

TEMA 1. Sistemas Combinacionales.

OBJETIVOS

1. **Introducción a los sistemas digitales. Familias lógicas (3-22)**
2. **Definición de circuito combinacional (23-27)**
3. **Funciones combinacionales. Simplificación e implementación (28-86)**
 - 3.1 Variables, funciones y representación de redes lógicas (29-32)
 - 3.2 Axiomas y teoremas del álgebra de Boole. Dualidad (33-36)
 - 3.3 Expresión de funciones como suma de productos y producto de sumas. Términos canónicos. Tablas de verdad. (37-44)
 - 3.4 Simplificación de funciones. Mapas de Karnaugh (45-46)
 - 3.5 Implementación (47-85)
4. **Estructuras combinacionales básicas (86-129)**
 - 4.1 Puertas lógicas básicas (86)
 - 4.2 Multiplexores y demultiplexores (87-112)
 - 4.3 Codificadores y decodificadores (113-122)
 - 4.4 Compradores (123-129) Comparadores



OBJETIVOS

El objetivo principal de este capítulo es presentar los sistemas lógicos combinacionales y compararlos con los sistemas lógicos secuenciales, objeto estos últimos, de capítulos posteriores.

Establecer los aspectos básicos sobre el diseño de los sistemas combinacionales y su construcción utilizando puertas lógicas básicas.

Introducir estructuras combinacionales más complejas y su utilización en la construcción de sistemas

Por último descender en los niveles de diseño hasta aquel en que se utilizan de Circuitos Integrados, en nuestro caso, de baja y media escalas de integración; así como presentar y simular modelos de sistemas combinacionales construidos mediante herramientas de diseño de alto nivel.



1. Introducción (I)

1. Distinción entre las representaciones digitales y las analógicas
2. Mención de las ventajas y desventajas de las tecnologías digitales comparadas con las analógicas
3. Introducción a los sistemas digitales básicos



1. Introducción (II) Analógico versus Digital (I)

La **información** viene dada por los valores que toman un conjunto de magnitudes significativas. Las **magnitudes** pueden ser de dos tipos: **analógicas** y **digitales**.

Magnitudes analógicas: toman valores en un **rango continuo**.
Ejemplos: temperatura, voltaje, corriente eléctrica, tiempo, etc.

La **ELECTRONICA ANALOGICA** es la parte de la Electrónica que trabaja con **variables continuas** de tal forma que un pequeño cambio en alguna variable puede producir un gran cambio en el comportamiento del circuito. Por lo tanto, las variables serán **números reales**.

Magnitudes digitales: su **rango** de posibles valores es **discreto**.
Ejemplos: número de personas en un lugar, número de libros en una biblioteca, etc.

La **ELECTRONICA DIGITAL** es la parte de la Electrónica que trabaja con **variables discretas**. Este hecho implica que un pequeño cambio en alguna de las variables del circuito (siempre que no cambie su valor su característica de "discreto") no producirá un cambio apreciable en el comportamiento del circuito.

Es decir, el comportamiento del circuito no depende del valor exacto de la señal. Se corresponden matemáticamente con el concepto de **números enteros**.

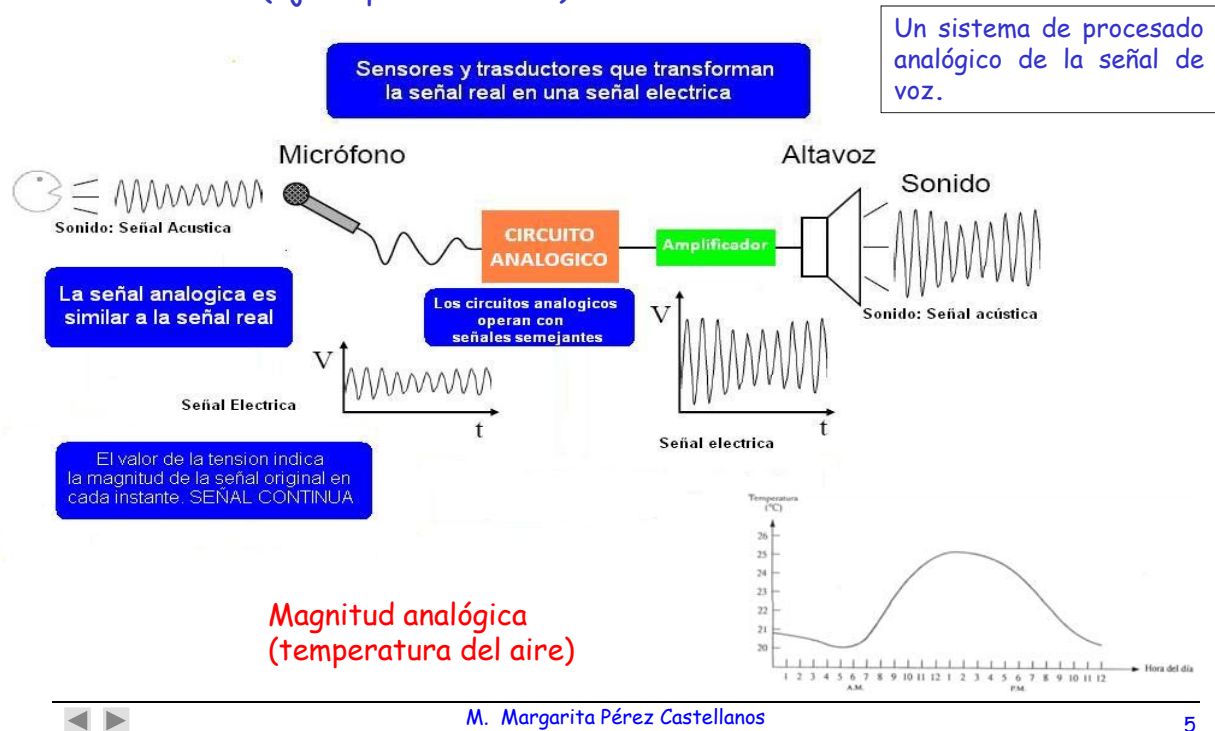


1. Introducción (III)

Analógico versus Digital (II)

Electrónica Analógica:

Trata con señales análogas a las que hay en el mundo real, modificando sus características (ej. amplificándolas).



1. Introducción (IV)

Analógico versus Digital (III)

En las señales analógicas, la información se encuentra en la forma de la onda.

Inconvenientes de los sistemas analógicos son:

1. La información está ligada a la forma de la onda.

Si la forma de onda se degrada, se pierde información.

2. Cada tipo de señal analógica necesita unos circuitos electrónicos particulares.

No es lo mismo un sistema electrónico para audio que para vídeo, puesto que las señales tienen características completamente diferentes.

1. Introducción (V)

Analógico versus Digital (IV)

Para minimizar los inconvenientes indicados

→ *conversión de las señales analógicas en digitales*

y posteriormente, si se requiere, reconstrucción de la señal

La validez de proceso de conversión analógico-digital y digital-analógico depende de la condición que impone *Nyquist*.



Harry Nyquist (1889-1976)

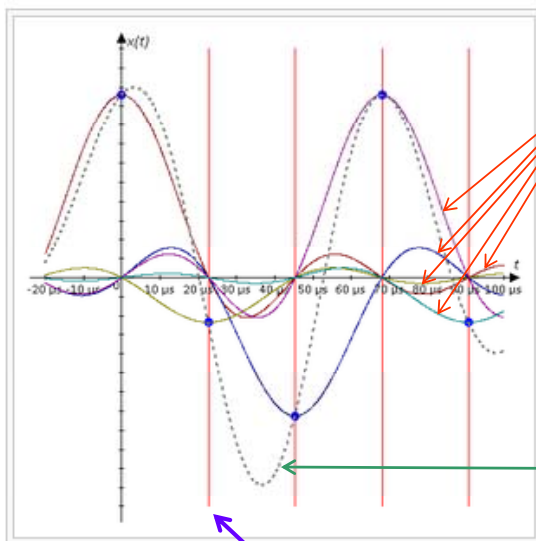
En 1927 el ingeniero sueco, Harry Nyquist, determinó que una señal analógica limitada en banda, para ser convertida en una representación adecuada en forma digital, debería ser muestreada, como mínimo, con una frecuencia doble que el ancho de banda de la señal.

Esta regla se conoce actualmente como el teorema de muestreo de *Nyquist-Shannon* y garantiza que cualquier señal se puede representar mediante números, y que con estos números se puede reconstruir la señal original.



1. Introducción (VI)

Analógico versus Digital (V)



La reconstrucción teórica se obtiene mediante la suma ponderada de la función de interpolación $g(t-nT)$ con $-\infty < n < \infty$, siendo los coeficientes de interpolación las muestras $x(n)$

Los 5 funciones de interpolación (5 colores), están ponderadas al valor de su correspondiente muestra (punto azul); el máximo de cada función pasa por el punto que representa su muestra

Ejemplo de reconstrucción de una señal de 14,7 kHz, con 5 muestras

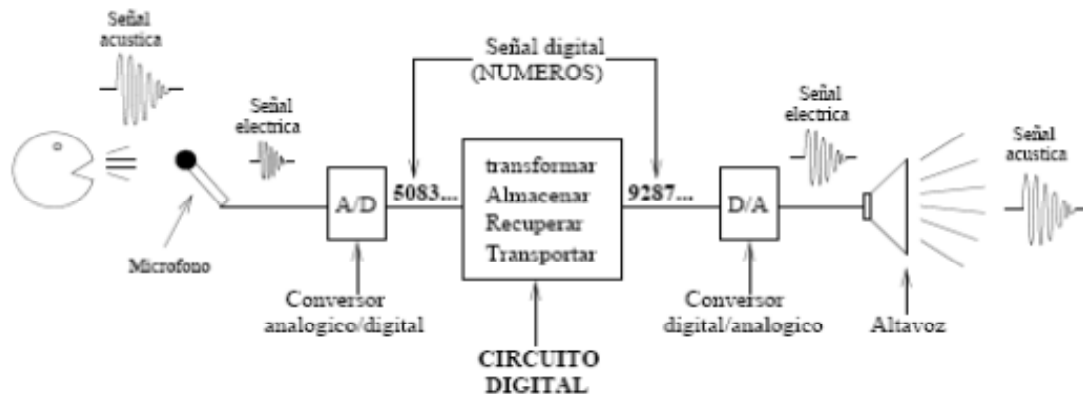
Cada ciclo se compone de solo 3 muestras a 44100 muestras/sg.



1. Introducción (VII)

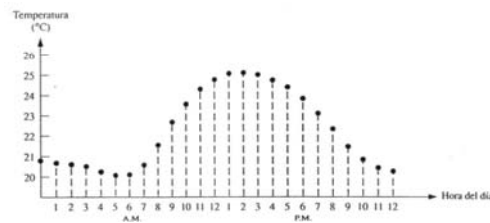
Analógico versus Digital (VI)

Una señal digital, es una señal que está descrita por números. La electrónica digital es la que trabaja con señales digitales.



Un sistema de tratamiento de voz, con electrónica digital

La temperatura como magnitud digital



1. Introducción (VIII)

Analógico versus Digital (VII)

Analógico vs. Digital

¿Por qué del éxito de los sistemas digitales?:

- Programables
- Flexibilidad y funcionalidad
- Mayor velocidad de procesamiento
- Mayor inmunidad al ruido
- Mayor capacidad de integración

Revolución digital:

- Cámaras Digitales
- DVD (video)
- CD (audio)
- Automóviles, teléfonos, tabletas, dispositivos 3D, efectos especiales...
- MEMS, NEMS..

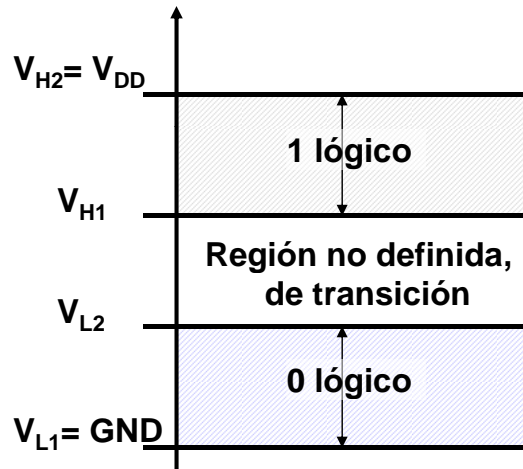


1. Introducción (IX)

Señales y sistemas digitales (I)

Un sistema digital es una combinación de dispositivos (eléctricos, mecánicos, fotoeléctricos,....) ensamblados con el fin de desempeñar funciones, en las cuales, las magnitudes se representan en forma digital.

- o Están diseñados para responder y producir tensiones en su entrada y salida respectivamente, que se clasifican dentro de los intervalos de tensión determinados como "0" y "1". Esto se traduce en que un circuito digital responde de la misma forma a todos los voltajes de entrada que se clasifiquen dentro del intervalo del 0 o 1 lógicos y, no diferenciará entre los voltajes de entrada que es clasifiquen dentro del 1 o 0 lógicos.

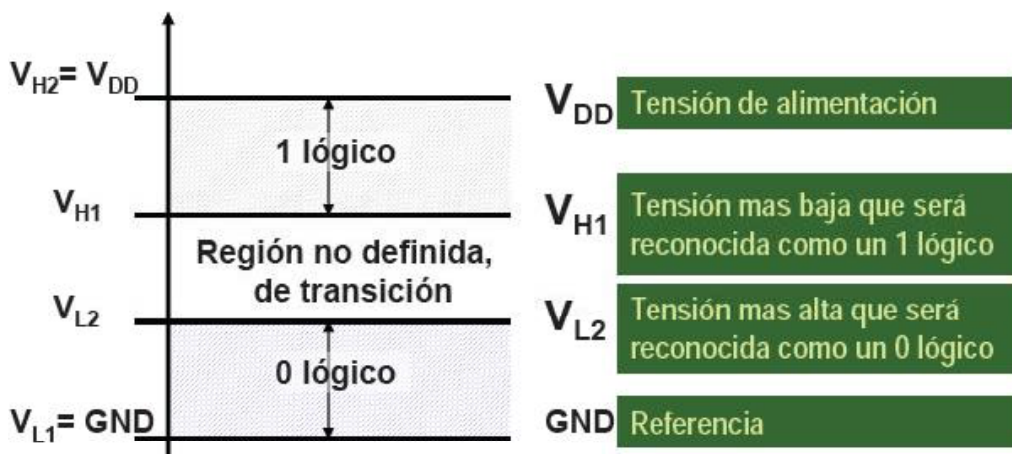


- o Las características de operación en modo binario nos va a permitir utilizar como herramienta, el álgebra booleana para analizar y diseñar sistemas digitales. Los sistemas digitales son: S. Combinacionales y S. Secuenciales



1. Introducción (X)

Señales y sistemas digitales (II)



Voltajes típicos		TTL	CMOS
V_{Hmax}		5 V	5 V
V_{Hmin}		2 V	3,5 V
Zona de incertidumbre			
V_{Lmax}		0,8 V	1 V
V_{Lmin}		0 V	0 V



1. Introducción (XI)

Señales y sistemas digitales (III)

Las señales digitales son, esencialmente, **niveles de tensión** que varían entre los estados alto y bajo. Una señal digital está compuesta por una **serie de pulsos**.

Formas de onda es la representación del conjunto de pulsos (tren de pulsos) que componen una señal digital

La información binaria que manejan los sistemas digitales aparece en forma de señales que representan **secuencias de bits**. Cuando la señal está en nivel alto, se representa con un 1 binario, mientras que si la señal está a a nivel bajo se indica con un 0 binario

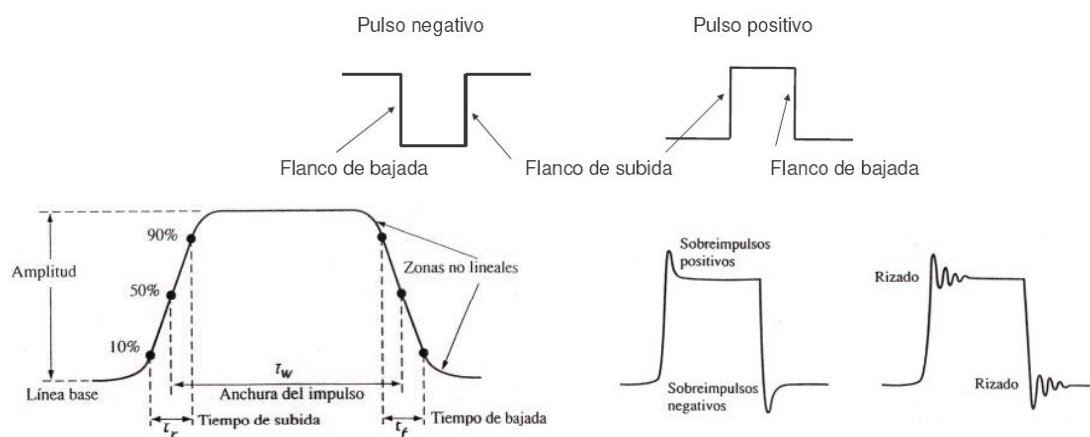
Diagrama de tiempos o cronograma es una gráfica que representa de forma precisa las relaciones temporales de varias señales y la variación de cada señal en función del tiempo



1. Introducción (XII)

Señales y sistemas digitales (IV)

- La evolución de una señal a lo largo del tiempo es: **la Forma de onda de la señal**
- Las formas de onda digitales se suelen representar en forma ideal, con transiciones instantáneas.
- Pulso: transiciones **H→L (alto → bajo)** y **L→H (bajo → alto)** (o viceversa) consecutivas y de una anchura determinada.

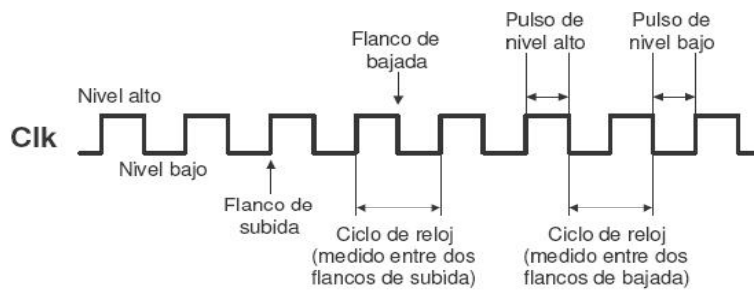


Representación de un pulso positivo no ideal



1. Introducción (XIII)

Señales y sistemas digitales (V)



Señal de Reloj (CLK): es una señal que varía periódicamente de forma infinita.

- Los sistemas digitales suelen contar con una señal de reloj (o varias) que sincroniza (n) a todas las demás.

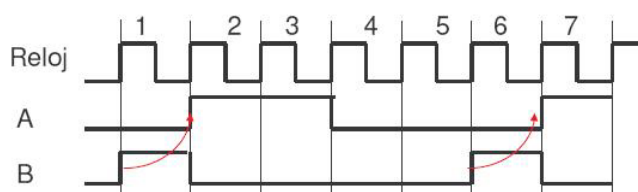


1. Introducción (XIV)

Señales y sistemas digitales (VI)

Cronograma o diagrama de tiempo: conjunto de formas de onda de varias señales, que normalmente están interrelacionadas

Los diagramas de tiempo suelen incluir una señal de reloj, que marca la evolución temporal de las señales



Evolución de las señales:

- En el periodo de reloj 1 A = "0" y B = "1".
- En el periodo 2 A = "1" y B = "0".



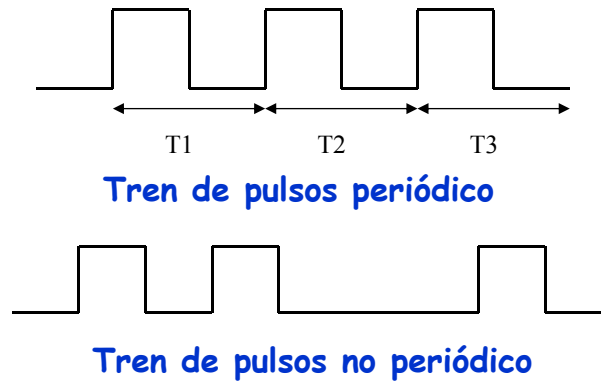
1. Introducción (XV)

Señales y sistemas digitales (VII)

Un tren de pulsos es un conjunto de pulsos continuos en el tiempo.

Cuando los intervalos de tiempo son fijos se forma un **tren periódico**, que queda definido mediante el valor de su periodo (T) o el de su inversa, la frecuencia (f)

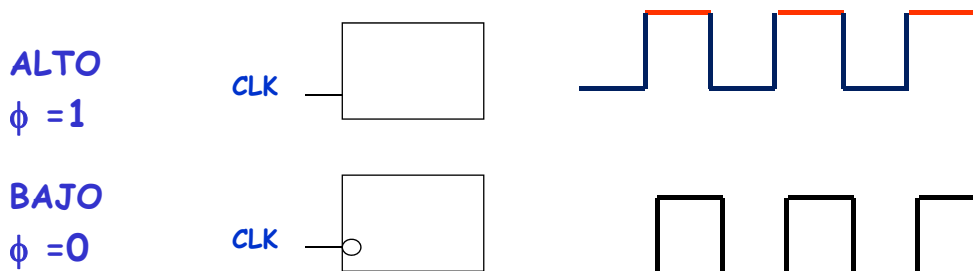
Si no tiene repetición de pulsos en forma periódica, se obtiene un tren de pulsos **no periódico**.



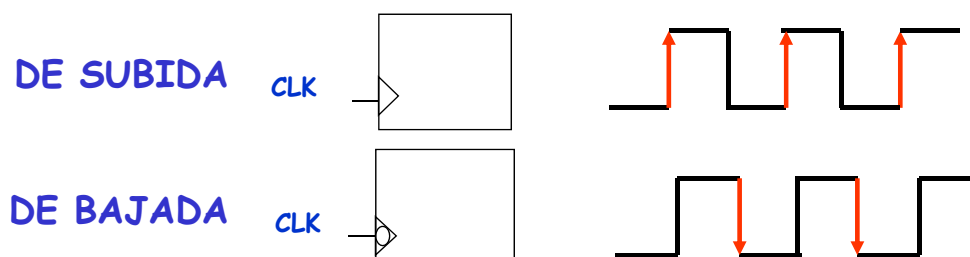
1. Introducción (XVI)

Señales y sistemas digitales (VIII)

• Activación de los sistemas mediante una señal de reloj (CLK): **POR NIVEL**



• Activación de los sistemas mediante una señal de reloj (CLK): **POR FLANCO**



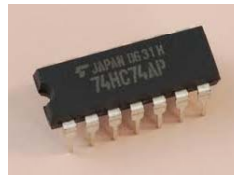
1. Introducción (XVII)

FAMILIAS LÓGICAS: Resumen (I)

Los circuitos digitales están agrupados en familias:

- Cada miembro de la familia, se fabrica con la misma tecnología, tiene una estructura similar y muestra las mismas características básicas
- Las características tanto eléctricas como lógicas, que son un conjunto de parámetros que discriminan a una familia. Son: tecnología de fabricación, retardos de propagación, fan-out, potencia disipada,

□ 74HC (XXXX)



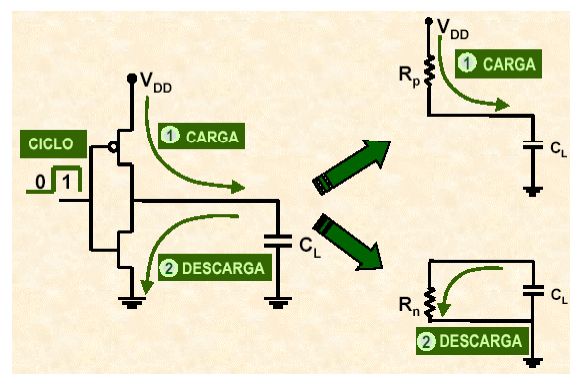
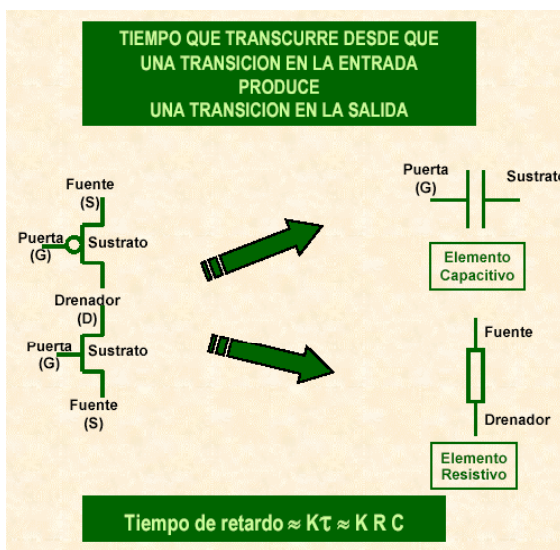
- Para elegir una familia u otra tendremos en cuenta: versatilidad lógica, velocidad, inmunidad al ruido, rango de temperaturas de operación, potencia disipada,



1. Introducción (XVIII)

FAMILIAS LÓGICAS: Resumen (II)

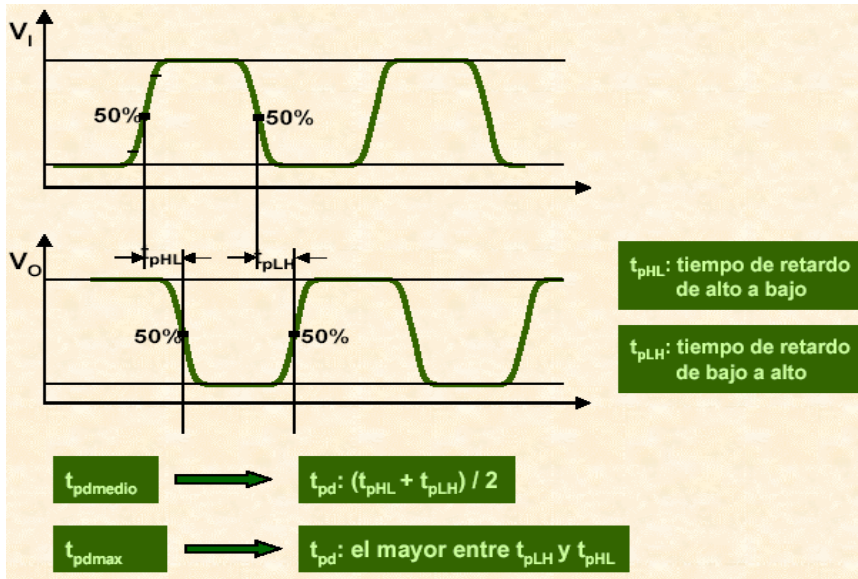
Retardos de propagación



1. Introducción (XIX)

FAMILIAS LÓGICAS: Resumen (III)

Retardos de propagación



t_{pHL} = tiempo de retardo de propagación desde la entrada (V_{input}) hasta la salida (V_{output}) para obtener una transición de salida de nivel alto (H) a nivel bajo (L).

t_{pLH} = tiempo de retardo de propagación desde la entrada (V_{input}) hasta la salida (V_{output}) para obtener una transición de salida de nivel bajo (L) a nivel alto (H).

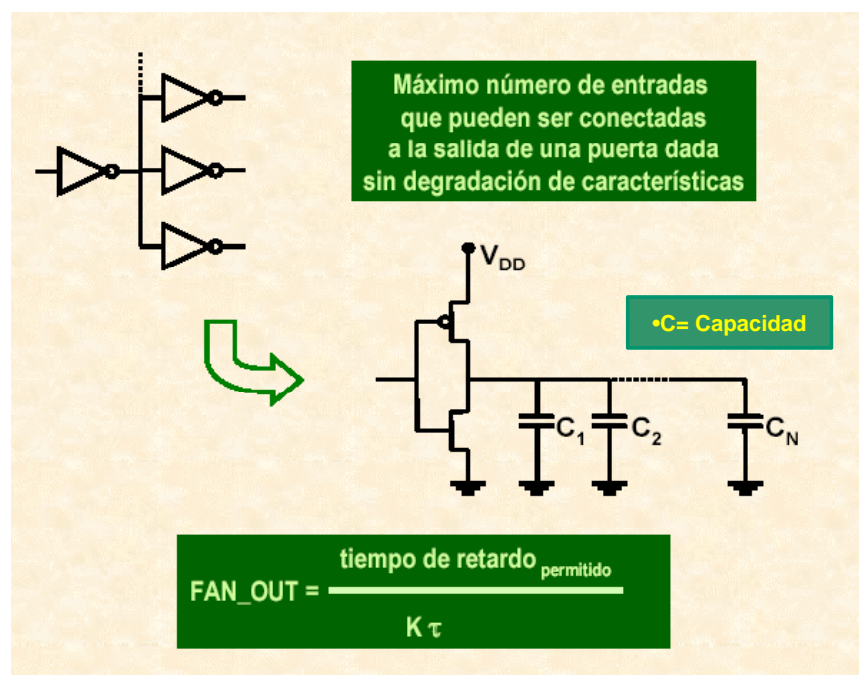
t_{pd} = retardo de propagación (propagation delay)



1. Introducción (XX)

FAMILIAS LÓGICAS: Resumen (IV)

Factor de carga (FAN-OUT)



2. DEFINICIÓN DE CIRCUITO COMBINACIONAL (I)

Un circuito lógico digital es puramente combinacional si la salida del mismo, en un instante dado, depende **única y exclusivamente** del valor que tengan sus entradas en el momento considerado.

En un circuito combinacional, salvo por el pequeño intervalo de tiempo que tardan en propagarse las señales, desde la entrada a la salida, dada la entrada, la salida estará determinada inmediatamente.



2. CIRCUITO COMBINACIONAL (II)

De la definición de circuito combinacional, podemos deducir:

- Las funciones de salida son una combinación de las variables de entrada presentes en cada momento → se puede representar mediante Funciones Lógicas de sus variables
- Es un sistema sin memoria
- Cada combinación de entrada sólo da lugar a un valor para la salida, por tanto el funcionamiento puede representarse mediante una tabla de verdad .



2. CIRCUITO COMBINACIONAL (III)

EJEMPLO: Sea un circuito de dos entradas que nos informe en su salida si ambas entradas son iguales entre si, o no lo son.

Entradas: A, B

Salida: z

$$A = B = 0 \rightarrow Z = 1$$

$$A = B = 1 \rightarrow Z = 1$$

$$A = 0, B = 1 \rightarrow Z = 0$$

$$A = 1, B = 0 \rightarrow Z = 0$$

A	B	Z
0	0	1
0	1	0
1	0	0
1	1	1

$$F(A,B) = Z = \bar{A}\bar{B} + AB$$

$$\bar{F}(A,B) = \bar{Z} = \bar{A}B + A\bar{B}$$



2. CIRCUITO COMBINACIONAL (IV)

Ejemplo de un circuito que NO ES combinacional: Constrúyase un circuito que nos indique si el número total de 1's presentados en su entrada hasta un instante determinado, es par o impar.



El sistema se puede especificar con una tabla de verdad en la que aparezcan las variables que intervienen en el sistema

Entrada x, salida z, "situación del sistema en un instante t"
estado Q



2. CIRCUITO COMBINACIONAL (V)

Tabla de verdad

	ENTRADA	SALIDA
Situación hasta el instante "t"	X(t)	Z(t)
Nº par de 1's	0	par
	1	impar
Nº impar de 1's	0	impar
	1	par

Consecuencias

- Este circuito no se puede representar en una tabla de verdad solamente con las variables de entrada
- Para el mismo valor de la entrada tenemos dos valores de la salida
- Tiene que recordar la información previa el instante actual: MEMORIA



3. Funciones combinacionales. Simplificación e implementación (I)

Para construir las funciones que representan a los circuitos combinacionales, se utiliza un conjunto de herramientas que permite especificar estos circuitos lógicos digitales.

SOPORTE ALGEBRAICO: ÁLGEBRA DE BOOLE.



3.1 Variables, funciones y representación de redes lógicas (I)

Un poco de historia.....

El matemático británico George Boole, publicó en 1854 la obra: INVESTIGACIÓN DE LAS LEYES DEL PENSAMIENTO, SOBRE LAS QUE SE BASAN LAS TEORÍAS MATEMÁTICAS DE LA LÓGICA Y LA PROBABILIDAD.

En esta publicación se generó la idea de "un álgebra de las operaciones lógicas" que se conoce en la actualidad como **ÁLGEBRA DE BOOLE**.

En 1938 Claude Shannon, publicó su tesis doctoral en el MIT ("A Symbolic Analysis of Relay and Switching Circuits") aplicando la obra de Boole al análisis y diseño de circuitos electrónicos.



3.1 Variables funciones y representación de redes lógicas (II)

TERMINOLOGÍA:

VARIABLE: es un símbolo que se utiliza para representar magnitudes lógicas (valor puede cambiar). Se designa a_i , A_i , x_i ...

Cualquier variable puede tener el valor 1 o 0

COMPLEMENTO: es el inverso de una variable y se indica mediante una barra encima de la misma \bar{a}_i , \bar{x}_i , \bar{A}_i ...

LITERAL: se define como una variable o el complemento de una variable

CONSTANTE: es un valor fijo (0,1)



3.1 Variables funciones y representación de redes lógicas (III)

TERMINOLOGÍA:

OPERACIONES en el álgebra de Boole son reglas que permiten diferentes combinaciones de elementos. Las básicas son:

ADICIÓN: $A+B$, $A \cup B$

MULTIPLICACIÓN: $A \cdot B$, $A \cap B$

INVERSIÓN: \bar{A} , $\neg A$, $\sim A$,

EXPRESIONES BOOLEANAS (Formas Booleanas, Expresiones Lógicas) son combinaciones de variables, constantes y operadores

FUNCIONES BOOLEANAS (Funciones Lógicas) son expresiones sin constantes



3.1 Variables funciones y representación de redes lógicas (IV)

TERMINOLOGÍA:

FORMAS ESTÁNDAR DE LAS EXPRESIONES BOOLEANAS: todas las expresiones booleanas independientemente de su forma pueden convertirse en: **suma de productos** o **producto de sumas**. Permite evaluar, simplificar e implementar expresiones booleanas de forma más sistemática.

Los términos suma o producto de una expresión si contienen todas las variables de la función afirmadas o negadas se denominan términos **canónicos**. **Minterms**, para el término producto y **Maxterms** para el término suma

TABLAS DE VERDAD es una forma de representar una expresión booleana utilizando todos los valores binarios de cada término de la expresión. Se forman con una columna por cada variable y otra para el valor de la función, y una fila por cada posible combinación de los valores de las variables.



3.2. AXIOMAS Y TEOREMAS DEL ÁLGEBRA DE BOOLE. DUALIDAD (I)

1. Ley conmutativa.

$$\square A+B = B+A$$

$$\square A \cdot B = B \cdot A$$

2. Ley Asociativa.

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

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

3. Ley distributiva.

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

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



3.2. AXIOMAS Y TEOREMAS DEL ÁLGEBRA DE BOOLE. DUALIDAD (II)

$$4. A+0 = A$$

$$5. A+1 = 1$$

$$6. A \cdot 0 = 0$$

$$7. A \cdot 1 = A$$

$$8. A+A = A$$

$$9. A+\bar{A} = 1$$

$$10. A \cdot A = A$$

$$11. A \bar{A} = 0$$

$$12. A = \bar{\bar{A}}$$

$$13. A+\bar{A} \cdot B = A+B$$

$$14. A+A \cdot B = A$$

$$15. (A+B) \cdot (A+C) = A+B \cdot C$$

$$16. A \cdot (A+B) = A \cdot B$$

$$17. A \cdot (\bar{A}+B) = A$$



3.2. AXIOMAS Y TEOREMAS DEL ÁLGEBRA DE BOOLE. DUALIDAD (III)

Teoremas de DE MORGAN

$$1. \overline{A \cdot B} = \overline{A} + \overline{B}$$

$$2. \overline{A + B} = \overline{A} \cdot \overline{B}$$

EJEMPLO:

$$\begin{aligned} Z &= \overline{A + B \cdot C} = \\ &= \overline{A} \cdot \overline{B \cdot C} = \overline{A} \cdot \overline{B + C} = \overline{A} \cdot (\overline{B} + \overline{C}) = \overline{A} \cdot \overline{B} + \overline{A} \cdot \overline{C} \end{aligned}$$



3.2. AXIOMAS Y TEOREMAS DEL ÁLGEBRA DE BOOLE. DUALIDAD (IV)

PRINCIPIO DE DUALIDAD: dado un teorema del Álgebra de Boole, existe otro teorema llamado TEOREMA DUAL que se obtiene sustituyendo:

- "+" por "•"
- "•" por "+"
- "0" por "1"
- "1" por "0"

EJEMPLO: de $A + B = B + A$ DUAL $A \cdot B = B \cdot A$



3.3. EXPRESIONES DE FUNCIONES COMO SUMAS DE PRODUCTOS O PRODUCTOS DE SUMAS. TÉRMINOS CANÓNICOS. TABLAS DE VERDAD (I)

EL ÁLGEBRA DE BOOLE proporciona una manera de expresar el funcionamiento de un circuito lógico formado por una combinación de puertas lógicas, de tal manera que la salida puede determinarse por la combinación de valores de entrada.

Cualquier circuito lógico se puede expresar mediante una EXPRESIÓN BOOLEANA.

La expresión booleana de un circuito lógico se puede desarrollar mediante una TABLA DE VERDAD. Dicho de otra forma, el funcionamiento de un circuito lógico se puede representar mediante una tabla de verdad.



3.3. EXPRESIONES DE FUNCIONES COMO SUMAS DE PRODUCTOS O PRODUCTOS DE SUMAS. TÉRMINOS CANÓNICOS. TABLAS DE VERDAD (II)

- La tabla de verdad representa todos los valores posibles que puede tomar la salida de un circuito lógico, para todas y cada una de las combinaciones posibles de las variables de entrada de las que depende.

- Es otra forma de representar una función lógica y se puede utilizar para obtener el desarrollo en forma canónica de la misma.

La tabla tiene una columna por cada variable del circuito

La tabla tiene una fila por cada posible combinación de valores de las variables de las que depende

$$\text{Sea } F(a,b) = \bar{a}\bar{b} + \bar{a}b$$

a	b	F
0	0	0
0	1	1
1	0	1
1	1	0



3.3. EXPRESIONES DE FUNCIONES COMO SUMAS DE PRODUCTOS O PRODUCTOS DE SUMAS. TÉRMINOS CANÓNICOS. TABLAS DE VERDAD (III)

- Cualquier expresión booleana que represente el funcionamiento de un circuito lógico, se puede convertir en una SUMA DE PRODUCTOS o en un PRODUCTO DE SUMAS.

- Si todos los términos de una suma de productos o de un producto de sumas, contienen todas las variables del sistema, estén complementadas (negadas) o no, se denominan EXPRESIONES CANÓNICAS O FORMAS NORMALES

-PRIMERA FORMA CANÓNICA, PRIMERA FORMA NORMAL O FORMA NORMAL DISYUNTIVA: es una expresión de una función booleana compuesta por suma de minitérminos (*minterms*)

- SEGUNDA FORMA CANÓNICA, SEGUNDA FORMA NORMAL O FORMA NORMAL CONJUNTIVA: es una expresión de una función booleana compuesta por suma de maxitérminos (*maxterms*)



3.3 EXPRESIONES DE FUNCIONES COMO SUMAS DE PRODUCTOS O PRODUCTOS DE SUMAS. TÉRMINOS CANÓNICOS. TABLAS DE VERDAD (IV)

- Si la suma o el producto está formada por términos que contienen todas las variables, negadas o sin negar, tenemos: SUMA CANÓNICA O PRODUCTO CANÓNICO
 - Cada PRODUCTO CANONICO corresponde a una fila de la tabla de verdad, en la cual la función toma el valor "1"
 - Cada SUMA CANÓNICA corresponde a una fila de la tabla de verdad, en la cual la función toma el valor "0"



3.3 EXPRESIONES DE FUNCIONES COMO SUMAS DE PRODUCTOS O PRODUCTOS DE SUMAS. TÉRMINOS CANÓNICOS. TABLAS DE VERDAD (V)

- Las formas canónicas se pueden extraer directamente de la tabla de verdad
- La primera forma canónica (1FC): suma de minterms, asociada a las filas cuyo valor de la función es 1.
 - $F(a,b,c,d) = \sum_4 m(0,1,8,9,10,11)$
- La segunda forma canónica (2FC): producto de maxterms, asociada a las filas cuyo valor de la función es 0.
 - $F(a,b,c,d) = \prod_4 M(2,3,4,5,6,7,12,13,14,15)$
- Las formas canónicas de una función son únicas. Las expresiones (1FC) y (2FC) son únicas.
- Las formas 1FC y 2FC de una función son equivalentes



3.3 EXPRESIONES DE FUNCIONES COMO SUMAS DE PRODUCTOS O PRODUCTOS DE SUMAS. TÉRMINOS CANÓNICOS. TABLAS DE VERDAD (VI)

a	b	c	d	Minitérmino	m_i	a	b	c	d	Maxitérmino	M_i
0	0	0	0	$\bar{a} \cdot \bar{b} \cdot \bar{c} \cdot \bar{d}$	m_0	0	0	0	0	$a+b+c+d$	M_0
0	0	0	1	$\bar{a} \cdot \bar{b} \cdot \bar{c} \cdot d$	m_1	0	0	0	1	$a+b+c+\bar{d}$	M_1
0	0	1	0	$\bar{a} \cdot \bar{b} \cdot c \cdot \bar{d}$	m_2	0	0	1	0	$a+b+\bar{c}+\bar{d}$	M_2
0	0	1	1	$\bar{a} \cdot \bar{b} \cdot c \cdot d$	m_3	0	0	1	1	$a+b+\bar{c}+d$	M_3
0	1	0	0	$\bar{a} \cdot b \cdot \bar{c} \cdot \bar{d}$	m_4	0	1	0	0	$a+\bar{b}+c+\bar{d}$	M_4
0	1	0	1	$\bar{a} \cdot b \cdot \bar{c} \cdot d$	m_5	0	1	0	1	$a+\bar{b}+c+d$	M_5
0	1	1	0	$\bar{a} \cdot b \cdot c \cdot \bar{d}$	m_6	0	1	1	0	$a+\bar{b}+\bar{c}+\bar{d}$	M_6
0	1	1	1	$\bar{a} \cdot b \cdot c \cdot d$	m_7	0	1	1	1	$a+\bar{b}+\bar{c}+d$	M_7
1	0	0	0	$a \cdot \bar{b} \cdot \bar{c} \cdot \bar{d}$	m_8	1	0	0	0	$\bar{a}+b+c+d$	M_8
1	0	0	1	$a \cdot \bar{b} \cdot \bar{c} \cdot d$	m_9	1	0	0	1	$\bar{a}+b+c+\bar{d}$	M_9
1	0	1	0	$a \cdot \bar{b} \cdot c \cdot \bar{d}$	m_{10}	1	0	1	0	$\bar{a}+b+\bar{c}+\bar{d}$	M_{10}
1	0	1	1	$a \cdot \bar{b} \cdot c \cdot d$	m_{11}	1	0	1	1	$\bar{a}+b+\bar{c}+d$	M_{11}
1	1	0	0	$a \cdot b \cdot \bar{c} \cdot \bar{d}$	m_{12}	1	1	0	0	$\bar{a}+\bar{b}+c+\bar{d}$	M_{12}
1	1	0	1	$a \cdot b \cdot \bar{c} \cdot d$	m_{13}	1	1	0	1	$\bar{a}+\bar{b}+c+d$	M_{13}
1	1	1	0	$a \cdot b \cdot c \cdot \bar{d}$	m_{14}	1	1	1	0	$\bar{a}+\bar{b}+\bar{c}+\bar{d}$	M_{14}
1	1	1	1	$a \cdot b \cdot c \cdot d$	m_{15}	1	1	1	1	$\bar{a}+\bar{b}+\bar{c}+d$	M_{15}



3.3 EXPRESIONES DE FUNCIONES COMO SUMAS DE PRODUCTOS O PRODUCTOS DE SUMAS. TÉRMINOS CANÓNICOS. TABLAS DE VERDAD (VII)

F (x₁, x₂, x₃, x₄),

X ₁	X ₂	X ₃	X ₄	F
0	0	0	0	1
0	0	0	1	1
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	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

□ Suma de productos (empleando los unos)

$$F = \overline{X_1}\overline{X_2}\overline{X_3}X_4 + \overline{X_1}\overline{X_2}X_3X_4 + \overline{X_1}\overline{X_2}X_3\overline{X_4} + X_1\overline{X_2}\overline{X_3}X_4 + X_1\overline{X_2}X_3\overline{X_4} + X_1\overline{X_2}X_3X_4$$

□ Producto de sumas (empleando los ceros)

$$F = (X_1 + X_2 + \overline{X_3} + X_4)(X_1 + X_2 + \overline{X_3} + \overline{X_4})$$

$$(X_1 + \overline{X_2} + X_3 + X_4)(X_1 + \overline{X_2} + X_3 + \overline{X_4})$$

$$(X_1 + \overline{X_2} + \overline{X_3} + X_4)(X_1 + \overline{X_2} + \overline{X_3} + \overline{X_4})$$

$$(\overline{X_1} + \overline{X_2} + X_3 + X_4)(\overline{X_1} + \overline{X_2} + X_3 + \overline{X_4})$$

$$(\overline{X_1} + \overline{X_2} + \overline{X_3} + X_4)(\overline{X_1} + \overline{X_2} + \overline{X_3} + \overline{X_4})$$



3.3 EXPRESIONES DE FUNCIONES COMO SUMAS DE PRODUCTOS O PRODUCTOS DE SUMAS. TÉRMINOS CANÓNICOS. TABLAS DE VERDAD (VIII)

□ Suma de productos (empleando los unos)

$$F = X_1\overline{X_2} + \overline{X_2}\overline{X_3}$$

$$F = \overline{X_1}\overline{X_2}\overline{X_3}X_4 + \overline{X_1}\overline{X_2}X_3X_4 + X_1\overline{X_2}\overline{X_3}X_4 + X_1\overline{X_2}X_3X_4 + X_1\overline{X_2}\overline{X_3}\overline{X_4} + X_1\overline{X_2}X_3\overline{X_4} + X_1\overline{X_2}X_3X_4$$

□ Producto de sumas (empleando los ceros)

$$F = \overline{X_2}(X_1 + \overline{X_3})$$

$$F = (X_1 + X_2 + \overline{X_3} + X_4)(X_1 + X_2 + \overline{X_3} + \overline{X_4})$$

$$(X_1 + \overline{X_2} + X_3 + X_4)(X_1 + \overline{X_2} + X_3 + \overline{X_4})$$

$$(X_1 + \overline{X_2} + \overline{X_3} + X_4)(X_1 + \overline{X_2} + \overline{X_3} + \overline{X_4})$$

$$(\overline{X_1} + \overline{X_2} + X_3 + X_4)(\overline{X_1} + \overline{X_2} + X_3 + \overline{X_4})$$

$$(\overline{X_1} + \overline{X_2} + \overline{X_3} + X_4)(\overline{X_1} + \overline{X_2} + \overline{X_3} + \overline{X_4})$$



3.4 SIMPLIFICACIÓN DE FUNCIONES. MAPAS DE KARNAUG (I)

Procedimiento regular de reducción de las expresiones Lógicas mediante una tabla.

Sea $P (c_1c_2c_3)$

c_1	c_2	c_3	P
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

P_1	C_2C_3			
C_1	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$$P = C_1C_2 + C_1C_3 + C_2C_3$$



3.4 SIMPLIFICACIÓN DE FUNCIONES. MAPAS DE KARNAUG (II)

$S (a_1a_0b_1b_0)$

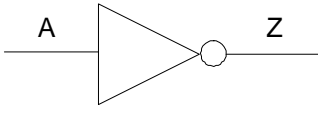
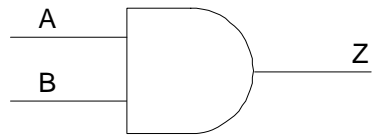
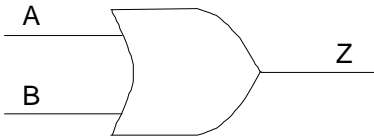
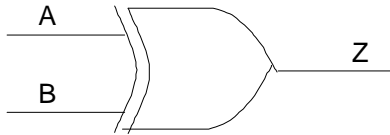
a_1	a_0	b_1	b_0	S
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

S	b_1b_0			
a_1a_0	00	01	11	10
00	1	1	1	1
01	0	1	1	1
11	0	0	1	0
10	0	0	1	1

$$S (a_1a_0b_1b_0) = \bar{a}_1\bar{a}_0 + \bar{a}_1b_1 + \bar{a}_0b_1 + \bar{a}_1b_0 + b_1b_0$$

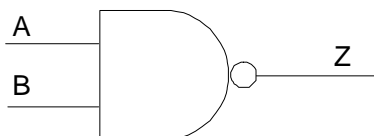
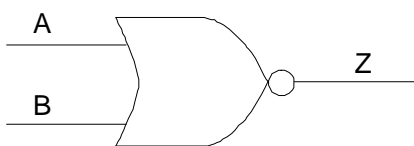
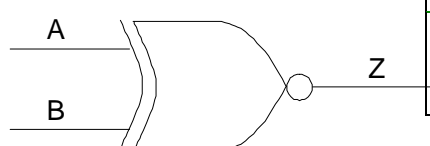


3.5 IMPLEMENTACIÓN: PUERTAS LÓGICAS BÁSICAS (I)

Función	Operación	Símbolo	Tabla de verdad															
NOT	$Z = \overline{A}$		<table border="1"> <thead> <tr> <th>A</th> <th>Z</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> </tr> </tbody> </table>	A	Z	0	1	1	0									
A	Z																	
0	1																	
1	0																	
AND	$Z = A \cdot B$		<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Z</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table>	A	B	Z	0	0	0	0	1	0	1	0	0	1	1	1
A	B	Z																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR	$Z = A + B$		<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Z</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table>	A	B	Z	0	0	0	0	1	1	1	0	1	1	1	1
A	B	Z																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
XOR	$Z = A \oplus B$		<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Z</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	A	B	Z	0	0	0	0	1	1	1	0	1	1	1	0
A	B	Z																
0	0	0																
0	1	1																
1	0	1																
1	1	0																

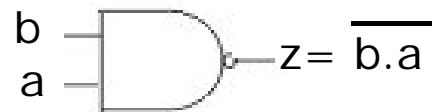
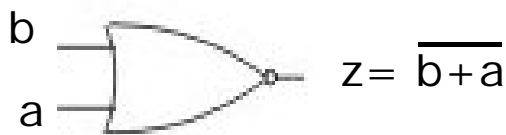
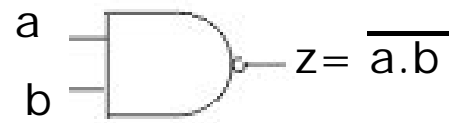
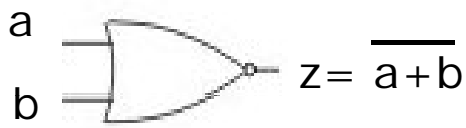


3.5 IMPLEMENTACIÓN: PUERTAS LÓGICAS BÁSICAS (II)

Función	Operación	Símbolo	Tabla de verdad															
NAND	$Z = \overline{A \cdot B}$		<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Z</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	A	B	Z	0	0	1	0	1	1	1	0	1	1	1	0
A	B	Z																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
NOR	$Z = \overline{A + B}$		<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Z</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	A	B	Z	0	0	1	0	1	0	1	0	0	1	1	0
A	B	Z																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
XNOR	$Z = \overline{A \oplus B}$		<table border="1"> <thead> <tr> <th>A</th> <th>B</th> <th>Z</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table>	A	B	Z	0	0	1	0	1	0	1	0	0	1	1	1
A	B	Z																
0	0	1																
0	1	0																
1	0	0																
1	1	1																



3.5 IMPLEMENTACIÓN DE AXIOMAS DE ÁLGEBRA DE BOOLE (III)



Conmutativa de la suma

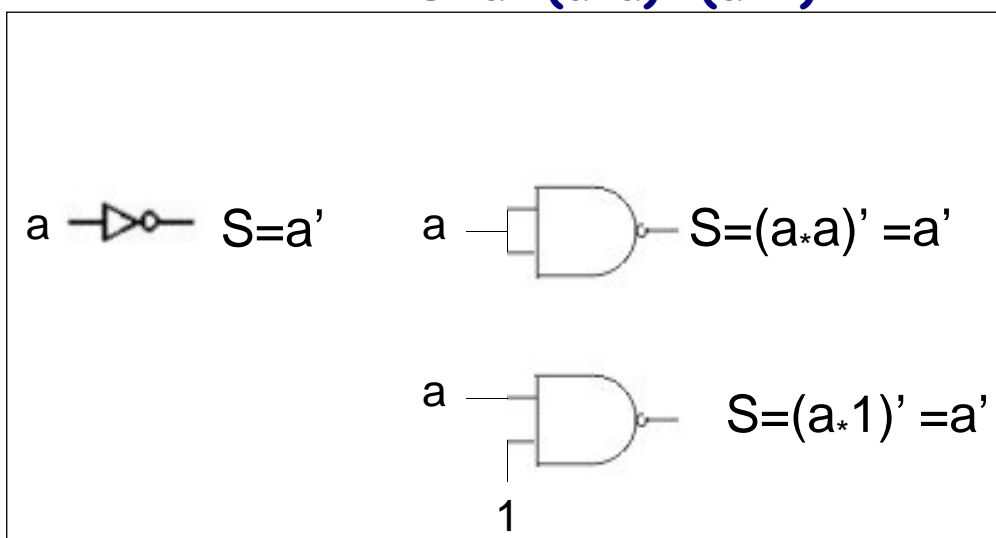
Conmutativa del producto



3.5 IMPLEMENTACIÓN DE AXIOMAS DE ÁLGEBRA DE BOOLE (IV)

T7 y T10

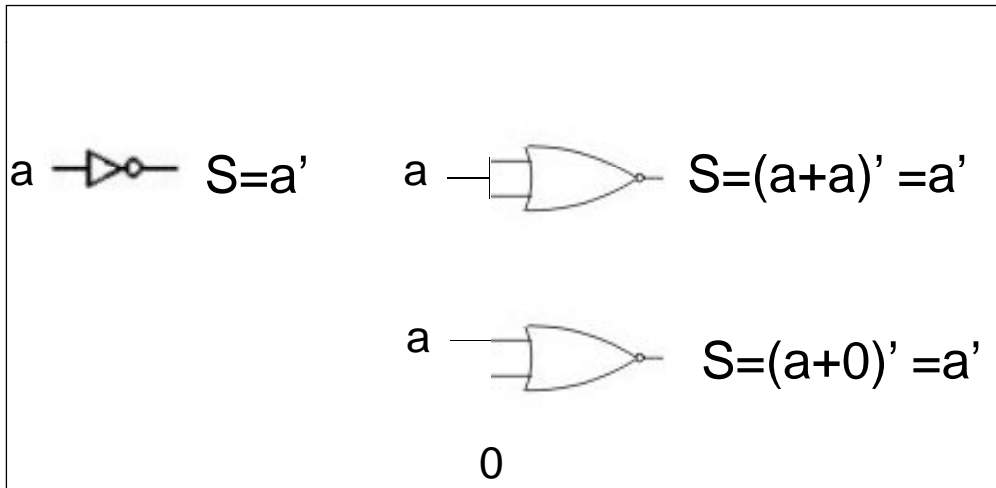
NOT: $S=a'=(a*a)'=(a*1)'$



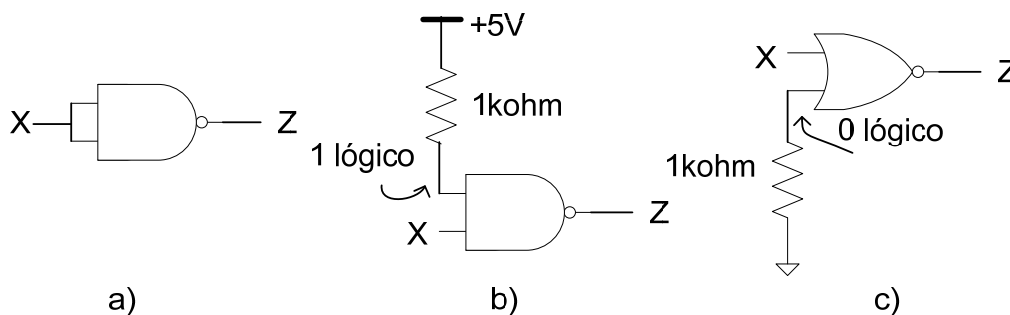
3.5 IMPLEMENTACIÓN DE AXIOMAS DE ÁLGEBRA DE BOOLE (V)

T4 Y T8

$$\text{NOT: } S = a' = (a+a)' = (a+0)'$$



3.5 IMPLEMENTACIÓN DE FUNCIONES CON PUERTAS LÓGICAS BÁSICAS (VI)



ENTRADAS NO UTILIZADAS:

- a) Dos entradas unidas
- b) NAND con entrada de valor alto
- c) NOR con entrada a valor bajo

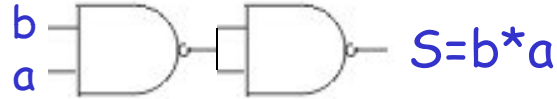
$$Z = \overline{X}$$



3.5 IMPLEMENTACIÓN DE FUNCIONES CON PUERTAS LÓGICAS BÁSICAS (VII)

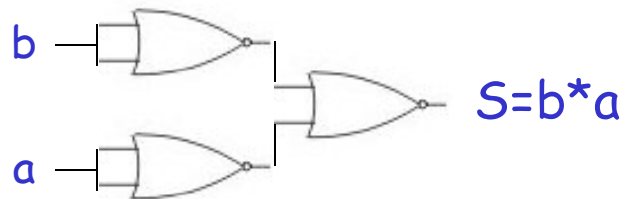
IMPLEMENTACIÓN DE UNA PUERTA LÓGICA CON PUERTAS **NAND**

AND: $S = b * a = ((b * a)')'$

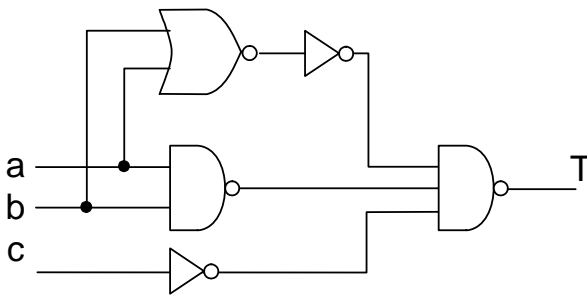


IMPLEMENTACIÓN DE UNA PUERTA LÓGICA CON PUERTAS **NOR**

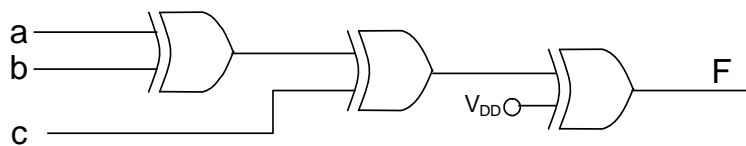
AND: $S = b * a = ((b * a)')' = (b' + a)'$



3.5 IMPLEMENTACIÓN DE FUNCIONES COMBINACIONALES (VIII)



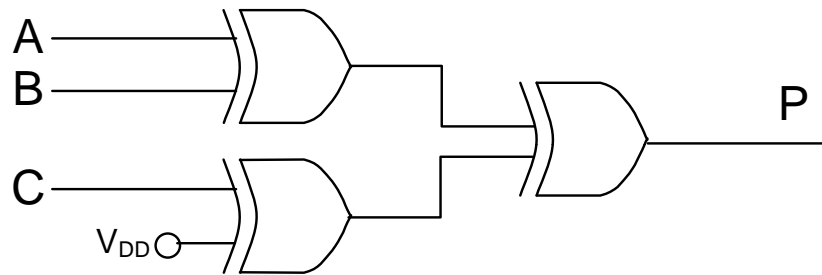
$T(a \ b \ c) = ab + \bar{a}b + c$



$F(a \ b \ c) = ((a \oplus b) \oplus c) \oplus 1 = \overline{a \oplus b \oplus c}$



3.5 IMPLEMENTACIÓN DE FUNCIONES COMBINACIONALES (IX)



$$P(A, B, C) = (A \oplus B) \oplus (C \oplus 1)$$



3.5 IMPLEMENTACIÓN DE FUNCIONES COMBINACIONALES (X)

Implementación con puertas AND y OR con cualquier n° de entradas, de la función: $F(A, B, C) = AB'C + A'B'C + A'BC + A'B'C + ABC + ABC'$

• $F(A, B, C) = \Sigma (1, 3, 5, 6, 7)$

• 2 niveles de puertas sin contar los inversores

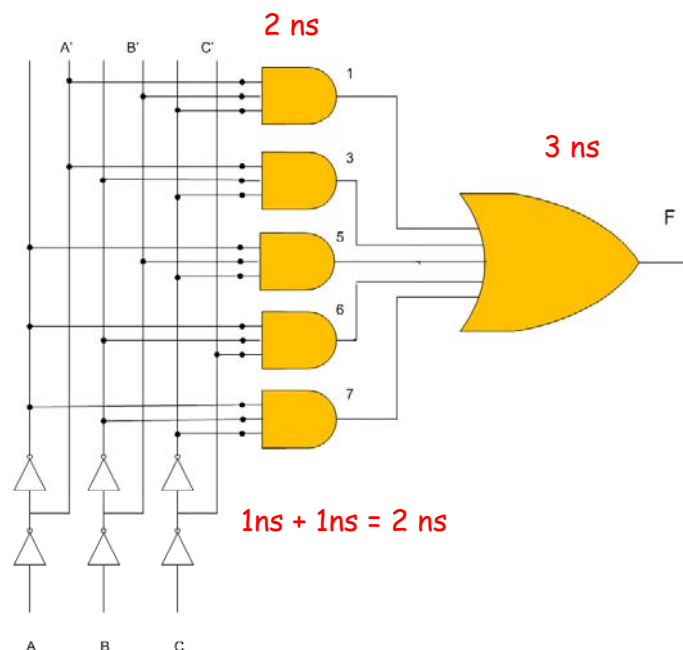
• Suponiendo que los retardos asociados a las puertas son:

Inversores 1ns

AND 2ns

OR 3ns

RETARDO TOTAL: 7 ns



3.5 IMPLEMENTACIÓN DE FUNCIONES COMBINACIONALES (XI)

Implementación , con puertas AND y OR de 2 entradas, de la función:

$$F(A,B,C) = AB'C + A'B'C + A'BC + A'B'C + ABC + ABC'$$

• $F(A,B,C) = \Sigma (1,3,5,6,7)$

• 5 niveles de puertas sin contar los inversores

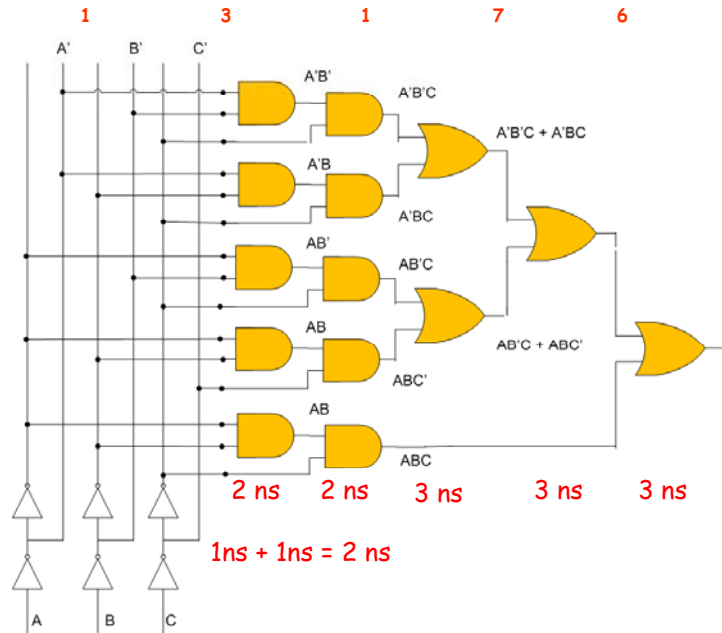
• Suponiendo que los retardos asociados a las puertas son:

Inversores 1ns

AND 2ns

OR 3ns

RETARDO TOTAL: 15 ns



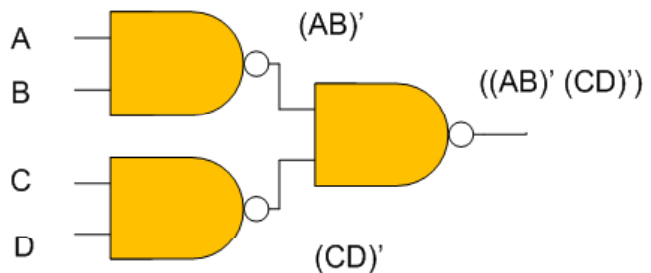
3.5 IMPLEMENTACIÓN DE FUNCIONES COMBINACIONALES (XII)

Sea la función: $F(ABCD) = AB + CD$

IMPLEMENTACIÓN CON PUERTAS NAND

$$F'' = (AB + CD)''$$

$$F = ((AB)' (CD)')$$



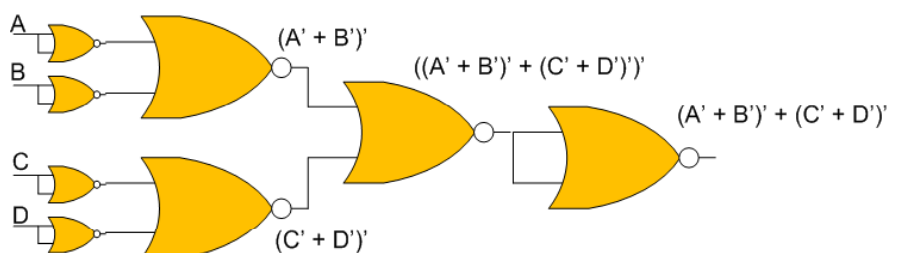
IMPLEMENTACIÓN CON PUERTAS NOR

$$F'' = (AB + CD)''$$

$$F = ((AB)' (CD)')$$

$$F = ((A' + B') (C' + D'))'$$

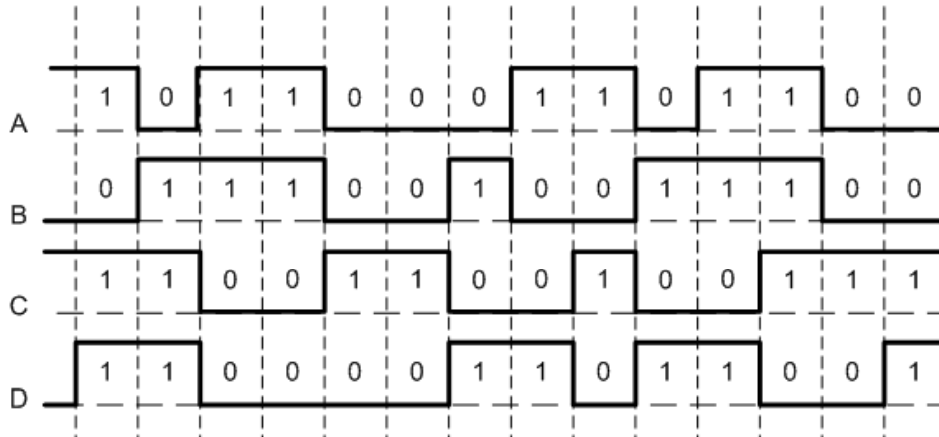
$$F = (A' + B') + (C' + D')$$



3.5 IMPLEMENTACIÓN DE FUNCIONES COMBINACIONALES (XIII)

Sea la función $F(A,B,C,D) = A'B'C + A'BD + ABD' + B'CD' + AB'C'D$

Dibújese la forma de onda de la salida cuando las entradas evolucionan según se indica en la gráfica siguiente:



3.5 IMPLEMENTACIÓN DE FUNCIONES COMBINACIONALES (XIV)

$$F(A,B,C,D) = A'B'C + A'BD + ABD' + B'CD' + AB'C'D$$

términos 2 3 5 7 12 14 2 10 9

$$F = A'B'CD + A'B'CD' + A'BCD + A'BC'D + ABCD' + ABC'D' + AB'CD' + A'B'CD' + AB'C'D$$

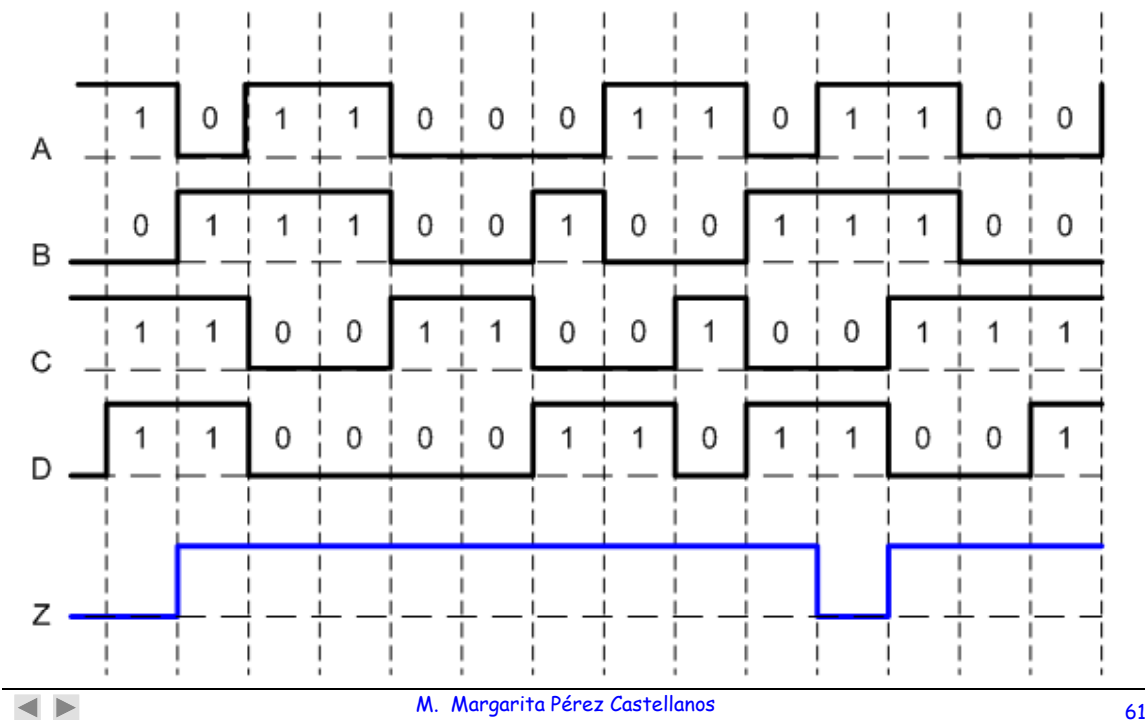
$$F(A,B,C,D) = \Sigma (2,3,5,7,9,10,12,14)$$

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	0



3.5 IMPLEMENTACIÓN DE FUNCIONES COMBINACIONALES (XV)

$$F = A' B' C D + A' B' C D' + A' B C D + A' B C' D + A B C D' + A B C' D' + A B' C D' + A' B' C' D$$



3.5 IMPLEMENTACIÓN DE FUNCIONES COMBINACIONALES (XVI)

EJEMPLOS para trabajo personal. Sean las funciones:

1.- $F(x,y,z,t) = x \cdot (y+z) \cdot t$

2.- $F(a,b,c) = \overline{ab} + ac$

3.- $F(a,b) = a \text{ XOR } b = \overline{a}b + a\overline{b}$

Impleméntense:

- a) Con cualquier tipo de puertas con el n° de entradas que se desee
- b) Con puertas NAND
- c) Con puertas NOR

3.5 IMPLEMENTACIÓN CON LENGUAJES DE ALTO NIVEL LENGUAJES DE DESCRIPCIÓN HARDWARE: Características (XVII)

Descripción y simulación de circuitos combinando diferentes niveles de abstracción



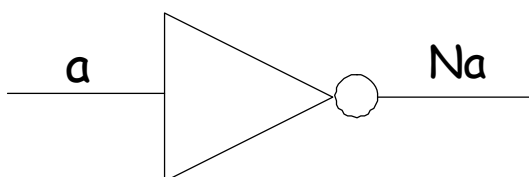
NIVELES DE ABSTRACCION								
Especificaciones	Función: Multiplexor Prestaciones: 3 MHz Limitaciones: Bajo consumo							
Comportamiento	Programa software ejecutable	<pre> PROCESS (entrada,control) BEGIN CASE control IS WHEN "00" => salida <= entrada(0); WHEN "01" => salida <= entrada(1); WHEN "10" => salida <= entrada(2); WHEN "11" => salida <= entrada(3); WHEN OTHERS => salida <= 'X'; END CASE; END PROCESS; END comportamiento; </pre>						
Transferencia entre registros	Estructura de bloques							
Lógico	Tablas de verdad	<table border="1"> <thead> <tr> <th>Control (C)</th> <th>salida</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>E0</td> </tr> <tr> <td>1</td> <td>E1</td> </tr> </tbody> </table>	Control (C)	salida	0	E0	1	E1
Control (C)	salida							
0	E0							
1	E1							
Circuito	Transistores Dispositivos reconfigurables							
Layout	Mascaras y geometría Bits de programación							



3.5 IMPLEMENTACIÓN CON LENGUAJES DE ALTO NIVEL (XVIII)

ESPECIFICACIÓN Y SIMULACIÓN : INVERSOR

El principal dominio de aplicación de los lenguajes de alto nivel es el modelado de dispositivos *hardware*, para la comprobación de su funcionalidad. Dado que es un lenguaje con una semántica orientada a la simulación, posteriormente al modelado, se pueden simular.

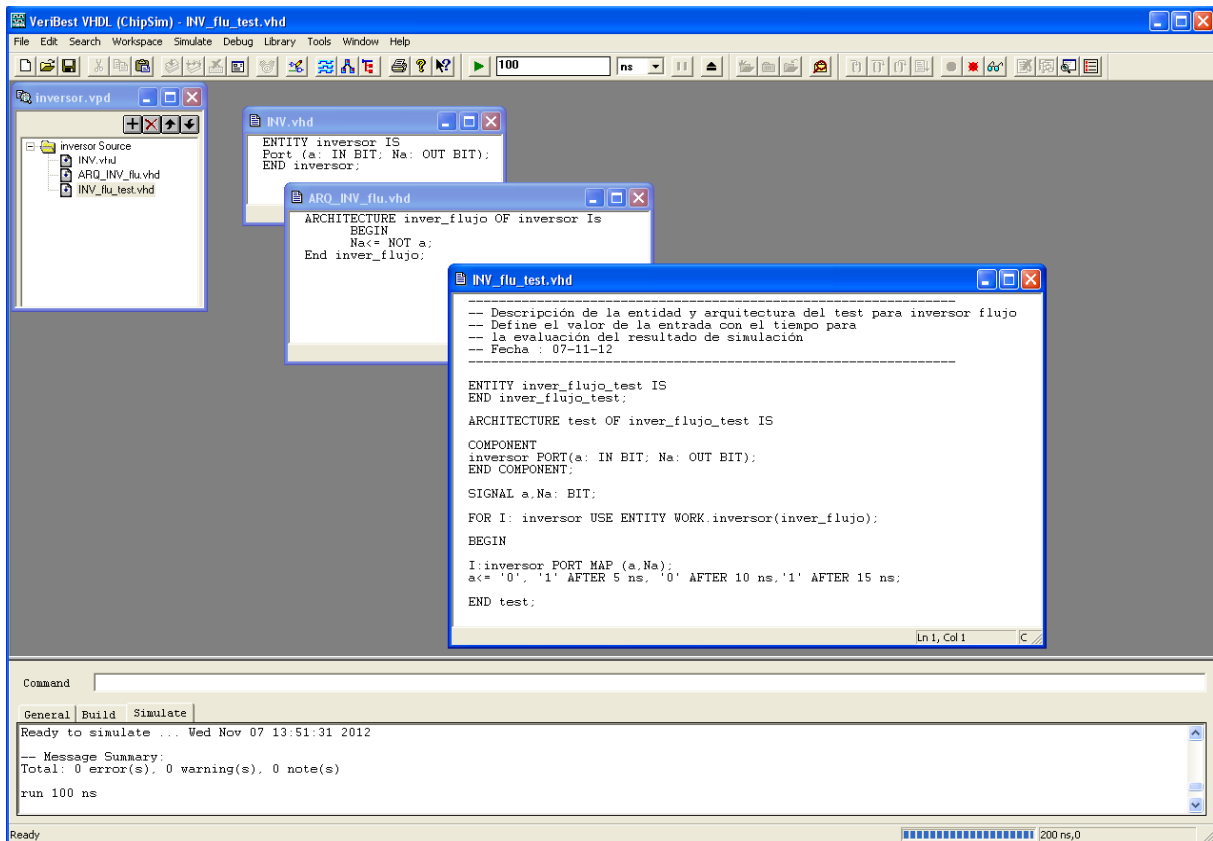


a	Na
0	1
1	0

Herramienta VeriBest VHDL.



3.5 IMPLEMENTACIÓN: LENGUAJE DE ALTO NIVEL (XIX)



```
ENTITY inversor IS
Port (a: IN BIT; Na: OUT BIT);
END inversor;

ARCHITECTURE inver_flujo OF inversor IS
BEGIN
Na<= NOT a;
End inver_flujo;

-- Descripción de la entidad y arquitectura del test para inversor flujo
-- Define el valor de la entrada con el tiempo para
-- la evaluación del resultado de simulación
-- Fecha : 07-11-12

ENTITY inver_flujo_test IS
END inver_flujo_test;

ARCHITECTURE test OF inver_flujo_test IS

COMPONENT
inversor PORT(a: IN BIT; Na: OUT BIT);
END COMPONENT;

SIGNAL a,Na: BIT;

FOR I: inversor USE ENTITY WORK inversor(inver_flujo);

BEGIN

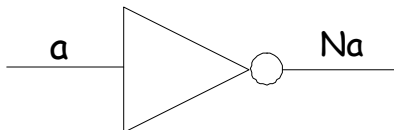
I:inversor PORT MAP (a,Na);
a<= '0', '1' AFTER 5 ns, '0' AFTER 10 ns, '1' AFTER 15 ns;

END test;
```

Ready to simulate ... Wed Nov 07 13:51:31 2012
-- Message Summary:
Total: 0 error(s), 0 warning(s), 0 note(s)
run 100 ns



3.5 IMPLEMENTACIÓN: LENGUAJE DE ALTO NIVEL (XX) ESPECIFICACIÓN DEL INVERSOR



a	Na
0	1
1	0

```
ENTITY inversor IS
Port (a: IN BIT; Na: OUT BIT);
END inversor;
```

```
ARCHITECTURE inver_flujo OF inversor IS
BEGIN
Na<= NOT a;
END inver_flujo;
```



3.5 IMPLEMENTACIÓN: LENGUAJE DE ALTO NIVEL (XXI)

ESPECIFICACIÓN del INVERSOR: FICHERO DE TEST

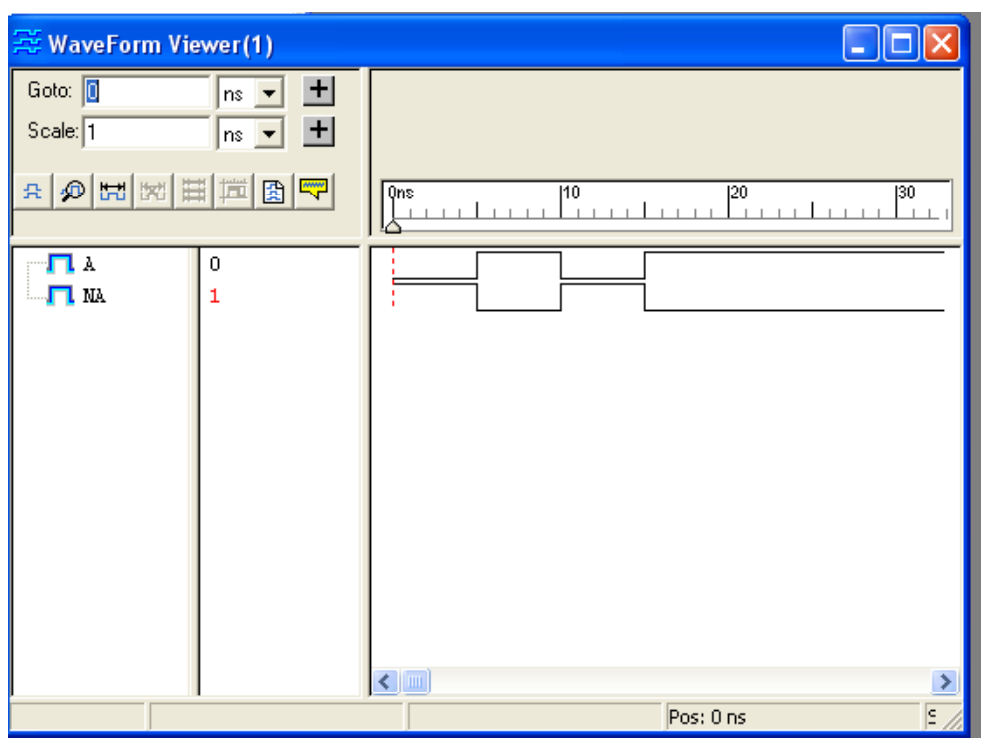
```
ENTITY inver_flujo_test IS
END inver_flujo_test;

ARCHITECTURE test OF inver_flujo_test IS
COMPONENT
inversor PORT(a: IN BIT; Na: OUT BIT);
END COMPONENT;
SIGNAL a, Na: BIT;
FOR I: inversor USE ENTITY WORK.inversor(inver_flujo);
BEGIN
I:inversor PORT MAP (a, Na);
a<='0', '1' AFTER 5 ns, '0' AFTER 10 ns, '1' AFTER 15 ns;
END test;
```

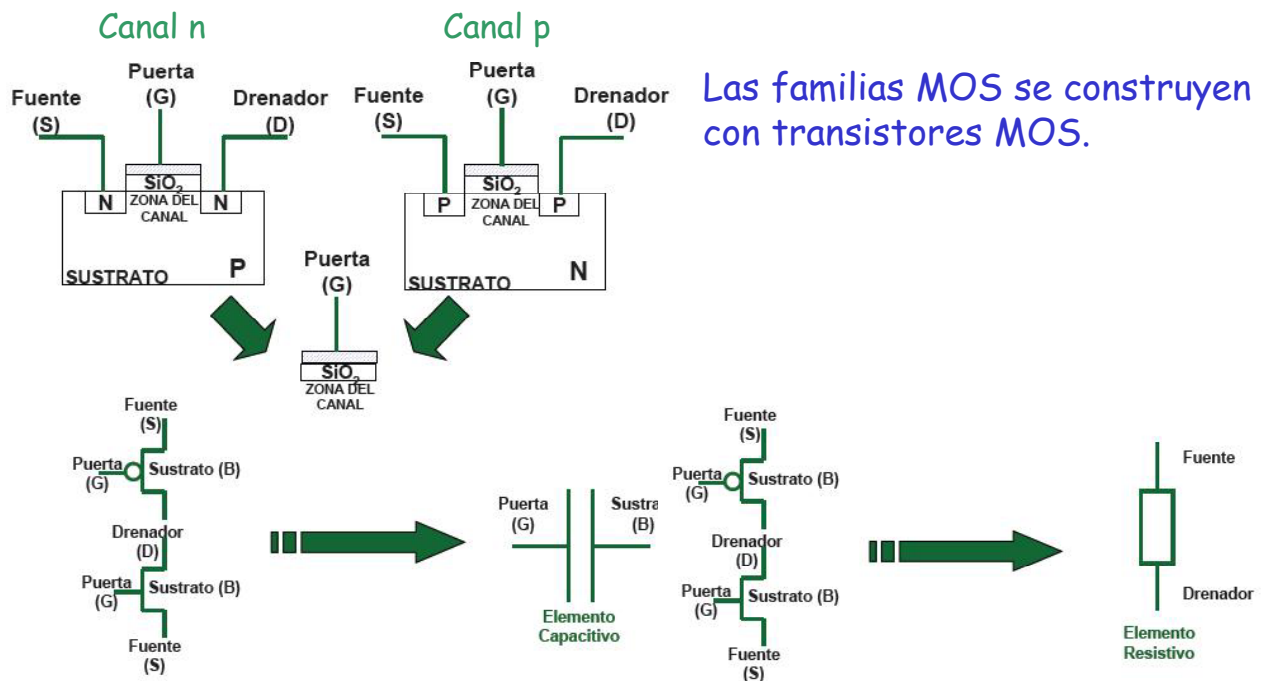


3.5 IMPLEMENTACIÓN: LENGUAJE DE ALTO NIVEL (XXII)

SIMULACIÓN DEL INVERSOR: CRONOGRAMA

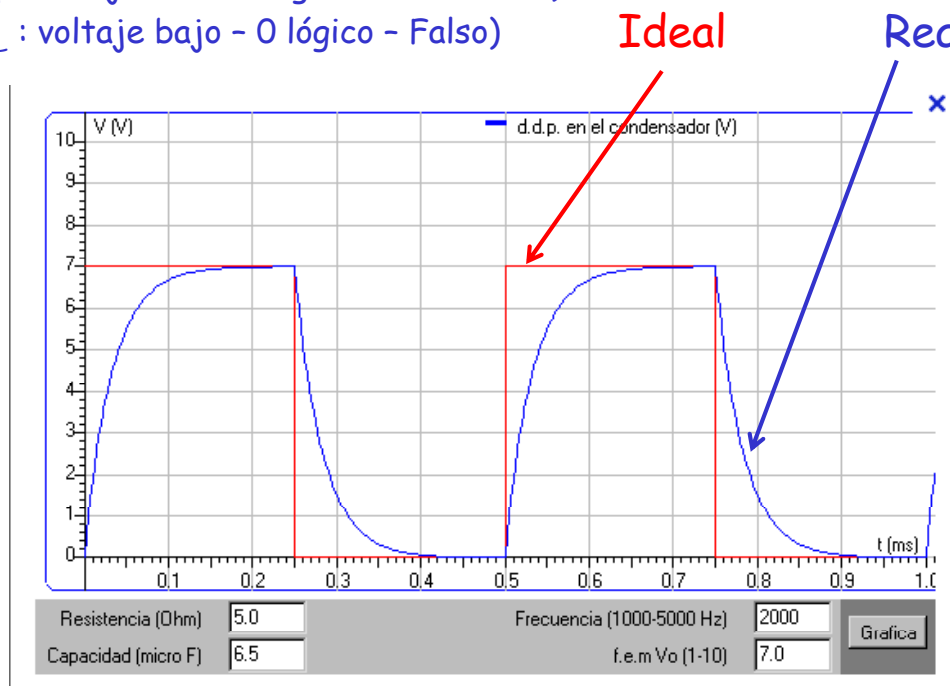


3.5 IMPLEMENTACIÓN MEDIANTE TECNOLOGÍAS MOS (XXIII) (REPASO de FFyTI) (I)



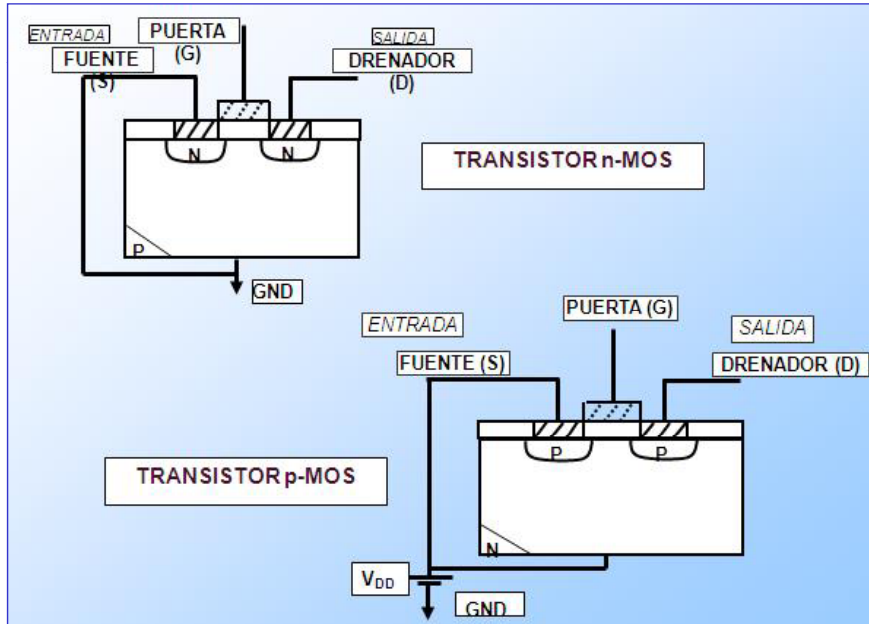
3.5 IMPLEMENTACIÓN MEDIANTE TECNOLOGÍAS MOS (XXIV) (REPASO de FFyTI) (II)

MODELO IDEAL DE TRANSISTOR: Las señales se consideran discretas
 ($V_{dd} - V_H$: voltaje alto - 1 lógico - Verdadero)
 ($GND - V_L$: voltaje bajo - 0 lógico - Falso)

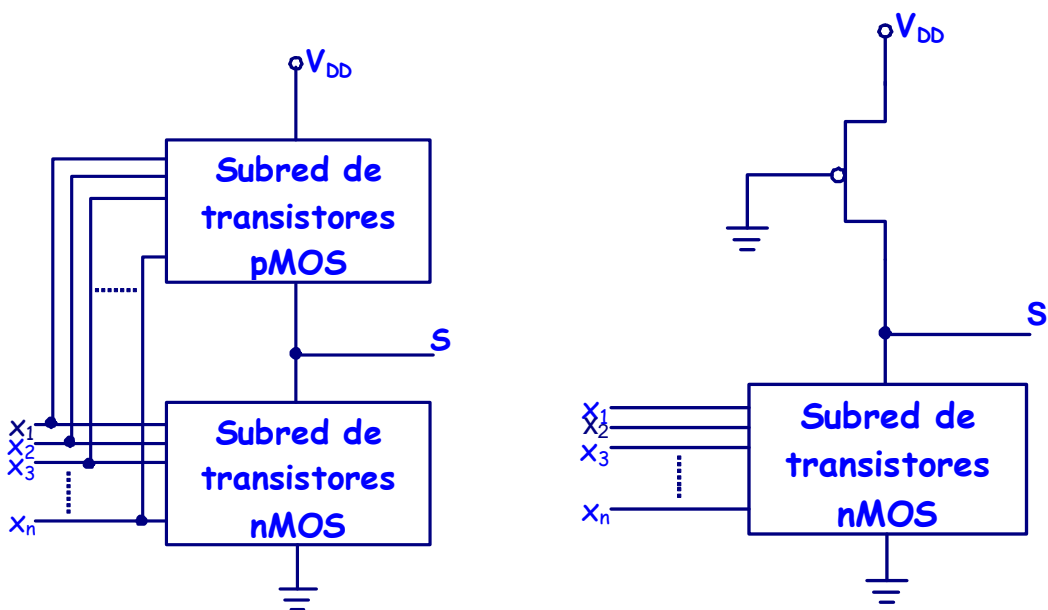


3.5 IMPLEMENTACIÓN MEDIANTE TECNOLOGÍAS MOS (XXV) (REPASO de FFyTI) (III)

La familia CMOS, se construye con utilizando el mismo número de transistores de canal p que de canal n.



3.5 IMPLEMENTACIÓN MEDIANTE TECNOLOGÍAS MOS (XXVI) (REPASO de FFyTI) (IV)

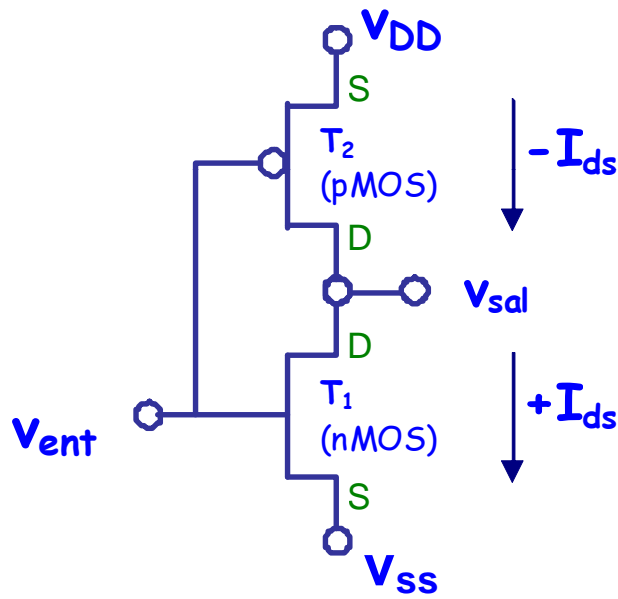


Tecnología CMOS

Tecnología pseudo-nMOS



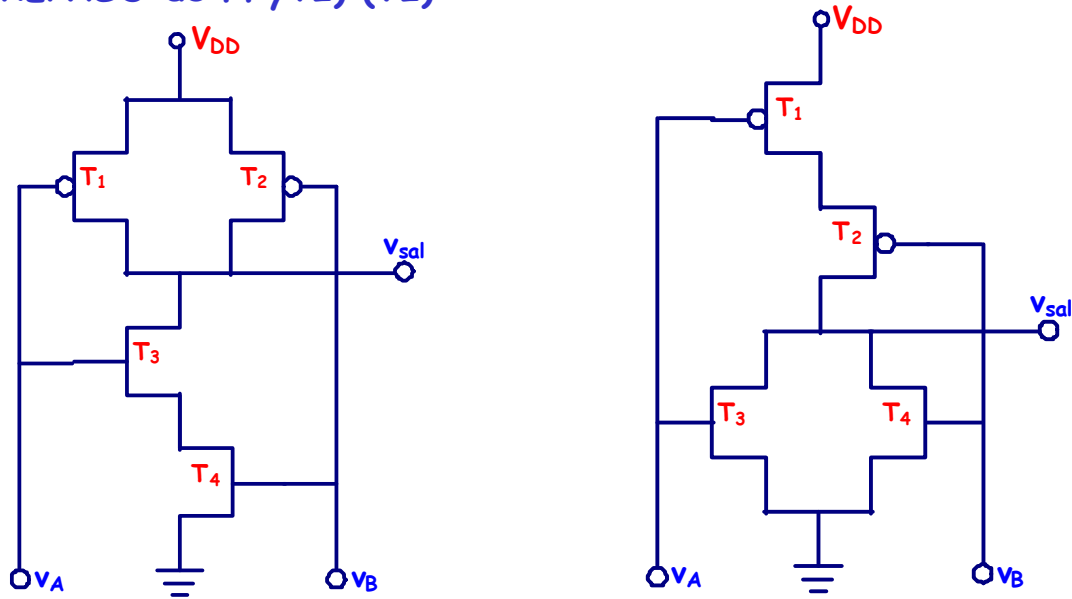
3.5 IMPLEMENTACIÓN MEDIANTE TECNOLOGÍAS MOS (XXVII) (REPASO de FFyTI) (V)



INVERSOR CMOS



3.5 IMPLEMENTACIÓN MEDIANTE TECNOLOGÍAS MOS (XXVIII) (REPASO de FFyTI) (VI)

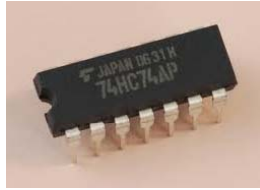


Puerta NAND

Puerta NOR



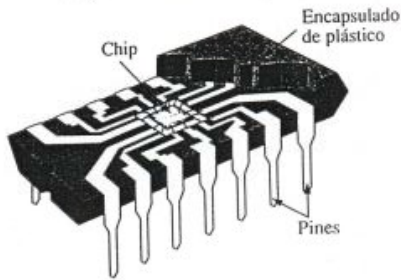
3.5 IMPLEMENTACIÓN MEDIANTE CIRCUITOS INTEGRADOS (XXIX)



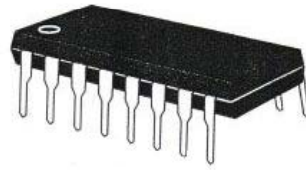
CIRCUITOS DIGITALES INTEGRADOS

Realizan funciones lógicas.

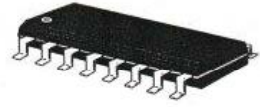
- Características:
- tamaño reducido
 - alta fiabilidad
 - bajo coste
 - bajo consumo de potencia



ENCAPSULADOS DE CI



(a) Encapsulado DIP



(b) Encapsulado SOIC

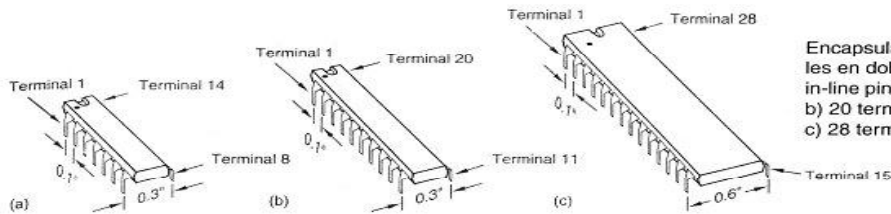
NUMERACIÓN DE LOS PINES



DIP o SOIC



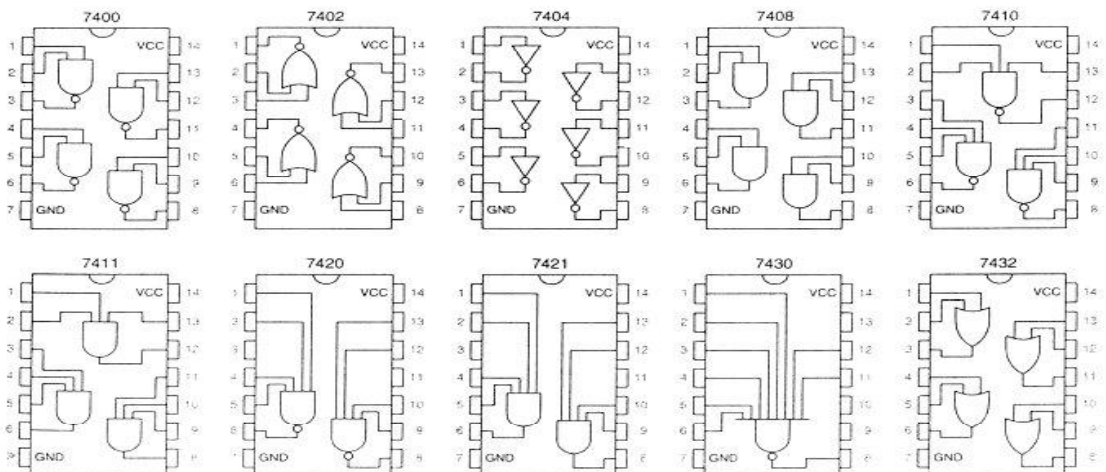
3.5 IMPLEMENTACIÓN MEDIANTE CIRCUITOS INTEGRADOS (XXX)



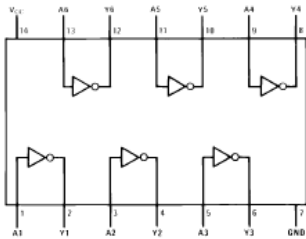
Encapsulados con terminales en doble línea (DIP, dual in-line pin): a) 14 terminales; b) 20 terminales; c) 28 terminales.



Diagramas de terminales para varios CIs SSI de la serie 7400.

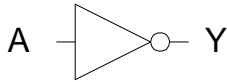


3.5 IMPLEMENTACIÓN MEDIANTE CIRCUITOS INTEGRADOS (XXXI): EL INVERSOR 74HC04 (III)

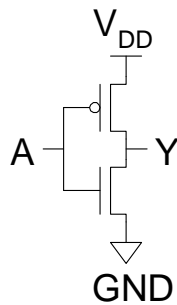
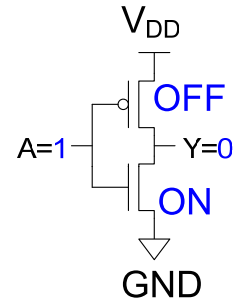


•6 INVERSORES

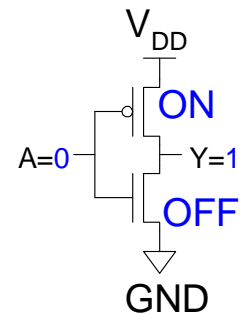
A	Y
0	1
1	0



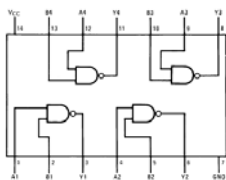
A	Y
1	0



A	Y
0	1

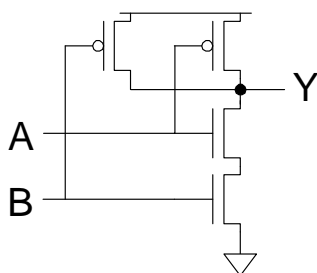
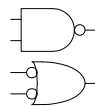


3.5 IMPLEMENTACIÓN MEDIANTE CIRCUITOS INTEGRADOS (XXXII): LA PUERTA NAND 74HC00 (IV)

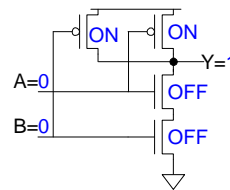


•4 puertas NAND de
•2 entradas

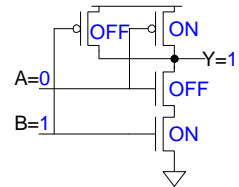
A	B	Y
0	0	
0	1	
1	0	
1	1	



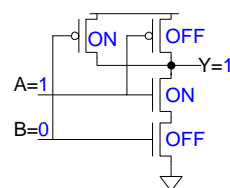
A	B	Y
0	0	1
0	1	
1	0	
1	1	



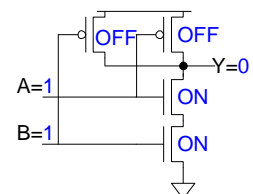
A	B	Y
0	0	1
0	1	1
1	0	
1	1	



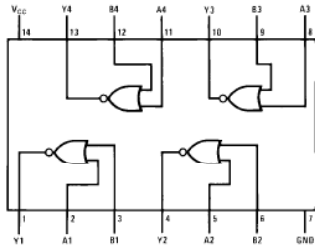
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	



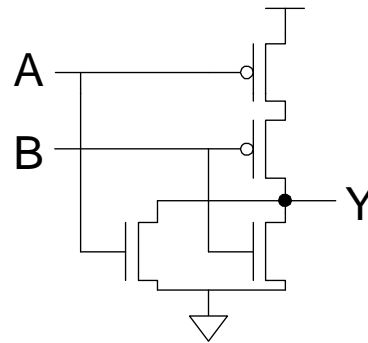
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0



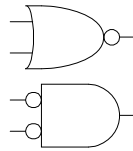
3.5 IMPLEMENTACIÓN MEDIANTE CIRCUITOS INTEGRADOS (XXXIII): LA PUERTA NOR 74HC02 (V)



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



- 4 puertas NOR de
- 2 entradas



3.5 IMPLEMENTACIÓN MEDIANTE CIRCUITOS INTEGRADOS (XXXIV):

EL INVERSOR 74HC04 (VI)

TC74HC04AP, TC74HC04AF, TC74HC04AFN

Hex Inverter

The TC74HC04A is a high speed CMOS INVERTER fabricated with silicon gate C²MOS technology.

It achieves the high speed operation similar to equivalent LSTTL while maintaining the CMOS low power dissipation.

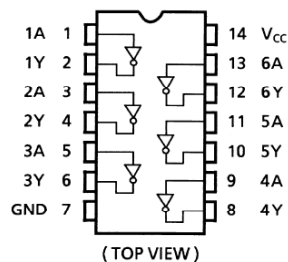
The internal circuit is composed of 3 stages, including buffered output, which provide high noise immunity and stable output.

All inputs are equipped with protection circuits against static discharge or transient excess voltage.

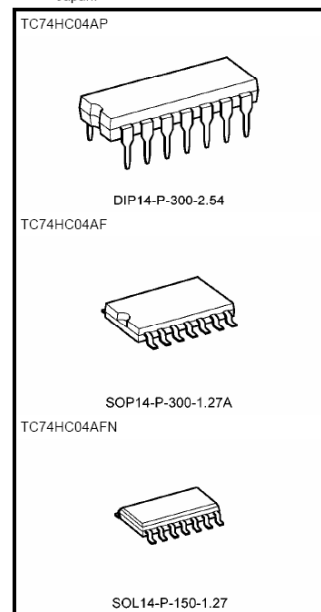
Features

- High speed: $t_{pd} = 6 \text{ ns (typ.)}$ at $V_{CC} = 5 \text{ V}$
- Low power dissipation: $I_{CC} = 1 \mu\text{A (max)}$ at $T_a = 25^\circ\text{C}$
- High noise immunity: $V_{NIH} = V_{NIL} = 28\% V_{CC} \text{ (min)}$
- Output drive capability: 10 LSTTL loads
- Symmetrical output impedance: $|I_{OH}| = I_{OL} = 4 \text{ mA (min)}$
- Balanced propagation delays: $t_{pLH} \approx t_{pHL}$
- Wide operating voltage range: $V_{CC} \text{ (opr)} = 2 \text{ to } 6 \text{ V}$
- Pin and function compatible with 74LS04

Pin Assignment



Note: xxxFN (JEDEC SOP) is not available in Japan.



Weight	
DIP14-P-300-2.54	: 0.96 g (typ.)
SOP14-P-300-1.27A	: 0.18 g (typ.)
SOL14-P-150-1.27	: 0.12 g (typ.)



3.5 IMPL. MEDIANTE CIRCUITOS INTEGRADOS (XXXIV): EL INVERSOR 74HC04 (VII)

Absolute Maximum Ratings (Note 1)

Characteristics	Symbol	Rating	Unit
Supply voltage range	V_{CC}	-0.5 to 7	V
DC input voltage	V_{IN}	-0.5 to $V_{CC} + 0.5$	V
DC output voltage	V_{OUT}	-0.5 to $V_{CC} + 0.5$	V
Input diode current	I_{IK}	± 20	mA
Output diode current	I_{OK}	± 20	mA
DC output current	I_{OUT}	± 25	mA
DC V_{CC} /ground current	I_{CC}	± 50	mA
Power dissipation	P_D	500 (DIP) (Note 2)/180 (SOP)	mW
Storage temperature	T_{stg}	-65 to 150	$^{\circ}C$

Operating Ranges (Note)

Characteristics	Symbol	Rating	Unit
Supply voltage	V_{CC}	2 to 6	V
Input voltage	V_{IN}	0 to V_{CC}	V
Output voltage	V_{OUT}	0 to V_{CC}	V
Operating temperature	T_{opr}	-40 to 85	$^{\circ}C$
Input rise and fall time	t_r, t_f	0 to 1000 ($V_{CC} = 2.0$ V) 0 to 500 ($V_{CC} = 4.5$ V) 0 to 400 ($V_{CC} = 6.0$ V)	ns



3.5 IMPLEMENTACIÓN MEDIANTE CIRCUITOS INTEGRADOS (XXXVI): EL INVERSOR 74HC04 (VIII)

Electrical Characteristics

DC Characteristics

Characteristics	Symbol	Test Condition	$T_a = 25^{\circ}C$			$T_a = -40$ to $85^{\circ}C$		Unit		
			V_{CC} (V)	Min	Typ.	Max	Min		Max	
High-level input voltage	V_{IH}	—	2.0	1.50	—	—	1.50	—	V	
			4.5	3.15	—	—	3.15	—		
			6.0	4.20	—	—	4.20	—		
Low-level input voltage	V_{IL}	—	2.0	—	—	0.50	—	0.50	V	
			4.5	—	—	1.35	—	1.35		
			6.0	—	—	1.80	—	1.80		
High-level output voltage	V_{OH}	$V_{IN} = V_{IH}$ or V_{IL}	$I_{OH} = -20 \mu A$	2.0	1.9	2.0	—	1.9	—	V
				4.5	4.4	4.5	—	4.4	—	
			$I_{OH} = -4$ mA	4.5	4.18	4.31	—	4.13	—	
				6.0	5.68	5.80	—	5.63	—	
Low-level output voltage	V_{OL}	$V_{IN} = V_{IH}$ or V_{IL}	$I_{OL} = 20 \mu A$	2.0	—	0.0	0.1	—	0.1	V
				4.5	—	0.0	0.1	—	0.1	
			$I_{OL} = 4$ mA	4.5	—	0.17	0.26	—	0.33	
				6.0	—	0.18	0.26	—	0.33	
Input leakage current	I_{IN}	$V_{IN} = V_{CC}$ or GND	6.0	—	—	± 0.1	—	± 1.0	μA	
Quiescent supply current	I_{CC}	$V_{IN} = V_{CC}$ or GND	6.0	—	—	1.0	—	10.0	μA	



3.5 IMPLEMENTACIÓN MEDIANTE CIRCUITOS INTEGRADOS (XXXVII): EL INVERSOR 74HC04 (IX)

AC Characteristics ($C_L = 50 \text{ pF}$, input: $t_r = t_f = 6 \text{ ns}$)

Characteristics	Symbol	Test Condition	Ta = 25°C			Ta = -40 to 85°C		Unit	
			VCC (V)	Min	Typ.	Max	Min		Max
Output transition time	t_{TLH} t_{THL}	—	2.0	—	30	75	—	95	ns
			4.5	—	8	15	—	19	
			6.0	—	7	13	—	16	
Propagation delay time	t_{pLH} t_{pHL}	—	2.0	—	27	75	—	95	ns
			4.5	—	9	15	—	19	
			6.0	—	8	13	—	16	
Input capacitance	C_{IN}	—	—	5	10	—	10	pF	
Power dissipation capacitance	C_{PD} (Note)	—	—	20	—	—	—	pF	

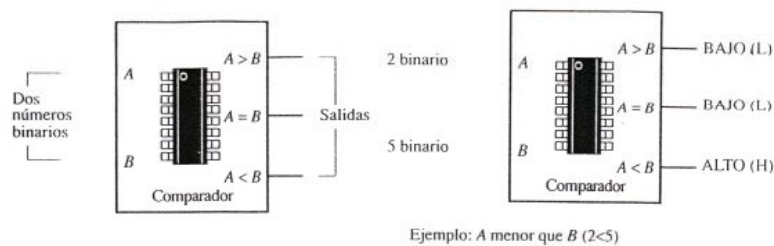
Note: C_{PD} is defined as the value of the internal equivalent capacitance which is calculated from the operating current consumption without load.



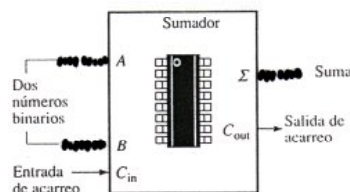
3.5 IMPLEMENTACIÓN MEDIANTE CIRCUITOS INTEGRADOS (XXXVIII) (X)

FUNCIONES LÓGICAS BÁSICAS

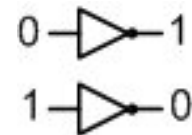
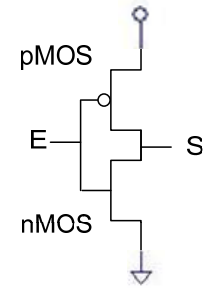
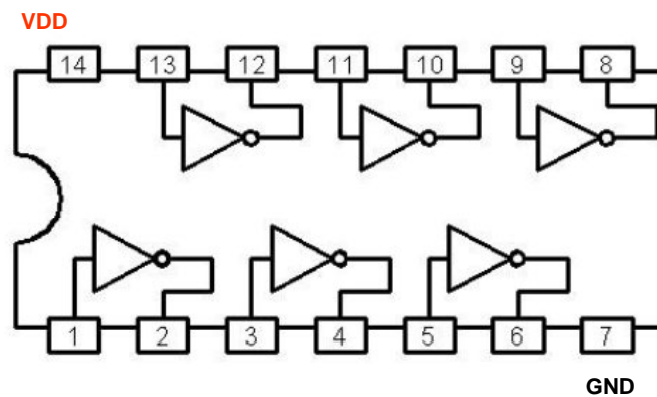
FUNCIÓN DE COMPARACIÓN



FUNCIONES ARITMÉTICAS



3.5 IMPLEMENTACIÓN: ENTORNO DE CONSTRUCCIÓN Y SIMULACIÓN (XXXIX): EL INVERSOR CMOS (I)



Constructor Virtual y Simulador de Circuitos Digitales 0.9.7



4. ESTRUCTURAS COMBINACIONALES BÁSICAS (I)

- Puertas lógicas básicas*.
- Multiplexores y demultiplexores.
- Codificadores y decodificadores.
- Comparadores.

• ver transparencias 47 y 48



4.2 MULTIPLEXORES (I)

Un multiplexor de información o selector de datos, es un circuito lógico combinacional que acepta varias entradas de datos y selecciona ("transmite") solamente una de ellas hasta su salida, dependiendo del valor de una señal de control.

Coloca en la salida la misma señal que haya en la entrada QUE SELECCIONAN las señales de control

ENTRADA DE DATOS. N° de entradas: n

ENTRADAS DE CONTROL. N° de entradas: $p \rightarrow 2^p \geq n$
 $2^p = n$ (en el caso más simple)

SALIDA (O SALIDAS). N° de salidas: 1



4.2 MULTIPLEXORES (II)

EJEMPLO para 2 entradas

Bloque funcional

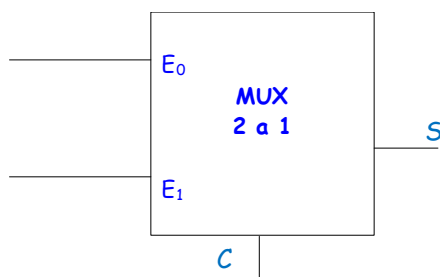


Tabla de verdad

C	S
0	E_0
1	E_1

Función Lógica

$$S (\text{MUX}_{2:1}) = E_0 C' + E_1 C$$

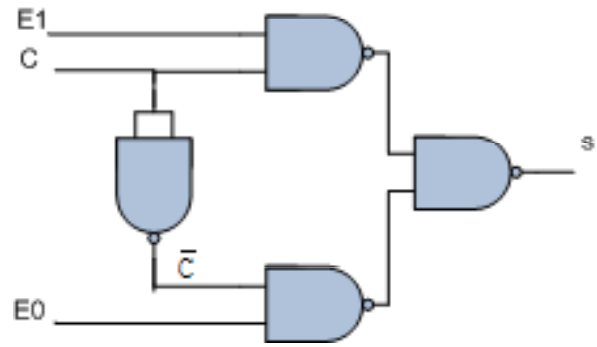
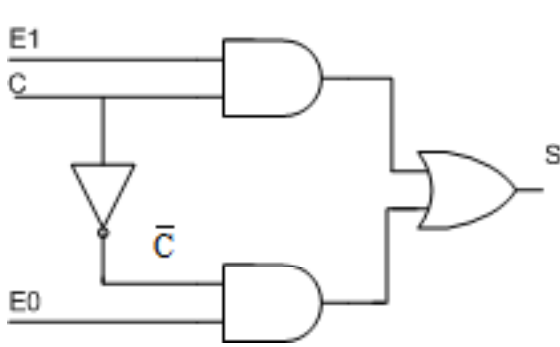


4.2 MULTIPLEXORES (III)

EJEMPLO para 2 entradas

Implementación con puertas lógicas

$$S = E_0 C' + E_1 C$$

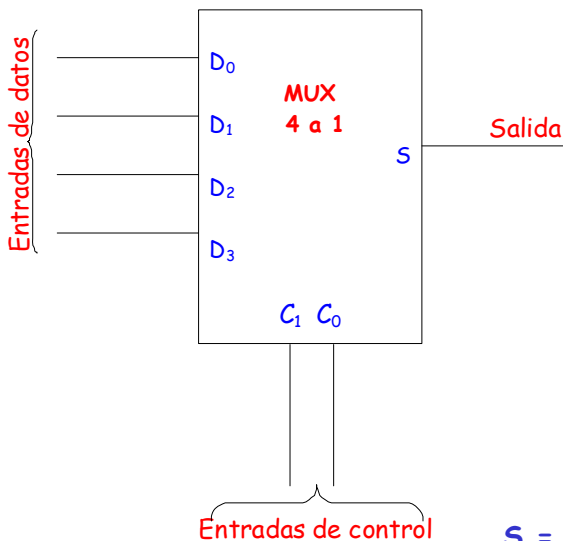


4.2 MULTIPLEXORES (IV)

EJEMPLO para 4 entradas

Bloque funcional

Tabla de verdad



C_1	C_0	S
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

Función Lógica

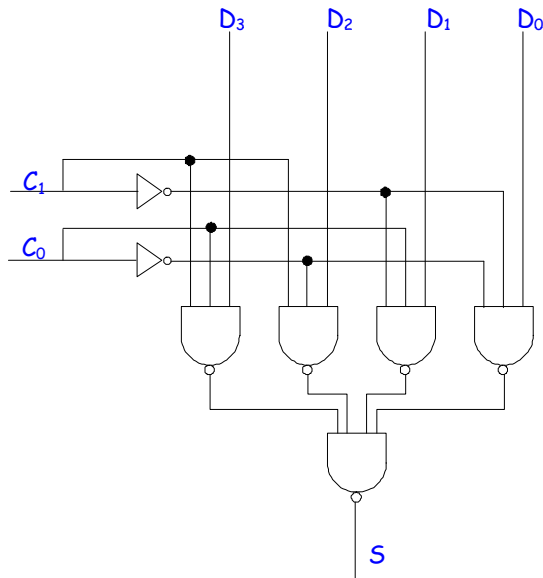
$$S = C_1' C_0' (D_0) + C_1' C_0 (D_1) + C_1 C_0' (D_2) + C_1 C_0 (D_3)$$



4.2 MULTIPLEXORES (V)

EJEMPLO para 4 entradas

Implementación con puertas lógicas (NAND)



Función lógica

$$S = D_0 \bar{C}_1 \bar{C}_0 + D_1 \bar{C}_1 C_0 + D_2 C_1 \bar{C}_0 + D_3 C_1 C_0$$

$$S = \overline{\overline{D_0 \bar{C}_1 \bar{C}_0} \overline{D_1 \bar{C}_1 C_0} \overline{D_2 C_1 \bar{C}_0} \overline{D_3 C_1 C_0}}$$

$$S = \overline{\overline{D_0 \bar{C}_1 \bar{C}_0} \overline{D_1 \bar{C}_1 C_0} \overline{D_2 C_1 \bar{C}_0} \overline{D_3 C_1 C_0}}$$



4.2 MULTIPLEXORES (XI)

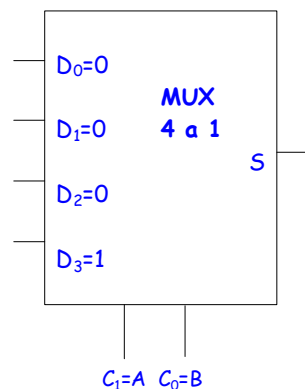
Aplicaciones: implementación de funciones lógicas

Implementación de la función $S = A.B$ (puerta AND) con un MUX 4:1

$$S_{(MUX\ 4:1)} = D_0 (C_1' C_0') + D_1 (C_1' C_0) + D_2 (C_1 C_0') + D_3 (C_1 C_0)$$

$$S_{(MUX\ 4:1)} = A.B = (D_0=0)(C_1' C_0') + (D_1=0)(C_1' C_0) + (D_2=0)(C_1 C_0') + (D_3=1)(C_1 C_0) = (D_0=0)(A' B') + (D_1=0)(A' B) + (D_2=0)(A B') + (D_3=1)(A B) =$$

$C_1(A)$	$C_0(B)$	S
0	0	0
0	1	0
1	0	0
1	1	1



4.2 MULTIPLEXORES (XII)

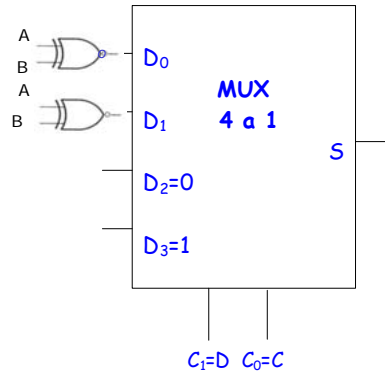
Aplicaciones: implementación de funciones lógicas

$S = (B' A' + BA) D' C' + (B' A + BA') D' C + DC$ directamente con un MUX 4:1

Ecuación general: $\rightarrow S_{(MUX\ 4:1)} = E_0 (C_1' C_0') + E_1 (C_1' C_0) + E_2 (C_1 C_0') + E_3 (C_1 C_0)$

$S_{(MUX\ 4:1)} = (D_0 \Rightarrow (A \oplus B)')(D' C') + (D_1 \Rightarrow A \oplus B)(D' C) + (D_2 \Rightarrow 0)(D C') + (D_3 \Rightarrow 1)(D C)$

$C_1(D)$	$C_0(C)$	S
0	0	$(A \oplus B)'$
0	1	$(A \oplus B)$
1	0	0
1	1	1



4.2 MULTIPLEXORES (XIII)

Aplicaciones: implementación de funciones lógicas

Implementación introduciendo en las entradas del MUX alguna de las variables de la función.

EJEMPLO: $S = A.B$ (AND)

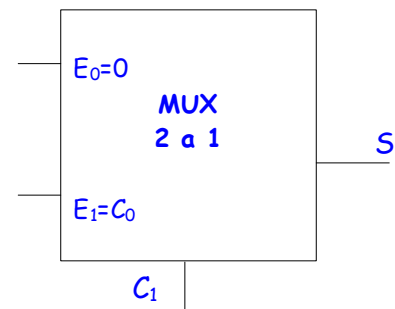
$C_1(A)$	$C_0(B)$	S
0	0	0
0	1	0
1	0	0
1	1	1

$S = 0$ cuando $C_1 = 0$

$S = C_0$ cuando $C_1 = 1$

$S = 0$ cuando $A=0$

$S = B$ cuando $A=1$



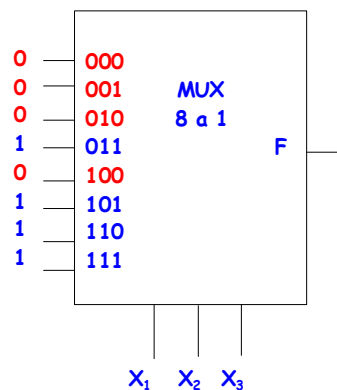
$S_{(MUX\ 2:1)} = (E_0=0) C_1' + (E_1=C_0) C_1 = (E_0=0) A' + (E_1=B) A = AB$



4.2 MULTIPLEXORES (XIV)

Sea la función $F(x_1x_2x_3)$ dada por la tabla de verdad adjunta, se puede construir utilizando un multiplexor con entradas valores constantes (0, 1) y tomando como variables de control ($x_1x_2x_3$)

x_1	x_2	x_3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

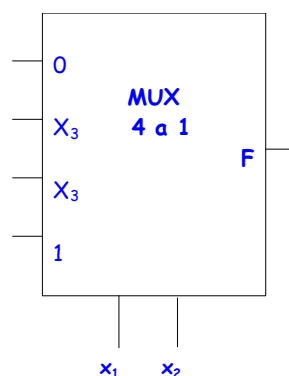


4.2 MULTIPLEXORES (XV)

La misma función $F(x_1x_2x_3)$, se puede construir con entradas que dependan de una variable (x_3) y tomando como variables de control (x_1, x_2):

x_1	x_2	x_3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

$F = 0$ cuando $x_1 = x_2 = 0$
 $F = X_3$ cuando $x_1 = 0$ y $x_2 = 1$
 $F = X_3$ cuando $x_1 = 1$ y $x_2 = 0$
 $F = 1$ cuando $x_1 = x_2 = 1$



4.2 MULTIPLEXORES (XVI)

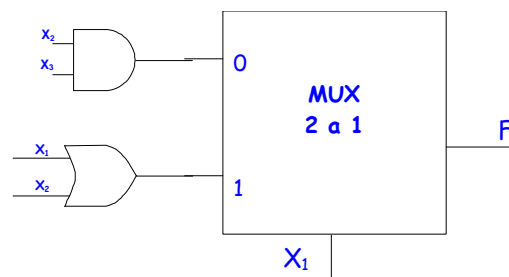
La misma función $F(x_1x_2x_3)$, se puede construir con entradas que dependan de las 2 variables (x_2 x_3) y tomando como variable de control (x_1):

x_1	x_2	x_3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

F será función de x_2 y de x_3

$$F = x_2 x_3 \text{ cuando } x_1 = 0$$

$$F = x_1 + x_2 \text{ cuando } x_1 = 1$$



4.2 MULTIPLEXORES (XVII):

TEOREMA DE EXPANSIÓN DE SHANNON

$$F(A_1, A_2, \dots, A_N) = \overline{A_1} \cdot F(0, A_2, \dots, A_N) + A_1 \cdot F(1, A_2, \dots, A_N)$$

$$F(A_1, A_2, \dots, A_N) = (\overline{A_1} + F(0, A_2, \dots, A_N)) \cdot (A_1 + F(1, A_2, \dots, A_N))$$

Si se emplea repetidamente el teorema de expansión de *Shannon*, hasta agotar todas las variables, nos proporciona directamente el desarrollo de una función como:

SUMA DE TÉRMINOS PRODUCTO (*MINTERMS*) o como
PRODUCTO DE TÉRMINOS SUMA (*MAXTERMS*).



4.2 MULTIPLEXORES (XVIII):

TEOREMA DE EXPANSIÓN DE SHANNON

Sabemos que:

$$\begin{aligned} F(A_1, A_2) &= \bar{A}_1 \cdot F(0, A_2) + A_1 \cdot F(1, A_2) = \\ &= \bar{A}_1 \cdot \bar{A}_2 \cdot F(0, 0) + \bar{A}_1 \cdot A_2 \cdot F(0, 1) + A_1 \cdot \bar{A}_2 \cdot F(1, 0) + A_1 \cdot A_2 \cdot F(1, 1) \end{aligned}$$

Hay que tener en cuenta que:

- $\bar{A}_1 \cdot F(0, A_2) = \bar{A}_1 \cdot \bar{A}_2 \cdot F(0, 0) + \bar{A}_1 \cdot A_2 \cdot F(0, 1)$
- $A_1 \cdot F(1, A_2) = A_1 \cdot \bar{A}_2 \cdot F(1, 0) + A_1 \cdot A_2 \cdot F(1, 1)$
- $F(0, 0)$, $F(0, 1)$, $F(1, 0)$ y $F(1, 1)$ solamente podrán tomar los valores: '1' o '0'.



4.2 MULTIPLEXORES (XIX):

TEOREMA DE EXPANSIÓN DE SHANNON

Ejemplo:

Si para el caso anterior tenemos la siguiente tabla de verdad...

X_1	X_2	$F(X_1, X_2)$
0	0	0
0	1	1
1	0	1
1	1	0

$$\begin{aligned} F(X_1, X_2) &= \bar{X}_1 \cdot \bar{X}_2 \cdot F(0, 0) + \bar{X}_1 \cdot X_2 \cdot F(0, 1) + X_1 \cdot \bar{X}_2 \cdot F(1, 0) + X_1 \cdot X_2 \cdot F(1, 1) = \\ &= \bar{X}_1 \cdot \bar{X}_2 \cdot (0) + \bar{X}_1 \cdot X_2 \cdot (1) + X_1 \cdot \bar{X}_2 \cdot (1) + X_1 \cdot X_2 \cdot 0 = \bar{X}_1 \cdot X_2 + X_1 \cdot \bar{X}_2 \end{aligned}$$



4.2 MULTIPLEXORES (XX)

o Generación de funciones lógicas

- Un multiplexor con N entradas de control permite generar cualquier función lógica de N+1 variables.
- Siguiendo el teorema de Shannon:

$$\begin{aligned}
 F(X_1, X_2, \dots, X_N, X_M) &= \overline{X_1} F(0, X_2, \dots, X_M) + X_1 F(1, X_2, \dots, X_M) = \\
 &= \overline{X_1} \overline{X_2} \dots \overline{X_N} F(0, 0, \dots, 0, X_M) + X_1 \overline{X_2} \dots \overline{X_N} F(1, 0, \dots, 0, X_M) + \dots + \\
 &+ X_1 X_2 \dots X_N F(1, 1, \dots, 1, X_M) = \\
 &= \overline{X_1} \overline{X_2} \dots \overline{X_N} G_0(X_M) + X_1 \overline{X_2} \dots \overline{X_N} G_1(X_M) + \dots + X_1 X_2 \dots X_N G_{2^N-1}(X_M)
 \end{aligned}$$

- Las funciones $G_i(X_M)$ serán del tipo: $\{1, 0, X_M, \overline{X_M}\}$

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

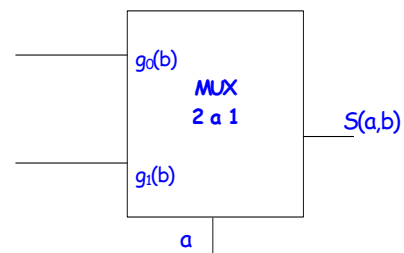


4.2 MULTIPLEXORES (XXI):

GENERADOR UNIVERSAL: EJEMPLO DE IMPLEMENTACIÓN PARA FUNCIONES DE DOS VARIABLES

Función S (a,b)	Puerta Lógica	Entradas g ₀ g ₁
ab	AND	0 b
a + b	OR	b 1
a' b'	NOR	b' 0
a' + b'	NAND	1 b'
ab' + a' b	X-OR	b b'
ab + a' b'	X-NOR	b' b

$$\begin{aligned}
 S(ab) &= \overline{a} S(0 b) + a S(1 b) \\
 &= \overline{a} g_0 + a g_1
 \end{aligned}$$

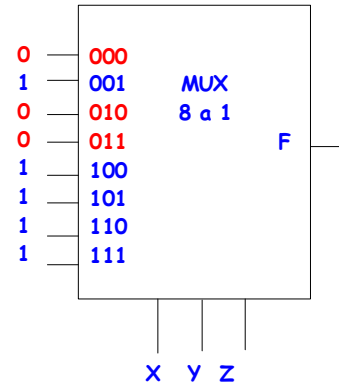


4.2 MULTIPLEXORES (XXII):

Sea la función: $F = X + \overline{Y}Z$,

1. Construimos la tabla de verdad que representa a esta función
2. Construimos una matriz que la implementa, tomando X, Y y Z como variables de control

X	Y	Z	F (X Y Z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1



4.2 MULTIPLEXORES (XXIII):

Sea la función: $F = X + \overline{Y}Z$

3. Desarrollamos por el teorema de Shannon en la variable Z; las entradas al multiplexor dependerán de la variable Z y se toman como variables de control X e Y

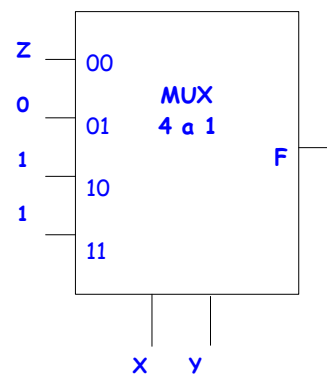
$$\begin{aligned}
 F(X Y Z) &= \overline{X}\overline{Y}F(0 0 Z) + \overline{X}YF(0 1 Z) + X\overline{Y}F(1 0 Z) + XYF(1 1 Z) = \\
 &= \overline{X}\overline{Y}(Z) + \overline{X}Y(0) + X\overline{Y}(1) + XY(1)
 \end{aligned}$$

Siendo $F(00Z) = z$

$F(01Z) = 0$

$F(10Z) = 1$

$F(11Z) = 1$



4.2 MULTIPLEXORES (XXIV):

Ejemplo:

$$\text{Función } F = X + \overline{Y}Z$$

La negada de dicha función es:

$$F = \overline{X} (Y + \overline{Z})$$

- Cuando 2 puertas estén situadas en serie → producto booleano.
- Cuando 2 puertas estén situadas en paralelo → suma booleana.
- Casos con salida a "1" separados de casos con salida a "0".

Restricciones

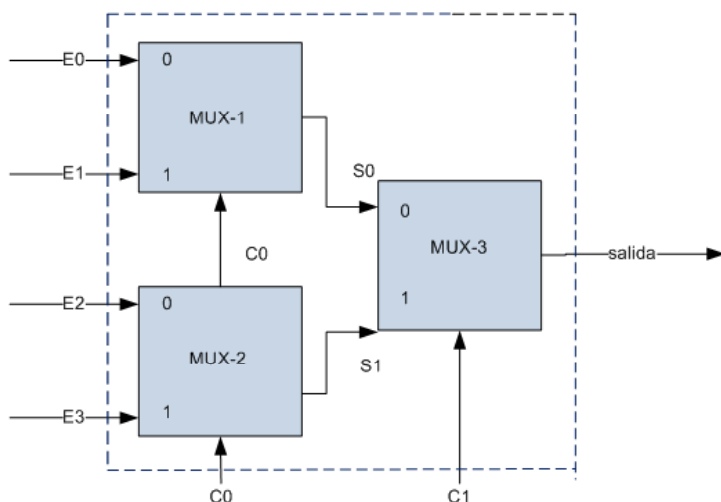
1. Un nodo no puede estar sometido simultáneamente a 2 tensiones diferentes. (no puede haber conducción por dos o más líneas simultáneamente)
2. No puede haber ninguna combinación de las entradas para cual la salida no esté definida. (siempre ha de haber conducción al menos por una línea)



4.2 MULTIPLEXORES (XXV)

EXPANSIÓN DE MULTIPLEXORES: construcción de multiplexores de órdenes superiores, utilizando multiplexores de órdenes inferiores.

EJEMPLO: MUX 4:1 con multiplexores 2:1

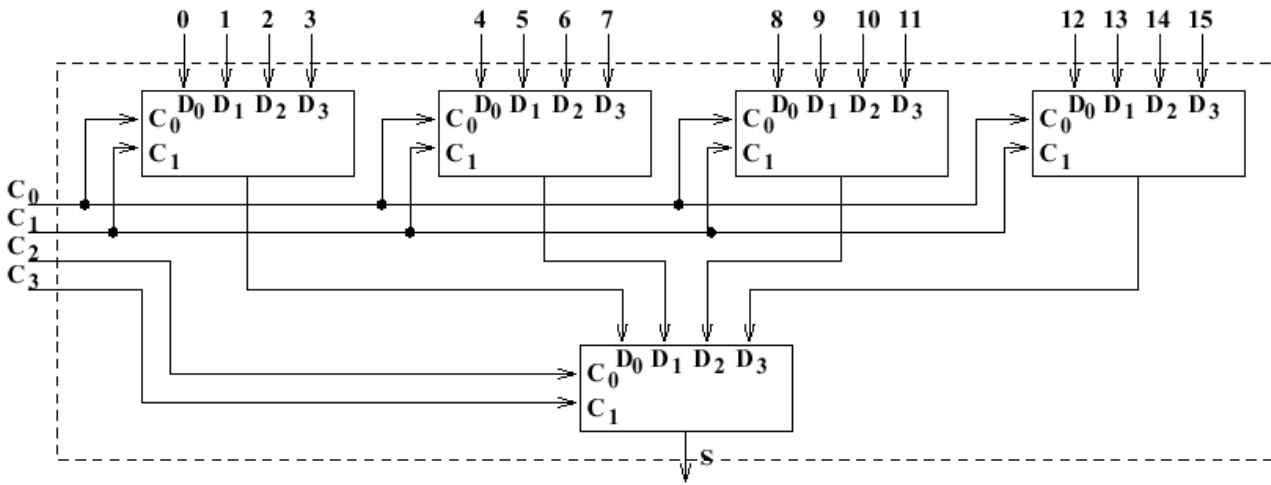


Señales control		Salidas intermedias y general		
C_1	C_0	S_0	S_1	Salida
0	0	E_0	E_2	E_0
0	1	E_1	E_3	E_1
1	0	E_0	E_2	E_2
1	1	E_1	E_3	E_3



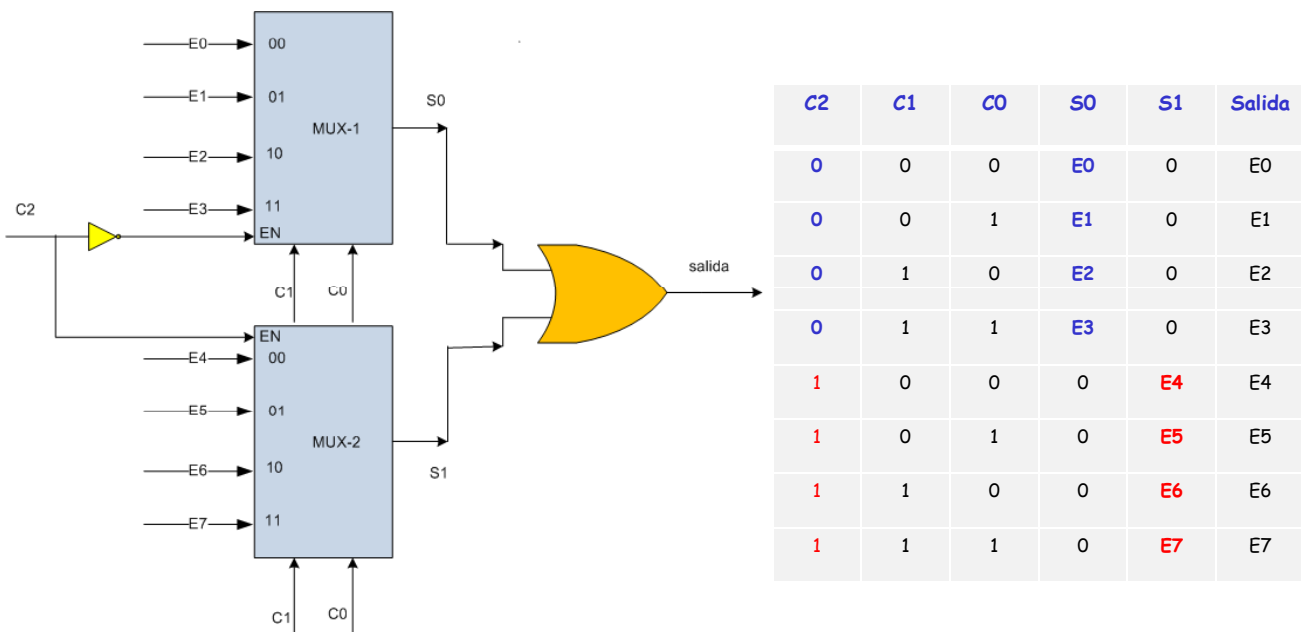
4.2 MULTIPLEXORES (XXVI)

EJEMPLO: MUX 16:1 con multiplexores 4:1



4.2 MULTIPLEXORES (XXVII)

EJEMPLO: MUX 8:1 con señal ENABLE y construido con multiplexores 4:1



4.2 DEMULTIPLEXORES (I)

Un demultiplexor realiza la operación inversa de la efectuada por un multiplexor.

Distribuye entre las salidas la señal que haya tomado de una entrada seleccionada por las señales de control

ENTRADA DE DATOS. N° de entradas: m

ENTRADAS DE SELECCIÓN. N° de entradas: $p \rightarrow 2^p \leq n$

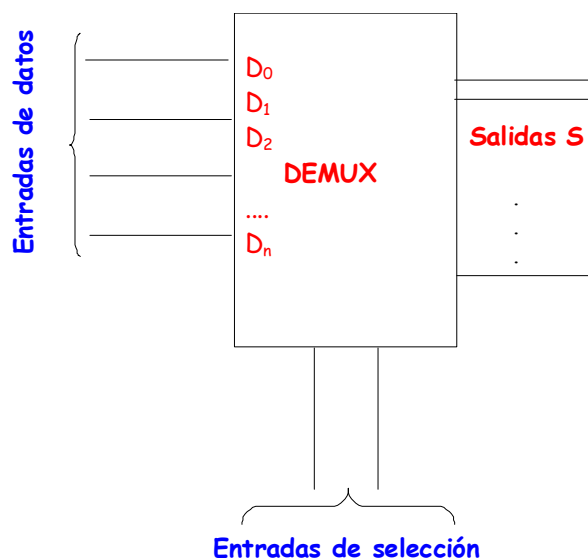
SALIDA (O SALIDAS). N° de salidas: n

Puede verse como un decodificador en el que la entrada de activación o *enable* (E) se utiliza como entrada de datos.



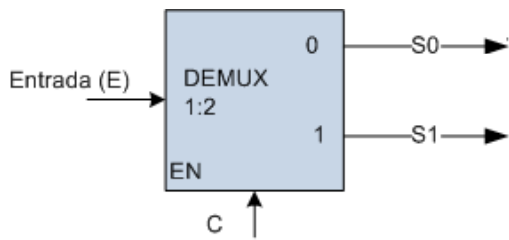
4.2 DEMULTIPLEXORES (II)

Bloque funcional



4.2 DEMULTIPLEXORES (III)

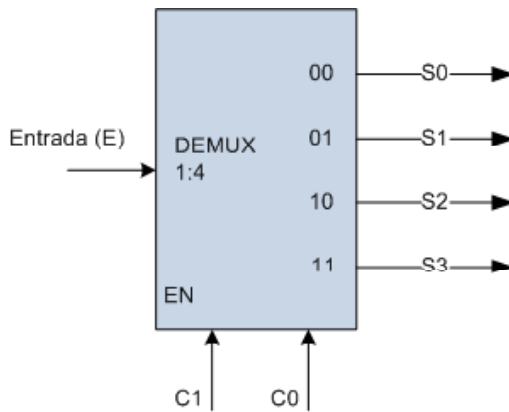
EJEMPLOS: DEMUX 1:2 y DEMUX 1:4



Control	Salidas	
C	S1	S0
0	0	E
1	E	0

$$S0 = EC'$$

$$S1 = EC$$



Control		Salidas			
C ₁	C ₀	S3	S2	S1	S0
0	0	0	0	0	E
0	1	0	0	E	0
1	0	0	E	0	0
1	1	E	0	0	0

$$S0 = E C1' C0'$$

$$S1 = E C1' C0$$

$$S2 = E C1 C0'$$

$$S3 = E C1 C0$$



4.2 DEMULTIPLEXORES (IV)

Demultiplexor 1:8, tabla de verdad

S ₂	S ₁	S ₀	O ₇	O ₆	O ₅	O ₄	O ₃	O ₂	O ₁	O ₀
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

$$(O_7 O_6 O_5 O_4 O_3 O_2 O_1 O_0) = F(S_2 S_1 S_0)$$

EJERCICIO DE APLICACIÓN: constrúyase con puertas AND



4.3 CODIFICADORES Y DECODIFICADORES (I)

Un Codificador binario es un circuito lógico combinacional que para cada entrada activada, produce un código de salida (combinación lógica) de N bits.

Para cada entrada activa un código de salida

ENTRADA DE DATOS. N° de entradas: m

SALIDAS N° de salidas: n

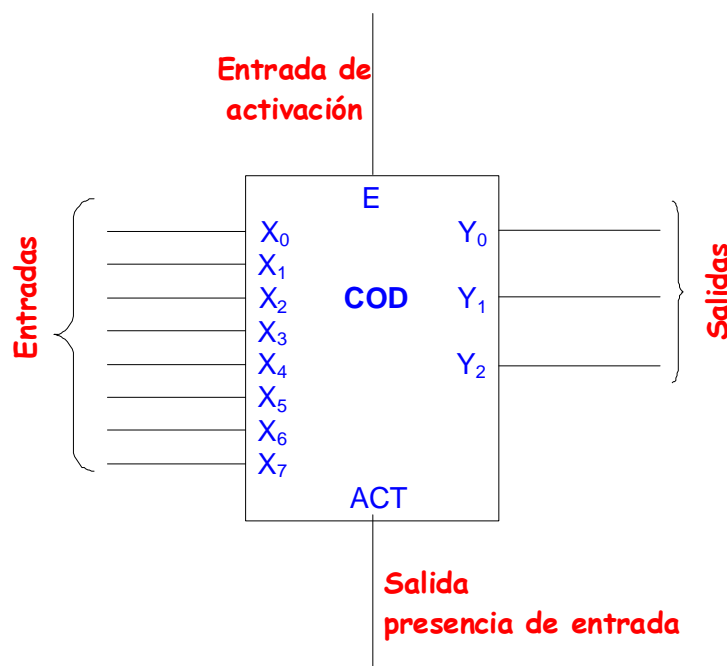
En el caso general: n y $m \in \mathbb{Z}$ y se cumple $2^n \geq m$

en el caso particular: $m = 2^n$



4.3 CODIFICADORES (II)

Bloque funcional



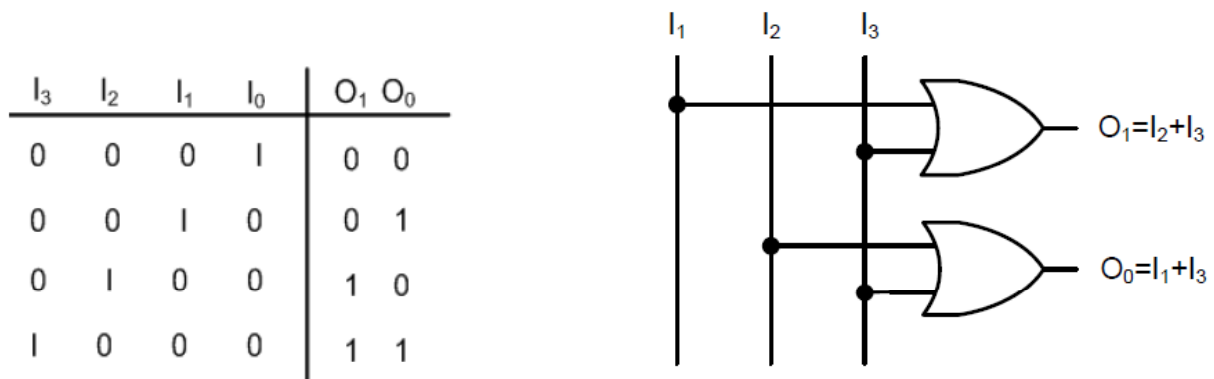
4.3 CODIFICADORES (III)

Ejemplo de codificador

Nº de entradas: 4

Nº de salidas: 2

se cumple $2^2 = 4$



4.3 CODIFICADORES (IV)

Codificador de octal a binario. Tabla de verdad

	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	O_2	O_1	O_0
	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	1	0	0	0	1
	0	0	0	0	0	1	0	0	0	1	0
	0	0	0	0	1	0	0	0	0	1	1
	0	0	0	1	0	0	0	0	1	0	0
	0	0	1	0	0	0	0	0	1	0	1
	0	1	0	0	0	0	0	0	1	1	0
	1	0	0	0	0	0	0	0	1	1	1

EJERCICIO DE APLICACIÓN: Impleméntese



4.3 DECODIFICADORES (I)

Un decodificador es un circuito lógico combinacional que para cada combinación lógica de los valores de sus entradas activa una y solamente una salida.

Para cada código de entrada se activa una salida

ENTRADA DE DATOS. N° de entradas: n

SALIDAS N° de salidas: m

$n, m \in \mathbb{Z}$ y cumplen $m \leq 2^n$

Las salidas de un decodificador generan todos los productos canónicos de las entradas



4.3 DECODIFICADORES (II)

Bloque funcional

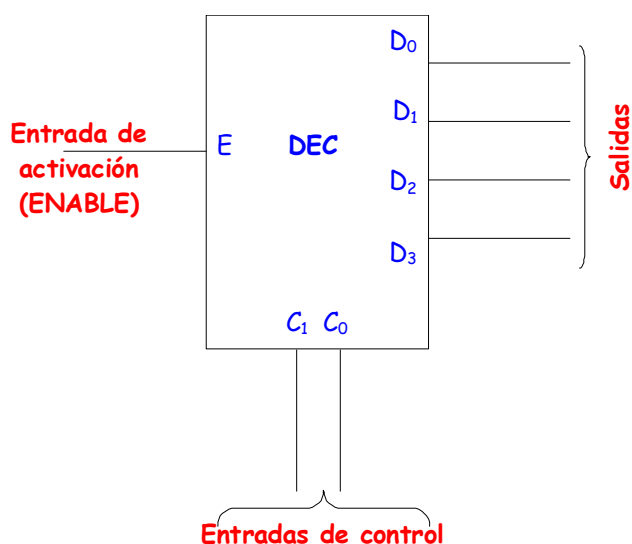


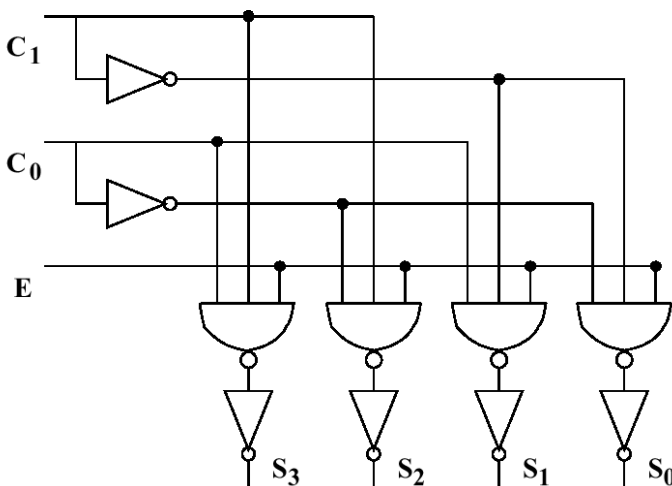
Tabla de verdad

E	C ₁	C ₀	D ₃	D ₂	D ₁	D ₀
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0



4.3 DECODIFICADORES (III)

Estructura interna basada en puertas lógicas NAND



Funciones lógicas de salida

$$S_0 = E \bar{C}_1 \bar{C}_0$$

$$S_1 = E \bar{C}_1 C_0$$

$$S_2 = E C_1 \bar{C}_0$$

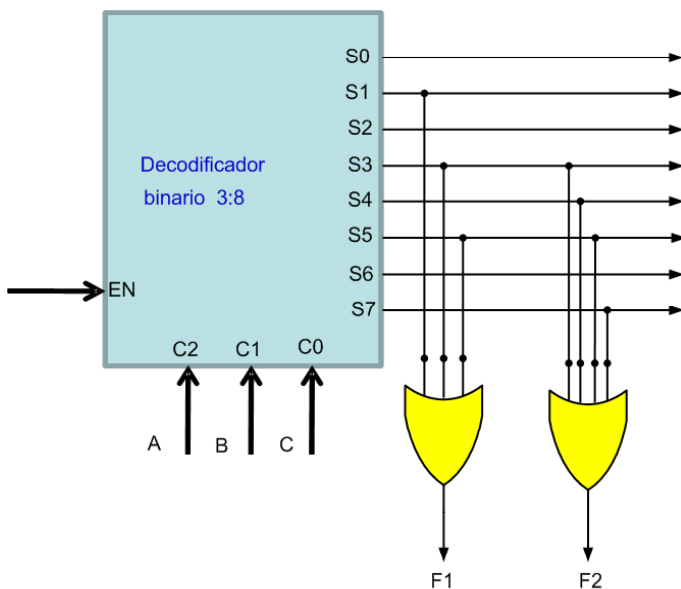
$$S_3 = E C_1 C_0$$

Ejemplo: $F(a,b,c) = ab + \bar{b}c$



4.3 DECODIFICADORES (IV)

Ejemplo de utilización de decodificadores para la implementación de funciones lógicas



Sean las funciones de 3 variables

$$F1 = B'C + A'C \quad \text{y}$$

$$F2 = AB' + BC$$

Transformación de las funciones a sus formas canónicas:

$$F1 = B'C + A'C =$$

$$= A'B'C + AB'C + A'B'C + A'BC =$$

$$= \Sigma (1, 3, 5)$$

$$F2 = AB' + BC =$$

$$= AB'C' + AB'C + A'BC + ABC =$$

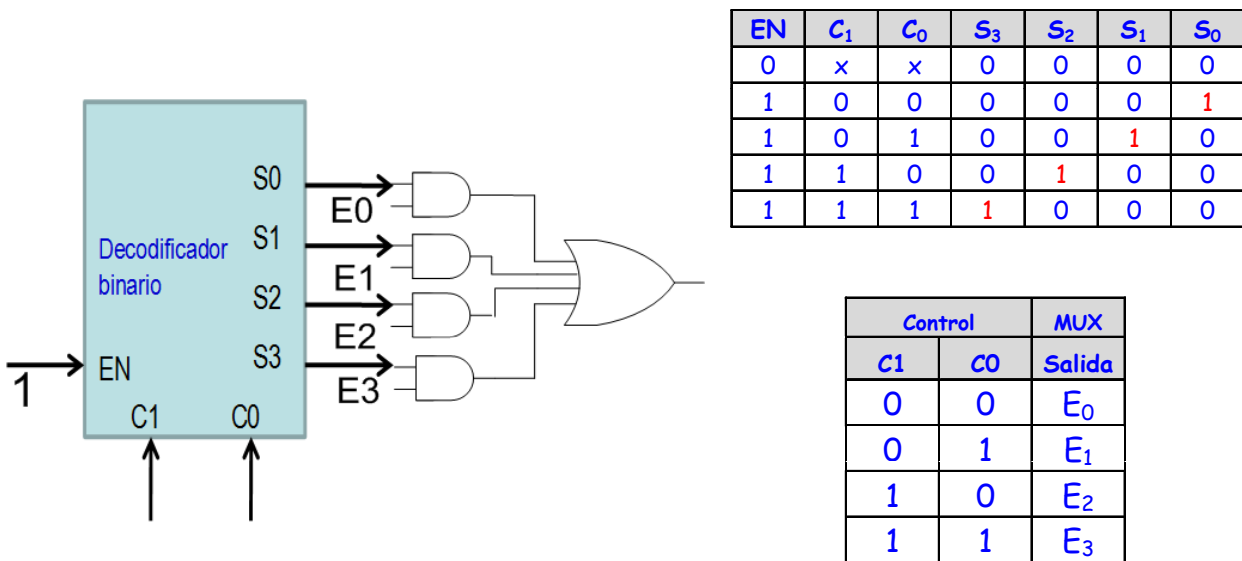
$$= \Sigma (3, 4, 5, 7)$$

Al tener 3 variables se requiere un decoder de 3:8 (3 variables de control)



4.3 DECODIFICADORES (V)

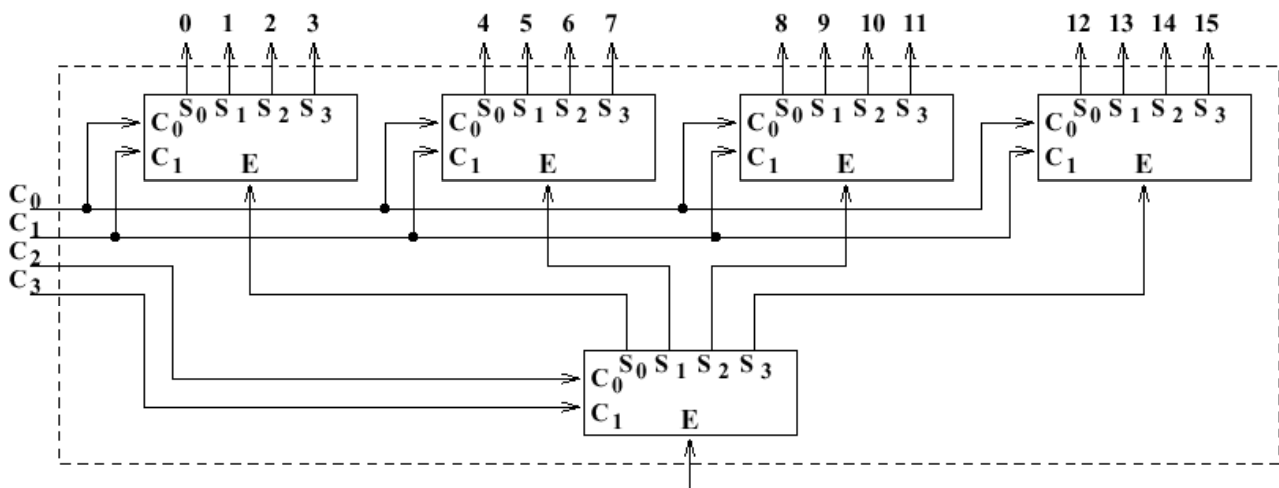
EJEMPLO: Construcción de un multiplexor 4:1, con un decodificador 2:4 y las puertas lógicas que se precisen



4.3 DECODIFICADORES (VI)

Obtención de decodificadores de órdenes superiores

EJEMPLO: decodificador 4:16 basado en decodificadores 2:4



4.4 COMPARADORES (I)

Los **COMPARADORES** son sistemas combinatoriales que comparan las magnitudes de cantidades binarias para determinar su relación.

- Se toman números enteros sin signo, representados con de N bits.
- El procedimiento general consiste en buscar una desigualdad en cualquier posición, comenzando por el bit más significativo. Quedando establecida la relación en el momento en que se encuentre una desigualdad entre bits.

Si $A_i=1$ y $B_i=0 \rightarrow A > B$

Si $A_i=0$ y $B_i=1 \rightarrow A < B$

Si $A_3= B_3 \rightarrow$ Hay que examinar los otros bits

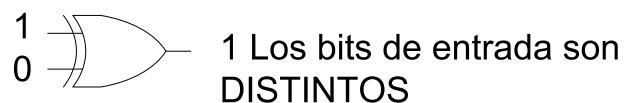
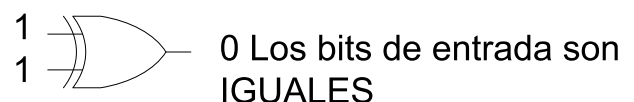
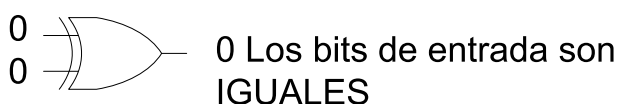
- Si se examinan todas las posiciones y no se encuentra ninguna desigualdad, los números son iguales.



4.4 COMPARADORES (II)

Determinación de **IGUALDAD**:

Comparación de números de 1 bit



4.4 COMPARADORES (III)

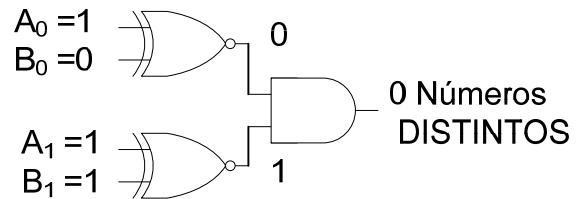
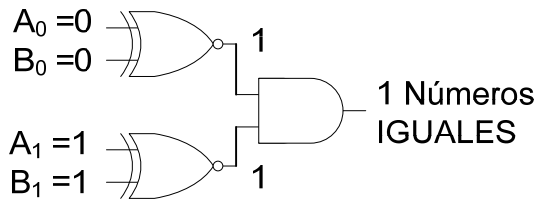
Comparación de números de 2 bits

Menor peso: subíndice 0

10
10

11
10

A_1A_0
 B_1B_0



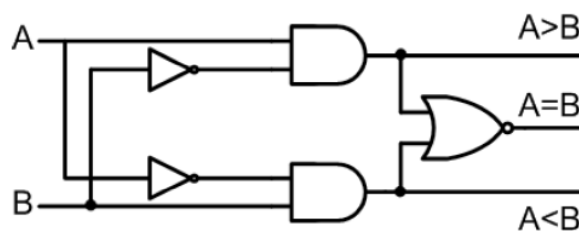
A	B	XNOR
0	0	1
0	1	0
1	0	0
1	1	1



4.4 COMPARADORES (IV)

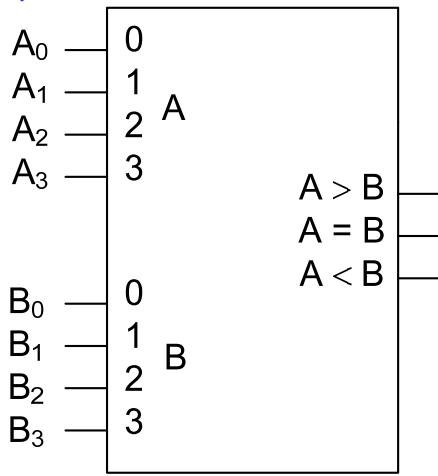
Comparación bit a bit de 2 números A y B

A_i	B_i	A=B	A>B	A<B
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0



4.4 COMPARADORES (V)

COMPARDOR DE 4 BITS: un ejemplo es el 74LS85. Es un comparador, de números de 4 bits. Dispone de otras 3 entradas de expansión, que permite conectar varios en cascada



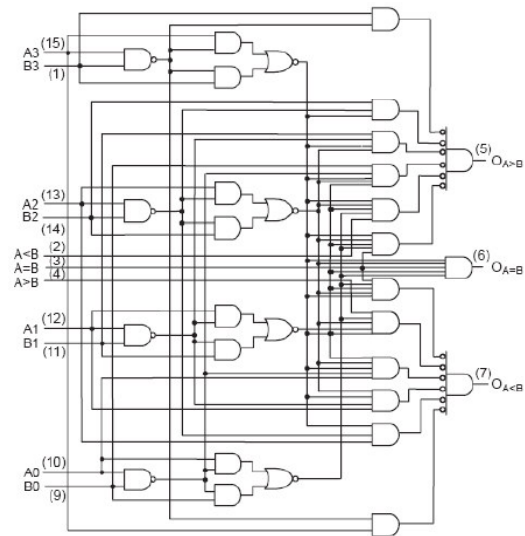
Menor peso: subíndice 0

Si $A_3=1$ y $B_3=0 \rightarrow A > B$

Si $A_3=0$ y $B_3=1 \rightarrow A < B$

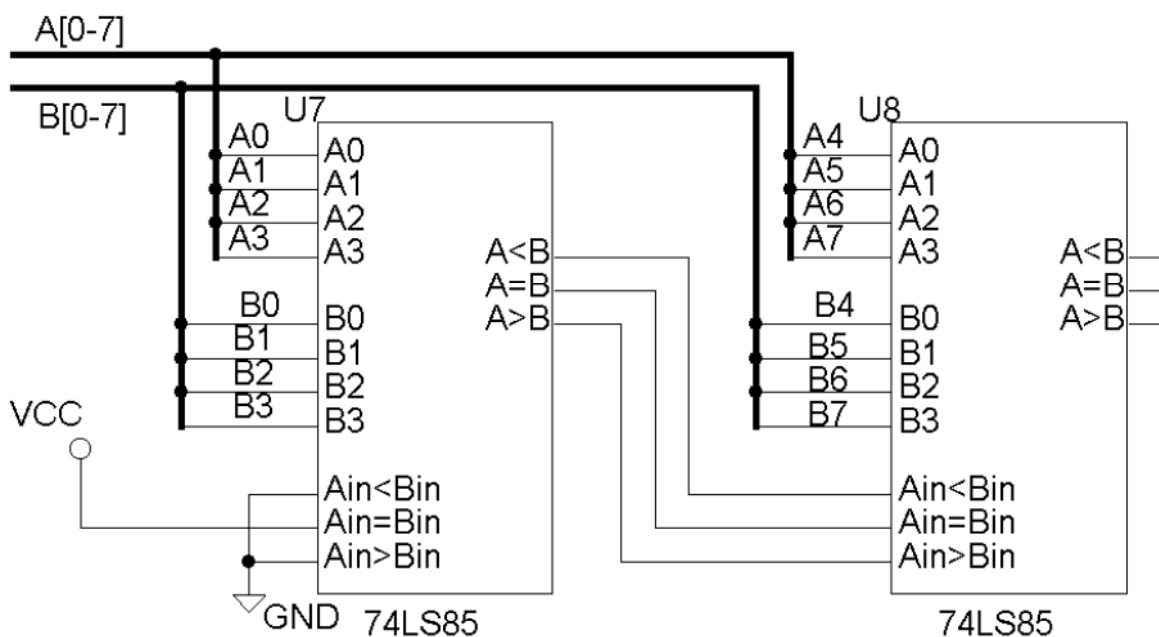
Si $A_3 = B_3 \rightarrow$ Hay que examinar los otros bits

Diagrama lógico del 74LS85



4.4 COMPARADORES (VI)

Comparador de números de 8 bits



4.4 COMPARADORES (VII)

Comparador de números de 16 bits

