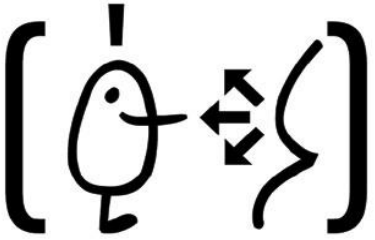
Juan Manuel Orduña Huertas

Fundamentos de computadores II

VNIVERSITAT DE VALÈNCIA

Escola Tècnica Superior d'Enginyeria ETSE-UV



VICERECTORAT DE PARTICIPACIÓ I PROJECCIÓ TERRITORIAL

Universitat i Societat

CULLERA

Objetivo

- ¿Cómo es posible que los computadores hagan todo tipo de cálculos? ¿Son tan listos como nosotros?

- ¿Por qué funcionan sólo con ceros y unos? ¿Cómo pueden hacer todo lo que hacen si sólo tienen ceros y unos?

- ¿Cómo funciona un computador (y todos los aparatos «digitales» de hoy en día) ?

Fundamentos de computadores

1. Sistemas de numeración
2. **Circuitos combinacionales y secuenciales**
3. Estructura de computadores

Sección 2. Circuitos combinacionales y secuenciales

1. Álgebra de Boole
2. Circuitos combinacionales
3. Circuitos secuenciales

2.- Álgebra de Boole

Las matemáticas básicas necesarias para el estudio de la Electrónica Digital las forman el *Álgebra de Boole.*

Fue desarrollada por George Boole en 1847 y utilizada para resolver problemas de lógica matemática.

Claude Shannon aplicó el álgebra booleana al diseño de circuitos de conmutación en 1939.

2.- Álgebra de Boole

Un ***Álgebra de Boole*** ($\mathcal{B}$) es una estructura algebraica formada por un conjunto de elementos que pueden tomar dos valores perfectamente diferenciados (que llamaremos 0 lógico y 1 lógico), junto con dos operadores binarios, **+** (suma lógica) y • (producto lógico), y que cumplen los siguientes postulados:

1. Ambas operaciones son **conmutativas**

$$\forall A, B \in \mathcal{B} \begin{cases} A + B = B + A \\ A \bullet B = B \bullet A \end{cases}$$

2. Elementos **neutros**

$$\forall A \in \mathcal{B} \begin{cases} A + 0 = A \\ A \bullet 1 = A \end{cases}$$

3. Elementos **complementados**

$$\forall A \in \mathcal{B} \;\; \exists \;\; B \in \mathcal{B} \begin{cases} A + B = 1 \\ A \bullet B = 0 \end{cases}$$

Al elemento B se le denomina **complemento** del elemento A, y se representa como $\bar{A}$.

4. Propiedades **distributivas**

$$\forall A, B, C \in \mathcal{B} \begin{cases} A \bullet (B + C) = (A \bullet B) + (A \bullet C) \\ A + (B \bullet C) = (A + B) \bullet (A + C) \end{cases}$$

2.- Álgebra de Boole

Propiedades del Álgebra de Boole:

1. Principio de **dualidad**.

2. Elementos **complementados.**

$$A \in \mathcal{B} \begin{cases} \text{Si } A = 0 \Rightarrow \overline{A} = 1 \\ \text{Si } A = 1 \Rightarrow \overline{A} = 0 \end{cases}$$

3. Operaciones con **0 y 1**.

$$\forall A \in \mathcal{B} \quad \begin{cases} A + 1 = 1 \\ A \bullet 0 = 0 \end{cases}$$

4. Leyes de **idempotencia**.

$$\forall A \in \mathcal{B} \quad \begin{cases} A + A = A \\ A \bullet A = A \end{cases}$$

5. Ley de **equivalencia**.

$$\overline{\overline{A}} = A$$

6. **Asociatividad**.

$$\forall A, B, C \in \mathcal{B} \begin{cases} A + (B + C) = (A + B) + C \\ A \bullet (B \bullet C) = (A \bullet B) \bullet C \end{cases}$$

7. Leyes de **absorción.**

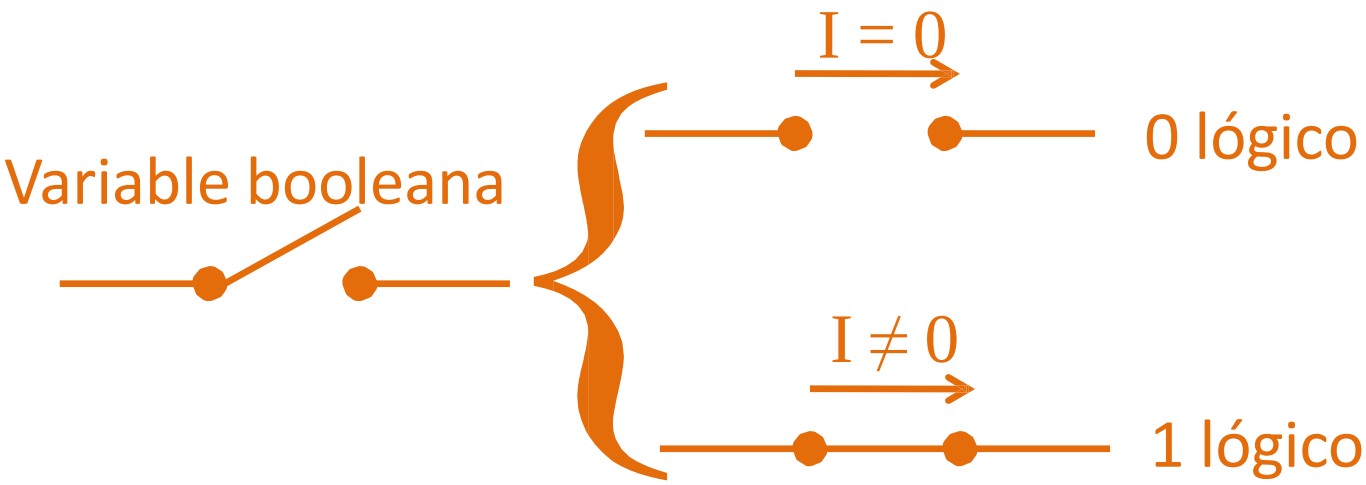
$$\forall A, B \in \mathcal{B} \begin{cases} A + A \bullet B = A \\ A \bullet (A + B) = A \end{cases}$$

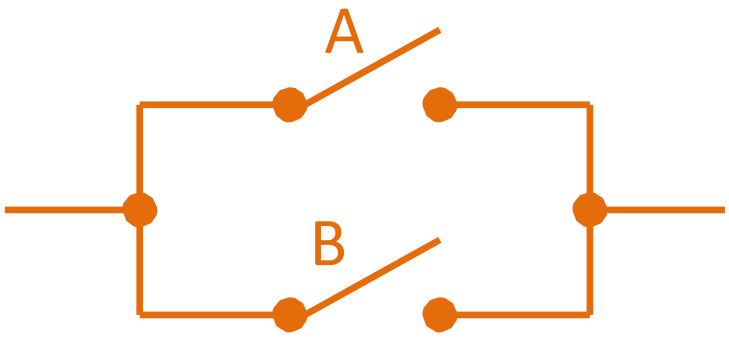
8. Leyes de **De Morgan.**

$$A, B, C, D, \ldots \in \mathcal{B} \begin{cases} \overline{A + B + C + D + \ldots} = \overline{A} \bullet \overline{B} \bullet \overline{C} \bullet \overline{D} \bullet \ldots \\ \overline{A \bullet B \bullet C \bullet D \bullet \ldots} = \overline{A} + \overline{B} + \overline{C} + \overline{D} + \ldots \end{cases}$$

2.- Álgebra de Boole

Ejemplo de Álgebra de Boole: Conmutadores





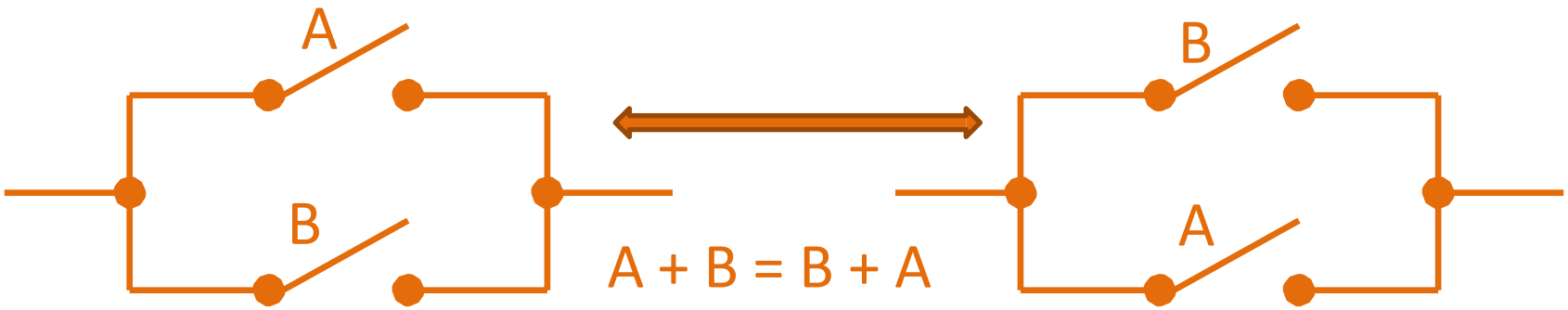
Suma lógica: A + B

Producto lógico: A · B

2.- Álgebra de Boole

Ejemplo de Álgebra de Boole: Conmutadores

Propiedad conmutativa



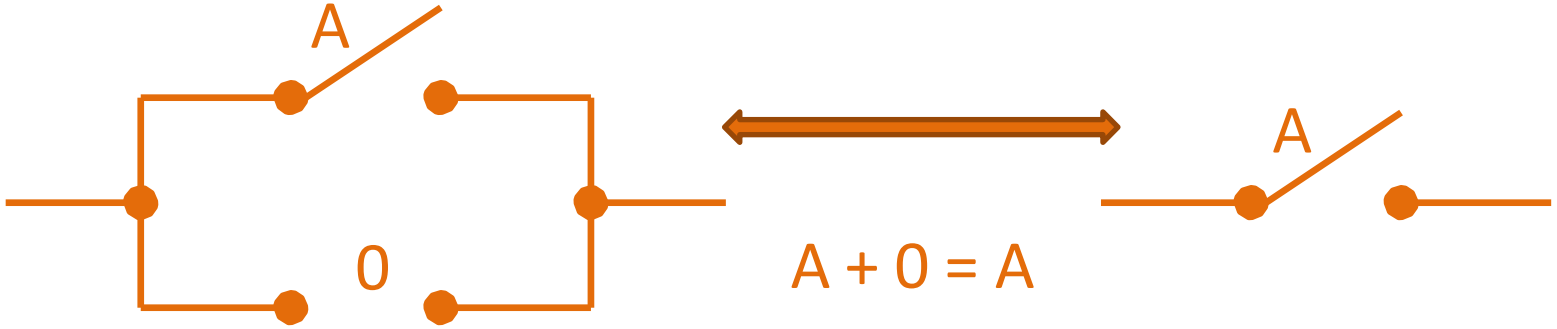
$A + B = B + A$

$A \cdot B = B \cdot A$

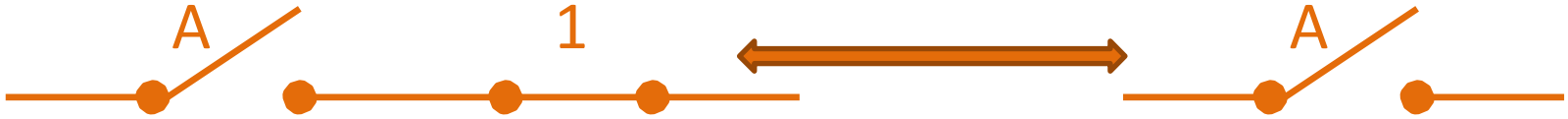
2.- Álgebra de Boole

Ejemplo de Álgebra de Boole: Conmutadores

Elementos neutros



$A + 0 = A$

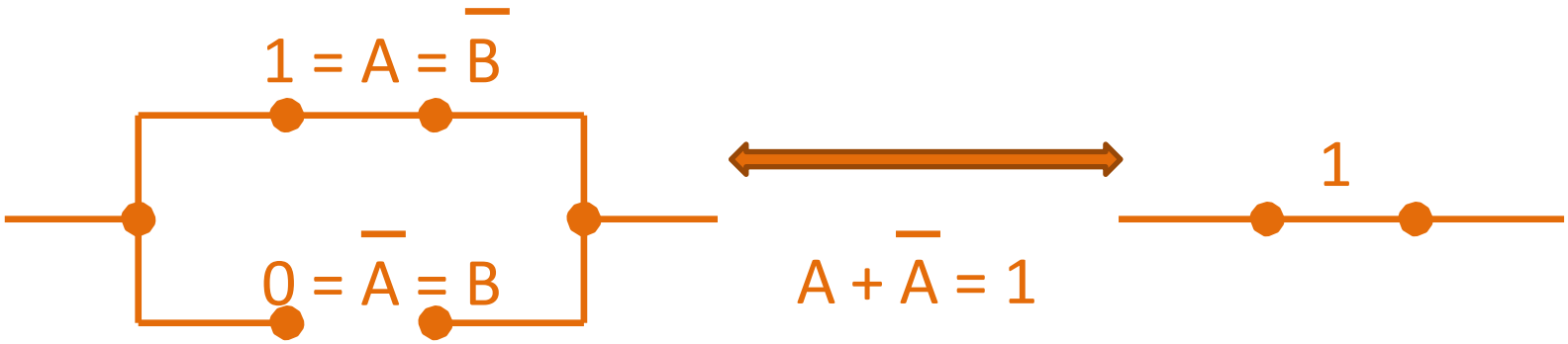


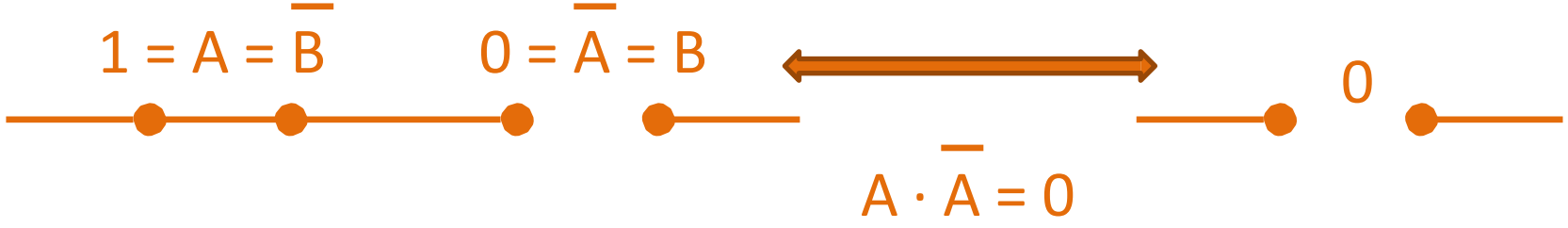
$A \cdot 1 = A$

2.- Álgebra de Boole

Ejemplo de Álgebra de Boole: Conmutadores

Elementos complementados

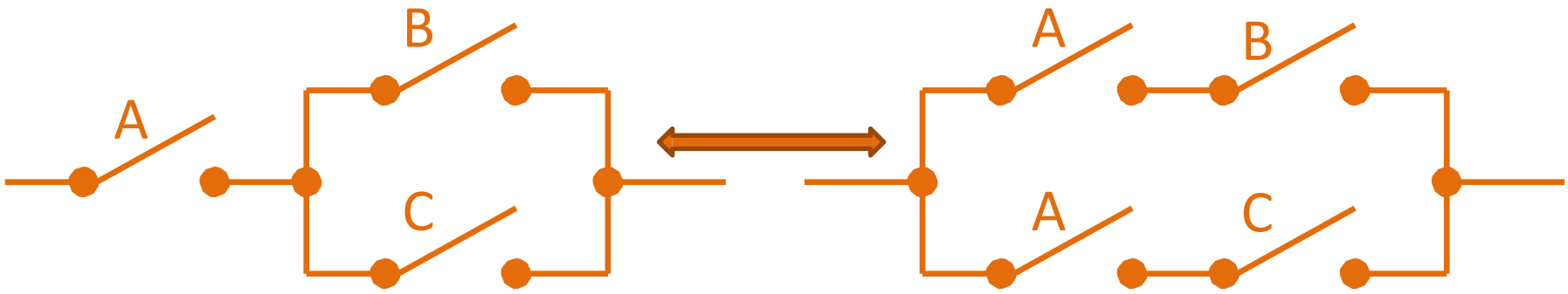




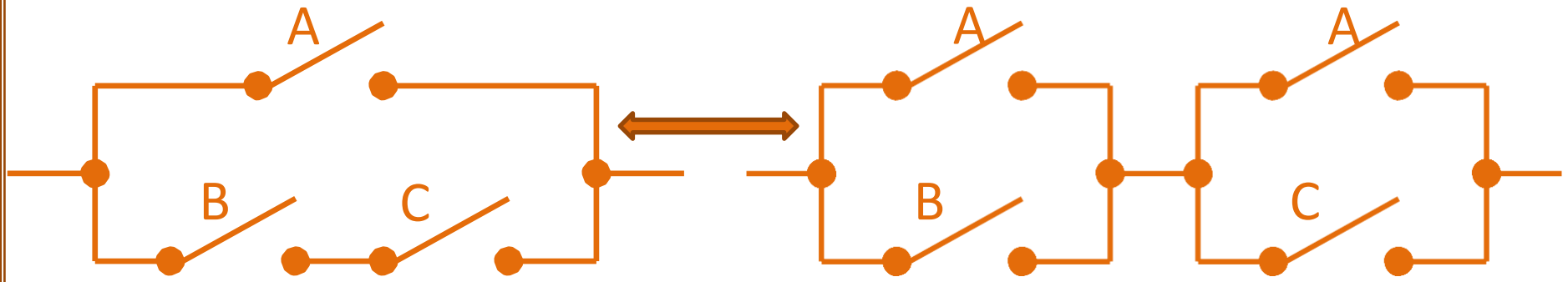
2.- Álgebra de Boole

Ejemplo de Álgebra de Boole: Conmutadores

Propiedad distributiva



$$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$$



$$A + (B \cdot C) = (A + B) \cdot (A + C)$$

2.- Álgebra de Boole

La Electrónica Digital y el Álgebra de Boole

Variable Digital ↔ Variable del Álgebra de Boole

¿Cómo implementamos la suma y el producto lógico en Electrónica Digital?

Mediante *Puertas Lógicas*

Puertas Lógicas: Circuitos electrónicos que implementan los operadores del Álgebra de Boole.

Trataremos los problemas de Electrónica Digital como problemas del Álgebra de Boole

3.- Funciones Lógicas

Función Lógica: Función en la que las variables independientes son variables lógicas, y el valor de la función o variable dependiente también es una variable lógica.

$$Z = f(A,B,C,\ldots,N)$$

donde Z,A,B,C,…,N son variables lógicas

Estudiaremos:
- Funciones lógicas de 1 variable
- Funciones lógicas de 2 variables
- Funciones lógicas de más variables

3.- Funciones Lógicas

Funciones lógicas de 1 variable

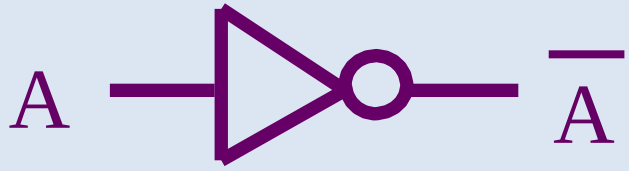
A	$f_0(A)$	$f_1(A)$	$f_2(A)$	$f_3(A)$
0	0	0	1	1
1	0	1	0	1

Tablas de verdad

- $f_0(A) = 0$ (función 0 lógico constante)
- $f_1(A) = A$ (función igual a la variable)
- $f_2(A) = \overline{A}$ (función complementación)
- $f_3(A) =$ (función 1 lógico constante)

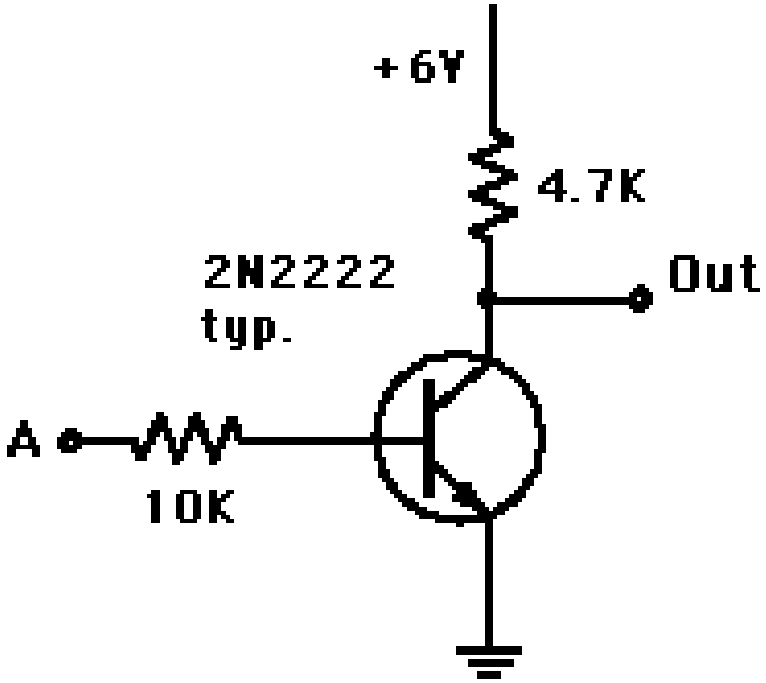
La función más interesante es la complementación.

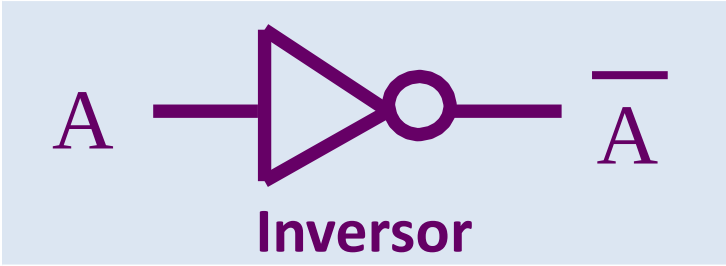
La puerta lógica que implementa esta función lógica se llama **inversor**.

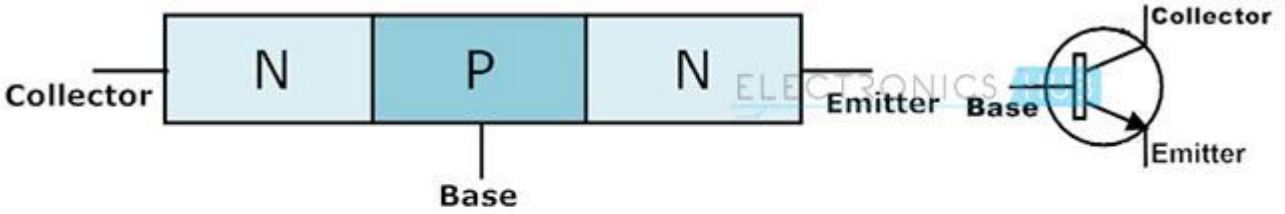


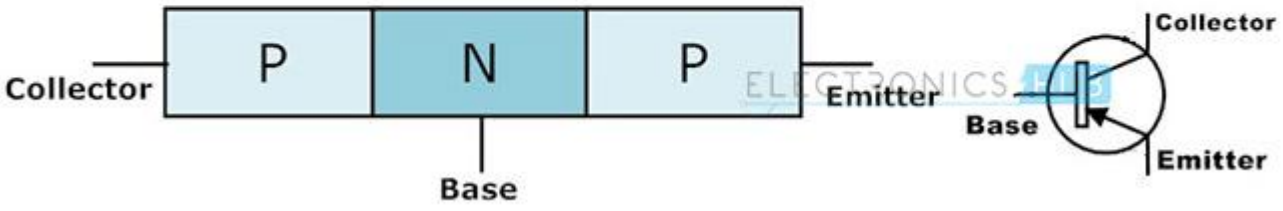
Inversor

Implementación física









3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Si tenemos *n* variables de entrada o independientes:

- 2^n estados de entrada
- 2^{2^n} funciones lógicas

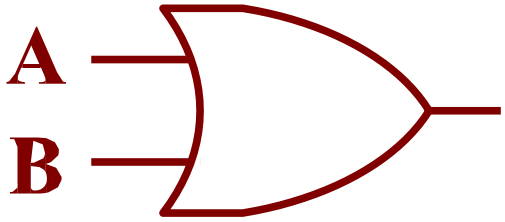
- $f_0(A,B) = 0$
- $f_{15}(A,B) = 1$
- $f_3(A,B) = A$
- $f_5(A,B) = B$
- $f_{10}(A,B) = \overline{B}$
- $f_{12}(A,B) = \overline{A}$

3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_7
0	0	0
0	1	1
1	0	1
1	1	1

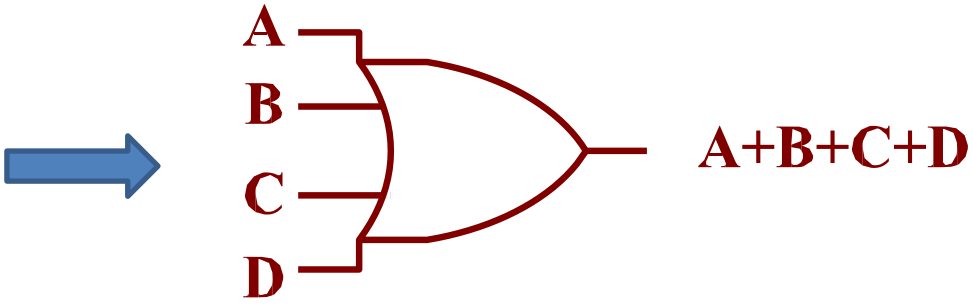
Función **OR** o **suma lógica** (**+**)



Puerta OR

Propiedades:

- Es conmutativa
- Es asociativa

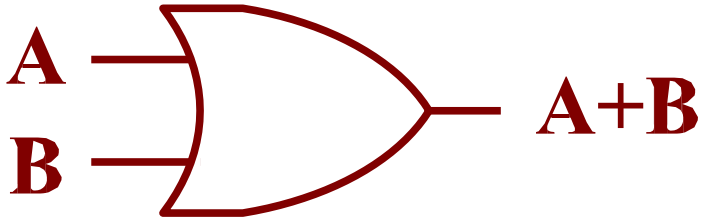


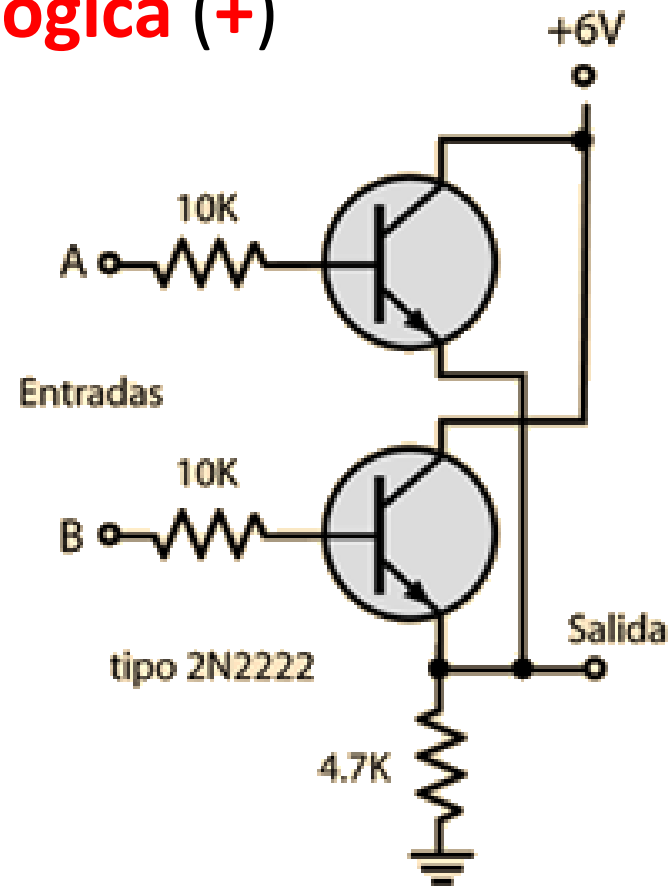
3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_7
0	0	0
0	1	1
1	0	1
1	1	1

Función **OR** o **suma lógica** (+)



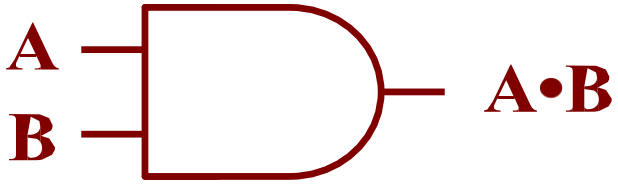


3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_1
0	0	0
0	1	0
1	0	0
1	1	1

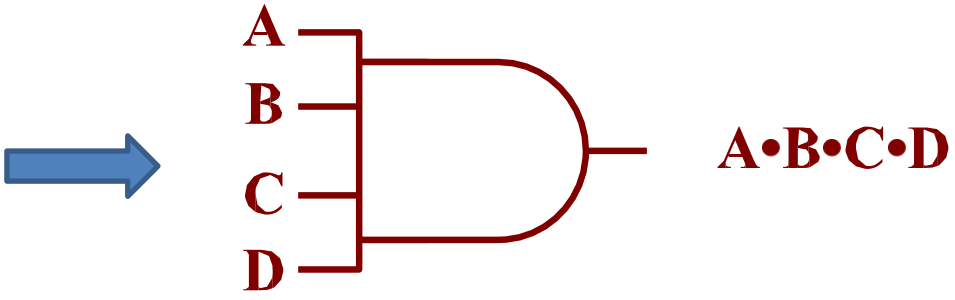
Función AND o **producto lógico** (•)



Puerta AND

Propiedades:

- Es conmutativa
- Es asociativa

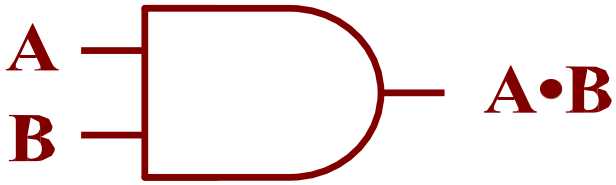


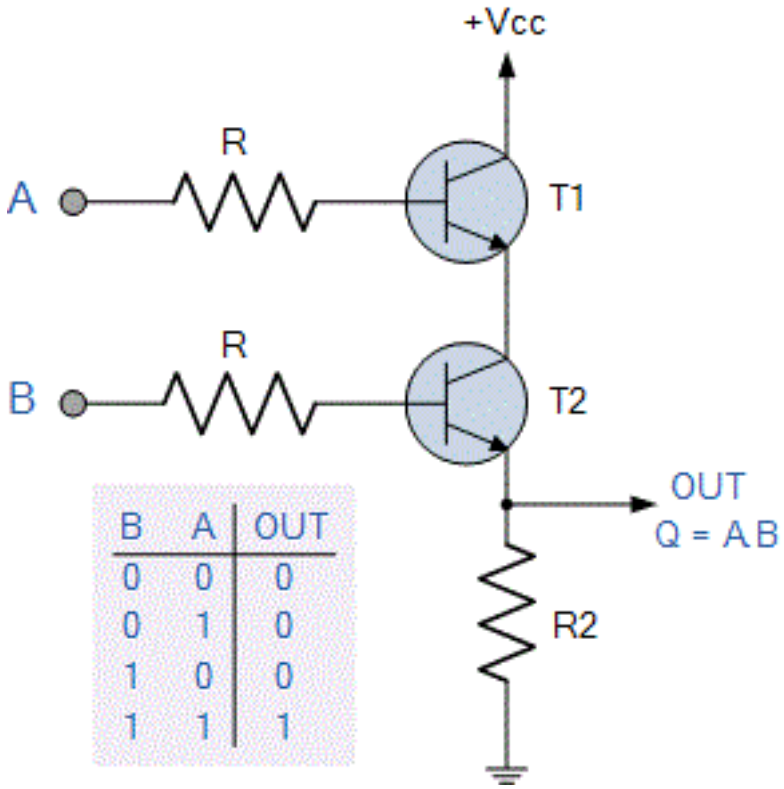
3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_1
0	0	0
0	1	0
1	0	0
1	1	1

Función AND o **producto lógico** (•)





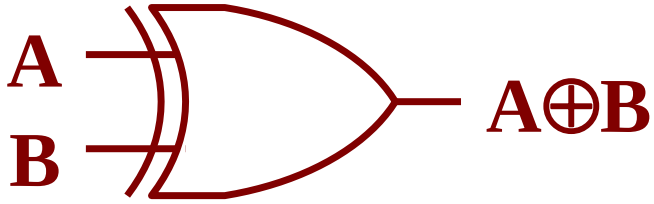
B	A	OUT
0	0	0
0	1	0
1	0	0
1	1	1

3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_6
0	0	0
0	1	1
1	0	1
1	1	0

Función X-OR o **suma exclusiva** (⊕)



Puerta X-OR

Propiedades:

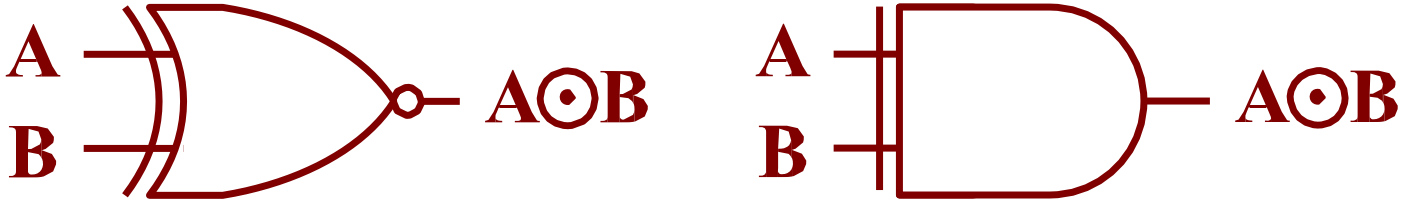
- Es conmutativa y asociativa
- $A(B \oplus C) = AB \oplus AC$
- $A \oplus B = \overline{A} \oplus B = A \oplus \overline{B} = \overline{A \oplus B}$
- La función X-OR de n variables toma el valor 1 lógico si un número impar de entradas valen 1, y 0 lógico en caso contrario

3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_9
0	0	1
0	1	0
1	0	0
1	1	1

Función X-NOR ($\odot$)



Propiedades: **Puertas X-NOR**

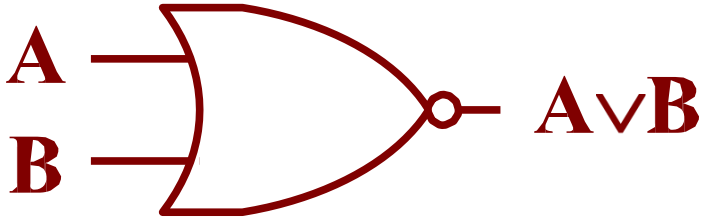
- Es conmutativa y asociativa
- $A+(B \odot C)=(A+B) \odot (A+C)$
- $A \odot B=\overline{A \oplus B}=A \oplus \overline{B}=\overline{A} \oplus B$
- La función X-NOR de n variables vale 1 lógico si un par de entradas valen 0, y 0 lógico en caso contrario

3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_8
0	0	1
0	1	0
1	0	0
1	1	0

Función **NOR** ($\vee$)



Puerta NOR

Propiedades:

- Es conmutativa
- **No es asociativa**

$$A \vee (B \vee C) \neq (A \vee B) \vee C$$

¿Cómo aplicamos esta función a más de 2 variables?

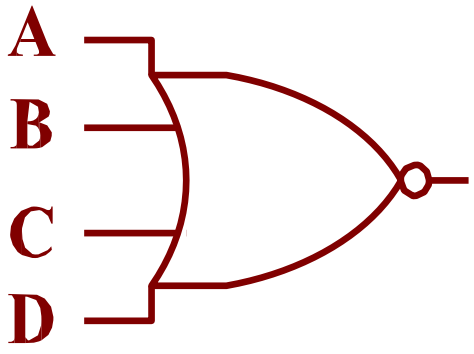
3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_8
0	0	1
0	1	0
1	0	0
1	1	0

Función **NOR** ($\vee$)

- Cuando se aplica esta función a más variables tiene otro significado: primero se suman todas las variables, y luego se complementa.
- Con esta definición, sí es asociativa.



$\overline{A+B+C+D}$

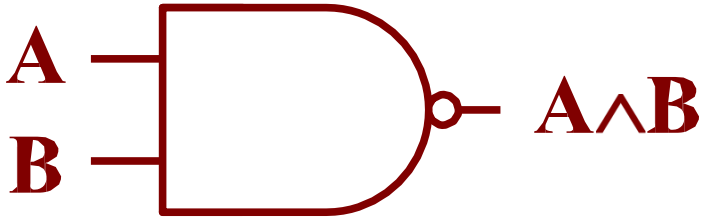
Puerta NOR

3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_{14}
0	0	1
0	1	1
1	0	1
1	1	0

Función **NAND** ($\uparrow$)



Puerta NAND

Propiedades:

- Es conmutativa
- **No es asociativa**

$$A \uparrow (B \uparrow C) \neq (A \uparrow B) \uparrow C$$

¿Cómo aplicamos esta función a más de 2 variables?

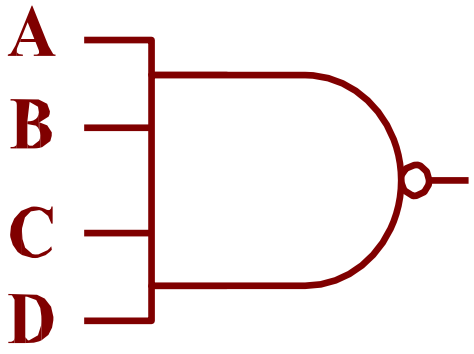
3.- Funciones Lógicas

Funciones lógicas de 2 variables

A	B	f_{14}
0	0	1
0	1	1
1	0	1
1	1	0

Función **NAND** ($\uparrow$)

- Cuando se aplica esta función a más variables tiene otro significado: primero se multiplican todas las variables, y luego se complementa.
- Con esta definición, sí es asociativa.



$\overline{A \cdot B \cdot C \cdot D}$

Puerta NAND

Simuladores web de funciones Lógicas

Logic.ly

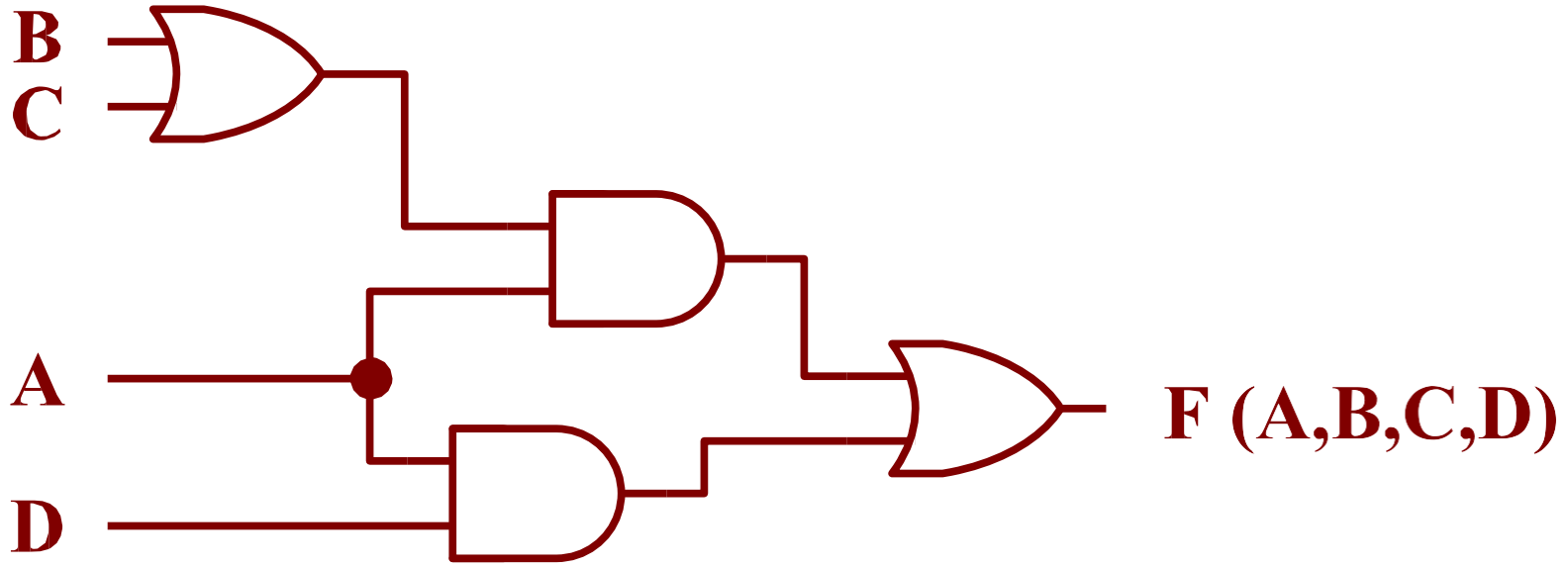
https://logic.ly/demo/

3.- Funciones Lógicas

Funciones lógicas de más de 2 variables

En lugar de definir nuevas funciones lógicas de 3 o más variables, utilizaremos las estudiadas hasta aquí aplicadas a un número mayor de variables.

Ejemplo: F(A,B,C,D)=A(B+C)+AD



3.- Funciones Lógicas

Funciones lógicas

Las funciones lógicas estudiadas no son independientes, si no que se relacionan entre sí. Todas la funciones lógicas pueden expresarse utilizando únicamente la suma lógica, el producto lógico y la función complementación.

- $F_0 = 0 = A\,\overline{A}$
- $F_1 = A\,B$
- $F_2 = \overline{A \supset B} = A\,\overline{B}$
- $F_3 = \overline{A}$
- $F_4 = B \supset A = \overline{A}\,B$
- $F_5 = B$
- $F_6 = A \oplus B = \overline{A}B + A\overline{B}$
- $F_7 = A + B$
- $F_8 = \overline{A \vee B} = \overline{A+B}$
- $F_9 = A \odot B = AB + \overline{A}\,\overline{B}$
- $F_{10} = \overline{B}$
- $F_{11} = B \supset A = A + \overline{B}$
- $F_{12} = \overline{A}$
- $F_{13} = A \supset B = \overline{A} + B$
- $F_{14} = A \wedge B = \overline{AB}$
- $F_{15} = 1 = A + A$

Ejercicios F. Lógicas

SÍNTESIS DE CIRCUITOS

Utilizando sólo puertas AND, OR e inversores, dibujar el circuito digital que realice las siguientes funciones:

1- F(A,B,C,D)=AB+B(C+D)

2- F(A,B,C) = A'BC+A(B'C)

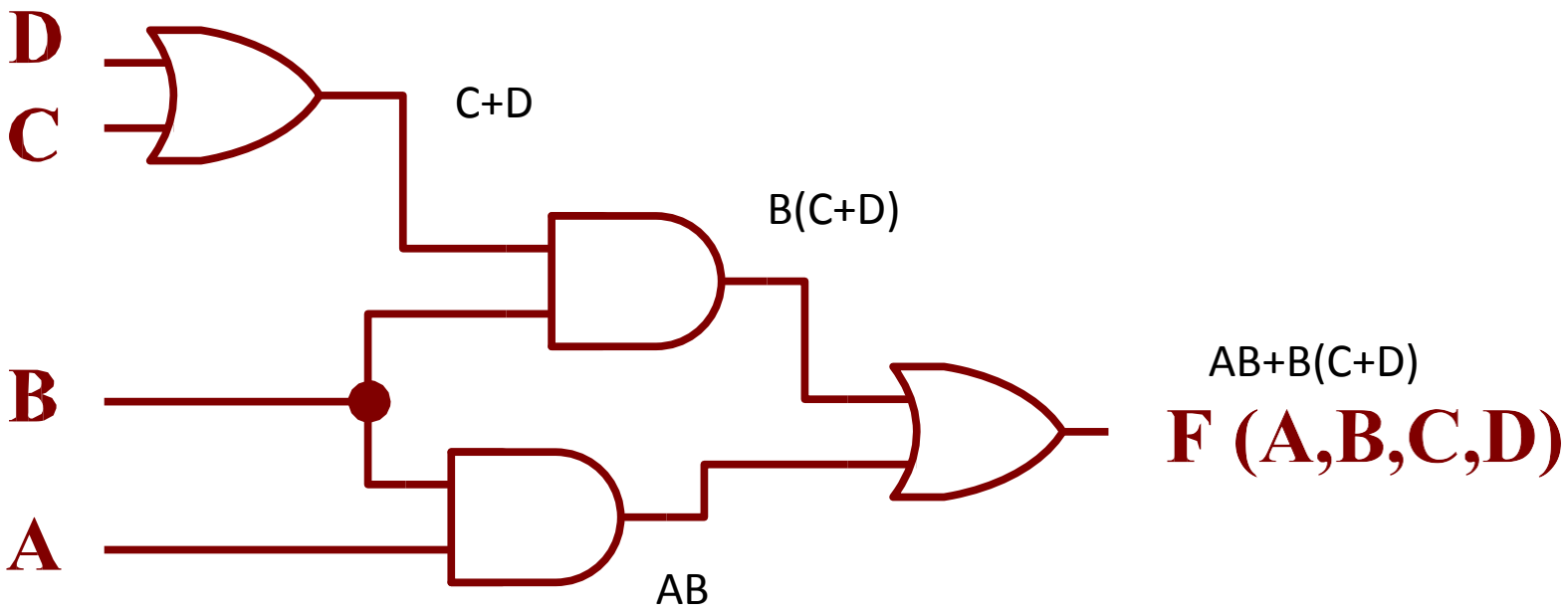
3- F(A,B,C,D) = (A+B'+C)D + B'C'D

Ejercicios F. Lógicas

SÍNTESIS DE CIRCUITOS

Utilizando sólo puertas AND, OR e inversores, dibujar el circuito digital que realice las siguientes funciones:

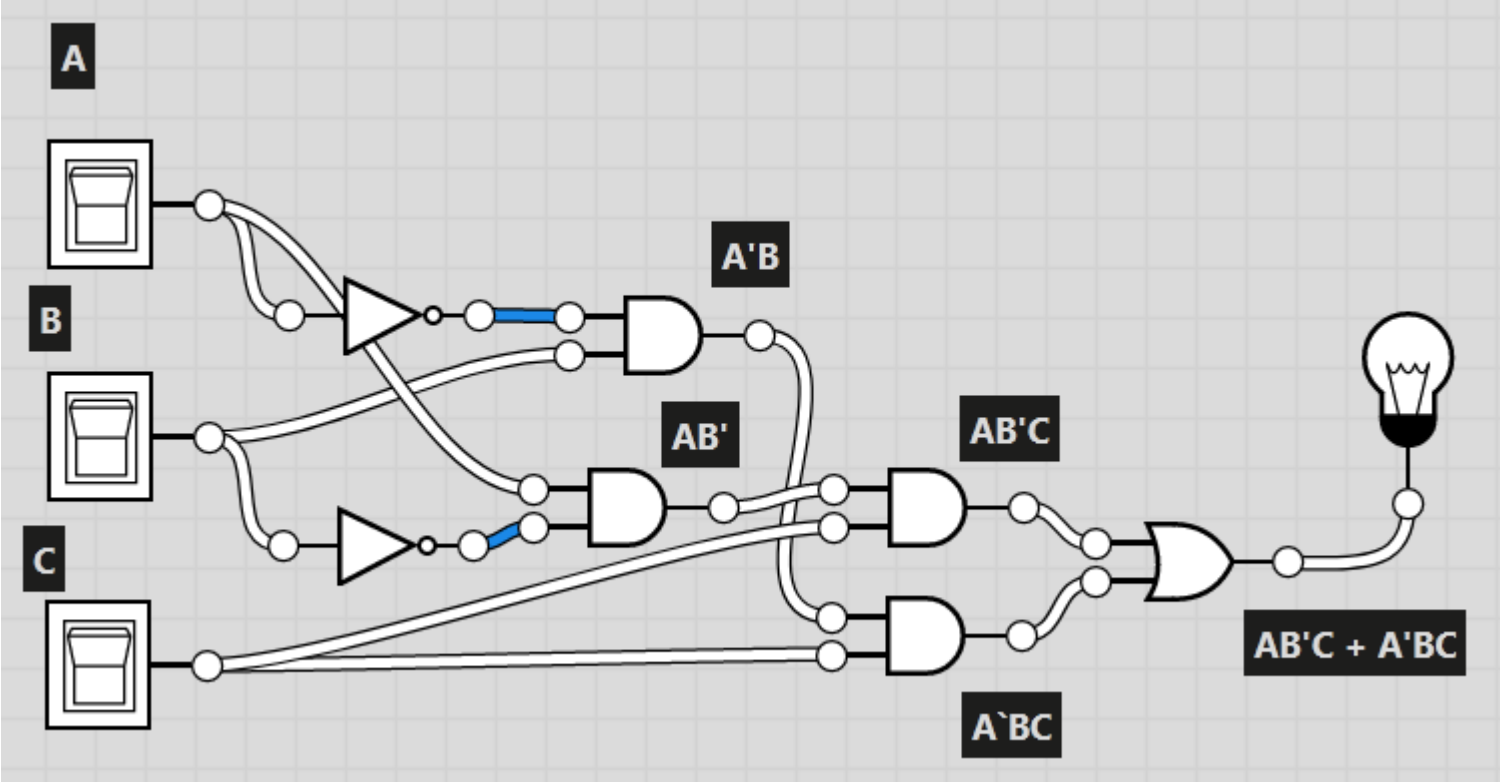
1- $F(A,B,C,D)=AB+B(C+D)$



Ejercicios F. Lógicas

Utilizando sólo puertas AND, OR e inversores, dibujar el circuito digital que realice las siguientes funciones:

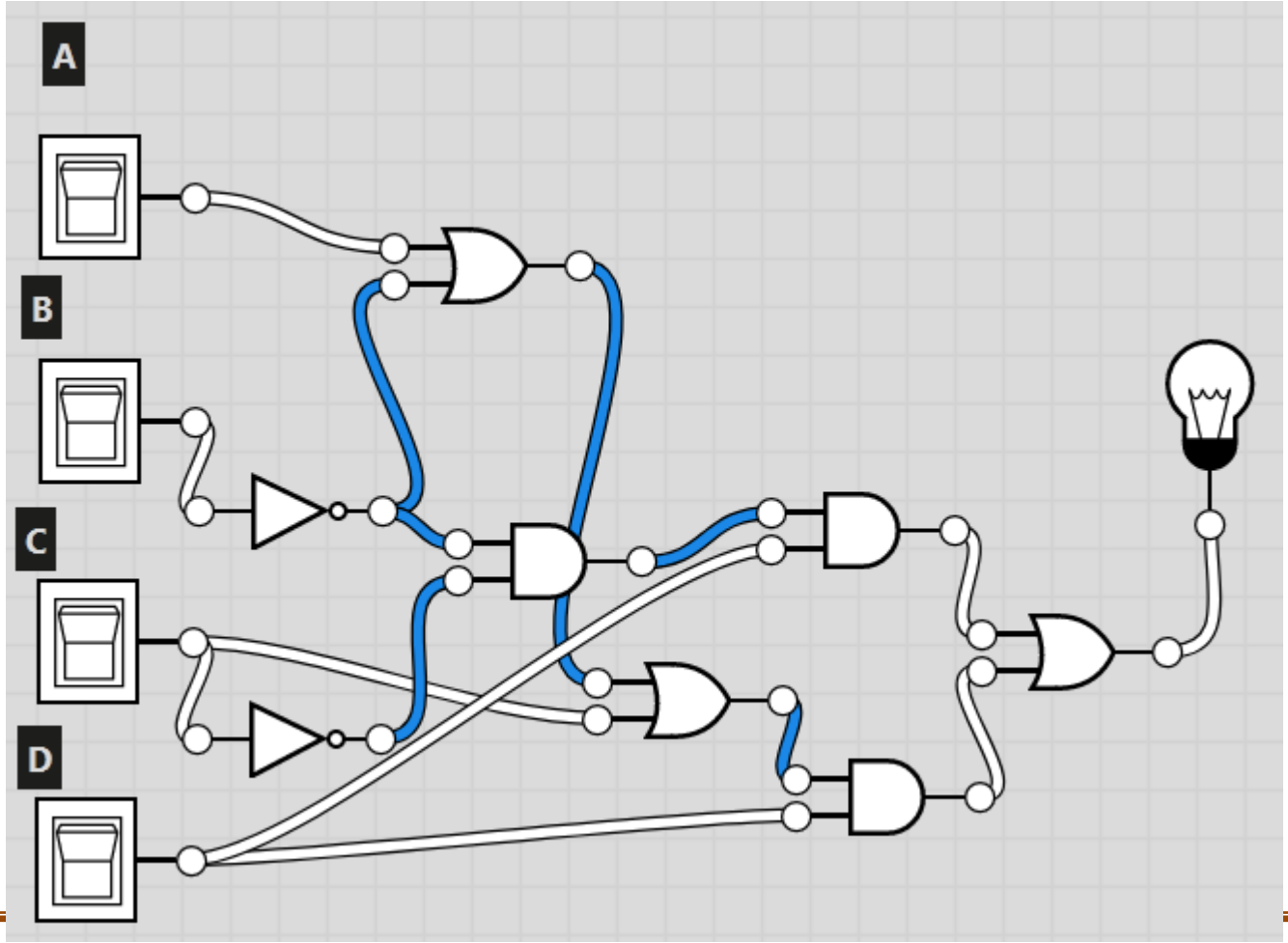
2- F(A,B,C) = A’BC+A(B’C)



Ejercicios F. Lógicas

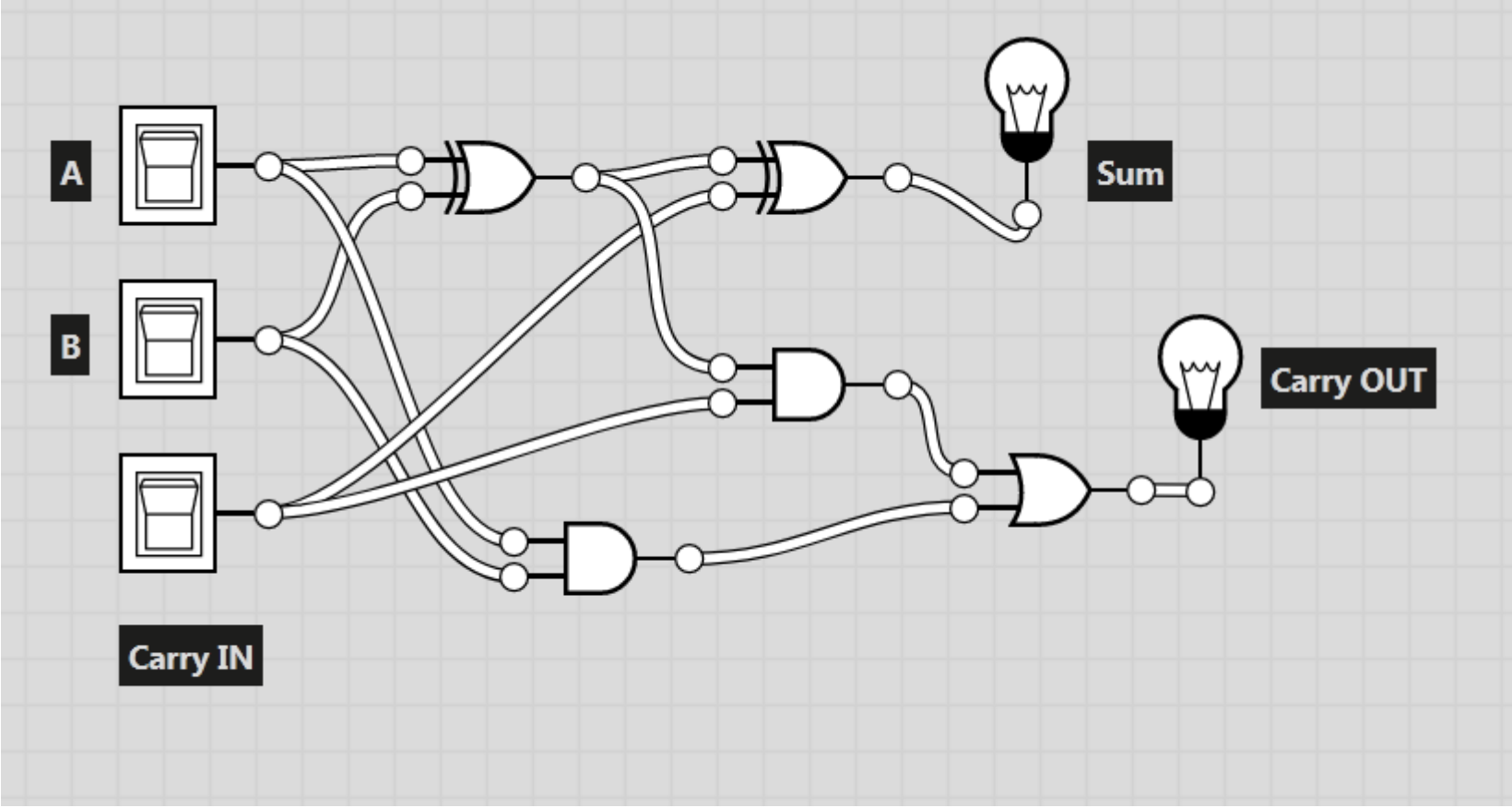
Utilizando sólo puertas AND, OR e inversores, dibujar el circuito digital que realice las siguientes funciones:

3- $F(A,B,C,D) = (A+B'+C)D + B'C'D$



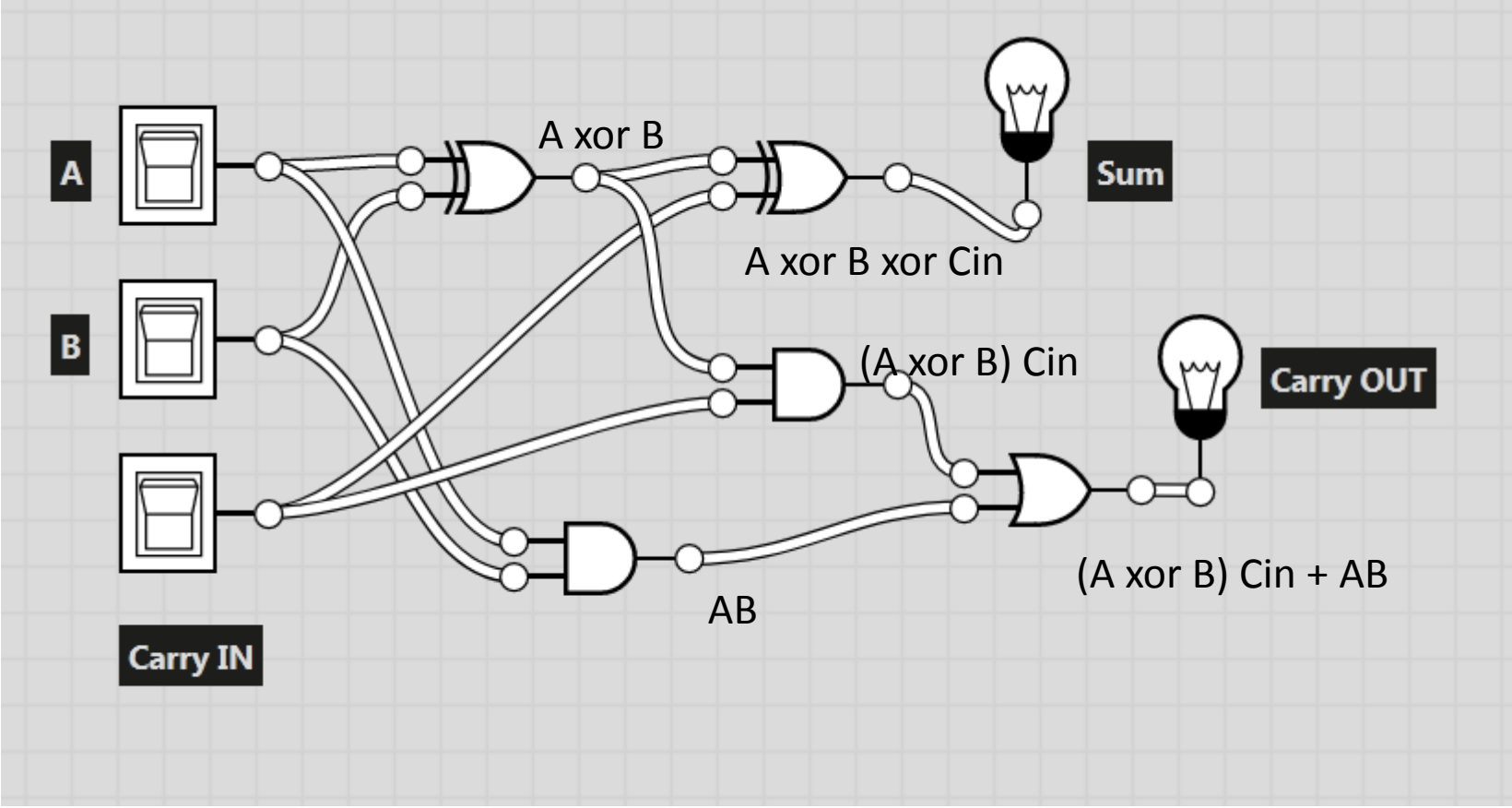
Ejercicios F. Lógicas

ANÁLISIS: Calcula las funciones lógicas de cada una de las salidas del siguiente circuito:



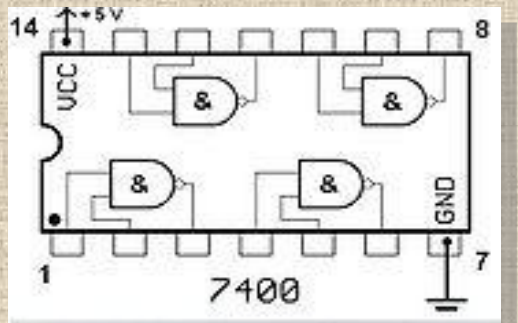
Ejercicios F. Lógicas

ANÁLISIS: Calcula las funciones lógicas de cada una de las salidas del siguiente circuito:



Circuitos lógicos combinacionales

- **CIRCUITO COMBINACIONAL** es todo sistema digital en el que sus salidas son función exclusiva del valor de sus entradas en un momento dado, sin que intervengan en ningún caso estados anteriores de las entradas o de las salidas. Por tanto, carecen de memoria y de retroalimentación.
- Los circuitos con **puertas lógicas** son circuitos combinacionales sencillos
- Están implementados mediante **circuitos integrados** (CI) en los que existen entradas, salidas, entradas de control, entrada de alimentación y entrada de puesta a tierra.



Circuitos combinacionales: tipos

- **Circuitos aritméticos:**
  - Semisumador
  - Sumador total
- **Circuitos lógicos:**
  - Codificador y decodificador
  - Multiplexor y demultiplexor
  - Comparador
  - Conversor de código
  - Generador/detector de paridad
- **Circuitos aritmético lógicos:**
  - Unidad aritmetico lógica (ALU)

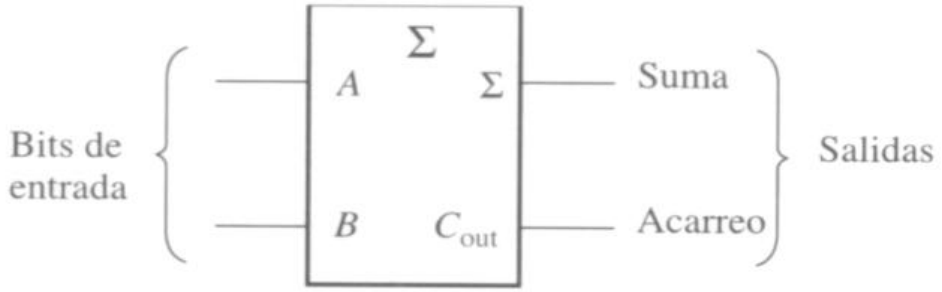
Semisumador (I)

- Realiza la suma de dos bits.
- Tiene dos entradas y dos salidas (suma y acarreo)
- La salida suma equivale a la función OR exclusiva, mientras que la salida de transporte equivale a las función AND

Tabla de verdad de un semisumador.

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

Σ = suma
C_{out} = acarreo de salida
A y B = variables de entrada (operandos)



Símbolo lógico de un semisumador.

Semisumador (II)

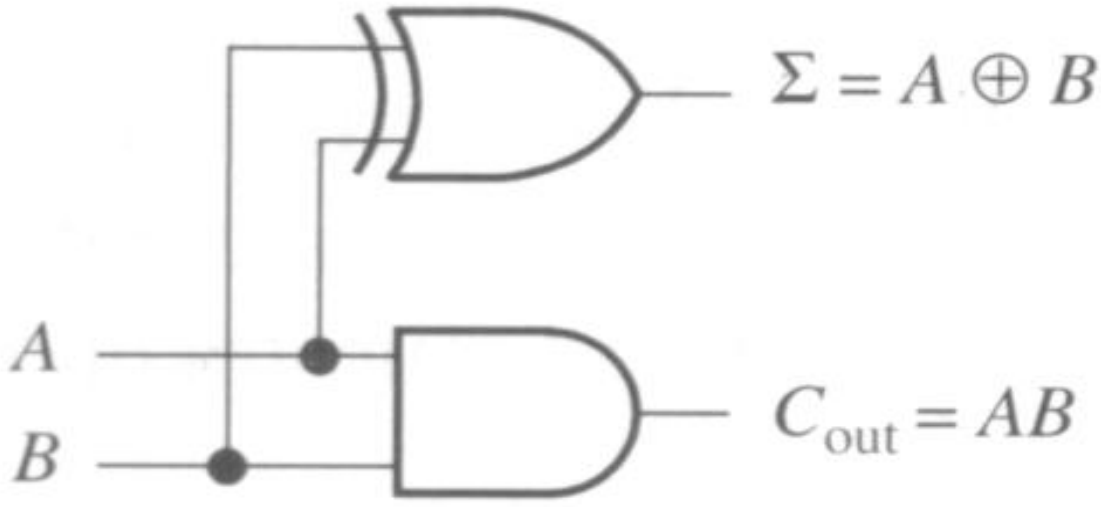


Diagrama lógico de un semisumador.

Sumador total (I)

- Realiza la suma de dos bits con acarreo anterior.
- Tiene tres entradas (dos bits y acarreo anterior) y dos salidas (suma y acarreo)
- Para sumar dos números en binario se utiliza un sumados por cada bit.

Símbolo lógico de un sumador completo.

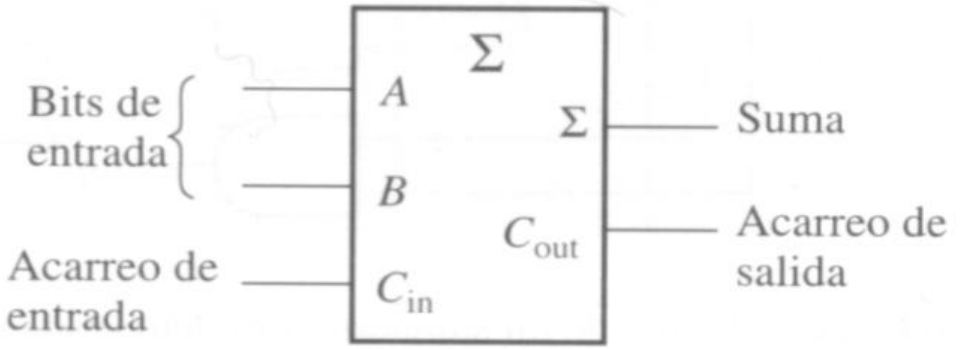


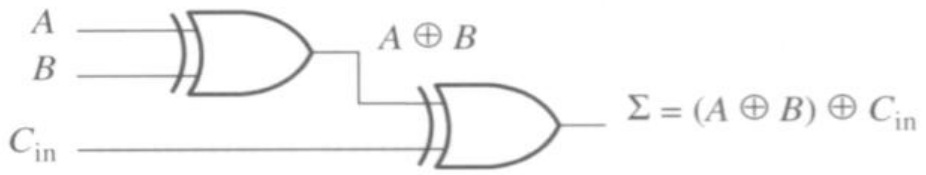
Tabla de verdad de un sumador completo.

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

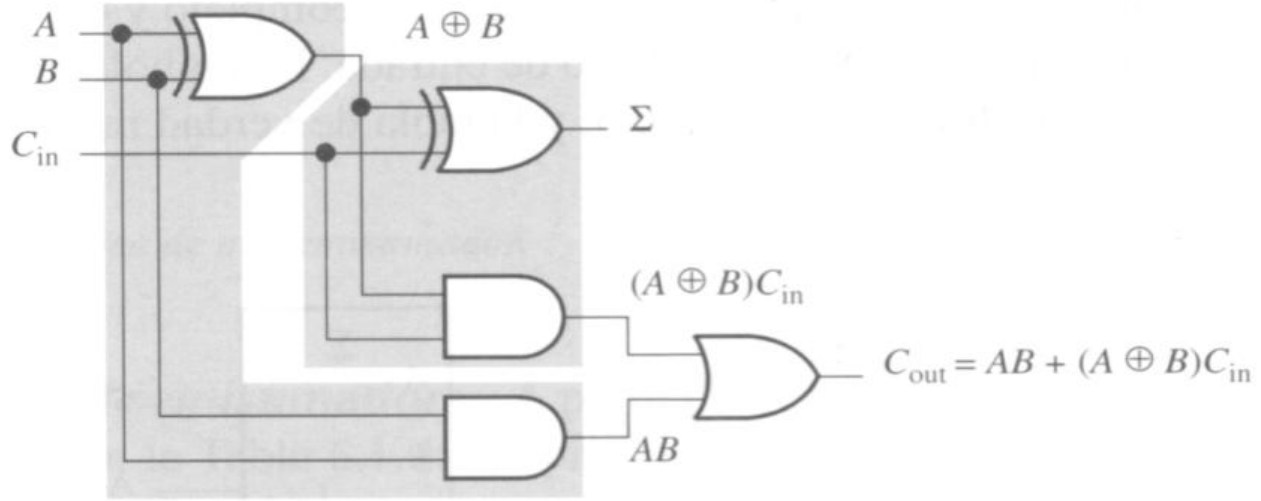
C_{in} = acarreo de entrada, algunas veces se designa por *CI*
C_{out} = acarreo de salida, algunas veces se designa por *CO*
Σ = suma
A y B = variables de entrada (operandos)

Sumador total (II)

Lógica de un sumador completo.



(a) Lógica necesaria para realizar la suma de tres bits

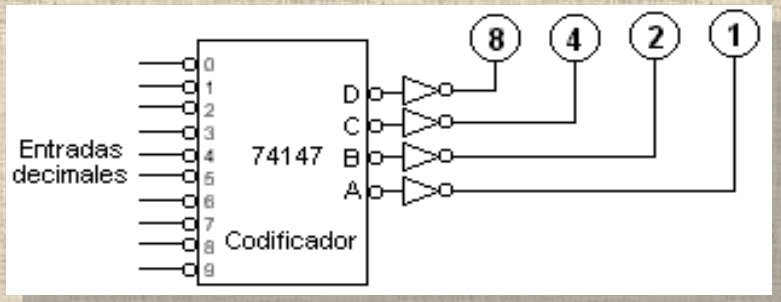


(b) Circuito lógico de un sumador completo (cada semisumador se representa por un área sombreada)

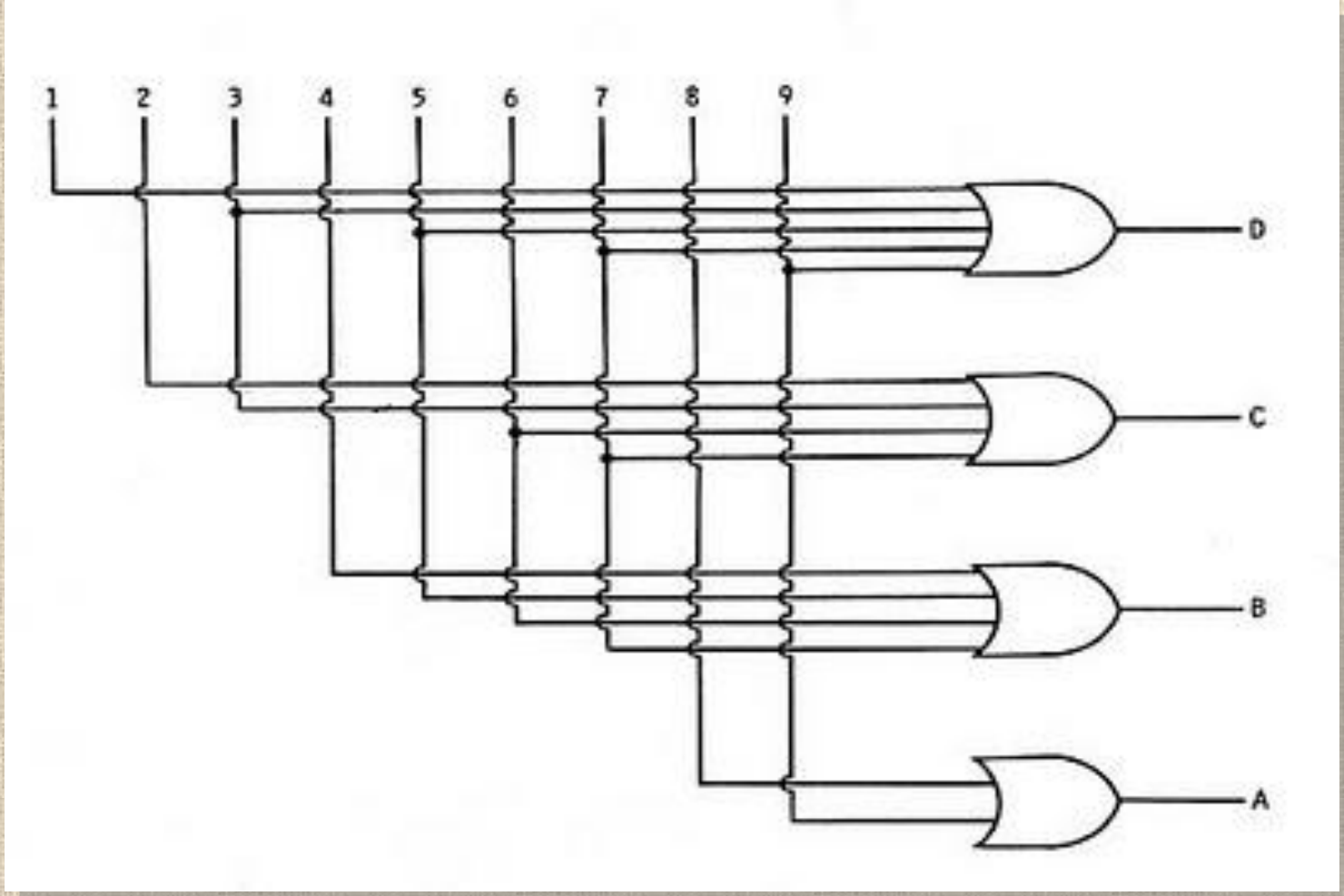
Codificador (I)

- Transforman un número en un código numérico en un número en código binario.
- Tiene N entradas y n salidas, cumpliéndose que $2^n \geq N$
- Los codificadores decimales tienen 10 entradas y 4 salidas.

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

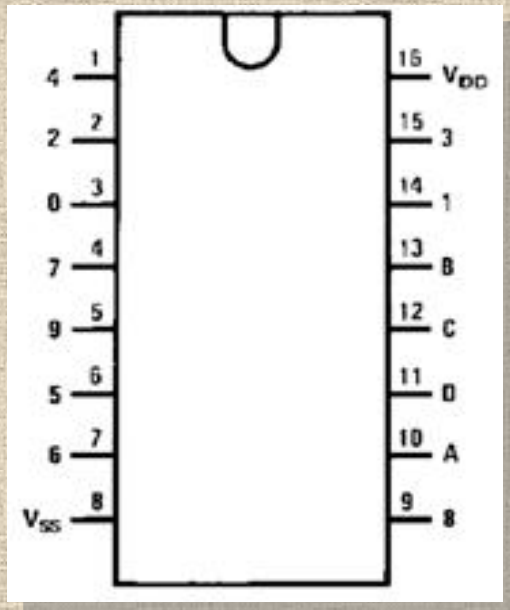


Codificador (I)



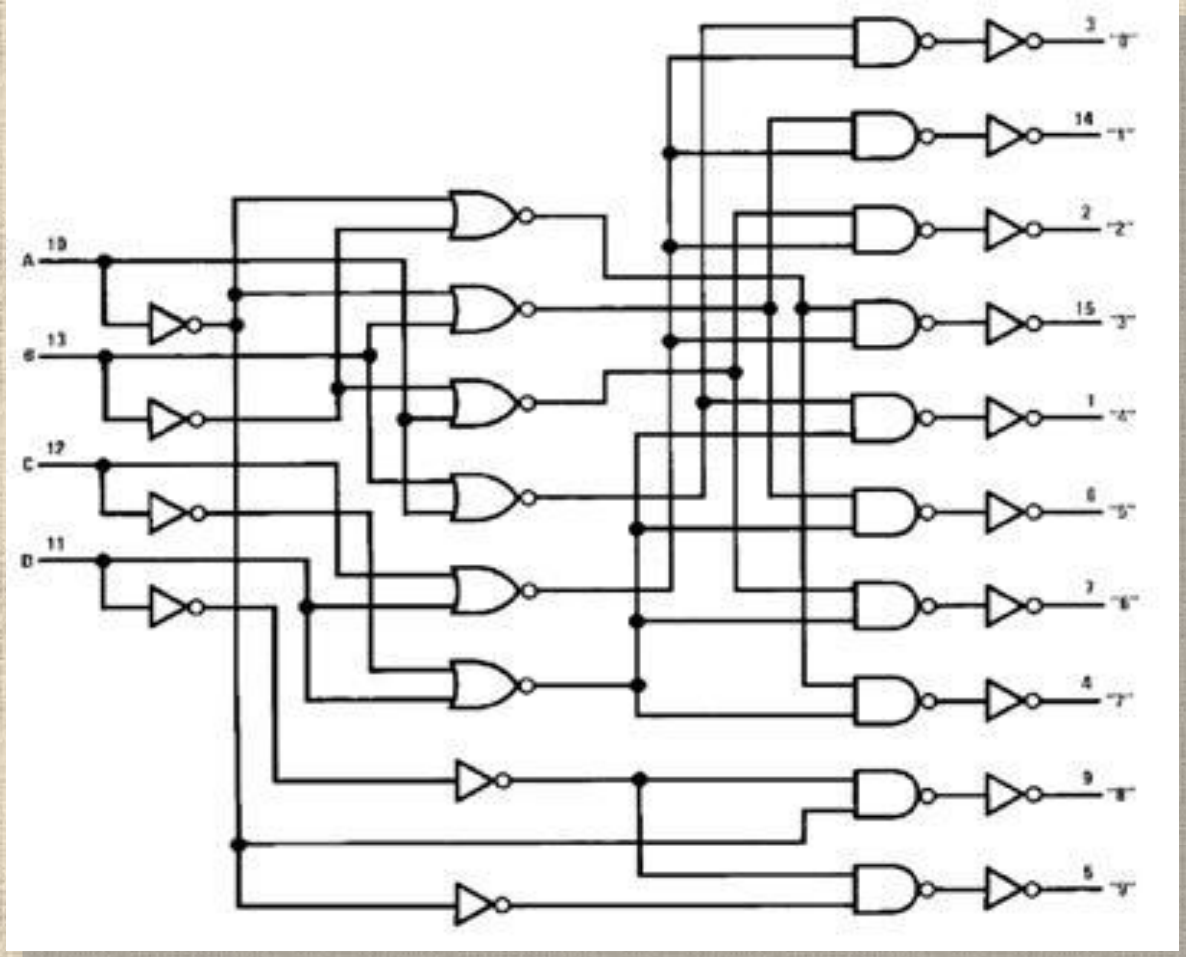
Decodificador (I)

- Transforman un número en código binario en un número en código decimal.
- Tiene n entradas y N salidas, cumpliéndose que $2^n \geq N$
- Los codificadores decimales tienen 4 entradas y 10 salidas.



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

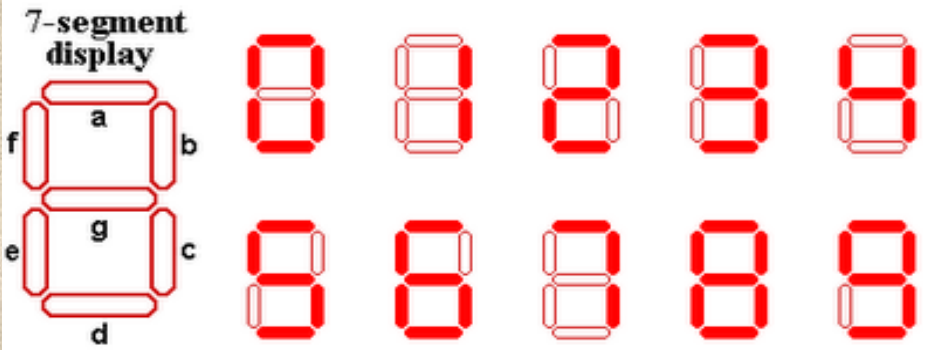
Decodificador (II)



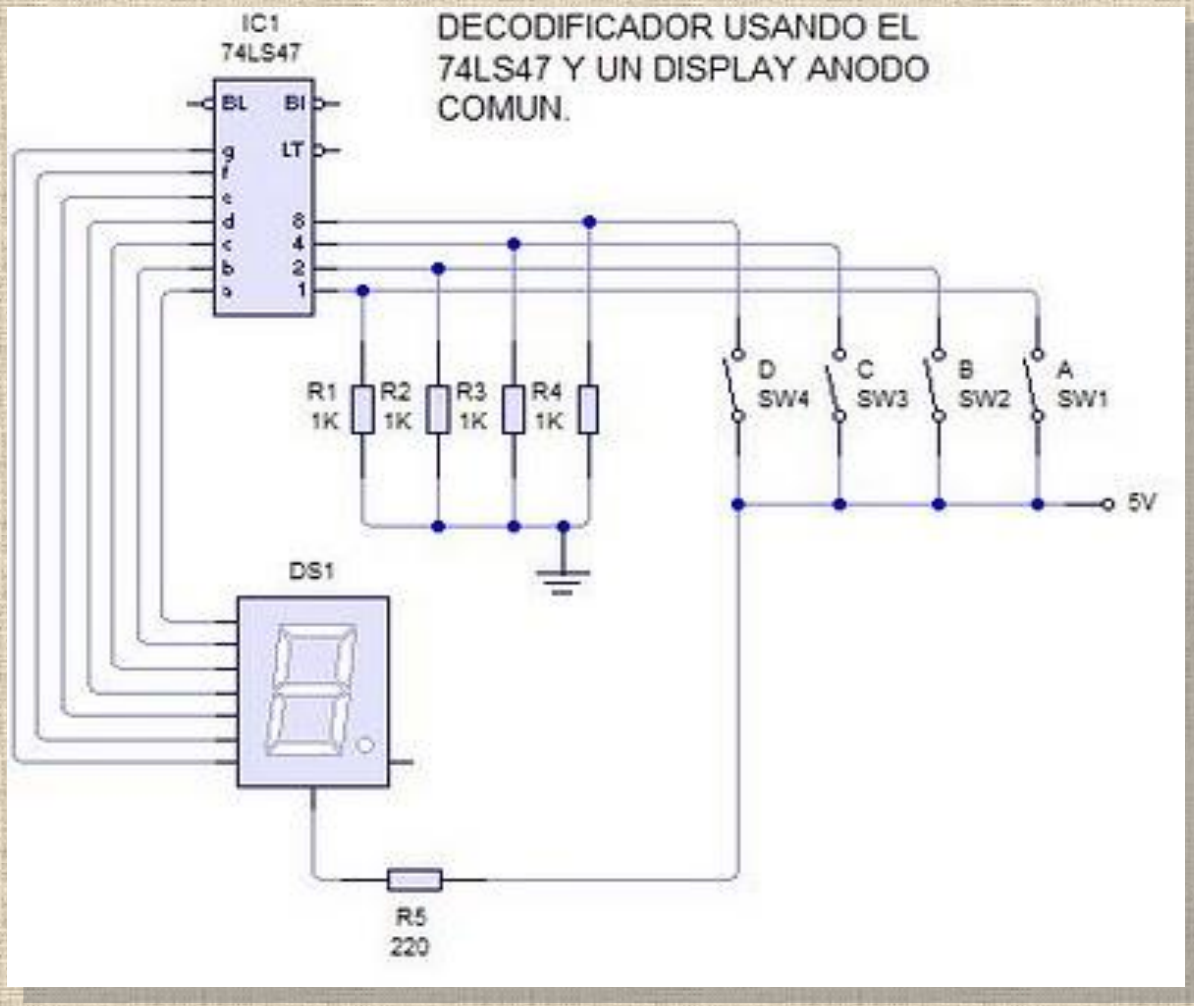
Decodificador BCD/7 segmentos (I)

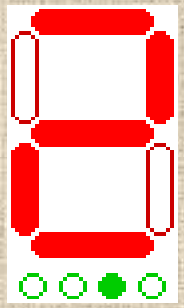
- Transforman un número en código binario en un número decimal que se implementa con un display de 7 segmentos.
- Tiene 4 entradas y 7 salidas.

Número Binario					LED						
					a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	1	0	1	1
1	0	1	0	10	0	0	0	0	0	0	0
1	0	1	1	11	0	0	0	0	0	0	0
1	1	0	0	12	0	0	0	0	0	0	0
1	1	0	1	13	0	0	0	0	0	0	0
1	1	1	0	14	0	0	0	0	0	0	0
1	1	1	1	15	0	0	0	0	0	0	0



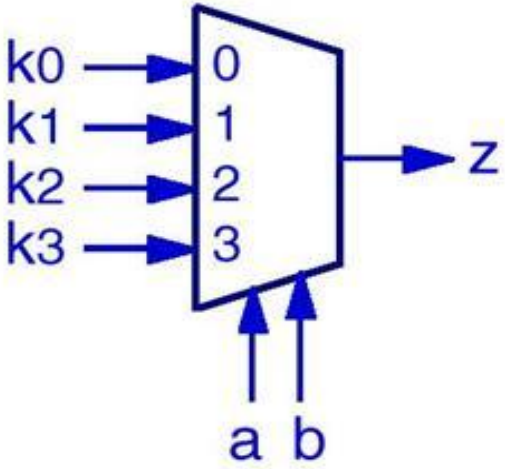
Decodificador BCD/7 segmentos (II)



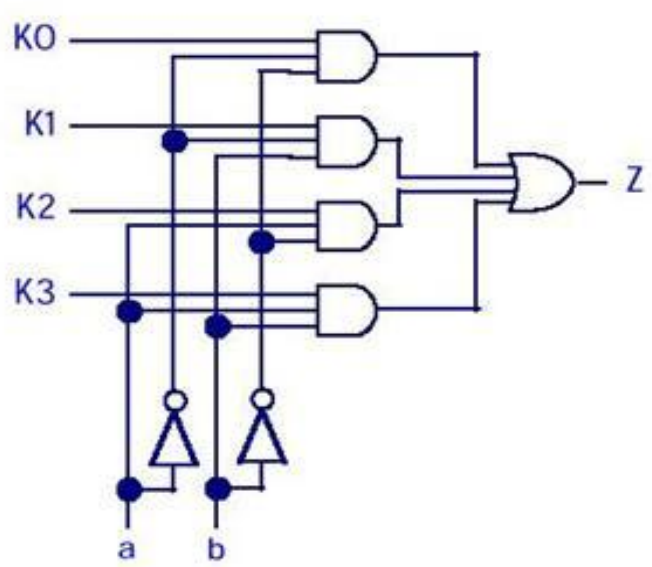


Multiplexor

- Permiten seleccionar la información de una de las entradas y ponerla en la salida.
- Para ello cuentan con n entradas de control, N entradas y 1 salida, cumpliéndose que 2n ≥ N

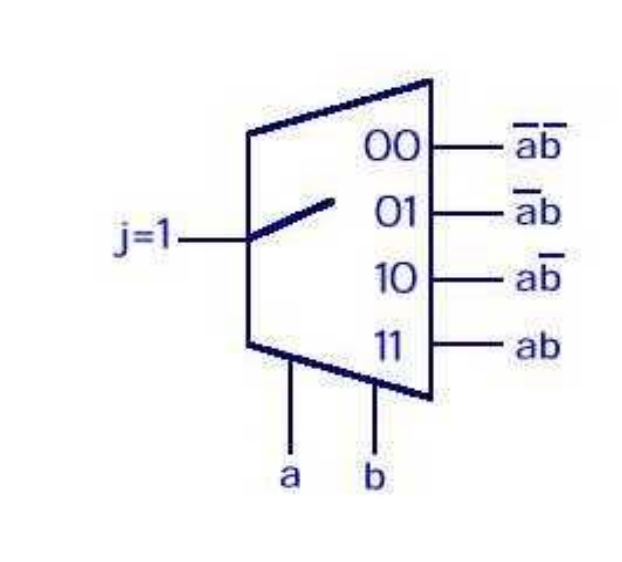


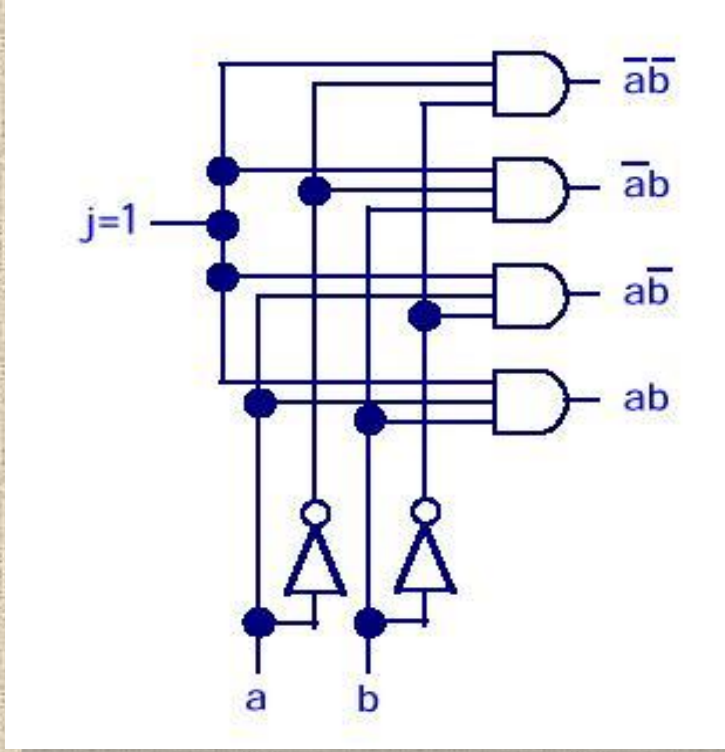
a	b	z
0	0	k0
0	1	k1
1	0	k2
1	1	k3



Demultiplexor

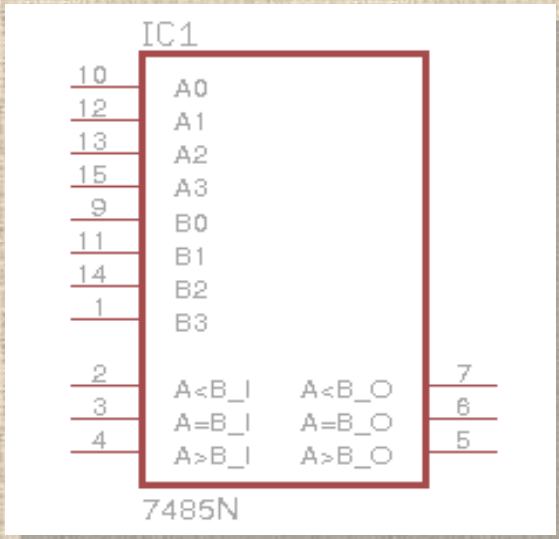
- Realiza la función inversa al multiplexor, permiten poner la información de la entrada en una de las salidas.
- Para ello cuentan con n entradas de control, N salidas y 1 entrada, cumpliéndose que $2n \geq N$

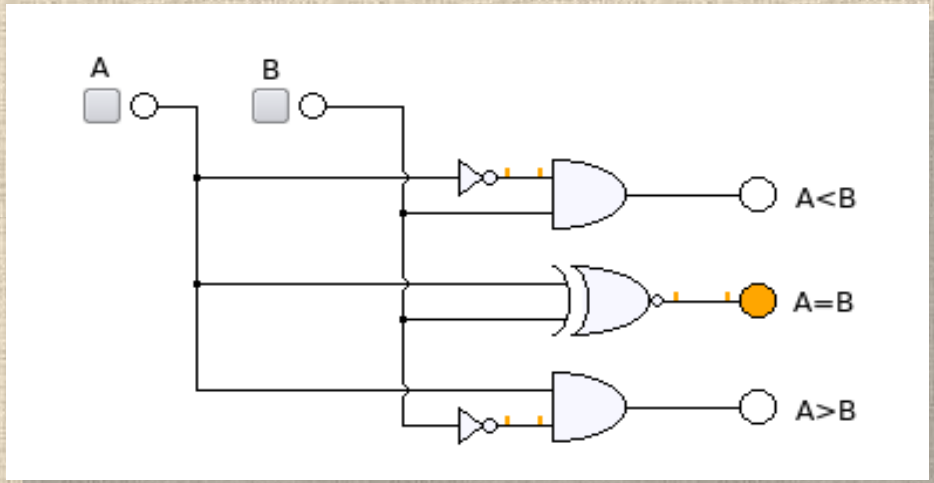




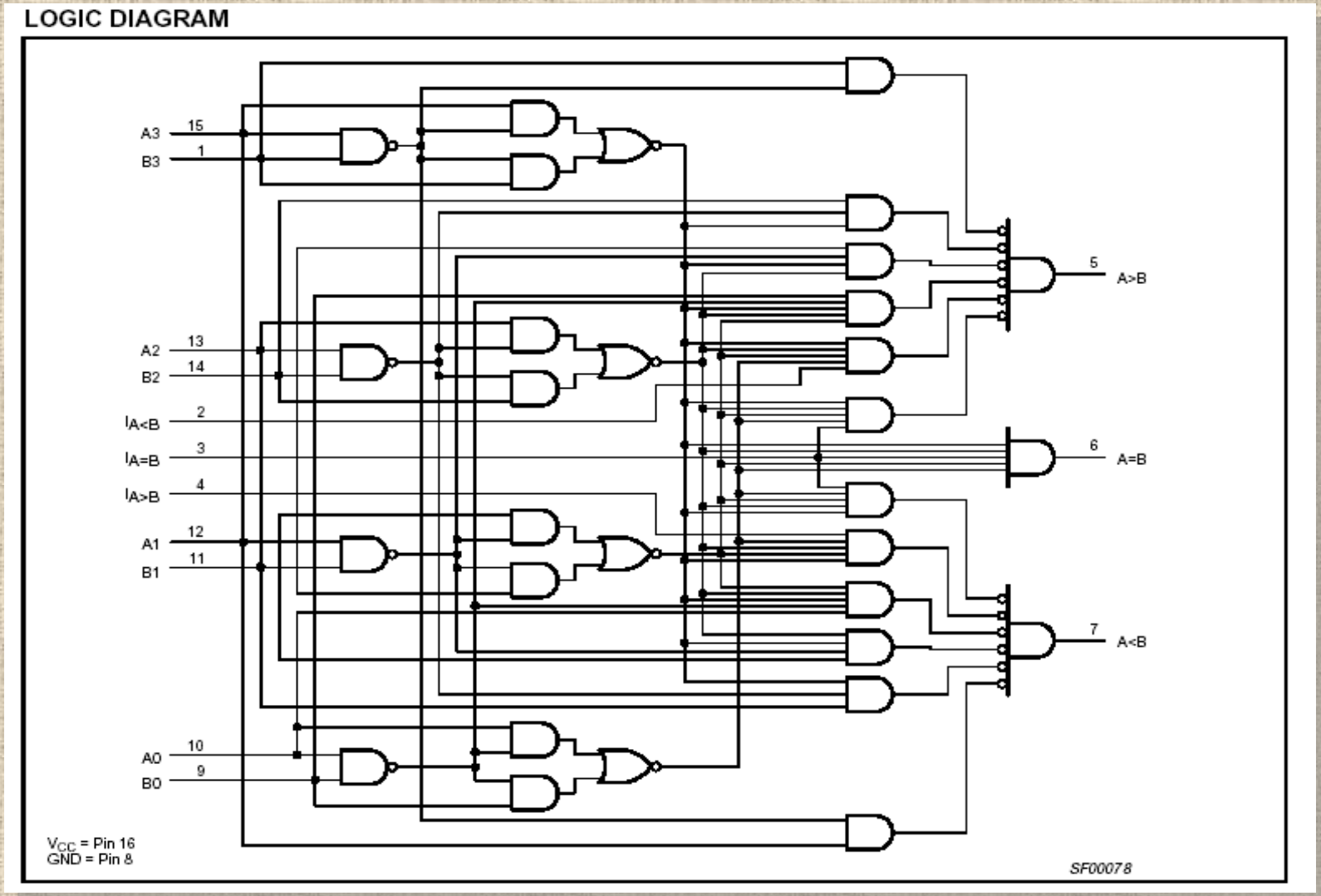
Comparador (I)

- Compara los números introducidos en las dos entradas activando una de las tres salidas: mayor, igual o menor.





Comparador (II)

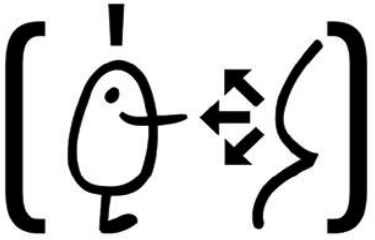


Juan Manuel Orduña Huertas

Fundamentos de computadores II

VNIVERSITAT DE VALÈNCIA

Escola Tècnica Superior d'Enginyeria ETSE-UV



VICERECTORAT DE PARTICIPACIÓ I PROJECCIÓ TERRITORIAL

Universitat i Societat

CULLERA