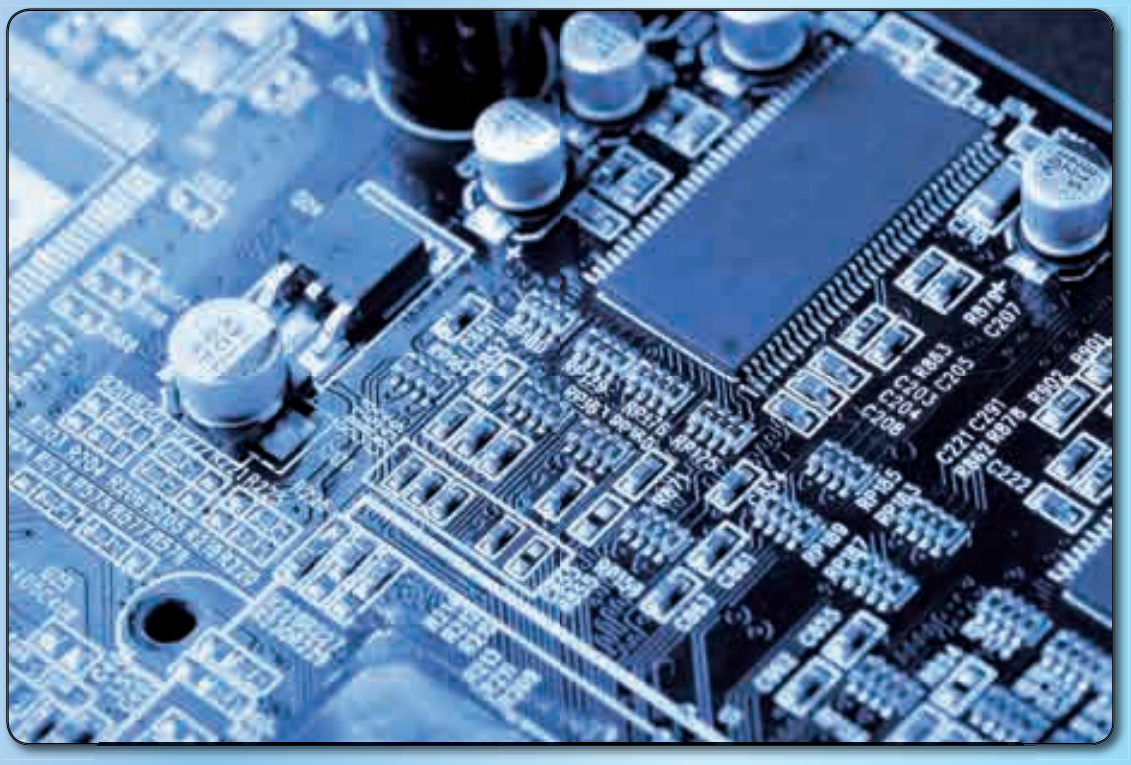
4 Circuitos y sistemas lógicos

La electrónica se divide en dos grandes áreas: la analógica y la digital. Esta última (objeto de esta unidad), gracias a los avances en el terreno de la integración de componentes en un solo chip, ha adquirido gran importancia y ha generado cambios importantes en las etapas de proceso y control de los sistemas electrónicos.



Conoce

1. **Electrónica digital**
2. **Sistemas de numeración**
3. **Álgebra de Boole**
4. **Puertas lógicas**
5. **Niveles lógicos**
6. **Obtención de la tabla de verdad de una función lógica**
7. **Simplif cación de funciones**
8. **Resolución de problemas y diseño de circuitos**
9. **Circuitos combinacionales integrados**

Practica paso a paso

Práctica 1. Simulación de circuitos con puertas lógicas

Taller

Práctica 1. Montaje de un circuito de control con puertas lógicas

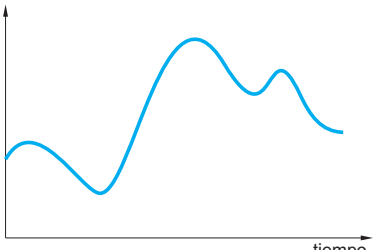
Ejercicios resueltos

Repasa

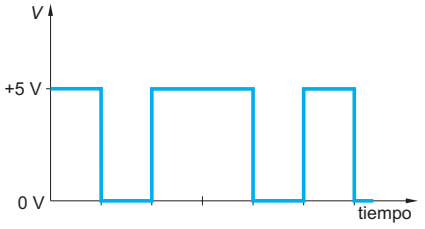
Refuerza

1. Electrónica digital

En la electrónica analógica se trabaja con **señales analógicas**, que son aquellas que pueden tomar un número infinito de valores. En cambio, en la electrónica digital trabajamos con **señales digitales**, que utilizan valores discretos, es decir, toman un número finito de valores.



Señal analógica.



Señal digital.

Podemos clasificar los circuitos de lógica digital en combinacionales y secuenciales:

- En los **circuitos combinacionales**, la salida únicamente depende de la combinación de las entradas.
- En los **circuitos secuenciales**, la salida no se puede conocer siempre conociendo la entrada, sino que es necesario saber la historia del circuito, es decir, la salida depende de la combinación de entradas y del estado anterior del circuito.

2. Sistemas de numeración

Los **sistemas de numeración** son los distintos conjuntos de símbolos capaces de representar la información numérica. Cada uno de ellos hace referencia a la **base** del sistema de numeración, que representa la cantidad de dígitos (símbolos distintos) que se utilizan para representar todos los números.

Los principales sistemas son:

Base	Número de dígitos
Base 10 o decimal	Diez dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
Base 2 o binario	Dos dígitos (0, 1)
Base 8 u octal	Ocho dígitos (0, 1, 2, 3, 4, 5, 6, 7)
Base 16 o hexadecimal	Dieciséis dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)

Sistema binario

En la lógica digital se emplean únicamente **dos tipos de estados** (0 y 1), también llamados *falso* y *verdadero*, *bajo* y *alto*, etc. Por ello, en los circuitos de lógica digital se emplea el sistema binario para codificar la información.

La unidad mínima de información es el **bit**, que corresponde a un solo digito. Y se denomina **byte** al conjunto de 8 bits seguidos; esta es la unidad fundamental de información de base utilizada en informática y telecomunicaciones (también se suele decir que es el tamaño de la "palabra" en un ordenador).

En el sistema binario, los múltiplos del byte se basan en potencias de 2^{10}. Así obtenemos:

byte	8 bits	$2^0 = 1$ byte
kilobyte	1.024 bytes	$2^{10} = 1.024$ bytes
megabyte	1.024 KB	$2^{20} = 1.048.576$ bytes
gigabyte	1.024 MB	$2^{30} = 1.073.741.824$ bytes
terabyte	1.024 GB	$2^{40} = 1.099.511.627.776$ bytes
petabyte	1.024 TB	$2^{50} = 1.125.899.906.842.624$ bytes

$$4732_{10} = 4 \cdot 10^3 + 7 \cdot 10^2 + 3 \cdot 10^1 + 2 \cdot 10^0$$

Fig. 1

Sistema decimal

En el sistema decimal empleamos **diez dígitos** para expresar cualquier cantidad, en forma de suma de potencias de 10, es decir, 10^n. Por ejemplo, el número 4.732 se puede expresar en potencias de 10 tal como muestra la figura 1.

$$110011_2 = 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 =$$
$$= 1 \cdot 32 + 1 \cdot 16 + 0 \cdot 8 + 0 \cdot 4 + 1 \cdot 2 + 1 \cdot 1 = 51_{10}$$

Fig. 2

Conversión del sistema binario al decimal

Seguimos el mismo procedimiento que para descomponer un número decimal, pero sustituyendo las potencias de 10^n por las de 2^n (figura 2).

Potencia de dos	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Decimal	64	32	16	8	4	2	1
Ejemplo: **53** =	0 +	32 +	16 +	0 +	4 +	0 +	1
	0	**1**	**1**	**0**	**1**	**0**	**1**

Fig. 3

Conversión del sistema decimal al binario

Podemos utilizar dos formas de transformación:

1. Colocar en una tabla todas las potencias e ir sumándolas hasta que la suma dé el número en decimal (figura 3).
2. Realizar divisiones sucesivas por la base binaria 2, guardar el resto de cada una de ellas y después colocar los ceros y unos en orden inverso (figura 4).

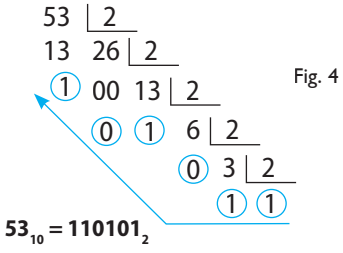


Fig. 4

$53_{10} = 110101_2$

Sistema hexadecimal

Es un sistema de numeración con base 16. Es el que emplean los microprocesadores, ya que parte de la base del byte (8 caracteres). Los 16 caracteres que componen el sistema hexadecimal son:

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Hexadecimal
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Digital

Conversión del sistema binario al hexadecimal

Agrupamos las cifras del número binario de cuatro en cuatro empezando por la derecha (se pueden añadir ceros por la izquierda hasta completar el grupo de cuatro). Convertimos cada grupo en número decimal. Cada número decimal obtenido lo convertimos a hexadecimal según la tabla de equivalencia (figura 5).

$$11010101_2 =$$
$$= \underbrace{1101}\ \underbrace{0101} =$$
$$= 13 \quad 5 =$$
$$= D \quad 5 = D5_{16}$$

Fig. 5

Conversión del sistema hexadecimal al binario

Se realiza el proceso contrario: se va sustituyendo cada carácter hexadecimal por su correspondiente número decimal. Cada uno de esos números decimales se sustituye por su equivalente grupo de cuatro bits en binario (figura 6).

$$27FA_{16} =$$
$$= 2 \quad 7 \quad F \quad A =$$
$$= 2 \quad 7 \quad 15 \quad 10 =$$
$$= 0010\ 0111\ 1111\ 1010_2$$

Fig. 6

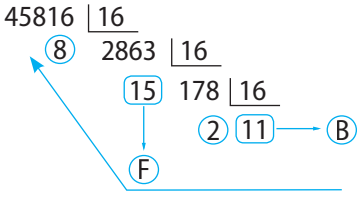
Conversión del sistema hexadecimal al decimal

Al igual que en la transformación de binario a decimal, se multiplica el número hexadecimal por las potencias, ahora de 16. Después se realiza la suma. Ejemplo:

$$B2F8_{16} = B \cdot 16^3 + 2 \cdot 16^2 + F \cdot 16^1 + 8 \cdot 16^0 = 45056 + 512 + 240 + 8 = 45816_{10}$$

Conversión del sistema decimal al hexadecimal

Se realiza con el mismo método que la conversión de decimal a binario, pero dividiendo ahora entre 16 (figura 7).



11-2-15-8 = B2F8

Fig. 7

$45816_{10} = B2F8_{16}$

3. Álgebra de Boole

El **álgebra de Boole** es una estructura matemática que nos permite representar un sistema electrónico digital matemáticamente mediante una función lógica.

Los valores que pueden tomar las variables de las funciones lógicas son el 0 y el 1 lógicos.

La función lógica puede representarse mediante una **expresión algebraica** o mediante su **tabla de verdad**. También podemos representar el circuito equivalente mediante interruptores o mediante puertas lógicas.

$F = a \cdot b$
Expresión algebraica.

a	b	F = a + b	F = a · b	$\overline{a}$
0	0	0	0	1
0	1	1	0	1
1	0	1	0	0
1	1	1	1	0

Tabla de verdad.

Leyes, propiedades y teoremas del álgebra de Boole

Mediante estas leyes, teoremas y propiedades podremos simplificar las funciones.

Las **leyes** conmutativa, asociativa y distributiva, para la suma y la multiplicación, son las mismas del álgebra ordinaria:

Ley conmutativa

$a + b = b + a$ $\qquad$ $a \cdot b = b \cdot a$

Ley asociativa

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

Ley distributiva

$a \cdot (b + c) = ab + ac$ $\qquad$ $a + (b \cdot c) = (a + b) \cdot (a + c)$

A continuación mostramos una serie de **propiedades básicas** que nos van a resultar muy útiles para simplificar funciones:

$a + 0 = a$	$a \cdot 0 = 0$
$a + 1 = 1$	$a \cdot 1 = a$
$a + a = a$	$a \cdot a = a$
$a + \overline{a} = 1$	$a \cdot \overline{a} = 0$
$\overline{\overline{a}} = a$	
$a \cdot (a + b) = a$	$(a + b) \cdot (a + \overline{b}) = a$
$a + (a \cdot b) = a$	$(a \cdot b) + (a \cdot \overline{b}) = a$

Por último, los dos **teoremas de Morgan** nos permitirán intercambiar las funciones de suma y producto, lo cual nos será muy útil a la hora de implementar los circuitos con puertas lógicas:

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

Escanea el código para ver cómo aplicar el **álgebra de Boole** en la simplificación de funciones.

EJERCICIOS

1. ✦✦ Haz las siguientes conversiones:

a.	$236_{10} =$ _____$_{16}$	b.	$AC3_{16} =$ _____$_{10}$	c.	$425_8 =$ _____$_2$
d.	$25D_{16} =$ _____$_2$	e.	$128_{10} =$ _____$_2$	f.	$0110100_2 =$ _____$_{16}$

4. Puertas lógicas

Circuitos integrados con puertas lógicas.

En este libro vamos a representar las puertas lógicas mediante los símbolos distintivos **tradicionales**. Sin embargo, también podemos encontrar los símbolos **rectangulares** en muchas ocasiones, por lo cual indicamos la equivalencia. Los símbolos lógicos rectangulares cumplen la normativa ANSI/IE EE91-1984.

Las **puertas lógicas** son circuitos electrónicos digitales, integrados en un chip, que realizan las operaciones lógicas básicas: suma lógica, producto lógico y negación.

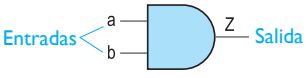
Internamente están compuestas fundamentalmente por combinaciones de transistores y resistencias.

Cada circuito integrado contiene varias puertas de un mismo tipo. Distintos tipos de puertas lógicas conectados adecuadamente entre sí forman los **circuitos lógicos de control**, a través de los cuales se implementa o genera físicamente una **función lógica**. Las entradas y salidas de las puertas lógicas solamente alcanzan **dos niveles de tensión**: nivel alto (que se representa mediante un 1) y nivel bajo (que se representa mediante un 0).

A continuación se presentan las puertas lógicas básicas (OR, AND y NOT), así como las puertas NOR y NAND, que son una combinación de las anteriores y facilitan el montaje de los circuitos electrónicos digitales.

Los circuitos integrados digitales se fabrican utilizando diferentes tecnologías, llamadas **familias lógicas**.

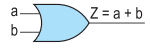
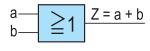
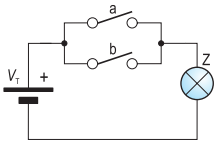
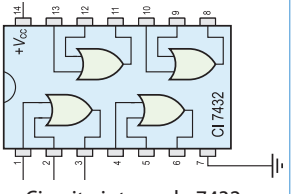
Las familias lógicas más utilizadas en la actualidad son la **TTL** y la **CMOS**.



Puerta lógica OR

La puerta lógica OR es una puerta formada por dos o más entradas y una salida y es el circuito lógico que realiza la **función lógica OR** o **suma lógica (+)**.

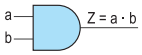
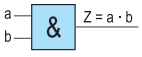
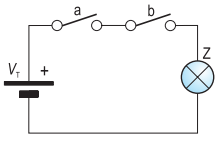
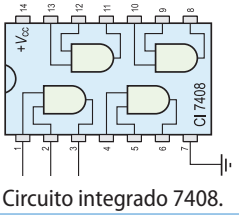
Su funcionamiento es el siguiente: la salida alcanza un nivel de tensión alto (1) si una o más de las entradas tiene un nivel de tensión alto.

Símbolos	Función lógica que realiza	Tabla de verdad	Circuito eléctrico equivalente	Circuito integrado formado por puertas OR
 Símbolo tradicional.  Según la norma ANSI.	$Z = a + b$	a b Z 0 0 0 0 1 1 1 0 1 1 1 1		 Circuito integrado 7432.

Puerta lógica AND

La puerta lógica AND consta de dos o más entradas y una salida y realiza la **función lógica AND** o **producto lógico (·)**.

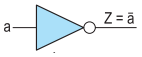
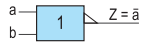
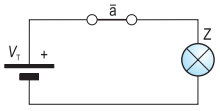
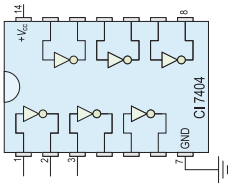
Funciona de la siguiente manera: la salida alcanza un nivel alto de tensión (1) solamente cuando todas las entradas están a nivel de tensión alto.

Símbolos	Función lógica que realiza	Tabla de verdad	Circuito eléctrico equivalente	Circuito integrado formado por puertas AND
 Símbolo tradicional.  Según la norma ANSI.	$Z = a \cdot b$	a b Z 0 0 0 0 1 0 1 0 0 1 1 1		 Circuito integrado 7408.

Puerta lógica NOT

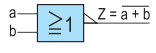
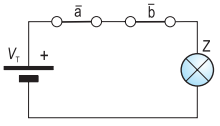
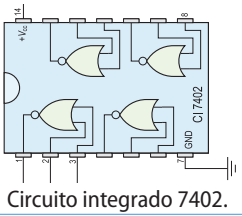
La puerta lógica NOT está formada por una entrada y una salida y es el circuito lógico que realiza la **función lógica complemento o negación**.

Su funcionamiento se basa en que la salida toma siempre el valor contrario a la entrada, es decir, si la entrada está a un nivel de tensión alto, la salida estará a un nivel de tensión bajo y viceversa.

Símbolos	Función lógica que realiza	Tabla de verdad	Circuito eléctrico equivalente	Circuito integrado formado por puertas NOT
 Símbolo tradicional.  Según la norma ANSI.	$Z = \overline{a}$	a | Z 0 | 1 1 | 0		 Circuito integrado 7404.

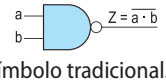
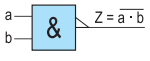
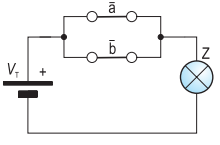
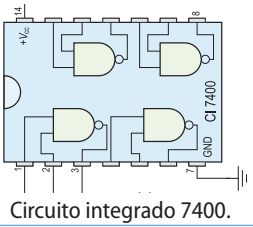
Puerta lógica NOR

La puerta lógica NOR es una **combinación** de las puertas **OR y NOT**. Realiza la función inversa de la puerta OR, es decir, su salida estará a un nivel de tensión alto (1) cuando todas sus entradas estén a un nivel bajo de tensión (0).

Símbolos	Función lógica que realiza	Tabla de verdad	Circuito eléctrico equivalente	Circuito integrado formado por puertas NOR
 Símbolo tradicional.  Según la norma ANSI.	$Z = \overline{a + b}$	a | b | OR | Z (NOR) 0 | 0 | 0 | 1 0 | 1 | 1 | 0 1 | 0 | 1 | 0 1 | 1 | 1 | 0		 Circuito integrado 7402.

Puerta lógica NAND

La puerta lógica NAND es una **combinación** de las puertas **AND y NOT**. Realiza la función inversa de la puerta AND, es decir, su salida estará a un nivel de tensión alto (1) cuando alguna de las entradas esté a un nivel de tensión bajo (0).

Símbolos	Función lógica que realiza	Tabla de verdad	Circuito eléctrico equivalente	Circuito integrado formado por puertas NAND
 Símbolo tradicional.  Según la norma ANSI.	$Z = \overline{a \cdot b}$	a | b | OR | Z (NAND) 0 | 0 | 0 | 1 0 | 1 | 0 | 1 1 | 0 | 0 | 1 1 | 1 | 1 | 0		 Circuito integrado 7400.

EJERCICIOS

2. ✦ Dibuja los símbolos de la puerta lógica NOT según la simbología tradicional y según la norma ANSI.

3. ✦ Dibuja la tabla de verdad y el circuito equivalente de la puerta lógica NOR.

4. ✦ Dibuja el circuito integrado formado por puertas lógicas NAND.

5. ✦✦ ¿Con qué tensión V_{CC} debe alimentarse una puerta lógica de la familia TTL?

La **familia lógica TTL** tiene las siguientes características:

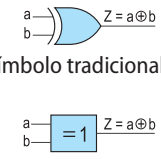
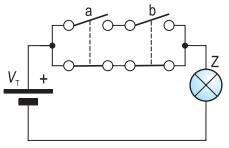
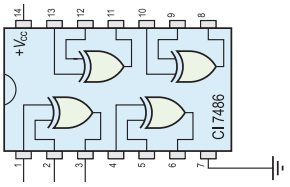
Tensión de alimentación: V_{CC} = 5 V	
Niveles de tensión de entrada (V_I)	
1 lógico	**0** lógico
2 V ≤ V_I ≤ 5 V	0 V ≤ V_I ≤ 0,8 V
Niveles de tensión de salida (V_O)	
1 lógico	**0** lógico
2,4 V ≤ V_O ≤ 5 V	0 V ≤ V_O ≤ 0,4 V

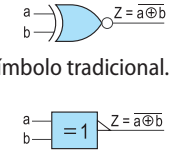
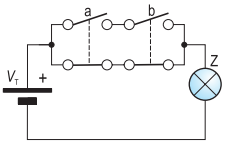
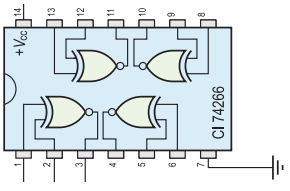
Escanea el código para ver más explicaciones sobre **puertas lógicas**.

Puertas lógicas XOR (u OR exclusiva) y XNOR (o NOR exclusiva)

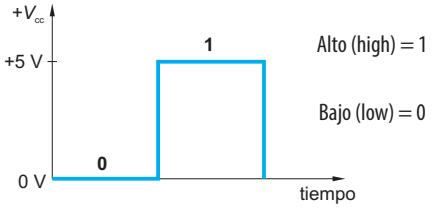
La **puerta lógica XOR** tiene sólo dos entradas. Su funcionamiento es el siguiente: la salida de una puerta XOR se pone a nivel alto sólo cuando las dos entradas están a **niveles lógicos opuestos**.

La **puerta lógica XNOR**, en cambio, se pone a nivel alto cuando las dos entradas tienen el **mismo nivel lógico**.

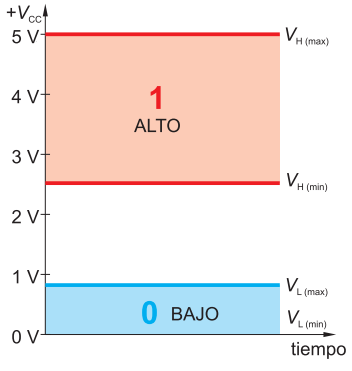
Símbolos	Función lógica que realiza	Tabla de verdad	Circuito eléctrico equivalente	Circuito integrado formado por puertas XOR
 Símbolo tradicional. Según la norma ANSI.	$Z = a \oplus b = a\bar{b} + \bar{a}b$	a b Z (XOR) 0 0 0 0 1 1 1 0 1 1 1 0		 Circuito integrado 7486.

Símbolos	Función lógica que realiza	Tabla de verdad	Circuito eléctrico equivalente	Circuito integrado formado por puertas XNOR
 Símbolo tradicional. Según la norma ANSI.	$Z = \overline{a \oplus b} = \bar{a}\bar{b} + ab$	a b Z (XNOR) 0 0 1 0 1 0 1 0 0 1 1 1		 Circuito integrado 74266.

5. Niveles lógicos



Lógica positiva.



Niveles de la familia lógica TTL.

Los circuitos digitales trabajan únicamente con dos niveles de tensión (alto o bajo), que se asignan a los elementos lógicos: el 1 lógico y el 0 lógico.

Utilizamos dos tipos de lógica: positiva y negativa.

- Trabajamos en **lógica positiva** cuando los valores de tensión cercanos a $+V$ (voltaje de alimentación) se consideran nivel alto y los que están cercanos a 0 voltios se consideran nivel bajo, es decir, cuando al nivel 1 se le asigna un valor de tensión más positivo y al 0 el más bajo.
- Trabajamos en **lógica negativa** cuando al nivel 1 se le asigna un valor de tensión más negativo que al 0.

Cuando definimos circuitos o trabajamos con ellos, tenemos que decir en qué lógica estamos trabajando. Normalmente será en lógica positiva.

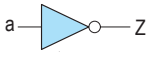
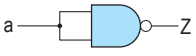
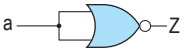
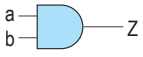
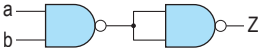
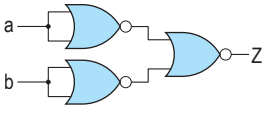
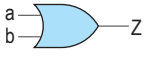
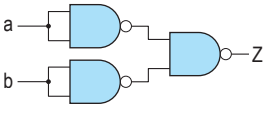
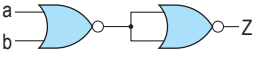
En realidad, los niveles altos y bajos no se definen por un único valor de voltaje, sino por una franja o zona en la que podemos considerar que tenemos un valor alto o bajo. La variable $V_{H\,(max)}$ representa el máximo valor para el nivel alto y $V_{H\,(min)}$ representa el valor mínimo para el nivel alto, mientras que el nivel bajo varía entre $V_{L\,(max)}$ y $V_{L\,(min)}$. Por ejemplo, en el caso de los circuitos TTL (*transistor-transistor logic*), si se aplica una tensión que varía entre 2,4 V y 5 V, se interpreta como un nivel alto, mientras que, si se aplica una tensión de entre 0 V y 0,8 V, se interpreta como un nivel bajo.

Obtención de funciones con puertas NAND y NOR

Podemos construir cualquier función lógica mediante una combinación de puertas NAND y NOR, por lo que éstas suelen ser las más utilizadas.

A la hora de montar un circuito con puertas lógicas debemos tratar de utilizar el menor número de chips. Por ello, como hemos visto antes, si cada chip lleva al menos cuatro puertas lógicas iguales, podremos economizar espacio y consumo construyendo nuestra función lógica únicamente con una combinación de puertas NAND y NOR.

Para poder dejar la función lógica como combinación de puertas NAND y NOR debemos aplicar los teoremas de Morgan las veces que sea necesario.

	Función	Puertas NAND	Puertas NOR	Teorema de Morgan
Negación				$Z=\overline{a\cdot a}=\overline{a}+\overline{a}=\overline{a}$ (NAND) $Z=\overline{a+a}=\overline{a}\cdot\overline{a}=\overline{a}$ (NOR)
AND				$Z=\overline{\overline{a\cdot b}}=ab$ (NAND) $Z=\overline{\overline{a}+\overline{b}}=ab$ (NOR)
OR				$Z=a+b=\overline{\overline{a+b}}=\overline{\overline{a}\,\overline{b}}$ (NAND) $Z=a+b=\overline{\overline{a+b}}$ (NOR)

6. Obtención de la tabla de verdad de una función lógica

La forma de representar las funciones lógicas es construir una **tabla de verdad** donde aparezcan recogidas todas las combinaciones de entradas y el valor que toma la salida para cada uno de los estados.

a	b	c	Z
0	0	0	
0	0	1	1
0	1	0	
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	
1	1	1	

Fig. 8.

Para ello debemos construir una tabla con tantas columnas como variables (n) y tantas filas como 2^n combinaciones de variables.

Por ejemplo, para la función $Z=a\overline{b}+\overline{a}c$, asignamos a la función Z los valores 1 que correspondan a la combinación de variables de entrada, al ser una suma de productos (figuras 8 y 9).

Una tabla de verdad tiene muchas expresiones. Las **formas canónicas** son dos expresiones normalizadas que nos resultan muy útiles a la hora de simplificar:

- **Primera forma canónica.** Es una suma de productos en que cada producto contiene todas las variables. Cada uno de estos productos recibe el nombre de ***minterm*** o **minitérmino**. El número de productos equivale al número de unos que tiene la tabla de verdad. Nuestra función Z en función de minterm quedará así:

$$Z=\overline{a}\,\overline{b}c+\overline{a}bc+a\overline{b}\,\overline{c}+a\overline{b}c$$

- **Segunda forma canónica.** Es un producto de sumas en que cada producto contiene todas las variables. Cada una de estas sumas recibe el nombre de ***maxterm*** o **maxitérmino**. El número de productos equivale al número de ceros que tiene la tabla de verdad. Nuestra función Z en función de maxterm quedará así:

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

a	b	c	Z	
0	0	0	0	$a+b+c$
0	0	1	1	$\overline{a}\,\overline{b}c$
0	1	0	0	$a+\overline{b}+c$
0	1	1	1	$\overline{a}bc$
1	0	0	1	$a\overline{b}\,\overline{c}$
1	0	1	1	$a\overline{b}c$
1	1	0	0	$\overline{a}+\overline{b}+c$
1	1	1	0	$\overline{a}+\overline{b}+\overline{c}$

Fig. 9.

7. Simplificación de funciones

Una función lógica se representa mediante una única tabla de verdad, aunque puede tener diferentes expresiones algebraicas o formas canónicas. Una vez obtenida la función lógica tendremos que simplificarla lo máximo posible para que el circuito sea, en la medida de lo posible, más sencillo y económico.

En esta unidad nos centraremos en el **método algebraico** y el **método gráfico de Karnaugh**.

7.1. Método algebraico

Utilizaremos las leyes, propiedades y teoremas del álgebra de Boole para simplificar lo máximo posible:

Por ejemplo:

$$Z = \overline{(c+b)\cdot(ca\overline{b})} = \overline{(c+b)} + \overline{(ca\overline{b})} = (\overline{c}\,\overline{b}) + (\overline{c} + \overline{a} + b) =$$
$$= \overline{c}b + \overline{c} + \overline{a} + b = \overline{c}\,(\overline{b} + 1) + \overline{a} + b = \overline{c}\cdot 1 + \overline{a} + b = \overline{c} + \overline{a} + b$$

7.2. Método gráfico de Karnaugh

El **método de Karnaugh** es uno de los métodos más sencillos de simplificación de funciones, con el que podemos generar expresiones suma de productos y productos de sumas lo más simples posibles.

a \ b	0	1
0	$\overline{a}\cdot\overline{b}$ 00	$\overline{a}\cdot b$ 01
1	$a\cdot\overline{b}$ 10	$a\cdot b$ 11

Fig. 10.

a \ bc	00	01	11	10
0	$\overline{a}\cdot\overline{b}\cdot\overline{c}$ 000	$\overline{a}\cdot\overline{b}\cdot c$ 001	$\overline{a}\cdot b\cdot c$ 011	$\overline{a}\cdot b\cdot\overline{c}$ 010
1	$a\cdot\overline{b}\cdot\overline{c}$ 100	$a\cdot\overline{b}\cdot c$ 101	$a\cdot b\cdot c$ 111	$a\cdot b\cdot\overline{c}$ 110

Fig. 11.

a \ bc	00	01	11	10
00	0000	0001	0011	0010
01	0100	0101	0111	0110
11	1100	1101	1111	1110
10	1000	1001	1011	1010

Fig. 12.

Situamos en una tabla los términos de las funciones (a, b, c, d, etc.), con la precaución de que los términos adyacentes en las filas y las columnas sólo se diferencian en una de sus variables, por lo cual siempre se disponen de la forma 00 - 01 - 11 - 10, tanto en las filas como en las columnas.

En las figuras 10, 11 y 12 podemos ver la disposición del mapa de Karnaugh para dos, tres y cuatro variables, respectivamente.

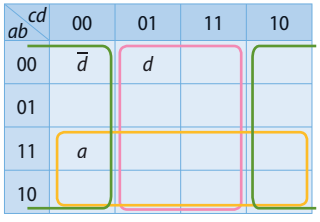
Una vez que hemos definido nuestra gráfica, vamos a ver cuál es el proceso para obtener una expresión suma de productos minimizada:

1. Mediante la tabla de verdad de la función obtenemos la forma canónica de minterms (suma de productos).
2. Con la función canónica que tenemos, vamos poniendo los unos en las celdas correspondientes al valor del producto.
3. Cuando hemos reflejado todos los unos, las celdas que no contienen unos las completamos con ceros.
4. Como lo que se pretende es obtener la función más simple, intentamos agrupar los unos adyacentes: primero, en grupos de ocho; los que queden, en grupos de cuatro; después, en grupos de dos; y, por último, habrá los que no se puedan agrupar.
5. En cada uno de los grupos formados, la variable (1 o 0) que cambia de valor se elimina y las variables que quedan se escriben asignando al 0 la variable negada y al 1 la variable directa.

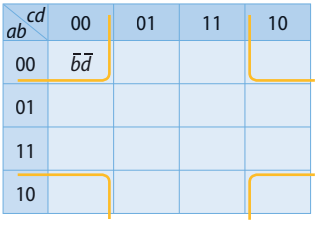
Como el objetivo es maximizar el tamaño de los grupos y minimizar el número de estos grupos, podemos observar las siguientes reglas:

- Los grupos deben ser potencias de 2^n, es decir, de 2, 4, 8... celdas.
- Siempre se debe incluir el mayor número posible de unos.
- Cada 1 del mapa tiene que estar incluido en al menos un grupo. Los unos que ya pertenezcan a un grupo pueden estar incluidos en otro, siempre que los grupos que se solapen contengan unos no comunes.

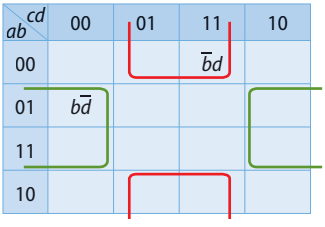
A continuación se muestran los distintos agrupamientos que se pueden formar. Hay que tener en cuenta que la tabla de Karnaugh se continúa de arriba abajo y de izquierda a derecha, como si fuese una esfera.



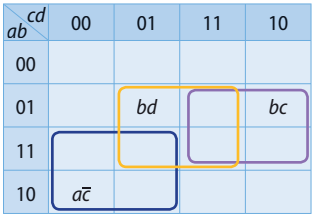
Agrupamiento de 8.



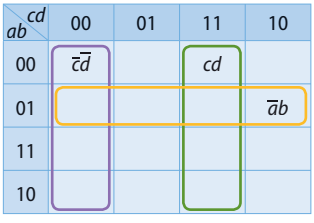
Agrupamiento de 4.



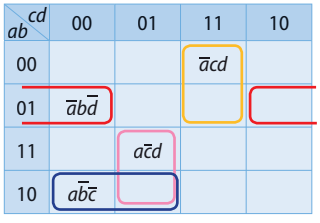
Agrupamiento de 4.



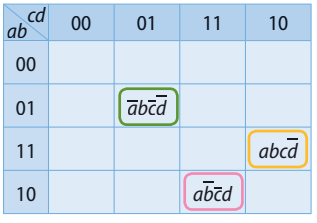
Agrupamiento de 4.



Agrupamiento de 4.



Agrupamiento de 2.



Agrupamiento de 1.

Cada agrupamiento da lugar al término simplificado que se muestra en su interior.

EJERCICIO RESUELTO

Dibuja la tabla de verdad de la siguiente función lógica suma de minterms, simplifícala y represéntala mediante puertas NAND o NOR: $Z(a, b, c) = \Sigma m(0, 2, 4, 6, 7)$. O, lo que es lo mismo: $Z = (\bar{a}\bar{b}\bar{c} + \bar{a}b\bar{c} + a\bar{b}\bar{c} + ab\bar{c} + abc)$.

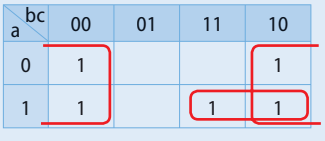
La forma canónica suma de minterms es la suma de productos. Cuando se representa la función como sumatorio, se indica la posición de la tabla de verdad que se corresponde con los unos (en este caso, las posiciones 0, 2, 4, 6 y 7).

Solución

Construimos la tabla de verdad:

Posición	a	b	c	Z
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Dibujamos la tabla de Karnaugh y simplificamos:

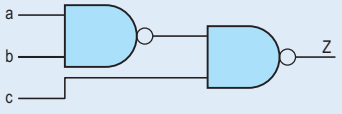


La función simplificada nos queda así:

$$Z = \bar{c} + ab$$

Para poder representarla con puertas NAND o NOR mediante los teoremas de Morgan usamos la doble negación de la función:

$$Z = \bar{c} + ab = \overline{\overline{\bar{c} + ab}} = \overline{\bar{\bar{c}} \cdot \overline{ab}} = \overline{c \cdot \overline{ab}}$$



8. Resolución de problemas y diseño de circuitos

Para resolver un problema de lógica digital y construir el correspondiente circuito electrónico debemos seguir los siguientes **pasos**:

Generalidades:

- La forma canónica tiene que tener todos los términos, todas las variables.
- Un circuito será una puerta determinada en lógica positiva o negativa.
- Varias salidas no se pueden juntar, pero una salida sí se puede unir a varias entradas.

1. Partimos de una situación inicial en la que **se describe el problema**. Declaramos las variables que necesitamos para definir el problema: *a*, *b*, *c*, etc.

   Por regla general, utilizamos lógica positiva y asignamos 0 a falso (apagado) y 1 a verdadero (encendido).

2. Dibujamos la **tabla de verdad** y completamos los unos y ceros.

3. Obtenemos la **función lógica** correspondiente.

   Si extraemos las expresiones donde tengamos unos, obtendremos la primera forma canónica (minterms) y escribiremos las variables como suma de productos.

   Si extraemos las expresiones donde tengamos ceros, obtendremos la segunda forma canónica (maxterms) y escribiremos las variables como producto de sumas.

4. **Simplificamos** la función lógica, bien mediante las propiedades del **álgebra de Boole**, bien mediante los **diagramas de Karnaugh**.

   Si extraemos y simplificamos los unos del diagrama de Karnaugh, obtendremos la función simplificada en forma de minterms (suma de productos).

   Si extraemos y simplificamos los ceros del diagrama de Karnaugh, obtendremos la función simplificada en forma de maxterms (producto de sumas).

5. Realizamos el **esquema de puertas lógicas**.

6. Dado que el objetivo es economizar el número de chips que se utilizan para construir los circuitos, aplicamos los teoremas de Morgan para construir el circuito lógico con **puertas NAND o NOR** y así reducir el número de componentes.

9. Circuitos combinacionales integrados

A continuación vamos a ver una serie de circuitos lógicos combinacionales que realizan cada uno de ellos una misión determinada. Aunque están constituidos por puertas lógicas, no estudiaremos la estructura interna sino la función que desempeñan y la expresión lógica.

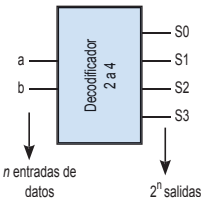
9.1. Decodificadores y codificadores

Decodificadores

Un **decodificador** es un circuito combinacional que tiene ***n* entradas** y **2^n salidas**. Su función consiste en que, cuando se presenta una determinada combinación binaria a la entrada, se activa una de las salidas, mientras que el resto permanecen desactivadas.

Los decodificadores se utilizan en muchos tipos de aplicaciones. Por ejemplo, en los ordenadores, para seleccionar los puertos de entrada/salida a través de los que se comunica con la impresora, el módem, el escáner, etc. Se emplea un decodificador para seleccionar el puerto de entrada/salida determinado por el ordenador, de forma que los datos puedan ser enviados o recibidos desde algún dispositivo externo concreto.

También se pueden usar para representar cifras en un display de 7 segmentos, para generar y convertir códigos, etc.



Símbolo del decodificador.

a	b	S_0	S_1	S_2	S_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Tabla de verdad del decodificador.

$$S_0 = \overline{a}\,\overline{b}$$
$$S_1 = \overline{a}\,b$$
$$S_2 = a\,\overline{b}$$
$$S_3 = ab$$

Expresión de salida del decodificador.

Codificadores

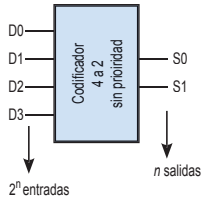
Un **codificador** es un circuito lógico combinacional que realiza la función inversa de la del codificador. Tiene ***n* salidas** y **2^n entradas**.

En la salida se muestra el código binario correspondiente a la línea de entrada activa en ese momento.

En los codificadores sin prioridad no puede activarse más de una entrada al mismo tiempo.

Los codificadores con prioridad permiten que varias entradas estén activas al mismo tiempo. En este caso, a la salida se codifica la entrada activa de mayor peso, es decir, de mayor valor decimal, mientras que el resto son ignoradas.

Los codificadores nos permiten transformar una señal o una información en una forma codificada usada para ser transmitida; por ejemplo, archivos multimedia para comprimir audio, imagen o vídeo.



Símbolo del codificador.

$$S_0 = D_2 + D_3$$
$$S_1 = D_1 + D_3$$

Expresión de un codificador sin prioridad.

D_0	D_1	D_2	D_3	S_0	S_1
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

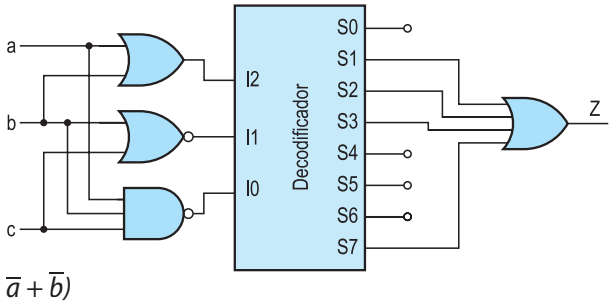
Tabla de verdad del codificador sin prioridad.

D_0	D_1	D_2	D_3	S_0	S_1
1	0	0	0	0	0
x	1	0	0	0	1
x	x	1	0	1	0
x	x	x	1	1	1

Tabla de verdad del codificador con prioridad.

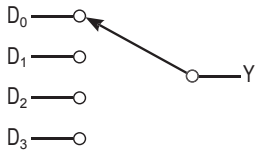
EJERCICIOS

6. ✦✦✦ Obtén la expresión lógica en forma de suma de minterms de la señal lógica *Z* como función de *a*, *b* y *c*. Simplifica dicha función por el método de Karnaugh.



(Solución: $Z = \overline{a} + \overline{b}$)

9.2. Multiplexores y demultiplexores



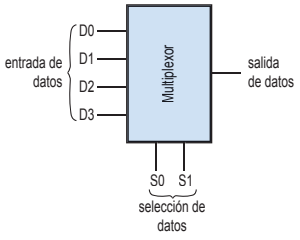
Multiplexores

Un **multiplexor** es un dispositivo que posee varias líneas de datos y una única línea de salida.

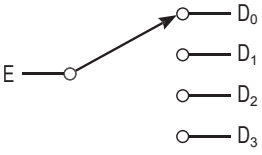
Permite dirigir la información digital procedente de diversas fuentes a esa única salida. Mediante unas entradas de control podemos seleccionar los datos y conmutarlos hacia la línea de salida. Es decir, se comporta como un conmutador de entradas múltiples y una salida única, pero con un control electrónico.

Tiene **2^n entradas**, **una salida** y ***n* líneas de control**.

$$Y = D_0\overline{S}_1\overline{S}_0 + D_1\overline{S}_1S_0 + D_2S_1\overline{S}_0 + D_3S_1S_0$$



Entradas de selección de datos		Entrada seleccionada
S_1	S_0	
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3



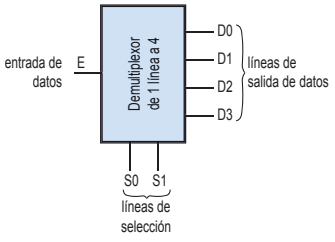
Demultiplexores

Un **demultiplexor** realiza la función contraria a la del multiplexor: toma los datos de una línea y los distribuye a un determinado número de líneas de salida, es decir, distribuye los datos.

Es un circuito integrado con **una entrada**, ***n* líneas de control** y **2^n salidas**.

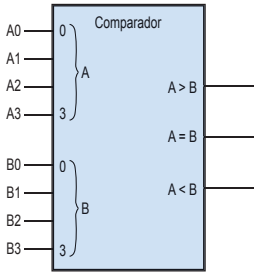
Un decodificador puede utilizarse como un demultiplexor.

Como se muestra en la tabla de verdad, el valor de la entrada *E* (0 o 1) se replicará en alguna de las salidas (D_0 a D_3), lo cual vendrá determinado por el valor de las líneas de control (S_0 y S_1).



Entradas de selección de datos		Salidas			
S_1	S_0	D_0	D_1	D_2	D_3
0	0	E	0	0	0
0	1	0	E	0	0
1	0	0	0	E	0
1	1	0	0	0	E

9.3. Comparadores



Un **comparador** es un circuito cuya función es comparar las magnitudes de dos cantidades binarias y determinar su relación.

Es decir, si queremos comparar dos números *A* y *B* que vienen definidos por dos grupos de *n* líneas de entradas, primero examinamos el bit de mayor orden de cada número y después los restantes. A la salida se activa la salida correspondiente: $A > B$, $A = B$ o $A < B$.

9.4. Sumadores

El complemento a 1 y el complemento a 2 de un número binario son importantes porque permiten la representación de números negativos. La aritmética en complemento a 2 se usa comúnmente en las computadoras para manipular los números negativos.

Sumadores

Un **sumador** es un circuito que se utiliza para sumar dos datos binarios. Si tenemos en cuenta el acarreo proveniente de una operación anterior, se denomina **sumador**; si no tenemos en cuenta el acarreo de entrada, se denomina **semisumador**.

El **acarreo**, cuando sumamos dos números binarios, es el bit resultante del desbordamiento:

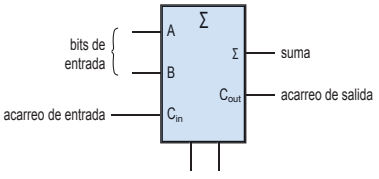
	1
+	1
1	0

1. Sumador de dos datos de un bit

La función lógica de salida será:

$$S = (A \oplus B) \oplus C_{in}$$

$$C_{out} = AB + (A \oplus B) \oplus C_{in}$$

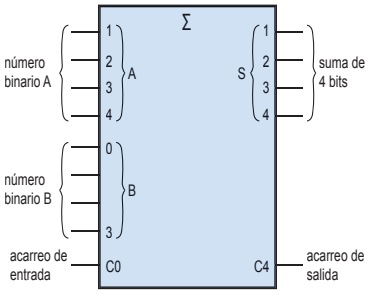


A	B	C_{in}	C_{out}	Σ
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Tabla de verdad de un sumador de 1 bit.

2. Sumador de datos de cuatro bits

Un sumador de 4 bits se implementa mediante cuatro sumadores de 1 bit. Primero se suman individualmente los dígitos binarios correspondientes al bit menos significativo, teniendo en cuenta el acarreo, y a continuación los de orden más alto.



C_{n-1}	A_n	B_n	S	C_n
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

Tabla para un sumador en paralelo de 4 bits.

Resta o diferencia

En los circuitos electrónicos digitales, para restar dos números realizamos el complemento a 2 de uno de ellos y después lo sumamos. De esta forma podemos hacer restas con un circuito sumador.

Para restar dos números utilizando el **complemento a 2** se siguen los siguientes pasos:

1. Pasamos los dos números a binario.
2. Si un número tiene menos bits que otro, se añaden ceros por delante.
3. En el sustraendo (número negativo), cambiamos los unos por los ceros y los ceros por los unos.
4. Sumamos al resultado el valor 1.
5. Añadimos delante el **bit de signo** (si es positivo, un 0; si es negativo, un 1). Es decir, normalmente, un 1 delante del sustraendo.
6. Sumamos normalmente. Si el resultado es positivo, el bit de signo será 0. Si es negativo y el bit de signo es 1.

$$\begin{array}{r} 29 \\ -\,12 \\ \hline 17 \end{array} \qquad \begin{array}{l} 29_{10} = 11101_2 \Rightarrow 011101_2 \\ 12_{10} = 1100_2 \Rightarrow 001100_2 \end{array}$$

Complemento a 2 del número 12:

$001100_2 \Rightarrow 110011 + 1 = 110100_{C2}$

bit de signo

$$\begin{array}{r} 0\,1\,1\,1\,0\,1 \\ +\,1\,1\,0\,1\,0\,0 \\ \hline \text{acarreo} \rightarrow \boxed{1}\;0\,1\,0\,0\,0\,1_2 \rightarrow (17)_{10} \end{array}$$

Ejemplo de resta de dos números con complemento a 2.

Práctica 1. Simulación de circuitos con puertas lógicas

Crocodile Clips 3.5 y Yenka disponen de unos dispositivos llamados **entradas lógicas** y otros llamados **salidas lógicas** en la opción del menú de puertas lógicas.

Las entradas lógicas se conectan a las entradas de las puertas lógicas, y las salidas lógicas, a la salida de la última puerta.

Las entradas lógicas se activan pulsando sobre ellas y suministran un nivel alto de tensión; si no se pulsan, suministran un nivel de tensión bajo (figura 13).

Las salidas lógicas se activan (color rojo) si les llega un nivel alto de tensión; o están desactivadas (sin color) si les llega un nivel bajo de tensión (figura 13).

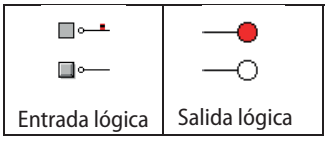


Fig. 13

La siguiente actividad está pensada para que practiques con la aplicación **Crocodile Clips 3.5**. También puedes descargarte la versión **Yenka** (*www.yenka.com*) con cualquiera de las tres licencias posibles: escolar (de pago), de prueba (15 días) o de uso en casa (gratuita).

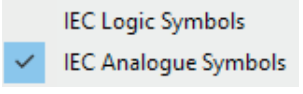
En un coche de fórmula 1 se enciende un led (*L*) de alarma cuando se da al menos una de las siguientes circunstancias: poco combustible (*C*), alta temperatura en frenos (*F*) o rotura del alerón (*A*). Cuando ocurre al menos una de estas circunstancias, el piloto puede activar un pulsador (*P*) para apagar el led. Éste no se apagará si el sensor que ha activado la alarma es el de rotura del alerón.

Lo que vamos a hacer en primer lugar es elaborar la tabla de verdad. Después simplificaremos por el método de Karnaugh y obtendremos el esquema del circuito con puertas lógicas.

1. Abre un procesador de texto y crea un documento nuevo en blanco.
2. Copia la tabla de verdad de la figura 14 y complétala con los valores que faltan.
3. Dibuja el diagrama de Karnaugh y comprueba que la función simplificada es la siguiente:

$$L = A + \overline{P}C + \overline{P}F$$

4. Explica cómo se realiza la simplificación.
5. Abre la aplicación Crocodile Clips.
6. Los elementos necesarios para montar el circuito son: entradas, salidas lógicas y puertas lógicas. Añade las puertas NOR, AND y OR que necesites. Puedes intercambiar la simbología haciendo clic en **View**.

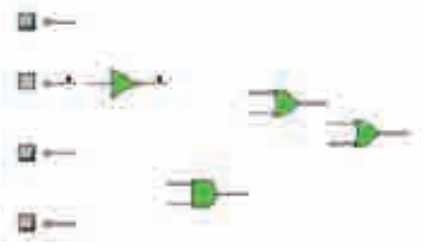


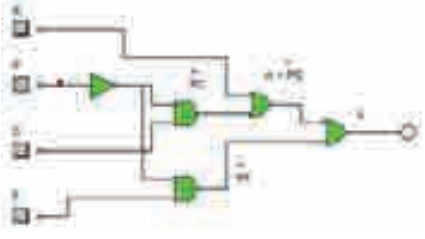
7. Une todos los elementos y añádeles las etiquetas con **Add** / **Text**.
8. Haz una captura y cópiala en el documento de texto.
9. Comprueba cómo cambia la salida *L* si vamos pulsando sobre cada una de las entradas. Comprueba que funciona correctamente en cada estado comparándolo con la tabla de verdad.
10. Guarda el documento de texto como **UD04_P1_nombre-apellido**.

P	C	F	A	L
0	0	0	0	0
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	1
0	1	0	1	
0	1	1	0	1
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Fig. 14

PC \ FA	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	0	1	1	0
10	0	1	1	0





Práctica 1. Montaje de un circuito de control con puertas lógicas

Vamos a diseñar un circuito de control de la luz (*L*) de una vivienda que está controlada por tres sensores (*a*, *b* y *c*) de forma que se encienda la luz en los siguientes casos:

a. Detector de oscuridad $a = 1$ cuando es de noche
b. Despertador activado $b = 1$ cuando está sonando
c. Persianas $c = 1$ cuando están bajadas
d. Detector de movimiento $d = 1$ cuando detecta movimiento

La luz de la vivienda se encenderá cuando sea de noche y se detecte movimiento (*a* y *d*) o bien cuando suene el despertador, estén las persianas bajadas y se detecte movimiento (*b*, *c* y *d*).

1. Creamos la tabla de verdad correspondiente. Escribe en tu cuaderno la declaración de variables y la tabla de verdad completa con todos los estados.
2. Simplifica la función mediante el método de Karnaugh. Compara tu mapa con el de la figura 14.
3. Comprueba que la función obtenida es la siguiente:

$$L = ad + bcd$$

4. Realiza el diagrama lógico (figura 15) y luego dibuja el esquema eléctrico (figura 16). En los circuitos digitales, las entradas de las puertas lógicas no deben dejarse al aire. Hemos de conectar las entradas a interruptores y la salida a una luz.
5. Ve montando los componentes en una placa protoboard. Lo primero será alimentar el chip. Para ello, conecta el polo positivo de la fuente a la patilla 14 y el negativo al 7.
6. Comprueba que el funcionamiento de cada una de las partes intermedias y el de la salida final se corresponden con lo indicado en la tabla de verdad.

ab \ cd	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	0	1	1	0
10	0	1	1	0

Fig. 14

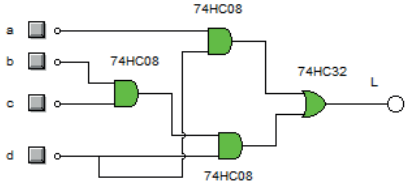


Fig. 15

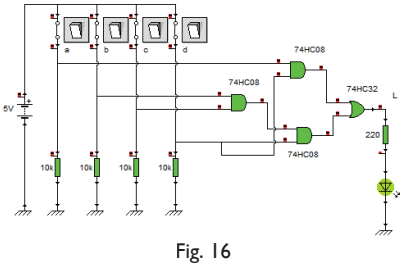
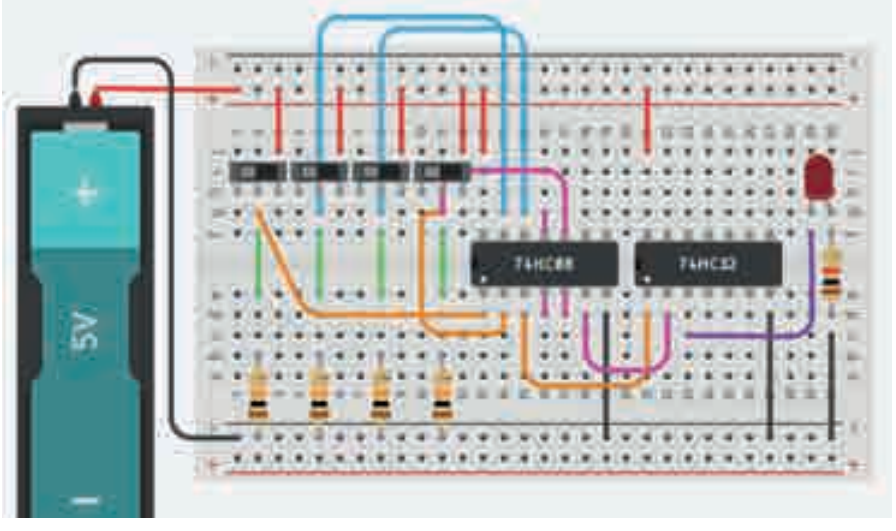
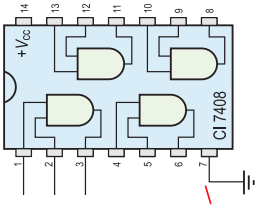


Fig. 16

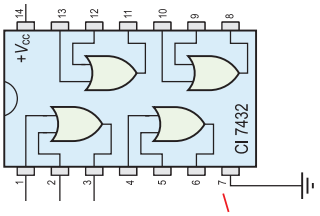


Patilla 14: conectarla al positivo de la fuente de alimentación



Patilla 7: conectarla al negativo de la fuente de alimentación

Patilla 14: conectarla al positivo de la fuente de alimentación



Patilla 7: conectarla al negativo de la fuente de alimentación

Ejercicios resueltos

1. ✦ Haz las siguientes conversiones:

a) $135B_{16}$ a decimal b) $247D_{16}$ a binario c) 49522_{10} a hexadecimal

d) $0101\ 1011\ 1111\ 0111_2$ a hexadecimal e) -78_{10} a binario

Solución

a) $135B_{16} = 1 \cdot 16^3 + 3 \cdot 16^2 + 5 \cdot 16^1 + B \cdot 16^0 = 1 \cdot 4096 + 3 \cdot 256 + 5 \cdot 16 + 11 \cdot 1 = 4096 + 768 + 80 + 11 = 4955_{10}$

b) $247D_{16}$ a binario

2	4	7	D
2	4	7	13
0010	0100	0111	1101

c) 49522_{10} a hexadecimal

49522 | 16
Resto = 2 — 3095 | 16
Resto = 7 — 193 | 16
Resto = 1 — 12 → C

$49522_{10} = C172_{16}$

d) $0101\ 1011\ 1111\ 0111_2$ a hexadecimal

5	11	15	7
5	B	F	7

$0101\ 1011\ 1111\ 0111_2 = 5BF7_{16}$

e) -78_{10} a binario ⇒ Pasamos el número +78 a binario y le añadimos un 0 delante para tener 8 bits.

78 | 2
0 — 39 | 2
1 — 19 | 2
1 — 9 | 2
1 — 4 | 2
0 — 2 | 2
0 — 1

$+78_{10} = 1001110_2$

Para transformar el número binario positivo a binario negativo se usa el complemento a dos (C2). Empezamos a leer el número por la derecha y buscamos el primer 1. Lo mantenemos como 1 (marcado en rojo). Continuamos leyendo los bits hacia la izquierda pero a partir de ahí cambiamos los 1 que encontremos por 0 y los 0 por 1 (marcados en azul).

0100 1110

1011 0010

$-78_{10} = 10110010_2$

2. ✦ Dado el siguiente circuito:

a) Obtén las expresiones de conmutación en función de *a*, *b*, *c* y *d* de las señales lógicas X_1, X_2, X_3, X_4 y *Z* mostradas en la figura.

b) Simplifica la función *Z* por el método de Karnaugh.

Solución

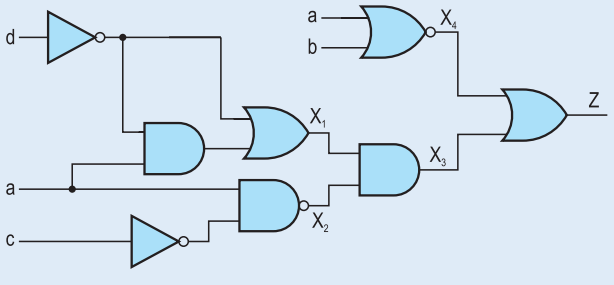
a) $X_1 = \overline{d} + a\overline{d} = \overline{d}(1 + a) = \overline{d} \cdot 1 = \overline{d}$

$X_2 = \overline{a\overline{c}} = \overline{a} + \overline{\overline{c}} = \overline{a} + c$

$X_3 = X_1 \cdot X_2 = \overline{d}(\overline{a} + c) = \overline{a}\overline{d} + c\overline{d}$

$X_4 = \overline{a + b} = \overline{a} \cdot \overline{b}$

$Z = X_3 + X_4 = \overline{a}\overline{d} + c\overline{d} + \overline{a}\overline{b}$



b)

ab \ cd	00	01	11	10
00	1	1	1	1
01	1	0	0	1
11	0	0	0	1
10	0	0	0	1

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

a	b	c	d	Z
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0

3. ✦✦ El encendido automático de las luces de un vehículo está formado por un sensor de luminosidad (*S*), un interruptor para seleccionar el encendido automático (*A*) y otro para el encendido normal (*E*). Las luces (*L*) se encienden si *S* está a 0 y *A* está a 1.

a) Obtén la tabla de verdad y la simplificación por el método de Karnaugh.

b) Obtén el circuito lógico de la función simplificada utilizando solamente puertas NAND.

Variables de entrada
sensor de luminosidad (*S*)
encendido automático (*A*)
encendido normal (*E*)

Variables de salida
luces (*L*)

SA \ *E*	0	1
00	0	1
01	1	1
11	0	1
10	0	1

Solución

a) Completamos la tabla de verdad en función de los estados en los que se activa la luz.

S	A	E	L
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

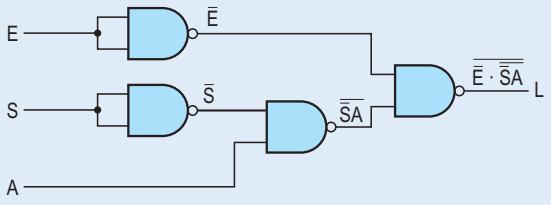
b) Dibujamos el diagrama de Karnaugh para realizar la simplificación:

$$L = E + \overline{S}A$$

Para construirlo con puertas NAND, aplicamos el teorema de Morgan:

$$L = \overline{\overline{E + \overline{S}A}} = \overline{\overline{E} \cdot \overline{\overline{S}A}}$$

Dibujamos el circuito:



4. ✦ Haz las siguientes conversiones:

a) Representa en complemento a 2 y usando 8 bits el número –67.

b) Representa en complemento a 2 y usando 8 bits el número +35.

c) Obtén el valor decimal de 1111 1000 sabiendo que está representado en complemento a 2 usando 8 bits.

Solución

a) C2 con 8 bits → –67

1. Escribimos el número en positivo: 67 = 0100 0011
2. Cambiamos 0 → 1, 1 → 0: 1011 1100
3. Sumamos +1: + 1

$1011\ 1101_{C2}$

Otra forma de hacerlo directamente es aplicando la siguiente regla práctica:

- De derecha a izquierda buscamos el primer 1 y lo mantenemos, y a partir de ahí los demás números los cambiamos (1 → 0 y 0 → 1):

67 = 0100 0011 ← El primer 1 se mantiene
$1011\ 1101_{C2}$

b) Los números positivos se representan igual que el binario natural. Por tanto, será 0010 0011.

c) 1111 1000 → en C2 con 8 bits
El primer número indica el signo { 1 negativo / 0 positivo

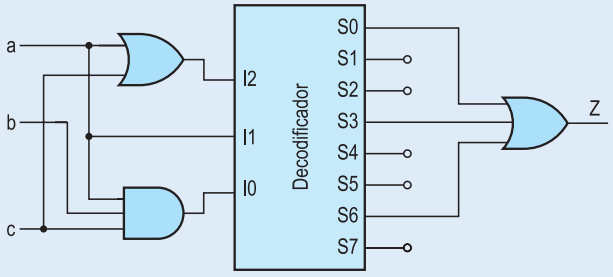
Como el número es negativo, de derecha a izquierda buscamos el primer 1. Lo mantenemos y cambiamos a partir de ahí los 1 → 0 y 0 → 1:

1111 1000
0000 1000 → Realizamos la conversión de binario a decimal
8 → Como el número original es negativo, será –8

5. ✦ Dado el siguiente circuito:

a) Obtén la expresión de conmutación en forma de suma de minterms de la señal lógica *Z*, como función de *a*, *b* y *c*.

b) Simplifica dicha función por el método de Karnaugh.



Solución

a) La expresión de conmutación en forma de suma de productos será:

$$Z = S_0 + S_3 + S_6$$

Para obtenerla en función de *a*, *b* y *c*, tenemos que sacar las expresiones de S_0, S_3 y S_6 de la tabla de verdad.

I_2	I_1	I_0	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	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
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Del gráfico deducimos:

$$I_2 = a + c$$
$$I_1 = a$$
$$I_0 = abc$$

De la tabla de verdad deducimos:

$$S_0 = \overline{I}_2 \overline{I}_1 \overline{I}_0$$
$$S_3 = \overline{I}_2 I_1 I_0$$
$$S_6 = I_2 I_1 \overline{I}_0$$

Sustituyendo cada uno de los valores I_0, I_1 e I_2 y simplificando obtenemos:

$$Z = S_0 + S_3 + S_6 = \overline{I}_2 \overline{I}_1 \overline{I}_0 + \overline{I}_2 I_1 I_0 + I_2 I_1 \overline{I}_0 = \overline{(a+c)}\,\overline{a}\,\overline{abc} + \overline{(a+c)}\,a\,abc + (a+c)\,a\,\overline{abc} =$$
$$= \overline{a}\,\overline{c}\,\overline{a}\,(\overline{a} + \overline{b} + \overline{c}) + (\overline{a}\,\overline{c})\,abc + (a+c)\,a\,(\overline{a} + \overline{b} + \overline{c}) =$$
$$= \overline{a}\,\overline{c} + \overline{a}\,\overline{b}\,\overline{c} + \overline{a}\,\overline{c} + 0 + a\,\overline{b} + a\,\overline{b}c + a\,\overline{c} = \overline{a}\,\overline{c} + \overline{a}\,\overline{bc} + a\,\overline{b} + a\,\overline{b}c + a\,\overline{c}$$

b) Para simplificar por el método de Karnaugh, partiendo de la tabla de verdad de la expresión anterior:

$$Z = \overline{a}\,\overline{c} + \overline{a}\,\overline{bc} + a\,\overline{b} + a\,\overline{b}c + a\,\overline{c}$$

a	b	c	Z
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Dibujamos el mapa de Karnaugh y simplificamos:

ab \ *c*	0	1
00	1	0
01	1	0
11	1	0
10	1	1

$$Z = \overline{c} + a\overline{b}$$

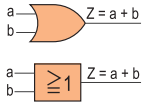
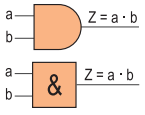
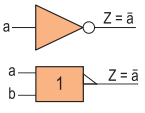
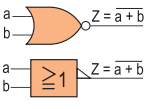
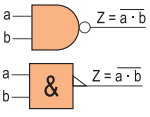
Repasa

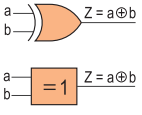
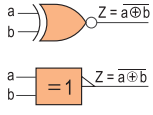
- En los **circuitos combinacionales**, la salida únicamente depende de la combinación de las entradas.
- En los **circuitos secuenciales**, la salida no se puede conocer siempre conociendo la entrada, sino que es necesario saber la historia del circuito, es decir, la salida depende de la combinación de entradas y del estado anterior del circuito.
- Los **sistemas de numeración** son los distintos conjuntos de símbolos capaces de representar la información numérica:

Base	Número de dígitos
Base 10 o decimal	Diez dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)
Base 2 o binario	Dos dígitos (0, 1)
Base 8 u octal	Ocho dígitos (0, 1, 2, 3, 4, 5, 6, 7)
Base 16 o hexadecimal	Dieciséis dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F)

- El **álgebra de Boole** es una estructura matemática que nos permite representar un sistema electrónico digital matemáticamente mediante una función lógica.
- Los dos **teoremas de Morgan** nos permiten intercambiar las funciones de suma y producto:

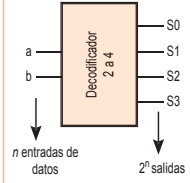
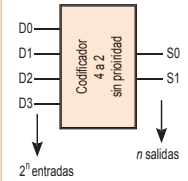
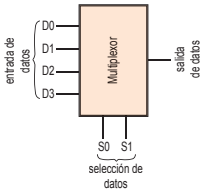
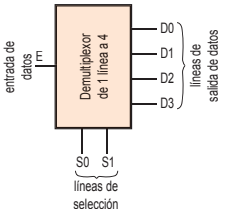
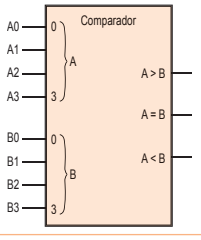
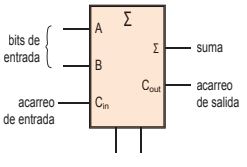
$\overline{(a+b)} = \overline{a} \cdot \overline{b}$	$\overline{(a \cdot b)} = \overline{a} + \overline{b}$

OR	AND	NOT	NOR	NAND
				
$Z = a + b$	$Z = a \cdot b$	$Z = \overline{a}$	$Z = \overline{a + b}$	$Z = \overline{a \cdot b}$

XOR	XNOR
	
$Z = a \oplus b$	$Z = \overline{a \oplus b}$

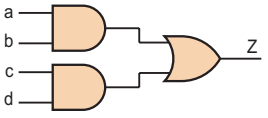
- El **método de Karnaugh** es un método de simplificación de funciones lógicas basado en la creación de una tabla, llamada *mapa de Karnaugh*, en la que se ordenan los términos de la función para formar agrupaciones de términos adyacentes, que permitirán expresar la función lógica de una forma más sencilla.
- **Resolución de problemas** y **diseño de circuitos**:

- **Circuitos lógicos combinacionales:**

Decodificador	Codificador	Multiplexor	Demultiplexor	Comparador	Sumador
					

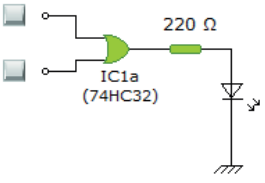
ACTIVIDADES DE REFUERZO

1. ✦ Explica qué diferencia hay entre una señal analógica y una señal digital.
2. ✦ Explica qué diferencia hay entre un circuito combinacional y un circuito secuencial.
3. ✦ Enuncia los teoremas de Morgan y describe el uso que se les da en esta unidad.
4. ✦ Dibuja con la simbología tradicional y con símbolos rectangulares las siguientes funciones lógicas: XOR, XNOR, AND, NOR, OR. Escribe debajo de cada una de ellas la función lógica de la operación que realizan.
5. ✦✦ ¿Qué son los niveles lógicos en las puertas digitales? ¿Qué utilidad tienen? ¿Qué pasa si tenemos un valor intermedio?
6. ✦✦✦ Queremos automatizar el proceso de control de temperatura de una habitación que permita regular la potencia en función de tres temperaturas y, además, el encendido en modo aire acondicionado. Describe todo el proceso que seguirías desde el planteamiento del problema hasta la construcción física con puertas lógicas.
7. ✦ Dibuja el símbolo de un decodificador y el de un codificador y explica sus aplicaciones.
8. ✦ Dibuja el símbolo de un multiplexor y el de un demultiplexor y explica sus aplicaciones.
9. ✦✦ Explica para qué se utiliza el complemento a 2. Explica las dos formas de convertir un número en complemento a 2 de decimal a binario.
10. ✦✦✦ Dado este circuito:



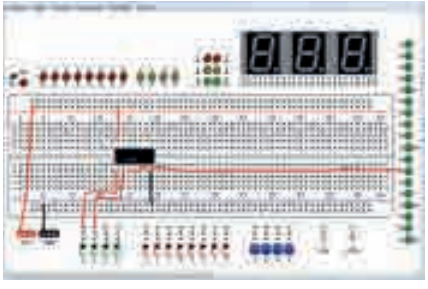
   a. Indica qué modelo de puertas lógicas utilizarías para construirlo.
   b. Dibuja la placa protoboard y las puertas físicas necesarias.
   c. Dibuja todas las conexiones necesarias para que se comporte como se muestra en la figura.

ACTIVIDADES PRÁCTICAS



Existen muchos simuladores de construcción de circuitos digitales. Uno de los más sencillos es **Simulador Digital 095**.

1. Abre en tu ordenador un buscador de Internet y encuentra el archivo llamado **SimuladorDigital_095.zip**.
2. Descárgalo, abre el archivo comprimido y ejecuta la aplicación.
3. Usando este programa, simula la construcción del circuito de funcionamiento de una puerta OR cuyo diagrama tienes al margen.
4. Tu circuito deberá ser similar al de la figura (no es necesario conectar una resistencia al led).



Además, puedes probar algunas de las apps para dispositivos móviles que permiten simular circuitos o realizar conversiones muy sencillas entre los distintos tipos de sistemas de numeración. Ejemplos de ellas son:

- Logic Simulator Pro
- Mini Karnaugh
- Logic Gates
- Conversor Binario

PROBLEMAS

1. ✦ Haz las siguientes conversiones:

a. $D4B0_{16}$ a decimal

b. 3053_8 a binario (un número en octal se representa con 3 bits en binario)

c. 39677_{10} a hexadecimal

d. $0001\ 1111\ 1101\ 0110_2$ a hexadecimal *(Solución: 54448; 0110 0010 1011; 9AFD; 1FD6)*

2. ✦✦ Representa en complemento a 2 y usando 8 bits los números –45 y +93. *(Solución: 93 = 01011101; –45 = 11010011)*

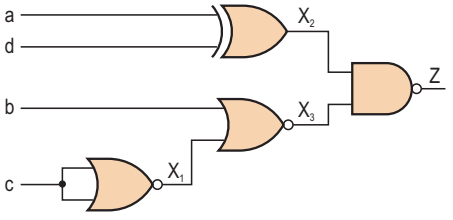
3. ✦✦ Obtén el valor decimal de 10110100 y de 01110001 sabiendo que están representados en complemento a 2 usando 8 bits. *(Solución: –76; 113)*

4. ✦✦ Dado el siguiente circuito:

a. Obtén las expresiones de conmutación en función de *a*, *b*, *c* y *d* de las señales lógicas X_1, X_2, X_3 y *Z* mostradas en la figura.

b. Obtén la tabla de verdad de la función lógica *Z*(*a*,*b*,*c*,*d*) que realiza el circuito mostrado en la figura.

c. Simplifica por el método de Karnaugh y obtén las dos expresiones en forma de minterms y maxterms.



(Solución: $Z = \overline{c} + b + \overline{a}\overline{d} + ad$)

5. ✦✦ Para que se active el motor de arranque (*MA*) de un motor diésel se deben cumplir las siguientes condiciones: que se presione el pulsador de arranque (*P*), que el sensor que detecta exceso de temperatura del motor diésel (*T*) esté a 0 y que la llave de contacto (*LC*) esté a 1. En el caso de que la temperatura sea excesiva ($T = 1$) el motor de arranque se podrá activar mediante un pulsador auxiliar (*PA*) independientemente del estado de las demás variables.

a. Obtén la tabla de verdad y la función lógica *MA* simplificada por el método de Karnaugh.

b. Obtén el circuito lógico mediante puertas.

(Solución: $MA = PA \cdot T + P \cdot \overline{T} \cdot LC$)

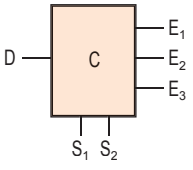
6. ✦✦ Se quiere diseñar un sistema con dos luces de alarma y tres sensores (entradas digitales). Llamaremos L_1 y L_2 a las luces de alarma y *A*, *B* y *C* a los sensores digitales. El sistema deberá funcionar así: la alarma L_1 se dispara si recibe señal del sensor *B* exclusivamente; la alarma L_2 se dispara si recibe señal del sensor *A* exclusivamente; las dos alarmas se disparan si reciben señal de al menos dos sensores cualesquiera.

a. Obtén la tabla de verdad y las funciones lógicas.

b. Obtén las funciones lógicas simplificadas y sus circuitos con puertas lógicas.

(Solución: $L_1 = B + AC$; $L_2 = A + BC$)

7. ✦✦✦ Una línea de datos digitales (*D*) puede ser enviada a tres equipos diferentes (E_1, E_2 y E_3) mediante un circuito (*C*) y dos señales de control (S_1 y S_2). La selección se realiza de forma que el número binario introducido en S_1 y S_2 se corresponde con el número del equipo conectado a *D*. Las señales de entrada de los equipos no conectados se ponen a 1.



a. Elabora la tabla de verdad para las variables de salida E_1, E_2 y E_3.

b. Simplifica por el método de Karnaugh e implementa el circuito con puertas lógicas.

(Solución: $E_1 = D + S_2 + \overline{S}_1$; $E_2 = D + \overline{S}_2 + S_1$; $E_3 = D + \overline{S}_2 + \overline{S}_1$)

ACTIVIDADES DE AMPLIACIÓN

1. ✦✦✦ Busca información y realiza una presentación o un informe sobre uno de los siguientes temas:

- Intensidad de entrada y de salida de las puertas lógicas. ¿Qué es? ¿Qué importancia tiene? ¿Problemas?
- Rebotes en las puertas lógicas. Qué son y cómo evitarlos
- Capacidad de entrada y de salida de las puertas lógicas