

Las constantes de funciones booleanas sólo pueden ser iguales a 1 o a 0. La función de complemento produce el complemento de cada una de las variables binarias. Se llama *transferencia* a toda función que es igual a una variable de entrada, porque la variable, x o y , se transfiere a través de la compuerta que forma la función sin alterar su valor. De los ocho operadores binarios, dos (inhibición e implicación) se usan en lógica, pero casi nunca en lógica de computadoras. Ya mencionamos los operadores AND y OR en relación con el álgebra booleana. Las otras cuatro funciones se usan extensamente en el diseño de sistemas digitales.

La función NOR es el complemento de la función OR y su nombre es la abreviatura de *no* OR. Asimismo, NAND es el complemento de AND y es la abreviatura de *no* AND. El OR exclusivo, que se abrevia XOR, es similar al OR, pero excluye la combinación en que *tanto x como y* son 1. La equivalencia es una función que es 1 cuando las dos variables binarias son iguales, es decir, cuando ambas son 0 o cuando ambas son 1. Las funciones de OR exclusivo y equivalencia son una el complemento de la otra. Esto se comprueba fácilmente inspeccionando la tabla 2-7. La tabla de verdad para el OR exclusivo es F_6 , y para la equivalencia, F_9 , y estas dos funciones son una el complemento de la otra. Por ello, llamamos NOR exclusivo a la función de equivalencia, y la abreviamos XNOR.

El álgebra booleana, tal como se definió en la sección 2-2, tiene dos operadores binarios, a los que hemos llamado AND y OR, y un operador unario, NOT (complemento). De las definiciones, hemos deducido varias propiedades de esos operadores y ahora hemos definido otros operadores binarios en términos de los primeros. Nada tiene de singular tal procedimiento. Bien podríamos haber partido del operador NOR (\downarrow), por ejemplo, y posteriormente haber definido AND, OR y NOT en términos de él. No obstante, hay motivos de peso para introducir el álgebra booleana de la manera en que lo hicimos. Los conceptos de “y”, “o” y “no” son conocidos y la gente los usa para expresar ideas lógicas cotidianas. Además, los postulados de Huntington reflejan la naturaleza dual del álgebra y hacen hincapié en la simetría mutua de $+$ y \cdot .

2-7 COMPUERTAS LÓGICAS DIGITALES

Puesto que las funciones booleanas se expresan en términos de operaciones AND, OR y NOT, es más fácil implementar una función booleana con estos tipos de compuertas. La posibilidad de construir compuertas para las otras operaciones lógicas tiene interés práctico. Los factores a considerar al investigar la construcción de otros tipos de compuertas lógicas son: 1) la factibilidad y economía de producir la compuerta con componentes físicos, 2) la posibilidad de extender la compuerta a más de dos entradas, 3) las propiedades básicas del operador binario, como conmutatividad y asociatividad, y 4) la capacidad de la compuerta para implementar funciones booleanas solas o junto con otras compuertas.

De las 16 funciones definidas en la tabla 2-8, dos son iguales a una constante y cuatro se repiten dos veces. Sólo quedan diez funciones que considerar como candidatas para compuertas lógicas. Dos —inhibición e implicación— no son conmutativas ni asociativas, por lo que no resulta práctico su uso como compuertas lógicas estándar. Las otras ocho: complemento, transferencia, AND, OR, NAND, NOR, OR exclusivo y equivalencia se emplean como compuertas estándar en diseño digital.

Los símbolos gráficos y tablas de verdad de las ocho compuertas aparecen en la figura 2-5. Cada compuerta tiene una o dos variables binarias de entrada designadas con x y y , y una variable binaria de salida designada con F . Ya definimos los circuitos de AND, OR y el inversor





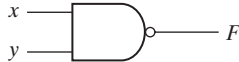
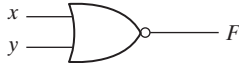


Nombre	Símbolo gráfico	Función algebraica	Tabla de verdad															
AND		$F = xy$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	x	y	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		$F = x'$	<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																	
Búfer		$F = x$	<table><tr><th>x</th><th>F</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	x	F	0	0	1	1									
x	F																	
0	0																	
1	1																	
NAND		$F = (xy)'$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	x	y	F	0	0	1	0	1	1	1	0	1	1	1	0
x	y	F																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
NOR		$F = (x + y)'$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	x	y	F	0	0	1	0	1	0	1	0	0	1	1	0
x	y	F																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
OR exclusivo (XOR)		$F = xy' + x'y$ $= x \oplus y$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	x	y	F	0	0	0	0	1	1	1	0	1	1	1	0
x	y	F																
0	0	0																
0	1	1																
1	0	1																
1	1	0																
NOR exclusivo o equivalencia		$F = xy + x'y'$ $= (x \oplus y)'$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>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 2-5
Compuertas lógicas digitales

en la figura 1-6. El circuito inversor invierte el sentido lógico de una variable binaria: produce la función NOT, o complemento. El pequeño círculo en la salida del símbolo gráfico de un inversor (llamado *burbuja*) indica el complemento lógico. El símbolo de triángulo, por sí solo, denota un circuito búfer. Un búfer produce la función de *transferencia*, pero no una operación lógica, ya que el valor binario de la salida es igual al valor binario de la entrada. Este circuito sirve para amplificar la potencia de la señal y equivale a dos inversores conectados en cascada.

La función NAND es el complemento de la función AND, como lo indica el símbolo gráfico que consiste en un símbolo gráfico AND seguido de una burbuja. La función NOR es el complemento de la función OR y su símbolo gráfico es el de OR seguido de una burbuja. Las compuertas NAND y NOR se usan mucho como compuertas lógicas estándar y, de hecho, son mucho más populares que las compuertas AND y OR. Ello se debe a que es fácil construir compuertas NAND y NOR con circuitos de transistores, y a que es fácil implementar con ellas circuitos digitales.

La compuerta de OR exclusivo tiene un símbolo gráfico parecido al de la compuerta OR, sólo que lleva una línea curva adicional del lado de la entrada. La compuerta de equivalencia, o NOR exclusivo, es el complemento del OR exclusivo, como indica la burbuja en el lado de salida del símbolo gráfico.

Extensión a múltiples entradas

Las compuertas que se muestran en la figura 2-5 —con excepción del inversor y el búfer— se pueden extender de modo que tengan más de dos entradas. Es posible extender una compuerta a múltiples entradas si la operación binaria que representa es conmutativa y asociativa. Las operaciones AND y OR, definidas en el álgebra booleana, poseen esas dos propiedades. Para la función OR, tenemos

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

y

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

lo que nos dice que las entradas de la compuerta son intercambiables y que la función OR se puede extender a tres o más variables.

Las funciones NAND y NOR son conmutativas, y sus compuertas se extienden a más de dos entradas, si se modifica ligeramente la definición de la operación. El problema radica en que los operadores NAND y NOR no son asociativos [es decir, $(x \downarrow y) \downarrow z \neq x \downarrow (y \downarrow z)$], como se indica en la figura 2-6 y en las ecuaciones siguientes:

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

Para superar este problema, definimos la compuerta NOR (o NAND) múltiple como una compuerta OR (o AND) complementada. Así, por definición, tenemos

$$\begin{aligned} x \downarrow y \downarrow z &= (x + y + z)' \\ x \uparrow y \uparrow z &= (xyz)' \end{aligned}$$

Los símbolos gráficos para las compuertas de tres entradas se incluyen en la figura 2-7. Al escribir operaciones NOR y NAND en cascada, hay que usar los paréntesis correctos para indi-

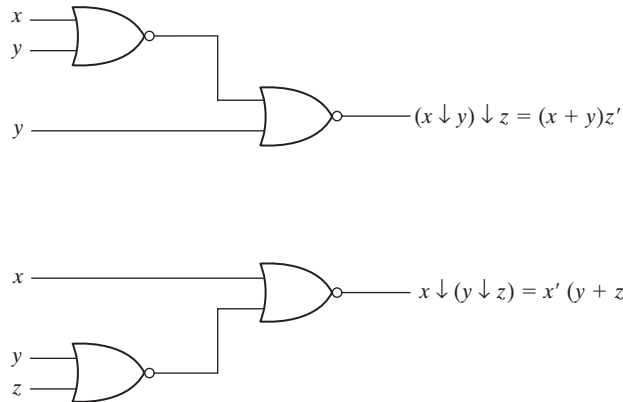


FIGURA 2-6
Demostración de la no asociatividad del operador NOR; $(x \downarrow y) \downarrow z \neq x \downarrow (y \downarrow z)$

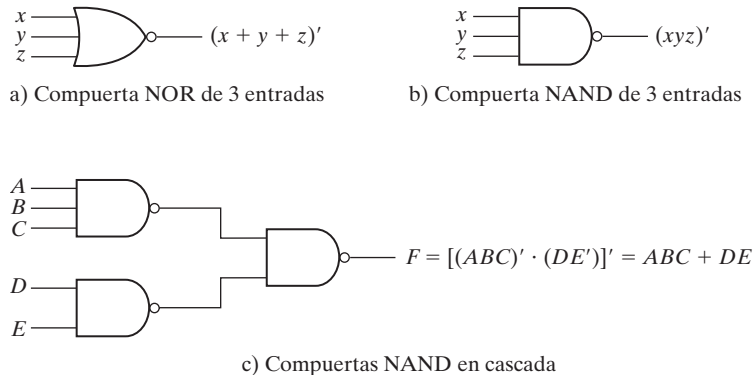


FIGURA 2-7
Compuertas NOR y NAND con múltiples entradas y en cascada

car el orden en que deben ir las compuertas. Para demostrar esto, consideremos el circuito de la figura 2-7c). La función booleana del circuito se escribe así:

$$F = [(ABC)'](DE')]' = ABC + DE$$

La segunda expresión se obtiene del teorema de DeMorgan, y también demuestra que una expresión en forma de suma de productos se puede implementar con compuertas NAND. En la sección 3-6 trataremos más a fondo las compuertas NAND y NOR.

Las compuertas OR exclusivo y de equivalencia son tanto conmutativas como asociativas y se pueden extender a más de dos entradas. No obstante, las compuertas OR exclusivo de varias entradas son poco comunes en hardware. De hecho, incluso la función de dos entradas suele construirse con otros tipos de compuertas. Además, es preciso modificar la definición de la función al extenderla a más de dos variables. El OR exclusivo es una función *impar*, es decir, es igual a 1 si las variables de entrada tienen un número impar de unos. En la figura 2-8 se representa la construcción de una función OR exclusivo de tres entradas,

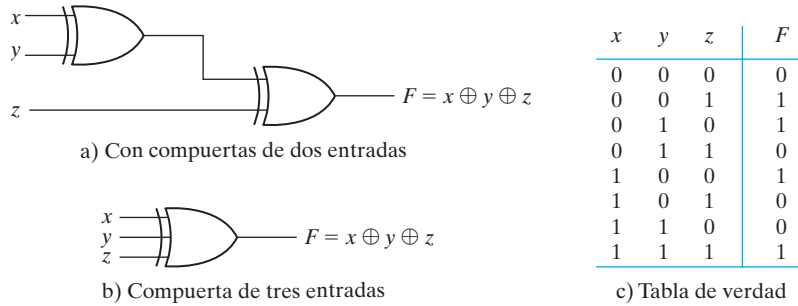


FIGURA 2-8
Compuerta OR exclusivo de tres entradas

aunque normalmente se la implementa conectando en cascada compuertas de dos entradas, como se observa en a). Gráficamente el OR exclusivo se representa con una sola compuerta de tres entradas, como en b). La tabla de verdad de c) indica claramente que la salida de F es igual a 1 si sólo una entrada es 1 o si las tres entradas son 1, es decir, si el número total de unos en las variables de entrada es *impar*. En la sección 3-8 se estudiará más a fondo el OR exclusivo.

Lógica positiva y negativa

La señal binaria en las entradas y salidas de cualquier compuerta tiene uno de dos valores, excepto durante una transición. Un valor de señal representa el 1 lógico, y el otro, el 0 lógico. Puesto que se asignan dos valores de señal a dos valores lógicos, puede haber dos asignaciones distintas de nivel de señal a valor lógico, como se indica en la figura 2-9. El nivel de señal más alto se designa con H , y el más bajo, con L . Si escogemos el nivel alto H para representar el 1 lógico, estaremos definiendo un sistema de lógica positiva. Si escogemos el nivel bajo L para representar el 1 lógico, definiremos un sistema de lógica negativa. Los términos positiva y negativa son un tanto engañosos porque ambas señales podrían ser positivas, o ambas negativas. No son los valores reales de la señal lo que determina el tipo de lógica, sino más bien la asignación de valores lógicos a las amplitudes relativas de los dos niveles de señal.

Las compuertas digitales en hardware se definen en términos de valores de señal como H y L . Corresponde al usuario decidir si la polaridad de la lógica va a ser positiva o negativa. Con-

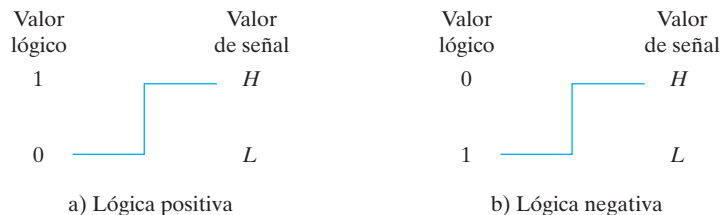
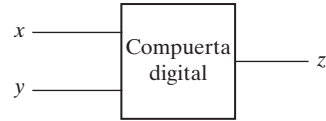


FIGURA 2-9
Asignación de señales y polaridad lógica

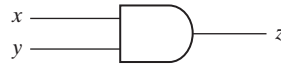
x	y	F
L	L	L
L	H	L
H	L	L
H	H	H

a) Tabla de verdad con H y L 

b) Diagrama de bloque de compuerta

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1

c) Tabla de verdad para lógica positiva



d) Compuerta AND de lógica positiva

x	y	z
1	1	1
1	0	1
0	1	1
0	0	0

e) Tabla de verdad para lógica negativa



f) Compuerta OR de lógica negativa

FIGURA 2-10
Demostración de lógica positiva y negativa

sideremos, por ejemplo, la compuerta electrónica que se muestra en la figura 2-10b). La tabla de verdad de esta compuerta se presenta en la figura 2-10a), y especifica el comportamiento físico de la compuerta cuando H es 3 volts y L es 0 volts. La tabla de verdad de la figura 2-10c) supone una asignación de lógica positiva, con $H = 1$ y $L = 0$. Esta tabla de verdad es igual a la de la operación AND. El símbolo gráfico para una compuerta AND con lógica positiva se muestra en la figura 2-10d).

Consideremos ahora la asignación de lógica negativa a la misma compuerta física, con $L = 1$ y $H = 0$. El resultado es la tabla de verdad de la figura 2-10e). Esta tabla representa la operación OR aunque las filas están invertidas. El símbolo gráfico para la compuerta OR de lógica negativa se aprecia en la figura 2-10f). Los pequeños triángulos en las entradas y la salida son *indicadores de polaridad*. La presencia de este indicador de polaridad en una terminal implica que se está suponiendo lógica negativa para la señal. Así, la misma compuerta física puede operar como compuerta AND de lógica positiva o como compuerta OR de lógica negativa.

La conversión de lógica positiva a lógica negativa, y viceversa, es básicamente una operación que cambia los unos a ceros y los ceros a unos tanto en las entradas como en la salida de la compuerta. Puesto que esta operación produce el dual de una función, el cambio de todas las terminales, de una polaridad a la otra, equivale a obtener 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 debemos olvidarnos de incluir el triángulo indicador de polaridad en los símbolos gráficos cuando se supone lógica negativa. En este libro no usaremos compuertas de lógica negativa, y supondremos que todas las compuertas operan con una asignación de lógica positiva.

2-8 CIRCUITOS INTEGRADOS

Un circuito integrado (que se abrevia CI) es un cristal semiconductor de silicio, llamado *chip*, que contiene los componentes electrónicos para construir compuertas digitales. Las diversas compuertas se interconectan dentro del chip para formar el circuito requerido. El chip se monta en un recipiente de cerámica o plástico, y las conexiones se sueldan a terminales externas para formar el circuito integrado. El número de terminales podría variar desde 14 en un paquete de CI pequeño hasta varios miles en los paquetes más grandes. Cada CI tiene una designación numérica impresa en la superficie del paquete, para poder identificarlo. Los fabricantes proporcionan libros de datos, catálogos y sitios Web de Internet que contienen descripciones e información acerca de los CI que producen.

Niveles de integración

Los CI digitales suelen clasificarse según la complejidad de sus circuitos, la cual se mide por el número de compuertas lógicas incluidas en el paquete. La diferenciación entre los chips que tienen pocas compuertas internas y los que tienen cientos de miles de compuertas suele hacerse diciendo que el paquete es un dispositivo de integración a pequeña, mediana, gran o muy grande escala.

Los dispositivos de *integración a pequeña escala* (SSI, *small-scale integration*) 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 que 10 y está limitado por el número de terminales con que cuenta el CI.

Los dispositivos de *integración a mediana escala* (MSI, *medium-scale integration*) tienen una complejidad de entre 10 y 1000 compuertas en un solo paquete. Por lo regular, efectúan operaciones digitales elementales específicas. Presentaremos las funciones digitales de MSI en el capítulo 4 como decodificadores, sumadores y multiplexores, y en el capítulo 6, como registros y contadores.

Los dispositivos de *integración a gran escala* (LSI, *large-scale integration*) contienen miles de compuertas en un solo paquete. Incluyen sistemas digitales como procesadores, chips de memoria y dispositivos de lógica programable. Presentaremos algunos componentes LSI en el capítulo 8.

Los dispositivos de *integración a muy grande escala* (VLSI, *very large-scale integration*) contienen cientos de miles de compuertas en un solo paquete. Como ejemplo podemos citar las grandes matrices de memoria y los microprocesadores complejos. En virtud de su pequeño tamaño y bajo costo, los dispositivos VLSI han revolucionado la tecnología de diseño de sistemas de cómputo y confieren al diseñador la capacidad de crear estructuras que antes no resultaban económico construir.

Familias de lógica digital

Los circuitos lógicos integrados se clasifican no sólo por su complejidad o por su funcionamiento lógico, sino también por la tecnología específica de circuitos utilizada en su construcción. Llamamos a esa tecnología familia de lógica digital. Cada familia de lógica tiene su propio circuito electrónico básico sobre el que se desarrollan circuitos digitales y componentes más complejos. El circuito básico en cada tecnología es una compuerta NAND, NOR o inversora. Por lo regular, se usan los componentes electrónicos empleados en la construcción del circuito básico para dar nombre a la tecnología. Se han introducido comercialmente muchas familias lógicas de circuitos integrados digitales. Las más populares son:

TTL	lógica transistor-transistor;
ECL	lógica acoplada por emisor;
MOS	metal-óxido-semiconductor;
CMOS	metal-óxido-semiconductor complementario.

TTL es una familia lógica que ha estado en operación mucho tiempo y se le considera estándar. ECL resulta ventajoso en sistemas que deben operar a alta velocidad. MOS es apropiado para circuitos que requieren una densidad elevada de componentes, y CMOS es preferible en sistemas que requieren bajo consumo de energía. Esto último es indispensable para el diseño de VLSI, así que CMOS se ha convertido en la familia lógica dominante, mientras que el uso de TTL y ECL ha decaído. El análisis del circuito electrónico básico de una compuerta digital para cada familia de lógica se presenta en el capítulo 10.

Las características de las familias de lógica digital suelen compararse analizando el circuito de la compuerta básica de cada familia. En la sección 10-2 veremos los parámetros más importantes que se evalúan y comparan. Aquí sólo se mencionan como referencia.

El *abanico de salida* (*fan-out*) especifica el número de cargas estándar que la salida de una compuerta representativa es capaz de alimentar sin merma de su funcionamiento normal. La carga estándar por lo regular se define como la cantidad de corriente que requiere en una de sus entradas otra compuerta similar de la misma familia.

El *abanico de entrada* (*fan-in*) es el número de entradas con que cuenta la compuerta.

La *disipación de potencia* es la energía consumida por la compuerta y que la fuente de potencia debe suministrar.

El *retardo de propagación* es el tiempo medio de transición que la señal tarda al propagarse de la entrada a la salida. La velocidad de operación es inversamente proporcional al retardo de propagación.

El *margen de ruido* es el voltaje externo máximo de ruido que puede añadirse a una señal de entrada sin causar un cambio indeseable en la salida del circuito.

Diseño asistido por computadora (CAD)

El diseño de sistemas digitales con circuitos VLSI que contienen millones de transistores es una tarea imponente. En general, es imposible desarrollar y verificar sistemas tan complejos sin la ayuda de herramientas computarizadas para diseño. Las herramientas de CAD consisten en programas de software que apoyan la representación computarizada y ayudan a desarrollar hardware digital automatizando el proceso de diseño. La automatización del diseño electrónico cubre todas las fases del diseño de circuitos integrados. Un flujo de diseño típico para crear circuitos VLSI consiste en una sucesión de pasos que inicia con la introducción del diseño y cul-