
Chapter 2: Boolean Algebra and Logic Gates

Prepared by

Abebe Teferi

Chapter 2 - Topics

- **Chapter 2: Boolean Algebra and Logic Gates**
- **2-1 Binary Logic**
- **2-2 Basic Definitions of Boolean Algebra**
- **2-3 Axiomatic Definition of Boolean Algebra**
- **2-4 Basic Theorems and Properties of Boolean Algebra**
- **2-5 Boolean Functions**
- **2-6 Canonical and Standard Forms**
- **2-7 Other Logic Operations**

2-1 Binary Logic

- Binary logic deals with variables that take on two discrete values and with operations that assume logical meaning.
- The two values the variables take may be called by different names (e.g., true and false, yes and no, etc.), but for our purpose, it is convenient to think in terms of bits and assign the values of 1 and 0.
- Binary logic is used to describe, in a mathematical way, the manipulation and processing of binary information.
- It is particularly suited for the analysis and design of digital systems.

Definition of Binary Logic

- The purpose of this section is to introduce Boolean algebra in a heuristic manner and relate it to digital logic circuits and binary signals.
- **Definition of Binary Logic**
- Binary logic consists of binary variables and logical operations.
- The variables are designated by letters of the alphabet such as A, B, C, x, y, z, etc., with each variable having two and only two distinct possible values: 1 and 0.

Cont...

- 1. AND: This operation is represented by a dot or by the absence of an operator.
 - For example, $xy = z$ or $xy = z$ is read "x AND y is equal to z."
 - The logical operation AND is interpreted to mean that $z=1$ if and only if $x=1$ and $y=1$; otherwise $z=0$. (Remember that x , y , and z are binary variables and can be equal either to 1 or 0 and nothing else)
- 2. OR: This operation is represented by a plus sign. For example, $x + y = z$ is read "x OR y is equal to z," meaning that $z=1$ if $x=1$ or if $y=1$ or if both $x=1$ and $y=1$. If both $x=0$ and $y=0$ then $z=0$.

Cont...

- 3. NOT: This operation is represented by a prime (sometimes by a bar). For example, $x' = z$ (or $=z$) is read "not x is equal to z," meaning that z is what x is not.
- In other words, if $x=1$, then $z=0$; but if $x=0$, then $z=1$.
- Binary logic resembles binary arithmetic, and the operations AND and OR have some similarities to multiplication and addition, respectively.
- In fact, the symbols used for AND and OR are the same as those used for

Cont...

- One should realize that an arithmetic variable designates a number that may consist of many digits.
- A logic variable is always either a 1 or 0.
- For example, in binary arithmetic, we have $1 + 1 = 10$ (read: "one plus one is equal to 2"), whereas in binary logic, we have $1 + 1 = 1$ (read: "one OR one is equal to one").
- For each combination of the values x and y , there is a value of z specified by the definition of the logical operation.

Truth Tables

- These definitions may be listed in a compact form using truth tables.
- A truth table is a table of all possible combinations of the variables showing the relation between the values that the variables may take and the result of the operation.
- For example, the truth tables for the operations AND and OR with variables x and y are obtained by listing all possible values that the variables may have when combined in pairs.
- The result of the operation for each combination is then listed in a separate

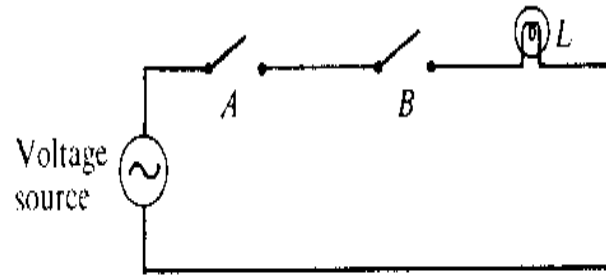
Cont..

TABLE 1-6
Truth Tables of Logical Operations

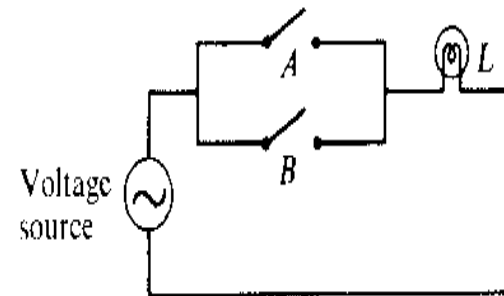
AND		OR		NOT	
x	y	$x \cdot y$	$x + y$	x	x'
0	0	0	0	0	1
0	1	0	1	1	0
1	0	0	1		
1	1	1	1		

Switching Circuits and Binary Signals

- The use of binary variables and the application of binary logic are demonstrated by the simple switching circuit



(a) Switches in series – logic AND



(b) Switches in parallel – logic OR

FIGURE 1-4

Switching circuits that demonstrate binary logic

Cont...

- Let the manual switches A and B represent two binary variables with values equal to 0 when the switch is open and 1 when the switch is closed.
- Similarly, let the lamp L represent a third binary variable equal to 1 when the light is on and 0 when off.
- For the switches in series, the light turns on if A and B are closed.
- For the switches in parallel, the light turns on if A or B is closed.

Cont...

- It is obvious that the two circuits can be expressed by means of binary logic with the AND and OR operations,

respectively,

$$L = A \cdot B \quad \text{for the circuit of Fig. 1-4(a)}$$

$$L = A + B \quad \text{for the circuit of Fig. 1-4(b)}$$

- Electronic digital circuits are sometimes called switching circuits because they behave like a switch, with the active element such as a transistor either conducting (switch closed) or not conducting (switch open).

Cont..

- Instead of changing the switch manually, an electronic switching circuit uses binary signals to control the conduction or non conduction state of the active element.
- Electrical signals such as voltages or current exist throughout a digital system in either one of two recognizable values (except during transition).
- Voltage-operated circuits, for example, respond to two separate voltage levels, which represent a binary variable equal to logic-1 or logic-0.

Cont...

- For example, a particular digital system may define logic-1 as a signal with a nominal value of 3 volts and logic-0 as a signal with a nominal value of 0 volt.
- As shown in Fig 1-5, each voltage level has an acceptable deviation from the nominal.
- The intermediate region between the allowed regions is crossed only during state transitions.
- The input terminals of digital circuits accept binary signals within the allowable tolerances and respond at the output terminal with binary signals that fall within the specified tolerances.

Cont...

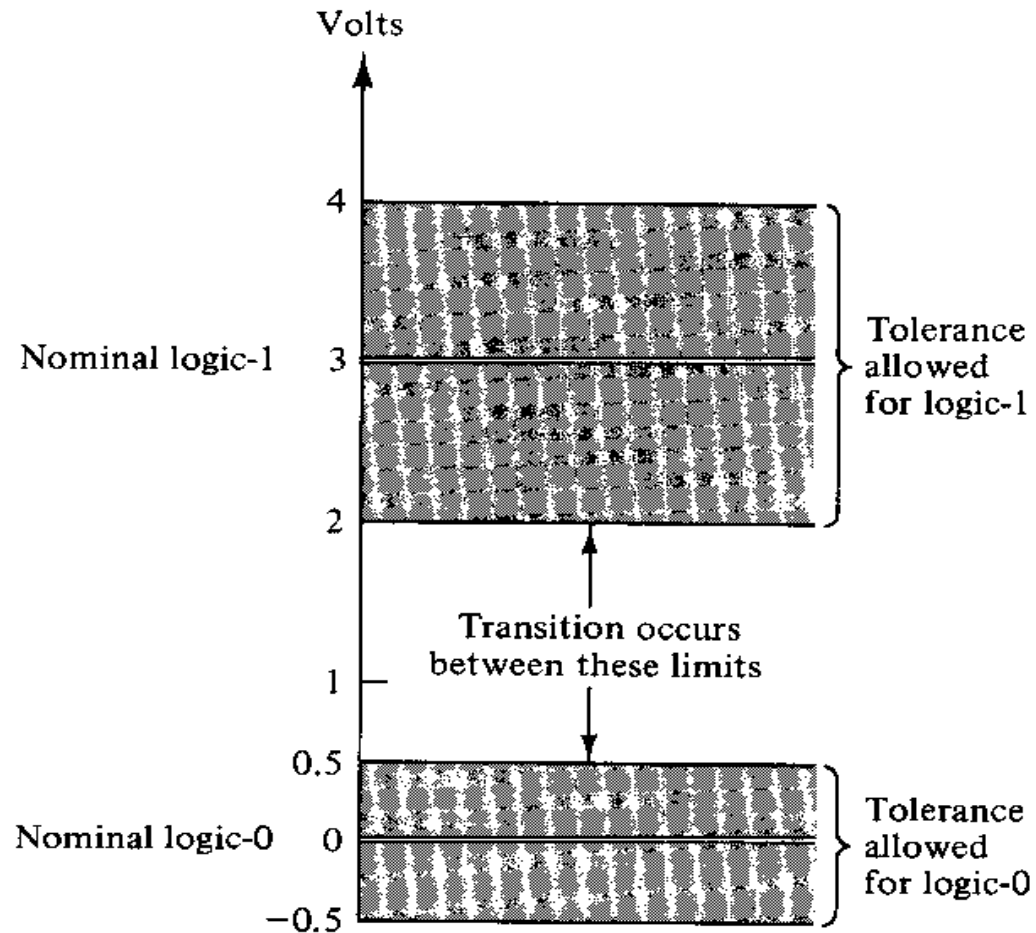


FIGURE 1-5

Example of binary signals

Logic Gates

- Electronics digital circuits are also called logic circuits, with the proper input, they establish logical manipulation paths.
- Any desired information for computing or control can be operated by passing binary signals through various combinations of logic circuits, each signal representing a variable and carrying one bit of information.
- Logic circuits that perform the logical operations of AND, OR, and NOT are shown with their symbols in Fig. 1-6.

Cont...

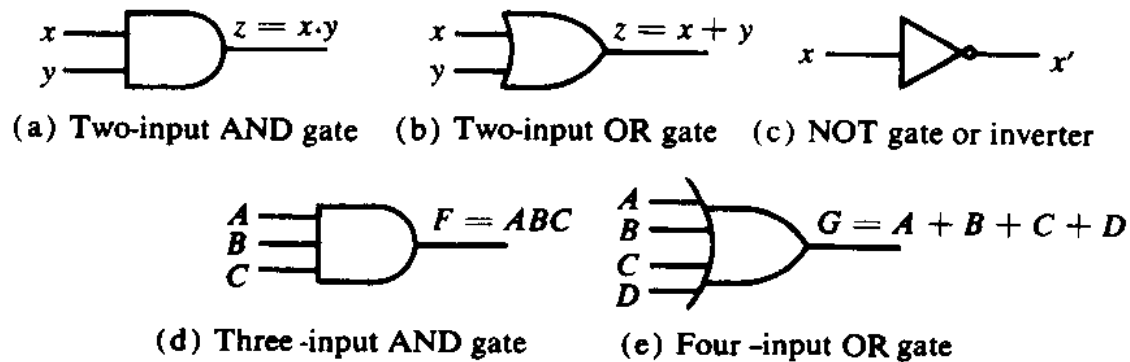


FIGURE 1-6

Symbols for digital logic circuits

Cont...

- These circuits, called gates, are blocks of hardware that produce a logic-1 or logic-0 output signal if input logic requirement are satisfied.
- Note that four different names have been used for the same type of circuits: digital circuits, switching circuits, logic circuits, and gates.
- All four names are widely used, but we shall refer to the circuits as AND, OR, and NOT gates.
- The NOT gate is sometimes called an inverter circuit since it inverts a binary

Cont..

These input signals are shown in Fig. 1-7, together with the output signals for the AND and OR gates.

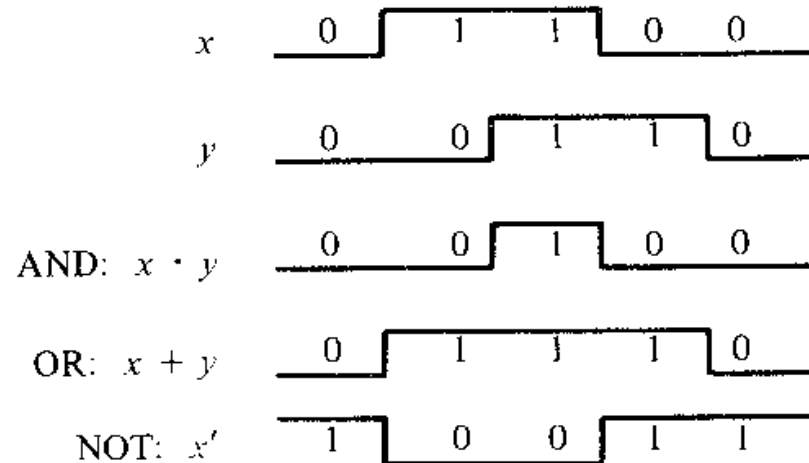


FIGURE 1-7

Input-output signals for gates (a), (b), and (c) of Fig. 1-6

Cont..

- The timing diagrams in Fig. 1-7 illustrate the response of each circuit to each of the four possible input binary combinations.
- AND and OR gates may have more than two inputs.
- An AND gate with three inputs and an OR gate with four inputs are shown in Fig 1-6.
- The three-input AND gate responds with a logic-1 output if all three input signals are logic-1.
- The output produces a logic-0 signal if any input is logic-0.

Cont...

- The four-input OR gate responds with a logic-1 when any input is a logic-1.
- Its output becomes logic-0 if all input signals are logic-0
- The reason for the name "inverter" for the NOT gate is apparent from a comparison of the signal x (input of inverter) and that of x' (output of inverter).
- The mathematical system of binary logic is better known as Boolean, or switching algebra.
- This algebra is conveniently used to describe the operation of complex networks

Cont...

- Designers of digital systems use Boolean algebra to transform circuit diagrams to algebraic expressions and vice versa.

2-2 Basic Definition of Boolean Algebra

Algebra

- Boolean algebra, like any other deductive mathematical system, may be defined with a set of elements, a set of operators, a number of unproved axioms or postulates.
- A set of elements is any collection of objects having a common property.
- If S is a set, and x and y are certain objects, then
- $x \in S$ denotes that x is a member of the set S , and $y \notin S$ denotes that y is not an element of S .
- A set with a denumerable number of elements is specified by braces: $A =$

Cont...

- A binary operator defined on a set S of elements is a rule that assigns to each pair of elements from S a unique element from S .
- As an example, consider the relation $a*b=c$.
- We say that $*$ is a binary operator if it specifies a rule for finding c from the pair (a, b) and also if $a, b, c \in S$.
- However, $*$ is not a binary operator if $a, b \in S$, where the rule finds $c \notin S$.

Cont...

- The postulates of a mathematical system form the basic assumptions from which it is possible to deduce the rules, theorems, and properties of the system.
- The most common postulates used to formulate various algebraic structures are:
- **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 $N = \{1, 2, 3, \dots\}$ is closed with respect to the binary operator plus (+) by the rules of

Cont...

- The set of natural numbers is not closed with respect to the binary operator minus (-) since for instance $2-3 = -1$ and $-1 \notin \mathbb{N}$.
- **2. Associative law.** A binary operator $*$ on a set S is said to be associative whenever
 - $(x*y)*z = x*(y*z)$ for all $x, y, z \in S$.
- **3. Commutative law.** A binary operator $*$ on a set S is said to be commutative whenever
 - $x*y = y*x$ for all $x, y \in S$

Cont...

- **4. Identity element.** A set S is said to have an identity element with respect to a binary operation $*$ on S if there exists element $e \in S$ with the property
 - $e * x = x * e = x$ for every $x \in S$
 - Example: The element 0 is an identity element with respect to operation $+$ on the set of integers $I = \{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$ since
 - $x + 0 = 0 + x = x$ for any $x \in I$

Cont...

- **6. Distributive law.** If $*$ and $\cdot$ are two binary operators on a set S , $*$ is said to be distributive over $\cdot$ whenever
 - $$x*(y \cdot z) = (x*y) \cdot (x*z)$$
 - An example of an algebraic structure is a field.
 - A field is a set of elements, together with two binary operators, each having properties 1 to 5 and both operators combined to give property 6.
 - The set of real number together with the binary operators $+$ and $\cdot$ is the field of real numbers and is the basis for arithmetic and ordinary algebra.

Cont...

- The operators and postulates have the following meanings:
- The binary operator $+$ defines addition.
- The additive identity is 0.
- The additive inverse defines subtraction.
- The binary operator $*$ defines multiplication.
- The multiplicative identity is 1.
- The multiplicative inverse of $a = 1/a$, defines division, i.e., $a \cdot 1/a = 1$.
- The only distributive law applicable is that of $\cdot$ over $+$:

2-3 Axiomatic Definition of Boolean Algebra

2-3 Axiomatic Definition of Boolean Algebra

- In 1854 George Boole introduced a systematic treatment of logic and developed for this purpose an algebraic system now called Boolean algebra.
- In 1938 C.E. Shannon introduced a two-valued Boolean algebra called switching algebra, in which he demonstrated that the properties of bistable electrical switching circuits can be represented by this algebra.
- For the formula definition of Boolean, we shall employ the postulates formulated by E.V. Huntington in 1904.
- Boolean algebra is an algebraic structure

Cont...

- 1. (a) Closure with respect to the operator $+$.
- (b) Closure with respect to the operator $\bullet$.
- 2. (a) An identity element with respect to $+$, designated by 0 : $x+0 = 0 + x = x$.
- (b) An identity element with respect to $\bullet$, designated by 1 : $x\bullet 1 = 1\bullet x = x$.
- (3) (a) Commutative with respect to $+$: $x + y = y + x$.
- (b) Commutative with respect to: $x\bullet y = y\bullet x$.

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

Cont...

- 5. For every element $x \in B$; there exists an element $x' \in B$ (called the complement of x) such that
 - (a) $x + x' = 1$ and
 - (b) $x \cdot x' = 0$.
- 6. There exists at least two elements $x, y \in B$ such that $x \neq y$.

Cont...

- Comparing Boolean algebra with arithmetic and ordinary algebra (the field of real numbers), we note the following differences:
- 1. Huntington postulates do not include the associative law. However, this law holds for Boolean algebra and can be derived (for both operators) from the other postulates.
- 2. The distributive law of $+$ over $\cdot$, ie., $x + (y \cdot z) = (x + y) \cdot (x + z)$, is valid for Boolean algebra, but not for ordinary algebra.
- 3. Boolean algebra does not have additive and multiplicative inverses; therefore, there is no subtraction or division operations.

Cont...

- 4. Postulate 5 defines an operator called complement that is not available in ordinary algebra.
- 5. Ordinary algebra deals with the real numbers, which constitute an infinite set of elements.
- Boolean algebra deals with as yet undefined set of elements, but in the two valued Boolean algebra defined below, B is defined as a set with only two elements, 0 and 1.
- Boolean algebra resembles ordinary algebra in some respects.

Cont...

- The choice of symbols $+$ and $\cdot$ is intentional to facilitate Boolean algebraic manipulations by persons already familiar with ordinary algebra.
- Although one can use some knowledge from ordinary algebra to deal with Boolean algebra, the beginner must be careful not to substitute the rules of ordinary algebra where they are not applicable.
- It is important to distinguish between the elements of the set of an algebraic structure and the variables of an algebraic system.

Cont...

- Similarly, in Boolean algebra, one defines the elements of the set B , and variables such as x , y , z are merely symbols that represent the elements.
- At this point, it is important to realize that in order to have Boolean algebra, one must show:
 - 1. the elements of the set B .
 - 2. The rules of operation for the two binary operators, and
 - 3. that the set of elements B , together with the two operators, satisfies the six Huntington postulates.

Cont...

- One can formulate many Boolean algebras, depending on the choice of the elements of B and the rules of operation.
- In our subsequent work, we deal only with a two-valued Boolean algebra, i.e., one with only two elements.
- Two-valued Boolean algebra has applications in set theory (the algebra of classes) and in propositional logic.
- Our interest here is with the application of Boolean algebra to gate-type circuits.

Two valued Boolean Algebra

- A two value Boolean algebra is defined on a set of two elements, $B = \{0, 1\}$, with rules for the two binary operators $+$ and $\cdot$ 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, respectively.

Cont...

- We must now show that the Huntington postulates are valid for the set $B=\{0,1\}$ and the two binary operators defined before.
- 1. Closure is obvious from the tables since the result of each operation is either 1 or 0 and $1,0 \in B$.
- 2. From the tables we see that
- (a) $0+1=1+0=1$, $0+0=0$
- (b) $1 \cdot 1 = 1$ $1 \cdot 0 = 0 \cdot 1 = 0$
- which established that the two identity elements 0 for $+$ and 1 for $\cdot$ as defined

Cont...

- 3. The commutative laws also hold:

$$x + y = y + x \qquad x \cdot y = y \cdot x$$
- 4. (a) The distributive law $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$ can be shown to hold true from the operator tables by forming a truth table of all possible values of x , y , and z .
- For each combination, we derive $x \cdot (y + z)$ and show that the value is the same as $(x \cdot y) + (x \cdot z)$

(x

x	y	z	$y + z$	$x \cdot (y + z)$	$x \cdot y$	$x \cdot z$	$(x \cdot y) + (x \cdot z)$
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

Cont...

- (b) The distributive law of $+$ over $\cdot$, $x + (y \cdot z) = (x + y) \cdot (x + z)$, can be shown to hold true by means of a truth table similar to the one above.
- 5. From the complement table it is easily shown that
 - (a) $x + x' = 1$, since $0 + 0' = 0 + 1 = 1$ and $1 + 1' = 1 + 0 = 1$.
 - (b) $x \cdot x' = 0$, since $0 \cdot 0' = 0 \cdot 1 = 0$ and $1 \cdot 1' = 1 \cdot 0 = 0$, which verifies postulate 5.
- 6. Postulate 6 is satisfied because the two-valued Boolean algebra has two distinct elements 1 and 0 with $1 \neq 0$.

2-4 Basic Theorems and Properties of Boolean Algebra

2-4 Basic Theorem and Principles of Boolean Algebra

- The Huntington postulates have been 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.
- This important property of Boolean algebra is called the **duality principle**.
- It states that every algebraic expression deducible from the postulates of Boolean algebra remains valid if the operators and identity elements are interchanged.

Basic Theorems

- 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.
- **Basic Theorems**
- Table 2-1 lists six theorems of Boolean algebra and four of its postulates.
- The notation is simplified by omitting the •

Cont...

- **you are advised to become familiar with them as soon as possible.**
- The postulates are basic axioms of the algebraic structure and need no proof.
- The theorems must be proven from the postulates.
- The proofs of the theorems with one variable are presented below.
- At the right is listed the number of the postulate that justifies each step of the proof.

Cont...

TABLE 2-1
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) $x \cdot 0 = 0$
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) $x(x + y) = x$

Cont...

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

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

THEOREM 1(b): $x \cdot x = x$.

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

Cont...

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

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

$x + 1 = 1 \cdot (x + 1)$	by 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.

Cont...

- **Theorem 3:** $(x')' = x$. From postulate 5, we have $x + x' = 1$ and $x \cdot x' = 0$, which defines the complement of x .
- The complement of x' is x and is also $(x')'$.
- Therefore, since the complement is unique, we have that $(x')' = x$.
- The theorems involving two or three variables may be proven algebraically from the postulates and the theorems that have already been proven.

Cont...

- Take, for example, the absorption theorem.

THEOREM 6(a): $x + xy = x$.

$$\begin{aligned}x + xy &= x \cdot 1 + xy && \text{by postulate:} && 2(b) \\&= x(1 + y) && && 4(a) \\&= x(y + 1) && && 3(a) \\&= x \cdot 1 && && 2(a) \\&= x && && 2(b)\end{aligned}$$

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

Cont...

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

Cont..

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

- The algebraic proofs of the associative law and Demorgan's theorem are long and will not be shown here. However, their validity is easily shown with truth tables.
- For example, the truth table for the first DeMorgan's theorem $(x + y)' = x'y'$ is shown

Operator Precedence

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

- **Operator Precedence**
- The operator precedence for evaluating Boolean expressions is (1) parentheses, (2) NOT, (3) AND, and (4) OR.
- As an example, consider the truth table for DeMorgan's theorem.

Cont...

- The left side of the expression is $(x + y)'$. Therefore, the expression inside the parentheses is evaluated first and the result then complemented.
- The right side of the expression is $x'y'$.
- Therefore, the complement of x and the complement of y are both evaluated first and the result is then ANDed.
- Note that in ordinary arithmetic, the same precedence holds (except for the complement) when multiplication and addition are replaced by AND and OR, respectively.

Venn Diagram

- **Venn Diagram**
- A helpful illustration that may be used to visualize the relationships among the variables of a Boolean expression is the Venn Diagram. This diagram consists of a rectangle such as shown in Fig 2-1, inside of which are two circles, one for each variable.

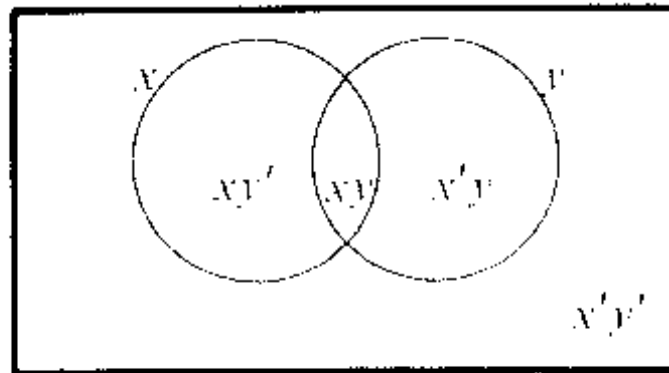


FIGURE 2-1

Venn diagram for two variables

Cont...

- Each circle is labeled by a variable.
- We designate all points inside a circle as belonging to the name variable and all points outside a circle as not belonging to the variable.
- Take, for example, the circle labeled x . If we are inside the circle, we say that $x=1$; when outside, we say $x=0$.
- Now, with two overlapping circles, there are four distinct areas inside the rectangle; the area not belonging to either x or y ($x'y'$), the area inside circle y but outside x ($x'y$), the area inside circle x but outside y (xy'),

Cont...

- Venn diagrams may be used to illustrate the postulates of Boolean algebra or to show the validity of theorems. Figure 2-2, for example, illustrates that the area belonging to xy is inside the circle x and t

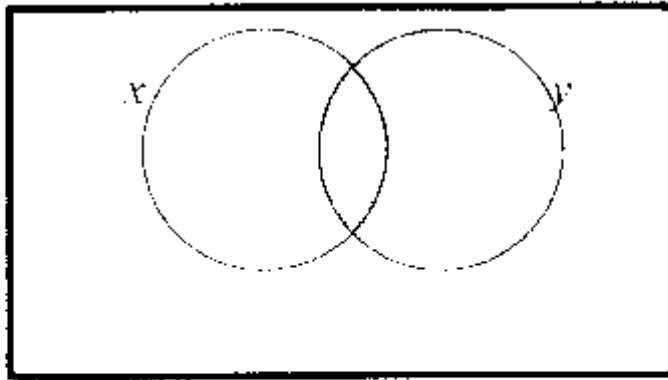


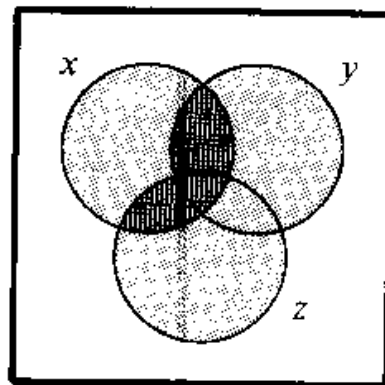
FIGURE 2-2

Venn diagram illustration $x = xy + x̄y$

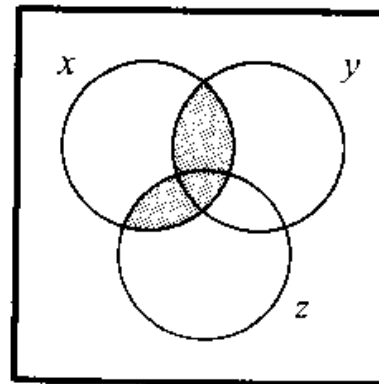
Cont...

- Figure 2-3 illustrates the distributive law $x(y + z) = xy + xz$.

Section 2-4 Boolean Functio



$x(y + z)$



$xy + xz$

FIGURE 2-3

Venn diagram illustration of the distributive law

Cont...

- In this diagram, we have three overlapping circles, one for each of the variables x , y , and z .
- It is possible to distinguish eight distinct areas in a three-variable Venn diagram.
- For this particular example, the distributive law is demonstrated by noting that the area intersecting the circle x with the area enclosing y or z is the same area belonging to xy or xz .

2-5 Boolean Functions

2-5 Boolean Functions

- A binary variable can take the value of 0 or 1.
- A Boolean function is an expression formed with binary variables, the two binary operators OR and AND, and unary operator NOT, parentheses, and an equal sign.
- For a given value of the variables, the function can be either 0 or 1.
- Consider, for example, the Boolean function
- $$F_1 = xyz'$$
- The function F_1 is equal to 1 if $x=1$ and $y=1$ and $z'=1$; otherwise $F_1=0$.

Cont...

- The above is an example of a Boolean function represented as an algebraic expression.
- A Boolean function may also be represented in a truth table.
- To represent a function in a truth table, we need a list of the 2^n combinations of 1's and 0's of the n binary variables, and a column showing the combinations for which the function is equal to 1 or 0.
- As shown in Table 2-2, there are eight possible distinct combinations for assigning bits to three variables.

Cont...

- The column labeled F_1 contains either a 0 or a 1 for each of these combinations.
- The table shows that the function F_1 is equal to 1 only when $x=1$, $y=1$, and $z=0$.
- It is equal to 0 otherwise. (Note that the statement $z'=1$ is equivalent to saying that $z=0$).

Cont...

TABLE 2-2

**Truth Tables for $F_1 = xyz'$, $F_2 = x + y'z$,
 $F_3 = x'y'z + x'yz + xy'$, and $F_4 = xy' + x'z$**

x	y	z	F_1	F_2	F_3	F_4
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	0

Cont...

- Consider now the function
- $F_2 = x + y'z$
- $F_2=1$ if $x=1$ or if $y=0$, while $z=1$.
- In Table 2-2, $x=1$ in the last four rows and $yz=01$ in rows 001 and 101.
- The latter combination applies also for $x=1$.
- Therefore, there are five combinations that make $F_2=1$.
- As a third example, consider the function
- $F_3 = x'y'z + x'yz + xy'$

Cont...

- This is shown in Table 2-2 with four 1's and four 0's.
- $F_4 = xy' + x'z$
- F_4 is the same as F_3 . This means that it is possible to have two or more algebraic equations that have the same output.
- F_4 is a simplified version of F_3 and can be implemented with fewer gates than F_3 .
- Therefore, F_4 is the preferable algebraic expression.
- And now we will explore how to simplify Boolean expressions.

Algebraic Manipulation

- A literal is a primed or unprimed variable.
- When a Boolean function is implemented with logic gates, each literal in the function designates an input to a gate, and each term is implemented with a gate.
- The minimization of the number of literals and the number of terms results in a circuit with less equipment.

Cont...

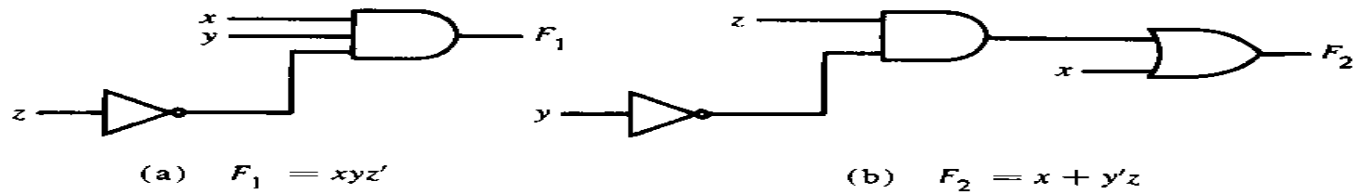
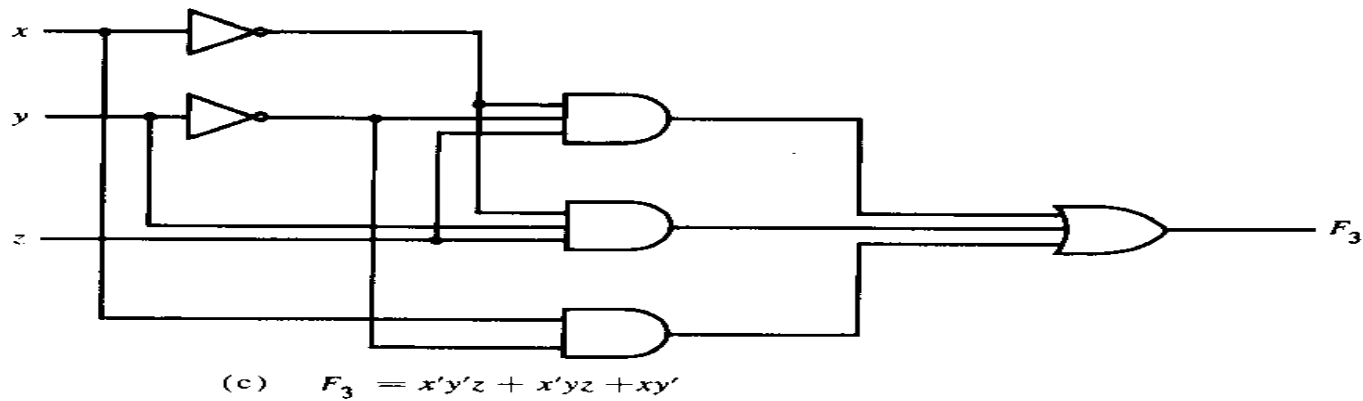
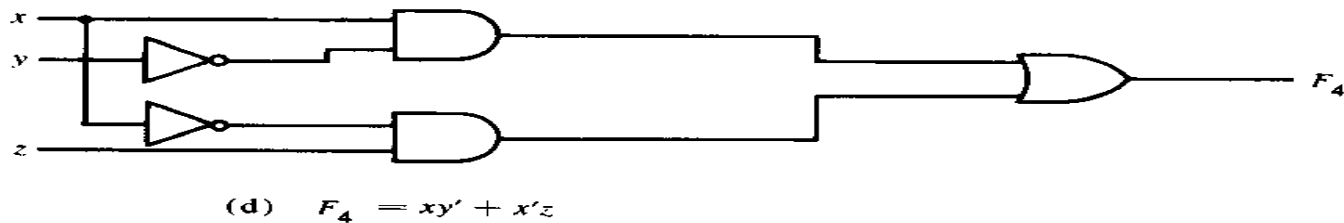


FIGURE 2-4

Implementation of Boolean functions with gates

Cont...

- It is not always possible to minimize both simultaneously; usually, further criteria must be available.
- At the moment, we shall narrow the minimization criteria to literal minimization.
- We shall discuss other criteria in the next Chapter.
- The number of literals in a Boolean function can be minimized by algebraic manipulation.
- There are no specific rules to follow that will guarantee the final answer.
- The only method available is a cut-and-try

Cont...

- The following examples illustrates this procedure.

- **Example 2-1** Simplify the following

Boo

1. $x + x'y = (x + x')(x + y) = 1 \cdot (x + y) = x + y$

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

3. $x'y'z + x'yz + xy' = x'z(y' + y) + xy' = x'z + xy'$

4. $xy + x'z + yz = xy + x'z + yz(x + x')$
 $= xy + x'z + xyz + x'yz$
 $= xy(1 + z) + x'z(1 + y)$
 $= xy + x'z$

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

Cont...

- Functions 1 and 2 are the duals of each other and use dual expressions in corresponding steps.
- Function 3 shows the equality of the functions F3 and F4 discussed previously.
- The fourth illustrates the fact that an increase in the number of literals sometimes leads to a final simpler expression.
- Function 5 is not minimized directly but can be derived from the dual of the steps used to derive function 4.

Complement of a Function

- The complement of a function F is F' and is obtained from an 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 theorem.
- This pair of theorems is listed in Table 2-1 for two variables.
- DeMorgan's theorems can be extended to three or more variables.
- The three-variables form of the first DeMorgan's theorem is derived below.

Cont...

- The postulates and theorems are those listed in Table 2-1.
- $(A + B + C)' = (A + X)'$ let $B + C = X$
- $= A'X'$ by theorem 5(a)
(DeMorgan)
- $= A' \cdot (B + C)'$ substitute $B + C = X$
- $= A' \cdot (B'C')$ by theorem 5(a)
(DeMorgan)
- $= A'B'C'$ by theorem 4(b)
(associative)
- DeMorgan's theorems for any number of variables resemble in form of the two variable case and can be derived by

Cont...

- These theorems can be generalized as follows:
- $(A + B + C + D + \dots + F)' = A'B'C'D' \dots F'$
- $(ABCD \dots F)' = A' + B' + C' + D' + \dots + F'$
- The generalized form of DeMorgan's theorem states that the complement of a function is obtained by interchanging AND and OR operators and complementing each literal.
- Example 2-2 Find the complement of the functions $F_1 = x'yz' + x'y'z$ and $F_2 = x(y'z' + yz)$. By applying DeMorgan's theorem as many times as necessary the

cont...

$$F_1' = (x'yz' + x'y'z)' = (x'yz')'(x'y'z)' = (x + y' + z)(x + y + z')$$

$$\begin{aligned} F_2' &= [x(y'z' + yz)]' = x' + (y'z' + yz)' = x' + (y'z')' \cdot (yz)' \\ &= x' + (y + z)(y' + z') \end{aligned}$$

- A simple procedure for deriving the complement of a function is to take the dual of the function and complement each literal.
- This method follows from the generalized DeMorgan's theorem.
- Remember that the dual of a function is obtained from the interchange of AND and OR operators and 1's and 0's.

Cont...

- Example 2-3 Find the complement of the functions F_1 and F_2 of Example 2-2 by taking their duals and complementing each

1. $F_1 = x'yz' + x'y'z.$

The dual of F_1 is $(x' + y + z')(x' + y' + z).$

Complement each literal: $(x + y' + z)(x + y + z') = F'_1.$

2. $F_2 = x(y'z' + yz).$

The dual of F_2 is $x + (y' + z')(y + z).$

Complement each literal: $x' + (y + z)(y' + z') = F'_2.$

Exercises

- Simplify the following functions.
- 1. $F = x + x'y$
- 2. $F = x'y'z + x'yz + xy$
- 3. $F = x'y' + xy + x'y$
- Find the simplified version of the complement of the following functions.
- 4. $F = xy + (x' + y)$
- 5. $F = xy + x'y'$

2-6 Canonical and Standard Forms

Forms

- **Minterms and Maxterms**
- A binary variable may appear either in its normal form (x) or in its complement form (x').
- Now consider two binary variables x and y combined with an AND operation.
- Since each variable may appear in either form, there are four possible combinations: xy' , $x'y$, xy , and $x'y'$.
- Each of these four AND terms represents one of the distinct area in the Venn diagram of Fig 2-1 and is called a **minterm**, or **standard product**.

Cont...

- The 2^n different minterms may be determined by a method similar to the one shown in Table 2-3 for three variables.
- The binary numbers from 0 to 2^n-1 are

lis

TABLE 2-3
Minterms and Maxterms for Three Binary Variables

<i>x</i>	<i>y</i>	<i>z</i>	Minterms		Maxterms	
			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

Cont...

- Each minterm is obtained from an AND term of the n variables, with each variable being primed if the corresponding bit of the binary number is 0 and unprimed if a 1.
- A symbol for each minterm is also shown in the table and is of the form m_j , where j denotes the decimal equivalent of the binary number of the minterm designated.
- In a similar fashion, n variables forming an OR term, with each variable being primed or unprimed, provide 2^n possible combinations, called **maxterms**, or **standard sums**.

Cont...

- The eight maxterms for three variables, together with their symbolic designation, are listed in Table 2-3.
- Any 2^n maxterms for n variables may be determined similarly.
- Each maxterm is obtained from an OR term of the n variables, with each variable being unprimed if the corresponding bit is a 0 and primed if a 1.
- Note that each maxterm is the complement of its corresponding minterm, and vice versa.
- A Boolean function may be expressed algebraically from a given truth table by

Cont...

- For example, let us look us Table 2-4.

TABLE 2-4
Functions of Three Variables

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

Cont...

- The function f_1 in Table 2-4 is determined by expressing the combinations 001, 100, and 111 as $x'y'z$, $xy'z'$, and xyz , respectively.
- Since each one of these minterms results in $f_1=1$, we should have
- $f_1 = x'y'z + xy'z' + xyz = m_1 + m_4 + m_7$
- Similarly, it may be easily verified that
- $f_2 = x'yz + xy'z + xyz' + xyz = m_3 + m_4 + m_6 + m_7$
- These examples demonstrate an important property of Boolean algebra: Any Boolean

Cont...

- Now consider the complement of a Boolean function.
- It may be read from the truth table by forming a minterm for each combination that produces a 0 in the function and then ORing those terms.
-
- The complement of f_1 is read as
- $f_1' = x'y'z' + x'yz' + x'yz + xy'x + xyz'$
- If we take the complement of f_1' , we obtain the function f_1 :
- $f_1 = (x + y + z)(x + y' + z)(x + y' + z')(x' + y' + z)$

Cont..

- $f_2 = (x + y + z)(x + y + z')(x + y' + z)(x' + y + z)$
- $= M_0 M_1 M_2 M_4$
- These examples demonstrate a second important property of Boolean algebra: Any Boolean function can be expressed as a product of maxterms (by "product" is meant the ANDing of terms).
- The procedure for obtaining the product of maxterms directly from the truth table is as follows.
- Form a maxterm for each combination of the variables that produces a 0 in the function, and then form the AND of all

Sum of Minterms

- It was previously stated that for n binary variables, one can obtain 2^n distinct minterms, and that any Boolean function can be expressed as a sum of minterms.
- The minterms whose sum defines the Boolean functions are those that give the 1's of the function in a truth table.
- It is sometimes convenient to express the Boolean function in its sum of minterms form.
- If not in this form, it can be made so by first expanding the expression into a sum of AND terms.

Cont...

- If it misses one or more variables, it is ANDed with an expression such as $x + x'$, where x is one of the missing variables.
- The following example clarifies this procedure.
- Example 2-4
- Express the Boolean function $F = A + B'C$ in a sum of minterms.
- The function has three variables, A , B , and C . The first term A is missing two variables; therefore:
 - $A = A(B + B') = AB + AB'$
 - This is still missing one variable:

Cont..

$$\begin{aligned} A &= AB(C + C') + AB'(C + C') \\ &= ABC + ABC' + AB'C + AB'C' \end{aligned}$$

- The second term $B'C$ is missing one var' ' '
$$B'C = B'C(A + A') = AB'C + A'B'C$$

- Combining all terms we have
$$\begin{aligned} F &= A + B'C \\ &= ABC + ABC' + AB'C + AB'C' + AB'C + A'B'C \end{aligned}$$

Cont...

- But $AB'C$ appear twice, and according to theorem 1 ($x + x = x$), it is possible to remove one of them.

- Rearranging the minterms in ascending

$$\begin{aligned} 0 | \quad F &= A'B'C + AB'C' + \overline{A}B'C + ABC' + ABC \\ &= m_1 + m_4 + m_5 + m_6 + m_7 \end{aligned}$$

- It is sometimes convenient to express the Boolean function. when in its sum of minter $F(A, B, C) = \Sigma(1, 4, 5, 6, 7)$ notation:

Cont...

- The summation symbol Σ stands for the ORing of terms: the numbers following it are the minterms of the function.
- The letters in parentheses following F form a list of the variables in the order taken when the minterm is converted to an AND term.
- An alternate procedure for deriving the minterms of a Boolean function is to obtain the truth table of the function directly from the algebraic expression and then read the minterms from the truth table.
- Consider the Boolean function given in

Cont..

- The truth table shown in Table 2-5 can be derived directly from the algebraic expression by listing the eight binary combinations under variables A, B, and C

ar

CC

TABLE 2-5

Truth Table for $F = A + B'C$

<i>A</i>	<i>B</i>	<i>C</i>	<i>F</i>
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 Maxterms

- From the truth table, we can then read the five minterms of the function to be 1,4,5,6 and 7.
- **Product of Maxterms**
- To express the Boolean function as a product of maxterms, it must first be brought into a form of OR terms.
- This may be done by using the distributive law,
- $x + yz = (x + y)(x + z)$.
- Then any missing variable x in each OR term is ORed with xx' .

- This procedure is clarified by the following

Cont...

- Example 2-5
- Express the Boolean function $F = xy + x'z$ in a product of maxterm form. First, convert the function into OR terms using the distributive law:
- $F = xy + x'z = (xy + x')(xy + z)$
- $= (x + x')(y + x')(x + z)(y + z)$
- $= (x' + y)(x + z)(y + z)$
- The function has three variables: x , y , and z . Each OR term is missing one variable;

Cont...

- therefore:

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

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

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

- Combining all the terms and removing those that appear more than once, we finally obtain:

$$\begin{aligned} F &= (x + y + z)(x + y' + z)(x' + y + z)(x' + y + z') \\ &= M_0 M_2 M_4 M_5 \end{aligned}$$

- A convenient way to express this function is as follow $F(x, y, z) = \Pi(0, 2, 4, 5)$

Forms

- The product symbol, $\prod$, denotes the ANDing of maxterms; the numbers are the maxterms of the function.
- **Conversion between Canonical Forms**
- The complement of a function expressed as the sum of minterms equals the sum of minterms missing from the original function.
- This is because the original function is expressed by those minterms that make the function equal to 1, whereas its complement is a 1 for those minterms that the function is a 0.

Cont...

$$F(A, B, C) = \Sigma(1, 4, 5, 6, 7)$$

- This has a complement that can be expressed as

$$F'(A, B, C) = \Sigma(0, 2, 3) = m_0 + m_2 + m_3$$

- Now, if we take the complement of F' by DeMorgan's theorem, we obtain F in a

$$F = (m_0 + m_2 + m_3)' = m_0' \cdot m_2' \cdot m_3' = M_0 M_2 M_3 = \Pi(0, 2, 3)$$

Canonical form.

Cont...

- The last conversion follows from the definition of minterms and maxterms as shown in Table 2-3.
- From the table it is clear that the following relation holds true:
- $m_j' = M_j$
- That is, the maxterm with subscript j is a complement of the minterm with the same subscript j , and vice versa.
- The last example demonstrates the conversion between a function expressed in sum of minterms and its equivalent in product of maxterms

Cont...

- A general conversion procedure:
- To convert from one canonical form to another, interchange the symbols Σ and Π and list those numbers missing from the original form.
- In order to find the missing terms, one must realize that the total number of minterms or maxterms is 2^n , where n is the number of binary variables in the function.
- If we have $F(x, y, z) = \Sigma(1, 3, 6, 7)$, and if the function is expressed as a sum of minterms as follows:

Standard Forms

- Then we can easily express as a product of maxterms by first realizing that there are total of eight minterms or maxterms in a function of three variable. Then we determine the missing terms to be 0, 2, 4 and 5.
- The function $F(x, y, z) = \Pi(0, 2, 4, 5)$ is a product of maxterm is

- **Standard Forms**

- The two canonical forms of Boolean algebra are basic forms that one obtains from

Cont...

- These forms are very seldom the ones with the least number of literals, because each minterm or maxterm must contain, by definition, all the variables either complemented or uncomplemented.
- Another way to express Boolean functions is in standard form.
- In this configuration, the terms that form the function may contain one, two, or any number of literals.
- There are two types of standard forms: the sum of products and product of sums.
- The **sum of products** is a Boolean expression containing AND terms called

Cont...

- The sum denotes the ORing of these terms.
- An example of a function expressed in sum of products is
- $F_1 = y' + xy + x'yz'$
- The expression has three product terms of one, two, and three literals each, respectively.
- Their sum is in effect an OR operation.
- A **product of sums** is a boolean expression containing OR terms, called **sum terms**.
- each term may have any number of literals.
- The **product** denotes the ANDing of these

Cont...

- This expression has three sum terms of one, two, and four literals each, respectively.
- The use of the words **product** and **sum** stems from the similarity of the AND operation to the arithmetic product (multiplication) and the similarity of the OR operation to the arithmetic sum (addition).
- A Boolean function may be expressed in a nonstandard form.
- For example, the function
- $F_3 = (AB + CD)(A'B' + C'D')$
- is neither in sum of products nor in product

-
- It can be changed to a standard form by using the distributive law to remove the parentheses:
 - $F_3 = A'B'CD + ABC'D'$

2-7 Other Logic Operations

2-7 Other Logic Operations

- When the binary operators AND and OR are placed between variables, x and y , they form two Boolean functions, $x.y$ and $x + y$, respectively.
- There are 2^{2^n} possible function for n variables, as there 2^n minterms and for each minterms the value of the function can be either 1 or 0.
- For two variables, $n=2$, the number of possible Boolean functions is 16.
- Therefore, the AND and OR functions are only two of a total of 16 possible functions formed with two binary variables.

Cont...

- It would be instructive to find the other 14 functions and investigate their properties.
- The truth tables for the 16 functions formed with two binary variables, x and y , are listed in Table 2-7.

TABLE 2-7
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
Operator symbol		.	/		/			$\oplus$	+	$\downarrow$	$\odot$	'	$\subset$	'	$\supset$	$\uparrow$	

Cont...

- In this table, each of the 16 columns, F_0 to F_{15} represents a truth table of one possible function for the two given variables, x and y .
- Note that the functions are determined from the 16 binary combinations that can be assigned to F .
- Some of the functions are shown with an operator symbol.
- For example, F_1 represents the truth table for AND and F_7 represents the truth table for OR.
- The operator symbols for these functions

Cont...

TABLE 2-8
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
$F_9 = xy + x'y'$	$x \odot y$	Equivalence	x equals y
$F_{10} = y'$	y'	Complement	Not y
$F_{11} = x + y'$	$x \supset y$	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

Cont...

- The Boolean expression listed are simplified to their minimum number of literals.
- Although each function can be expressed in terms of the Boolean operators AND, OR, and NOT, there is no reason one cannot assign special operator symbols for expressing the other functions.
- Such operator symbols are listed in the second column of Table 2-8.
- However, all the new symbols shown, except for the exclusive-OR symbol, $\oplus$, are not in common use by digital designers.

Cont....,

- Each of the functions in Table 2-8 is listed with an accompanying name and a complement that explains the function in some way.
- The 16 functions listed can be subdivided into three categories:
 - 1. Two functions that produce a constant 0 or 1.
 - 2. Four functions with unary operations: complement and transfer
 - 3. Ten functions with binary operators the define eight different operations: AND, OR, NAND, NOR, exclusive-OR, equivalence,

Cont...

- Any function can be equal to a constant, but a binary function can be equal to only 1 or 0.
- The complement function produces the complement of each of the binary variables.
- A function that is equal to an input variable has been given the name **transfer**, because the variable x or y is transferred through the gate that forms the function without changing its value.
- Of the eight binary operators, two (inhibition and implication) are used by logicians but

Cont...

- The AND and OR operators have been mentioned in conjunction with Boolean algebra.
- The other four functions are extensively used in the design of digital system.
- The NOR function is the complement of the OR function and its name is abbreviated of not-OR.
- Similarly, NAND is the complement of AND and is abbreviation of not-AND.
- The exclusive-OR, abbreviated XOR or EOR, is similar to OR but excludes the combination of both x and y being equal to

Cont...

- The exclusive-OR and equivalence are complements of each other. This can be easily shown by inspecting Table 2-7.
- For this reason, the equivalence function is often called exclusive-NOR, i.e. exclusive-OR-NOT.
- Boolean algebra, as defined in section 2-3 has two binary operators, which we have called AND and OR, and a unary operator, NOT(complement).
- From the definitions, we have deduced a number of properties of these operators and now have defined other binary

Cont...

- We could have just as well started with the operator NOR , for example, and later defined AND, OR, and NOT in terms of it.
- There are, nevertheless, a good reasons for introducing Boolean algebra in the way it has been introduced.
- The concepts of "and," "or," and "not" are familiar and are used by people to express everyday logical ideas.
- Moreover, the Huntington postulates reflect the dual nature of the algebra, emphasizing the symmetry of $+$ and $\cdot$ with respect to each other.

2-8 Digital Logic Gates

2-8 Digital Logic Gates

- Of the 16 function, only eight are considered for gates.
- These are: complement, transfer, AND, OR, NAND, NOR, exclusive-OR, and equivalence
- The graphic symbols and truth tables of the eight gates are shown in Fig 2-5.
- Each gate has only one or two binary input variables designated by x and y .
- Each gate has one binary output variable designated by F .

Cont...

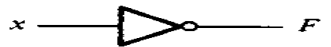
Name	Graphic symbol	Algebraic function	Truth table															
AND		$F = xy$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	x	y	F	0	0	0	0	1	0	1	0	0	1	1	1
x	y	F																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR		$F = x + y$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	x	y	F	0	0	0	0	1	1	1	0	1	1	1	1
x	y	F																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
Inverter		$F = x'$	<table><tr><th>x</th><th>F</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	x	F	0	1	1	0									
x	F																	
0	1																	
1	0																	
Buffer		$F = x$	<table><tr><th>x</th><th>F</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	x	F	0	0	1	1									
x	F																	
0	0																	
1	1																	
NAND		$F = (xy)'$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	x	y	F	0	0	1	0	1	1	1	0	1	1	1	0
x	y	F																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
NOR		$F = (x + y)'$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	x	y	F	0	0	1	0	1	0	1	0	0	1	1	0
x	y	F																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
Exclusive-OR (XOR)		$F = xy' + x'y$ $= x \oplus y$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	x	y	F	0	0	0	0	1	1	1	0	1	1	1	0
x	y	F																
0	0	0																
0	1	1																
1	0	1																
1	1	0																
Exclusive-NOR or equivalence		$F = xy + x'y'$ $= x \odot y$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	x	y	F	0	0	1	0	1	0	1	0	0	1	1	1
x	y	F																
0	0	1																
0	1	0																
1	0	0																
1	1	1																

FIGURE 2-5
Digital logic gates

Cont...

- The inverter circuit inverts the logic sense of a binary variable.
- It produces the NOT, or complement, function.
- The small circle in the output of the graphic symbol of an inverter designates the logic complement.
- The triangle symbol by itself designates a buffer circuit.
- A buffer produces the transfer function but does not produce any particular logic operation.
- The binary value of the output is equal

Cont...

- This circuit is mainly used for power amplification and is equivalent to two inverters connected in cascade.
- The NAND function is the complement of the AND function.
- The NAND function is indicated by a graphic symbol that consists of AND graphic symbol followed by a small circle.
- The NOR function is the complement of the OR function and uses an OR graphic symbol followed by a small circle.

Cont...

- The exclusive -OR gate has a graphic symbol similar to that of the OR gate.
- It has an additional curved line on the input side.
- The equivalence, or exclusive-NOR gate is the complement of the exclusive-OR.
- You can see the small circle on the output side of the graphic symbol to indicate complement.

Cont...

- **Extension to Multiple Inputs**
- The gates shown in Fig 2-5, except for the inverter and buffer, can be extended to have more than two inputs.
- A gate can be extended to have multiple inputs if the binary operation it represents is commutative and associative.
- The AND and OR operations defined in Boolean algebra have these two properties.
- For the OR function, we have

Cont...

$$x + y = y + x \quad \text{commutative}$$

and

$$(x + y) + z = x + (y + z) = x + y + z \quad \text{associative}$$

- The NAND and NOR functions are commutative and other gates can be extended to have more than two inputs, with slight definition change.
- NOR (or NAND) gate can be defined as complemented OR (or AND) gate.

Cont...

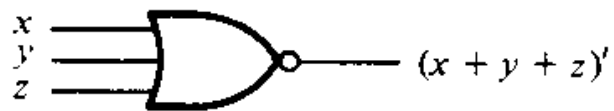
- Thus, by definition, we have

$$x \downarrow y \downarrow z = (x + y + z)'$$

$$x \uparrow y \uparrow z = (xyz)'$$

- The graphic symbols for the 3-input gates are shown in Fig .2-7.

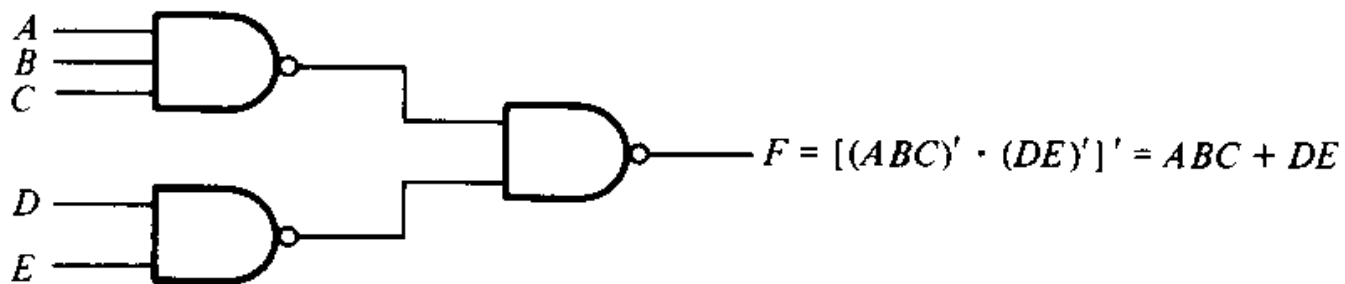
Cont...



(a) Three-input NOR gate



(b) Three-input NAND gate



(c) Cascaded NAND gates

FIGURE 2-7

Multiple-input and cascaded NOR and NAND gates

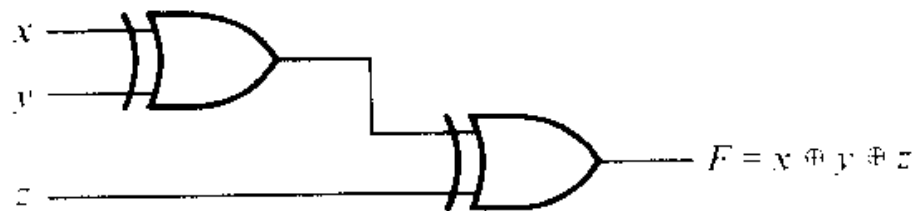
Cont...

- In writing cascaded NOR and NAND operations, one must use the correct parentheses to signify proper sequence of gates.
- For example, consider the circuit of Fig 2-7 (c)
- The Boolean function for the circuit must be written as
- $$F = [(ABC)'(DE)']' = ABC + DE$$
- The second expression is obtained from DeMorgan's theorem.
- It also shows that an expression in sum

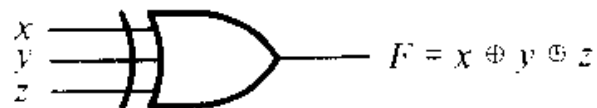
Cont...

- The exclusive-OR and equivalence gates are both commutative and associative and can be extended to more than two inputs.
- The exclusive-OR is an odd function, i.e., it is equal to 1 if an odd number of input variables are equal to 1.
- The construction of a 3-input exclusive-OR function is shown in (a).

Cont...



(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

FIGURE 2-8

3-input exclusive-OR gate

Cont...

- It is normally implemented by cascading 2-input gates, as shown in (a).
- Graphically, it can be represented with a single 3-input gate.
- Notice that for three input exclusive-OR gate(s), the output is 1 when an odd number of inputs, 1 or 3 in this case, are 1.

2-9 Integrated Circuits

2-8 Integrated Circuits

- Digital circuits are constructed with integrated circuits.
- An integrated circuit (IC) is a small silicon semiconductor crystal, called a chip, containing the electronic components for the digital gates.
- The various gates are interconnected inside the chip to form the required circuit.
- The chip is mounted in a ceramic or plastic container, and connections are welded to external pins to form the

Cont...

- The size of the IC package is very small.
- For example, an entire microprocessor is enclosed within a 64-pin package with dimensions of 50 X15X4 millimeters.
- Each IC has a numeric designation printed on the surface of the package for identification.
- Vendors publish data books that contain descriptions and other information about the IC's that they manufacture.

Levels of Integration

- Digital ICs are often categorized according to their circuit complexity as measured by the number of logic gates in a single package.
- They are categorized as small scale, medium scale, and large scale integration devices depending on the number of internal gates they have.
- **Small-scale integration (SSI)** devices contain several independent gates in a single package.
- The inputs and outputs of the gates are

Cont...

- The number of gates is usually fewer than 10 and is limited by the number of pins available in the IC.
- **Medium-scale integration (MSI)** have approximately 10 to 100 gates in a single package and are used to build decoders, adders or multiplexers.
- **Large-scale integration (LSI)** devices contain between 100 and a few thousands gates in a single package.
- They include digital systems as processors, memory chips, and

Digital Logic Families

- **Very large-scale integration (VLSI)**
devices contain thousands of gates within a single package.
- Examples are large memory arrays and complex microcomputer chips.
- **Digital Logic Families**
 - Digital integrated circuits are classified by not only their complexity or logical operation, but also by the specific circuit technology to which they belong.
 - The circuit technology is referred to as digital logic family.

Cont...

- Each logic family has its own basic electronic circuit upon which more complex digital circuits and components are developed.
- The basic circuit in each technology is a NAND, NOR, or an inverter gate.
- The electronic components used in the construction of the basic circuit are usually used as the name of the technology.
- Many different logic family of integrated circuits have been introduced

Cont...

- The following are the most popular
 - TTL transistor-transistor logic
 - ECL emitter-coupled logic
 - MOS metal-oxide semiconductor
 - CMOS complementary metal-oxide semiconductor
- TTL is a widespread logic family that has been in operation for some time and is considered as standard.
- ECL has an advantage in systems requiring high-speed operation.
- MOS is suitable for circuits that need

Cont...

- The transistor-transistor logic family evolved from a previous technology that used diodes and transistors for the basic NAND gate.
- This technology was called DTL for diode-transistor logic.
- Later the diodes were replaced by transistors to improve the circuit operation.
- The name of the logic family was changed to TTL as a result.

Cont...

- Emitter-coupled logic (ECL) circuits provide the highest speed among the integrated digital logic families.
- ECL is used in systems such as supercomputers and signal processors, where high speed is essential.
- The transistors in ECL gates operate in a non-saturated state, a condition that allows the achievement of propagation delays of 1 to 2 nanoseconds.
- The metal-oxide semiconductor (MOS) is a unipolar transistor that depends upon

Cont...

- This is in contrast to the bipolar transistor used in TTL and ECL gates, where both carriers exist during normal operation.
- A p-channel MOS is referred to as PMOS and n-channel as NMOS.
- NMOS is the one that is currently used in circuits with only one type of MOS transistor.
- Complementary MOS (CMOS) technology uses one PMOS and one NMOS transistor connected in a

Cont...

- The most important advantages of MOS over bipolar transistors are the high packing density of circuits, and a simpler processing technique during fabrication.
- Also MOS technology encompasses a more economical operation because of the low-power consumption.
- The characteristics of digital logic families are usually compared by analyzing the circuit of the basic gate in each family.
- The most important parameters that are

Cont...

- *Fan-out* specifies the number of standard loads that the output of a typical gate can drive without impairing its normal operation.
- A standard load is usually defined as the amount of current needed by an input of another similar gate of the same family.
- *Power dissipation* is the power consumed by the gate that must be available from the power supply.
- *Propagation delay* is the average transition delay time for the signal to

Integrated Circuits

- The operating speed is inversely proportional to the propagation delay.
- *Noise Margin* is the minimum external noise voltage that causes an undesirable change in the circuit output.
- **Integrated Circuits**
- Some typical SSI circuit are shown in Fig 2-9.
- Each IC is enclosed within a 14 or 16-pin package.
- A notch placed on the left side of the package is used to reference the pin

Cont...

- The inputs and outputs of the gates are connected to the package pins, as indicated in each diagram.

Cont...

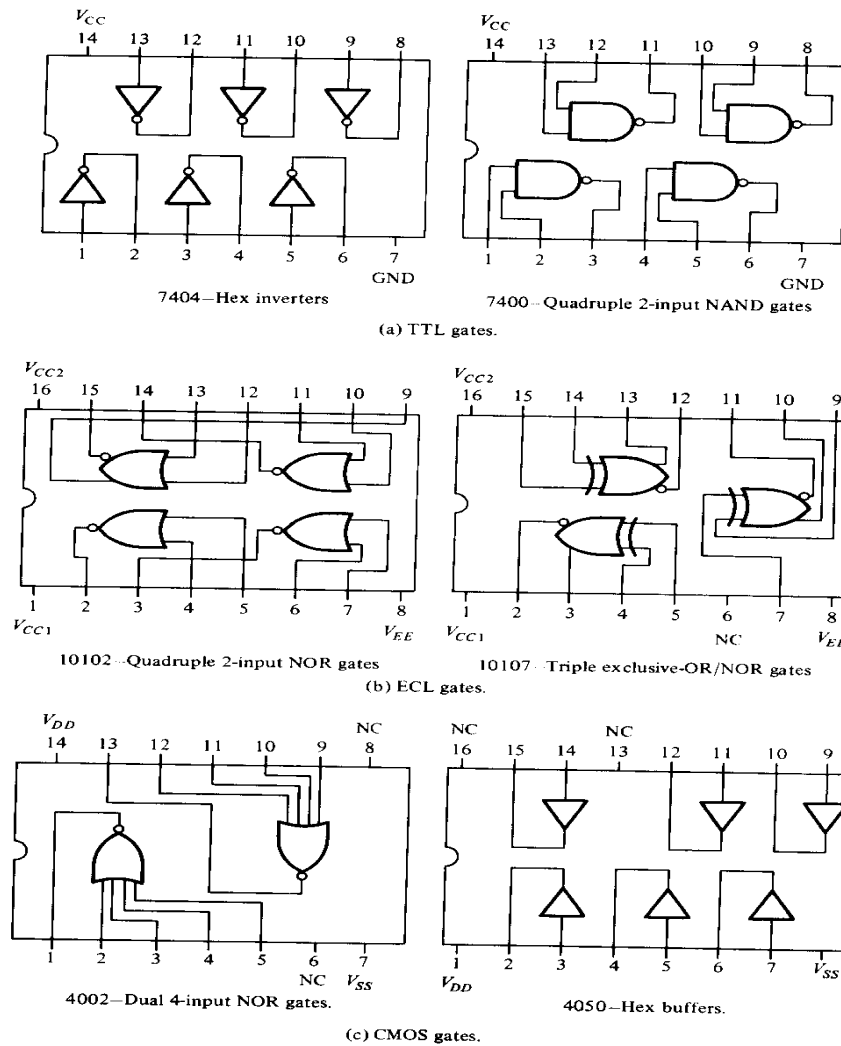


FIGURE 2-9
 Some typical integrated-circuit gates

Cont...

- TTL IC's are usually distinguished by their numerical designation as the 5400 and 7400 series.
- The former has a wide operating temperature range, suitable for military use.
- And the latter has a narrower temperature range, suitable for commercial use.
- The numerical designation of 7400 series means that IC packages are numbered as 7400, 7401, 7402, etc.

Cont...

- The 7400 provides four (quadruple) 2-input NAND gates
- The terminal marked Vcc and GRD (ground) are the power supply pins that require a voltage of 5 volts for proper operation.
- The two logic levels for TTL are 0 and 3.5 volts.
- The TTL logic family actually consists of several subfamilies or series.
- Table 2-9 lists the name of each series and the prefix designation that identifies

Cont..

TABLE 2-9
Various Series of the TTL Logic Family

TTL Series	Prefix	Example
Standard TTL	74	7486
High-speed TTL	74H	74H86
Low-power TTL	74L	74L86
Schottky TTL	74S	74S86
Low-power Schottky TTL	74LS	74LS86
Advanced Schottky TTL	74AS	74AS86
Advanced Low-power Schottky TTL	74ALS	74ALS86

Cont...

- The differences between the various TTL series are in their electrical characteristic, such as power dissipation, propagation delay, and switching speed.
- They do not differ in the pin assignment or logic operation performed by the internal circuits.
- The most common ECL ICs are designated as 10000 series.
- Fig 2-9(b) shows two ECL circuits.
- The 10102 provides four 2-input NOR

Cont...

- Note that an ECL gate may have two outputs, one for the NOR function and another for the OR function.
- The 10107 IC provides three exclusive-OR gates.
- ECL gates have three terminals for power supply.
- V_{cc1} and V_{cc2} are usually connected to ground, and V_{EE} to a -5.2 volt supply.
- The two logic level for ECL are -0.8 and -1.8
- CMOS circuits of the 4000 series are

Cont...

- The 4050 IC provides six buffer gates.
- Both ICs have unused terminals marked NC (no connection).
- The terminals marked V_{DD} requires a power-supply voltage from 3 to 15 volts.
- V_{SS} is usually connected to ground.
- The two logic level are 0 and V_{DD} volts.
- The original 4000 series of CMOS circuits was designed independently from the TTL series.
- Since TTL became a standard in the industry, vendors started to supply other

Cont...

- The CMOS series available commercially are listed in Table 2-10.
- The 74H series operates at higher speeds than the 74 C series.
- The 74HCT series is both electrically and pin compatible with TTL devices.
- This means that 74HCT ICs can be connected directly to TTL ICs without interfacing.

Cont...

TABLE 2-10
Various Series of the CMOS Logic Family

CMOS series	Prefix	Example
Original CMOS	40	4009
Pin compatible with TTL	74C	74C04
High-speed and pin compatible with TTL	74HC	74HCO4
High-speed and electrically compatible with TTL	74HCT	74HCT04

Review Exercises

1. Demonstrate by means of truth tables the validity of the following identities.

(a) DeMorgan's theorem for three variables :

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

(b) The second distributive law: $x + yz = (x + y)(x + z)$.

(c) The consensus theorem: $xy + x'z + yz = xy + x'z$ (This is done algebraically in Example 2-1, Part-4)

2. Simplify the following Boolean expression to a minimum number of literals.

(a) $x'y' + xy + x'y$

(b) $x'z + xy + x'z + xy + x'z + xy + x'z + xy$ (Use the consensus

Cont..

3. Simplify the following Boolean expressions to a minimum number of literals:

(a) $ABC + A'B + ABC'$

(b) $(x + y)'(x' + y')$

4. Find the complement of $F = x + yz$; then show that $F.F' = 0$ and $F + F' = 1$.

5. Express the following functions in sum of minterms and product of maxterms:

(a) $F(A, B, C, D) = B'D + A'D + BD$

Cont..

6. Draw the logic diagram corresponding for the following functions before and after simplifying it.

(a) $F = BC' + AB + BC$

7. How many gates does **small scale integrated circuits**, **medium scale integrated circuits**, and **Large scale integrated circuits** have?
8. What are the most popular types of digital logic families?
9. What is fan out?
10. What is noise margin?