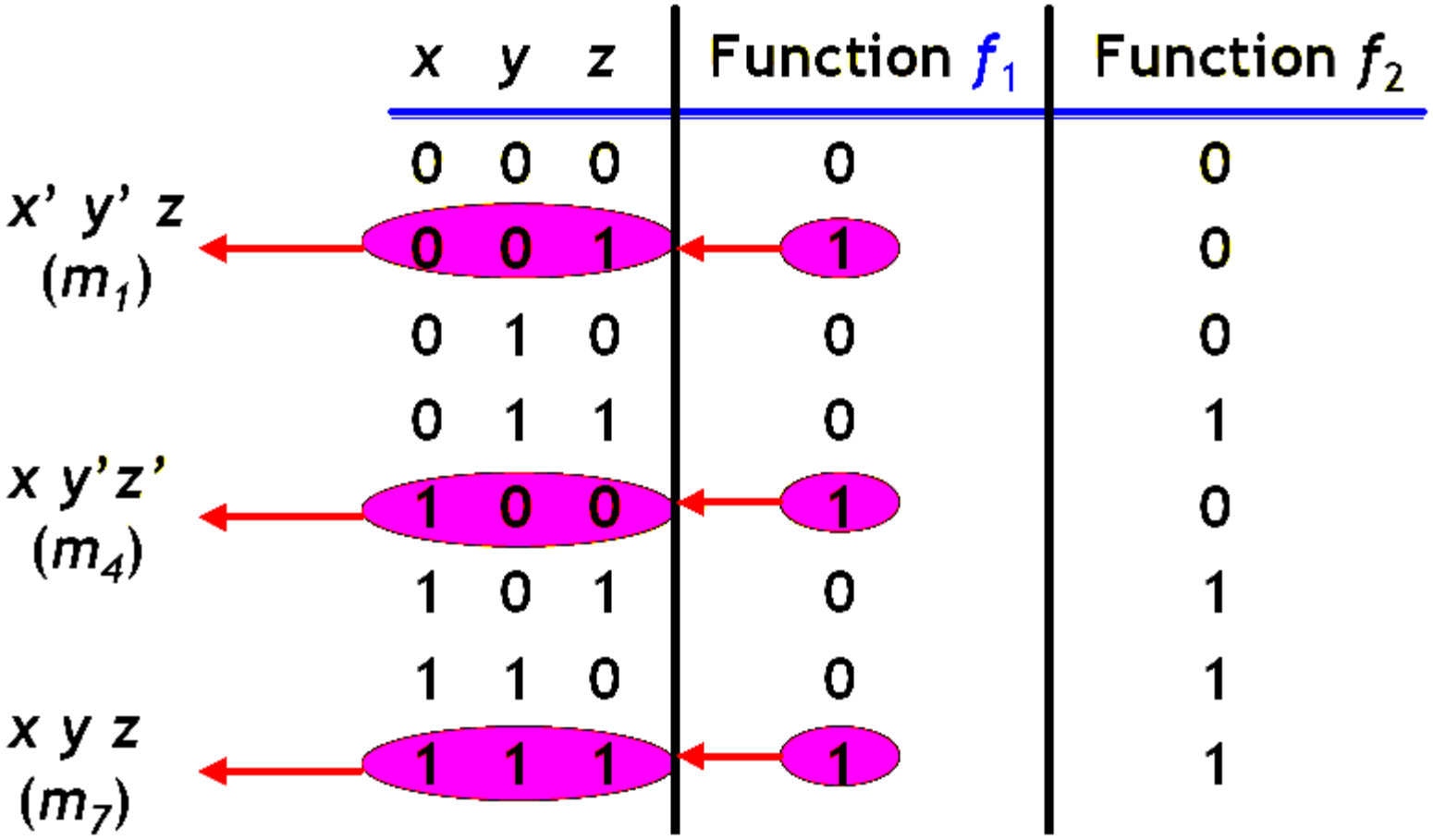
Example

- Truth table of f_1 and f_2

x	y	z	Function f_1	Function f_2
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

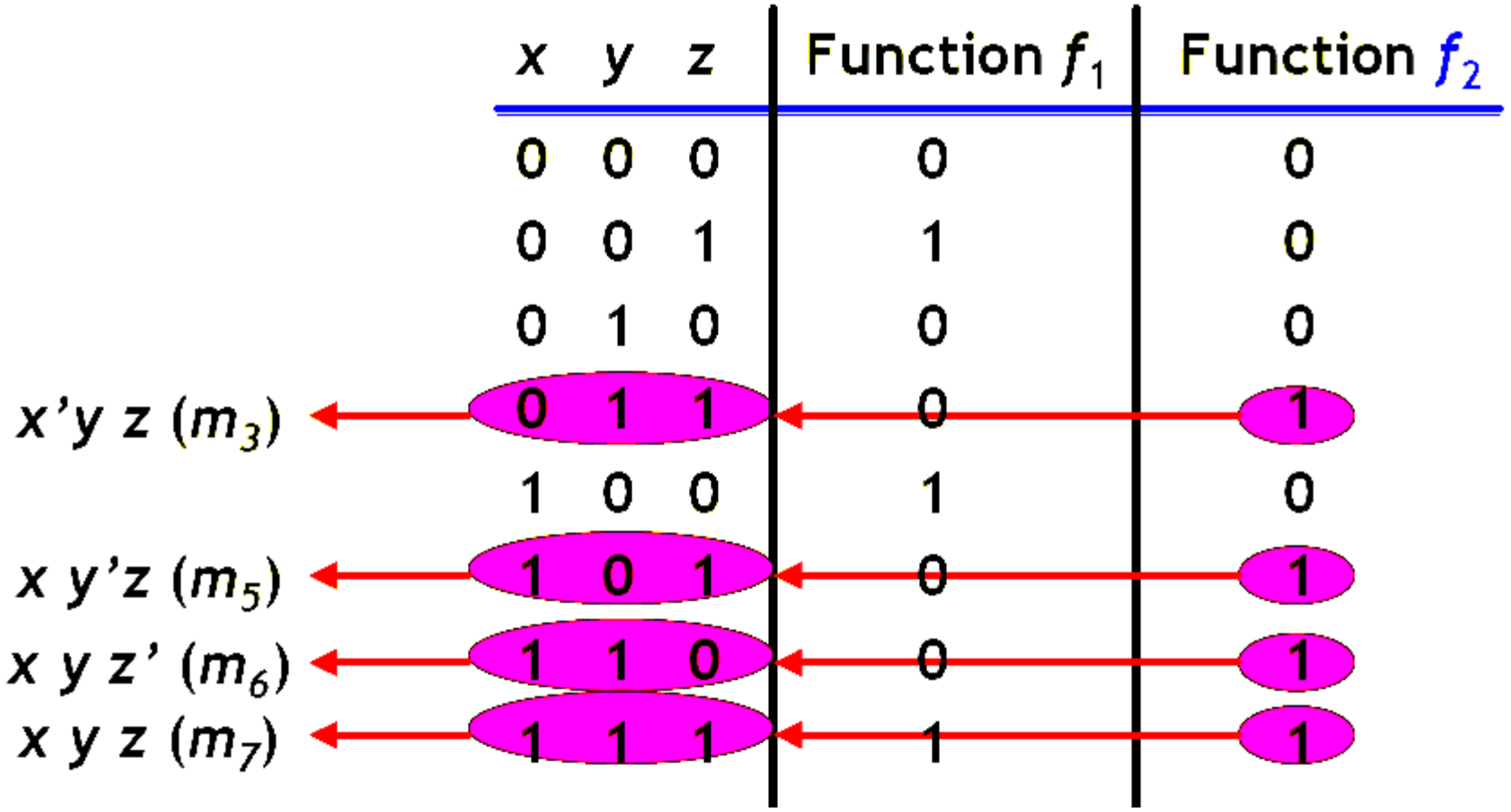
Minterm Result

$f_1 = x'y'z + x\,y'z' + x\,y\,z = m_1 + m_4 + m_7$



Minterm Result

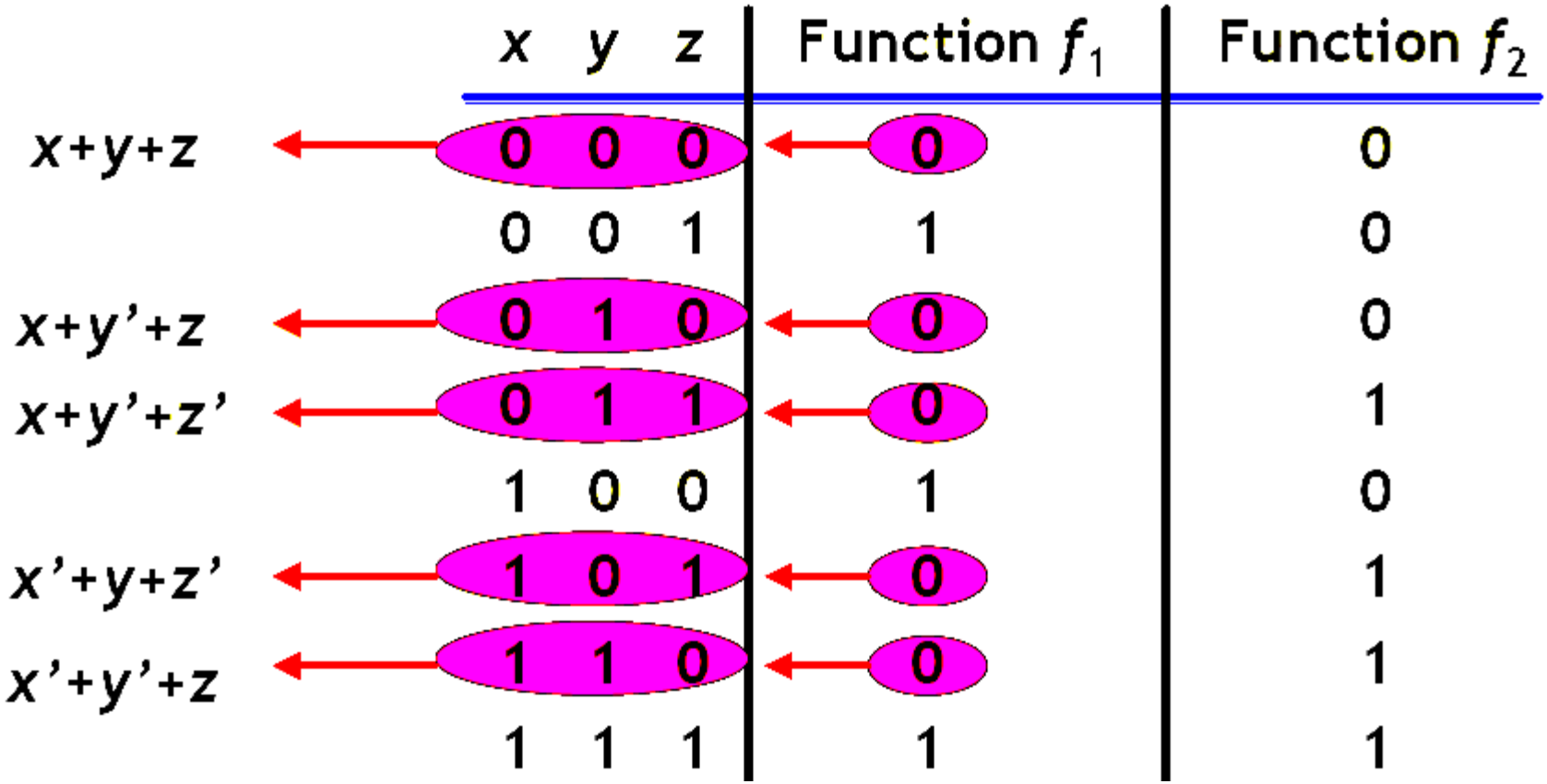
$f_2 = x'y\,z + x\,y'z + x\,y\,z' + x\,y\,z = m_3 + m_5 + m_6 + m_7$



Maxterm Result

$f_1 = (x+y+z)(x+y'+z)(x+y'+z')(x'+y+z')(x'+y'+z)$

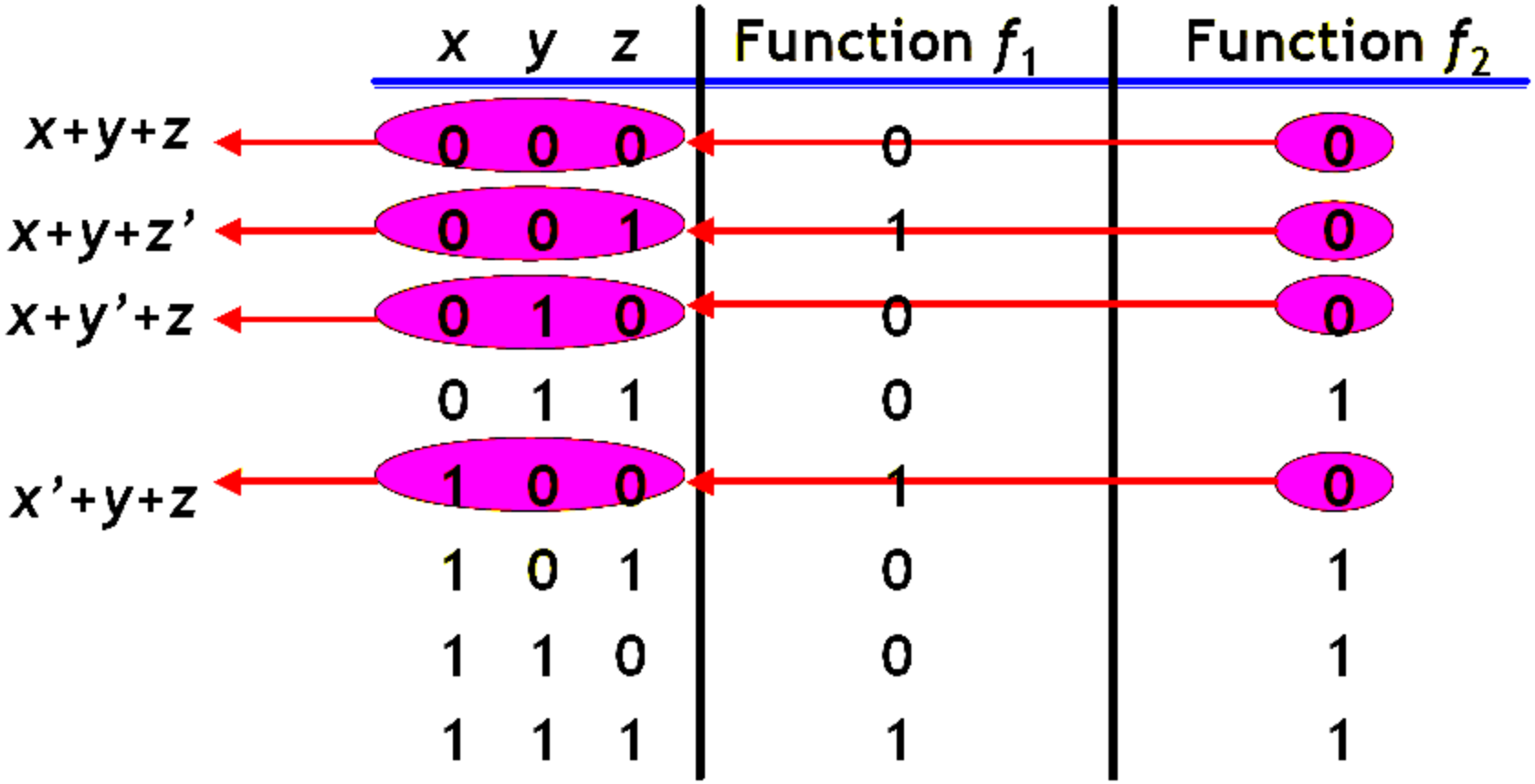
$= M_0 M_2 M_3 M_5 M_6$



Maxterm Result

$f_2 = (x+y+z)\,(x+y+z')\,(x+y'+z)\,(x'+y+z)$

$= M_0\,M_1\,M_2\,M_4$



Canonical forms

- Sum of minterms (Sum of Products SoP)

F1= x'y'z + xy'z' + xyz

= $m_1 + m_4 + m_7$

= $\sum(1, 4, 7)$

F2= x'yz + xy'z + xyz' + xyz

= $m_3 + m_5 + m_6 + m_7$

= $\sum(3, 5, 6, 7)$

Canonical forms (cont.)

- Product of maxterms (Product of Sums PoS)

F1= (x+y+z) (x+y'+z) (x+y'+z') (x'+y+z') (x'+y'+z)

= $M_0M_2M_3M_5M_6$

= $\pi(0, 2, 3, 5, 6)$

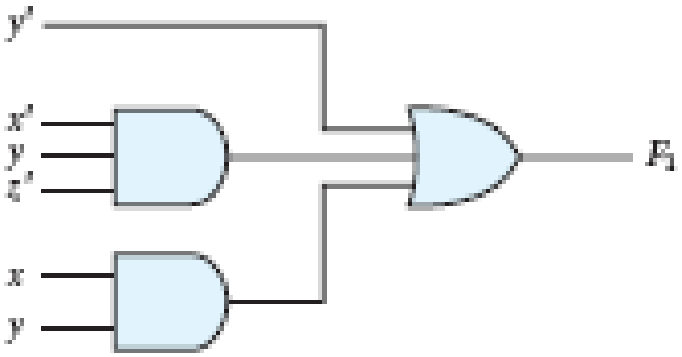
F2= (x+y+z) (x+y+z') (x+y'+z) (x'+y+z)

= $M_0M_1M_2M_4$

= $\pi(0, 1, 2, 4)$

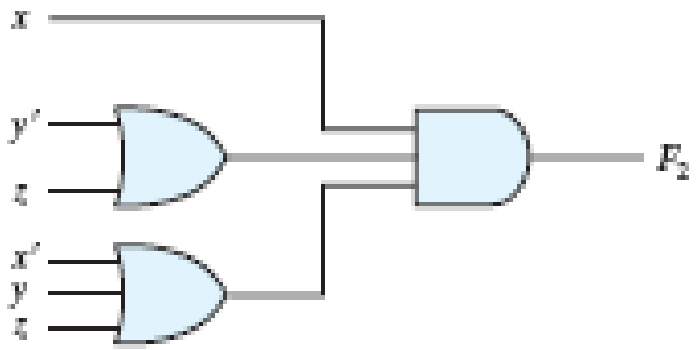
Standard form gate implementation

F1 = y’ + xy + x’yz’



Sum of Products

F2 = x(y’ + z)(x’+y+z’)



Product of Sums

Functions of 2 binary variables

Truth Tables for the 16 Functions of Two Binary Variables

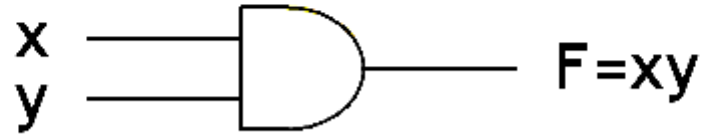
x	y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Boolean Expressions for the 16 Functions of Two 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

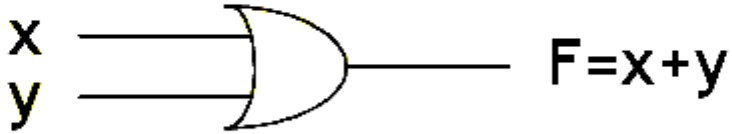
Digital Logic Gates (1/4)

AND



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

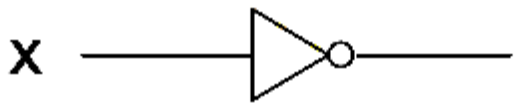
OR



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

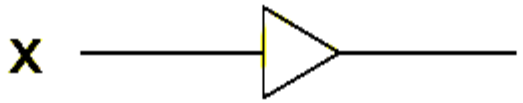
Digital Logic Gates (2/4)

Inverter



x	F
0	1
1	0

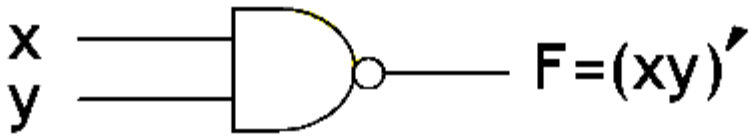
Buffer



x	F
0	0
1	1

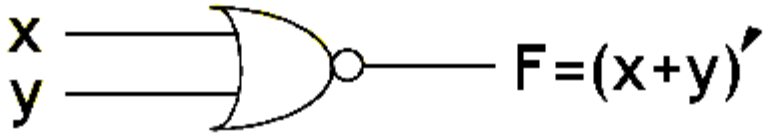
Digital Logic Gates (3/4)

NAND



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

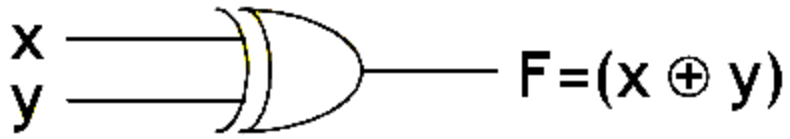
NOR



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

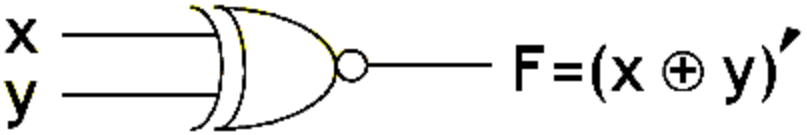
Digital Logic Gates (4/4)

XOR
(Exclusive-OR)



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

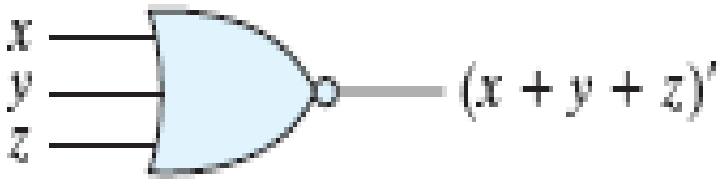
Equivalence
(Exclusive-NOR)



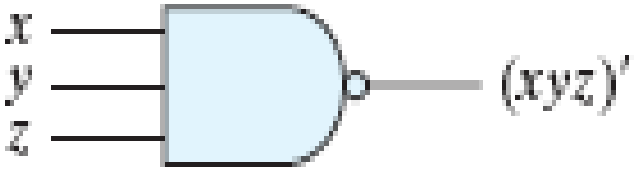
x	y	F
0	0	1
0	1	0
1	0	0
1	1	1

Multiple input logic gate

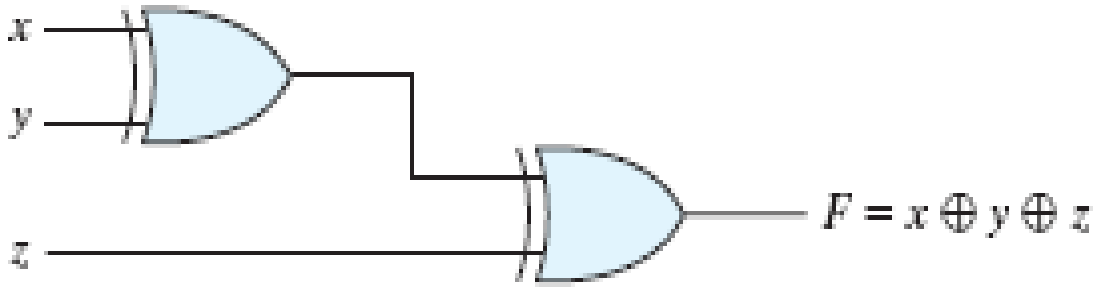
3-input NOR gate



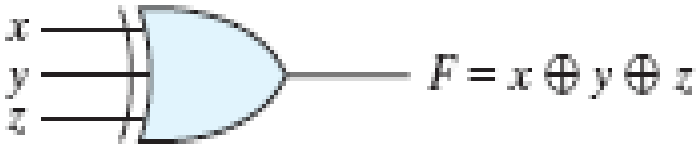
3-input NAND gate



3-input exclusive-OR gates (odd Parity checker)



(a) Using 2-input gates



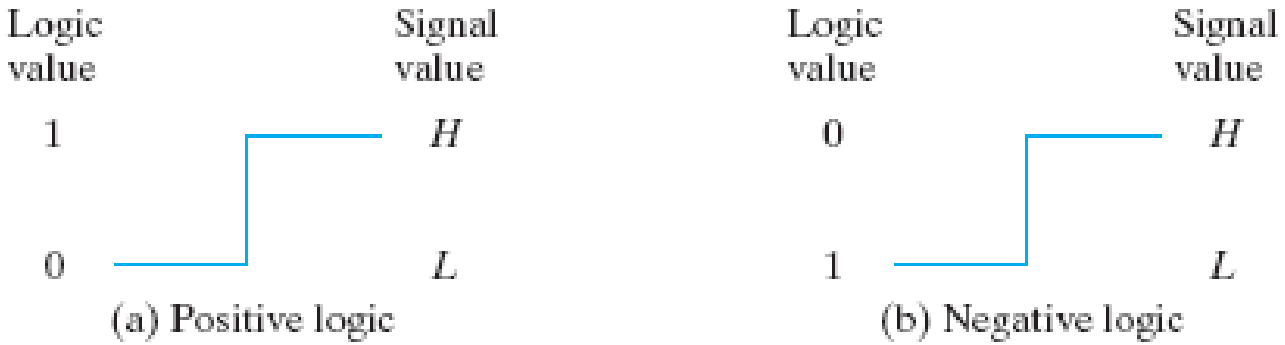
(b) 3-input gate

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

Positive and Negative Logic

Positive Logic		Negative Logic	
Logic level	Signal level	Logic level	Signal level
0	**L**	**0**	**H**
1	**H**	**1**	**L**



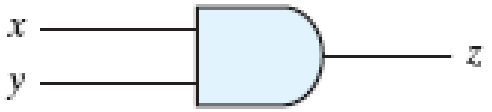
Example of Negative Logic

Truth table with H and L

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

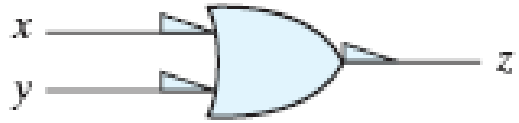
Positive logic
AND gate

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



Negative logic
OR gate

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



Integrated Circuits (ICs)

- Level of integration

1. SSI: Small-scale Integration, Gates < 10
2. MSI: Medium-scale Integration, 10<Gates < 1000
3. LSI: Large-scale Integration, Gates > 1000
4. VLSI: Very Large-scale Integration, Gates > 100000

Digital gates ICs