

# **Chapter 11**

# **Boolean Algebra**

# What is Boolean Algebra?

- A minor generalization of propositional logic.
  - In general, an *algebra* is any mathematical structure satisfying certain standard algebraic axioms.
    - Such as associative/commutative/transitive laws, *etc.*
  - General theorems that are proved about an algebra then apply to *any* structure satisfying these axioms.
- *Boolean algebra* just generalizes the rules of propositional logic to sets other than **{T,F}**.
  - E.g., to the set **{0, 1}** of base-2 digits, or the set **{V<sub>L</sub>, V<sub>H</sub>}** of low and high voltage levels in a circuit.
- We will see that this algebraic perspective lends itself to the design of *digital logic circuits*.

Claude Shannon's  
Master's thesis!

# Boolean Algebra

§1 – Boolean Functions

§2 – Representing Boolean Functions

§3 – Logic Gates

§4 – Minimization of Circuits

## §11.1 – Boolean Functions

- Boolean complement, sum, product.
- Boolean expressions and functions.
- Boolean algebra identities.
- Duality.
- Abstract definition of a Boolean algebra.

# Complement, Sum, Product

- Correspond to logical NOT, OR, and AND.
- We will denote the two logic values as **0**: $\equiv$ **F** and **1**: $\equiv$ **T**, instead of **False** and **True**.
  - Using numbers encourages algebraic thinking.
- New, more algebraic-looking notation for the most common Boolean operators:

$$\bar{x} \equiv \neg x$$

$$x \cdot y \equiv x \wedge y$$

$$x + y \equiv x \vee y$$

Precedence order  $\rightarrow$

# Boolean Sum, Boolean Product, Complement

Boolean Sum (or)

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

Boolean Product (and)

$$1 \cdot 1 = 1, \quad 1 \cdot 0 = 0, \quad 0 \cdot 1 = 0, \quad 0 \cdot 0 = 0$$

Complement

$$\bar{1} = 0, \quad \bar{0} = 1$$


**Example 1** Find the value of  $1 \cdot 0 + \overline{(0 + 1)}$

$$1 \cdot 0 + \overline{(0 + 1)} = 0 + \bar{1} = 0 + 0 = 0.$$

# Boolean Expressions and Boolean Functions

- Let  $B = \{0, 1\}$ , the set of Boolean values.
- For all  $n \in \mathbf{Z}^+$ , any function  $f: B^n \rightarrow B$  is called a *Boolean function of degree  $n$* .
- There are  $2^{2^n}$  (wow!) distinct Boolean functions of degree  $n$ .
  - $\exists 2^n$  rows in truth table, with 0 or 1 in each.

<u>Degree</u>	<u>How many</u>	<u>Degree</u>	<u>How many</u>
0	2	4	65,536
1	4	5	4,294,967,296
2	16	6	18,446,744,073,709,551,616.
3	256		

**Example 2** Find the values of Boolean function  
(of degree 2)  $F(x, y) = x \bar{y}$ .  Boolean expression

<b>TABLE 1</b>		
<b><i>x</i></b>	<b><i>y</i></b>	<b><i>F(x, y)</i></b>
1	1	0
1	0	1
0	1	0
0	0	0



**Example 3** Find the values of Boolean function  
(of degree 3)  $F(x, y, z) = xy + \bar{z}$ .

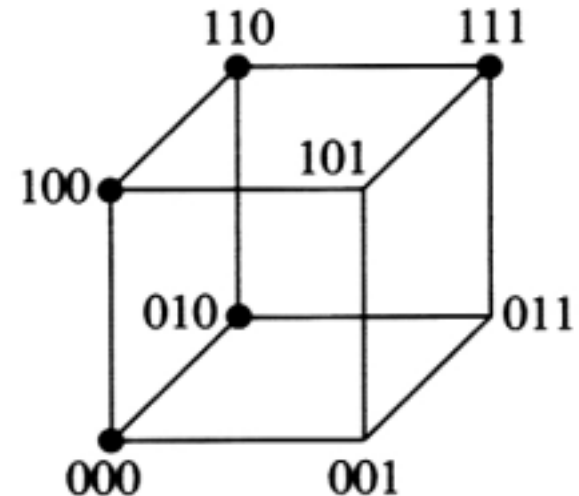
<b>TABLE 2</b>					
<b><math>x</math></b>	<b><math>y</math></b>	<b><math>z</math></b>	<b><math>xy</math></b>	<b><math>\bar{z}</math></b>	<b><math>F(x, y, z) = xy + \bar{z}</math></b>
1	1	1	1	0	1
1	1	0	1	1	1
1	0	1	0	0	0
1	0	0	0	1	1
0	1	1	0	0	0
0	1	0	0	1	1
0	0	1	0	0	0
0	0	0	0	1	1

A Boolean function of degree  $n$  can be represented by an  $n$ -cube (hypercube) with the corresponding function value at each vertex.

**Example 4** Express the Boolean function

$$F(x, y, z) = xy + \bar{z} \quad \text{by a hypercube.}$$

TABLE 2					
$x$	$y$	$z$	$xy$	$\bar{z}$	$F(x, y, z) = xy + \bar{z}$
1	1	1	1	0	1
1	1	0	1	1	1
1	0	1	0	0	0
1	0	0	0	1	1
0	1	1	0	0	0
0	1	0	0	1	1
0	0	1	0	0	0
0	0	0	0	1	1



**FIGURE 1**

# Boolean Expressions

- Let  $x_1, \dots, x_n$  be  $n$  different Boolean variables.
  - $n$  may be as large as desired.
- A *Boolean expression* (recursive definition) is a string of one of the following forms:
  - Base cases:  $0$ ,  $1$ ,  $x_1$ , ..., or  $x_n$ .
  - Recursive cases:  $E_1$ ,  $(E_1E_2)$ , or  $(E_1+E_2)$ , where  $E_1$  and  $E_2$  are Boolean expressions.
- A Boolean expression represents a Boolean function.
  - Furthermore, *every* Boolean function (of a given degree) can be represented by a Boolean expression.

## Boolean equivalents, operations on Boolean expressions

- Two Boolean expressions  $e_1$  and  $e_2$  that represent the exact *same* function  $f$  are called *equivalent*. We write  $e_1 \Leftrightarrow e_2$ , or just  $e_1 = e_2$ .
  - Implicitly, the two expressions have the same value for *all* values of the free variables appearing in  $e_1$  and  $e_2$ .
- The operators  $\bar{\phantom{x}}$ ,  $+$ , and  $\cdot$  can be extended from operating on expressions to operating on the functions that they represent, in the obvious way.

$$\overline{\overline{F}(x_1, x_2, \dots, x_n)} = \overline{\overline{F(x_1, x_2, \dots, x_n)}} \quad \text{i.e. } F(x, y, z) = xy + \bar{z}, \quad \overline{\overline{F}(x, y, z)} = \overline{\overline{xy + \bar{z}}} = \overline{\overline{xy} + \overline{\bar{z}}} = \overline{\overline{xy} + z}$$

$$(F + G)(x_1, x_2, \dots, x_n) = F(x_1, x_2, \dots, x_n) + G(x_1, x_2, \dots, x_n)$$

$$(FG)(x_1, x_2, \dots, x_n) = F(x_1, x_2, \dots, x_n)G(x_1, x_2, \dots, x_n)$$

How many different Boolean functions of degree 3?

**TABLE 3 The Boolean Functions of Degree Two.**

$x$	$y$	$F_1$	$F_2$	$F_3$	$F_4$	$F_5$	$F_6$	$F_7$	$F_8$	$F_9$	$F_{10}$	$F_{11}$	$F_{12}$	$F_{13}$	$F_{14}$	$F_{15}$	$F_{16}$
1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
1	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
0	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

**Example 5** (How many different Boolean functions of degree 3?)

**TABLE 4 The Number of Boolean Functions of Degree  $n$ .**

<i>Degree</i>	<i>Number</i>
1	4
2	16
3	256
4	65,536
5	4,294,967,296
6	18,446,744,073,709,551,616

# Identities of Boolean Algebra

TABLE 5 Boolean Identities.	
<i>Identity</i>	<i>Name</i>
$\overline{\overline{x}} = x$	Law of the double complement
$x + x = x$ $x \cdot x = x$	Idempotent laws
$x + 0 = x$ $x \cdot 1 = x$	Identity laws
$x + 1 = 1$ $x \cdot 0 = 0$	Domination laws
$x + y = y + x$ $xy = yx$	Commutative laws
$x + (y + z) = (x + y) + z$ $x(yz) = (xy)x$	Associative laws
$x + yz = (x + y)(x + z)$ $x(y + z) = xy + xz$	Distributive laws
$\overline{(xy)} = \overline{x} + \overline{y}$ $\overline{(x + y)} = \overline{x} \overline{y}$	De Morgan's laws
$x + xy = x$ $x(x + y) = x$	Absorption laws
$x + \overline{x} = 1$	Unit property
$x\overline{x} = 0$	Zero property

Not true  
in ordinary  
algebras.



**Example 6** Prove that the Boolean equation  $x(y + z) = xy + xz$  is true.

<b>TABLE 6</b>							
<b><math>x</math></b>	<b><math>y</math></b>	<b><math>z</math></b>	<b><math>y + z</math></b>	<b><math>xy</math></b>	<b><math>xz</math></b>	<b><math>x(y + z)</math></b>	<b><math>xy + xz</math></b>
1	1	1	1	1	1	1	1
1	1	0	1	1	0	1	1
1	0	1	1	0	1	1	1
1	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0

## **Example 7** Prove the **absorption law**

$$x(x + y) = x.$$

*Solution:*

$x(x + y) = (x + 0)(x + y)$	Identity law for the Boolean sum
$= x + 0 \cdot y$	Distributive law of the Boolean sum over the Boolean product
$= x + y \cdot 0$	Commutative law for the Boolean product
$= x + 0$	Domination law for the Boolean product
$= x.$	Identity law for the Boolean sum



# Duality

**Definition:** The *dual*  $e^d$  of a Boolean expression  $e$  representing function  $f$  is obtained by exchanging  $+$  with  $\cdot$ , and  $0$  with  $1$  in  $e$ .

The function represented by  $e^d$  is denoted  $f^d$ .

**Duality principle:** If  $e_1 \Leftrightarrow e_2$  then  $e_1^d \Leftrightarrow e_2^d$ .

**Example:** The equivalence  $x(x+y) = x$  implies (and is implied by)  $x + xy = x$ .

**Example 8 :** Find the *dual* of the Boolean expressions

$$f(x, y) = x(y + 0), \quad g(x, y, z) = \bar{x} \cdot 1 + (\bar{y} + z).$$

*Solution:*

$$f^d(x, y) = x + (y \cdot 1),$$

$$g^d(x, y, z) = (\bar{x} + 0)(\bar{y}z).$$

**Example 9 :** Take the duals of both sides of the Boolean expression.

$$x(x + y) = x.$$

*Solution:*

$$x + (xy) = x.$$

# The Abstract Definition of Boolean Algebra

**Definition 1** A *Boolean algebra* is a set  $B$  with operators  $\vee$  and  $\wedge$ , element 0 and 1, and a unary operator  $\bar{\phantom{x}}$  such that the following properties holds for  $x, y, z$  in  $B$ .

$$\left. \begin{array}{l} x \vee 0 = x \\ x \wedge 1 = x \end{array} \right\}$$

Identity laws

$$\left. \begin{array}{l} x \vee \bar{x} = 1 \\ x \wedge \bar{x} = 0 \end{array} \right\}$$

Complement laws

$$\left. \begin{array}{l} (x \vee y) \vee z = x \vee (y \vee z) \\ (x \wedge y) \wedge z = x \wedge (y \wedge z) \end{array} \right\}$$

Associative laws

$$\left. \begin{array}{l} x \vee y = y \vee x \\ x \wedge y = y \wedge x \end{array} \right\}$$

Commutative laws

$$\left. \begin{array}{l} x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z) \\ x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z) \end{array} \right\}$$

Distributive laws

Note that  $B$  may generally have other elements besides **0**, **1**, and we have not fully defined any of the operators!

# More about Boolean algebras

- *Any* Boolean algebra can be proven to satisfy all the theorems of “ordinary” Boolean algebra!
- An example of another Boolean algebra:
  - For any set  $U$ , let  $B = 2^U$ ,  $\mathbf{0} = \emptyset$ ,  $\mathbf{1} = U$ ,  $\vee = \cup$ ,  $\wedge = \cap$ , and  $\neg = \overline{\phantom{x}}$  (set complement).
  - Then,  $(B, \mathbf{0}, \mathbf{1}, \wedge, \vee, \neg)$  is a Boolean algebra!
- Boolean algebras can also be defined in terms of *lattices* (in chapter 7, though we skipped it).
  - A poset where every pair  $x, y$  has a lub and a glb.
  - A complemented, distributed lattice is a Boolean alg.

## §11.2 Representing Boolean Functions

- Sum-of-products Expansions
  - A.k.a. Disjunctive Normal Form (DNF)
- Product-of-sums Expansions
  - A.k.a. Conjunctive Normal Form (CNF)
- Functional Completeness
  - Minimal functionally complete sets of operators.

# Sum-of-Products Expansions

**Example 1 :** Find the Boolean expressions that represent the function  $F(x, y, z)$  and  $G(x, y, z)$ .

**TABLE 1**

$x$	$y$	$z$	$F$	$G$
1	1	1	0	0
1	1	0	0	1
1	0	1	1	0
1	0	0	0	0
0	1	1	0	0
0	1	0	0	1
0	0	1	0	0
0	0	0	0	0

*Solution:*

For  $F$ ,  $F(x, y, z) = 1$  iff  $x = \bar{y} = z = 1$ .

$$\Rightarrow F(x, y, z) = x\bar{y}z.$$

For  $G$ ,  $G(x, y, z) = 1$  iff  $x = y = \bar{z} = 1$  or

$$\bar{x} = y = \bar{z} = 1.$$

$$\Rightarrow F(x, y, z) = xy\bar{z} + \bar{x}y\bar{z}.$$

**Definition 1:** A *literal* is a Boolean variable or its complement. A *minterm* of Boolean variables  $x_1, x_2, \dots, x_n$  is Boolean product  $y_1 y_2 \dots y_n$  where  $y_i = x_i$  or  $y_i = \overline{x_i}$ .

**Example 2** (P.710): Find the minterm  $F$  such that  $F = 1$  if  $x_1 = x_3 = 0$  and  $F = 0$  if  $x_2 = x_4 = x_5 = 1$ .

*Solution:* The minterm is  $\overline{x_1} \overline{x_2} \overline{x_3} x_4 x_5$ .

The sum of minterms that represents the functions is called the *sum-of-products expansion* or the **disjunctive normal form** of the Boolean function.

$$F(x, y, z) = \overline{x} y \overline{z} + x \overline{y} z + \overline{x} y z$$

**Example 3 (P.710):** Find the **sum-of-product expansion** for the function  $F(x, y, z) = (x + y)\bar{z}$ .

*Solution:*

(i)

$$F(x, y, z) = (x + y)\bar{z}$$

$$= x\bar{z} + y\bar{z}$$

**Distributive law**

$$= x1\bar{z} + 1y\bar{z}$$

**Identity law**

$$= x(y + \bar{y})\bar{z} + (x + \bar{x})y\bar{z}$$

**Unit property**

$$= xy\bar{z} + x\bar{y}\bar{z} + xy\bar{z} + \bar{x}y\bar{z}$$

**Distributive law**

$$= xy\bar{z} + x\bar{y}\bar{z} + \bar{x}y\bar{z}.$$

**Idempotent law**

$$F(x, y, z) = \overline{xy\bar{z}} + \overline{x\bar{y}\bar{z}} + \overline{\bar{x}y\bar{z}}$$



(ii)

TABLE 2					
$x$	$y$	$z$	$x + y$	$\bar{z}$	$(x + y)\bar{z}$
1	1	1	1	0	0
1	1	0	1	1	1
1	0	1	1	0	0
1	0	0	1	1	1
0	1	1	1	0	0
0	1	0	1	1	1
0	0	1	0	0	0
0	0	0	0	1	0

$F(x, y, z) = 1$  iff  $x = y = \bar{z} = 1$ ,  $x = \bar{y} = \bar{z} = 1$ , or  $\bar{x} = y = \bar{z} = 1$ .

$$\Rightarrow F(x, y, z) = xy\bar{z} + x\bar{y}\bar{z} + \bar{x}y\bar{z}.$$

# Sum-of-Products Expansions

**Theorem:** Any Boolean function can be represented as a sum of products of variables and their complements.

–**Proof:** By construction from the function's truth table. For each row that is 1, include a term in the sum that is a product representing the condition that the variables have the values given for that row.

Show an example on the board.

# Literals, Minterms, DNF

- A *literal* is a Boolean variable or its complement.
- A *minterm* of Boolean variables  $x_1, \dots, x_n$  is a Boolean product of  $n$  literals  $y_1 \dots y_n$ , where  $y_i$  is either the literal  $x_i$  or its complement  $\overline{x_i}$ .
  - Note that at most one minterm can have the value 1.
- The *disjunctive normal form* (DNF) of a degree- $n$  Boolean function  $f$  is the unique sum of minterms of the variables  $x_1, \dots, x_n$  that represents  $f$ .
  - A.k.a. the sum-of-products expansion of  $f$ .

# Conjunctive Normal Form

- A *maxterm* is a sum of literals.
- CNF is a *product-of-maxterms* representation.
- To find the CNF representation for  $f$ ,
- take the DNF representation for complement  $\neg f$ ,

$$\neg f = \sum_i \prod_j y_{ij}$$

- and then complement both sides & apply DeMorgan's laws to get:

$$f = \prod_i \sum_j \neg y_{ij}$$

Can also get CNF more directly, using the 0 rows of the truth table.

# Functional Completeness

- Since every Boolean function can be expressed in terms of  $\cdot, +, \bar{\phantom{x}}$ , we say that the set of operators  $\{\cdot, +, \bar{\phantom{x}}\}$  is *functionally complete*.
- There are smaller sets of operators that are also functionally complete.
  - We can eliminate either  $\cdot$  or  $+$  using DeMorgan's law.
- NAND  $|$  and NOR  $\downarrow$  are also functionally complete, each by itself (as a singleton set).
  - E.g.,  $\neg x = x|x$ , and  $xy = (x|y)|(x|y)$ .

# Reversible Boolean Logic

- A *reversible* Boolean function of degree  $n$  is a bijective function  $f:B^n \leftrightarrow B^n$ .
  - Also corresponds to a permutation of  $B^n$ .
- Reversible unary and binary Boolean operators are bijective operators on  $B$  and  $B^2$ , respectively.
  - Unary  $f:B \leftrightarrow B$ , binary  $f:B^2 \leftrightarrow B^2$ .
  - It turns out that *no* set of reversible unary and binary Boolean operators is functionally complete!
  - However, there are many ternary reversible operators that are functionally complete, even as singletons.

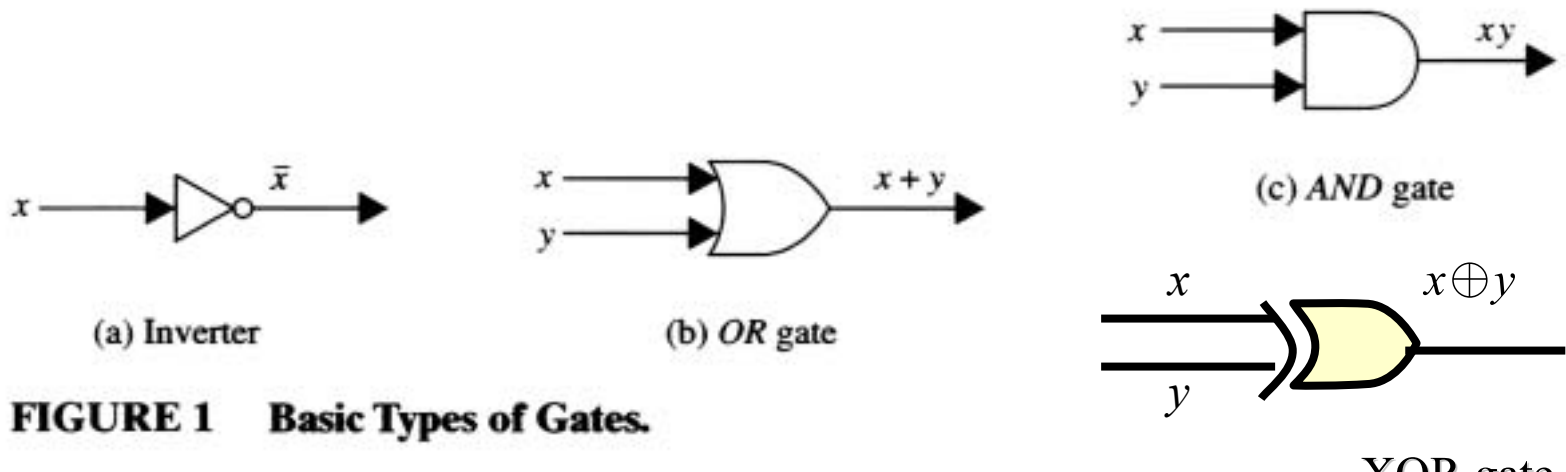
# A little Quantum Logic

- A *quantum Boolean function* is a bijective and linear function  $f: \mathbb{C}^{2^n} \leftrightarrow \mathbb{C}^{2^n}$ .
  - That is, it maps vectors of  $2^n$  complex numbers (one for each  $n$ -bit string of Boolean values) reversibly and linearly.
  - Any reversible Boolean function corresponds to a quantum Boolean function where a string in  $B^n$  is represented by  $c = 1$  for that string,  $c = 0$  for all others.
- Any quantum Boolean function can be built out of *quantum operators* operating on just  $\mathbb{C}$  and  $\mathbb{C}^2$ .
  - Quantization removes the need for ternary gates!

## §11.3 Logic Gates

**Boolean Algebra** can be used to model the circuits of electronic devices. A computer or other electronic device, is made up of a number of circuits. Each circuit can be designed using the rules of Boolean algebra. The basic elements of circuits are called **gates**.

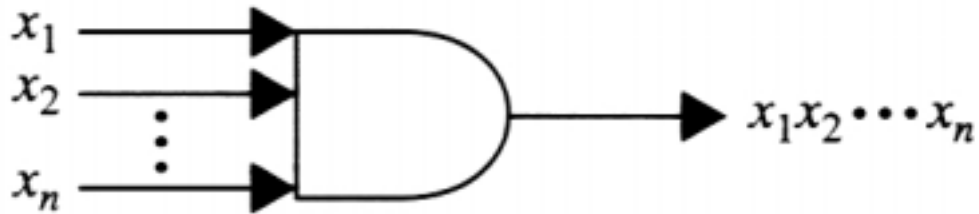
The circuit that we will study give output that depend only on the input and not on the current state of the circuit. Such circuits have no memory capacities. They are called **combinational circuits** or **gating networks**.



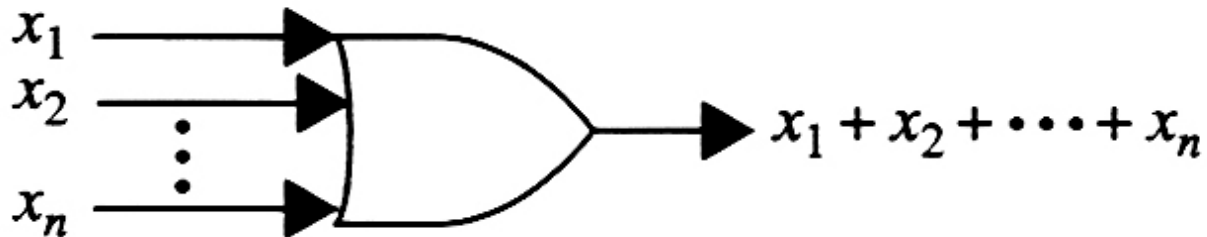
**FIGURE 1 Basic Types of Gates.**



We can extend these gates to arbitrarily many inputs.

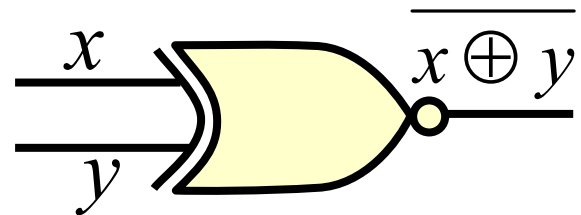
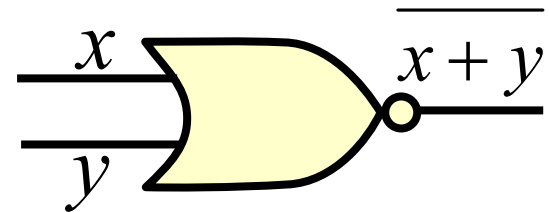
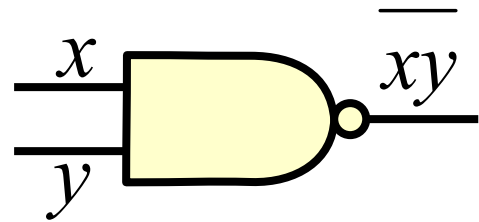


**FIGURE 2**    **Gates with  $n$  Inputs.**



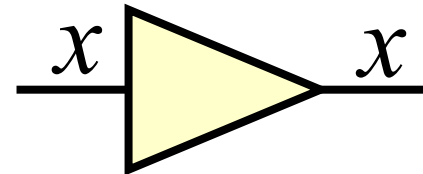
# NAND, NOR, XNOR

- Just like the earlier icons, but with a small circle on the gate's output.
  - Denotes that output is complemented.
- The circles can also be placed on inputs.
  - Means, input is complemented before being used.



# Buffer

- What about an inverter symbol *without* a circle?
- This is called a *buffer*. It is the identity function.
- It serves no logical purpose, but...
- It represents an explicit delay in the circuit.
  - This is sometimes useful for timing purposes.
- All gates, when physically implemented, incur a non-zero delay between when their inputs are seen and when their outputs are ready.



# Combinational Logic Circuits

- **Note:** The correct word to use here is “combinational,” **NOT** “combinatorial!”
  - Many sloppy authors get this wrong.
- These are circuits composed of Boolean gates whose outputs depend only on their most recent inputs, not on earlier inputs.
  - Thus these circuits have no useful memory.
    - Their state persists while the inputs are constant, but is irreversibly lost when the input signals change.

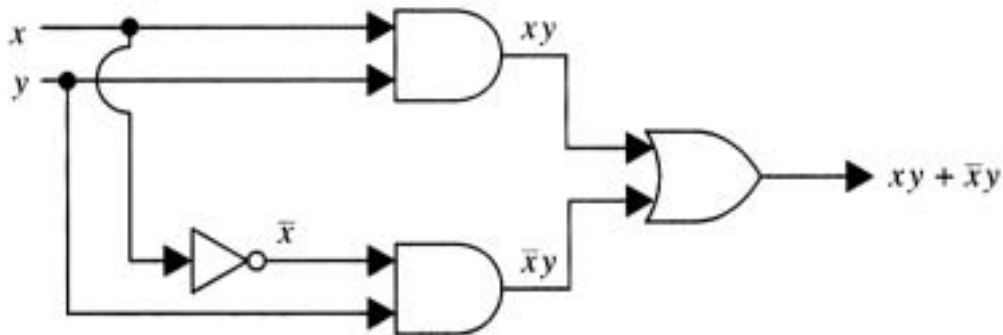
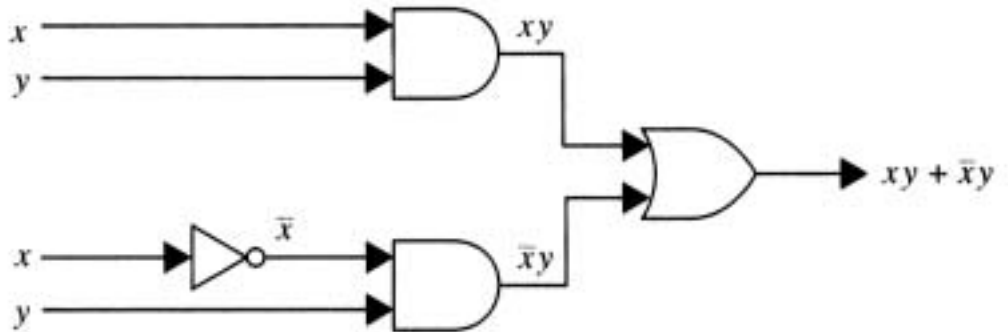
# Combinational Circuit Examples

- Draw a few examples on the board:
  - Majority voting circuit.
  - XOR using OR / AND / NOT.
  - 3-input XOR using OR / AND / NOT.
- Also, show some binary adders:
  - Half adder using OR/AND/NOT.
  - Full adder from half-adders.
  - Ripple-carry adders.

# Combinational Gates

**Example:** Construct circuits that produce the output  $xy + \bar{x}y$ .

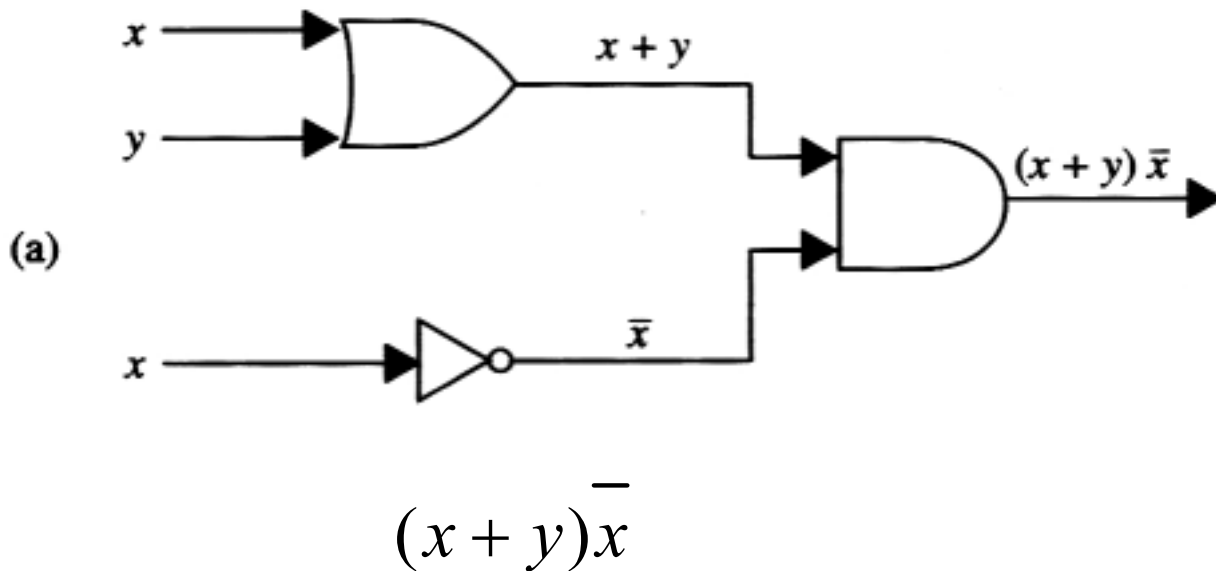
*Solution:*



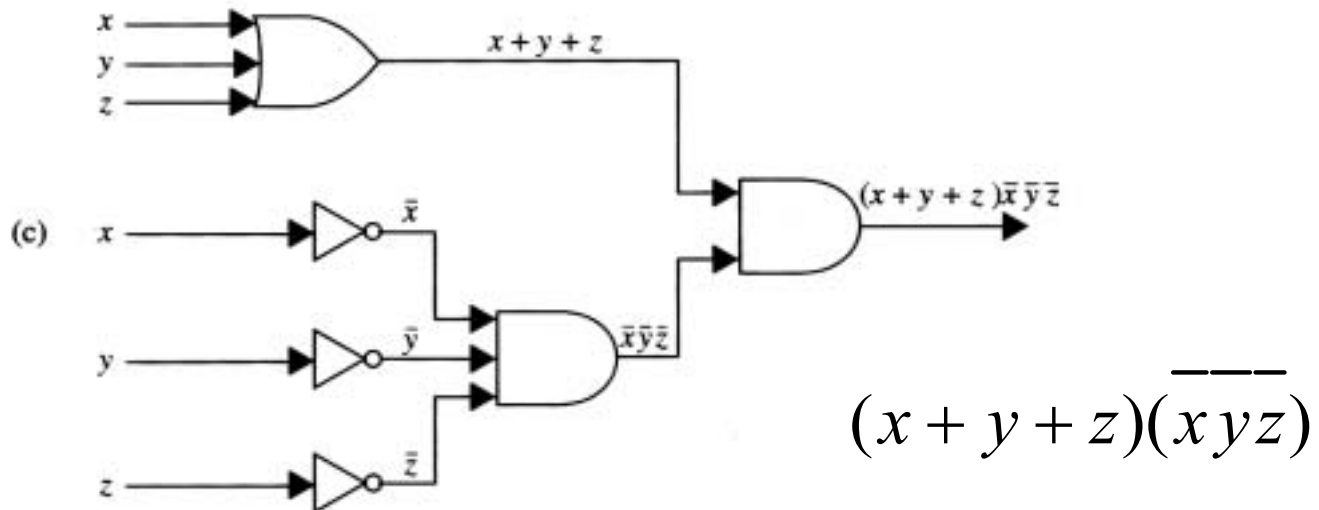
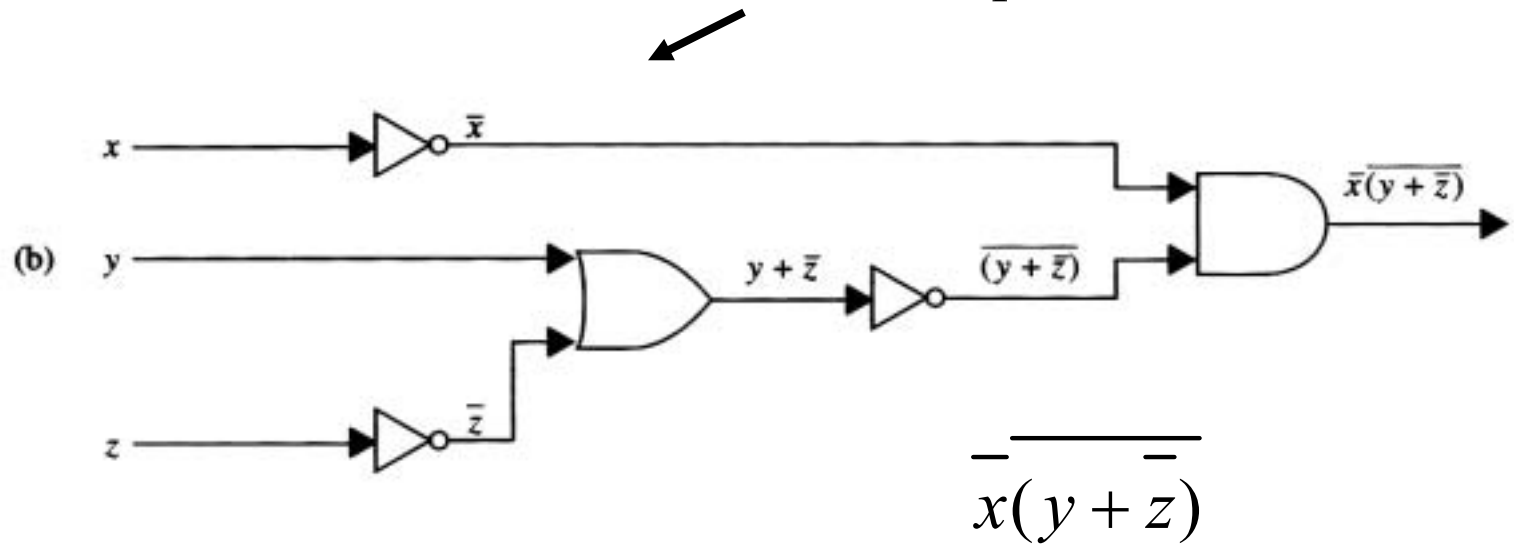
**FIGURE 3** Two Ways to Draw the Same Circuit.

**Example 1 (P. 714):** Construct circuits that produce the output (a)  $(x + y)\bar{x}$ , (b)  $\overline{\bar{x}(y + z)}$ , (c)  $(x + y + z)(\overline{\bar{x}y z})$ .

*Solution:*



not unique



**FIGURE 4** Circuits that Produce the Outputs Specified in Example 1.



## Example:

<i>Temperature &gt; 80°?</i>	<i>Humidity &gt; 50%</i>	<i>Air Conditioner on?</i>
no	no	no
no	yes	yes
yes	no	yes
yes	yes	yes

Design a circuit that will turn on or turn off the air conditioner according to the above conditions.

## Solution:

$$\text{Let } x = \begin{cases} 1, & \text{if temperature} > 80^\circ, \\ 0 & \text{otherwise.} \end{cases} \quad y = \begin{cases} 1, & \text{if humidity} > 50\%, \\ 0 & \text{otherwise.} \end{cases}$$

$$F(x, y) = \begin{cases} 1, & \text{the air conditioner is on,} \\ 0, & \text{otherwise.} \end{cases}$$

$x$	$y$	$F$
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

By sum-of-product expansions,

$$F(x, y) = \bar{x}y + x\bar{y} + xy = y + x\bar{y}$$

## Example 2 (P. 714):

$$x = \begin{cases} 1 & \text{if the 1st individual votes yes,} \\ 0 & \text{otherwise} \end{cases} \quad y = \begin{cases} 1 & \text{if the 2nd individual votes yes,} \\ 0 & \text{otherwise} \end{cases}$$

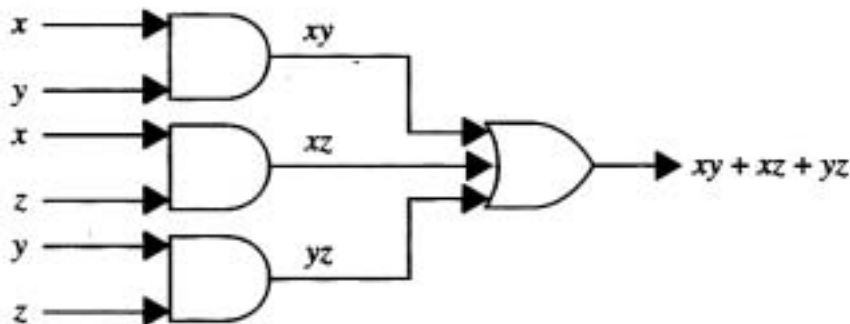
$$z = \begin{cases} 1 & \text{if the 3rd individual votes yes,} \\ 0 & \text{otherwise} \end{cases}$$

Design a circuit that determine a proposal passes.

*Solution:*

$$F(x, y, z) = \begin{cases} 1, & \text{if } x + y + z \geq 2 \text{ (ordinary addition),} \\ 0, & \text{otherwise.} \end{cases}$$

or  $F(x, y, z) = xy + xz + yz$  (Boolean operations)



**FIGURE 5 A Circuit for Majority Voting.**

**Example 3(a)** (P. 714): To design a **two** switches circuit so that flipping any one of the switches for the fixture turns the light on when it is off and turns the light off when it is on.

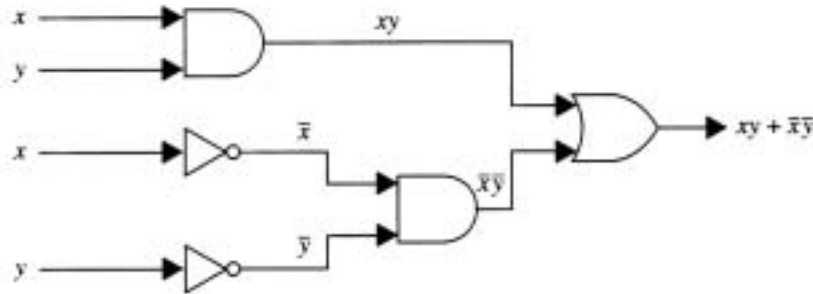
*Solution:*

Let  $x = \begin{cases} 1 & \text{if the 1st switch is closed,} \\ 0 & \text{otherwise.} \end{cases}$   $y = \begin{cases} 1 & \text{if the 2nd switch is closed,} \\ 0 & \text{otherwise.} \end{cases}$

$F(x, y) = \begin{cases} 1 & \text{if the light is on,} \\ 0 & \text{otherwise.} \end{cases} \Rightarrow F(1,1) = 1, F(1,0) = F(0,1) = 0, F(0,0) =$

TABLE 1		
$x$	$y$	$F(x, y)$
1	1	1
1	0	0
0	1	0
0	0	1

$$\Rightarrow F(x, y) = xy + \overline{x}\overline{y}$$



**Example 3(b) (P. 714):** To design a **three switches** circuit so that flipping any one of the switches for the fixture turns the light on when it is off and turns the light off when it is on.

*Solution:*

$$\text{Let } x = \begin{cases} 1 & \text{if the 1st switch is closed,} \\ 0 & \text{otherwise.} \end{cases} \quad y = \begin{cases} 1 & \text{if the 2nd switch is closed,} \\ 0 & \text{otherwise.} \end{cases}$$

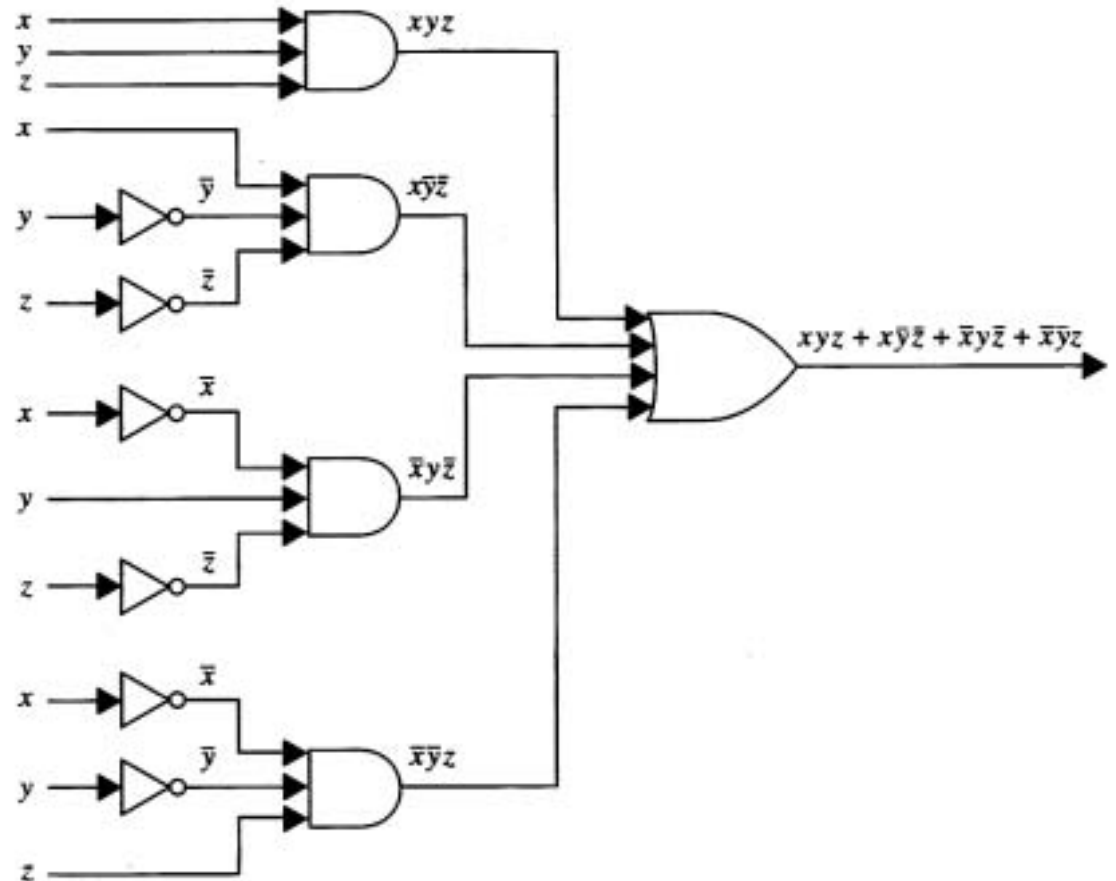
$$z = \begin{cases} 1 & \text{if the 3rd switch is closed,} \\ 0 & \text{otherwise.} \end{cases}$$

$$F(x, y, z) = \begin{cases} 1 & \text{if the light is on,} \\ 0 & \text{otherwise.} \end{cases}$$

$$\Rightarrow F(1,1,1) = 1, F(1,1,0) = F(1,0,1) = F(0,1,1) = 0, F(1,0,0) = F(0,1,0) = F(0,0,1) = 1 \\ F(0,0,0) = 0.$$

$$\Rightarrow F(x, y, z) = xyz + x\bar{y}z + x\bar{y}\bar{z} + x\bar{y}z \quad (\text{Using sum-of-products expansion.})$$

TABLE 2			
$x$	$y$	$z$	$F(x, y, z)$
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	1
0	0	1	1
0	0	0	0



**FIGURE 7** A Circuit for a Fixture Controlled by Three Switches.

# Adders

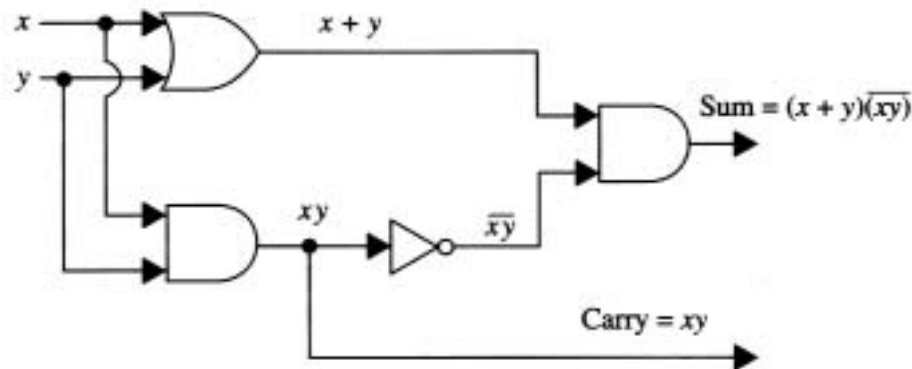
To design a circuit that carries out addition of two positive integers from their binary expansions.

The output will consist of two bits, namely  $s$  and  $c$ , where  $s$  is the sum bit and  $c$  is the carry bit. This circuit is called a **multiple output circuit**.

**TABLE 3**  
Input and  
Output for  
the Half  
Adder.

Input		Output	
$x$	$y$	$s$	$c$
1	1	0	1
1	0	1	0
0	1	1	0
0	0	0	0

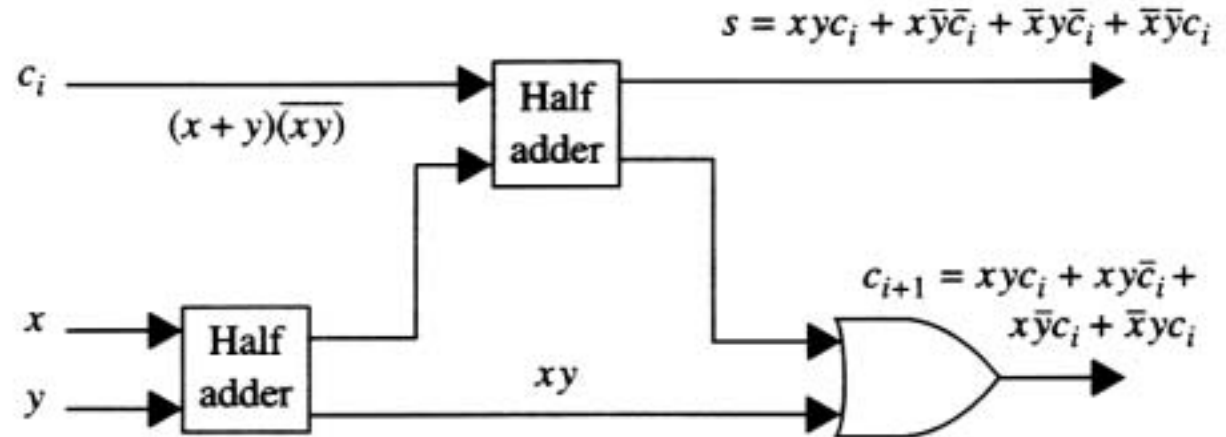
**Half adder:** It adds two bits, without considering a carry from the previous addition.



**Full adder:** It compute the sum of bit and the carry bit when two bits and carry are added.

The input are the bits  $x$  and  $y$  and the carry  $c_i$ . The outputs are the sum bit  $s$  and the new carry  $c_{i+1}$ .

TABLE 4 Input and Output for the Full Adder.				
Input			Output	
$x$	$y$	$c_i$	$s$	$c_{i+1}$
1	1	1	1	1
1	1	0	0	1
1	0	1	0	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0



**FIGURE 9 A Full Adder.**

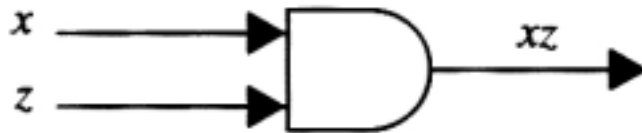
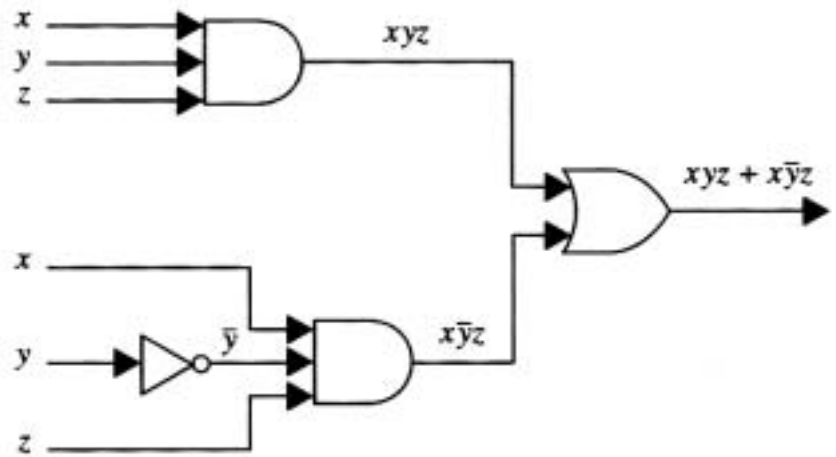


## §11.4 – Minimization of Circuits

$$\begin{aligned}xyz + x\bar{y}z &= (y + \bar{y})(xz) \\&= 1 \cdot (xz) \\&= xz.\end{aligned}$$

$$F(x, y, z) = xyz + x\bar{y}z, \quad G(x, z) = xz$$

$$\Rightarrow F(x, y, z) \equiv G(x, z).$$



# Karnaugh Maps

	$y$	$\bar{y}$
$x$	$xy$	$x\bar{y}$
$\bar{x}$	$\bar{x}y$	$\bar{x}\bar{y}$

**FIGURE 2**  
**K-maps in Two**  
**Variables.**



Maurice  
Karnaugh  
1924 -

**Example 1** (P. 721): Find the K-maps for

(a)  $xy + \bar{x}y$ , (b)  $x\bar{y} + \bar{x}y$ , and (c)  $x\bar{y} + \bar{x}y + \bar{x}\bar{y}$ .

	$y$	$\bar{y}$
$x$	1	
$\bar{x}$	1	

(a)

	$y$	$\bar{y}$
$x$		1
$\bar{x}$	1	

(b)

	$y$	$\bar{y}$
$x$		1
$\bar{x}$	1	1

(c)

**FIGURE 3** K-maps for the Sum-of-Products Expansions in Example 1.

# Karnaugh Maps

- (1) If there are 1s in two adjacent cells in the K-map, the minterm represented by these cells can be combined into a product involving just one of the variables.

e.g.  $\overline{x}y + x\overline{y} = (\overline{x} + x)y.$

	y	$\overline{y}$
x	xy	$x\overline{y}$
$\overline{x}$	$\overline{x}y$	$\overline{x}\overline{y}$

**FIGURE 2**  
**K-maps in Two**  
**Variables.**

- (2) If 1s are in all four cells, the four minterms can be combined into one term, i.e. 1 involving none of the variables.
- (3) To circle the blocks of cells that can be combined. The goal is to identify the largest possible blocks, and to cover all the 1's with the fewest blocks using the largest blocks first and always using the largest possible blocks.

# Goals of Circuit Minimization

- (1) Minimize the number of primitive Boolean logic gates needed to implement the circuit.
  - Ultimately, this also roughly minimizes the number of transistors, the chip area, and the cost.
    - Also roughly minimizes the energy expenditure
      - among traditional irreversible circuits.
  - This will be our focus.
- (2) It is also often useful to minimize the number of combinational *stages* or logical *depth* of the circuit.
  - This roughly minimizes the *delay* or *latency* through the circuit, the time between input and output.

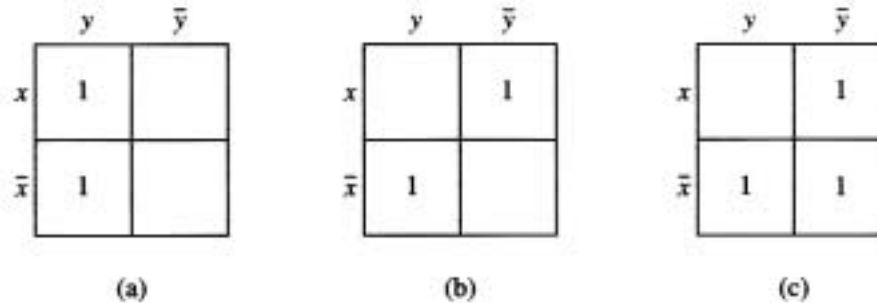
# Minimizing DNF Expressions

- Using DNF (or CNF) guarantees there is always *some* circuit that implements any desired Boolean function.
  - However, it may be far larger than needed!
- We would like to find the *smallest* sum-of-products expression that yields a given function.
  - This will yield a fairly small circuit.
  - However, circuits of other forms (not CNF or DNF) might be even smaller for complex functions.

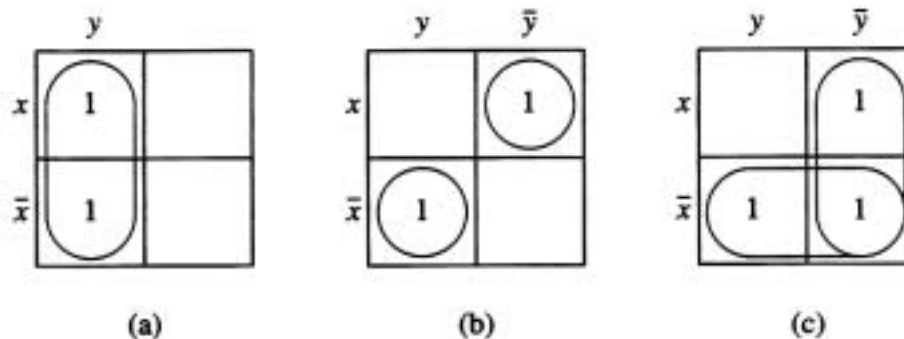
**Example 2 (P. 721):** Simplify the sum-of products expansions

(a)  $xy + \bar{x}y$ , (b)  $x\bar{y} + \bar{x}y$ , and (c)  $x\bar{y} + \bar{x}y + \bar{x}\bar{y}$ .

*Solution:*



**FIGURE 3 K-maps for the Sum-of-Products Expansions in Example 1.**



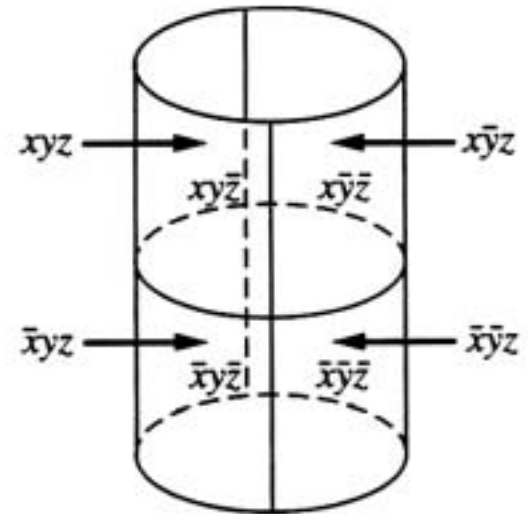
**FIGURE 4 Simplifying the Sum-of-Products Expansion from Example 1.**

$\Rightarrow$  (a)  $y$ , (b)  $x\bar{y} + \bar{x}y$ , and (c)  $\bar{x} + \bar{y}$ .

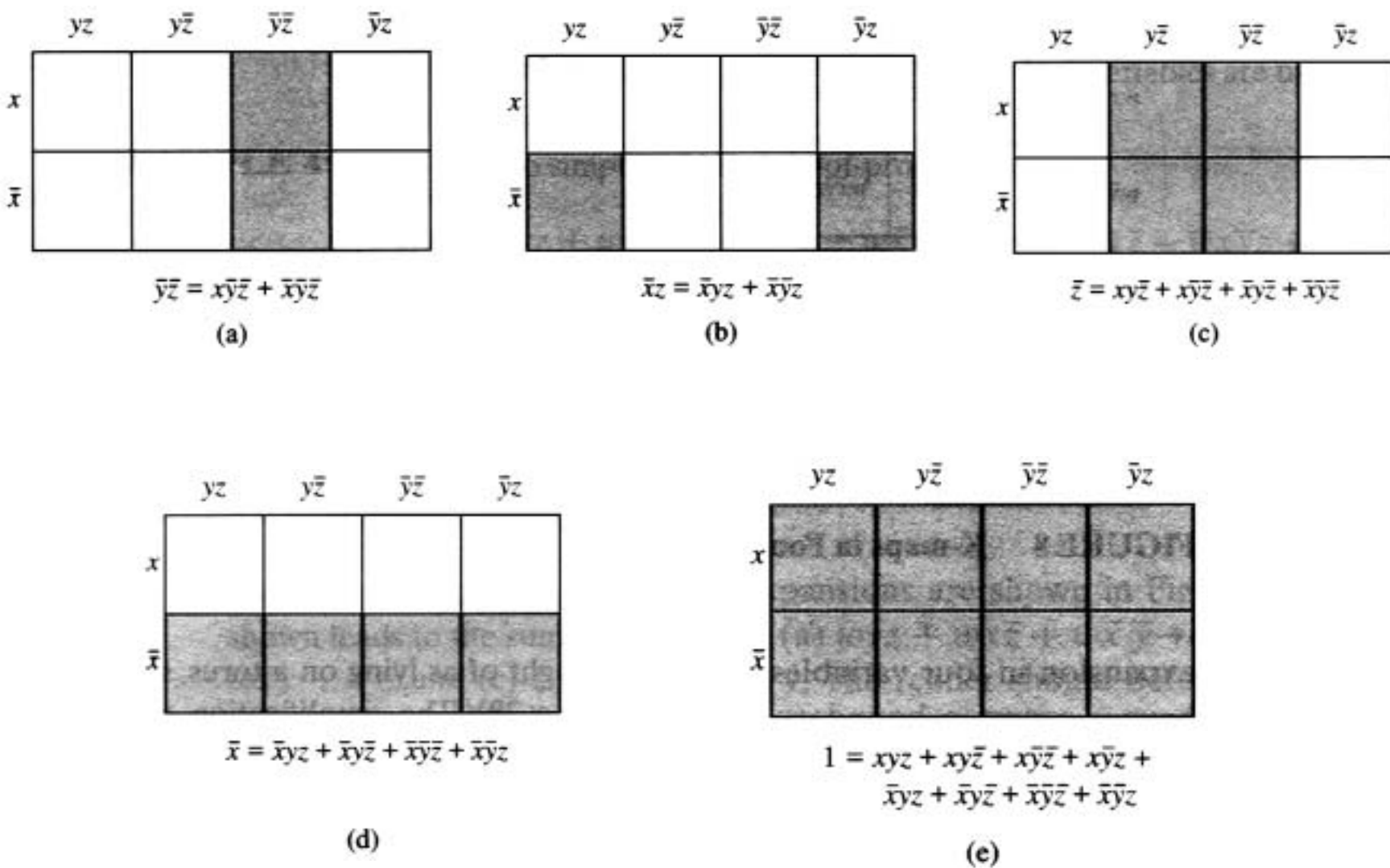
To form a K-map in three variables. Such a map can be thought of as lying on a cylinder. On the cylinder two cells have a common border if and only if they are adjacent.

	$yz$	$y\bar{z}$	$\bar{y}z$	$\bar{y}\bar{z}$
$x$	$xyz$	$xy\bar{z}$	$x\bar{y}z$	$x\bar{y}\bar{z}$
$\bar{x}$	$\bar{x}yz$	$\bar{x}y\bar{z}$	$\bar{x}\bar{y}z$	$\bar{x}\bar{y}\bar{z}$

(a)



(b)



**FIGURE 6 Blocks in K-maps in Three Variables.**



**Example 3 (P. 722):** Simplify the sum-of products expansions

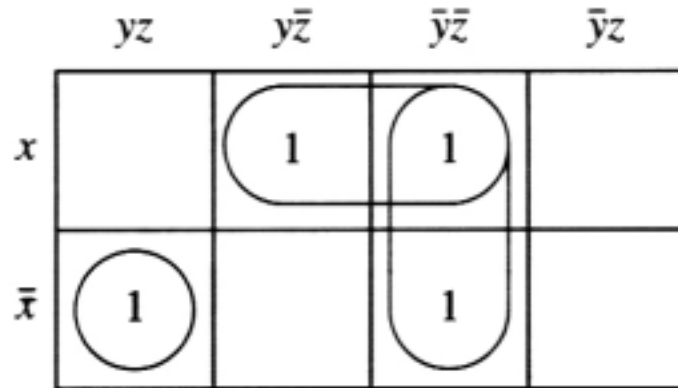
$$(a) \ x \bar{y} \bar{z} + x \bar{\bar{y}} \bar{z} + \bar{x} \bar{y} z + \bar{\bar{x}} \bar{\bar{y}} \bar{\bar{z}}$$

$$(b) \ x \bar{y} z + x \bar{\bar{y}} z + \bar{x} \bar{y} z + \bar{\bar{x}} \bar{\bar{y}} z + \bar{\bar{x}} \bar{\bar{y}} \bar{\bar{z}}$$

$$(c) \ x \bar{y} z + x \bar{y} \bar{z} + x \bar{\bar{y}} \bar{z} + x \bar{\bar{y}} z + \bar{x} \bar{y} z + \bar{\bar{x}} \bar{\bar{y}} z + \bar{\bar{x}} \bar{\bar{y}} \bar{\bar{z}}$$

$$(d) \ x \bar{y} \bar{z} + x \bar{\bar{y}} \bar{z} + \bar{x} \bar{y} z + \bar{\bar{x}} \bar{\bar{y}} z$$

*Solution:*



(a)

$$\Rightarrow \ x \bar{y} \bar{z} + x \bar{\bar{y}} \bar{z} + \bar{x} \bar{y} z + \bar{\bar{x}} \bar{\bar{y}} z = x \bar{z} + \bar{y} z + \bar{x} \bar{\bar{y}} z$$

	$yz$	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
$x$			1	1
$\bar{x}$	1		1	1

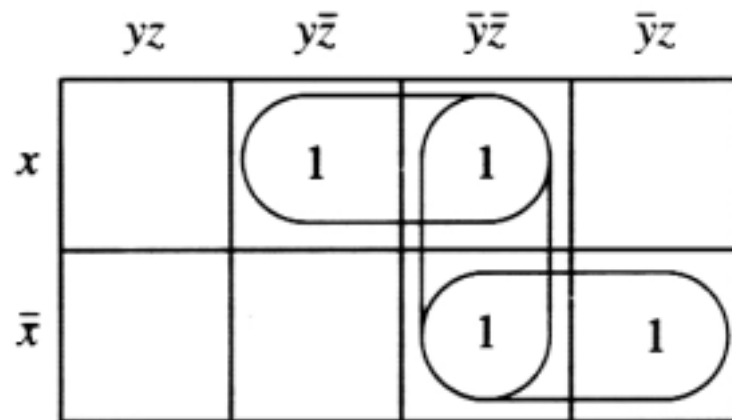
(b)

$$\Rightarrow \bar{x}\bar{y}z + x\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}z + \bar{x}\bar{y}z = \bar{y} + \bar{x}z$$

	$yz$	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
$x$	1	1	1	1
$\bar{x}$	1		1	1

(c)

$$\Rightarrow x\bar{y}z + x\bar{y}\bar{z} + x\bar{y}z + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}y\bar{z} + \bar{x}y\bar{z} = x + \bar{y} + z$$



(d)

$$\Rightarrow x \bar{y} \bar{z} + x \bar{y} z + x y \bar{z} + x y z = x \bar{z} + \bar{x} y$$

	$yz$	$y\bar{z}$	$\bar{y}z$	$\bar{y}\bar{z}$
$wx$	$wxyz$	$wxy\bar{z}$	$wx\bar{y}z$	$wx\bar{y}\bar{z}$
$w\bar{x}$	$w\bar{x}yz$	$w\bar{x}y\bar{z}$	$w\bar{x}\bar{y}z$	$w\bar{x}\bar{y}\bar{z}$
$\bar{w}x$	$\bar{w}xyz$	$\bar{w}xy\bar{z}$	$\bar{w}x\bar{y}z$	$\bar{w}x\bar{y}\bar{z}$
$\bar{w}\bar{x}$	$\bar{w}\bar{x}yz$	$\bar{w}\bar{x}y\bar{z}$	$\bar{w}\bar{x}\bar{y}z$	$\bar{w}\bar{x}\bar{y}\bar{z}$

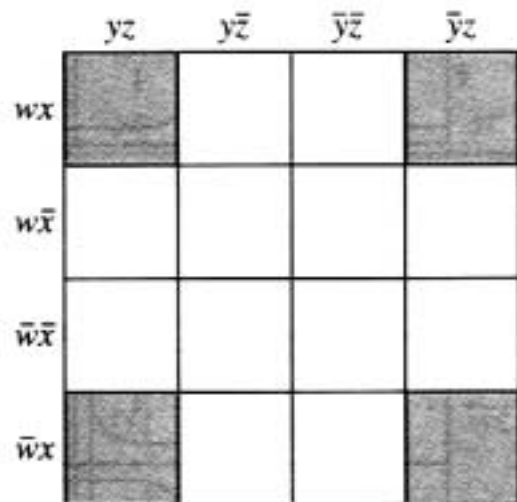
**FIGURE 8 K-maps in Four Variables.**

	$yz$	$y\bar{z}$	$\bar{y}z$	$\bar{y}\bar{z}$
$wx$				
$w\bar{x}$				
$\bar{w}x$				
$\bar{w}\bar{x}$				

$$w\bar{x}z = w\bar{x}yz + w\bar{x}\bar{y}z$$

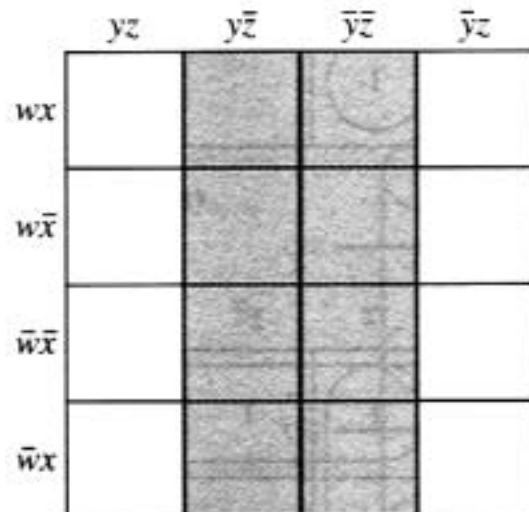
	$yz$	$y\bar{z}$	$\bar{y}z$	$\bar{y}\bar{z}$
$wx$				
$w\bar{x}$				
$\bar{w}x$				
$\bar{w}\bar{x}$				

$$\bar{w}\bar{x} = \bar{w}\bar{x}yz + \bar{w}\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}z + \bar{w}\bar{x}\bar{y}\bar{z}$$



$$x\bar{z} = wxyz + wx\bar{y}z + \bar{w}xyz + \bar{w}x\bar{y}z$$

(c)



$$\bar{z} = wxy\bar{z} + wx\bar{y}\bar{z} + w\bar{x}y\bar{z} + w\bar{x}\bar{y}\bar{z} + \bar{w}xy\bar{z} + \bar{w}x\bar{y}\bar{z} + \bar{w}\bar{x}y\bar{z} + \bar{w}\bar{x}\bar{y}\bar{z}$$

(d)

**FIGURE 9 Blocks in K-maps in Four Variables.**

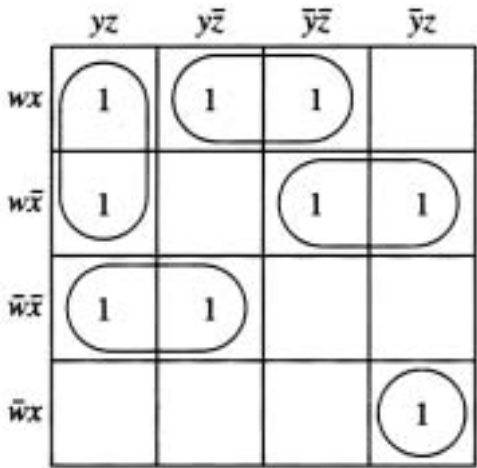
# **Example 4 (P. 725):** Simplify the sum-of products expansions

$$(a) \quad wxyz + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z + wx\bar{y}\bar{z}$$

$$(b) \quad wx\bar{y}z + wx\bar{y}z + wx\bar{y}z + wx\bar{y}z + wx\bar{y}z + wx\bar{y}z + wx\bar{y}z$$

$$(c) \quad wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z \\ + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z$$

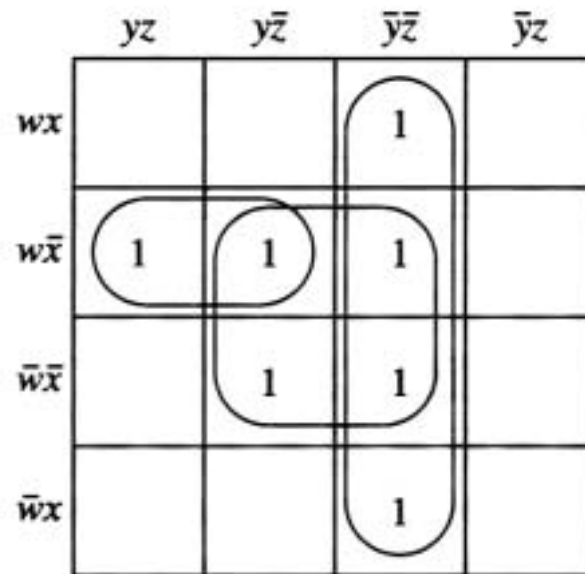
*Solution:*



(a)

$$\Rightarrow \quad wxyz + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z + wx\bar{y}\bar{z} + wx\bar{y}z + wx\bar{y}\bar{z} \\ = wyz + wx\bar{z} + wx\bar{y} + wx\bar{y} + wx\bar{y}z$$

$$(b) \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z$$



(b)

$$\Rightarrow \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z$$

$$= \overline{y}z + \overline{w}x\overline{y} + \overline{x}z$$

	$yz$	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
$wx$		1	1	
$w\bar{x}$	1	1	1	
$\bar{w}\bar{x}$		1	1	
$\bar{w}x$	1	1	1	1

(c)

$$\begin{aligned}
 \Rightarrow & \bar{w}x\bar{y}z + \bar{w}x\bar{y}\bar{z} + \bar{w}x\bar{y}z + \bar{w}x\bar{y}\bar{z} + \bar{w}x\bar{y}z + \bar{w}x\bar{y}\bar{z} + \bar{w}x\bar{y}z + \bar{w}x\bar{y}\bar{z} + \bar{w}x\bar{y}z \\
 & + \bar{w}x\bar{y}z + \bar{w}x\bar{y}\bar{z} + \bar{w}x\bar{y}z \\
 & = \bar{z} + \bar{w}x + \bar{w}x\bar{y}
 \end{aligned}$$



# Don't Care Conditions

In some circuits, some combinations of input values are not possible or never occur.

**Example 8 (P. 727):**

$$f(x) = \begin{cases} 1, & \text{if } 5 \leq x \leq 10 \\ 0, & \text{if } 0 \leq x \leq 4. \end{cases} \quad F(w, x, y, z) = \begin{cases} 1, & \text{if } 5 \leq (wxyz)_2 \leq 10, \\ 1, & \text{if } 0 \leq (wxyz)_2 \leq 4. \end{cases}$$

Construct a circuit using *OR* gates, *AND* gates, and inverters only.

TABLE 1					
<i>Digit</i>	<i>w</i>	<i>x</i>	<i>y</i>	<i>z</i>	<i>F</i>
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1

By sum- of- products expansion,

$$F(w,x,y,z) = \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z + \overline{w}x\overline{y}z.$$

**TABLE 1**

<i>Digit</i>	<i>w</i>	<i>x</i>	<i>y</i>	<i>z</i>	<i>F</i>
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1

	<i>yz</i>	<i>y<math>\overline{z}</math></i>	$\overline{y}z$	$\overline{y}\overline{z}$
<i>w</i> <i>x</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>
<i>w</i> $\overline{x}$	<i>d</i>	<i>d</i>	1	1
$\overline{w}$ $\overline{x}$				
$\overline{w}$ <i>x</i>	1	1		1

(a)

	<i>yz</i>	<i>y<math>\overline{z}</math></i>	$\overline{y}z$	$\overline{y}\overline{z}$
<i>w</i> <i>x</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>
<i>w</i> $\overline{x}$	<i>d</i>	<i>d</i>	1	1
$\overline{w}$ $\overline{x}$				
$\overline{w}$ <i>x</i>	1	1		1

(b)

	<i>yz</i>	<i>y<math>\overline{z}</math></i>	$\overline{y}z$	$\overline{y}\overline{z}$
<i>w</i> <i>x</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>
<i>w</i> $\overline{x}$	<i>d</i>	<i>d</i>	1	1
$\overline{w}$ $\overline{x}$				
$\overline{w}$ <i>x</i>	1	1		1

(c)

	<i>yz</i>	<i>y<math>\overline{z}</math></i>	$\overline{y}z$	$\overline{y}\overline{z}$
<i>w</i> <i>x</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>
<i>w</i> $\overline{x}$	<i>d</i>	<i>d</i>	1	1
$\overline{w}$ $\overline{x}$				
$\overline{w}$ <i>x</i>	1	1		1

(d)

$$\Rightarrow F(w,x,y,z) = w + xy + xz.$$

# The Quine-McCluskey Method

**K-maps** are awkward to use when there are more than four variables. Furthermore, the use of K-maps relies on visual inspection to identify terms to group.

The **Quine-McCluskey method** can be used for Boolean functions in any number of variables.

## Quine-McCluskey method

- (1) The first part finds those terms that are candidate for inclusion in minimal expansion as a Boolean sum of Boolean products.
- (2) The second part determine which of these terms to actually use.



Edward J.  
McCluskey  
1929 -



Willard Van  
Orman Quine  
1908 - 2000

**Example 9 (P.728):** Use Quine-McCluskey method to find a minimal expansion which is equivalent to

*Solution:* 
$$xyz + x\bar{y}z + \bar{x}yz + \bar{x}\bar{y}z + \bar{x}\bar{y}\bar{z}.$$

TABLE 2		
Minterm	Bit String	Number of 1s
xyz	111	3
xȳz	101	2
ȳxz	011	2
ȳȳz	001	1
ȳȳȳ	000	0

TABLE 3							
		Step 1			Step 2		
	Term	Bit String		Term	String	Term	String
1	xyz	111	(1,2)	xz	1-1	(1,2,3,4)	z
2	xȳz	101	(1,3)	yz	-11		
3	ȳxz	011	(2,4)	ȳz	-01		
4	ȳȳz	001	(3,4)	ȳȳ	0-1		
5	ȳȳȳ	000	(4,5)	ȳȳȳ	00-		

$$xyz + x\bar{y}z + x\bar{y}\bar{z} + x\bar{y}\bar{z} + x\bar{y}\bar{z}.$$

TABLE 2		
Minterm	Bit String	Number of 1s
$xyz$	111	3
$x\bar{y}z$	101	2
$\bar{x}yz$	011	2
$\bar{x}\bar{y}z$	001	1
$\bar{x}\bar{y}\bar{z}$	000	0

TABLE 4					
	$xyz$	$x\bar{y}z$	$\bar{x}yz$	$\bar{x}\bar{y}z$	$\bar{x}\bar{y}\bar{z}$
$z$	X	X	X	X	
$\bar{x}\bar{y}$				X	X

TABLE 3							
			Step 1			Step 2	
	Term	Bit String		Term	String	Term	String
1	$xyz$	111	(1,2)	$xz$	1-1	(1,2,3,4)	$z$
2	$x\bar{y}z$	101	(1,3)	$yz$	-11		
3	$\bar{x}yz$	011	(2,4)	$\bar{y}z$	-01		
4	$\bar{x}\bar{y}z$	001	(3,4)	$\bar{x}z$	0-1		
5	$\bar{x}\bar{y}\bar{z}$	000	(4,5)	$\bar{x}\bar{y}$	00-		

$$\Rightarrow xyz + x\bar{y}z + x\bar{y}\bar{z} + x\bar{y}\bar{z} + x\bar{y}\bar{z} = z + \bar{x}\bar{y}.$$

**Example 10 (P.731):** Use Quine-McCluskey method to find a minimal expansion which is equivalent to

$$\overline{w}x\overline{y}z + w\overline{x}y\overline{z} + \overline{w}x\overline{y}z + w\overline{x}y\overline{z} + \overline{w}x\overline{y}z + w\overline{x}y\overline{z}.$$

*Solution:*

<b>TABLE 5</b>		
<i><b>Term</b></i>	<i><b>Bit String</b></i>	<i><b>Number of 1s</b></i>
$wxy\overline{z}$	1110	3
$w\overline{x}yz$	1011	3
$\overline{w}xyz$	0111	3
$w\overline{x}y\overline{z}$	1010	2
$\overline{w}x\overline{y}z$	0101	2
$\overline{w}\overline{x}yz$	0011	2
$\overline{w}\overline{x}\overline{y}z$	0001	1

$$\overline{w}x y z + w \overline{x} y z + w x \overline{y} z + w x y \overline{z} + \overline{w} x y \overline{z} + \overline{w} \overline{x} y z.$$

TABLE 6								
		Step 1				Step 2		
	<i>Term</i>	<i>Bit String</i>		<i>Term</i>	<i>String</i>		<i>Term</i>	<i>String</i>
1	$wxy\overline{z}$	1110	(1,4)	$wy\overline{z}$	1-10	(3,5,6,7)	$\overline{w}z$	0- -1
2	$w\overline{x}yz$	1011	(2,4)	$w\overline{x}y$	101-			
3	$\overline{w}xyz$	0111	(2,6)	$\overline{x}yz$	-011			
4	$w\overline{x}y\overline{z}$	1010	(3,5)	$\overline{w}xz$	01-1			
5	$\overline{w}x\overline{y}z$	0101	(3,6)	$\overline{w}yz$	0-11			
6	$\overline{w}\overline{x}yz$	0011	(5,7)	$\overline{w}\overline{y}z$	0-01			
7	$\overline{w}x\overline{y}\overline{z}$	0001	(6,7)	$\overline{w}\overline{x}z$	00-1			

TABLE 7							
	$wxy\overline{z}$	$w\overline{x}yz$	$\overline{w}xyz$	$w\overline{x}y\overline{z}$	$\overline{w}x\overline{y}z$	$\overline{w}\overline{x}yz$	$\overline{w}x\overline{y}\overline{z}$
$\overline{w}z$			X		X	X	X
$wy\overline{z}$	X			X			
$w\overline{x}y$		X		X			
$\overline{x}yz$		X				X	

$$\overline{w}x y z + w \overline{x} y z + w x \overline{y} z + w x y \overline{z} + \overline{w} x y \overline{z} + \overline{w} \overline{x} y z$$

$$= w z + w y \overline{z} + w x \overline{y} \quad \text{OR} \quad \overline{w} z + w y \overline{z} + \overline{x} y z.$$