Álgebra de Boole

Fundamentos y Aplicaciones Básicas en la Electrónica Digital Moderna.

Ing. Arturo Gustavo Tajani

• La profundización teórica del tema **“Algebra de Boole”** puede ser consultada en una extensa bibliografía, a la que no se pretende reemplazar.

Simplemente entrando en “Internet”, en un “buscador” como “Google”, “Algebra de Boole” permite acceder a muchos artículos de gran calidad.

• Solo daremos una definición y mencionaremos los enunciados de algunas leyes básicas (sin discriminar entre “postulados” y “teoremas”), como para iniciarnos en este tema.

• La idea fundamental es empezar a entender el Álgebra de Boole en el contexto de las aplicaciones en la electrónica digital moderna.

• Con los ejemplos que se verán, se pretende tener una idea razonablemente clara sobre los principios elementales de funcionamiento que rigen los sistemas de cálculo de máquinas ariméticas y computadoras electrónicas.

Definición del Álgebra de Boole

- *Es toda clase o conjunto de elementos que:*

1. *pueden tomar* ***dos*** *valores o estados* ***claramente distintos*** *(o perfectamente diferenciados)*
2. *están relacionados entre sí por* ***dos operaciones binarias****, llamadas* ***suma lógica (+)*** *y* ***producto lógico (·).***

   ***** *operación binaria es aquella que, definida entre elementos de un conjunto, da por resultado un elemento del mismo conjunto.*

- *Se incorpora también la* ***negación (´),*** *aunque no entre en la definición.*

- Son ejemplos de álgebras de Boole: el álgebra de proposiciones o de juicios formales y el álgebra de redes eléctricas o de conmutación, vistos anteriormente.

• Las variables o elementos, se indican con letras mayúsculas: A, B, C, D, etc. .(aunque en “Álgebra Proposicional”, como se acostumbra, se hayan utilizado letras minúsculas). También se pueden utilizar números o nombres representativos.

• Los dos estados posibles se anotan: “0” y “1”.

• De igual manera que en el álgebra convencional, la suma lógica se indica con (**+**) y el producto lógico con (·) o simplemente se elimina.

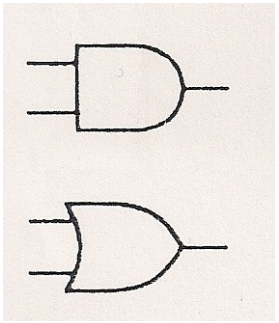
• Así el producto **a·b** se puede poner **ab** . La negación puede señalarse con: **- , ~,** con un guión superior o simplemente con **´**.

Por ejemplo la función lógica: $F = A+B+(B\cdot(-C))\cdot((\sim D)+E)$

puede escribirse: $F = A+B+(B\overline{C})(\overline{D}+E)$

• Definidas las cuatro Compuertas Lógicas básicas, es oportuno agregar a las anteriores, otras cuatro compuertas, que se obtienen respectivamente, conectando un inversor (negación), a continuación de cada una de las compuertas básicas.

• Esto, lejos de ser una complicación y gracias a la tecnología del “circuito integrado”, permite simplificar y abaratar las cosas.

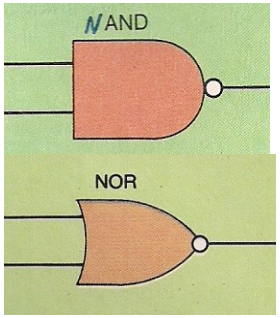


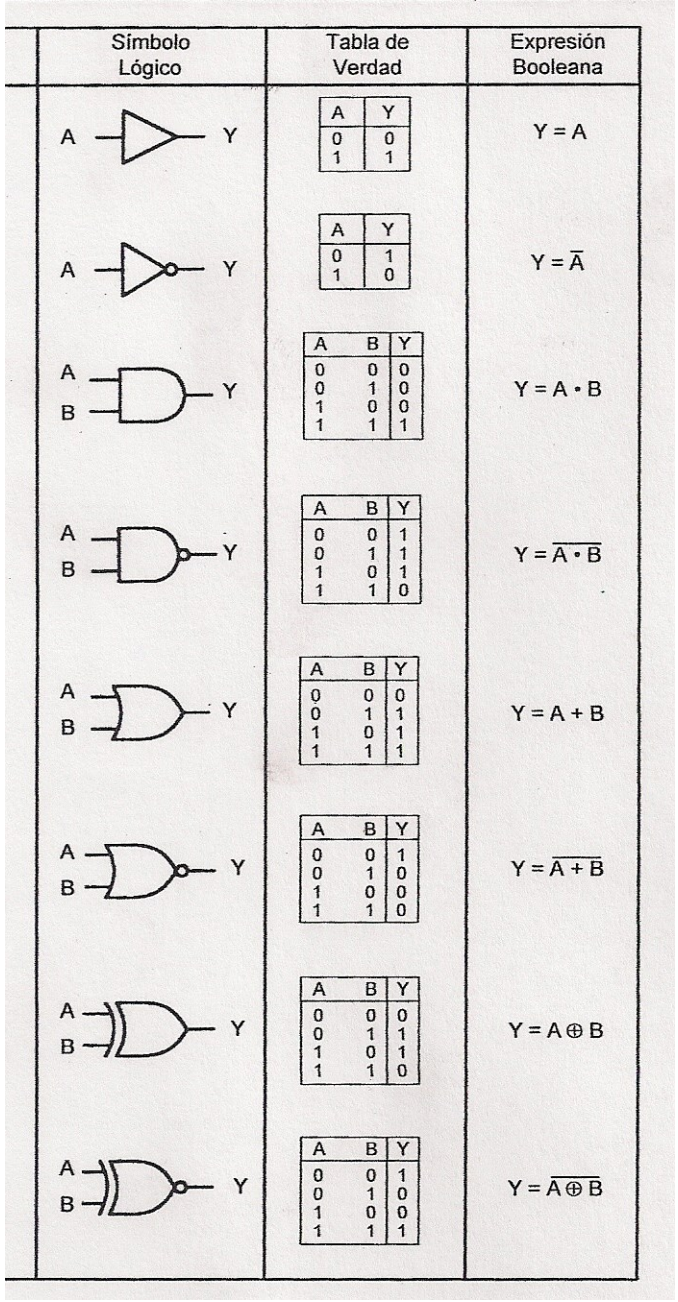
Y negada ó “NOY” (NAND)

O negada ó “NOO” (NOR)

O negada ó “NOO” (EXNOR)

Doble negación (Búfer)





- En la tabla se indican los símbolos gráficos de las nuevas compuertas, que se forman agregándole en la salida de las básicas, un pequeño circulo que indica la inversión.

- También en cada caso está la “tabla de verdad” y la correspondiente “Expresión Booleana”.

• Existen compuertas de hasta ocho entradas. Ejemplos de símbolos y Tablas de Verdad, se dan a continuación, para tres y cuatro entradas, tanto para NAND como para NOR.

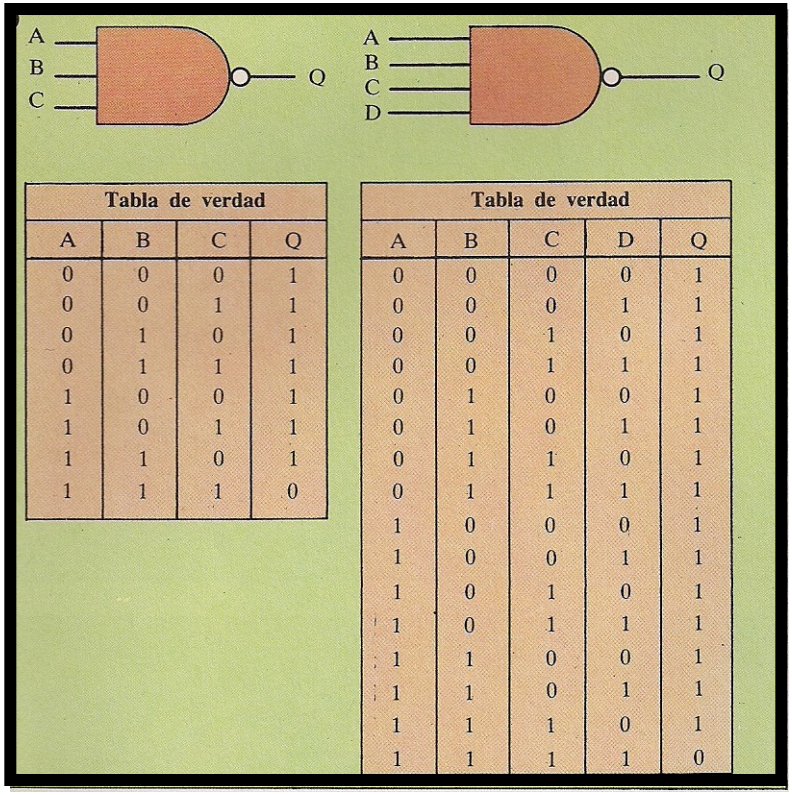


Tabla de verdad			
A	B	C	Q
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Tabla de verdad				
A	B	C	D	Q
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	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

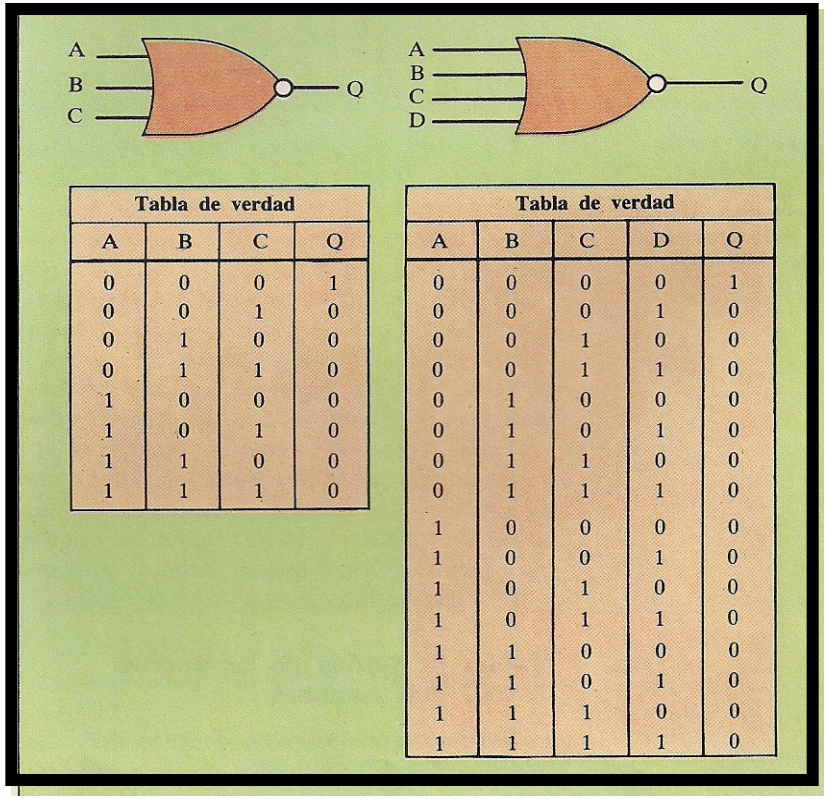


Tabla de verdad			
A	B	C	Q
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Tabla de verdad				
A	B	C	D	Q
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

• Habiendo definido :

las “**proposiciones**”

los “**conectivos lógicos**”

las “**compuertas lógicas**”

y sus correspondientes “**símbolos gráficos**”,

podemos a partir de ahora operar con estos conceptos a través del:

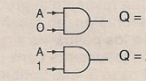
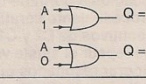
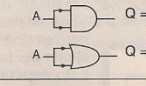
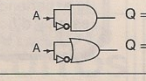
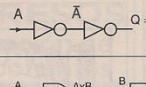
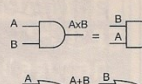
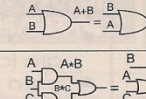
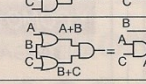
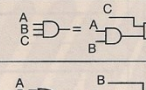
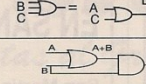
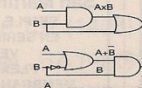
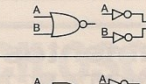
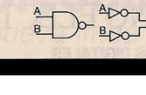
“**Algebra de Boole**”.

• En nuestros razonamientos nos independizaremos de los elementos materiales, aunque a título informativo mencionaremos, cuando corresponda, los componentes reales existentes.

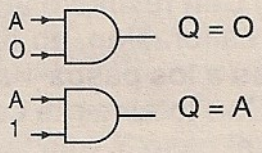
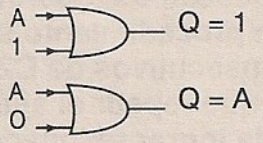
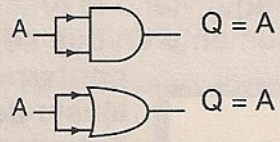
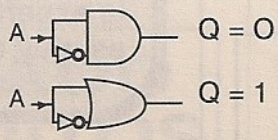
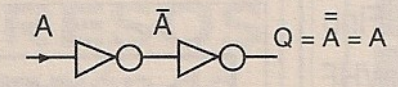
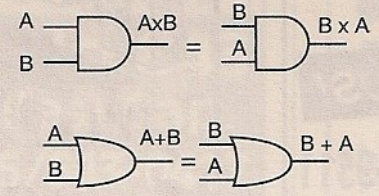
Enunciados, teoremas, propiedades, leyes ó reglas del Algebra de Boole.

En cada caso, se da a continuación del nombre de la propiedad, la expresión matemática Booleana, la materialización en forma de circuito de compuertas simples y una breve explicación.

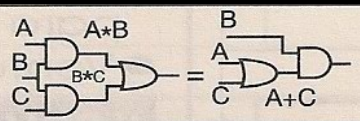
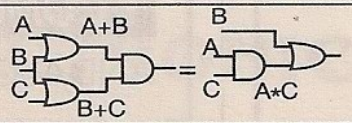
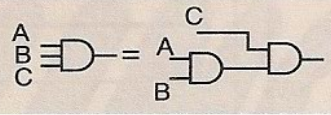
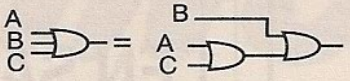
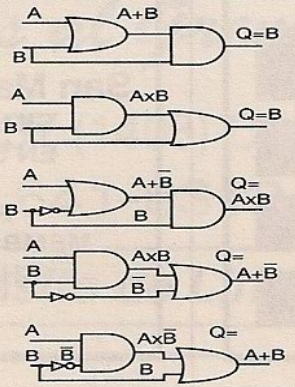
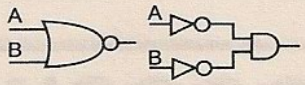
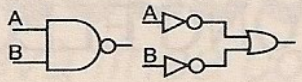
Para mayor claridad, se presenta la misma tabla en dos secciones.

N°	NOMBRE	EXPRESION	MATERIALIZACION	EXPLICACION
1 2	Propiedades de la operación AND	$A * O = O$ $A * 1 = A$		La operación AND de una variable A con O es siempre O y con 1 es siempre igual a la variable A (A puede ser O ó 1)
3 4	Propiedades de la operación OR	$A + 1 = 1$ $A + O = A$		La operación OR de una variable A con 1 es siempre 1 y con O es siempre igual a la variable A (A puede ser O ó 1)
5 6	Leyes de tautología (o redundancia)	$A * A = A$ $A + A = A$		La operaciones AND y OR de una variable A con sí misma (o sea: se aplica A a las dos entradas), dan como resultado A.
7 8	Leyes del complemento	$A * \overline{A} = O$ $A + \overline{A} = 1$		La operación AND de A con su complemento ($\overline{A}$) es siempre O y la operación OR es siempre 1.
9	Ley de la doble inversión	$\overline{\overline{A}} = A$		Si A se inverte dos veces, se vuelve a obtener A
10 11	Propiedades conmutativas	$A * B = B * A$ $A + B = B + A$		Las entradas de una compuerta AND u OR se pueden permutar sin que se altere el resultado (la salida) Estas reglas son similares a las del álgebra común (el orden de los factores no altera el producto y el orden de los sumandos no altera la suma)
12	Propiedad distributiva de la operación AND	$A * B + B * C = B*(A+C)$		Regla similar al primer caso de factoreo del álgebra común
13	Propiedad distributiva de la operación OR	$(A + B) * (B + C) = B + A * C$		Cuando un termino B está en ambos factores de un producto de sumas, el producto puede transformarse en una suma de productos
14	Propiedad asociativa de la operación AND	$A * B * C = (A * B) * C$		Similar a la propiedad asociativa del álgebra común (son también válidas las demás combinaciones de A, B, y C)
15	Propiedad asociativa de la operación OR	$A + B + C = (A + C) + B$		Similar a la propiedad asociativa del álgebra común (son también válidas las demás combinaciones de A, B, y C)
16 17 18 19 20	Propiedades de absorción	$(A + B) * B = B$ $(A * B) + B = B$ $(A + \overline{B}) * B = A * B$ $(A * B) + \overline{B} = A + \overline{B}$ $(A * \overline{B}) + B = A + B$		Estas propiedades son exclusivas del álgebra de Boole y no se cumplen en el álgebra común. Al "absorber" variables, permiten simplificar el diseño con compuertas lógicas.
21	Primer teorema de De Morgan	$\overline{A + B} = \overline{A} * \overline{B}$		La inversa del resultado de la operación OR de dos variables es igual a la operación AND entre las inversas de dichas variables
22	Segundo teorema de De Morgan	$\overline{A * B} = \overline{A} + \overline{B}$		La inversa del resultado de la operación AND de dos variables es igual a la operación OR entre las inversas de dichas variables

Leyes Básicas del Algebra de Boole (1 a 11)

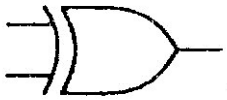
N°	NOMBRE	EXPRESION	MATERIALIZACION	EXPLICACION
1 2	Propiedades de la operación AND	$A * O = O$ $A * 1 = A$		La operación AND de una variable A con O es siempre O y con 1 es siempre igual a la variable A (A puede ser O ó 1)
3 4	Propiedades de la operación OR	$A + 1 = 1$ $A + O = A$		La operación OR de una variable A con 1 es siempre 1 y con O es siempre igual a la variable A (A puede ser O ó 1)
5 6	Leyes de tautología (o redundancia)	$A * A = A$ $A + A = A$		La operaciones AND y OR de una variable A con sí misma (o sea: se aplica A a las dos entradas), dan como resultado A.
7 8	Leyes del complemento	$A * \overline{A} = O$ $A + \overline{A} = 1$		La operación AND de $\overline{A}$ con su complemento ($\overline{A}$) es siempre O y la operación OR es siempre 1.
9	Ley de la doble inversión	$\overline{\overline{A}} = A$		Si A se inverte dos veces, se vuelve a obtener A
10 11	Propiedades conmutativas	$A * B = B * A$ $A + B = B + A$		Las entradas de una compuerta AND u OR se pueden permutar sin que se altere el resultado (la salida) Estas reglas son similares a las del álgebra común (el orden de los factores no altera el producto y el orden de los sumandos no altera la suma)

(12 a 22)

12	Propiedad distributiva de la operación AND	$A * B + B * C = B*(A+C)$		Regla similar al primer caso de factoreo del álgebra común
13	Propiedad distributiva de la operación OR	$(A + B) * (B + C) = B + A * C$		Cuando un termino B está en ambos factores de un producto de sumas, el producto puede transformarse en una suma de productos
14	Propiedad asociativa de la operación AND	$A * B * C = (A * B) * C$		Similar a la propiedad asociativa del álgebra común (son también válidas las demás combinaciones de A, B, y C)
15	Propiedad asociativa de la operación OR	$A + B + C = (A + C) + B$		Similar a la propiedad asociativa del álgebra común (son también válidas las demás combinaciones de A, B, y C)
16 17 18 19 20	Propiedades de absorción	$(A + B) * B = B$ $(A * B) + B = B$ $(A + \overline{B}) * B = A * B$ $(A * B) + \overline{B} = A + \overline{B}$ $(A * \overline{B}) + B = A + B$		Estas propiedades son exclusivas del álgebra de Boole y no se cumplen en el álgebra común. Al "absorber" variables, permiten simplificar el diseño con compuertas lógicas.
21	Primer teorema de De Morgan	$\overline{A + B} = \overline{A} * \overline{B}$		La inversa del resultado de la operación OR de dos variables es igual a la operación AND entre las inversas de dichas variables
22	Segundo teorema de De Morgan	$\overline{A * B} = \overline{A} + \overline{B}$		La inversa del resultado de la operación AND de dos variables es igual a la operación OR entre las inversas de dichas variables

• Una primera aplicación de los conceptos anteriores es proponerse sintetizar una “Disyunción Excluyente **O** (EXOR)”, utilizando solo compuertas básicas **O, Y y NO (OR, AND y NOT).**

Recordemos el símbolo y su tabla de verdad:



A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

- La tabla se puede describir con palabras, de varias formas, por ejemplo:

1. *La salida toma el estado “1”, si una y solo una de las entradas está en “1”.*
2. *La salida es “1”, si A = 1 ó B = 1, pero* no *ambas a la vez. Esta expresión nos llevaría a :* $A \underline{O} B = (A+B) \cdot \overline{(AB)}$
3. *La salida es “0”, si A y B son ambas iguales a “1” ó ambas iguales a “0”. La función Booleana sería:*

$$A \underline{O} B = \overline{(A\,B)} + \overline{(\overline{A}\,\overline{B})}$$

4. *La salida es “0”, si ambas entradas son iguales entre sí.* *Etc. Etc. Etc.*

• De esta manera en forma intuitiva, se puede encontrar la expresión Booleana conveniente. Pero cuando el problema se complica porque el enunciado que se plantea es mas avanzado, se requiere entonces tener alguna forma sistemática para expresar la función lógica correspondiente.

• El método que se propone, entre otros, se llama

"Suma de Productos" y consiste en:

1º Crear la tabla de verdad del enunciado planteado.

A	B	Y
0	0	0
0	1	1 ←
1	0	1 ←
1	1	0

2º Disponer de una suma de tantos paréntesis como "**unos**" haya en la tabla de verdad, en la proposición de salida. En nuestro caso: dos.

$$A \underline{O} B = (\quad) + (\quad)$$

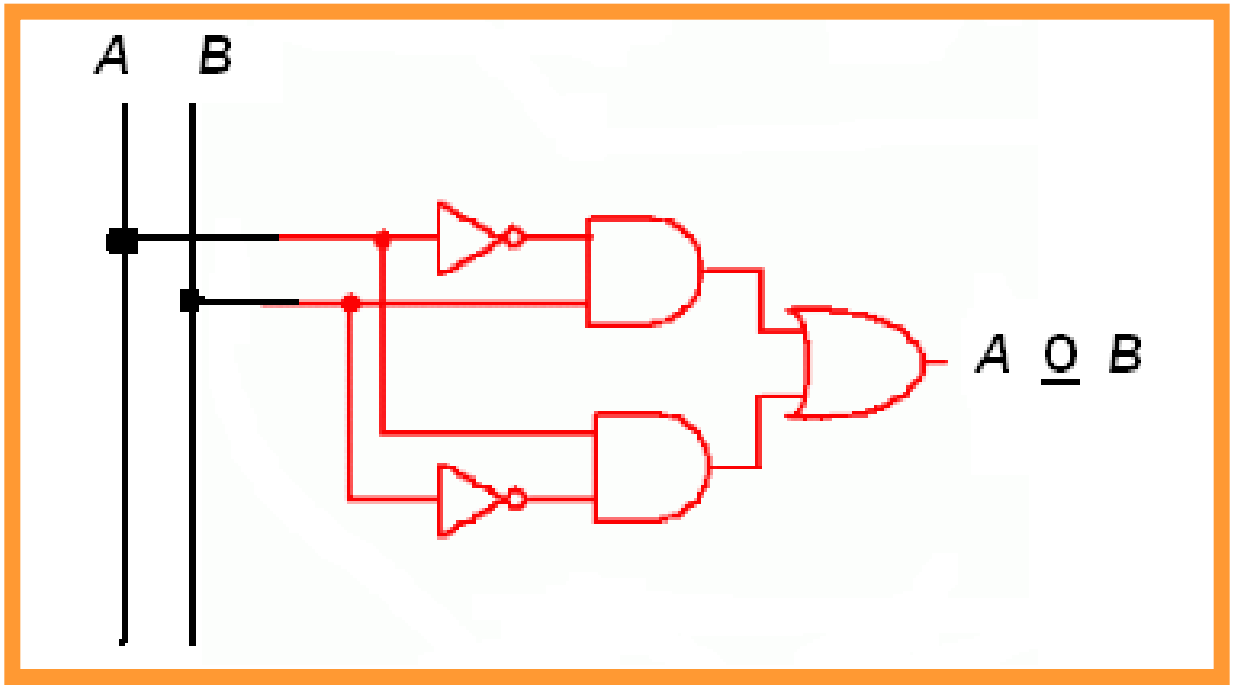
3º Dentro de cada paréntesis irá un producto lógico entre todas las variables de la entrada, tomándolas **directas** cuando valgan "**1**" y **negadas** cuando su valor sea "**0**":

$$A \underline{O} B = (\overline{A}. B) + (A . \overline{B})$$

• Cada producto formado se llama “minitérmino” y resulta claro que su valor será “1”, solo cuando se dé la combinación de “0s y 1s” correspondiente.

$$A \underline{O} B = (\overline{A}B) + (A\overline{B})$$

• Finalmente el circuito de compuertas lógicas para el O excluyente, será:



- Para continuar con aplicaciones significativas, se propone tomar como objeto, el funcionamiento macroscópico de una simple calculadora digital de cuatro operaciones.

- Recordemos que en la vida diaria se utiliza la familiar numeración decimal, pero toda máquina que realice operaciones aritméticas, desde una simple calculadora hasta la mas compleja computadora, opera internamente, sin ninguna excepción, en el sistema de **numeración binaria**.

- Se crea entonces la necesidad de introducir un **"sistema codificador"** en la entrada de la máquina, que vincule el teclado numérico exterior con los elementos internos de cálculo.

- **Enunciado:** **Codificador Decimal a Binario:** Se trata de convertir al sistema binario natural, un dígito expresado en forma decimal. Tendrá evidentemente diez entradas vinculadas a las teclas, de forma tal que cuando solo una de ellas es oprimida, aparezca en cuatro puntos internos, el conjunto de ceros y unos de la combinación binaria natural correspondiente.

• **Planteo Lógico:** Llamaremos con los dígitos 0; 1; 2; 3; 4; 5; 6; 7; 8 y 9, a las diez proposiciones de entrada (cada tecla tiene dos estados: oprimida o no).

• Las proposiciones de salida serán D; C; B y A y estarán en correspondencia con los valores de cada posición binaria 8; 4; 2 y 1. El valor lógico "1" para las entradas, se toma como "tecla oprimida".

• **Tabla de Verdad:**

0	1	2	3	4	5	6	7	8	9	D	C	B	A
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0	0	0	0	1	0	1
0	0	0	0	0	0	1	0	0	0	0	1	1	0
0	0	0	0	0	0	0	1	0	0	0	1	1	1
0	0	0	0	0	0	0	0	1	0	1	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	1

• Con la observación de la tabla se deducen, en forma intuitiva, las correspondientes **"Funciones Booleanas":**

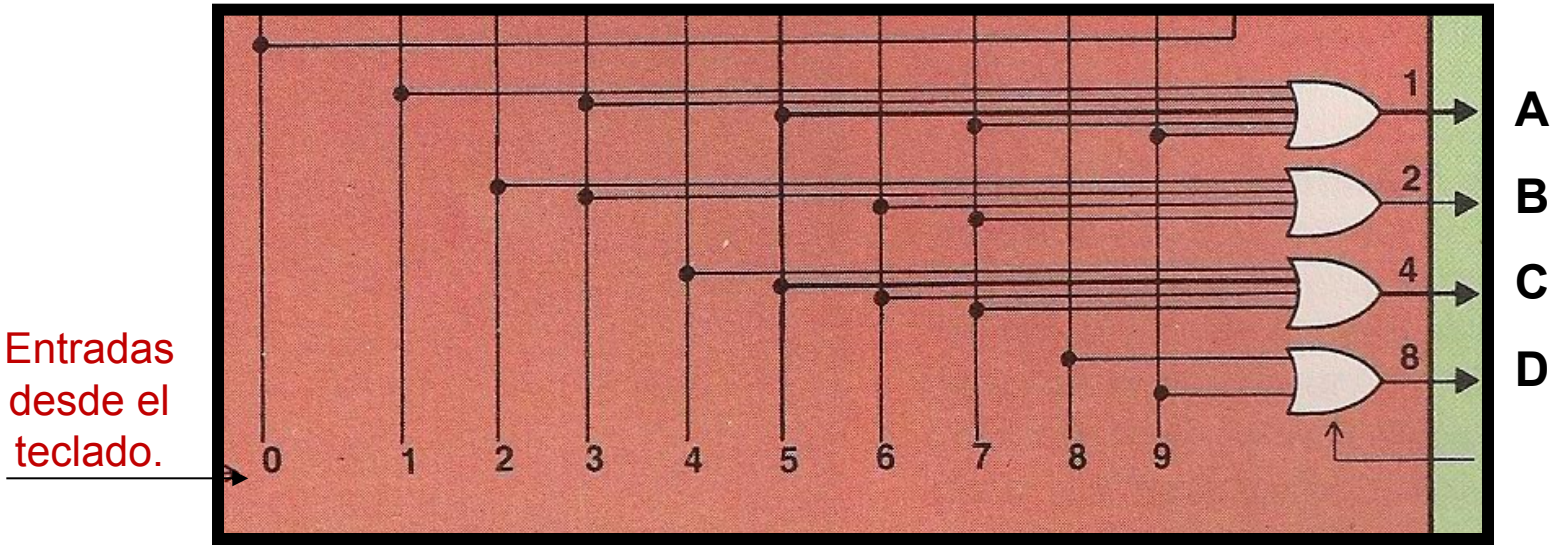
$$A = 1 + 3 + 5 + 7 + 9$$

$$B = 2 + 3 + 6 + 7$$

$$C = 4 + 5 + 6 + 7$$

$$D = 8 + 9$$

Que se materializan con el **circuito de compuertas lógicas " O ",** de varias entradas.



• La siguiente aplicación nos lleva a la operación inversa de la anterior, es decir el órgano de cálculo arroja un resultado numérico binario, y debe interpretarse como un número decimal.

• La necesidad ahora es de crear un sistema “decodificador” en el interior de la máquina.

• **Enunciado:** **Decodificador Binario a Decimal** . Se trata de convertir un binario natural en un dígito decimal. El sistema tendrá cuatro proposiciones de entrada, diez de salida y estarán vinculadas de manera tal que para cada combinación de los estados binarios, se excite solo la salida decimal correspondiente.

• **Planteo Lógico:** En forma semejante al caso anterior, esta vez las proposiciones de entrada serán las letras D; C; B; y A , mientras que con los dígitos 0; 1; 2; . . .9 se nombran las diez proposiciones de salida.

- **Tabla de Verdad:**

Los espacios en blanco son obviamente “ceros”, que no se colocan para mayor claridad de la tabla.

D	C	B	A	0	1	2	3	4	5	6	7	8	9
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						1				
0	1	1	0							1			
0	1	1	1								1		
1	0	0	0									1	
1	0	0	1										1

Funciones Booleanas:

Aplicando el método de **Suma de productos** se tendrá para cada proposición de salida, un único producto lógico entre las variables de entrada. Se toman directas si valen “1” y serán negadas si valen “0”.

- $0 = \overline{D} . \overline{C} . \overline{B} . \overline{A}$
- $1 = \overline{D} . \overline{C} . \overline{B} . A$
- $2 = \overline{D} . \overline{C} . B . \overline{A}$
- $3 = \overline{D} . \overline{C} . B . A$
- $4 = \overline{D} . C . \overline{B} . \overline{A}$
- $5 = \overline{D} . C . \overline{B} . A$
- $6 = \overline{D} . C . B . \overline{A}$
- $7 = \overline{D} . C . B . A$
- $8 = D . \overline{C} . \overline{B} . \overline{A}$
- $9 = D . \overline{C} . \overline{B} . A$

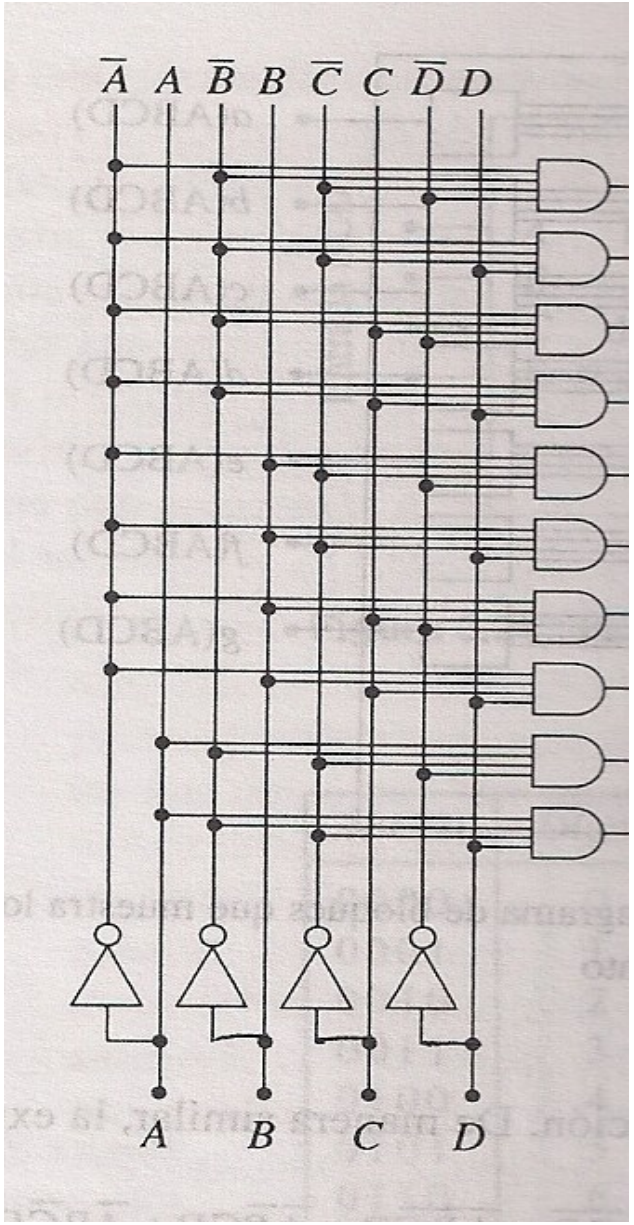
Circuito de Compuertas Lógicas:

Los inversores colocados en cada entrada, proveen al sistema la posibilidad de elegir la proposición directa o negada según la tabla.

Es muy utilizado el recurso de generar barras con las proposiciones directas y negadas.

No solo se ahorran inversores, sino que se hace mas sencilla la interpretación del gráfico y se simplifica el cableado.

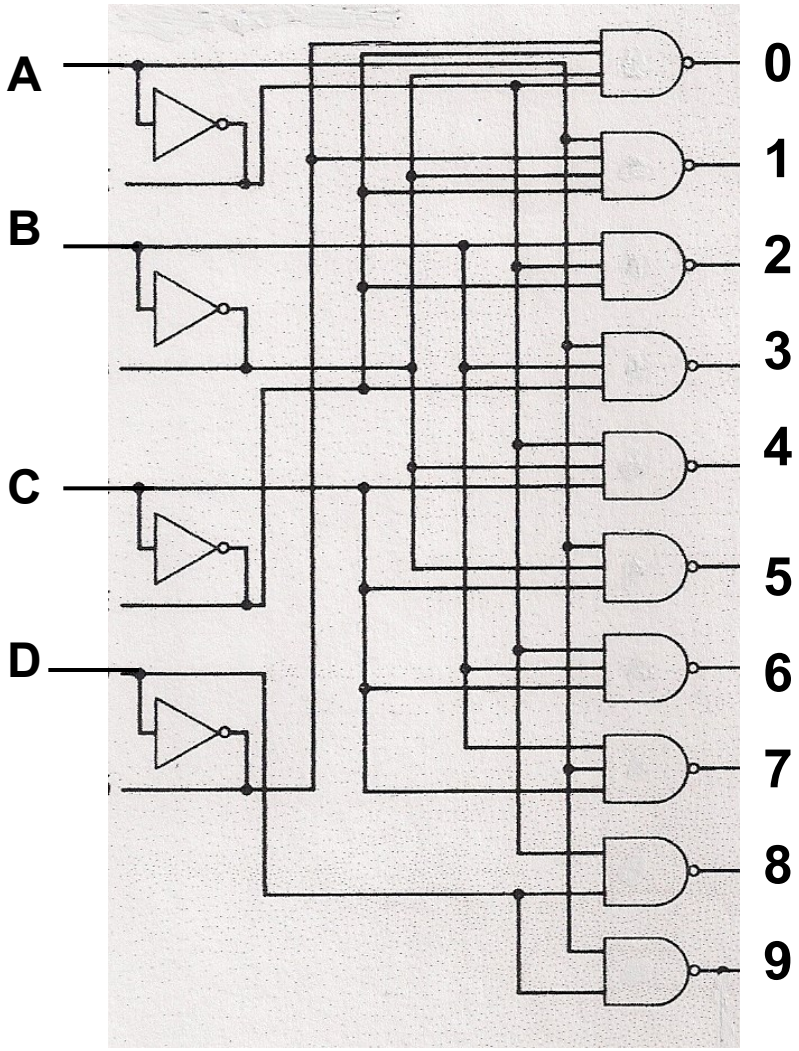
Se utilizan solo compuertas Y (AND).



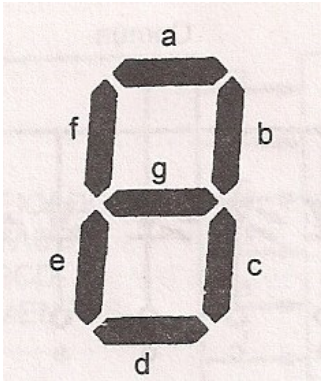
- En este segundo circuito hay algunas simplificaciones.

- Como vemos solo hay dos compuertas NAND de cuatro entradas, seis de tres entradas y dos de dos entradas, en vez de utilizar diez compuertas de cuatro entradas.

- Las compuertas NAND proveen simplificaciones en los sistemas posteriores.



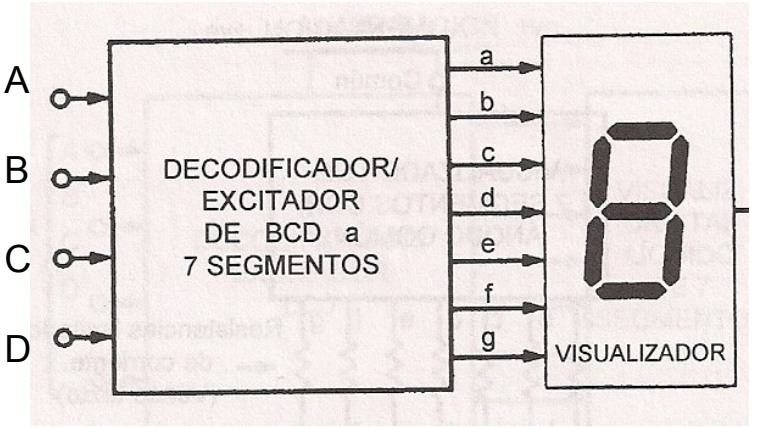
• Siguiendo con la calculadora elemental, observemos que la indicación numérica del visor, se realiza mediante una representación llamada de “ Siete segmentos “ ó de “Siete Barras”. Todos estamos familiarizados con esta indicación, que también se utiliza en ascensores, indicadores numéricos, relojes digitales, etc.



• Los segmentos luminosos pueden ser “cristales líquidos (LCD) en calculadoras y relojes, ó diodos LED, lámparas comunes o hasta tubos fluorescentes en otros sistemas . Pero la denominacion de los segmentos generalmente aceptada es la que se indica.

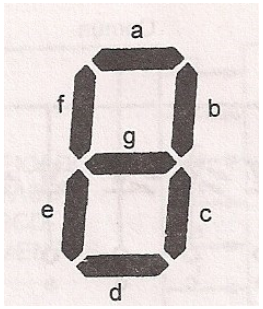
• **Decodificador ABCD a 7 segmentos.**

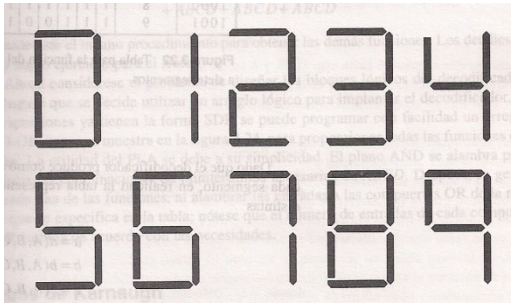
• El enunciado y el planteo lógico de este circuito, resulta claro con el gráfico que se agrega.



• **Tabla de Verdad** del decodificador BCD a 7 Seg.

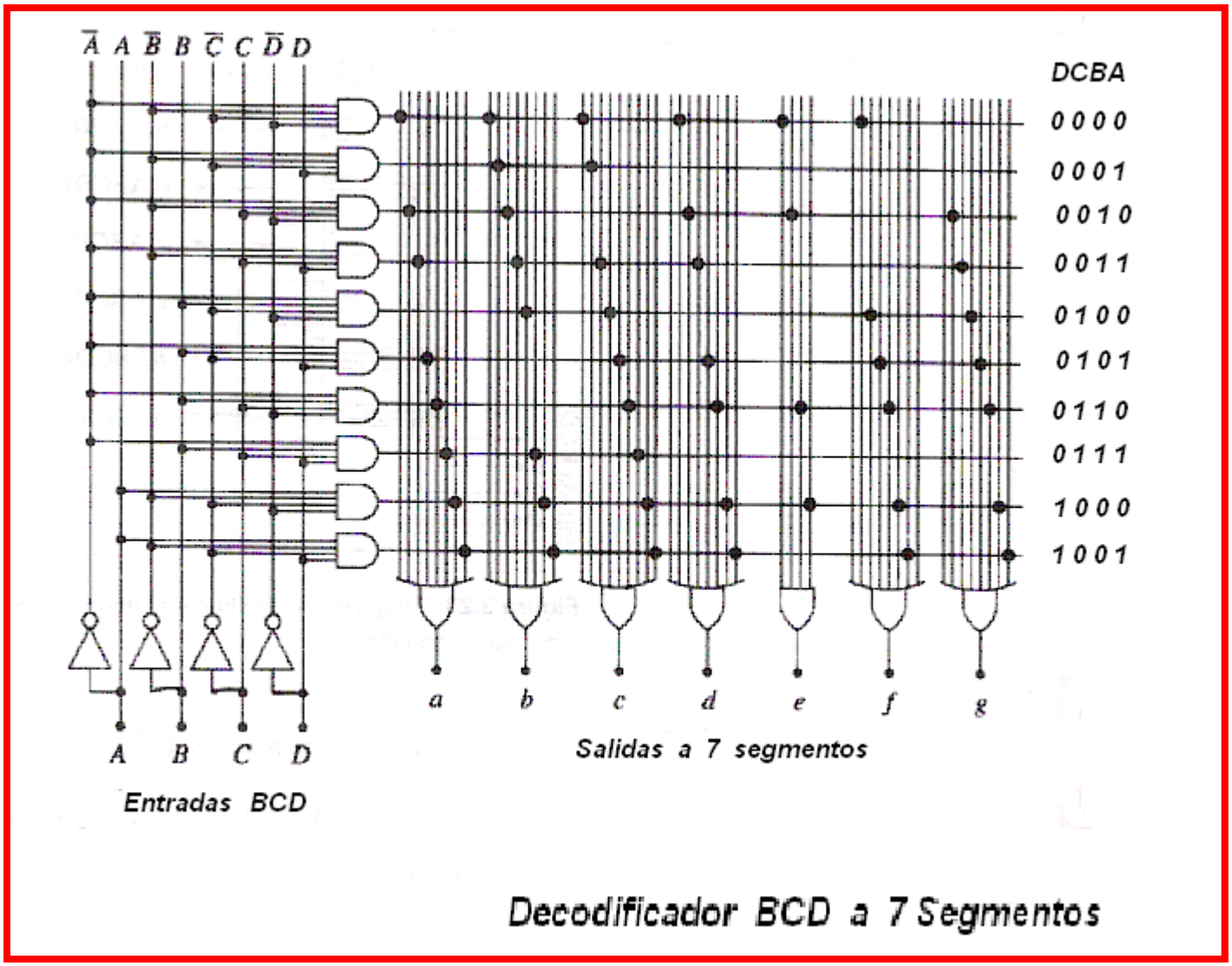
Nº	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1





• Para expresar las **funciones Booleanas** se puede aplicar el método de la **Suma de Productos**. Por ejemplo para reconocer el segmento **"a"**se tendría una suma lógica de ocho productos; para el segmento "e" solo cuatro productos; etc.

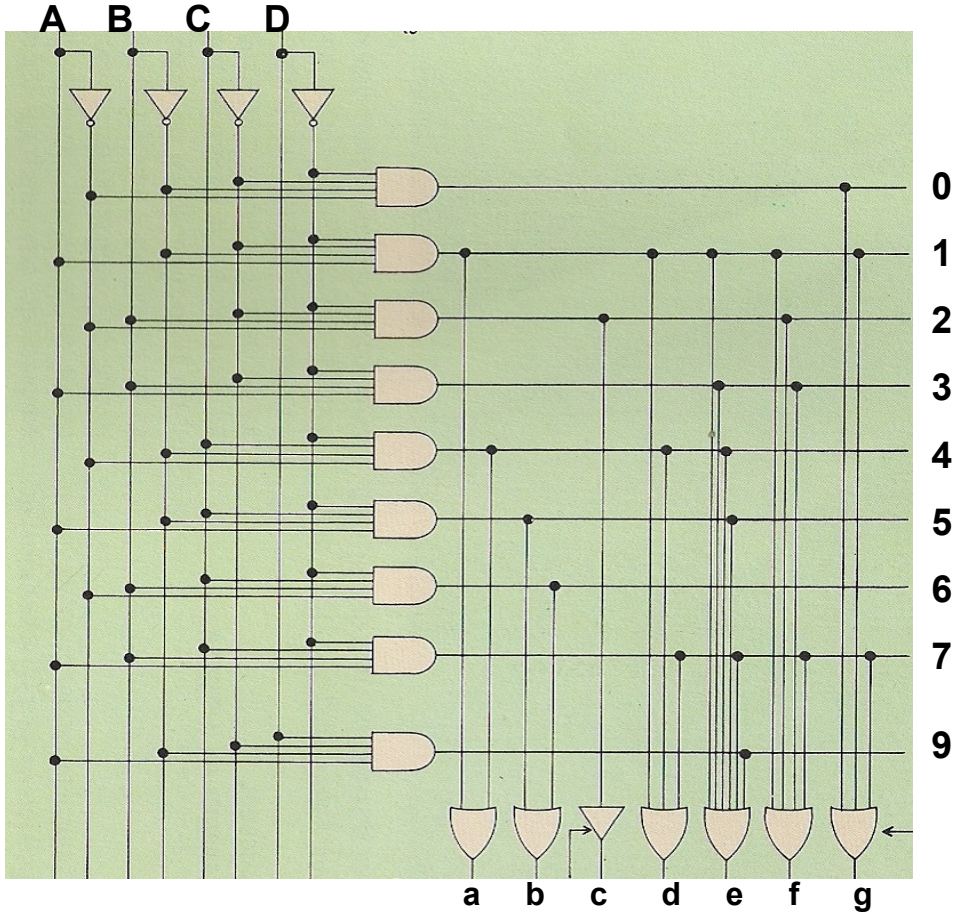
• Inmediatamente se comprueba que las expresiones son muy largas.


$\overline{A}$ A $\overline{B}$ B $\overline{C}$ C $\overline{D}$ D
DCBA
0 0 0 0
0 0 0 1
0 0 1 0
0 0 1 1
0 1 0 0
0 1 0 1
0 1 1 0
0 1 1 1
1 0 0 0
1 0 0 1
a b c d e f g
A B C D
Entradas BCD
Salidas a 7 segmentos
Decodificador BCD a 7 Segmentos

• Cabe hacer varias simplificaciones Booleanas y algunas experimentales, pero por sencillez no daremos el detalle.

• El circuito de **Compuertas Lógicas** ya simplificado, pero razonablemente entendible se muestra a continuación:

• Se puede reconocer a la izquierda del dibujo, una versión del decodificador binario a decimal anterior.



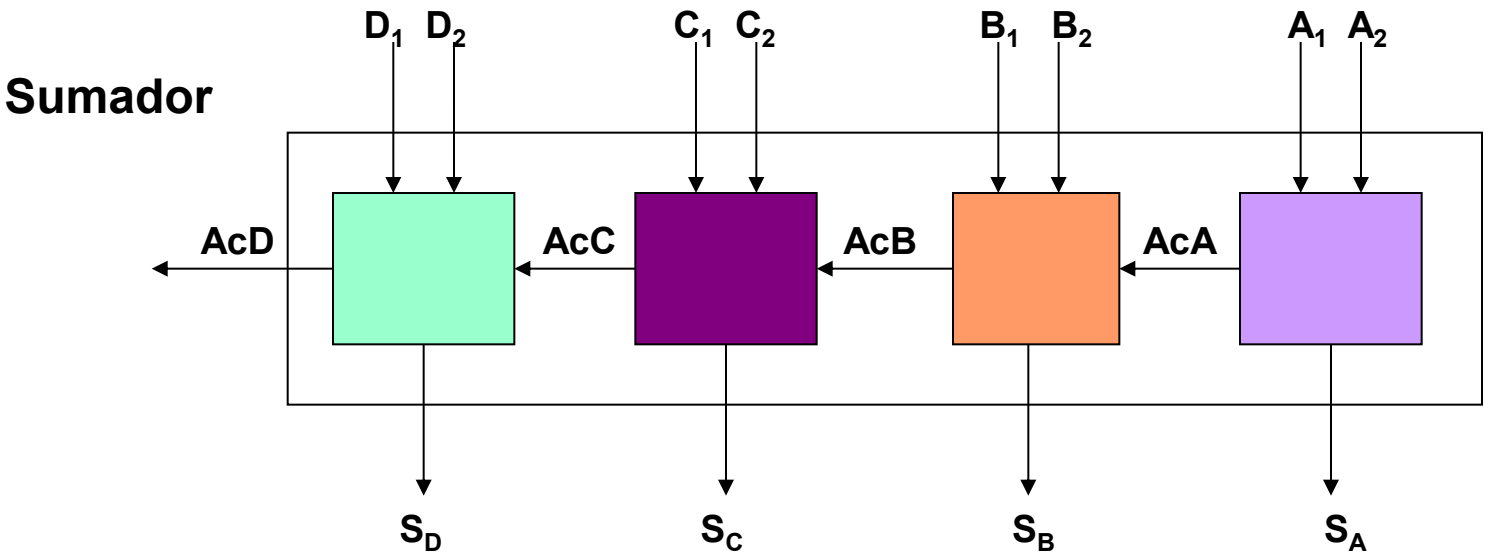
• De los Circuitos Lógicos anteriores vimos que, aún cuando se han simplificado bastante, la implementación práctica de los mismos requirió el empleo de muchas compuertas elementales.

• Si se tiene en cuenta que la solución encontrada fue para un solo dígito decimal y normalmente su manejan como mínimo ocho dígitos, empezamos a vislumbrar el crecimiento de la cantidad de "compuertas lógicas" que son necesarias disponer en cuanto se avanza en el diseño de un dispositivo tan "simple" como una sencilla calculadora.

• Nos ocuparemos ahora, en forma muy somera, de lo relativo a los elementos de cálculo, que naturalmente también serán resueltos con compuertas lógicas elementales.

• Mostraremos primero, como funciona un circuito **"Sumador Digital".**

• Se considera necesario hacer un esquema en bloques que permita fijar conceptualmente lo que todos sabemos: como se realiza una **"Suma".** Por supuesto que se describe en términos de una adición entre dos números binarios de cuatro dígitos cada uno.



• Distinguimos entre el primer bloque de la derecha, que recibe dos entradas y genera dos salidas y que se lo llama "**Semisumador"** y los siguientes bloques que reciben tres entradas y tienen también dos salidas y son nombrados como **"Sumador Total".**

• **Tabla de verdad del Semisumador:** Llamaremos A_1 y A_2 a los dígitos de entrada; S_A a la suma directa y AcA al acarreo, arrastre o el simple "***me llevo***".

• En la tabla, podemos ver que la suma equivale a un "o excluyente" (O), mientras que el acarreo es claramente equivalente a la "conjunción" (Y).

A_1	A_2	AcA	S_A
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

• **Tabla de verdad del Semisumador:** Llamaremos A_1 y A_2 a los dígitos de entrada; S_A a la suma directa y AcA al acarreo, arrastre o el simple "***me llevo***".

• En la tabla, podemos ver que la suma equivale a un "o excluyente" (O), mientras que el acarreo es claramente equivalente a la "conjunción" (Y).

• Estas observaciones nos llevan directamente a las **funciones Booleanas:**

A_1	A_2	AcA	S_A
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$S_A = A_1 \,\underline{O}\, A_2 = (\overline{A}_1 \,.\, A_2) + (A_1 \,.\, \overline{A}_2)$$

$$AcA = A_1 \,.\, A_2$$

• **Tabla de verdad del Semisumador:** Llamaremos A_1 y A_2 a los dígitos de entrada; S_A a la suma directa y AcA al acarreo, arrastre o el simple "***me llevo***".

• En la tabla, podemos ver que la suma equivale a un "o excluyente" (O), mientras que el acarreo es claramente equivalente a la "conjunción" (Y).

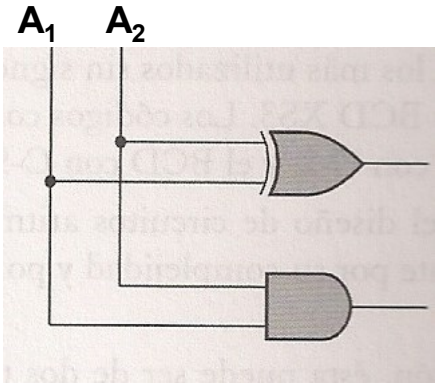
• Estas observaciones nos llevan directamente a las **funciones Booleanas:**

A_1	A_2	AcA	S_A
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$S_A = A_1 \,\underline{O}\, A_2 = (\overline{A}_1 \,.\, A_2) + (A_1 \,.\, \overline{A}_2)$$

$$AcA = A_1 \,.\, A_2$$

• El **circuito de compuertas lógicas** muestra una compuerta Y y un O excluyente (O) que por supuesto puede ser reemplazado por el diagrama ya visto.



• **Tabla de verdad del Sumador Total:** En este caso las entradas son tres, no solo los dos dígitos a sumar B_1 Y B_2, sino también el arrastre anterior AcA. Las salidas son solo dos, una la suma S_B y la segunda el acarreo AcB.

AcA	B_2	B_1	AcB Acarreo	S_B Suma
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

• **Ecuaciones Booleanas:** Aplicamos a la tabla precedente el método de la "suma de productos". Tenemos en cada caso cuatro unos, que originan la suma lógica de cuatro productos, respectivamente para cada salida.

$$S_B = (\overline{\overline{A}cA}\ \overline{B}_1\ B_2) + (\overline{\overline{A}cA}\ B_1\ \overline{B}_2) + (AcA\ \overline{B}_1\ \overline{B}_2) + (AcA\ B_1\ B_2)$$

1 2 3 4

$$AcB = (\overline{\overline{A}cA}\ B_1\ B_2) + (AcA\ \overline{B}_1\ B_2) + (AcA\ B_1\ \overline{B}_2) + (AcA\ B_1\ B_2)$$

5 6 7 8

• **Ecuaciones Booleanas:** Aplicamos a la tabla precedente el método de la "suma de productos". Tenemos en cada caso cuatro unos, que originan la suma lógica de cuatro productos, respectivamente para cada salida.

$$S_B = (\overline{AcA}\ \overline{B}_1\ B_2) + (\overline{AcA}\ B_1\ \overline{B}_2) + (AcA\ \overline{B}_1\ \overline{B}_2) + (AcA\ B_1\ B_2)$$

1 2 3 4

$$AcB = (\overline{AcA}\ B_1\ B_2) + (AcA\ \overline{B}_1\ B_2) + (AcA\ B_1\ \overline{B}_2) + (AcA\ B_1\ B_2)$$

5 6 7 8

Se pueden simplificar sacando factor común $\overline{AcA}$ de 1 y 2 y luego AcA de 3 y 4. Se obtiene que:

$$S_B = \overline{AcA}\ (B_1\ \underline{O}\ B_2) + AcA\ \overline{(B_1\ \underline{O}\ B_2)}$$

• **Ecuaciones Booleanas:** Aplicamos a la tabla precedente el método de la "suma de productos". Tenemos en cada caso cuatro unos, que originan la suma lógica de cuatro productos, respectivamente para cada salida.

$$S_B = (\overline{AcA}\ \overline{B_1}\ B_2) + (\overline{AcA}\ B_1\ \overline{B_2}) + (AcA\ \overline{B_1}\ \overline{B_2}) + (AcA\ B_1\ B_2)$$

1 2 3 4

$$AcB = (\overline{AcA}\ B_1\ B_2) + (AcA\ \overline{B_1}\ B_2) + (AcA\ B_1\ \overline{B_2}) + (AcA\ B_1\ B_2)$$

5 6 7 8

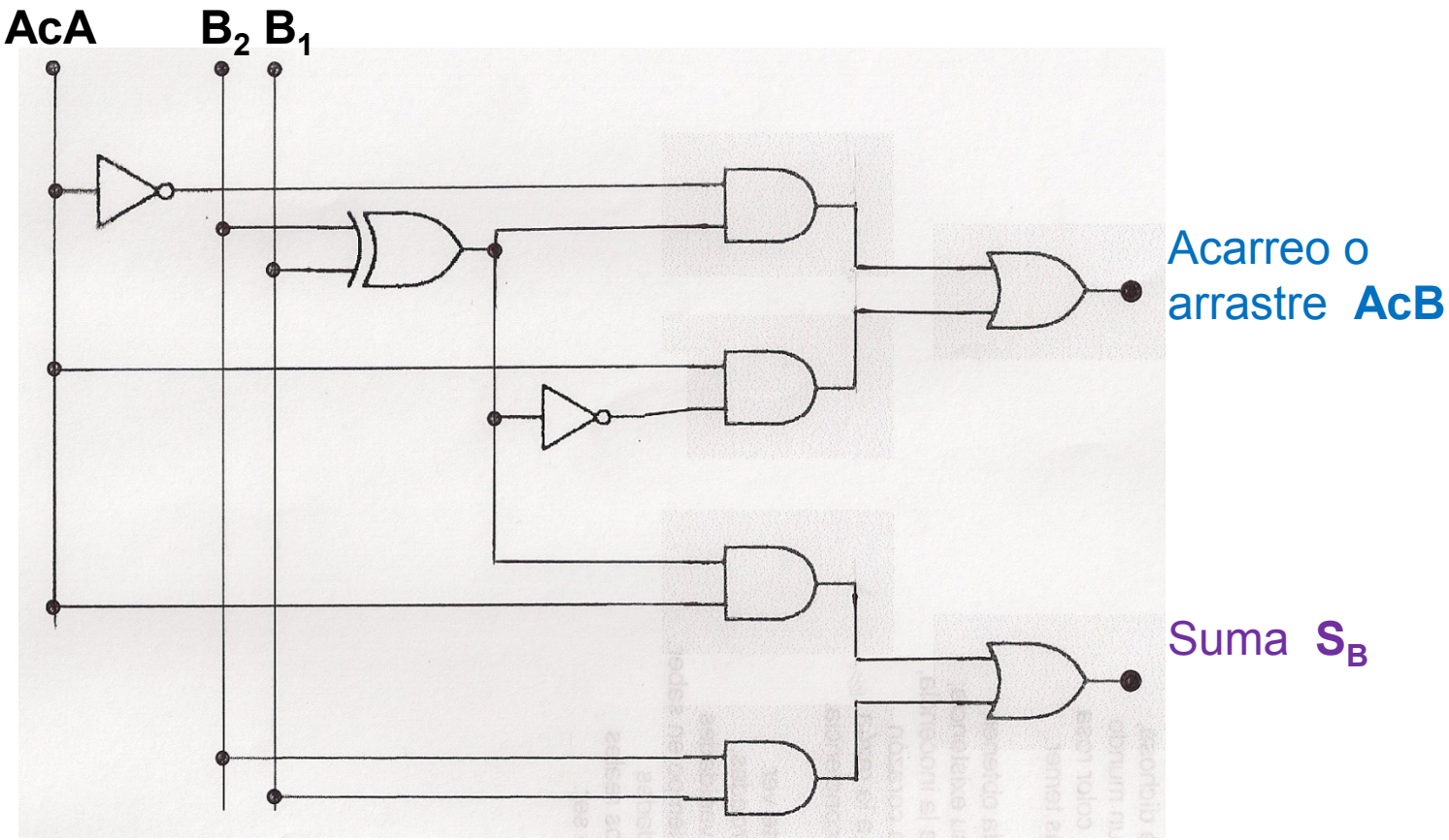
Se pueden simplificar sacando factor común $\overline{AcA}$ de 1 y 2 y luego AcA de 3 y 4. Se obtiene que:

$$S_B = \overline{AcA}\ (B_1\ \underline{O}\ B_2) + AcA\ \overline{(B_1\ \underline{O}\ B_2)}$$

Y además de 5 y 8 y de 6 y 7 :

$$AcB = B_1\ B_2 + AcA\ (B_1\ \underline{O}\ B_2)$$

• **Circuito de compuertas lógicas** debidamente simplificado. Responde a las expresiones recuadradas anteriores. El O entre las entradas B_1 y B_2, aparece en ambas ecuaciones, pero no se repite en el esquema.



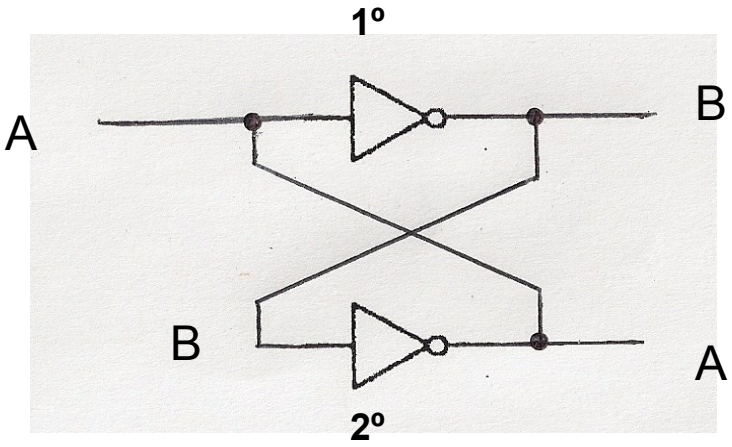
• Se demuestra que la “Resta” puede ser efectuada, en numeración binaria, mediante una suma, por supuesto que utilizando un cierto artificio de electrónica digital que no merece llamarse operación, pero donde también se usan compuertas lógicas. (Se demuestra en el apéndice).

• La Multiplicación, sabemos que por definición es una suma reiterada.

• La División también puede verse como la repetición de una resta, que a su vez se puede convertir en sumas.

• Por lo tanto, aún sin demostración, se puede afirmar que con “sumadores” y algunos “artilugios” adecuados, concebimos con relativa facilidad la idea de que, las operaciones aritméticas elementales, se realizan en los sistemas de cálculo actuales, respaldados por las ideas que aquí se han desarrollado.

• Para finalizar, es útil mostrar como se puede concebir un elemento de “memoria” de un “bit”, empleando solamente dos inversores. Se trata de “retener” un “0” ó un “1”, cuando éste se aplique durante un corto período de tiempo.



• Un “**0**” de corta duración en la entrada **A,** impone un “**1**” en **B** a través del **1º** inversor; a la vez que el **2º** inversor coloca un “**0**” permanente en **A**, aunque el valor original en la entrada haya desaparecido.

• En igual forma un “**1**” de corta duración en A, es retenido o memorizado con una consideración complementaria de la anterior.

• Existen muchas variantes de memoria, basadas en otras compuertas lógicas, pero esta es significativa por su simpleza.

• Se puede continuar con la síntesis de muchos circuitos que cumplan diferentes tareas en los sistemas de computación, pero creemos que el objetivo de fundamentar las bases “Lógico-matemáticas” de esta moderna técnica, ha sido cumplido.

• Gracias por su atención.

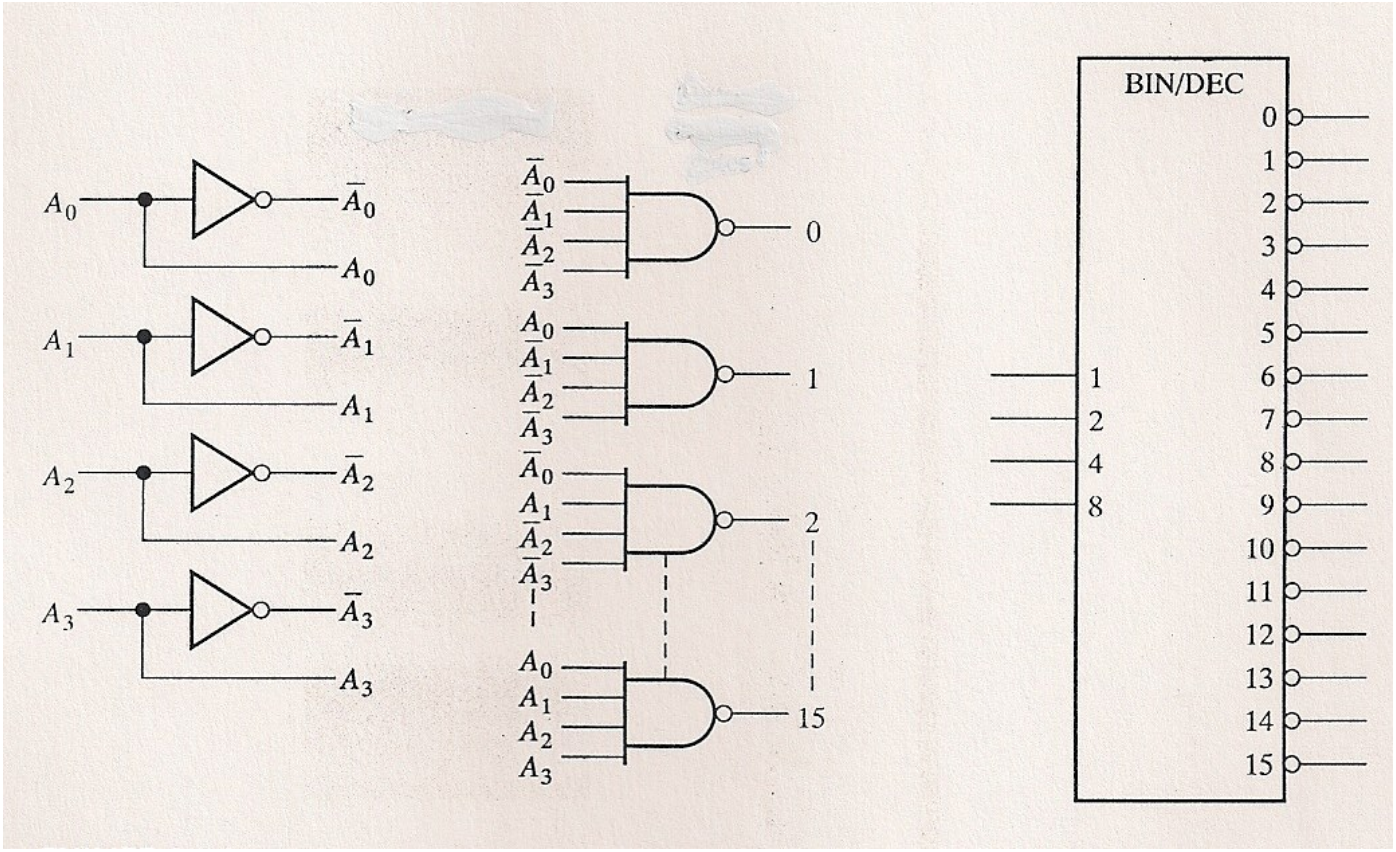
• Se sugiere comentar el tema.

F I N

- **APÉNDICES:**

1. "Decodificador o Selector de 16 Direcciones".
2. Código ASCII .
3. Familia TTL de compuertas lógicas integradas.
4. Multiplexor y Demultiplexor
5. Varios.

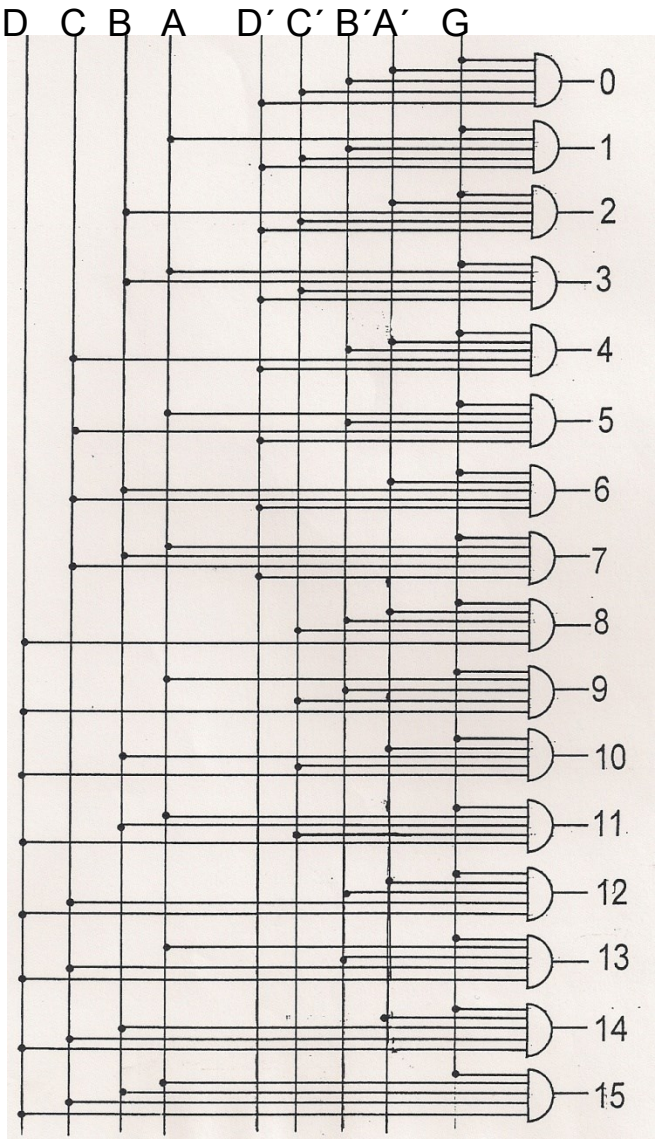
Decodificador “ Binario a Decimal o a Hexadecimal(16 salidas)” (4 : 16), tambien llamado “Selector de Direcciones”.



• Se muestra un diagrama de un “selector de 16 posiciones, “sencillo” de interpretar, en cuanto al conexionado de las compuertas “Y”, en relación con las ecuaciones del sistema.

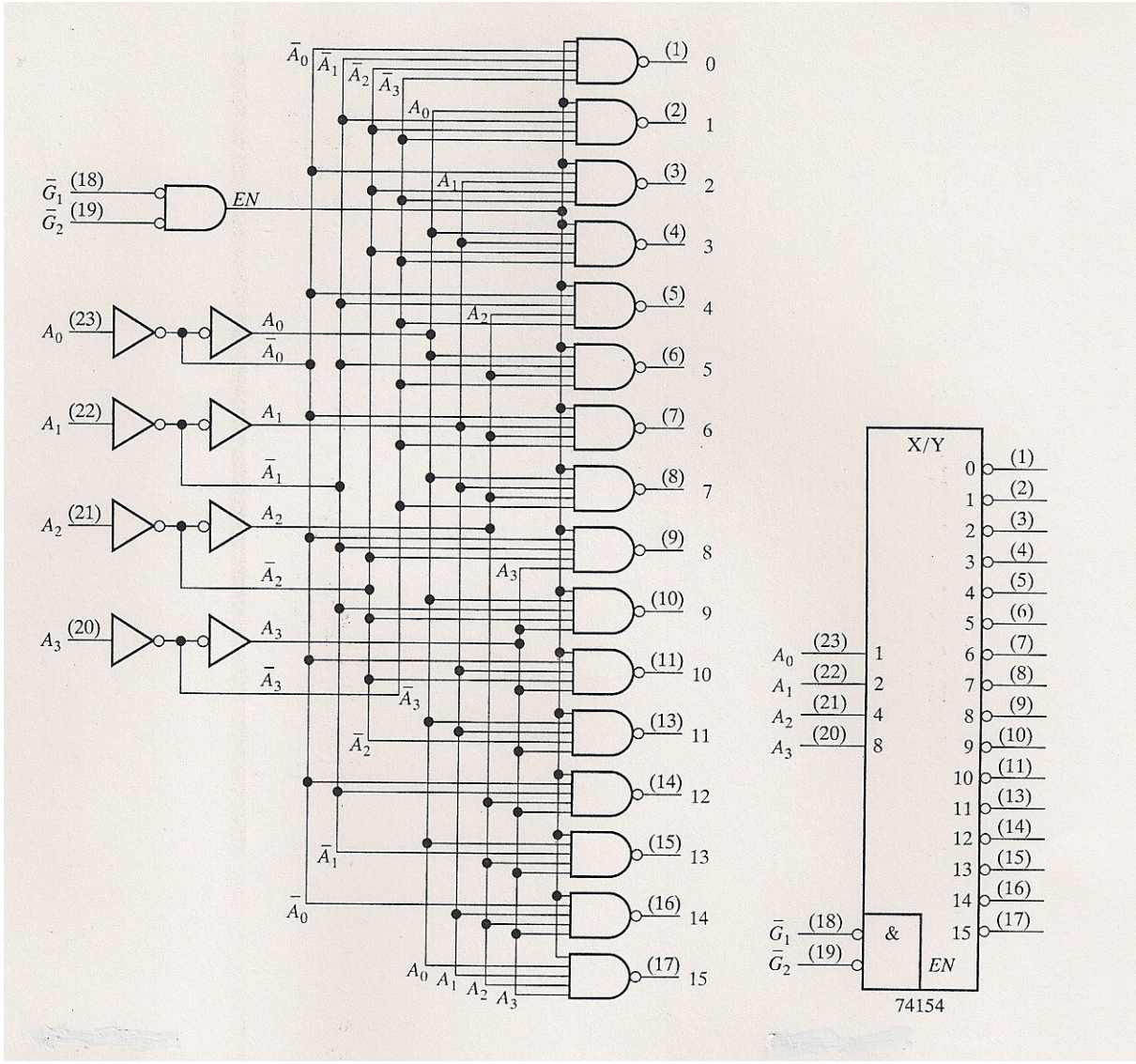
• Por supuesto que entre las entradas D, C, B y A y las barras D´, C´, B´y A´, se interponen cuatro inversores.

• La entrada auxiliar G, permite controlar a todas las salidas en forma simultánea, a los efectos de que no se tengan salidas no deseadas mientras cambian las entradas. También se puede interpretar como un “distribuidor de datos”.



Se muestra el esquema del circuito integrado 74154 (Selector de Direcciones 4:16), de uso comercial.

También se da el detalle de las interconexiones de las compuertas lógicas básicas.



Código ASCII (American Standard Code for Information Interchange)

Control Characters				Graphic Symbols											
Name	Dec	Binary	Hex	Symbol	Dec	Binary	Hex	Symbol	Dec	Binary	Hex	Symbol	Dec	Binary	Hex
NUL	0	0000000	00	space	32	0100000	20	@	64	1000000	40	`	96	1100000	60
SOH	1	0000001	01	!	33	0100001	21	A	65	1000001	41	a	97	1100001	61
STX	2	0000010	02	"	34	0100010	22	B	66	1000010	42	b	98	1100010	62
ETX	3	0000011	03	#	35	0100011	23	C	67	1000011	43	c	99	1100011	63
EOT	4	0000100	04	$	36	0100100	24	D	68	1000100	44	d	100	1100100	64
ENQ	5	0000101	05	%	37	0100101	25	E	69	1000101	45	e	101	1100101	65
ACK	6	0000110	06	&	38	0100110	26	F	70	1000110	46	f	102	1100110	66
BEL	7	0000111	07	´	39	0100111	27	G	71	1000111	47	g	103	1100111	67
BS	8	0001000	08	(	40	0101000	28	H	72	1001000	48	h	104	1101000	68
HT	9	0001001	09	)	41	0101001	29	I	73	1001001	49	i	105	1101001	69
LF	10	0001010	0A	*	42	0101010	2A	J	74	1001010	4A	j	106	1101010	6A
VT	11	0001011	0B	+	43	0101011	2B	K	75	1001011	4B	k	107	1101011	6B
FF	12	0001100	0C	,	44	0101100	2C	L	76	1001100	4C	l	108	1101100	6C
CR	13	0001101	0D	-	45	0101101	2D	M	77	1001101	4D	m	109	1101101	6D
SO	14	0001110	0E	.	46	0101110	2E	N	78	1001110	4E	n	110	1101110	6E
SI	15	0001111	0F	/	47	0101111	2F	O	79	1001111	4F	o	111	1101111	6F
DLE	16	0010000	10	0	48	0110000	30	P	80	1010000	50	p	112	1110000	70
DC1	17	0010001	11	1	49	0110001	31	Q	81	1010001	51	q	113	1110001	71
DC2	18	0010010	12	2	50	0110010	32	R	82	1010010	52	r	114	1110010	72
DC3	19	0010011	13	3	51	0110011	33	S	83	1010011	53	s	115	1110011	73
DC4	20	0010100	14	4	52	0110100	34	T	84	1010100	54	t	116	1110100	74
NAK	21	0010101	15	5	53	0110101	35	U	85	1010101	55	u	117	1110101	75
SYN	22	0010110	16	6	54	0110110	36	V	86	1010110	56	v	118	1110110	76
ETB	23	0010111	17	7	55	0110111	37	W	87	1010111	57	w	119	1110111	77
CAN	24	0011000	18	8	56	0111000	38	X	88	1011000	58	x	120	1111000	78
EM	25	0011001	19	9	57	0111001	39	Y	89	1011001	59	y	121	1111001	79
SUB	26	0011010	1A	:	58	0111010	3A	Z	90	1011010	5A	z	122	1111010	7A
ESC	27	0011011	1B	;	59	0111011	3B	[	91	1011011	5B	{	123	1111011	7B
FS	28	0011100	1C	<	60	0111100	3C	\	92	1011100	5C	|	124	1111100	7C
GS	29	0011101	1D	=	61	0111101	3D	]	93	1011101	5D	}	125	1111101	7D
RS	30	0011110	1E	>	62	0111110	3E	^	94	1011110	5E	~	126	1111110	7E
US	31	0011111	1F	?	63	0111111	3F	_	95	1011111	5F	Del	127	1111111	7F

Caracteres del ASCII extendido

Symbol	Dec	Hex	Symbol	Dec	Hex	Symbol	Dec	Hex	Symbol	Dec	Hex
Ç	128	80	á	160	A0	└	192	C0	α	224	E0
ü	129	81	í	161	A1	┴	193	C1	β	225	E1
é	130	82	ó	162	A2	┬	194	C2	Γ	226	E2
â	131	83	ú	163	A3	├	195	C3	π	227	E3
ä	132	84	ñ	164	A4	─	196	C4	Σ	228	E4
à	133	85	Ñ	165	A5	┼	197	C5	σ	229	E5
å	134	86	ª	166	A6	╞	198	C6	µ	230	E6
ç	135	87	º	167	A7	╟	199	C7	τ	231	E7
ê	136	88	¿	168	A8	╚	200	C8	Φ	232	E8
ë	137	89	⌐	169	A9	╔	201	C9	Θ	233	E9
è	138	8A	¬	170	AA	╩	202	CA	Ω	234	EA
ï	139	8B	½	171	AB	╦	203	CB	δ	235	EB
î	140	8C	¼	172	AC	╠	204	CC	∞	236	EC
ì	141	8D	¡	173	AD	═	205	CD	φ	237	ED
Ä	142	8E	«	174	AE	╬	206	CE	ε	238	EE
Å	143	8F	»	175	AF	╧	207	CF	∩	239	EF
É	144	90	▒	176	B0	╨	208	D0	≡	240	F0
æ	145	91	▒	177	B1	╤	209	D1	±	241	F1
Æ	146	92	▓	178	B2	╥	210	D2	≥	242	F2
ô	147	93	│	179	B3	╙	211	D3	≤	243	F3
ö	148	94	┤	180	B4	╘	212	D4	⌠	244	F4
ò	149	95	╡	181	B5	╒	213	D5	⌡	245	F5
û	150	96	╢	182	B6	╓	214	D6	÷	246	F6
ù	151	97	╖	183	B7	╫	215	D7	≈	247	F7
ÿ	152	98	╕	184	B8	╪	216	D8	°	248	F8
Ö	153	99	╣	185	B9	┘	217	D9	∙	249	F9
Ü	154	9A	║	186	BA	┌	218	DA	·	250	FA
¢	155	9B	╗	187	BB	▉	219	DB	√	251	FB
£	156	9C	╝	188	BC	▊	220	DC	η	252	FC
¥	157	9D	╜	189	BD	▋	221	DD	²	253	FD
Pₜ	158	9E	╛	190	BE	▌	222	DE	■	254	FE
ƒ	159	9F	┐	191	BF	▍	223	DF		255	FF

• **Familias de Compuertas lógicas Integradas.**

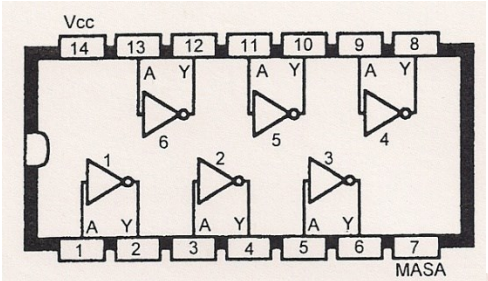
• En la segunda mitad de la década de los años 60 del siglo pasado, fueron desarrollados, con la técnica del circuito integrado, algunas familias de componentes electrónicos que presentan conjuntos de compuertas lógicas básicas, agrupadas de forma conveniente.

• Este es un tema muy amplio y especializado, pero como para tener una referencia, nombraremos a una familia llamada “TTL” (Transistor-Transistor-Logic), considerada muy representativa.

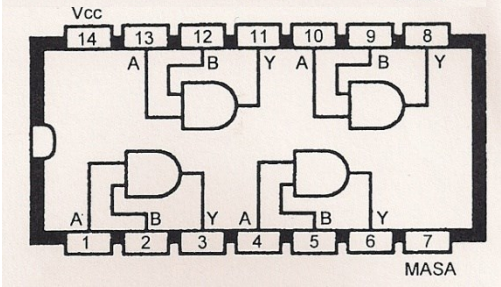
• Actualmente, aunque con características eléctricas muy mejoradas respecto de las primitivas, se siguen empleando en forma normal.

• Se presentan en cápsulas de plástico o también de cerámica y sus medidas son del orden de 20 mm de largo por 8 mm de ancho y solo 3,5 mm de espesor, con una separación entre patitas de conexión de 2,5 mm

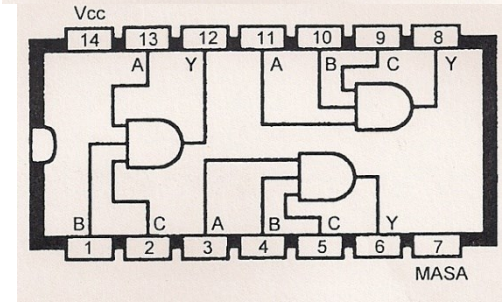
Diagramas funcionales de algunos circuitos integrados comerciales, de uso normal y que pertenecen a la familia TTL.



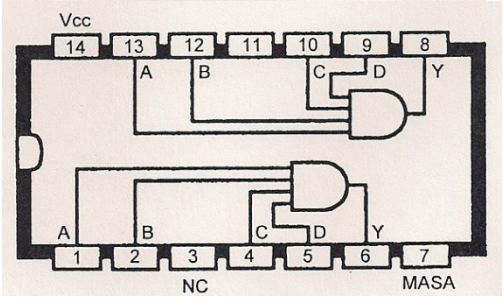
74LS04 - Seis Inversores.



74LS08 - Cuatro compuertas Y (AND) de dos entradas.



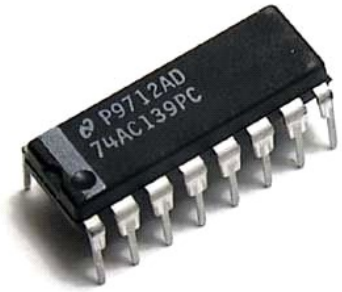
74LS11 - Tres compuertas Y (AND) de tres entradas.

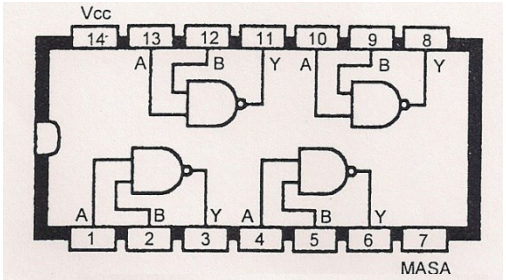


74LS21 - Dos compuertas Y (AND) de cuatro entradas.

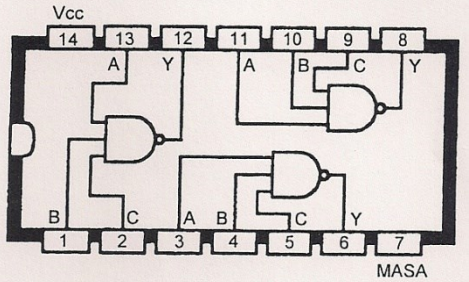
Diagramas funcionales de algunos circuitos integrados comerciales, de uso normal y que pertenecen a la familia TTL.

2

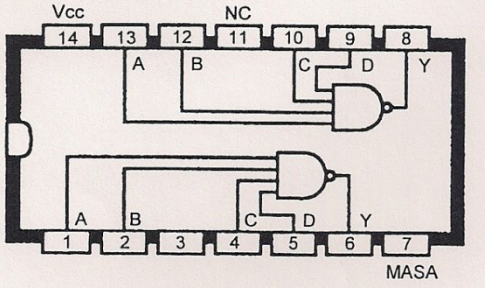




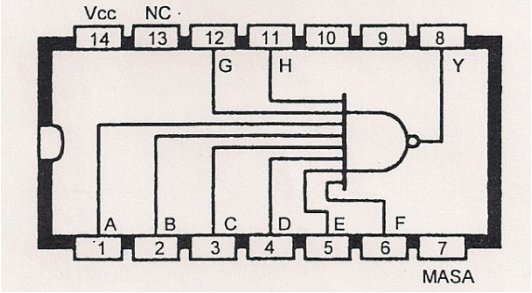
74LS00 - Cuatro compuertas NOY (NAND) de dos entradas.



74LS10 - Tres compuertas NOY (NAND) de tres entradas.



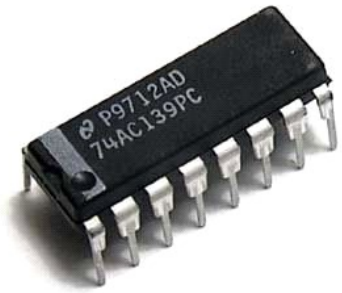
74LS20 - Dos compuertas NOY (NAND) de cuatro entradas.

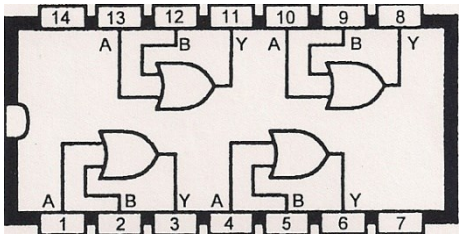


74LS30 – Una compuerta NOY (NAND) de ocho entradas.

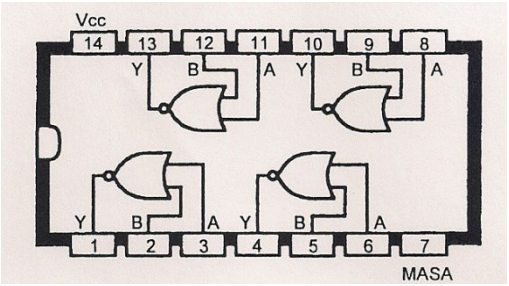
Diagramas funcionales de algunos circuitos integrados comerciales, de uso normal y que pertenecen a la familia TTL.

3

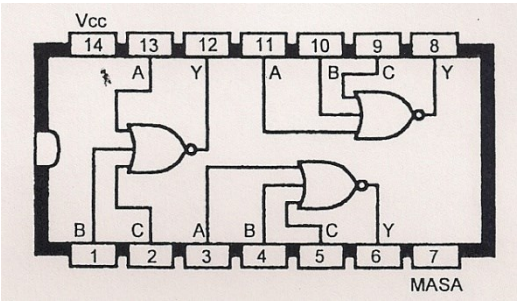




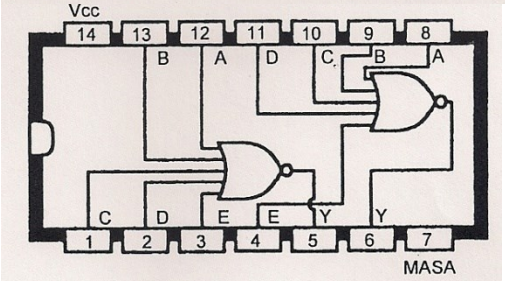
74LS32 - Cuatro compuertas OR (OR) de dos entradas.



74LS02 - Cuatro compuertas NOO (NOR) de dos entradas.

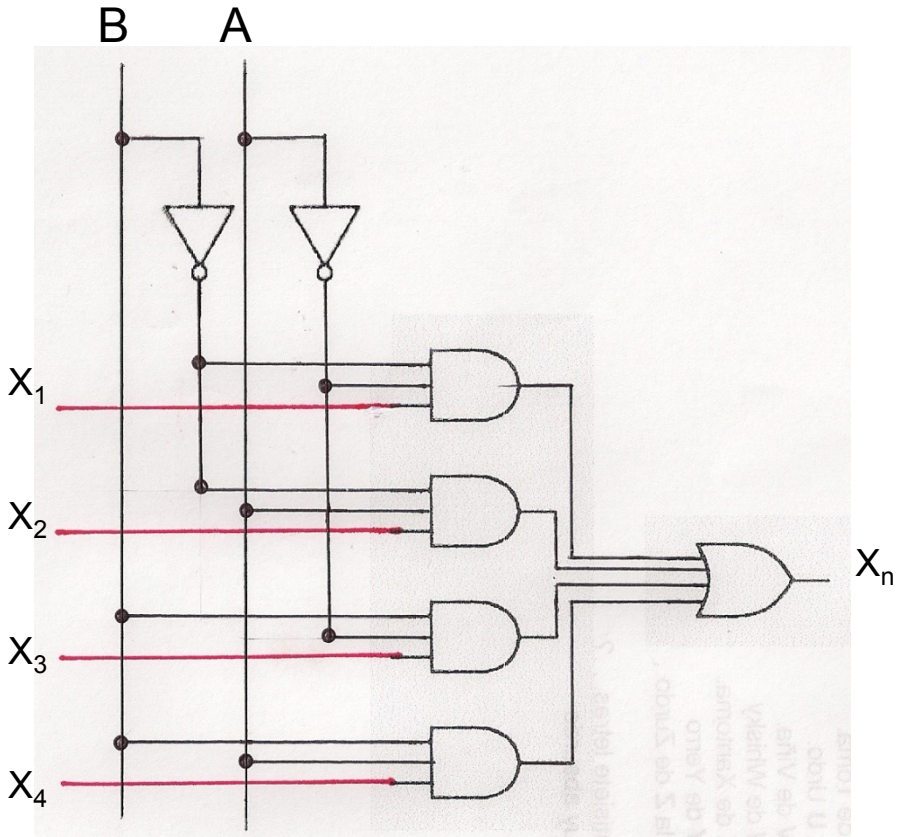


74LS27 - Dos compuertas NOO (NOR) de tres entradas.

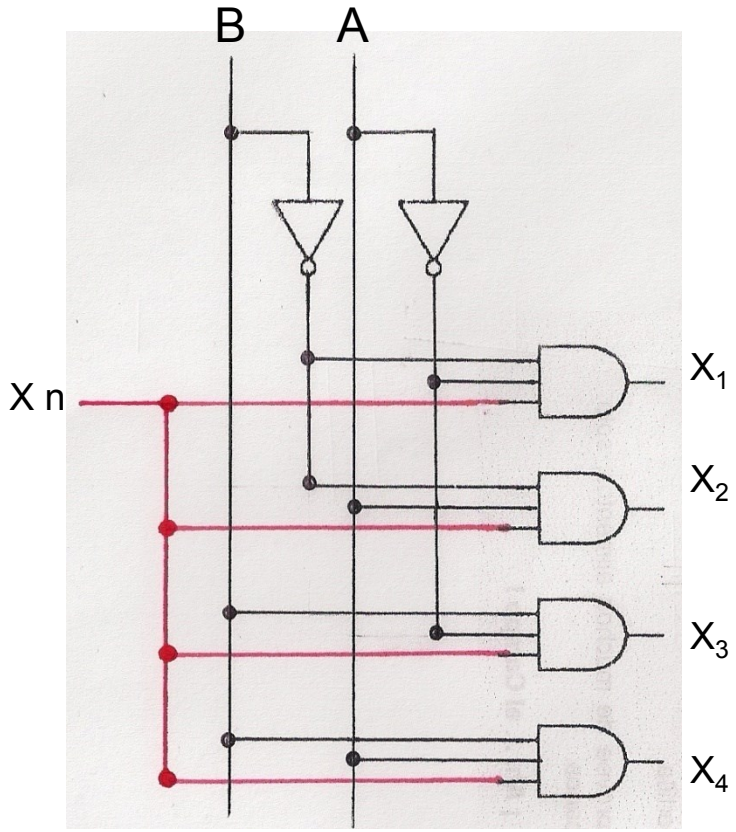


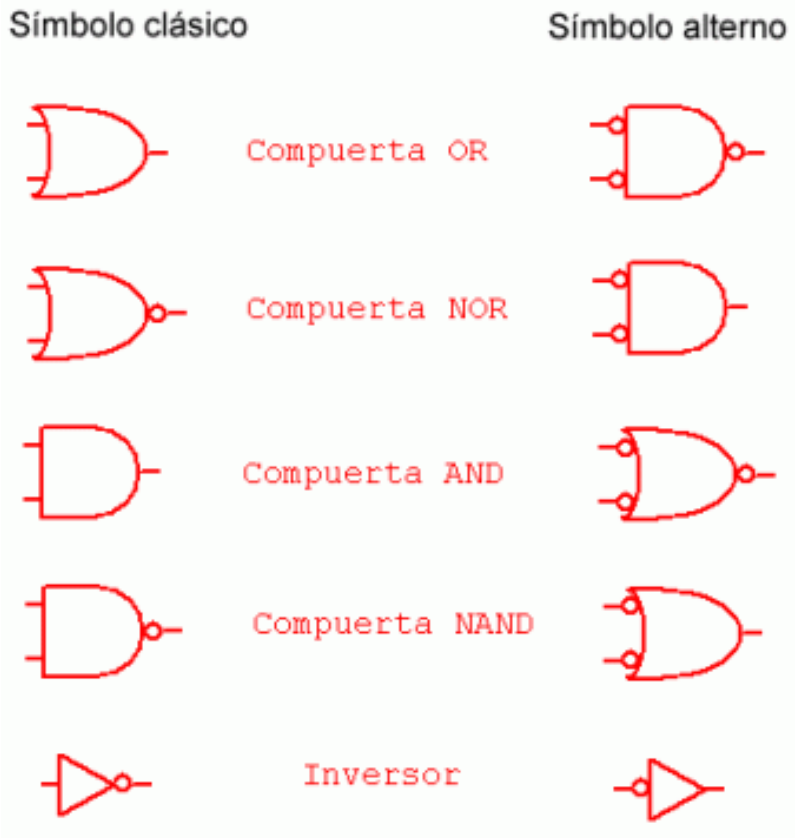
74LS30 – Dos compuertas NOO (NOR) de cinco entradas.

• Selector de Datos de Cuatro canales - (Multiplexor)



• Distribuidor de Datos de cuatro canales - (Demultiplexor).




Símbolo clásico
Símbolo alterno
Compuerta OR
Compuerta NOR
Compuerta AND
Compuerta NAND
Inversor

Algunas de las Leyes Básicas del Álgebra de Boole.

Leyes de absorción:	$A+(A.B) = A$
	$A.(A+B) = A$
Leyes de anulación:	$A+1 = 1$
	$A.0 = 0$
Leyes asociativas:	$(A+B)+C = A+(B+C) = A+B+C$
	$(A.B).C = A.(B.C) = A.B.C$
Leyes conmutativas:	$A+B = B+A$
	$A.B = B.A$
Leyes de complementación:	$A+\overline{A} = 1$
	$A.\overline{A} = 0$
Leyes distributivas:	$A.(B+C) = (A.B)+(A.C)$
	$A+(B.C) = (A+B).(A+C)$
	$(A+B).(A+C).(A+D) = A+(B.C.D)$
Doble negación:	$\text{not}\,\overline{A} = \overline{\overline{A}} = A$
Leyes de expansión:	$(A+B).(A+B) = A$
	$(A.B)+(A.B) = A$
Igualdad:	$A+0 = A$
	$A.1 = A$
1ª Regla De Morgan:	$\overline{A+B} = \overline{A}.\overline{B}$
2ª Regla De Morgan:	$\overline{A.B} = \overline{A}+\overline{B}$
Leyes tautológicas:	$A.A = A$
	$A+A = A$