Tema 3. Circuitos Combinacionales

Fundamentos de los Computadores

Índice

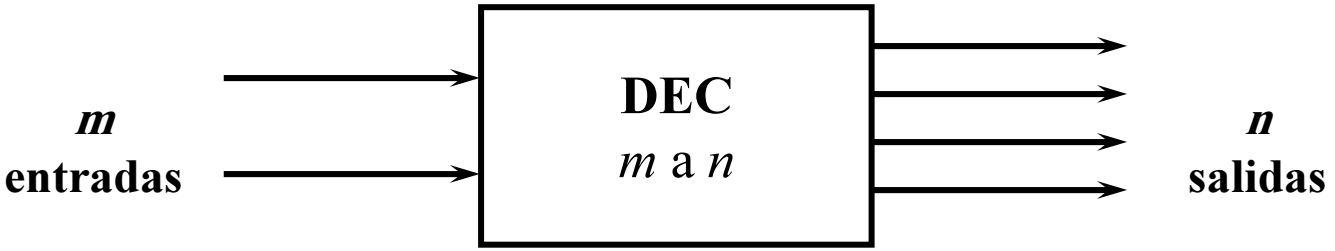
1. Introducción
Circuitos combinacionales (I)

- En el tema anterior se han estudiado los principios básicos necesarios para abordar la descripción e implementación de circuitos digitales a partir de las puertas lógicas o funciones lógicas elementales.
- En este tema se aplicarán dichos principios para comprender el funcionamiento e implementación de los circuitos combinacionales básicos más utilizados.
- Se estudiará la importancia de dichos circuitos como elementos básicos en la implementación de las diferentes unidades funcionales del computador y en la transferencia de información entre ellas.

1. Introducción
Circuitos combinacionales (II)

- En un circuito combinacional, la relación entre la(s) entrada(s) y la(s) salida(s) puede expresarse mediante una función lógica
  - El valor de la(s) salida(s) en un instante dado depende exclusivamente del valor de la(s) entrada(s) en ese instante
- Las puertas lógicas introducen un pequeño retardo entre la entrada y la salida (del orden de nanosegundos)
  - Un circuito combinacional real presenta un retardo entre la entrada y la salida
  - El retardo depende del tipo de puertas, su nº de entradas y el nivel del circuito
- Estudiaremos circuitos que implementan funciones sencillas
  - Están integrados en pastillas (chips)

2. Decodificadores (I)



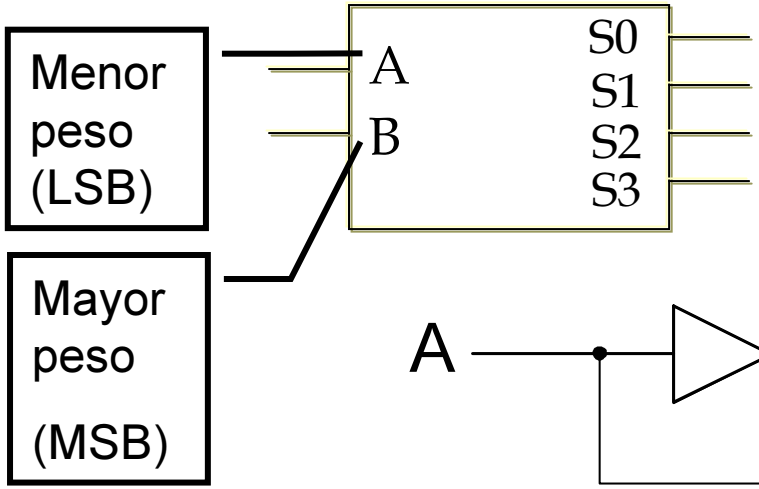
Son de gran utilidad para habilitar dispositivos

- Decodificadores binarios
  - m entradas y n = 2^m salidas
    - 2 a 4, 3 a 8, 4 a 16
- Decodificadores de BCD a 7 segmentos
  - 4 entradas y 7 salidas
- Decodificadores de BCD a decimal
  - 4 entradas y 10 salidas

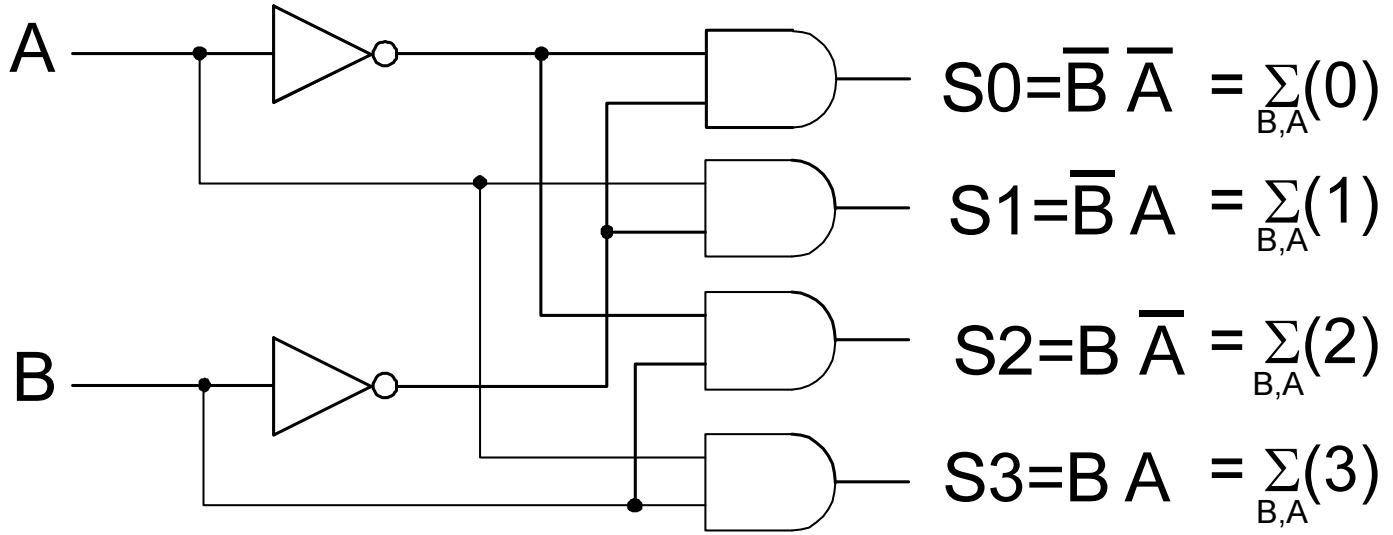
Decodificadores (II)

(Las salidas son mutuamente excluyentes)

- Decodificador binario



ENTRADAS		SALIDAS			
B	A	S3	S2	S1	S0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

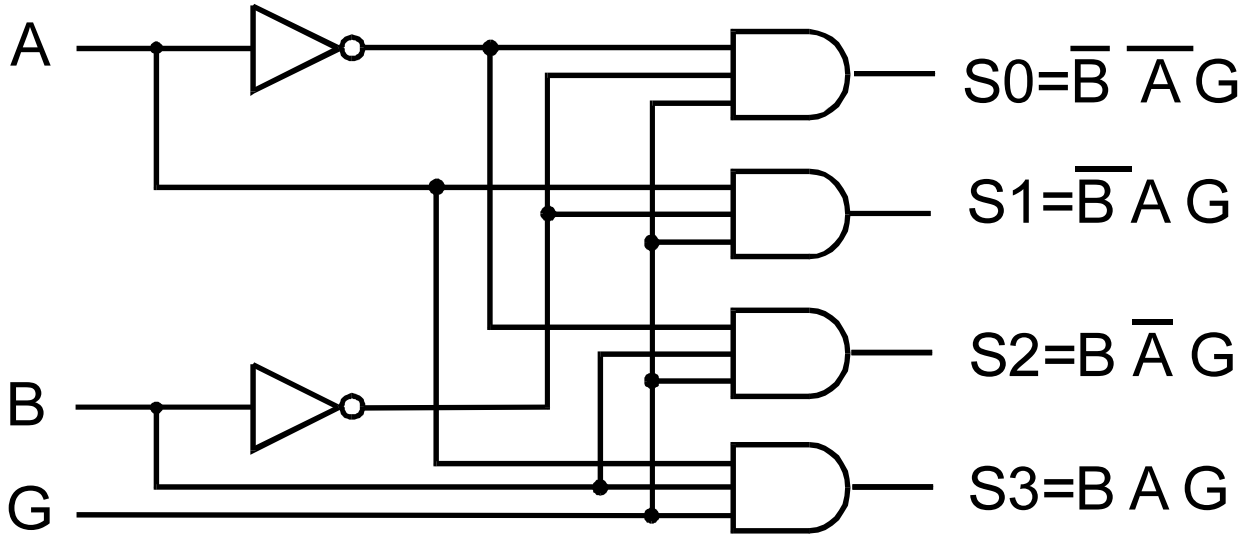


Decodificadores (III)

(Enable o strobe)

- Entrada de habilitación

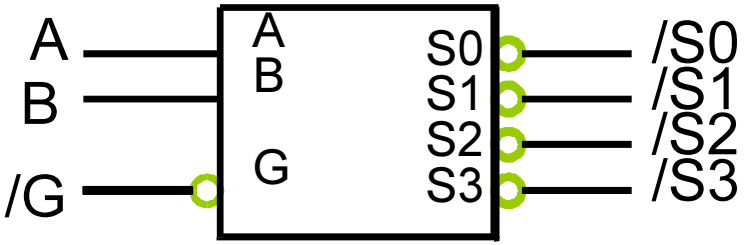
ENTRADAS			SALIDAS			
G	B	A	S3	S2	S1	S0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0



(habilitación)

Decodificadores (IV)

- Circuito integrado 74LS139



ENTRADAS			SALIDAS			
/G	B	A	/S3	/S2	/S1	/S0
1	X	X	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	1

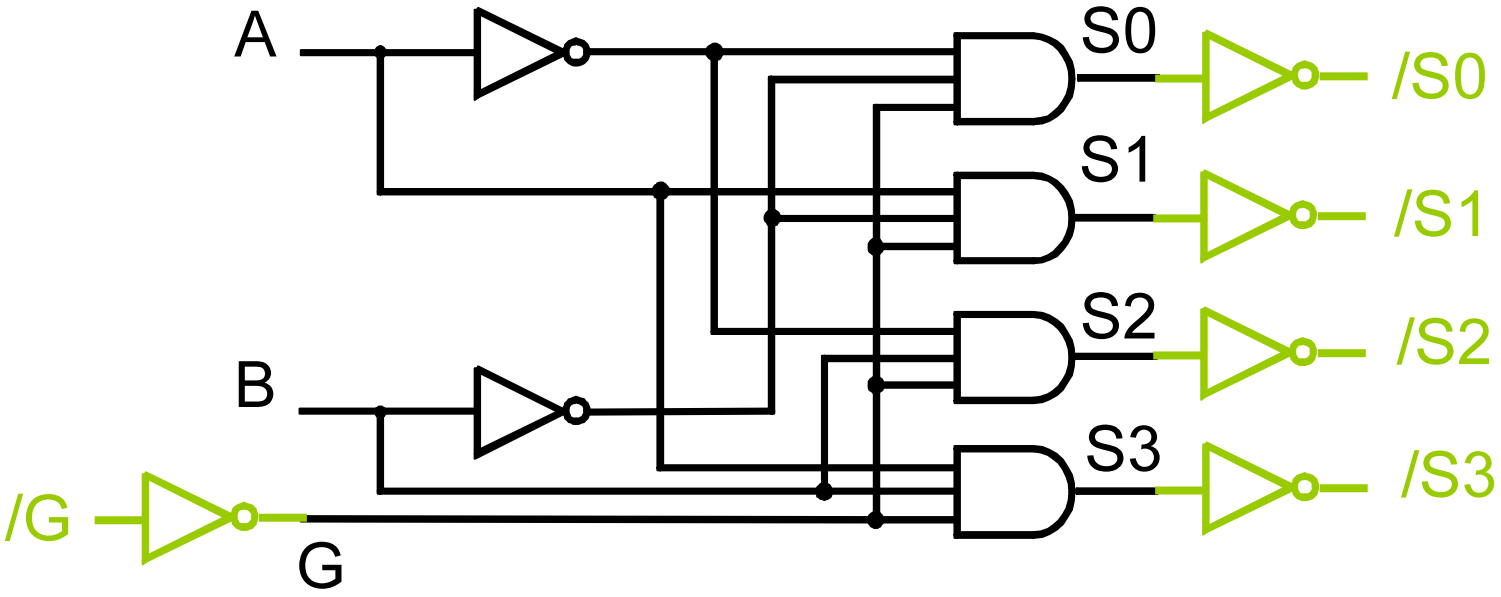
Entrada de habilitación y salidas activas a nivel bajo. Se indica con los círculos y nombres de las variables.

Ejercicio: Diseñad el circuito lógico

Dos formas:

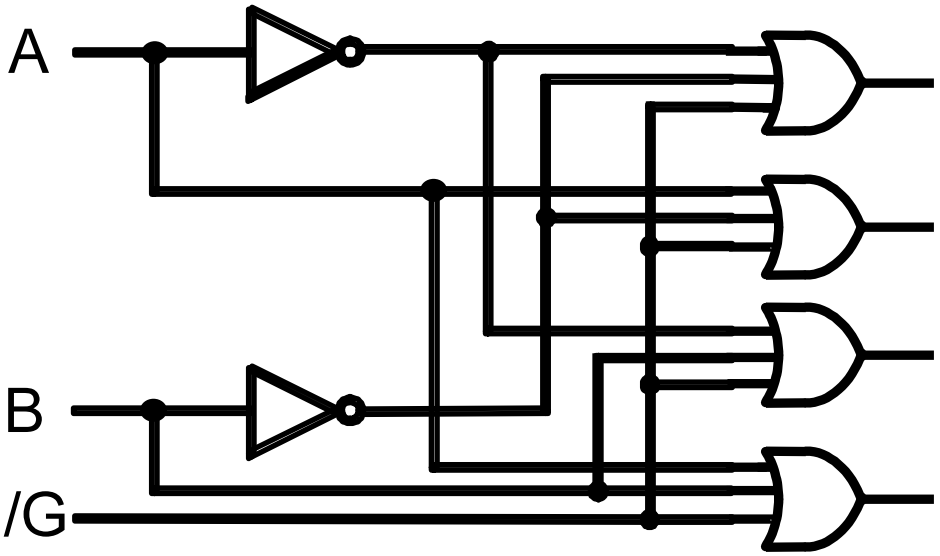
- Partiendo del diseño con entrada de habilitación y salidas activas a nivel alto
- Partiendo de la tabla verdad mediante producto de maxitérminos

Posible Solución



En esta solución el nivel del circuito es tres, a continuación se presenta una solución más eficiente

Solución Eficiente



$/S3 = /G + \overline{B} + \overline{A} = /G + \prod_{B,A}(3)$

$/S2 = /G + \overline{B} + A = /G + \prod_{B,A}(2)$

$/S1 = /G + B + \overline{A} = /G + \prod_{B,A}(1)$

$/S0 = /G + B + A = /G + \prod_{B,A}(0)$

(habilitación)

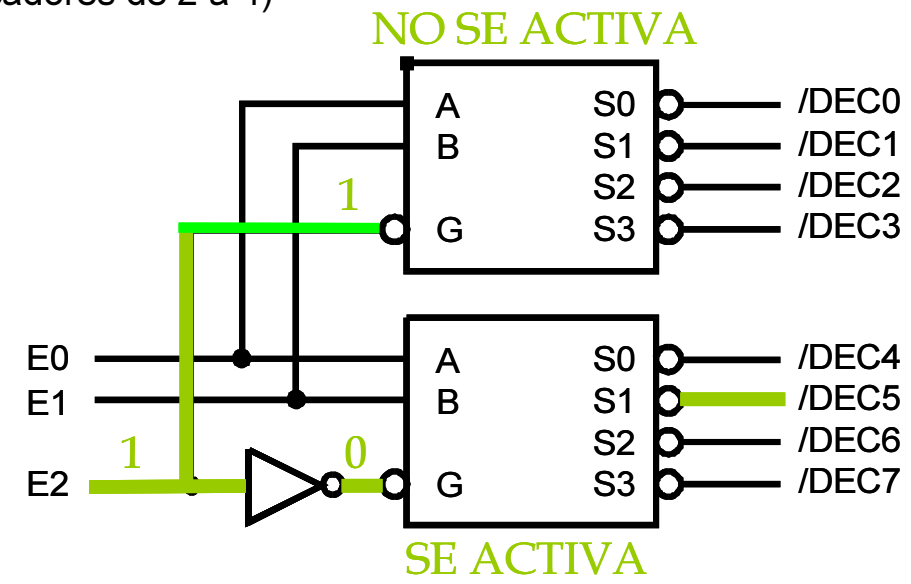
Cada salida /Si implementa el maxitérmino correspondiente a la fila i

En este caso, el nivel del circuito es 2

Decodificadores (V)

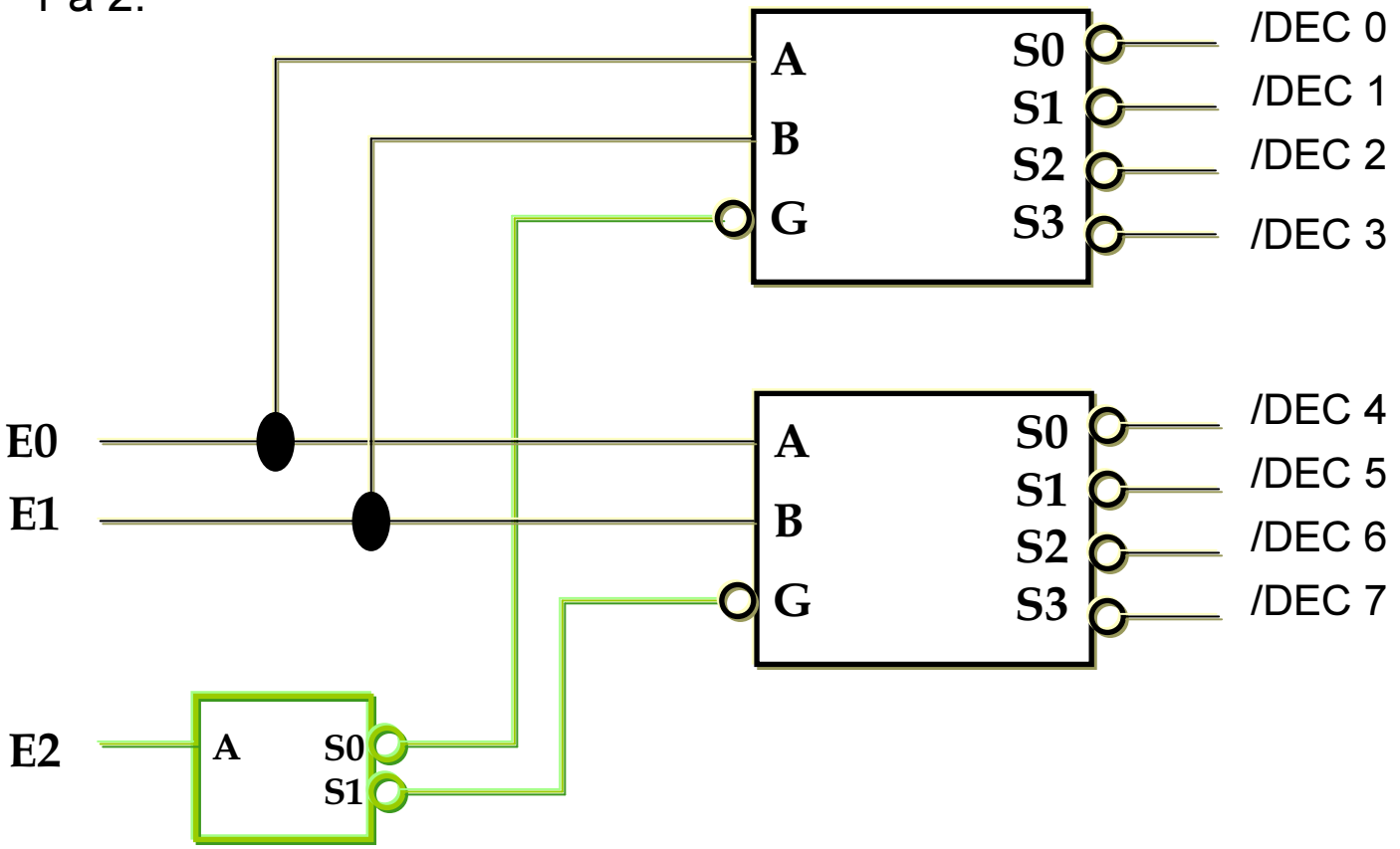
- Composición de decodificadores
  - Tamaño más grande existente en el mercado: 4 a 16
  - Podemos implementar decodificadores mayores combinando o componiendo decodificadores más pequeños en paralelo
  - Ejemplos de esta técnica:
- Decodificador de 3 a 8 (con decodificadores de 2 a 4)

E2	E1	E0	
0	0	0	/DEC0
0	0	1	/DEC1
0	1	0	/DEC2
0	1	1	/DEC3
1	0	0	/DEC4
1	0	1	/DEC5
1	1	0	/DEC6
1	1	1	/DEC7



Decodificadores (VI)

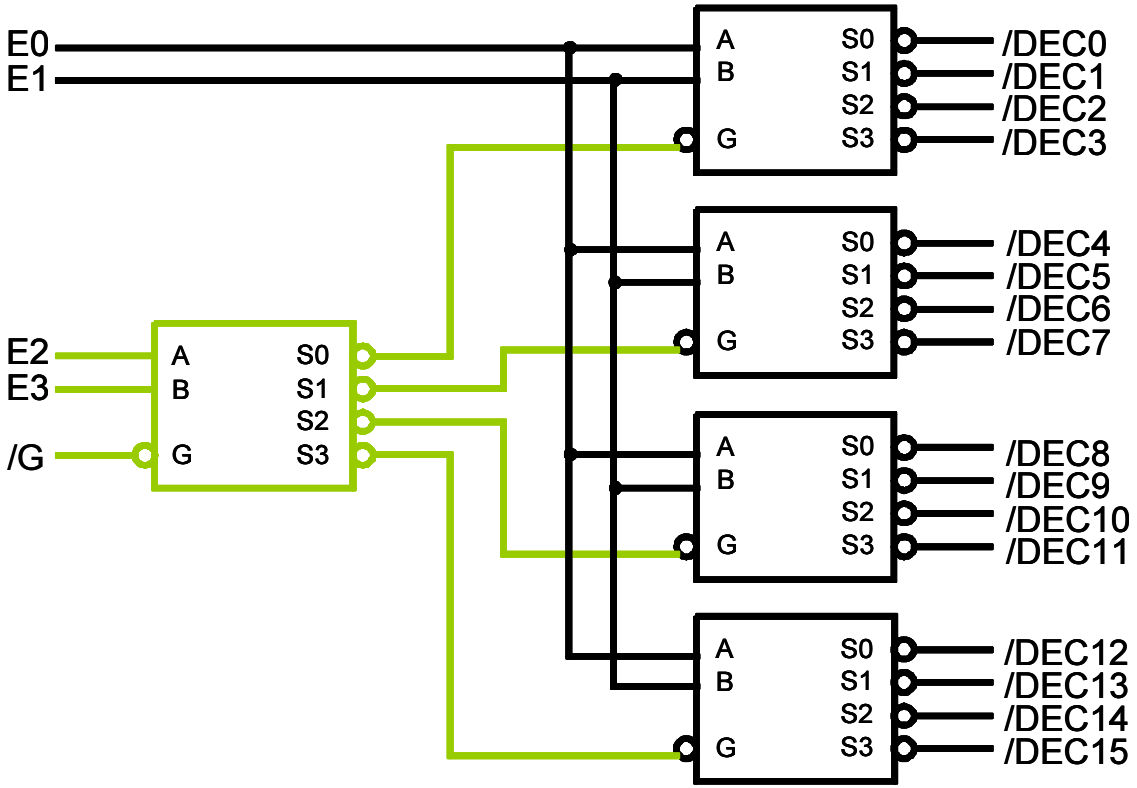
- Composición de decodificadores:
  - Decodificador de 3 a 8 utilizando dos decodificadores de 2 a 4 y otro de 1 a 2:



Decodificadores (VII)

– Decodificador de 4 a 16 con decodificadores de 2 a 4 (74LS139) :

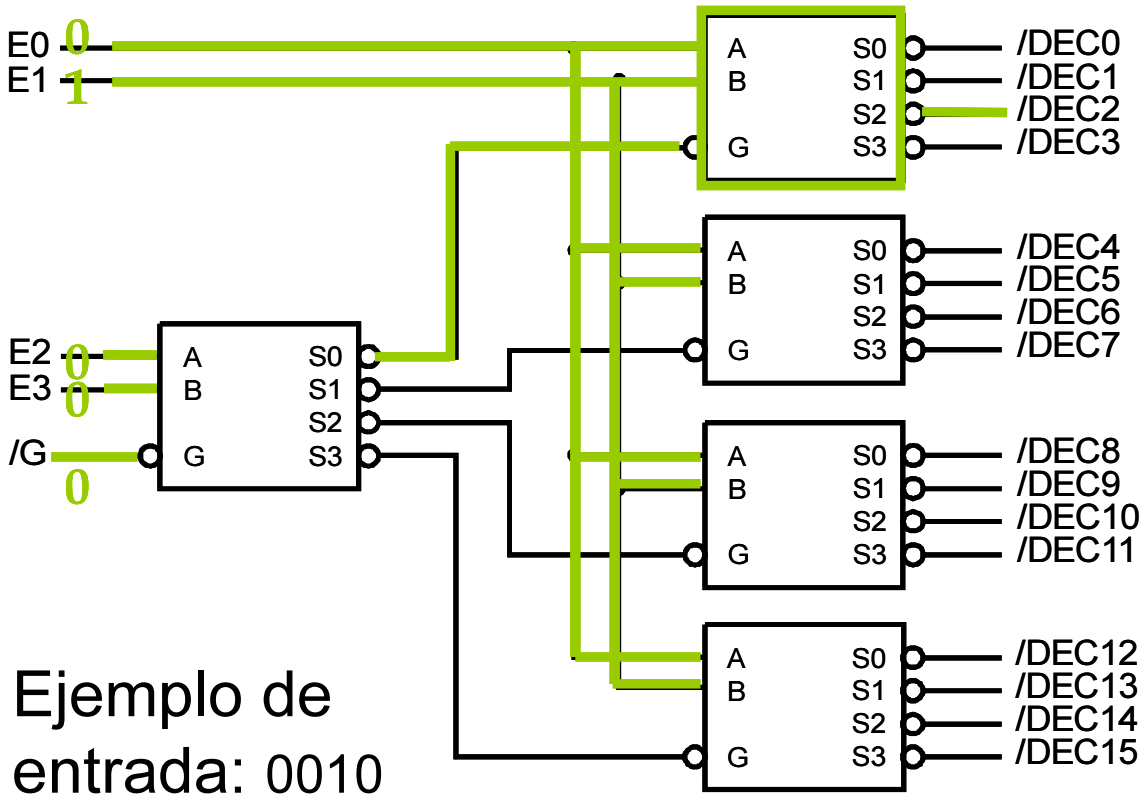
	E3	E2	E1	E0	
(0)	0	0	0	0	/DEC0
(1)	0	0	0	1	/DEC1
(2)	0	0	1	0	/DEC2
(3)	0	0	1	1	/DEC3
(4)	0	1	0	0	/DEC4
(5)	0	1	0	1	/DEC5
(6)	0	1	1	0	/DEC6
(7)	0	1	1	1	/DEC7
(8)	1	0	0	0	/DEC8
(9)	1	0	0	1	/DEC9
(10)	1	0	1	0	/DEC10
(11)	1	0	1	1	/DEC11
(12)	1	1	0	0	/DEC12
(13)	1	1	0	1	/DEC13
(14)	1	1	1	0	/DEC14
(15)	1	1	1	1	/DEC15



Ejemplo

– Decodificador de 4 a 16 con decodificadores de 2 a 4

	E3	E2	E1	E0	
(0)	0	0	0	0	/DEC0
(1)	0	0	0	1	/DEC1
(2)	0	0	1	0	/DEC2
(3)	0	0	1	1	/DEC3
(4)	0	1	0	0	/DEC4
(5)	0	1	0	1	/DEC5
(6)	0	1	1	0	/DEC6
(7)	0	1	1	1	/DEC7
(8)	1	0	0	0	/DEC8
(9)	1	0	0	1	/DEC9
(10)	1	0	1	0	/DEC10
(11)	1	0	1	1	/DEC11
(12)	1	1	0	0	/DEC12
(13)	1	1	0	1	/DEC13
(14)	1	1	1	0	/DEC14
(15)	1	1	1	1	/DEC15



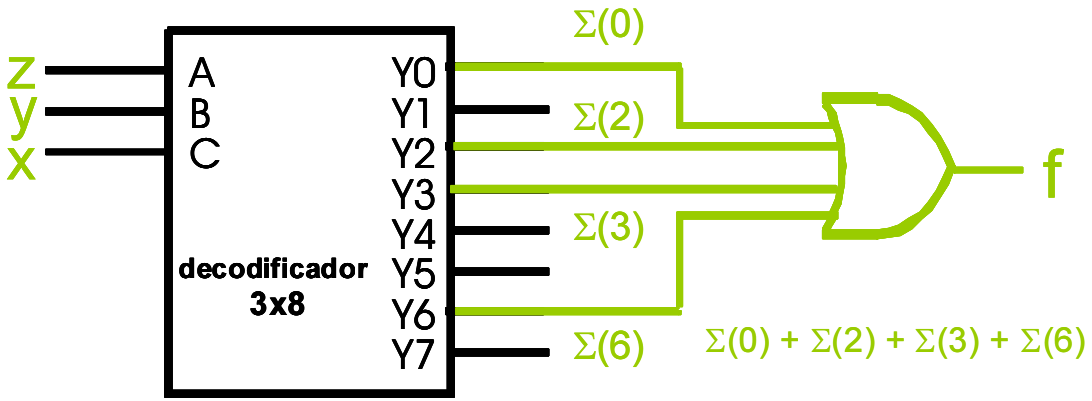
⇨ Ejemplo de entrada: 0010

Decodificadores (VIII)

- El decodificador como Generador de Funciones (1/4)
  - Decodificador binario con salidas activas a nivel alto + puerta OR
  - Salida i-ésima: minitérmino i = Σ(i)
  - Se puede usar para implementar funciones lógicas expresadas en forma de suma de minitérminos

Ejemplo: Implementar la función Mediante un decodificador de 3 a 8

$$f = \sum_{x,y,z}(0,2,3,6)$$



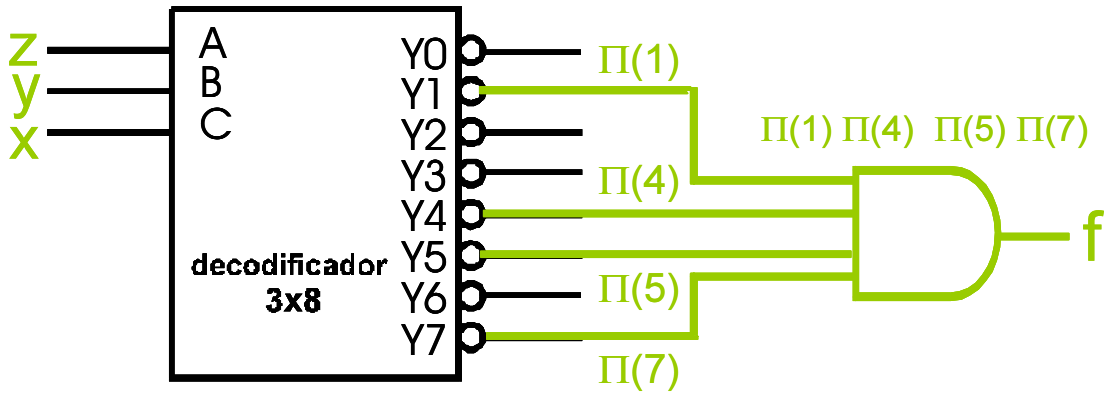
x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Decodificadores (IX)

- El decodificador como Generador de Funciones (2/4)
  - Decodificador binario con salidas activas a nivel bajo + puerta AND
  - Salida i-ésima: maxitérmino i = $\Pi(i)$
  - Se puede usar para implementar funciones lógicas expresadas en forma de producto de maxitérminos

Ejemplo: Implementar la función
Mediante un decodificador de 3 a 8

$$f = \prod_{x,y,z} (1,4,5,7)$$



x	y	z	f
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Decodificadores (X)

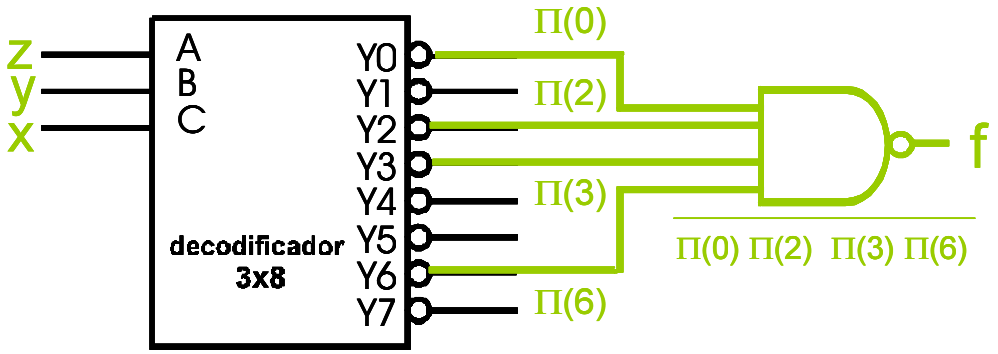
- El decodificador como Generador de Funciones (3/4)
  - Decodificador binario con salidas activas a nivel bajo + puerta NAND

$$f = \sum_{x,y,z}(0,2,3,6) = \sum_{x,y,z}(0) + \sum_{x,y,z}(2) + \sum_{x,y,z}(3) + \sum_{x,y,z}(6) =$$

$$= \overline{\sum_{x,y,z}(0) + \sum_{x,y,z}(2) + \sum_{x,y,z}(3) + \sum_{x,y,z}(6)} = \overline{\sum_{x,y,z}(0)}\,\overline{\sum_{x,y,z}(2)}\,\overline{\sum_{x,y,z}(3)}\,\overline{\sum_{x,y,z}(6)}$$

$$= \overline{\prod_{x,y,z}(0)\prod_{x,y,z}(2)\prod_{x,y,z}(3)\prod_{x,y,z}(6)}$$

$$\overline{\sum_{x,y,z}(i)} = \prod_{x,y,z}(i)$$



Decodificadores (XI)

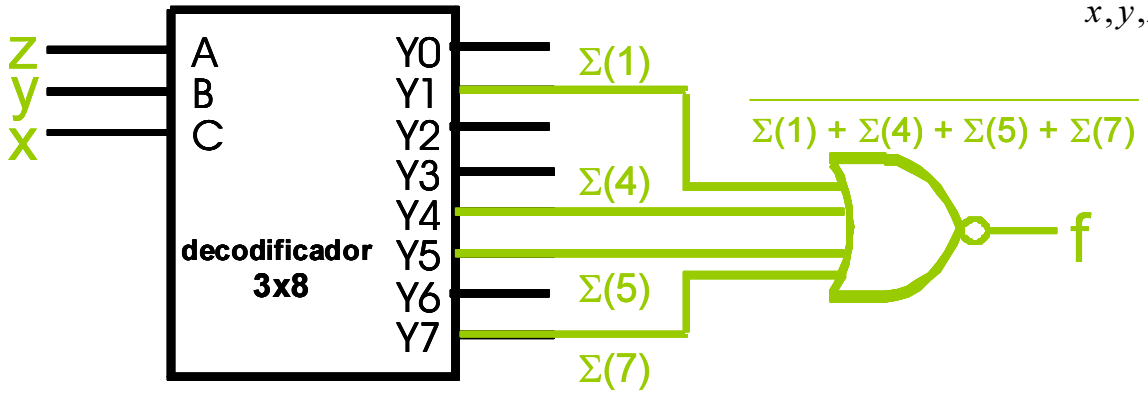
- El decodificador como Generador de Funciones (4/4)
  - Decodificador binario con salidas activas a nivel alto + puerta NOR

$$f = \prod_{x,y,z}(1,4,5,7) = \prod_{x,y,z}(1)\prod_{x,y,z}(4)\prod_{x,y,z}(5)\prod_{x,y,z}(7) =$$

$$= \overline{\prod_{x,y,z}(1)\prod_{x,y,z}(4)\prod_{x,y,z}(5)\prod_{x,y,z}(7)} = \overline{\prod_{x,y,z}(1) + \prod_{x,y,z}(4) + \prod_{x,y,z}(5) + \prod_{x,y,z}(7)}$$

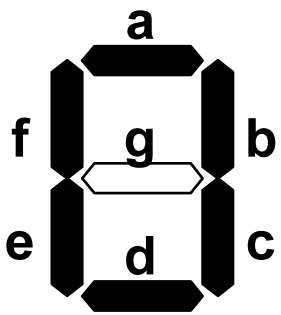
$$= \overline{\sum_{x,y,z}(1) + \sum_{x,y,z}(4) + \sum_{x,y,z}(5) + \sum_{x,y,z}(7)}$$

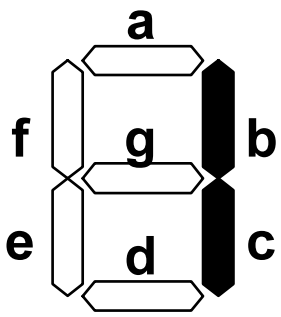
$$\sum_{x,y,z}(i) = \overline{\prod_{x,y,z}(i)}$$

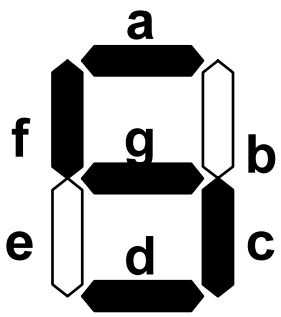


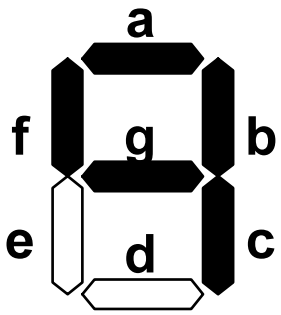
Decodificadores (XII)

- Decodificadores BCD a 7 segmentos (salidas no excluyentes)





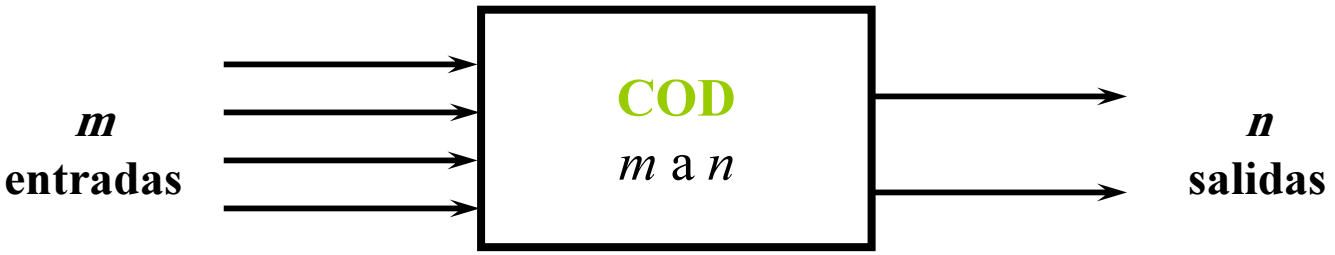




DECIMAL	ENTRADAS				SALIDAS						
	D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	0	0	1	1

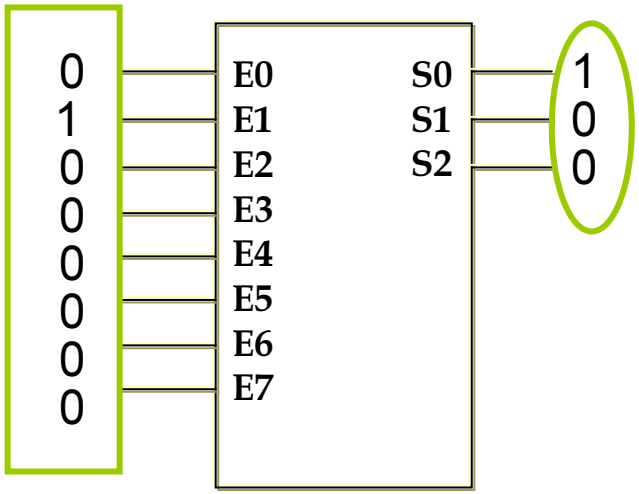
3. Codificadores (I)

- Función opuesta al decodificador



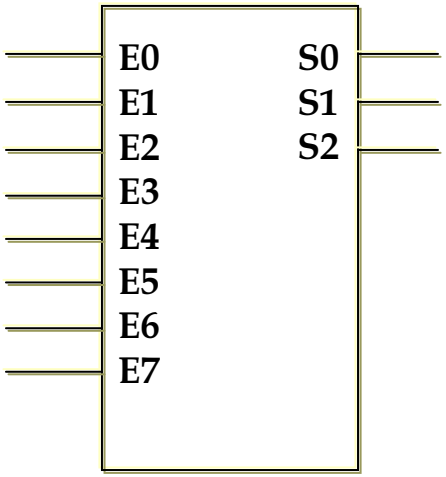
  - ⇨ Codificador binario
    - m=2^n entradas y n salidas
  - ⇨ Se emplean en subsistemas de entrada/salida
    - el código de salida identifica el dispositivo que realiza una petición al procesador

Codificadores (II)



ENTRADAS								SALIDAS		
E7	E6	E5	E4	E3	E2	E1	E0	S2	S1	S0
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0

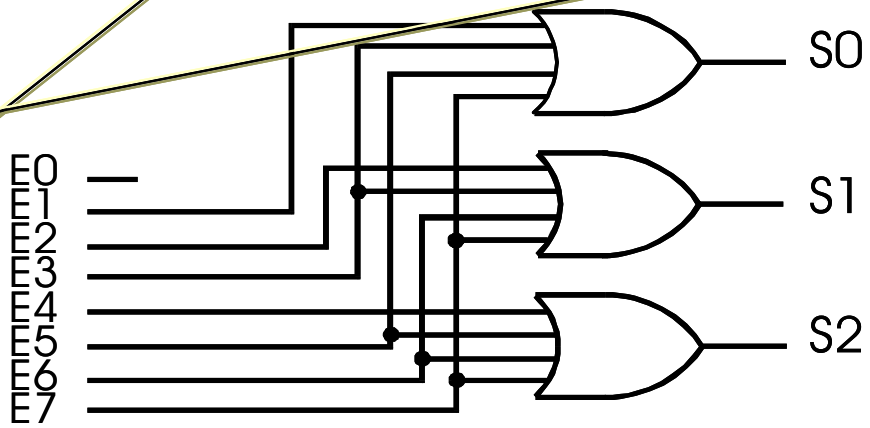
Codificadores (III)



ENTRADAS								SALIDAS		
E7	E6	E5	E4	E3	E2	E1	E0	S2	S1	S0
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1	1	1
0	1	0	0	0	0	0	0	1	1	0
0	0	1	0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	1	0	0	0

Inconveniente:

La entrada E0, genera un código de salida que es imposible de distinguir del código generado cuando todas las entradas son cero.

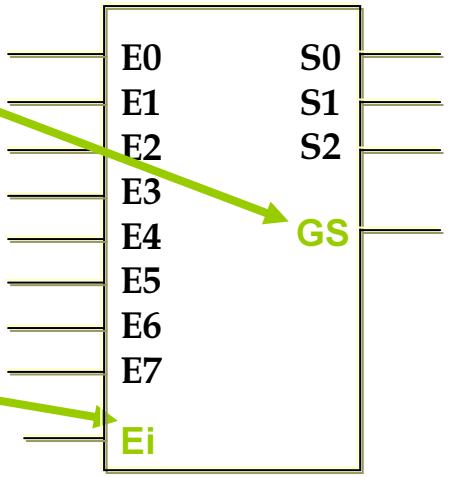


Codificadores (IV)

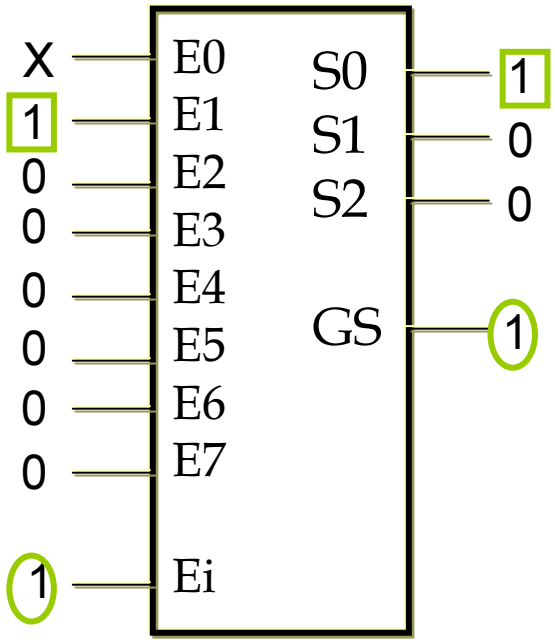
Inconvenientes:

- La entrada E0 genera un código idéntico al código generado cuando todas las entradas son cero (es imposible saber cuándo está activa)
  - Solución: GS (group selection). Se activa si alguna entrada es activa
- Si hay más de una entrada activa se genera un conflicto
  - Solución: Codificadores PRIORITARIOS

Es posible añadir una entrada de habilitación Ei (enable input)



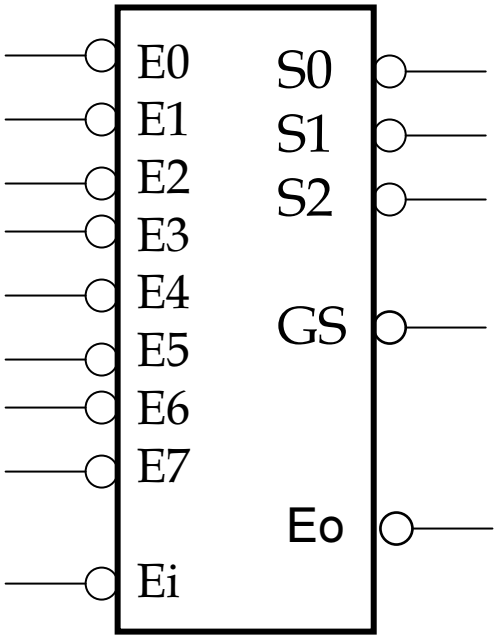
Codificadores (V)



ENTRADAS									SALIDAS			
Ei	E7	E6	E5	E4	E3	E2	E1	E0	S2	S1	S0	GS
0	×	×	×	×	×	×	×	×	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	×	×	×	×	×	×	×	1	1	1	1
1	0	1	×	×	×	×	×	×	1	1	0	1
1	0	0	1	×	×	×	×	×	1	0	1	1
1	0	0	0	1	×	×	×	×	1	0	0	1
1	0	0	0	0	1	×	×	×	0	1	1	1
1	0	0	0	0	0	1	×	×	0	1	0	1
1	0	0	0	0	0	0	1	×	0	0	1	1
1	0	0	0	0	0	0	0	1	0	0	0	1

Codificadores (VI)

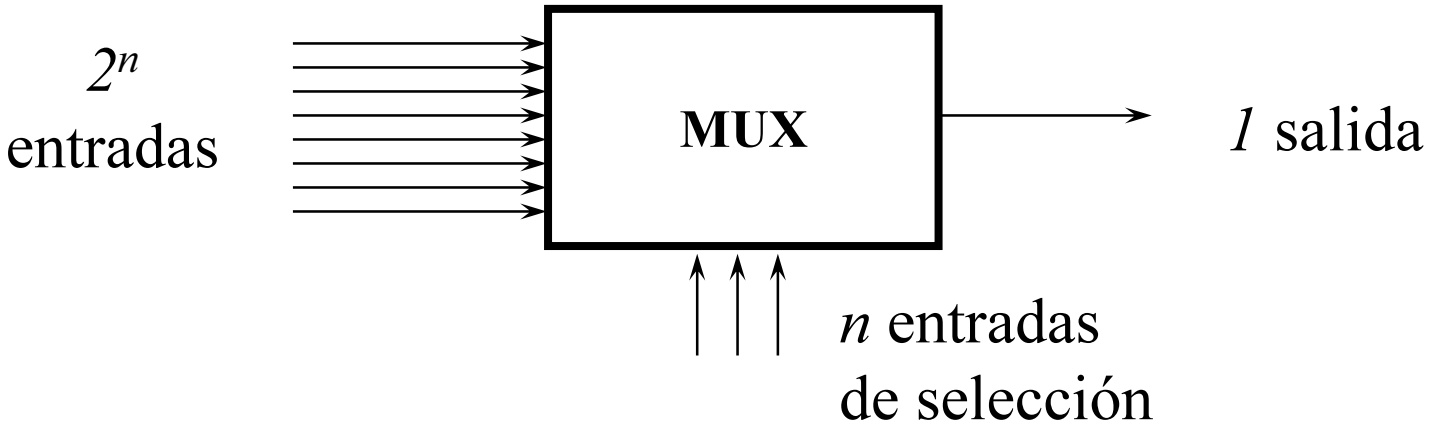
- Codificador comercial 74LS148



ENTRADAS									SALIDAS				
/Ei	/E7	/E6	/E5	/E4	/E3	/E2	/E1	/E0	/S2	/S1	/S0	/GS	/Eo
1	×	×	×	×	×	×	×	×	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	×	×	×	×	×	×	×	0	0	0	0	1
0	1	0	×	×	×	×	×	×	0	0	1	0	1
0	1	1	0	×	×	×	×	×	0	1	0	0	1
0	1	1	1	0	×	×	×	×	0	1	1	0	1
0	1	1	1	1	0	×	×	×	1	0	0	0	1
0	1	1	1	1	1	0	×	×	1	0	1	0	1
0	1	1	1	1	1	1	0	×	1	1	0	0	1
0	1	1	1	1	1	1	1	0	1	1	1	0	1

4. Multiplexores (I)

- Las líneas de selección indican qué entrada se obtendrá en la salida
- Muy utilizados en los caminos que sigue la información en los sistemas informáticos



Multiplexores (II)

ENTRADAS DE SELECCION		SALIDA
B	A	S
0	0	E0
0	1	E1
1	0	E2
1	1	E3

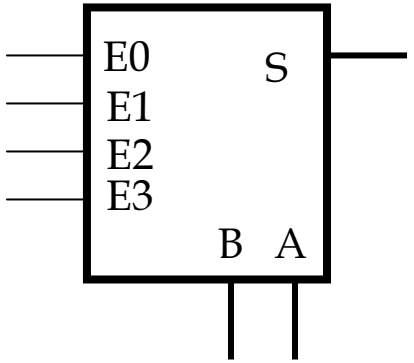
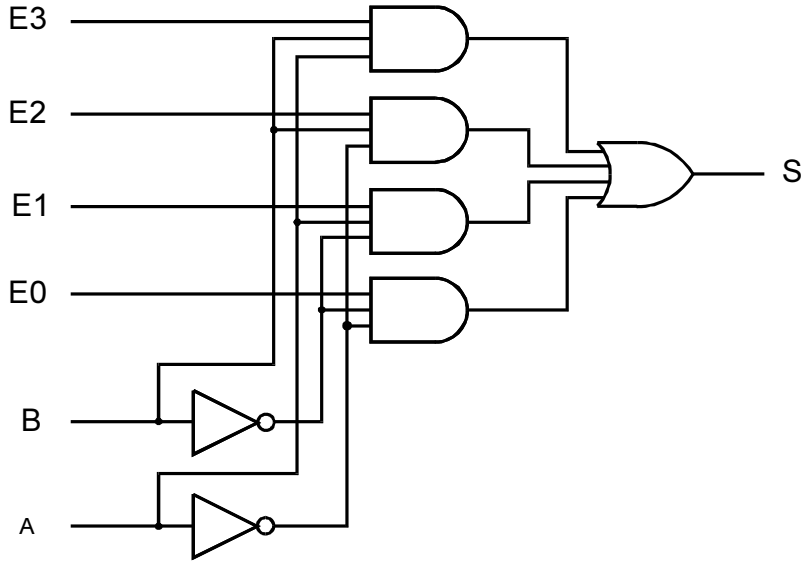


Tabla de verdad extendida

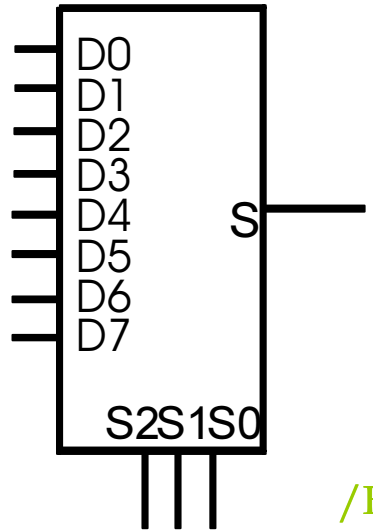
ENTRADAS DE SELECCION		ENTRADAS DE DATOS				SALIDA
B	A	E3	E2	E1	E0	S
0	0	X	X	X	0	0
0	0	X	X	X	1	1
0	1	X	X	0	X	0
0	1	X	X	1	X	1
1	0	X	0	X	X	0
1	0	X	1	X	X	1
1	1	0	X	X	X	0
1	1	1	X	X	X	1

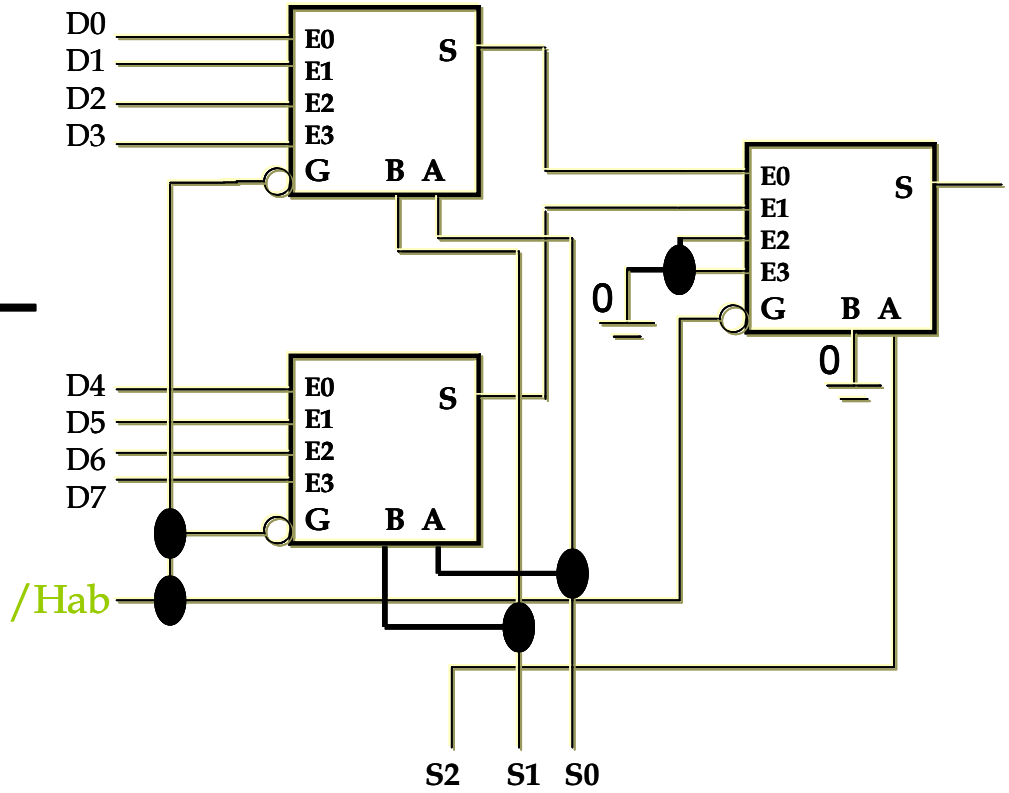


Multiplexores (III)

- Composición de multiplexores
  - MUX de 8 entradas de datos con MUX´s de 4 entradas de datos

S2	S1	S0	S
0	0	0	D0
0	0	1	D1
0	1	0	D2
0	1	1	D3
1	0	0	D4
1	0	1	D5
1	1	0	D6
1	1	1	D7



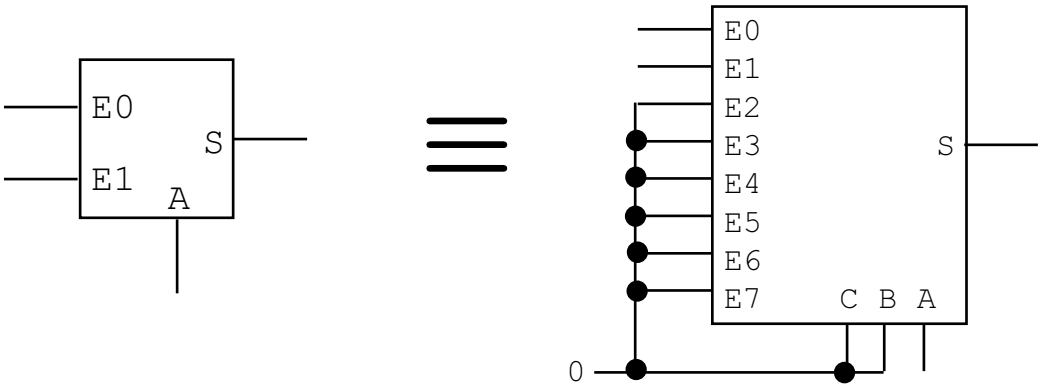


Multiplexores (IV)

- Ejercicio:
  - Se desea implementar un multiplexor de 1024 a 1 a base de una composición con multiplexores de 3 entradas de selección.
    - Indique el número de dichos multiplexores que serán necesarios.
    - Detalle el razonamiento seguido para obtener la respuesta.

Multiplexores (V)

- Solución:
  - Para cubrir 1024 entradas con multiplexores de 8 entradas necesitamos: 1024 / 8 = 128 multiplexores de 8 entradas.
  - Para cubrir las salidas de esos 128 multiplexores necesitamos: 128 / 8 = 16 multiplexores de 8 entradas.
  - Para cubrir las salidas de esos 16 multiplexores necesitamos: 16 / 8 = 2 multiplexores de 8 entradas.
  - Para cubrir las salidas de esos 2 multiplexores necesitamos: 1 multiplexor de 2 entradas
    - Se puede implementar con 1 multiplexor de 8 entradas.



128 + 16 + 2 + 1 =

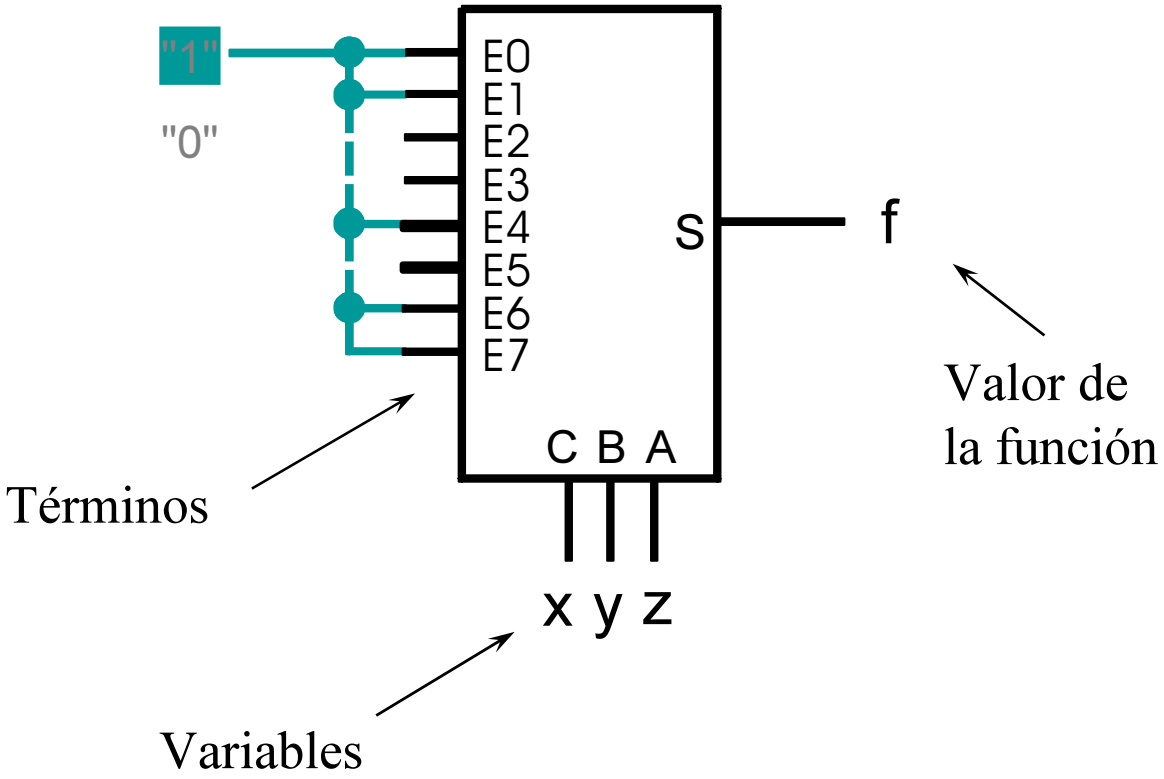
147 multiplexores de 8 entradas

Multiplexores (VI)

- Generación de funciones (I)

$$f = \sum_{x,y,z}(0,1,4,6,7)$$

x	y	z	f
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1



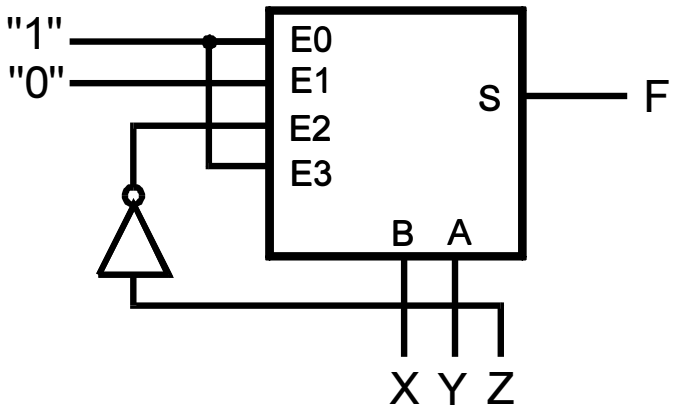
Multiplexores (VII)

- Generación de funciones (II)
  - Se pueden implementar funciones de n variables con un multiplexor de n-1 entradas de selección

X	Y	Z	F	F(Z)
0	0	0	1	1
0	0	1	1	
0	1	0	0	0
0	1	1	0	
1	0	0	1	$\overline{Z}$
1	0	1	0	
1	1	0	1	1
1	1	1	1	

X	Y	F	
0	0	1	E0
0	1	0	E1
1	0	$\overline{Z}$	E2
1	1	1	E3

$$F = \sum_{X,Y,Z}(0,1,4,6,7)$$



Multiplexores (VIII)

- Ejercicio:
  - Dada la función:

$$k = \prod_{d,c,b,a}(2,6,7) \bullet \prod_{\Phi}(1,3,4,5,14,15)$$

  - Impleméntese mediante un multiplexor y ninguna puerta adicional. Este multiplexor deberá disponer de una entrada de selección menos que el número de variables de la función. Muéstrese:
    - la tabla de verdad de la función k
    - la tabla de verdad reducida
    - el circuito resultante

Multiplexores (IX)

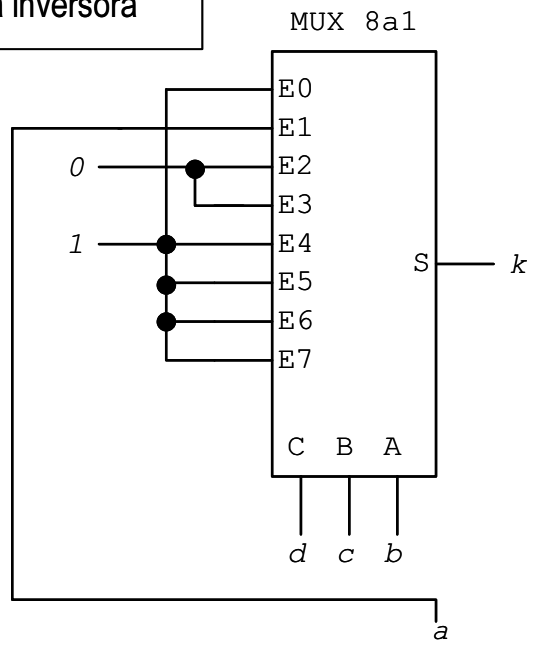
- Solución:

d	c	b	a	k
0	0	0	0	1
0	0	0	1	x
0	0	1	0	0
0	0	1	1	x
0	1	0	0	x
0	1	0	1	x
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	1
1	1	0	1	1
1	1	1	0	x
1	1	1	1	x

d	c	b	k(a)
0	0	0	1
0	0	1	a
0	1	0	x
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	x

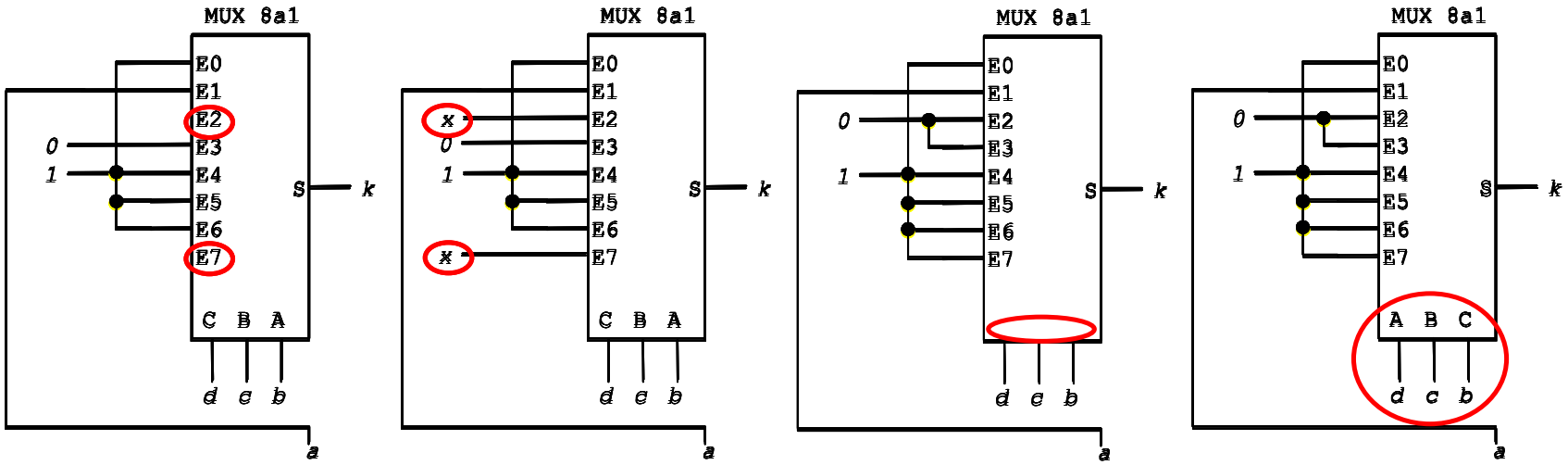
Este valor podría ser 1 o not(a), pero este último caso nos obligaría a añadir una puerta inversora

Este valor podría ser 0 o a



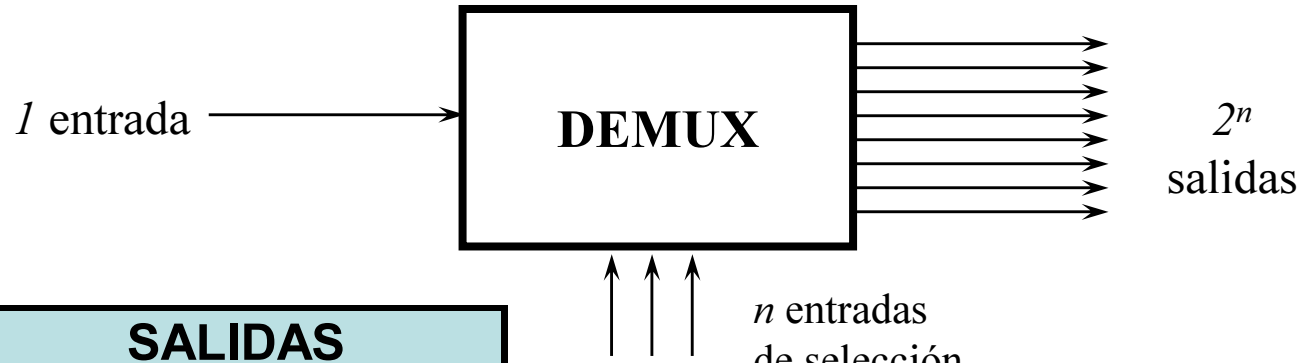
Multiplexores (X)

- Errores más comunes:
  - Dejar entradas al aire
  - Conectar “x” a las entradas
  - Conectar las variables de la función a entradas sin nombre
  - Conectar las variables de la función a entradas cuyo peso no es el correspondiente

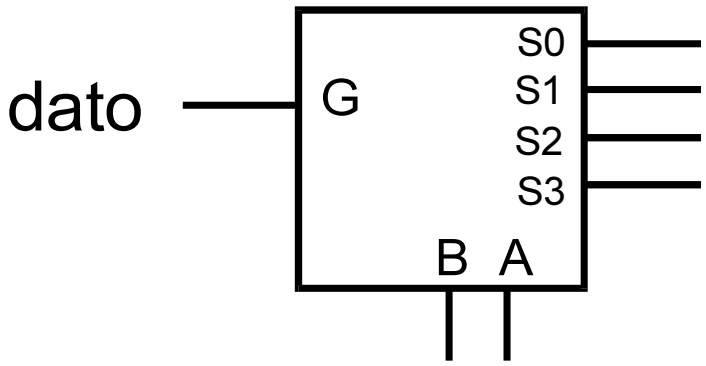


5. Demultiplexores

- Se pueden construir a partir de decodificadores
- Pueden ser utilizados para habilitar dispositivos

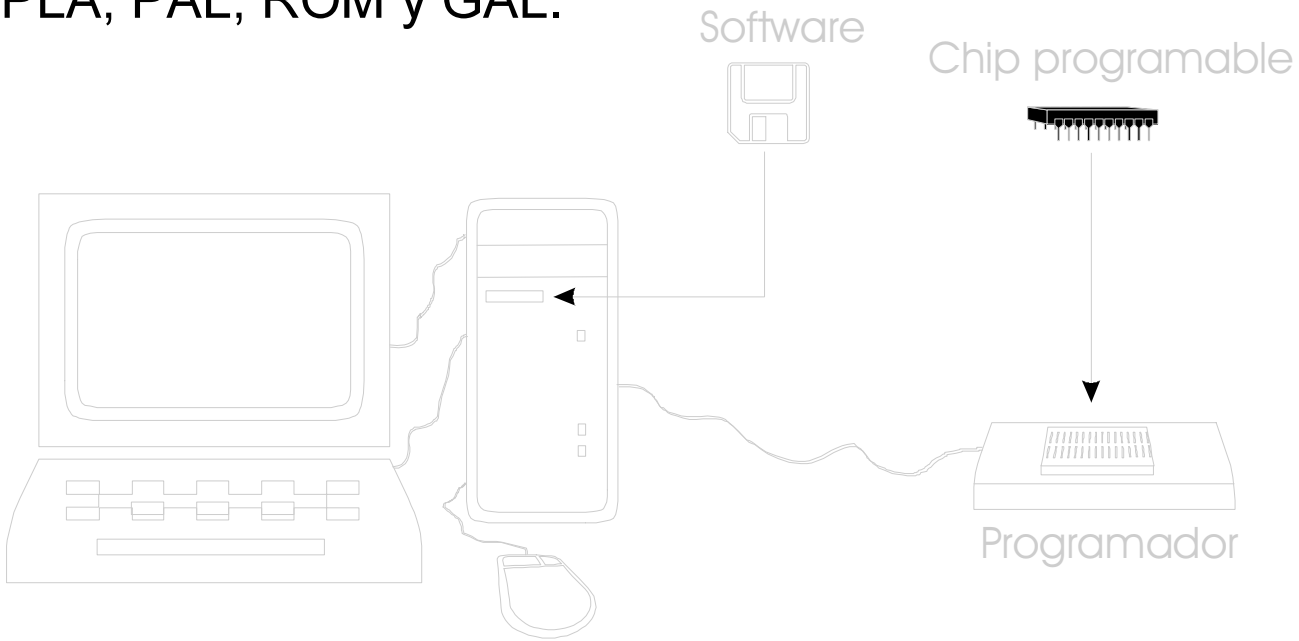


ENTRADAS			SALIDAS			
	SELECCION					
Dato	B	A	S3	S2	S1	S0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0



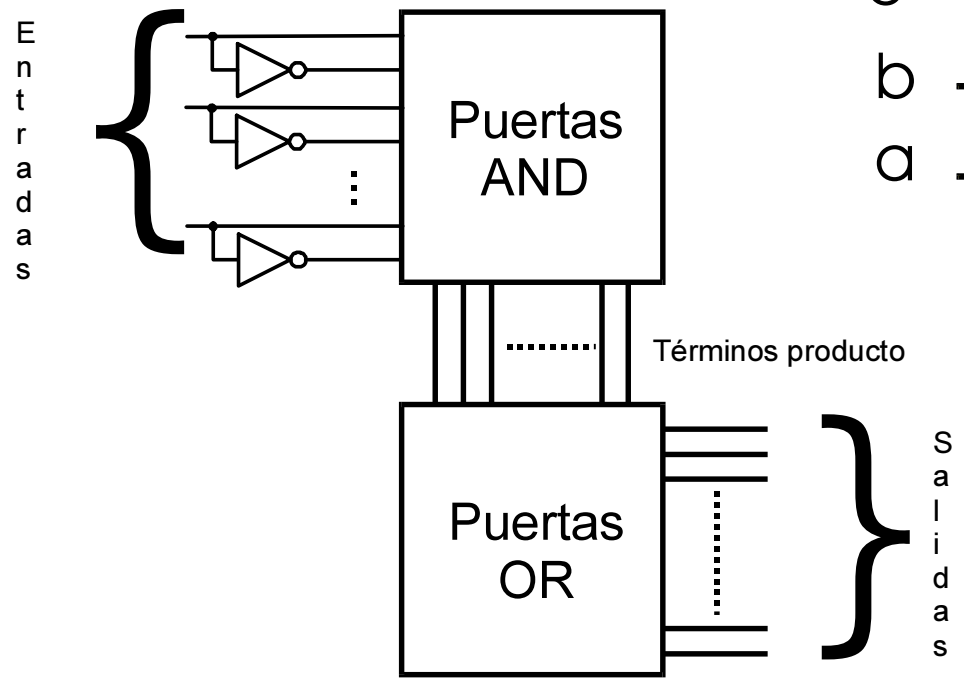
6. Dispositivos Lógicos Programables. PLDs (I)

- Son circuitos integrados que se utilizan con frecuencia para implementar funciones lógicas combinacionales.
- Ahorran espacio y reducen el coste de implementación, ya que en un mismo chip se implementan una o varias funciones.
- Tipos: PLA, PAL, ROM y GAL.



PLDs (II)

- Diagrama de bloques de una PLA
  matrices AND y OR



- Implementación de
  3 funciones mediante una PLA