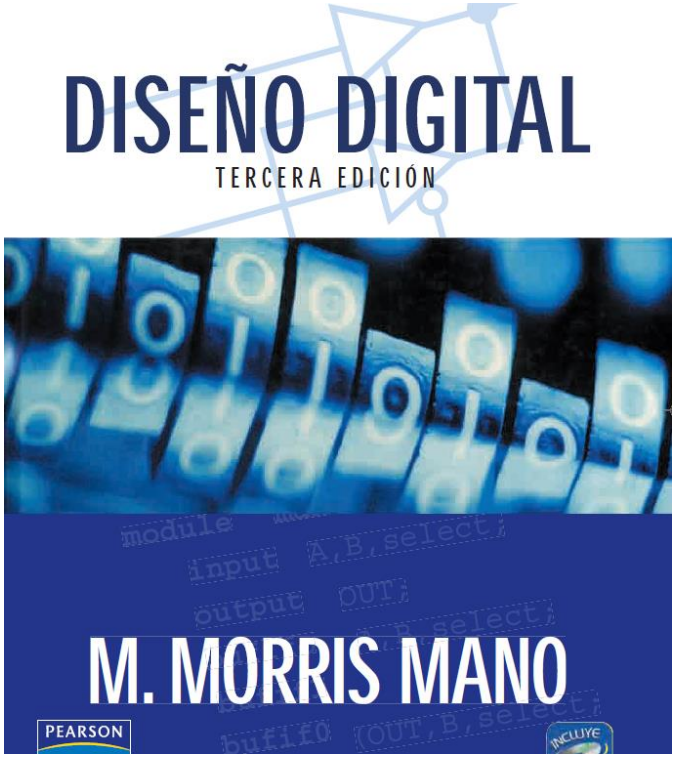
MINIMIZACIÓN DE FUNCIONES LÓGICAS

MÉTODO GRÁFICOS DE KARNAUGH

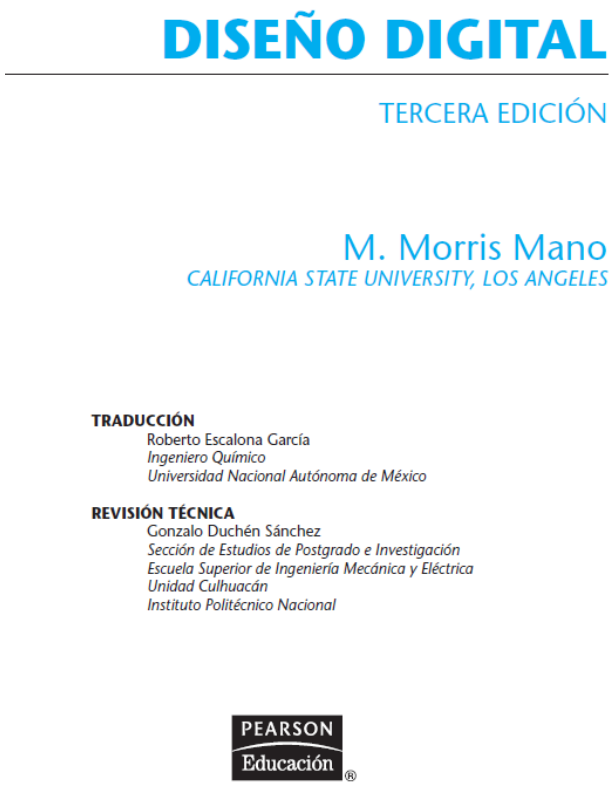
Libro de Base para este Tema

- 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.



Libro de Base para este Tema

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

CONTENIDO

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

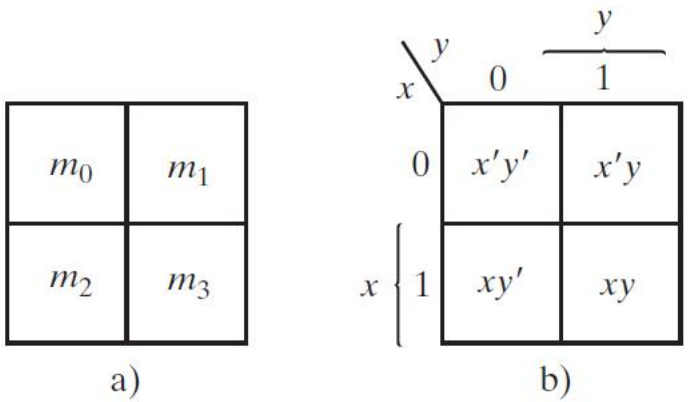


FIGURA 3-1
Mapa de dos variables

Mapa de 2 Variables

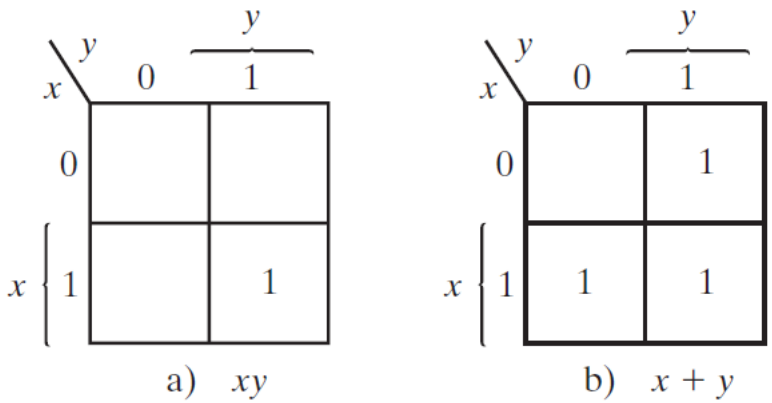


FIGURA 3-2
Representación de funciones en el mapa

Mapa de 3 Variables

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

a)

	xz / y	00	01	11	10
x	0	$x'y'z'$	$x'y'z$	$x'yz$	$x'yz'$
x	1	$xy'z'$	$xy'z$	xyz	xyz'

z

b)

FIGURA 3-3
Mapa de tres variables

Mapa de 3 Variables

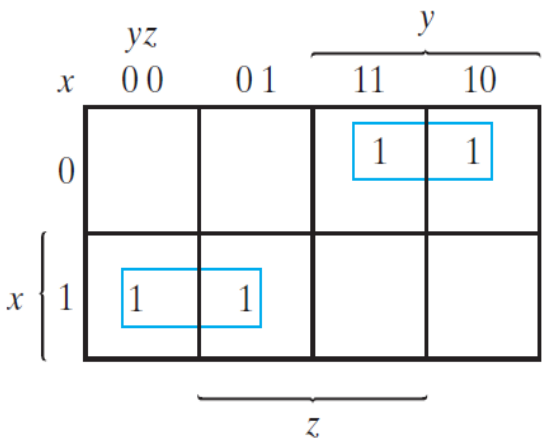


FIGURA 3-4
Mapa para el ejemplo 3-1; $F(x, y, z) = \sum(2, 3, 4, 5) = x'y + xy'$

EJEMPLO 3-1

Simplifique la función booleana

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

Mapa de 3 Variables

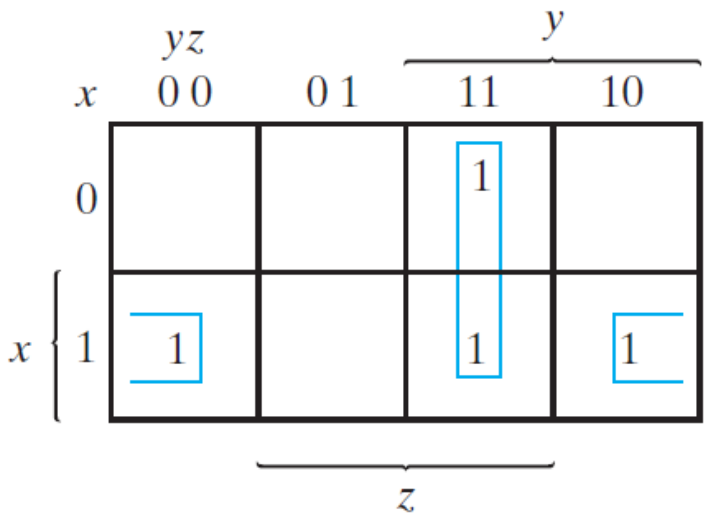


FIGURA 3-5
Mapa para el ejemplo 3-2; $F(x, y, z) = \sum(3, 4, 6, 7) = yz + xz'$

Mapa de 3 Variables

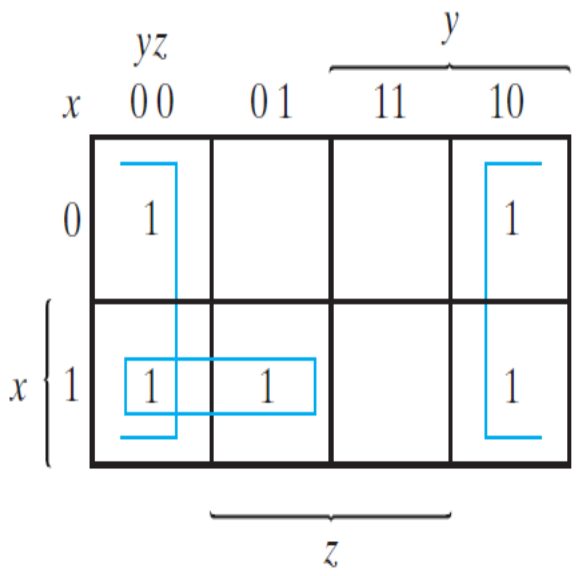


FIGURA 3-6

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

Mapa de 3 Variables

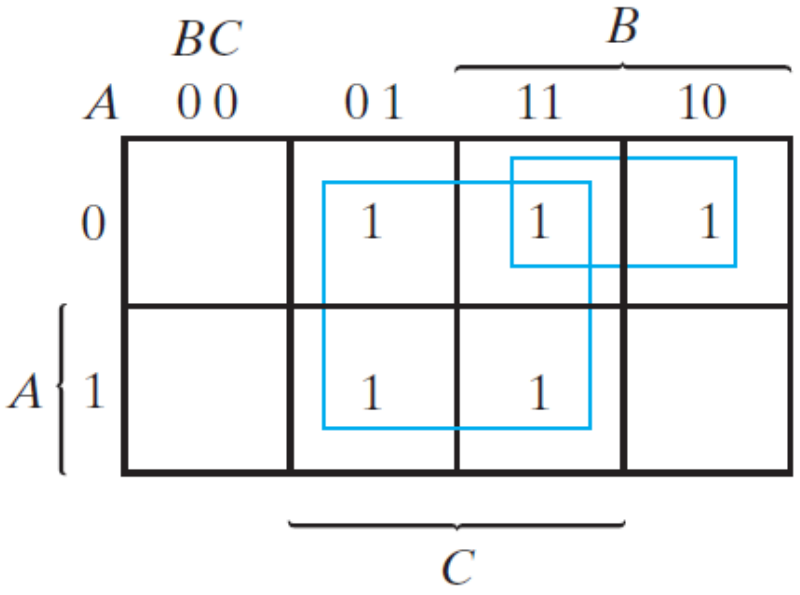
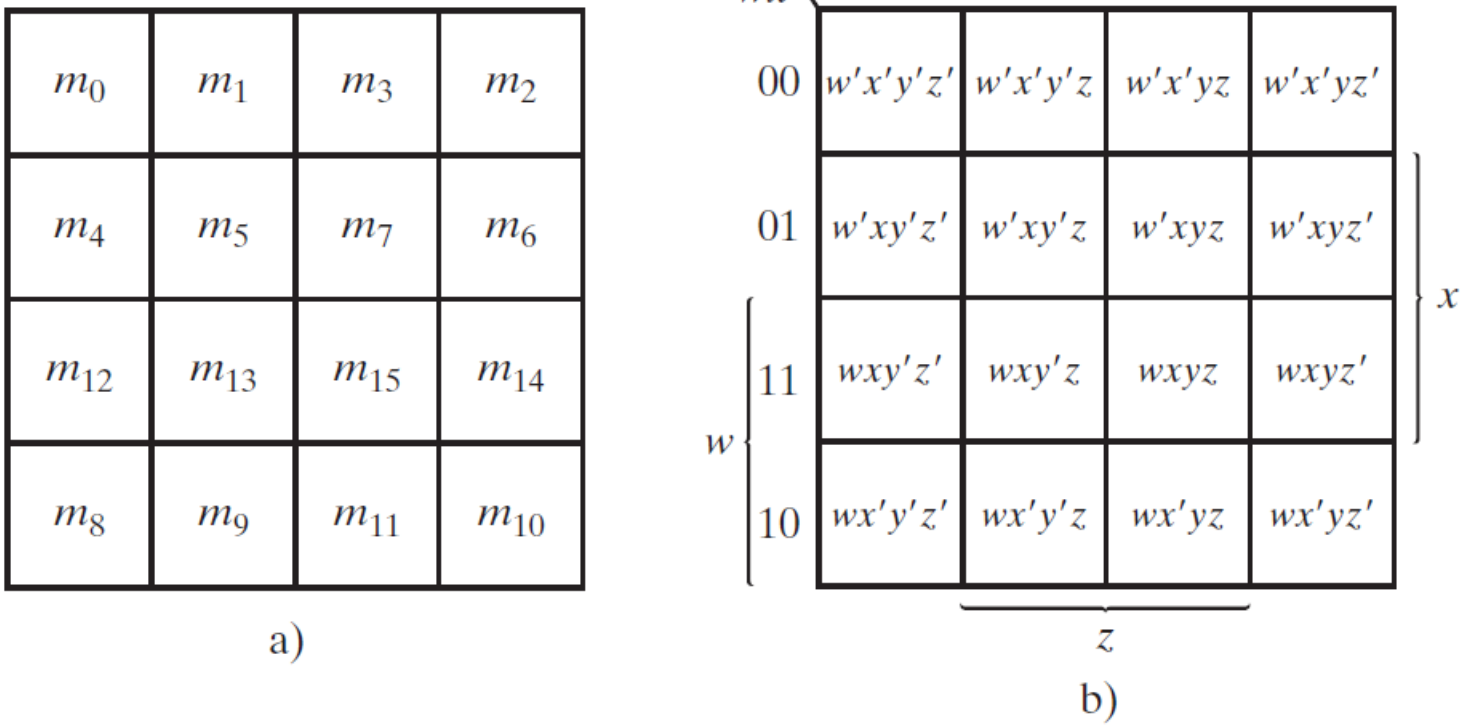


FIGURA 3-7

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)

wx \ yz	00	01	11	10
00	$w'x'y'z'$	$w'x'y'z$	$w'x'yz$	$w'x'yz'$
01	$w'xy'z'$	$w'xy'z$	$w'xyz$	$w'xyz'$
11	$wxy'z'$	$wxy'z$	$wxyz$	$wxyz'$
10	$wx'y'z'$	$wx'y'z$	$wx'yz$	$wx'yz'$

y (11, 10); x (01, 11, 10); w (11, 10); z (01, 11)

b)

FIGURA 3-8
Mapa de cuatro variables

Mapa de 4 Variables

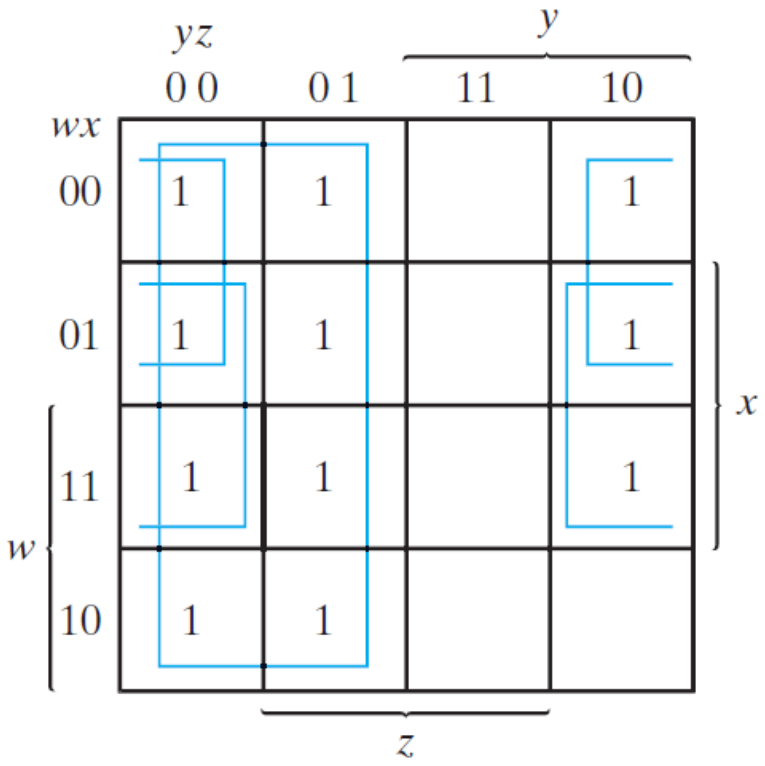


FIGURA 3-9

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

Mapa de 4 Variables

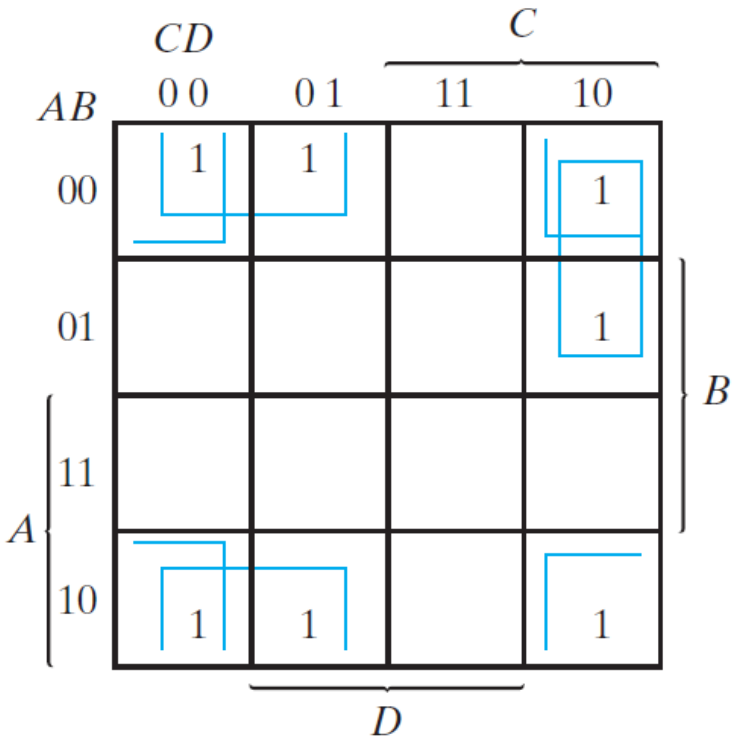
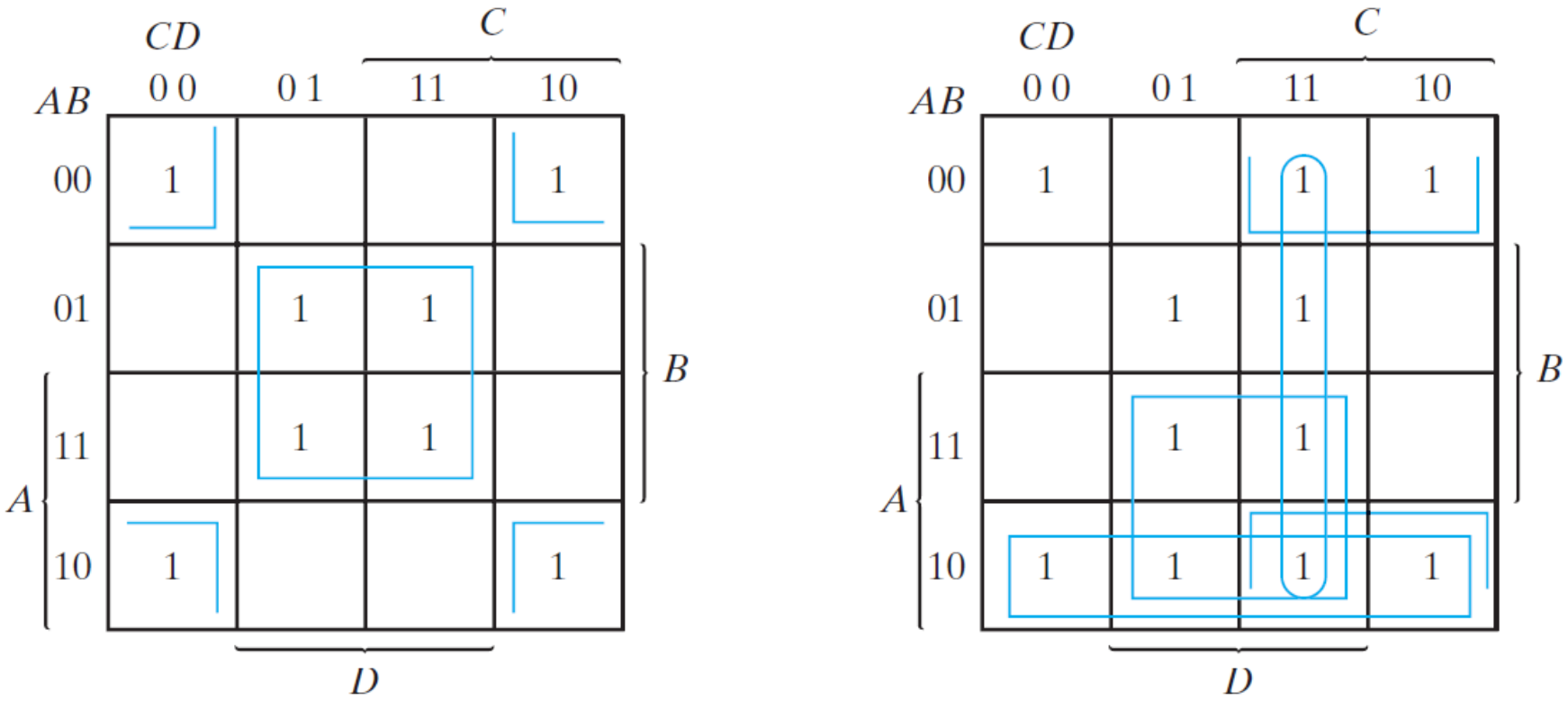


FIGURA 3-10
Mapa para el ejemplo 3-6; $A'B'C' + B'CD' + A'BCD' + AB'C' = B'D' + B'C' + A'CD'$

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

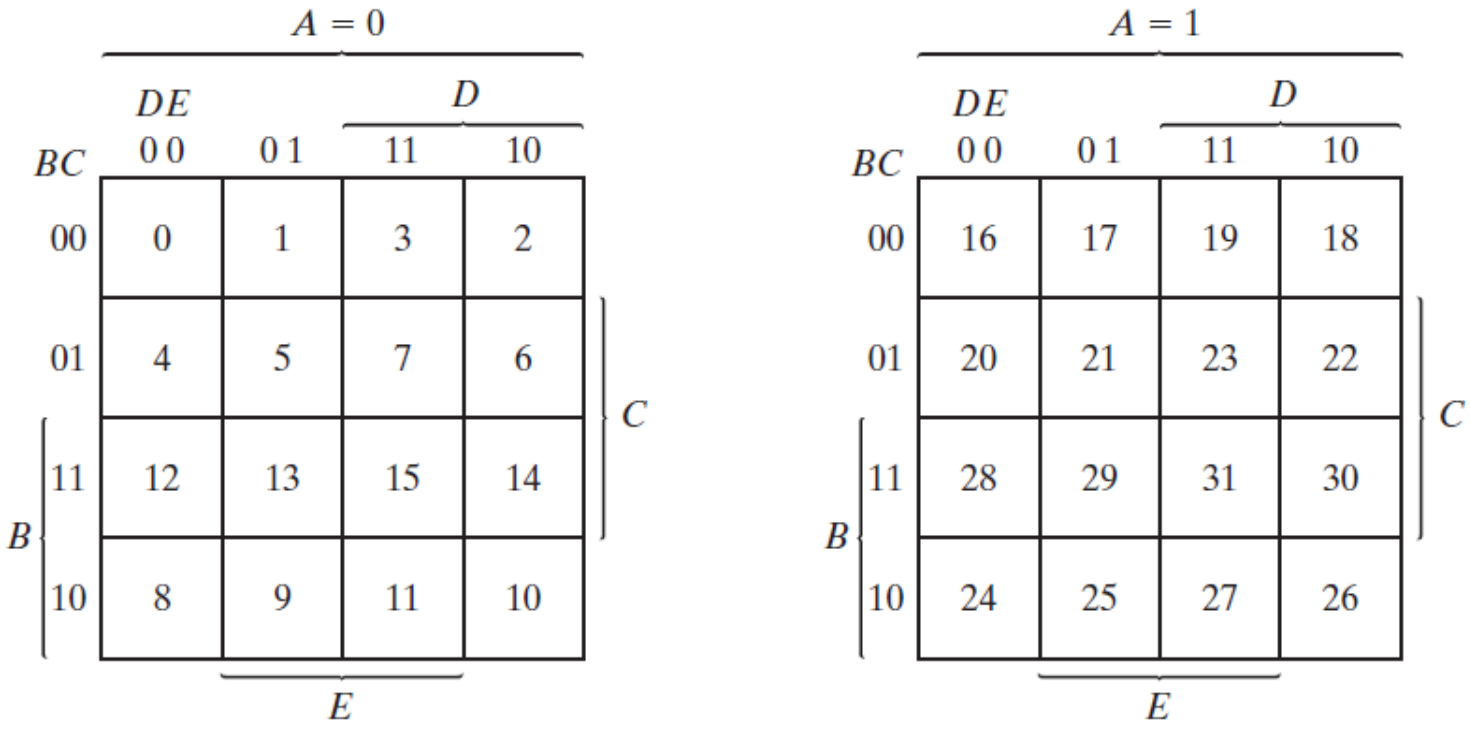


FIGURA 3-12
Mapa de cinco variables

Mapa de 5 Variables

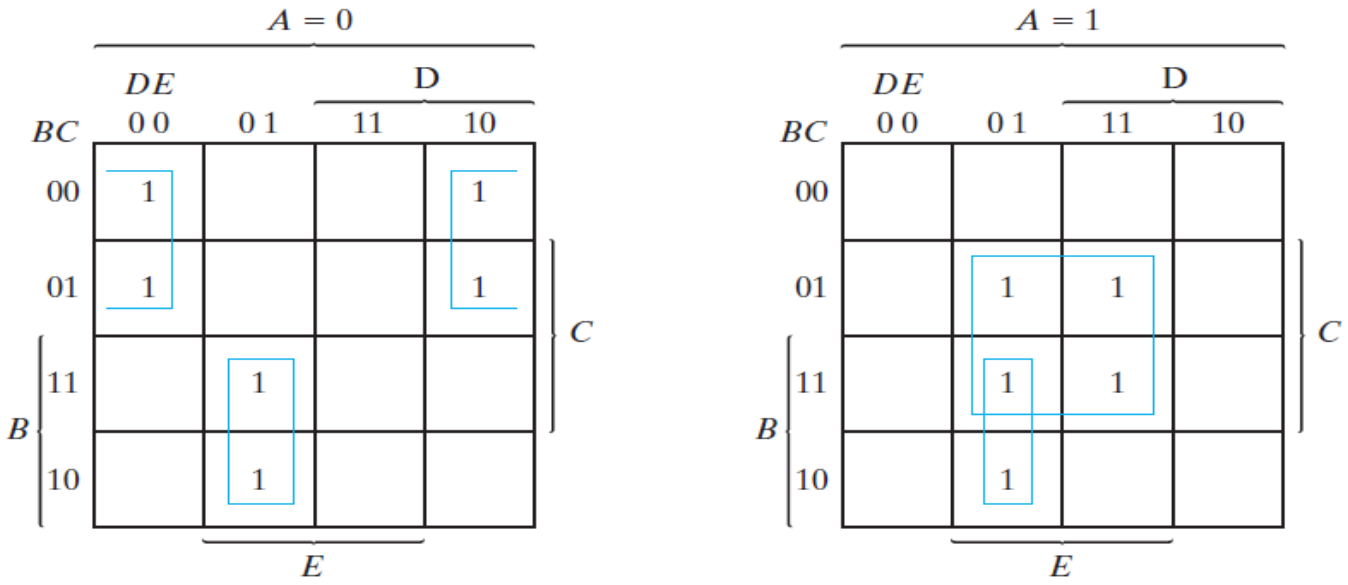


FIGURA 3-13
Mapa para el ejemplo 3-7; $F = A'B'E' + BD'E + ACE$

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

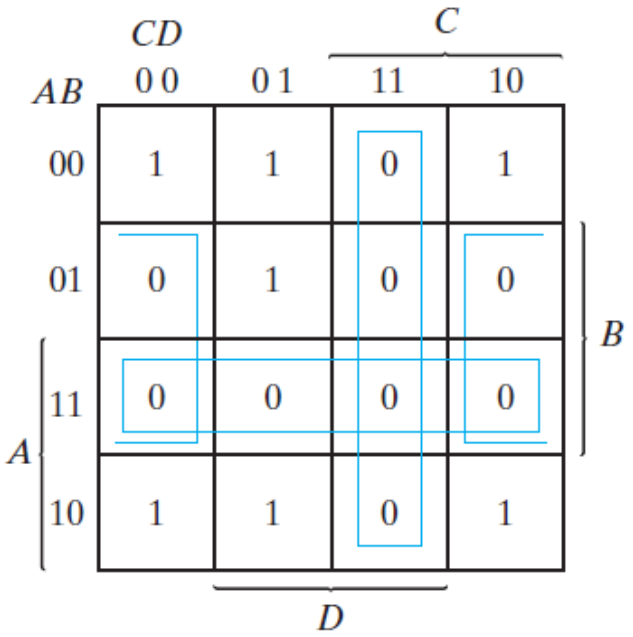
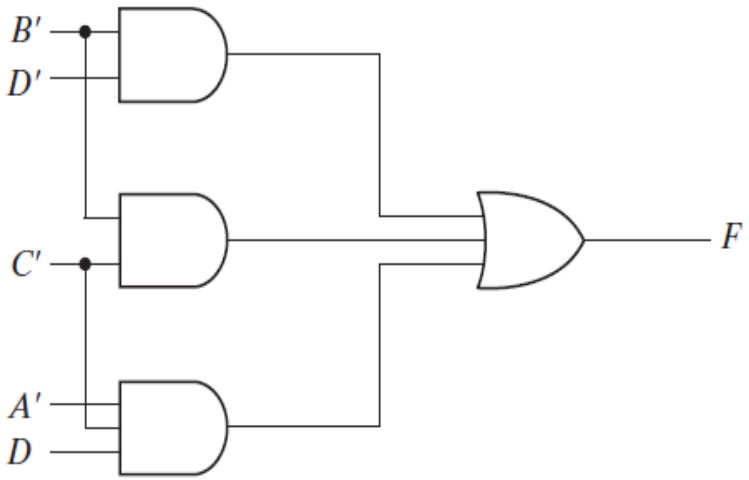
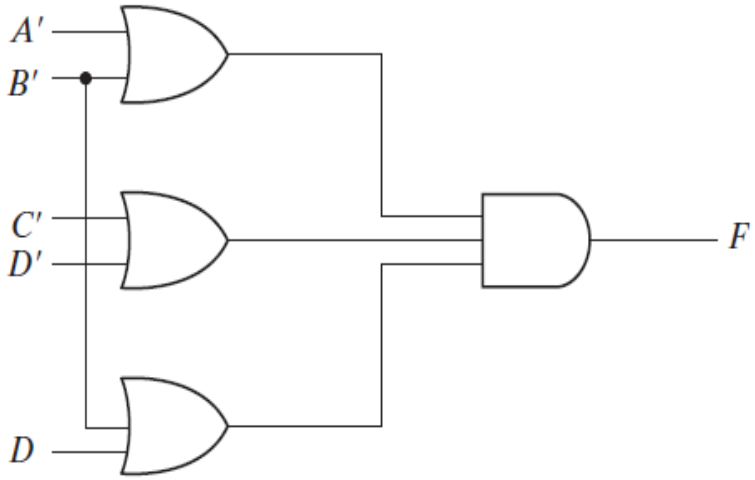


FIGURA 3-14
Mapa para el ejemplo 3-8; $F(A, B, C, D) = \sum(0, 1, 2, 5, 8, 9, 10)$
$= B'D' + B'C' + A'C'D = (A' + B')(C' + D')(B' + D)$

Suma de Productos y Producto de Sumas



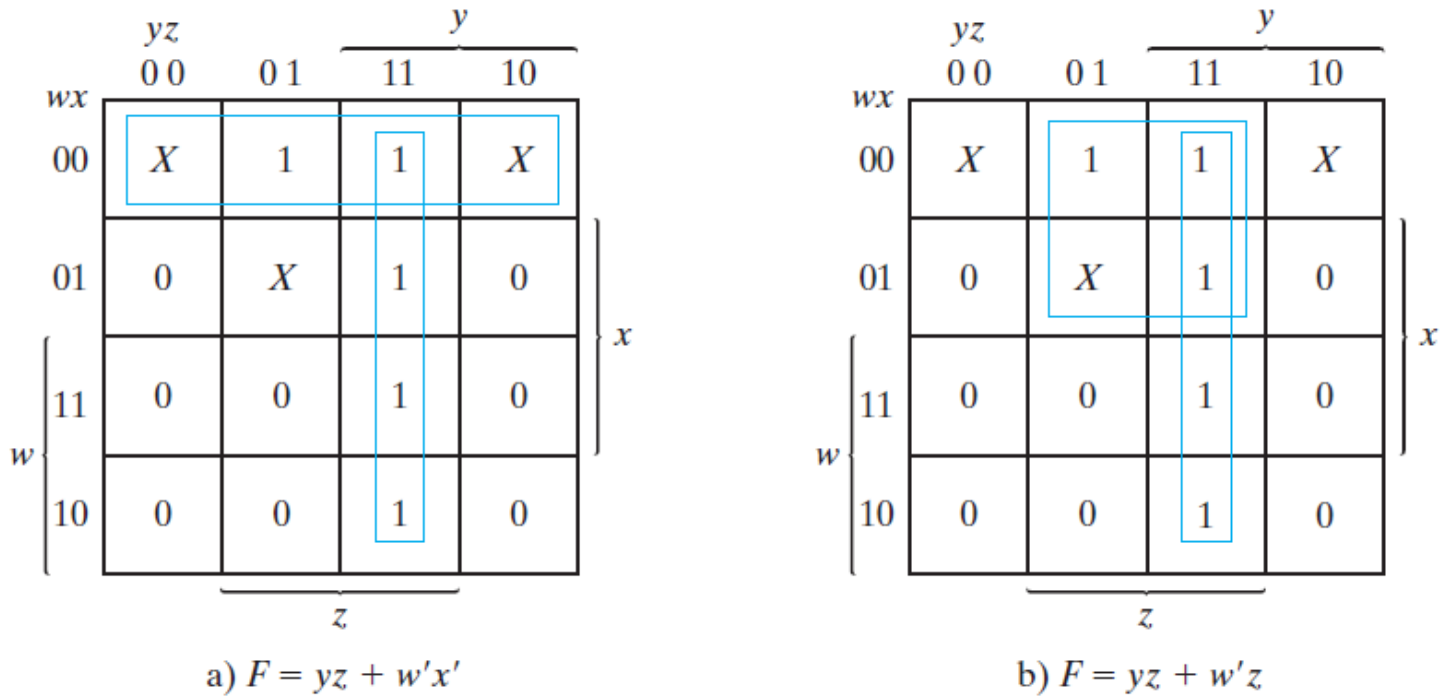
a) $F = B'D' + B'C' + A'C'D$



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)



a) $F = yz + w'x'$

b) $F = yz + w'z$

FIGURA 3-17
Ejemplo con condiciones de indiferencia

Implementación con NAND y NOR

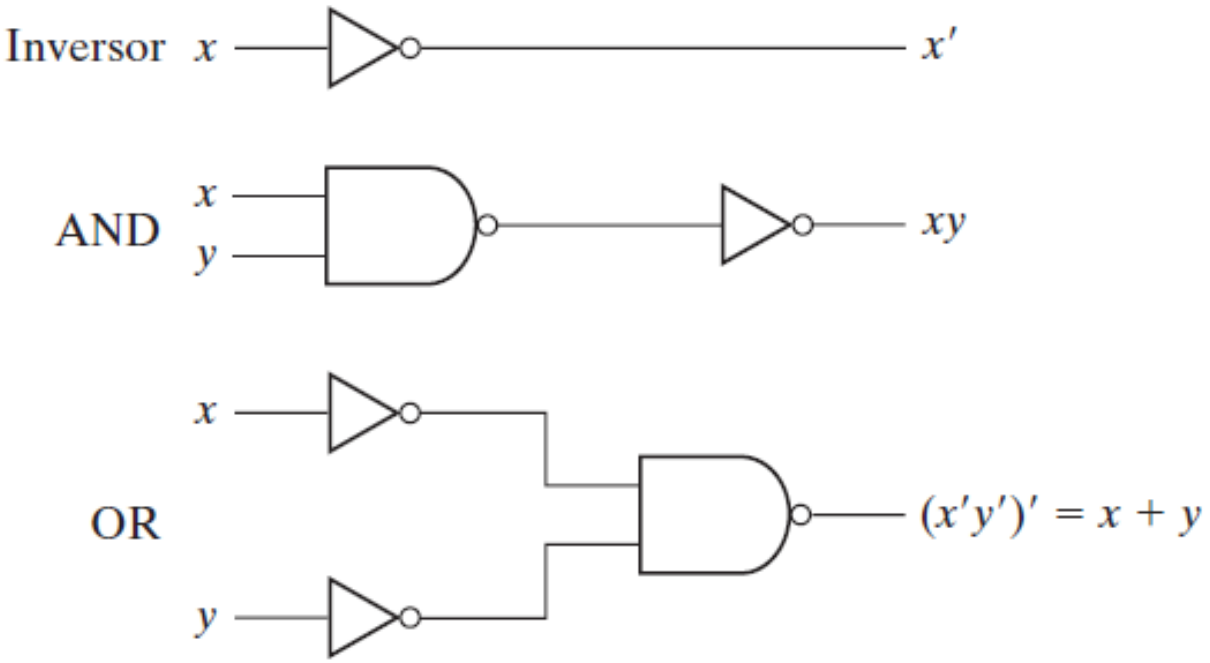
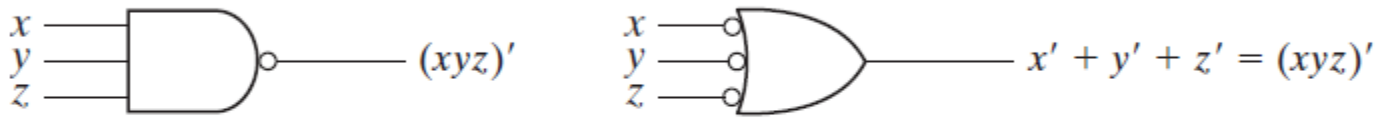


FIGURA 3-18
Operaciones lógicas con compuertas NAND

Implementación con NAND y NOR



a) AND–invertir b) Invertir–OR

FIGURA 3-19
Dos símbolos gráficos para la compuerta NAND

Implementación con NAND y NOR

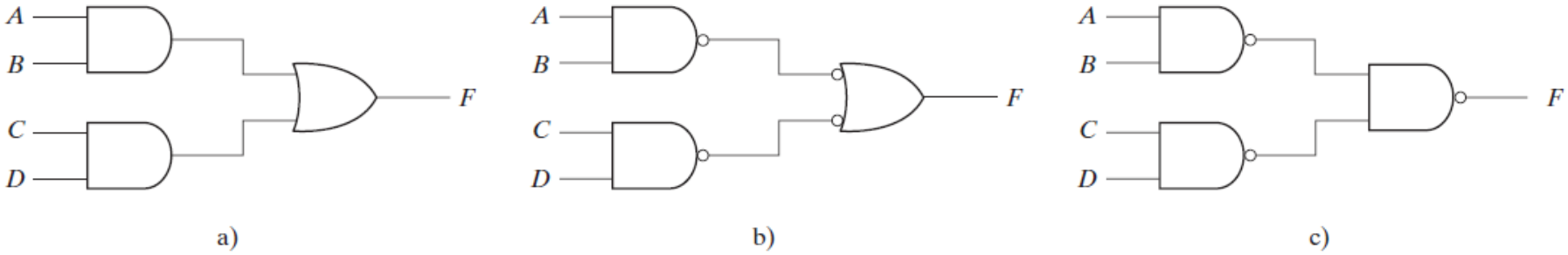


FIGURA 3-20
Tres formas de implementar $F = AB + CD$

Implementación con Nand y Nor

- También es una aplicación directa de De Morgan

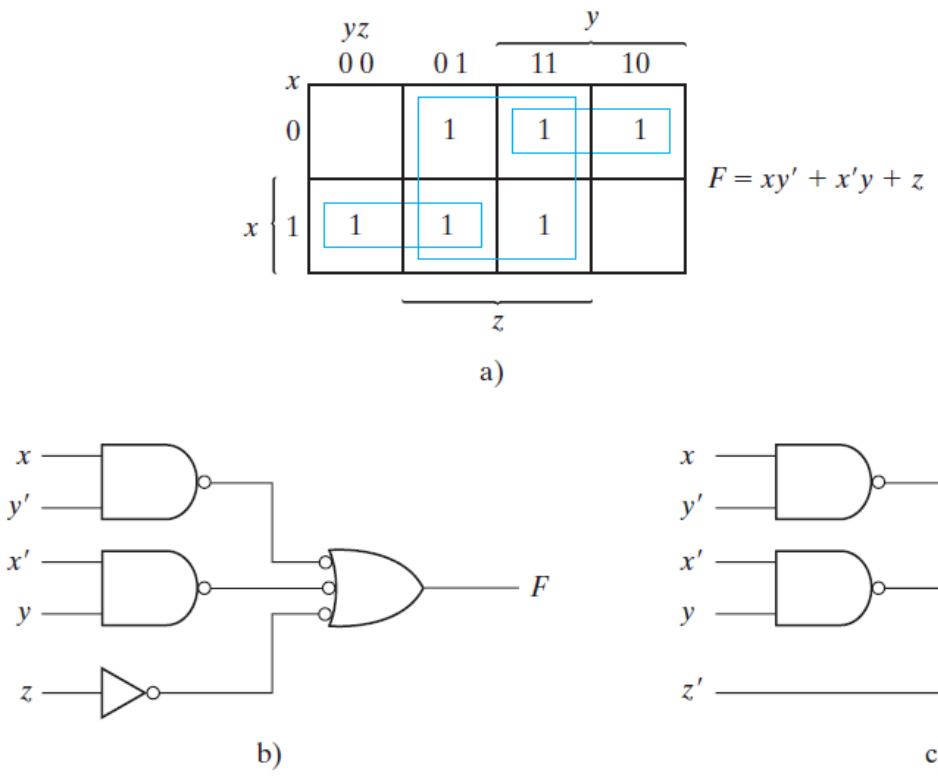
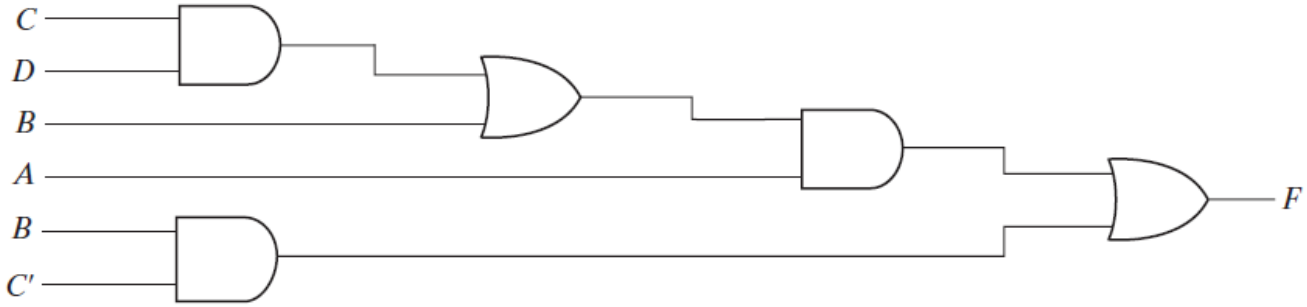
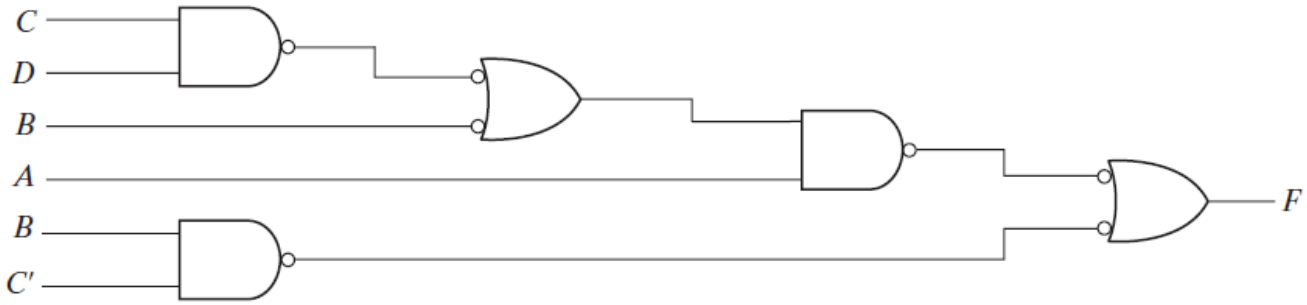


FIGURA 3-21
Solución del ejemplo 3-10

Implementación con Nand y Nor



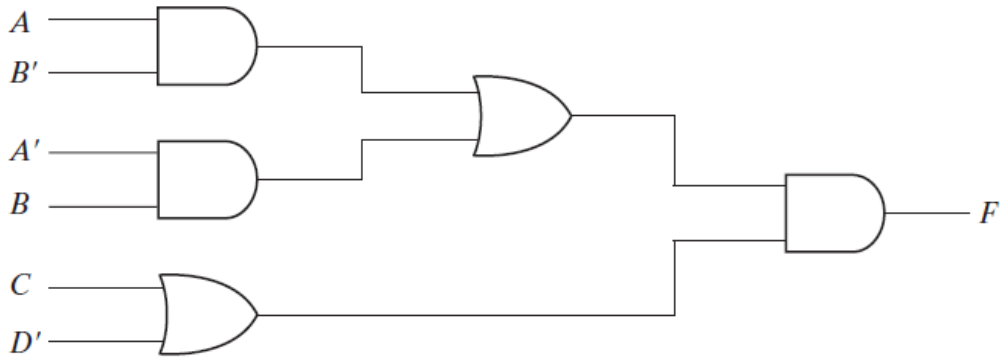
a) Compuertas AND-OR



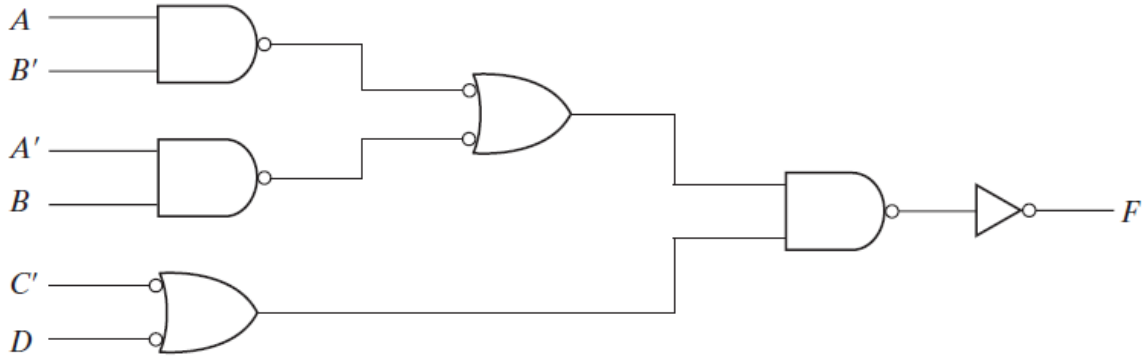
b) Compuertas NAND

FIGURA 3-22
Implementación de $F = A(CD + B) + BC'$

Implementación con Nand y Nor



a) Compuertas AND-OR



b) Compuertas NAND

FIGURA 3-23
Implementación de $F = (AB' + A'B)(C + D')$

Implementación con Nand y Nor

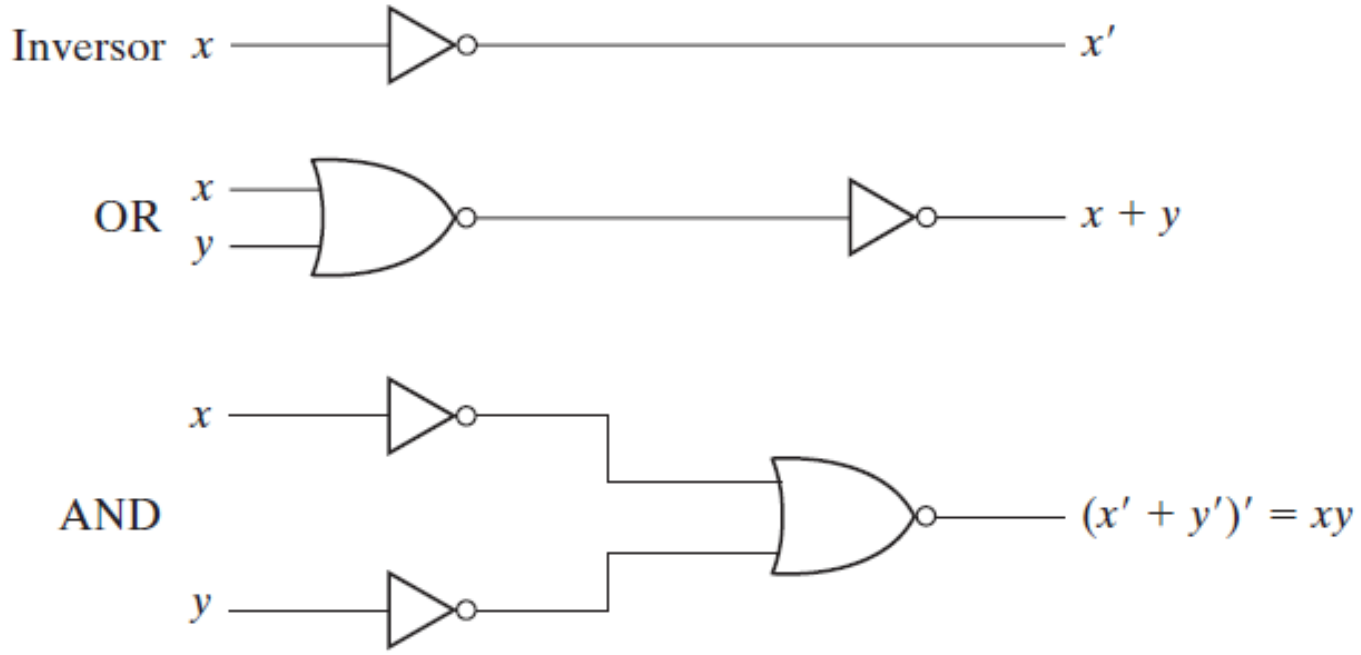
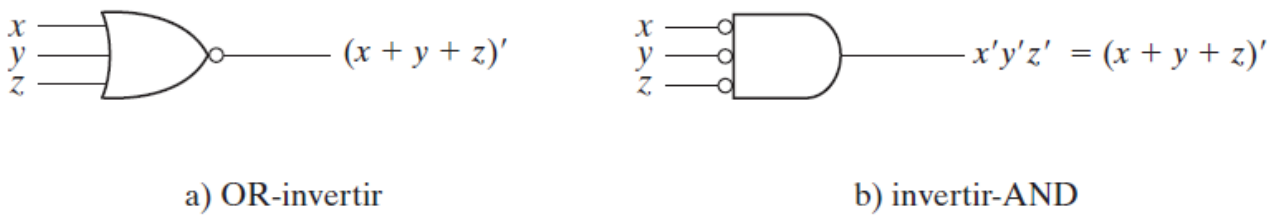


FIGURA 3-24
Operaciones lógicas con compuertas NOR

Implementación con Nand y Nor



a) OR-invertir b) invertir-AND

FIGURA 3-25
Dos símbolos gráficos para la compuerta NOR

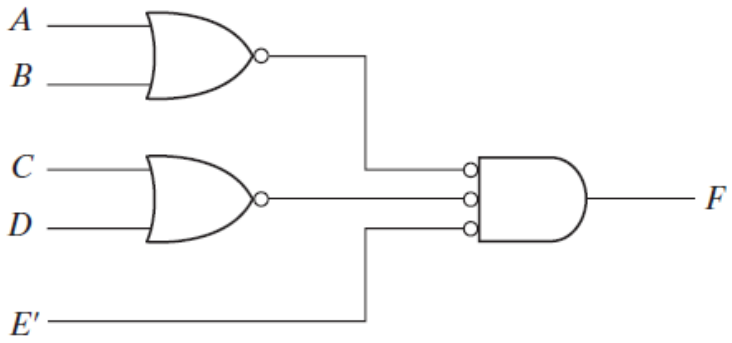


FIGURA 3-26
Implementación de $F = (A + B)(C + D)E$

Otras Implementaciones

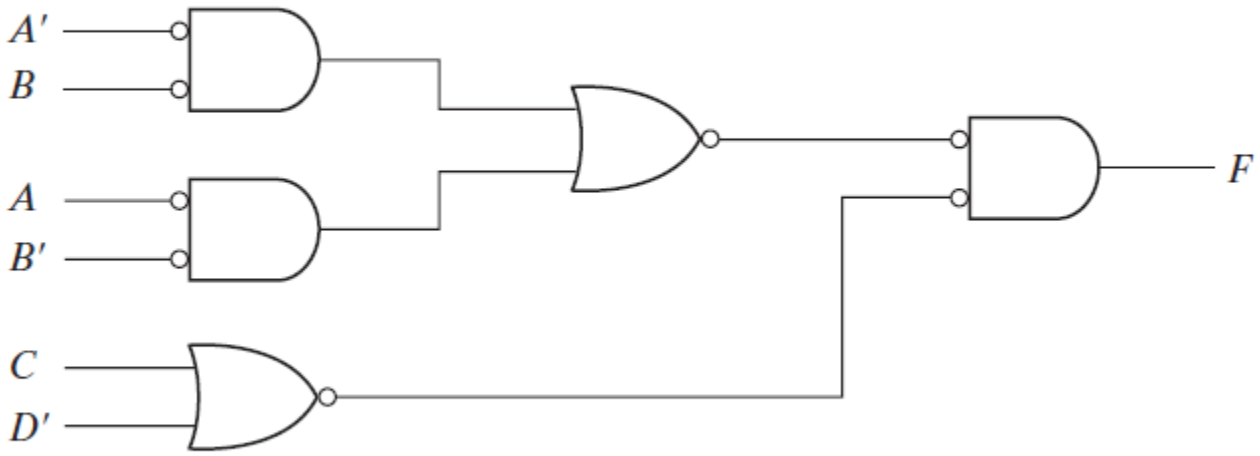
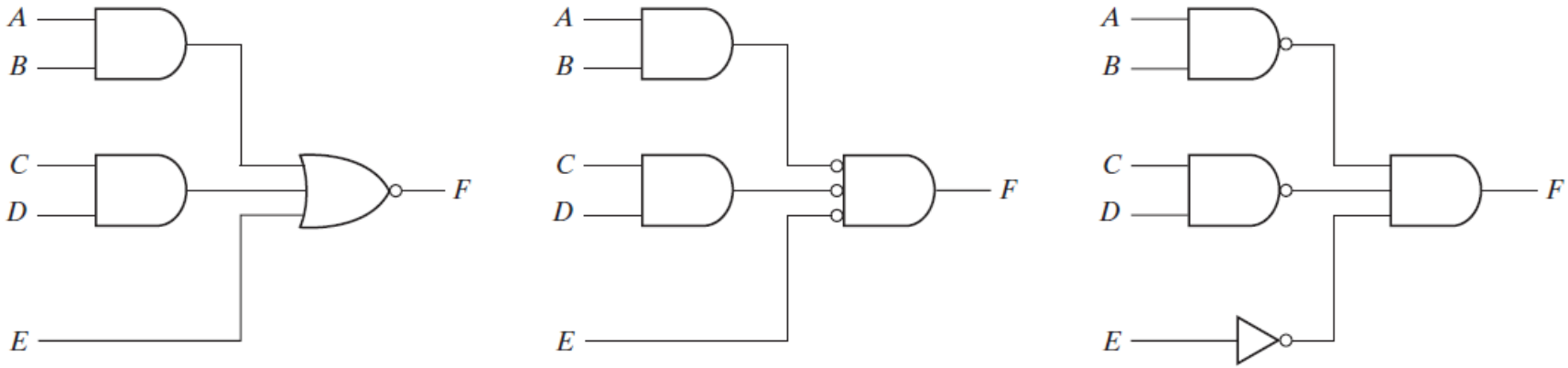


FIGURA 3-27
Implementación de $F = (AB' + A'B)(C + D')$ con compuertas NOR

Otras Implementaciones

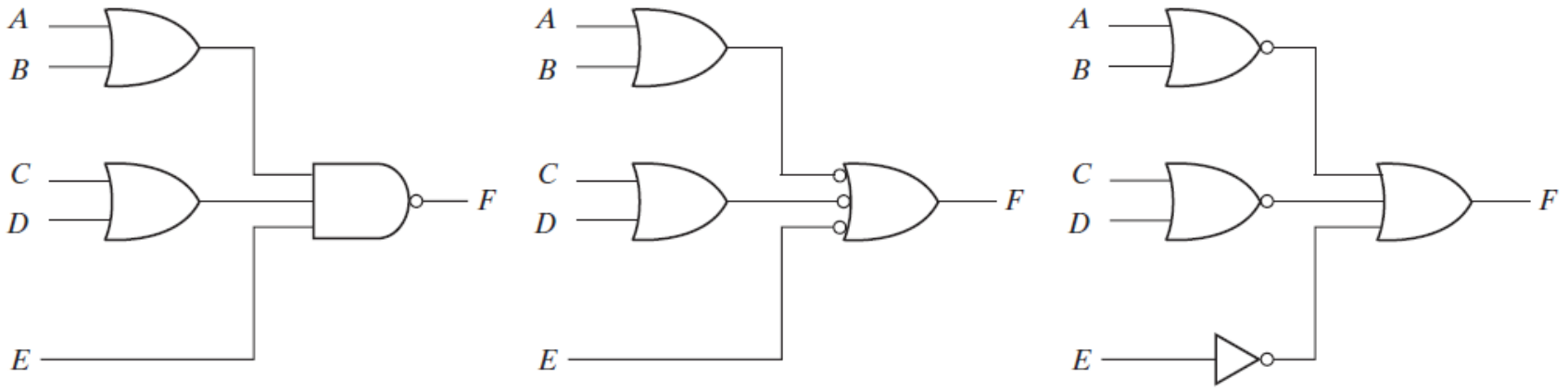


a) AND-NOR b) AND-NOR c) NAND-AND

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:

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

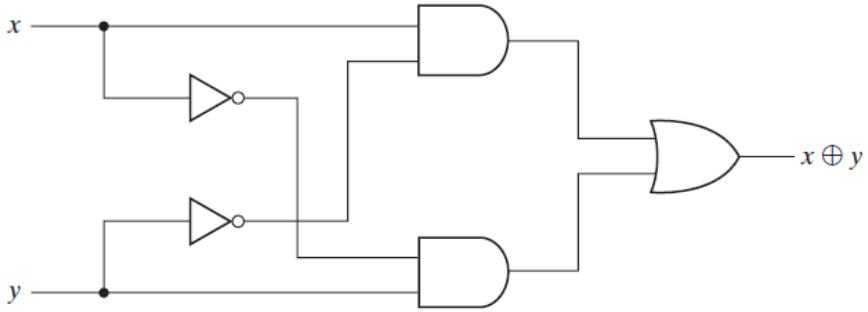
Es igual a 1 si sólo x es igual a 1 o sólo y es igual a 1, pero no si ambas son 1. El NOR exclusivo, también llamado equivalencia, realiza la operación booleana siguiente:

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

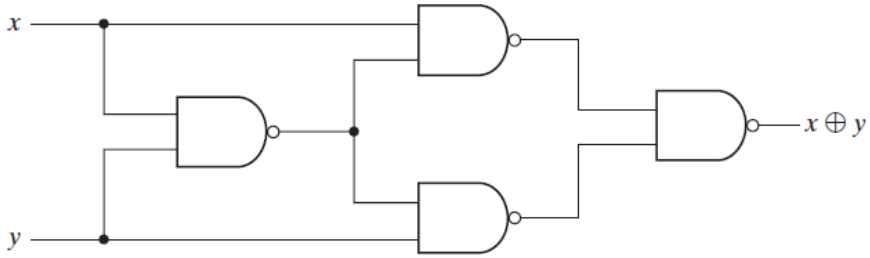
Es igual a 1 si tanto x como y son 1 o si ambas son 0. Se puede demostrar que el NOR exclusivo es el complemento del OR exclusivo con la ayuda de una tabla de verdad o por manipulación algebraica:

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

Función O-Exclusivo



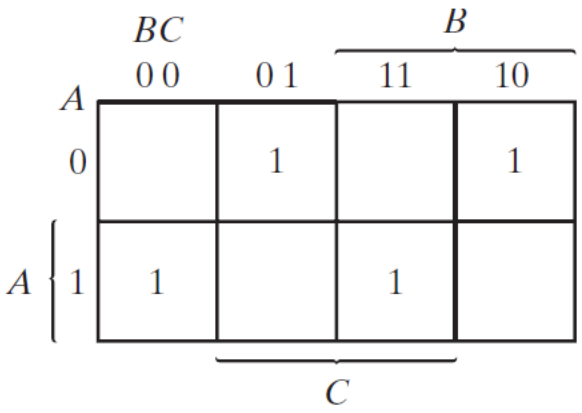
a) Con compuertas AND-OR-NOT



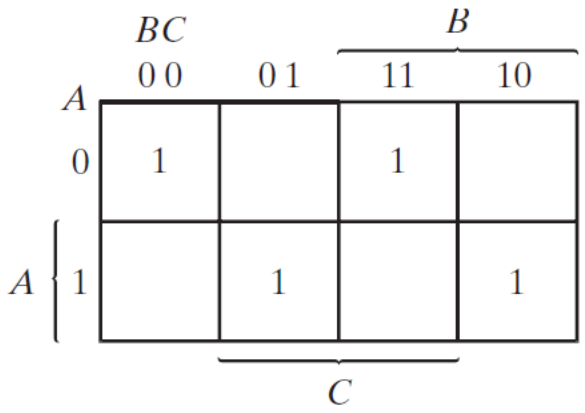
b) Con compuertas NAND

FIGURA 3-32
Implementaciones del OR exclusivo

Funciones Par e Impar



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

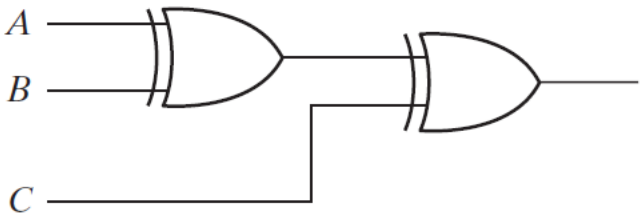


b) Función par
$F = (A \oplus B \oplus C)'$

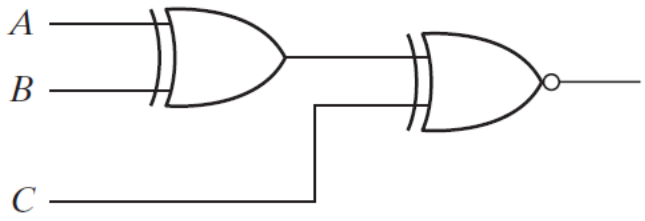
FIGURA 3-33
Mapa para una función OR exclusivo de tres variables

Funciones Par e Impar

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



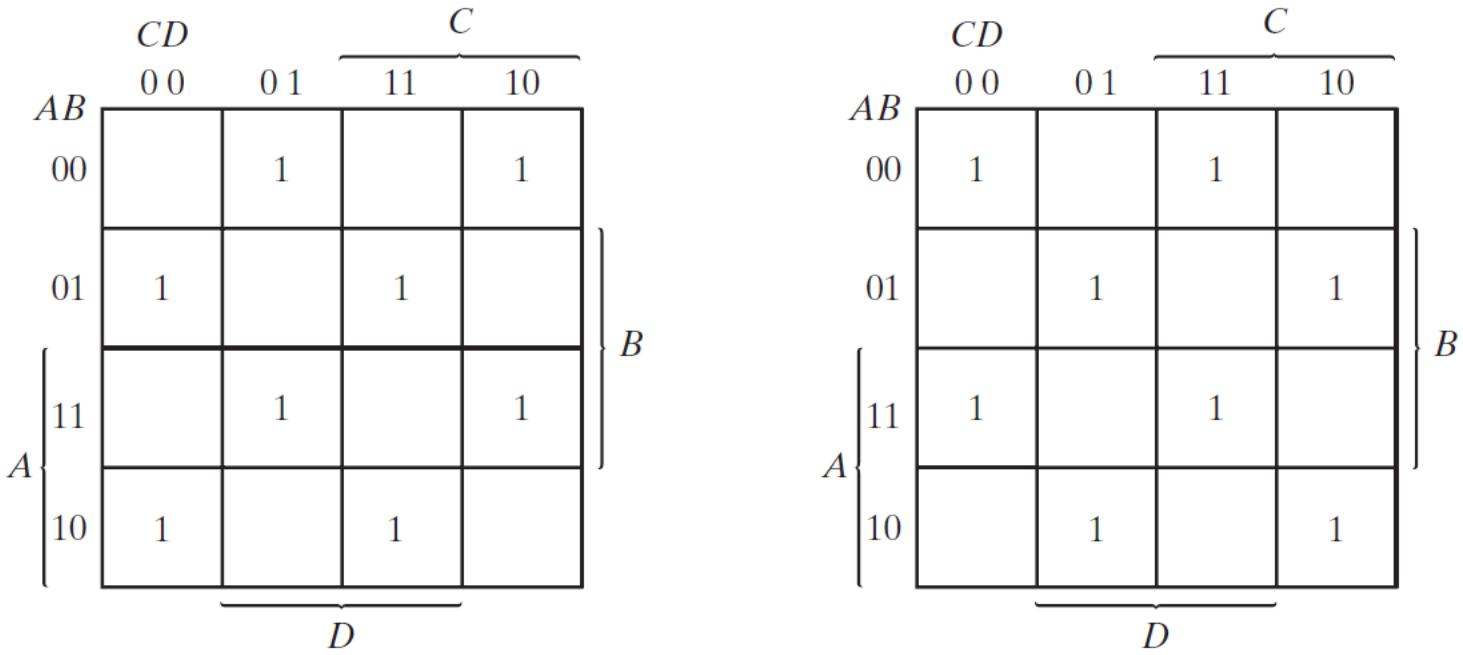
a) Función impar de tres entradas



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

$$A \oplus B \oplus C \oplus D = (AB' + A'B) \oplus (CD' + C'D)$$

$$= (AB' + A'B)(CD + C'D') + (AB + A'B')(CD' + C'D)$$

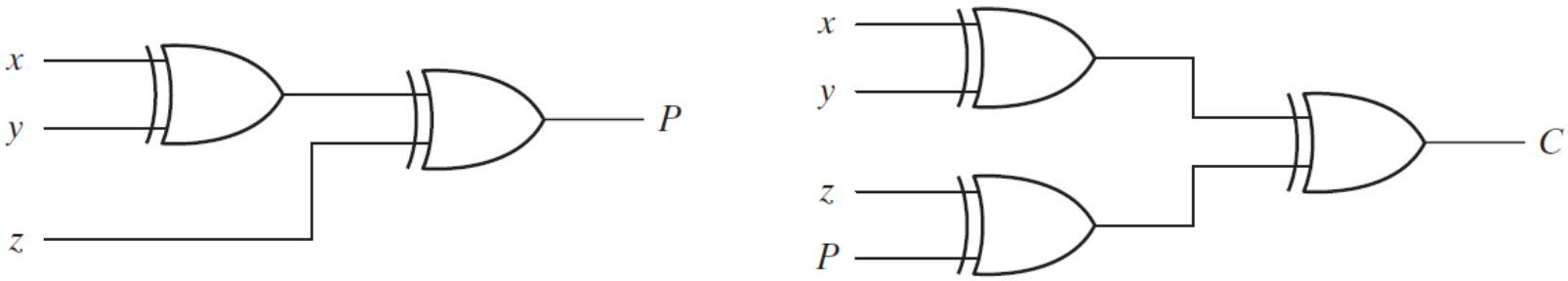
$$= \sum(1, 2, 4, 7, 8, 11, 13, 14)$$

Generador de Paridad Par

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;
    outputx,y;
    wire e;
    and g1(e,A,B);
    not g2(y, C);
    or g3(x,e,y);
endmodule
```

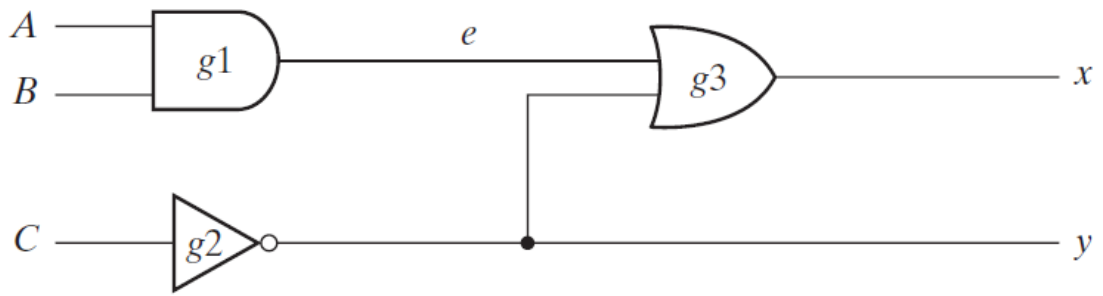


FIGURA 3-37
Circuito para ilustrar HDL

Lenguajes de Descripción de Hardware

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	Entrada	Salida
	(ns)	ABC	*y* e *x*
Inicial	–	000	1 0 1
Cambio	–	111	1 0 1
	10	111	0 0 1
	20	111	0 0 1
	30	111	0 1 0
	40	111	0 1 0
	50	111	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

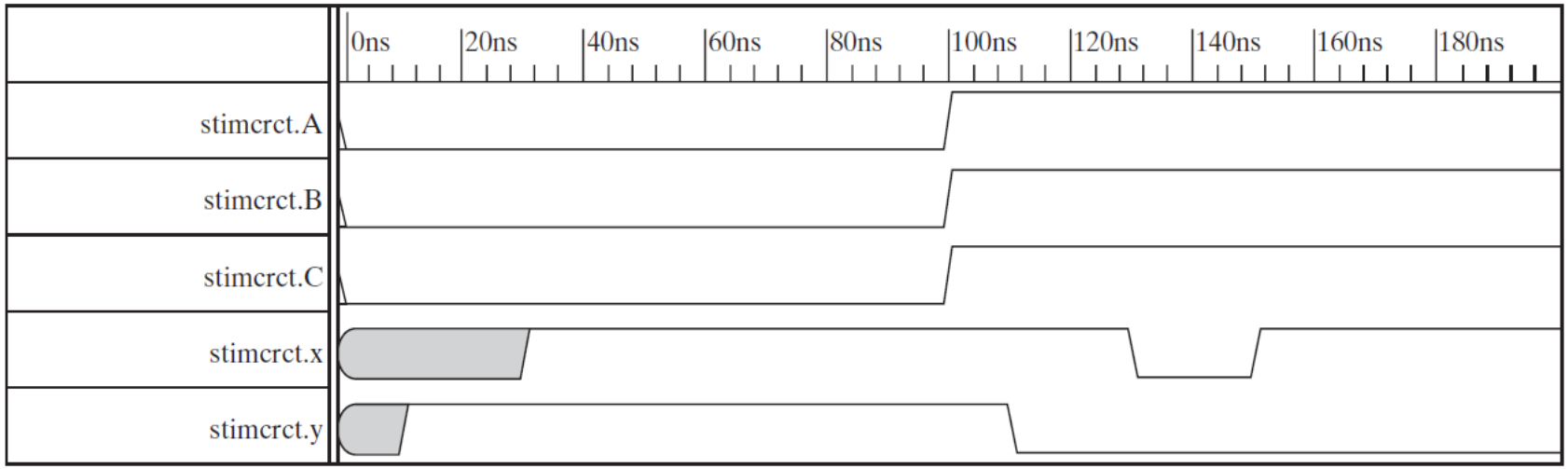


FIGURA 3-38
Salida de la simulación del ejemplo HDL 3-3

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 Hardware

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
```