Tema 19. Fundamentos de electrónica digital. Tratamiento digital de la información. Sistemas de numeración. Álgebra de Boole: Variables y operaciones. Aritmética binaria. Funciones lógicas y tablas de verdad. Simplificación de funciones. Puertas lógicas: Tipología, funciones y características. Familias lógicas y tecnologías digitales.

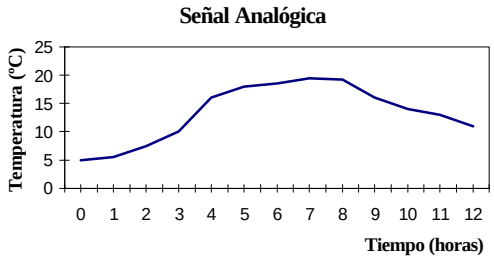
19.1. Fundamentos de electrónica digital

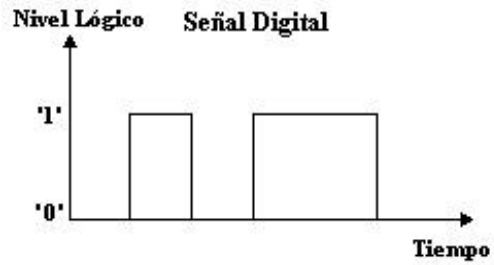
La rápida evolución experimentada por la tecnología electrónica permite diseñar y construir sistemas para procesado y tratamiento de información de bajo coste, reducido volumen, gran capacidad de almacenamiento y unas altas prestaciones. Esto hace que los sistemas electrónicos aparezcan cada vez más y en un mayor número de áreas; desde productos de uso doméstico hasta en complejos procesos de producción industrial.

Los sistemas electrónicos pueden clasificarse en dos grandes grupos: analógicos y digitales, en función de las señales que manipulan; es decir, de los valores que pueden tomar las diferentes variables que intervienen en el sistema.

Los **sistemas analógicos** van a trabajar con señales analógicas; es decir, señales cuya magnitud toma valores continuos. Ejemplos de señales analógicas son: temperatura, altura, sonido, … En los sistemas analógicos, los dispositivos electrónicos que los constituyen trabajan en zona lineal.

Los **sistemas digitales** son sistemas para procesamiento, tratamiento o transmisión de la información, en el que dicha información está limitada a tomar valores en un conjunto discreto. Estas señales, cuya magnitud sólo puede tomar un valor de entre un conjunto discreto de valores son las señales digitales. A diferencia de los sistemas analógicos, en los digitales los dispositivos que los constituyen van a funcionar como interruptores.





19.2. Tratamiento digital de la información

Muchos sistemas analógicos están siendo sustituidos por sistemas digitales que realizan funciones similares debido a sus **ventajas** inherentes:

- Mayor fiabilidad, propia de los circuitos integrados.
- Mayor facilidad de diseño.
- Flexibilidad, debido al carácter programable de muchos circuitos digitales.
- Procesado y transmisión de datos de una forma más eficiente y fiable.

- Facilidad de almacenamiento.
- Menor coste en general.

En los sistemas electrónicos digitales hay dos posibles valores de magnitud; es decir, se trabaja con señales binarias. Estos estados se representan con los dígitos '0' y '1', y van a estar asociados a unos niveles de tensión. En la figura anterior el '1' está asociado al valor más alto de tensión (V_H) y el '0' al nivel bajo (V_L), se habla de lógica positiva. Si se asocia el '0' al valor más elevado de voltaje, y el '1' al más bajo, se habla de lógica negativa.

Los sistemas digitales se clasifican en dos grandes grupos:

- Combinacionales: las salidas en cualquier instante de tiempo dependen del valor de las entradas en ese mismo instante de tiempo (salvo los retardos propios de los dispositivos electrónicos). Son por tanto, sistemas sin memoria.
- Secuenciales: la salida del sistema va a depender del valor de las entradas en ese instante de tiempo y del estado del sistema; es decir, de la historia pasada del sistema. Son sistemas con memoria.

En este tema nos centraremos en el diseño de sistemas digitales combinacionales con puertas lógicas, y en un tema posterior analizaremos los sistemas secuenciales.

19.3. Sistemas de numeración

Los sistemas digitales trabajan con variables binarias (pueden tomar dos valores o estados diferentes). Por consiguiente, el desarrollo matemático de los sistemas digitales se basa en el sistema de numeración binario. Además, se emplean el sistema octal y hexadecimal para facilitar la representación de cantidades expresadas en binario. Todos estos sistemas de numeración son polinómicos y presentan las siguientes características:

- Todo número es una expresión formada por un conjunto de símbolos (dígitos o cifras), cada uno con un valor fijo y diferente a los demás.
- El número de símbolos distintos que se pueden usar en el sistema de numeración, constituye su base.
- El valor que expresa una determinada combinación de dígitos en una base determinada depende de:
  - El valor de los símbolos.
  - La posición de los dígitos dentro de la combinación.
- Cada posición del dígito tiene un valor intrínseco que aumenta de derecha a izquierda según potencias sucesivas de la base.

En un sistema de numeración de base b un número cualquiera N_b puede ser expresado mediante su expresión polinómica:

$$N_b = a_n b^n + a_{n-1} b^{n-1} + \ldots + a_1 b^1 + a_0 b^0 + a_{-1} b^{-1} + a_{-2} b^{-2} + \ldots a_{-m+1} b^{-m+1} + a_{-m} b^{-m}$$

Donde: a_i = coeficientes (son los dígitos del número).

b^i = valores intrínsecos de cada posición. Si i$0 estamos en la parte entera del número, y si i<0 estamos en la parte fraccionaria.

Sin embargo, la forma más usual de representar un número es mediante sus coeficientes: $N_b = a_n\, a_{n-1} \ldots a_1 a_0\, a_{-1} a_{-2} \ldots a_{-m}$

A continuación se representan los sistemas de numeración empleados en digital:

Sistema	Base	Dígitos
Binario	dos	0,1
Octal	ocho	0,1,2,3,4,5,6,7
Hexadecimal	dieciseis	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

19.4. Álgebra de Boole: Variables y operaciones

19.4.1 Definición

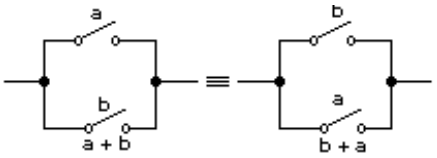
George Boole formuló en el siglo XIX un álgebra de aplicación en la teoría de conjuntos. Lo realmente interesante del álgebra de Boole es que permite expresar y analizar de una forma muy eficiente las operaciones de los circuitos lógicos.

Un álgebra de Boole es toda clase o conjunto B de elementos que pueden tomar dos valores perfectamente diferenciados (representados por 0 y 1) y que están relacionados por dos operaciones binarias, denominadas suma lógica + y producto lógico · que cumplen los siguientes postulados:

(a) Ambas operaciones son conmutativas:

$a + b = b + a$

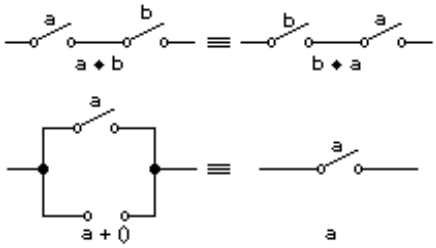
$a \cdot b = b \cdot a \qquad \forall\, a, b \in B$



(b) Existen elementos neutros dentro de B para ambas operaciones:

$a + 0 = 0 + a = a \qquad \forall\, a \in B$

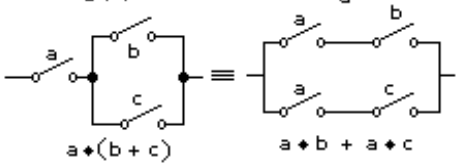
$a \cdot 1 = 1 \cdot a = a \qquad \forall\, a \in B$



(c) Las operaciones son distributivas cada una respecto a la otra:

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

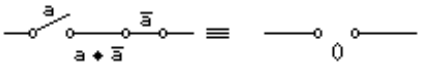
$a \cdot (b + c) = (a \cdot b) + (a \cdot c) \quad \forall\, a, b, c \in B$



(d) Cada elemento de B tiene su complemento:

$a + a' = 1$

$a \cdot a' = 0 \qquad \forall\, a \in B$



Se puede recurrir a los diagramas de contactos (fueron los primeros circuitos digitales

utilizados) para comprender mejor estas leyes. La operación suma se asimila a la conexión en paralelo de contactos y la operación producto se asimila a la conexión serie. El inverso de un contacto es otro cuyo estado es siempre el opuesto del primero (está cerrado cuando aquel está abierto, y viceversa). El elemento 0 es un contacto siempre abierto y el 1 un contacto siempre cerrado.

19.4.2. Teoremas

Basándose en los postulados anteriores se deducen los teoremas que se exponen a continuación:

Teorema 1: Ley de idempotencia

$a = a + a$

$a = a \cdot a$

Teorema 2:

$a + 1 = 1$

$a \cdot 0 = 0$

Teorema 3: Ley de absorción

$a + ab = a$

$a (a + b) = a$

Teorema 4 : En un álgebra de Boole las operaciones + y · son asociativas:

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

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

Teorema 5: se verifica que para todo elemento a de un álgebra de Boole:

$a'' = a$

Teorema 6: Leyes de De Morgan

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

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

En general:

$(b_1 + b_2 + \ldots b_n)' = b_1' \cdot b_2' \cdot \ldots \cdot b_n'$

$(b_1 . b_2 . \ldots \cdot b_n)' = b_1' + b_2' + \ldots b_n'$

Teorema 7: Principio de dualidad: Si en una expresión válida intercambiamos las operaciones (+, ·) y los elementos neutros ($0 \leftrightarrow 1$), la nueva expresión así obtenida también sigue siendo válida.

19.5. Aritmética binaria

La aritmética binaria es la base de muchas de las operaciones que realizan los sistemas digitales y las computadoras. Las cuatro operaciones básicas se muestran en las siguientes tablas:

Sumando 1		Sumando 2	Suma	Acarreo
0	+	0	0	0
0	+	1	1	0
1	+	0	1	0
1	+	1	1	1

Minuendo		Sustraendo	Diferencia	Acarreo
0	-	0	0	0
0	-	1	1	1
1	-	0	1	0
1	-	1	0	0

Factor 1		Factor 2	Producto
0	·	0	0
0	·	1	0
1	·	0	0
1	·	1	1

Dividendo		Divisor	Cociente	Resto
0	/	0	No def.	No def.
0	/	1	0	0
1	/	0	No def.	No def.
1	/	1	1	0

19.6. Funciones lógicas y tablas de verdad

19.6.1. Definición

Una función lógica es una variable binaria F, cuyo valor es igual al de una expresión algebraica de variables lógicas (complementadas o no) unidas por las operaciones lógicas (+,·).

Ejemplos de funciones lógicas serían:

$F_1 = F_1(a,b,c) = ab + c$

$F_2 = F_2(a,b,c,d) = a' + bc + ad'$

Donde $F_1(a,b,c)$ indica que la función depende de las variables a, b y c.

Las funciones lógicas se clasifican en:

a) Completamente especificadas: si a cada una de las posibles combinaciones de las variables de entrada corresponde un valor único y definido de la función.

b) Incompletas: si a una o más combinaciones de entrada se le puede asignar el valor 0 ó 1 indistintamente (por ejemplo: si nunca van a aparecer determinadas combinaciones de entrada o si aparecen me da igual el valor que tome la función). Se habla de indiferencias.

19.6.2. Representación de funciones digitales

Además de las expresiones algebraicas vistas en el apartado anterior las funciones digitales se van a representar de otras formas:

a) Tabla de verdad: se indica el valor 0 ó 1 que toma la función para cada una de las combinaciones posibles de las variables de la función.

a	b	F(a,b)
0	0	1
0	1	0
1	0	1
1	1	1

Se correspondería a $F(a,b) = a + b'$

En el caso de una función incompleta, las salidas no definidas se representan mediante un guión o una x en la tabla de la verdad.

b) Forma canónica: representación basada en el Teorema de Shannon que definiremos a continuación:

Minterm: dadas n variables, un *minterm* es un término producto en el que aparecen todas las variables solo una vez, complementadas o sin complementar.
Maxterm: dadas n variables, un *maxterm* es un término suma en el que aparecen todas las variables solo una vez, complementadas o sin complementar.

Teorema de expansión de Shannon: Toda función digital de n variables se puede expresar como suma de *minterms* o producto de *maxterms*.

La forma canónica de una función digital sería su expresión como suma de *minterms* o como producto de *maxterms*. La siguiente tabla nos indica como obtenerla:

Tipo de ecuación	Método de obtención	Convenio a aplicar
Ecuación *minterms*	Obtener suma de *minterms* que hacen 1 la función	0 Variable negada 1 Variable sin negar
Ecuación *maxterms*	Obtener producto de *maxterms* que hacen 0 la función	0 Variable sin negar 1 Variable negada

Filas	x	y	z	f
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Lo ejemplificaremos con la siguiente función dada por la siguiente tabla:

- Suma de *minterms*: $f(x,y,z) = x´yz´ + x´yz + xy´z´ + xyz´ + xyz$
  Que en forma abreviada sería: $f(x,y,z) = \sum m(2,3,4,6,7)$
- Producto de *maxterms*: $f(x,y,z) = (x+y+z)\cdot(x+y+z´)\cdot(x´+y+z´)$
  Que en forma abreviada sería: $f(x,y,z) = \prod M\ (0,1,5)$

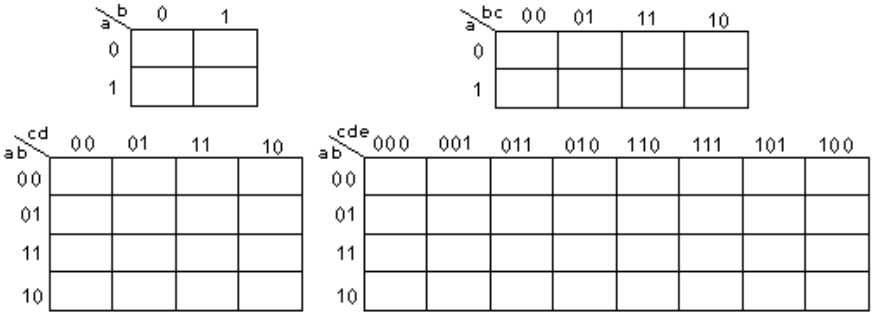
Filas	x	y	z	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	X
5	1	0	1	0
6	1	1	0	X
7	1	1	1	1

Si hay indiferencias, el proceso anterior se repite, como se puede ver en el siguiente ejemplo:

- Expresiones válidas, considerando las indiferencias, son:
  $f = x´y´z´ + x´yz´ + xyz$
  $f = x´y´z´ + x´yz´ + xyz + xy´z´ + xyz´.$
- De forma abreviada: $f = \sum m(0,2,7) + \sum d(4,6)$ ó $f = \prod M\ (1,3,5) + \prod d\ (4,6)$

c) Mapa de Karnaugh: Esta forma de representación va a permitir al diseñador lógico simplificar las funciones, logrando que el circuito digital resultante sea más sencillo y económico a la hora de implementarse. El mapa de Karnaugh para una

función de n variables está constituido por un rectángulo dividido en 2^n celdas, que representan cada una de las posibles combinaciones de las variables de la función. Se coloca un 1 en aquellas celdas tales que, para la combinación de las variables correspondiente, la función tome el valor 1 y un guión en las indiferencias; en las restantes no se coloca nada. En la siguiente figura aparecen las distintas formas que adoptan los mapas de Karnaugh para funciones de hasta cinco variables:



19.7. Simplificación de funciones

Lo que se busca con la simplificación o minimización es obtener **el circuito óptimo** asociado a una función digital respecto a algún criterio. Dependiendo de cada situación, se puede optimizar el circuito pensando en:

- El número de niveles de puertas que deben atravesar las señales de entrada hasta alcanzar la salida (hablamos del tiempo de propagación de las señales).
- El coste en cuanto al nº de puertas o al nº de conexiones del circuito.
- El coste económico del sistema final.
- Tamaño del sistema.
- Tiempo de diseño y fabricación.
- Uso de circuitos disponibles o fácilmente accesibles.
- Fiabilidad, mantenibilidad del sistema.

Existen diversos métodos de simplificación, entre los que destacan el método de los diagramas de Karnaugh y el método de Quine-McCluskey. El primero se basa en la capacidad humana para percibir patrones en representaciones gráficas, mientras que el de Quine-McCluskey es un método sistemático, pensado para ser programado (computerizado).

De este proceso de simplificación, obtendremos una expresión óptima de la función que se implementará a nivel *hardware* (montaje del circuito).

19.7.1. Método de Karnaugh

La técnica que explicaremos aquí está basada en la utilización del mapa de Karnaugh para la localización y simplificación de adyacencias. Aunque este método permite una rápida minimización, tiene la desventaja de que para funciones de 6 o

más variables el proceso se complica demasiado, siendo más recomendable usar el método de Quine McCluskey.

Dos *minterms* constituyen una adyacencia de primer orden si la expresión de ellos solo difiere en una variable; son adyacencias de primer orden, por ejemplo, $xy´z$ y $xy´z´$. Aplicando la propiedad asociativa a la suma de *minterms* adyacentes, pueden reducirse a uno solo en el que falta la variable que difería en ambos términos:

$$xy´z + xy´z´ = xy´(z + z´) = xy´\cdot 1 = xy´$$

Asimismo, dos adyacencias de primer orden constituyen una adyacencia de segundo orden si sus expresiones (ya reducidas) solo difieren en una variable.

$$\left.\begin{array}{ll} xy´z & xy´z \rightarrow \quad xy´ \\ xyz & xyz´ \rightarrow \quad xy \end{array}\right\} \rightarrow x$$

El fundamento del método de simplificación basado en mapas de Karnaugh consiste en que aquellos *minterms* que se puedan reducir serán adyacentes y ocuparán:

- Casillas contiguas, en grupos de 2,4,8,16,32, ...
- Grupos de casillas sobre los bordes del mapa opuestos entre sí.
- Grupos de casillas constituidas por las cuatro esquinas del mapa.

El **algoritmo** del método consta de los siguientes pasos:

a) Selección de todos los grupos de casillas indicados anteriormente, de modo que cada uno cumpla las siguientes normas:

   - Las casillas que forman las agrupaciones (también denominadas cubos) deben contener 1 o indiferencia.

   - Los grupos de casillas deben contener al menos un 1 (no pueden ser todas indiferencias)

   - Debe procurarse conseguir grupos del máximo número de casillas (este número, como se ha indicado, ha ser potencia de 2). Para ello, seguimos el siguiente orden:

     - Se toman los unos que no se puedan agrupar con ningún otro.
     - Se forman grupos de dos unos (o 1 e indiferencia) que no pueden constituir ningún grupo de cuatro.
     - Se itera el proceso hasta cubrir todos los unos.

   -Al hacer los agrupamientos, debe procurarse formar grupos con el mayor número de términos posibles, no existiendo ningún problema en que un término aparezca en más de un agrupamiento. Las indiferencias servirán para conseguir grupos más grandes.

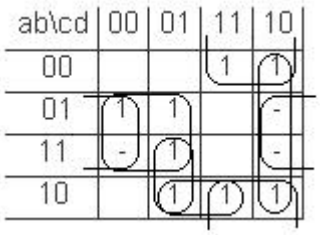
b) Se identifican los términos productos que representa cada grupo.

c) Se construye la expresión como la suma de implicantes primos, incluyendo:

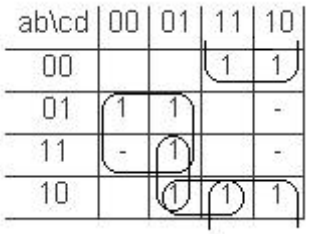
- Todos los *minterms* correspondientes a grupos que cubren algún 1 en exclusividad (estos *minterms* se denominan implicantes primos esenciales).
- Se comprueba sobre el mapa qué unos no han quedado cubiertos por los implicante primos esenciales y se seleccionan los implicantes primos (no esenciales) de menor coste que los incluyan.

Se ejemplificará este método con un **ejemplo**:

Dada la función F = Σm(2,3,4,5,9,10,11,13) + Σd(6,12,14) dibujamos su diagrama de Karnaugh y extraemos todos los cubos.



Hay que seleccionar todos los implicantes primos esenciales (1 cubiertos en exclusividad), y los 1 no cubiertos se deben cubrir con el mínimo coste (implicantes con el menor número posible de variables).



Finalmente, obtenemos dos expresiones mínimas equivalentes:

F = bc´ + b´c + ac´d = bc´ + b´c + ab´d

19.8. Puertas lógicas: Tipología, funciones y características

19.8.1. Tipología y funciones

En este apartado se van a definir las funciones lógicas básicas, que son las que se utilizan fundamentalmente en la realización de sistemas digitales. Estas funciones existen implementadas en electrónica digital, denominándose puertas lógicas a los circuitos que las realizan.

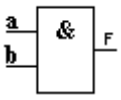
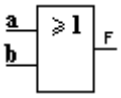
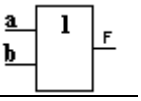
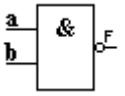
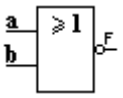
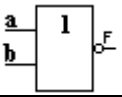
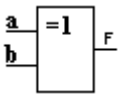
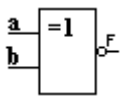
En la siguiente figura aparecen las funciones básicas, su nombre, su tabla de verdad y su simbología. Se incluye la nueva simbología (símbolos rectangulares, norma IEEE Std. 91-1984) y la simbología antigua (IEEE Std. 91-1973), debido a que ambas son ampliamente usadas y coexisten en la actualidad.

Se han representado las funciones AND, OR, NAND, NOR, EXOR y NEXOR de dos variables; ya que, para más variables lo único que cambia es que las puertas tendrán más entradas y en las tablas de verdad aparecen más variables; pero su comportamiento será como el indicado.

Hay que señalar que aunque las funciones EXOR y NEXOR se pueden realizar asociando otras puertas (AND, OR y NOT por ejemplo), el perfeccionamiento de las tecnologías de fabricación ha permitido fabricar circuitos integrados que realizan estas funciones.

Este último apunte sirve para definir lo que se conoce como conjunto suficiente de conectores o funcionalmente completo: viene a ser un conjunto de conectores (puertas) unitarios y/o binarios tal que cualquier otra función de n variables se puede expresar relacionando las n variables a través de este conjunto de conectores. Por ejemplo, con el conjunto {OR,AND,NOT} se va a poder implementar cualquier función digital; por tanto, sería un conjunto suficiente de conectores. Otros conjuntos son {NAND} y {NOR}, lo que nos quiere decir que cualquier circuito digital se puede realizar usando sólo puertas NAND, o sólo puertas NOR.

- La salida de una puerta AND es 1 solo si todas las variables son simultáneamente 1.
- La salida de una puerta OR es 0 cuando todas las variables valen simultáneamente 0.
- En un buffer la salida sigue a la entrada.
- La salida de una puerta NAND es la negada de la función AND.
- La salida de una puerta NOR es la negada de la función OR.
- En un inversor, obtenemos a la salida la negación de la entrada.
- La puerta EXOR tendrá un 1 en la salida cuando una de sus entradas sea 1 y la otra 0.
- La salida de una puerta NEXOR es la negada de la función EXOR.

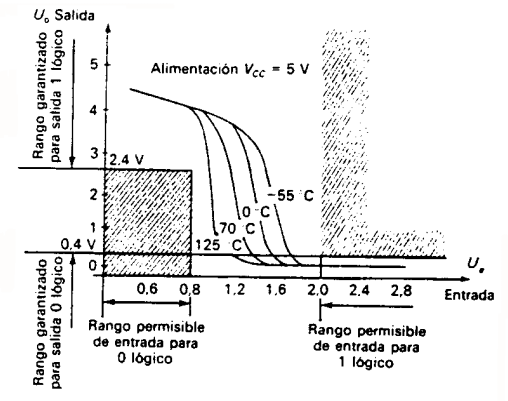
TablaVerdad	Función	Nombre	Símbolo	Símbolo
a b | F(a,b) 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1	F(a,b)=a·b	AND		
a b | F(a,b) 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1	F(a,b)=a+b	OR		
a | F(a) 0 | 0 1 | 1	F(a) = a	BUFFER		
a b | F(a,b) 0 0 | 1 0 1 | 1 1 0 | 1 1 1 | 0	$F(a,b)=\overline{a \cdot b}$	NAND		
a b | F(a,b) 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 0	$F(a,b)=\overline{a+b}$	NOR		
a | F(a) 0 | 1 1 | 0	$F(a) = \overline{a}$	NOT		
a b | F(a,b) 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 0	$F(a,b)=\overline{a} \cdot b+a \cdot \overline{b}$ ó $F(a,b)=a \oplus b$	EXOR		
a b | F(a,b) 0 0 | 1 0 1 | 0 1 0 | 0 1 1 | 1	$F(a,b)=\overline{a} \cdot \overline{b}+a \cdot b$ ó $F(a,b)=a \ ? \ b$	NEXOR		

19.8.2. Características

En los catálogos del fabricante de puertas lógicas se indican un elevado número de parámetros y características de cada puerta lógica integrada, necesarios para realizar los diseños de circuitos prácticos. Seguidamente enunciaremos los más importantes:

a. Niveles lógicos de funcionamiento. La información, que inicialmente se representa por dos valores de tensión discreta, en la práctica, a causa de variaciones en valores de componentes (tolerancias), fuentes de alimentación, temperatura, ..., se representa por dominios o bandas de tensión. Los niveles lógicos de funcionamiento son los márgenes de valores de tensión que el fabricante permite o garantiza para cada uno de los dos estados lógicos entre los que puede funcionar un circuito digital.

b. Característica de transferencia (*voltage transfer function)* Es una gráfica que relaciona la tensión de entrada con la de salida en una puerta lógica.



c. Inmunidad al ruido(*noise margins).* Se define como el margen de ruido electrónico que es capaz de soportar la puerta sin que se produzcan alteraciones en su funcionamiento. Se mide en voltios.

d. Tiempo de propagación (*propagation delay*). Es el tiempo que transcurre entre el momento de introducir una información en la entrada de una puerta lógica y el instante en que se produce la respuesta en la salida de esta. La inversa de esta característica define la **frecuencia** máxima de trabajo de la puerta.

e. Cargabilidad de salida (*fan out)* Es un número entero que nos indica la cantidad de entradas de puertas lógicas de la misma familia que se pueden conectar a la salida de una puerta.

19.9. Familias lógicas y tecnologías digitales

Por tecnología de fabricación y familia lógica se entiende:

a) Tecnología de fabricación: es la forma de construir un circuito integrado desde el punto de vista de sus principios de funcionamiento o fabricación. Como ejemplo de diferentes tecnologías están el empleo de transistores bipolares, el hecho de que los transistores del circuito trabajen entre corte o saturación, o que el circuito se fabrique sobre una base de zafiro.

b) Familia lógica: es el conjunto de circuitos integrados digitales que, dentro de una misma tecnología, emplean el mismo tipo de componentes y de circuito base en su estructura.

En el siguiente esquema se expone la clasificación general de las tecnologías de fabricación y de sus correspondientes familias lógicas:

1**. Tecnología de base**

- Tecnología bipolar.
  - -Tecnología saturada: *Familias TTL, RTL, DTL, HTL.*
  - -Tecnología no saturada: *Familias TTL, Schottkey, ECL.*
- Tecnología MOS: *Familias PMOS, CMOS, NMOS.*
- Tecnología BICMOS.
- Tecnología CCD.

2**. Tecnología de apoyo**

. I^2L para tecnologías bipolares.

. SOS, implantación iónica para MOS.

19.9.1. Aplicaciones

El hecho de tener una amplia gama de dispositivos lógicos entre los que elegir está muy bien, pero puede producir muchas confusiones en el momento de tomar una decisión sobre la familia más adecuada para una determinada aplicación. En general, los tipos **CMOS** normales son los más adecuados cuando es importante una **baja corriente de consumo**, pero no lo son para altas velocidades. Por el contrario, si lo que se busca es la máxima velocidad de conmutación la familia ECL es la más rápida.

Puede decirse que la gama **74LSxx** es la mejor cuando se necesita una **corriente moderada y una velocidad de funcionamiento bastante elevada**.

Las familias **74HCxx y 74HCTxx** son unas propuestas muy interesantes que ofrecen al mismo tiempo un bajo consumo y una elevada velocidad de funcionamiento, pero aún son mucho **más caras** que sus equivalentes normales de la familia 74LSxx, y en realidad no son interesantes a no ser que se necesiten realmente una elevada velocidad y una economía de corriente.

También es posible adoptar una mezcla de varios tipos, empleando solamente los dispositivos de alta velocidad en los puntos del circuito en que se necesitan, pero entonces hay que tener cierto cuidado en asegurarse de que todos los circuitos integrados son compatibles con los que excitan y que no se superan las limitaciones de carga (*fan-out*) nominales.