- **Sistema**
  - Estructura y comportamiento
  - Señal analógica y señal digital
  - Señal binaria
- **Sistemas de numeración**
- **Representación de números enteros**
  - Signo-magnitud
  - Complemento a dos
- **Codificación**
  - Códigos numéricos
  - Códigos alfanuméricos

- **Álgebra de Boole**
  - Definiciones y axiomas
  - Propiedades

- **Variables y funciones booleanas**
  - Definiciones
  - Propiedades
  - Formas de representación
  - Funciones booleanas y circuitos combiancionales

- **Puertas lógicas**
  - Puertas lógicas fundamentales
  - Puertas lógicas derivadas

Sistema

- **Estructura y Comportamiento**

Un sistema físico establece una relación causal entre excitaciones (señales de entrada) y respuestas a tales excitaciones (señales de salida)

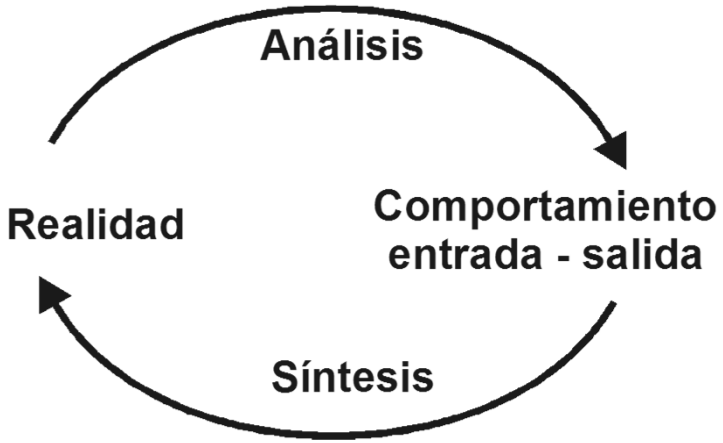
Alfabeto de entrada. Conjunto de valores posibles de las señales de entrada

$$\sum = \{\sigma_1, \sigma_2, \ldots\ldots \sigma_n\}$$

Alfabeto de salida. Conjunto de valores posibles de las señales de salida

$$\Omega = \{\omega_1, \omega_2, \ldots\ldots \omega_m\}$$

Comportamiento Entrada/Salida.



Conjunto de relaciones que el sistema establece entre las señales de entrada y las de salida

Puede ser punto de partida en el diseño del sistema, ó punto final en el análisis

Para facilitar la resolución de sistemas digitales de gran complejidad, podemos establecer tres niveles jerárquicos:

a) Nivel de arquitectura

También llamado nivel de sistema. Es el nivel más alto, y se caracteriza por la administración global del sistema. Permite descomponer un sistema complejo en varios subsistemas menos complejos

b) Nivel lógico

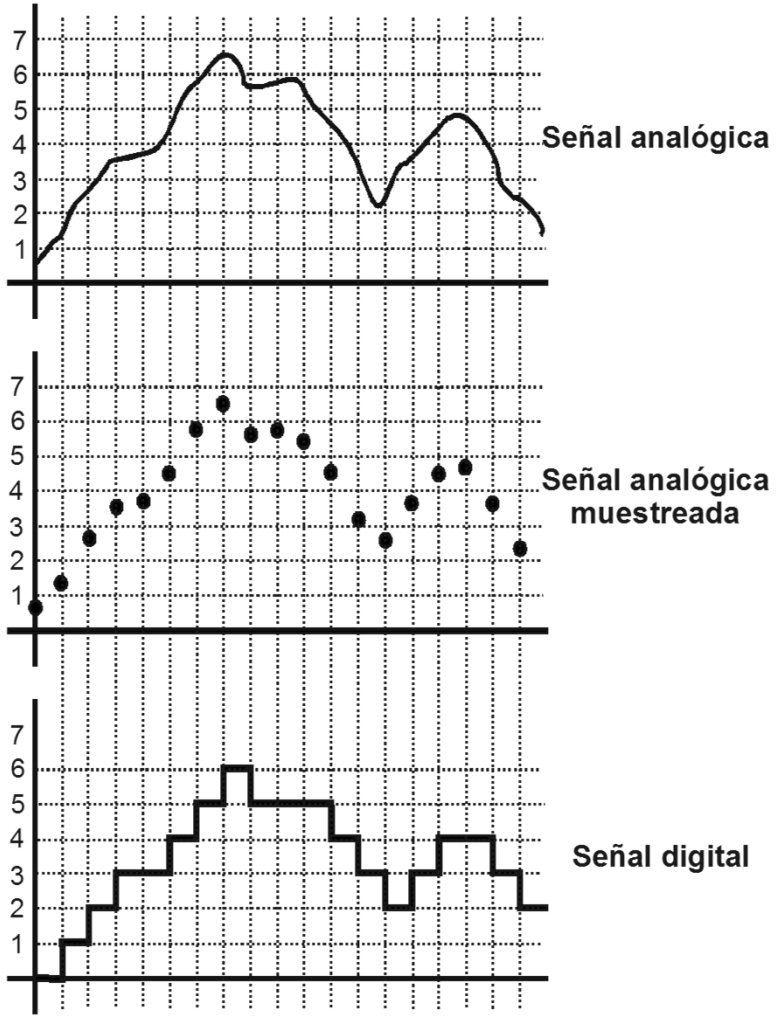
Nivel intermedio. Se preocupa por detalles del sistema, proponiendo una solución desde consideraciones lógicas, sin tener en cuenta el problema de la fabricación física de los dispositivos lógicos que utiliza

c) Nivel físico

Nivel inferior. Se preocupa de los detalles necesarios para fabricar o montar el sistema.

La asignatura de Fundamentos Electrónicos trata del estudio de los circuitos digitales en el **Nivel lógico** y en el **Nivel físico**.

- **Señal analógica y señal digital**



a) Señal analógica.

Consiste en representar la información de forma que admita cualquier valor dentro del rango definido para la señal, en cualquier instante de tiempo.

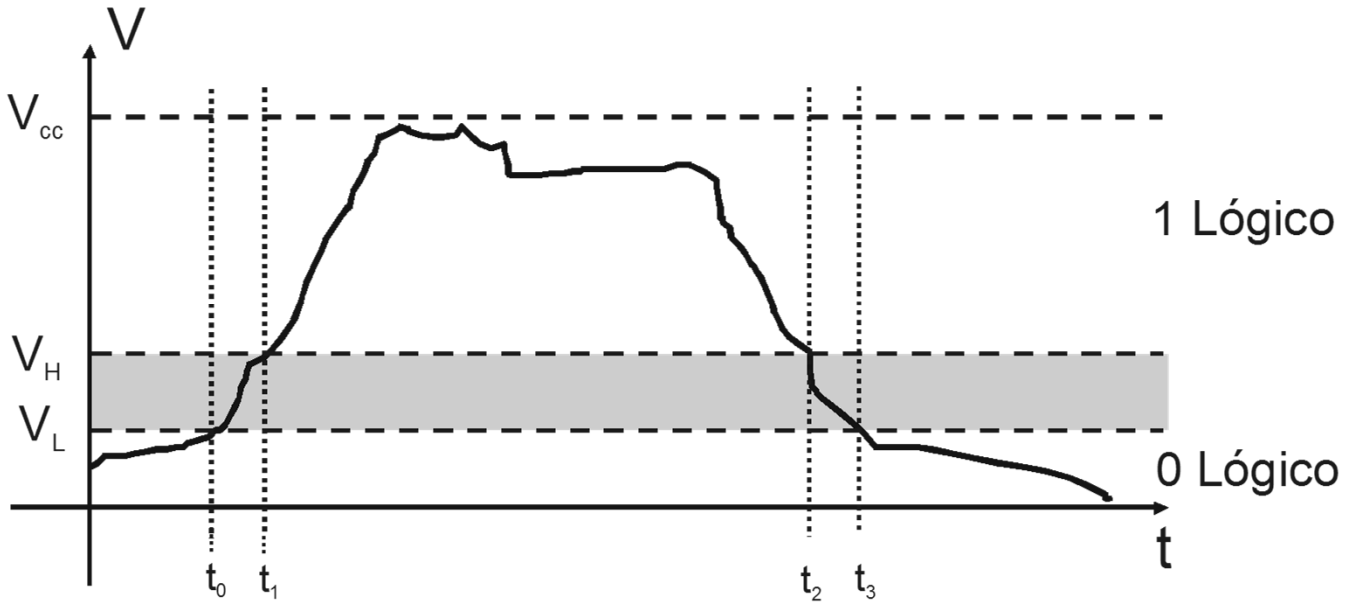
b) Señal analógica muestreada.

Consiste en representar la información de forma que admita cualquier valor dentro del rango definido para la señal, en unos instantes de tiempo determinados.

c) Señal digital.

Consiste en representar la información de forma que admita un conjunto de valores discretos, en unos instantes de tiempo determinados.

- **Señal binaria**



Señal binaria

Consiste en representar la información mediante dos únicos valores que generalmente se asimilan al **0** y al **1**

Circuito digital

Se caracteriza por utilizar señales binarias

Diseño lógico

Consiste en la construcción de circuitos digitales utilizando como elementos base, las puertas lógicas

Puerta lógica

Circuito electrónico, de comportamiento específico, que permite ejecutar operaciones lógicas específicas para unas señales de entrada, produciendo como resultado unas señales de salida.

Las puertas lógicas están disponibles comercialmente empaquetadas en circuitos integrados (CI)

La teoría de diseño lógico de circuitos digitales se conoce como teoría de conmutación y su base teórico-matemática es el álgebra de conmutación o álgebra de Boole

Los sistemas digitales se dividen en sistemas combinacionales y sistemas secuenciales

Sistema combinacional

La salida en un instante de tiempo t sólo depende del valor de la entrada en dicho instante t

$$Z(t) = G(x(t))$$

El sistema NO tiene memoria, por lo que realmente

$$Z = G(x)$$

Sistema secuencial

La salida en un instante de tiempo t depende del valor de la entrada en dicho instante t y de las entradas previas

$$Z(t) = G_1(x(t),s(t))$$
$$s(t+1) = G_2(x(t),s(t))$$

Donde:

G_1	Función de Salida.
G_2	Función de Transición de Estados.
s(t)	Estado del sistema en el instante t

Sistemas de numeración

- **Representación numérica en base B**

  Es una codificación de $Z_N = \{0, 1, ... , N-1\}$, en el conjunto de símbolos $B=\{x_0, x_1, x_2, ... , x_{B-1}\}$ utilizando un alfabeto $A = \{s_0, s_1, s_2, ... , s_K\}$, donde cualquier natural X tiene asociada una palabra en A

- **Valor numérico**

  Dado el elemento X representado por la palabra $[x_{n-1}, x_{n-2} , ... , x_1 , x_0]$ en una base B, su valor numérico viene dado por:

$$[x_{n-1},x_{n-2}, \dots ,x_1,x_0] = x_{n-1}B^{n-1}+x_{n-2}B^{n-2}+ \dots +x_1B^1+x_0B^0$$

- **Propiedades de los sistemas de numeración**

a) La representación de cada X es única

b) A cada elemento de ***B*** se denomina dígito

c) El número de elementos de ***B*** se denomina base del sistema |B|

d) El dígito de mayor valor es |B|-1

e) A cada elemento del alfabeto se le denomina ***palabra***

f) El número de palabras de un alfabeto viene dado por $|B|^n$ donde:

|B|	número de dígitos de la base B
n	longitud de palabra de los elementos del alfabeto

- **Sistema Binario**

B = {0,1} |B| = 2

Cada uno de los dígitos recibe el nombre de ***bit***

El bit más a la izquierda del número, se denomina ***bit más significativo*** y el bit más a la derecha del número, se denomina ***bit menos significativo***

Ejp. 0100011101110001

↑ ↑

Bit más significativo Bit menos significativo

- **Sistema Octal**

B = {0,1,2,3,4,5,6,7} |B| = 8

- **Sistema Hexadecimal**

B = {0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F} |B| = 16

Decimal	Binario	Octal	Hexadecimal
0	0 000	00	0
1	0 001	01	1
2	0 010	02	2
3	0 011	03	3
4	0 100	04	4
5	0 101	05	5
6	0 110	06	6
7	0 111	07	7
8	1 000	10	8
9	1 001	11	9
10	1 010	12	A
11	1 011	13	B
12	1 100	14	C
13	1 101	15	D
14	1 110	16	E
15	1 111	17	F

Representación de números enteros

- **Representación signo-magnitud**

Para el caso binario, dado un tamaño de palabra de n bits, podemos representar el siguiente rango de números sin signo

$$0 \leq x \leq 2^n - 1$$

La representación signo-magnitud consiste en añadir un bit adicional (bit de signo), que tendrá el valor 0 para los números positivos y el valor 1 para los números negativos

Para un tamaño de palabra de n bits, podemos representar el siguiente rango de números signados en formato signo-magnitud

$$-(2^{n-1} - 1) \leq x \leq (2^{n-1} - 1)$$

n	Bin.	Dec.	Rango
1	0 1	0 1	$0 \leq x \leq 2^1-1$ $0 \leq x \leq 1$
2	00 01 10 11	0 1 2 3	$0 \leq x \leq 2^2-1$ $0 \leq x \leq 3$
3	000 001 010 011 100 101 110 111	0 1 2 3 4 5 6 7	$0 \leq x \leq 2^3-1$ $0 \leq x \leq 7$

Sin signo

n	Bin.	Dec.	Rango
2	00 01 10 11	0 1 0 -1	$-(2^{2-1}-1) \leq x \leq (2^{2-1}-1)$ $-1 \leq x \leq 1$
3	000 001 010 011 100 101 110 111	0 1 2 3 0 -1 -2 -3	$-(2^{3-1}-1) \leq x \leq (2^{3-1}-1)$ $-3 \leq x \leq 3$

Signo-magnitud

- **Representación complemento-a-dos**

La representación complemento-a-dos de un número con signo consiste en:

a) Los números con el bit más significativo con valor 0, serán considerados como positivos

b) Los números con el bit más significativo con valor 1, serán considerados como negativos

c) Para un tamaño de palabra de n bits, podemos representar el siguiente rango de números signados en formato complemento-a-dos:

$$-(2^{n-1}) \leq x \leq (2^{n-1} - 1)$$

Dado un número, el cálculo de su opuesto, se efectúa en 2 etapas (método 1):

1) Se obtiene el complementario del número
2) Se suma 1 al resultado obtenido en el paso 1

Dado un número, el cálculo de su opuesto, se efectúa en 2 etapas (método 2):

1) Se busca el primer bit con valor 1 partiendo del bit menos significativo (bit diferenciador)
2) Se complementan todos los bits a la izquierda del bit diferenciador

Operandos		Resultado	Acarreo
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Suma binaria

Operando	Resultado
0	1
1	0

Operación complementación

n	Bin.	Dec.	Rango
2	00	0	$-(2^{2-1}) \leq x \leq (2^{2-1}-1)$
	01	1	$-2 \leq x \leq 1$
	10	-2	
	11	-1	
3	000	0	$-(2^{3-1}) \leq x \leq (2^{3-1}-1)$
	001	1	$-4 \leq x \leq 3$
	010	2	
	011	3	
	100	-4	
	101	-3	
	110	-2	
	111	-1	
4	0000	0	$-(2^{4-1}) \leq x \leq (2^{4-1}-1)$
	0001	1	$-8 \leq x \leq 7$
	0010	2	
	0011	3	
	0100	4	
	0101	5	
	0110	6	
	0111	7	
	1000	-8	
	1001	-7	
	1010	-6	
	1011	-5	
	1100	-4	
	1101	-3	
	1110	-2	
	1111	-1	

Codificación

- **Códigos numéricos**

Código BCD

Dado un número sin signo en base 10 $N_{10}= d_{n-1}d_{n-2}...d_1d_0$ se convierte a BCD estándar representando en binario de 4 bits cada uno de sus dígitos d_i

d_{n-1}	.....	d_i	.....	d_0
$b_3^{n-1}b_2^{n-1}b_1^{n-1}b_0^{n-1}$	.....	$b_3^ib_2^ib_1^ib_0^i$	.....	$b_3^0b_2^0b_1^0b_0^0$

Código Exceso 3

Dado un número sin signo en base 10 $N_{10}= d_{n-1}d_{n-2}...d_1d_0$ se convierte a E3 estándar representando en binario de 4 bits cada uno de sus dígitos d_i más tres

Presenta como ventaja al código BCD que genera automáticamente el acarreo cuando se suman dos dígitos

Código Gray

Fórmula de generación recursiva:

$$G_{n+1} = \{0G_n, 1G_n^{ref}\} \quad \text{donde } G_1 = \{0,1\} \quad n \geq 1$$

Es un código reflejado, no posicional

Presenta la propiedad de que dos números consecutivos, representados en este código, sólo difieren en 1 bit

	BCD	E3	GRAY
0	0000 0000	0000 0011	0000
1	0000 0001	0000 0100	0001
2	0000 0010	0000 0101	0011
3	0000 0011	0000 0110	0010
4	0000 0100	0000 0111	0110
5	0000 0101	0000 1000	0111
6	0000 0110	0000 1001	0101
7	0000 0111	0000 1010	0100
8	0000 1000	0000 1011	1100
9	0000 1001	0000 1100	1101
10	0001 0000	0100 0011	1111
11	0001 0001	0100 0100	1110
12	0001 0010	0100 0101	1010
13	0001 0011	0100 0110	1011
14	0001 0100	0100 0111	1001
15	0001 0101	0100 1000	1000

- **Códigos de caracteres**

Son adecuados para almacenar información no numérica

Asignan una palabra binaria de longitud fija, para representar la menor unidad de información alfanumérica, esto es, para cada carácter gráfico

a) Código ASCII. (American Standard Code for Information Interchange).

Utiliza un tamaño de palabra binaria de 8 bits, siendo siempre el bit más significativo 0, por lo que puede representar 128 caracteres distintos

b) Código EBCDIC. Desarrollado por IBM.

Utiliza un tamaño de palabra binaria de 8 bits, por lo que puede representar 256 caracteres distintos

Álgebra de Boole

- **Definición de Álgebra de Boole**

Un conjunto es un álgebra de Boole se verifica:

a) Es un conjunto finito B con al menos dos elementos, N (elemento nulo), U (elemento universal), y tres operaciones: dos binarias y una unaria

$$(B, *, +, \bar{} \;)$$

$$N \in B$$

$$U \in B$$

b) Cumple los 6 axiomas de HUNTINGTON

- **Axiomas de HUNTINGTON**

1. Las operaciones *, + ,¯ , deben ser cerradas

$$\forall x,y \in B \begin{cases} x*y \in B \\ x+y \in B \\ \overline{x} \in B \end{cases}$$

2. Operaciones con N,U

$$x * N = N \qquad x + N = x$$
$$x * U = x \qquad x + U = U$$

3. Conmutatividad

$$x * y = y * x$$
$$x + y = y + x$$

4. Distributiva.

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

5. Complementatividad.

$$\forall x \in B, \exists\, \overline{x} \in B \,/ \begin{cases} x * \overline{x} = N \\ x + \overline{x} = U \end{cases}$$

6. Hay por lo menos dos elementos distintos en B.

- **Propiedades del Álgebra de Boole**

1. Propiedad de Idempotencia

$$x * x = x$$
$$x + x = x$$

2. Propiedad Asociativa

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

3. Propiedad de Absorción

$$x + (x * y) = x$$
$$x * (x + y) = x$$

4. Ley del consenso

$$x + (\overline{x} * y) = x + y$$
$$x * (\overline{x} + y) = x * y$$

5. Ley de involución

$$\overline{(\overline{x})} = x$$

Se puede demostrar que B={0,1}, N=0, U=1, junto con las operaciones * , + , $^{-}$, definidas por las siguientes tablas de verdad, forman un álgebra de boole.

+	0	1
0	0	1
1	1	1

OR

*	0	1
0	0	0
1	0	1

AND

$^{-}$	
0	1
1	0

NOT

Variables y funciones booleanas

- **Definiciones**

Definimos constante sobre B, a todo elemento de B

Definimos variable de B, todo símbolo x que representa a cualquier elemento de B

Definimos literal de B, a toda constante ó variable

Definimos función booleana a la aplicación:

$$f : B^n \rightarrow B$$

$$\text{donde: } B^n = B \times B \times .... \times B \,/\, \underline{x} = (x_1, x_2, ... , x_n) \in B^n$$

Definimos ***término producto*** a todo literal o producto de literales, en los que cada variable aparece como máximo una vez.

Definimos ***mintérmino*** o ***producto canónico*** al término producto de una función, que está formado por las ***n*** variables de dicha función, apareciendo éstas una sola vez de forma complementada o sin complementar.

Definimos ***forma normal disyuntiva*** de una función, a su representación algebraica, que consta de un sólo término producto o de la suma de varios de ellos.

Definimos ***término suma*** a todo literal o suma lógica de literales, en las que cada variable aparece como máximo una vez.

Definimos ***maxtérmino*** o ***suma canónica*** al término suma de una función, que está formado por las ***n*** variables de dicha función, apareciendo éstas una sola vez de forma complementada o sin complementar.

Definimos ***forma normal conjuntiva*** de una función, a su representación algebraica, que consta de un sólo término suma o del producto de varios de ellos.

Para ***n*** variables podemos formar ***2^n*** mintérminos y ***2^n*** maxtérminos.

Notación para mintérminos y maxtérminos:

$$\bar{x}_1 * \bar{x}_2 * .... * \bar{x}_{n-1} * \bar{x}_n = m_0$$
$$\bar{x}_1 * \bar{x}_2 * .... * \bar{x}_{n-1} * x_n = m_1$$
$$\bar{x}_1 * \bar{x}_2 * .... * x_{n-1} * \bar{x}_n = m_2$$
$$\bar{x}_1 * \bar{x}_2 * .... * x_{n-1} * x_n = m_3$$
$$.....$$
$$x_1 * x_2 * .... * x_{n-1} * x_n = m_{2^n-1}$$

$$x_1 + x_2 + ... + x_{n-1} + x_n = M_0$$
$$x_1 + x_2 + ... + x_{n-1} + \bar{x}_n = M_1$$
$$x_1 + x_2 + ... + \bar{x}_{n-1} + x_n = M_2$$
$$x_1 + x_2 + ... + \bar{x}_{n-1} + \bar{x}_n = M_3$$
$$.....$$
$$\bar{x}_1 + \bar{x}_2 + ... + \bar{x}_{n-1} + \bar{x}_n = M_{2^n-1}$$

Definimos vector asociado a un mintérmino ***m_i*** de ***n*** variables, al obtenido colocando 1 en las posiciones correspondientes a las variables NO complementadas, y colocando 0 en las posiciones correspondientes a las variables complementadas.

Ejp:

$$m_5 = \bar{x}_1 * x_2 * \bar{x}_3 * x_4 \rightarrow 0101$$

Definimos vector asociado a un maxtérmino ***M_i*** de ***n*** variables, al obtenido colocando 0 en las posiciones correspondientes a las variables NO complementadas, y colocando 1 en las posiciones correspondientes a las variables complementadas.

Ejp:

$$M_{10} = \bar{x}_1 + x_2 + \bar{x}_3 + x_4 \rightarrow 1010$$

- **Propiedades**

Dadas las funciones booleanas, f, g, las siguientes funciones f*g, f+g, $\overline{g}$, también son booleanas.

$$f*g(\underline{x}) = f(\underline{x}) * g(\underline{x}) \quad \forall \underline{x} = (x_1, x_2, \ldots, x_n) \in B^n$$

$$f+g(\underline{x}) = f(\underline{x}) + g(\underline{x}) \quad \forall \underline{x} = (x_1, x_2, \ldots, x_n) \in B^n$$

$$\overline{g}(\underline{x}) = \overline{g(\underline{x})} \quad \forall \underline{x} = (x_1, x_2, \ldots, x_n) \in B^n$$

Teorema de Dualidad.

Si a una identidad o teorema de conmutación se sustituyen {+,*,0,1} por {*,+,1,0} respectivamente, se obtiene otra identidad o teorema dual al original.

Teorema de DeMORGAN.

$$\overline{x_1 + x_2 + \ldots + x_n} = \overline{x}_1 * \overline{x}_2 * \ldots * \overline{x}_n$$

$$\overline{x_1 * x_2 * \ldots * x_n} = \overline{x}_1 + \overline{x}_2 + \ldots + \overline{x}_n$$

Teorema de SHANON (Teorema generalizado de DeMorgan).

$$\overline{f(x_1,x_2,\ldots,x_n,*,+,0,1)} = f(\overline{x}_1,\overline{x}_2,\ldots,\overline{x}_n,+,*,1,0)$$

Teorema de los mintérminos para n variables.

$$\sum_{i=0}^{2^n-1} m_i(x_1,x_2,\ldots,x_n) = 1$$

Teorema de los maxtérminos para n variables.

$$\prod_{i=0}^{2^n-1} M_i(x_1,x_2,\ldots,x_n) = 0$$

Teorema del desarrollo de Shanon, para mintérminos, para una función $f(\underline{x}) = f(x_1, x_2, ... , x_n)$, de ***n*** variables.

$$\begin{aligned} f(\underline{x}) &= \left[(\overline{x}_1 * \overline{x}_2 * ....* \overline{x}_n) * f(0,0, ... ,0)\right] + \left[(\overline{x}_1 * \overline{x}_2 * ....* x_n) * f(0,0, ... ,1)\right] + \\ &\quad + ... + \left[(x_1 * x_2 * ....* x_n) * f(1,1, ... ,1)\right] = \\ &= \left[m_0 * f(0,0, ... ,0)\right] + \left[m_1 * f(0,0, ... ,1)\right] + ... + \left[m_{2^n-1} * f(1,1, ... ,1)\right] \end{aligned}$$

Teorema del desarrollo de Shanon, para maxtérminos, para una función $f(\underline{x}) = f(x_1, x_2, ... , x_n)$, de ***n*** variables.

$$\begin{aligned} f(\underline{x}) &= \left[(x_1 + x_2 + ... + x_n) + f(0,0, ... ,0)\right] * \left[(x_1 + x_2 + ... + \overline{x}_n) + f(0,0, ... ,1)\right] * \\ &\quad * ..... * \left[(\overline{x}_1 + \overline{x}_2 + ... + \overline{x}_n) + f(1,1, ... ,1)\right] = \\ &= \left[M_0 + f(0,0, ... ,0)\right] * \left[M_1 + f(0,0, ... ,1)\right] * ..... * \left[M_{2^n-1} + f(1,1, ... ,1)\right] \end{aligned}$$

Como consecuencia del teorema del desarrollo de Shanon para mintérminos, tenemos que toda función f(x), admite una representación, que denominaremos ***forma canónica disyuntiva***, formada por la suma de los mintérminos cuyos vectores asociados V_K verifican que $f(V_K)=1$.

$$f(x) = \sum m_k \qquad \text{donde } f(V_k) = 1$$

Como consecuencia del teorema del desarrollo de Shanon para maxtérminos, tenemos que toda función f(x), admite una representación, que denominaremos ***forma canónica conjuntiva***, formada por el producto de los maxtérminos cuyos vectores asociados V_K verifican que $f(V_K)=0$.

$$f(x) = \prod M_k \qquad \text{donde } f(V_k) = 0$$

Teorema para obtener la función complementada de una dada en forma canónica disyuntiva.

> Dada una función $g(\underline{x})$, expresada en forma canónica disyuntiva, su función complementada $\overline{g}(\underline{x})$, estará dada por la suma de los mintérminos que no aparecen en $g(\underline{x})$.

Teorema para obtener la función complementada de una dada en forma canónica conjuntiva.

> Dada una función $g(\underline{x})$, expresada en forma canónica conjuntiva, su función complementada $\overline{g}(\underline{x})$, estará dada por el producto de los maxtérminos, que no aparecen en $g(\underline{x})$.

- **Formas de representación**

a) Esquemas de Circuitos

Un circuito electrónico, descrito a nivel de dispositivos, puede ser considerado como una forma de representar a la función booleana que implementa

b) Diagrama de puertas lógicas

Consiste en representar una función, mediante su implementación utilizando puertas lógicas

c) Expresión algebraica

Permite una representación más compacta, pero la información se presenta más oculta

d) Métodos de enumeración

d.1)Tabla de verdad.	Listado que de forma explícita representa el resultado de la función, para cada una y todas las combinaciones de las entradas
d.2) Vector de valores.	Vector formado por el resultado de la función, en el orden creciente de las combinaciones en las variables de entrada
d.3) Mintérminos.	La función en forma canónica disyuntiva
d.4) Maxtérminos.	La función en forma canónica conjuntiva

e) Mapas de Karnaugh

Es una representación gráfica de la tabla de verdad mediante una matriz bidimensional, donde cada posible combinación de los valores binarios de las variables de entrada, está representada por una celda ó casilla. Las entradas están ordenadas en código Gray, de forma que dos casillas adyacentes (horizontal ó verticalmente), sólo tienen distinto valor en una de las entradas

- **Mapas de Karnaugh de 3 variables y de 4 variables**

x_1,x_2 / x_3	00	01	11	10
0	0	2	6	4
1	1	3	7	5

x_1,x_2 / x_3,x_4	00	01	11	10
00	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

- **Mapa de Karnaugh de 5 variables**

x_4,x_5 \ x_2,x_3	00	01	11	10
00	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

$x_1 = 0$

x_4,x_5 \ x_2,x_3	00	01	11	10
00	16	20	28	24
01	17	21	29	25
11	19	23	31	27
10	18	22	30	26

$x_1 = 1$

- **Mapa de Karnaugh de 6 variables**

x_5,x_6 \ x_3,x_4	00	01	11	10
00	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

x_1x_2= 00

x_5,x_6 \ x_3,x_4	00	01	11	10
00	16	20	28	24
01	17	21	29	25
11	19	23	31	27
10	18	22	30	26

x_1x_2= 01

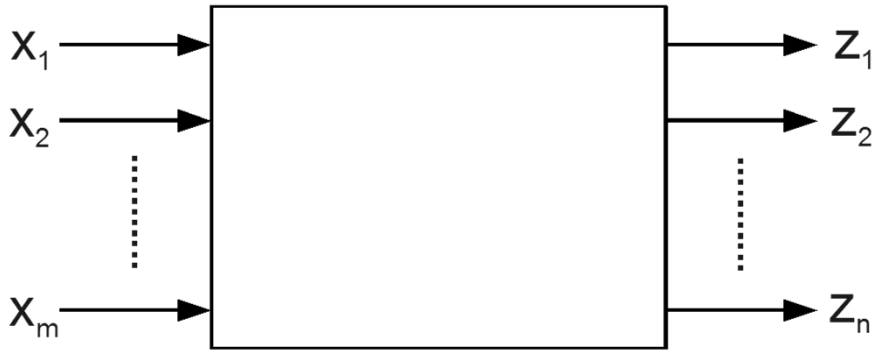
x_5,x_6 \ x_3,x_4	00	01	11	10
00	32	36	44	40
01	33	37	45	41
11	35	39	47	43
10	34	38	46	42

x_1x_2= 10

x_5,x_6 \ x_3,x_4	00	01	11	10
00	48	52	60	56
01	49	53	61	57
11	51	55	63	59
10	50	54	62	58

x_1x_2= 11

- **Funciones booleanas y circuitos combinacionales**

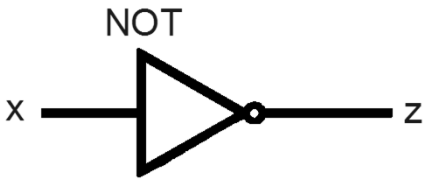


Para implementar un circuito digital combinacional de ***m*** entradas y ***n*** salidas, será necesario implementar ***n*** funciones booleanas cada una de ellas dependiente de ***m*** variables.

$$f_1(x_1 , x_2, ... , x_m)$$
$$f_2(x_1 , x_2, ... , x_m)$$
$$.....$$
$$f_n(x_1 , x_2, ... , x_m)$$

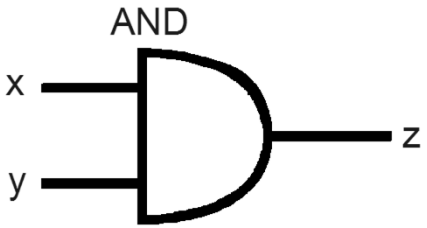
Puertas lógicas

- **Puertas lógicas fundamentales**



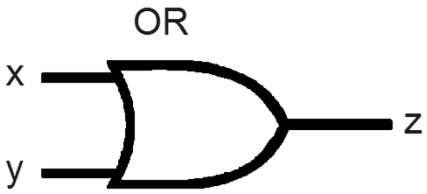
$z = \overline{x}$

NOT	
0	1
1	0



$z = x\,y$

AND	0	1
0	0	0
1	0	1

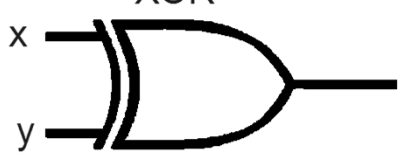


$z = x + y$

OR	0	1
0	0	1
1	1	1

- **Puertas lógicas derivadas**

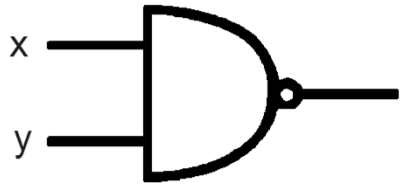
XOR



$z = x \oplus y = \bar{x}\, y + x\, \bar{y}$

XOR	0	1
0	0	1
1	1	0

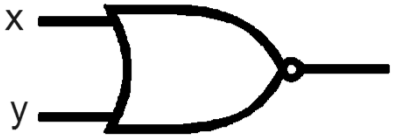
NAND



$z = \overline{x\, y}$

NAND	0	1
0	1	1
1	1	0

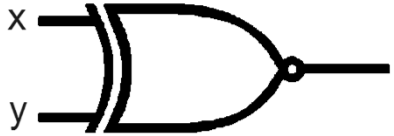
NOR



$z = \overline{x + y}$

NOR	0	1
0	1	0
1	0	0

XNOR



$z = \overline{x \oplus y}$

XNOR	0	1
0	1	0
1	0	1

Ejemplo de función booleana

Diagrama de circuitos

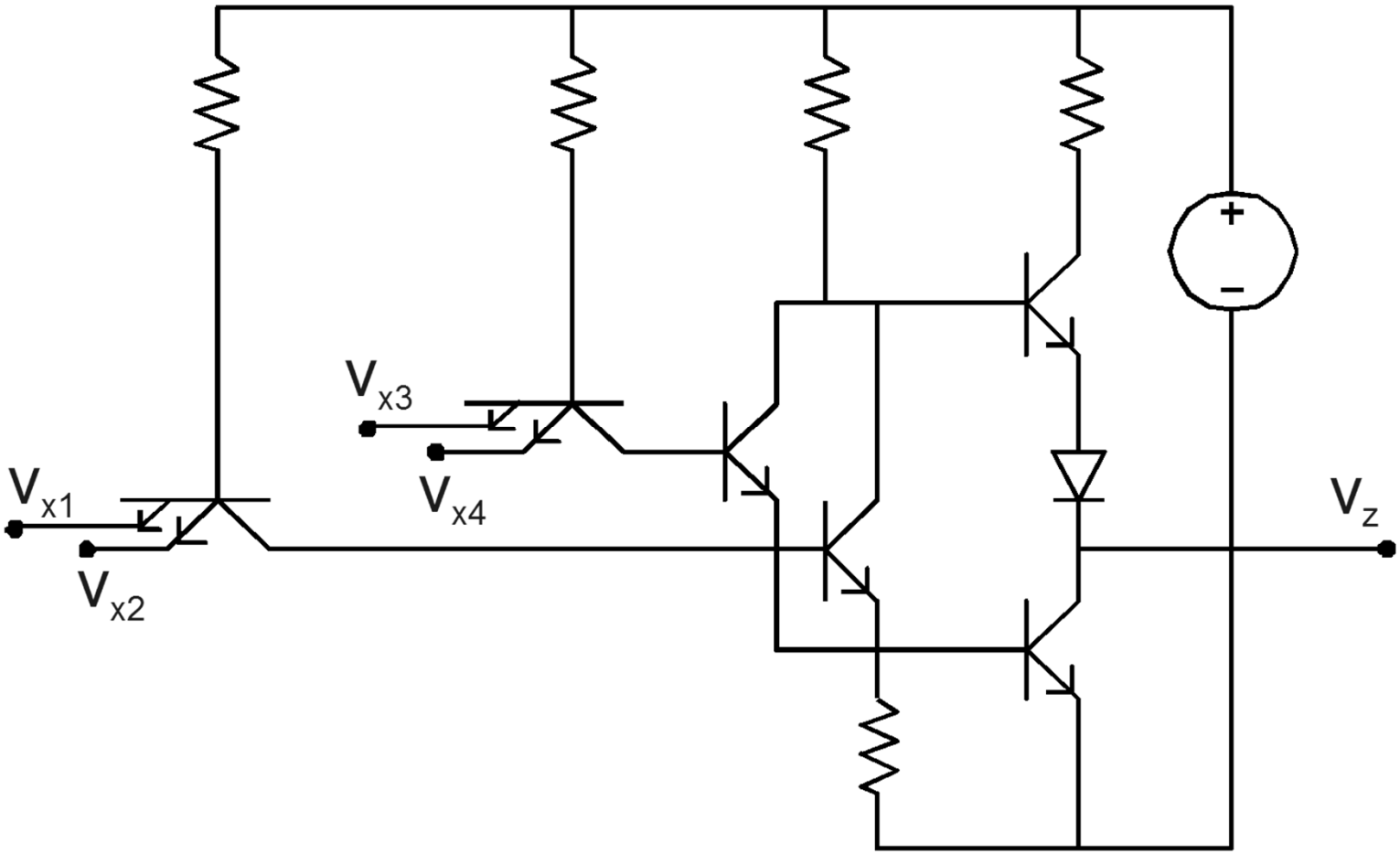
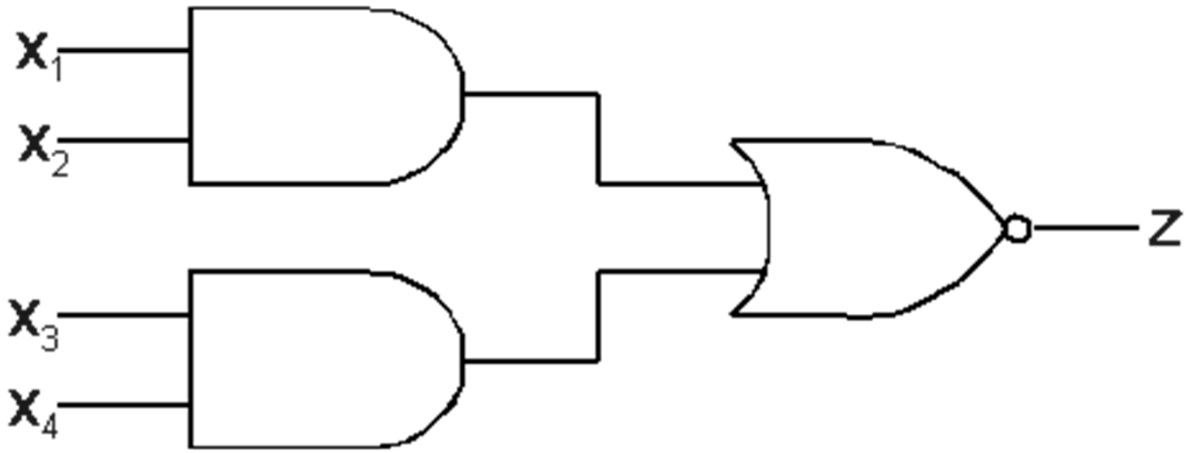


Diagrama de puertas lógicas



Expresión algebraica $f(x_1, x_2, x_3, x_4) = \overline{(x_1 * x_2) + (x_3 * x_4)}$

Tabla de verdad

$x_1\ x_2\ x_3\ x_4$	z
0000	1
0001	1
0010	1
0011	0
0100	1
0101	1
0110	1
0111	0
1000	1
1001	1
1010	1
1011	0
1100	0
1101	0
1110	0
1111	0

Vector de salida $f(x_1,x_2,x_3,x_4) = (1,1,1,0,1,1,1,0,1,1,1,0,0,0,0,0)$

Forma canónica disjuntiva $f(x_1,x_2,x_3,x_4) = \sum m(0,1,2,4,5,6,8,9,10)$

Forma canónica conjuntiva $f(x_1,x_2,x_3,x_4) = \prod M(3,7,11,12,13,14,15)$

Mapa de Karnaugh.

x_1,x_2 / x_3,x_4	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	0	0	0	0
10	1	1	0	1