**ÁLGEBRA DE BOOLE Y COMPUERTAS**

**Introducción al álgebra de Boole**

Muchos componentes utilizados en sistemas de control, como contactores y relés, presentan dos estados claramente diferenciados (abierto o cerrado, conduce o no conduce). A este tipo de componentes se les denomina componentes todo o nada o también componentes lógicos.

Para estudiar de forma sistemática el comportamiento de estos elementos, se representan los dos estados por los símbolos 1 y 0 (0 abierto, 1 cerrado). De esta forma podemos utilizar una serie de leyes y propiedades comunes con independencia del componente en sí; da igual que sea una puerta lógica, un relé, un transistor, etc...

Atendiendo a este criterio, todos los elementos del tipo todo o nada son representables por una variable lógica, entendiendo como tal aquella que sólo puede tomar los valores 0 y 1. El conjunto de leyes y reglas de operación de variables lógicas se denomina álgebra de Boole, ya que fué George Boole quien desarrolló las bases de la lógica matemática.

**Operaciones lógicas básicas**

Sea un conjunto formado por sólo dos elementos que designaremos por 0 y 1. Llamaremos variables lógicas a las que toman sólo los valores del conjunto, es decir 0 o 1.   
En dicho conjunto se definen tres operaciones básicas:

SUMA LOGICA:

Denominada también operación "O" (OR). Esta operación responde a la siguiente tabla:

|  |  |  |
| --- | --- | --- |
| **a** | **b** | **a+b** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

PRODUCTO LOGICO:

Denominada también operación "Y" (AND). Esta operación responde a la siguiente tabla:

|  |  |  |
| --- | --- | --- |
| **a** | **b** | **a\*b** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

NEGACION LOGICA:

Denominada también operación "N" (NOT). Esta operación responde a la siguiente tabla:

|  |  |
| --- | --- |
| **a** | **a'** |
| 0 | 1 |
| 1 | 0 |

**Propiedades del álgebra de Boole**

Las propiedades del conjunto en el que se han definido las operaciones (+, \*, ') son las siguientes:

PROPIEDAD CONMUTATIVA:

De la suma: a+b = b+a  
Del producto: a\*b = b\*a

PROPIEDAD ASOCIATIVA:

De la suma: (a+b)+c = a+(b+c) = a+b+c  
Del producto: (a\*b)\*c = a\*(b\*c) = a\*b\*c

LEYES DE IDEMPOTENCIA:

De la suma: a+a = a ; a+a' = 1  
Del producto: a\*a = a ; a\*a' = 0

PROPIEDAD DISTRIBUTIVA:

De la suma respecto al producto: a\*(b+c) = (a\*b) + (a\*c)  
Del producto respecto a la suma: a + (b\*c) = (a+b) \* (a+c)

LEYES DE DE MORGAN:

(a+b+c)' = a'\*b'\*c'  
(a\*b\*c)' = a'+b'+c'

**Otras operaciones lógicas**

A partir de las operaciones lógicas básicas se pueden realizar otras operaciones booleanas, las cuales son:

NAND, cuya tabla correspondiente es:

|  |  |  |
| --- | --- | --- |
| **a** | **b** | **(a\*b)'** |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

NOR, cuya tabla correspondiente es:

|  |  |  |
| --- | --- | --- |
| **a** | **b** | **(a+b)'** |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

XOR, también llamada función OR-EXCLUSIVA. Responde a la tabla:

|  |  |  |
| --- | --- | --- |
| **a** | **b** | **a(+)b** |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

**Puertas lógicas**

Todas las funciones lógicas vistas hasta el momento poseen una representación normalizada, la cual se muestra en la figura siguiente:

|  |
| --- |
| http://www.profesormolina.com.ar/electronica/componentes/int/elec_digit/image001.gif |

Toda puerta lógica consta de 1 o más entradas y 1 o 2 salidas (puede darse el caso de proporcionarse la salida y su negada). En todos los símbolos las entradas se encuentran a la izquierda y las salidas a la derecha.

**Funciones lógicas**

La aplicación más directa de las puertas lógicas es la combinación entre dos o más de ellas para formar circuitos lógicos que responden a funciones lógicas. Una función lógica hace que una o más salidas tengan un determinado valor para un valor determinado de las entradas.

Supongamos que tenemos dos entradas, A y B, y una salida F. Vamos a hacer que la salida sea 1 lógico cuando A y B tengan el mismo valor, siendo 0 la salida si A y B son diferentes.

En primer lugar veamos los valores de A y B que hacen 1 la función:

A = 1 y B = 1   
A = 0 y B = 0

Es decir, podemos suponer dos funciones de respuesta para cada caso:

F1 = A\*B (A y B a 1 hacen F1 1)   
F2 = A'\*B' (A y B a 0 hacen F2 1)

La suma de estas funciones será la función lógica final que buscamos:

F = F1 + F2 = (A\*B)+(A'\*B')

A continuación vamos a ver como en muchos casos es posible simplificar la función lógica final en otra más simple sin alterar el funcionamiento del circuito.

**Simplificación de funciones**

Supongamos que tenemos un circuito donde "F" es la respuesta (salida) del mismo en función de las señales A, B, y C (entradas):

F = A\*B\*C + A'\*B\*C + B\*C

Esta función puede ser simplificable aplicando las propiedades del álgebra de Boole. En primer lugar aplicamos la propiedad distributiva:

F = B\*C\*(A+A') + B\*C

Ahora aplicamos las leyes de idempotencia:

F = B\*C + B\*C = B\*C

Como hemos podido ver en este ejemplo en muchas ocasiones se puede simplificar la función (y por tanto el circuito) sin que ello afecte al resultado. Más adelante veremos como simplificar funciones empleando otros métodos más sencillos y fiables.

**Tabla de verdad**

DEFINICION:

Es una forma de representación de una función en la que se indica el valor 0 o 1 para cada valor que toma ésta por cada una de las posibles combinaciones que las variables de entrada pueden tomar.

Anteriormente hemos visto las tablas de respuesta de cada una de las operaciones lógicas; estas tablas son tablas de verdad de sus correspondientes puertas lógicas.

La tabla de verdad es la herramienta que debemos emplear para obtener la forma canónica de la función del circuito, para así poder simplificar y conseguir la función más óptima.

Veamos un ejemplo de un circuito y la tabla de verdad correspondiente:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| http://www.profesormolina.com.ar/electronica/componentes/int/elec_digit/image002.gif | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **A** | **B** | **C** | **D** | **F** | | 0 | 0 | 0 | 0 | 1 | | 0 | 0 | 0 | 1 | 1 | | 0 | 0 | 1 | 0 | 1 | | 0 | 0 | 1 | 1 | 1 | | 0 | 1 | 0 | 0 | 1 | | 0 | 1 | 0 | 1 | 1 | | 0 | 1 | 1 | 0 | 1 | | 0 | 1 | 1 | 1 | 1 | | 1 | 0 | 0 | 0 | 1 | | 1 | 0 | 0 | 1 | 1 | | 1 | 0 | 1 | 0 | 1 | | 1 | 0 | 1 | 1 | 1 | | 1 | 1 | 0 | 0 | 1 | | 1 | 1 | 0 | 1 | 1 | | 1 | 1 | 1 | 0 | 1 | | 1 | 1 | 1 | 1 | 0 | |

Como podemos ver, si simplificamos la función obtenemos:

F = (A\*B\*C\*D)'

es decir, un puerta NAND de 4 entradas.