

UNIVERSIDAD NACIONAL DE CÓRDOBA

FACULTAD DE MATEMÁTICA ASTRONOMÍA, FÍSICA Y
COMPUTACIÓN.

ORGANIZACIÓN DEL COMPUTADOR

TEÓRICOS:

PABLO A. FERREYRA - NICOLAS WOLOVIC

PRÁCTICOS:

DELFINA VELEZ

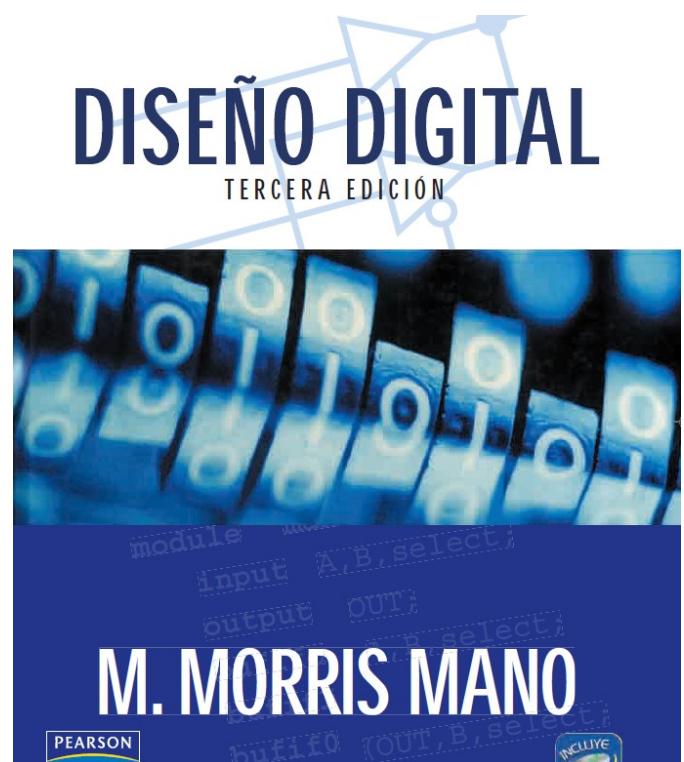
AGUSTÍN LAPROVITA

GONZALO VODANOVICK

VALENTIN BASEL

Algebra Booleana y Compuestas

- ▶ Para facilitar el seguimiento del tema nos basaremos fielmente en el siguiente libro



Algebra Booleana y Compuestas

- ▶ Entiendo que está en Biblioteca y si no por favor pídanlo a los profesores del práctico.

DISEÑO DIGITAL

TERCERA EDICIÓN

M. Morris Mano

CALIFORNIA STATE UNIVERSITY, LOS ANGELES

TRADUCCIÓN

Roberto Escalona García
Ingeniero Químico
Universidad Nacional Autónoma de México

REVISIÓN TÉCNICA

Gonzalo Duchén Sánchez
Sección de Estudios de Postgrado e Investigación
Escuela Superior de Ingeniería Mecánica y Eléctrica
Unidad Culhuacán
Instituto Politécnico Nacional



Libro de Base para este Tema

- ▶ En particular este tema se desarrolla en el capítulo 2 del libro:

2 ÁLGEBRA BOOLEANA Y COMPUERTAS LÓGICAS 33

2-1	Definiciones básicas	33
2-2	Definición axiomática del álgebra booleana	34
2-3	Teoremas y propiedades básicos del álgebra booleana	37
2-4	Funciones booleanas	40
2-5	Formas canónicas y estándar	44
2-6	Otras operaciones lógicas	51
2-7	Compuertas lógicas digitales	53
2-8	Circuitos integrados	59

Algebra Booleana y Compuertas

2-4 FUNCIONES BOOLEANAS

$$F_1 = x + y'z$$

$$F_2 = x'y'z + x'yz + xy'$$

Tabla 2-2

Tablas de verdad para F_1 y F_2

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

Algebra Booleana y Compuertas

2-4 FUNCIONES BOOLEANAS

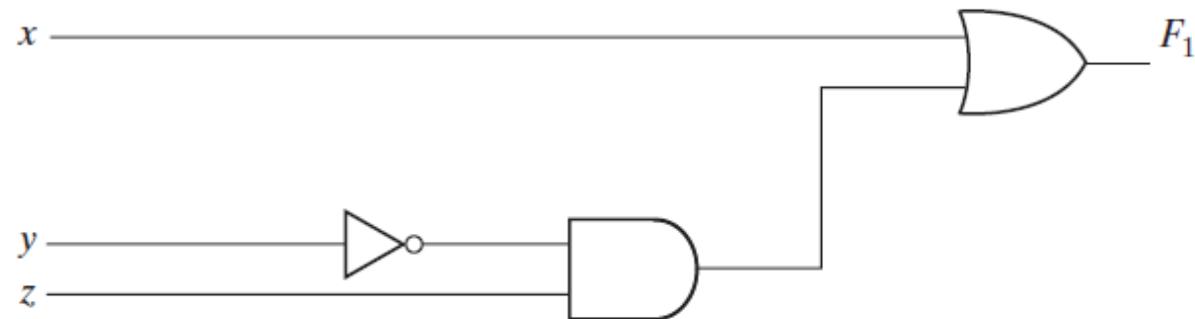
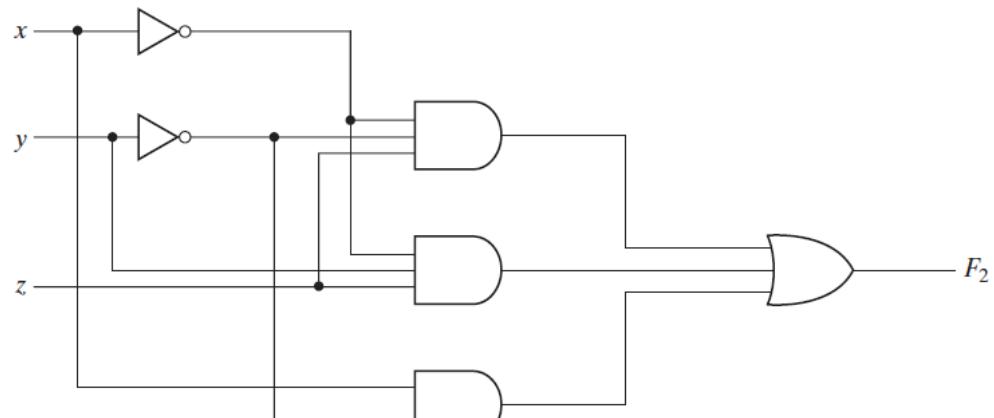


FIGURA 2-1

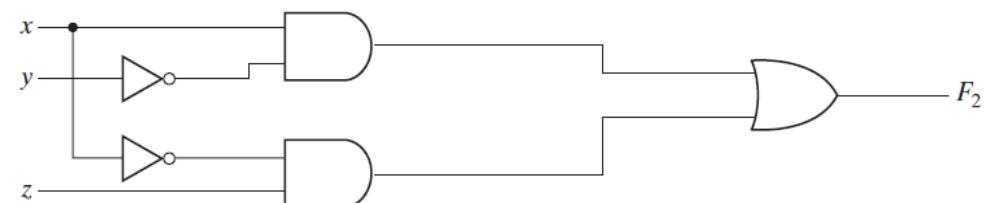
Implementación de $F_1 = x + y'z$ con compuertas

Algebra Booleana y Compuertas

2-4 FUNCIONES BOOLEANAS



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



$$b) \quad F_2 = xy' + x'z$$

FIGURA 2-2
Implementación de la función booleana F_2 con compuertas

Algebra Booleana y Compuertas

2-5 FORMAS CANÓNICAS Y ESTÁNDAR

Minitérminos y maxitérminos

Tabla 2-3

Minitérminos y maxitérminos para tres variables binarias

x	y	z	Minitérminos		Maxitérminos	
			Términos	Designación	Términos	Designación
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

Algebra Booleana y Compuertas

2-5 FORMAS CANÓNICAS Y ESTÁNDAR

Minitérminos y maxitérminos

Tabla 2-4
Funciones de tres variables

x	y	z	Función f_1	Función 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

Algebra Booleana y Compuertas

2-5 FORMAS CANÓNICAS Y ESTÁNDAR

Minitérminos y maxitérminos

$$f_1 = x'y'z + xy'z' + xyz = m_1 + m_4 + m_7$$

$$f_2 = x'yz + xy'z + xyz' + xyz = m_3 + m_5 + m_6 + m_7$$

Algebra Booleana y Compuertas

2-5 FORMAS CANÓNICAS Y ESTÁNDAR

Minitérminos y maxitérminos

$$f'_1 = x'y'z' + x'yz' + x'yz + xy'z + xyz'$$

$$\begin{aligned} f_1 &= (x + y + z)(x + y' + z)(x' + y + z')(x' + y' + z) \\ &= M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6 \end{aligned}$$

Algebra Booleana y Compuertas

2-5 FORMAS CANÓNICAS Y ESTÁNDAR

Minitérminos y maxitérminos

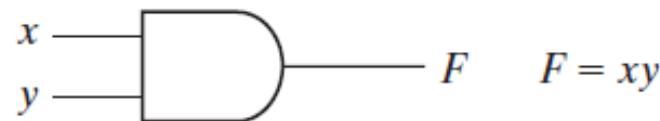
$$\begin{aligned}f_2 &= (x + y + z)(x + y + z')(x + y' + z)(x' + y + z) \\&= M_0M_1M_2M_4\end{aligned}$$

Algebra Booleana y Compuertas

2-7 COMPUERTAS LÓGICAS DIGITALES

Nombre	Símbolo gráfico	Función algebraica	Tabla de verdad
--------	-----------------	--------------------	-----------------

AND



$$F = xy$$

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

OR



$$F = x + y$$

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

Algebra Booleana y Compuertas

2-7 COMPUERTAS LÓGICAS DIGITALES

Nombre	Símbolo gráfico	Función algebraica	Tabla de verdad
--------	-----------------	--------------------	-----------------

Inversor



$$F = x'$$

x	F
0	1
1	0

Búfer



$$F = x$$

x	F
0	0
1	1

Algebra Booleana y Compuertas

2-7 COMPUERTAS LÓGICAS DIGITALES

Nombre	Símbolo gráfico	Función algebraica	Tabla de verdad
--------	-----------------	--------------------	-----------------

NAND



$$F = (xy)'$$

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

NOR



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

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

Algebra Booleana y Compuertas

2-7 COMPUERTAS LÓGICAS DIGITALES

Nombre	Símbolo gráfico	Función algebraica	Tabla de verdad
--------	-----------------	--------------------	-----------------

OR exclusivo
(XOR)



$$\begin{aligned}F &= xy' + x'y \\&= x \oplus y\end{aligned}$$

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

NOR exclusivo
o
equivalencia



$$\begin{aligned}F &= xy + x'y' \\&= (x \oplus y)'\end{aligned}$$

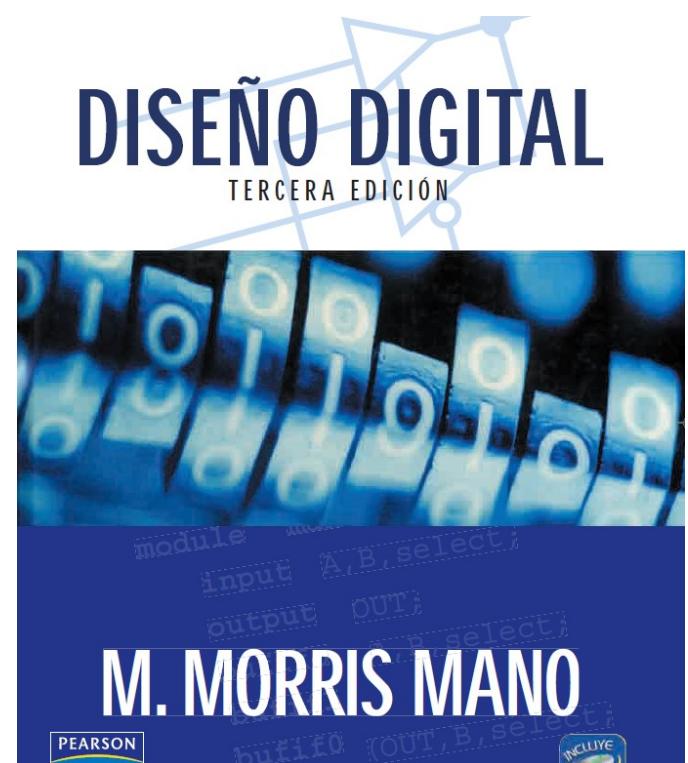
x	y	F
0	0	1
0	1	0
1	0	0
1	1	1

MINIMIZACIÓN DE FUNCIONES LÓGICAS

MÉTODO GRÁFICOS DE KARNAUGH

Libro de Base para este Tema

- ▶ Para facilitar el seguimiento del tema nos basaremos fielmente en el siguiente libro



Libro de Base para este Tema

- ▶ Entiendo que está en Biblioteca y si no por favor pídanlo a los profesores del práctico.

DISEÑO DIGITAL

TERCERA EDICIÓN

M. Morris Mano

CALIFORNIA STATE UNIVERSITY, LOS ANGELES

TRADUCCIÓN

Roberto Escalona García
Ingeniero Químico
Universidad Nacional Autónoma de México

REVISIÓN TÉCNICA

Gonzalo Duchén Sánchez
Sección de Estudios de Postgrado e Investigación
Escuela Superior de Ingeniería Mecánica y Eléctrica
Unidad Culhuacán
Instituto Politécnico Nacional



Libro de Base para este Tema

- ▶ En particular este tema se desarrolla en el capítulo 3 del libro:

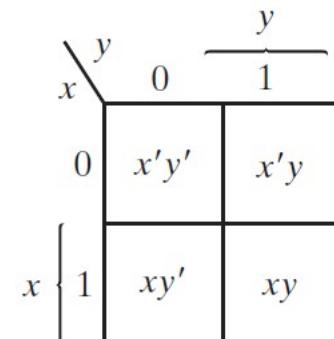


PREFACIO	ix
1 SISTEMAS BINARIOS	1
1-1 Sistemas digitales	1
1-2 Números binarios	3
1-3 Conversiones de base numérica	5
1-4 Números octales y hexadecimales	7
1-5 Complementos	9
1-6 Números binarios con signo	13
1-7 Códigos binarios	16
1-8 Almacenamiento binario y registros	24
1-9 Lógica binaria	27
2 ÁLGEBRA BOOLEANA Y COMPUERTAS LÓGICAS	33
2-1 Definiciones básicas	33
2-2 Definición axiomática del álgebra booleana	34
2-3 Teoremas y propiedades básicos del álgebra booleana	37
2-4 Funciones booleanas	40
2-5 Formas canónicas y estándar	44
2-6 Otras operaciones lógicas	51
2-7 Compuertas lógicas digitales	53
2-8 Circuitos integrados	59
3 MINIMIZACIÓN EN EL NIVEL DE COMPUERTAS	64
3-1 El método del mapa	64
3-2 Mapa de cuatro variables	70

Mapa de 2 Variables

m_0	m_1
m_2	m_3

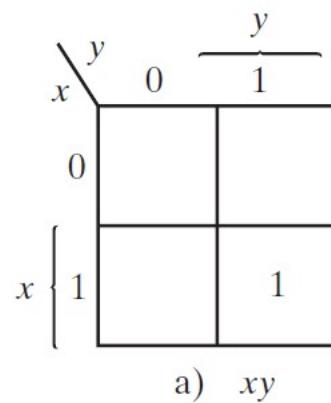
a)



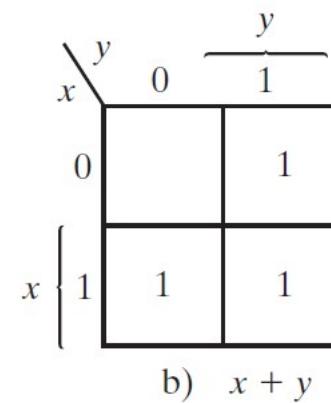
b)

FIGURA 3-1
Mapa de dos variables

Mapa de 2 Variables



a) xy



b) $x + y$

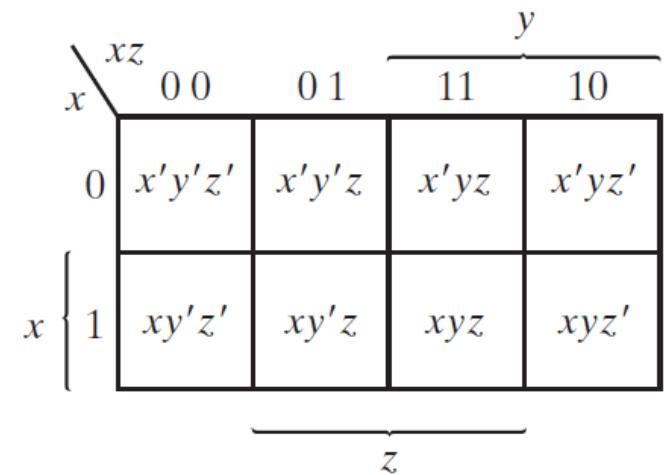
FIGURA 3-2
Representación de funciones en el mapa

Mapa de 3 Variables

66 Capítulo 3 Minimización en el nivel de compuertas

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

a)



b)

FIGURA 3-3
Mapa de tres variables

Mapa de 3 Variables

Sección 3-1 El método del mapa

67

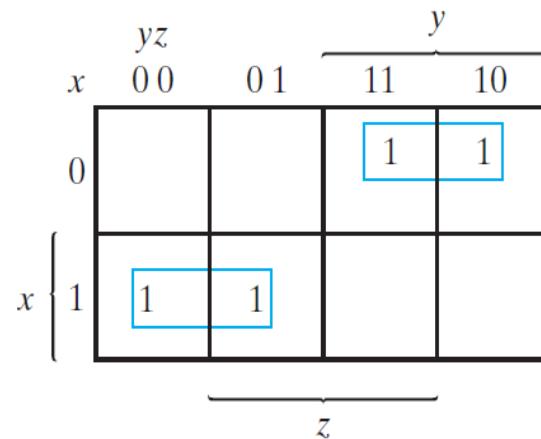


FIGURA 3-4

Mapa para el ejemplo 3-1; $F(x, y, z) = \Sigma(2, 3, 4, 5) = x'y + xy'$

EJEMPLO 3-1

Simplifique la función booleana

$$F(x, y, z) = \Sigma(2, 3, 4, 5)$$

Mapa de 3 Variables

68 Capítulo 3 Minimización en el nivel de compuertas

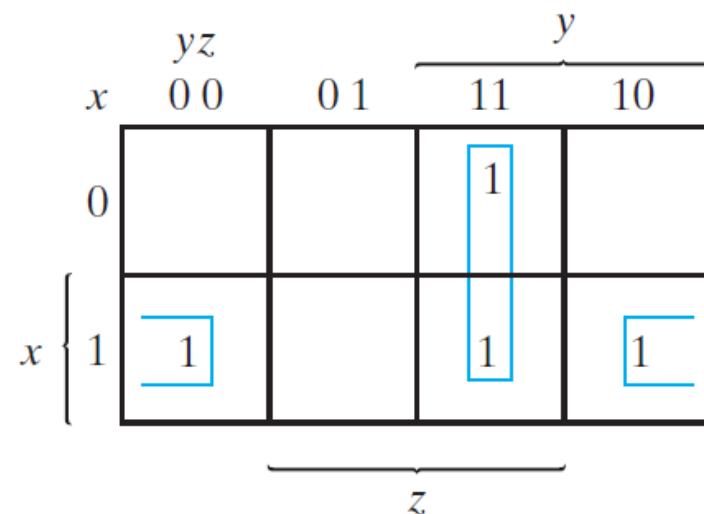


FIGURA 3-5

Mapa para el ejemplo 3-2; $F(x, y, z) = \Sigma(3, 4, 6, 7) = yz + xz'$

Mapa de 3 Variables

Sección 3-1 El método del mapa

69

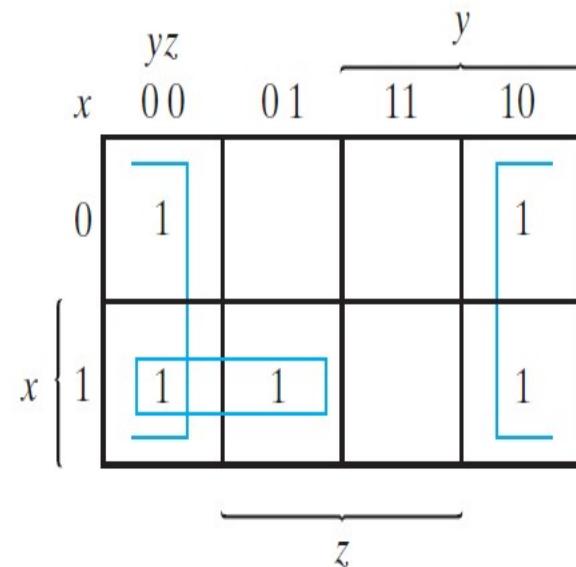


FIGURA 3-6

Mapa para el ejemplo 3-3; $F(x, y, z) = \Sigma(0, 2, 4, 5, 6) = z' + xy'$

Mapa de 3 Variables

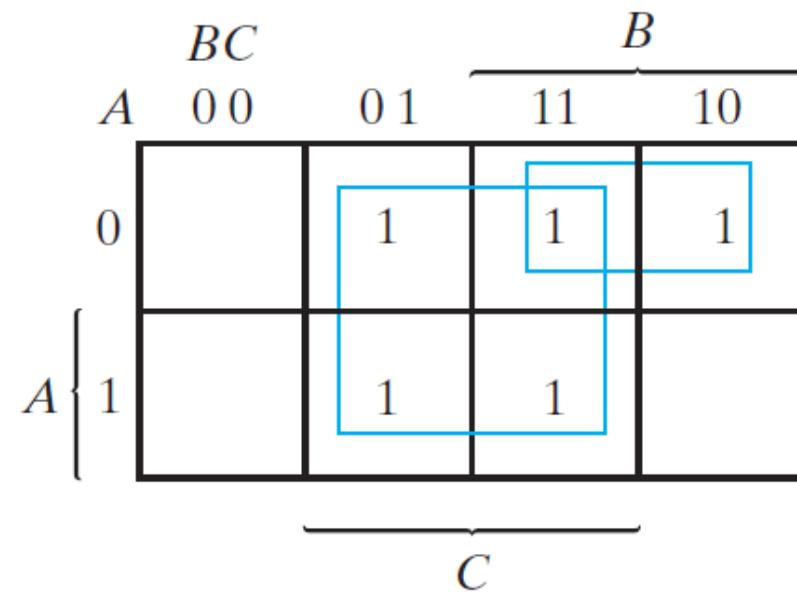


FIGURA 3-7

Mapa para el ejemplo 3-4; $A'C + A'B + AB'C + BC = C + A'B$

Mapa de 4 Variables

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

a)

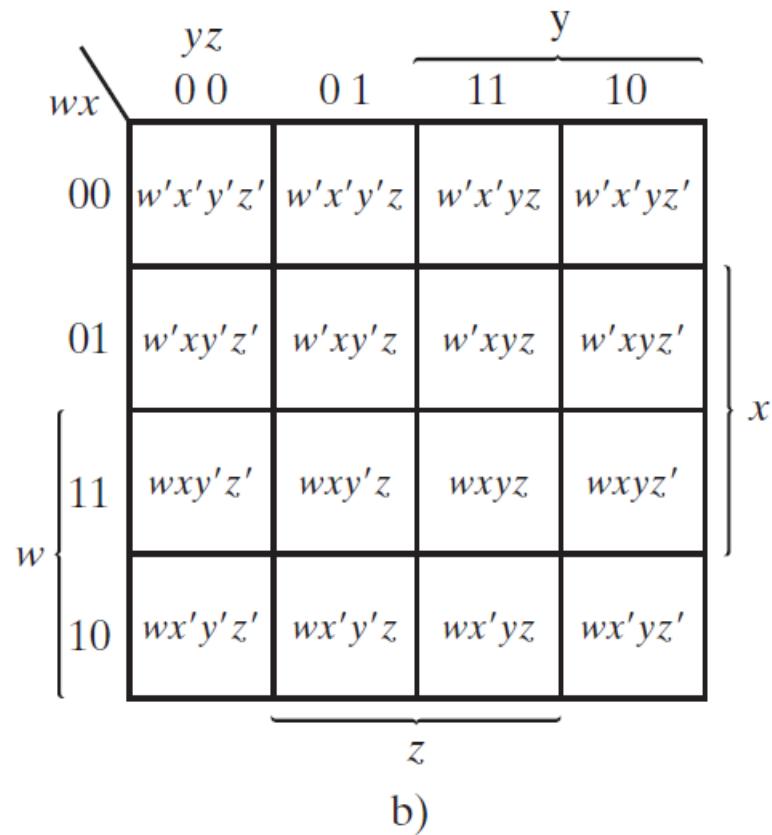


FIGURA 3-8
Mapa de cuatro variables

Mapa de 4 Variables

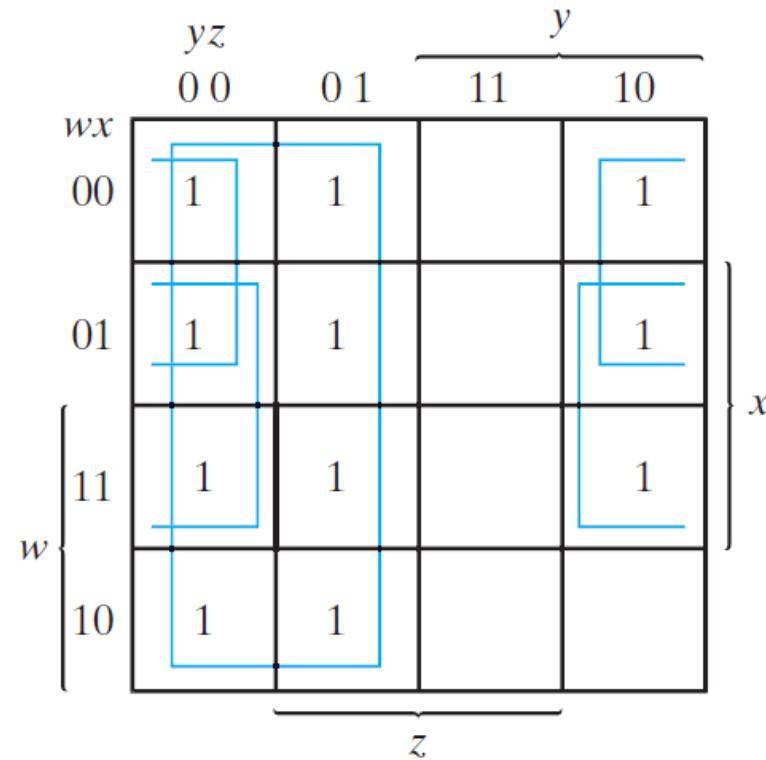


FIGURA 3-9

Mapa para el ejemplo 3-5; $F(w, x, y, z) = \Sigma(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$
 $= y' + w'z' + xz'$

Mapa de 4 Variables

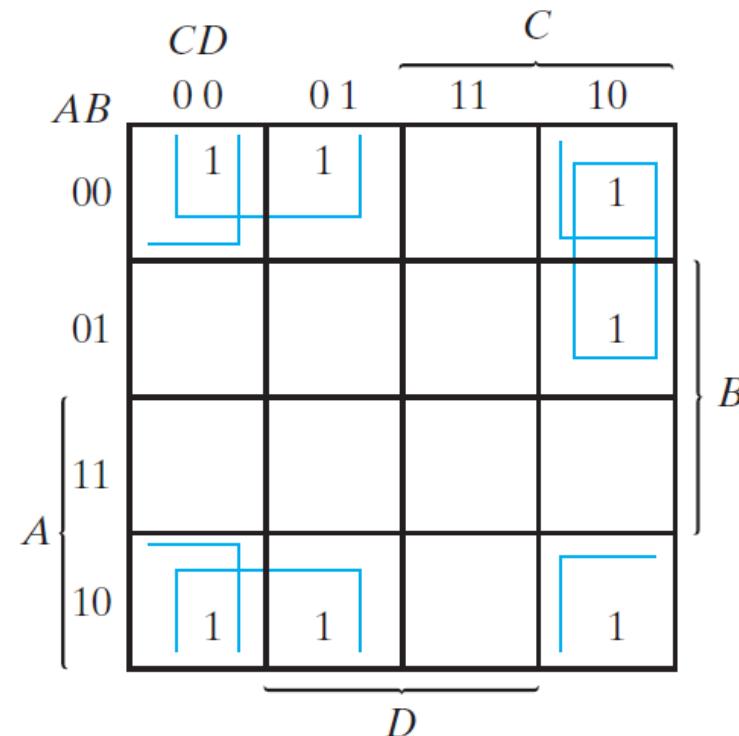
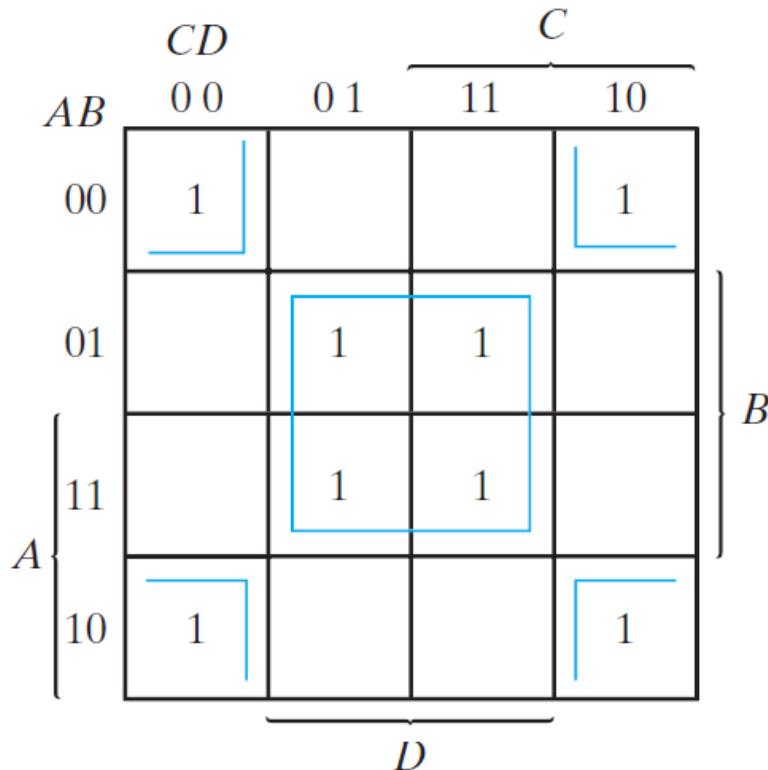


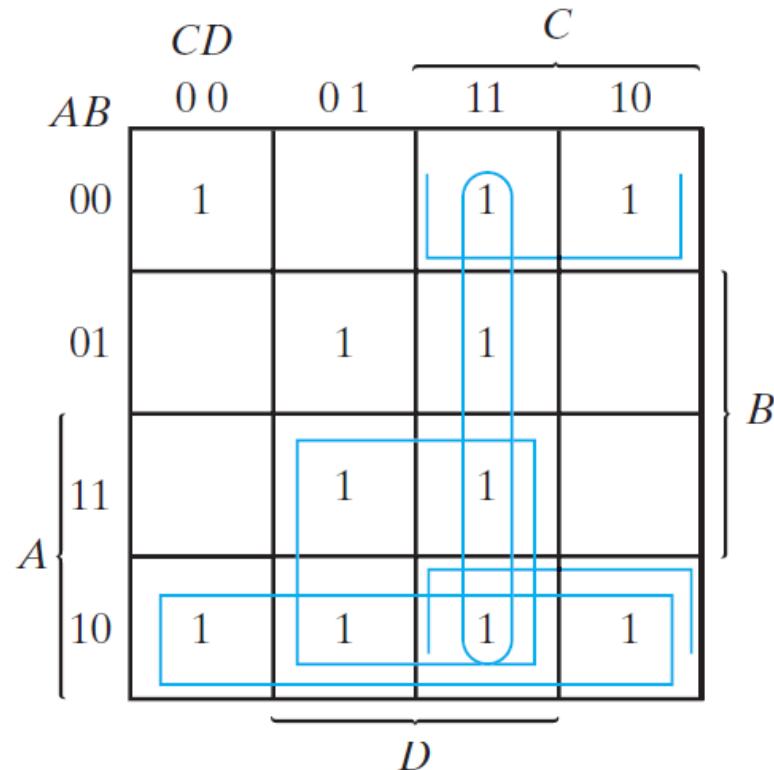
FIGURA 3-10

Mapa para el ejemplo 3-6; $A'B'C' + B'CD' + A'BCD' + AB'C' = B'D' + B'C' + A'CD'$

Mapa de 4 Variables



a) Implicantes primos esenciales
BD y B'D'



b) Implicantes primos CD, B'C
AD y AB'

FIGURA 3-11

Simplificación empleando implicantes primos

Mapa de 5 Variables

Sección 3-3 Mapa de cinco variables

75

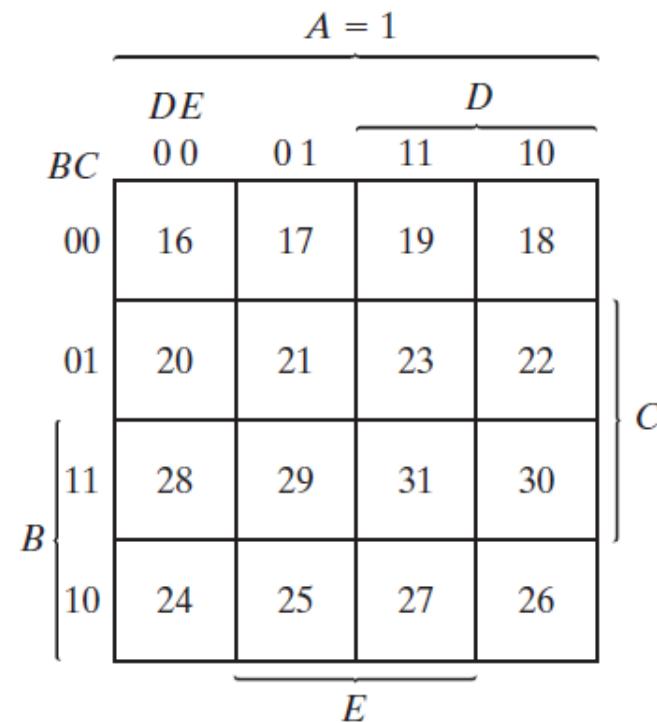
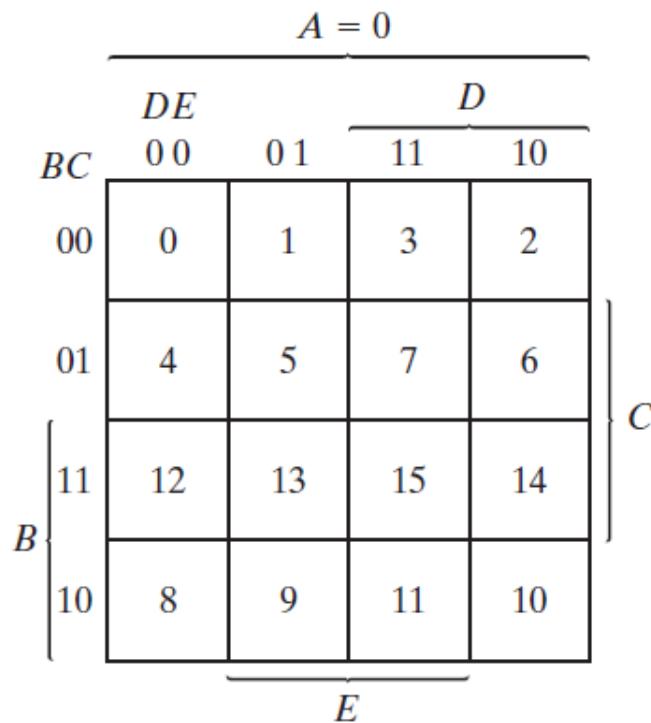


FIGURA 3-12
Mapa de cinco variables

Mapa de 5 Variables

76

Capítulo 3 Minimización en el nivel de compuertas

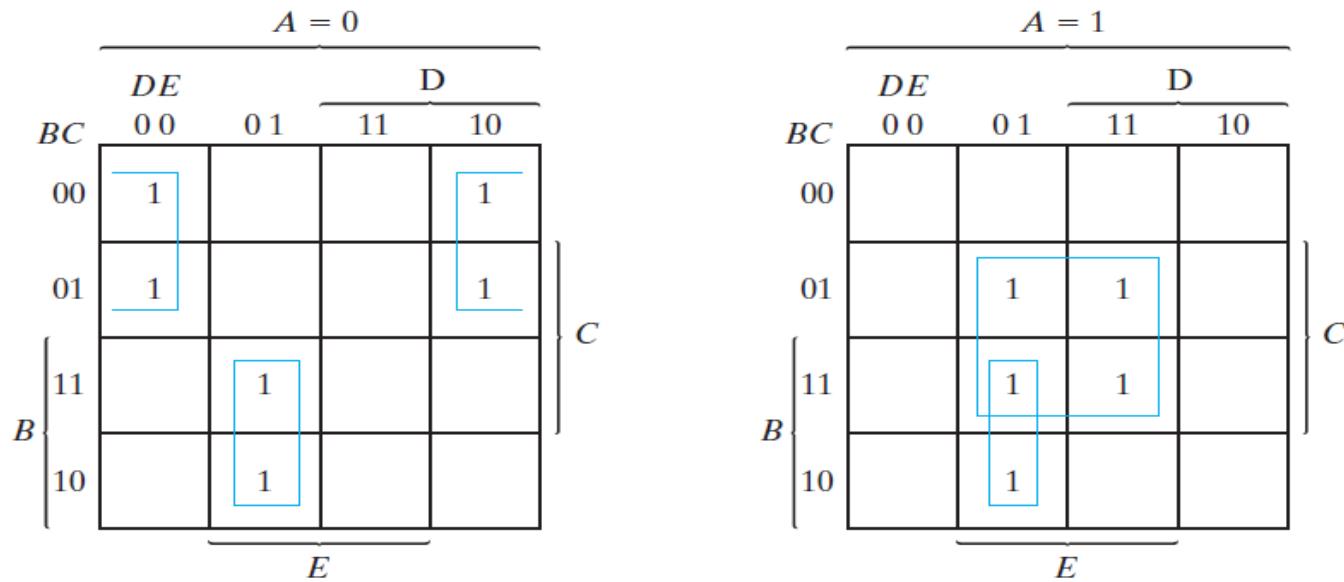


FIGURA 3-13

Mapa para el ejemplo 3-7; $F = A'B'E' + BD'E + ACE$

EJEMPLO 3-7

Simplifique la función booleana

$$F(A, B, C, D, E) = (0, 2, 4, 6, 9, 13, 21, 23, 25, 29, 31)$$

Suma de Productos y Producto de Sumas

- ▶ Es una aplicación directa de De Morgan

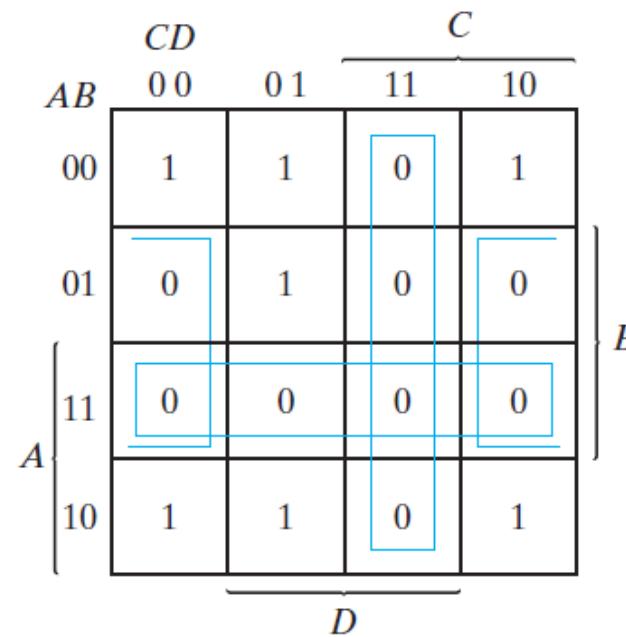


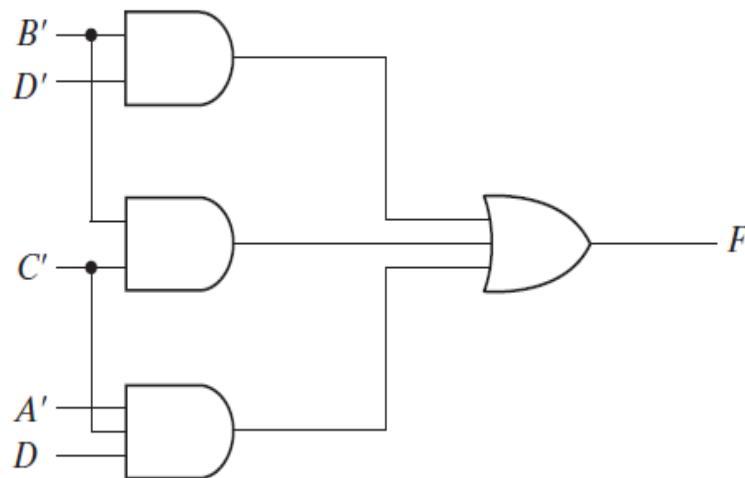
FIGURA 3-14

Mapa para el ejemplo 3-8; $F(A, B, C, D) = \sum(0, 1, 2, 5, 8, 9, 10)$
 $= B'D' + B'C' + A'C'D = (A' + B')(C' + D')(B' + D)$

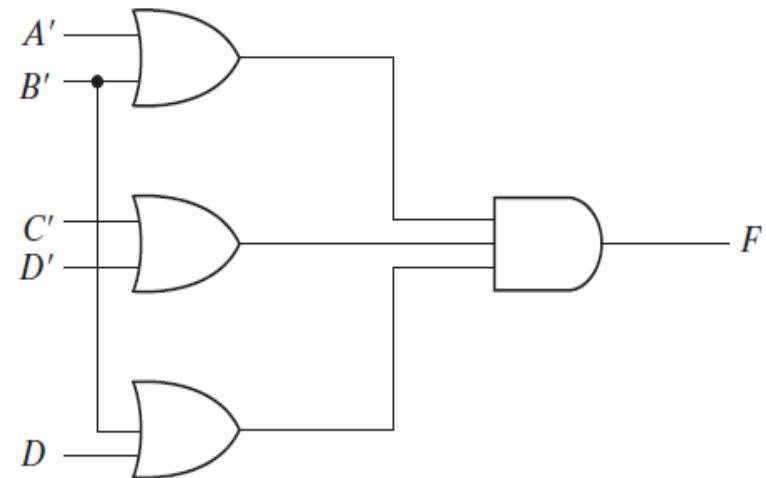
Suma de Productos y Producto de Sumas

78

Capítulo 3 Minimización en el nivel de compuertas



$$a) F = B'D' + B'C' + A'C'D$$



$$b) F = (A' + B') (C' + D') (B' + D)$$

FIGURA 3-15

Implementación con compuertas de la función del ejemplo 3-8

Condiciones Sin Cuidado o de Indiferencia (x)

Sección 3-5 Condiciones de indiferencia

81

		yz		y	
		00	01	11	10
wx		X	1	1	X
w	00	0	X	1	0
	01	0	0	1	0
	11	0	0	1	0
	10	0	0	1	0

$$\text{a) } F = yz + w'x'$$

		yz		y	
		00	01	11	10
wx		X	1	1	X
w	00	0	X	1	0
	01	0	0	X	0
	11	0	0	1	0
	10	0	0	1	0

$$\text{b) } F = yz + w'z$$

FIGURA 3-17

Ejemplo con condiciones de indiferencia

Implementación con NAND y NOR

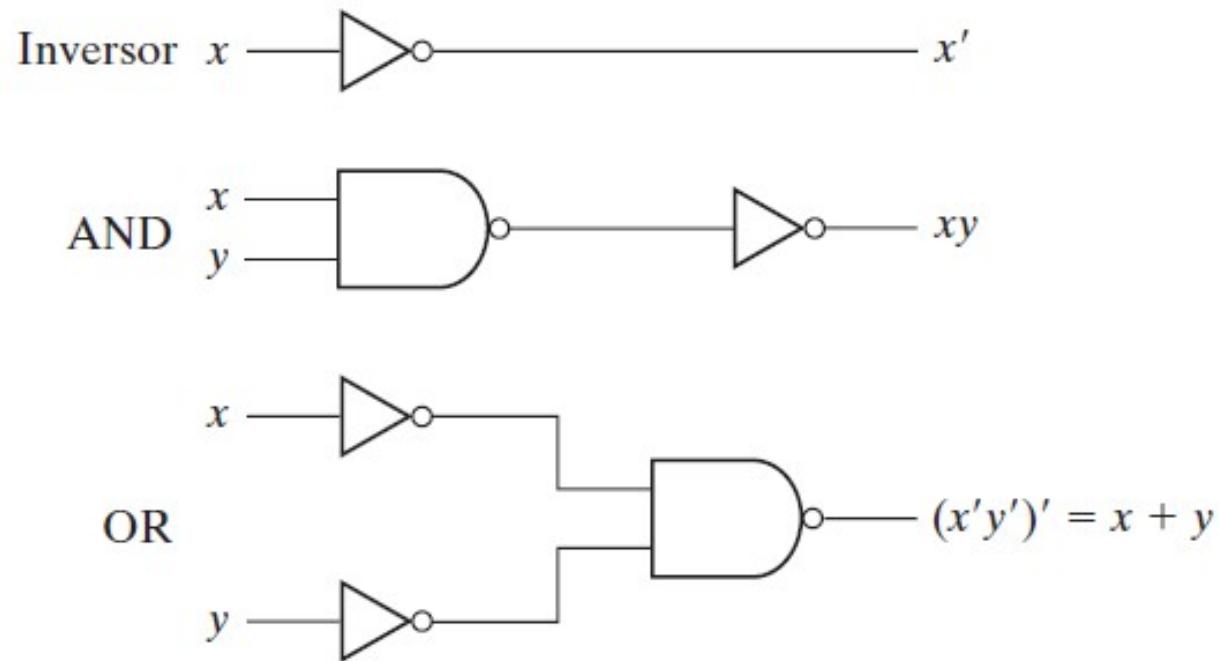
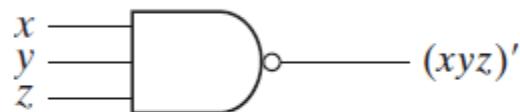


FIGURA 3-18
Operaciones lógicas con compuertas NAND

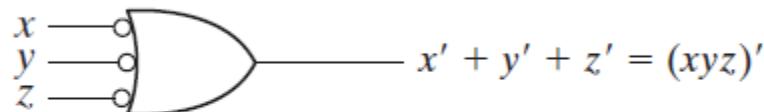
Implementación con NAND y NOR

Sección 3-6 Implementación con NAND y NOR

83



a) AND-invertir



b) Invertir-OR

FIGURA 3-19

Dos símbolos gráficos para la compuerta NAND

Implementación con NAND y NOR

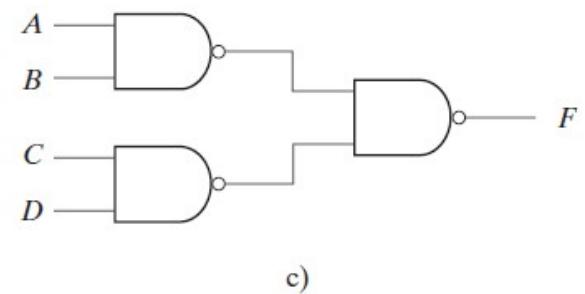
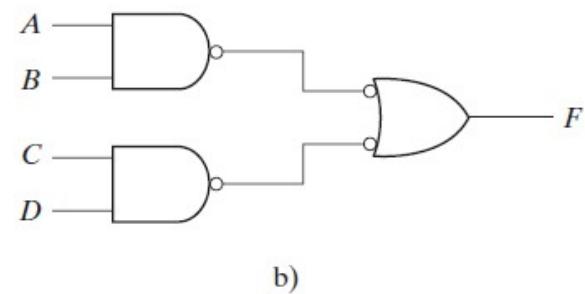
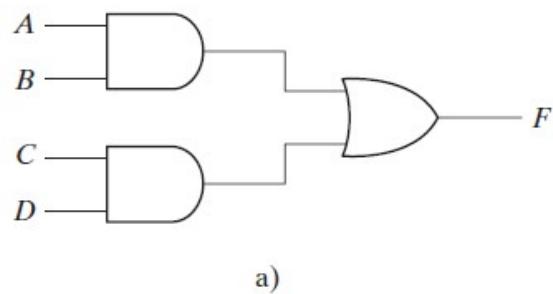


FIGURA 3-20
Tres formas de implementar $F = AB + CD$

Implementación con Nand y Nor

- ▶ También es una aplicación directa de De Morgan

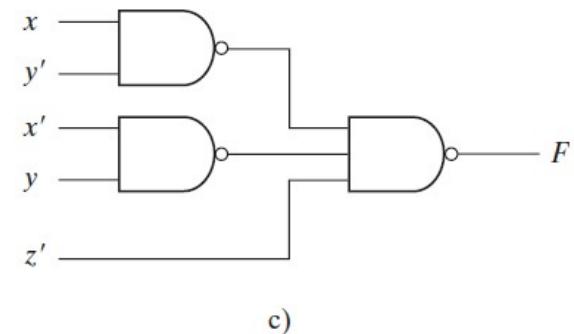
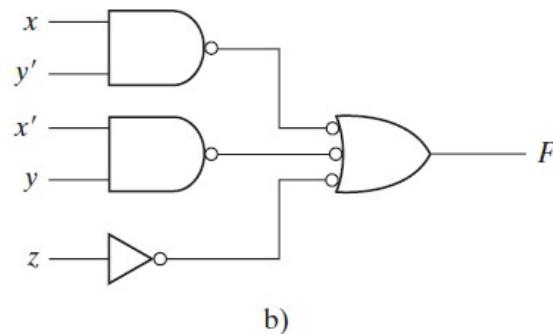
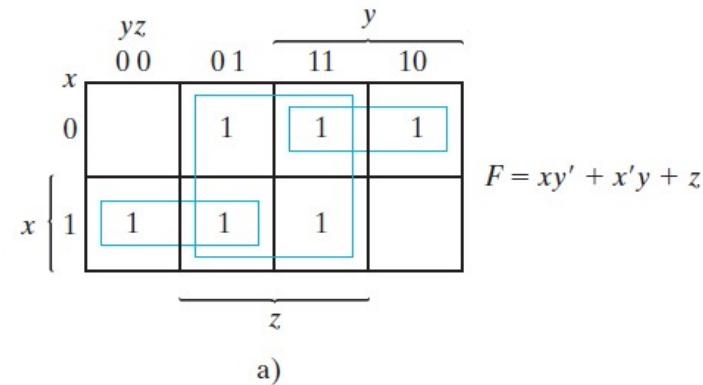
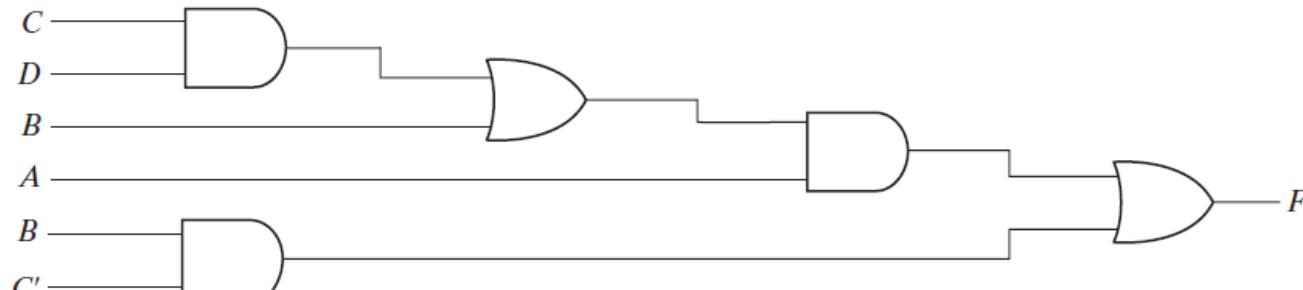


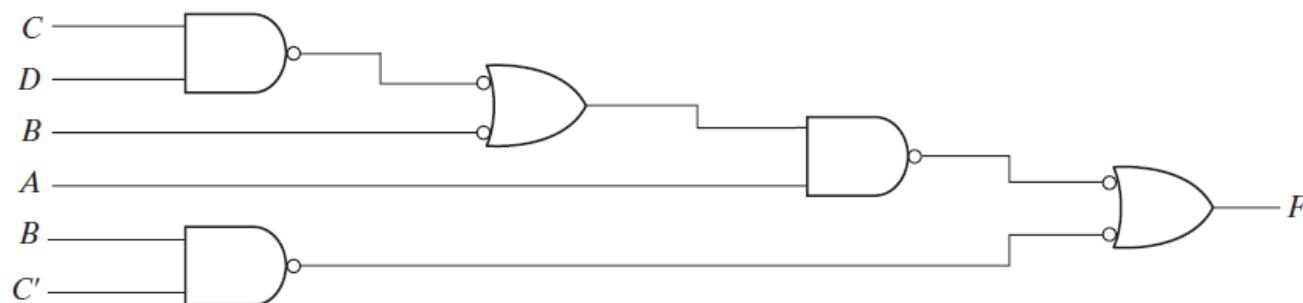
FIGURA 3-21
Solución del ejemplo 3-10

Implementación con Nand y Nor

86 Capítulo 3 Minimización en el nivel de compuertas



a) Compuertas AND-OR



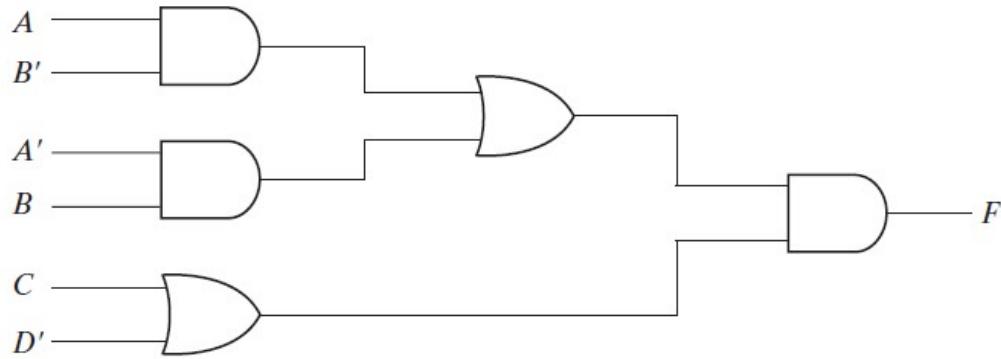
b) Compuertas NAND

FIGURA 3-22

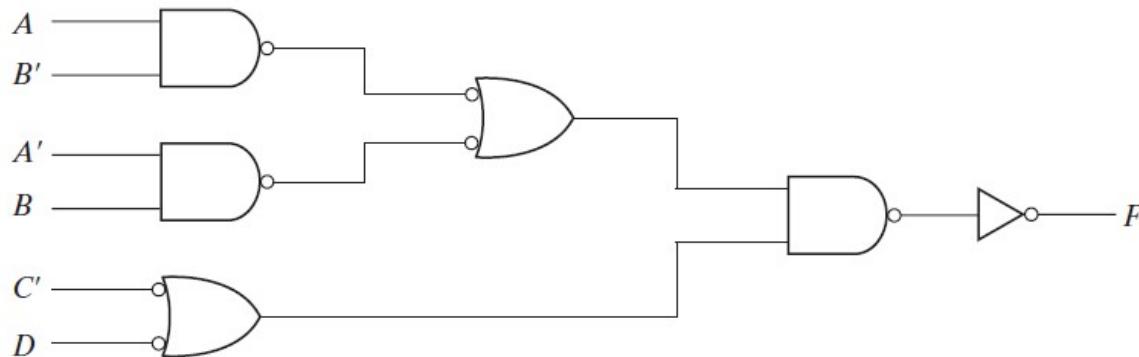
Implementación de $F = A(CD + B) + BC'$

Implementación con Nand y Nor

Sección 3-6 Implementación con NAND y NOR 87



a) Compuertas AND-OR



b) Compuertas NAND

FIGURA 3-23

Implementación de $F = (AB' + A'B)(C + D')$

Implementación con Nand y Nor

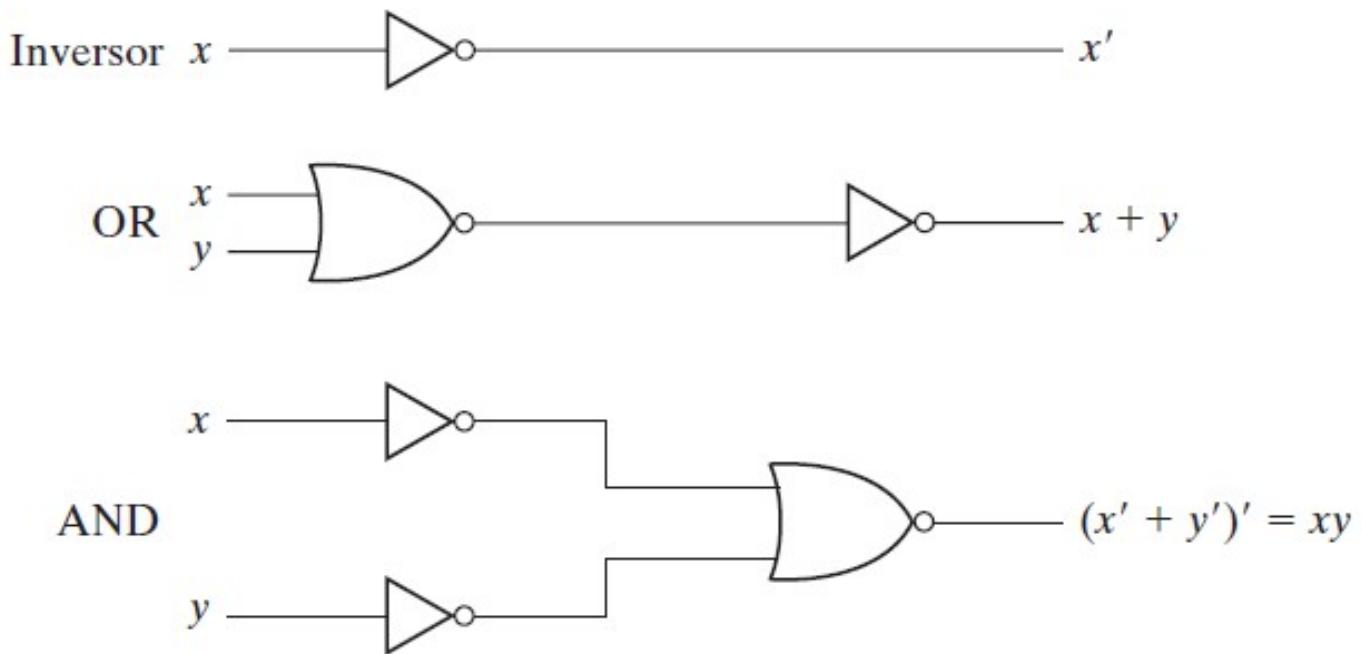


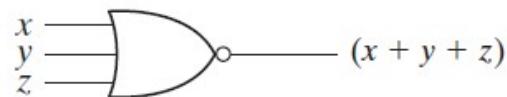
FIGURA 3-24

Operaciones lógicas con compuertas NOR

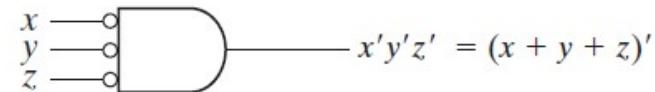
Implementación con Nand y Nor

88

Capítulo 3 Minimización en el nivel de compuertas



a) OR-invertir



b) invertir-AND

FIGURA 3-25

Dos símbolos gráficos para la compuerta NOR

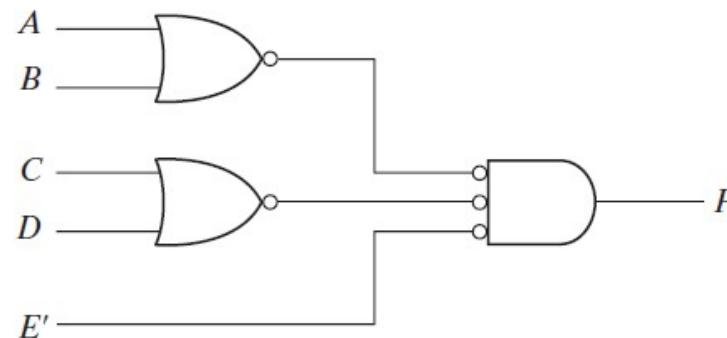


FIGURA 3-26

Implementación de $F = (A + B)(C + D)E$

Otras Implementaciones

Sección 3-7 Otras implementaciones de dos niveles

89

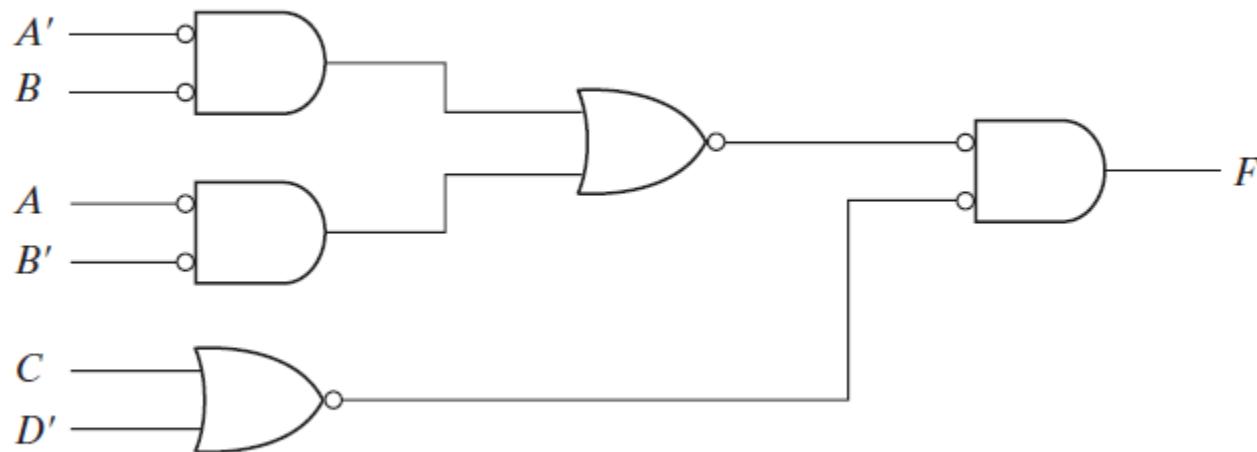


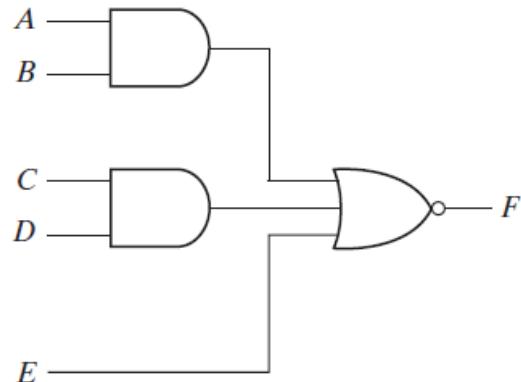
FIGURA 3-27

Implementación de $F = (AB' + A'B)(C + D')$ con compuertas NOR

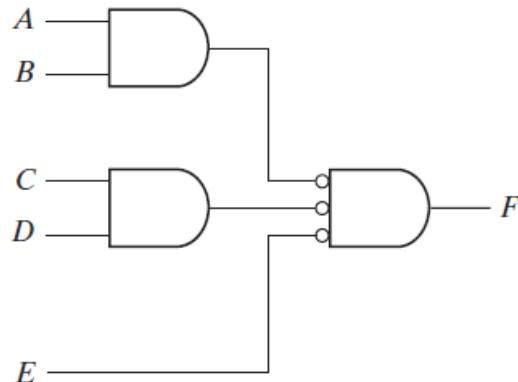
Otras Implementaciones

Sección 3-7 Otras implementaciones de dos niveles

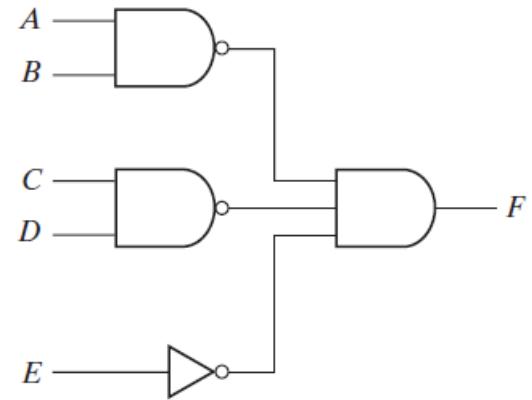
91



a) AND-NOR



b) AND-NOR



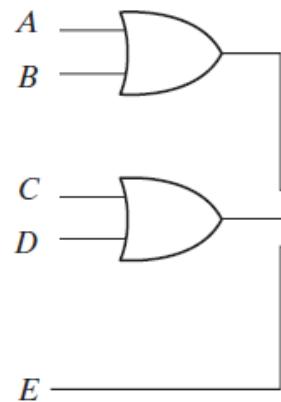
c) NAND-AND

FIGURA 3-29

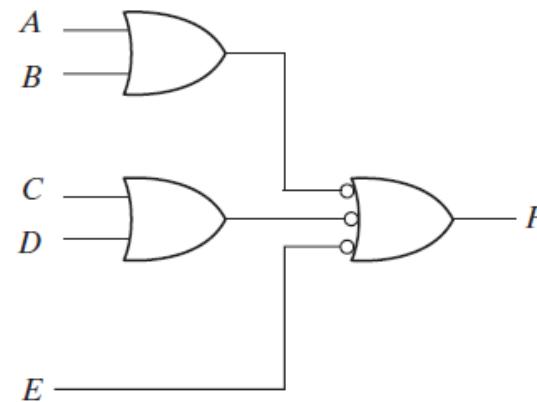
Circuitos AND-OR-INVERT; $F = (AB + CD + E)'$

Otras Implementaciones

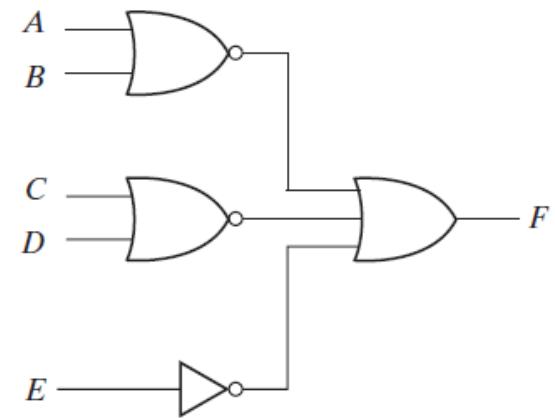
$$F = [(A + B)(C + D)E]'$$



a) OR-NAND



b) OR-NAND



c) NOR-OR

FIGURA 3-30

Circuitos OR-AND-INVERT; $F = [(A + B)(C + D)E]'$

Función O-Exclusivo

3-8 FUNCIÓN OR EXCLUSIVO

La función OR exclusivo (XOR), denotada por el símbolo \oplus , es una operación lógica que efectúa la operación booleana siguiente:

$$x \oplus y = xy' + x'y$$

Es igual a 1 si sólo x es igual a 1 o sólo y es igual a 1, pero no si ambas son 1. El NOR exclusivo, también llamado equivalencia, realiza la operación booleana siguiente:

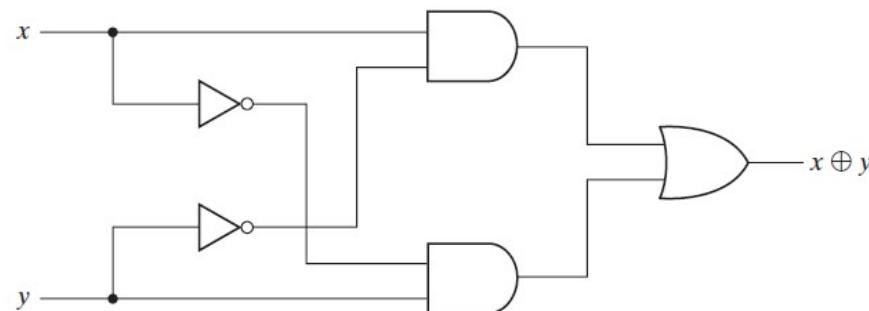
$$(x \oplus y)' = xy + x'y'$$

Es igual a 1 si tanto x como y son 1 o si ambas son 0. Se puede demostrar que el NOR exclusivo es el complemento del OR exclusivo con la ayuda de una tabla de verdad o por manipulación algebraica:

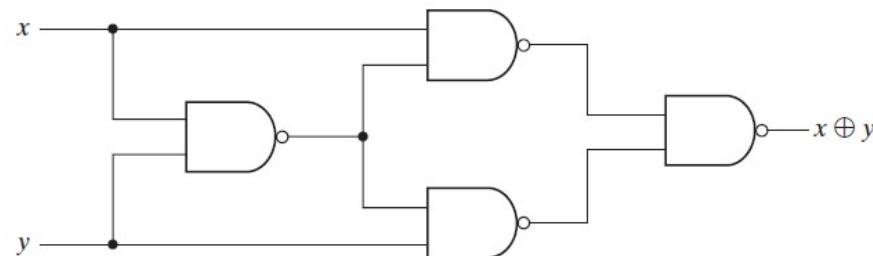
$$(x \oplus y)' = (xy' + x'y)' = (x' + y)(x + y') = xy + x'y'$$

Función O-Exclusivo

Sección 3-8 Función OR exclusivo 95



a) Con compuertas AND-OR-NOT



b) Con compuertas NAND

FIGURA 3-32
Implementaciones del OR exclusivo

Funciones Par e Impar

96 Capítulo 3 Minimización en el nivel de compuertas

		BC		B	
		00	01	11	10
A					
0			1		1
A	1	1		1	

C

a) Función impar
 $F = A \oplus B \oplus C$

		BC		B	
		00	01	11	10
A					
0		1			1
A	1			1	
			1		1

C

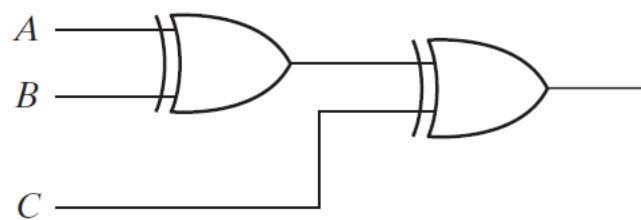
b) Función par
 $F = (A \oplus B \oplus C)'$

FIGURA 3-33

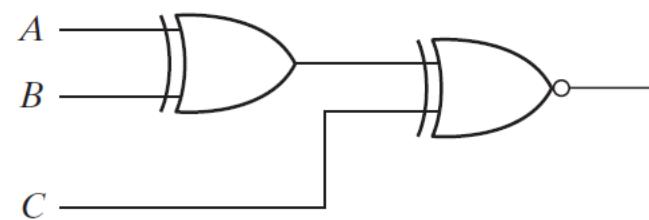
Mapa para una función OR exclusivo de tres variables

Funciones Par e Impar

$$\begin{aligned}A \oplus B \oplus C &= (AB' + A'B)C' + (AB + A'B')C \\&= AB'C' + A'BC' + ABC + A'B'C \\&= \Sigma(1, 2, 4, 7)\end{aligned}$$



a) Función impar de tres entradas

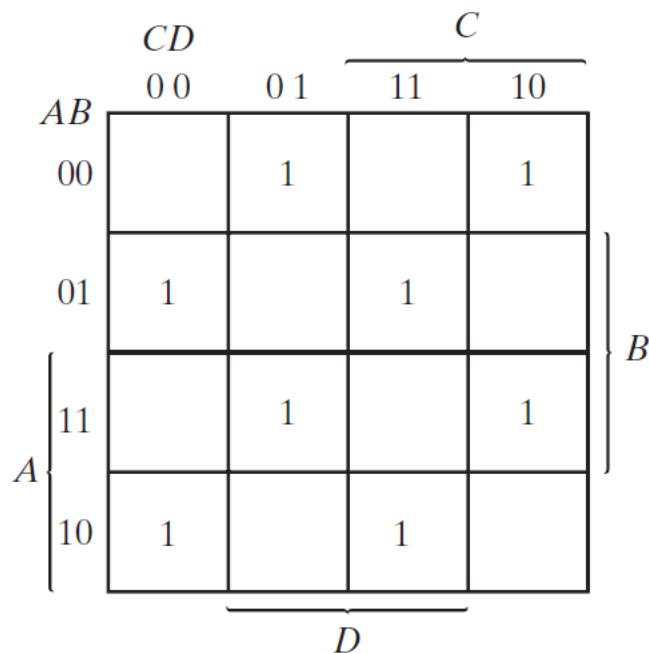


b) Función par de tres entradas

FIGURA 3-34

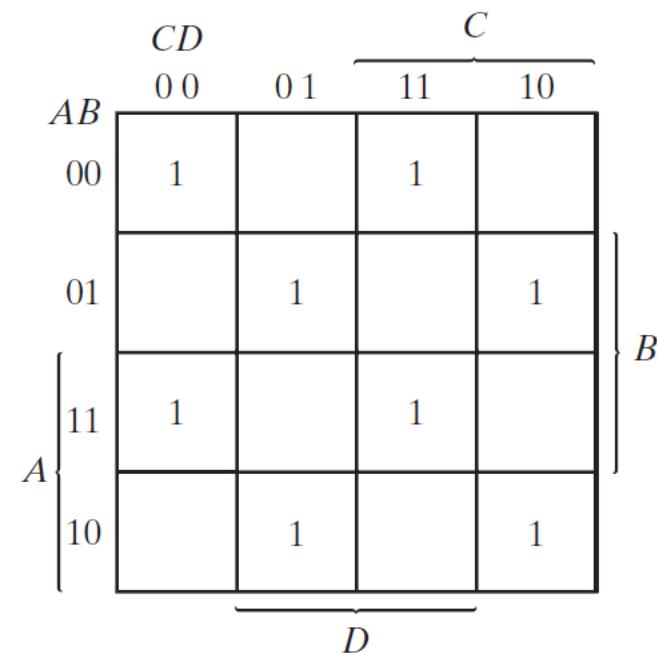
Diagrama lógico de funciones impar y par

Función Par e Impar



a) Función impar

$$F = A \oplus B \oplus C \oplus D$$



b) Función par

$$F = (A \oplus B \oplus C \oplus D)'$$

FIGURA 3-35

Mapa de una función OR exclusivo de cuatro variables

Función Par e Impar

$$\begin{aligned} A \oplus B \oplus C \oplus D &= (AB' + A'B) \oplus (CD' + C'D) \\ &= (AB' + A'B)(CD + C'D') + (AB + A'B')(CD' + C'D) \\ &= \sum(1, 2, 4, 7, 8, 11, 13, 14) \end{aligned}$$

Generador de Paridad Par

98

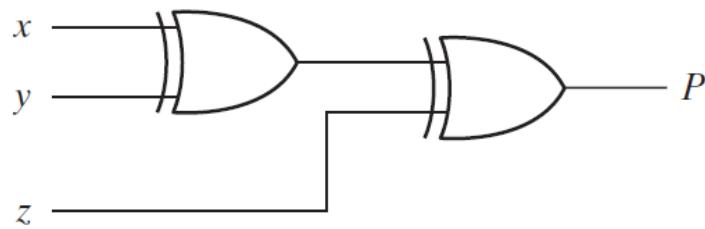
Capítulo 3 Minimización en el nivel de compuertas

Tabla 3-4

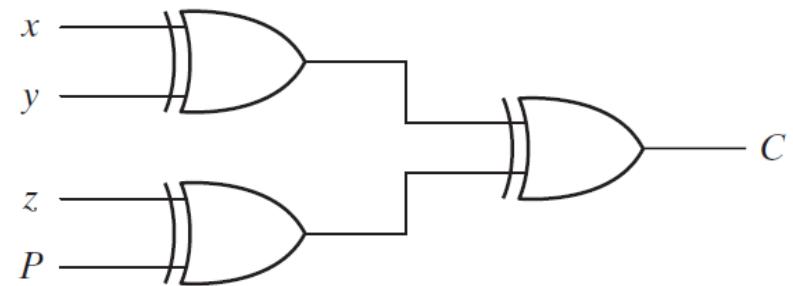
Tabla de verdad de un generador de paridad par

Mensaje de tres bits			Bit de paridad
x	y	z	P
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

Generador de Paridad Par



a) Generador de paridad par de tres bits



b) Verificador de paridad par de cuatro bits

FIGURA 3-36

Diagrama lógico de un generador y un verificador de paridad

Generador de Paridad

Tabla 3-5

Tabla de verdad de un verificador de paridad par

Cuatro bits recibidos				Verificador de errores de paridad
x	y	z	P	C
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

Lenguajes de Descripción de Hardware

Ejemplo HDL 3-1

```
//Descripción del circuito simple de la fig. 3-37
module circuito_smpl(A,B,C,x,y);
    input A,B,C;
    output x,y;
    wire e;
    and g1(e,A,B);
    not g2(y, C);
    or g3(x,e,y);
endmodule
```

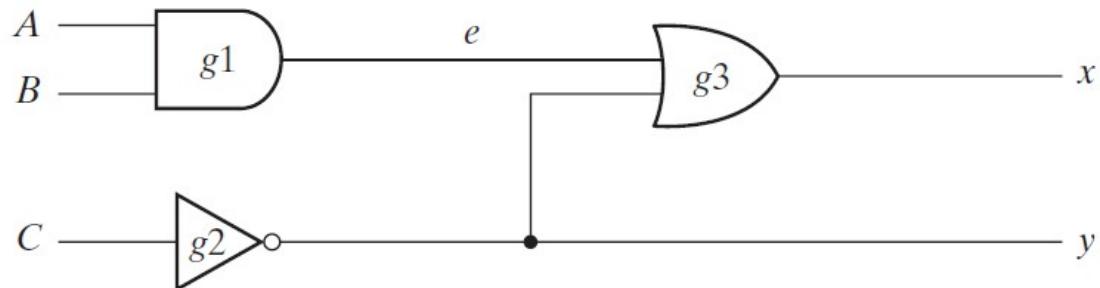


FIGURA 3-37
Circuito para ilustrar HDL

Lenguajes de Descripción de Hardware

102 Capítulo 3 Minimización en el nivel de compuertas

Retardos de compuerta

Cuando se usa HDL para hacer simulaciones, tal vez sea necesario especificar la magnitud del retardo que hay entre la entrada y la salida de las compuertas. En Verilog, el retardo se especifica en términos de *unidades de tiempo* y el símbolo #. La asociación de la unidad de tiempo con el tiempo físico se efectúa con la directriz de compilador `timescale. (Las directrices al compilador inician con el símbolo ` (acento grave).) Tales directrices se especifican antes de declarar módulos. Un ejemplo de directriz de escala de tiempo es:

```
`timescale 1ns/100ps
```

Lenguajes de Descripción de Hardware

Ejemplo HDL 3-2

```
//Descripción de circuito con retardo
module circuito_con_retardo(A,B,C,x,y);
    input A,B,C;
    output x,y;
    wire e;
    and #(30) g1(e,A,B);
    or #(20) g3(x,e,y);
    not #(10) g2(y,C);
endmodule
```

Lenguajes de Descripción de Hardware

Tabla 3-6

Salida de las compuertas después del retardo

Unidades de tiempo	(ns)	Entrada			Salida		
		A	B	C	y	e	x
Inicial	—	0	0	0	1	0	1
Cambio	—	1	1	1	1	0	1
	10	1	1	1	0	0	1
	20	1	1	1	0	0	1
	30	1	1	1	0	1	0
	40	1	1	1	0	1	0
	50	1	1	1	0	1	1

Lenguajes de Descripción de Hardware

Ejemplo HDL 3-3

```
//Estímulo para el circuito simple
module stimcrct;
reg A,B,C;
wire x,y;
circuito_con_retardo ccr(A,B,C,x,y);
initial
    begin
        A = 1'b0; B = 1'b0; C = 1'b0;
        #100
        A = 1'b1; B = 1'b1; C = 1'b1;
        #100 $finish;
    end
endmodule

//Descripción de circuito con retardo
module circuito_con_retardo (A,B,C,x,y);
    input A,B,C;
    output x,y;
    wire e;
    and #(30) g1(e,A,B);
    or  #(20) g3(x,e,Y);
    not #(10) g2(y,C);
endmodule
```

Lenguajes de Descripción de Hardware

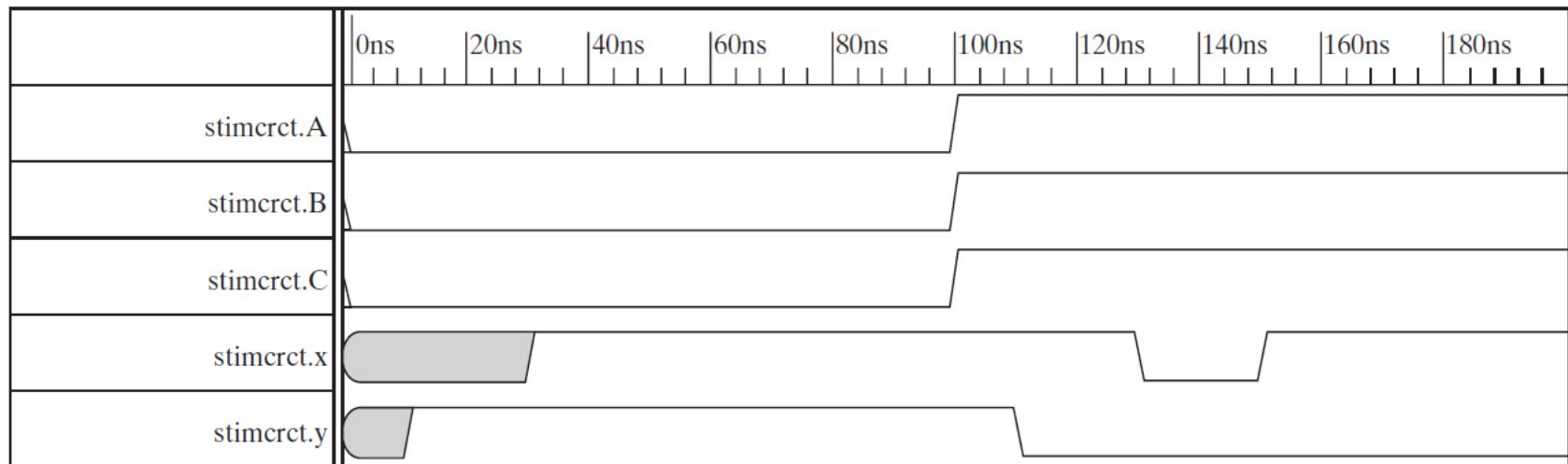


FIGURA 3-38
Salida de la simulación del ejemplo HDL 3-3

Lenguajes de Descripción de Hardware

Ejemplo HDL 3-4

```
//Circuito especificado con expresiones booleanas
module circuit_bln (x,y,A,B,C,D);
    input A,B,C,D;
    output x,y;
    assign x = A | (B & C) | (~B & D);
    assign y = (~B & C) | (B & ~C & ~D);
endmodule
```

Lenguajes de Descripción de Hard

Ejemplo HDL 3-5

```
//Primitiva definida por el usuario (UDP)
primitive crctp (x,A,B,C);
    output x;
    input A,B,C;
//Tabla de verdad para x(A,B,C) = Σ(0,2,4,6,7)
    table
        //      A      B      C   :   x   (Esto es sólo un comentario)
        0      0      0   :   1;
        0      0      1   :   0;
        0      1      0   :   1;
        0      1      1   :   0;
        1      0      0   :   1;
        1      0      1   :   0;
        1      1      0   :   1;
        1      1      1   :   1;
    endtable
endprimitive

//Crear una copia de la primitiva
module declare_crctp;
    reg x,y,z;
    wire w;
    crctp (w,x,y,z);
endmodule
```

LÓGICA COMBINACIONAL ALGUNOS MÓDULOS BÁSICOS



Circuitos Combinacionales

► Esquema General

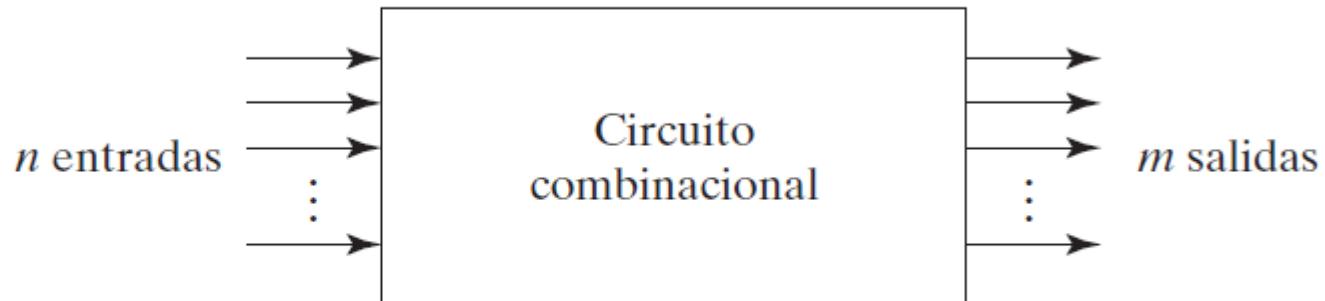


FIGURA 4-1
Diagrama de bloques de un circuito combinacional

Circuitos Combinacionales

► Análisis

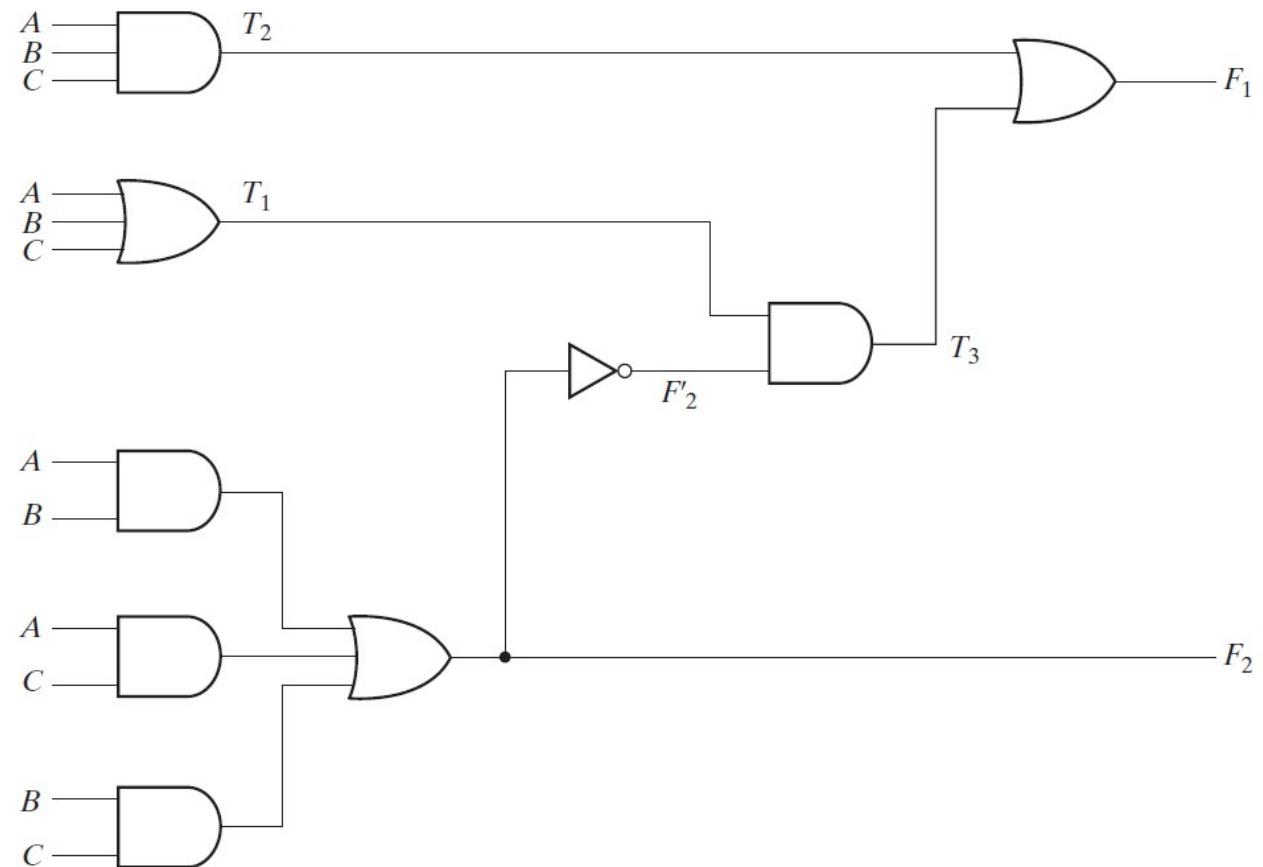


FIGURA 4-2

Diagrama lógico para el ejemplo de análisis

Circuitos Combinacionales

- ▶ Ejemplo de Diseño

Tabla 4-2

Tabla de verdad para el ejemplo de conversión de código

Entrada BCD				Salida código exceso-3			
A	B	C	D	w	x	y	z
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

Circuitos Combinacionales

► Ejemplo de Diseño

Sección 4-3 Procedimiento de diseño 117

		CD		C		
		00	01	11	10	
AB	00	1			1	B
	01	1			1	
	11	X	X	X	X	
	10	1		X	X	

D

$z = D'$

		CD		C		
		00	01	11	10	
AB	00	1		1		B
	01	1		1		
	11	X	X	X	X	
	10	1		X	X	

D

$y = CD + C'D'$

Circuitos Combinacionales

► Ejemplo de Diseño

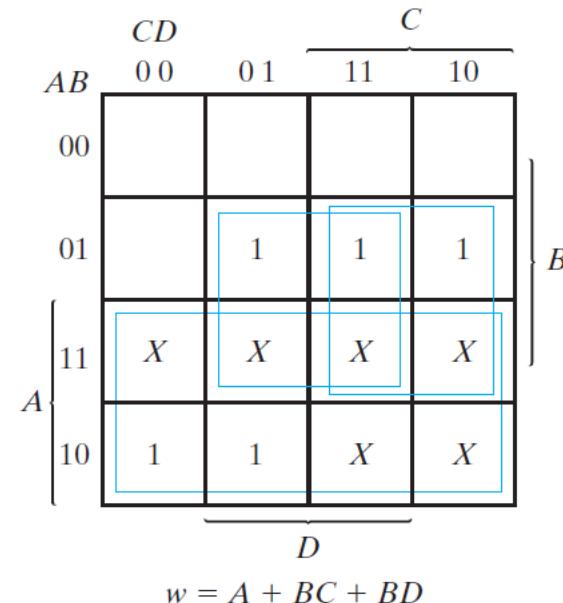
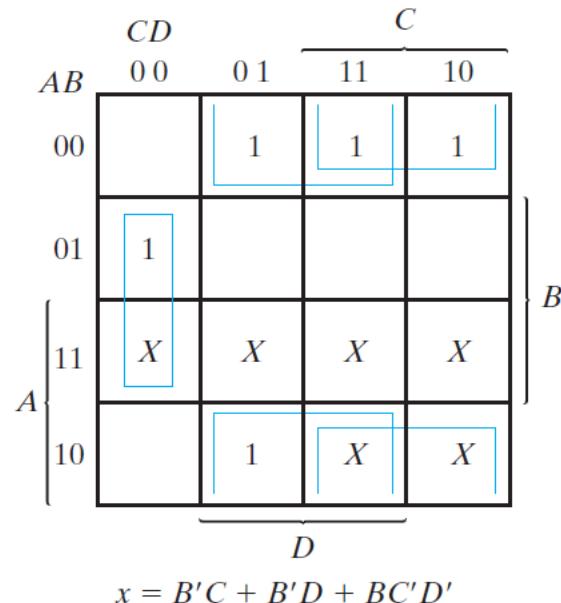


FIGURA 4-3
Mapas para el convertidor de código BCD a exceso-3

Circuitos Combinacionales

A blue triangle icon pointing to the right, indicating a continuation or example.

Ej:

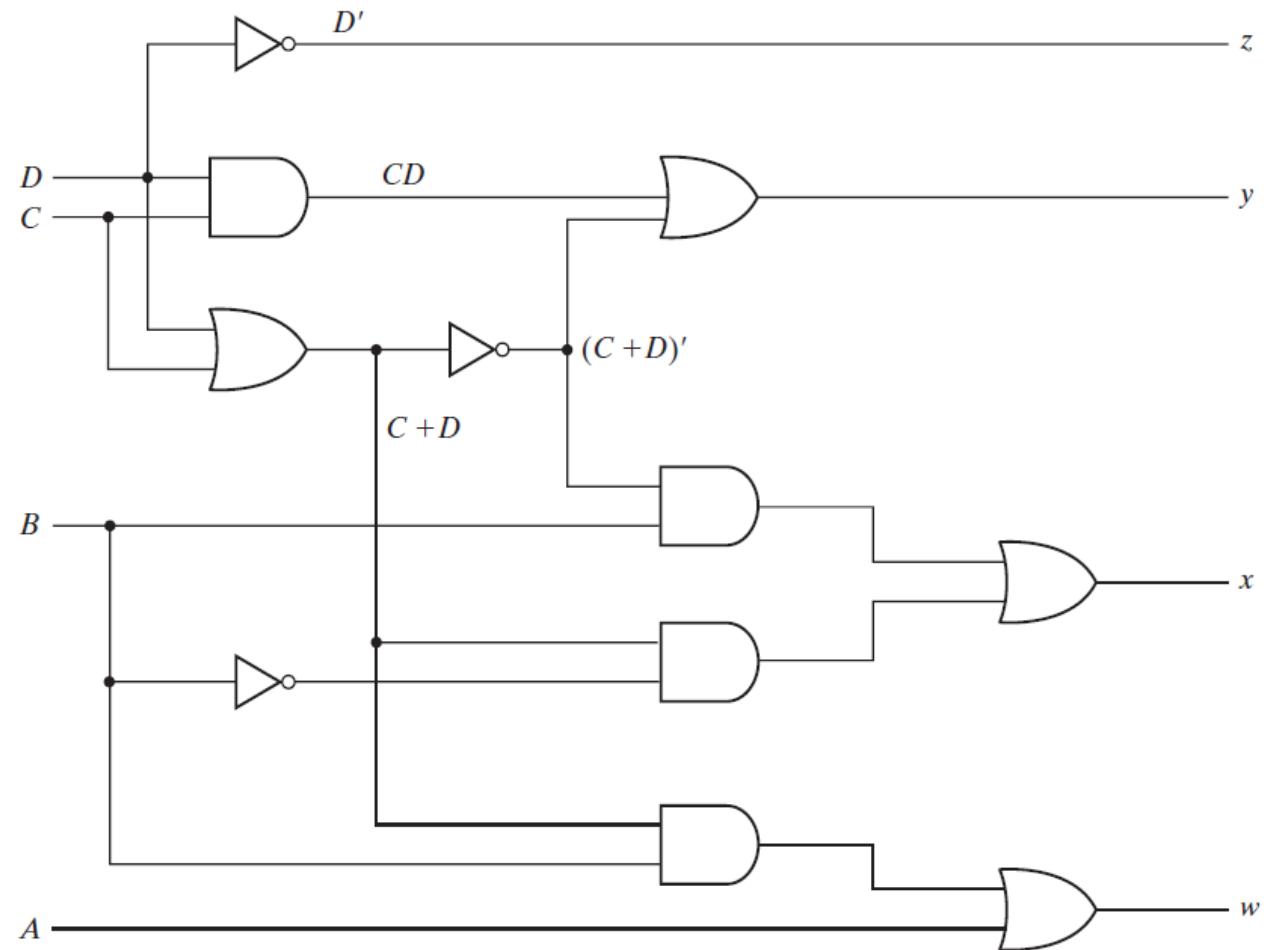


FIGURA 4-4

Diagrama lógico para el convertidor de código BCD a exceso-3

Ejemplo de un Sumador Restador

- ▶ Semisumador

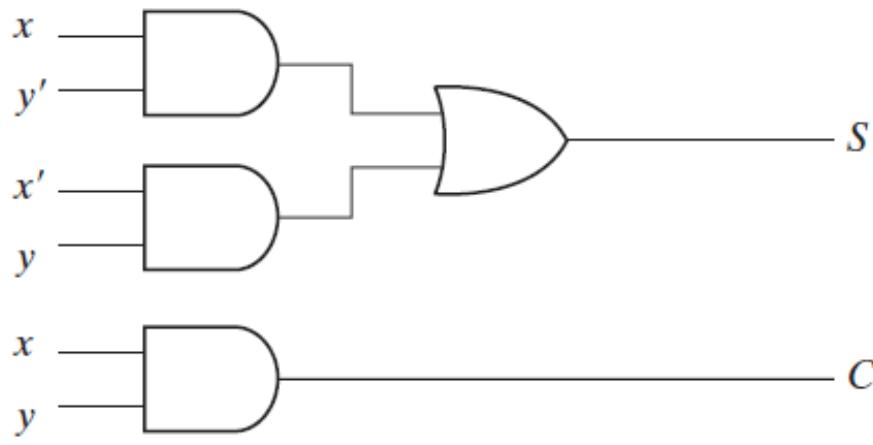
Tabla 4-3
Semisumador

<i>x</i>	<i>y</i>	<i>c</i>	<i>s</i>
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

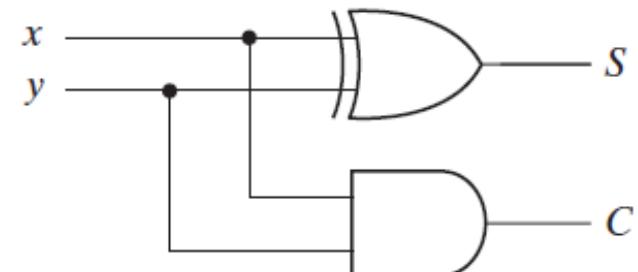
Ejemplo de un Sumador Restador

► Semisumador

Capítulo 4 Lógica combinacional



$$\text{a) } S = xy' + x'y \\ C = xy$$



$$\text{b) } S = x \oplus y \\ C = xy$$

FIGURA 4-5

Implementación de semisumador

Ejemplo de un Sumador Restador

- ▶ Sumador Completo

Tabla 4-4
Sumador completo

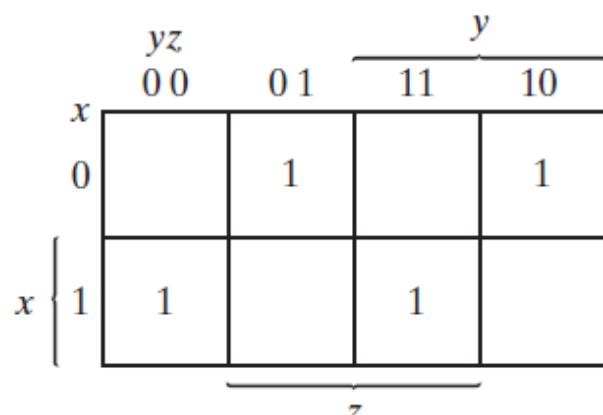
<i>x</i>	<i>y</i>	<i>z</i>	<i>c</i>	<i>s</i>
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Ejemplo de un Sumador Restador

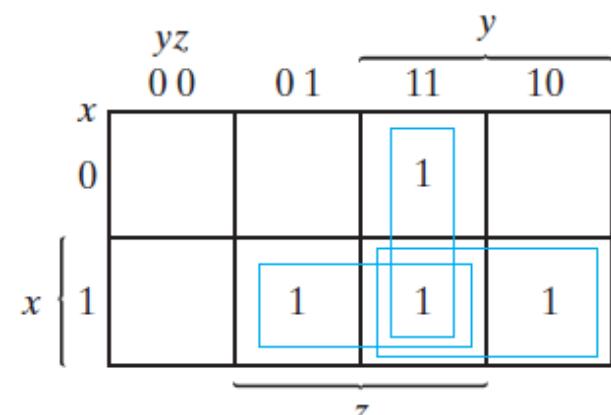
► Sumador Completo

Sección 4-4 Sumador-restador binario

121



$$S = x'y'z + x'yz' + xy'z' + xyz$$



$$\begin{aligned}C &= xy + xz + yz \\&= xy + xy'z + x'yz\end{aligned}$$

FIGURA 4-6

Mapas para el sumador completo

Ejemplo de un Sumador Restador

▶ Sumador Completo

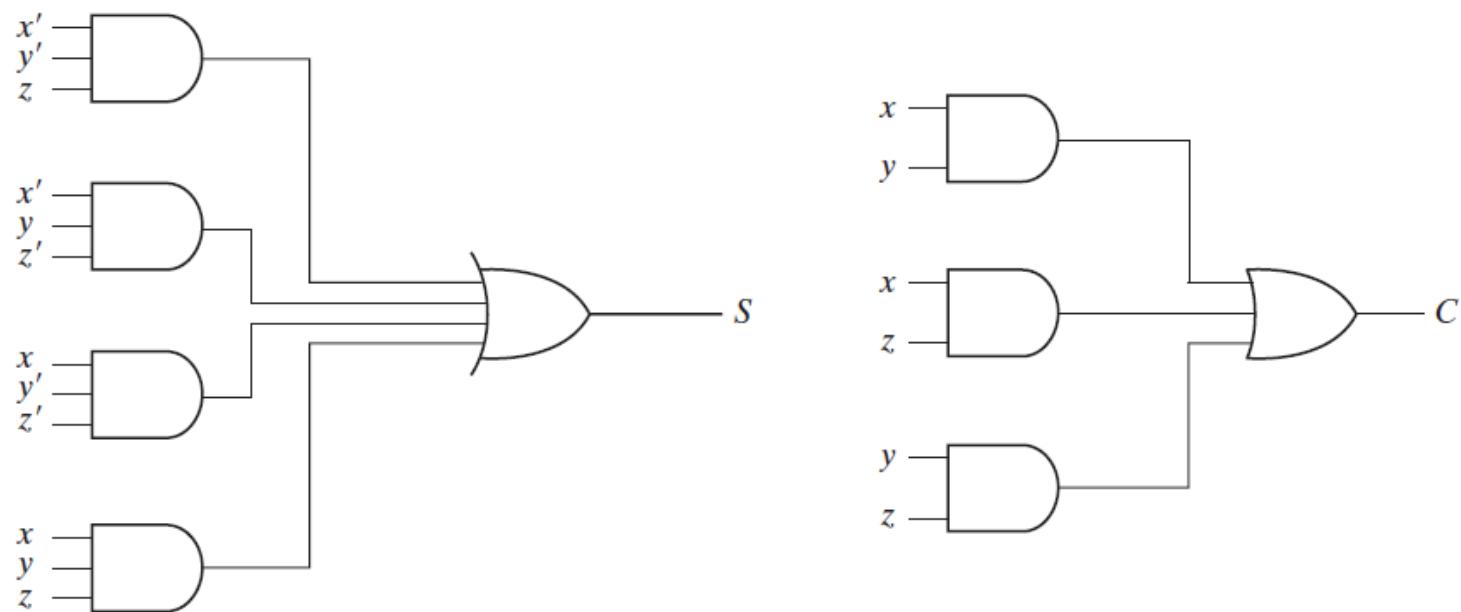


FIGURA 4-7

Implementación de un sumador completo como suma de productos

Ejemplo de un Sumador Restador

▶ Sumador Completo

122 Capítulo 4 Lógica combinacional

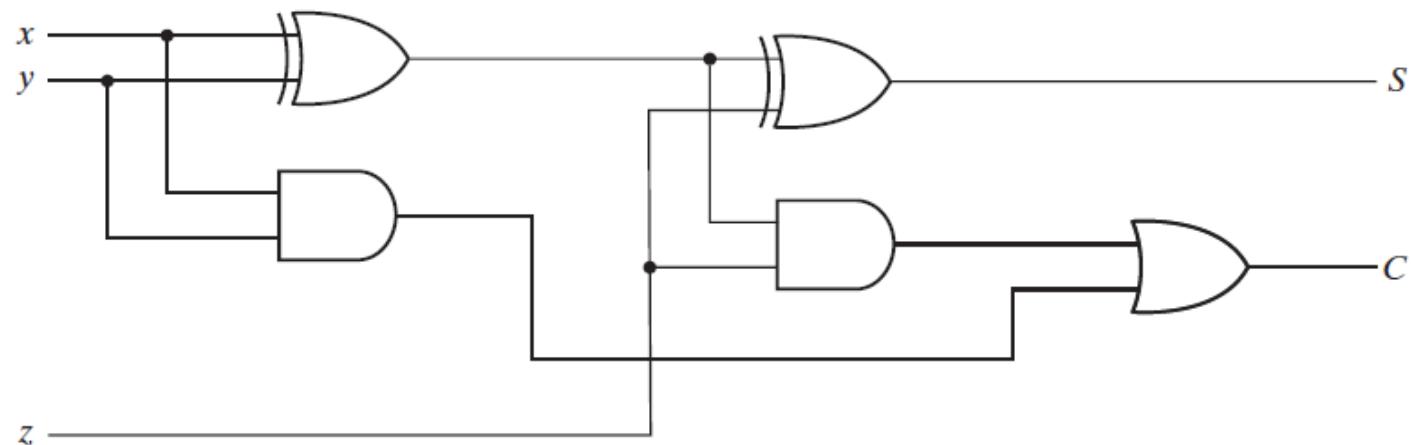


FIGURA 4-8

Implementación de un sumador completo con dos semisumadores y una compuerta OR

Ejemplo de Sumador Restador

► Sumador Completo

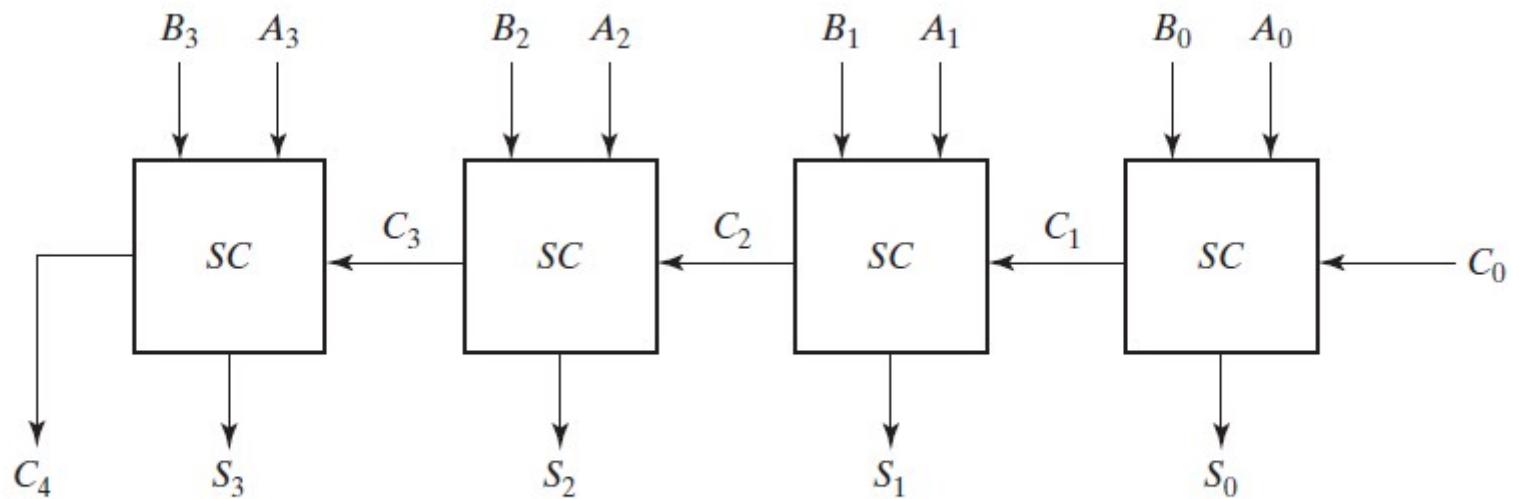


FIGURA 4-9
Sumador de cuatro bits

Ejemplo de Sumador Restador

- ▶ Sumador Completo

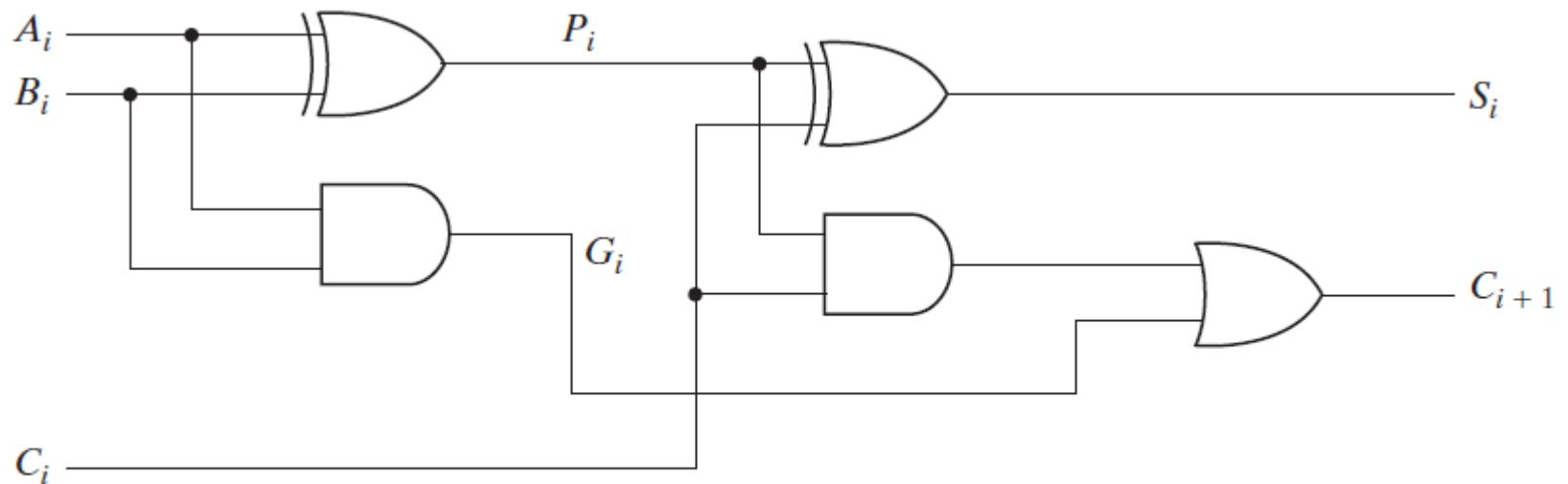


FIGURA 4-10
Sumador completo en el que se indican P y G

Ejemplo de Sumador Restador

► Acarreo Anticipado

Considere el circuito del sumador completo que se aprecia en la figura 4-10. Si definimos dos nuevas variables binarias

$$P_i = A_i \oplus B_i$$

$$G_i = A_i B_i$$

la suma y el acarreo se expresarán así:

$$S_i = P_i \oplus C_i$$

$$C_{i+1} = G_i + P_i C_i$$

Ejemplo de Sumador Restador

► Acarreo Anticipado

C_0 = acarreo de entrada

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 C_1 = G_1 + P_1(G_0 + P_0 C_0) = G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$C_3 = G_2 + P_2 C_2 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

Ejemplo de Sumador Restador

Sección 4-4 Sumador-restador binario 125

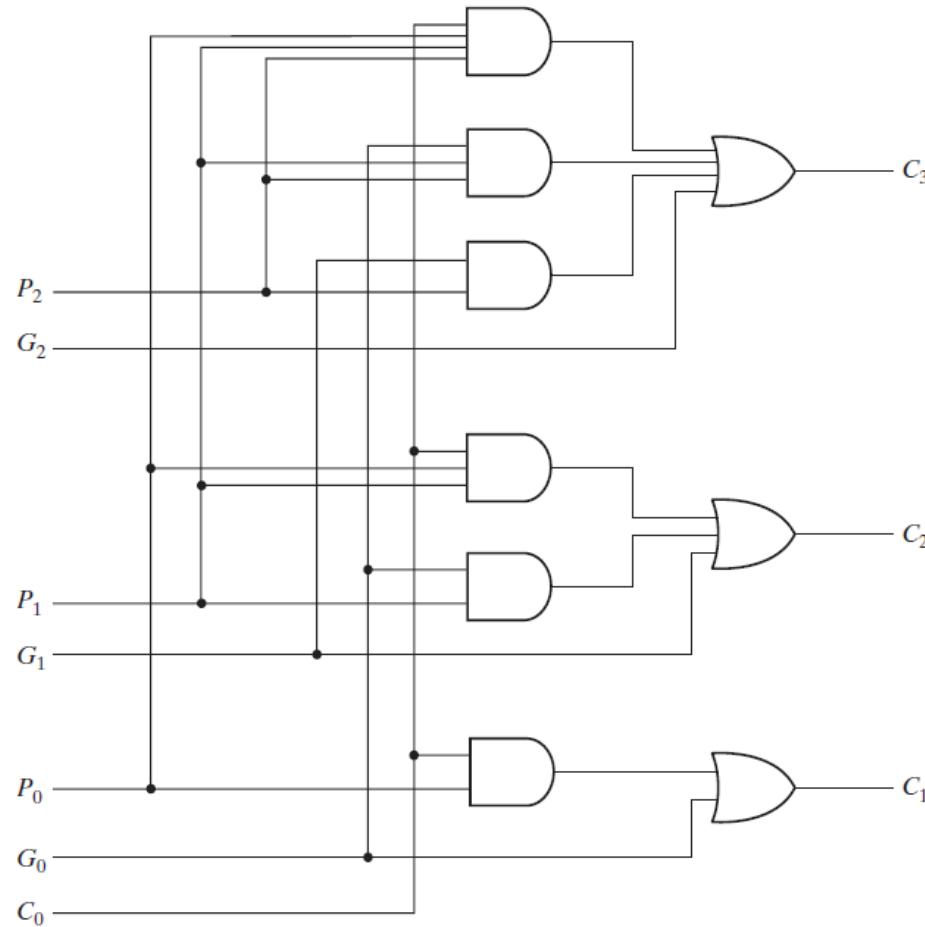


FIGURA 4-11
Diagrama lógico del generador de acarreo anticipado

Ejemplo de Sumador Restador

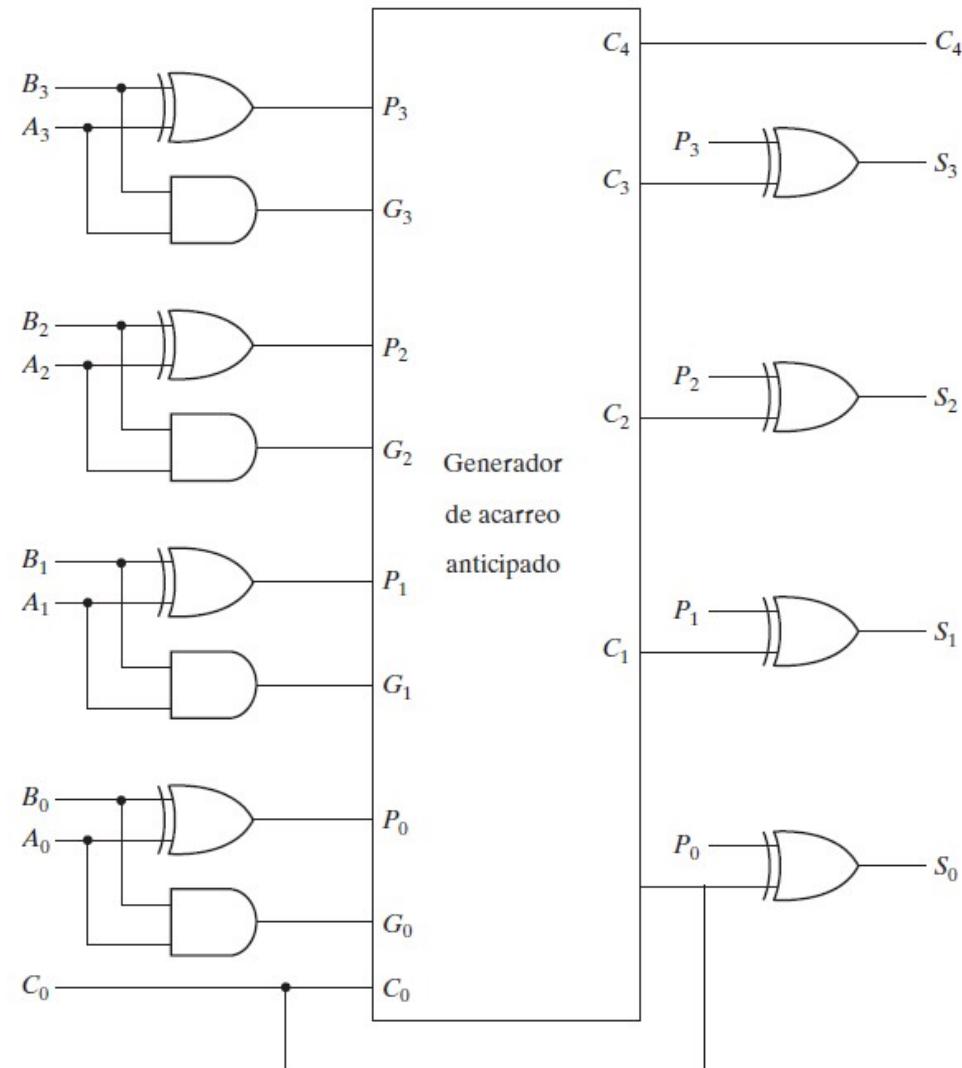


FIGURA 4-12

Sumador de cuatro bits con acarreo anticipado

Ejemplo de Sumador Restador

Sección 4-4 Sumador-restador binario

127

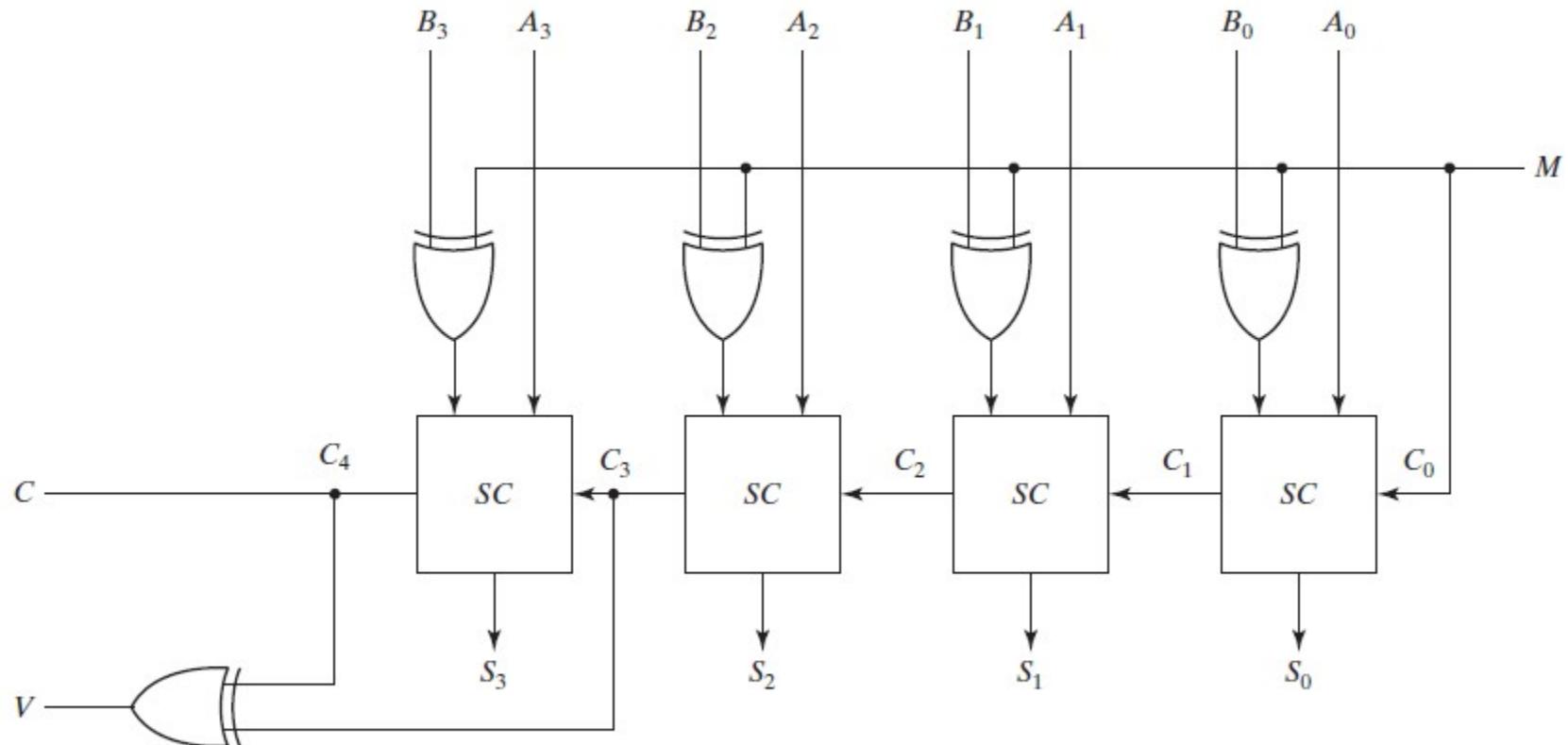


FIGURA 4-13
Sumador-restador de cuatro bits

Ejemplo de Multiplicador Binario

$$\begin{array}{r} B_1 \quad B_0 \\ A_1 \quad A_0 \\ \hline A_0B_1 \quad A_0B_0 \\ A_1B_1 \quad A_1B_0 \\ \hline C_3 \quad C_2 \quad C_1 \quad C_0 \end{array}$$

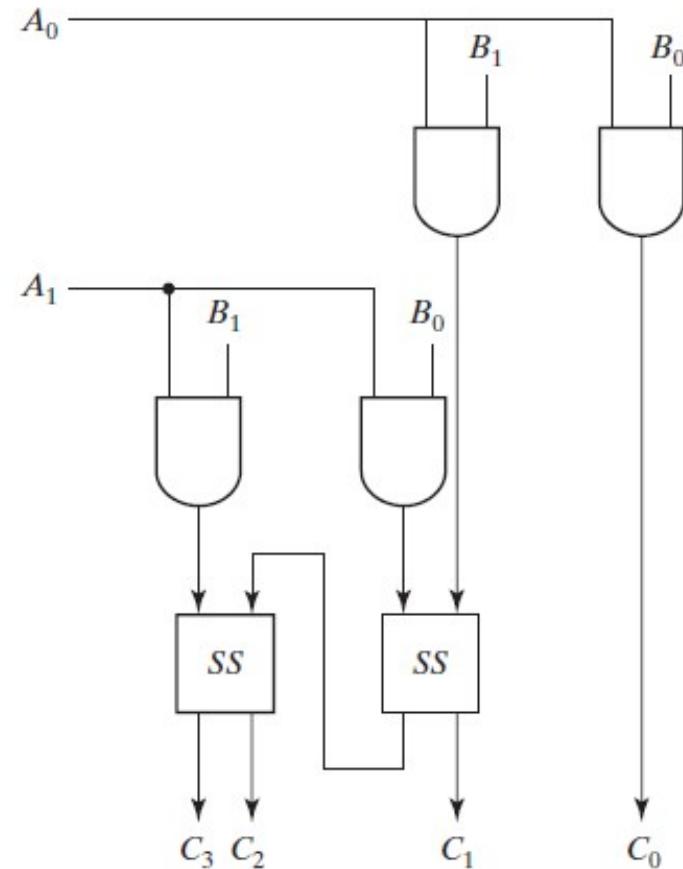


FIGURA 4-15
Multiplicador binario de dos bits por dos bits

Ejemplo de Multiplicador Binario

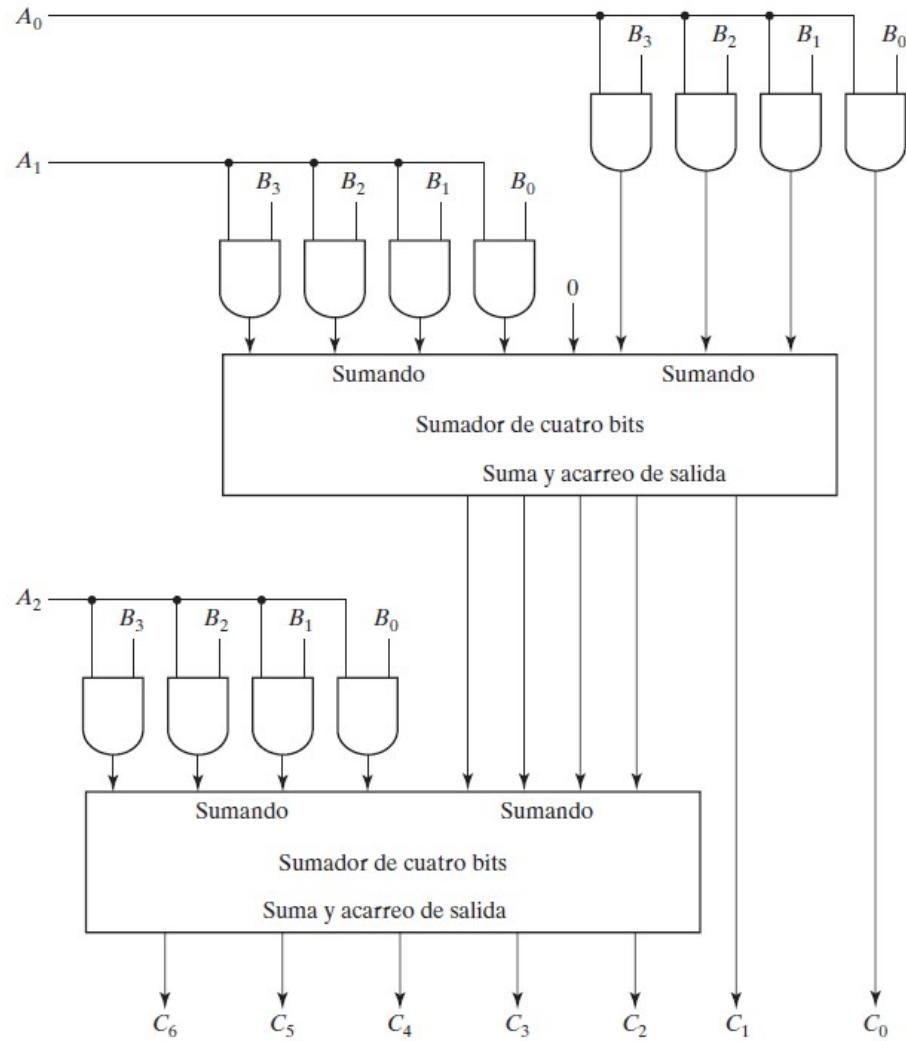


FIGURA 4-16
Multiplicador binario de 4 bits por 3 bits

DECODIFICADORES

Sección 4-8 Decodificadores

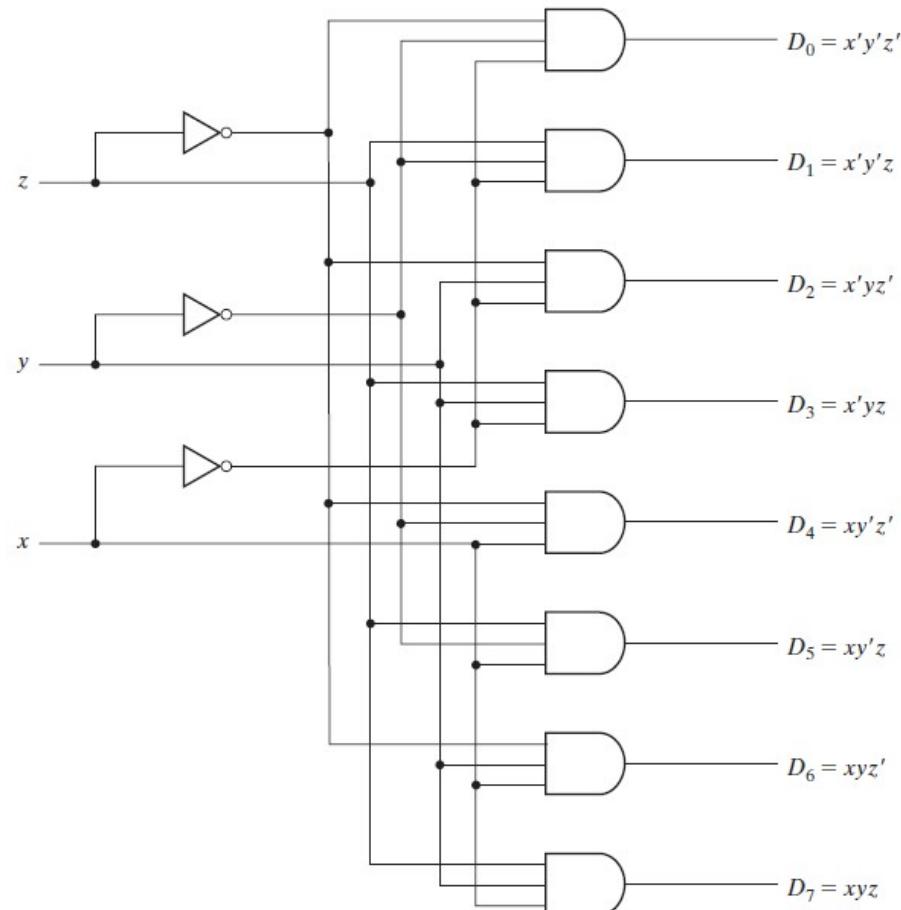
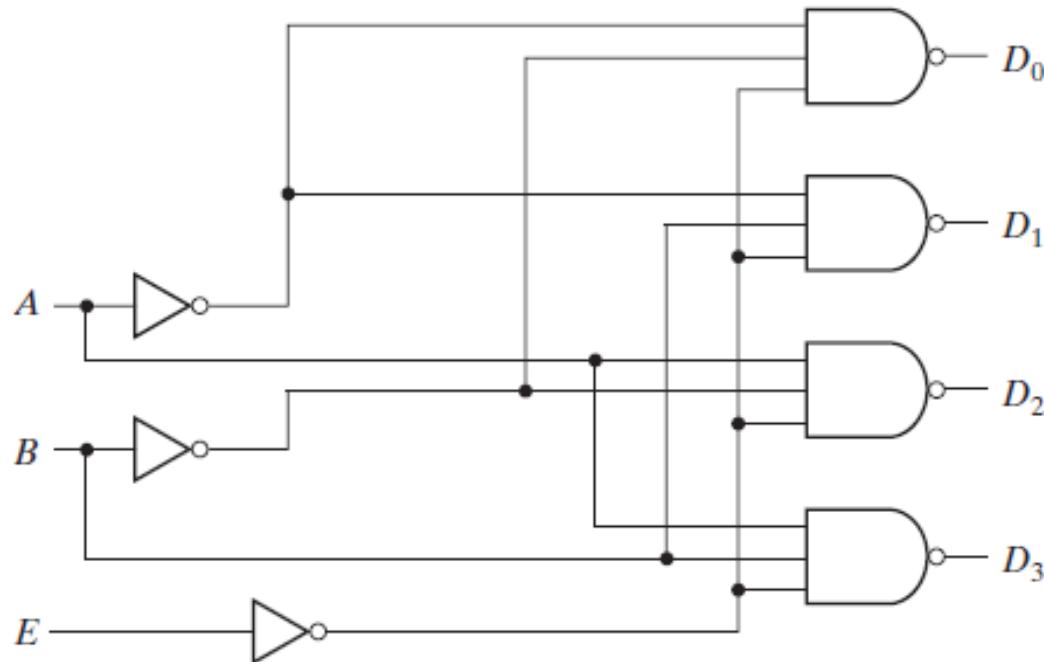


FIGURA 4-18
Decodificador de 3 a 8 líneas

DECODIFICADORES

Tabla 4-6
Tabla de verdad de un decodificador de 3 a 8 líneas

DECODIFICADORES



a) Diagrama lógico

b) Tabla de verdad

E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

FIGURA 4-19

Decodificador de 2 a 4 líneas con entrada habilitadora

DECODIFICADORES

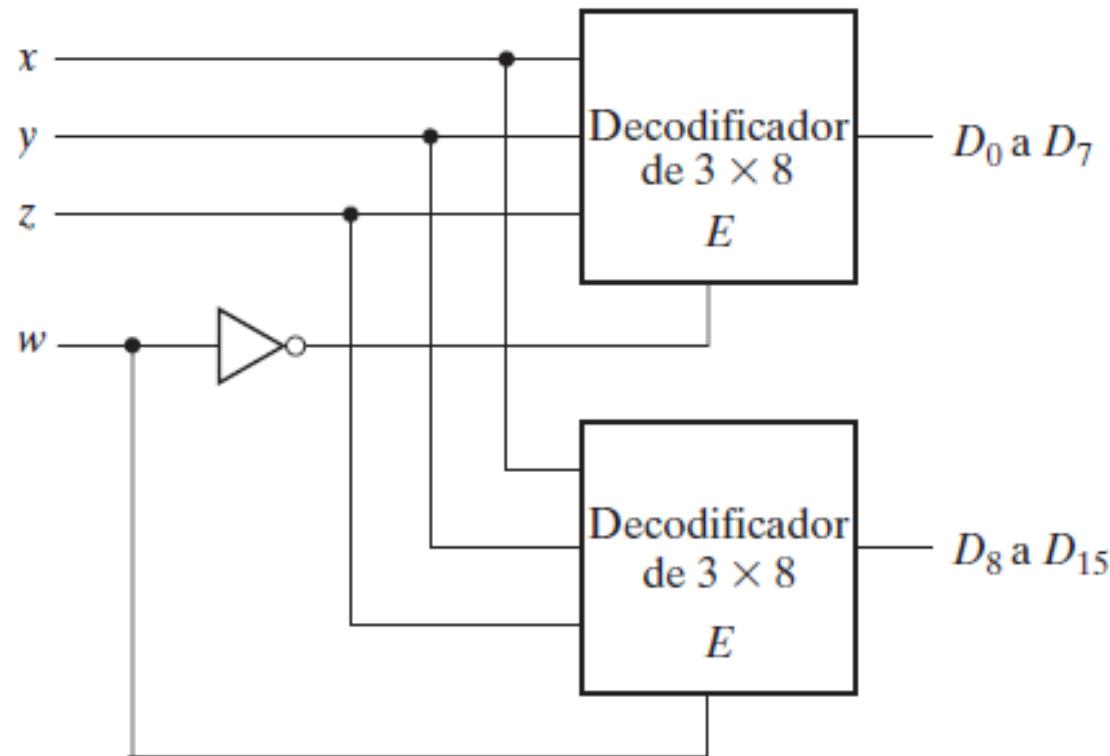


FIGURA 4-20

Decodificador 4×16 construido con dos decodificadores 3×8

Implementación de lógica con decodificadores

- ▶ Suma de Productos

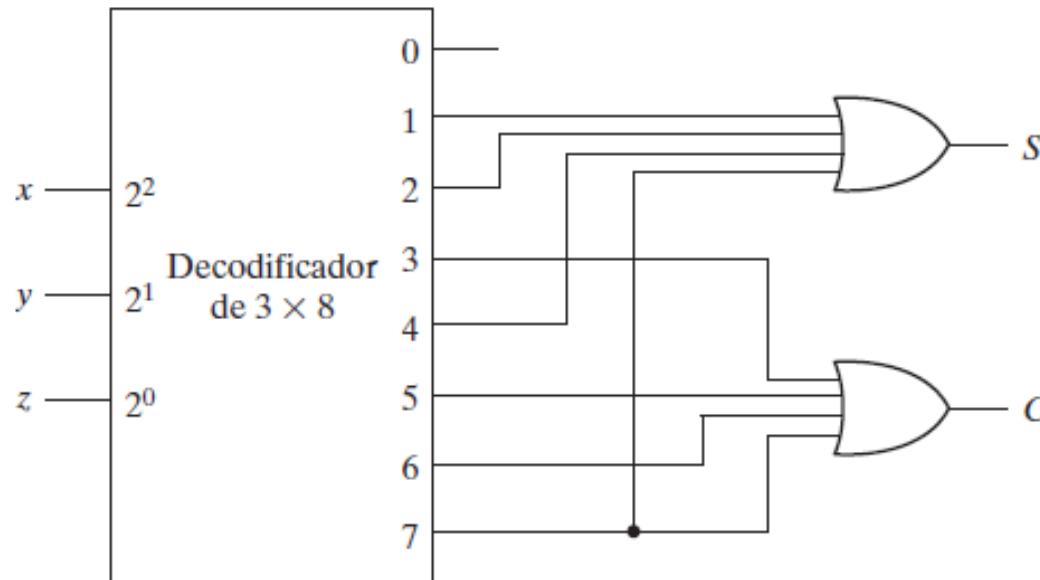
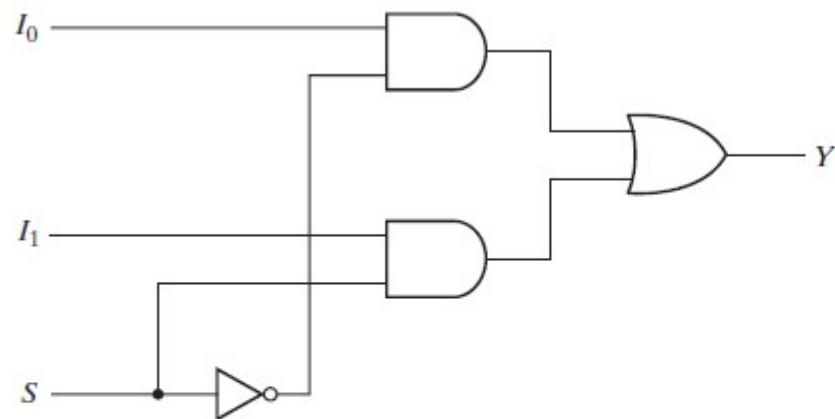


FIGURA 4-21

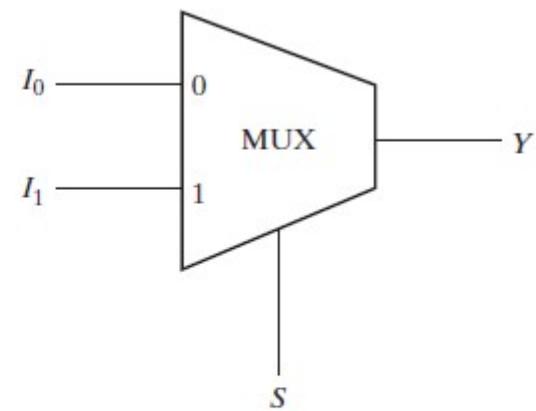
Implementación de un sumador completo con un decodificador

MULTIPLEXORES

- De 2 a 1



a) Diagrama lógico

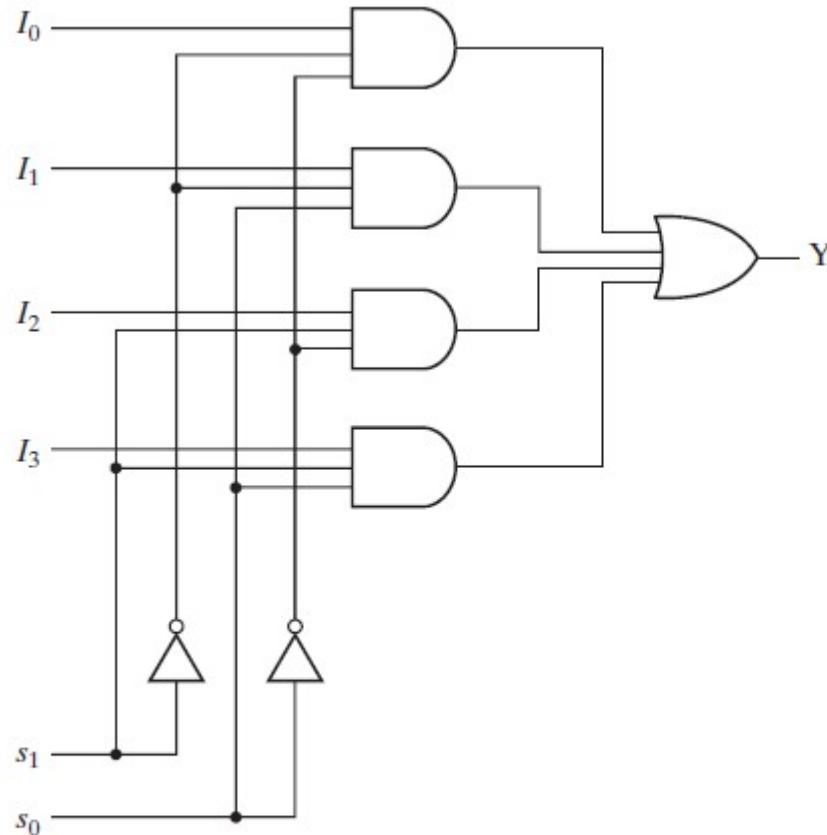


b) Diagrama de bloque

FIGURA 4-24
Multiplexor de 2 líneas a 1

MULTIPLEXORES

► De 4 a 1



a) Diagrama lógico

s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

b) Tabla de función

FIGURA 4-25
Multiplexor de 4 líneas a 1

MULTIPLEXORES

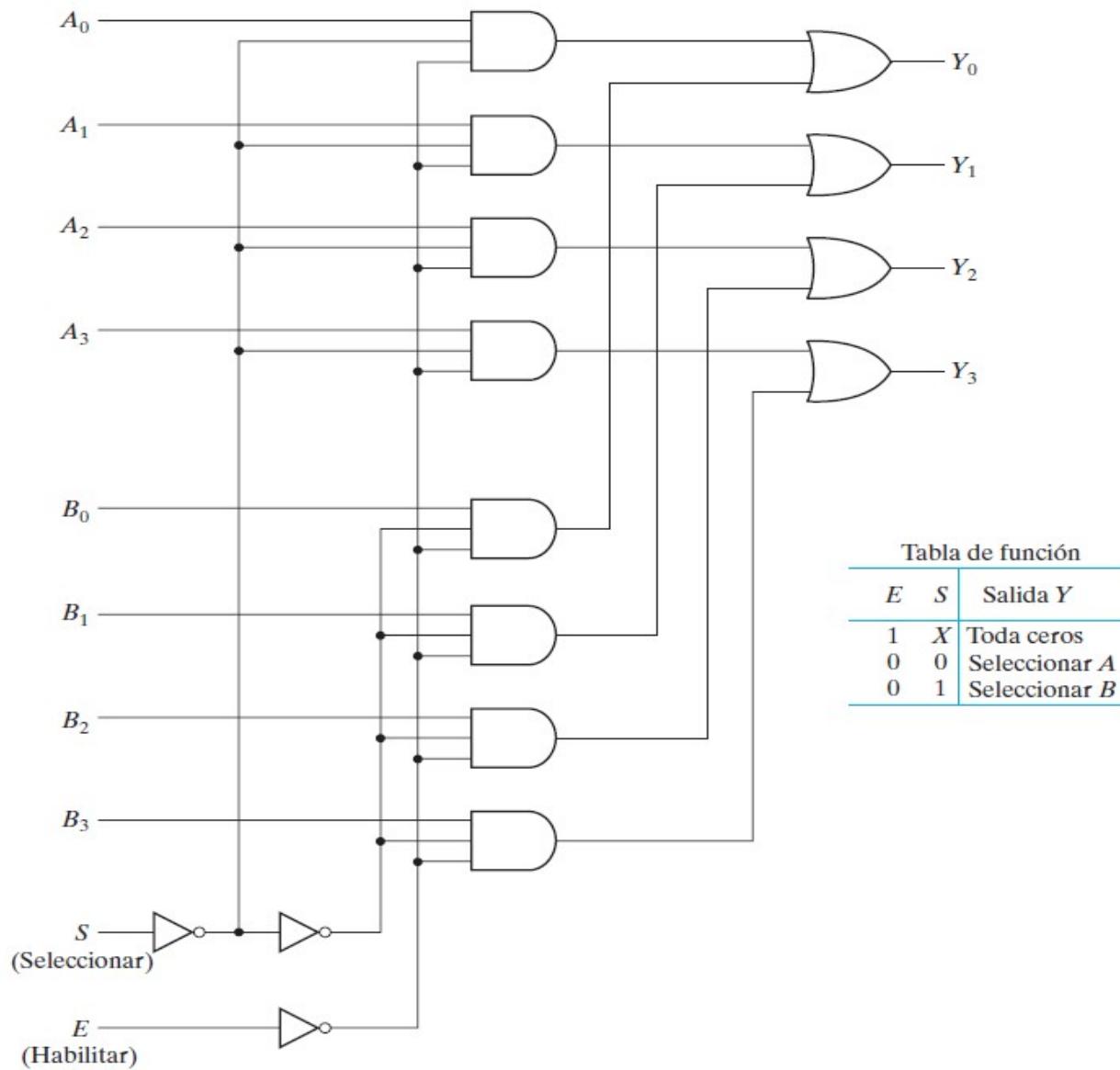
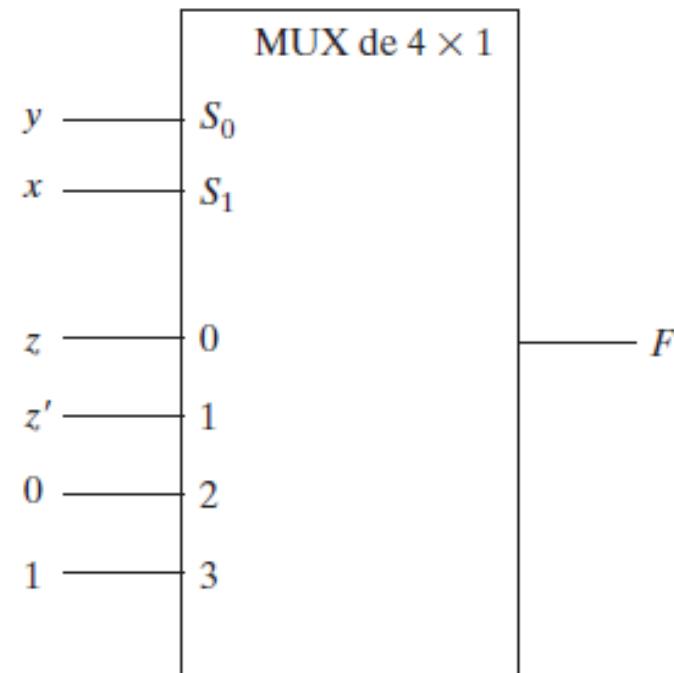


FIGURA 4-26
Multiplexor cuádruple de 2 líneas a 1

IMPLEMENTACIÓN DE LÓGICA CON MULTIPLEXORES

x	y	z	F
0	0	0	0 $F = z$
0	0	1	1
0	1	0	1 $F = z'$
0	1	1	0
1	0	0	0 $F = 0$
1	0	1	0
1	1	0	1
1	1	1	1 $F = 1$

a) Tabla de verdad



b) Implementación con multiplexor

FIGURA 4-27

Implementación de una función booleana con un multiplexor

IMPLEMENTACIÓN DE LÓGICA CON MULTIPLEXORES

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

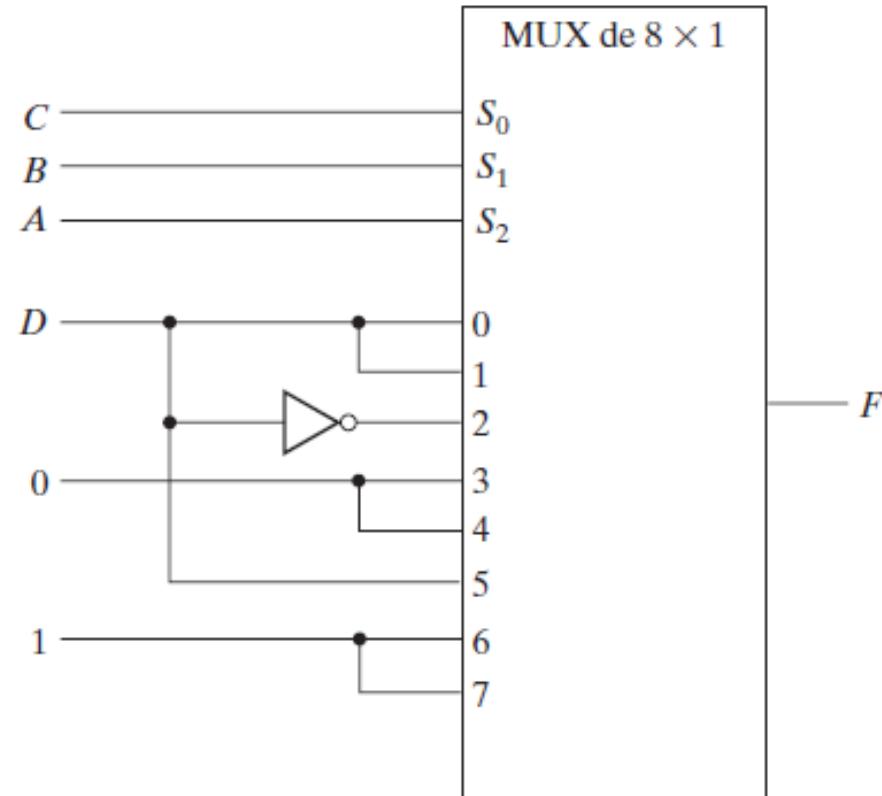
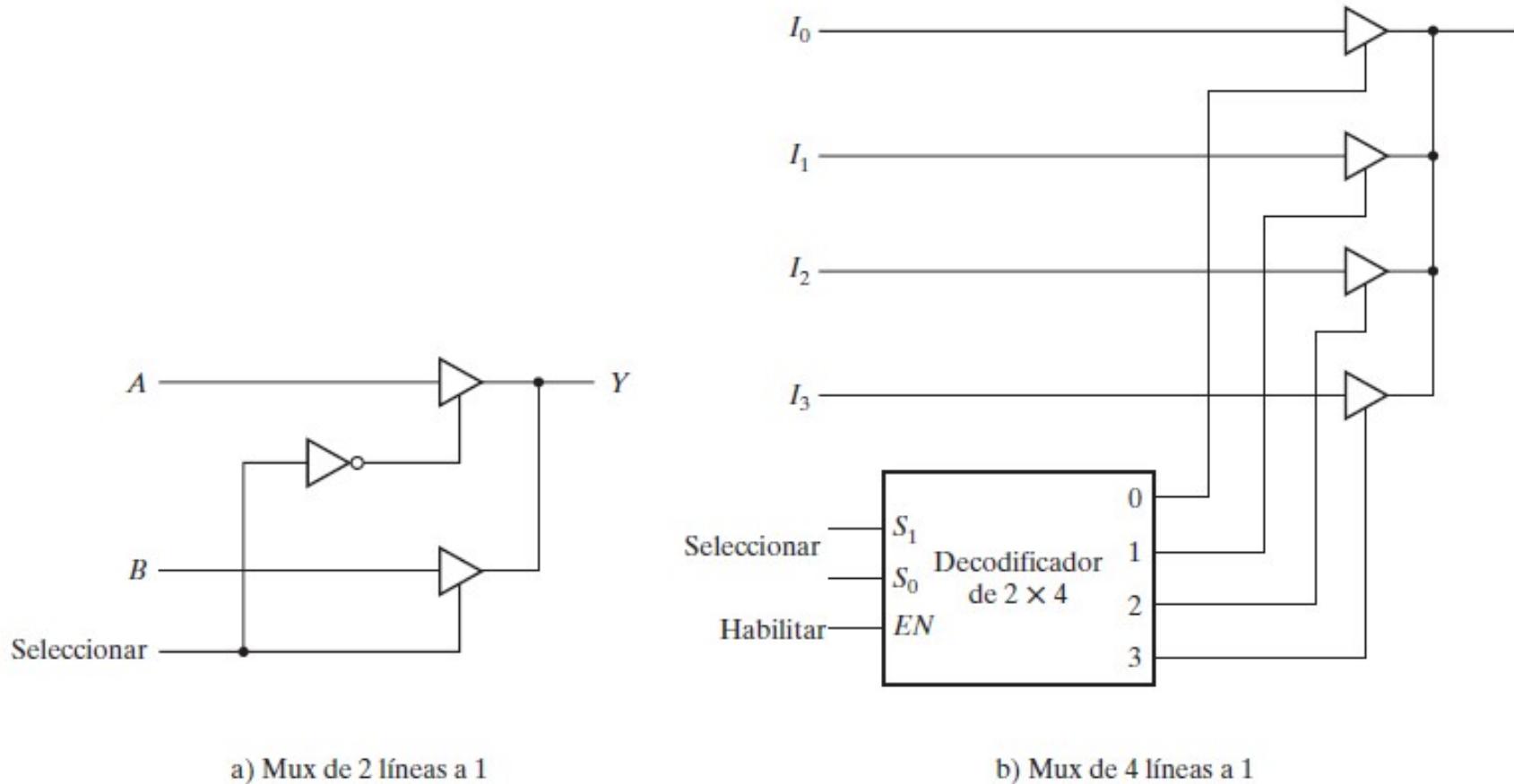


FIGURA 4-28

Implementación de una función de cuatro entradas con un multiplexor

IMPLEMENTACIÓN DE LÓGICA CON BUFFERS DE ALTA CON ALTA IMPEDANCIA



a) Mux de 2 líneas a 1

b) Mux de 4 líneas a 1

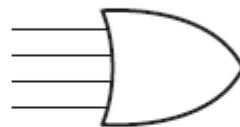
FIGURA 4-30

Multiplexores con compuertas de tres estados

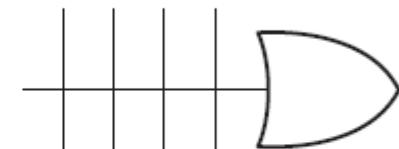
Memoria y Lógicas Programables

256

Capítulo 7 Memoria y lógica programable



a) Símbolo convencional



b) Símbolo de arreglo lógico

FIGURA 7-1

Diagramas convencional y de arreglo lógico para la compuerta OR

Read Only Memories ROM

Sección 7-5 Memoria de sólo lectura

271

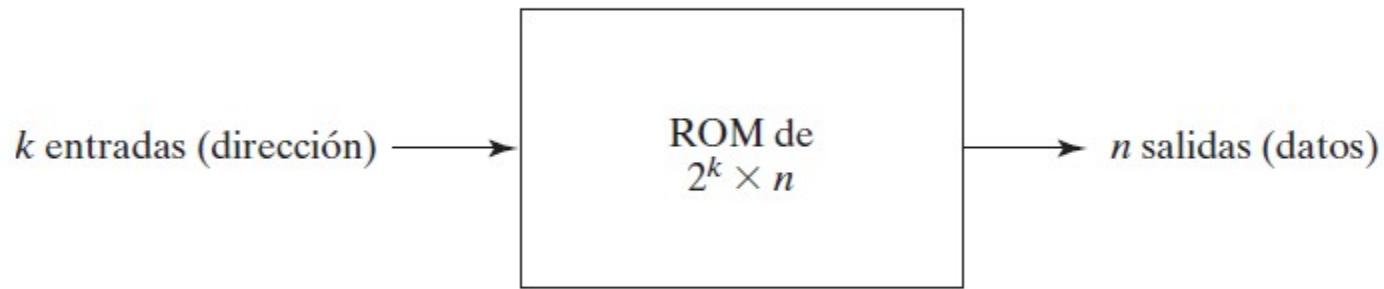


FIGURA 7-9
Diagrama de bloques de ROM

Read Only Memories ROM

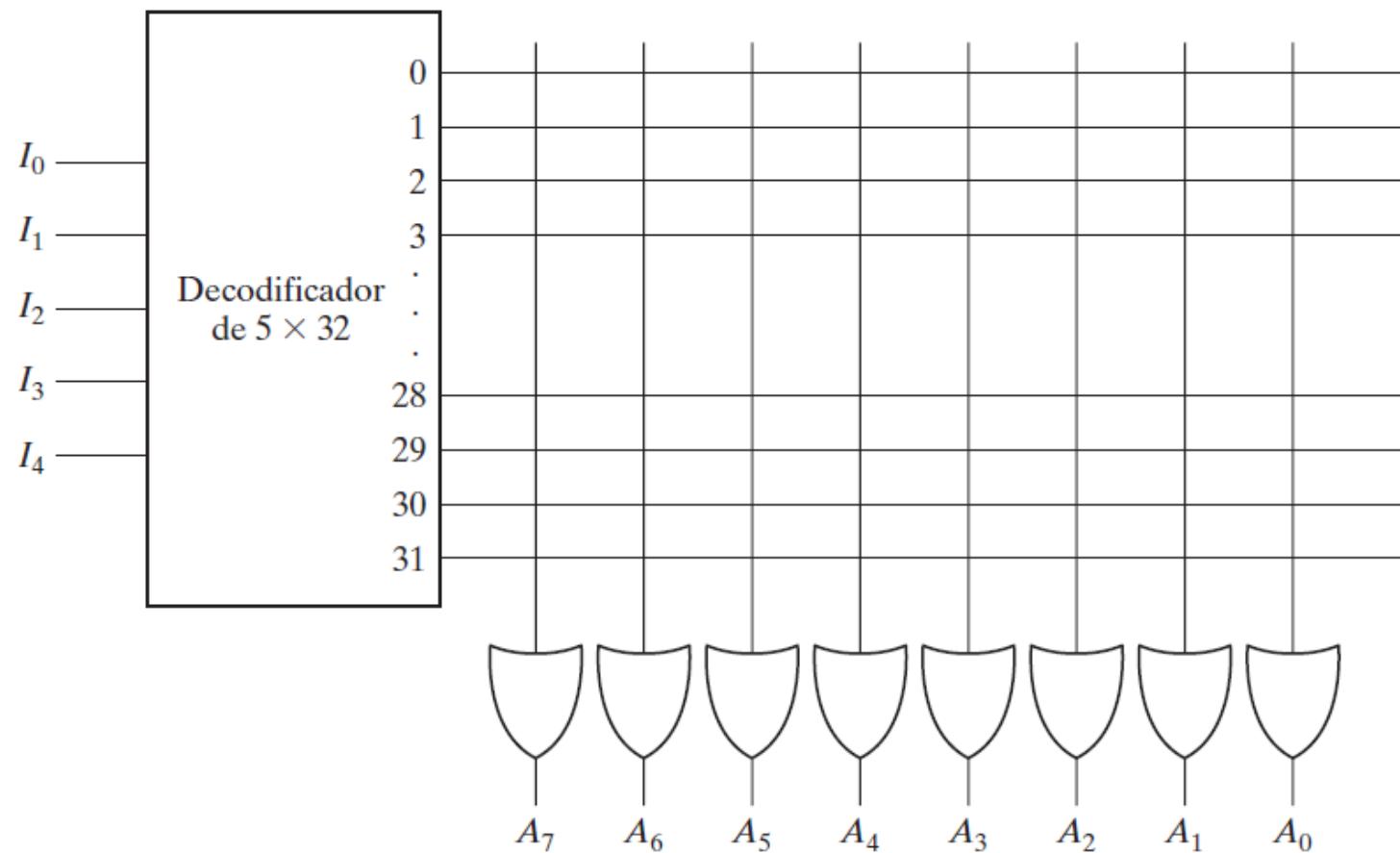


FIGURA 7-10
Lógica interna de una ROM de 32×8

Implementando Combinacionales con ROM

272 Capítulo 7 Memoria y lógica programable

Tabla 7-3
Tabla de verdad de ROM (parcial)

Entradas					Salidas							
I4	I3	I2	I1	I0	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	1	0	1	1	0	1	1	0
0	0	0	0	1	0	0	0	1	1	1	0	1
0	0	0	1	0	1	1	0	0	0	1	0	1
0	0	0	1	1	1	0	1	1	0	0	1	0
⋮					⋮							
1	1	1	0	0	0	0	0	0	1	0	0	1
1	1	1	0	1	1	1	1	0	0	0	1	0
1	1	1	1	0	0	1	0	0	1	0	1	0
1	1	1	1	1	0	0	1	1	0	0	1	1

Implementando Combinacionales con ROM

Sección 7-5 Memoria de sólo lectura

273

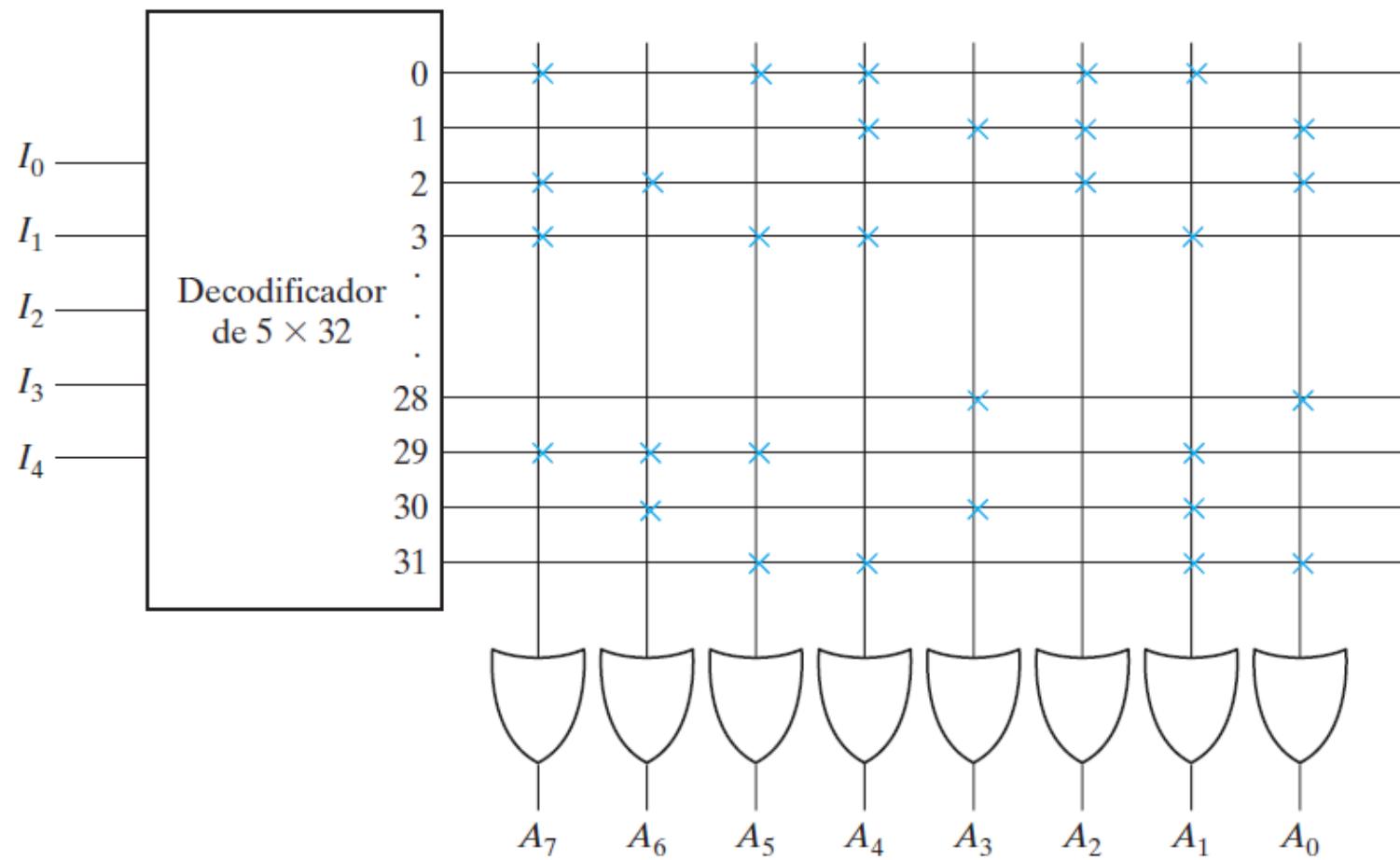


FIGURA 7-11
Programación de la ROM según la tabla 7-3

Programmable Logic Devices (PLDs)

276 Capítulo 7 Memoria y lógica programable

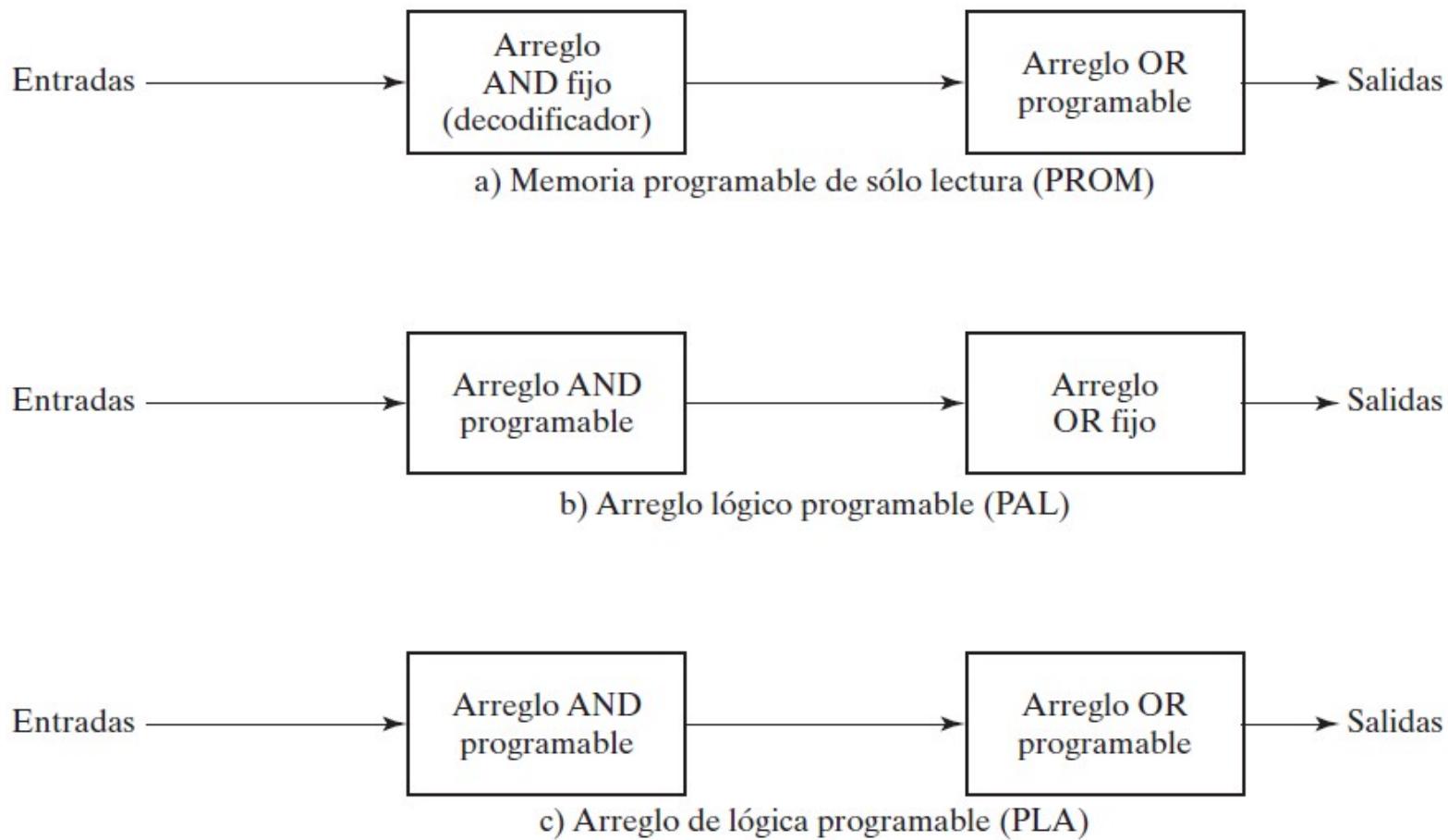


FIGURA 7-13

Configuración básica de tres PLD

Programmable Logic Devices (PLDs)

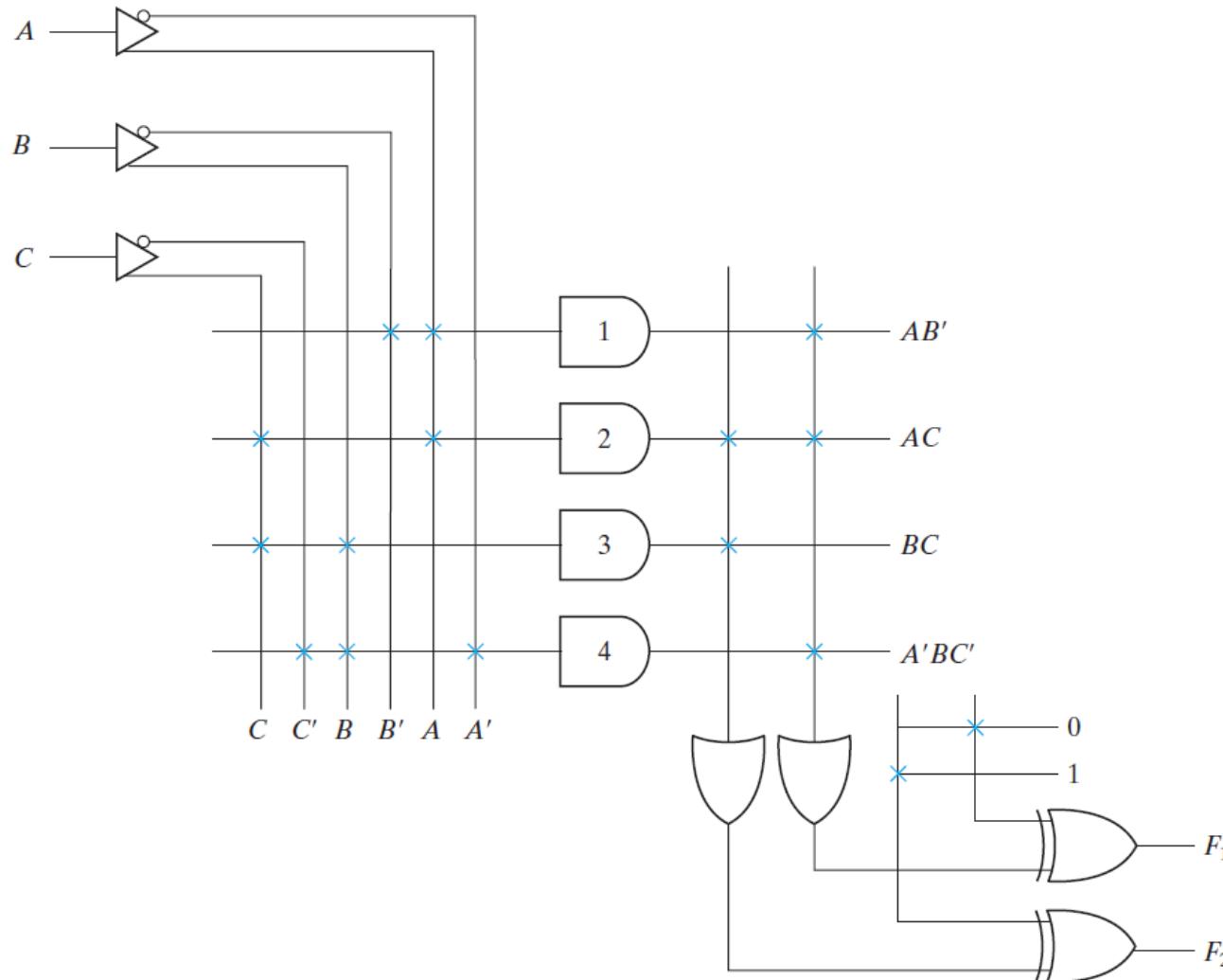


FIGURA 7-14

PLA con tres entradas, cuatro términos producto y dos salidas

Programmable Logic Devices (PLDs)

SECCIÓN 7-7 Arreglo lógico programable

281

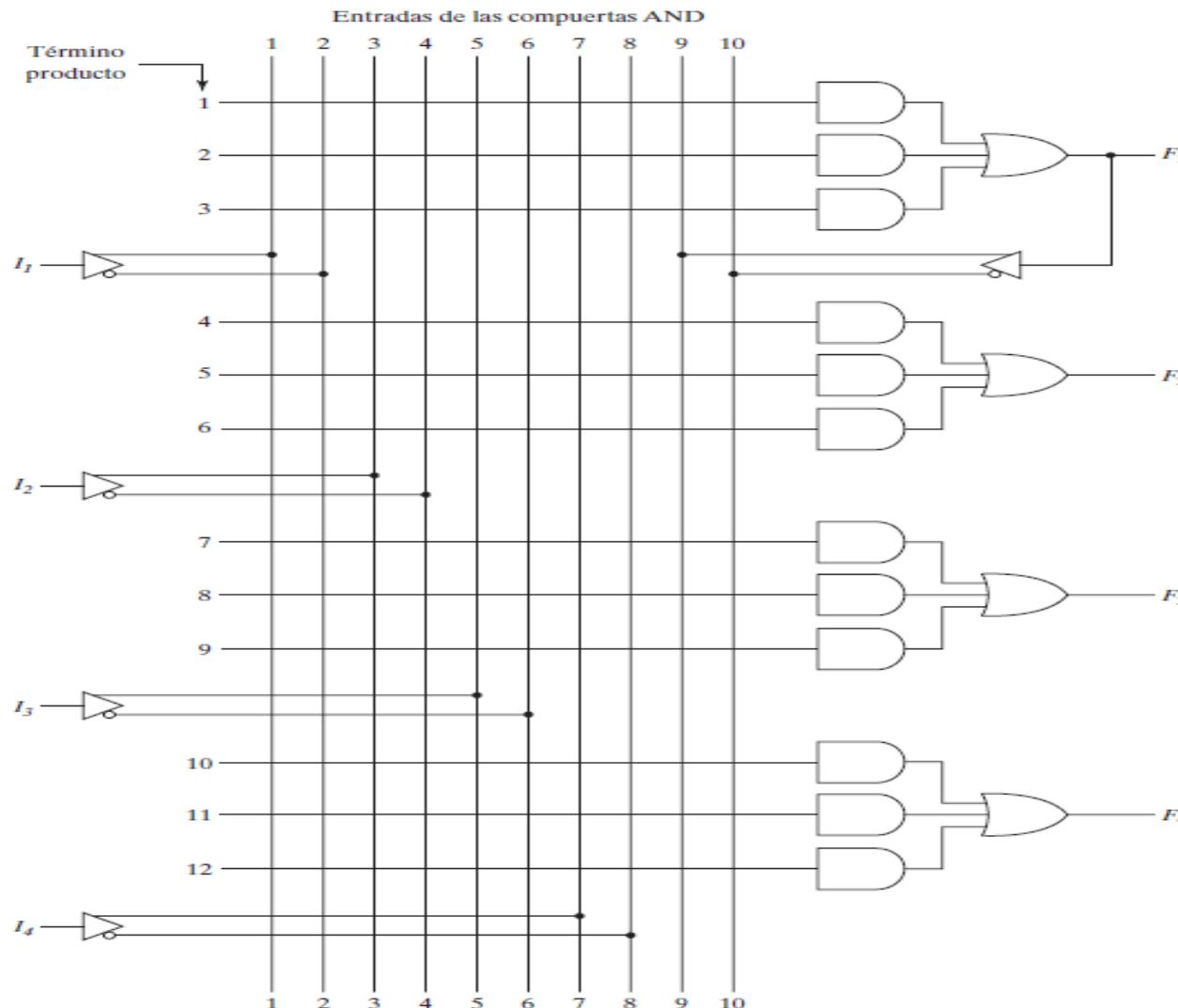


FIGURA 7-16

PAL con cuatro entradas, cuatro salidas y una estructura AND-OR de anchura tres