1 ÁLGEBRAS DE BOOLE DE 2 ELEMENTOS OPERACIONES BOOLEANAS Y PUERTAS LÓGICAS

La finalidad de la Electrónica Digital es procesar la información. Para ello utiliza las operaciones definidas por George Boole en su «investigación sobre las leyes del pensamiento», publicada en 1854. En una época de triunfo de las matemáticas en la tarea de «modelizar» el mundo físico, George Boole dio también forma matemática a la combinación de proposiciones; Boole introdujo, a la vez, un lenguaje formal (la lógica proposicional) y una estructura matemática (el álgebra de Boole) capaz de representar y de validar tal lenguaje.

Casi un siglo después, en 1938, al estudiar los complejos circuitos de relés que se utilizaban en la comunicación telefónica, Claude E. Shannon demostró que las operaciones booleanas son aptas para describir los circuitos con conmutadores y, también, para expresar cálculos en el sistema de numeración de base 2. Shannon estableció la posibilidad de utilizar la misma estructura matemática (el álgebra de Boole) como soporte de un sistema de numeración y cálculo (el sistema binario) y proporcionó una forma de «construir» las operaciones del álgebra booleana mediante la conexión de dispositivos físicos (los conmutadores).

Boole y Shannon fijaron los «cimientos» conceptuales para el procesamiento digital de la información. Gracias a ellos disponemos de un lenguaje formalizado capaz de expresar la combinación de proposiciones, de un sistema de numeración capaz de soportar cálculos aritméticos y de una forma de «materializar» (es decir, de construir máquinas capaces de ejecutar) tanto el lenguaje como el sistema de numeración.

La base matemática que soporta todo esto corresponde a la estructura de álgebra de Boole de dos elementos (el 0 y el 1): álgebra booleana binaria. Las «máquinas digitales», aunque solamente «saben» trabajar con el 0 y el 1 (una lógica dual muy limitada), son capaces de manejar, a más alto nivel (por programación), la lógica difusa, las redes neuronales, la inferencia matemática, la inteligencia artificial,...

Este tema presenta un resumen general de los conceptos fundamentales del álgebra de Boole y de sus operaciones, considerando en particular las tres álgebras binarias citadas: la lógica proposicional, el sistema de numeración con base 2 y el álgebra de conmutadores. Además, se expone la representación gráfica de las operaciones booleanas mediante puertas lógicas, como esquema para describir (y forma de construir) los circuitos digitales, y se introducen las operaciones «unitarias» que permiten expresar, con sólo una de ellas, todo el álgebra booleana y, por lo mismo, permiten construir cualquier circuito digital con un solo tipo de puertas.

Tomás Pollán Santamaría. Tecnología Electrónica. E.U.I.T.I.Z. Universidad de Zaragoza. tpollan@posta.unizar.es

1.1. Estructura de Álgebra de Boole

El álgebra de Boole es una estructura matemática definida sobre un conjunto de elementos $\{$**a, b, c,...**$\}$ por tres operaciones con las propiedades siguientes:

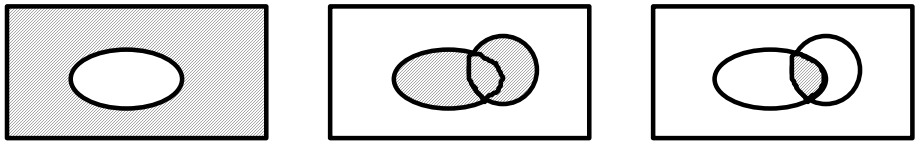
- la complementación o negación, $\bar{a}$, con propiedad de involución, $\bar{\bar{a}} = a$
- la operación "**o**", $a + b$, asociativa y conmutativa
- la operación "**y**", $a.b$, también asociativa y conmutativa
- siendo estas dos últimas operaciones distributivas entre sí

$$a \,.\, (b + c) = (a \,.\, b) + (a \,.\, c)$$
$$a + (b \,.\, c) = (a + b) \,.\, (a + c)$$

- y existiendo dos elementos únicos, **0** y **1**, tales que $\bar{0} = 1$ $\bar{1} = 0$ y

$$a + 0 = a \qquad a + 1 = 1 \qquad a + \bar{a} = 1$$
$$a \,.\, 0 = 0 \qquad a \,.\, 1 = a \qquad a \,.\, \bar{a} = 0$$

Un ejemplo característico de álgebra booleana lo constituye el conjunto partes de un conjunto dado (conjunto de todos sus subconjuntos) con las operaciones de complementación, unión e intersección; tales operaciones pueden representarse gráficamente mediante los diagramas de Venn.



$\bar{A}$ $A \cup B$ $A \cap B$

En relación con el cuerpo de los números reales, contrastando sus operaciones aritméticas de suma y resta con las operaciones "**o**" e "**y**" booleanas, el álgebra de Boole presenta las siguientes diferencias:

- La propiedad distributiva es doble; no sólo de **.** respecto a **+**, sino también de **+** respecto a **.**.
- No existen elementos inversos respecto a las operaciones "**o**" e "**y**" y por ello no están definidas las operaciones inversas (como son la resta y división aritméticas).
- Existe, en cambio, el elemento complementario.

[Considérese el diferente papel que desempeña el elemento complementario en relación con los elementos inversos: - complementario $a + \bar{a} = 1$ $a \,.\, \bar{a} = 0$
- inversos $a + (-a) = 0$ $a \,.\, (1/a) = 1$].

A partir de los axiomas que definen el álgebra de Boole pueden deducirse directamente, los siguientes teoremas operativos:

• Dualidad : toda expresión booleana sigue siendo válida si se efectúan, a la vez, los siguientes cambios: $a \leftrightarrow \bar{a}$ + (operación "**o**") $\leftrightarrow$ **.** (operación "**y**") **0** $\leftrightarrow$ **1**

• Idempotencia: $a + a = a$ $\quad$ $a \,.\, a = a$

• Absorción: $a + a.b = a$ $\quad$ $a + \bar{a}.b = a + b$

$a.(a + b) = a$ $\quad$ $a.(\bar{a} + b) = a.b$

• de Morgan: $\overline{a + b} = \bar{a} \,.\, \bar{b}$ $\quad$ $\overline{a \,.\, b} = \bar{a} + \bar{b}$

• de Consenso: $a.b + \bar{a}.c + b.c = a.b + \bar{a}.c$

$(a + b).(\bar{a} + c).(b + c) = (a + b).(\bar{a} + c)$

Como simple ejemplo de demostración de teoremas, se incluye a continuación la correspondiente a los teoremas de idempotencia:

$a + a = a.1 + a.1 = a.(1 + 1) = a.1 = a$

$a.a = (a + 0).(a + 0) = a + 0.0 = a + 0 = a$

[En el primer paso, se hace uso de los axiomas a.1=a, a+0=a; en el segundo, se aplica la propiedad distributiva; en el tercer paso, se utilizan los axiomas a+1=1, a.0=0 y en el cuarto, se emplean los mismos axiomas que en el primer paso.]

El teorema de Dualidad se deduce de que todos los axiomas siguen siendo válidos si se aplican sobre ellos dichos cambios (los tres a la vez) y, por tanto, tales cambios pueden generalizarse a cualquier expresión booleana.

[Debe tenerse en cuenta que la Dualidad ha de ser aplicada a ambos miembros de una expresión booleana y no solamente a uno de ellos:

sea la función $y = a + \bar{c}.b$

no es válido hacer $y = a + \bar{c}.b$ "$= \bar{a}.(c + \bar{b})$" sino $\bar{y} = \bar{a}.(c + \bar{b})$.]

El álgebra booleana más simple y de mayor interés práctico es la definida sobre un conjunto de sólo dos elementos, que necesariamente han de ser el **0** y el **1**:

negación	operación "**o**"	**a+b**	operación "**y**"	**a.b**
$\bar{0} = 1$	**0 + 0**	**0**	**0 . 0**	**0**
	0 + 1	**1**	**0 . 1**	**0**
$\bar{1} = 0$	**1 + 0**	**1**	**1 . 0**	**0**
	1 + 1	**1**	**1 . 1**	**1**

En la operación "**o**" predomina el valor **1**, en el sentido de que si un operando tiene dicho valor **1**, el resultado también es **1**; mientras que para la operación "**y**" el valor que prevalece es el **0**.

Una operación compuesta muy frecuente en el álgebra binaria es la denominada "**o-exclusiva**" $a \oplus b$, que corresponde a la función «ser diferente» o «desigualdad»; la denominación "**o-exclusiva**" deriva de que esta operación coincide con la "**o**", salvo en el caso **11** que adopta el valor **0**:

$0 \oplus 0 = 0$; $0 \oplus 1 = 1$; $1 \oplus 0 = 1$; $1 \oplus 1 = 0$

$a \oplus b$ vale **1** cuando $a \neq b$ y vale **0** cuando $a = b$,

es decir, $a \oplus b = 1$ si $a = 0$ y $b = 1$ o, también, si $a = 1$ y $b = 0$

o sea, $a \oplus b = \bar{a}.b + a.\bar{b}$.

La expresión $\bar{a}.b + a.\bar{b}$ representa la función "**o-exclusiva**" en términos de operaciones booleanas básicas, en la forma de «suma de productos»; también puede expresarse dicha operación como «producto de sumas», en la forma siguiente:

$$a \oplus b = \bar{a}.b + a.\bar{b} = \overline{\overline{\bar{a}.b + a.\bar{b}}} = \overline{\overline{(\bar{a}.b)} \,.\, \overline{(a.\bar{b})}} = \overline{(a + \bar{b}).(\bar{a} + b)} = \overline{a.\bar{a} + a.b + \bar{b}.\bar{a} + \bar{b}.b} =$$

$$= \overline{a.b + \bar{b}.\bar{a}} = \overline{(a.b)} \,.\, \overline{(\bar{b}.\bar{a})} = (\bar{a} + \bar{b}) \,.\, (b + a) = (\bar{a} + \bar{b}) \,.\, (a + b)$$

1.2. Casos de interés de Álgebras de Boole binarias

1.2.1. Lógica Proposicional

Entenderemos por proposición toda frase (afirmación o negación) que admite la asignación de valores verdad (**1**) y mentira (**0**). La lógica proposicional trata de la combinación de proposiciones para formar proposiciones compuestas; en tal sentido, deduce nuevas proposiciones a partir de las iniciales.

Las funciones básicas de la lógica proposicional son: negación (**no a**), disyunción (**a ó b**), conjunción (**a y b**), implicación ($\mathbf{a \Rightarrow b}$) y equivalencia ($\mathbf{a \equiv b}$). Las tres primeras operaciones coinciden, respectivamente, con las operaciones básicas booleanas (negación, "**o**" e "**y**") y las otras dos pueden expresarse en la siguiente forma:

• la implicación $\mathbf{a \Rightarrow b}$ equivale a la expresión $a.\bar{b} = 0$
(suceder **a** y que no suceda **b** es falso)
o a la expresión dual $\bar{a} + b = 1$
(o no sucede **a** o necesariamente sucede **b**).

• la equivalencia $\mathbf{a \equiv b}$ equivale a la expresión $a.b + \bar{a}.\bar{b} = 1$
(**a** y **b** son, a la vez, ciertas o son ambas falsas).

La lógica proposicional constituye un lenguaje formalizado, capaz de configurar decisiones lógicas a partir de la combinación de proposiciones; tal lenguaje puede ser expresado mediante las funciones del álgebra booleana de dos elementos.

Ejemplos:

a) A Dios rogando y con el mazo dando: $rD \cdot dm = 1$

(rD : rogar a Dios; dm : dar con el mazo).

b) O me toca la lotería o me pego un tiro $lot + tir = 1$

(lot : tener suerte en la lotería; tir : pegarme un tiro).

c) En el país de los ciegos el tuerto es rey: $pc.t \Rightarrow R$ $\quad pc \cdot t \cdot \overline{R} = 0$

o también $\quad \overline{pc} + \overline{t} + R = 1$

(pc : estar en el país de los ciegos; t : ser tuerto; R : ser rey).

d) Todo hombre es mortal $h \Rightarrow m$ $\quad h \cdot \overline{m} = 0$

Juan es hombre $j \Rightarrow h$ $\quad J \cdot \overline{h} = 0$

Luego Juan es mortal $¿ j \Rightarrow m ?$ $\quad ¿ J \cdot \overline{m} = 0 ?$

(Efectuando la operación "o" de las dos premisas $h \cdot \overline{m} + J \cdot \overline{h} = 0$

y por el teorema de consenso $h \cdot \overline{m} + \overline{h} \cdot J + \overline{m} \cdot J = 0$

para lo cual han de ser nulos todos los «sumandos» $J \cdot \overline{m} = 0$ *).*

e) ***p*** *es una condición necesaria y suficiente para* ***q :*** $\boldsymbol{p \equiv q}$ $\quad p \cdot q + \overline{p} \cdot \overline{q} = 1.$

f) Ser aragonés y ser tozudo es una misma cosa : $\boldsymbol{a \equiv t}$ $\quad a \cdot t + \overline{a} \cdot \overline{t} = 1$

(todos los aragoneses son tozudos y todas las personas tozudas son aragonesas).

g) Me casaré con una persona joven e inteligente o que tenga fortuna y no ronque

$c = j \cdot i + f \cdot \overline{r}$

(c : decisión de casarme; j : ser joven; i : inteligente; f : rico; r : roncar).

h) La alarma de un automóvil suena cuando la llave de contacto está puesta y no lo está el cinturón de seguridad y, también, con el cinturón de seguridad puesto si la puerta del conductor está abierta $a = ll \cdot \overline{c} + c \cdot \overline{p}$

(a : alarma; ll : llave de contacto puesta; c : cinturón de seguridad puesto; p : puerta del conductor cerrada).

i) En una oficina trabajan cuatro personas y han decidido que la musica ambiental estará activada solamente cuando lo deseen tres de ellas

$m = p1.p2.p3 + p1.p2.p4 + p1.p3.p4 + p2.p3.p4$

(m : suena la música;

p1,p2,p3,p4 : la correspondiente persona -1,2,3,4- desea escuchar música).

1.2.2. Sistema binario de numeración

El sistema de numeración en base 2 emplea solamente los dígitos **0** y **1**, siendo 2^n el valor relativo de la cifra que ocupa el lugar n (contado de derecha a izquierda, partiendo de 0).

El esquema operativo de los cambios de base binario-decimal es el siguiente:

$$e\ d\ c\ b\ a\ _{(base\ 2} = a.2^0 + b.2^1 + c.2^2 + d.2^3 + e.2^4\ _{(base\ 10}$$

Ejemplo:

$$10011000100101_{(2} = 1x1 + 0x2 + 1x4 + 0x8 + 0x16 + 1x32 + 0x64 + 0x128 + 0x256 + 1x512 + 1x1024 + 0x2048 + 0x4096 + 1x8192 = 9765_{(10}$$

$n^o{}_{(10}$	2	*Ejemplo:*	*cocientes*	*restos*
r_1	c_1	2	*9765*	*1*
r_2	c_2	2	*4882*	*0*
r_3	c_3	2	*2441*	*1*
r_4	c_4	2	*1220*	*0*
r_5	c_5	2	*610*	*0*
r_6	1		*305*	*1*
			152	*0*
			76	*0*
			38	*0*
			19	*1*
			9	*1*
			4	*0*
			2	*0*
			1	*1*

$n^o{}_{(base\ 10} = 1\ r_6\ r_5\ r_4\ r_3\ r_2\ r_1\ _{(base\ 2}$

$9765_{(10} = 10011000100101_{(2}$

Cualquier operación de un sistema de numeración puede realizarse mediante un algoritmo que contenga solamente sumas, restas y comparaciones. Ahora bien, en el caso del sistema binario estas tres operaciones pueden construirse con funciones booleanas sencillas; véase a continuación para dos números binarios de 1 solo dígito:

suma:

a	b	suma	acarreo
0	**0**	**0**	**0**
0	**1**	**1**	**0**
1	**0**	**1**	**0**
1	**1**	**0**	**1**

acarreo = «me llevo»

suma = $\bar{a}.b + a.\bar{b}$

$= a \oplus b$

acarreo = a.b

resta:

a	b	resta	acarreo
0	**0**	**0**	**0**
0	**1**	**1**	**1**
1	**0**	**1**	**0**
1	**1**	**0**	**0**

acarreo = «me llevo»

resta = $\bar{a}.b + a.\bar{b}$

$= a \oplus b$

acarreo = $\bar{a}.b$

comparación :

a	b	igual	mayor	menor
0	**0**	**1**	**0**	**0**
0	**1**	**0**	**0**	**1**
1	**0**	**0**	**1**	**0**
1	**1**	**1**	**0**	**0**

igual = $\bar{a}.\bar{b} + a.b$

$= \overline{a \oplus b}$

mayor = $a.\bar{b}$

menor = $\bar{a}.b$

Obviamente las anteriores expresiones booleanas para la suma, resta y comparación de dos dígitos pueden extenderse a números binarios de más de 1 dígito; precisamente en el capítulo 3 dedicado a bloques aritméticos se estudian sumadores, restadores y comparadores para números binarios de **n** dígitos.

La expresión de estas tres operaciones aritméticas en términos de operaciones booleanas implica que cualquier cálculo en sistema binario puede realizarse algorítmicamente mediante operaciones del álgebra de Boole de dos elementos.

Sea **dcba** un número binario (base 2): **a** es la cifra de las unidades; el valor relativo de **b** es 2 y podemos referirnos a tal cifra como «dosenas»; **c** puede ser nominada como «cuatrenas» ya que su valor relativo es 4 y **d** cuyo valor relativo es 8 puede ser llamada «octenas».

Habrá 16 números de 4 dígitos, desde el **0000** al **1111** (del 0 al 15 en base 10). Como regla general, toda la numeración digital comienza por 0 = **000…**, no por 1, de forma que **n** dígitos o bits contienen 2^n posibilidades que irán de 0 a 2^n - 1.

1.3.3. Álgebra de Conmutadores

Un conmutador o interruptor es un sistema de dos estados:

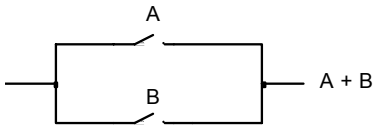
- cerrado, permitiendo el paso a su través
- abierto, interrumpiendo dicho paso.

Consideremos un conjunto de interruptores controlados de alguna forma y asignemos a cada interruptor una letra **A**, **B**, **C**,...:

• el símbolo A representa a todos aquellos interruptores, relacionados entre sí, que se encuentran en todo momento en el mismo estado;

• a su vez, se denotan con $\bar{A}$ aquellos interruptores que, en todo momento, se encuentran en estado contrario al que tienen los interruptores **A** en dicho momento; la negación o inversión se realiza mediante algún mecanismo que obliga a un interruptor a adoptar el estado contrario al de otro interruptor.

La operación "**o**" corresponde a conectar dos interruptores en paralelo mientras que la operación "**y**" supone disponerlos en serie:



A partir de estas operaciones y del concepto de negación indicado puede describirse cualquier combinación de interruptores, por compleja que sea la malla que dichos interruptores configuren.

Ejemplo:

Consideremos el caso de dos interruptores eléctricos conmutados: al accionar cualesquiera de ellos cambia el estado (encendida-apagada) de la lámpara conectada al conjunto; su tabla de verdad puede ser la siguiente:

A	*B*	*luz*
0	*0*	*1*
0	*1*	*0*
1	*0*	*0*
1	*1*	*1*

$$luz = \overline{A}.\overline{B} + A.B$$

Esta función corresponde a la combinación de conmutadores de la figura a, pero también puede construirse en la forma representada en la figura b que es el esquema circuital correspondiente a los interruptores conmutados comerciales.

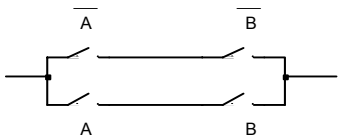


figura a

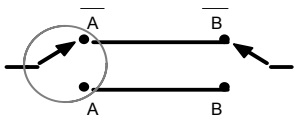


figura b

El álgebra de conmutadores puede realizarse físicamente mediante dispositivos muy diversos: mecánicos, fluídicos, neumáticos, relés, electrónicos..., que constituyen diversas formas de materializar el álgebra de Boole de dos elementos.

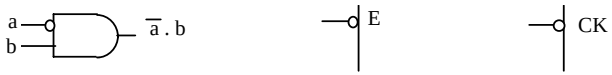
Existe un isomorfismo en cuanto a estructura matemática entre el sistema binario de numeración, la lógica proposicional y el álgebra de conmutadores, como álgebras booleanas que son los tres. De dicho isomorfismo se deduce que *el hombre dispone de mecanismos físicos capaces de realizar cálculos numéricos en el sistema binario y de automatizar decisiones lógicas por combinación de proposiciones en el marco de la lógica proposicional*.

Este isomorfismo estructural entre los conmutadores como dispositivos físicos, el sistema binario como sistema de numeración en el que realizar cálculos y la lógica proposicional como lenguaje formal en el que desarrollar deducciones proporciona a las técnicas digitales su magnífica capacidad para procesar información.

1.3. Representación de las operaciones booleanas: puertas lógicas

Las operaciones básicas del álgebra de Boole (negación, "**o**", "**y**") se representan gráficamente en la forma que sigue:

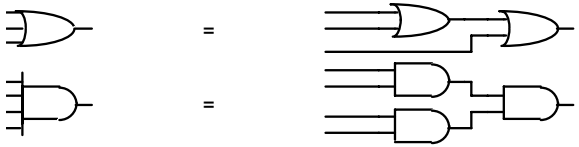
En el primer caso, es el círculo el que expresa directamente la negación o inversión; a veces se utiliza dicho círculo sin más para indicar una negación, como por ejemplo en las siguientes figuras:



En el caso de una entrada de habilitación **E**, la presencia de un círculo en la misma indica que se activa con **0**, en lugar de con **1** (la habilitación se produce cuando **E = 0**); si es una entrada de reloj **CK**, el círculo previo indica que dicho reloj actúa en su flanco negativo (bajadas ↓), en lugar del flanco positivo (subidas ↑).

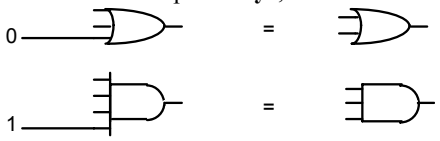
Los símbolos que representan gráficamente a las operaciones booleanas reciben el nombre de puertas lógicas. En tal sentido las tres primeras figuras son, respectivamente, un inversor, una puerta "**o**" y una puerta "**y**". También se emplea el nombre de puerta lógica para designar la realización física de tales operaciones, en particular cuando se construyen con dispositivos electrónicos.

La propiedad asociativa permite utilizar puertas "**o**" y puertas "**y**" de tres o más entradas, que expresan la realización sucesiva de la operación correspondiente.



Habida cuenta de que en la operación "**o**" predomina el valor **1**, las puertas "**o**" quedan «trabadas» con valor **1** en la salida si una de sus entradas se fija a **1**; lo mismo sucede en las puertas "**y**" respecto al valor **0**.

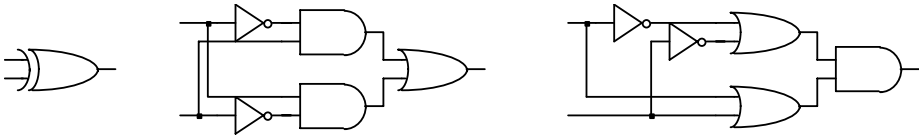
En cambio, la forma de «anular» una entrada de una puerta "**o**" consiste en conectarla a valor **0** y, en el caso de una puerta "**y**", a valor **1**.



Una puerta "**y**" de dos entradas **a b** puede ser utilizada para «controlar el paso» de una de ellas **a**, permitiéndolo o interrumpiéndolo según que el valor de la otra entrada **b** sea **1** ó **0**; la puerta "**y**" actúa como interruptor controlado por **b**, que «deja pasar» la señal **a** cuando **b = 1** y no le permite el paso (salida **0**) cuando **b = 0**.

Una puerta "**o-exclusiva**" (función «ser diferente» o «desigualdad») se representa con el símbolo de la figura y equivale a la combinación de cinco puertas básicas:

$$a \oplus b = \bar{a}.b + a.\bar{b} = (\bar{a} + \bar{b}).(a + b)$$



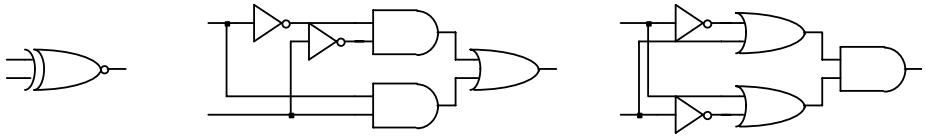
La puerta **o-exclusiva** puede ser utilizada como «inversor de **a** controlado por **b**»:

- cuando **b = 0**, $y = a \oplus 0 = \bar{a}.0 + a.\bar{0} = a$, la señal **a** pasa directamente,

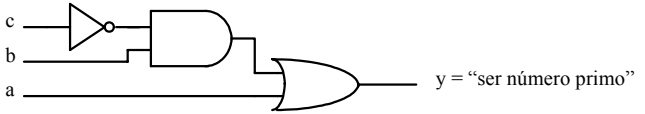
- para **b = 1**, $y = a \oplus 1 = \bar{a}.1 + a.\bar{1} = \bar{a}$, la señal **a** pasa en forma invertida.

La negación de la puerta "**o-exclusiva**" se denomina "**y-inclusiva**" (porque coincide con la operación "**y**" incluyendo el caso **00**) y corresponde a la función «igualdad»: da resultado **1** cuando los dos operandos son iguales **00**, **11.**

$$a \otimes b = \bar{a}.\bar{b} + a.b = (\bar{a} + b).(a + \bar{b})$$



Veamos una función booleana sencilla: para números binarios de 3 dígitos **bca** (del 0 al 7) la condición "ser número primo" afecta a todos los impares (**a = 1**) y al número 2 (**010**), resultando la función siguiente: $y = a + \bar{c}.b.\bar{a} = a + \bar{c}.b$



1.4. Operaciones unitarias

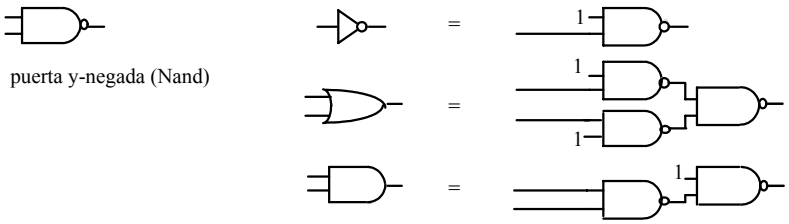
Aunque en la definición del álgebra de Boole se establecen tres operaciones básicas, existe un tipo especial de operaciones que podemos llamar «unitarias» porque con sólo una de ellas se puede realizar toda el álgebra; las principales operaciones unitarias son la operación "**y-negada**" (*Nand*) y la operación "**o-negada**" (*Nor*).

• operación "**y-negada**" (*Nand*): $a * b = \overline{a \,.\, b}$

$$\bar{a} = \overline{a \,.\, 1} = a * 1$$

$$a + b = \overline{\bar{a} \,.\, \bar{b}} = \bar{a} * \bar{b} = (a * 1) * (b * 1)$$

$$a \,.\, b = \overline{\overline{a \,.\, b}} = \overline{a * b} = (a * b) * 1$$

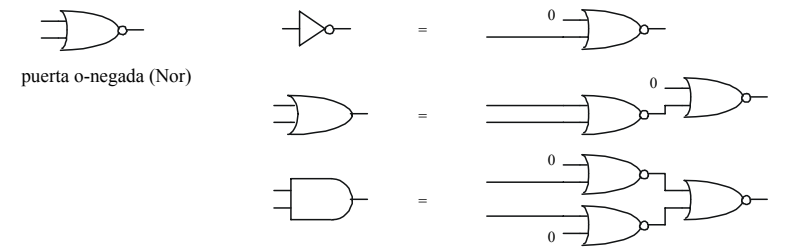


• operación "**o-negada**" (*Nor*): $a \Delta b = \overline{a + b}$

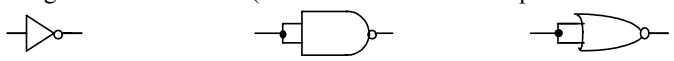
$$\bar{a} = \overline{a + 0} = a \Delta 0$$

$$a + b = \overline{\overline{a + b}} = \overline{a \Delta b} = (a \Delta b) \Delta 0$$

$$a \,.\, b = \overline{\bar{a} + \bar{b}} = \bar{a} \Delta \bar{b} = (a \Delta 0) \Delta (b \Delta 0)$$



Comentario: la inversión puede expresarse según la figura siguiente, pero (aunque «en el papel» no hay diferencia con la anterior) técnicamente es peor solución porque supone una «carga» de dos entradas (con doble intensidad o capacidad de entrada).

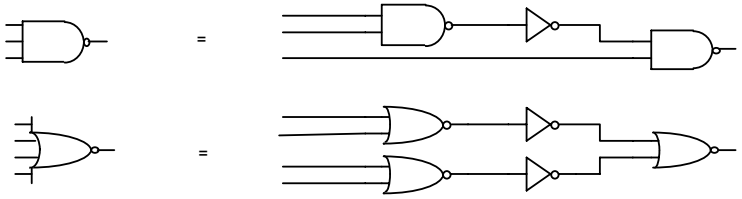


También otras operaciones como $\bar{a} \,.\, b$ o $\bar{a} + b$ son unitarias, pero no son utilizadas por no poseer la propiedad conmutativa.

Las operaciones "**y-negada**" y "**o-negada**" son conmutativas pero no asociativas:

$$(a*b)*c \neq a*(b*c) \qquad (a\Delta b)\Delta c \neq a\Delta(b\Delta c)$$

Por ello, las correspondientes puertas "**o-negada**" e "**y-negada**" de tres o más entradas no equivalen directamente a la realización sucesiva de tales operaciones con dos entradas; es preciso añadir una inversión intermedia en cada paso (tal inversión restituye la propiedad asociativa al conjunto, al negar la operación "**o/y-negada**" y convertirla en operación "**o/y**"):



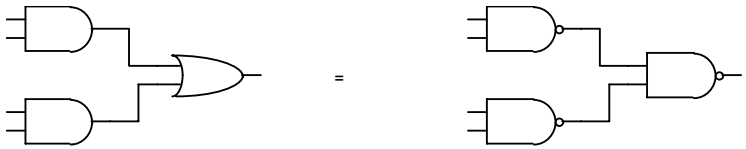
El álgebra de Boole realizada con una de estas operaciones unitarias es muy uniforme (solamente utiliza un tipo de puertas lógicas), pero menos potente por carecer de la propiedad asociativa. Esta carencia determina la ineludible necesidad de colocar los correspondientes paréntesis:

- las expresiones $a*b*c$ y $a\Delta b\Delta c$ carecen de sentido, pues no indican el orden en que deben aplicarse las dos operaciones contenidas en ellas;
- las puertas de tres o más entradas deben expresarse como **Nand(a,b,c)** y **Nor(a,b,c)**, respectivamente.

La operación "**y-negada**" se acomoda bien a expresiones del tipo «suma de productos», **a.b + c.d**, ya que pueden convertirse directamente en la forma que sigue:

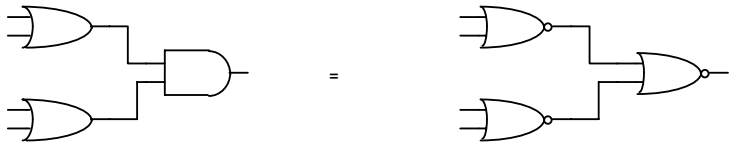
$$a.b + c.d = \overline{\overline{a.b + c.d}} = \overline{\overline{(a.b)}.\overline{(c.d)}} = (a*b)*(c*d)$$

[En las dos expresiones, el número de puertas y la conexión entre ellas son idénticos; basta sustituir directamente todas las puertas "**o**" e "**y**" por puertas "**y-negada**"].



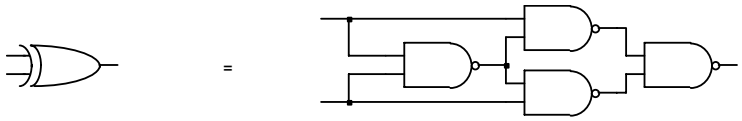
En cambio, la operación "**o-negada**" se acomoda bien a expresiones del tipo «producto de sumas», **(a+b) . (c+d)**:

$$(a+b).(c+d) = \overline{\overline{(a+b).(c+d)}} = \overline{\overline{(a+b)} + \overline{(c+d)}} = (a\Delta b)\Delta(c\Delta d)$$



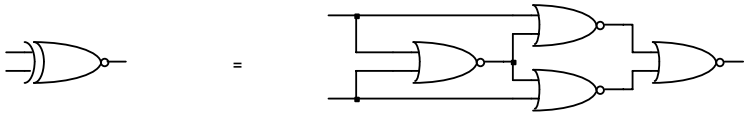
La operación "**o-exclusiva**" (*ser diferentes*) puede construirse con solamente cuatro puertas "**y-negada**" mediante la siguiente transformación:

$$a \oplus b = \bar{a}.b + a.\bar{b} = \bar{a}.b + a.\bar{b} + a.\bar{a} + b.\bar{b} = a.(\bar{a} + \bar{b}) + b.(\bar{a} + \bar{b}) =$$
$$= a.\overline{(a.b)} + b.\overline{(a.b)} = a.(a*b) + b.(a*b) = \left[a*(a*b)\right]*\left[b*(a*b)\right]$$



Y, del mismo modo la operación "**y-inclusiva**" (*igualdad*) puede construirse con solamente cuatro puertas "**o-negada**" mediante la transformación:

$$a \otimes b = (\bar{a} + b).(a + \bar{b}) = (\bar{a} + b).(a + \bar{b}).(a + \bar{a}).(b + \bar{b}) = (a + \bar{a}.\bar{b}).(b + \bar{a}.\bar{b}) =$$
$$= [a + (a\Delta b)].[b + (a\Delta b)] = \left[a\Delta(a\Delta b)\right]\Delta\left[b\Delta(a\Delta b)\right]$$

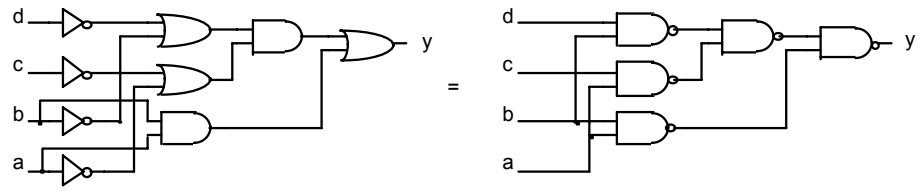


Veamos un par de ejemplos de transformación de expresiones booleanas con operaciones básicas a expresiones con puertas "**y-negada**":

$$y_1 = \text{"ser número primo de 3 dígitos"} = a + \bar{c}.b = \overline{\overline{a + \bar{c}.b}} = \bar{a}.\overline{(\bar{c}.b)} = \bar{a}*(\bar{c}*b)$$

Por último, sea la siguiente función y_2:

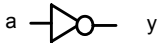
$$y_2 = \bar{d}.\bar{c} + \bar{d}.\bar{a} + \bar{c}.\bar{b} + b.a + \bar{b}.\bar{a} = \bar{d}.(\bar{c} + \bar{a}) + \bar{b}.(\bar{c} + \bar{a}) + b.a =$$
$$= (\bar{c} + \bar{a}).(\bar{d} + \bar{b}) + b.a = \overline{\overline{(\bar{c} + \bar{a}).(\bar{d} + \bar{b})} + b.a} = \overline{[(\bar{c} + \bar{a}).(\bar{d} + \bar{b})]}.\overline{(b.a)} =$$
$$= [(\bar{c} + \bar{a}) * (\bar{d} + \bar{b})] * (b*a) = [\overline{\overline{(\bar{c} + \bar{a})} * \overline{(\bar{d} + \bar{b})}}] * (b*a) =$$
$$= [\overline{(\bar{c}.a)} * \overline{(\bar{d}.b)}] * (b*a) = [(c*a)*(d*b)] * (b*a)$$



1.5 Las operaciones booleanas en lenguaje de descripción circuital: VHDL

Actualmente el diseño de circuitos digitales no se expresa en forma gráfica sino en forma de texto, a través de lenguajes de descripción circuital; el más utilizado de tales lenguajes es el VHDL.

A fin de desarrollar una aproximación gradual al diseño con lenguajes de descripción circuital, se incluye aquí la forma de describir las operaciones lógicas en VHDL; asimismo, al final del capítulo 4 se incluye la descripción de bloques combinacionales en VHDL.



a) y <= not a;
b) y <= '1' when a = '0' else '0';

a) y <= a or b;
b) y <= a when b = '0' else '1';

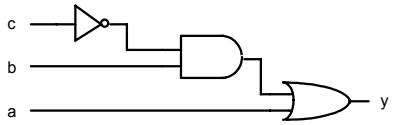
a) y <= a and b;
b) y <= a when b = '1' else '0';

a) y <= a xor b;
b) y <= not a when b = '1' else a;
c) y <= '0' when a = b else '1';

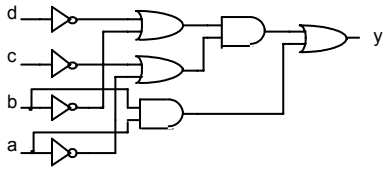
a) y <= a nand b;
b) y <= '0' when (a = '1') and (b = '1') else '1';

a) y <= a nor b;
b) y <= '1' when (a = '0') and (b = '0') else '0';

a) y <= not (a and b and c);
y <= '0' when (a = '1') and (b = '1') and (c = '1') else '1';



y <= a or (b and not c);



y <= ((not d or not b) and (not c or not a))
or (b and a);

y <= not ((d and b) or (c and a))
or (b and a);