Nivel de Lógica Digital: Álgebra booleana y compuertas lógicas

Ing. Ronald Caravaca Mora

IF4000 - Arquitectura de Computadores
Informática Empresarial

3 de septiembre de 2021

UNIVERSIDAD DE COSTA RICA

1. Introducción a la lógica binaria
2. Álgebra booleana
3. Teoremas y propiedades básicas del Álgebra booleana

1. Introducción a la lógica binaria
2. Álgebra booleana
3. Teoremas y propiedades básicas del Álgebra booleana

Lógica binaria

1. Se ocupa de variables que adoptan dos valores discretos y de operaciones que asumen un significado lógico.
2. En nuestro caso las variable discretas serán los bits (0, 1).
3. La lógica binaria que estudiaremos equivale a un álgebra llamada álgebra booleana.
4. Consiste en variables binarias y operaciones lógicas.
5. Las variables se designan con letras del alfabeto, como A, B, C, x, y, z, etc.
6. Hay tres operaciones lógicas básicas: AND, OR y NOT.

Lógica binaria

1. **AND**: Esta operación se representa con un punto u omitiendo el operador. Por ejemplo, $x \cdot y = z$ o $xy = z$ se lee “x AND y es igual a z”. La operación lógica AND significa que $z = 1$ si y sólo si $x = 1$ y $y = 1$; de lo contrario, $z = 0$.
2. **OR**: Esta operación se representa con un signo más. Por ejemplo, $x + y = z$ se lee “x OR y es igual a z”, y significa que $z = 1$ si $x = 1$ o si $y = 1$ o si $x = 1$ y $y = 1$. Si $x = 0$ y $y = 0$, entonces $z = 0$.
3. **NOT**: Esta operación se representa con un apostrofe o “techito”. Por ejemplo, $x' = z$ ó $\overline{x} = z$ se lee como ”no x es igual a z” y significa que z es lo contrario de x. Dicho de otro modo, si $x = 1$, entonces $z = 0$; pero si $x = 0$, entonces $z = 1$. La operación NOT también se llama operación de complemento, ya que cambia un 1 por 0 y un 0 por 1.

Tablas de verdad

1. Para cada combinación de los valores de x y y, la definición de la operación lógica especifica un valor de z. Dichas definiciones se pueden presentar en forma compacta con tablas de verdad.
2. Una tabla de verdad es una tabla de todas las posibles combinaciones de las variables, y muestra la relación entre los valores que las variables pueden adoptar y el resultado de la operación.

Tablas de verdad

1. Para cada combinación de los valores de x y y, la definición de la operación lógica especifica un valor de z. Dichas definiciones se pueden presentar en forma compacta con tablas de verdad.
2. Una tabla de verdad es una tabla de todas las posibles combinaciones de las variables, y muestra la relación entre los valores que las variables pueden adoptar y el resultado de la operación.

AND

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

OR

x	y	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

NOT

x	x'
0	1
1	0

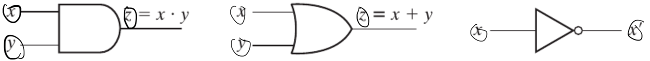
Figura 1: Tablas de verdad para las operaciones AND, OR, NOT.

Compuertas lógicas

1. Las compuertas lógicas son circuitos electrónicos que operan con una o más señales de entrada para producir una señal de salida.
2. En los sistemas digitales, son las señales eléctricas como voltajes o corrientes dentro de un rango determinado, las que se utilizan para el manejo de los valores discretos.
3. Estos circuitos están construidos a base de transistores y están diseñados para satisfacer requisitos lógicos de entrada. Es decir, para comportarse como una operación lógica.

Compuertas lógicas

1. Las compuertas lógicas son circuitos electrónicos que operan con una o más señales de entrada para producir una señal de salida.
2. En los sistemas digitales, son las señales eléctricas como voltajes o corrientes dentro de un rango determinado, las que se utilizan para el manejo de los valores discretos.
3. Estos circuitos están construidos a base de transistores y están diseñados para satisfacer requisitos lógicos de entrada. Es decir, para comportarse como una operación lógica.

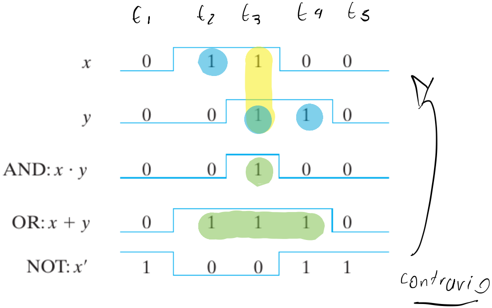


a) Compuerta AND de dos entradas

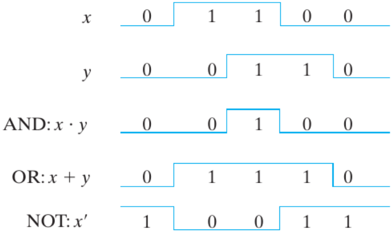
b) Compuerta OR de dos entradas

(c) Compuerta NOT o inversor

Compuertas lógicas

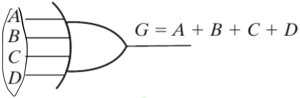


Compuertas lógicas



Las compuertas pueden tener mas de 2 entradas:

a) Compuerta AND de tres entradas



b) Compuerta OR de cuatro entradas

1. Introducción a la lógica binaria
2. Álgebra booleana
3. Teoremas y propiedades básicas del Álgebra booleana

Álgebra booleana I

1. El álgebra booleana, al igual que todos los sistemas matemáticos deductivos, se define con un conjunto de elementos, un conjunto de operadores y varios axiomas o postulados.
    1. Un *conjunto de elementos* es cualquier colección de objetos con alguna propiedad en común.
    2. Un *operador* es una regla que asigna a cada par de elementos de un conjunto un elemento único del mismo conjunto.
    3. Los *axiomas o postulados* de un sistema matemático constituyen los supuestos básicos a partir de los cuales es posible deducir las reglas, teoremas y propiedades del sistema.
2. Los postulados mas comunes son:
    1. *Cerradura:* Un conjunto S es cerrado respecto a un operador binario si, por cada par de elementos de S, el operador especifica una regla para obtener un elemento único de S.

Álgebra booleana II

2. *Ley asociativa:* Decimos que un operador binario $*$ sobre un conjunto S es asociativo si: $(x * y) * z = x * (y * z)$.
3. *Ley conmutativa:* Decimos que un operador binario $*$ sobre un conjunto S es conmutativo si: $x * y = y * x$ para todos x, $y \in S$.
4. *Elemento de identidad:* Decimos que un conjunto S tiene un elemento de identidad respecto a una operación binaria $*$ sobre S si existe un elemento $e \in S$ con la propiedad: $e * x = x * e = x$ para todos $x \in S$.
5. *Inverso:* Decimos que un conjunto S, que tiene el elemento de identidad e respecto a un operador $*$, tiene un inverso si, para todo $x \in S$, existe un elemento y $\in S$ tal que $x * y = e$.
6. *Ley distributiva:* Si $*$ y $\cdot$ son dos operadores binarios sobre un conjunto S, decimos que $*$ es distributivo sobre $\cdot$ si: $x * (y \cdot z) = (x * y) \cdot (x * z)$.

$1 \cdot n = 1$

$0 + n = n$

$2 \cdot \frac{1}{2} = 1$

Definición axiomática del Álgebra booleana

1. En 1854 George Boole definió un sistema algebraico para el tratamiento de la lógica a la que llamó álgebra de Boole o álgebra Booleana.
2. En 1904 Huntington propuso un sistema de axiomas para la definición formal del álgebra de Boole.
3. En 1938 Shannon demostró que podía aplicar el álgebra Booleana a los circuitos de conmutación eléctricas biestables.

Definición axiomática del Álgebra booleana I

El álgebra booleana es una estructura algebraica definida por un conjunto de elementos, B, junto con dos operadores binarios, $+$ y $\cdot$, a condición de que se satisfagan los postulados siguientes:

1. Cerradura respecto al operador $+$.
2. Cerradura respecto al operador $\cdot$.
3. Un elemento de identidad con respecto a $+$, designado por 0: $x + 0 = 0 + x = x$.
4. Un elemento de identidad con respecto a $\cdot$, designado por 1: $x \cdot 1 = 1 \cdot x = x$.
5. Conmutativa respecto a $+$: $x + y = y + x$.
6. Conmutativa respecto a $\cdot$: $x \cdot y = y \cdot x$.

Definición axiomática del Álgebra booleana II

AND OR

7. $+$ es distributivo sobre $\cdot$: $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$.
8. $\cdot$ es distributivo sobre $+$: $x + (y \cdot z) = (x + y) \cdot (x + z)$.
9. Para cada elemento $x \in B$, existe un elemento $x' \in B$ (llamado complemento de x) tal que a) $x + x' = 1$ y b) $x \cdot x' = 0$.
10. Existen por lo menos dos elementos $x, y \in B$ tales que $x \neq y$.

Definición axiomática del Álgebra booleana I

Como ejemplo, tenemos el álgebra de dos valores: Un álgebra booleana de dos valores se define sobre un conjunto de dos elementos, $B = 1, 0$, con las reglas para los dos operadores binarios, $+$ y $\cdot$, que se muestran en las siguientes tablas de operador:

x	y	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

x	y	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

x	x'
0	1
1	0

Como ejercicio para la casa, queda demostrar los postulados para este conjunto.

1. Introducción a la lógica binaria
2. Álgebra booleana
3. Teoremas y propiedades básicas del Álgebra booleana

Teoremas y propiedades básicas del Álgebra booleana

Principio de dualidad: Establece que toda expresión algebraica que pueda deducirse de los postulados del álgebra booleana seguirá siendo válida si se intercambian los operadores y los elementos de identidad. En un álgebra booleana de dos valores, los elementos de identidad y los elementos del conjunto, B, son los mismos: 1 y 0. El principio de dualidad tiene muchas aplicaciones. Si queremos el dual de una expresión algebraica, simplemente intercambiamos los operadores OR y AND y sustituimos los unos por ceros y los ceros por unos. Por ejemplo:

$$A + (B \cdot C) = 1, \text{ por dualidad se tiene: } A \cdot (B + C) = 0$$

Teoremas y propiedades básicas del Álgebra booleana

Postulado 2	a)	$x + 0 = x$	b)	$x \cdot 1 = x$
Postulado 5	a)	$x + x' = 1$	b)	$x \cdot x' = 0$
Teorema 1	a)	$x + x = x$	b)	$x \cdot x = x$
Teorema 2	a)	$x + 1 = 1$	b)	$x \cdot 0 = 0$
Teorema 3, involución		$(x')' = x$		
Postulado 3, conmutatividad	a)	$x + y = y + x$	b)	$xy = yx$
Teorema 4, asociatividad	a)	$x + (y + z) = (x + y) + z$	b)	$x(yz) = (xy)z$
Postulado 4, distributividad	a)	$x(y + z) = xy + xz$	b)	$x + yz = (x + y)(x + z)$
Teorema 5, DeMorgan	a)	$(x + y)' = x'y'$	b)	$(xy)' = x' + y'$
Teorema 6, absorción	a)	$x + xy = x$	b)	$x(x + y) = x$

Funciones booleanas (ecuaciones booleana o expresiones booleanas

Una función Booleana es una expresión formada por variables binarias, operadores AND, OR y NOT, paréntesis y signo igual (=). Para un valor dado de las variables, la función puede ser 0 o 1.

Funciones booleanas (ecuaciones booleana o expresiones booleanas)

Ejemplos:

And Not

$F1 = xyz'$ $\quad$ $F1 = 1 <=> x = 1, y = 1, z = 0$

OR Not And

$F2 = x + y'z$ $\quad$ $F2 = 1 <=> x = 1$ ó $(y = 0$ y $z = 1)$

Not Not Not Not

$F3 = x'y'z + x'yz + xy'$ $\quad$ $F3 = 1 <=> (x = 0$ y $y = 0$ y $z = 1)$

and and Or and and Or and

ó $(x = 0$ y $y = 1$ y $z = 1)$ ó $(x = 1$ y $y = 0)$

$F4 = xy' + x'z$ $\quad$ $F4 = 1 <=> (x = 1$ y $y = 0)$

and Or and

ó $(x = 0$ y $z = 1)$

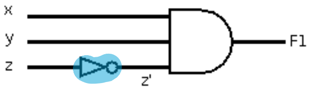
Funciones booleanas (ecuaciones booleana o expresiones booleanas)

Representación en tabla de verdad:

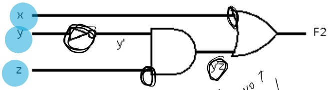
entradas	x	y	z	F1	F2	F3	F4
	0	0	0	0	0	0	0
1	0	0	1	0	1	1	1
	0	1	0	0	0	0	0
2	0	1	1	0	0	1	1
1	1	0	0	0	1	1	1
1	1	0	1	0	1	1	1
2	1	1	0	1	1	0	0
3	1	1	1	0	1	0	0

Funciones booleanas (ecuaciones booleana o expresiones booleanas)

Representación en compuertas lógicas:

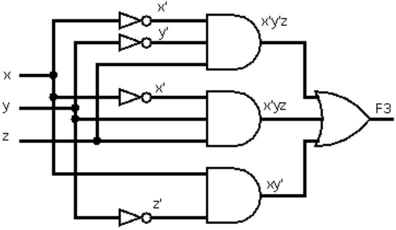


(a) $F1 = xyz'$

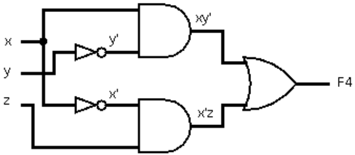


or not and

(b) $F2 = x + y'z$



(c) $F3 = x'y'z + x'yz + xy'$



(d) $F4 = xy' + x'z$

Reducción de funciones booleanas

Dos funciones booleanas de n variables binarias son iguales si ellas tienen el mismo valor para todas las 2^n combinaciones posibles de las en variables.

Reducción de funciones booleanas

Dos funciones booleanas de n variables binarias son iguales si ellas tienen el mismo valor para todas las 2^n combinaciones posibles de las en variables.

x	y	z	F1	F2	F3	F4
0	0	0	0	0	0	0
0	0	1	0	1	1	1
0	1	0	0	0	0	0
0	1	1	0	0	1	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	0

Reducción de funciones booleanas

La funciones $F3$ y $F4$ son equivalentes, lo cual significa que puede reducirse hasta obtener expresión dada en $F4$, de la siguitente manera:

$F3 = x'y'z + x'yz + xy'$	
$F3 = x'(y'z + yz) + xy'$	distributividad
$F3 = x'(z(y' + y)) + xy'$	distributividad
$F3 = x'(z(\cancel{1})) + xy'$	postulado 5
$F3 = x'z + xy'$	postulado 2
$F3 = xy' + x'z$	Por conmutatividad $F3 = F4$

4 Literal
2 termios

→ Cada variable = literal

→ cada conjunto = terminos

Reducción de funciones booleanas

Cuando se implementa una expresión booleana con compuertas lógicas, cada *término* requiere una compuerta y cada variable dentro del término implica una entrada a la compuerta. Definimos un *literal* como una sola variable dentro de un término, que podría estar complementada. En el ejemplo anterior, $F3$ contiene 2 términos y 4 literales.

Si reducimos el número de términos, el número de literales, o ambas cosas, en una expresión booleana, podría obtenerse un circuito más sencillo. La manipulación del álgebra booleana consiste en su mayor parte en reducir una expresión con el objetivo de obtener un circuito más simple.

Reducción de funciones booleanas

Al realizar las expresiones booleanas se debe tomar en cuenta la prioridad de signos en el siguiente orden:

1. Paréntesis ( )
2. NOT ·
3. AND +
4. OR '

Ejemplos: I

Simplifique las funciones booleanas siguientes al número mínimo de literales.

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

$$F = xx' + xy \qquad \text{distributividad}$$
$$F = 0 + xy \qquad \text{Postulado 5}$$
$$F = xy \qquad \text{postulado 2}$$

2. $F = x + x'y$.

$$F = (x + x')(x + y) \qquad \text{distributividad}$$
$$F = 1(x + y) \qquad \text{Postulado 5}$$
$$F = x + y \qquad \text{postulado 2}$$

Ejemplos: II

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

$F = x + xy + xy' + yy'$ — distributividad

$F = x(1 + \underbrace{y + y'}_{})$ — distributividad y postulado 5

$F = x$ — postulado 5

1 + 1 = 1

4. $F = xy + x'z + yz$.

identidad = 1

$F = xy + x'z + yz(x + x')$ — mas literales con postulado 5

$F = xy + x'z + xyz + x'yz$ — distributividad

ordenar terminos

$F = \underbrace{xy + xyz} + \underbrace{x'z + x'yz}$ — conmutatividad

$F = xy(1 + z) + x'z(1 + y)$ — distributividad y conmutatividad

$F = xy + x'z$ — teorema 2

Ejemplos: III

And or and or and

5. $F = (x + y)(x' + z)(y + z)$.

Cambiar And por Or y OR por And.

$F = xy + x'z + yz$ por dualidad. Esto es igual a la función en 4

$F = xy + x'z$

$F = (x + y)(x' + z)$ por dualidad

Las funciones en 4 y 5 llevan el nombre de *teorema de consenso*, donde:

contrario

$$F_4 = xy + x'z + yz = xy + x'z$$
$$F_5 = (x + y)(x' + z)(y + z) = (x + y)(x' + z)$$

Complemento de una función booleana

El complemento de una función F es F' y se obtiene intercambiando ceros por unos y unos por ceros en el valor de F. El complemento de una función podría deducirse algebraicamente empleando el *teorema de DeMorgan*. Los teoremas de DeMorgan para cualquier número de variables tienen una forma similar al caso de dos variables y se deducen por sustituciones sucesivas como se hizo en la deducción anterior. Estos teoremas se pueden generalizar así:

Cambiar signo

$$(A + B + C + D + \ldots + F)' = A'B'C'D' \ldots F'$$
$$(ABCD \ldots F)' = A' + B' + C' + D' + \ldots + F'$$

La forma generalizada del teorema de DeMorgan dice que el complemento de una función se obtiene intercambiando operadores AND y OR y complementando cada literal.

Complemento de una función booleana

Ejemplo: Halle el complemento de las funciones $F1 = x'yz' + x'y'z$ y $F2 = x(y'z' + yz)$.

$$\begin{aligned} F1' &= (x'yz' + x'y'z)' \\ &= (x'yz')'(x'y'z)' && \text{DeMorgan} \\ &= (x + y' + z)(x + y + z') && \text{DeMorgan} \end{aligned}$$

$$\begin{aligned} F2' &= [x(y'z' + yz)]' \\ &= x' + (y'z' + yz)' && \text{DeMorgan} \\ &= x' + (y'z')'(yz)' && \text{DeMorgan} \\ &= x' + (y + z)(y' + z') && \text{DeMorgan} \end{aligned}$$

Complemento de una función booleana

Un procedimiento más sencillo para deducir el complemento de una función consiste en obtener el **dual de la función y complementar cada literal**.

Complemento de una función booleana

Ejemplo: Determine el complemento de las funciones $F1$ y $F2$ del ejemplo anterior obteniendo sus duales y complementando cada literal.

cambia todo

$$F1 = x'yz' + x'y'z$$

$$= (x' + y + z')(x' + y' + z) \quad \text{por dualidad}$$

$$F1' = (x + y' + z)(x + y + z') \quad \text{Complementando cada literal}$$

$$F2 = x(y'z' + yz)$$

$$= x + (y' + z')(y + z) \quad \text{por dualidad}$$

$$F2' = x' + (y + z)(y' + z') \quad \text{Complementando cada literal}$$

1 Introducción

2 Álgebra booleana

3 Teoremas y propiedades

4 Funciones booleanas

5 Formas canónicas y estándar

Formas canónicas y estándar: Minitérminos y maxitérminos

si hay 0 = negado
→ producto

si hay 1 = negado
→ suma

Forma canónica:

			Minitérminos		Maxitérminos	
x	***y***	***z***	**Términos**	**Designación**	**Términos**	**Designación**
0	0	0	$x'y'z'$	m_0	$x + y + z$	M_0
0	0	1	$x'y'z$	m_1	$x + y + z'$	M_1
0	1	0	$x'yz'$	m_2	$x + y' + z$	M_2
0	1	1	$x'yz$	m_3	$x + y' + z'$	M_3
1	0	0	$xy'z'$	m_4	$x' + y + z$	M_4
1	0	1	$xy'z$	m_5	$x' + y + z'$	M_5
1	1	0	xyz'	m_6	$x' + y' + z$	M_6
1	1	1	xyz	m_7	$x' + y' + z'$	M_7

Formas canónicas y estándar: Minitérminos y maxitérminos

Se puede expresar algebraicamente una función booleana a partir de una tabla de verdad dada formando un minitérmino para cada combinación de las variables que produce un 1 en la función, y formando después el OR de todos esos términos.

Formas canónicas y estándar: Minitérminos y maxitérminos

Por ejemplo, la función $f1$ de la tabla de verdad:

x	y	z	Función f_1	Función f_2
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

m_0 M_0

m_7 M_7

se obtiene expresando las combinaciones 001, 100 y 111 como $x'y'z$, $xy'z'$ y xyz, respectivamente. Puesto que cada uno de estos minitérminos hace que $f1 = 1$, tenemos: $f1 = x'y'z + xy'z' + xyz = m_1 + m_4 + m_7$.

hace 1

función booleana

Se tiene que: Cualquier función booleana se puede expresar como una suma de *minitérminos o producto estándar* (donde "suma" se refiere al OR de los términos).

Formas canónicas y estándar: Minitérminos y maxitérminos

Si se considera el complemento de una función, se toman los minitérminos para cada combinación que produce un 0 en la función, y haciendo después el OR de esos términos. El complemento de f1 se lee así: $f1' = x'y'z' + x'yz' + x'yz + xy'z + xyz'$. Ahora, obteniedo el complemento de $f1' = f1$ se tiene:

$$f1 = (x + y + z)(x + y' + z)(x' + y + z')(x' + y' + z)$$
$$= M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6$$

no hace 0

Se tiene que: Cualquier función booleana se puede expresar como un producto de *maxitérminos o sumas estándar* (donde "producto" se refiere a hacer el AND de los términos).

Formas canónicas y estándar: Minitérminos y maxitérminos

El procedimiento para obtener el producto de maxitérminos directamente de la tabla de verdad es el siguiente. Se forma un maxitérmino para cada combinación de las variables que produce un 0 en la función y luego se hace el AND de todos esos maxitérminos. Se dice que las funciones booleanas expresadas como suma de minitérminos o producto de maxitérminos están en **forma canónica**.

Formas canónicas y estándar: Minitérminos y maxitérminos I

Ejemplo: Exprese la función booleana $F = A + B'C$ como suma de minitérminos.

necesita
todos los terminos
vistos

A B C

Formas canónicas y estándar: Minitérminos y maxitérminos II

I

1 $A = A(B + B')$ Postulado 5

$= AB + AB'$ distributividad

$= AB(C + C') + AB'(C + C')$ Postulado 5

$= ABC + ABC' + AB'C + AB'C'$ distributividad

2 $B'C = B'C(A + A')$ Postulado 5

$= AB'C + A'B'C$ distributividad

3 $F = A + B'C$

$= ABC + ABC' + AB'C + AB'C' + AB'C + A'B'C$

$= ABC + ABC' + AB'C + AB'C' + A'B'C$ teorema 1

$\boxed{= m_1 + m_4 + m_5 + m_6 + m_7}$

Formas canónicas y estándar: Minitérminos y maxitérminos III

También se puede expresar como: $F(A, B, C) = \sum(1, 4, 5, 6, 7)$.

Ejemplo: Exprese la función booleana $F = xy + x'z$ en forma de producto de maxitérminos. Primero, se convierte la función en términos OR empleando la ley distributiva:

$$
\begin{aligned}
&= (xy + x')(xy + z) \text{ distributividad} \\
&= (x \cancel{+} x')(y + x')(x + z)(y + z) \text{ distributividad} \\
&= (x' + y)(x + z)(y + z) \text{ Postulado 5}
\end{aligned}
$$

falta una variable

La función tiene tres variables, x, y y z. A cada término OR le falta una variable; por tanto, utilizando el postulado 5 y distributividad:

Formas canónicas y estándar: Minitérminos y maxitérminos IV

$$x' + y = x' + y + zz' = (x' + y + z)(x' + y + z')$$
$$x + z = x + z + yy' = (x + y + z)(x + y' + z)$$
$$y + z = y + z + xx' = (x + y + z)(x' + y + z)$$
$$F = (x + y + z)(x + y' + z)(x' + y + z)(x' + y + z')$$
$$F = M_0 M_2 M_4 M_5$$
$$F(x, y, z) = \prod(0, 2, 4, 5)$$

Conversión en formas canónica y estándar

El complemento de una función expresado como la suma de minitérminos es igual a la suma de los minitérminos que faltan en la función original.

$$F(A,B,C) = \sum(1,4,5,6,7)$$
$$F'(A,B,C) = \sum(0,2,3)$$

Ahora bien, si se determina el complemento de F' por el teorema de DeMorgan, se obtiene F en una forma distinta:

$$F = (m_0 + m_2 + m_3)' = m_0' m_2' m_3' = M_0 M_2 M_3 = \prod(0,2,3)$$

Entonces, se tiene que: $m_j' = M_j$

Conversión en formas canónica y estándar

Para convertir de una forma canónica a otra, intercambiamos los símbolos $\sum$ y $\prod$ e incluimos en la lista sólo los números que faltaban en la forma original. Para hallar los términos faltantes, debemos recordar que el número total de minitérminos o maxitérminos es 2^n, donde n es el número de variables binarias en la función. Por ejemplo:

$$F(x, y, z) = \sum(1, 3, 6, 7)$$

$$F(x, y, z) = \prod(0, 2, 4, 5)$$

Forma estándar

Otra forma de expresar funciones booleanas es en forma estándar. En esta configuración, los términos que forman la función podrían contener una, dos o cualquier número de literales. Hay dos tipos de formas estándar: la *suma de productos* y el *producto de sumas*.

$$F1 = y' + xy + x'yz'$$ forma estandar

$$F2 = x(y' + z)(x' + y + z)$$ forma estandar

Otras operaciones lógicas

Se ha señalado ya que hay 2^{2^n} funciones para n variables binarias. En el caso de dos variables, n=2, el número de posibles funciones booleanas es 16.

Otras operaciones lógicas

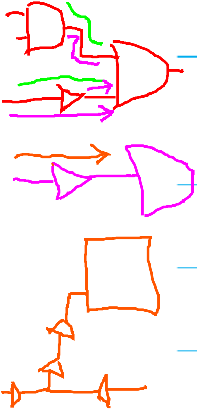
Se ha señalado ya que hay 2^{2n} funciones para n variables binarias. En el caso de dos variables, n=2, el número de posibles funciones booleanas es 16.

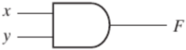
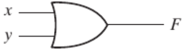
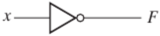
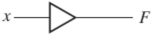
x	y	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Otras operaciones lógicas

Funciones booleanas	Símbolo de operador	Nombre	Comentarios
$F_0 = 0$		Nula	Constante binaria 0
$F_1 = xy$	$x \cdot y$	AND	x y y
$F_2 = xy'$	x/y	Inhibición	x, pero no y
$F_3 = x$		Transferencia	x'
$F_4 = x'y$	y/x	Inhibición	y, pero no x
$F_5 = y$		Transferencia	y
$F_6 = xy' + x'y$	$x \oplus y$	OR exclusivo	x o y, pero no ambas
$F_7 = x + y$	$x + y$	OR	x o y
$F_8 = (x + y)'$	$x \downarrow y$	NOR	No OR
$F_9 = xy + x'y'$	$(x \oplus y)'$	Equivalencia	x es igual a y
$F_{10} = y'$	y'	Complemento	No y
$F_{11} = x + y'$	$x \subset y$	Implicación	Si y, entonces x
$F_{12} = x'$	x'	Complemento	No x
$F_{13} = x' + y$	$x \supset y$	Implicación	Si x, entonces y
$F_{14} = (xy)'$	$x \uparrow y$	NAND	No AND
$F_{15} = 1$		Identidad	Constante binaria 1

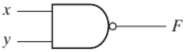
Compuertas lógicas



AND		$F = xy$	x y | F 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1
OR		$F = x + y$	x y | F 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1
Inversor		$F = x'$	x | F 0 | 1 1 | 0
Búfer		$F = x$	x | F 0 | 0 1 | 1

Compuertas lógicas

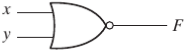
NAND



$F = (xy)'$

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

NOR



$F = (x + y)'$

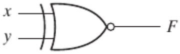
x	y	F
0	0	1
0	1	0
1	0	0
1	1	0

OR exclusivo (XOR)

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

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

NOR exclusivo o equivalencia



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

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

Compuertas lógicas con múltiples entradas

Las compuertas mencionadas anteriormente, 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.

Compuertas lógicas con múltiples entradas

Las compuertas mencionadas anteriormente, 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.

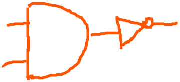
$F = (x + y + z + w)$

Los operadores NAND y NOR no son asociativos. Para superar este problema, definimos la compuerta NOR o NAND múltiple como una compuerta OR o AND complementada. Así, por definición, tenemos:

Compuertas lógicas con múltiples entradas

Las compuertas mencionadas anteriormente, 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.

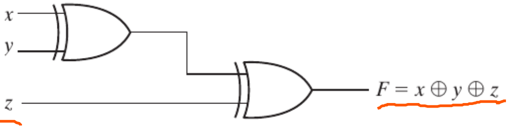
Los operadores NAND y NOR no son asociativos. Para superar este problema, definimos la compuerta NOR o NAND múltiple como una compuerta OR o AND complementada. Así, por definición, tenemos:



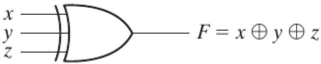
$$x \downarrow y \downarrow z = (x + y + z)'$$
$$x \uparrow y \uparrow z = (xyz)'$$

Compuertas lógicas con múltiples entradas

También se pueden tener la operación con mas entradas poniendo las operaciones de dos entradas en cascada.



a) Con compuertas de dos entradas

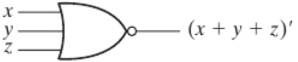


b) Compuerta de tres entradas

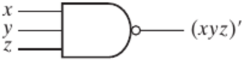
x	y	z	F
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

c) Tabla de verdad

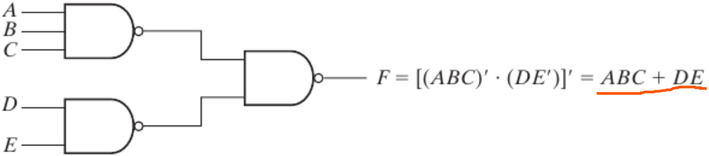
Compuertas lógicas con múltiples entradas



a) Compuerta NOR de 3 entradas



b) Compuerta NAND de 3 entradas



c) Compuertas NAND en cascada

Preguntas?