PRINCIPIOS DE ELECTRÓNICA DIGITAL

La electrónica digital es una herramienta muy importante en los sistemas de control industriales, procesos de datos e infinidad de equipos como son: calculadoras electrónicas, vídeo juegos, ordenadores, telefonía móvil, etc. Sin embargo la lógica en que se basa, o lógica de conmutación, basada a su vez en el álgebra de Boole, está siendo rápidamente suplantada, en diferentes campos de aplicación, por la lógica denominada Fuzzy o lógica difusa.

ÁLGEBRA DE BOOLE

Es la herramienta fundamental de la electrónica digital, constituyendo su base matemática. El álgebra de Boole es un conjunto que consta de dos elementos **0** y **1** que no siempre representan números. Pueden ser:

0	$\Rightarrow$	Falso	$\Rightarrow$	Apagado	$\Rightarrow$	No tensión	$\Rightarrow$	Interruptor abierto	$\Rightarrow$	etc.
1	$\Rightarrow$	Verdadero	$\Rightarrow$	Encendido	$\Rightarrow$	Tensión	$\Rightarrow$	Interruptor cerrado	$\Rightarrow$	etc.

Operadores, postulados, propiedades, teoremas y leyes

Operadores		
Suma $\boldsymbol{a+b}$	Producto $\boldsymbol{a\cdot b}$	Complementación $\boldsymbol{a'}$ o $\boldsymbol{\overline{a}}$

Postulados		
Existe un complementario	$a+\overline{a}=1$	$a\cdot\overline{a}=0$
Idempotencia	$a+a=a$	$a\cdot a=a$
Existe un elemento neutro	$a+0=a$	$a\cdot 0=0$
Dominio del 0 y del 1	$a+1=1$	$a\cdot 1=a$
Doble complementación	$\overline{\overline{a}}=a$	

Propiedades		
conmutativa	$a+b=b+a$	$a\cdot b=b\cdot a$
distributiva	$a+b\cdot c=(a+b)\cdot(a+c)$	$a\cdot(b+c)=a\cdot b+a\cdot c$
asociativa	$a\cdot(b\cdot c)=(a\cdot b)\cdot c=a\cdot b\cdot c$	$a+(b+c)=(a+b)+c=a+b+c$

Teoremas		
Absorción	$a+(a\cdot b)=a$	$a\cdot(a+b)=a$
Unicidad de complementario	$a=1 \xrightarrow{\text{será}} \overline{a}=0$	$a \xrightarrow{\text{sólo}} \overline{a}$
	$\overline{a}=1 \xrightarrow{\text{será}} a=0$	$\overline{a} \xrightarrow{\text{sólo}} a$
Dualidad	$a\cdot\overline{b}+\overline{a}\cdot b \equiv \overline{(\overline{a}+b)\cdot(a+\overline{b})}$	

Leyes de De Morgan	
$\overline{a\cdot b\cdot c\cdot d}=\overline{a}+\overline{b}+\overline{c}+\overline{d}$	$\overline{a+b+c+d}=\overline{a}\cdot\overline{b}\cdot\overline{c}\cdot\overline{d}$

Funciones algebraicas

Función lógica

Es una expresión algebraica en la que se relacionan entre sí las variables binarias por medio de operaciones básicas: producto lógico, suma lógica e inversión.

De forma general podemos expresar una función lógica de la forma:

$$f = f(a,b,c,\cdots)$$

El valor de ***f*** depende del valor de las variables ***a, b, c …***

Una función lógica podría ser: $f=\overline{a}\cdot b+a\cdot(\overline{b}+c)+\overline{c}\cdot\overline{a}\cdot(\overline{a+\overline{b}})$. Las variables pueden tomar los valores **0** o **1**. Si a una variable le asignamos el valor $a=1$ la variable complementada es $\overline{a}=0$, pero si asignamos $a=0$ entonces $\overline{a}=1$.

De una función lógica, se dice, **que la función se hace cierta para las combinaciones de las variables que expresa**.

Función canónica

Es una expresión en la que todos sus términos contienen todas las variables, bien de forma directa o complementada.

Se denomina **minterm**, al término expresado como productos de las variables, y **maxterm** al expresado como sumas.

Es un minterm: $a\cdot\overline{b}\cdot c\cdot\overline{d}$. Es un maxterm: $\overline{a}+\overline{b}+c+\overline{d}$

Una función expresada en minterms:

$$f=\overline{a}\cdot\overline{b}\cdot\overline{c}\cdot\overline{d}+\overline{a}\cdot\overline{b}\cdot c\cdot\overline{d}+\overline{a}\cdot\overline{b}\cdot c\cdot d+a\cdot\overline{b}\cdot c\cdot\overline{d}+a\cdot b\cdot c\cdot d$$

Una función expresada en maxterms:

$$f=(\overline{a}+\overline{b}+c+\overline{d})\cdot(\overline{a}+b+\overline{c}+\overline{d})\cdot(a+b+\overline{c}+\overline{d})$$

El número máximo de términos que puede tener una función canónica será igual a $\mathbf{2^n}$, donde $\mathbf{n}$ es el número de variables.

Tabla de verdad

Es una relación ordenada donde se indican los términos canónicos que hacen verdadera la función. Se suele colocar en la primera columna el equivalente decimal del término, en la segunda columna los términos en binario y en la tercera se indican con **1** los que hacen verdadera la función y con **0** los que no.

Decimal	***a b c***	***f***
0	**0 0 0**	**1**
1	**0 0 1**	**1**
2	0 1 0	0
3	**0 1 1**	**1**
4	**1 0 0**	**1**
5	**1 0 1**	**1**
6	1 1 0	0
7	1 1 1	0

De la tabla de verdad indicada podemos obtener la función canónica:

$$f = f(0,1,3,4,5) = \overline{a}\cdot\overline{b}\cdot\overline{c} + \overline{a}\cdot\overline{b}\cdot c + \overline{a}\cdot b\cdot c + \overline{a}\cdot b\cdot c + \overline{a}\cdot b\cdot\overline{c}$$

Paso de función no canónica a canónica

Si tenemos una función no canónica, expresada como suma de productos, podemos convertirla en canónica multiplicando cada término por la suma de la variable que le falte en forma directa y complementada

$$f(a,b,c) = a + \overline{a}\cdot b + \overline{b}\cdot\overline{c} + a\cdot b\cdot c$$

$$f = a\cdot\left(b+\overline{b}\right)\cdot\left(c+\overline{c}\right) + \overline{a}\cdot b\cdot\left(c+\overline{c}\right) + \overline{b}\cdot\overline{c}\cdot\left(a+\overline{a}\right) + a\cdot b\cdot c$$

Deshaciendo los paréntesis tendremos:

$$f = abc + ab\overline{c} + a\overline{b}c + a\overline{b}\,\overline{c} + \overline{a}bc + \overline{a}b\overline{c} + a\overline{b}\,\overline{c} + \overline{a}\,\overline{b}\,\overline{c} + abc$$

Observamos si existen términos repetidos. Si así fuera los simplificamos, **dejando sólo uno de ellos**.

$$f = abc + ab\overline{c} + a\overline{b}c + a\overline{b}\,\overline{c} + \overline{a}bc + \overline{a}b\overline{c} + \cancel{a\overline{b}\,\overline{c}} + \overline{a}\,\overline{b}\,\overline{c} + \cancel{ab\overline{c}}$$

Quedando la función:

$$f = abc + ab\overline{c} + a\overline{b}c + a\overline{b}\,\overline{c} + \overline{a}bc + \overline{a}b\overline{c} + \overline{a}\,\overline{b}\,\overline{c}$$

En el caso que la función venga expresada como productos de sumas, a cada término le sumaremos el producto de la variable directa y complementada.

$$f(a,b,c) = \left(\overline{a} + \overline{b}\right)\cdot\left(\overline{a} + b\right)\cdot\left(\overline{a} + b + \overline{c}\right)$$

Para transformarla en canónica:

$$f(a,b,c) = \left(\overline{a} + \overline{b} + c\cdot\overline{c}\right)\cdot\left(\overline{a} + b + c\cdot\overline{c}\right)\cdot\left(\overline{a} + b + \overline{c}\right)$$

Aplicamos la propiedad distributiva a la función

$$f = \left(\overline{a} + \overline{b} + c\right)\cdot\left(\overline{a} + \overline{b} + \overline{c}\right)\cdot\left(\overline{a} + b + c\right)\cdot\left(\overline{a} + b + \overline{c}\right)\cdot\left(\overline{a} + b + \overline{c}\right)$$

y eliminamos los términos repetidos:

$$f = \left(\overline{a} + \overline{b} + c\right)\cdot\left(\overline{a} + \overline{b} + \overline{c}\right)\cdot\left(\overline{a} + b + c\right)\cdot\left(\overline{a} + b + \overline{c}\right)\cdot\left(\cancel{\overline{a} + b + \overline{c}}\right)$$

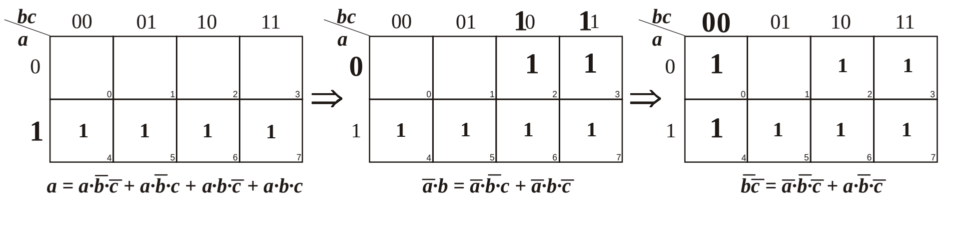
Sobre la tabla

Directamente sobre la tabla se pueden obtener los términos canónicos. Si suponemos la función:

$$f(a,b,c) = a + \overline{a}\cdot b + \overline{b}\cdot\overline{c}$$

tenemos tres términos no canónicos.

El término a comprenderá todos los términos canónicos que tengan dicha variable en forma directa. Sobre la tabla colocaremos un **1** en las casillas que le correspondan. Lo mismo haremos para los otros dos términos no canónicos.



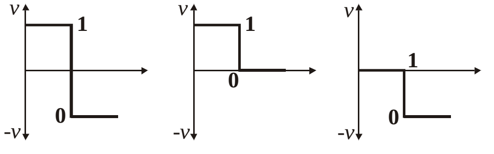
De esta forma, hemos ocupado todas las casillas que contempla la función.

ELECTRÓNICA DIGITAL

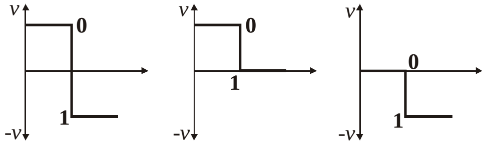
Lógica positiva y lógica negativa

Las variables lógicas sólo podrán tomar numéricamente los valores 0 y 1, pero eléctricamente estos dos valores vienen definidos por dos niveles de tensión bien distintos.

Debido a como se asignen estos niveles de tensión pueden aparecer dos tipos de lógica: lógica positiva y lógica negativa.



Si al 1 lógico se le asigna un valor de tensión más positivo que al 0 lógico, como en los casos representados, la lógica es positiva.



Si por el contrario el 1 lógico tiene un valor más negativo de tensión que el 0 lógico, la lógica es negativa.

Representación de operadores lógicos

Eléctricamente las operaciones del álgebra de Boole son realizables por medio de interruptores. Un interruptor puede tener sólo dos estados: abierto y cerrado.

Podemos asignar el estado abierto al estado directo de la variable y el estado cerrado al estado complementado de la misma (o a la inversa).

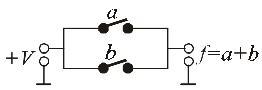
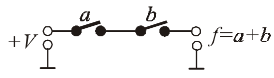
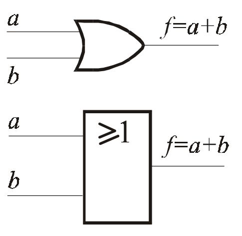
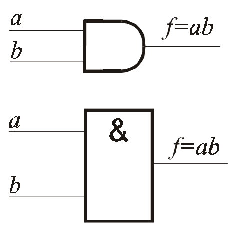
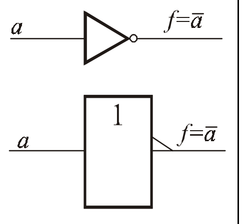
No se utiliza la simbología de contactos para representar, en electrónica, las operaciones lógicas. Utilizamos otros símbolos, que reciben el nombre de **puertas,** con formas diferentes para indicar el tipo de puerta en la simbología antigua y no estandarizada, y con igual forma, pero con indicaciones del tipo de puerta, en la simbología nueva y estandarizada según la norma IEC.

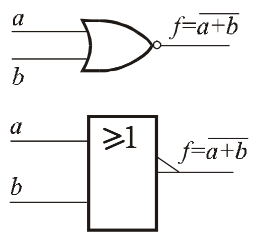
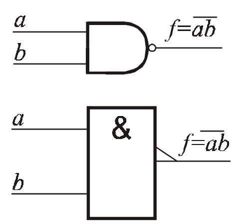
Las puertas indicadas son de dos entradas para que resulte más sencillo comprender su función. Existen puertas de mayor número de entradas.

El símbolo de complementación o inversión

En la simbología antigua o americana, la inversión se indica con un círculo tanto en las entradas como en las salidas de los símbolos.

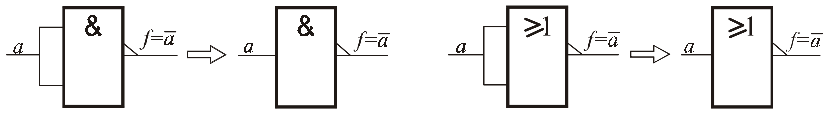
En la simbología IEC se pueden utilizar estos símbolos de inversión, pero se está extendiendo más la indicación con triángulos, tal como se ve en el símbolo de inversión dibujado posteriormente. Esta será la indicación de inversión utilizada para los símbolos IEC que utilicemos normalmente.

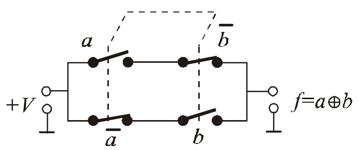
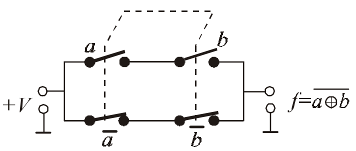
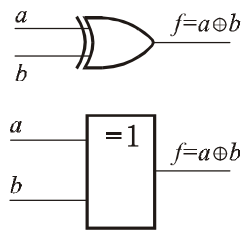
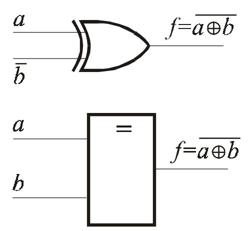
<table>
<tr><th></th><th>Operación suma</th><th>Operación producto</th><th>Operación inversión</th></tr>
<tr><td>Función lógica</td><td>$f = a + b$</td><td>$f = a \cdot b$</td><td>$f = \overline{a}$</td></tr>
<tr><td>Operador eléctrico</td><td></td><td></td><td></td></tr>
<tr><td>Símbolos lógicos</td><td></td><td></td><td></td></tr>
<tr><td>Tabla de verdad</td>
<td><table><tr><th>a</th><th>b</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></td>
<td><table><tr><th>a</th><th>b</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></td>
<td><table><tr><th>a</th><th>f</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table></td></tr>
</table>

<table>
<tr><th></th><th>Operación suma NOR</th><th>Operación producto NAND</th></tr>
<tr><td>Función lógica</td><td>$S = \overline{a + b} = \overline{a} \cdot \overline{b}$</td><td>$f = \overline{a \cdot b} = \overline{a} + \overline{b}$</td></tr>
<tr><td>Símbolos lógicos</td><td></td><td></td></tr>
<tr><td>Tabla de verdad</td>
<td><table><tr><th>a</th><th>b</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></td>
<td><table><tr><th>a</th><th>b</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></td></tr>
</table>

Otros inversores

Con una puerta NOR o una puerta NAND podemos obtener inversores al conectar entre sí sus entradas, tal como se indica



<table>
<tr><td></td><td>OR exclusiva, exclusión o EXOR</td><td>NOR exclusiva, equivalencia o EXNOR</td></tr>
<tr><td>Función lógica</td><td>$f = a \oplus b$</td><td>$S = \overline{a \oplus b}$</td></tr>
<tr><td>Operador eléctrico</td><td>
</td><td>
</td></tr>
<tr><td>Símbolos lógicos</td><td>
</td><td>
</td></tr>
<tr><td>Tabla de verdad</td><td>
<table>
<tr><th>a</th><th>b</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>
</td><td>
<table>
<tr><th>a</th><th>b</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>
</td></tr>
</table>

Simplificación de funciones

Una vez obtenida la función canónica de un determinado proceso, es posible encontrar una función lógica, equivalente a la anterior, que tenga el mínimo número de términos, sin que por ello varíe la función.

Son tres los métodos de simplificación que se pueden utilizar: algebraico o por Boole, tabular o de Karnaugh y numérico o de Quine - McCluskey.

Los tres métodos se basan en la existencia del complementario del álgebra de Boole, así recordarás que: $a + \overline{a} = 1$ y que $a \cdot \overline{a} = 0$.

También necesitamos aplicar $a + a = a$, lo que nos indica que ante términos repetidos sólo es necesario dejar uno, o por el contrario, podremos duplicar o utilizar un término cuantas veces necesitemos en la simplificación.

Esto nos produce, ante la existencia de dos términos canónicos que tengan todas sus variables iguales salvo una, que se puedan simplificar como sigue:

$$a\overline{b}c\overline{d}+\overline{a}\overline{b}c\overline{d}=\overline{b}c\overline{d}\left(a+\overline{a}\right)=\overline{b}c\overline{d}\cdot 1=\overline{b}c\overline{d}$$

$$\left(a+\overline{b}+\overline{c}+\overline{d}\right)\cdot\left(a+\overline{b}+\overline{c}+d\right)=\left(a+\overline{b}+\overline{c}\right)\cdot\left(d+\overline{d}\right)=\left(a+\overline{b}+\overline{c}+\right)\cdot 1=\left(a+\overline{b}+\overline{c}\right)$$

En este capítulo sólo trataremos el método algebraico y el método de Karnaugh.

Nos encontraremos con términos que no provocan en las salidas ni el estado lógico 1 ni el estado 0. En estos casos, la salida se representa por X y, en los mapas de Karnaugh, podemos considerar que su valor es 1 o 0 según interese. Éstos términos se denominan *indiferentes.*

Método algebraico

Utiliza los postulados, propiedades, teoremas y leyes del álgebra de Boole cuando la función a simplificar no es canónica.

Pero, en el ejemplo que sigue se aplica directamente lo indicado en la sección anterior, pues partimos de una función canónica.

Si tenemos la función:

$$f\left(1,4,6,8,12,13,14,15\right)=$$

$$=\overline{a}\overline{b}\overline{c}d+\overline{a}b\overline{c}\overline{d}+\overline{a}bc\overline{d}+a\overline{b}\overline{c}\overline{d}+ab\overline{c}\overline{d}+ab\overline{c}d+abc\overline{d}+abcd$$

Agruparemos, separados por unas barras, los términos simplificables si cambian en una sola de sus variables, duplicando, en estos agrupamientos, el término que creamos conveniente.

A continuación, eliminamos la variable que cambia en cada par de términos agrupados, quedando un solo término, como resultado del agrupamiento, con una variable menos.

$$f=\overline{a}\overline{b}\overline{c}d+\left|\overline{a}b\overline{c}\overline{d}+ab\overline{c}\overline{d}\right|+\left|a\overline{b}\overline{c}\overline{d}+ab\overline{c}\overline{d}\right|+\left|ab\overline{c}d+ab\overline{c}\overline{d}\right|+\left|abcd+abc\overline{d}\right|$$

$$+\left|abc\overline{d}+\overline{a}bc\overline{d}\right|=$$

$$=\overline{a}\overline{b}\overline{c}d+b\overline{c}\overline{d}\left(a+\overline{a}\right)+a\overline{c}\overline{d}\left(b+\overline{b}\right)+ab\overline{c}\left(d+\overline{d}\right)+abc\left(d+\overline{d}\right)+bc\overline{d}\left(a+\overline{a}\right)=$$

$$=\overline{a}\overline{b}\overline{c}d+b\overline{c}\overline{d}+a\overline{c}\overline{d}+ab\overline{c}+abc+bc\overline{d}$$

Con los términos resultantes volvemos a realizar agrupamientos con la misma intención, hasta llegar a términos no simplificables.

$$f=\overline{a}\overline{b}\overline{c}d+a\overline{c}\overline{d}+\left|b\overline{c}\overline{d}+bc\overline{d}\right|+\left|ab\overline{c}+abc\right|=$$

$$=\overline{a}\overline{b}\overline{c}d+a\overline{c}\overline{d}+b\overline{d}\left(c+\overline{c}\right)+ab\left(c+\overline{c}\right)=$$

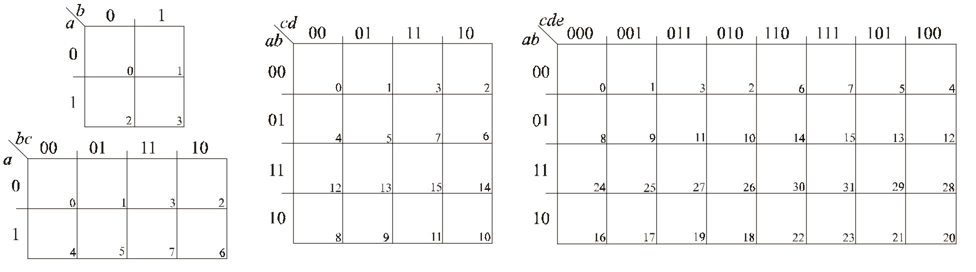
$$=\overline{a}\overline{b}\overline{c}d+a\overline{c}\overline{d}+b\overline{d}+ab$$

El resultado obtenido ha sido intencionado, ya que es difícil llegar normalmente, por este método, a una de las funciones más simplificadas.

Por esto debemos utilizar alguno de los métodos tabulares o numéricos que hay.

Método tabular de Karnaugh y Veitch

Para este método se utilizan tablas gráficas para funciones de 2, 3, 4, 5 y hasta de 6 variables. La utilización de este método para funciones de más variables resulta compleja, por lo que se debe utilizar el método numérico de Quine – McCluskey de forma manual con funciones no muy extensas o por medios informáticos, ya que este método lo permite.



Esta disposición conveniente, nos permite visualizar rápidamente los términos que se diferencian en una sola variable y, por lo tanto, eliminarla. Para ello las tablas tienen la particularidad de que los términos adyacentes en las filas y columnas sólo se diferencian en una de sus variables; para eso se disponen de la forma: 00, 01, 11, 10.

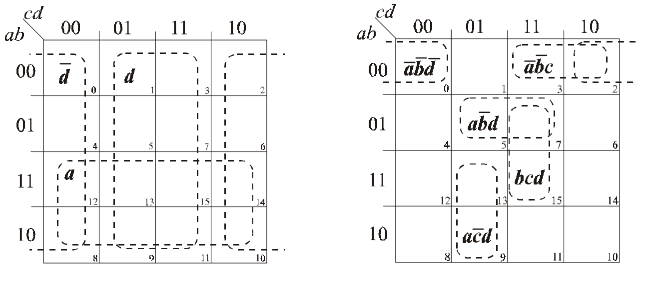
Ante una función canónica determinada, se colocará un **1** en las casillas correspondientes a los términos que contenga la función.

Es importante señalar que las tablas son cíclicas por los lados, por arriba y por abajo, es decir, en la tabla para cuatro variables los términos de la columna de la izquierda son adyacentes con los de la derecha, como en el caso del 0100 con el 0110; y los de la fila superior son adyacentes con los de la fila inferior, como el 0011 con el 1011.

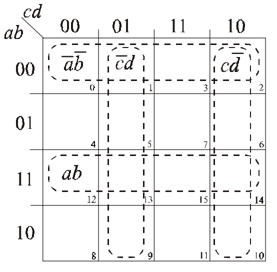
En esta representación intentamos agrupar un número de términos adyacentes, siempre en un número que sea una potencia de dos, y que varían en una, dos tres, etc., variables.

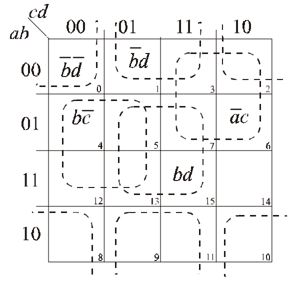
Sobre las tablas que siguen se indican algunos tipos de agrupamientos.

Serian ejemplos de agrupamientos de 8 ⇙ y de 2 ⇘:



Los que siguen son ejemplos de agrupamientos de 4:





Método

Se pretende obtener la función más simple, partiendo de una expresión canónica.

Agrupamientos mayores dan lugar a términos simplificados de menor número de variables.

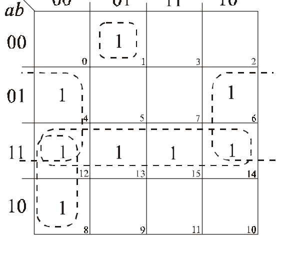
Se puede recoger un término en cuantos agrupamientos sean necesarios, ya que esto nos llevará a una expresión más simple.

Si tenemos la función de cuatro variables:

$$f(1,4,6,8,12,13,14,15) =$$

$$= \overline{a}\,\overline{b}\,\overline{c}\,d + \overline{a}\,b\,\overline{c}\,\overline{d} + \overline{a}\,b\,c\,\overline{d} + a\,\overline{b}\,\overline{c}\,\overline{d} + a\,b\,\overline{c}\,\overline{d} + a\,b\,\overline{c}\,d + a\,b\,c\,\overline{d} + abcd$$

1. Instalamos los términos en sus casillas, asignándoles un uno.
2. Realizamos los agrupamientos mayores posibles, que en este caso son dos de cuatro términos.
3. Realizamos los agrupamientos de menor tamaño, en este caso tenemos uno de dos términos.
4. Los términos que no se puedan agrupar con otros se dejan íntegros en la función simplificada. En este caso hay uno.



La función resultante de la simplificación será:

$$f = \overline{a}\,\overline{b}\,\overline{c}\,d + a\,\overline{c}\,\overline{d} + b\,\overline{d} + ab$$

Se han podido realizar otros tipos de agrupamientos, pero ninguno de ellos nos dará una función más simple.

Si la función viene expresada en maxterms el proceso sería el mismo.

Es importante obtener la función en minterms y en maxterms para comprobar cual de ella resulta más simple.

Realización de funciones con puertas lógicas

Podemos utilizar cualquier tipo de puerta en la realización de una función lógica pero la tendencia, por economía a nivel industrial, es utilizar un solo tipo de ellas, fundamentalmente NAND o NOR, ya que podemos obtener las otras a partir de ellas y tienen la ventaja de llevar implícita también la inversión.

Venga la función expresada como sumas de productos o productos de sumas, se puede realizar tanto con puertas NAND como con puertas NOR. El procedimiento seguido sería similar, teniendo en cuenta qué tipo de función tenemos y qué tipo de puertas queremos utilizar.

Realización con puertas NAND

Función expresada como suma de productos

$$f = \overline{a}\overline{b}\overline{c}d + a\overline{c}\overline{d} + b\overline{d} + ab$$

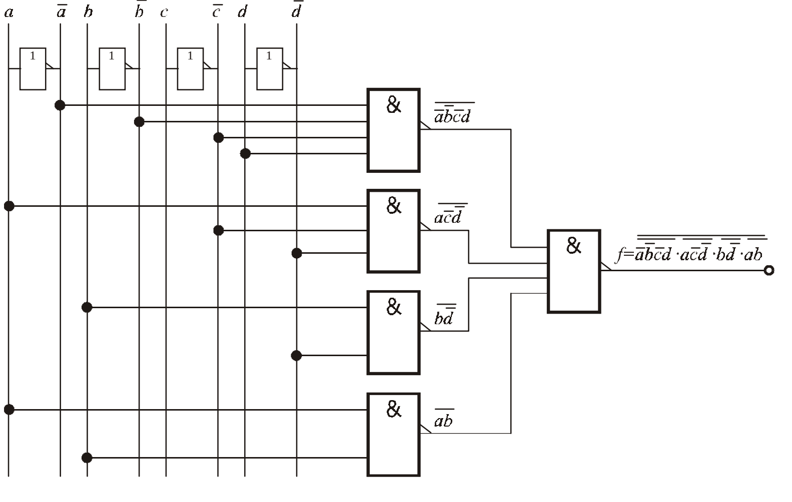
1. Se aplica siempre a toda la función dos inversiones. De esta forma la función no varía.

$$f = \overline{\overline{\overline{a}\overline{b}\overline{c}d + a\overline{c}\overline{d} + b\overline{d} + ab}}$$

2. Deshacemos una de las inversiones generales aplicando De Morgan.

$$f = \overline{\overline{\overline{a}\overline{b}\overline{c}d + a\overline{c}\overline{d} + b\overline{d} + ab}} = \overline{\overline{\overline{a}\overline{b}\overline{c}d} \cdot \overline{a\overline{c}\overline{d}} \cdot \overline{b\overline{d}} \cdot \overline{ab}}$$

3. Todos son productos invertidos, luego todos ellos se pueden realizar con puertas NAND.



Circuitos combinacionales

Todos los circuitos digitales, por muy complejos que estos sean, están realizados con puertas lógicas. Podemos diferenciar entre unos que se denominan combinacionales y otros denominados secuenciales. Podemos decir, genéricamente, que los primeros no son función del tiempo y los segundos sí.

Definimos como:

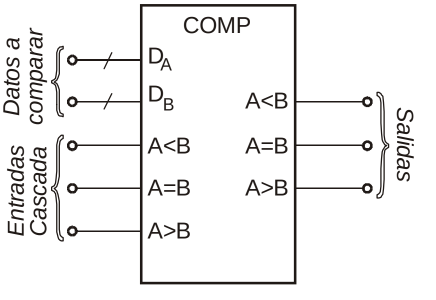
- **Circuito combinacional** aquel cuya salida **sólo depende del estado que tengan las variables de entrada,** cuando se actúa sobre él.
- **Circuito secuencial** aquel cuya salida **no sólo depende del estado de sus entradas sino también del estado que tenga su salida,** al actuar sobre él.

Comparadores

Un circuito comparador, como su nombre indica, se encarga de comparar dos datos binarios, A y B, de igual número de bits, entregándonos en sus salidas la información del resultado de la comparación.

Nos puede dar tres informaciones, según la *salida* activada:

1. Si A < B
2. Si A = B
3. Si A > B



El símbolo genérico que podemos asociar a un comparador sería el indicado, donde en las entradas introducimos D_A y D_B, que son los datos a comparar. Las *entradas* en cascada, indicadas por A < B, A = B y A > B, se utilizan para introducir la información proveniente de otro comparador, de forma que se puedan comparar datos de un número cualquiera de bits.

Representamos a continuación la tabla de verdad del comparador más simple, de un bit.

Entradas		Salidas		
A	B	A < B	A = B	A > B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

De la tabla podemos deducir los siguientes productos lógicos que nos producen indicación sobre la salida correspondiente:

- Para la salida A<B se produce $\mathbf{\overline{A}B}$ que se corresponde a una puerta AND.
- Para la salida A = B se producen $\mathbf{AB + \overline{A}\,\overline{B}}$ que se corresponde a una puerta XNOR.
- Para la salida A>B se produce $\mathbf{A\overline{B}}$ que se corresponde a una puerta AND.

Con estas últimas indicaciones se pretende que se comprenda que cualquier circuito, ya sea combinacional o secuencial, se realiza mediante puertas lógicas, como iremos viendo en los ejercicios.

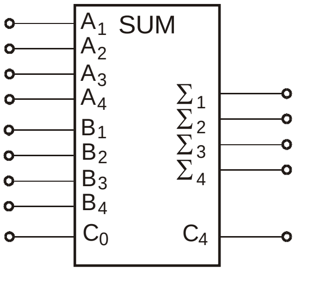
Sumador

Cuando es necesario sumar dos datos binarios, teniendo en cuanta el acarreo proveniente de una operación anterior, se utiliza un circuito denominado *sumador*. Sin embargo, si no es necesario tener en cuenta el acarreo de una operación anterior, se utiliza un circuito denominado *semisumador*.

Las entradas al sumador serán los dos datos a sumar A y B y el acarreo anterior denominado C_0; las funciones de salida, la suma S y el acarreo C.

Se expone por simplicidad la tabla de un sumador de dos datos de 1 bit.

A	B	C_0	S	C_1
0	0	0	0	0
0	0	1	1	0
0	1	0	1	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



En la figura de la derecha aparece el símbolo de un sumador de dos datos de cuatro bits.

Codificación, decodificación y transcodificación

Codificar consiste en establecer una correspondencia entre una información primaria de cualquier tipo, normalmente decimal, y una información secundaria siempre en binario, es decir, partimos de una información de cualquier tipo y obtenemos una información binaria. Ejemplos, de decimal a binario o de hexadecimal a binario.

Decodificar es la operación contraria, es decir, partiendo de una información binaria obtenemos una información de otro tipo. Ejemplos, de binario a decimal o de binario a hexadecimal.

Transcodificar o convertir el código, es partir de una información no binaria a otra información no binaria. Ejemplos, de hexadecimal a decimal o de decimal a hexadecimal.

A continuación se expone la tabla de verdad de un decodificador de decimal a binario.

DECIMAL (E)									BCD (S)			
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**D**	**C**	**B**	**A**
0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	**1**	**1**	0	0	**1**
0	0	0	0	0	0	0	**1**	0	0	0	0	**1**
0	0	0	0	0	0	**1**	0	0	**1**	**1**	**1**	0
0	0	0	0	0	**1**	0	0	0	0	**1**	**1**	0
0	0	0	0	**1**	0	0	0	0	**1**	0	**1**	0
0	0	0	**1**	0	0	0	0	0	0	0	**1**	0
0	0	**1**	0	0	0	0	0	0	**1**	**1**	0	0
0	**1**	0	0	0	0	0	0	0	0	**1**	0	0
1	0	0	0	0	0	0	0	0	**1**	0	0	0

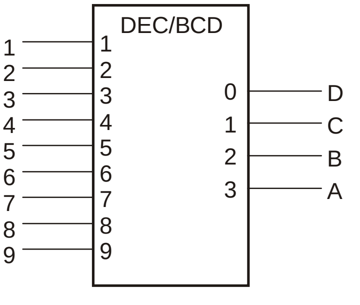
De la tabla anterior obtenemos las funciones de salida:

$$A = E_8 + E_9$$

$$B = E_4 + E_5 + E_6 + E_7$$

$$C = E_2 + E_3 + E_6 + E_7$$

$$D = E_1 + E_3 + E_5 + E_7 + E_9$$



realizables con puertas lógicas

El símbolo normalizado para un tipo de codificador de decimal a binario es el dibujado.

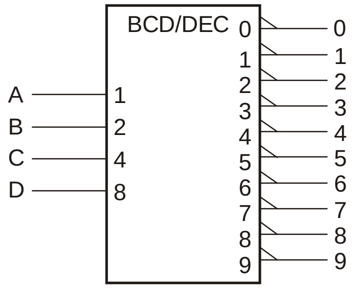
Decodificadores

Son circuitos combinacionales de varias entradas y varias salidas. Tienen un número **n** de entradas para **2^n** salidas.

Con una combinación binaria de la entrada se selecciona una de sus salidas.

En los decodificadores las entradas suelen ser activas a nivel alto mientras las salidas se hacen activas por niveles bajos.

El símbolo de este tipo de circuito, para un caso de decodificador de binario a decimal, es el que se indica.



Su tabla de verdad

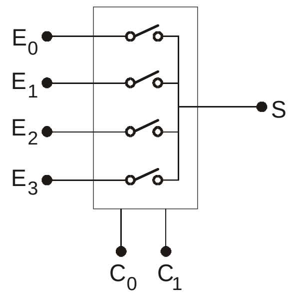
	ENTRADAS				SALIDAS									
Nº	D	C	B	A	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
1	0	0	0	1	1	0	1	1	1	1	1	1	1	1
2	0	0	1	0	1	1	0	1	1	1	1	1	1	1
3	0	0	1	1	1	1	1	0	1	1	1	1	1	1
4	0	1	0	0	1	1	1	1	0	1	1	1	1	1
5	0	1	0	1	1	1	1	1	1	0	1	1	1	1
6	0	1	1	0	1	1	1	1	1	1	0	1	1	1
7	0	1	1	1	1	1	1	1	1	1	1	0	1	1
8	1	0	0	0	1	1	1	1	1	1	1	1	0	1
9	1	0	0	1	1	1	1	1	1	1	1	1	1	0
INVÁLIDAS	1	0	1	0	1	1	1	1	1	1	1	1	1	1
	1	0	1	1	1	1	1	1	1	1	1	1	1	1
	1	1	0	0	1	1	1	1	1	1	1	1	1	1
	1	1	0	1	1	1	1	1	1	1	1	1	1	1
	1	1	1	0	1	1	1	1	1	1	1	1	1	1
	1	1	1	1	1	1	1	1	1	1	1	1	1	1

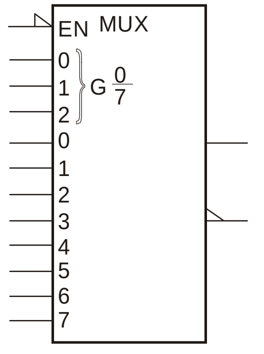
Multiplexores

Son circuitos en los que sus entradas de control seleccionan una entrada entre varias, para llevar la información de ésta a una única salida.

Para N líneas de entrada y n entradas de control la relación entre ellas es $N = 2^n$.

A la izquierda, representamos un esquema eléctrico de un multiplexor de cuatro entradas que nos ayuda a comprender el funcionamiento de estos circuitos. A la derecha, el símbolo normalizado de un multiplexor de ocho entradas.





La tabla de verdad de un multiplexor de cuatro entradas es:

Entradas de control		Salida
C_1	C_0	S
0	0	E_0
0	1	E_1
1	0	E_2
1	1	E_3

Su función lógica:

$$S = \overline{C}_1\overline{C}_0E_0 + \overline{C}_1C_0E_1 + C_1\overline{C}_0E_2 + C_1C_0E_3$$

Demultiplexores

Son circuitos que, con sus entradas de control, seleccionan una línea de salida entre varias, para llevar la información de su única entrada a la salida seleccionada.

Los circuitos decodificadores comerciales realizan también la función de demultiplexado.

Circuitos secuenciales

Los circuitos lógicos reseñados hasta ahora los hemos denominado combinacionales porque sus salidas sólo dependían de los valores de sus entradas.

Sin embargo, en un circuito secuencial el estado de sus salidas depende del estado de sus entradas, pero también depende del estado interno del circuito y de la secuencia con que se introduzcan sus entradas.

Biestables

Un biestable es un circuito electrónico capaz de memorizar una información. Dicho de otra forma, capaz de posicionarse en un estado interno indefinidamente (estado estable), mientras no se actúe sobre él, entregándonos en su salida un nivel alto o bajo de información.

Tipos de biestables

Los podemos clasificar según diversas características.

- Lógica de disparo: **RS** (Reset-Set), **JK**, **D** (Delay) y **T** (Toggle).
- Tipo de disparo: por **nivel**, **flanco** de subida o de bajada.
- Sincronismo de disparo: **asíncronos** y **síncronos**.

Un biestable asíncrono tiene poca utilidad o se utiliza en aplicaciones donde realiza una función individualizada. La mayoría de los biestables comercializados son síncronos o como tales forman un conjunto con una función muy específica, como contadores o registros.

Un biestable síncrono puede ser disparado de dos formas: **por nivel o por flanco**.

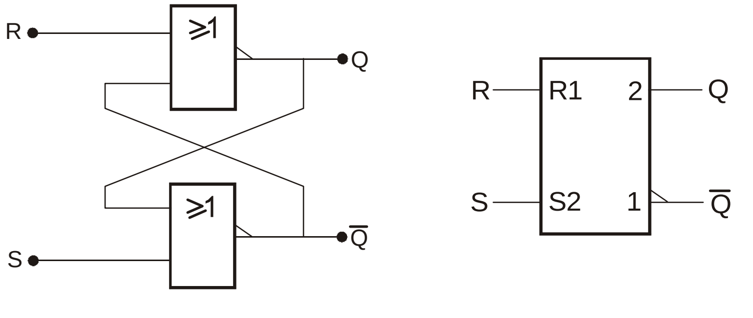
Un biestable, se dice, que es disparado por **nivel** si sólo es necesario que esté presente un valor característico (nivel lógico) de tensión en su entrada de reloj, para que al presentar un nivel lógico en su entrada de información el biestable se dispare.

Si para disparar el biestable es necesario que, estando presente la información, la entrada de reloj reciba un flanco ascendente o descendente con el cual se dispara, decimos que el biestable está disparado por **flanco** y en este caso suele recibir el nombre de biestable Edge-Triggered.

Los biestables suelen recibir también los nombres de **básculas** y **flip-flop**.

Biestable RS asíncrono con puertas NOR

Está formado por dos puertas NOR conectadas tal como se indica, y su símbolo normalizado es el representado.



En las tablas que siguen, se puede observar los diferentes efectos de las entradas R y S sobre las salidas (Q_{t+1} y $\overline{Q}_{t+1}$), teniendo en cuenta el estado anterior de las mismas (Q_t y $\overline{Q}_t$).

S	R	Q_t	Q_{t+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	Ind.
1	1	1	Ind.

Tabla característica

S	R	Q_{t+1}
0	0	Q_t
1	0	1
0	1	0
1	1	Ind.

Tabla de transición o próximo estado

Q_t	Q_{t+1}	S	R
0	0	0	Ind.
0	1	1	0
1	0	0	1
1	1	Ind.	0

Tabla de excitación

Cualquiera que sea la combinación de las salidas, si las dos entradas se ponen a 0, las salidas no cambian. A este estado de las entradas, en el que se conserva el estado que tenían las salidas, se le denomina **cerrojo** (**Latch**), y es el principio del funcionamiento de estos circuitos como elementos de *memoria*.

Si las dos entradas se ponen a 1, las dos salidas se ponen a 0. Este último efecto nos produce un estado de **indeterminación** (Ind.), del que es necesario conocer su existencia para evitarlo.

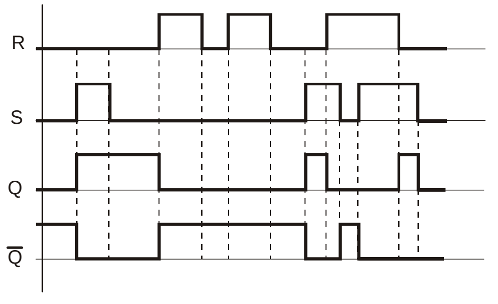
Cualquiera que sea el estado de las salidas al poner S a 1, la salida Q_{t+1} pasa a 1 y la $\overline{Q}_{t+1}$ a 0, denominamos este efecto puesta a 1 al poner la salida Q_{t+1} a 1.

Cualquiera que sea el estado de las salidas, al poner R a 1 la salida Q_{t+1} pasa a 0 y la $\overline{Q}_{t+1}$ a 1, denominamos este efecto puesta a 0 al poner la salida Q_{t+1} a 0.

Cronogramas

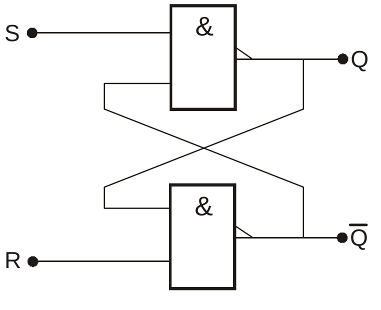
Para completar el análisis de un bloque secuencial se utiliza una representación gráfica de todas las señales que se producen en sus entradas y salidas en función del tiempo.

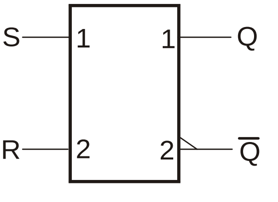
Se dibujan sobre unos ejes horizontales las entradas de un bloque secuencial en función del tiempo, indicando como evolucionan las salidas para una determinada combinación de las entradas y salidas. Para el biestable con puertas NOR sería el indicado.



Biestable RS asincrono con puertas NAND

El esquema del biestable R-S con puertas NAND es el indicado





Sus tablas

S	R	Q_t	Q_{t+1}
0	0	0	Ind.
0	0	1	Ind.
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Tabla característica

S	R	Q_{t+1}
0	0	Ind.
1	0	0
0	1	1
1	1	Q_t

Tabla de transición o próximo estado

Q_t	Q_{t+1}	S	R
0	0	1	Ind.
0	1	0	1
1	0	1	0
1	1	Ind.	1

Tabla de excitación

En las tablas de estados se indican las diferencias con el biestable de puertas NOR, indicadas también en el cronograma.