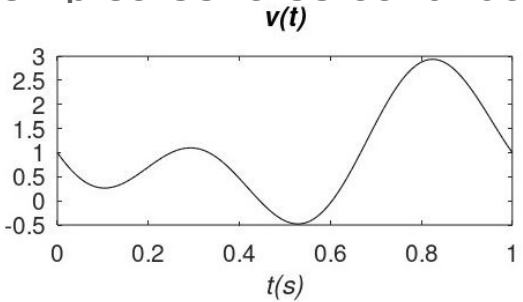
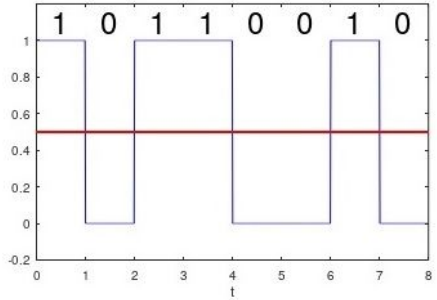
UD1 FUNDAMENTOS DE ELECTRÓNICA DIGITAL

1.- Diferencia entre sistemas digitales y analógicos

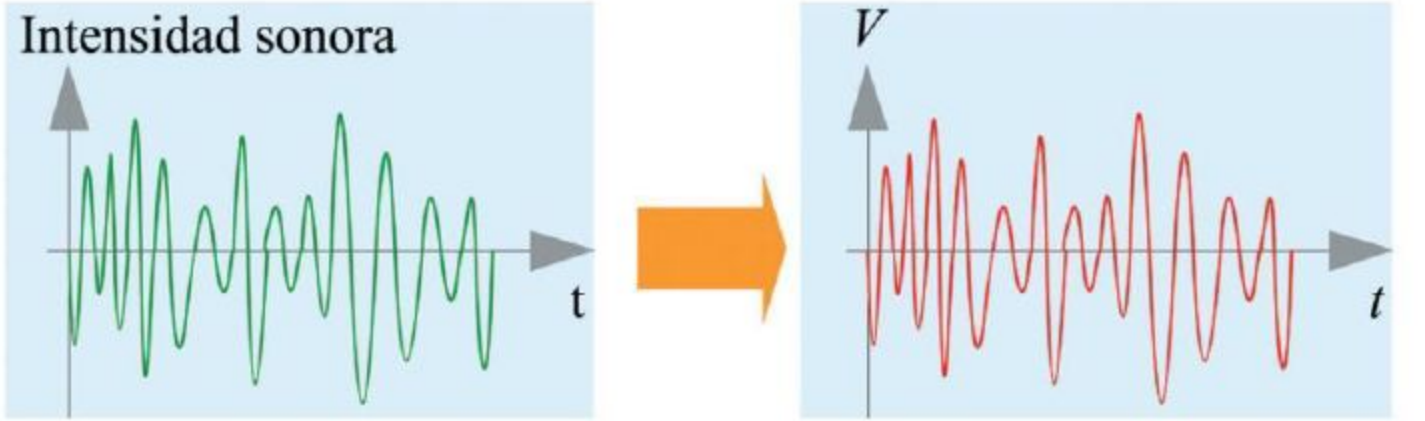
La electrónica analógica emplea señales continuas, donde cualquier valor es posible.



Sin embargo, la electrónica digital se basa en señales binarias, donde solo dos posibles estados están permitidos.

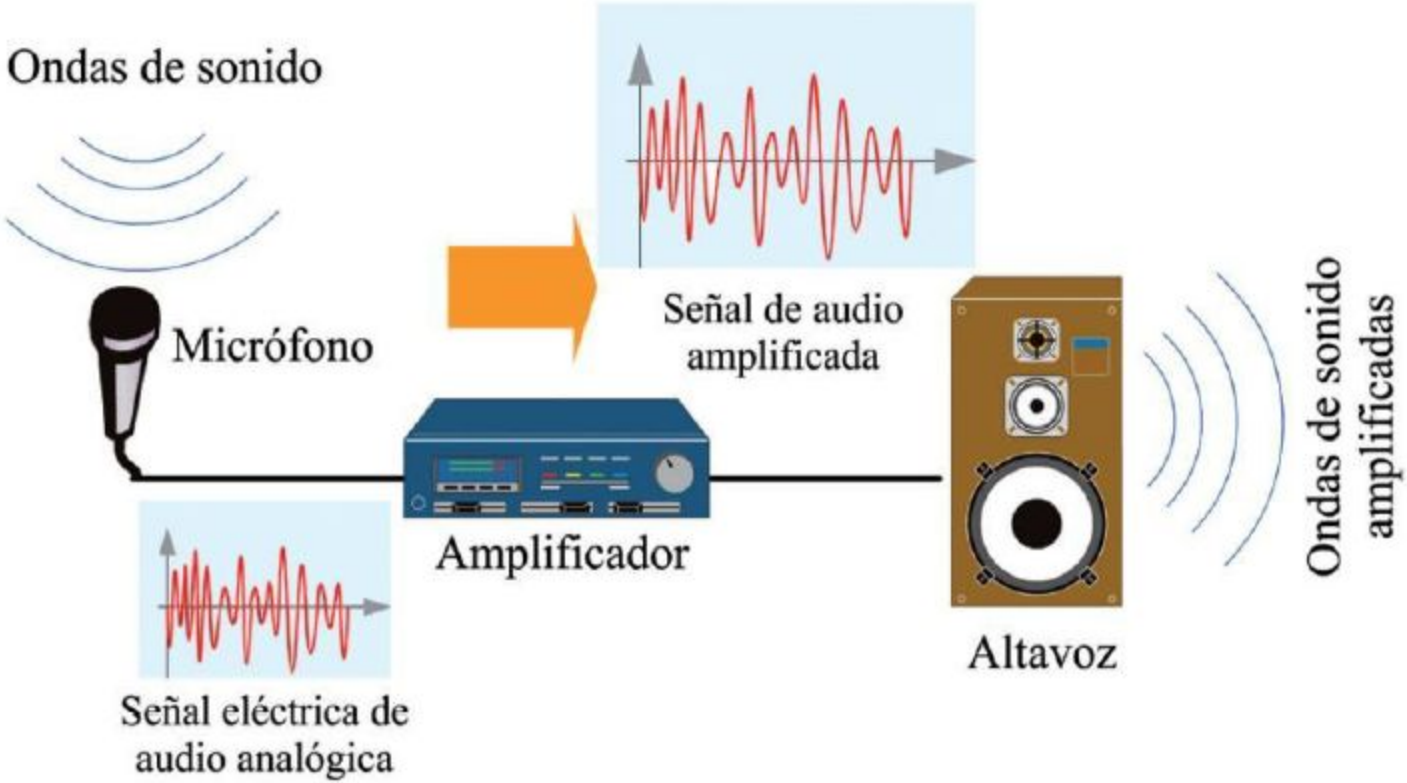


Señales analógicas:



Señal de audio

Señal eléctrica de audio analógica


Ondas de sonido
Micrófono
Señal de audio amplificada
Amplificador
Señal eléctrica de audio analógica
Altavoz
Ondas de sonido amplificadas

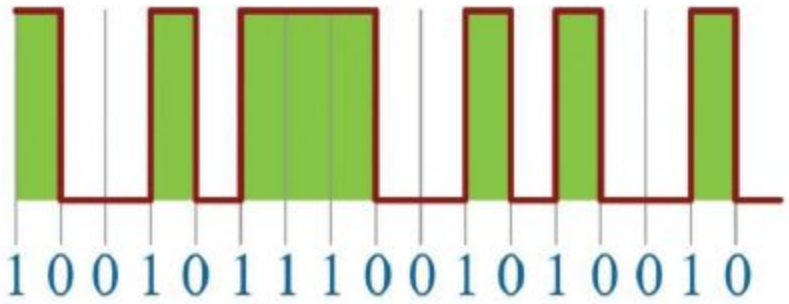
En la Figura, el sonido provoca la vibración de la membrana del micrófono, lo que hace que se genere una señal eléctrica analógica de muy poco nivel (en torno a unos pocos milivoltios); el amplificador de audio eleva dicho nivel (en torno a unos cuantos voltios) utilizando circuitos analógicos.

La señal ya tiene suficiente nivel (volumen) para poder mover la membrana de un altavoz, donde obtenemos el sonido original captado por el micrófono pero con un volumen mucho mayor.

Señales digitales:

Otra forma de tratar las señales eléctricas que vamos a procesar es convertirlas en números.

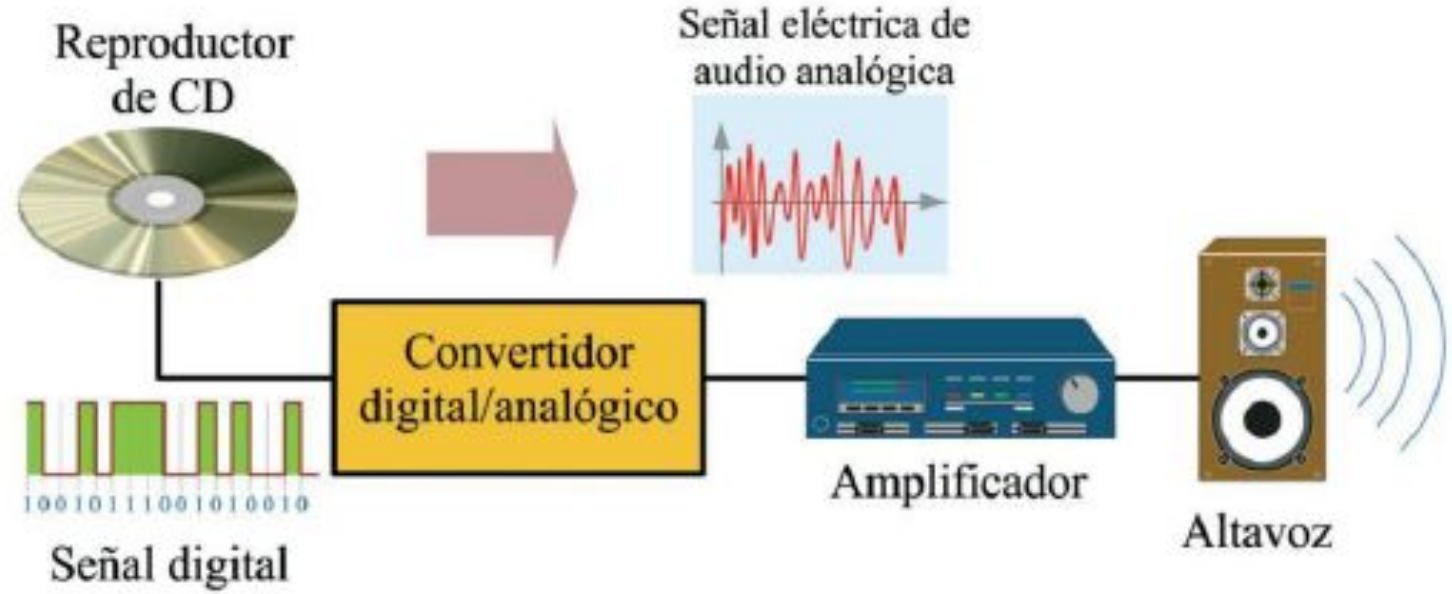
Las señales digitales son mucho más simples que las analógicas, ya que la información se procesa y codifica en dos únicos estados.



Estados de una señal digital:

1.er estado	2.º estado
1	0
Sí	No
Verdadero	Falso
Nivel alto de tensión	Nivel bajo de tensión
5 V	0 V
Interruptor cerrado 	Interruptor abierto 

Proceso de conversión de una señal digital en una señal analógica:



En el ejemplo de la Figura,el sonido está grabado en formato digital en un disco compacto (CD). El reproductor de CD lee los datos digitales gracias a un sistema óptico basado en diodos láser.

Luego se convierte a formato analógico mediante un decodificador o convertidor digital-analógico (DAC). A la salida del DAC obtenemos una señal analógica que se corresponde con la señal eléctrica original del sonido.

Por último, la señal se amplifica y se escucha en un altavoz.

Ventajas de la electrónica digital

Circuitos digitales	Circuitos analógicos
Siempre reproducen exactamente los mismos resultados.	La salida puede variar con la temperatura, la tensión de alimentación, estado de los componentes, etc.
El diseño de circuitos lógicos es sencillo.	Para diseñar circuitos hay que realizar operaciones complejas y conocer muy bien sus componentes.
Se pueden programar para hacer que un mismo circuito pueda ser utilizado para varias funciones.	Los circuitos solo realizan la función para la que han sido diseñados.

Mayor facilidad de integración para circuitos repetitivos.	Coste más elevado de los circuitos.
Menor posibilidad de interferencias en las señales digitales.	Susceptible de sufrir interferencias de otros sistemas.
Facilidad de almacenamiento de la información en soporte magnético u óptico sin deterioro de la fidelidad de la señal, aunque se realicen muchas copias.	La información almacenada se va deteriorando, sobre todo si se realizan copias.

2.- Sistemas de Numeración

Un **sistema de numeración** es el conjunto ordenado de símbolos o dígitos y las reglas con que se combinan para representar cantidades numéricas. Existen diferentes sistemas numéricos, cada uno de ellos se identifica por su base.

Un **dígito** en un sistema numérico es un símbolo que no es combinación de otros y que representa un entero positivo.

Un **bit** es un dígito binario (abreviación del inglés binary digit), es decir, un 0 o un 1.

La **base** de un sistema numérico es el número de dígitos diferentes usados en ese sistema.

El sistema de numeración que mejor se adapta a la codificación de las señales digitales es el binario, ya que solamente utiliza dos dígitos, el 1 y el 0.

Sistema	Base	Dígitos
Binario	2	0, 1
Octal	8	0, 1, 2, 3, 4, 5, 6, 7
Decimal	10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Hexadecimal	16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Sistema decimal

El sistema decimal tiene su base en diez dígitos: 0, 1, 2, 3, 4, 5, 6, 7,8 y 9. El número de dígitos o símbolos diferentes que se utilizan en un sistema de numeración constituye su base. Para el sistema decimal la base es 10.

El lugar que ocupa cada dígito en una determinada cifra nos indica su valor. Así, por ejemplo el 956,y se puede descomponer de la siguiente forma:

956_{10}= 900 + 50 + 6 = 9.100 + 5.10 + 6.1

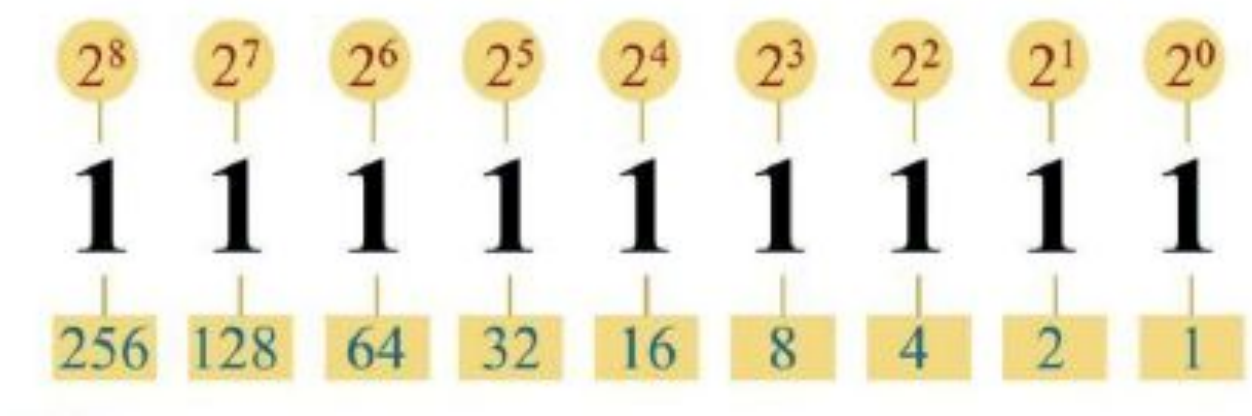
Otra forma de expresarlo sería en forma polinómica:

$$956_{10} = 9 \cdot 10^2 + 5 \cdot 10^1 + 6 \cdot 10^0$$

En conclusión, la cifra se descompone multiplicando cada dígito por su base elevada al número que representa la posición que ocupa.

Sistema binario

Valores de las posiciones de los términos de un número binario de 8 bits:



¿Cuál es el valor decimal del número binario 11001_2?

¿Cuál es el valor decimal del número binario 11001_2?

Solución:

Aplicamos la expresión polinómica:

$$11001_2 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 =$$
$$1 \cdot 16 + 1 \cdot 8 + 0 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 = 25_{10}$$

Convierte los siguientes números binarios a decimales:

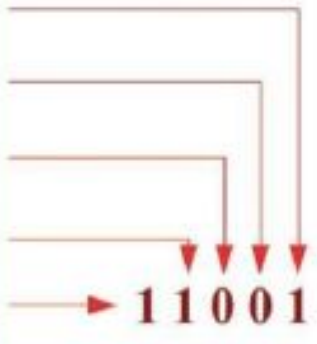
d) 11101001_2; e) $1010100 11_2$; f) 1100110010_2

¿Cuál es el valor binario del número decimal 25_{10}?

¿Cuál es el valor binario del número decimal 25_{10}?

Solución:

División	Cociente	Resto
25 : 2 =	12	1
12 : 2 =	6	0
6 : 2 =	3	0
3 : 2 =	1	1
1 : 2 =	0	1



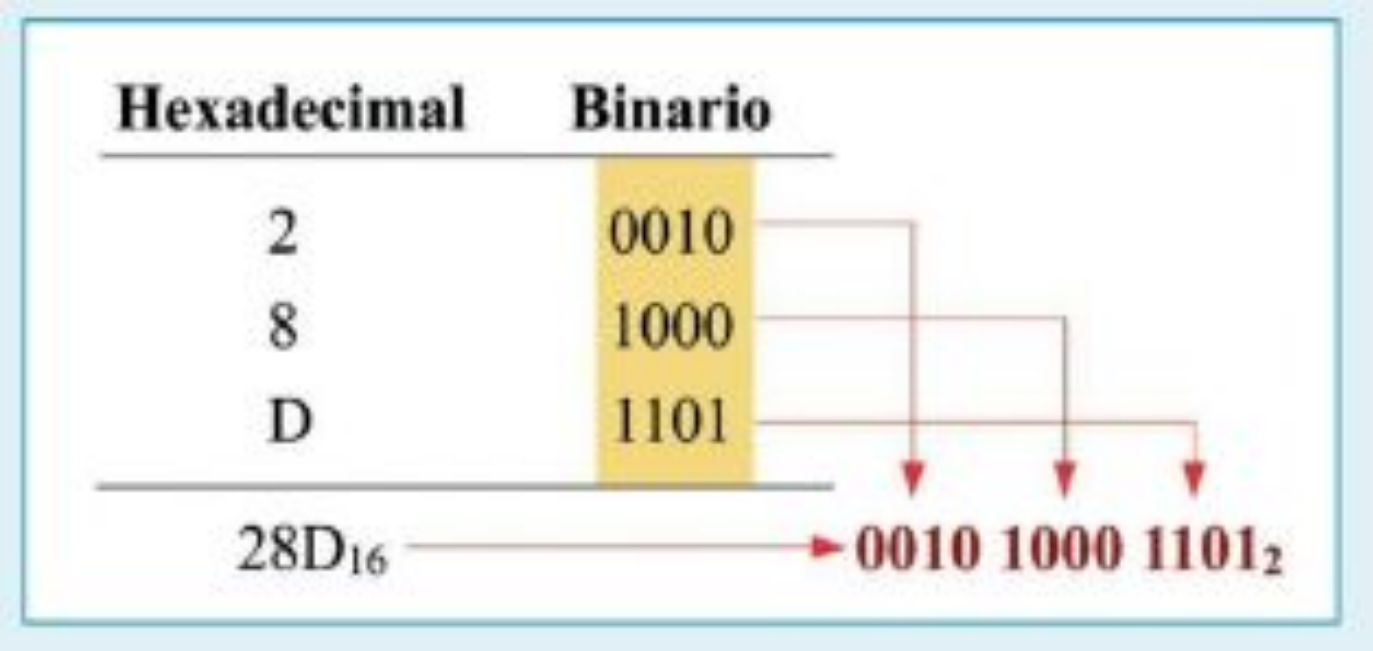
Convierte los siguientes números decimales a binarios:

a) 48_{10}; b) 375_{10}; c) 4.356_{10}

Sistema Hexadecimal

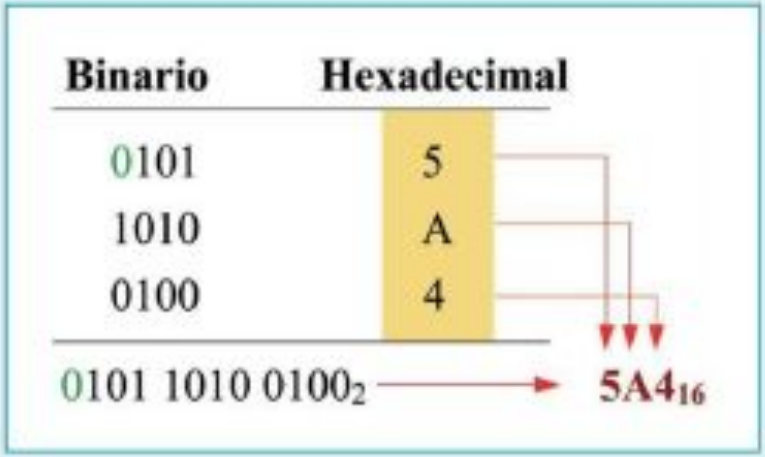
$Decimal_{(10)}$	$Hexadecimal_{(16)}$	$Binario_{(2)}$
0	0	0 0 0 0
1	1	0 0 0 1
2	2	0 0 1 0
3	3	0 0 1 1
4	4	0 1 0 0
5	5	0 1 0 1
6	6	0 1 1 0
7	7	0 1 1 1
8	8	1 0 0 0
9	9	1 0 0 1
10	A	1 0 1 0
11	B	1 0 1 1
12	C	1 1 0 0
13	D	1 1 0 1
14	E	1 1 1 0
15	F	1 1 1 1

¿Cuál es el número binario del número hexadecimal $28D_{16}$?



¿Cuál es el número hexadecimal del número binario 10110100100_2?

Se agrupan los bits de cuatro en cuatro comenzando por el bit menos significativo. Como en nuestro ejemplo el número de bits no es múltiplo de cuatro, se añaden a la izquierda del bit más significativo los ceros necesarios para completar un grupo de cuatro.



Ejercicios:

Convertir de hexadecimal a binario:

a) $45B_{16}$
b) $675D_{16}$

Convertir de binario a hexadecimal:

a) 1100010_{2}
b) 101010101010_{2}

2.- Códigos

Cuando se representan números, letras o palabras mediante un grupo especial de símbolos, decimos que están siendo codificados y al grupo de símbolos se le llama código.

Todos los sistemas digitales utilizan cierta forma de números binarios para su operación interna.

El más famoso quizás sea el código Morse.

Algunos de los códigos más empleados son el **BCD natural** y el **ASCII**.

Código Morse:

Letra	Código		Letra/Número	Código
A	.–		R	.–.
B	–...		S	...
C	–.–.		T	–
Ch	––––		U	..–
D	–..		V	...–
E	.		W	.––
F	..–.		X	–..–
G	––.		Y	–.––
H	....		Z	––..
I	..		1	.––––
J	.–––		2	..–––
K	–.–		3	...––
L	.–..		4	....–
M	––		5	.....
N	–.		6	–....
Ñ	––.––		7	––...
O	–––		8	–––..
P	.––.		9	––––.
Q	––.–		0	–––––

Código BCD natural

BCD natural (Binary-Coded Decimal): es un estándar para representar números decimales en el sistema binario, en donde cada dígito decimal es codificado con una secuencia de 4 bits.

Los números decimales, se codifican en BCD con los bits que representan sus dígitos. Por ejemplo, la codificación en BCD del número decimal 59237 es:

Decimal: 59237_{10}

BCD: $0101\ 1001\ 0010\ 0011\ 0111_{BCD}$

La representación anterior (en BCD) es diferente de la representación del mismo número decimal en binario puro (vista en páginas anteriores): 1110011101100101_2

Equivalencias Decimal-BCD natural

$Decimal_{(10)}$	BCD natural
0	0 0 0 0
1	0 0 0 1
2	0 0 1 0
3	0 0 1 1
4	0 1 0 0
5	0 1 0 1
6	0 1 1 0
7	0 1 1 1
8	1 0 0 0
9	1 0 0 1

a) Convierte el número 928_{10} en BCD.

b) Convierte el número 100001110011_{BCD} en decimal.

a) Convierte el número 928_{10} en BCD.

b) Convierte el número 100001110011_{BCD} en decimal.

a) $928_{10} = 1001\ 0010\ 1000_{BCD}$

b) $1000\ 0111\ 0011_{BCD} = 873_{10}$

Código ASCII

El código ASCII (*American Standard Code for Information Interchange*) es un código alfanumérico que utiliza 7 bits para codificar números, letras, símbolos especiales e instrucciones de control para periféricos. Es el código más utilizado en los teclados de los ordenadores.

Así, por ejemplo, la palabra «Hola» se presenta en código ASCII de la siguiente forma:

H	**o**	**l**	**a**
1001000	1101111	1101100	1100001

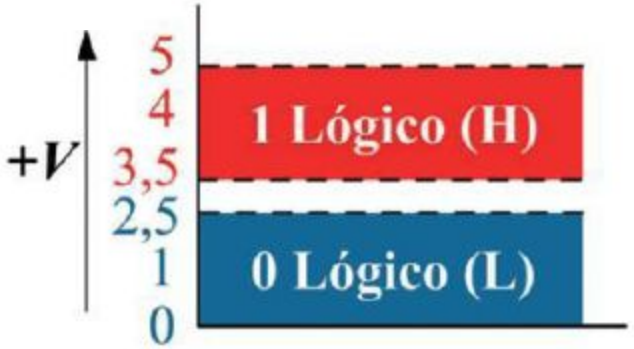
Encuentra el significado de la siguiente expresión codificada en ASCII:

1000010 1001001 1000101 1001110

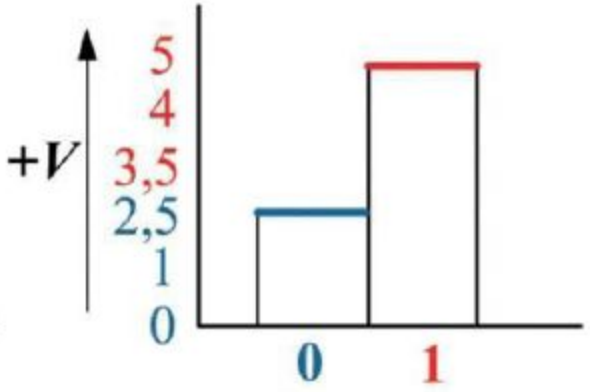
3.- Niveles lógicos de las señales digitales

- **Lógica positiva:** el «1» equivale a una tensión de nivel alto (**H**igh) y el «0» a una tensión de nivel bajo (**L**ow) (Figura 1.11).
- **Lógica negativa:** el «1» equivale a una tensión de nivel bajo (**L**ow) y el «0» a una tensión de nivel alto (**H**igh) (Figura 1.12).

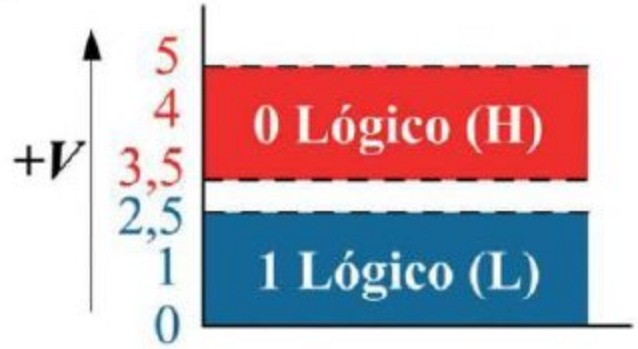
Lógica positiva.



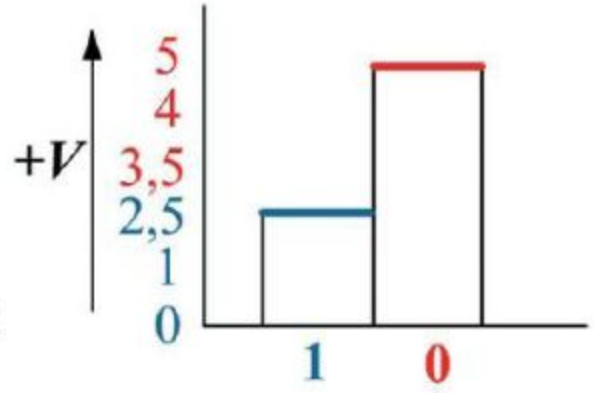
o bien



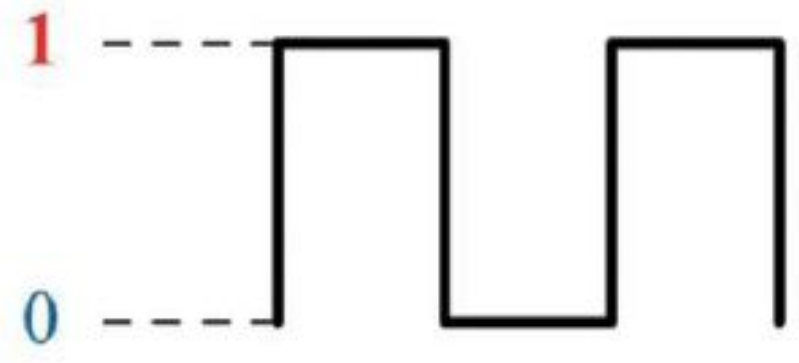
Lógica negativa.



o bien



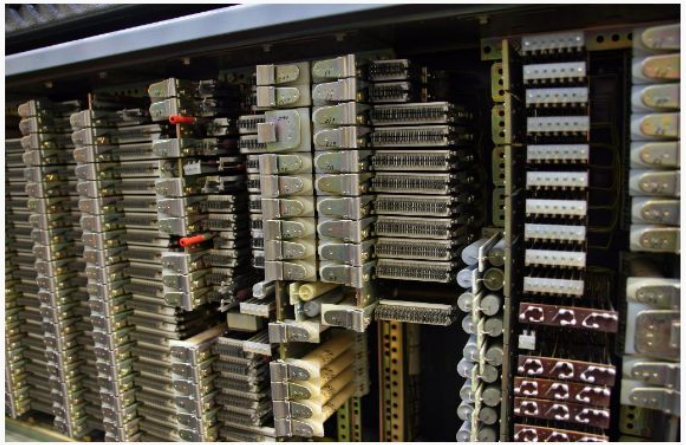
Niveles lógicos de una señal digital.



3.- Puertas lógicas

El origen de los circuitos lógicos comienza con la necesidad de construir automatismos y es anterior al desarrollo de la electrónica digital integrada.

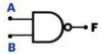
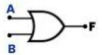
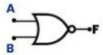
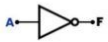
Una de las primeras aplicaciones fue la sustitución de los relés electromagnéticos, que ocupaban un gran volumen y requerían de operaciones de mantenimiento constantes, por puertas logicas integradas en un solo «chip», en las centrales telefónicas.



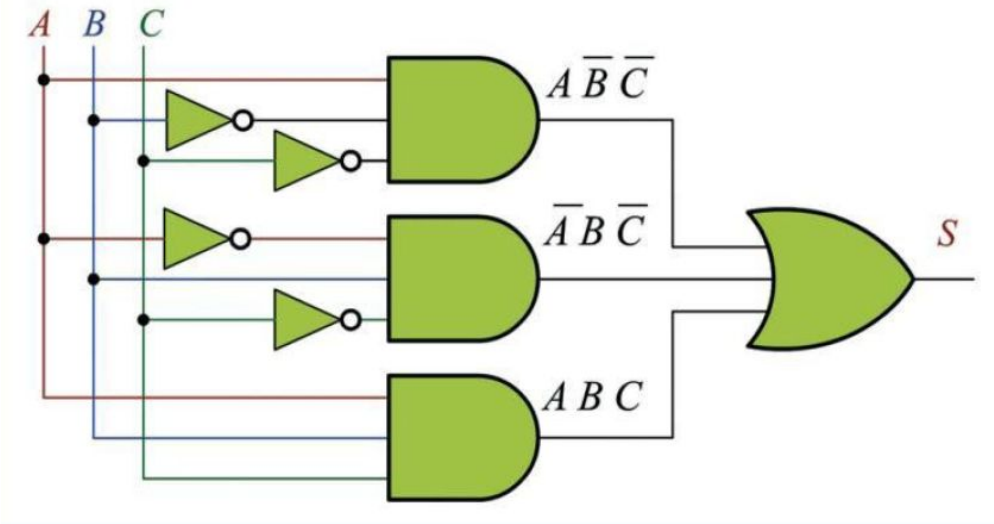
Detalle cuadro relés P1000. Central Madrid/S. Cristóbal en 2012 ya fuera de servicio.

Los componentes básicos que se utilizan en la electrónica digital para realizar las diferentes funciones elementales reciben el nombre de puertas lógicas.

Las puertas lógicas se consiguen gracias a los circuitos integrados, y constan de diferentes entradas y de una salida. A las entradas de las mismas solo se aplica uno de los dos niveles lógicos: «1» o «0», y en función del tipo de puerta utilizada, obtendremos a la salida uno de dichos niveles lógicos.

NOMBRE	TABLA DE VERDAD	CIRCUITO	OPERACIÓN
AND 	A B F 0 0 0 0 1 0 1 0 0 1 1 1	74LS08 	F=AB
NAND 	A B F 0 0 1 0 1 1 1 0 1 1 1 0	74LS00 	$F=\overline{AB}$ $=\overline{A}+\overline{B}$
OR 	A B F 0 0 0 0 1 1 1 0 1 1 1 1	74LS32 	F=A+B
NOR 	A B F 0 0 1 0 1 0 1 0 0 1 1 0	74LS02 	$F=\overline{A+B}$ $=\overline{A}\cdot\overline{B}$
XOR 	A B F 0 0 0 0 1 1 1 0 1 1 1 0	74LS86 	$F=A\oplus B$ $=A\overline{B}+\overline{A}B$
NOT 	A F 0 1 1 0	74LS04 	$F=\overline{A}$

En este tema estudiaremos los circuitos combinacionales, donde las salidas dependen directamente del valor de las entradas, y no pueden por tanto almacenar ningún tipo de información, solo realizan transformaciones en las entradas. Más adelante estudiaremos los circuitos secuenciales, que son capaces de recordar números que han recibido anteriormente.



Para representar las combinaciones de las entradas posibles y el nivel obtenido a la salida se utiliza la tabla de la verdad:

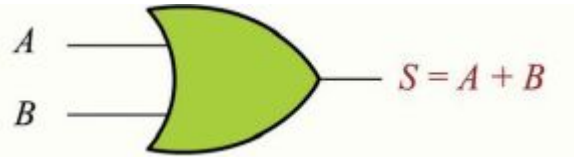
A B C	*S*
0 0 0	0
0 0 1	0
0 1 0	0
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	0
1 1 1	1

Puerta O (OR)

Es una puerta lógica de varias entradas.

Para el caso de dos entradas, la salida obtenida es de nivel alto si cualquiera de sus entradas o ambas están a nivel alto, y su salida será de nivel bajo si ambas entradas están a nivel bajo.

Puerta lógica OR y su tabla de la verdad.



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

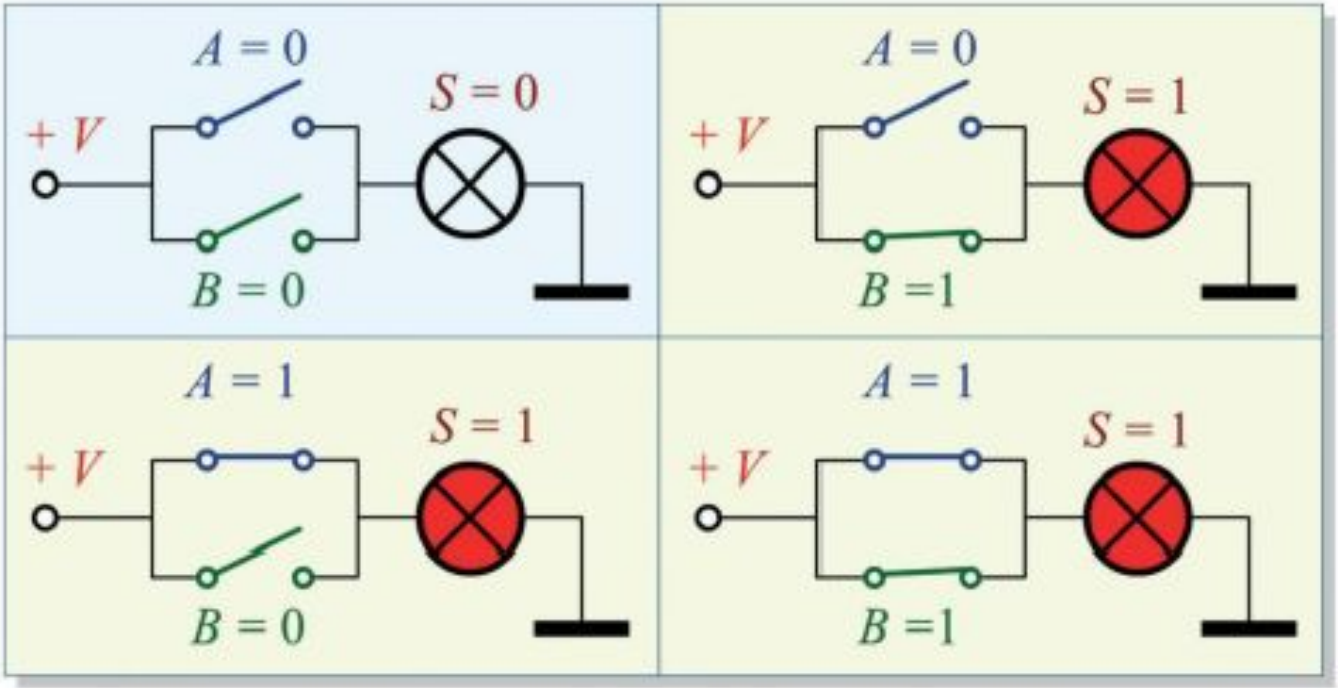
Esta puerta lógica realiza la función «O», conocida también con el nombre de suma:

$$S = A + B$$

Desde el punto de vista de la lógica, esta función se puede interpretar así:

- La salida S será verdadera cuando cualquiera de las entradas A o B lo sea.

Simulación de una puerta OR mediante interruptores



Dibuja el símbolo de una puerta OR con tres entradas y escribe su tabla de la verdad.

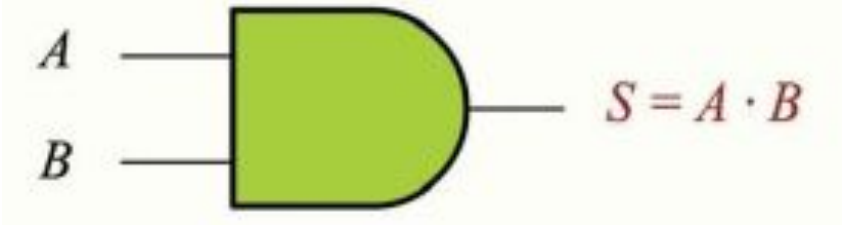
Puerta Y (AND)

Esta puerta lógica realiza la función Y, conocida también con el nombre de producto:

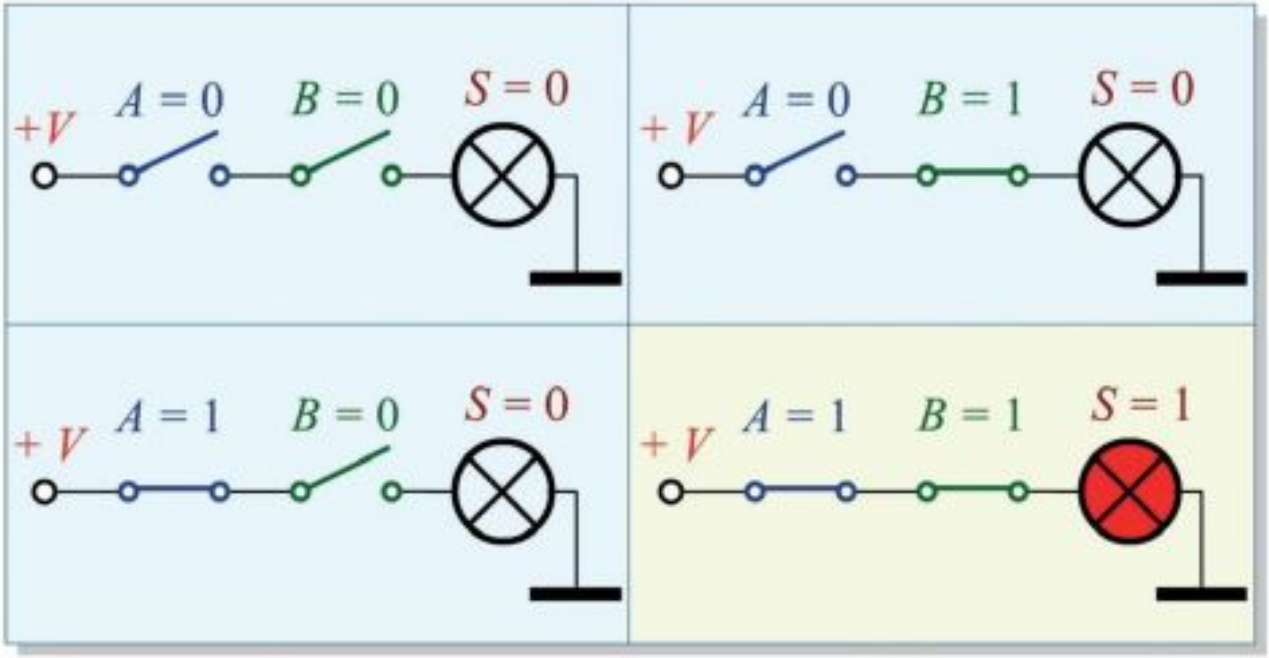
$$S = A \cdot B$$

Para el caso de dos entradas, la salida obtenida es de nivel alto solo si ambas entradas están a nivel alto.

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



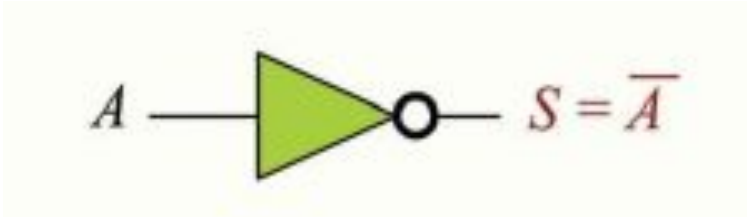
Simulación de una puerta AND mediante interruptores.



Dibuja el símbolo de una puerta AND con cuatro entradas y escribe su tabla de la verdad.

Puerta inversora NOT

Es una puerta lógica de una sola entrada. La salida obtenida es siempre el inverso al nivel lógico de la entrada, es decir convertir unos a ceros y ceros a unos.



A	S
0	1
1	0

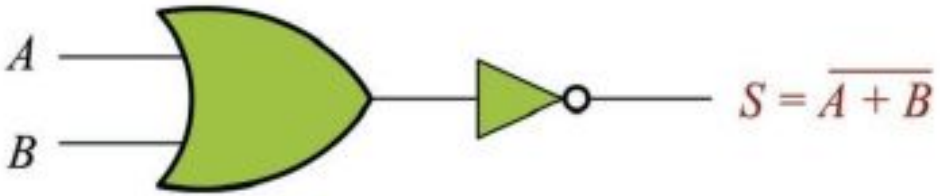
La función que realiza la puerta NOT es:

$$S = \overline{A}$$

Puerta NO O (NOR)

La puerta NOR, desde un punto de vista functional, está formada per una puerta OR y una puerta NOT.

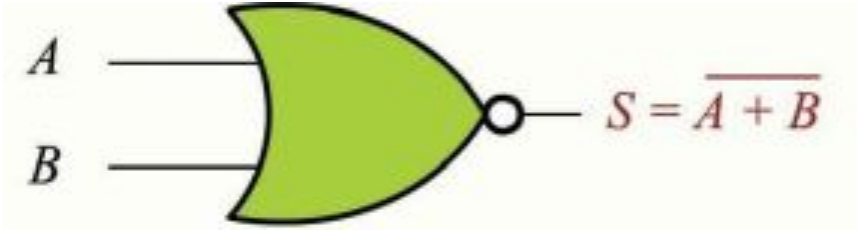
Su funcionamiento es el inverso de la puerta OR. Para el caso de dos entradas, la salida obtenida es de nivel alto sólo si ambas entradas están a nivel bajo.



La función que realiza la puerta NOR es:

$$S = \overline{A + B}$$

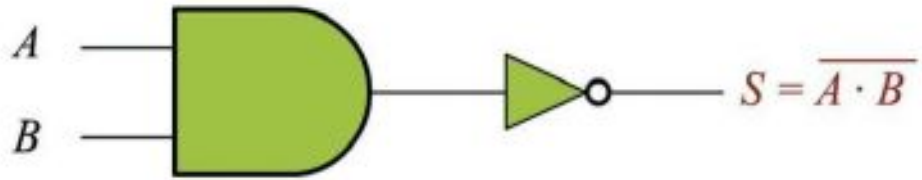
Puerta lógica NOR y su tabla de la verdad.



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

Puerta NO Y (NAND)

La puerta NAND funcionalmente está formada por una puerta AND y una puerta NOT

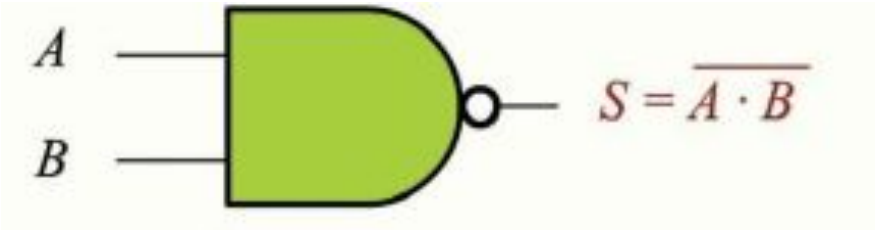


Su funcionamiento es el **inverso** de la puerta AND.

La función que realiza la puerta NAND es:

$$S = \overline{A \cdot B}$$

Puerta lógica NAND y su tabla de la verdad.



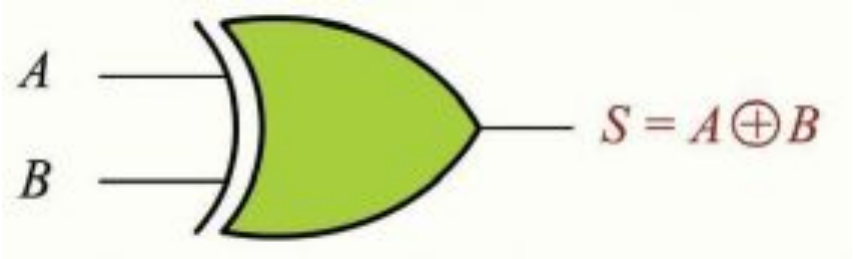
A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Puerta O exclusiva (XOR)

La salida obtenida en una puerta XOR es de nivel alto solo cuando lo sea **exclusivamente** alguna de sus entradas.

La función que realiza la puerta XOR es:

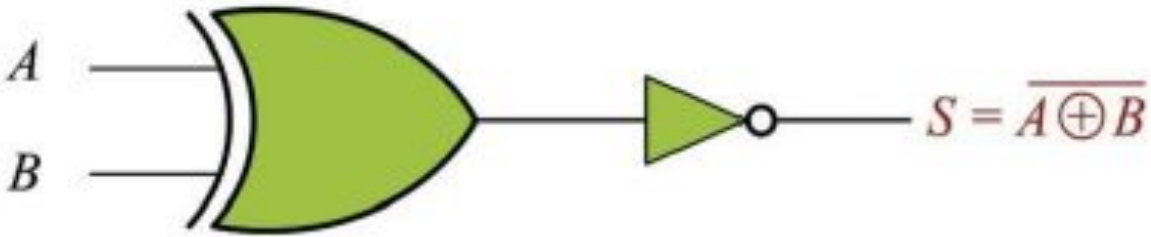
$$S = A \oplus B$$



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

Puerta NO XOR (XNOR)

La puerta XNOR funcionalmente está formada por una puerta XOR y una puerta NOT

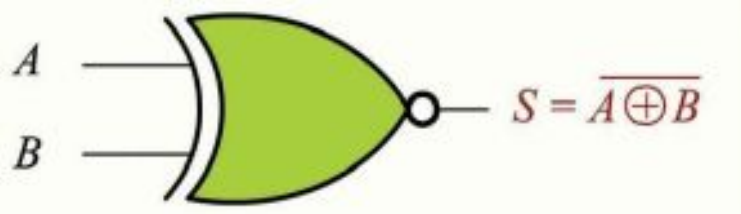


Su funcionamiento es el **inverso** de la puerta XOR. La salida obtenida es de nivel alto si ambas entradas son iguales.

La función que realiza la puerta XOR es:

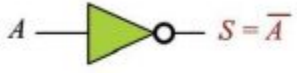
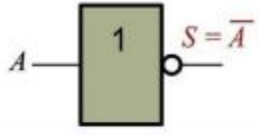
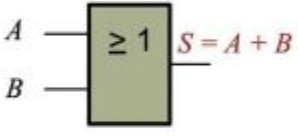
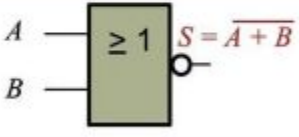
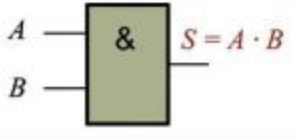
$$S = \overline{A \oplus B}$$

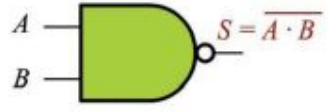
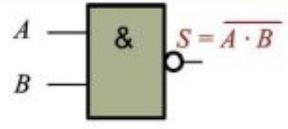
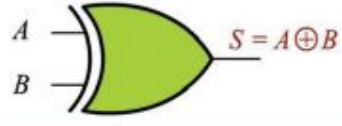
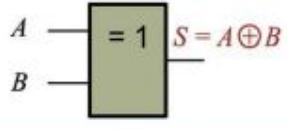
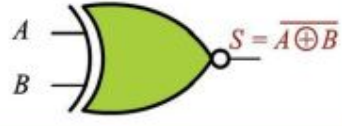
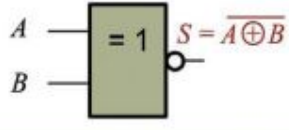
Puerta lógica XNOR y su tabla de la verdad.



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

Simbología tradicional y ANSI

Puerta	Símbolo tradicional	Símbolo ANSI
NOT		
OR		
NOR		
AND		

Símbolo tradicional	Símbolo ANSI
NAND 	
XOR 	
XNOR 	

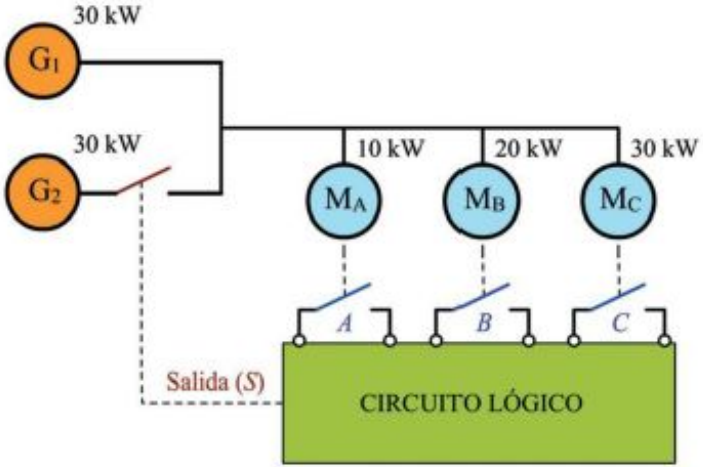
4.- Diseño de circuitos combinacionales con puertas lógicas

Con la combinación de diferentes puertas lógicas se puede conseguir dar respuesta a una determinada aplicación práctica. Para ello, lo primero es definir el nfimero de entradas y establecer las asociaciones de las señales de entrada con la salida.

Esto se lleva a cabo con la ayuda de la tabla de la verdad, de la que se obtiene la función que se corresponde con la salida.

Una vez obtenida dicha función, ya se puede realizar el circuito o diagrama lógico formado por la interconexión de las puertas lógicas necesarias.

En una nave industrial se dispone de tres motores de las siguientes potencias: 10 kW, 20 kW y 30 kW, para lo que se dispone de dos generadores de 30 kW cada uno. Se desea diseñar un sistema automático que ponga en funcionamiento el segundo generador cuando la potencia de los motores supere los 30 kW suministrados por el primer generador.



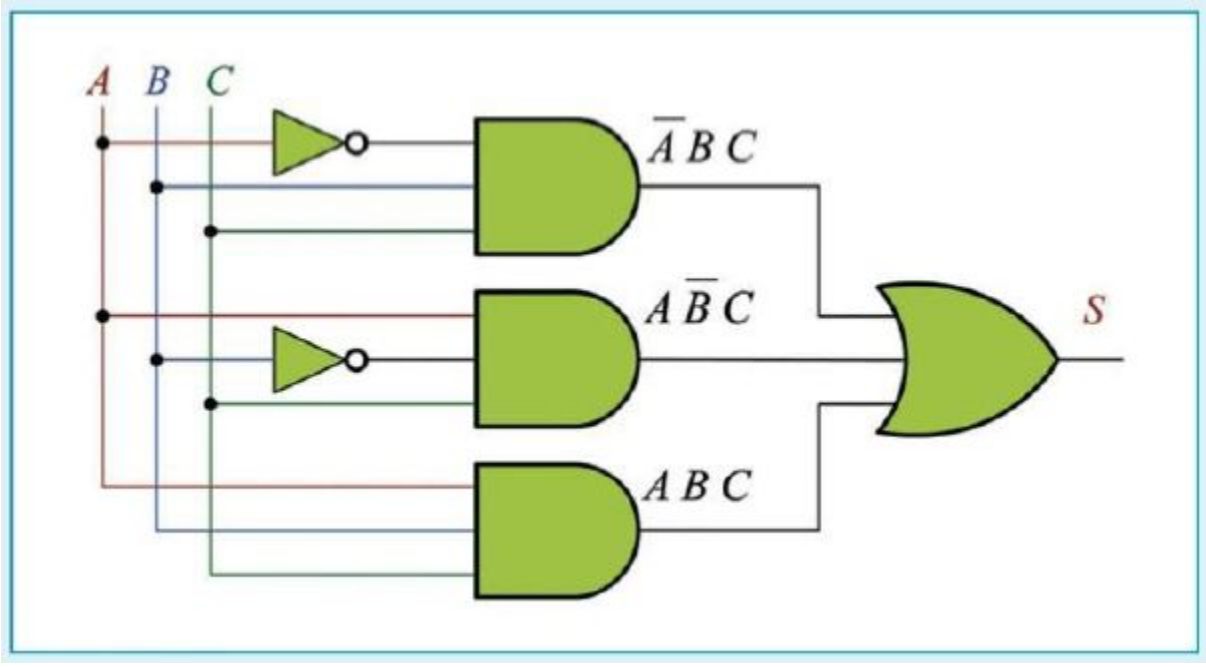
Hacer la tabla de la verdad y circuito lógico.

Con esta información ya estamos en disposición de realizar la tabla de la verdad que cumpla con las condiciones dadas en el diseño Como las entradas son 3, el número de combinaciones posibles será: $2^3 = 8$.

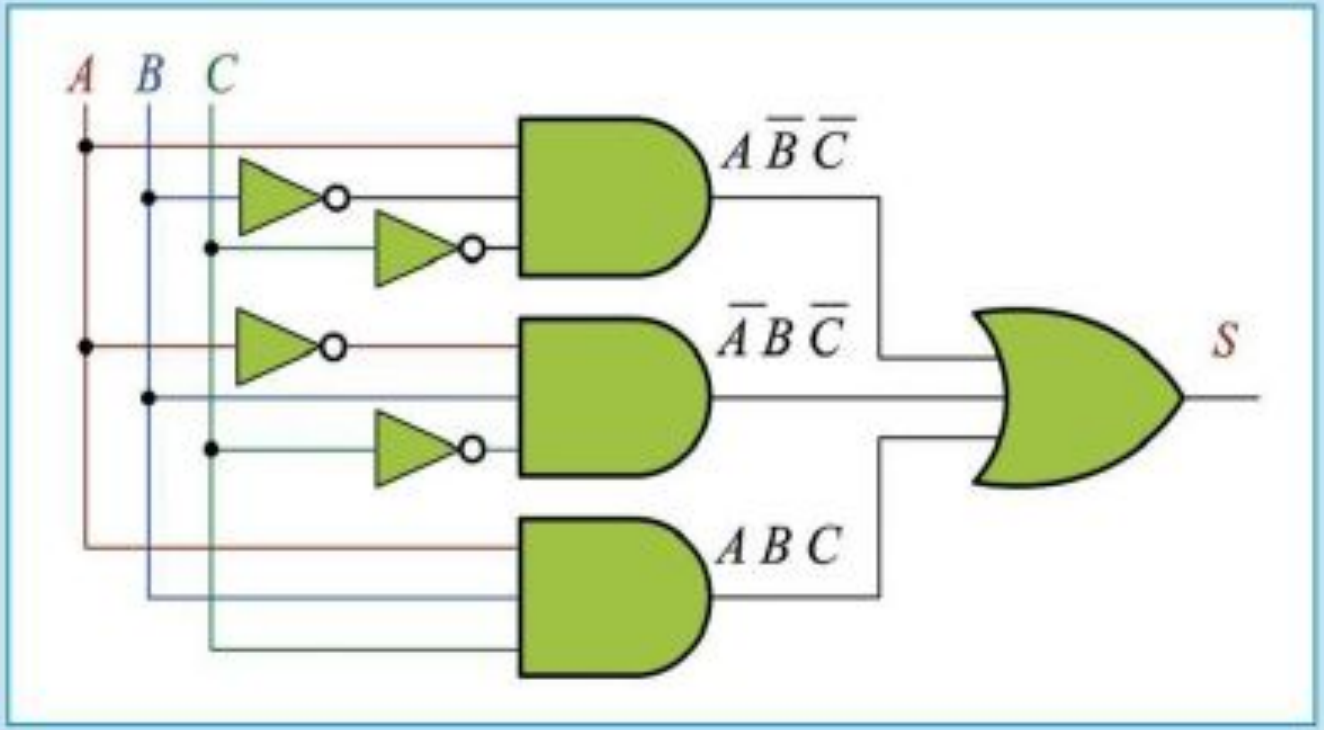
$A\ B\ C$	S	
0 0 0	0	
0 0 1	0	
0 1 0	0	
0 1 1	1	$\overline{A}\,B\,C$
1 0 0	0	
1 0 1	1	$A\,\overline{B}\,C$
1 1 0	0	
1 1 1	1	$A\,B\,C$

Si ahora tomamos solamente los valores de la salida que han dado como resultado un «1» lógico, podremos escribir la función que deberá realizar nuestro circuito lógico combinacional.

$$S = \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot C + A \cdot B \cdot C$$



Obtén la función y la tabla de la verdad del diagrama lógico de la Figura



Al observar la función que realiza cada puerta lógica individual obtenemos la función que se corresponde con la salida:

$$S = A \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot B \cdot C$$

Para realizar la tabla de la verdad de forma más sencilla, situaremos los términos parciales de la función con las salidas obtenidas en las diferentes puertas lógicas

$A\,B\,C$	$A\,\overline{B}\,\overline{C}$	$\overline{A}\,B\,\overline{C}$	ABC	S
0 0 0	0	0	0	0
0 0 1	0	0	0	0
0 1 0	0	1	0	1
0 1 1	0	0	0	0
1 0 0	1	0	0	1
1 0 1	0	0	0	0
1 1 0	0	0	0	0
1 1 1	0	0	1	1

4.- Construcción de puertas lógicas con circuitos integrados

Para la fabricación de las puertas lógicas se utilizan los circuitos integrados (CI).

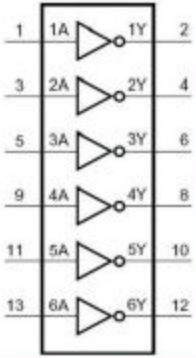
Estos están formados por un conjunto de componentes electrónicos (resistencias, diodos, transistores) integrados en una sola pieza de material semiconductor a base de silicio e insertada en el interior de un encapsulado.

Los circuitos integrados han ido evolucionando con el tiempo, consiguiéndose integrar cada vez un mayor número de puertas lógicas en un solo componente o **CI.**

Escala de integración	N.º de puertas
SSI - Integración a pequeña escala	Menos de 12
MSI - Integración a media escala	12 a 99
LSI - Integración a gran escala	100 a 9.999
VLSI - Integración a escala muy grande	10.000 a 99.999
ULSI - Integración a escala ultragrande	100.000 a 999.999
GSI - Integración a gigaescala	1.000.000 o más

SSI

Las entradas y salidas están directamente conectadas a los pins, como por ejemplo el CI 7404 que contiene 6 puertas NOT.



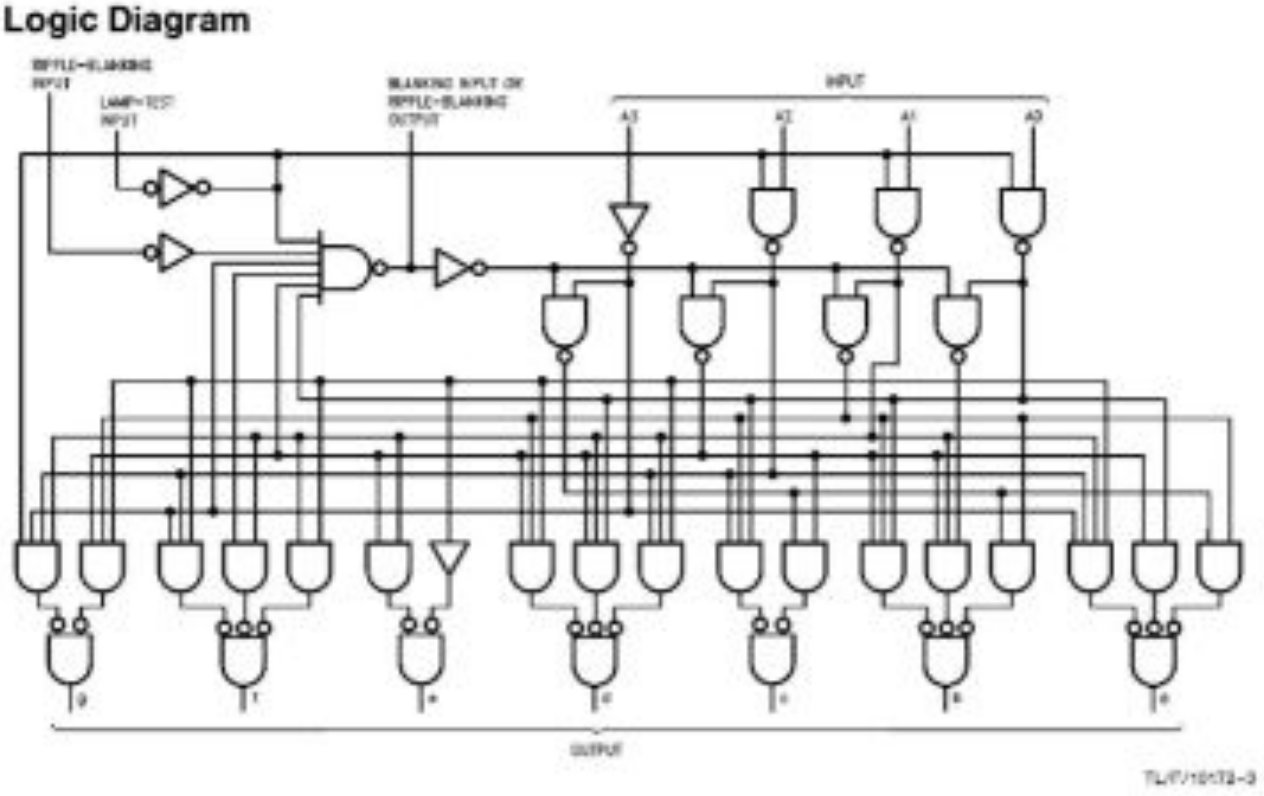
MSI

Hasta 100 puertas. Realizan una tarea específica simple, como por ejemplo un codificador de BCD a decimal.

GSI

Puede llegar a contener millones de puertas, como por ejemplo la CPU de un ordenador.

BCD a 7 segmentos



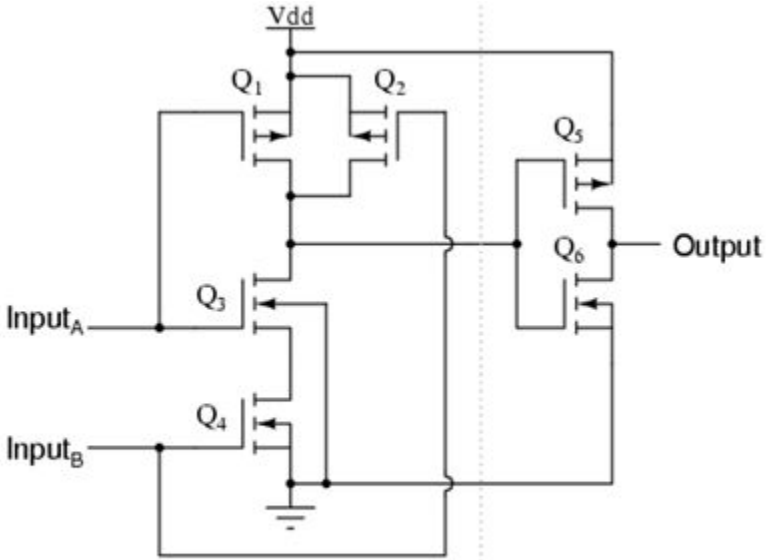
Ejemplos de encapsulados

Familias lógicas

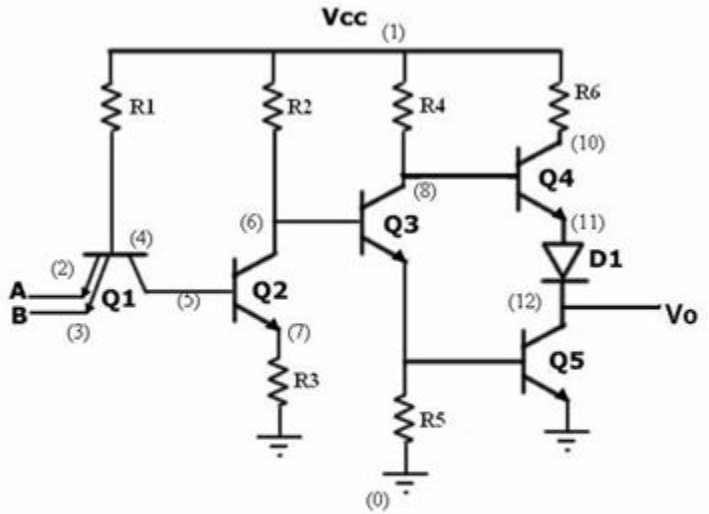
Una **familia lógica** es el conjunto de todos los componentes lógicos fabricados con la misma tecnología.

RTL	*Resistor-Transistor Logic*
DTL	*Diode-Transistor Logic*
ECL	*Emitter-Coupled Logic*
TTL	*Transistor-Transistor Logic*
CMOS	*Complementary Metal-Oxide Semiconductor*
BiCMOS	*Bipolar Complementary Metal-Oxide Semiconductor*

Las dos familias lógicas más utilizadas en la actualidad son la TTL basada en los transistores bipolares, y MOSFET basada en los transistores unipolares de efecto de campo.



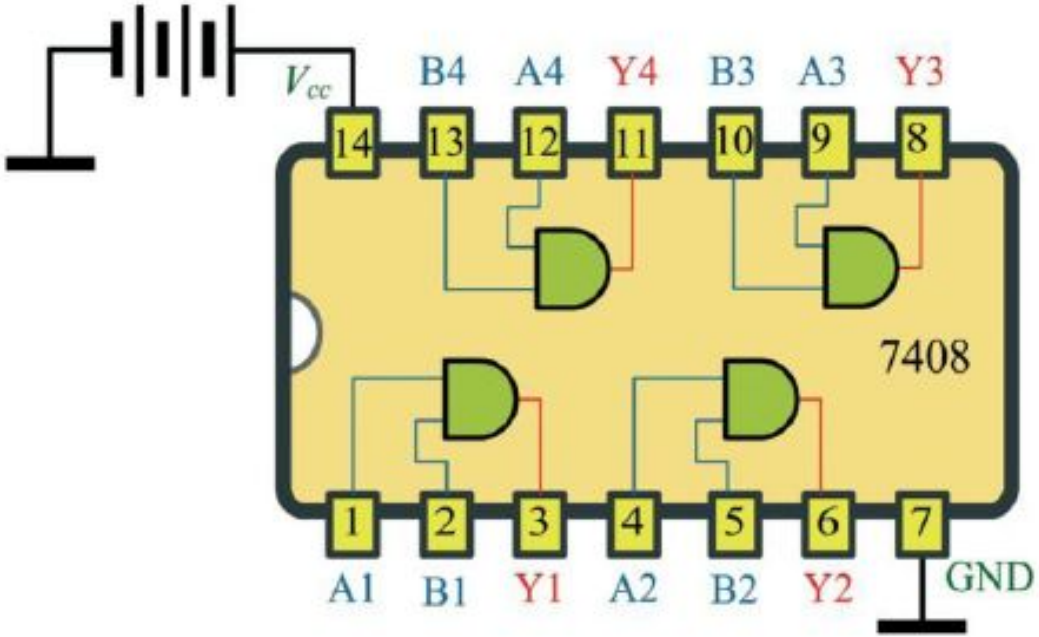
CMOS Logic



TTL Logic

5.- Características de una familia lógica

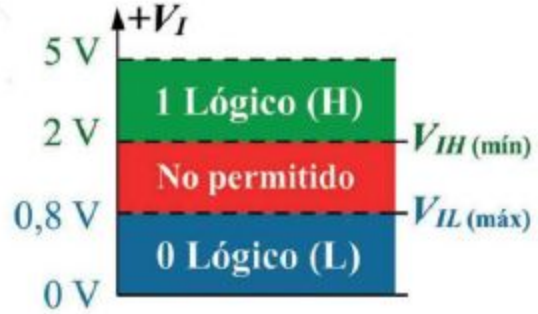
Tensión de alimentación (V_{CC})



Niveles de tensión de entrada y salida

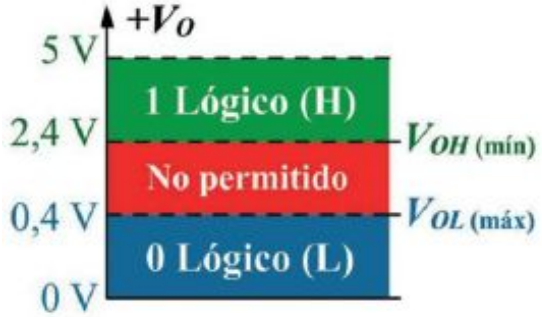
$V_{IL(\text{máx})}$ = máxima tensión de entrada (I) admisible para que la puerta interprete un «0» lógico o un nivel bajo (L).

$V_{IH(\text{mín})}$ = mínima tensión de entrada (I) admisible para que la puerta interprete un «1» lógico o un nivel alto (H).



$V_{OL(\text{máx})}$ =máxima tensión que aparece en la salida (O) para el estado lógico «0» o nivel bajo (L).

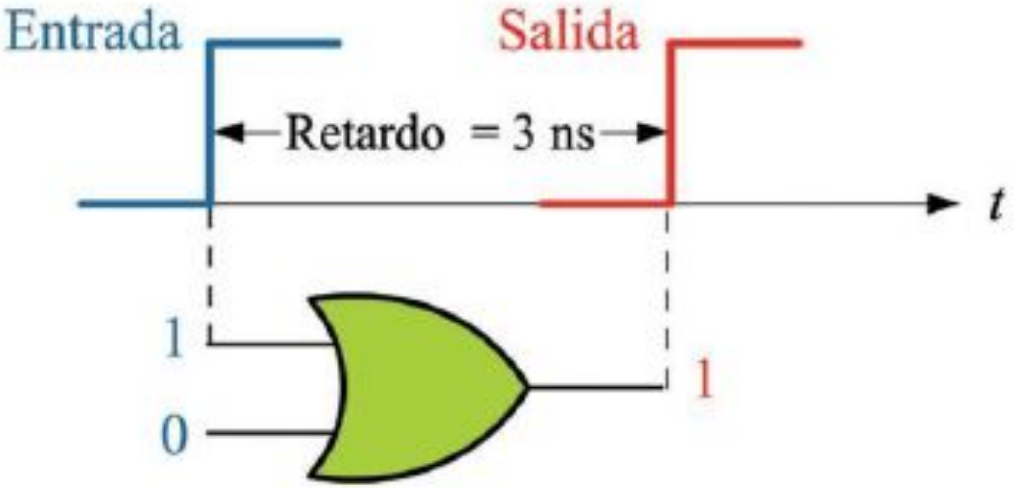
$V_{OH(\text{mín})}$ =mínima tensión que aparece en la salida (O) para el estado lógico «1» o nivel alto (H).



Disipación de potencia

Así, por ejemplo, la familia TTL 74 posee una disipación de potencia por puerta de 10 mW, mientras que la familia CMOS 74HC posee una disipación de potencia mucho menor, del orden de 0,0025 mW.

Retardo de propagación

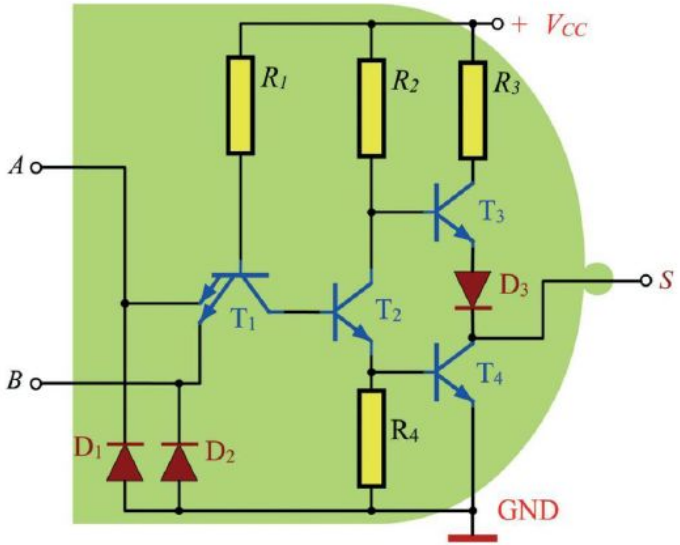


Comparativa entre las familias lógicas

		TTL 74	CMOS 74HC
Tensión de alimentación	V_{CC} (V)	4,5-5,5	3-15
Margen de ruido	V_{NH} (V)	0,4	1,4
	V_{NL} (V)	0,4	0,9
Potencia consumida	P (mW)	10	0,0025
Capacidad de carga	Fan-out	10	100
Tiempo de propagación	t_p (ns)	9	8
Frecuencia máxima	f (MHz)	35	40
Niveles de tensión de entrada	$V_{IL(máx)}$	0,8	1
	$V_{IH(mín)}$	2	3,5
Niveles de tensión de salida	$V_{OL(máx)}$	0,4	0,1
	$V_{OH(mín)}$	2,4	4,9

Familia lógica TTL

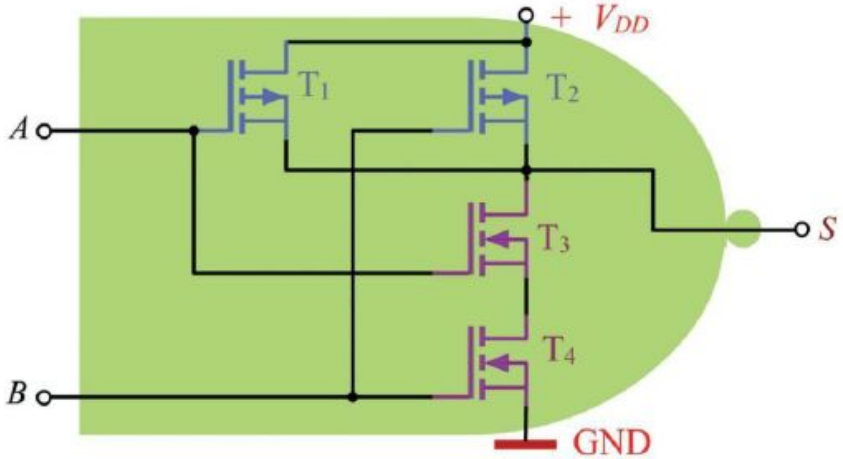
La familia TTL (*Transistor-Transistor-Logic*), que proviene del término lógica de transistor a transistor, está constituida por resistencias, diodos y transistores bipolares.



puerta NAND de dos entradas.

Familia lógica CMOS

La familia CMOS (*Complementary Metal-Oxide Semiconductor*) construye sus puertas lógicas con transistores unipolares MOSFET de canal N y de canal P



Puerta NAND CMOS.

Ejercicios de repaso

Calcula el valor decimal de los siguientes números binarios:

- 111111_2.
- 101010101_2.
- 10111_2.

Calcula el valor binario de los siguientes números decimales:

- 1.458_{10}.
- 324_{10}.
- 86_{10}.

$$111111_2 = 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 =$$
$$1 \cdot 32 + 1 \cdot 16 + 1 \cdot 8 + 1 \cdot 4 + 1 \cdot 2 + 1 \cdot 1 = 63_{10}$$

$$101010101_2 = 1 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 =$$
$$1 \cdot 256 + 0 \cdot 128 + 1 \cdot 64 + 0 \cdot 32 + 1 \cdot 16 + 0 \cdot 8 + 1 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 = 341_{10}$$

$$10111_2 = 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 =$$
$$1 \cdot 16 + 0 \cdot 8 + 1 \cdot 4 + 1 \cdot 2 + 1 \cdot 1 = 23_{10}$$

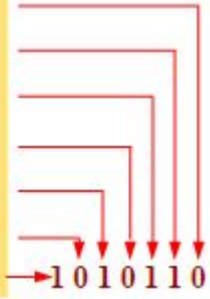
División = Cociente	Resto
1458/2 = 729	0
729/2 = 364	1
364/2 = 182	0
182/2 = 91	0
91/2 = 45	1
45/2 = 22	1
22/2 = 11	0
11/2 = 5	1
5/2 = 2	1
2/2 = 1	0
1/2 = 0	1

Solución: 10110110010

División = Cociente	Resto
324/2 = 162	0
162/2 = 81	0
81/2 = 40	1
40/2 = 20	0
20/2 = 10	0
10/2 = 5	0
5/2 = 2	1
2/2 = 1	0
1/2 = 0	1

Solución: 101000100

División	Cociente	Resto
86 : 2 =	43	0
43 : 2 =	21	1
21 : 2 =	10	1
10 : 2 =	5	0
5 : 2 =	2	1
2 : 2 =	1	0
1 : 2 =	0	1



Ejercicios de repaso

Convierte en código binario:

100110010010000_{BCD}

Convierte el número binario en código BCD

11001010.

1010101110.

Ejercicios de repaso

Escribe la función de salida y la tabla de la verdad que se corresponde con el circuito lógico de la Figura