

## SISTEMA BINARIO

En el sistema decimal tenemos unidades, decenas, centenas,... en realidad lo que hacemos sin darnos cuenta es utilizar las potencias de base 10, para expresar cualquier número:

$$538 = 5 \times 10^2 + 3 \times 10^1 + 8 \times 10^0 \quad \text{Recordar que } 10^0 = 1$$

Usamos las potencias de base 10 puesto que podemos combinar 10 elementos.

Con el sistema binario ocurre algo similar, pero solo tenemos dos elementos: 0 y 1.

Cuando nos quedamos sin cifras hacemos lo mismo, se aumenta una cifra cada vez, por lo tanto, después del 1 viene el 10. Tras el 10 viene el 11, y después, como hemos vuelto a agotar las cifras, se aumenta una más, es decir, viene el número 100. La cuenta es, por lo tanto, la siguiente:

0, 1, 10, 11, 100, 101, 110, 111, 1000, 1001, 1010, ...

En este caso estamos usando las potencias de base 2, puesto que solo tenemos dos elementos. Así podemos expresar 101 como  $1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$ .

Notar que en ambos sistemas un cero a la izquierda del número no añade ningún valor.

## EQUIVALENCIA ENTRE NÚMEROS BINARIOS Y DECIMALES

Si hacemos la comparación entre los sistemas binarios y decimal, tendremos:

Decimal	0	1	2	3	4	5	6	7	8	9	10	11
Binario	0	1	10	11	100	101	110	111	1000	1001	1010	1011

Tener una tabla de este tipo no es siempre posible, pero a continuación vamos saber cómo se calcula la equivalencia entre los números de los dos sistemas.

### - Para pasar de binario a decimal

Debemos calcular el número decimal que corresponde a un binario dado, para ello se multiplica cada uno y cada cero del binario por potencias sucesivas de 2, empezando por  $2^0$  en la derecha ya que es la cifra de menos peso, y luego por  $2^1$  después por  $2^2$ , etc ... y por último sumamos todos

los valores:

$$\begin{array}{r} 101 \\ \begin{array}{l} \text{---} 1 \times 2^0 = 1 \times 1 = 1 \\ \text{---} 0 \times 2^1 = 0 \times 2 = 0 \\ \text{---} 1 \times 2^2 = 1 \times 4 = 4 \end{array} \\ \hline 5 \end{array}$$

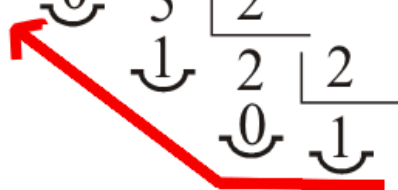
Es decir, basta con sumar las potencias en base 2 que hemos visto antes:

$$101 \rightarrow 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 5$$

Podemos comprobar la correspondencia en la tabla anterior.

### - Para pasar de decimal a binario

Para calcular el número binario que representa a un número decimal, tenemos que dividir el número decimal por 2 tantas veces como sea necesario hasta obtener un cociente menor que dos, por tanto el último cociente siempre será uno. Dicho cociente y los restos de las anteriores divisiones formaran el número binario.

$$\begin{array}{r} 10 \quad | \quad 2 \\ \hline 0 \quad 5 \quad | \quad 2 \\ \hline 1 \quad 2 \quad | \quad 2 \\ \hline 0 \quad 1 \end{array}$$


Se toman los unos y ceros empezando desde el último cociente y siguiendo con los restos desde abajo, y así tenemos el número binario: 1010

10 en decimal equivale a 1010 en binario.

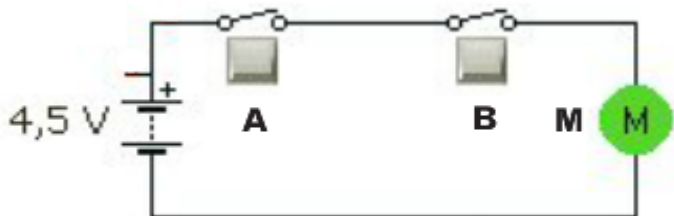
Observar que el último cociente sale dividir 3 veces por 2, será por tanto la potencia más alta  $2^3$  y la que más peso tiene, por eso se lee en ese sentido.

Con este ejemplo también se puede comprobar que se pueden confundir números decimales y binarios. Para evitarlo, a los números decimales se les puede añadir un 10 como subíndice, y a los números binarios, se les añade un 2. Por tanto, tendríamos:

$$10_{10} = 1010_2$$

## PROBLEMAS TECNOLÓGICOS

Si queremos que un motor eléctrico funcione solo cuando estén cerrados simultáneamente dos interruptores A y B es la solución es sencilla conectar dos interruptores en serie con el motor.



Para traducir este problema al lenguaje de la lógica digital seguimos las siguientes instrucciones:

**1. Identificar cada elemento de maniobra o control con una variable de entrada**, que solo puede tomar los valores 0 (abierto) y 1 (cerrado), aquí tenemos **A** y **B** que son interruptores.

**2. Identificar cada actuador con una función de salida.**

Esta función también tomará únicamente dos valores dependiendo de los que tomen las variables. En nuestro caso los valores 0 y 1 de la función **M** corresponderán respectivamente a los estados: parado y movimiento del motor. Las variables y las funciones que sólo pueden tomar dos valores se llaman variables y funciones lógicas.

**3. Elaborar la tabla de verdad de los actuadores.**

Dicha tabla recoge todos los valores que pueden tomar una función lógica según los valores que toman las variables lógicas de las que depende

VARIABLES		FUNCIÓN
A	B	M
0	0	0
0	1	0
1	0	0
1	1	1

Tabla de verdad.

**4. Expresar algebraicamente las funciones lógicas.**

En este circuito los valores que toma la función M pueden obtenerse multiplicando los valores que toman las variables lógicas A y B. Por tanto la función M puede expresarse así:  $M=A \cdot B$ . Por muy complicada que sea una función lógica siempre podrá expresarse como la suma de productos de sus variables negadas o no.

A esta forma de expresión algebraica se llama la **primera forma canónica** (FC1).

$$M=A \cdot B$$

## PUERTAS LOGICAS

Las puertas lógicas básicas son las tres correspondientes a las operaciones definidas en el álgebra de Boole, se llaman AND, OR y NOT y corresponden respectivamente a las operaciones: multiplicación, suma y negación, veámoslo:

### PUERTA AND (Y)

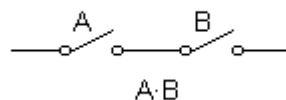
Su símbolo y tabla de verdad son las siguientes:



A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

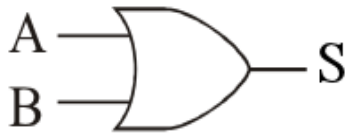
La salida de la puerta AND es uno solamente cuando todas las entradas son uno, es decir cuando A es uno y B es uno (AND=y). Esto es en realidad el producto de las entradas, por tanto la operación lógica AND se indica como  **$S = A \cdot B$** .

Si pensamos cómo obtener este mismo resultado físicamente con interruptores, llegamos a que el circuito equivalente son dos interruptores en serie, ya que solo habrá corriente cuando los dos a la vez estén activados (1 en A y en B).



## PUERTA OR (O)

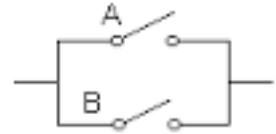
En este caso, el símbolo y la tabla de verdad son:



A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

La salida es uno cuando alguna de las entradas es uno, es decir cuando A es uno o B es uno (OR=0). Aunque no lo sea en realidad, a la función OR se le suele llamar suma y se describe como:  **$S=A+B$** .

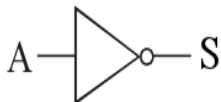
Si pensamos cómo obtener este mismo resultado físicamente con interruptores, llegamos a que el circuito equivalente son dos interruptores en paralelo, cuando cualquiera de los dos esté activado (1 en A o en B) habrá corriente.



Aunque existen puertas AND y OR de más entradas, es más barato construirlas sólo de dos entradas.

## PUERTA NOT

Esta puerta sólo tiene una entrada y una salida, y se limita a cambiar el valor de la entrada por su opuesto. Por esta razón, también se llama inversor o negación. Veamos su símbolo y su tabla de verdad:



A	S
0	1
1	0

Esta función se describe mediante la expresión:  **$S = \bar{A}$**

## EJEMPLO DE ASOCIACIÓN PUERTAS

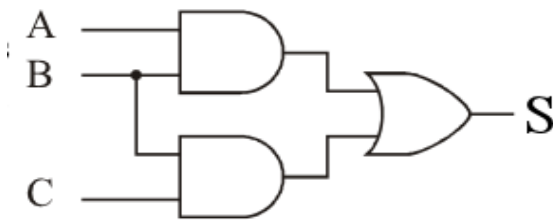
Es normal acoplar dos o más tipos de puertas lógicas para conseguir una salida determinada cuando se den unas condiciones en entradas, veamos cómo se obtiene la tabla de verdad de un circuito con varias puertas lógicas.

A la vista de un circuito, se debe empezar por asignar las letras en las entradas y salidas, por ejemplo nombramos las entradas con A, B, C,... y S a la salida. Después se anota en la tabla la expresión que describe cada operación lógica parcial para facilitarnos las operaciones (puede hacerse directamente pero es más fácil equivocarse), al final se pone la operación completa.

A la hora de escribir las entradas, tenemos que tener en cuenta todas las combinaciones posibles según los diferentes valores que podamos tener, en este caso cada variable solo puede tener dos valores, el número de combinaciones será por tanto  $2^n$  donde n es el número de entradas.

Seguimos siempre la misma técnica al rellenar la tabla, la última entrada se rellena con ceros y unos consecutivamente: 0,1,0,1, la penúltima con el doble de ceros y unos 0,0,1,1,, la siguiente con el doble de la anterior: 0,0,0,0,1,1,1,1, y así hasta que lleguemos a la primera.

Escribimos la tabla de verdad del siguiente circuito:



A	B	C	$A \cdot B$	$B \cdot C$	S
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	1	1
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	1	1	1
1	1	1	1	1	1

La salida podemos expresarla  $S = A \cdot B + B \cdot C$

VARIABLES			FUNCIÓN
A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Obtener la primera forma canónica (FC1) a partir de la tabla de verdad:

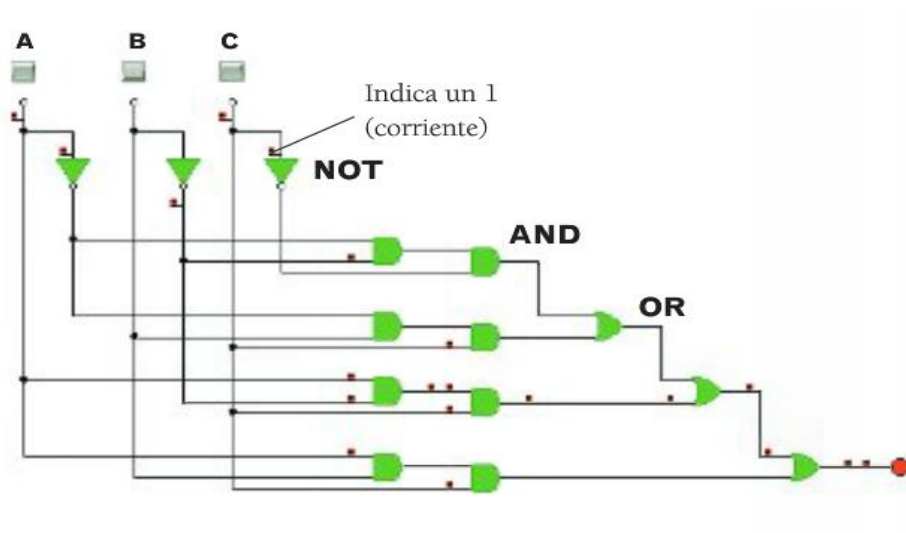
Corresponde a una función  $F$  que depende de tres variables ( $A$ ,  $B$  y  $C$ ).  $F$  toma el valor 1 para 4 combinaciones de valores de  $A$ ,  $B$  y  $C$ . La primera forma canónica de  $F$  tendrá cuatro términos. En cada término las variables deben aparecer negadas si toma el valor cero en la combinación correspondiente.

Fíjate en la primera combinación para la que se toma el valor 1. Las tres variables de entrada tienen el valor 0, así que deberán aparecer negadas en el primer término:  $\bar{A} \cdot \bar{B} \cdot \bar{C}$ . Los términos restantes son:  $\bar{A} \cdot B \cdot C$ ,  $A \cdot \bar{B} \cdot C$  y  $A \cdot B \cdot C$ . Por tanto, la primera forma canónica de la función  $F$  es:

$$F_{FC1} = \bar{A} \cdot \bar{B} \cdot \bar{C} + \bar{A} \cdot B \cdot C + A \cdot \bar{B} \cdot C + A \cdot B \cdot C$$

Ahora implementaremos con puertas lógicas la función anterior:

1. Dibujaremos tantos terminales de entrada como variables haya en la función, en este caso tres:  $A$ ,  $B$  y  $C$ . Cuando estén negadas debemos incluir una puerta NOT.
2. Usaremos puertas AND para construir cada término producto, en este caso vamos a usar puertas AND de dos entradas, pero tenemos tres entradas, por tanto para multiplicar  $A \cdot B \cdot C$ , lo haremos en dos pasos, primero  $A \cdot B$  y su salida la multiplicamos por  $C$ , puesto que se cumple la propiedad asociativa del producto.
3. Por último conectamos las ultimas puertas AND con puertas OR para implementar la suma, al igual que antes usaremos varias puertas de dos entradas en cadena, ya que se cumple la propiedad asociativa de la suma.



## ALGEBRA DE BOOLE

– Multiplicación ( $\cdot$ ):

$0 \cdot 0 = 0$	$0 \cdot 1 = 0$
$1 \cdot 0 = 0$	$1 \cdot 1 = 1$

– Suma (+):

$0 + 0 = 0$	$0 + 1 = 1$
$1 + 0 = 1$	$1 + 1 = 1$

– Negación ( $\bar{\phantom{x}}$ ):

$\bar{0} = 1$	$\bar{1} = 0$
---------------	---------------

La prioridad de estos operadores es:

1º la negación, después la multiplicación y por último la suma.

– Las operaciones suma y multiplicación son **conmutativas**:

$$x \cdot y = y \cdot x \quad x + y = y + x$$

– Las operaciones suma y multiplicación son **asociativas**:

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

$$(x \cdot y) \cdot z = x \cdot (y \cdot z)$$

– Las operaciones suma y multiplicación son **distributivas** una respecto a otra:

$$x \cdot (y + z) = (x \cdot y) + (x \cdot z)$$

$$x + (y \cdot z) = (x + y) \cdot (x + z)$$

➤ POSTULADO 1:  $a + 1 = 1$

➤ POSTULADO 2:  $a + 0 = a$

➤ POSTULADO 3:  $a \cdot 1 = a$

➤ POSTULADO 4:  $a \cdot 0 = 0$

➤ POSTULADO 5:  $a + a = a$

➤ POSTULADO 6:  $a \cdot a = a$

➤ POSTULADO 7:  $a + \bar{a} = 1$

➤ POSTULADO 8:  $a \cdot \bar{a} = 0$

➤ POSTULADO 9:  $\bar{\bar{a}} = a$

➤ POSTULADO 10:  $S = a + b; \bar{S} = \overline{a + b}$   
 $S = a \cdot b; \bar{S} = \overline{a \cdot b}$

➤ TEOREMA 1: LEY DE ABSORCIÓN.

$$a + (a \cdot b) = a$$

$$a(a + b) = a$$

➤ TEOREMA 2:

$$a + \bar{a} \cdot b = a + b$$

$$b(a + \bar{b}) = a \cdot b$$

➤ TEOREMA 3: LEYES DE MORGAN.

$$\overline{a + b} = \bar{a} \cdot \bar{b}$$

$$\overline{a \cdot b} = \bar{a} + \bar{b}$$

➤ TEOREMA 4:

$$ab + \bar{a}c + bc = ab + \bar{a}c$$

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