
CIRCUITOS DIGITALES

2-1 LOGICA Y COMPUERTAS BINARIAS

Los circuitos digitales son componentes de *hardware* que manipulan información binaria. Los circuitos se construyen con partes electrónicas como transistores, diodos y resistores. Cada circuito recibe el nombre de *compuerta*. El diseñador de un sistema digital no tiene que ocuparse de la construcción interna de las compuertas individuales, sino sólo de sus propiedades lógicas externas. Cada compuerta realiza una operación lógica específica, y la salida de una compuerta se aplica a las entradas de otras compuertas, en secuencia, para formar el circuito digital requerido.

A fin de describir las propiedades operacionales de los circuitos digitales, es necesario presentar una notación matemática que especifique la operación de cada compuerta. Este sistema matemático es un sistema lógico binario que se conoce como *álgebra booleana*, en honor del matemático inglés George Boole, quien en 1854 publicó un libro donde se presentaba la teoría matemática de la lógica. Hoy día, el álgebra booleana se utiliza para describir la interconexión de compuertas digitales y para transformar diagramas de circuitos en expresiones algebraicas. Primero se presentará el concepto de lógica binaria y se mostrará su relación con compuertas digitales y señales binarias. Después presentaremos las propiedades del álgebra booleana junto con otros métodos de diseño para tratar diversos aspectos de los circuitos y sistemas digitales.

Lógica binaria

La lógica binaria tiene que ver con variables que asumen dos valores discretos y con operaciones que asumen un significado lógico. A los dos valores que toman las va-

riables se les pueden dar nombres diferentes, pero para nuestros fines conviene pensar en términos de valores binarios y asignar 1 y 0 a cada variable. Las variables son designadas por letras del alfabeto como A , B , C , X , Y , Z . Existen tres operaciones lógicas asociadas con los valores binarios llamadas AND, OR y NOT.

1. AND. Esta operación se representa por un punto o por la ausencia de un operador. Por ejemplo, $X \cdot Y = Z$ o $XY = Z$ se leen como “ X y Y es igual a Z ”. La operación lógica AND se interpreta como $Z = 1$ si y sólo si $X = 1$ y $Y = 1$; de lo contrario $Z = 0$. (Recuérdese que X , Y y Z son variables binarias y pueden ser iguales a 1 ó 0 nada más.)
2. OR. Esta operación está representada por un símbolo o signo más (+). Por ejemplo, $X + Y = Z$ se lee “ X o Y es igual a Z ”, lo que significa que $Z = 1$ si $X = 1$ o $Y = 1$, o si $X = 1$ y $Y = 1$. Sólo si $X = 0$ y $Y = 0$, es $Z = 0$.
3. NOT. Esta operación se representa por medio de una barra colocada arriba de una variable. Por ejemplo, $\bar{X} = Z$ se lee “no X es igual a Z ”, que quiere decir que Z es lo que no es X . Dicho de otra manera, si $X = 1$, entonces $Z = 0$; pero si $X = 0$, entonces $Z = 1$. La operación NOT se conoce también como operación de *complemento*, porque cambia un 1 por 0 y un 0 por 1.

La lógica binaria se parece a la aritmética binaria, y las operaciones AND y OR tienen similitudes con la multiplicación y la adición o suma, respectivamente. De hecho, los símbolos que se utilizan para AND y OR son los mismos que se emplean para la multiplicación y la adición. Sin embargo, la lógica binaria no debe confundirse con la aritmética binaria. Se debe comprender que una variable aritmética designa un número que puede constar de muchos dígitos. Una variable lógica es siempre un 1 o un 0. Los posibles valores binarios de la operación OR lógica son los siguientes:

$$0 + 0 = 0$$

$$0 + 1 = 1$$

$$1 + 0 = 1$$

$$1 + 1 = 1$$

Estas operaciones se asemejan a la suma binaria excepto en la última operación. En la lógica binaria se tiene $1 + 1 = 1$ (que se lee “uno o uno es igual a uno”), pero en la aritmética binaria se tiene $1 + 1 = 10$ (que se lee “uno más uno es igual a dos”). Para evitar que haya ambigüedad, a veces se emplea el símbolo \vee para denotar la operación OR en vez del signo +. Pero en tanto que no se mezclen las operaciones aritméticas y lógicas, cada una puede utilizar el signo de + con su significado independiente.

Los valores binarios de la operación AND son

$$0 \cdot 0 = 0$$

$$0 \cdot 1 = 0$$

$$1 \cdot 0 = 0$$

$$1 \cdot 1 = 1$$

Esta es idéntica a la multiplicación binaria siempre que se use sólo un bit. La operación AND lógica se conoce a veces como *multiplicación lógica*, y la operación OR lógica como *adición o suma lógica*.

TABLA 2—1
Tablas de verdad de las tres operaciones lógicas

AND			OR			NOT	
X	Y	X·Y	X	Y	X + Y	X	\bar{X}
0	0	0	0	0	0	0	1
0	1	0	0	1	1	1	0
1	0	0	1	0	1		
1	1	1	1	1	1		

Para cada combinación de los valores de las variables binarias, tales como *X* y *Y*, existe un valor de *Z* especificado por la definición de la operación lógica. Estas definiciones se pueden acomodar en forma compacta en una *tabla de verdad*, que es una disposición de combinaciones de las variables binarias que muestra la relación entre los valores que pueden tomar las variables y el resultado de la operación. Las tablas de verdad de las operaciones AND, OR y NOT se ilustran en la tabla 2-1, y presentan todos los valores posibles de las variables y los resultados de la operación. Estas tablas demuestran con claridad la definición de las tres operaciones.

Compuertas lógicas

Las compuertas lógicas son circuitos electrónicos que operan con una o más señales de entrada para producir una señal de salida. Existen señales como voltajes (o tensiones) o corrientes eléctricas en un sistema digital en uno u otro de dos valores reconocibles. Los circuitos operados por tensión responden a dos niveles de voltaje independientes que representan una variable binaria igual a un 1 lógico o 0 lógico. Por ejemplo, un sistema digital en particular puede definir el cero lógico como una señal igual a 0 voltios, y el 1 lógico como una señal igual a 4 voltios. En la práctica, cada nivel de voltaje tiene un intervalo aceptable como se indica en la figura 2-1. Las terminales de entrada de circuitos digitales aceptan señales binarias dentro del intervalo admisible y responden en las terminales de salida con señales binarias que entran en

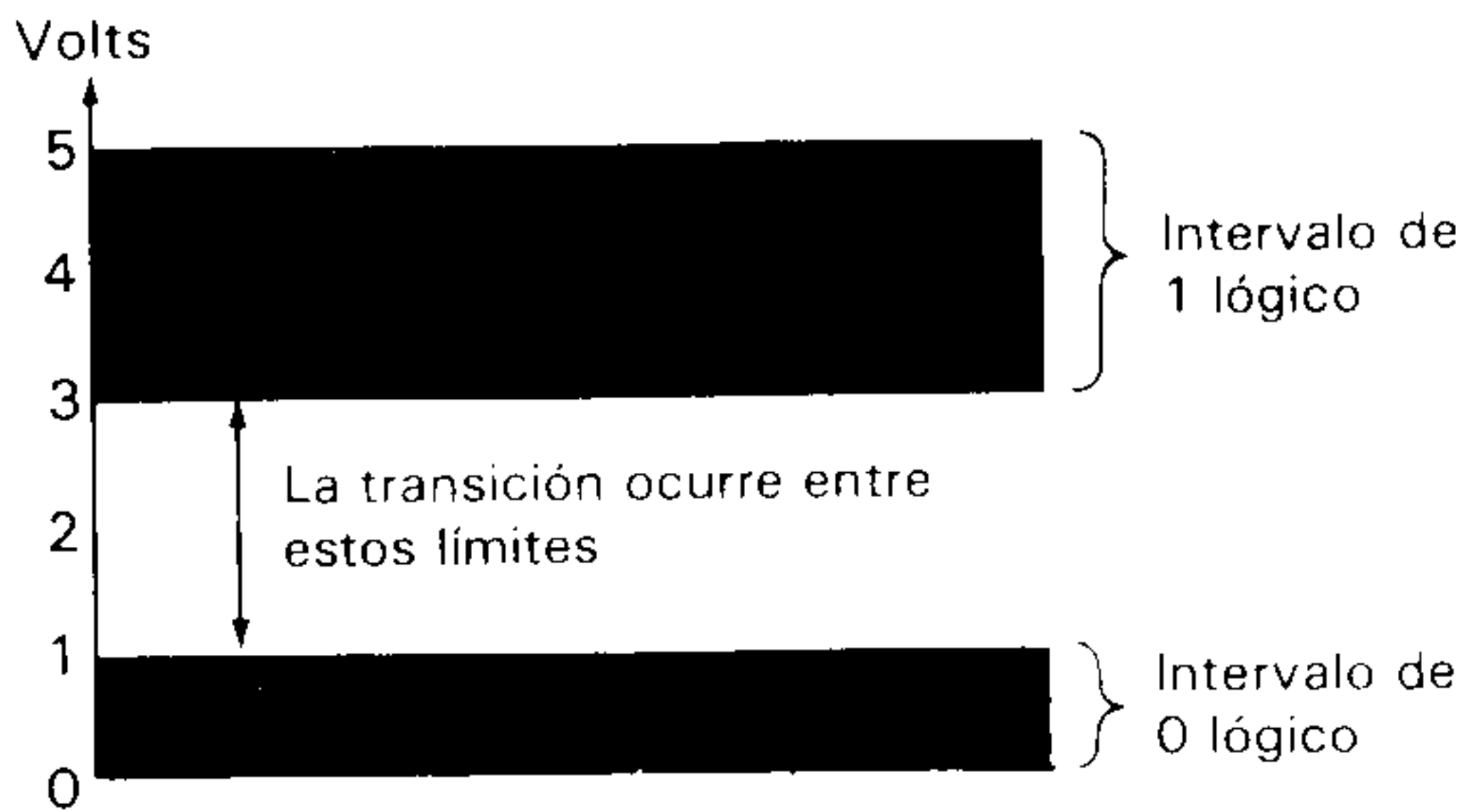
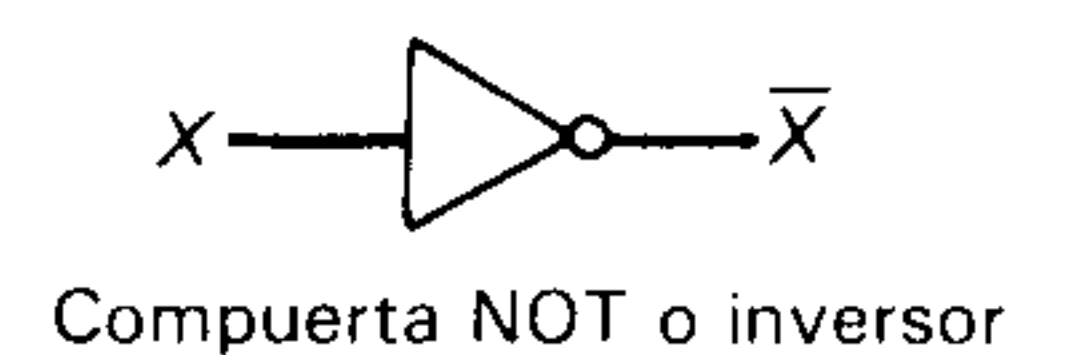
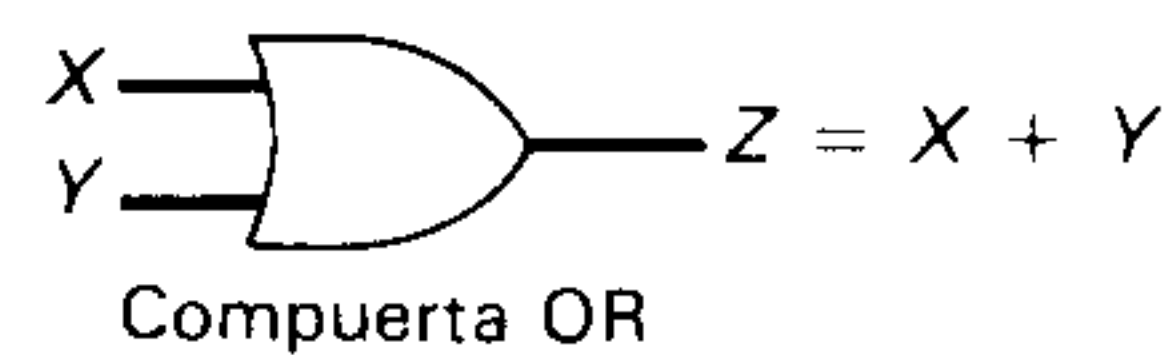
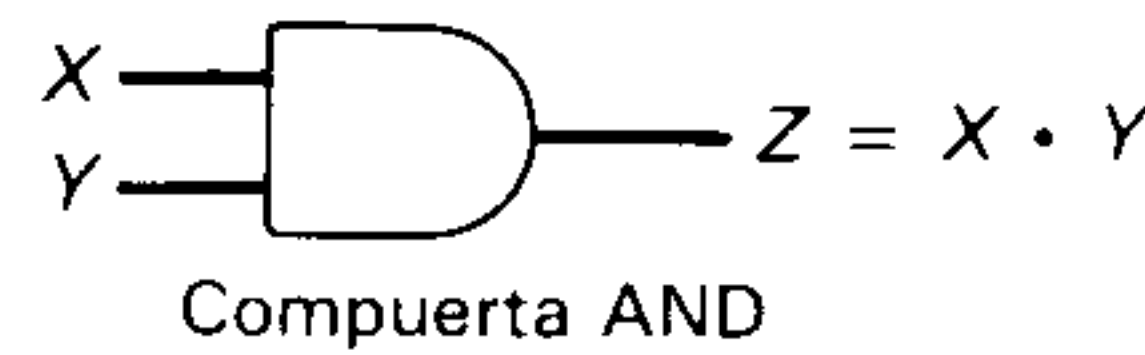
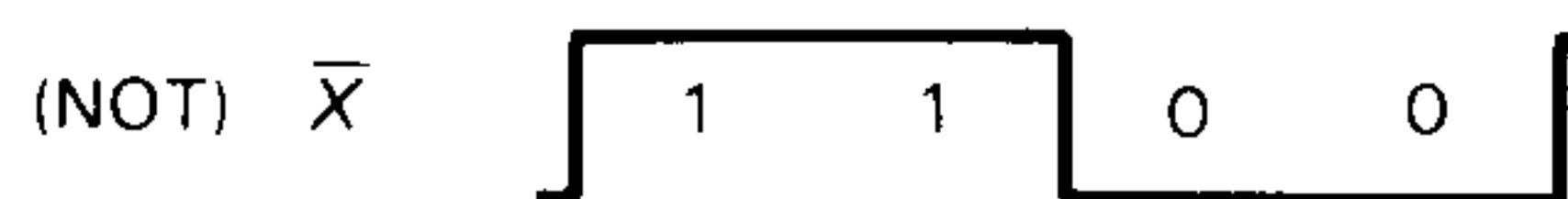
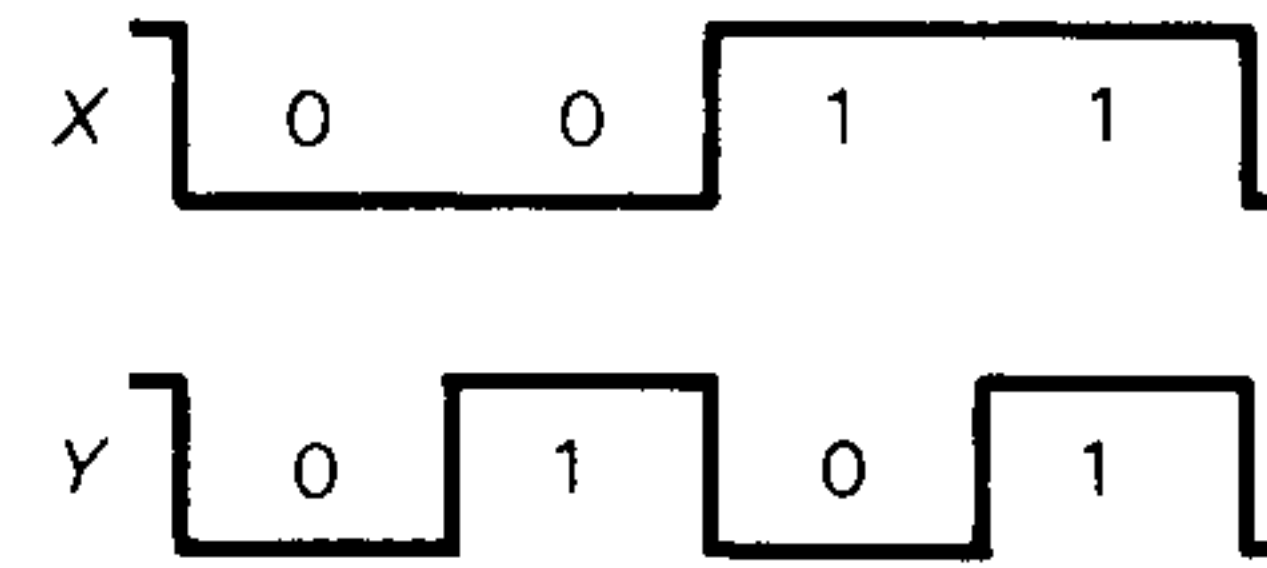


FIGURA 2—1
Ejemplo de señales binarias



(a) Símbolos gráficos



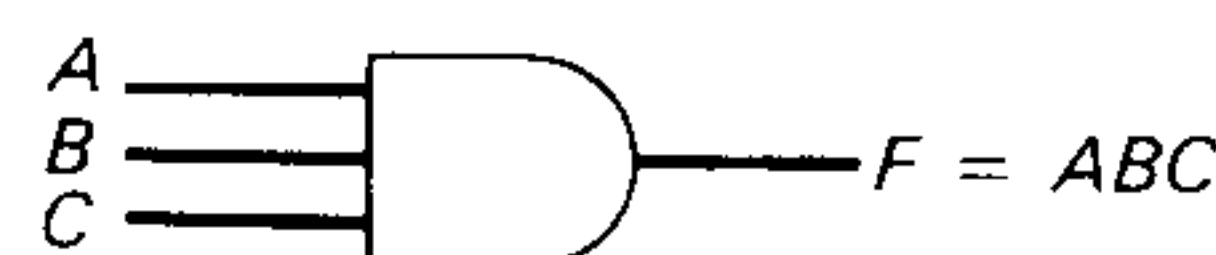
(b) Diagrama de sincronización

FIGURA 2-2

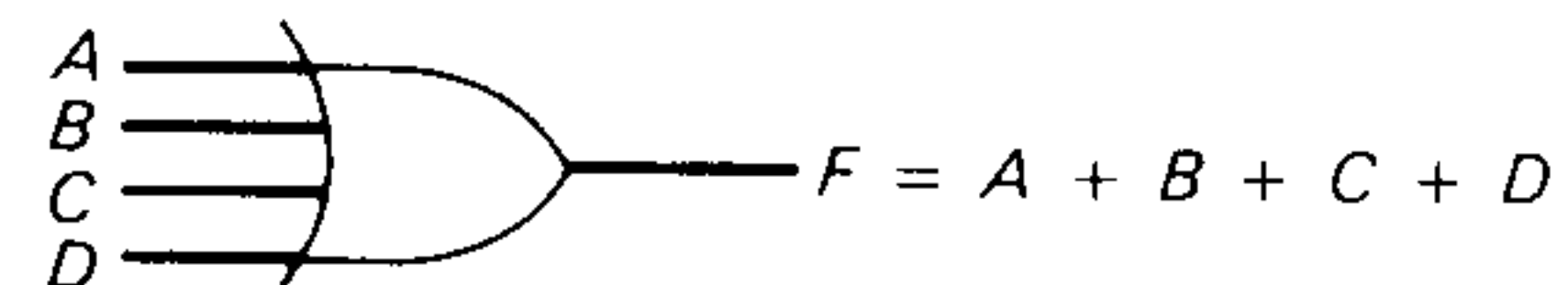
Compuertas de lógica digital

el intervalo especificado. La región intermedia entre las regiones admitidas se cruza sólo durante la transición de estados. Cualquier información deseada de cómputo o control puede sencillamente operarse transmitiendo señales binarias a través de diversas combinaciones de compuertas lógicas, y cada señal representa una variable binaria en particular.

Los símbolos gráficos que se utilizan para designar los tres tipos de compuertas se ilustran en la figura 2-2(a). Las compuertas son bloques de *hardware* que producen el equivalente de señales de salida, 1 y 0 lógicos, si se satisfacen requisitos de lógica de entrada. Las señales de entrada X y Y pueden existir en las compuertas AND y OR en uno de cuatro estados posibles: 00, 01, 10 u 11. Estas señales de entrada se ilustran en la figura 2-2(b) junto con la señal de salida correspondiente para cada compuerta. Los diagramas de sincronización ilustran la respuesta de cada compuerta a las cuatro combinaciones de señales de entrada posibles. El eje horizontal del diagrama de sincronización representa el tiempo y el eje vertical muestra la señal conforme ésta cambia entre los dos niveles de voltaje posibles. El nivel bajo representa el 0 lógico y el nivel alto representa el 1 lógico. La compuerta AND responde



(a) Compuerta AND de tres entradas



(b) Compuerta OR de cuatro entradas

FIGURA 2-3

Compuertas con entradas múltiples

con una señal de salida 1 lógico cuando ambas señales de entrada son 1 lógicos. La compuerta OR responde con una señal de salida 1 lógico si alguna señal de entrada es 1 lógico. La compuerta NOT se conoce comúnmente como *inversor*. La razón de ser de este nombre es obvia por la respuesta de señal en el diagrama de sincronización donde se muestra que la señal de salida invierte el sentido lógico de la señal de entrada.

Las compuertas AND y OR pueden tener más de dos entradas. En la figura 2-3 se ilustran una compuerta AND con tres entradas y una compuerta OR con cuatro entradas. La compuerta AND de tres entradas responde con una salida de 1 lógico si las tres entradas son también 1 lógicos. La salida produce un 0 lógico si alguna entrada es 0 lógico. La compuerta OR de cuatro entradas responde con un 1 lógico si alguna entrada es 1 lógico; su salida se convierte en 0 lógico sólo cuando todas las entradas son 0 lógico.

2-2 ALGEBRA BOOLEANA

El álgebra booleana es la que tiene que ver con variables binarias y operaciones lógicas. Las variables se designan por letras del alfabeto, y las tres operaciones lógicas básicas son AND, OR y el complemento. Una función booleana consta de una expresión algebraica formada con variables binarias, las constantes 0 y 1, los símbolos de las operaciones lógicas, paréntesis y un signo igual. Para un valor dado de las variables binarias, la función booleana puede ser igual a 1 o 0. Considérese como ejemplo la siguiente función booleana:

$$F = X + \bar{Y}Z$$

La función F es igual a 1 si X es igual a 1, o si Y y Z son iguales a 1. De lo contrario F es igual a 0. La operación de complemento prescribe que cuando $\bar{Y} = 1$ entonces $Y = 0$. Por lo tanto, podemos decir que $F = 1$ si $X = 1$, o si $Y = 0$ y $Z = 1$. Una función booleana expresa la relación lógica entre variables binarias. Se evalúa determinando el valor binario de la expresión de todos los valores posibles de las variables.

Una función booleana se puede representar en una tabla de verdad, la cual es una lista de combinaciones de unos y ceros asignados a las variables binarias y una columna que muestra el valor de la función para cada combinación binaria. El número de renglones de la tabla de verdad es 2^n , donde n es el número de variables de la función. Las combinaciones binarias de la tabla de verdad se obtienen a partir de los números binarios contando de 0 a $2^n - 1$. La tabla 2-2 muestra la tabla de verdad de las funciones antes citadas. Existen ocho posibles combinaciones binarias para asignar bits a las variables X , Y y Z . La columna con la letra F contiene un 0 o 1 para cada una de estas combinaciones. La tabla muestra que la función es igual a 1 cuando $X = 1$ o cuando $YZ = 01$. En caso contrario es igual a 0.

Una función booleana puede transformarse de una expresión algebraica en un diagrama de circuitos compuesto de compuertas lógicas. El diagrama de circuitos lógicos de F se muestra en la figura 2-4. Existe un inversor para la entrada Y que genera el complemento \bar{Y} . Existe una compuerta AND para el término $\bar{Y}Z$ y una compuerta OR que combina los dos términos. En los diagramas de circuitos lógicos, las variables de la función se toman como las entradas del circuito y la variable binaria F se toma como la salida del circuito.

Identi

TABLA 2—2
Tabla de verdad de la
función $F = X + \bar{Y}X$

X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

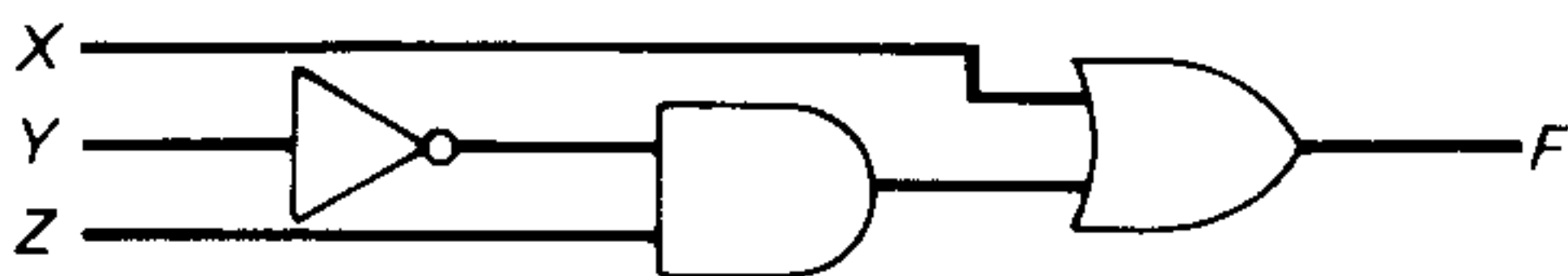


FIGURA 2—4
Diagrama de circuito lógico para $F = X + YZ$

Existe una sola forma en la que se puede representar una función booleana en una tabla de verdad. Sin embargo, cuando la función está en forma algebraica, puede expresarse de diversas maneras. La expresión particular que se usa para designar la función, también determinará la interconexión de compuertas en el diagrama de circuitos lógicos. Mediante la manipulación de una expresión booleana según las reglas del álgebra booleana, a veces es posible obtener una expresión más simple para la misma función y en consecuencia para reducir el número de compuertas en el circuito. Para ver cómo se hace esto, es necesario estudiar antes las reglas básicas del álgebra.

Identidades básicas del álgebra booleana

La tabla 2-3 presenta una lista de las identidades más básicas del álgebra booleana. La notación se simplifica omitiendo el símbolo \cdot para la operación AND siempre que esto no cause confusión. Las nueve primeras identidades muestran la relación entre una variable X , su complemento \bar{X} y las constantes binarias 0 y 1. Las cinco identidades siguientes, de la 10 a la 14, son similares al álgebra ordinaria. Las tres últimas, de la 15 a la 17, no se aplican en el álgebra ordinaria pero son muy útiles en la manipulación de expresiones booleanas.

Las reglas básicas que se muestran en la tabla 2-3 se han ordenado en dos columnas. Las dos partes demuestran la propiedad de dualidad del álgebra booleana. El *dual* de una expresión algebraica se obtiene intercambiando operaciones OR y AND y reemplazando unos por ceros y ceros por unos. Una ecuación en una columna de la tabla se puede obtener a partir de la ecuación correspondiente de la otra columna calculando el dual de las expresiones a ambos lados del signo igual. Por ejemplo, la relación 2 es el dual de la 1 porque la operación OR se ha reemplazado por una AND y el 0 por 1.

Las nueve identidades en que interviene una sola variable pueden verificarse fácilmente sustituyendo X por sus dos valores posibles. Por ejemplo, para demostrar que $X + 0 = X$, sea $X = 0$ para obtener $0 + 0 = 0$; después, sea $X = 1$ para obtener $1 + 1 = 1$. Ambas ecuaciones se cumplen de acuerdo con la definición de la operación lógica OR. Se puede sustituir con cualquier expresión, a la variable X , en todas las ecuaciones booleanas de la tabla. Por lo tanto, con la identidad 3, y con $X = AB + C$, se obtiene

$$AB + C + 1 = 1$$

Nótese que la ecuación 9 señala que la complementación doble devuelve a la variable su valor original. En consecuencia, si $X = 0$ entonces $\overline{\overline{X}} = 1$ y $\overline{\overline{\overline{X}}} = 0 = X$.

Las leyes conmutativas indican que el orden en el cual se escriben las variables no afectará el resultado cuando se utilicen las operaciones OR y AND. Las leyes aso-

TABLA 2—3
Identidades básicas del álgebra booleana

1. $X + 0 = X$	2. $X \cdot 1 = X$	
3. $X + 1 = 1$	4. $X \cdot 0 = 0$	
5. $X + X = X$	6. $X \cdot X = X$	
7. $X + \overline{X} = 1$	8. $X \cdot \overline{X} = 0$	
9. $\overline{\overline{X}} = X$		
10. $X + Y = Y + X$	11. $XY = YX$	Conmutativa Asociativa Distributiva DeMorgan
12. $X + (Y + Z) = (X + Y) + Z$	13. $X(YZ) = (XY)Z$	
14. $X(Y + Z) = XY + XZ$	15. $X + YZ = (X + Y)(X + Z)$	
16. $\overline{X + Y} = \overline{X} \cdot \overline{Y}$	17. $\overline{X \cdot Y} = \overline{X} + \overline{Y}$	

ciativas postulan que el resultado de formar una operación entre tres variables es independiente del orden que se siga y, por lo tanto, pueden eliminarse sin excepción todos los paréntesis.

$$X + (Y + Z) = (X + Y) + Z = X + Y + Z$$

$$X(YZ) = (XY)Z = XYZ$$

Estas dos leyes y la primera ley distributiva del álgebra ordinaria son bien conocidas y por consiguiente no deben presentar dificultad alguna. La segunda ley distributiva dada por la identidad 15 es el dual de la ley distributiva ordinaria y es muy útil en la manipulación de funciones booleanas.

$$X + YZ = (X + Y)(X + Z)$$

Esta ecuación se puede usar para otra combinación de variables. Considérese la expresión $(A + B)(A + CD)$. Haciendo que $X = A$, $Y = B$ y $Z = CD$, y aplicando la segunda ley distributiva, se obtiene

$$(A + B)(A + CD) = A + BCD$$

Las dos últimas identidades de la tabla 2-3 se conocen como el teorema de DeMorgan.

$$\overline{(X + Y)} = \overline{X} \cdot \overline{Y} \quad \text{y} \quad \overline{(X \cdot Y)} = \overline{X} + \overline{Y}$$

Este es un teorema muy importante y se aplica para obtener el complemento de una expresión. El teorema de DeMorgan se puede verificar por medio de tablas de verdad que asignan todos los valores binarios posibles a X y Y . La tabla 2-4 presenta dos tablas de verdad que verifican la primera parte de este teorema. En A, se evalúa $(\overline{X + Y})$ para todos los valores posibles de X y Y . Esto se hace evaluando en primer término $X + Y$ y después calculando su complemento. En B, se evalúan \overline{X} y \overline{Y} y después se suman (con la operación AND). El resultado es el mismo para las cuatro combinaciones binarias de X y Y , el cual verifica la identidad de la ecuación.

Nótese el orden en el cual se efectúan las operaciones cuando se evalúa una expresión. Se evalúa el complemento con respecto a una variable, luego la operación AND y después la operación OR, como se hace en el álgebra ordinaria con la multiplicación y la adición. Un complemento sobre una expresión como $(\overline{X + Y})$ se considera como especificar NOT ($X + Y$); así, primero se evalúa el valor que está dentro de los paréntesis y después se toma el resultado del complemento. Se acostumbra excluir los paréntesis cuando se complementa una expresión con una barra sobre toda la expresión. Por lo tanto $(\overline{X + Y})$ se expresa como $\overline{X + Y}$ cuando se designa el complemento de $(X + Y)$.

TABLA 2—4
Tablas de verdad para verificar el teorema de DeMorgan

A.	X	Y	X + Y	($\overline{X + Y}$)	B.	X	Y	\overline{X}	\overline{Y}	$\overline{X} \cdot \overline{Y}$
	0	0	0	1		0	0	1	1	1
	0	1	1	0		0	1	1	0	0
	1	0	1	0		1	0	0	1	0
	1	1	1	0		1	1	0	0	0

El teorema de DeMorgan se puede extender a tres o más variables. El teorema de DeMorgan general puede expresarse como sigue:

$$\overline{X_1 + X_2 + X_3 + \dots + X_n} = \overline{X_1} \overline{X_2} \overline{X_3} \dots \overline{X_n}$$
$$\overline{X_1 X_2 X_3 \dots X_n} = \overline{X_1} + \overline{X_2} + \overline{X_3} + \dots + \overline{X_n}$$

La operación lógica cambia de OR a AND o de AND a OR. Además, se elimina el complemento de toda la expresión y se coloca sobre cada variable. Por ejemplo,

$$\overline{XYZ} = \overline{X} + \overline{Y} + \overline{Z} \quad \text{y} \quad \overline{A + B + C + D} = \overline{A} \overline{B} \overline{C} \overline{D}.$$

Manipulación algebraica

El álgebra booleana es una herramienta útil para simplificar circuitos digitales. Considérese por ejemplo la siguiente función booleana:

$$F = \overline{X}YZ + \overline{X}Y\overline{Z} + XZ$$

La ejecución de esta función con compuertas lógicas se representa en la figura 2-5(a). Las variables de entrada X y Z se complementan con inversores para obtener \overline{X} y \overline{Z} . Los tres términos de la expresión se ejecutan con tres compuertas AND. La compuerta OR forma la suma lógica de los tres términos. Ahora considérese la posible simplificación de la función a través de la aplicación de las identidades citadas en la tabla 2-3.

$$\begin{aligned} F &= \overline{X}YZ + \overline{X}Y\overline{Z} + XZ \\ &= \overline{X}Y(Z + \overline{Z}) + XZ \\ &= \overline{X}Y \cdot 1 + XZ \\ &= \overline{X}Y + XZ \end{aligned}$$

por la identidad 14
por la identidad 7
por la identidad 2

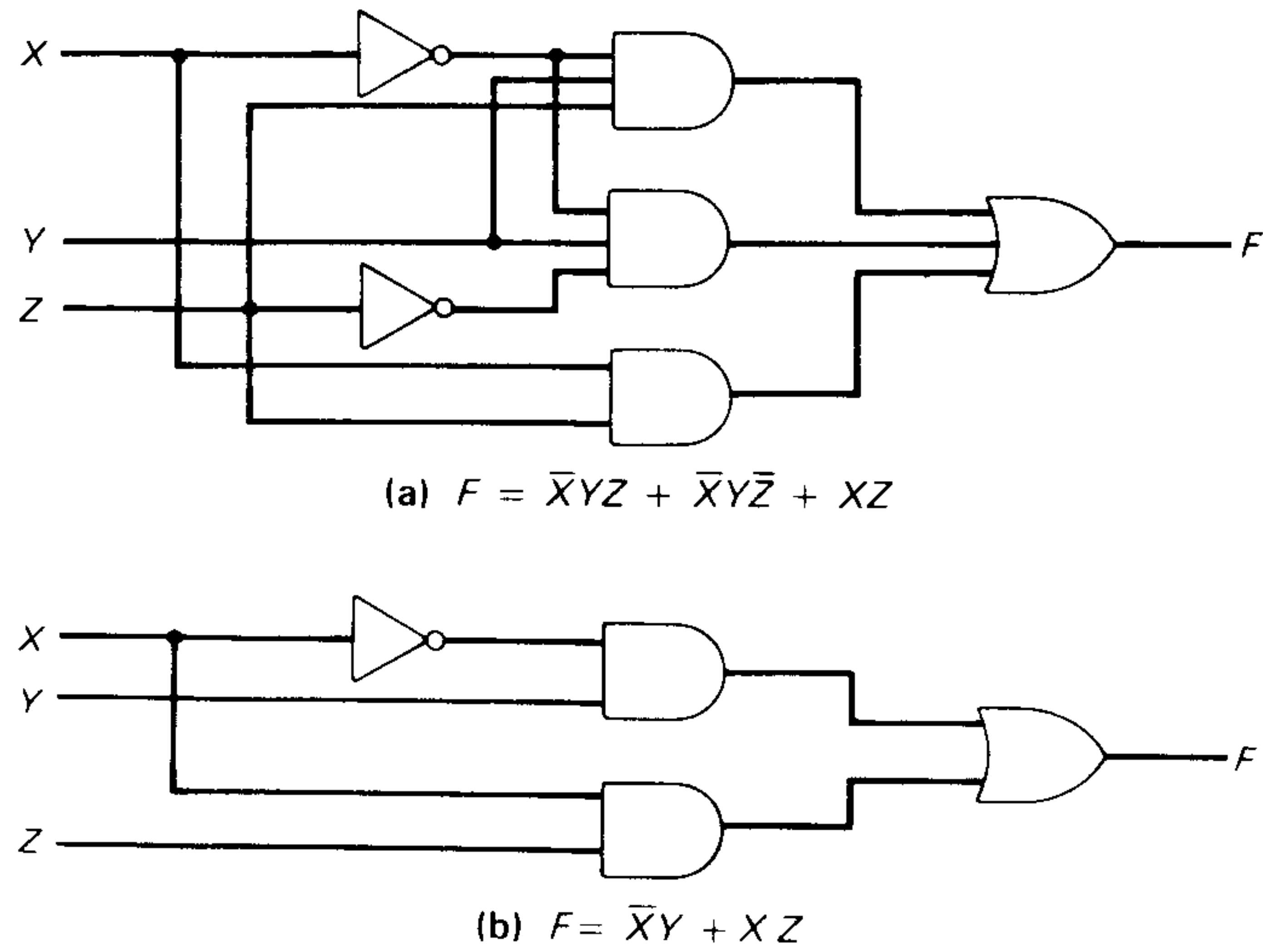


FIGURA 2—5
Ejecución de una función booleana con compuertas

La función se reduce a sólo dos términos y se puede ejecutar con compuertas como se indica en la figura 2-5(b). Es evidente que el circuito de la parte (b) es más simple que el de la parte (a); no obstante ambos efectúan la misma función. Es posible emplear una tabla de verdad para verificar que las dos expresiones sean equivalentes, como se ilustra en la tabla 2-5. La función, como se expresa en la figura 2-5(a), es igual a 1 cuando $XYZ = 011$, o bien cuando $XYZ = 010$ o incluso cuando $XZ = 11$. Esto produce cuatro unos para F en la tabla. La función según se expresa en la figura 2-5(b) es igual a 1 cuando $XY = 01$, o bien cuando $XZ = 11$. Esto produce los cuatro unos de la tabla. Como ambas expresiones generan la misma tabla de verdad, se

TABLA 2—5
Tabla de verdad de una
función booleana

<i>X</i>	<i>Y</i>	<i>Z</i>	<i>F</i>
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

ad 14
ad 7
ad 2

dice que son equivalentes. Por lo tanto, los dos circuitos tienen la misma salida para todas las combinaciones binarias de entrada posibles de las tres variables. Cada uno efectúa la misma función; pero es preferible el que tiene menos compuertas porque requiere un menor número de componentes.

Cuando una expresión booleana se ejecuta con compuertas lógicas, cada término requiere una compuerta y cada variable del término designa una entrada a la compuerta. Definimos una *literal* como una variable única contenida en un término que puede o no estar complementada. La función de la figura 2-5(a) tiene tres términos y ocho literales; la de la figura 2-5(b) tiene dos términos y cuatro literales. Reduciendo el número de términos, el número de literales o ambos en una expresión booleana, a veces es posible obtener un circuito más sencillo. La manipulación del álgebra booleana consiste principalmente en reducir una expresión con el fin de obtener un circuito más simple. Por desgracia, no hay reglas específicas que garanticen un resultado óptimo. El único método de que se dispone es un procedimiento rutinario que emplea las relaciones básicas y otras manipulaciones que se vuelven familiares con el uso. Los ejemplos que siguen ilustran algunas de las posibilidades.

- a. $X + XY = X(1 + Y) = X$
- b. $XY + X\bar{Y} = X(Y + \bar{Y}) = X$
- c. $X + \bar{X}Y = (X + \bar{X})(X + Y) = X + Y$

Obsérvese que el paso intermedio $X = X \cdot 1$ se ha omitido cuando X se factoriza en la primera ecuación. La relación $1 + Y = 1$ es útil para eliminar términos redundantes, como se hace con el término XY de la primera ecuación. La relación $Y + \bar{Y} = 1$ es útil para combinar dos términos, como se hace en la segunda ecuación. Los dos términos que se combinarán deben contener la misma variable, pero ésta debe complementarse en un término y no complementarse en el otro. La tercera ecuación se simplifica por medio de la segunda ley de distribución (identidad 15 de la tabla 2-3).

Los siguientes son tres ejemplos más de la simplificación de expresiones booleanas.

- d. $X(X + Y) = X + XY = X(1 + Y) = X$
- e. $(X + Y)(X + \bar{Y}) = X + Y\bar{Y} = X$
- f. $X(\bar{X} + Y) = X\bar{X} + XY = XY$

Nótese que los pasos intermedios $XX = X = X \cdot 1$ se han pasado por alto durante la manipulación de la ecuación d. La expresión en e se simplifica por medio de la segunda ley distributiva. Aquí se vuelve a omitir el paso intermedio $Y\bar{Y} = 0$ y $X + 0 = X$.

Las tres últimas ecuaciones son el dual de las tres primeras. Recuérdese que el dual de una expresión se obtiene cambiando AND por OR y OR por AND en todo el proceso (y los unos por ceros y los ceros por unos, si figuran en la expresión). El principio de dualidad del álgebra booleana señala que una ecuación booleana se mantiene válida si se toma el dual de la expresión a ambos lados del signo de igual. Por lo tanto, las ecuaciones d, e y f se pueden verificar calculando el dual de las ecuaciones a, b y c, respectivamente.

El teorema de consenso, que se representa a continuación, a veces es útil cuando se simplifican expresiones booleanas.

$$XY + \bar{X}Z + YZ = XY + \bar{X}Z$$

mo se
que el
mplear
como
ual a 1
. Esto
figura
s cua-
ad, se

Esto demuestra que el tercer término YZ es redundante y se puede eliminar. Nótese que Y y Z están asociadas también con X y \bar{X} en los dos primeros términos. La prueba de la ecuación se obtiene sumando (con AND) primero YZ con $(X + \bar{X}) = 1$.

$$\begin{aligned} XY + \bar{X}Z + YZ &= XY + \bar{X}Z + YZ(X + \bar{X}) \\ &= XY + \bar{X}Z + XYZ + \bar{X}YZ \\ &= XY + XYZ + \bar{X}Z + \bar{X}YZ \\ &= XY(1 + Z) + \bar{X}Z(1 + Y) \\ &= XY + \bar{X}Z \end{aligned}$$

El dual del teorema de consenso es

$$(X + Y)(\bar{X} + Z)(Y + Z) = (X + Y)(\bar{X} + Z)$$

El ejemplo que sigue demuestra cómo se puede aplicar el teorema de consenso cuando se manipula una expresión booleana.

$$\begin{aligned} (A + B)(\bar{A} + C) &= A\bar{A} + AC + \bar{A}B + BC \\ &= AC + \bar{A}B + BC \\ &= AC + \bar{A}B \end{aligned}$$

Nótese que $A\bar{A} = 0$ y $0 + AC = AC$. Aquí, el término redundante es BC .

Complemento de una función

El complemento de una función, F , se obtiene a partir de un intercambio de unos por ceros y de ceros por unos en los valores de F de la tabla de verdad. El complemento de una función puede determinarse en forma algebraica aplicando el teorema de DeMorgan. Recuérdese que la forma generalizada de este teorema señala que el complemento de una expresión se obtiene intercambiando operaciones AND y OR y complementando cada variable.

Ejemplo 2-1

Determinése el complemento de las dos funciones que siguen: $F_1 = \bar{X}Y\bar{Z} + \bar{X}\bar{Y}Z$ y $F_2 = X(\bar{Y}\bar{Z} + YZ)$.

Aplicando el teorema de DeMorgan tantas veces como sea necesario, los complementos se obtienen de la manera siguiente:

$$\begin{aligned} \bar{F}_1 &= \overline{\bar{X}Y\bar{Z} + \bar{X}\bar{Y}Z} = (\overline{\bar{X}Y\bar{Z}}) \cdot (\overline{\bar{X}\bar{Y}Z}) \\ &= (X + \bar{Y} + Z)(X + Y + \bar{Z}) \end{aligned}$$

$$\begin{aligned} \bar{F}_2 &= \overline{X(\bar{Y}\bar{Z} + YZ)} = \bar{X} + \overline{(\bar{Y}\bar{Z} + YZ)} \\ &= \bar{X} + (\overline{\bar{Y}\bar{Z}} \cdot \overline{YZ}) \\ &= \bar{X} + (Y + Z)(\bar{Y} + \bar{Z}) \end{aligned}$$

Un método simple para determinar el complemento de una función consiste en calcular el dual de la función y complementar cada literal. Este método sigue del teorema de DeMorgan generalizado. Recuérdese que el dual de una expresión se obtiene intercambiando las operaciones AND y OR y los unos y ceros.

Ejemplo 2-2

Obtégase el complemento de las funciones del ejemplo 2-1 calculando su dual y complementando cada literal

$$F_1 = \overline{X}Y\overline{Z} + \overline{X}\overline{Y}Z$$

El dual de F_1

$(\overline{X} + Y + \overline{Z})(\overline{X} + \overline{Y} + Z)$

Complementése cada literal

$(X + \overline{Y} + Z)(X + Y + \overline{Z}) = \overline{F}_1$

$$F_2 = X(\overline{Y}\overline{Z} + YZ)$$

El dual de F_2

$X + (\overline{Y} + \overline{Z})(Y + Z)$

Complementése cada literal

$\overline{X} + (Y + Z)(\overline{Y} + \overline{Z}) = \overline{F}_2$

2-3

FORMAS ESTANDAR

Una función booleana se puede escribir en diversas formas cuando se expresa en forma algebraica. Sin embargo, hay pocas expresiones algebraicas que se consideran en forma estándar. Las formas estándar facilitan los procedimientos de simplificación de expresiones booleanas y con frecuencia producen circuitos de compuertas más deseables.

Las formas estándar contienen términos que se conocen como términos de *productos* y términos de *sumas*. Un ejemplo de un término de producto es $X\overline{Y}Z$. Este es un producto lógico que consta de una operación AND entre varias variables. Un ejemplo de un término de suma es $\overline{X} + Y + Z$. Esta es una suma lógica que consta de una operación OR entre las variables. Debe entenderse que los términos *producto* y *suma* no implican operaciones aritméticas cuando se maneja el álgebra booleana. En su lugar, especifican operaciones *lógicas* equivalentes a las operaciones booleanas de AND y OR, respectivamente.

Minitérminos (términos mínimos) y maxitérminos (términos máximos)

Se ha probado que una tabla de verdad define una función booleana. Una expresión algebraica que representa la función se determina a partir de la tabla obteniendo la suma lógica de todos los términos de productos para los cuales la función asume el valor binario de 1. Un término de producto en el cual todas las variables figuran exactamente una vez complementadas o no complementadas recibe el nombre de *minitérmino* (o término mínimo). Su propiedad característica es que presenta exactamente una combinación de las variables binarias en una tabla de verdad. Existen 2^n minitérminos distintos para n variables. Los cuatro minitérminos de las variables X

TABLA 2—6
Minitérminos y maxitérminos para tres variables

			Minitérminos		Maxitérminos	
X	Y	Z	Término de producto	Símbolo	Término de suma	Símbolo
0	0	0	$\overline{X}\overline{Y}\overline{Z}$	m_0	$X + Y + Z$	M_0
0	0	1	$\overline{X}\overline{Y}Z$	m_1	$X + Y + \overline{Z}$	M_1
0	1	0	$\overline{X}Y\overline{Z}$	m_2	$X + \overline{Y} + Z$	M_2
0	1	1	$\overline{X}YZ$	m_3	$X + \overline{Y} + \overline{Z}$	M_3
1	0	0	$X\overline{Y}\overline{Z}$	m_4	$\overline{X} + Y + Z$	M_4
1	0	1	$X\overline{Y}Z$	m_5	$\overline{X} + Y + \overline{Z}$	M_5
1	1	0	$XY\overline{Z}$	m_6	$\overline{X} + \overline{Y} + Z$	M_6
1	1	1	XYZ	m_7	$\overline{X} + \overline{Y} + \overline{Z}$	M_7

y Y son $\overline{X}\overline{Y}$, $\overline{X}Y$, $X\overline{Y}$ y XY . Los ocho minitérminos de las tres variables X , Y y Z se citan en la tabla 2-6. Los números binarios del 000 al 111 se presentan debajo de las variables. Cada minitérmino se obtiene a partir del término de producto de exactamente tres variables, donde cada una de ellas se complementa si el bit correspondiente del número binario es 0 y no se complementa si es 1. En la tabla se muestra también un símbolo para cada minitérmino y es de la forma m_j , donde el subíndice j denota el equivalente decimal del número binario del minitérmino. La lista del minitérminos para n variables dadas cualesquiera se puede formar de manera análoga a partir de una lista de los números binarios del 0 al $2^n - 1$.

Un término de una suma que contiene todas las variables en forma complementada o no complementada recibe el nombre de *maxitérmino* (o término máximo). Una vez más, es posible formular 2^n maxitérminos con n variables. Los ocho maxitérminos de tres variables se presentan en la tabla 2-6. Cada maxitérmino se obtiene a partir de la suma lógica de las tres variables, donde cada variable está complementada si el bit correspondiente es 1 y no complementada si es 0. El símbolo de un maxitérmino es M_j , donde j denota el número binario del maxitérmino. Nótese que un minitérmino y un maxitérmino con el mismo número de subíndice son los complementos de sí mismos, es decir, $M_j = \overline{m_j}$. Por ejemplo, para $j = 3$, se tiene

$$\overline{m_3} = \overline{\overline{X}\overline{Y}Z} = X + \overline{Y} + \overline{Z} = M_3$$

Una función booleana se puede expresar algebraicamente a partir de una tabla de verdad dada formando la suma lógica de todos los minitérminos que producen un 1 en la función. Considérese la función booleana F de la tabla 2-7(a). La función es igual a 1 para cada una de las siguientes combinaciones binarias de las variables X , Y y Z : 000, 010, 101 y 111. Estas combinaciones corresponden a los minitérminos 0, 2, 5 y 7. La función F se puede expresar de manera algebraica como la suma lógica de estos cuatro minitérminos.

$$F = \overline{X}\overline{Y}\overline{Z} + \overline{X}Y\overline{Z} + X\overline{Y}Z + XYZ = m_0 + m_2 + m_5 + m_7$$

Esta expresión puede abreviarse todavía más escribiendo sólo los subíndices decimales de los minitérminos.

$$F(X, Y, Z) = \Sigma m(0, 2, 5, 7)$$

TABLA 2—7
Funciones booleanas de tres variables

(a)	X	Y	Z	F	\bar{F}	(b)	X	Y	Z	E
	0	0	0	1	0		0	0	0	1
	0	0	1	0	1		0	0	1	1
	0	1	0	1	0		0	1	0	1
	0	1	1	0	1		0	1	1	0
	1	0	0	0	1		1	0	0	1
	1	0	1	1	0		1	0	1	1
	1	1	0	0	1		1	1	0	0
	1	1	1	1	0		1	1	1	0

El símbolo Σ representa la suma lógica (OR booleana) de los minitérminos. Los números que le siguen son los minitérminos de la función. Las letras entre paréntesis que siguen de F forman una lista de las variables en el orden que se considera cuando los minitérminos se convierten en términos de productos.

Ahora considérese el complemento de una función booleana. Los valores binarios de \bar{F} en la tabla 2-7(a) se obtienen cambiando unos por ceros y ceros por unos en los valores de F . Calculando la suma lógica de los minitérminos de \bar{F} se obtiene

$$\bar{F} = \bar{X}\bar{Y}Z + \bar{X}YZ + X\bar{Y}\bar{Z} + XY\bar{Z} = m_1 + m_3 + m_4 + m_6$$

o, en forma abreviada,

$$\bar{F}(X, Y, Z) = \Sigma m(1, 3, 4, 6)$$

Obsérvese que los números de los minitérminos de \bar{F} son los que faltan en la lista de los minitérminos de F . Ahora se toma el complemento de \bar{F} a fin de obtener F .

$$\begin{aligned} F &= \overline{m_1 + m_3 + m_4 + m_6} = \bar{m}_1 \cdot \bar{m}_3 \cdot \bar{m}_4 \cdot \bar{m}_6 \\ &= M_1 \cdot M_3 \cdot M_4 \cdot M_6 \quad (\text{ya que } \bar{m}_j = M_j) \\ &= (X + Y + \bar{Z})(X + \bar{Y} + \bar{Z})(\bar{X} + Y + Z)(\bar{X} + \bar{Y} + Z) \end{aligned}$$

Esto muestra el procedimiento para expresar una función booleana en producto de maxitérminos. La forma abreviada del producto de maxitérminos es

$$F(X, Y, Z) = \Pi M(1, 3, 4, 6)$$

El símbolo Π denota el producto lógico (AND booleana) de los maxitérminos entre paréntesis. Obsérvese que los números decimales incluidos en el producto de maxitérminos serán siempre los mismos que la lista de minitérminos de la función complementada (1, 3, 4, 6 en el ejemplo anterior). Los maxitérminos rara vez se utilizan cuando se manejan funciones booleanas, ya que siempre se pueden reemplazar con la lista de minitérminos de \bar{F} . Lo que sigue es un resumen de las propiedades más importantes de los minitérminos:

1. Hay 2^n minitérminos para n variables booleanas. Estos se pueden evaluar a partir de los números binarios de 0 a $2^n - 1$.
2. Cualquier función booleana se puede expresar como una suma lógica de minitérminos.

3. El complemento de una función contiene los minitérminos no incluidos en la función original.
4. Una función que incluyera los 2^n minitérminos es igual al 1 lógico.

Una función que no esté en forma de suma de minitérminos se puede convertir en la suma de minitérminos por medio de una tabla de verdad, ya que esta tabla siempre especifica los minitérminos de la función. Considérese por ejemplo la función booleana

$$E = \bar{Y} + \bar{X}Z$$

La expresión no está en suma de minitérminos porque cada término no contiene las tres variables X , Y y Z . La tabla de verdad para esta función se representa en la tabla 2-7(b). De la tabla de verdad se obtienen los minitérminos de la función.

$$E(X, Y, Z) = \Sigma m(0, 1, 2, 4, 5)$$

Los minitérminos del complemento de E son

$$\bar{E}(X, Y, Z) = \Sigma m(3, 6, 7)$$

Nótese que el número total de minitérminos en E y \bar{E} es igual a ocho porque la función tiene tres variables y tres variables pueden producir un total de ocho términos mínimos. Con cuatro variables, habrá un total de 16 minitérminos y, en el caso de dos variables, habrá cuatro términos mínimos. Un ejemplo de una función que incluye todos los minitérminos es

$$G(X, Y) = \Sigma m(0, 1, 2, 3) = 1$$

Puesto que G es una función de dos variables y contiene los cuatro minitérminos, siempre será igual a 1 lógico.

Suma de productos

La forma de la suma de minitérminos es una expresión algebraica estándar que se obtiene directamente a partir de una tabla de verdad. La expresión que así se obtiene contiene el número máximo de términos de producto y el número máximo de literales en cada término. Esto se debe a que, por definición, cada término mínimo debe incluir todas las variables de la función complementada o no complementada. Cuando se obtiene la suma de minitérminos de la tabla de verdad, el paso siguiente es intentar simplificar la expresión para ver si es posible reducir el número de términos de productos y el número de literales en los términos. El resultado es una expresión simplificada en suma de productos. La suma de productos es una forma de expresión estándar alternativa que contiene términos de productos con una, dos o cualquier número de literales. Un ejemplo de una función booleana expresada en suma de productos es

$$F = \bar{Y} + \bar{X}Y\bar{Z} + XY$$

La expresión tiene tres términos de producto. El primer término tiene una literal, el segundo tres literales y el tercero, dos.

El diagrama lógico de una expresión de suma de productos consta de un grupo de compuertas AND seguido de una compuerta OR. Este patrón de configuración se

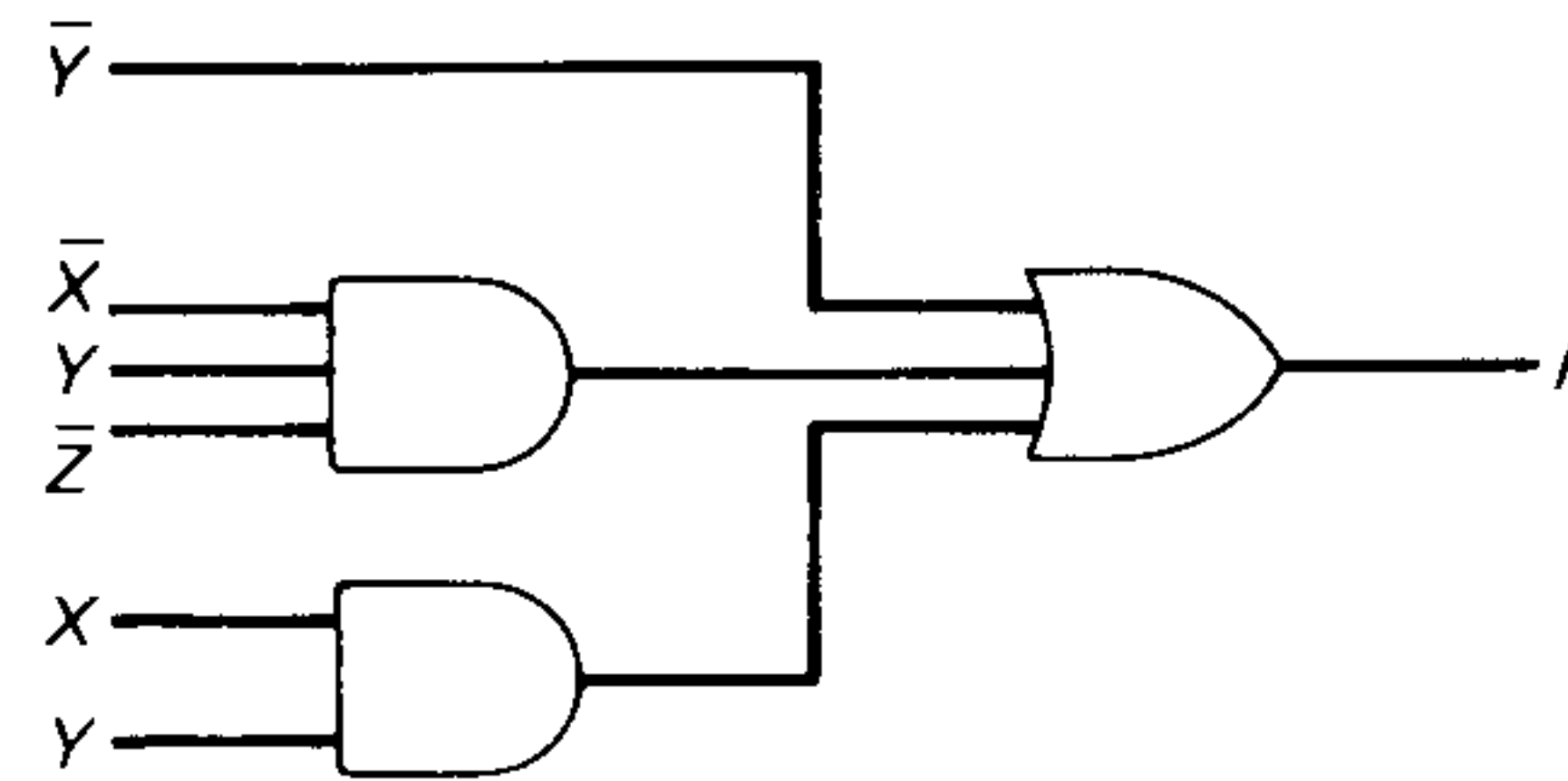


FIGURA 2—6

Ejecución de suma de productos

representa en la figura 2-6. Cada término de producto requiere una compuerta AND salvo para un término con una sola literal. La suma lógica se forma con una compuerta OR que recibe sus entradas de las salidas de las compuertas AND o de la variable única. Se supone que las variables de entrada se pueden obtener directamente en su complemento, de modo que el diagrama no tiene inversores. Las compuertas AND seguidas de una OR forman una configuración de circuitos que se conoce como ejecución *en dos niveles* o *binivel*.

Si una expresión no está en la forma de suma de productos, se puede convertir en la forma estándar por medio de las leyes distributivas. Considérese la expresión

$$F = AB + C(D + E)$$

Esta expresión no está en forma de suma de productos porque el término $(D + E)$ es parte de un producto pero no es una variable única. La expresión se puede convertir en una suma de productos eliminando los paréntesis.

$$F = AB + C(D + E) = AB + CD + CE$$

La ejecución de esta función se muestra en la figura 2-7. La función se ejecuta en forma no estándar en (a). Esto requiere dos compuertas AND y dos OR. Existen tres niveles de compuertas en este circuito. La expresión se ejecuta en forma de suma de productos en (b). Este circuito requiere tres compuertas AND y una OR y emplea dos niveles de compuertas. En general, es preferible una ejecución en dos niveles porque produce el menor tiempo de retraso a través de las compuertas cuando la señal se propaga de las entradas a la salida.

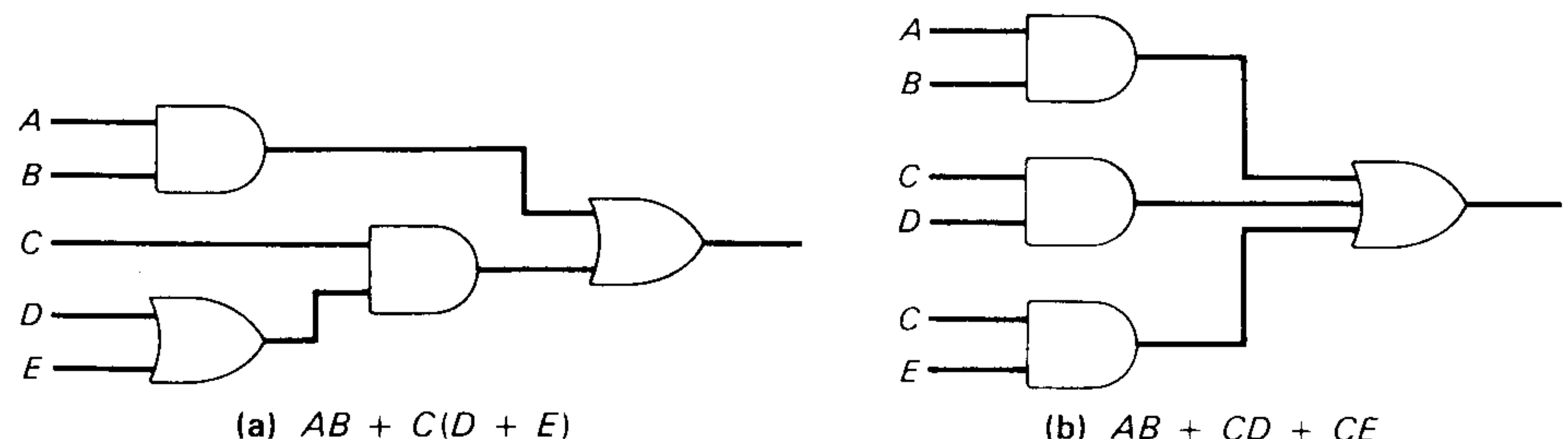


FIGURA 2—7

Ejecución en dos y tres niveles

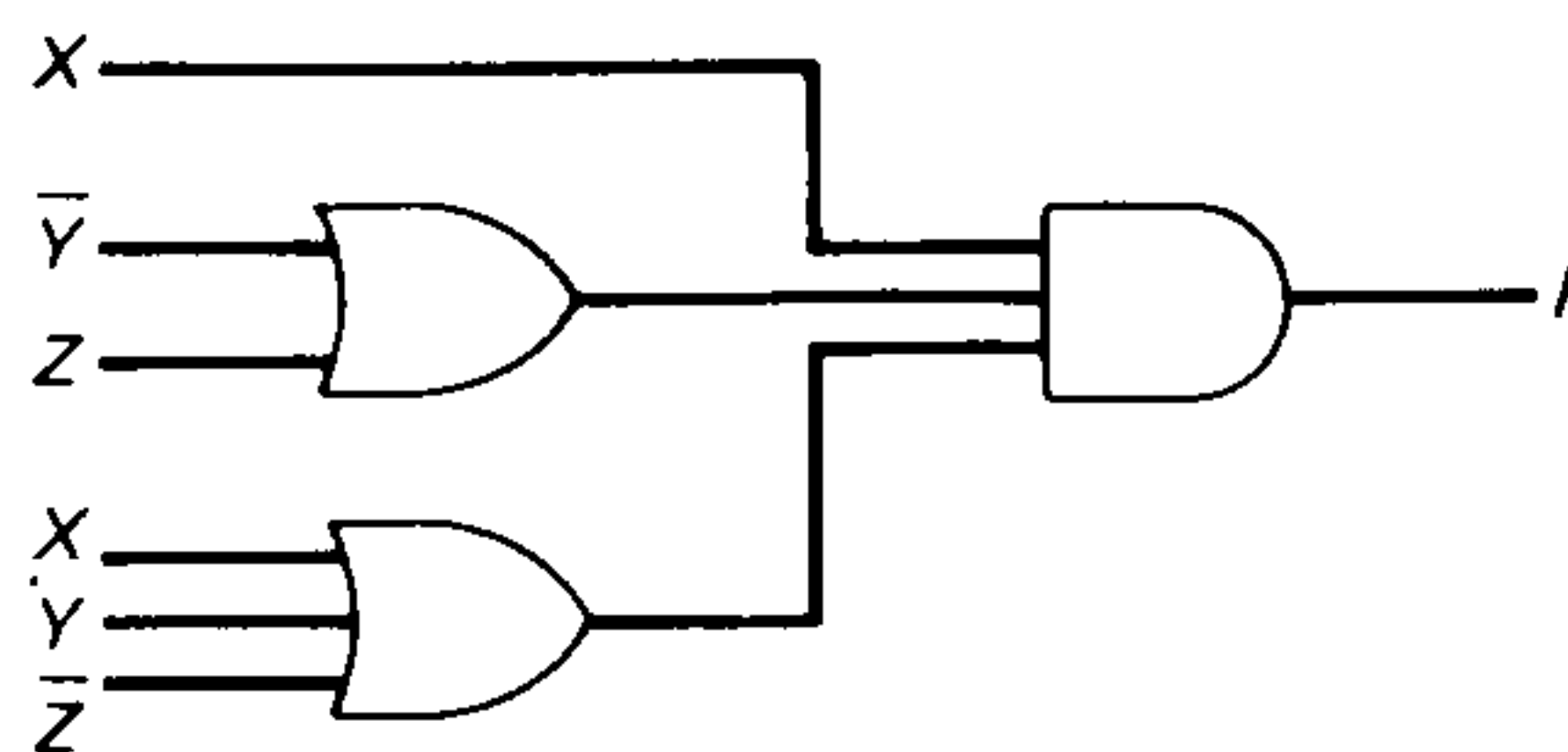


FIGURA 2—8

Ejecución de producto de sumas

Producto de sumas

Otra forma estándar de expresar funciones booleanas en forma algebraica es el producto de sumas, y se obtiene formando el producto lógico de términos de suma. Cada término de suma lógico puede tener un número cualquiera de literales. Un ejemplo de una función expresada en producto de sumas es

$$F = X(\bar{Y} + Z)(X + Y + \bar{Z})$$

Esta expresión tiene tres términos de suma de una, dos y tres literales. Los términos de suma realizan una operación OR y el producto es una operación AND.

La estructura de compuertas de la expresión del producto de sumas consta de un grupo de compuertas OR para los términos de suma (salvo por un término literal único) seguido de una compuerta AND. Esto se ilustra en la figura 2-8. El tipo de expresión estándar da origen a una estructura de compuertas en dos niveles.

2-4 SIMPLIFICACION DE UN MAPA

La complejidad de las compuertas lógicas digitales que ejecutan una función booleana está relacionada directamente con la expresión algebraica a partir de la cual se ejecuta la función. Aunque la representación de la tabla de verdad de una función es única cuando se expresa en forma algebraica, la función puede asumir muchas formas. Las expresiones booleanas se pueden simplificar por manipulación algebraica como se vio en la sección 2-2. No obstante, este procedimiento de simplificación es complicado porque carece de reglas específicas para anticipar cada paso sucesivo del proceso de manipulación y es difícil determinar si se ha obtenido la expresión más simple. El método del mapa ofrece un procedimiento directo para simplificar funciones booleanas de hasta cuatro variables. Se pueden trazar mapas para un número mayor de variables pero son más difíciles de usar. El mapa se conoce también como mapa de Karnaugh o mapa K.

El mapa es un diagrama formado por cuadrados, donde cada uno de ellos representa un minitérmino de la función. Como cualquier función booleana se puede expresar como una suma de minitérminos, se deduce que una función booleana se reconoce gráficamente en el mapa a partir del área encerrada por esos cuadrados cuyos términos mínimos están incluidos en la función. De hecho, el mapa presenta un diagrama visual de todas las formas posibles en que se puede expresar una función en forma estándar. Por el reconocimiento de diversos patrones, el usuario puede determinar otras expresiones algebraicas para la misma función, de las cuales se pueden seleccionar las más sencillas.

Las expresiones simplificadas que produce el mapa están siempre en una de las dos formas estándar: en suma de productos o bien en producto de sumas. Se supondrá que la expresión algebraica más simple es una con un número mínimo de términos con el menor número posible de literales en cada término. Esto produce un diagrama de circuitos lógicos con un número mínimo de compuertas y el número mínimo de entradas a las compuertas. Veremos más adelante que la expresión más simple no es necesariamente única. A veces es posible hallar dos o más expresiones que satisfagan los criterios de simplificación. En ese caso, una u otra solución sería satisfactoria. En esta sección sólo se cubre la simplificación de la suma de productos. En la sección que sigue se mostrará cómo obtener la simplificación del producto de sumas.

Mapa de dos variables

Hay cuatro minterminos para una función booleana con dos variables. Por lo tanto, el mapa de dos variables consta de cuatro cuadrados, uno para cada término mínimo, según se ve en la figura 2-9. El mapa se vuelve a trazar en la parte (b) para señalar la relación entre los cuadros y las variables X y Y . El 0 y 1 marcados del lado izquierdo y en la parte superior del mapa designan los valores de las variables. La variable X figura complementada en el renglón 0 y no complementada en el renglón 1. De la misma manera, Y aparece complementada en la columna 0 y no complementada en la columna 1.

Una función de dos variables se puede representar en un mapa marcando los cuadrados que corresponden a los minterminos de la función. Para poner un ejemplo, la función XY se muestra en la figura 2-10(a). Como XY es igual al mintermino m_3 , se coloca un 1 dentro del cuadrado que pertenece a m_3 . La figura 2-10(b) muestra el mapa de la suma lógica de tres minterminos.

La expresión simplificada $X + Y$ se determina a partir del área de dos cuadrados de la variable X en el segundo renglón y el área de dos cuadrados de Y en la segunda co-

$$m_1 + m_2 + m_3 = \bar{X}Y + X\bar{Y} + XY = X + Y$$

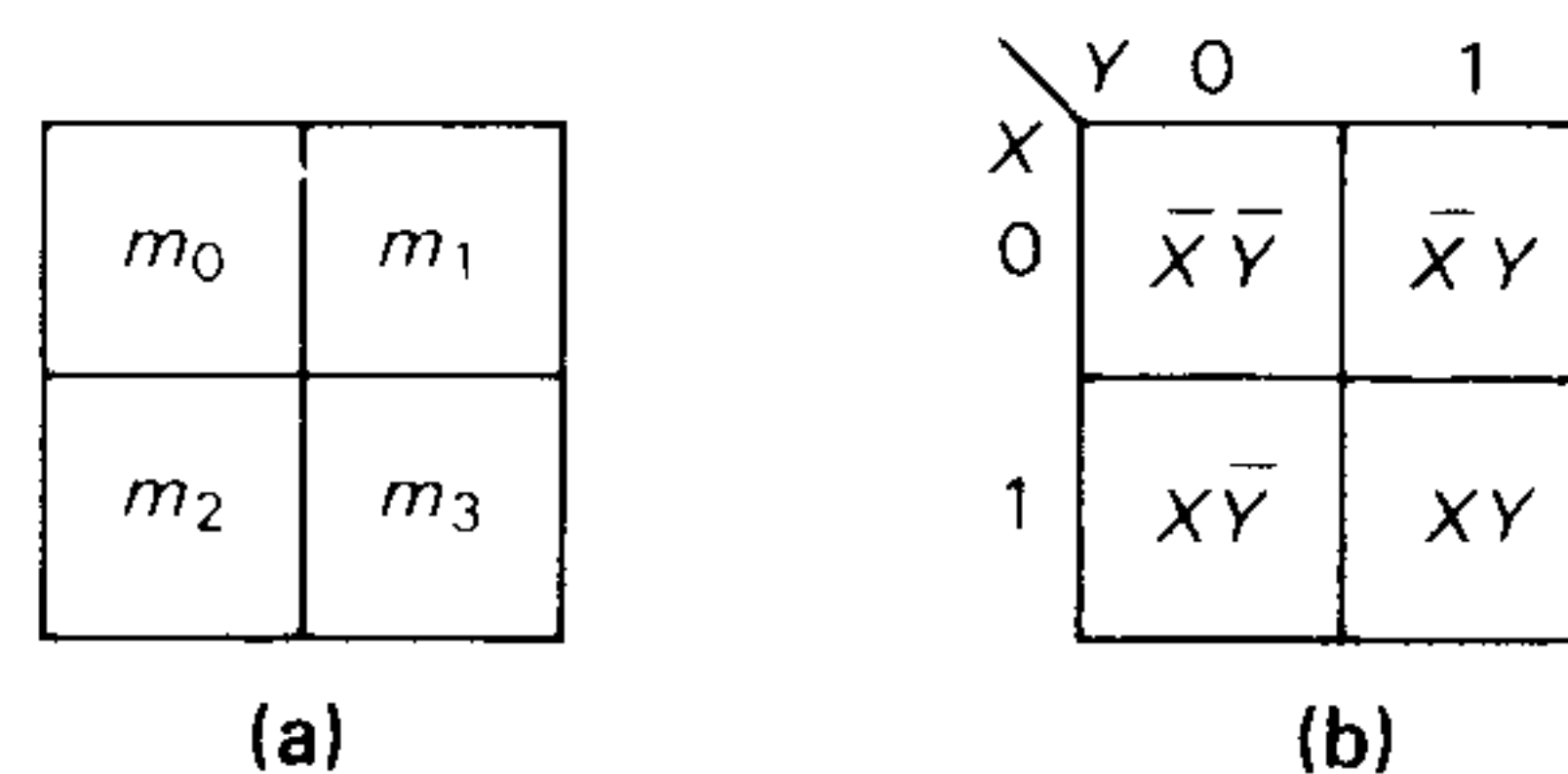


FIGURA 2—9

Mapa de dos variables

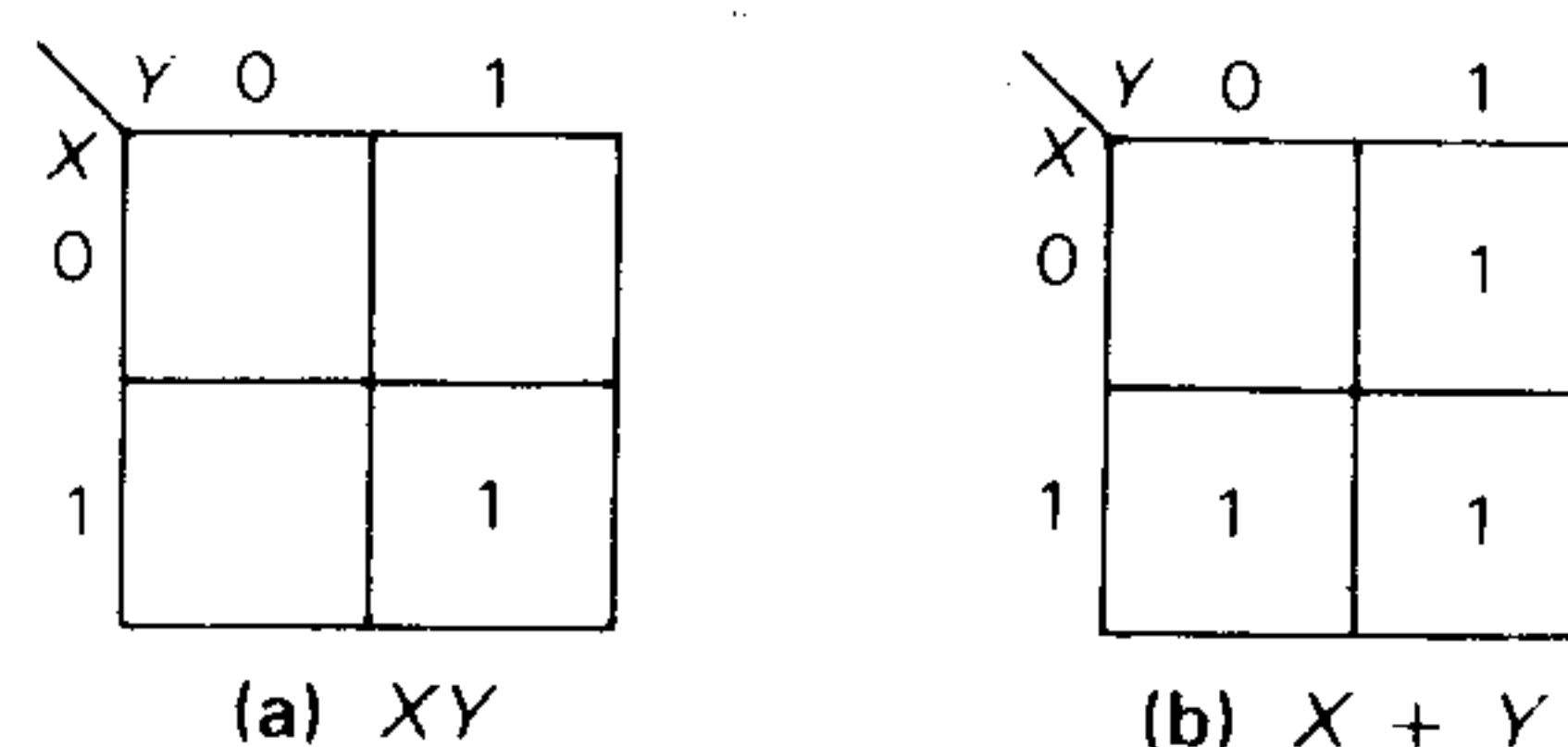


FIGURA 2—10

Representación de funciones en el mapa

lumna. Juntas, estas dos áreas encierran los tres cuadrados que pertenecen a X o Y . Esta simplificación se puede justificar por manipulación algebraica.

$$\bar{X}Y + X\bar{Y} + XY = \bar{X}Y + X(\bar{Y} + Y) = (\bar{X} + X)(Y + X) = X + Y$$

El procedimiento exacto para combinar cuadrados en el mapa se describirá con claridad en los ejemplos que siguen.

Mapa de tres variables

Hay ocho minitérminos para tres variables binarias. Por lo tanto, un mapa de tres variables consta de ocho cuadrados como se muestra en la figura 2-11. El mapa trazado en la parte (b) se marca con números binarios en cada renglón y cada columna para señalar los valores binarios de los términos mínimos. Nótese que los números de las columnas no siguen la secuencia de cuenta binaria. La característica de la secuencia mostrada es que sólo un bit cambia de valor de una columna adyacente a la que sigue.

Un cuadrado de minitérmino puede localizarse en el mapa de dos maneras. Podemos memorizar los números que aparecen en la figura 2-11(a) para cada ubicación de un minitérmino, o bien podemos referirnos a los números binarios de los renglones y columnas. Por ejemplo, el cuadrado asignado a m_5 corresponde al renglón 1 y columna 01. Cuando se eslabonan estos dos números, producen el número binario 101, cuyo equivalente decimal es 5. Otra manera de observar el cuadrado $m_5 = X\bar{Y}Z$ es considerar que está en el renglón marcado X y la columna que pertenece a $\bar{Y}Z$ (columna 01). Nótese que hay cuatro cuadrados donde cada variable es igual a 1 y cuatro cuadrados donde cada una es igual a cero. La variable figura no complementada en los cuatro cuadrados donde es igual a 1 y complementada en los 4 cuadrados donde es igual a 0. Por conveniencia, escribimos el nombre de la variable con el símbolo de la letra en los cuatro cuadrados donde no está complementada.

Para entender la utilidad del mapa para simplificar funciones booleanas, debemos reconocer la propiedad básica que poseen cuadrados adyacentes. Dos cuadrados adyacentes cualesquiera, colocados horizontal o verticalmente (pero no en diagonal), corresponden a términos mínimos que difieren sólo en una variable única. La variable figura no complementada en un cuadrado y complementada en el otro. Por ejemplo, m_5 y m_7 están en dos cuadrados adyacentes. La variable Y está complementada en m_5 y no complementada en m_7 , mientras que las otras dos variables son las mismas en los dos cuadrados. La suma lógica de dos minitérminos adyacentes, puede simplificarse en un término de producto de dos variables.

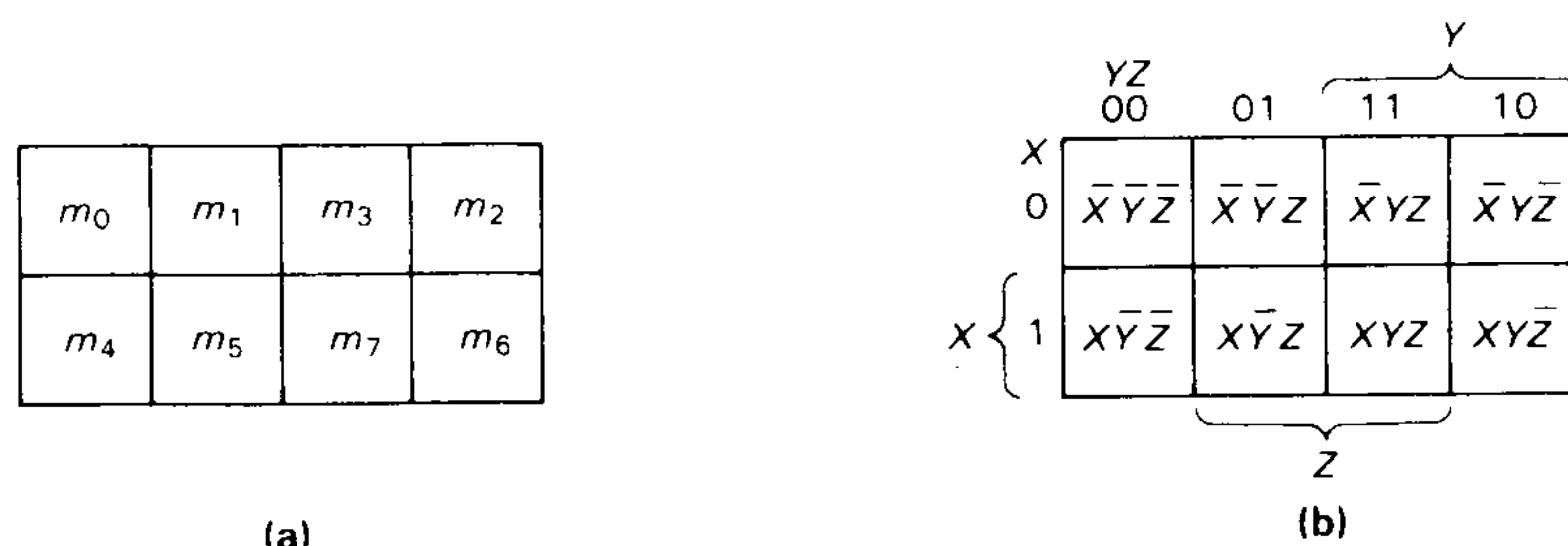


FIGURA 2—11

Mapa de tres variables

$$m_5 + m_7 = X\bar{Y}Z + XYZ = XZ(\bar{Y} + Y) = XZ$$

Aquí, los dos cuadrados difieren en la variable Y , que se puede suprimir cuando se forme la suma lógica (OR) de los dos minitérminos. Por lo tanto, dos minitérminos cualesquiera en cuadrados adyacentes que se sumen (con OR), producen un término de producto de dos variables.

Ejemplo 2-3

Simplifíquese la función booleana

$$F(X, Y, Z) = \sum m(2, 3, 4, 5)$$

Primero, se marca un 1 en cada minitérmino que representa a la función. Esto se muestra en la figura 2-12 donde los cuadrados de los minitérminos 010, 011, 100 y 101 se marcan con unos. El paso siguiente consiste en hallar posibles cuadrados adyacentes. Estos se indican en el mapa por medio de dos rectángulos, donde cada uno encierra dos unos. El rectángulo del lado superior derecho representa el área encerrada por $\bar{X}Y$. Esto se determina observando que el área de dos cuadrados está en el renglón 0, que corresponde a \bar{X} , y en las dos últimas columnas, correspondientes a Y . En forma análoga, el rectángulo del lado inferior de la izquierda representa el término de producto $X\bar{Y}$. (El segundo renglón representa a X y las dos columnas de la izquierda representan a \bar{Y} .) La suma lógica de estos dos términos de productos produce la expresión simplificada

$$F = \bar{X}Y + X\bar{Y}$$

Hay casos donde dos cuadrados del mapa se consideran adyacentes aunque no se llegan a tocar. En la figura 2-11, m_0 es adyacente a m_2 y m_4 es adyacente a m_6 porque los minitérminos difieren en una variable. Esto se puede verificar fácilmente en forma algebraica.

$$m_0 + m_2 = \bar{X}\bar{Y}\bar{Z} + \bar{X}Y\bar{Z} = \bar{X}\bar{Z}(\bar{Y} + Y) = \bar{X}\bar{Z}$$

$$m_4 + m_6 = X\bar{Y}\bar{Z} + XY\bar{Z} = X\bar{Z}(\bar{Y} + Y) = X\bar{Z}$$

En consecuencia, debemos modificar la definición de cuadrados adyacentes para incluir éste y otros casos semejantes. Esto se hace considerando que el mapa está trazado en una superficie donde los bordes de la derecha y de la izquierda se tocan entre sí para formar cuadrados adyacentes.

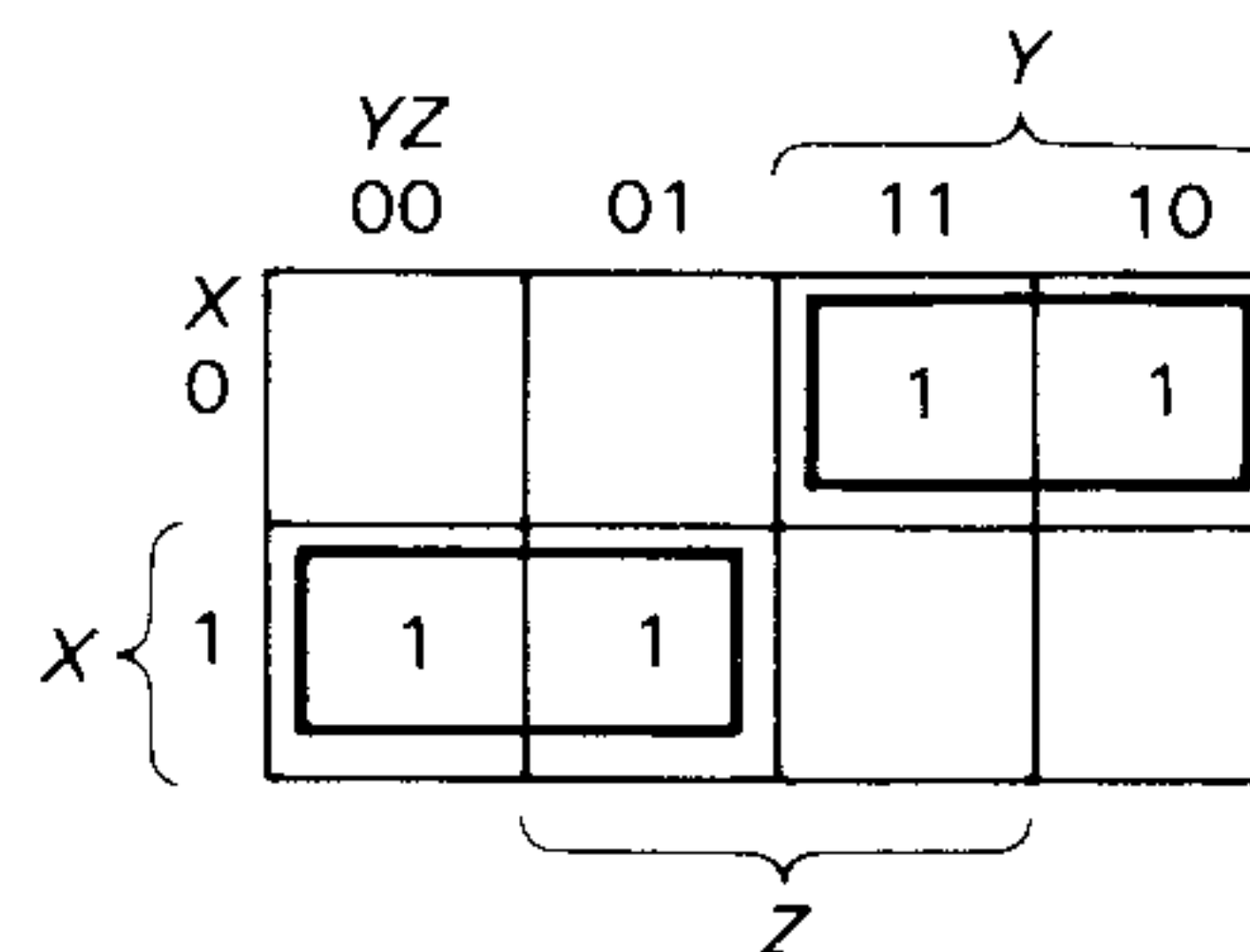


FIGURA 2—12

Mapa del ejemplo 2-3; $F(X, Y, Z) = \sum m(2, 3, 4, 5) = \bar{X}Y + X\bar{Y}$

Considérese ahora cualquier combinación de cuatro cuadrados adyacentes en el mapa de tres variables. Cualquier combinación de este tipo representa la suma lógica de cuatro minitérminos y da origen a una expresión de una sola literal. Para poner un ejemplo, la suma lógica de los cuatro minitérminos adyacentes 0, 2, 4 y 6 se reduce a un solo término literal Z .

$$\begin{aligned} m_0 + m_2 + m_4 + m_6 &= \bar{X}\bar{Y}\bar{Z} + \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + XY\bar{Z} \\ &= \bar{X}\bar{Z}(\bar{Y} + Y) + X\bar{Z}(\bar{Y} + Y) \\ &= \bar{X}\bar{Z} + X\bar{Z} = \bar{Z}(\bar{X} + X) = \bar{Z} \end{aligned}$$

El número de cuadrados adyacentes que se pueden combinar debe representar siempre un número que sea una potencia de dos como 1, 2, 4 y 8. Conforme se combina un número mayor de cuadrados adyacentes, se obtiene un término de producto con menos literales.

Un cuadrado representa un minitérmino de tres literales.

Dos cuadrados adyacentes representan un término de producto de dos literales.

Cuatro cuadrados adyacentes representan un término de producto de una literal.

Ocho cuadrados adyacentes abarcan todo el mapa y producen una función que siempre es igual al 1 lógico.

Ejemplo 2-4

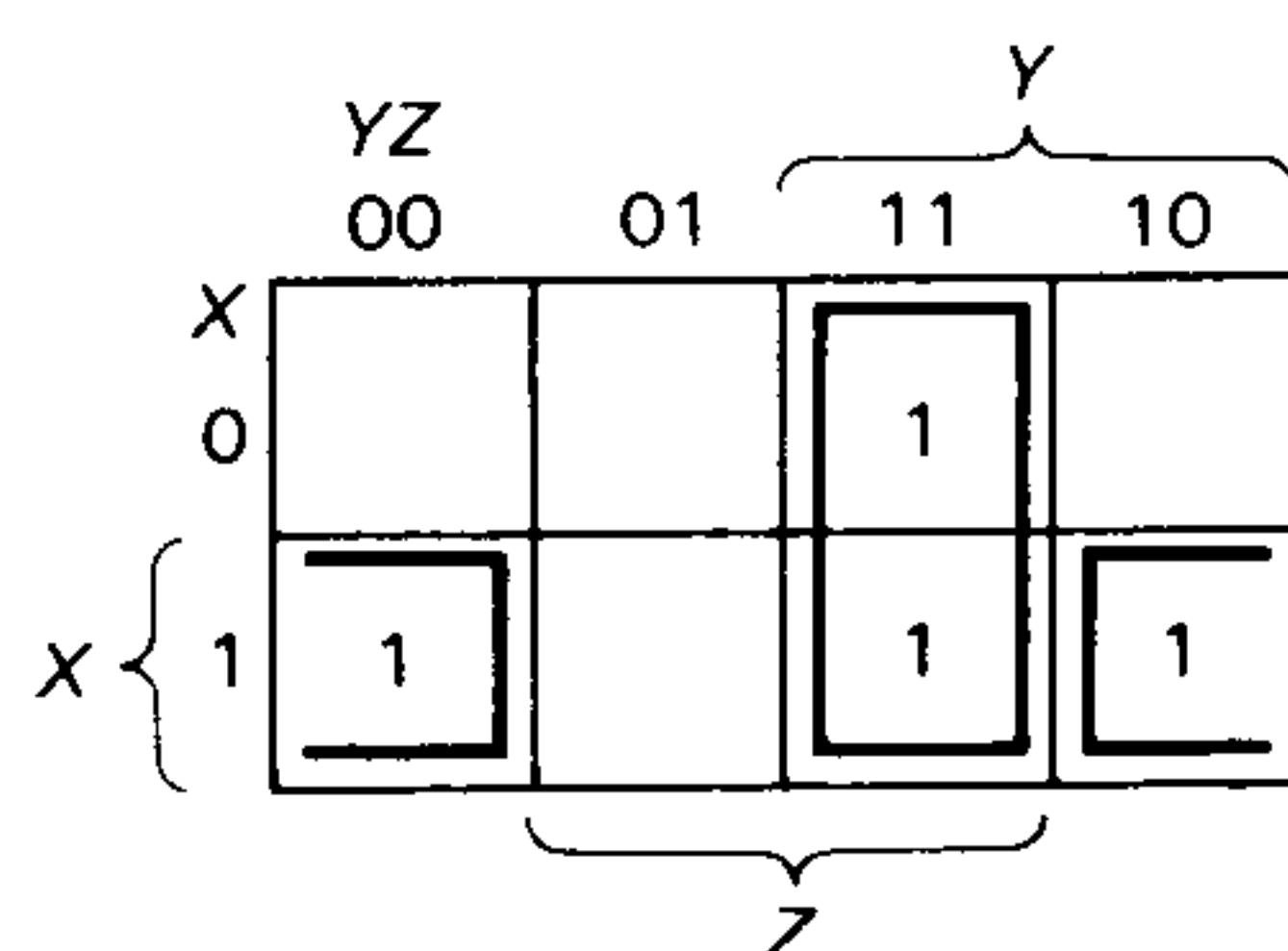
Simplifiquense las funciones booleanas

$$F_1(X, Y, Z) = \sum m(3, 4, 6, 7)$$

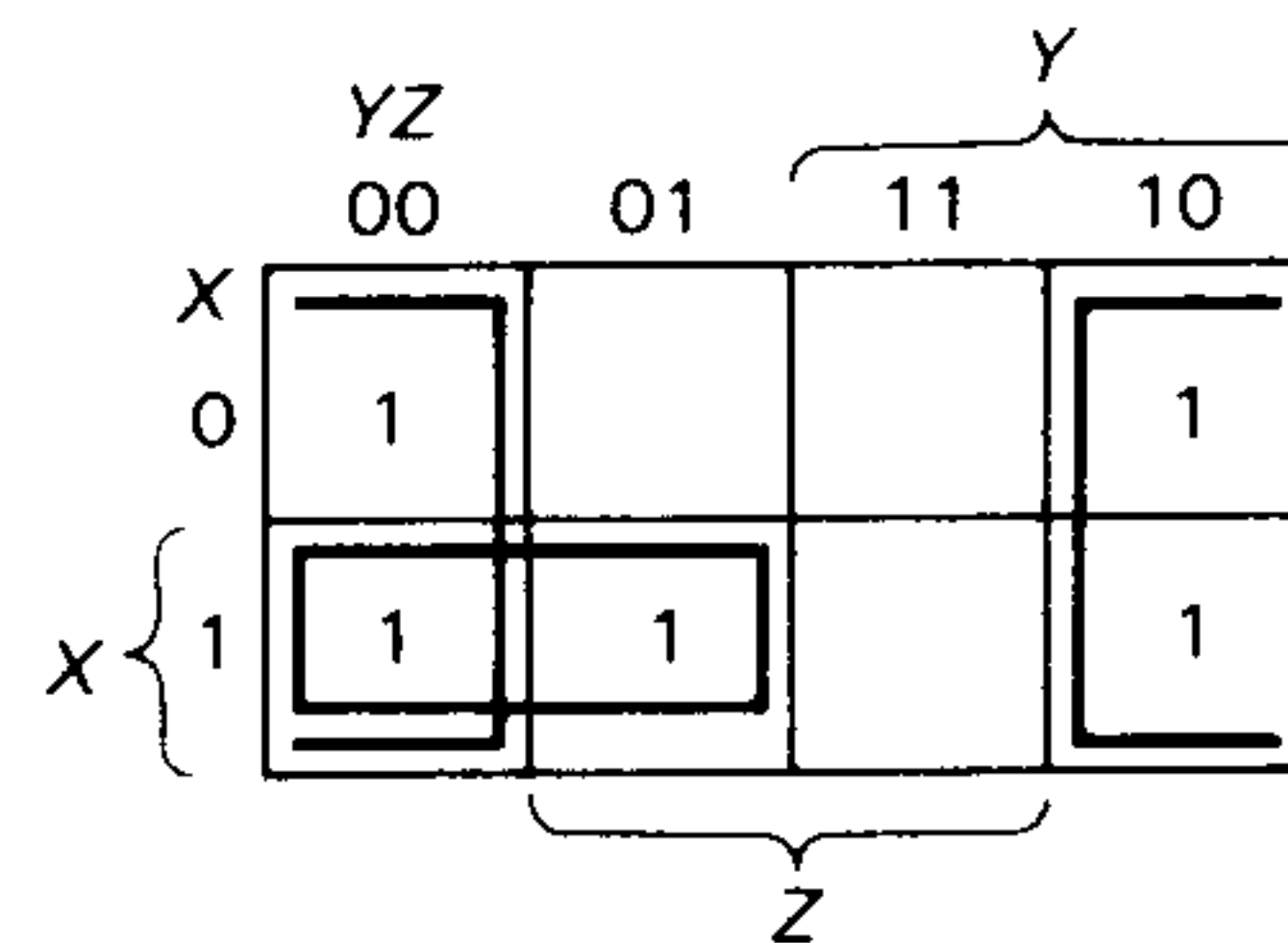
$$F_2(X, Y, Z) = \sum m(0, 2, 4, 5, 6)$$

El mapa de F_1 se ilustra en la figura 2-13(a). Hay cuatro cuadrados marcados con unos, uno para cada minitérmino de la función. Se combinan dos cuadros adyacentes en la tercera columna para producir un término YZ de dos literales. Los dos cuadrados restantes con unos son también adyacentes por la nueva definición y se muestran en el diagrama con sus valores encerrados en medios rectángulos (o áreas semirrectangulares). Estos dos cuadrados, cuando se combinan, producen el término $X\bar{Z}$ de dos literales. La función simplificada se convierte en

$$F_1 = YZ + X\bar{Z}$$



$$\begin{aligned} \text{(a) } F_1(X, Y, Z) &= \sum m(3, 4, 6, 7) \\ &= YZ + X\bar{Z} \end{aligned}$$



$$\begin{aligned} \text{(b) } F_2(X, Y, Z) &= \sum m(0, 2, 4, 5, 6) \\ &= \bar{Z} + X\bar{Y} \end{aligned}$$

FIGURA 2—13

Mapas del ejemplo 2-4

		YZ		Y	
		00	01	11	10
X	0		1	1	
X	1	1	1		1
		Z			

FIGURA 2-14

$$\begin{aligned}
 F(X, Y, Z) &= \sum m(1, 3, 4, 5, 6) \\
 &= \bar{X}Z + X\bar{Z} + X\bar{Y} \\
 &= \bar{X}Z + X\bar{Z} + \bar{Y}Z
 \end{aligned}$$

El mapa de F_2 se presenta en la figura 2-13(b). Primero se combinan los cuatro cuadrados adyacentes en la primera y última columnas para generar el término \bar{Z} de una sola literal. El cuadrado restante que representa al minitérmino 5 se combina con un cuadrado adyacente que ya se ha usado una vez. Esto no sólo es permisible sino deseable, puesto que los dos cuadrados adyacentes producen el término $X\bar{Y}$ de dos literales, mientras que el cuadrado individual representa el minitérmino $X\bar{Y}Z$ de tres literales. La función simplificada es

$$F_2 = \bar{Z} + X\bar{Y}$$

En algunas ocasiones hay dos maneras alternativas de combinar cuadrados para producir expresiones igualmente simplificadas. Un ejemplo de esto se demuestra en el mapa de la figura 2-14. Los minitérminos 1 y 3 se combinan para producir el término $\bar{X}Z$ y los minitérminos 4 y 6 producen el término $X\bar{Z}$. Sin embargo, hay dos formas en las que se puede combinar el cuadrado del minitérmino 5 con otro cuadrado adyacente para producir un tercer término de dos literales. Al combinarlo con el minitérmino 4 se produce el término $X\bar{Y}$. Podríamos optar en cambio por combinarlo con el minitérmino 1 para generar el término $\bar{Y}Z$. Cada una de las dos expresiones simplificadas posibles de la figura 2-14 tiene tres términos de dos literales cada uno; así que hay dos posibles soluciones simplificadas para esta función.

Si una función no se expresa como una suma de minitérminos, podemos emplear el mapa para obtener los minitérminos de la función y luego simplificar la función. Es necesario tener la expresión algebraica en suma de productos, de la cual cada término de producto se grafica en el mapa. Los minitérminos de la función se leen después en forma directa del mapa. Considérese la siguiente función booleana.

$$F = \bar{X}Z + \bar{X}Y + X\bar{Y}Z + YZ$$

Tres términos de productos de la expresión tienen dos literales y se representan en un mapa de tres variables por medio de dos cuadrados cada uno. Los dos cuadrados correspondientes al primer término, $\bar{X}Z$, se obtienen en la figura 2-15 a partir de la coincidencia de \bar{X} (primer renglón) y Z (dos columnas del centro) para producir los cuadrados 001 y 011. Nótese que cuando se marcan unos en los cuadrados, es posible encontrar un 1 ya colocado ahí de un término anterior. Esto sucede con el segundo término $\bar{X}Y$ que tiene unos en los cuadrados 011 y 010, pero el cuadrado 011 es común con el primer término $\bar{X}Z$ de modo que sólo se marca un 1 en él. Procediendo de esta manera vemos que la función tiene cinco minitérminos como lo indican los cinco unos del mapa de la figura 2-15. Los minitérminos se leen directamente

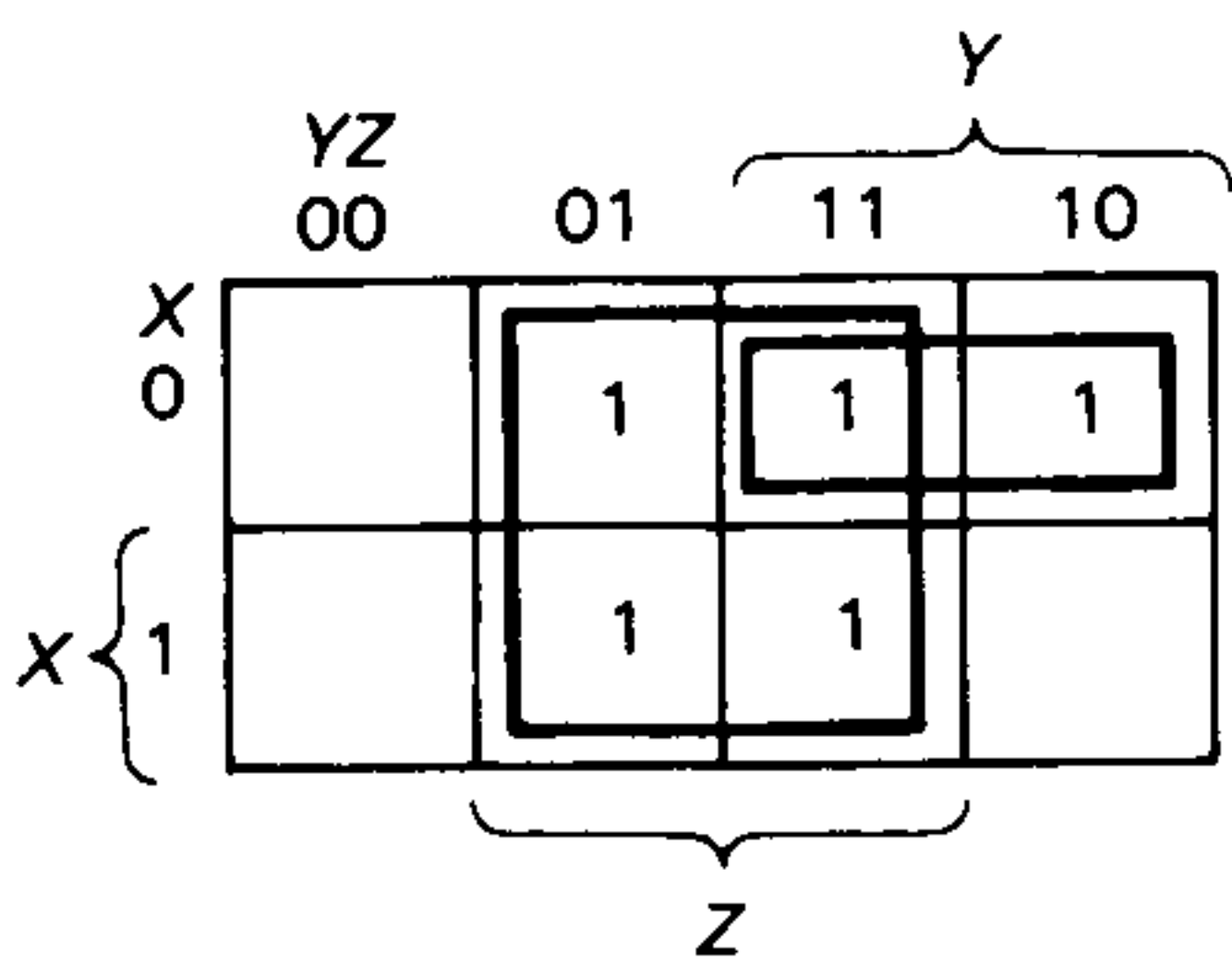


FIGURA 2—15
 $F(X, Y, Z) = \sum m(1, 2, 3, 5, 7) =$
 $Z + \bar{X}Y$

del mapa y son 1, 2, 3, 5 y 7. La función como está expresada originalmente tiene demasiados términos de productos. Se puede simplificar sólo a dos términos.

$$F = Z + \bar{X}Y$$

Mapa de cuatro variables

Hay 16 minitérminos para cuatro variables binarias y, en consecuencia, un mapa de cuatro variables consta de 16 cuadrados, como se ve en la figura 2-16. La asignación de minitérminos en cada cuadrado se indica en la parte (a) del diagrama. El mapa se vuelve a trazar en la parte (b) para mostrar la relación de las cuatro variables. Los renglones y las columnas están numerados en una secuencia especial, de manera que sólo un bit del número binario cambia de valor entre dos cuadrados adyacentes cualesquiera. Los minitérminos correspondientes a cada cuadrado se pueden obtener a partir del eslabonamiento del número de renglón con el número de columna. Por ejemplo, los números del tercer renglón (11) y la segunda columna (01) cuando se eslabonan dan el número binario 1101, que es el equivalente binario de 13. Por lo tanto, el cuadrado del tercer renglón y segunda columna representa el minitérmino m_{13} . Además, cada variable se marca en el mapa para mostrar los ocho cuadrados donde aparece no complementada. Los otros ocho cuadrados en los que no se indica

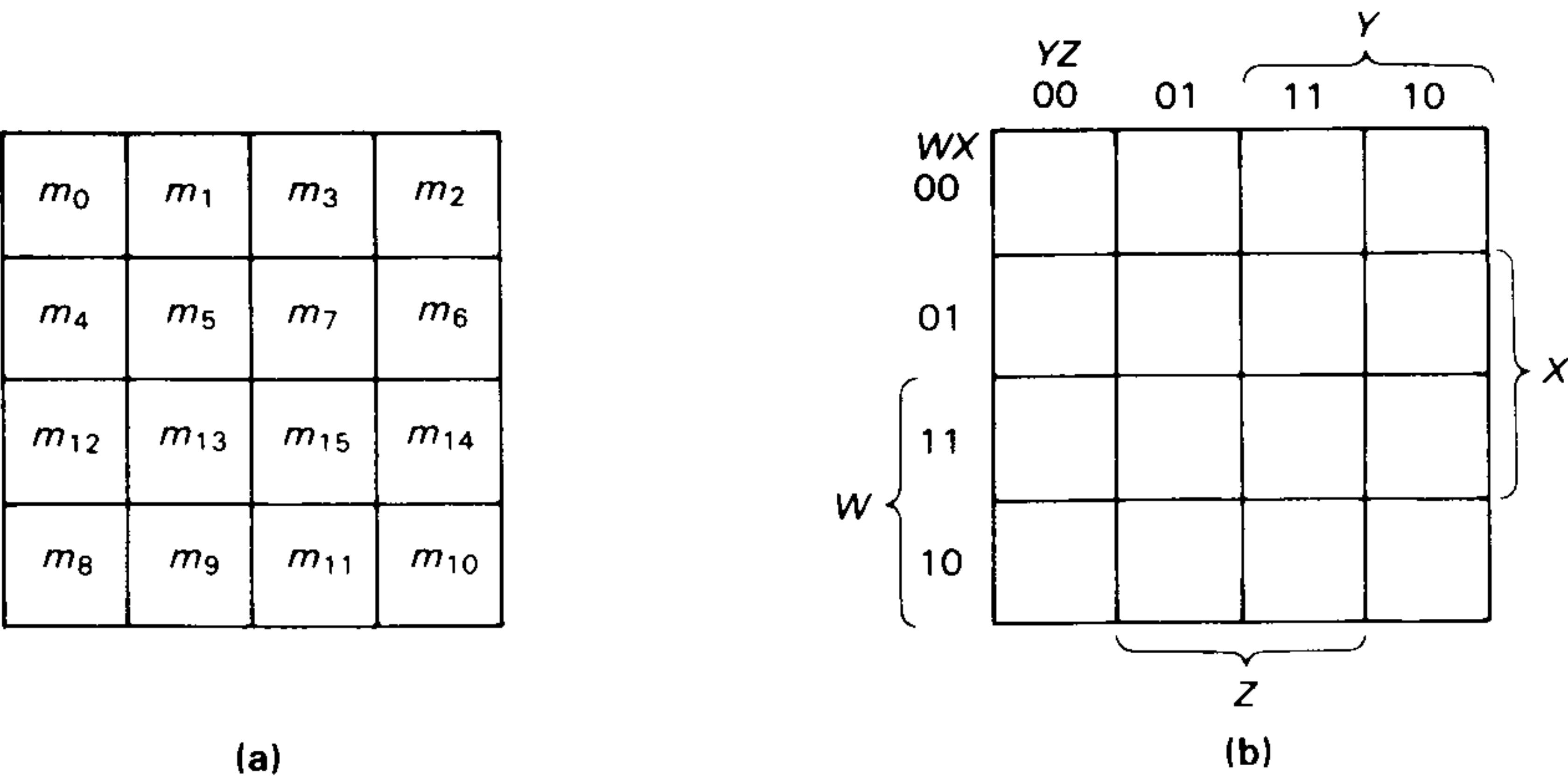


FIGURA 2—16
Mapa de cuatro variables

ningún rótulo corresponden a la variable que se complementa. Por lo tanto, W figura complementada en los dos primeros renglones y no complementada en los dos segundos renglones.

La simplificación del mapa de funciones de cuatro variables es similar al método que se emplea para simplificar funciones de tres variables. Los cuadrados adyacentes se definen como cuadrados próximos entre sí. Además, se considera que el mapa está sobre una superficie con los bordes superior e inferior, así como también los bordes de la derecha y la izquierda, haciendo contacto para formar cuadrados adyacentes. Por ejemplo, m_0 y m_2 son dos cuadrados adyacentes, como lo son también m_3 y m_{11} . La combinación de cuadrados que se pueden tomar durante el proceso de simplificación en el mapa de cuatro variables, es como sigue:

Un cuadrado representa un minitérmino de cuatro literales.

Dos cuadrados adyacentes representan un término de producto de tres literales.

Cuatro cuadrados adyacentes representan un término de producto de dos literales.

Ocho cuadrados adyacentes denotan un término de producto de una literal.

Dieciséis cuadrados producen una función que siempre es igual al 1 lógico.

No se puede utilizar ninguna otra combinación de cuadrados adyacentes. Los ejemplos que siguen ilustran el procedimiento de simplificación de funciones booleanas de cuatro variables.

Ejemplo 2-5

Simplifíquese la función booleana

$$F(W, X, Y, Z) = \sum m(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$$

Los minitérminos de la función se marcan con unos en el mapa de la figura 2-17. Ocho cuadrados adyacentes en las dos columnas de la izquierda se combinan para formar el único término literal \bar{Y} . Los 3 unos restantes no se pueden combinar para producir un término simplificado. Deben combinarse como dos o cuatro cuadrados adyacentes. Los 2 unos de la parte superior de la derecha se combinan con los 2 unos de la parte superior de la izquierda para generar el término $\bar{W}Z$. Obsérvese una

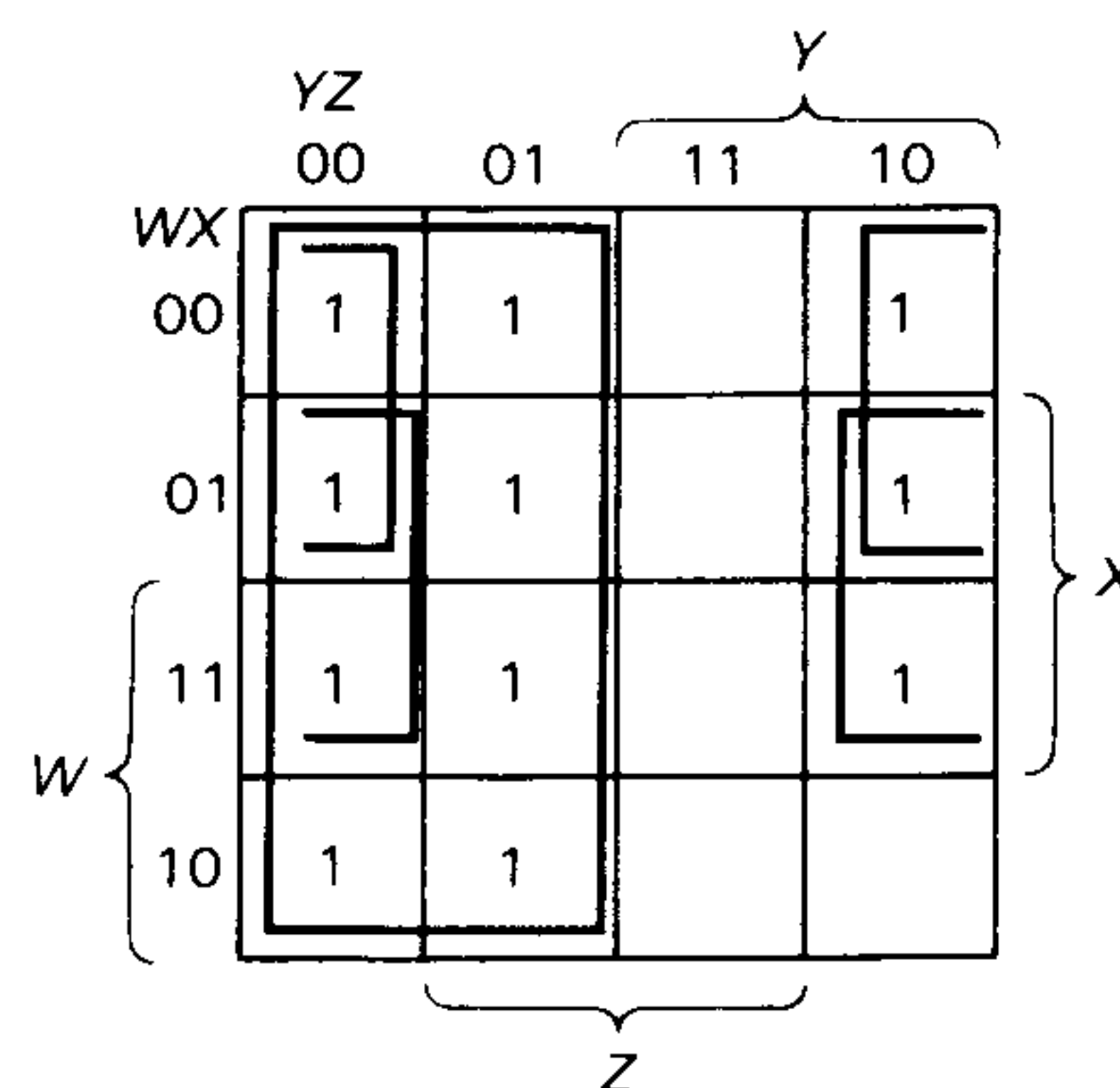


FIGURA 2-17

Mapa del ejemplo 2-5; $F = \bar{Y} + \bar{W}Z + X\bar{Z}$

vez más que es permisible tomar el mismo cuadrado más de una vez. Ahora nos queda un cuadrado marcado con un 1 en el tercer renglón y la cuarta columna (minitérmino 1100). En vez de tomar este cuadrado solo, que producirá un término de cuatro literales, se combina con otros que ya se utilizaron para formar un área de cuatro cuadrados adyacentes en los dos renglones del centro y las dos columnas finales, que da origen al término $X\bar{Z}$. La expresión simplificada es la suma lógica de los términos

$$F = \bar{Y} + \bar{W}\bar{Z} + X\bar{Z}$$

Ejemplo 2-6

Simplifíquese la función booleana

$$F = \bar{A}\bar{B}\bar{C} + \bar{B}\bar{C}\bar{D} + \bar{A}BC\bar{D} + A\bar{B}\bar{C}$$

Esta función tiene cuatro variables, A , B , C y D . Esta se expresa en suma de productos con tres términos de tres literales cada uno y un término de cuatro literales. El área representada en el mapa que está cubierta por esta función se muestra en la figura 2-18. Cada término de tres literales se representa en el mapa con dos cuadrados. $\bar{A}\bar{B}\bar{C}$ se representa en los cuadrados 0000 y 0001, $\bar{B}\bar{C}\bar{D}$ en los cuadrados 0010 y 1010, y $A\bar{B}\bar{C}$ en 1000 y 1001. El término con cuatro literales es el minitérmino 0110. La función se simplifica en el mapa tomando los unos de las cuatro esquinas para producir el término $\bar{B}\bar{D}$. Esto es posible porque estos cuatro cuadrados son adyacentes cuando el mapa se traza en una superficie donde los bordes superior e inferior o de la izquierda y la derecha están en contacto. Los 2 unos del renglón de arriba se combinan con los 2 unos del renglón de abajo para producir el término $\bar{B}\bar{C}$. El 1 restante del cuadrado 0110 se combina con el cuadrado adyacente 0010 para dar el término $\bar{A}\bar{C}\bar{D}$. La función simplificada es

$$F = \bar{B}\bar{D} + \bar{B}\bar{C} + \bar{A}\bar{C}\bar{D}$$

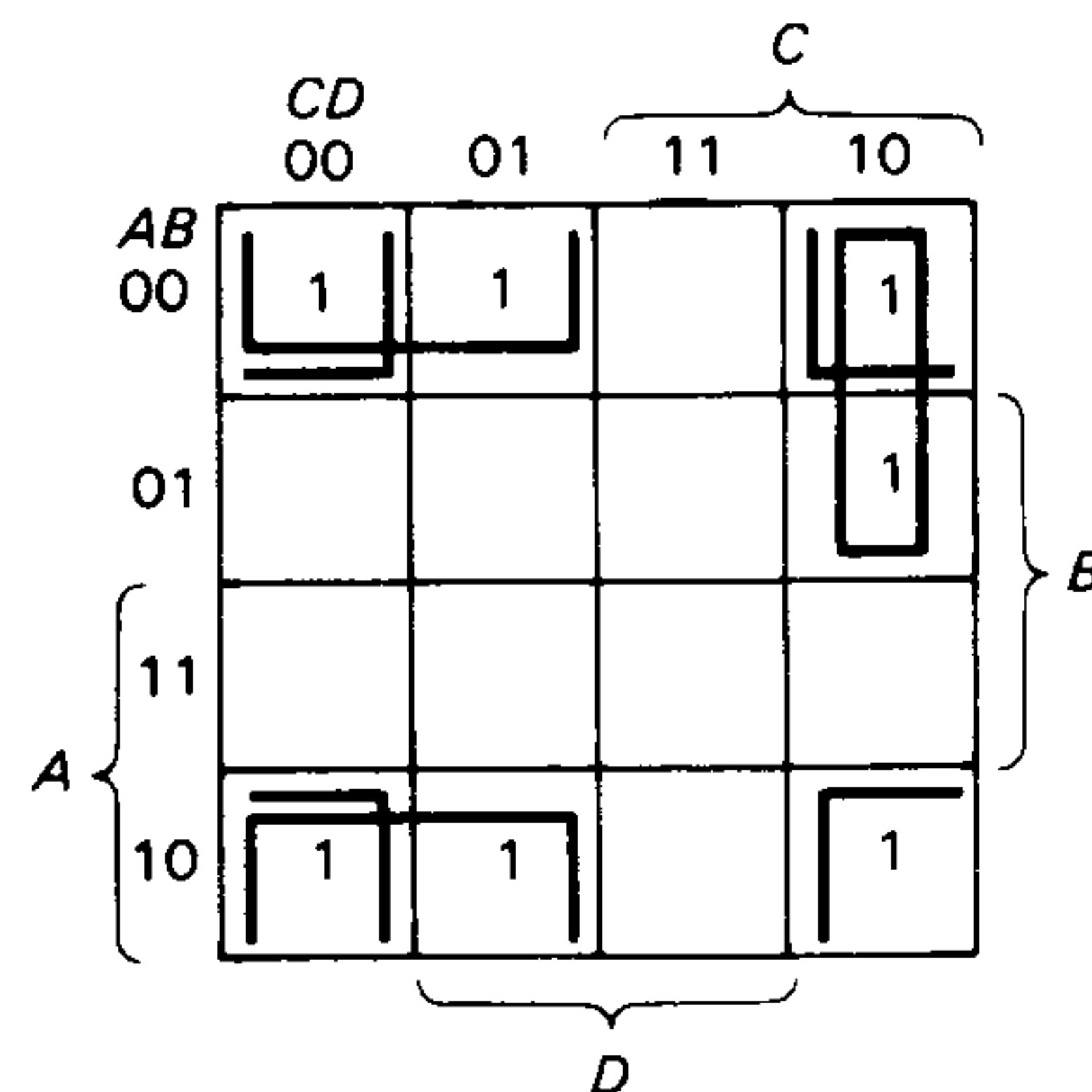


FIGURA 2—18

Mapa del ejemplo 2-6; $F = \bar{B}\bar{D} + \bar{B}\bar{C} + \bar{A}\bar{C}\bar{D}$
+ $\bar{A}\bar{C}\bar{D}$

2-5 MANIPULACION DE MAPAS

Cuando se combinan cuadrados adyacentes de un mapa, es necesario asegurarnos de que estén incluidos todos los minitérminos de la función. Al mismo tiempo es necesario minimizar el número de términos en la función simplificada evitando cualquier término redundante cuyos términos mínimos ya estén cubiertos por otros. En esta sección consideramos un procedimiento de manipulación que facilita el reconocimiento de patrones correctos en el mapa. Se abarcarán otros temas, como la simplificación del producto de sumas y la simplificación de funciones especificadas en forma incompleta. Asimismo, mostraremos cómo se puede extender el mapa de cuatro variables para cubrir funciones booleanas de cinco variables.

Implicantes primos esenciales

El procedimiento de combinación de cuadrados en el mapa puede hacerse más sistemático si entendemos el significado de los términos conocidos como implicante primo e implicante primo esencial. Un implicante primo es un término de producto que se obtiene combinando el máximo número posible de cuadrados adyacentes en el mapa. Si un minitérmino de un cuadrado está cubierto sólo por un implicante primo, se dice que tal implicante primo es esencial. En la figura 2-14, los términos $\bar{X}Z$ y $X\bar{Z}$ son implicantes primos esenciales y los términos $X\bar{Y}$ y $\bar{Y}Z$ son implicantes primos no esenciales.

Los implicantes primos de una función se pueden obtener del mapa combinando el número máximo de cuadrados posibles. Esto quiere decir que un 1 en un mapa representa un implicante primo si no es adyacente a ningún otro 1. Dos unos adyacentes forman un implicante primo siempre que no estén dentro de un grupo de cuatro cuadrados adyacentes. Cuatro unos adyacentes forman un implicante primo si no se encuentran dentro de un grupo de ocho cuadrados adyacentes, etc. Los implicantes primos esenciales se determinan observando cada cuadrado marcado con un 1 y revisando el número de implicantes primos que lo cubren. El implicante primo es esencial si es el único que cubre el minitérmino. El procedimiento para obtener la expresión simplificada del mapa requiere que se determinen primero todos los implicantes primos esenciales. La expresión simplificada se obtiene a partir de la

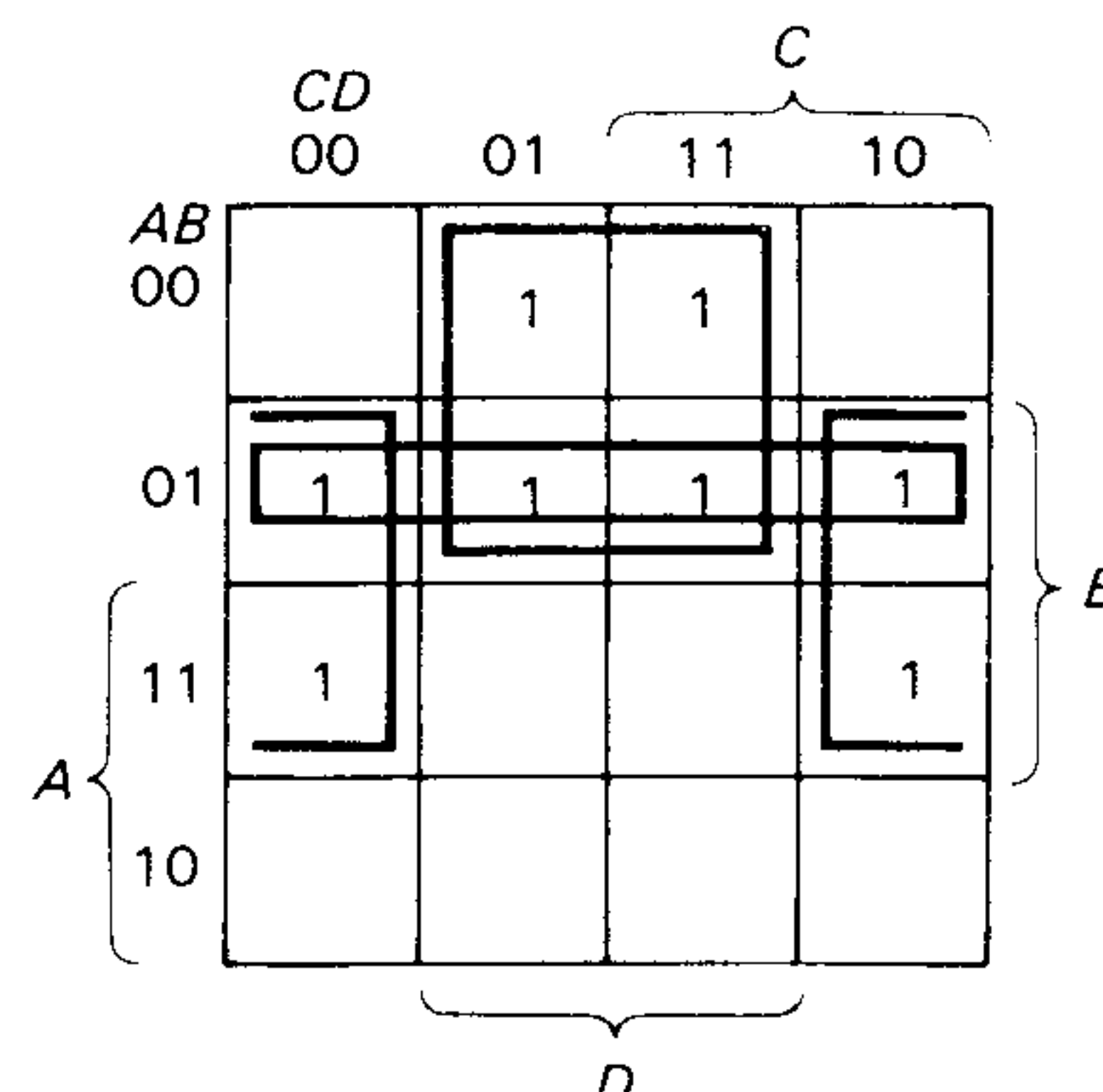


FIGURA 2—19

Implicantes primos $\bar{A}D$, $B\bar{D}$ y $\bar{A}B$

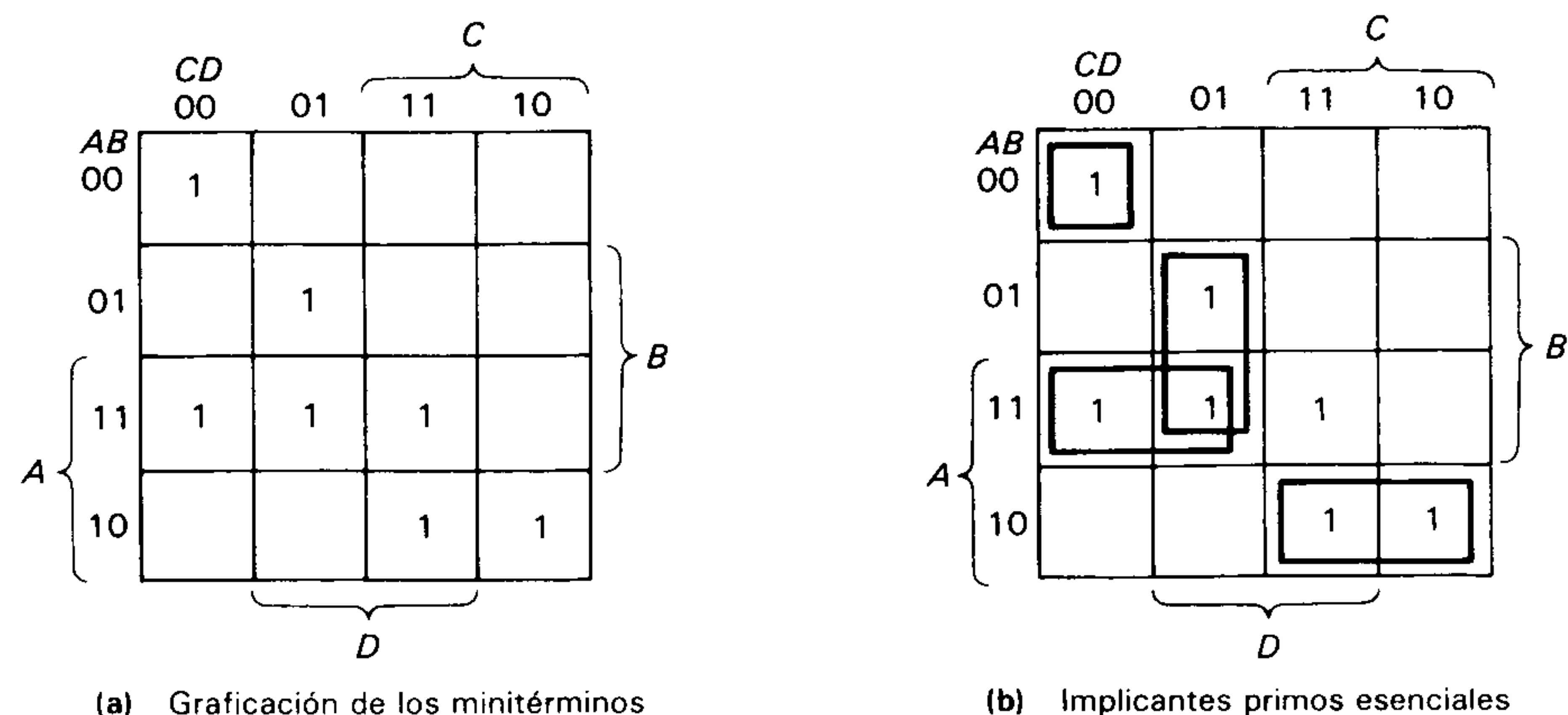


FIGURA 2—20

Simplificación con implicantes primos

suma lógica de todos los implicantes primos esenciales, más otros implicantes primos que puedan necesitarse para cubrir cualquier minitérmino restante no cubierto por los implicantes primos esenciales. Este procedimiento se aclarará por medio de dos ejemplos.

Considérese el mapa de la figura 2-19. Hay tres formas en las que podemos combinar cuatro cuadrados adyacentes. Los términos de productos que se obtienen a partir de estas combinaciones son los implicantes primos de la función. Los términos $\overline{A}D$ y $B\overline{D}$ son implicantes primos esenciales pero $\overline{A}B$ no es esencial. Esto se debe a que los minitérminos 1 y 3 sólo pueden ser cubiertos por el término $\overline{A}D$ y los minitérminos 12 y 14 sólo puede cubrirlos el término $B\overline{D}$. Pero los términos mínimos 4, 5, 6 y 7 son cubiertos por dos implicantes primos, de modo que el término $\overline{A}B$ no es un implicante primo esencial. De hecho, una vez que se toman los implicantes primos esenciales, el tercer término no se necesita en este caso porque todos los minitérminos ya han sido cubiertos por los implicantes primos esenciales. La expresión simplificada de la función de la figura 2-19 es

$$F = \overline{A}D + B\overline{D}$$

En la figura 2-20 se presenta un segundo ejemplo. La función graficada en la parte (a) tiene siete minitérminos. Si intentamos combinar cuadrados adyacentes, veremos que hay seis implicantes primos. A fin de obtener un número mínimo de términos para la función, debemos determinar antes los implicantes primos que sean esenciales. Como se indica en la parte (b) de la figura, la función tiene cuatro implicantes primos esenciales. El término de producto $\overline{A}BCD$ es esencial porque es el único implicante primo que cubre el minitérmino 0. Análogamente, los términos de producto $B\overline{C}D$, $AB\overline{C}$ y $\overline{A}BC$ son implicantes primos esenciales puesto que son los únicos que cubren los minitérminos 5, 12 y 10, respectivamente. El minitérmino 15 puede ser cubierto por dos implicantes primos. La expresión simplificada de la función consta de la suma lógica de los cuatro implicantes primos esenciales y un implicante primo que cubre el minitérmino 15.

$$F = \overline{A}BCD + B\overline{C}D + AB\overline{C} + \overline{A}BC + \begin{cases} ACD \\ 0 \\ ABD \end{cases}$$

La identificación de implicantes primos esenciales en el mapa aumenta el patrón de cuadrados adyacentes y muestra las alternativas disponibles para simplificar la función booleana.

Simplificación de productos de sumas

Las funciones booleanas simplificadas que se obtienen del mapa en todos los ejemplos anteriores se expresaron en suma de productos. Con una mínima modificación, se puede obtener la forma de producto de sumas.

El procedimiento para obtener una expresión simplificada en producto de sumas sigue las propiedades básicas de las funciones booleanas. Los unos colocados en los cuadrados del mapa representan los minitérminos de la función. Los minitérminos no incluidos en la función pertenecen al complemento de ésta. De esto vemos que el complemento de una función se representa en el mapa por medio de los cuadrados no marcados con unos. Si marcamos los cuadrados vacíos con ceros y combinamos éstos en cuadrados adyacentes válidos, obtenemos una expresión simplificada del complemento de la función. Después se toma el complemento de \bar{F} para obtener la función F como un producto de sumas. Esto se hace calculando el dual y complementando cada literal, según se describe en el ejemplo 2-2.

Ejemplo 2-7

Simplifíquese la siguiente función booleana en forma de producto de sumas.

$$F(A, B, C, D) = \sum m(0, 1, 2, 5, 8, 9, 10)$$

Los unos marcados en el mapa de la figura 2-21 representan los minitérminos de la función. Los cuadrados marcados con ceros representan los términos mínimos no incluidos en F y por lo tanto denotan el complemento de F . Combinando los cuadrados marcados con ceros se obtiene la función complementada simplificada

$$\bar{F} = AB + CD + B\bar{D}$$

Calculando el dual y complementando cada literal se tiene el complemento de \bar{F} . Esta expresión es F en forma de producto de sumas.

$$F = (\bar{A} + \bar{B})(\bar{C} + \bar{D})(\bar{B} + D)$$

El ejemplo anterior muestra el procedimiento de obtención de la simplificación del producto de sumas cuando la función se expresa originalmente en producto de maxitérminos o producto de sumas. Recuérdese que los números maxitérminos son los mismos que los números minitérminos de la función complementada; así, se colocan ceros en el mapa para representar los maxitérminos o para el complemento de la función. Para colocar una función expresada como un producto de sumas en el mapa debemos tomar el complemento de la función y, a partir de éste, determinar qué cuadrados se marcarán con ceros. Por ejemplo, la función

$$F = (\bar{A} + \bar{B} + C)(B + D)$$

se puede graficar en el mapa calculando primero su complemento

$$\bar{F} = ABC + \bar{B}\bar{D}$$

y después marcando ceros en los cuadrados que representan los minitérminos de \bar{F} .

		CD	00	01	11	10	
AB	00		1	1	0	1	
	01		0	1	0	0	B
	11		0	0	0	0	
A	10		1	1	0	1	
			D				

FIGURA 2—21

Mapa del ejemplo 2-7;

$$F = (\bar{A} + \bar{B}) (\bar{C} + \bar{D}) (\bar{B} + D)$$

Los cuadrados restantes se marcan con unos. Luego, al combinar los unos se obtiene la expresión simplificada en forma de suma de productos. Combinando los ceros y después complementando se obtiene la expresión simplificada en producto de sumas. En consecuencia, para cualquier función trazada en el mapa podemos determinar la función simplificada en una u otra de las dos formas estándar.

Condiciones no importa

La lista de minitérminos de una función booleana especifica las condiciones en las cuales la función es igual a 1. Se supone que la función es igual a 0 para el resto de los minitérminos. Esta hipótesis no siempre es válida porque hay aplicaciones donde la función no se especifica para ciertas combinaciones de las variables. Para poner un ejemplo, el código binario de cuatro bits de los dígitos decimales tiene seis combinaciones que no se utilizan y en consecuencia se consideran como no especificadas. Las funciones que tienen salidas no especificadas para algunas combinaciones de entrada se llaman funciones especificadas en forma incompleta. En la mayoría de las aplicaciones, simplemente no nos importa qué valor asume la función para los minitérminos no especificados de una función. Por esta razón se acostumbra llamar condiciones no importa a los términos mínimos no especificados de una función. Estas condiciones no importa se pueden utilizar en un mapa para simplificar más la función.

Debe entenderse que un minitérmino no importa no se puede marcar con un 1 en el mapa porque esto requeriría que la función sea siempre 1 para esta combinación. De igual manera, colocar un 0 en el cuadrado requiere que la función sea 0. Para distinguir la condición no importa de unos y ceros, se utiliza una cruz (X). Por lo tanto, una cruz dentro de un cuadrado del mapa indica que no nos importa si el valor de 0 o 1 es asignado a F para el minitérmino en particular.

Cuando se eligen cuadrados adyacentes para simplificar la función en un mapa, los minitérminos no importa pueden suponerse como 1 o 0. Cuando se simplifica la función, se puede optar por incluir cada término no importa con los unos o los ceros, dependiendo de qué combinación genere la expresión más simple. Además, un minitérmino no importa no necesita tomarse en absoluto si no contribuye a cubrir

un número mayor de cuadrados. La elección depende enteramente de la simplificación que se pueda lograr.

Para aclarar el procedimiento de manejo de las condiciones no importa, considérese la siguiente función especificada en forma incompleta que tiene tres términos mínimos no importa.

$$F(W, X, Y, Z) = \Sigma m(1, 3, 7, 11, 15)$$

$$d(W, X, Y, Z) = \Sigma m(0, 2, 5)$$

Los minitérminos de F son las combinaciones variables que hacen que la función sea igual a 1. Los minitérminos de d son los minitérminos no importa a los que se les puede asignar 0 o 1. La simplificación del mapa se presenta en la figura 2-22. Los minitérminos de F se marcan con unos, los de d se marcan con cruces y los cuadrados restantes se llenan con ceros. Para obtener la función simplificada en suma de productos, debemos incluir los cinco unos en el mapa, pero podemos o no incluir alguna de las cruces, según la forma en que se simplifique la función. El término YZ cubre los cuatro minitérminos de la tercera columna. El minitérmino restante del cuadrado 0001 se puede combinar con el cuadrado 0011 para producir un término de tres literales. Sin embargo, incluyendo una o dos cruces adyacentes podemos combinar cuatro cuadrados adyacentes para obtener un término de dos literales. En la parte (a) del diagrama se incluyen los minitérminos no importa 0 y 2 con los unos, lo que da lugar a la función simplificada

$$F = YZ + \overline{W}X$$

En la parte (b), el minitérmino no importa 5 se incluye con los unos y ahora la función simplificada es

$$F = YZ + \overline{W}Z$$

Las dos expresiones citadas representan dos funciones que son algebraicamente desiguales. Ambas cubren los minitérminos especificados de la función, pero cada una cubre diferentes minitérminos no importa. Hasta donde concierne a la función especificada en forma incompleta, ambas expresiones son aceptables. La única diferencia está en el valor de F para los minitérminos no especificados.

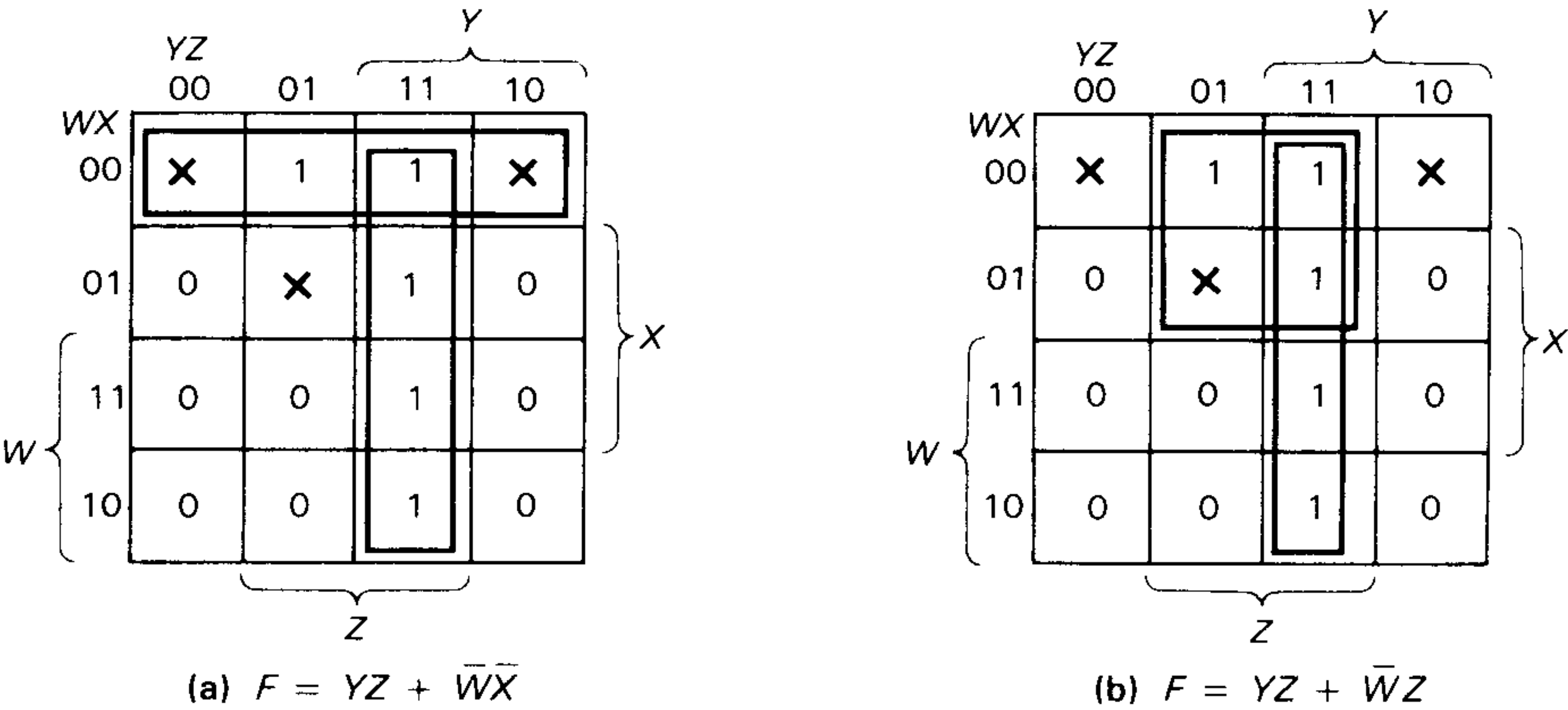


FIGURA 2—22
Ejemplo con condiciones no importa

También es posible obtener una expresión simplificada de producto de sumas para la función de la figura 2-22. En este caso, la manera de combinar los ceros es incluir los minitérminos no importa 0 y 2 con los ceros para producir una función complementada simplificada

$$\overline{F} = \overline{Z} + W\overline{Y}$$

Al tomar el complemento de \overline{F} se obtiene la expresión simplificada en forma de producto de sumas.

$$F = Z(\overline{W} + Y)$$

Los ejemplos anteriores indican que los términos mínimos no importa del mapa se consideran inicialmente representativos de una elección de 1 o 0. La elección se hace dependiendo de la forma en que se desee simplificar la función especificada en forma incompleta. Sin embargo, una vez que se hace la elección, la función simplificada tendrá valores específicos para todos los minitérminos de la función, incluyendo los que inicialmente no estaban especificados.

Mapa de cinco variables

Los mapas de más de cuatro variables no son sencillos de usar. Uno de cinco variables necesita 32 cuadrados y uno de seis variables necesita 64. Cuando el número de variables se vuelve grande, el número de cuadrados se hace excesivamente grande y tiene una mayor participación la geometría de combinación de cuadrados adyacentes.

El mapa de cinco variables se presenta en la figura 2-23 y consta de dos mapas de cuatro variables, con variables *A*, *B*, *C*, *D* y *E*. La variable *A* distingue los dos mapas, según se indica en la parte superior del diagrama. El mapa de cuatro variables del lado izquierdo representa los 16 cuadrados donde *A* = 0 y el otro mapa de cuatro variables representa los cuadrados donde *A* = 1. Los minitérminos del 0 al 15 pertenecen a *A* = 0 y los minitérminos del 16 al 31, a *A* = 1. Cada mapa de cuatro variables conserva la adyacencia antes definida cuando se toman por separado. Además, cada cuadrado del mapa *A* = 0 es adyacente al cuadrado correspondiente en el mapa *A* = 1. Por ejemplo, el término mínimo 4 es adyacente al minitérmino 20 y

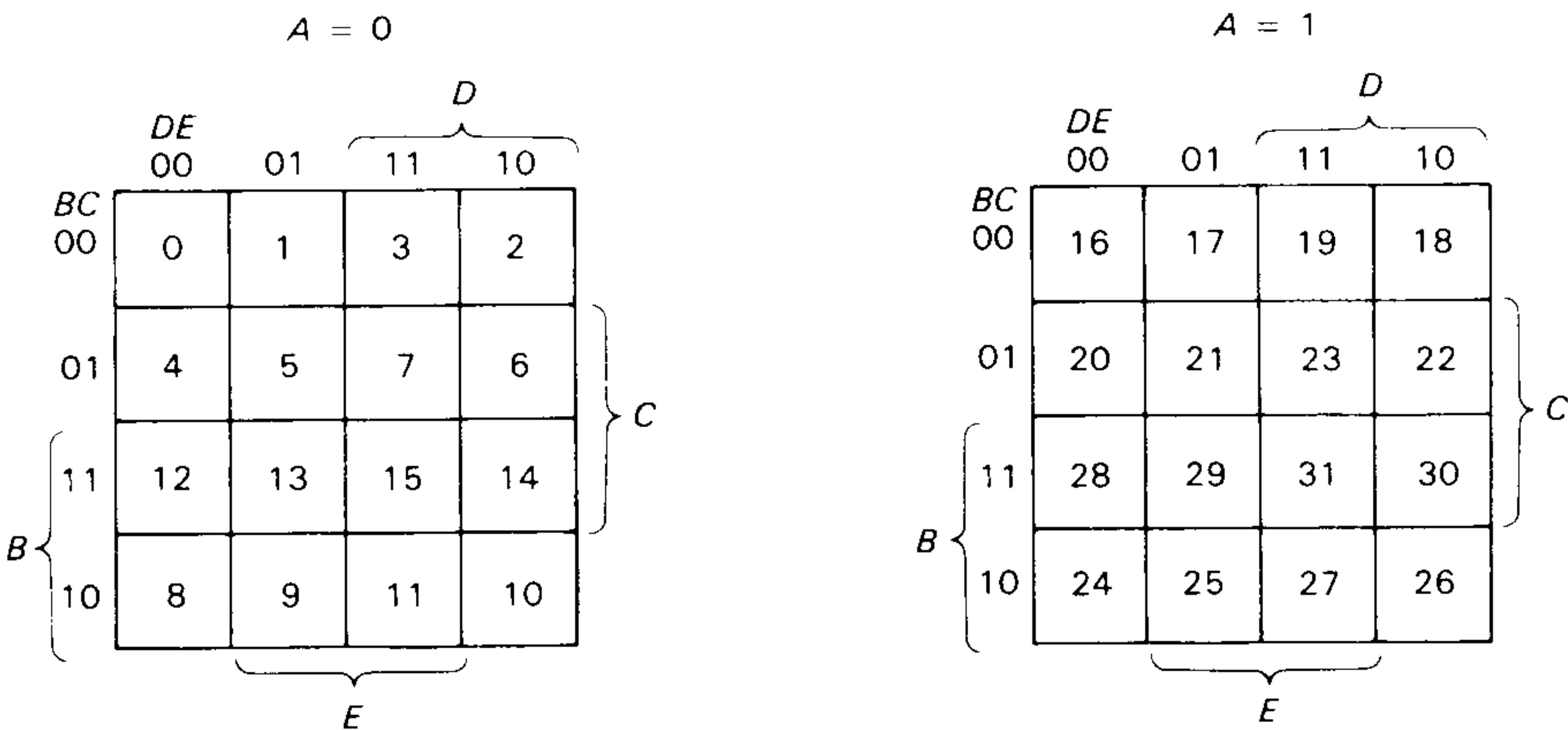


FIGURA 2—23
Mapa de cinco variables

Ejemplo

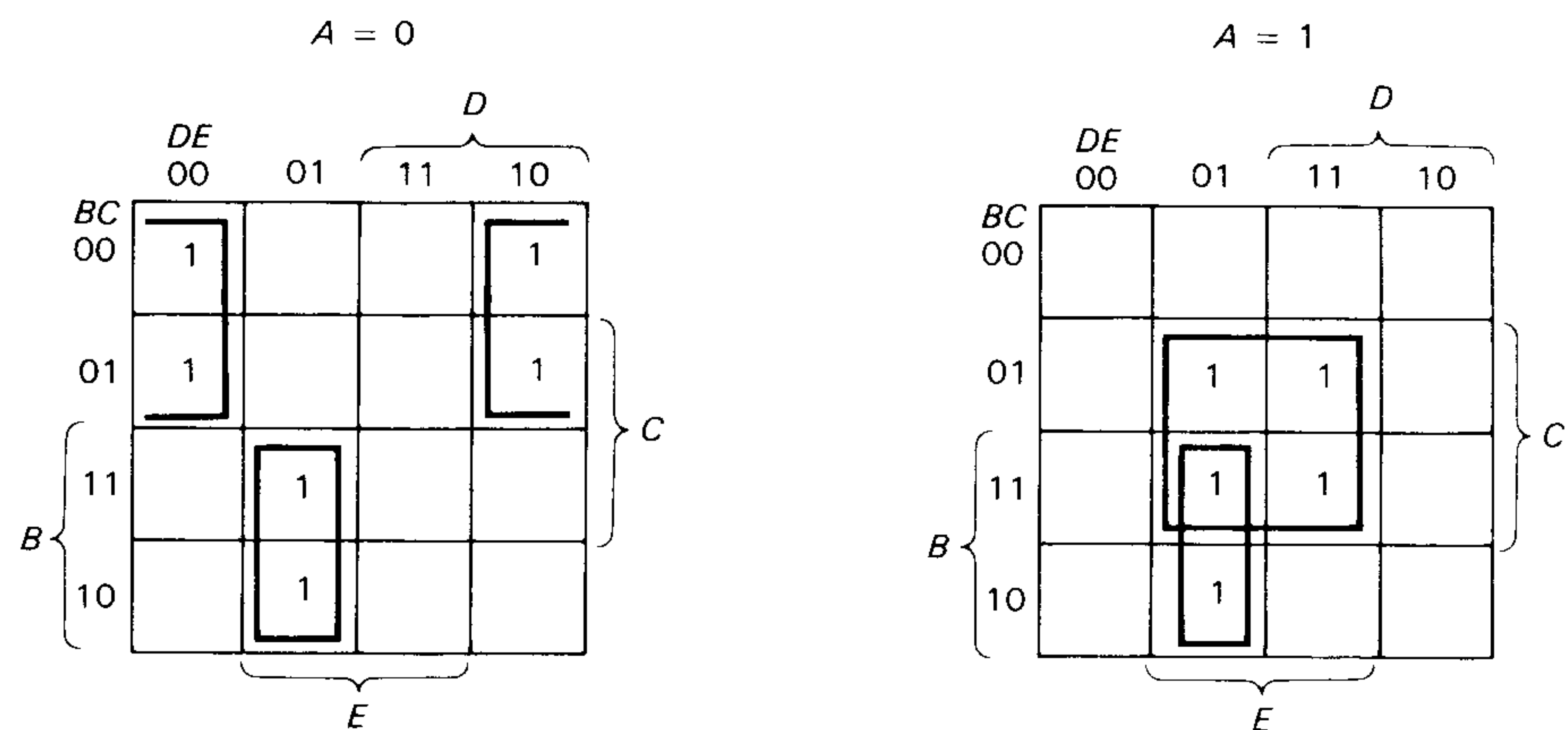


FIGURA 2-24

Mapa del ejemplo 2-8; $F = \overline{A}BE + B\overline{D}E + ACE$

el 15 al 31. La mejor manera de visualizar esta nueva regla para cuadrados adyacentes consiste en considerar que los dos semimapas (o medios mapas) están uno arriba del otro. Dos cuadrados cualquiera que estén uno sobre el otro se consideran adyacentes.

A partir de una inspección, y tomando en cuenta la nueva definición de cuadrados adyacentes, es posible demostrar que 2^k cuadrados adyacentes cualesquiera, para $k = 0, 1, 2, 3, 4$ del mapa de cinco variables representan un término de producto de $5 - k$ literales. Por ejemplo, cuatro cuadrados adyacentes combinan un área del mapa de cinco variables que representa un término de producto de $5 - 2 = 3$ literales.

Siguiendo el procedimiento que se emplea para el mapa de cinco variables, es posible construir un mapa de seis variables con mapas de cuatro variables a fin de obtener los 64 cuadrados que se requieren. Los mapas con seis o más variables necesitan demasiados cuadrados y son poco prácticos de usar. La alternativa es emplear programas de computadora escritos específicamente para facilitar la simplificación de funciones booleanas con un gran número de variables.

Ejemplo 2-8

Simplifíquese la función booleana

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

El mapa de cinco variables de esta función se ilustra en la figura 2-24. Hay seis mini-términos del 0 al 15 que pertenecen a la parte del mapa con $A = 0$. Los otros cinco términos mínimos pertenecen a $A = 1$. Cuatro cuadrados adyacentes en el mapa $A = 0$ se combinan para dar el término de tres literales $\overline{A}BE$. Obsérvese que es necesario incluir \overline{A} con el término porque todos los cuadrados están asociados con $A = 0$. Los dos cuadrados de la columna 01 y los dos últimos renglones son comunes a ambas partes del mapa. Por lo tanto, constituyen cuatro cuadrados adyacentes y producen el término de tres literales $B\overline{D}E$. La variable A no está incluida aquí porque los cuadrados adyacentes pertenecen a $A = 0$ y a $A = 1$. El término ACE se obtiene de los cuatro cuadrados adyacentes que están completamente dentro del mapa $A = 1$. La función simplificada es la suma lógica de los tres términos.

$$F = \overline{A}BE + B\overline{D}E + ACE$$

2-6 COMPUERTAS NAND Y NOR

Como las funciones booleanas se expresan en términos de operaciones AND, OR y de complemento, la ejecución de una función booleana con compuertas AND, OR e inversoras es un procedimiento directo. La posibilidad de construir compuertas con otras operaciones lógicas es de interés práctico. Los factores que se tomarán en consideración cuando se construyan otros tipos de compuertas son la factibilidad y la economía de producir la compuerta con componentes electrónicas, la posibilidad de extender la compuerta a más de dos entradas y la posibilidad de la compuerta de ejecutar funciones booleanas sola o en conjunto con otras compuertas.

Además de las compuertas AND, OR e inversora, hay otras compuertas lógicas en el mercado y se utilizan en forma extensiva en el diseño de circuitos digitales. Los símbolos gráficos y tablas de verdad de ocho compuertas lógicas se presentan en la figura 2-25. Las compuertas se muestran con dos variables binarias de entrada X y Y y una variable binaria de salida F . Se trazan dos símbolos gráficos para cada compuerta. Los símbolos de forma rectangular son recomendados por el *Standard Graphics Symbols for Logic Functions* (IEEE Standard 91-1984) del *Institute of Electrical and Electronics Engineers* (IEEE). Los símbolos diferenciados se han utilizado en el pasado y se consideran como una alternativa al estándar. Los símbolos rectangulares se consideran más convenientes cuando se utilizan con gráficas de computadora. En este libro conservaremos los símbolos de forma diferenciada debido a su uso intenso en la literatura técnica.

Las compuertas AND, OR e inversora ya se definieron antes. El circuito inversor invierte el sentido lógico de una señal binaria para producir la operación de complemento. El círculo pequeño en la salida del símbolo gráfico de un inversor designa el complemento lógico. El símbolo del triángulo por sí solo designa un circuito *buffer*. Un *buffer* no produce ninguna operación lógica en particular, ya que el valor binario de la salida es igual al valor binario de la entrada. Este circuito se utiliza meramente para amplificar la señal eléctrica.

La compuerta NAND es el complemento de la operación AND. Su nombre es abreviatura de Not AND. El símbolo gráfico consta de un símbolo AND seguido de un pequeño círculo. La compuerta NOR (abreviatura de Not OR) es el complemento de la operación OR y se representa por un símbolo gráfico OR seguido de un pequeño círculo. Las compuertas NAND y NOR se utilizan ampliamente como compuertas lógicas estándar y de hecho son más populares que las compuertas AND y OR. Esto se debe a que las compuertas NAND y NOR se construyen fácilmente con circuitos electrónicos y debido a que las funciones booleanas se pueden ejecutar fácilmente con ellas.

La compuerta OR excluyente (XOR) es semejante a la compuerta OR pero excluye la combinación de X y Y que es igual a 1. El símbolo gráfico de la compuerta XOR es semejante al de la compuerta OR salvo por la línea curva adicional del lado de la entrada. La compuerta NOR excluyente es el complemento de la OR excluyente, como lo indica el pequeño círculo en el lado de la salida del símbolo gráfico. La OR excluyente tiene el símbolo especial \oplus que designa su operación.

En lo que resta de esta sección estudiaremos la construcción de circuitos digitales con compuertas NAND y NOR. La compuerta XOR se analizará en la sección que sigue.

Símbolos gráficos

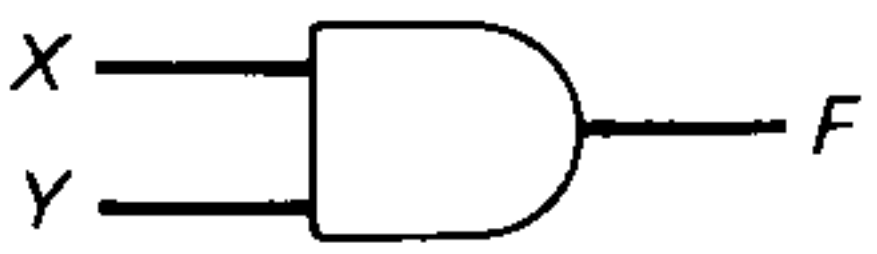
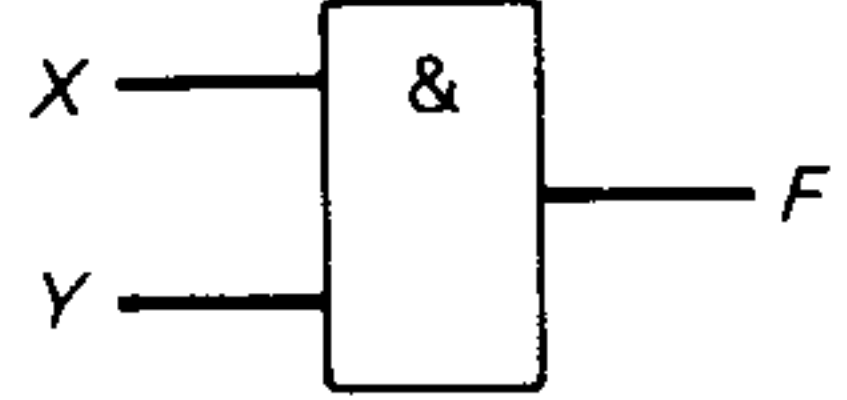
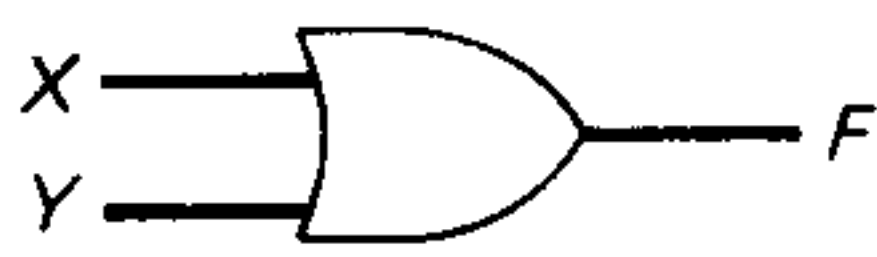
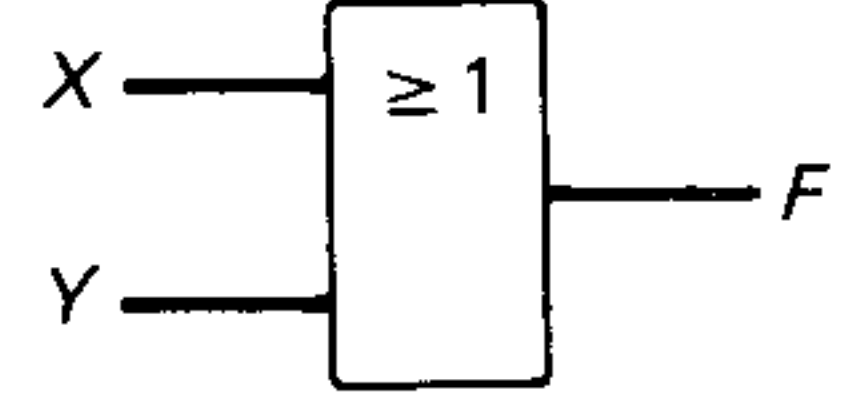
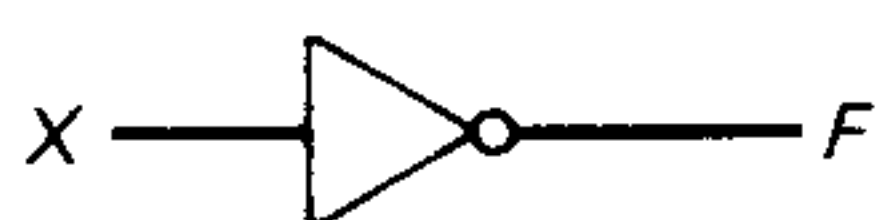
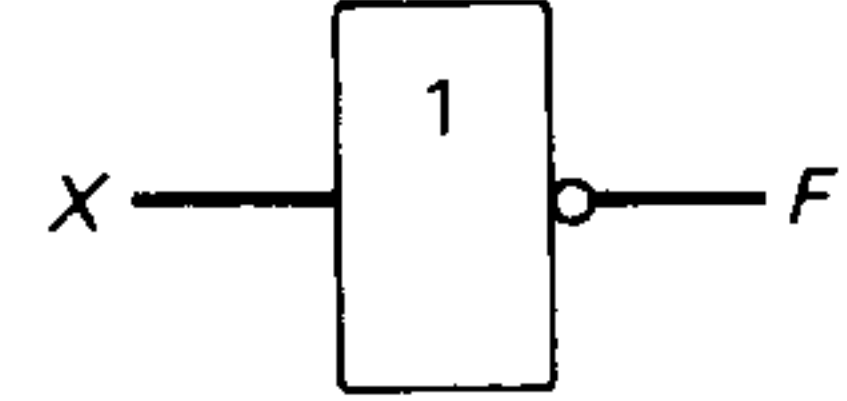
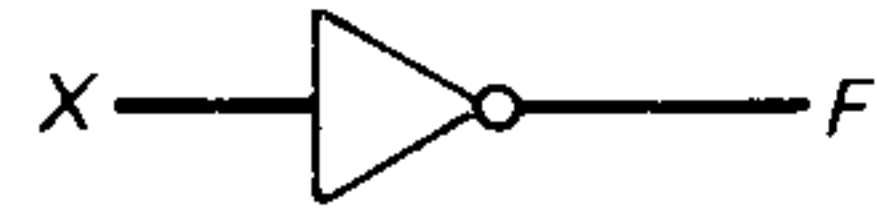
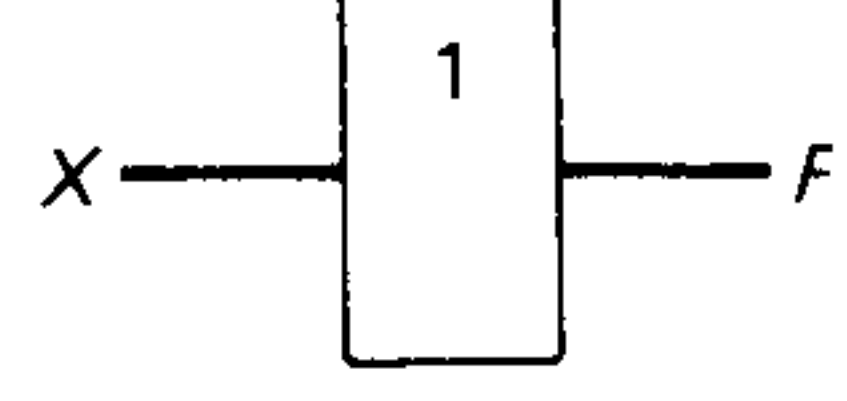
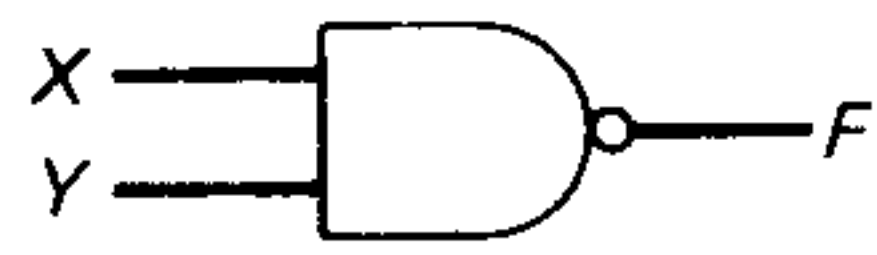
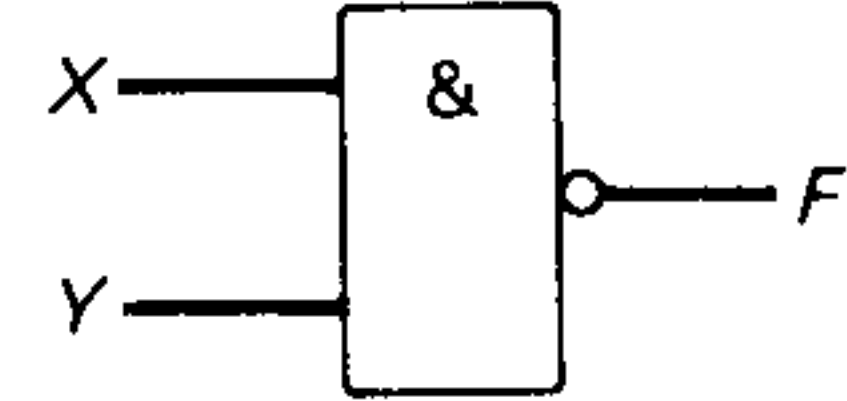
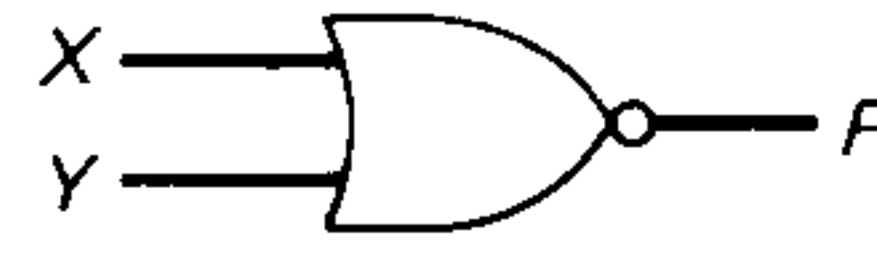
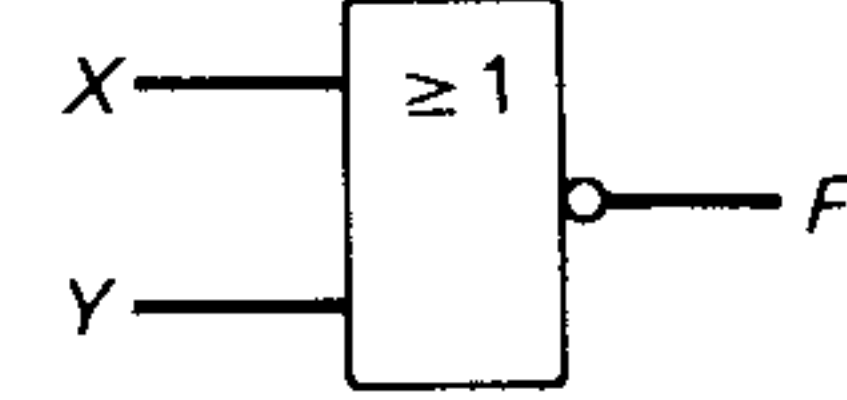

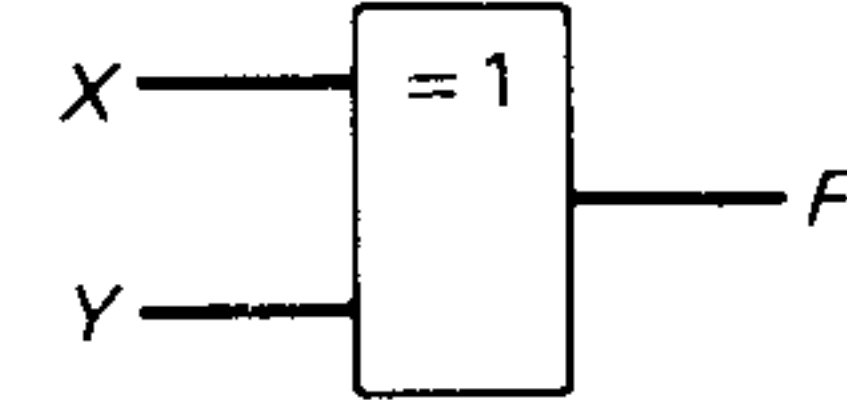
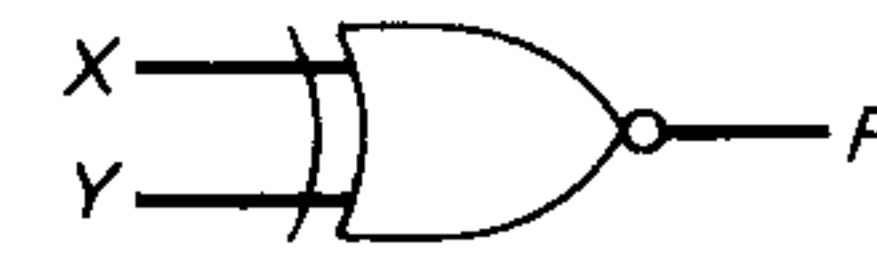
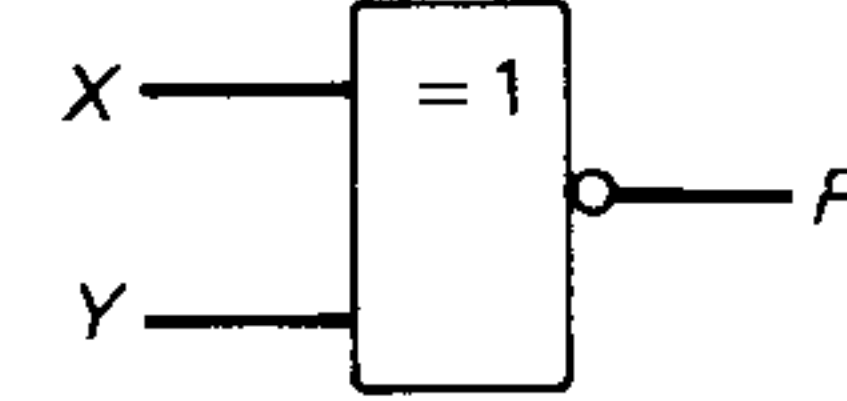
Nombre	Forma diferenciada	Forma rectangular	Ecuación algebraica	Tabla de verdad															
AND			$F = XY$	<table><tr><th>X</th><th>Y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	X	Y	F	0	0	0	0	1	0	1	0	0	1	1	1
X	Y	F																	
0	0	0																	
0	1	0																	
1	0	0																	
1	1	1																	
OR			$F = X + Y$	<table><tr><th>X</th><th>Y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	X	Y	F	0	0	0	0	1	1	1	0	1	1	1	1
X	Y	F																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	1																	
Inversor			$F = \bar{X}$	<table><tr><th>X</th><th>F</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	X	F	0	1	1	0									
X	F																		
0	1																		
1	0																		
Buffer			$F = X$	<table><tr><th>X</th><th>F</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	X	F	0	0	1	1									
X	F																		
0	0																		
1	1																		
NAND			$F = \overline{X \cdot Y}$	<table><tr><th>X</th><th>Y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	X	Y	F	0	0	1	0	1	1	1	0	1	1	1	0
X	Y	F																	
0	0	1																	
0	1	1																	
1	0	1																	
1	1	0																	
NOR			$F = \overline{X + Y}$	<table><tr><th>X</th><th>Y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	X	Y	F	0	0	1	0	1	0	1	0	0	1	1	0
X	Y	F																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	0																	
OR excluyente (XOR)			$F = X\bar{Y} + \bar{X}Y$ $= X \oplus Y$	<table><tr><th>X</th><th>Y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	X	Y	F	0	0	0	0	1	1	1	0	1	1	1	0
X	Y	F																	
0	0	0																	
0	1	1																	
1	0	1																	
1	1	0																	
NOR excluyente (XNOR)			$F = \overline{X\bar{Y} + \bar{X}Y}$ $= X \oplus Y$	<table><tr><th>X</th><th>Y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	X	Y	F	0	0	1	0	1	0	1	0	0	1	1	1
X	Y	F																	
0	0	1																	
0	1	0																	
1	0	0																	
1	1	1																	

FIGURA 2—25

Compuertas de lógica digital

Compuerta NAND

Se dice que la compuerta NAND es una compuerta universal porque con ella se puede diseñar cualquier sistema digital. Para demostrar que cualquier función booleana se puede ejecutar con compuertas NAND, sólo necesitamos mostrar que las operaciones lógicas AND, OR y de complemento pueden obtenerse únicamente con compuertas NAND. Esto se ilustra en la figura 2-26. La operación de complemento se obtiene a partir de una compuerta NAND de una entrada que se conduce exactamente como una compuerta inversora. La operación AND requiere de dos compuertas NAND. La primera produce la operación NAND y la segunda invierte el sentido lógico de la señal. La operación OR se realiza a través de una compuerta NAND con inversores adicionales en cada entrada.

Una manera adecuada de ejecutar una función booleana con compuertas NAND consiste en obtener la función booleana simplificada en términos de operadores booleanos y después convertir la función a lógica NAND. La conversión de una expresión algebraica en NAND a partir de AND, OR y complemento, se puede efectuar a través de técnicas simples de manipulación de circuitos que cambian diagramas de AND y OR por diagramas de NAND.

Para facilitar la conversión a la lógica NAND, conviene definir un símbolo gráfico alternativo para la compuerta. En la figura 2-27 se presentan dos símbolos gráficos equivalentes para la compuerta NAND. El símbolo AND-inversión, como se definió antes, consta de un símbolo gráfico AND seguido de un círculo pequeño. Sin embargo, en su lugar es posible representar una compuerta NAND por un símbolo gráfico OR precedido por un pequeño círculo en cada entrada. El símbolo inversión-OR de la compuerta NAND se apega al teorema de DeMorgan y a la convención de que los círculos pequeños denotan complementación.

Los dos símbolos gráficos son útiles en el análisis y diseño de circuitos NAND. Cuando se combinan ambos símbolos en el mismo diagrama, se dice que el circuito está en *notación mixta*.

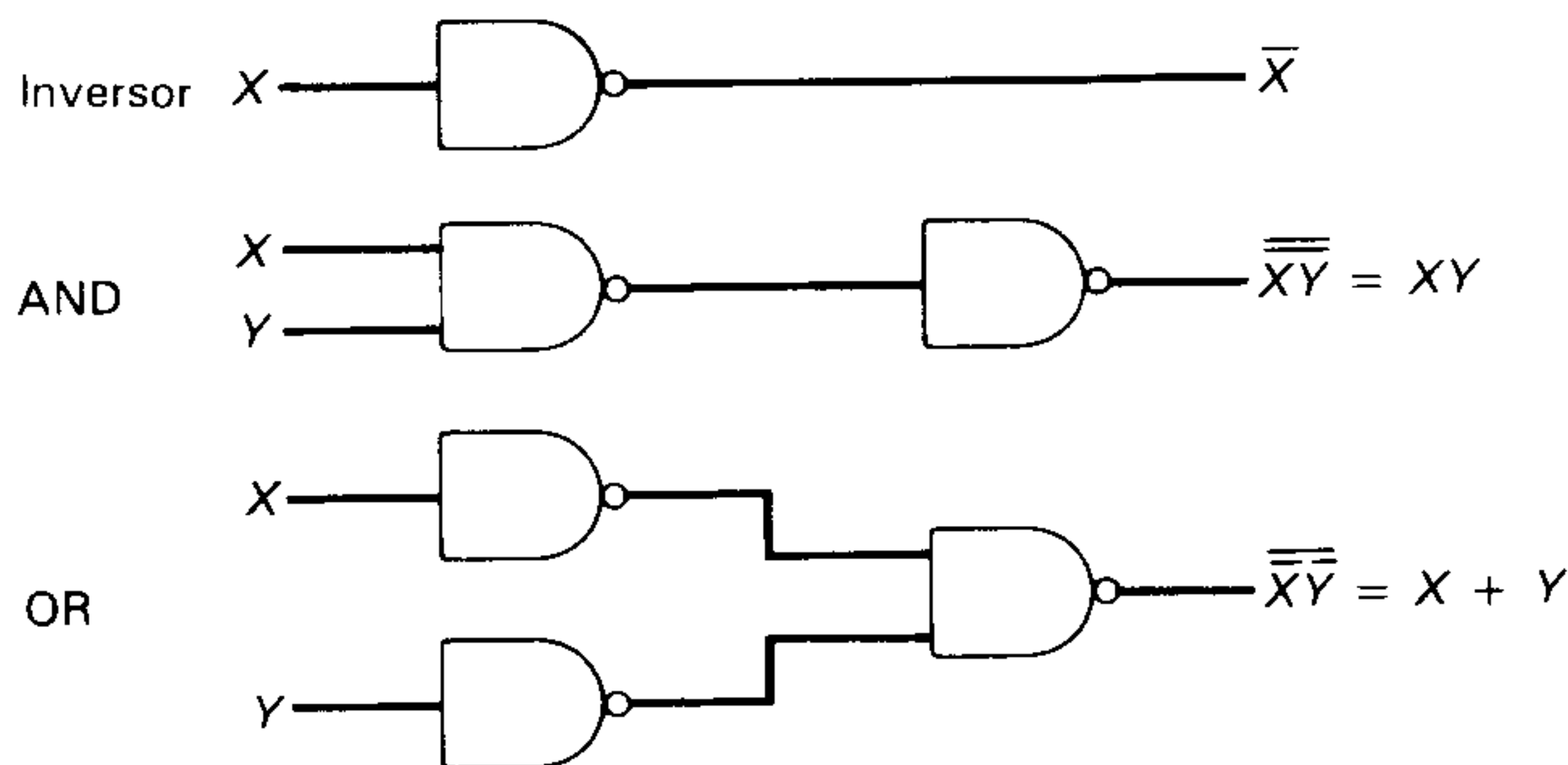


FIGURA 2—26
Operaciones lógicas con compuertas NAND

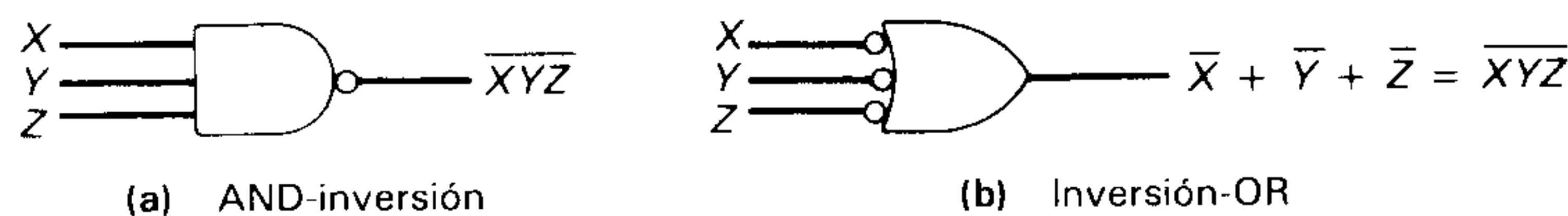
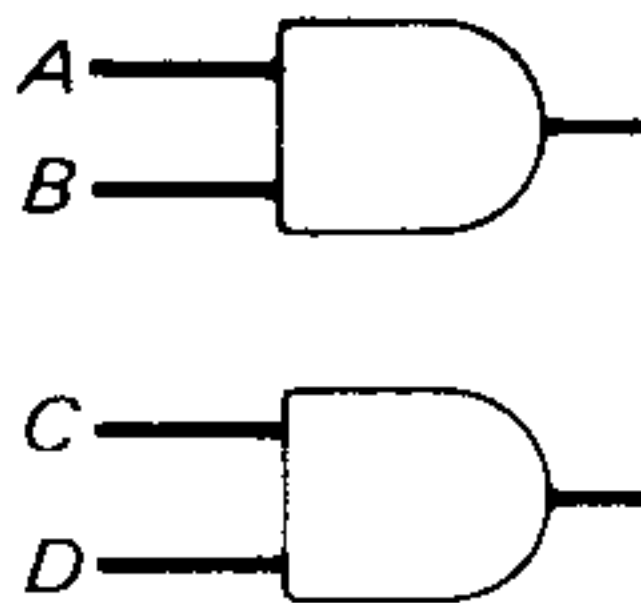


FIGURA 2—27
Dos símbolos gráficos de la compuerta NAND

Ejecución e



Ejemplo 2-9

Ejecución en dos niveles

La ejecución de funciones booleanas con compuertas NAND requiere que la función esté en forma de suma de productos. Para apreciar la relación entre una expresión de suma de productos y su ejecución equivalente con NAND, considérense los diagramas de lógica trazados en la figura 2-28. Los tres diagramas son equivalentes y aplican la función

$$F = AB + CD$$

La función se ejecuta en (a) con compuertas AND y OR. En (b) las compuertas AND se reemplazan por compuertas NAND y la compuerta OR se sustituye por una compuerta NAND con un símbolo gráfico de OR-inversión. Recuerdese que un círculo pequeño denota complementación y que dos círculos en la misma línea representan complementación doble; de modo que ambos se pueden suprimir. La supresión de los círculos pequeños en las compuertas de (b) produce el circuito de (a). Por lo tanto, los dos diagramas aplican la misma función y son equivalentes.

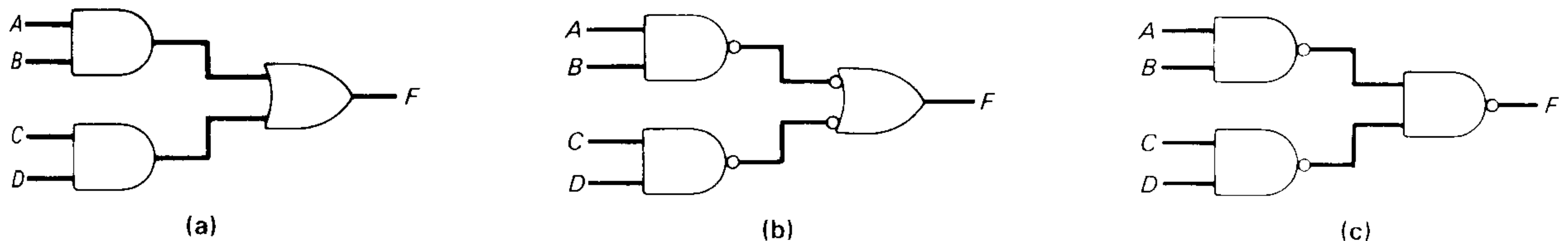


FIGURA 2—28

Tres formas de ejecutar $F = AB + CD$

En la figura 2-28(c), la compuerta de salida NAND se vuelve a representar con el símbolo gráfico AND-inversión. Cuando se trazan diagramas de lógica NAND, el circuito que se presenta en (b) o (c) es aceptable. El de (b) está en notación mixta y representa una relación más directa con la expresión booleana que ejecuta. La aplicación de NAND en la figura 2-28(c) se puede verificar en términos algebraicos. La función que ésta ejecuta se puede convertir sin dificultad a una forma de suma de productos aplicando el teorema de DeMorgan.

$$F = \overline{\overline{AB} \cdot \overline{CD}} = AB + CD$$

Ejemplo 2-9

Ejecútese la siguiente función booleana con compuertas NAND.

$$F(X, Y, Z) = \Sigma m(1, 2, 3, 4, 5, 7)$$

El primer paso consiste en simplificar la función en forma de suma de productos. Esto se hace por medio del mapa de la figura 2-29(a) del cual se obtiene la función simplificada.

$$F = X\bar{Y} + \bar{X}Y + Z$$

La aplicación de NAND en dos niveles se muestra en la figura 2-29(b) en notación mixta. Obsérvese que la entrada Z debe tener una compuerta NAND (inversora) de

una entrada para compensar el círculo pequeño en la compuerta del segundo nivel. Una manera alternativa de trazar el diagrama de lógica se muestra en la figura 2-29(c). Aquí, todas las compuertas NAND se trazan con el mismo símbolo gráfico. El inversor con entrada Z se ha eliminado pero la variable de entrada se complementa y se denota por \bar{Z} .

El procedimiento descrito en el ejemplo anterior indica que una función booleana se puede ejecutar con dos niveles de compuertas NAND. El procedimiento para obtener una función booleana es el siguiente:

1. Simplifíquese la función y exprese en suma de productos.
2. Trácese una compuerta NAND para cada término de producto de la expresión que tenga cuando menos dos literales. Las entradas a cada compuerta NAND son las literales del término. Esto constituye un grupo de compuertas de primer nivel.
3. Trácese una compuerta individual utilizando el símbolo gráfico AND-inversión o inversión-OR en el segundo nivel, donde las entradas provengan de salidas de compuertas del primer nivel.
4. Un término con una literal requiere un inversor en el primer nivel. Sin embargo, si se complementa la literal única se puede conectar directamente a una entrada de la compuerta NAND del segundo nivel.

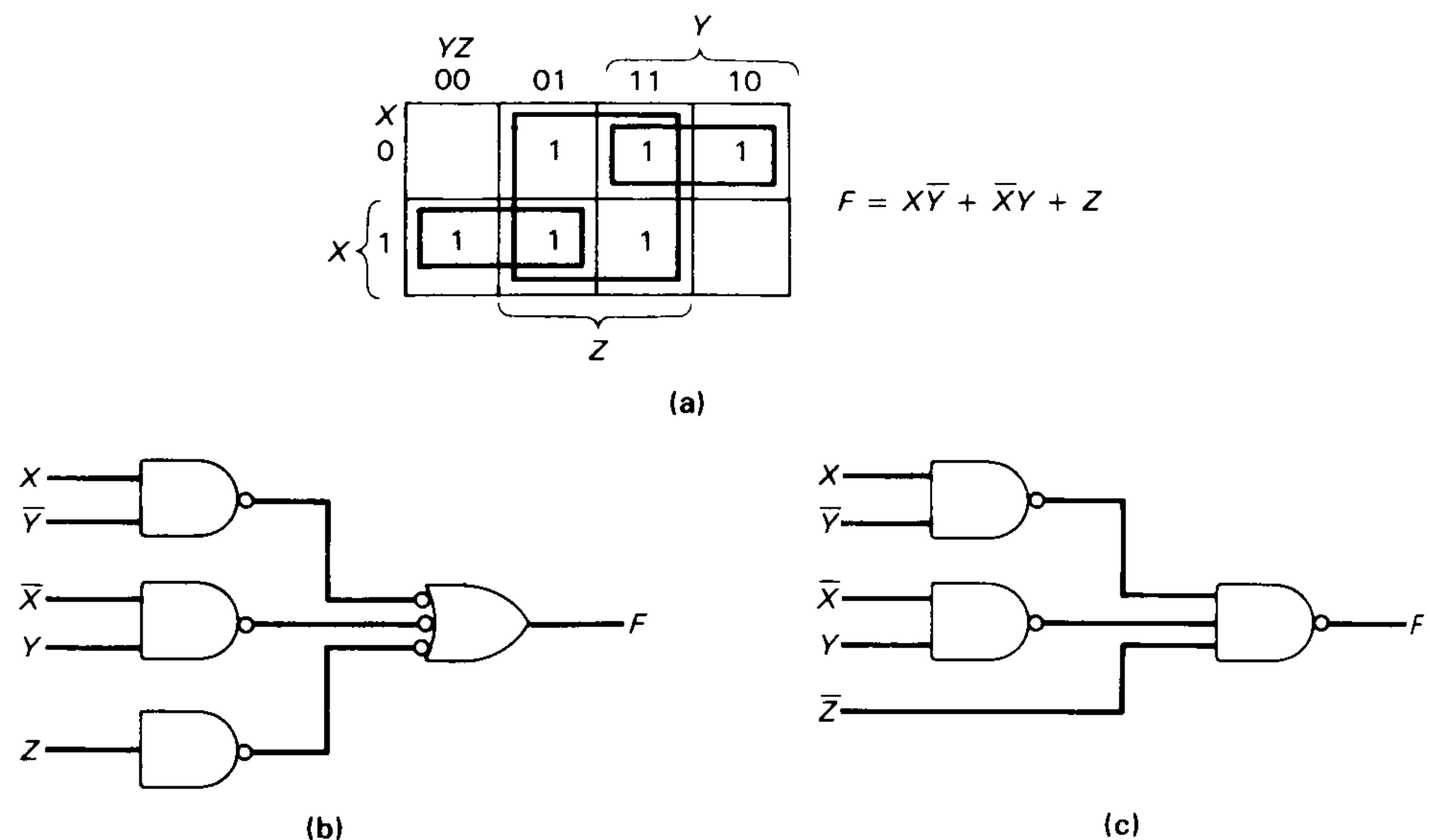


FIGURA 2—29

Solución al ejemplo 2-9

Circuitos NAND de múltiples niveles

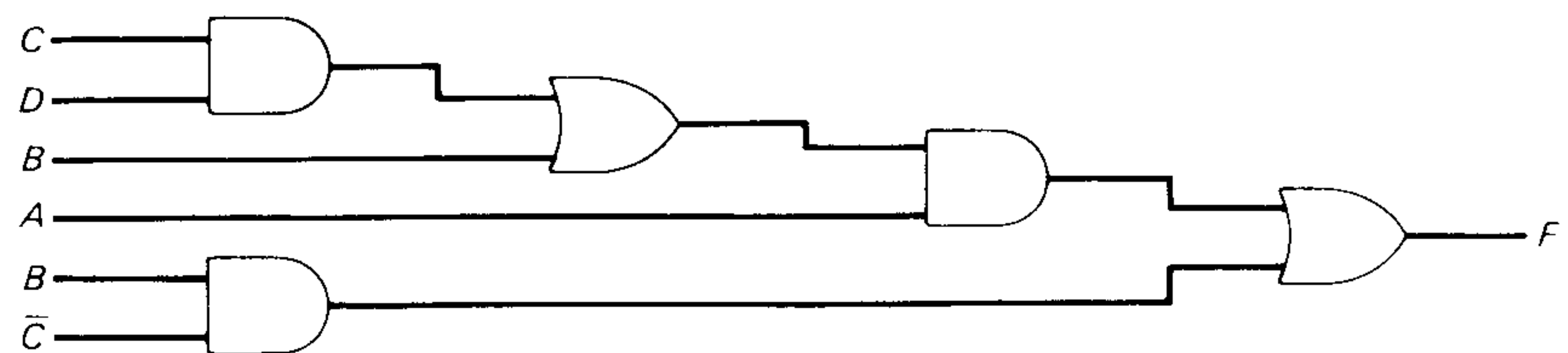
La forma estándar de expresar funciones booleanas da origen a una ejecución en dos niveles. Hay ocasiones en las que el diseño de sistemas digitales da origen a estructuras de compuertas con tres o más niveles. El procedimiento más común en el diseño

de circuitos de múltiples niveles consiste en expresar la función booleana en términos de operaciones AND, OR y de complemento. Después, la función se puede ejecutar directamente con compuertas AND y OR. Luego, de ser necesario, se puede convertir en un circuito de compuertas todas NAND.

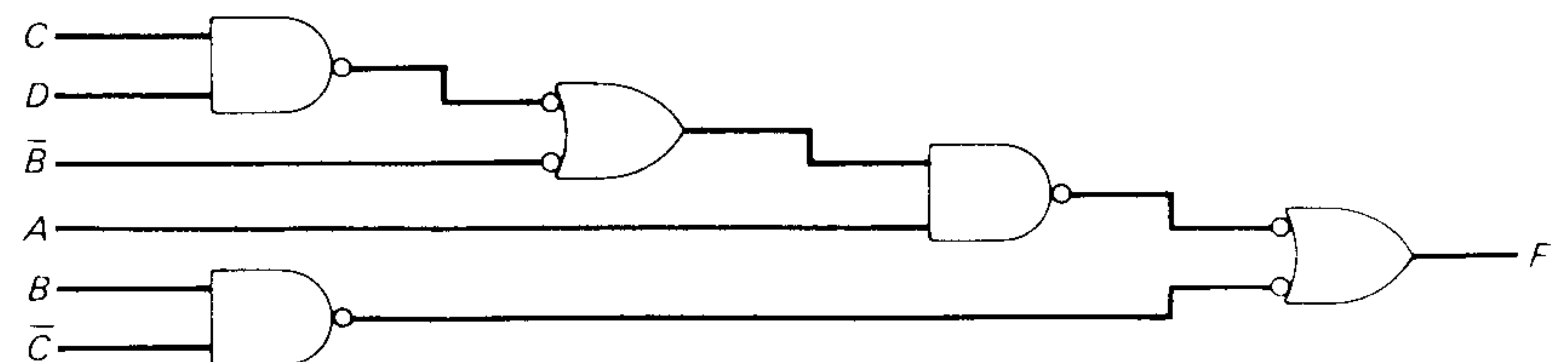
Considérese por ejemplo la función booleana

$$F = A(CD + B) + B\bar{C}$$

Aunque es posible eliminar los paréntesis y reducir la expresión a una forma estándar de suma de productos, optamos por aplicarla como un circuito de múltiples niveles con fines ilustrativos. La construcción de AND-OR se muestra en la figura 2-30(a). Hay cuatro niveles de compuertas en el circuito. El primer nivel tiene dos compuertas AND. El segundo nivel tiene una compuerta OR seguida de una compuerta AND en el tercer nivel y una compuerta OR en el cuarto nivel. Un diagrama de lógica con un patrón de niveles alternativos de compuertas AND y OR se puede convertir fácilmente en un circuito NAND utilizando la notación mixta. Esto se muestra en la figura 2-30(b). El procedimiento consiste en cambiar toda compuerta AND por un símbolo gráfico AND-inversión y toda compuerta OR por un símbolo gráfico inversión-OR. El circuito NAND realiza la misma lógica que el diagrama AND-OR en tanto haya dos círculos pequeños en la misma línea. El círculo pequeño asociado con la entrada R hace que se realice una complementación extra que se debe compensar cambiando la literal de entrada por \bar{B} .



(a) Compuertas AND-OR



(b) Compuertas NAND

FIGURA 2—30

Ejecución de $F = A(CD + B) + B\bar{C}$

El procedimiento general de conversión de un diagrama AND-OR de múltiples niveles en un diagrama de compuertas todas NAND utilizando la notación mixta es como sigue:

1. Conviértanse todas las compuertas AND en compuertas NAND con símbolos gráficos AND-inversión.

2. Conviértanse todas las compuertas OR en compuertas NAND con símbolos gráficos inversión-OR.
3. Revísense todos los círculos pequeños del diagrama. Para todo círculo pequeño que no sea compensado por otro pequeño círculo en la misma línea, insértese un inversor (una compuerta NAND de una entrada) o complementétese la variable de entrada.

Para poner otro ejemplo, considérese la función booleana de múltiples niveles.

$$F = (A\bar{B} + \bar{A}B)(C + \bar{D})$$

La construcción de AND-OR se muestra en la figura 2-31(a) con tres niveles de compuertas. La conversión en NAND con notación mixta se presenta en la parte (b) del diagrama. Los dos círculos pequeños adicionales asociados con las entradas C y \bar{D} causan que estas dos literales se complementen a \bar{C} y D . El pequeño círculo de la

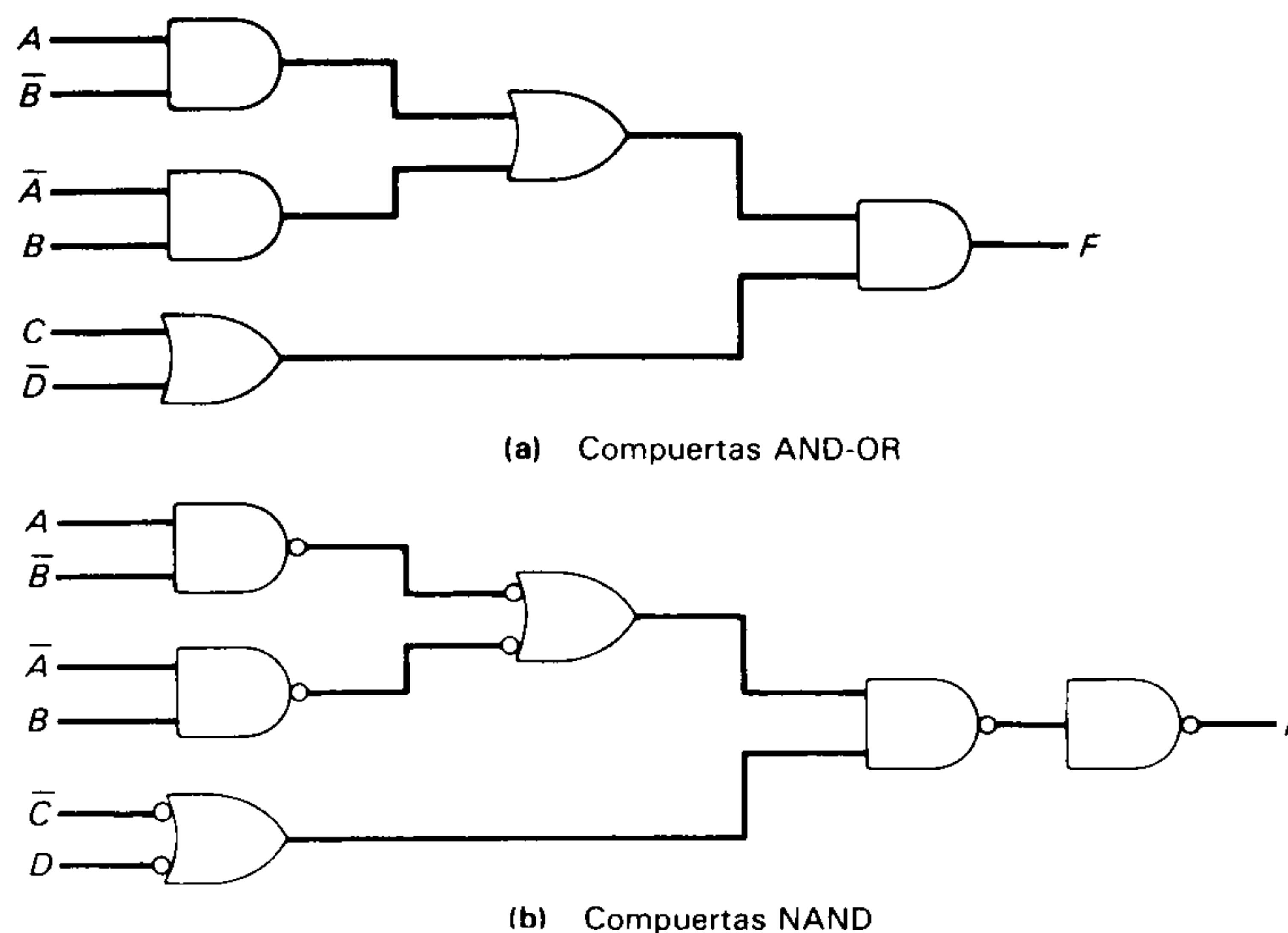


FIGURA 2—31

Ejecución de $F = (A\bar{B} + \bar{A}B)(C + \bar{D})$

compuerta NAND de salida complementa el valor de salida; de modo que necesitamos insertar una compuerta inversora en la salida a fin de volver a complementar la señal y obtener el valor original.

Compuerta NOR

La operación NOR es el dual de la operación NAND. Por lo tanto, todos los procedimientos y reglas de la lógica NOR son los duales de los procedimientos y reglas correspondientes desarrollados para la lógica NAND. La compuerta NOR es otra compuerta universal que se puede utilizar para ejecutar cualquier función booleana. La construcción de las operaciones AND, OR y de complemento con compuertas NOR se ilustra en la figura 2-32. La operación de complemento se obtiene a partir de una compuerta NOR de una entrada. La operación OR requiere dos compuertas NOR y la operación AND se obtiene con una compuerta NOR que tiene inversores en cada entrada.

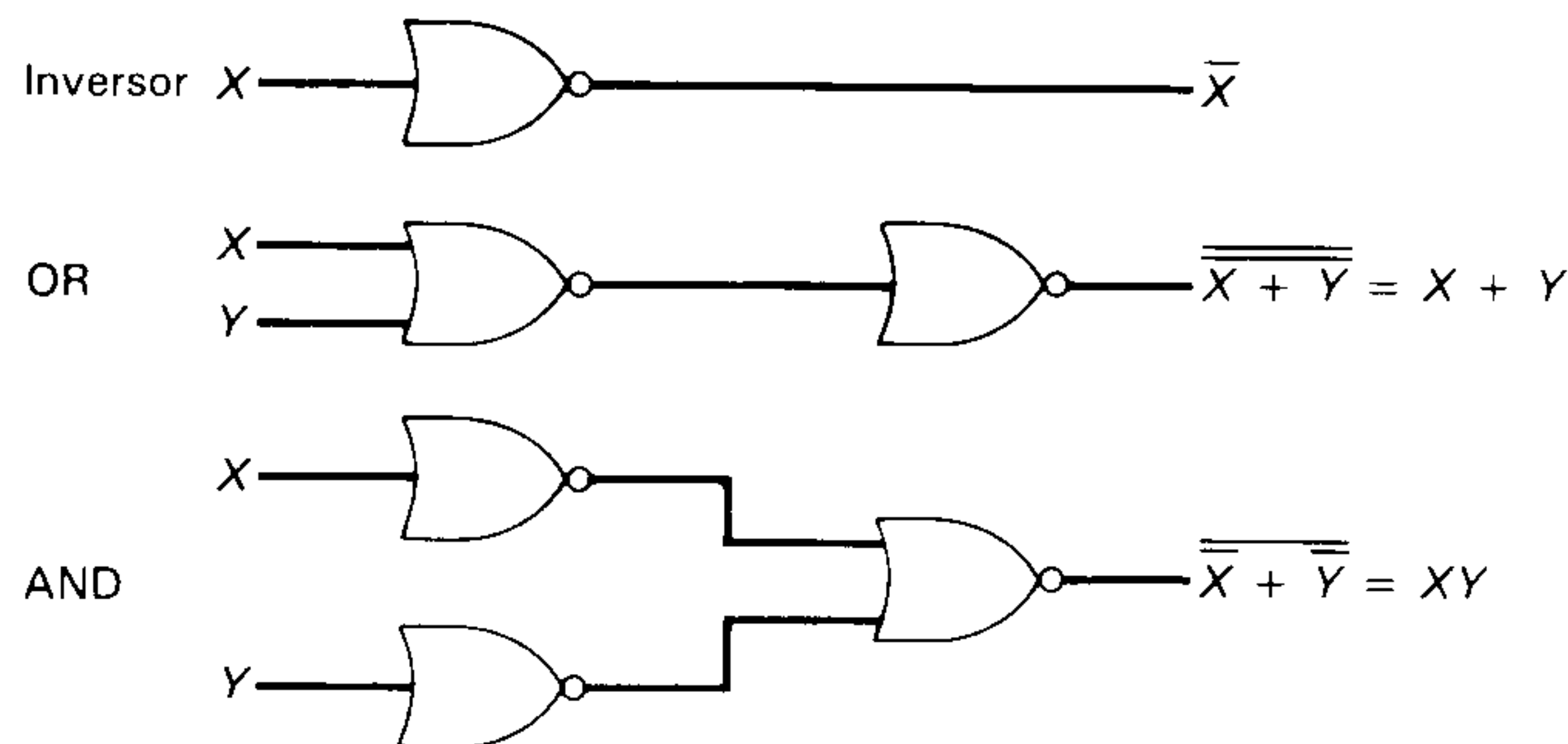


FIGURA 2-32

Operaciones de lógica con compuertas NOR

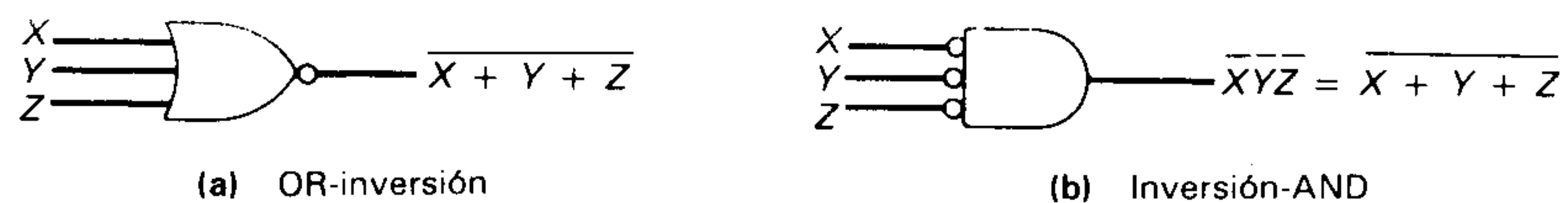


FIGURA 2-33

Dos símbolos gráficos de la compuerta NOR

Los dos símbolos gráficos de la notación mixta se muestran en la figura 2-33. El símbolo OR-inversión define la operación NOR como una OR seguida de un complemento. El símbolo inversión-AND complementa cada entrada y después efectúa una operación AND. Los dos símbolos designan la misma operación NOR y son lógicamente idénticos en virtud del teorema de DeMorgan.

La aplicación en dos niveles con compuertas NOR requiere que la función se simplifique en forma de producto de sumas. Recuerdese que la expresión simplificada de producto de sumas se obtiene a partir del mapa mediante la combinación de los ceros y realizando una operación de complementación. Una expresión de producto de sumas se ejecuta con un primer nivel de compuertas OR que producen los términos de suma seguidos de una compuerta AND de segundo nivel para generar el producto. La transformación del diagrama OR-AND a un diagrama NOR se logra cambiando las compuertas OR por compuertas NOR con símbolos gráficos OR-inversión y la AND por una compuerta NOR con un símbolo gráfico inversión-AND. Un término literal único que vaya en la compuerta del segundo nivel deberá complemen-

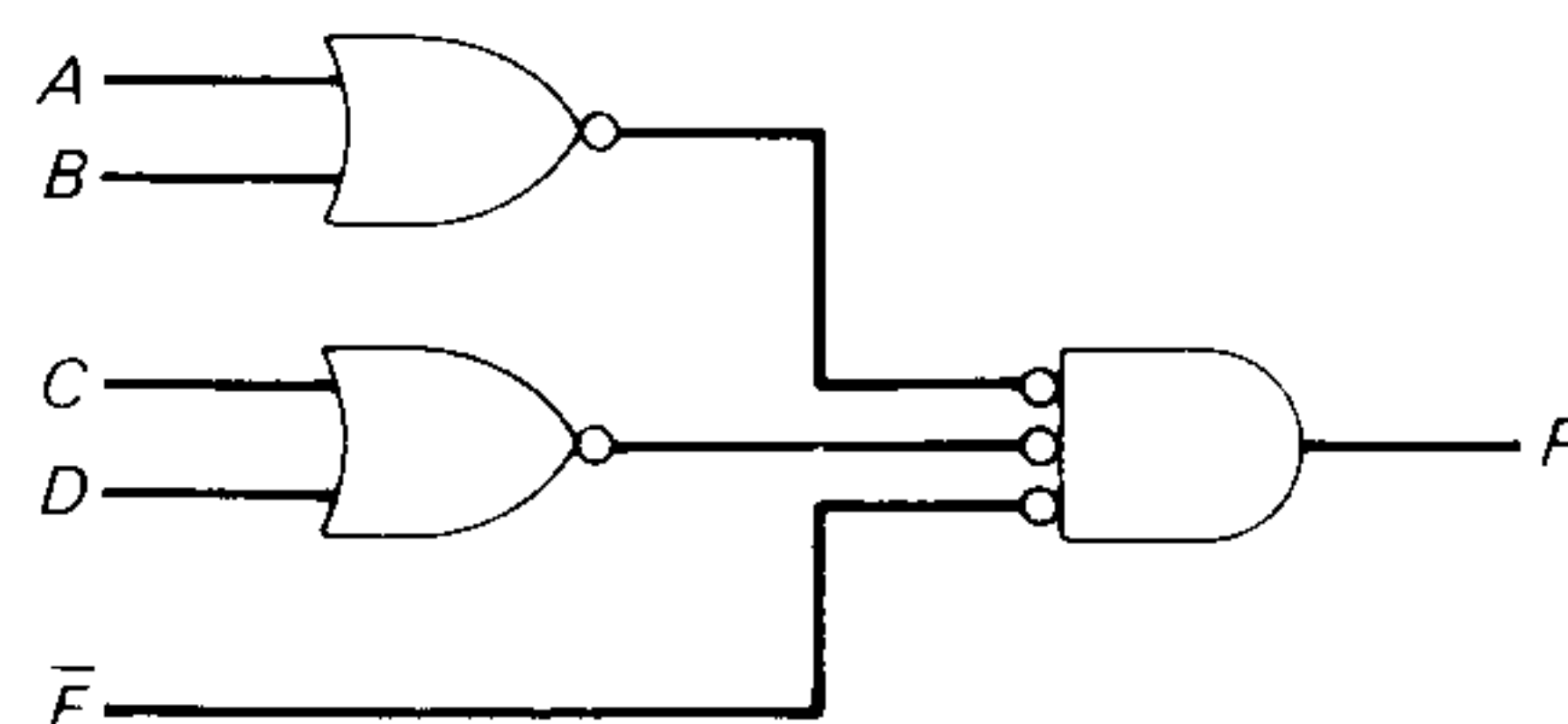


FIGURA 2-34

Ejecución de $F = (A + B)(C + D)\bar{E}$ con compuertas NOR

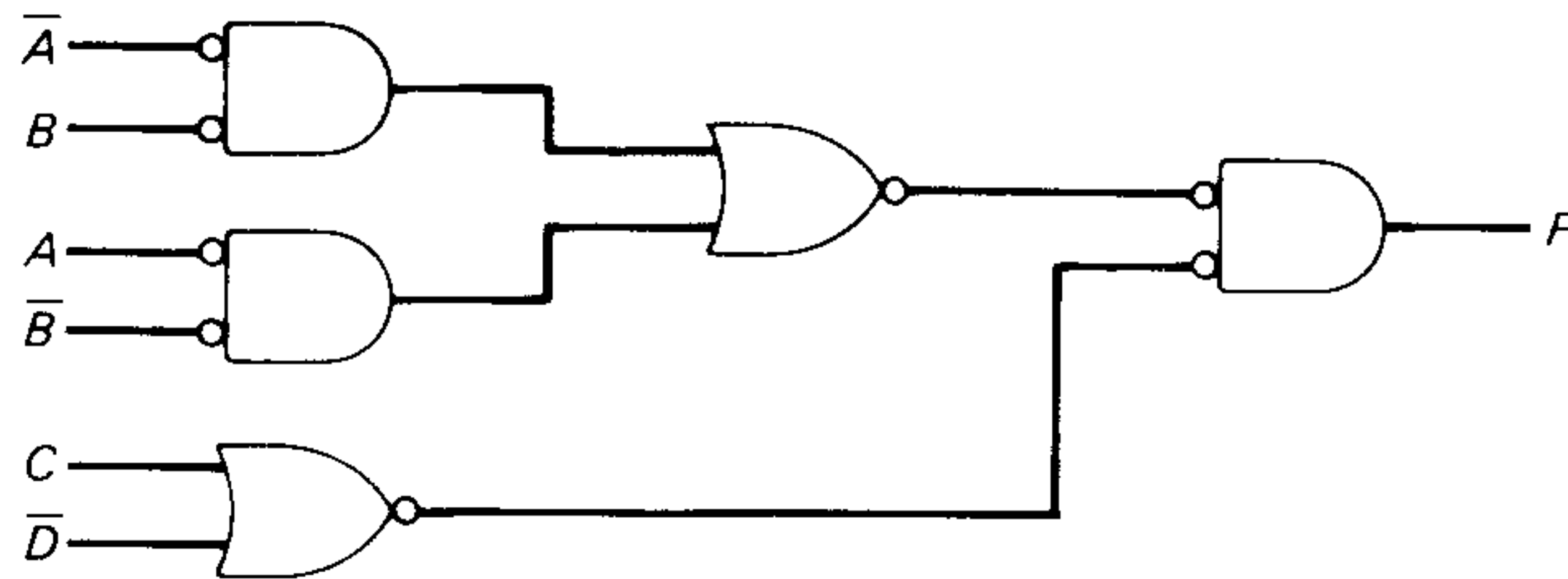


FIGURA 2—35

Ejecución de $F = (A\bar{B} + \bar{A}B)(C + \bar{D})$ con compuertas NOR

tarse. La figura 2-34 muestra la construcción NOR de una función expresada en producto de sumas.

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

El patrón OR-AND se puede detectar fácilmente mediante la supresión de los círculos pequeños que están en la misma línea. La variable E se complementa para compensar el tercer círculo pequeño en la entrada de la compuerta de segundo nivel.

El procedimiento para convertir un diagrama AND-OR de múltiples niveles en un diagrama con todas las compuertas NOR es similar al que se presenta para las compuertas NAND. En el caso de NOR debemos convertir cada compuerta OR en un símbolo OR-inversión y cada compuerta AND en un símbolo inversión-AND. Cualquier círculo pequeño que no sea compensado por otro en la misma línea necesita un inversor o bien la complementación de la literal de entrada.

La transformación del diagrama AND-OR de la figura 2-31(a) en un diagrama NOR se ilustra en la figura 2-35. La función booleana para este circuito es:

$$F = (A\bar{B} + \bar{A}B)(C + \bar{D})$$

El diagrama AND-OR equivalente puede reconocerse en el diagrama NOR suprimiendo todos los círculos pequeños. La supresión de los círculos pequeños en las entradas requiere que se complementen las literales de entrada.

2-7 COMPUERTA OR EXCLUYENTE

La compuerta OR excluyente (XOR), denotada por \oplus , es una operación lógica que desempeña la función

$$X \oplus Y = X\bar{Y} + \bar{X}Y$$

Es igual a 1 si sólo una variable es igual a 1 pero no ambas. La operación NOR excluyente (o exclusive-NOR), que se conoce también como la equivalencia, es el complemento de la OR excluyente y se expresa a través de la función

$$\overline{X \oplus Y} = XY + \bar{X}\bar{Y}$$

Es igual a 1 si X y Y son iguales a 1 o si ambas son iguales a 0. Se puede demostrar que las dos funciones se complementan entre sí. Esto se puede lograr por medio de una tabla de verdad o bien por manipulación algebraica.

$$\overline{X \oplus Y} = \overline{X\bar{Y} + \bar{X}Y} = (\bar{X} + Y)(X + \bar{Y}) = XY + \bar{X}\bar{Y}$$

Comp

Las identidades que siguen se aplican a la operación OR excluyente:

$$\begin{aligned} X \oplus 0 &= X & X \oplus 1 &= \bar{X} \\ X \oplus X &= 0 & X \oplus \bar{X} &= 1 \\ X \oplus \bar{Y} &= \overline{X \oplus Y} & \bar{X} \oplus Y &= \overline{X \oplus Y} \end{aligned}$$

Cualquiera de estas identidades puede demostrarse utilizando una tabla de verdad o reemplazando la operación \oplus por su expresión booleana equivalente. Se puede probar también que la operación OR excluyente es tanto acumulativa como asociativa.

$$A \oplus B = B \oplus A$$

$$(A \oplus B) \oplus C = A \oplus (B \oplus C) = A \oplus B \oplus C$$

Esto quiere decir que las dos entradas a una compuerta OR excluyente se pueden intercambiar sin que se afecte a la operación. Significa también que podemos evaluar una operación OR excluyente de tres variables en cualquier orden y, por este motivo, se pueden expresar tres o más variables sin usar paréntesis. Esto implicaría la posibilidad de utilizar compuertas OR excluyentes con tres o más entradas. Sin embargo, las compuertas OR excluyentes de entradas múltiples son difíciles de fabricar con *hardware*. De hecho, incluso una función de dos entradas suele construirse con otros tipos de compuertas. Una función OR excluyente de dos entradas se construye con compuertas convencionales utilizando dos inversores, dos compuertas AND y una OR. En la figura 2-36 se presenta la construcción con cuatro compuertas NOR. El diagrama NAND de notación mixta realiza la siguiente operación:

$$X(\bar{X} + \bar{Y}) + Y(\bar{X} + \bar{Y}) = X\bar{Y} + \bar{X}Y$$

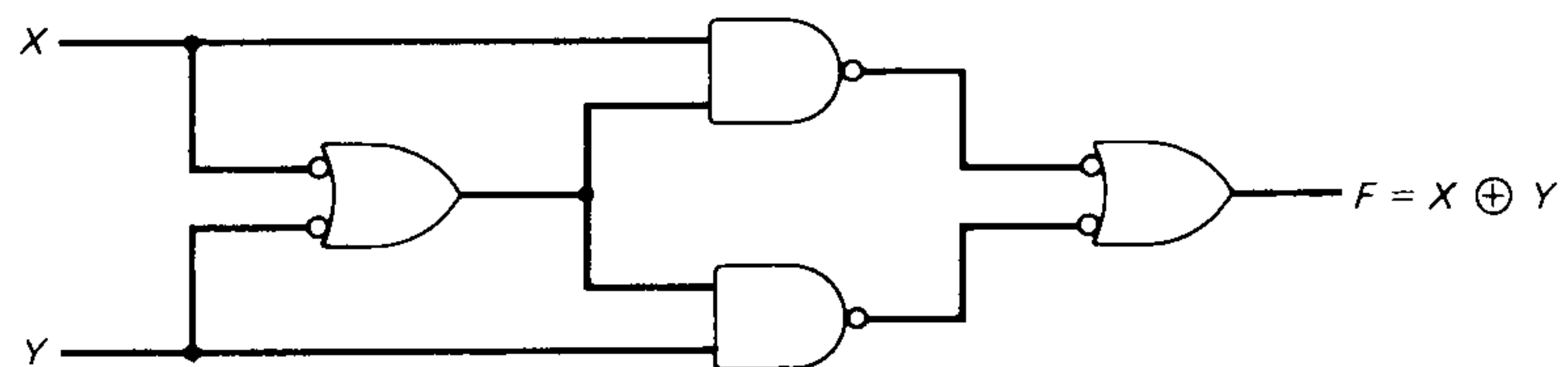


FIGURA 2—36

OR excluyente construida con compuertas NAND

Compuerta de transmisión

La compuerta OR excluyente se puede construir por medio del circuito especial llamado *compuerta de transmisión*. Este tipo de compuerta se tiene a disposición con circuitos electrónicos del tipo CMOS (que se analizarán en la sección que sigue). La operación de la compuerta de transmisión (TG) se ilustra en la figura 2-37. X es la entrada, Y es la salida, y las terminales C y \bar{C} son entradas de control. El circuito funciona como un interruptor electrónico. Cuando $C = 1$ y $\bar{C} = 0$, hay una trayectoria cerrada entre X y Y para que pase la señal. Cuando $C = 0$ y $\bar{C} = 1$, la trayectoria se desconecta y el circuito se comporta como un interruptor abierto. Normalmente, las entradas de control se conectan a través de un inversor, como se muestra en la figura 2-37(c), de manera que C y \bar{C} se complementan entre sí.

Una compuerta OR excluyente se puede construir con dos compuertas de transmisión y dos inversores, como se muestra en la figura 2-38. La entrada *A* controla las trayectorias en las compuertas de transmisión y la entrada *B* ofrece la salida para *Y*. Cuando la entrada *A* es igual a 0, la compuerta de transmisión TG1 se cierra y la salida *Y* es igual a la entrada *B*. Cuando la entrada *A* es igual a 1, TG2 se cierra y la salida *Y* es igual al complemento de la entrada *B*. Esto da origen a la tabla de verdad OR excluyente, según se representa en la tabla de la figura 2-38.

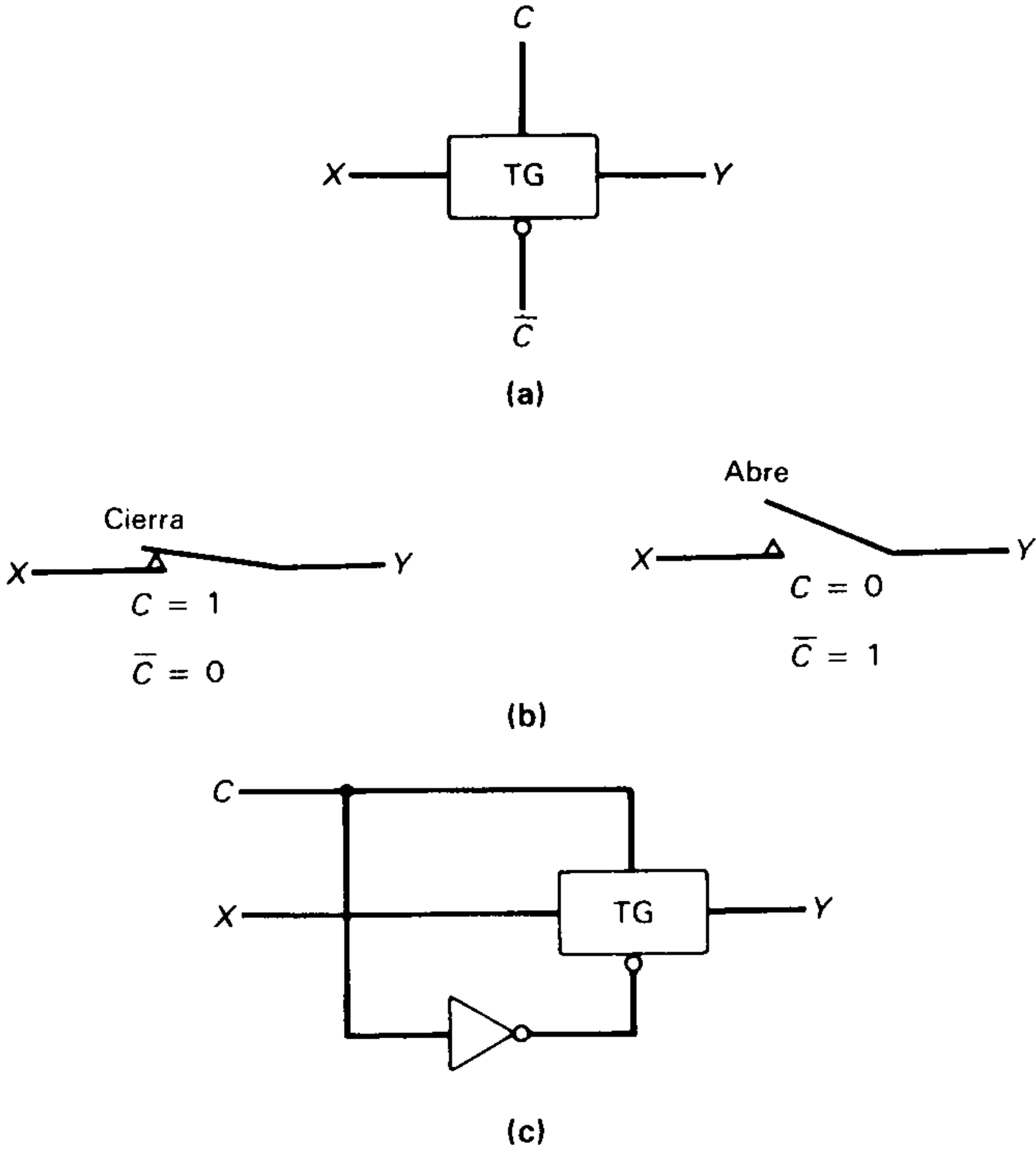


FIGURA 2—37
Compuerta de transmisión (TG)

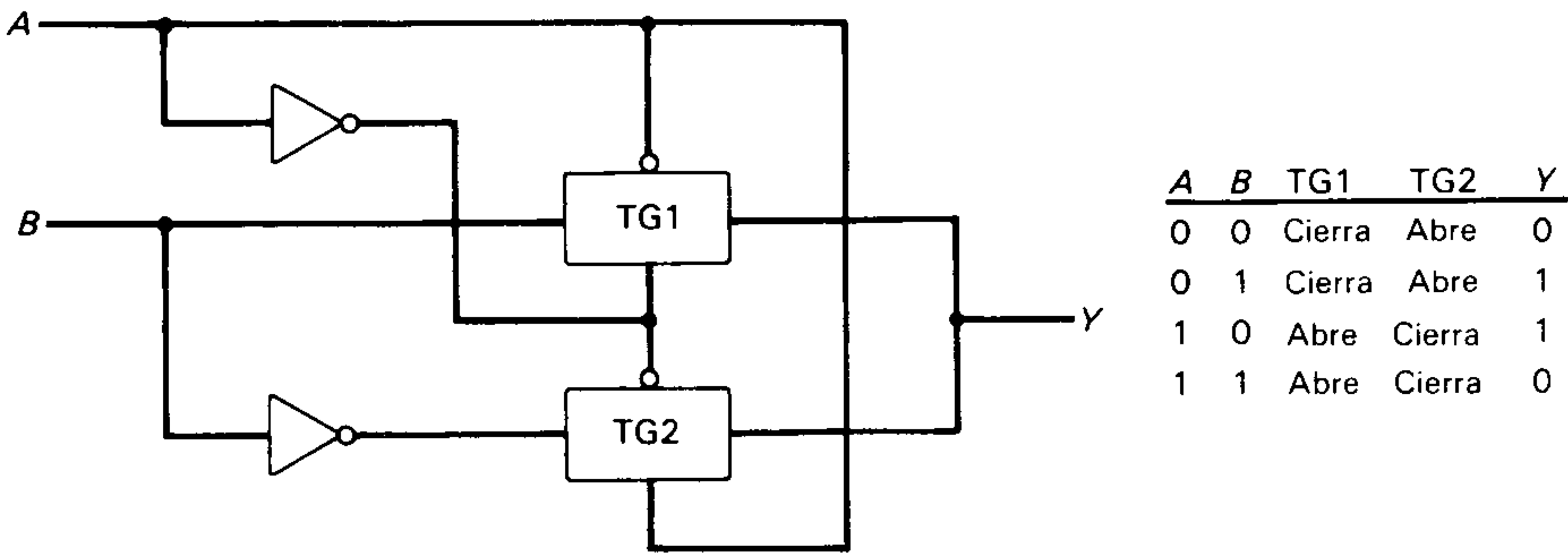


FIGURA 2—38
OR excluyente construida con compuertas de transmisión

Función impar

La operación OR excluyente con tres o más variables se puede convertir en una función booleana ordinaria reemplazando el símbolo \oplus por su expresión booleana equivalente. En particular, el caso de tres variables se puede convertir en una expresión booleana como sigue:

$$\begin{aligned} X \oplus Y \oplus Z &= (X\bar{Y} + \bar{X}Y)\bar{Z} + (XY + \bar{X}\bar{Y})Z \\ &= X\bar{Y}\bar{Z} + \bar{X}Y\bar{Z} + \bar{X}\bar{Y}Z + XYZ \end{aligned}$$

La expresión booleana indica con claridad que la función OR excluyente de tres variables es igual a 1 sólo si una variable es igual a 1, o si las tres variables son iguales a 1. Mientras que en la función de dos variables sólo una de ellas necesita ser igual a 1; con tres o más variables, una variable de número impar debe ser igual a 1. Como consecuencia, la operación OR excluyente de múltiples variables se define como una *función impar*.

La función booleana que se obtiene de la operación OR excluyente de tres variables se expresa como la suma lógica de cuatro minitérminos (o términos mínimos) cuyos valores numéricos binarios son 001, 010, 100 y 111. Cada uno de estos números binarios tiene un número impar de unos (1). Los otros cuatro minitérminos no incluidos en la función son 000, 011, 101 y 110, y tienen un número par de unos. En general, una función OR excluyente de n variables es una función impar que se define como la suma lógica de los $2^n/2$ minitérminos cuyos valores numéricos binarios tienen un número impar de unos.

La definición de función impar puede explicarse con claridad trazándola en un mapa. La figura 2-39(a) muestra el mapa de la función OR excluyente de tres variables. Los cuatro minitérminos de la función están separados a una distancia de una unidad entre sí. La función impar se identifica de los cuatro términos mínimos cuyos valores binarios tienen un número impar de unos. El caso de las cuatro variables se muestra en la figura 2-39(b). Los ocho minitérminos marcados con unos en el mapa constituyen la función impar. Nótese el patrón característico de la distancia entre los unos en el mapa. Debe mencionarse que los minitérminos no marcados con unos en el mapa tienen un número par de unos y constituyen el complemento de la función impar. Una función impar se ejecuta por medio de compuertas OR excluyente de dos entradas, según se ve en la figura 2-40. El complemento de una función impar se obtiene reemplazando la compuerta de salida por una compuerta NOR excluyente.

Generación y verificación de la paridad

Las funciones OR excluyente son muy útiles en sistemas que requieren códigos de detección y corrección de errores. Como se vio en la sección 1-5, un bit de paridad es un esquema para detectar errores durante la transmisión de información binaria. Un bit de paridad es un bit extra incluido en un mensaje binario para hacer que el número de unos sea impar o par. El mensaje, incluyendo el bit de paridad, se transmite y después se verifica en el extremo receptor para detectar errores. Hay un error si la paridad verificada no corresponde a la transmitida. El circuito que genera el bit de paridad en el extremo transmisor recibe el nombre de *generador de paridad*. El

tas de trans-
a A controla
a salida para
se cierra y la
cierra y la sa-
la de verdad

G2	Y
bre	0
bre	1
erra	1
erra	0

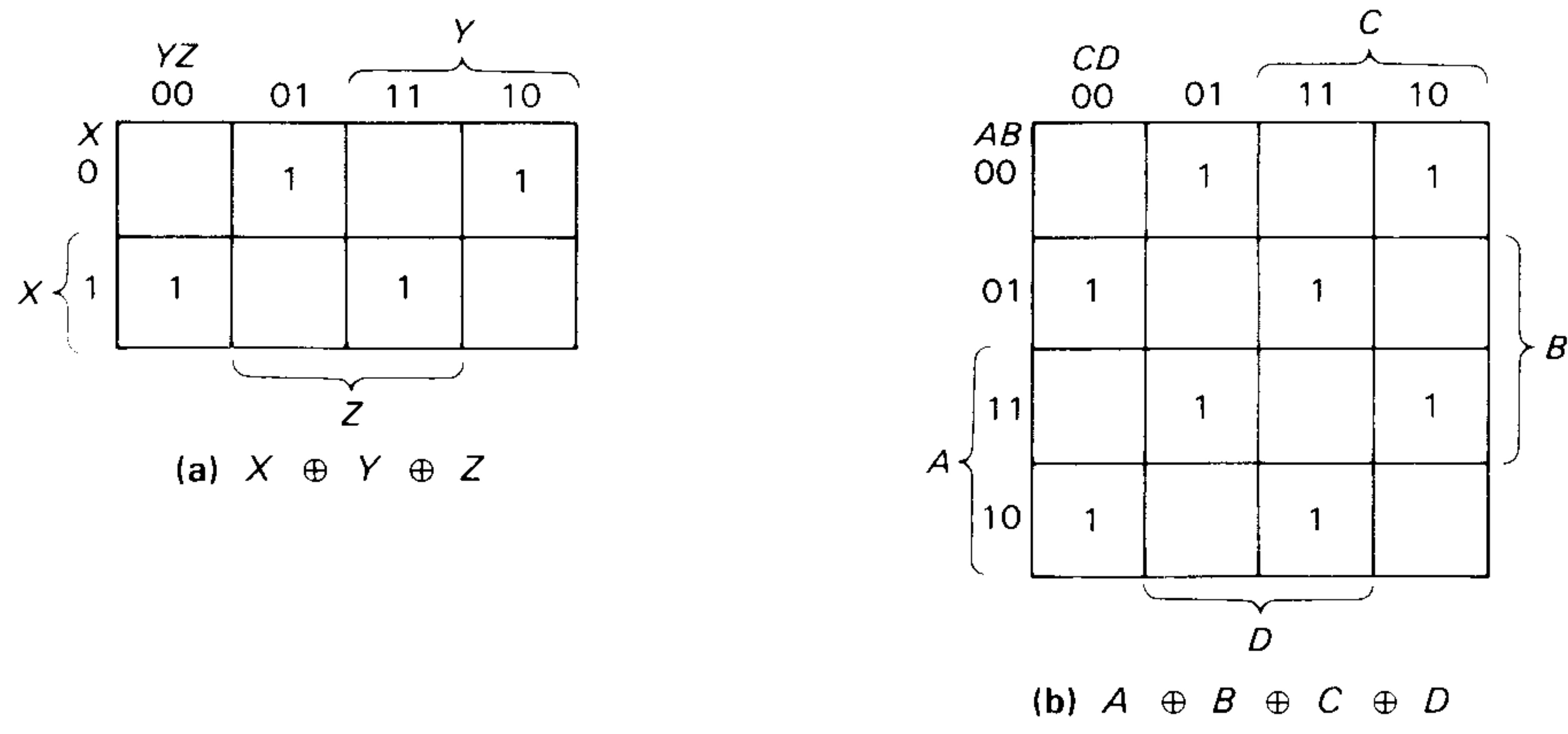


FIGURA 2—39
Mapas de OR excluyente de múltiples variables

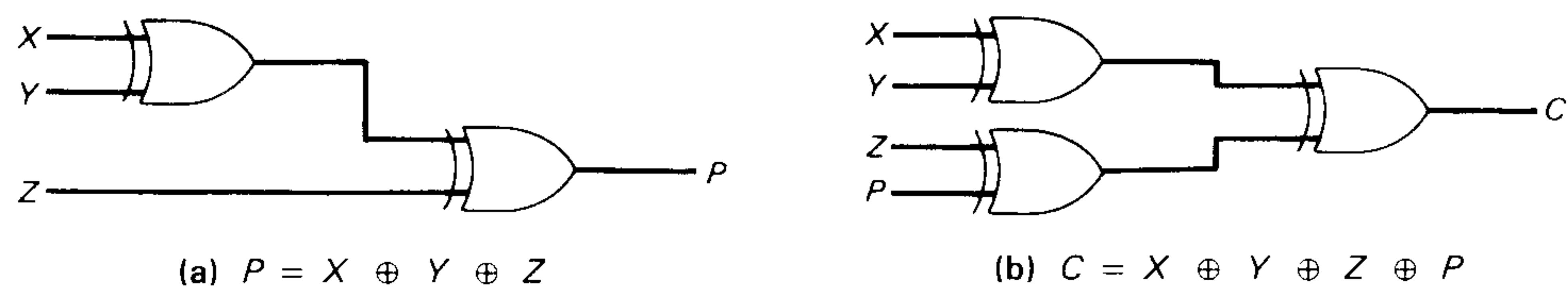


FIGURA 2—40
Funciones OR excluyente de múltiples entradas

circuito que revisa la paridad en el extremo receptor se denomina *verificador de paridad*.

Para poner un ejemplo, considérese un mensaje de tres bits que se transmitirá con un bit de paridad par. La tabla 2-8 muestra la tabla de verdad del generador de paridad. Los bits X , Y y Z constituyen el mensaje y son las entradas al circuito. El bit de paridad P es la salida. En el caso de la paridad par, el bit P debe generarse para hacer que el número total de unos sea par (incluyendo a P). En la tabla de verdad se ve que P constituye una función impar porque es igual a 1 para los minitérminos cuyos valores numéricos tienen un número impar de unos. Por lo tanto, P se puede expresar como una función OR excluyente de tres variables.

$$P = X \oplus Y \oplus Z$$

El diagrama de lógica del generador de paridad se muestra en la figura 2-40(a). Los tres bits del mensaje, junto con el de paridad, se transmiten a su destino, donde se aplican a un circuito verificador de la paridad para detectar posibles errores en la transmisión. Como la información se transmitió con paridad par, los cuatro bits recibidos deben tener un número par de unos. Ocurre un error si los cuatro bits recibidos tienen un número impar de unos, lo que indica que cuando menos un bit ha cambiado de valor durante la transmisión. La salida del verificador de paridad, denotada por C , será igual a 1 si ocurre un error, es decir, si los cuatro bits recibidos tienen un número impar de unos. Esta es, por definición, una función impar y se puede ejecutar con compuertas OR excluyente.

$$C = X \oplus Y \oplus Z \oplus P$$

TABLA 2—8
Tabla de verdad del generador de paridad par

Mensaje de tres bits			Bit de paridad <i>P</i>
<i>X</i>	<i>Y</i>	<i>Z</i>	
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

El diagrama de lógica del verificador de paridad se ilustra en la figura 2-40(b). Es evidente en el ejemplo anterior que los circuitos de generación y verificación de la paridad tienen siempre una función de salida que incluye a la mitad de los mini-
términos cuyos valores numéricos tienen un número impar o par de unos. Como consecuencia, se pueden construir con compuertas OR excluyente. Una función con un número par de unos es el complemento de una función impar. Esta se ejecuta con compuertas OR excluyente, salvo que la compuerta asociada con la salida debe ser una NOR excluyente para ofrecer la complementación.

2-8 CIRCUITOS INTEGRADOS

Los circuitos digitales se construyen con circuitos integrados. Un circuito integrado (que se abrevia IC) es un pequeño cristal de silicón semiconductor, llamado *chip* (pastilla), que contiene los componentes electrónicos de las compuertas digitales. Las diversas compuertas están interconectadas dentro del chip para formar el circuito que se requiere. El chip está montado en un recipiente de cerámica o de plástico, con las conexiones soldadas a terminales externas que forman el circuito integrado. El número de terminales puede variar de 14 en un paquete IC pequeño a 64 o más en un paquete mayor. Cada IC tiene una designación numérica impresa en la superficie del paquete para su identificación. Cada fabricante publica un libro o catálogo con la descripción exacta y toda la información necesaria acerca de los IC que manufactura.

El tamaño del paquete de IC es muy pequeño. Por ejemplo, dentro de un paquete de IC de 14 terminales hay cuatro compuertas AND, con dimensiones de 20×8×3 milímetros. Un microprocesador entero está contenido en el interior de un paquete de IC de 40 terminales con dimensiones de 50×15×4 milímetros.

Niveles de integración

A medida que ha mejorado la tecnología de los circuitos integrados, el número de compuertas que se pueden colocar en un solo chip de silicón ha aumentado en forma considerable. La diferencia entre los chips que tienen pocas compuertas internas y los que tienen cientos o miles de compuertas, se define casi siempre por costumbre

dando a un paquete la designación de dispositivo de integración a pequeña, mediana o gran escala.

Los dispositivos de *integración a pequeña escala* (SSI) contienen varias compuertas independientes en un solo paquete. Las entradas y salidas de las compuertas se conectan directamente a las terminales del paquete. El número de compuertas suele ser menor de 10 y está limitado por el número de terminales disponibles en el IC.

Los dispositivos de *integración a mediana escala* (MSI) tienen una complejidad de aproximadamente 10 a 100 compuertas en un solo paquete. Por lo general realizan funciones digitales elementales específicas, que sirven como decodificadores, sumadores y registros. Las funciones digitales MSI se presentan en los capítulos 3 y 5.

Los dispositivos de *integración a gran escala* (LSI) contienen entre 100 y 2,000 compuertas en un paquete. Incluyen sistemas digitales como procesadores, chips de memoria y módulos programables. Los sistemas de lógica LSI se presentan en los capítulos 6 y 8.

Los dispositivos de *integración a muy grande escala* (VLSI) contienen miles de compuertas dentro de un solo paquete. Algunos ejemplos de ellos son los grandes arreglos de memoria y los chips complejos de microcomputadora. Debido a su tamaño compacto y bajo costo, los dispositivos VLSI han revolucionado la tecnología de diseño de los sistemas de computación, dando al diseñador los recursos para generar estructuras que antes no resultaban económicas.

Familias de lógica digital

Los circuitos integrados digitales se clasifican no sólo por su operación lógica sino también por la tecnología específica de circuitos a la cual pertenecen. La tecnología de circuitos se denomina familia de lógica digital. Cada familia de lógica tiene un circuito electrónico básico en el cual se basa el desarrollo de circuitos y funciones digitales más complejos. El circuito básico de cada tecnología es una compuerta NAND, una OR o una inversora. Los componentes electrónicos que se usan en la construcción del circuito básico suelen servir para dar nombre a la tecnología aplicada. Han aparecido en el mercado muchas familias de circuitos integrados de lógica diferente. Las que siguen son las más populares.

TTL	Lógica transistor-transistor
ECL	Lógica acoplado al emisor
MOS	Semiconductor de óxido metálico
CMOS	Semiconductor de óxido metálico complementario.

TTL Es una familia de lógica ampliamente utilizada que ha estado en operación por algún tiempo y se considera como estándar. ECL tiene una ventaja en sistemas que requieren operación de alta velocidad. MOS es adecuada para circuitos que necesitan alta densidad de componentes y CMOS se prefiere en sistemas que requieren un bajo nivel de consumo de energía.

La familia de lógica transistor-transistor evolucionó de una tecnología previa que empleaba diodos y transistores para la compuerta NAND básica. Esta tecnología fue llamada DTL por lógica diodo-transistor. Después, los diodos fueron reemplazados por transistores para mejorar la operación del circuito y el nombre de la familia se cambió por TTL. Esta es la razón para mencionar la palabra transistor dos veces.

Existen algunas variaciones de la familia TTL además de TTL estándar, como TTL de alta velocidad, TTL de bajo consumo de energía, TTL Schottky, TTL Schottky de bajo consumo de energía y TTL Schottky avanzada de bajo consumo de energía. El voltaje de suministro de energía de los circuitos TTL es de 5 voltios y los dos niveles de lógica son aproximadamente de 0 y 3.5 voltios.

La familia de lógica acoplada al emisor (ECL) proporciona los circuitos digitales de más alta velocidad en forma integrada. ECL se utiliza en sistemas como supercomputadoras y procesadores de señales donde la alta velocidad es fundamental. Los transistores de compuertas ECL operan en un estado no saturado, condición que hace posible que se obtengan demoras en la propagación de uno a dos nanosegundos.

El semiconductor de óxido metálico (MOS) es un transistor unipolar que depende del flujo de sólo un tipo de portador que pueden ser electrones (canal n) u orificios (canal p). Contrasta con el transistor bipolar que se utiliza en compuertas TTL y ECL, donde ambos portadores existen durante la operación normal. La tecnología MOS de canal p se conoce como PMOS y la de canal n como NMOS. NMOS es la que se utiliza comúnmente en circuitos con un solo tipo de transistor MOS. La tecnología MOS complementaria (CMOS) utiliza un transistor PMOS y uno NMOS, conectados en forma complementaria en todos los circuitos. Las ventajas más importantes de los transistores MOS sobre los bipolares son la alta densidad de agrupación de circuitos, que es una técnica de procesamiento más simple durante la fabricación, y es una operación más económica debido al bajo consumo de energía.

Las características de las familias de lógica digital suelen compararse analizando el circuito de la compuerta básica de cada familia. Los parámetros más importantes que se evalúan y comparan son los siguientes.

La *salida en abanico* especifica el número de cargas estándar que puede manejar la salida de una compuerta típica sin alterar su operación normal. Una carga estándar suele definirse como la cantidad de corriente que se necesita en una entrada de otra compuerta semejante de la misma familia.

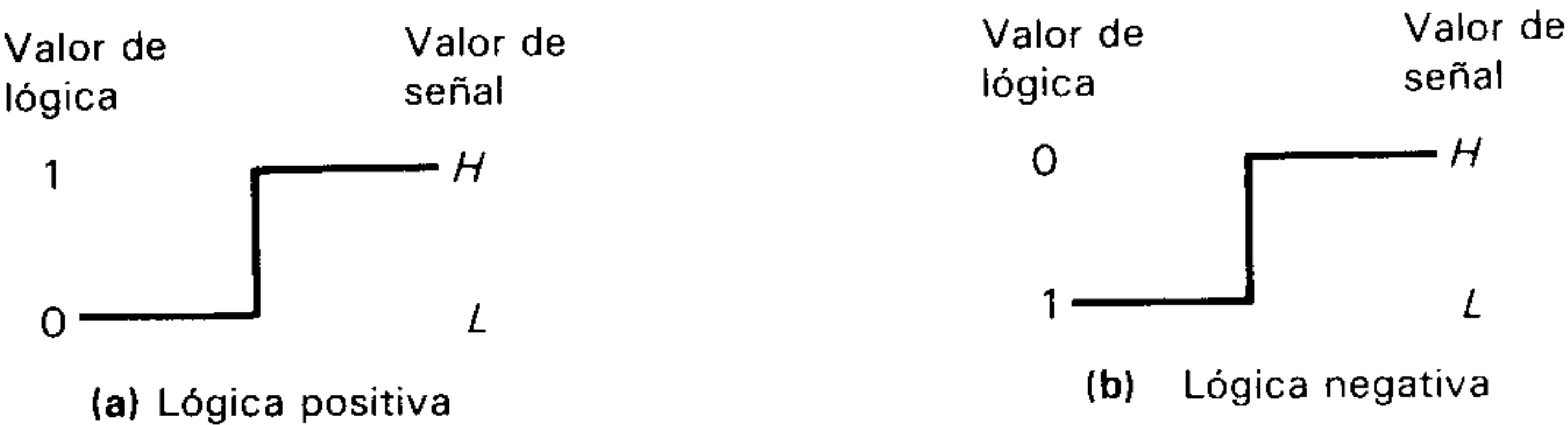
La *disipación de energía* es la energía que consume la compuerta que debe estar disponible de la fuente de suministro de energía.

La *demora en la propagación* es el tiempo promedio de retraso de transición para que la señal se propague de la entrada a la salida cuando la señal binaria cambia de valor. La velocidad de operación es inversamente proporcional a la demora en la propagación.

El *margen de ruido* es el voltaje de ruido externo mínimo que produce un cambio indeseable en la salida del circuito.

Lógica positiva y negativa

La señal binaria en las entradas y salidas de cualquier compuerta tiene uno de dos valores, salvo durante la transición. Un valor de la señal representa el 1 lógico y el otro el 0 lógico. Puesto que se asignan dos valores de señales a dos valores lógicos, existen dos asignaciones diferentes de nivel de la señal al valor lógico, como se muestra en la figura 2-41. El nivel de señal superior está designado por H y el de señal inferior por L. Al elegir el nivel alto H para representar el 1 lógico se define un sistema de lógica positiva. Al elegir el nivel bajo L para representar el 1 lógico se define un sistema de lógica negativa. Los términos positivos y negativos son un tanto confusos por-



Valor de
lógica

0

1

Valor de
señal

H

L

(b) Lógica negativa

FIGURA 2—41
Asignación de señales y polaridad lógica

que ambas señales pueden ser positivas o ambas negativas. No son los valores reales de las señales los que determinan el tipo de lógica, sino la asignación de valores lógicos a las amplitudes relativas de los dos niveles de señales.

Las hojas de datos de los circuitos integrados definen compuertas digitales no en términos de valores lógicos sino en términos de valores de señales como H y L. Corresponde al usuario decidir si se hace una asignación lógica positiva o negativa. Considérese por ejemplo la compuerta TTL que se ilustra en la figura 2-42(b). La tabla de verdad de esta compuerta, como se presenta en un libro de datos, se representa en la figura 2-42(a). Esta especifica el comportamiento físico de la compuerta donde H es 3.5 voltios y L es 0 voltios. La tabla de verdad de la figura 2-42(c) asume una asignación de lógica positiva con H = 1 y L = 0. Esta tabla de verdad es la misma que la de la operación AND. El símbolo gráfico de una compuerta AND de lógica positiva se muestra en la figura 2-42(d).

Ahora considérese la asignación de lógica negativa de la misma compuerta física con L = 1 y H = 0. El resultado es la tabla de verdad de la figura 2-42(e). Esta tabla representa la operación OR aunque las entradas están invertidas. El símbolo gráfico de la

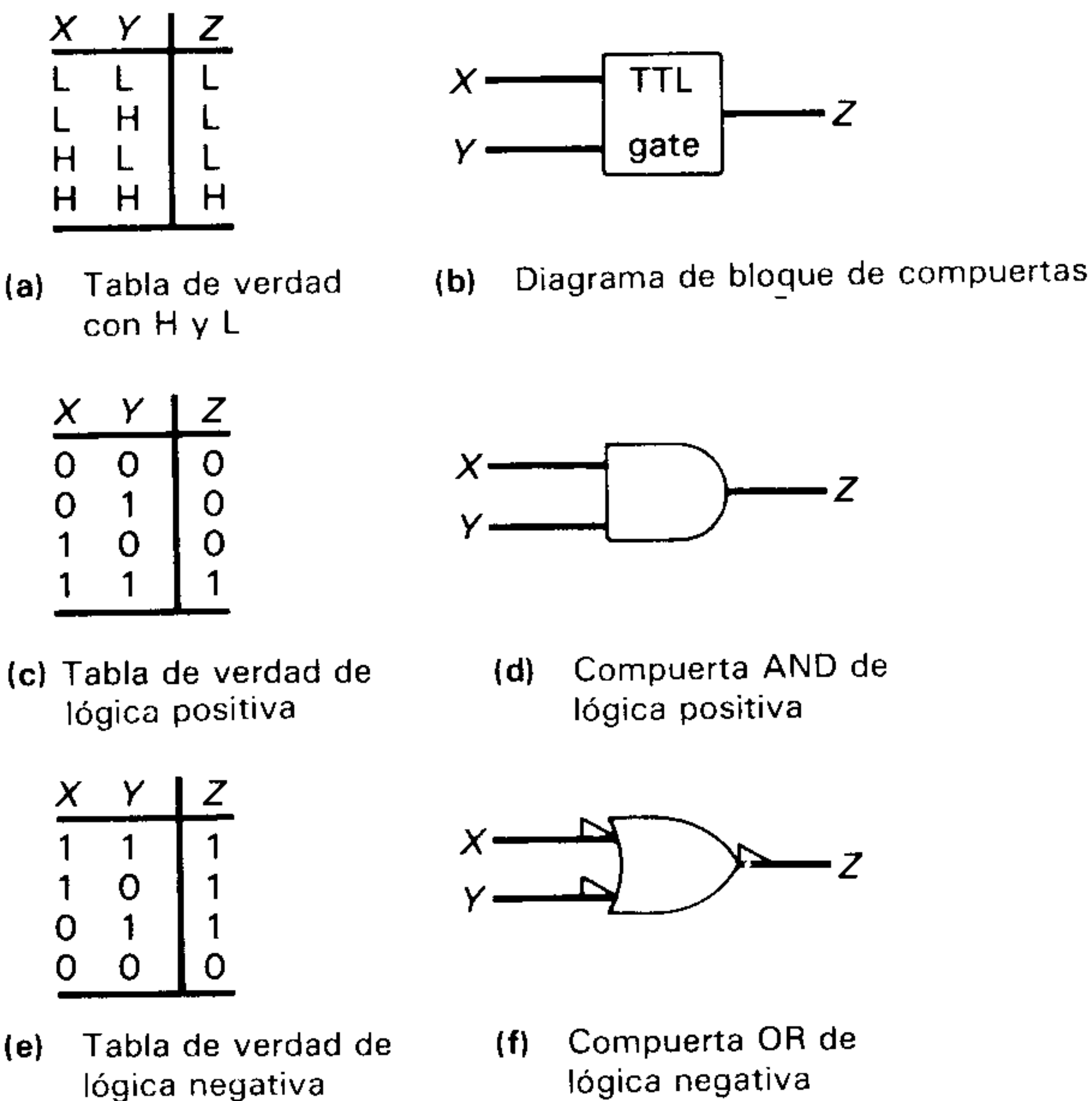


FIGURA 2—42
Demostración de lógica positiva y negativa

compuerta OR de lógica negativa se muestra en la figura 2-42(f). Los triángulos pequeños en las entradas y la salida designan un *indicador de polaridad*. La presencia de este indicador de polaridad junto con una terminal significa que se supone la lógica negativa para la señal. En consecuencia, la misma compuerta física puede operar como una compuerta AND de lógica positiva o bien como una compuerta OR de lógica negativa.

La conversión de lógica negativa a positiva, y viceversa, es básicamente una operación que cambia unos por ceros y ceros por unos en las entradas y la salida de una compuerta. Como esta operación produce el dual de una función, el cambio de todas las terminales de una polaridad a la otra hace que se tome el dual de la función. El resultado de esta conversión es que todas las operaciones AND se convierten en operaciones OR (o símbolos gráficos) y viceversa. Además, no se debe olvidar incluir el triángulo indicador de la polaridad en los símbolos gráficos cuando se suponga la lógica negativa. En este libro no utilizaremos compuertas de lógica negativa, pero supondremos que todas las compuertas operan con una asignación de lógica positiva.

BIBLIOGRAFIA

1. BOOLE, G., *An Investigation of the Laws of Thought*. Nueva York: Dover, 1954.
2. ERCEGOVAC, M. D. y LANG, T., *Digital Systems and Hardware/Firmware Algorithms*. Nueva York: Wiley, 1985.
3. FLETCHER, W. I., *An Engineering Approach to Digital Design*. Englewood Cliffs: Prentice-Hall, 1980.
4. *High-Speed CMOS Logic Data Book*. Dallas: Texas Instruments, 1984.
5. *IEEE Standard Graphic Symbols for Logic Functions*. (ANSI/IEEE Std 91-1984). Nueva York: The Institute of Electrical and Electronics Engineers.
6. KARNAUGH, M., "A Map Method for Synthesis of Combinational Logic Circuits", *Transactions of AIEE, Communications and Electronics*, 72, Part I (nov. 1953), 593-99.
7. MANO, M. M., *Digital Design*. Englewood Cliffs: Prentice-Hall, 1984.
8. ROTH, C. H., *Fundamentals of Logic Design*, 3a./edic. St. Paul: Oeste, 1985.
9. *The TTL Data Book for Design Engineers*. Dallas: Texas Instruments, 1983.

PROBLEMAS

- 2-1 Demuestre por medio de tablas de verdad la validez de las siguientes identidades:
 - (a) Teorema de DeMorgan para tres variables $\overline{XYZ} = \overline{X} + \overline{Y} + \overline{Z}$
 - (b) La segunda ley distributiva. $X + YZ = (X + Y)(X + Z)$
 - (c) $\overline{XY} + \overline{YZ} + X\overline{Z} = X\overline{Y} + Y\overline{Z} + \overline{X}Z$
- 2-2 Pruebe la identidad de cada una de las ecuaciones booleanas que siguen utilizando la manipulación algebraica.
 - (a) $\overline{X}\overline{Y} + XY + \overline{X}Y = \overline{X} + Y$
 - (b) $\overline{X}Y + X\overline{Y} + XY + \overline{X}\overline{Y} = 1$
 - (c) $\overline{X} + XY + X\overline{Z} + X\overline{Y}\overline{Z} = \overline{X} + Y + \overline{Z}$
 - (d) $X\overline{Y} + \overline{Y}\overline{Z} + \overline{X}\overline{Z} = X\overline{Y} + \overline{X}\overline{Z}$

2-3 Simplifique las expresiones booleanas siguientes a un número mínimo de literales.

(a) $XYZ + \bar{X}Y + XY\bar{Z}$

(b) $\bar{X}YZ + XZ$

(c) $(\bar{X} + \bar{Y})(\bar{X} + \bar{Y})$

(d) $XY + X(WZ + W\bar{Z})$

(e) $(X + \bar{Y} + X\bar{Y})(XY + \bar{X}Z + YZ)$ [Respuesta: $XY + \bar{X}\bar{Y}Z$]

2-4 Reduzca las expresiones booleanas que siguen al número de literales que se indica.

(a) $\bar{A}\bar{C} + ABC + A\bar{C}$ A tres literales

(b) $(\bar{C}\bar{D} + A) + A + CD + AB$ A tres literales

(c) $\bar{A}B(\bar{D} + \bar{C}D) + B(A + \bar{A}CD)$ A una literal

(d) $(\bar{A} + C)(\bar{A} + \bar{C})(A + B + \bar{C}D)$ A cuatro literales

2-5 Aplicando el teorema de DeMorgan, exprese la siguiente función

$$F = XY + \bar{X}\bar{Y} + \bar{Y}Z$$

(a) sólo con operaciones OR y de complemento.

(b) sólo con operaciones AND y de complemento.

2-6 Determine el complemento de las expresiones siguientes:

(a) $X\bar{Y} + \bar{X}Y$

(b) $(\bar{A}\bar{B} + C)\bar{D} + E$

(c) $AB(\bar{C}D + C\bar{D}) + \bar{A}\bar{B}(\bar{C} + D)(C + \bar{D})$

(d) $(A + \bar{B} + C)(\bar{A} + \bar{C})(A + B)$

2-7 Obtenga la tabla de verdad de las funciones siguientes y exprese cada una de ellas en suma de minitérminos y producto de maxitérminos.

(a) $(XY + Z)(Y + XZ)$

(b) $(\bar{A} + B)(\bar{B} + C)$

(c) $\bar{Y}Z + WX\bar{Y} + WX\bar{Z} + \bar{W}\bar{X}Z$

2-8 En relación con las funciones booleanas E y F que se dan en la tabla de verdad que sigue

X	Y	Z	E	F
0	0	0	1	0
0	0	1	1	0
0	1	0	1	1
0	1	1	0	1
1	0	0	0	0
1	0	1	0	0
1	1	0	0	1
1	1	1	0	1

(a) Mencione los minitérminos y maxitérminos de cada función.

(b) Señale los minitérminos de \bar{E} y \bar{F} .

(c) Exprese E y F en suma de minitérminos en forma algebraica.

(d) Simplifique cada función a una expresión con un número mínimo de literales.

2-9 Convierta las siguientes expresiones en suma de productos y producto de sumas.

(a) $(AB + C)(B + \bar{C}D)$

(b) $\bar{X} + X(X + \bar{Y})(Y + \bar{Z})$

2-10 Trace el diagrama de lógica de las siguientes expresiones booleanas.

- (a) $B\bar{C} + AB + ACD$
- (b) $(A + B)(C + D)(\bar{A} + B + D)$
- (c) $(AB + \bar{A}\bar{B})(\bar{C}\bar{D} + \bar{C}D)$

2-11 Simplifique las siguientes funciones booleanas por medio de un mapa de tres variables.

- (a) $F(X, Y, Z) = \sum m(2, 3, 6, 7)$
- (b) $F(X, Y, Z) = \sum m(3, 5, 6, 7)$
- (c) $F(A, B, C) = \sum m(0, 2, 3, 4, 6)$
- (d) $F(A, B, C) = \sum m(1, 3, 5, 7)$

2-12 Simplifique las expresiones booleanas que siguen usando un mapa.

- (a) $XY + Y\bar{Z} + \bar{X}\bar{Y}\bar{Z}$
- (b) $\bar{A}\bar{B} + BC + \bar{A}BC$

2-13 Simplifique las siguientes funciones booleanas por medio de un mapa de cuatro variables.

- (a) $F(A, B, C, D) = \sum m(3, 7, 11, 13, 14, 15)$
- (b) $F(W, X, Y, Z) = \sum m(2, 3, 10, 11, 12, 13, 14, 15)$
- (c) $F(A, B, C, D) = \sum m(0, 2, 4, 5, 6, 7, 8, 10, 13, 15)$

2-14 Simplifique las funciones booleanas que siguen usando un mapa.

- (a) $F(W, X, Y, Z) = \sum m(1, 4, 5, 6, 12, 14, 15)$
- (b) $F(A, B, C, D) = \sum m(0, 1, 2, 4, 5, 7, 11, 15)$

2-15 Simplifique las expresiones que siguen por medio de un mapa de cuatro variables.

- (a) $\bar{A}D + BD + \bar{B}C + \bar{A}\bar{B}D$
- (b) $\bar{X}Z + \bar{W}X\bar{Y} + W(\bar{X}Y + X\bar{Y})$
- (c) $\bar{A}\bar{B}C + \bar{B}\bar{C}\bar{D} + BCD + A\bar{C}\bar{D} + \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{C}D$
- (d) $ABC + CD + \bar{B}\bar{C}D + \bar{B}C$

2-16 Determine los minitérminos de las expresiones siguientes trazando primero cada expresión en un mapa.

- (a) $XY + YZ + X\bar{Y}Z$
- (b) $\bar{Y}Z + WXY + WX\bar{Z} + \bar{W}\bar{X}Z$
- (c) $ABC + \bar{B}\bar{D} + \bar{A}BD$

2-17 Obtenga todos los implicantes primos de las siguientes funciones booleanas y determine cuáles de ellos son esenciales.

- (a) $F(W, X, Y, Z) = \sum m(0, 2, 4, 5, 6, 7, 8, 10, 13, 15)$
- (b) $F(A, B, C, D) = \sum m(0, 2, 3, 5, 7, 8, 10, 11, 14, 15)$
- (c) $F(A, B, C, D) = \sum m(1, 3, 4, 5, 10, 11, 12, 13, 14, 15)$

2-18 Simplifique las funciones booleanas que siguen en producto de sumas.

- (a) $F(W, X, Y, Z) = \sum m(0, 2, 5, 6, 7, 8, 10)$
- (b) $F(A, B, C, D) = \Pi M(1, 3, 5, 7, 13, 15)$

2-19 Simplifique las expresiones siguientes en (1) suma de productos y (2) producto de sumas.

- (a) $A\bar{C} + \bar{B}D + \bar{A}CD + ABCD$
- (b) $(\bar{A} + \bar{B} + \bar{D})(A + \bar{B} + \bar{C})(\bar{A} + B + \bar{D})(B + \bar{C} + \bar{D})$
- (c) $(\bar{A} + \bar{B} + D)(\bar{A} + \bar{D})(A + B + \bar{D})(A + \bar{B} + C + D)$

2-20 Simplifique la siguiente función booleana F junto con las condiciones no importa d.

- (a) $F(X, Y, Z) = \sum m(0, 1, 2, 4, 5)$ $d(X, Y, Z) = \sum m(3, 6, 7)$
- (b) $F(A, B, C, D) = \sum m(0, 6, 8, 13, 14)$ $d(A, B, C, D) = \sum m(2, 4, 10)$
- (c) $F(A, B, C, D) = \sum m(1, 3, 5, 7, 9, 15)$ $d(A, B, C, D) = \sum m(4, 6, 12, 13)$

- 2-21** Simplifique las funciones booleanas que siguen por medio de un mapa de cinco variables.
- (a) $F(A, B, C, D, E) = \sum m(0, 1, 4, 5, 16, 17, 21, 25, 29)$
 (b) $\overline{A}\overline{B}\overline{C}\overline{E} + \overline{A}\overline{B}\overline{C}\overline{D} + \overline{B}\overline{D}\overline{E} + \overline{B}\overline{C}\overline{D} + \overline{C}\overline{D}\overline{E} + \overline{B}\overline{D}\overline{E}$
- 2-22** Simplifique la función booleana F junto con las condiciones no importa d , en (1) suma de productos y (2) producto de sumas.
- (a) $F(W, X, Y, Z) = \sum m(0, 1, 2, 3, 7, 8, 10)$
 $d(W, X, Y, Z) = \sum m(5, 6, 11, 15)$
 (b) $F(A, B, C, D) = \sum m(3, 4, 13, 15)$
 $d(A, B, C, D) = \sum m(1, 2, 5, 6, 8, 10, 12, 14)$
- 2-23** Simplifique cada una de las expresiones siguientes y ejecútelas con compuertas NAND.
- (a) $\overline{A}\overline{B} + \overline{A}BD + \overline{A}B\overline{D} + \overline{A}\overline{C}\overline{D} + \overline{A}\overline{B}\overline{C}$ (b) $BD + \overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}\overline{C}\overline{D}$
- 2-24** Ejecute la siguiente expresión con compuertas NAND de dos entradas.
- $(AB + \overline{A}\overline{B})(\overline{C}\overline{D} + \overline{C}D)$
- 2-25** Trace el diagrama de lógica NAND para cada una de las siguientes expresiones utilizando un circuito NAND de múltiples niveles.
- (a) $W(X + Y + Z) + XYZ$ (b) $(\overline{A}\overline{B} + \overline{C}\overline{D})E + BC(A + B)$
- 2-26** Simplifique cada una de las expresiones siguientes y ejecútelas con compuertas NOR.
- (a) $\overline{A}\overline{B} + \overline{C}\overline{D} + \overline{A}\overline{C}\overline{D}$ (b) $\overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}\overline{B}\overline{C}D$
- 2-27** Repita los problemas 2-23 y 2-25 utilizando compuertas NOR.
- 2-28** Demuestre que el dual de la operación OR excluyente es además su complemento.
- 2-29** Determine los circuitos de un generador de paridad de tres bits y de un verificador de paridad de cuatro bits usando un bit de paridad impar.
- 2-30** Ejecute la función booleana que sigue:
- $$F = \overline{A}\overline{B}\overline{C}\overline{D} + \overline{A}B\overline{C}\overline{D} + \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}\overline{C}D$$
- con compuertas OR y AND excluyentes.
- 2-31** Construya un circuito NOR excluyente con dos inversores y dos compuertas de transmisión.
- 2-32** Los SSI TTL vienen incluidos principalmente en paquetes de 14 terminales. Dos de ellas están reservadas para el suministro de energía y las otras se emplean como terminales de entrada y salida. Indique el número de compuertas que se pueden contener en un paquete si éste tiene los siguientes tipos de compuertas:
- (a) Compuertas OR excluyente de dos entradas
 (b) Compuertas AND de tres entradas
 (c) Compuertas NAND de cuatro entradas
 (d) Compuertas NOR de cinco entradas
 (e) Compuertas NAND de ocho entradas
- 2-33** Demuestre que la compuerta NAND de lógica positiva es una compuerta NOR de lógica negativa y viceversa.
- 2-34** Una familia de lógica de circuitos integrados tiene compuertas AND con salida en abanico de 5 y compuertas *buffer* con salida en abanico de 10. Demuestre cómo se puede aplicar la señal de salida de una compuerta AND a otras 50 entradas de compuertas utilizando *buffers*.