Ollscoil na Gaillimhe
University of Galway

CT101 Computing Systems

Dr. Bharathi Raja Chakravarthi
Lecturer-above-the-bar
Email: bharathi.raja@universityofgalway.ie

University
ofGalway.ie

Ollscoil na Gaillimhe
University of Galway

Recap

Binary Logic

There are three basic logical operations: AND, OR, and NOT. Each operation produces a binary result, denoted by z.

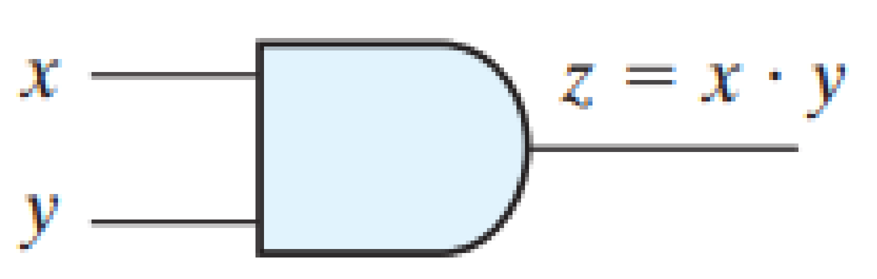
- AND – represented by a dot or absence of an operator. **E.g.**, $x \cdot y = z$ or $xy = z$
- OR – represented by a plus sign. **E.g.,** $x + y = z$
- NOT – represented by a prime (sometimes by an overbar). **E.g.**, $x' = z$ or $\bar{x} = z$

Logic Gates

A logic gate is a simple switching circuit that determines whether an input pulse can pass through to the output in digital circuits.

(a) **AND Gate:**

Two-input AND gate

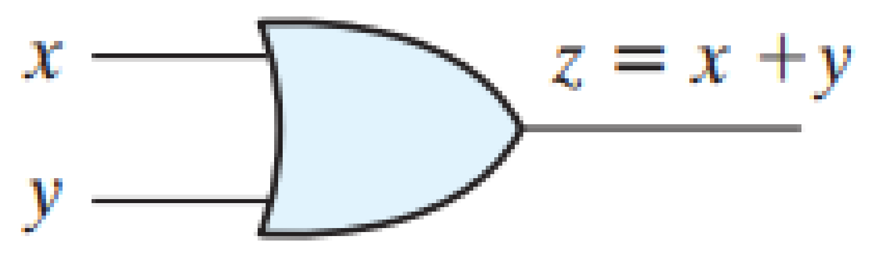


Truth Table

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

Logic Gates

(b) **OR Gate:**

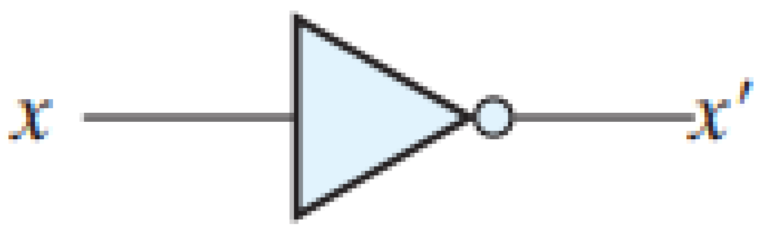


Truth Table

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

Logic Gates

(c) **NOT Gate:**



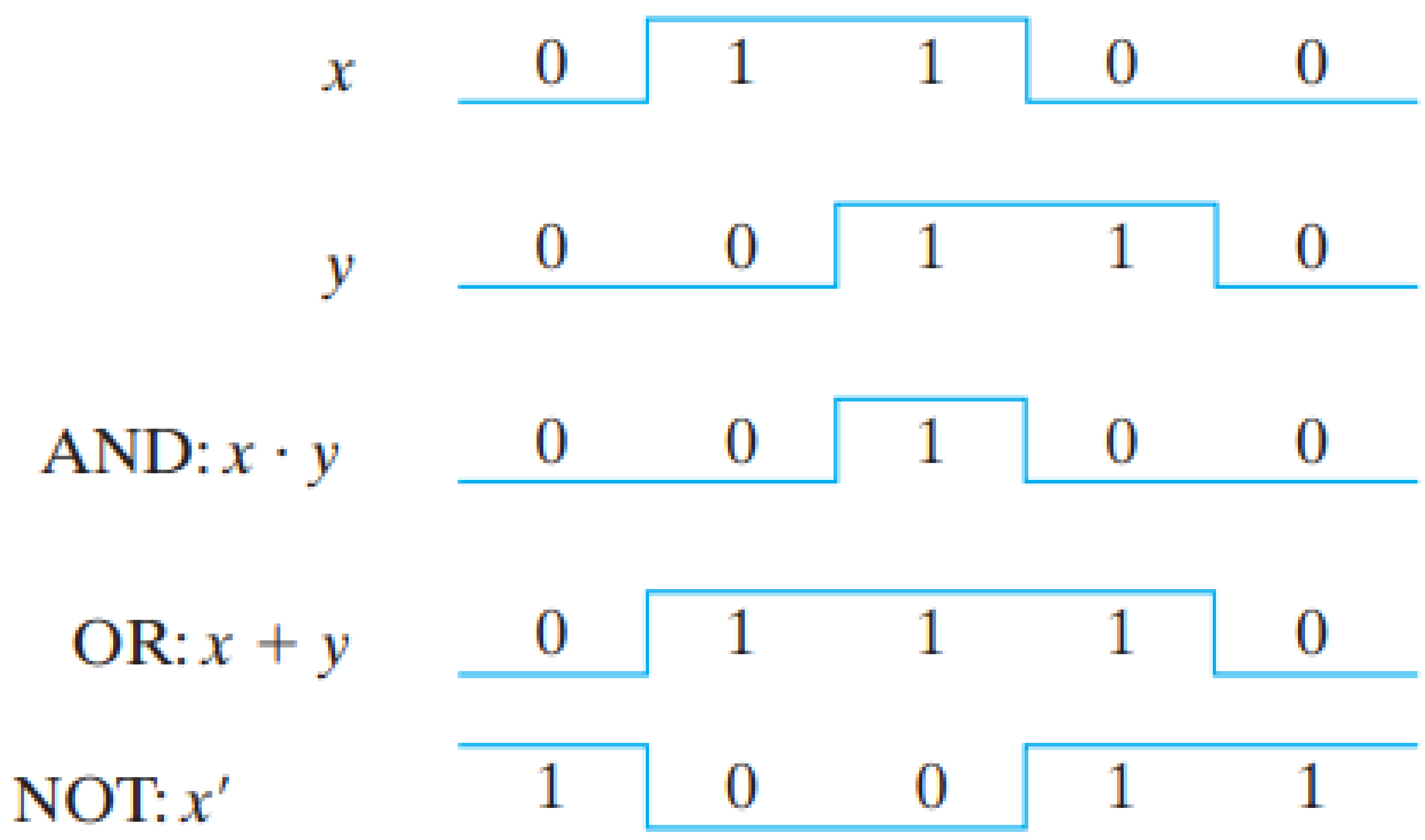
Truth Table

NOT

x	x'
0	1
1	0

Logic Gates

Input – Output signals for gates



Boolean Algebra - Definitions

- Boolean algebra defined with a set of elements, a set of operators, and a number of unproved axioms or postulates.

- A set of elements is any collection of objects, usually having a common property.

  If **S** → Set
  x,y → objects
  x ∈ **S** → x is an element of S
  y ∈ **S** → y is not an element of S

- A = {1, 2, 3, 4} → the elements of set A are the numbers 1, 2, 3, and 4.

Boolean Algebra - Postulates

- ❖ Postulates of a mathematical system form the basic assumptions to deduce the rules, theorems, and properties of the system.
- ❖ The most common postulates used to formulate various algebraic structures are as follows:

Postulate 1: Closure

- A set S is closed with respect to a binary operator if, for every pair of elements of S, the binary operator specifies a rule for obtaining a unique element of S.

- **For example**, the set of natural numbers **N = {1, 2, 3, 4, ...}** is closed with respect to the binary operator + by the rules of arithmetic addition, since, for any **a, b** ∈ **N**, there is a unique **c** ∈ **N** such that **a + b = c**.

Boolean Algebra - Postulates

Postulate 2: Associative law

- A binary operator * on a set S is said to be associative whenever

$$(x * y) * z = x * (y * z) \text{ for all } x, y, z, \in S$$

Postulate 3: Commutative law

- A binary operator * on a set S is said to be commutative whenever

$$x * y = y * x \text{ for all } x, y \in S$$

Boolean Algebra - Postulates

Postulate 4: Identity element

- A binary operation * on S if there exists an element $e \in S$ with the property that

  $e * x = x * e = x$ for every $x \in S$

- E.g., $x + 0 = 0 + x = x$ for any $x \in I$ where $I = \{ c, -3, -2, -1, 0, 1, 2, 3, c\}$,

Postulate 5: Inverse

- a binary operator * is said to have an inverse whenever, for every $x \in S$, there exists an element $y \in S$ such that

  $x * y = e$

- E.g., $x + x' = 1$ and $x \cdot x' = 0$

Boolean Algebra - Postulates

Postulate 6: Distributive law

- If * and · are two binary operators on a set S, * is said to be distributive over . whenever

$$x * (y \cdot z) = (x * y) \cdot (x * z)$$

- A field is a set of elements, together with two binary operators, each having properties 1 through 5 and both operators combining to give property 6.

- The field of real numbers is the basis for arithmetic and ordinary algebra.

Huntington postulates

1. (a) The structure is closed with respect to the operator +.
   (b) The structure is closed with respect to the operator · .

2. (a) The element 0 is an identity element with respect to +; that is, $\mathbf{x + 0 = 0 + x = x}$.
   (b) The element 1 is an identity element with respect to · ; that is, $\mathbf{x \cdot 1 = 1 \cdot x = x}$.

3. (a) The structure is commutative with respect to +; that is, $\mathbf{x + y = y + x}$.
   (b) The structure is commutative with respect to · ; that is, $\mathbf{x \cdot y = y \cdot x}$.

Huntington postulates

4. (a) The operator · is distributive over +; that is, $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$.
   (b) The operator + is distributive over · ; that is, $x + (y \cdot z) = (x + y) \cdot (x + z)$.

5. For every element $x \in B$, there exists an element $x' \in B$ (called the complement of x)
   (a) $x + x' = 1$
   (b) $x \cdot x' = 0$.

6. There exist at least two elements $x, y \in B$ such that $x \neq y$.

Two-Valued Boolean Algebra

- **Two-valued Boolean algebra** is defined on a set of two elements, **B = {0, 1}** with rules for the two binary operators **+** and **·** as shown in the following operator tables.

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

x	y	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

x	x'
0	1
1	0

- These rules are exactly the same as the AND, OR, and NOT operations

Ollscoil na Gaillimhe
University of Galway

Theorems and Properties
Boolean Functions

Theorems and Properties

- The Huntington postulates were listed in pairs and designated by part (a) and part (b).

- One part may be obtained from the other if the binary operators and the identity elements are interchanged which is called dual principle.

- **Duality principle** states that every algebraic expression deducible from the postulates of Boolean algebra remains valid if the operators and identity elements are interchanged.

Theorems and Properties

- In a two-valued Boolean algebra, the identity elements and the elements of the set B are the same: 1 and 0.

- The duality principle has many applications.

- If the dual of an algebraic expression is desired, we simply interchange OR and AND operators and replace 1's by 0's and 0's by 1's.

Theorems and Properties

Postulates and Theorems of Boolean Algebra

Postulate 2	(a)	$x + 0 = x$	(b)	$x \cdot 1 = x$
Postulate 5	(a)	$x + x' = 1$	(b)	$x \cdot x' = 0$
Theorem 1	(a)	$x + x = x$	(b)	$x \cdot x = x$
Theorem 2	(a)	$x + 1 = 1$	(b)	$x \cdot 0 = 0$

Theorems and Properties

Postulates and Theorems of Boolean Algebra

Theorem 3, involution		$(x')' = x$		
Postulate 3, commutative	(a)	$x + y = y + x$	(b)	$xy = yx$
Theorem 4, associative	(a)	$x + (y + z) = (x + y) + z$	(b)	$x(yz) = (xy)z$
Postulate 4, distributive	(a)	$x(y + z) = xy + xz$	(b)	$x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a)	$(x + y)' = x'y'$	(b)	$(xy)' = x' + y'$
Theorem 6, absorption	(a)	$x + xy = x$	(b)	$x(x + y) = x$

Theorems and Properties

- The theorems, like the postulates, are listed in pairs; each relation is the dual of the one paired with it.
- The postulates are basic axioms of the algebraic structure and need no proof.
- The theorems must be proven from the postulates.
- Proofs of the theorems with one variable are presented in *upcoming slides*.
- At the right is listed the number of the postulate which justifies that particular step of the proof.

Theorems and Properties

THEOREM 1(a): x + x = x

Statement	Justification
$x + x = (x + x) \cdot 1$	postulate 2(b)
$= (x + x)(x + x')$	5(a)
$= x + xx'$	4(b)
$= x + 0$	5(b)
$= x$	2(a)

Theorems and Properties

THEOREM 1(b): x · x = x

Statement	Justification
$x \cdot x = xx + 0$	postulate 2(a)
$= xx + xx'$	5(b)
$= x(x + x')$	4(a)
$= x \cdot 1$	5(a)
$= x$	2(b)

OLLSCOIL NA GAILLIMHE
UNIVERSITY OF GALWAY

Note:

- theorem 1(b) is the dual of theorem 1(a) and that each step of the proof in part (b) is the dual of its counterpart in part (a).
- Any dual theorem can be similarly derived from the proof of its corresponding theorem.

Theorems and Properties

THEOREM 2(a): $x + 1 = 1$.

Statement	Justification
$x + 1 = 1 \cdot (x + 1)$	postulate 2(b)
$= (x + x')(x + 1)$	5(a)
$= x + x' \cdot 1$	4(b)
$= x + x'$	2(b)
$= 1$	5(a)

THEOREM 2(b): $x \cdot 0 = 0$ by duality.

Theorems and Properties

THEOREM 3: $(x')' = x$.

- From postulate 5, we have $x + x' = 1$ and $x \cdot x' = 0$, which together define the complement of x. The complement of x' is x and is also (x')'.

- Therefore, since the complement is unique, we have $(x')' = x$.

Theorems and Properties

- The theorems involving two or three variables may be proven algebraically from the postulates and the theorems that have already been proven.

- For example, the absorption theorem:

  - **THEOREM 6(a):** x + xy = x.

Statement	Justification
$x + xy = x \cdot 1 + xy$	postulate 2(b)
$= x(1 + y)$	4(a)
$= x(y + 1)$	3(a)
$= x \cdot 1$	2(a)
$= x$	2(b)

  - **THEOREM 6(b):** x(x + y) = x by duality.

Theorems and Properties

- The theorems of Boolean algebra can be proven by means of truth tables.
- In truth tables, both sides of the relation are checked to see whether they yield identical results for all possible combinations of the variables involved.
- The following truth table verifies the first absorption theorem:

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

Theorems and Properties

The truth table for the first **DeMorgan's theorem**, $(x + y)' = x'y'$, is as follows:

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

x'	y'	$x'y'$
1	1	1
1	0	0
0	1	0
0	0	0

Operator Precedence

- The operator precedence for evaluating Boolean expressions is
  1. parentheses
  2. NOT
  3. AND
  4. OR

- Expressions inside parentheses must be evaluated before all other operations.

- The next operation that holds precedence is the complement, and then follows the AND and, finally, the OR.

Operator Precedence

- For E.g., Truth table for one of DeMorgan's theorems.
- $(x + y)' = x'y'$

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

x'	y'	$x'y'$
1	1	1
1	0	0
0	1	0
0	0	0

- The left side of the expression is $(x + y)'$.
- The expression inside the parentheses is evaluated first i.e., **x+y** and the results then complemented.

Operator Precedence

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

x'	y'	$x'y'$
1	1	1
1	0	0
0	1	0
0	0	0

- The right side of the expression is **x'y'**,
- The **complement of x** and the **complement of y** are both evaluated first, and the result is then **ANDed**.

Note: In ordinary arithmetic, the same precedence holds (except for the complement) when multiplication and addition are replaced by AND and OR, respectively.

Boolean Functions

- **Boolean algebra** is an algebra that deals with binary variables and logic operations.
- A **Boolean function** described by an algebraic expression consists of binary variables, the constants 0 and 1, and the logic operation symbols.
- For a given value of the binary variables, the function can be equal to either **1** or **0**.
- As an example, consider the Boolean function

$$\mathbf{F1 = x + y'z}$$

Boolean Functions

F1 = x + y'z

- The function **F1 is equal to 1** if x is equal to 1 or if both y' and z are equal to 1.
- **F1 is equal to 0** otherwise.
- The complement operation dictates that when **y' = 1, y = 0**.
  Therefore, F1 = 1 if x = 1 or if y = 0 and z = 1.

Boolean Functions

A **Boolean function** expresses the logical relationship between binary variables and is evaluated by determining the binary value of the expression for all possible values of the variables.

- A Boolean function can be represented in a truth table.

- The number of rows in the truth table is $\mathbf{2^n}$,
  where
  n - the number of variables in the function.

- The binary combinations for the truth table are obtained from the binary numbers by counting from **0 through $\mathbf{2^n - 1}$**.

Boolean Functions

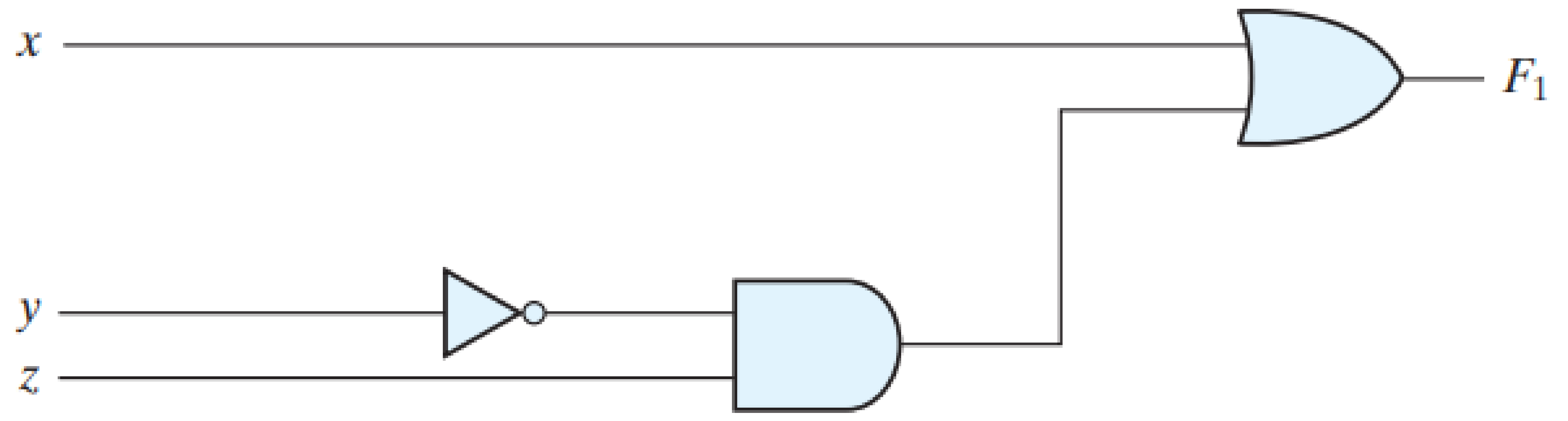
- Consider the truth table for the function F1.
- There are eight possible binary combinations for assigning bits to the three variables x, y, and z.
- The column labeled F1 contains either 0 or 1 for each of these combinations.
- The table shows that the function is
  equal to 1 when $x = 1$ or when $yz = 01$ and is
  equal to 0 otherwise.

Truth Tables for F_1 and F_2

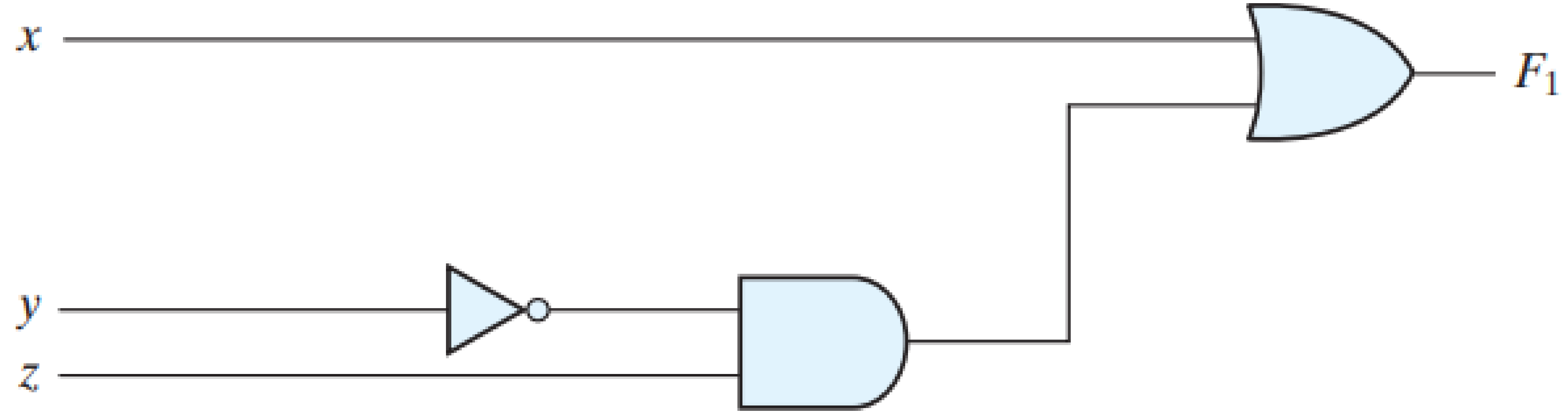
x	y	z	F_1	F_2
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	0	1
1	0	0	1	1
1	0	1	1	1
1	1	0	1	0
1	1	1	1	0

Boolean Functions

- A Boolean function can be transformed from an algebraic expression into a circuit diagram composed of logic gates connected in a particular structure.
- The **logic-circuit diagram** (also called a schematic) for **F1** is shown below.
- There is an inverter for input y to generate its complement.
- There is an AND gate for the term **yz** and an OR gate that combines **x** with **y’z**.



Boolean Functions



- In logic-circuit diagrams, the variables of the function are taken as the inputs of the circuit and the binary variable **F1** is taken as the output of the circuit.
- The schematic expresses the relationship between the output of the circuit and its inputs.
- It indicates how to compute the logic value of each output from the logic values of the inputs.

Boolean Functions

Consider, for example, the following Boolean function:

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

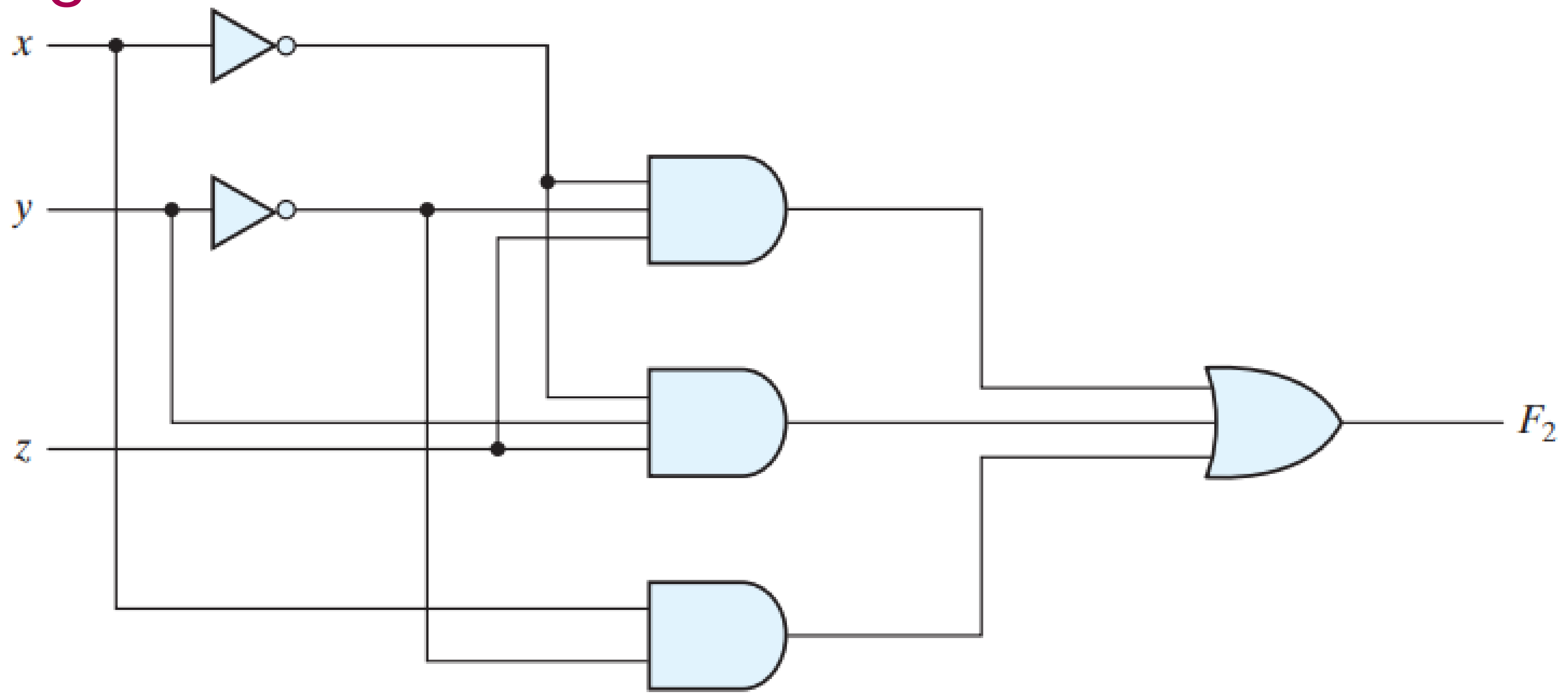
- The truth table for F2 is shown right.
- The function is **equal to 1**
  when xyz = 001 or 011 or
  when xy = 10 (irrespective of the value of z)
- The function is **equal to 0** otherwise.
- This set of conditions produces four 1's and four 0's for F2.

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

Boolean Functions

- A schematic with logic gates of **F2 = x'y'z + x'yz + xy'** is shown below.
- Input variables x and y are complemented with inverters to obtain x' and Y'.
- The three terms in the expression are implemented with three AND gates.
- The OR gate forms the logical OR of the three terms.



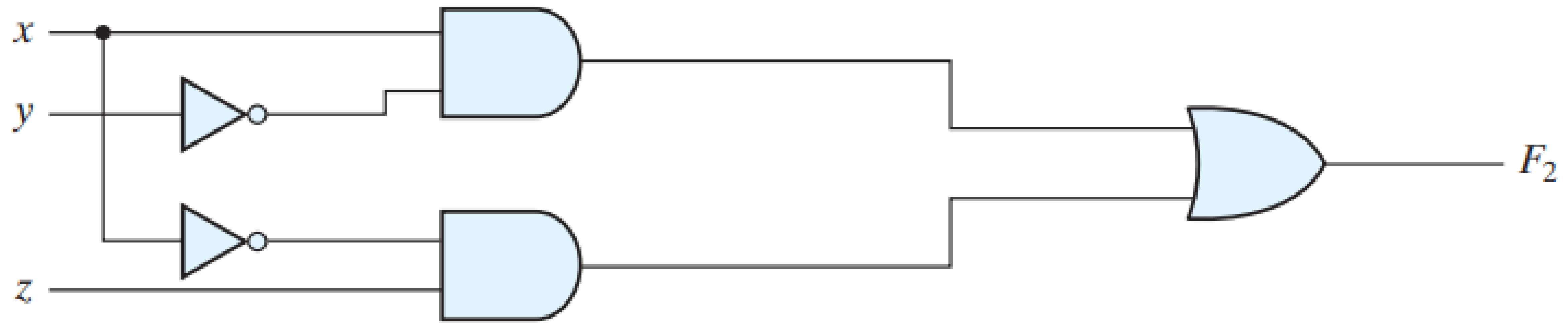
(a) $F_2 = x'y'z + x'yz + xy'$

Boolean Functions

- Now consider the possible simplification of the function by applying some of the identities of Boolean algebra:

$$\begin{aligned} F2 &= x'y'z + x'yz + xy' \\ &= x'z(y' + y) + xy' \\ &= \mathbf{x'z + xy'} \end{aligned}$$

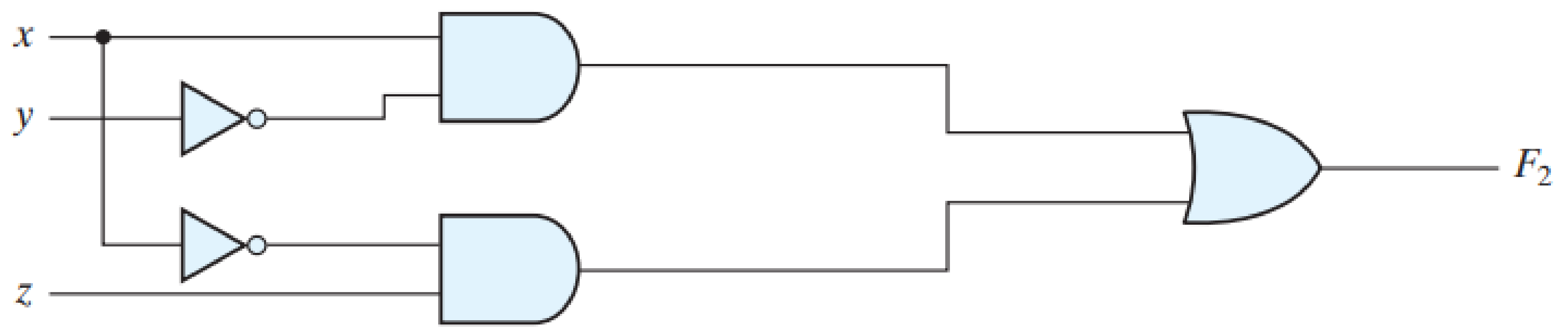
- The function is reduced to only two terms and can be implemented with gates as shown below.



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

Boolean Functions

- The circuit in (b) is simpler than the one in (a), yet both implement the same function.
- By means of a truth table, it is possible to verify that the two expressions are equivalent.
- The simplified expression is **equal to 1** when xz = 01 or when xy = 10.
- This produces the same four 1's in the truth table. Since both expressions produce the same truth table, they are equivalent.



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

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

Boolean Functions

- Therefore, the two circuits have the same outputs for all possible binary combinations of inputs of the three variables.

- Each circuit implements the same identical function, but the one with fewer gates and fewer inputs to gates is preferable because it **requires fewer wires and components**.

- In general,
    - There are many equivalent representations of a logic function.

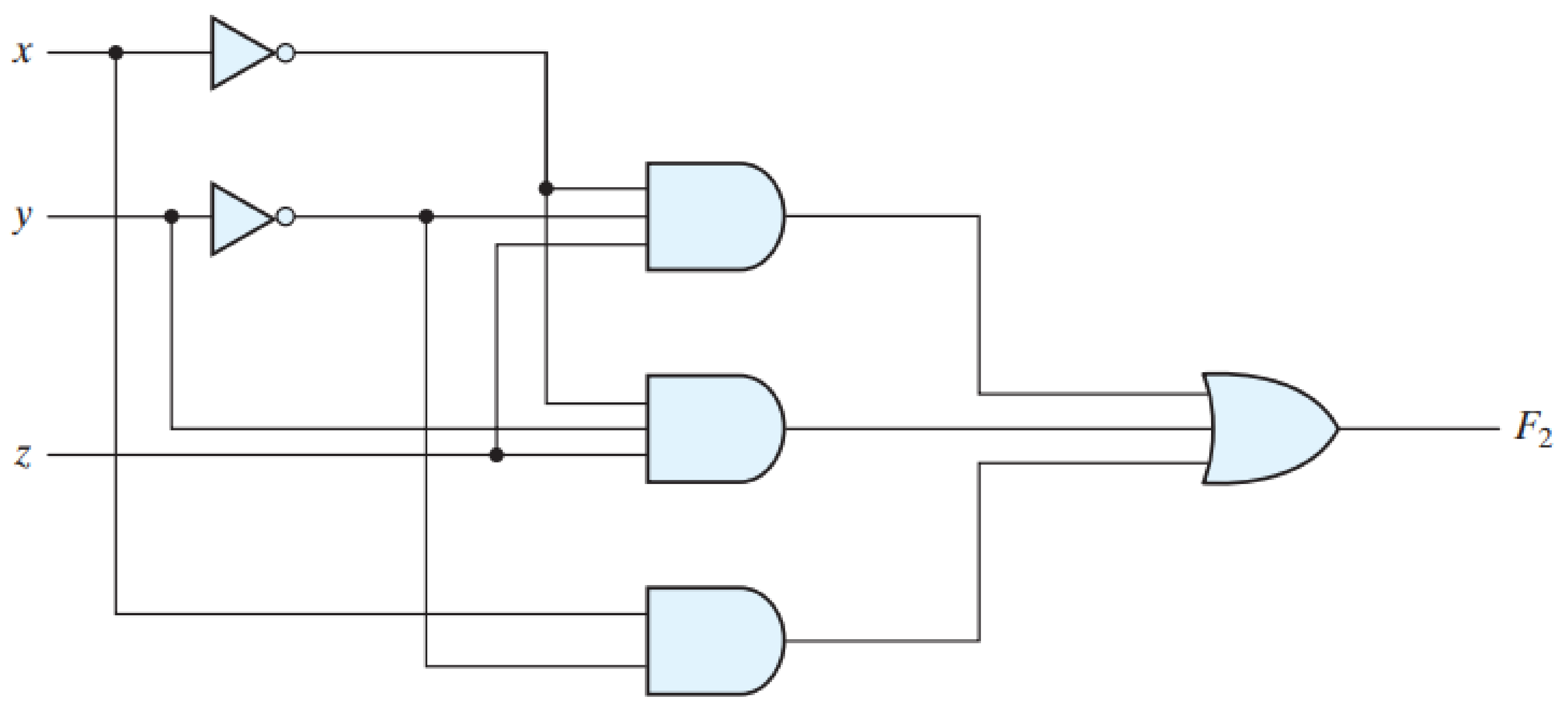
    - Finding the most economic representation of the logic is an important design task.

Algebraic Manipulation

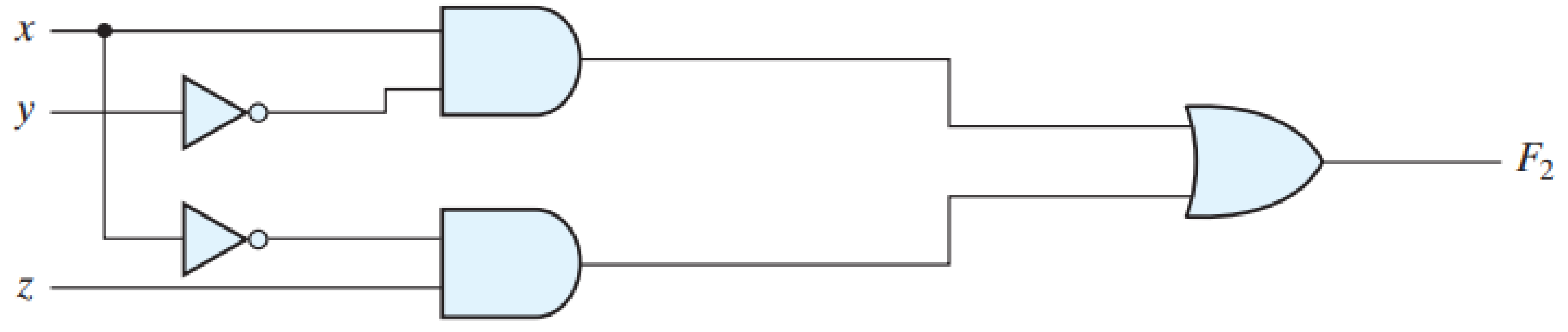
- When a Boolean expression is implemented with logic gates, each term requires a gate and each variable within the term designates an input to the gate.
- We define a literal to be a single variable within a term, in complemented or uncomplemented form.

Algebraic Manipulation

- The function of **Fig. a** has three terms and eight literals, and the one in **Fig. b** has two terms and four literals.



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



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

Algebraic Manipulation

- By reducing the number of terms, the number of literals, or both in a Boolean expression, it is often possible to obtain a simpler circuit.

- The manipulation of Boolean algebra consists mostly of reducing an expression for the purpose of obtaining a simpler circuit.

- For complex Boolean functions and many different outputs, designers of digital circuits use computer minimization programs that are capable of producing optimal circuits with millions of logic gates.

Algebraic Manipulation

- The examples that follow illustrate the algebraic manipulation of Boolean algebra to acquaint the reader with this important design task.

1. $x(x' + y) = xx' + xy = 0 + xy = xy.$

2. $x + x'y = (x + x')(x + y) = 1(x + y) = x + y.$

3. $(x + y)(x + y') = x + xy + xy' + yy' = x(1 + y + y') = x.$

4.
$$\begin{aligned} xy + x'z + yz &= xy + x'z + yz(x + x') \\ &= xy + x'z + xyz + x'yz \\ &= xy(1 + z) + x'z(1 + y) \\ &= xy + x'z. \end{aligned}$$

5. $(x + y)(x' + z)(y + z) = (x + y)(x' + z)$, by duality from function 4.

Algebraic Manipulation

- **Functions 1 and 2** are the dual of each other and use dual expressions in corresponding steps.
- An easier way to simplify **function 3** is by means of postulate 4(b) $(x + y)(x + y') = x + yy' = x$.
- The **4th function** illustrates the fact that an increase in the number of literals sometimes leads to a simpler final expression.
- **Function 5** is not minimized directly but can be derived from the dual of the steps used to derive function 4.
- Functions 4 and 5 are together known as the **consensus theorem**.

Complement of a Function

- The complement of a function **F** is **F'** **i.e.,** interchange of 0's for 1's and 1's for 0's in the value of F.

- The complement of a function may be derived algebraically through DeMorgan's theorems for two variables. $\mathbf{(x + y)' = x'y'}$

- DeMorgan's theorems can be extended to three or more variables.

Complement of a Function

- The three-variable form of the first **DeMorgan's theorem** is derived as follows, from postulates and theorems.

$$
\begin{aligned}
(A + B + C)' &= (A + x)' && \text{let } B + C = x \\
&= A'x' && \text{by theorem 5(a) (DeMorgan)} \\
&= A'(B + C)' && \text{substitute } B + C = x \\
&= A'(B'C') && \text{by theorem 5(a) (DeMorgan)} \\
&= A'B'C' && \text{by theorem 4(b) (associative)}
\end{aligned}
$$

Complement of a Function

- DeMorgan's theorems for any number of variables resemble the two-variable case in form and can be derived by successive substitutions similar to the method used in the preceding derivation.

- These theorems can be generalized as follows:

$$(A + B + C + D + \cdots + F)' = A'B'C'D' \ldots F'$$

$$(ABCD \ldots F)' = A' + B' + C' + D' + \cdots + F'$$

- The generalized form of DeMorgan's theorems states that the complement of a function is obtained by interchanging AND and OR operators and complementing each literal.

References

- Computer Organization and Architecture Designing for Performance Tenth Edition by William Stallings
- Digital Design With an Introduction to the Verilog HDL FIFTH EDITION by M Morris, M. and Michael, D., 2013.

Ollscoil na Gaillimhe
University of Galway

Thank *you*