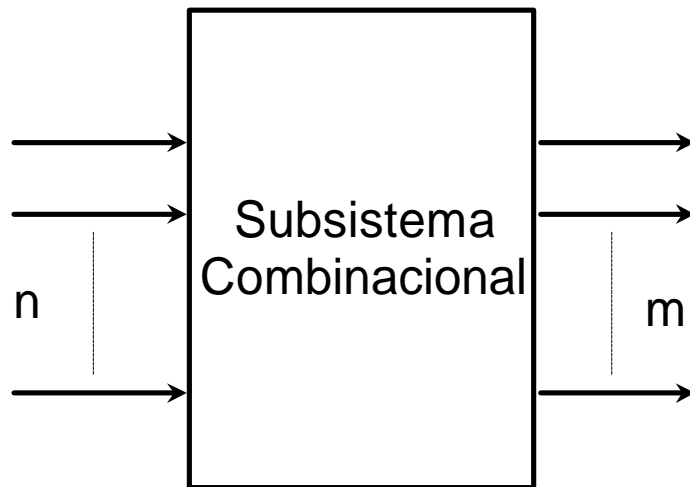


TEMA 5

SUBSISTEMAS COMBINACIONALES

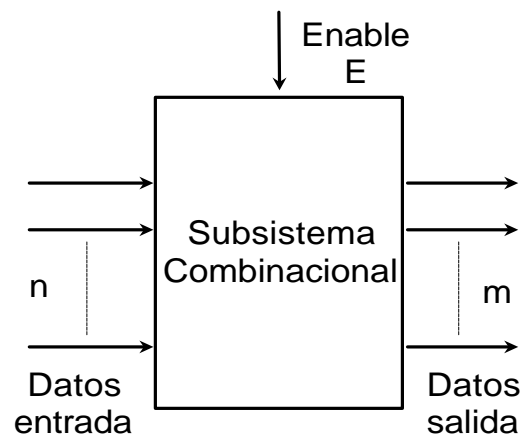
1. CIRCUITOS INTEGRADOS MSI/LSI



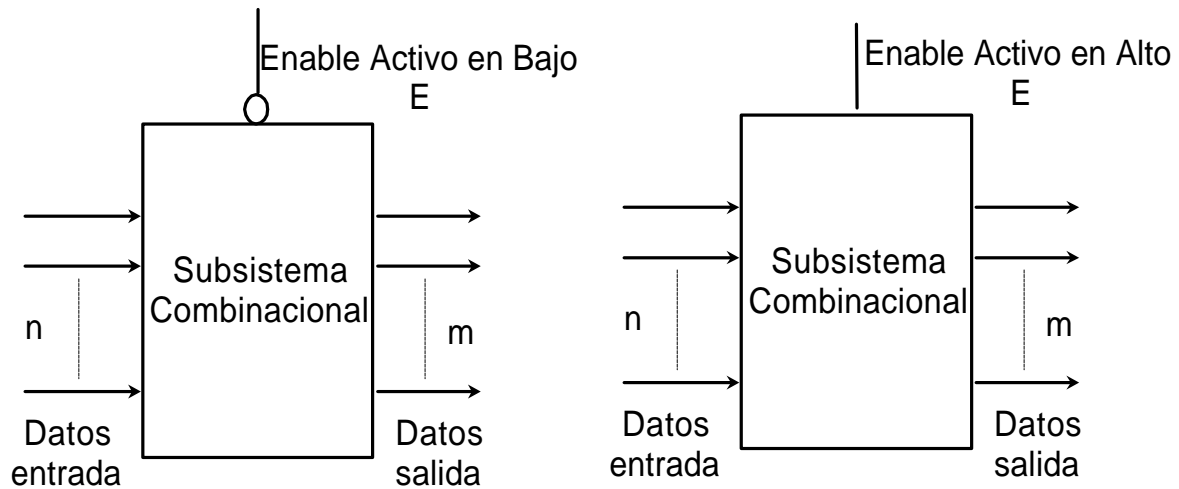
Estas líneas de entrada y salida pueden clasificarse en:

- líneas de datos
- líneas de control

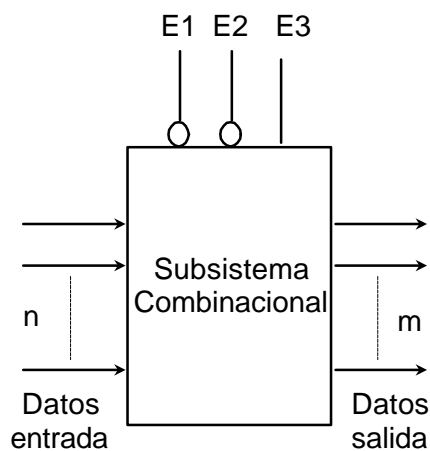
Una de las líneas de control más usadas es el Enable (Habilitador).



La habilitación de un subsistema por la línea de Enable, puede obtenerse cuando dicha línea de control tenga un nivel alto “1” o cuando tenga un nivel bajo “0”. En el primer caso, se dice que el Enable es activo en alta, mientras que para el segundo caso, el Enable es activo en baja.



Podemos encontrar subsistemas que tengan múltiples entradas de habilitación (enables).

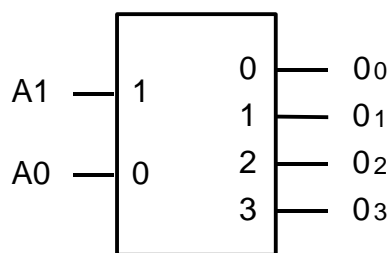


2. SUBSISTEMAS COMBINACIONES DE PROPÓSITO ESPECÍFICO

2.1 Decodificadores

El decodificador es un circuito integrado de n entradas de datos y m salidas de datos, donde $m \leq 2^n$. Cuando se cumple la igualdad, se dice que el decodificador es completo. Se especifican haciendo de la siguiente manera: *DEC $n:m$* o *DEC de n a m* .

Su propósito es generar los 2^n minterminos o maxtérminos asociados a las n variables de entrada, por tanto, el funcionamiento del mismo se deriva de esta propiedad, es decir, sólo hay una salida activa para cada combinación de entrada.



Entradas		Salidas			
A_1	A_0	O_0	O_1	O_2	O_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Las salidas tienen la siguiente expresión lógica:

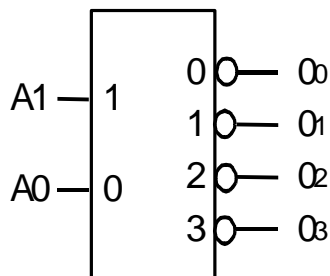
$$O_0 = A_1' \bullet A_0'$$

$$O_1 = A_1' \bullet A_0$$

$$O_2 = A_1 \bullet A_0'$$

$$O_3 = A_1 \bullet A_0$$

En el caso de que las salidas del decodificador de 2 a 4 fueran activas en bajo.



Entradas		Salidas			
A_1	A_0	O_0	O_1	O_2	O_3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

Las salidas tienen la siguiente expresión lógica:

$$O_0 = A_1 + A_0$$

$$O_1 = A_1 + A_0'$$

$$O_2 = A_1' + A_0$$

$$O_3 = A_1' + A_0'$$

Entradas			Salidas			
E	A ₁	A ₀	O ₀	O ₁	O ₂	O ₃
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

Las salidas tienen la siguiente expresión lógica:

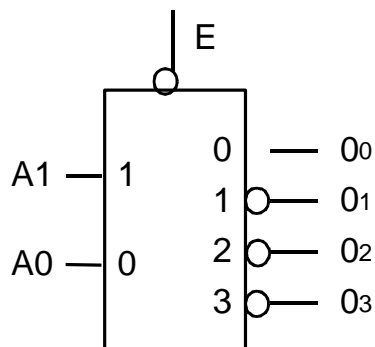
$$O_0 = A_1' \bullet A_0' \bullet E$$

$$O_1 = A_1' \bullet A_0 \bullet E$$

$$O_2 = A_1 \bullet A_0' \bullet E$$

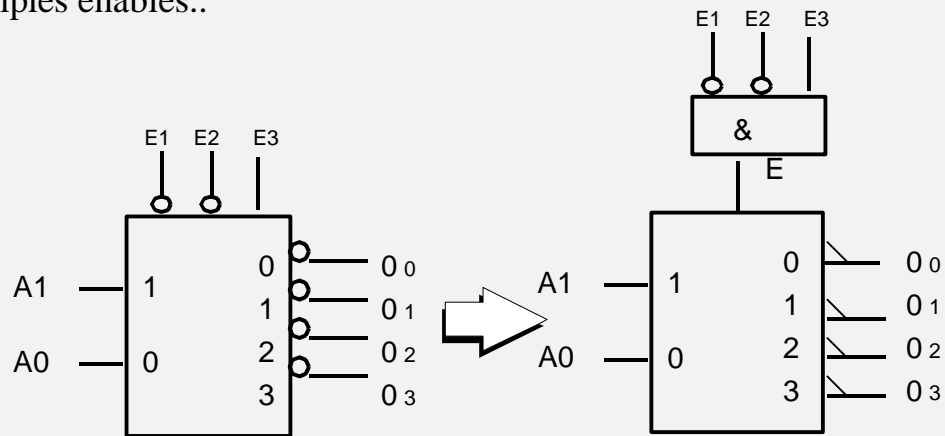
$$O_3 = A_1 \bullet A_0 \bullet E$$

Podemos decir que la expresión de salida de un decodificador con salidas activas en alto que posee entrada de habilitación también activa en alto equivale a $O_i = m_i \bullet E$, donde m_i es el mintermino asociado a la salida i . Si la entrada de habilitación fuese activa en bajo, la expresión de la salida i del decodificador sería $O_i = m_i \bullet E'$.



Entradas			Salidas			
E	A ₁	A ₀	O ₀	O ₁	O ₂	O ₃
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

Para múltiples enables..



2.1.1 Aplicaciones de los decodificadores

Ejemplo: Implementar con un decodificador las siguientes funciones de conmutación

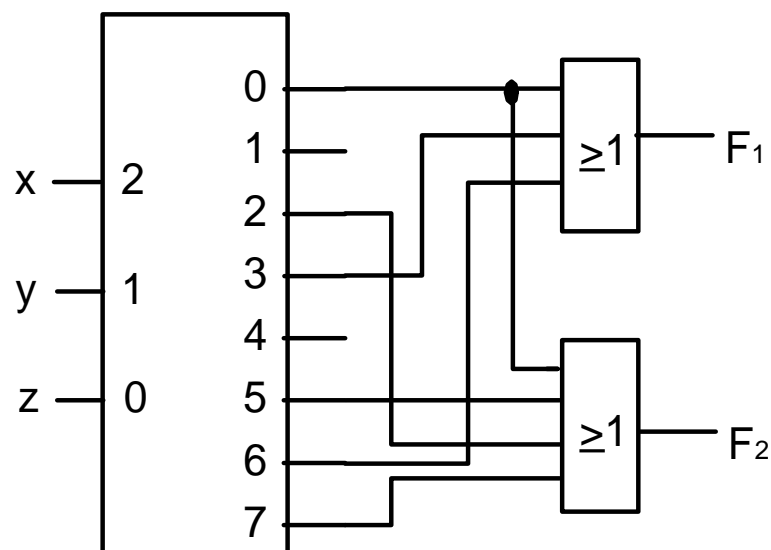
$$F_1 = \Sigma(0,3,6)$$

$$F_2 = \Pi(1,3,4,6)$$

Solución 1.- El decodificador tiene las salidas activas en alto.

$$F_2 = \Pi(1,3,4,6) = \Sigma(0,2,5,7)$$

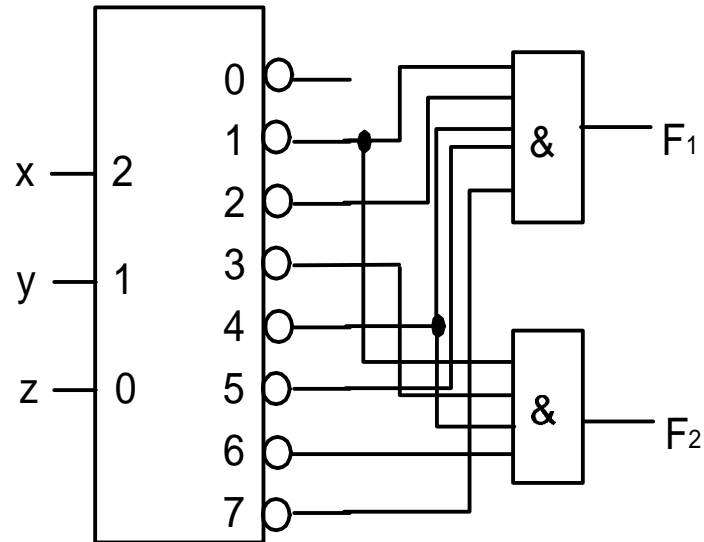
Y el circuito resultante es:



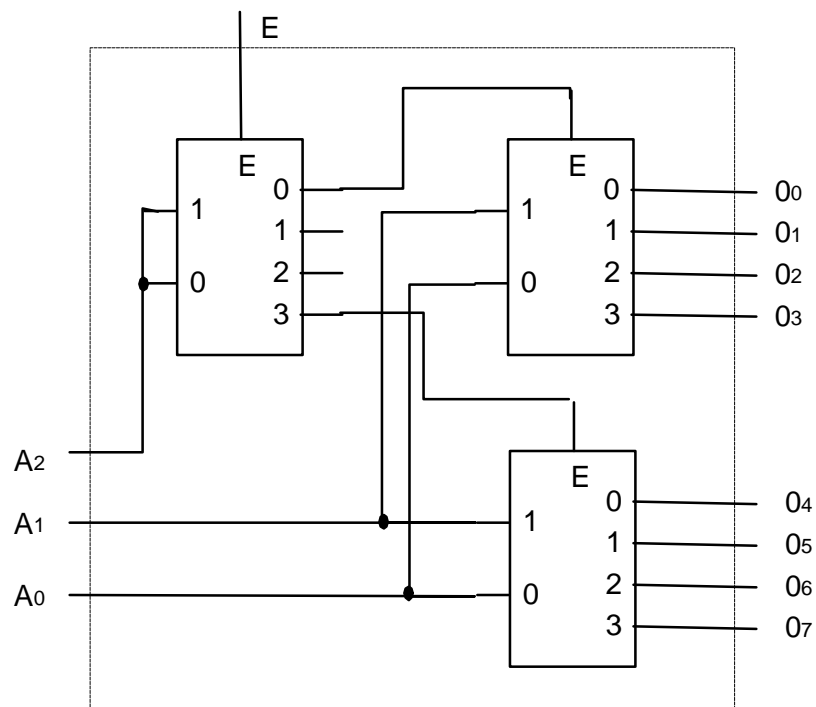
Solución 2.- El decodificador tiene las salidas activas en bajo.

$$F_1 = \Sigma(0,3,6) = \Pi(1,2,4,5,7)$$

Y el circuito resultante es:

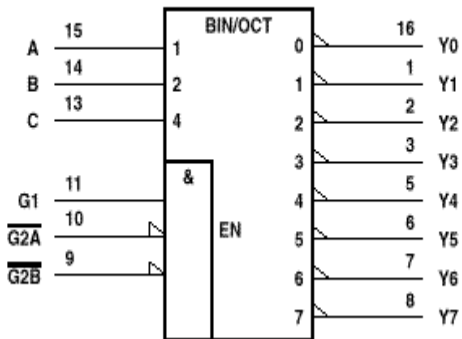


2.1.2 Asociación de decodificadores



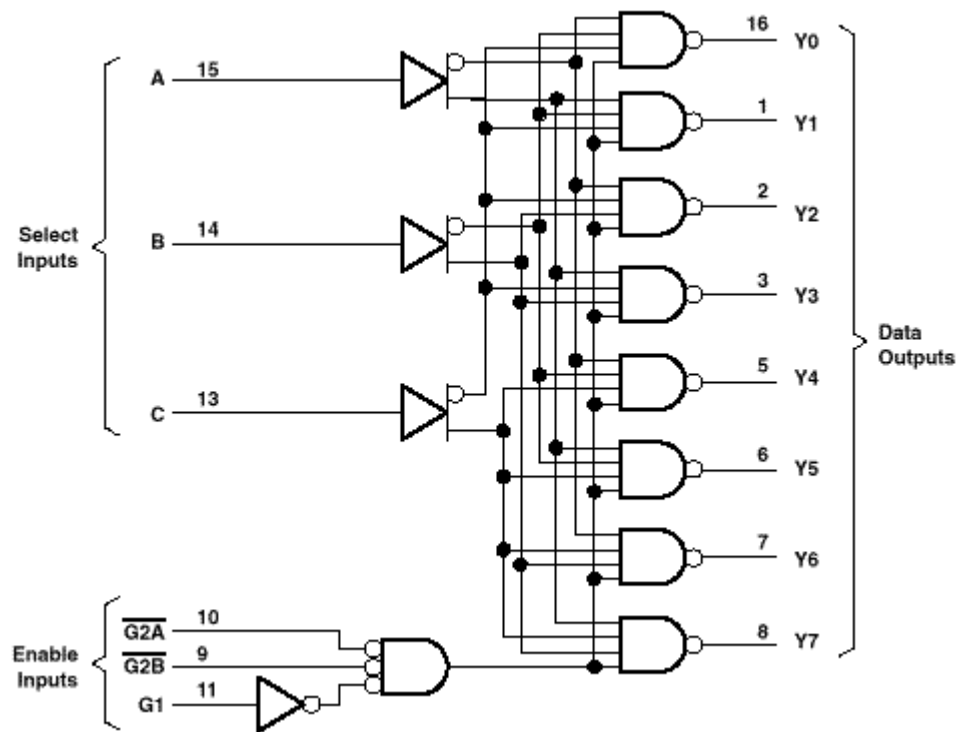
Decodificador comercial

logic symbols (alternatives)[†]

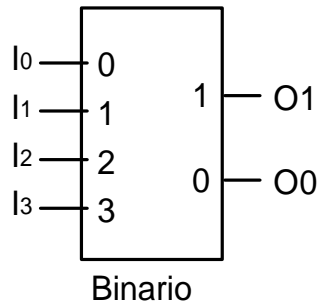


FUNCTION TABLE

ENABLE INPUTS			SELECT INPUTS			OUTPUTS							
G1	$\overline{G2A}$	$\overline{G2B}$	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	X	H	H	H	H	H	H	H	H
X	X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	L	H	H	H	H	H	L	H	H	H	H
H	L	L	H	L	L	H	H	H	H	L	H	H	H
H	L	L	H	L	H	H	H	H	H	H	L	H	H
H	L	L	H	H	L	H	H	H	H	H	H	L	H
H	L	L	H	H	H	H	H	H	H	H	H	H	L

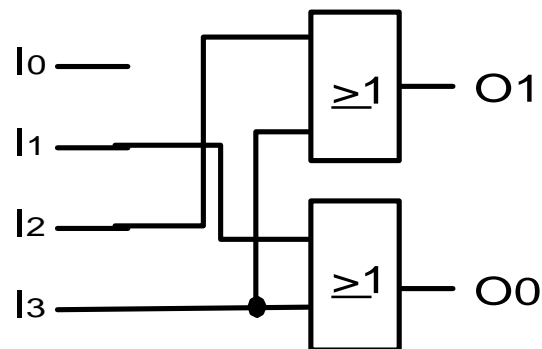


2.2 Codificadores

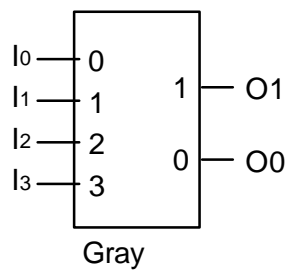


Entradas				Salidas	
I ₀	I ₁	I ₂	I ₃	O ₁	O ₀
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

I



Existen otros tipos de codificadores, como el Gray, representado en la siguiente tabla.

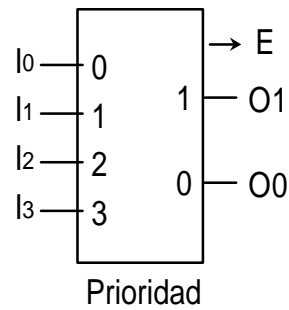


Entradas				Salidas	
I ₀	I ₁	I ₂	I ₃	O ₁	O ₀
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	1
0	0	0	1	1	0

$$O_0 = I_1 + I_2$$

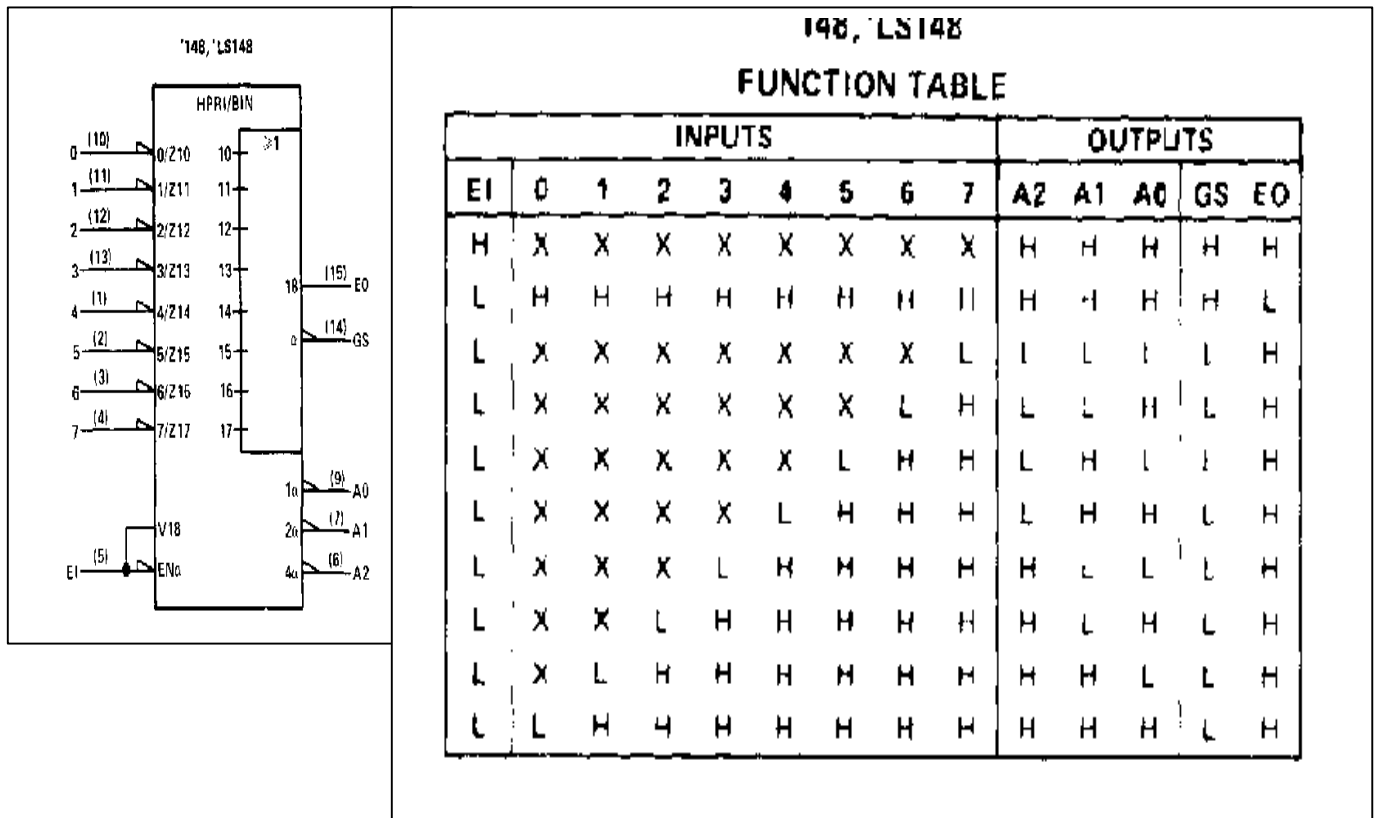
$$O_1 = I_2 + I_3$$

Existe otro tipo de codificador, denominado CODIFICADOR DE PRIORIDAD,



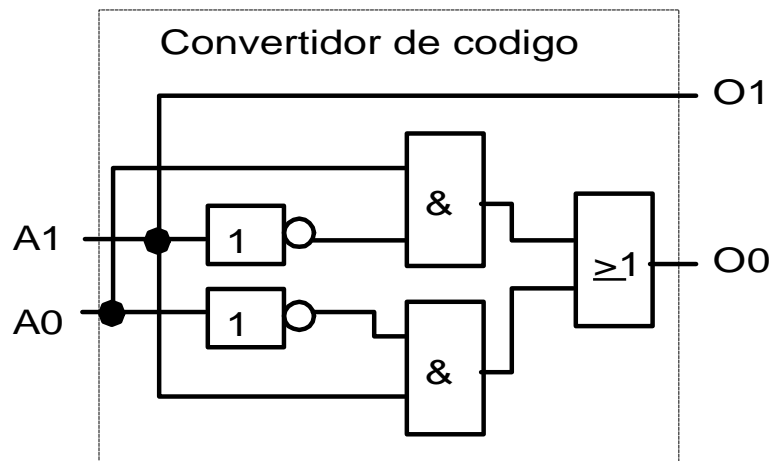
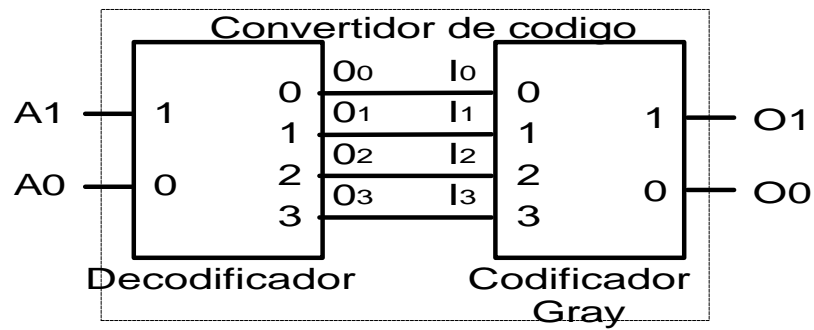
Entradas				Salidas		
I ₃	I ₂	I ₁	I ₀	E	O ₁	O ₀
1	x	x	X	0	1	1
0	1	x	x	0	1	0
0	0	1	x	0	0	1
0	0	0	1	0	0	0
0	0	0	0	1	0	0

Codificador comercial

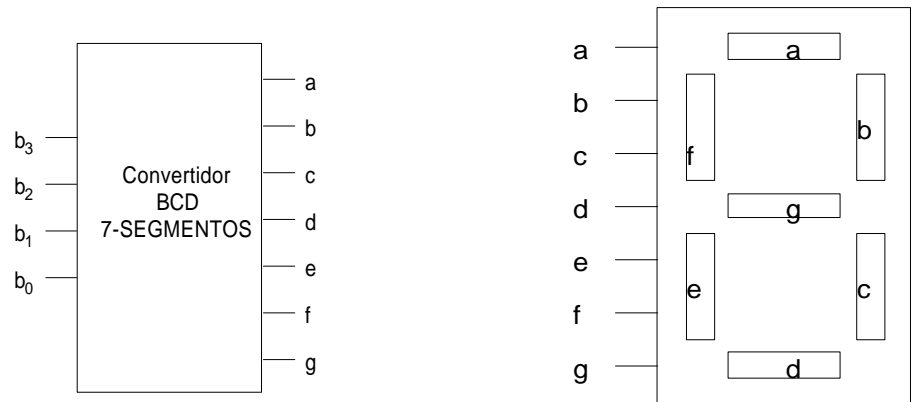


2.3 Convertidores de código

BINARIO/GRAY
GRAY/BINARIO
BCD/GRAY
BCD/7-SEGMENTOS
....

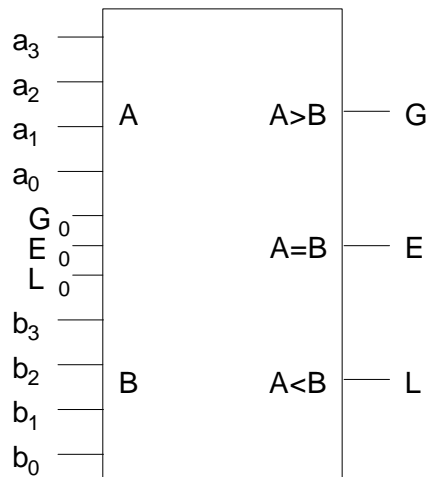


Un tipo de convertidor muy utilizado es el convertidor BCD/7-SEGMENTOS. El código 7-segmentos es utilizado para iluminar los distintos segmentos de un display numérico.



b_3	b_2	b_1	b_0	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	0	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1

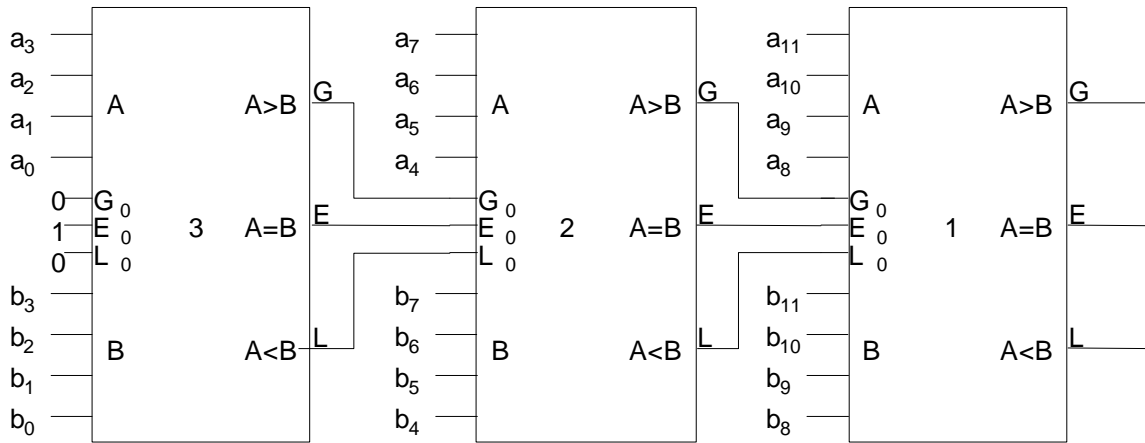
2.4 Comparadores de magnitud



FUNCTION TABLE

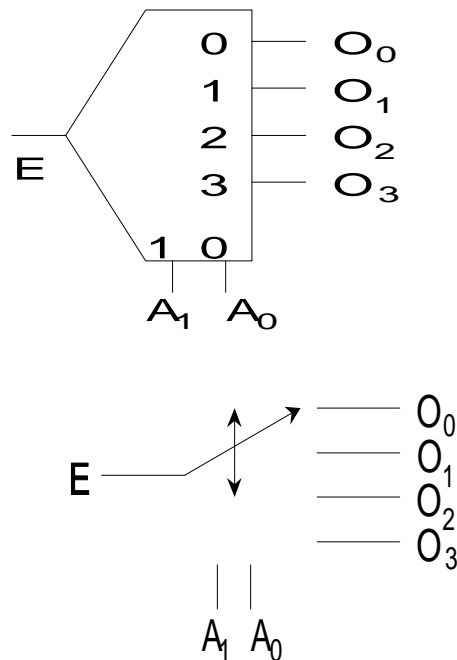
COMPARING INPUTS				CASCADING INPUTS			OUTPUTS		
A3, B3	A2, B2	A1, B1	A0, B0	A > B	A < B	A = B	A > B	A < B	A = B
A3 > B3	X	X	X	X	X	X	H	L	L
A3 < B3	X	X	X	X	X	X	L	H	L
A3 = B3	A2 > B2	X	X	X	X	X	H	L	L
A3 = B3	A2 < B2	X	X	X	X	X	L	H	L
A3 = B3	A2 = B2	A1 > B1	X	X	X	X	H	L	L
A3 = B3	A2 = B2	A1 < B1	X	X	X	X	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 > B0	X	X	X	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 < B0	X	X	X	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	L	L	H	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	H	L	L	H	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	X	X	H	L	L	H
A3 = B3	A2 = B2	A1 = B1	A0 = B0	H	H	L	L	L	L
A3 = B3	A2 = B2	A1 = B1	A0 = B0	L	L	L	H	H	L

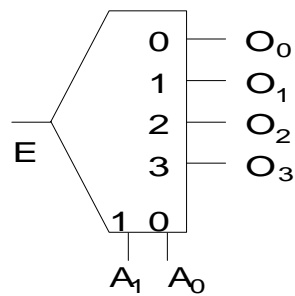
2.4.1. Asociación de comparadores



2.5 Demultiplexor

Es un circuito MSI que dispone de n líneas de control y $2^n + 1$ líneas de datos. 2^n líneas de datos son de salida (también llamadas canales) frente a una única línea de entrada. Se pueden designar mediante DEMUX 1:m, donde m es el número de canales. Recordemos que los canales deben ser una potencia del número 2. Otra forma de designarlos es DEMUX de n entradas de control. El símbolo usado para representar al demultiplexor es el mostrado en la siguiente figura

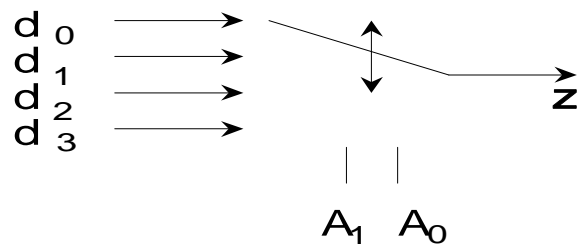
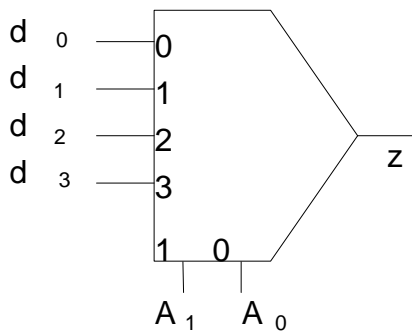




Entradas			Salidas			
E	A ₁	A ₀	O ₀	O ₁	O ₂	O ₃
0	X	X	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

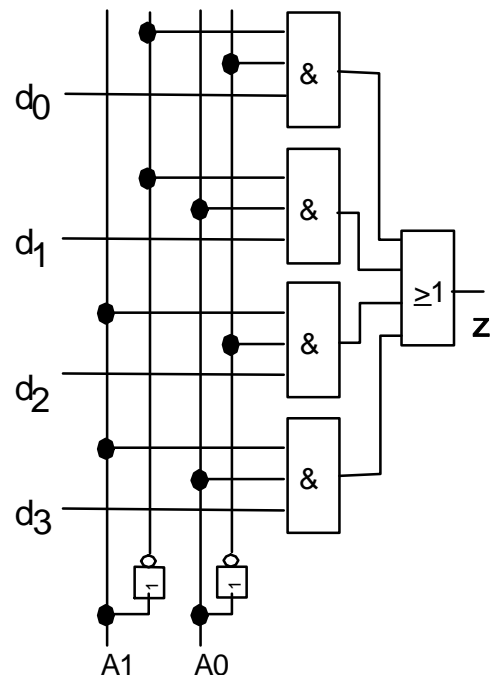
3. SUBSISTEMAS DE PROPÓSITO GENERAL

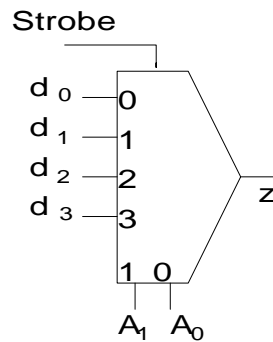
3.1 Multiplexor



A ₁	A ₀	Z
0	0	d ₀
0	1	d ₁
1	0	d ₂
1	1	d ₃

$$Z = d_0 A_1' A_0' + d_1 A_1' A_0 + d_2 A_1 A_0' + d_3 A_1 A_0$$





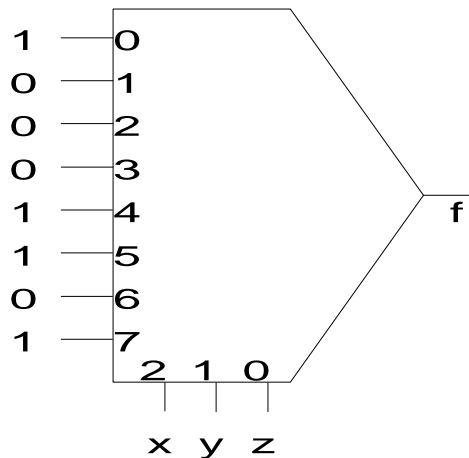
3.1.1 El multiplexor como generador de funciones

El teorema de expansión de Shannon estudiado en el tema 3 indica que cualquier función de conmutación completa de n variables puede ser expresada como

$$F(x_1, \dots, x_n) = \sum f(i) * m_i(x_1, \dots, x_n)$$

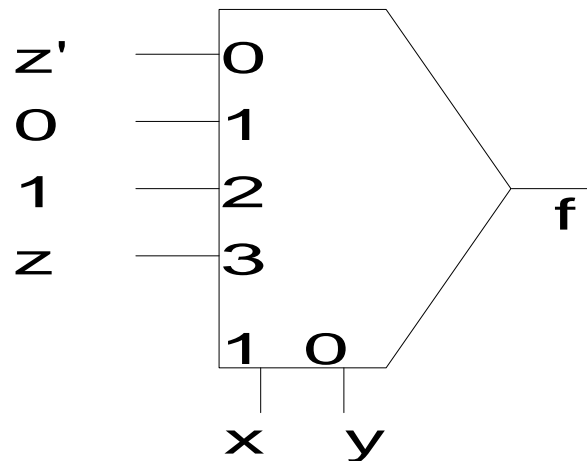
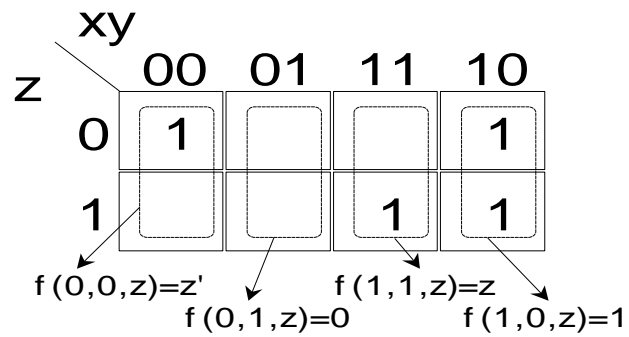
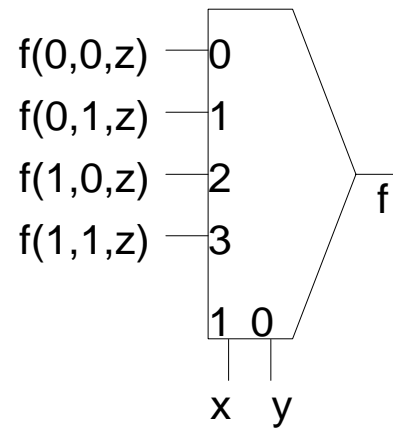
$$Z = \sum d_i * m_i(x_1, \dots, x_n)$$

Ejemplo1 : Usando un multiplexor de 8 canales, implementar la función de conmutación $f = \sum(0, 4, 5, 7)$



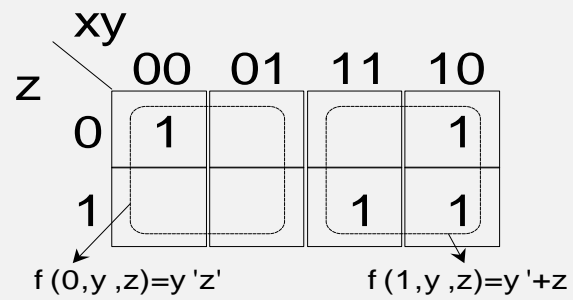
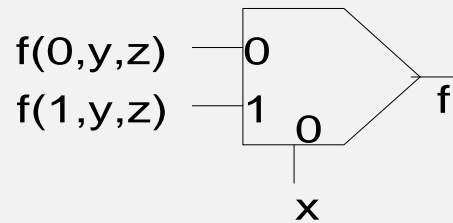
Ejemplo 2: Diseñar la función de conmutación $f = \Sigma(0,4,5,7)$ usando multiplexores de 4 canales.

$$F(x,y,z) = d_0 x'y' + d_1 x'y + d_2 xy' + d_3 xy$$

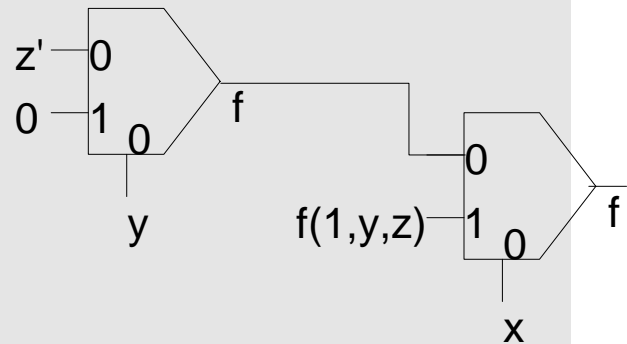
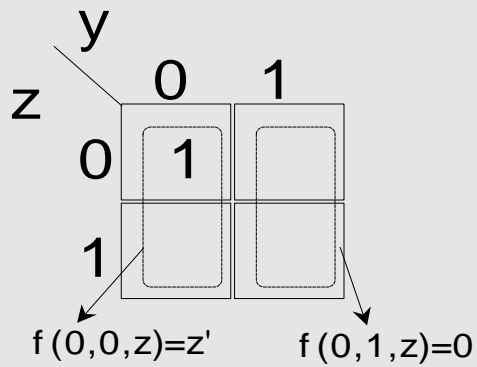
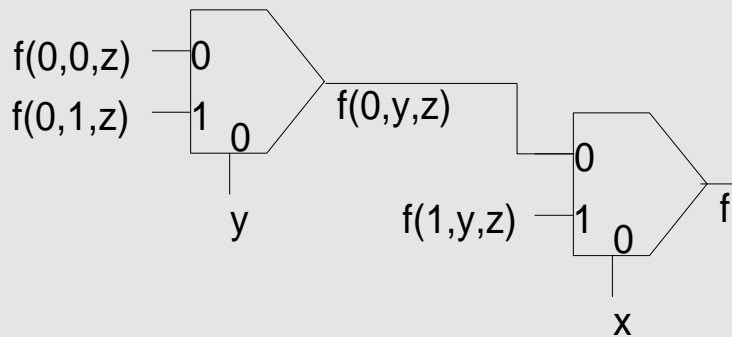


Ejemplo 3. Diseñar la función de conmutación $f = \Sigma(0,4,5,7)$ usando multiplexores de 2 canales.

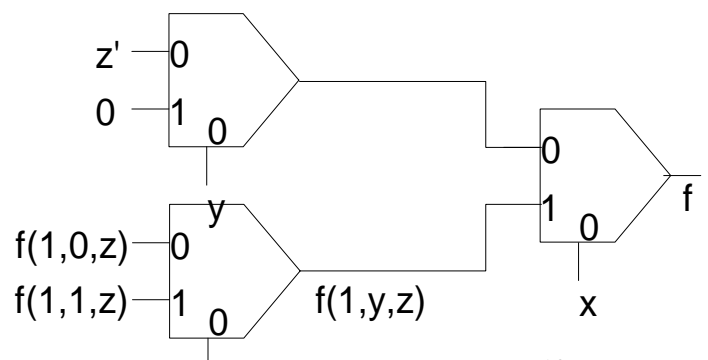
$$F(x,y,z) = d_0 x' + d_1 x$$

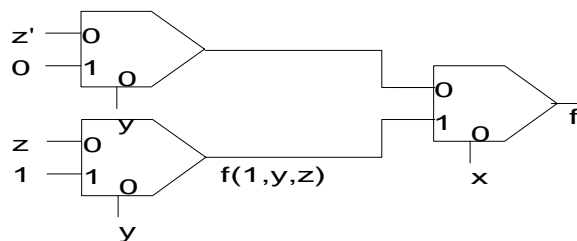
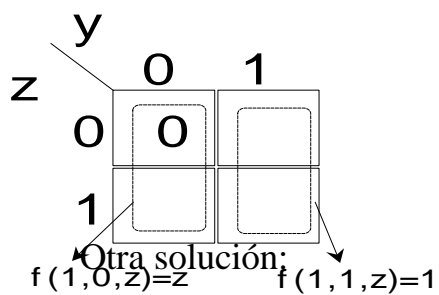


$$F(0,y,z) = d_0 y' + d_1 y$$

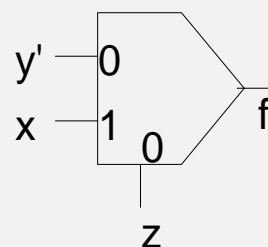
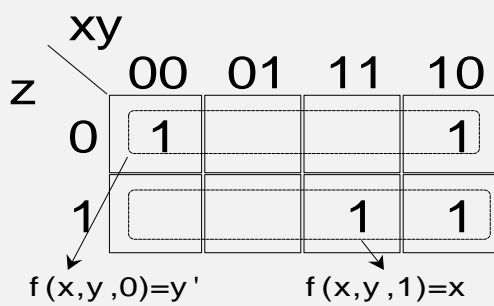
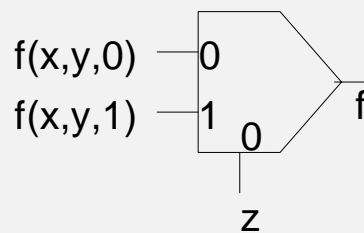


$$F(1,y,z) = d_0 y' + d_1 y$$



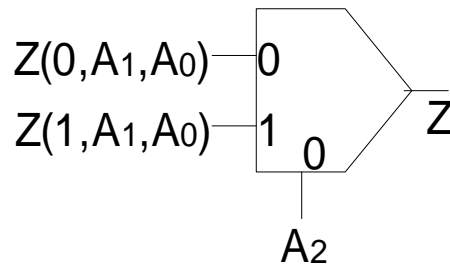


$$F(x,y,z) = d_0 z' + d_1 z$$

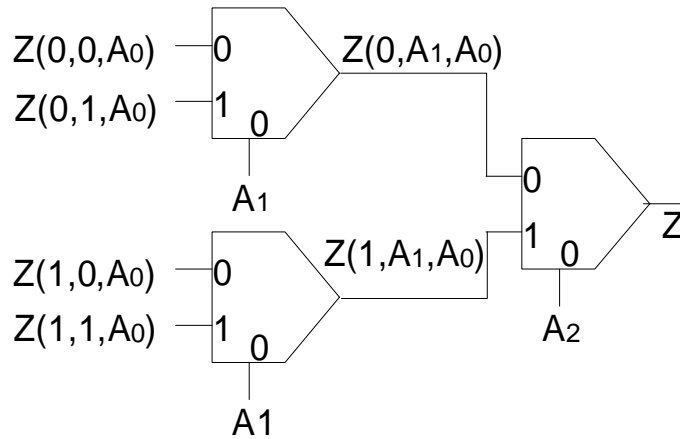


3.1.2 Asociación de multiplexores

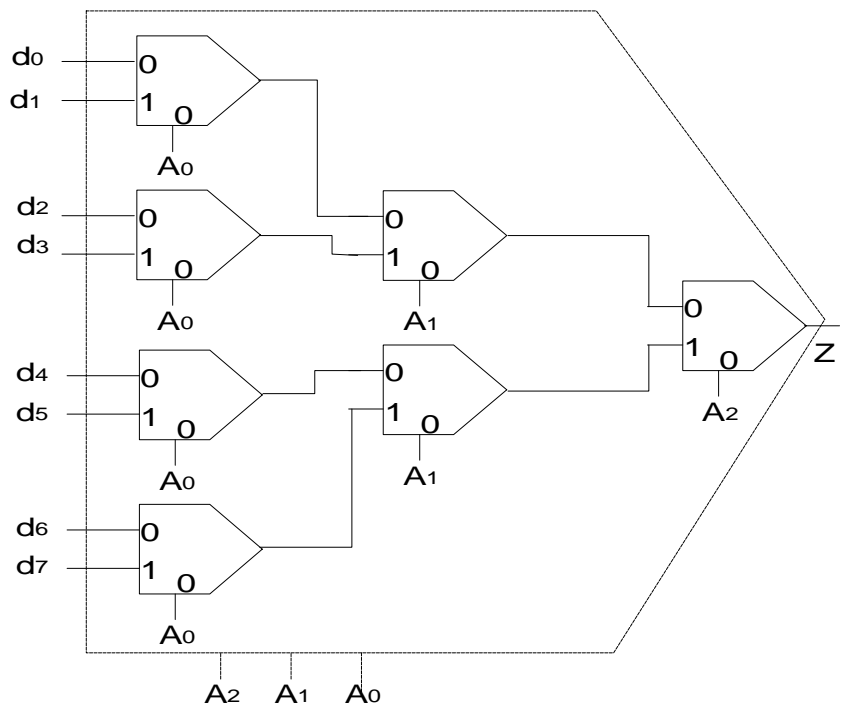
$$Z(A_2A_1A_0) = d_0 A_2' A_1' A_0' + d_1 A_2' A_1' A_0 + d_2 A_2' A_1 A_0' + d_3 A_2' A_1 A_0 + d_4 A_2 A_1' A_0' + d_5 A_2 A_1' A_0 + d_6 A_2 A_1 A_0' + d_7 A_2 A_1 A_0$$



La función residuo $Z(0A_1A_0) = d_0 A_1' A_0' + d_1 A_1' A_0 + d_2 A_1 A_0' + d_3 A_1 A_0$ y la función residuo $Z(1A_1A_0) = d_4 A_1' A_0' + d_5 A_1' A_0 + d_6 A_1 A_0' + d_7 A_1 A_0$.

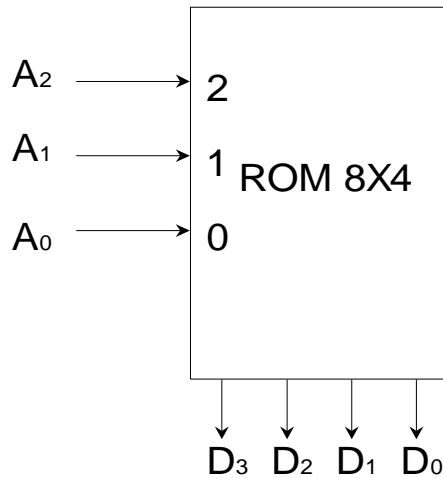


$$\begin{aligned} Z(00A_0) &= d_0 A_0' + d_1 A_0 \\ Z(01A_0) &= d_2 A_0' + d_3 A_0 \\ Z(10A_0) &= d_4 A_0' + d_5 A_0 \\ Z(11A_0) &= d_6 A_0' + d_7 A_0 \end{aligned}$$



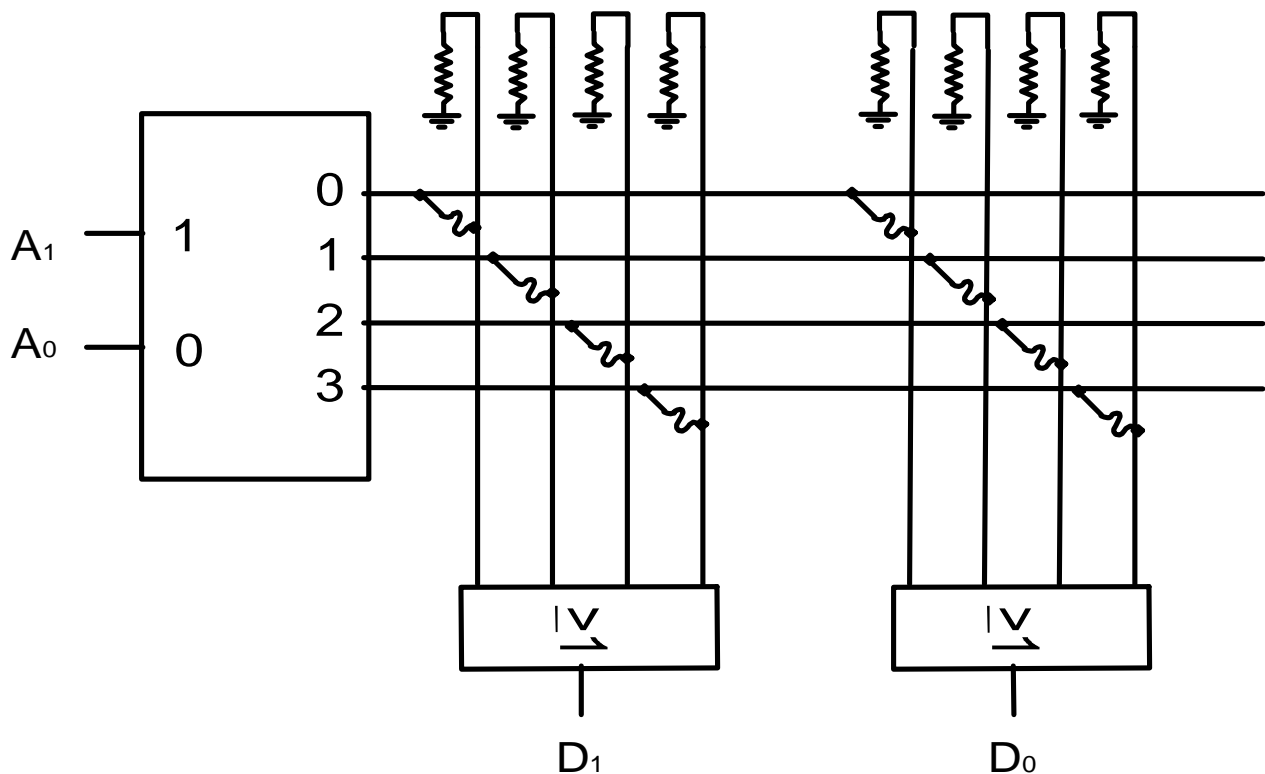
3.2 ROM (Read Only Memory)

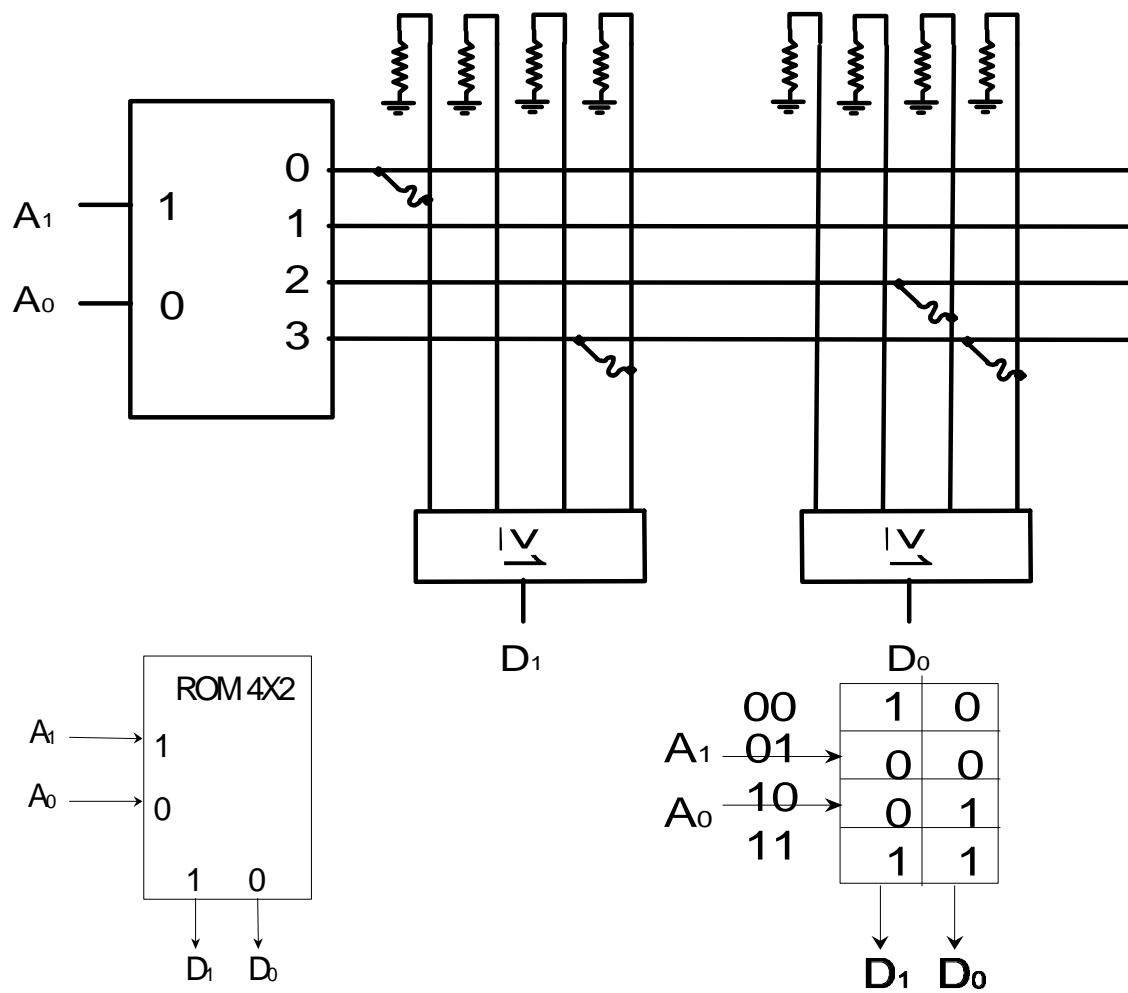
$$\text{Capacidad(bits)} = 2^n \times m$$



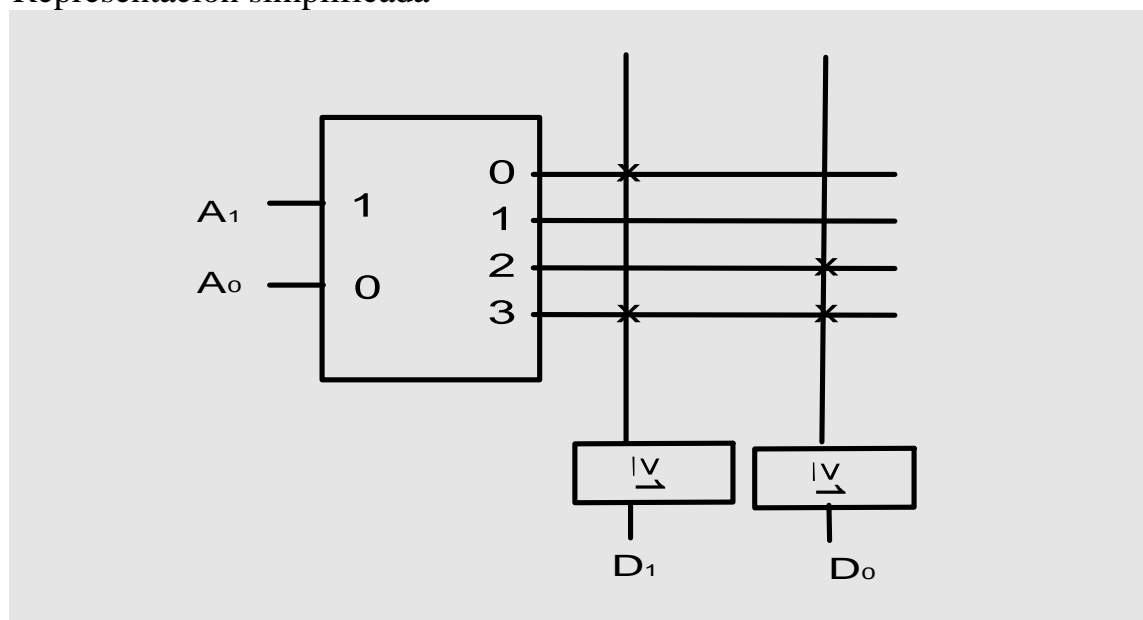
A_2	000	1	1	0	0
A_1	001	0	0	0	1
	010	1	0	1	0
A_0	011	1	1	1	0
	100	0	0	1	1
	101	0	0	0	0
	110	0	0	0	1
	111	1	0	0	0
		D_3	D_2	D_1	D_0

Una ROM de $2^n \times m$, está formada por un decodificador de m líneas de entrada, $2^n \times m$ fusibles o interconexiones y m puertas OR. La siguiente figura esquematiza una ROM de 4x2.

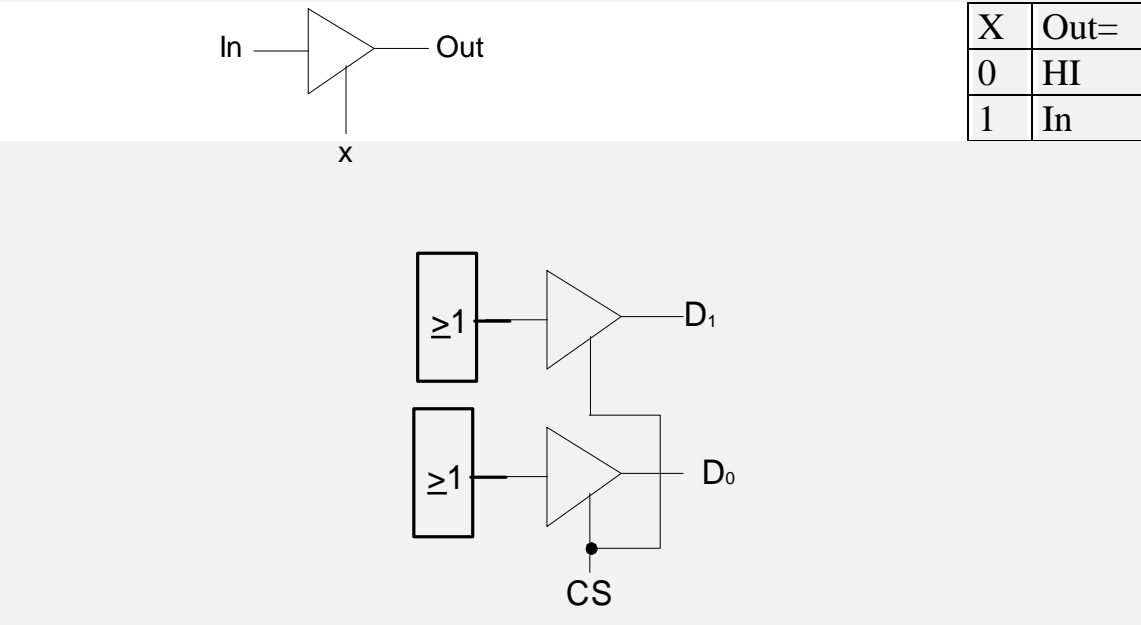




Representación simplificada



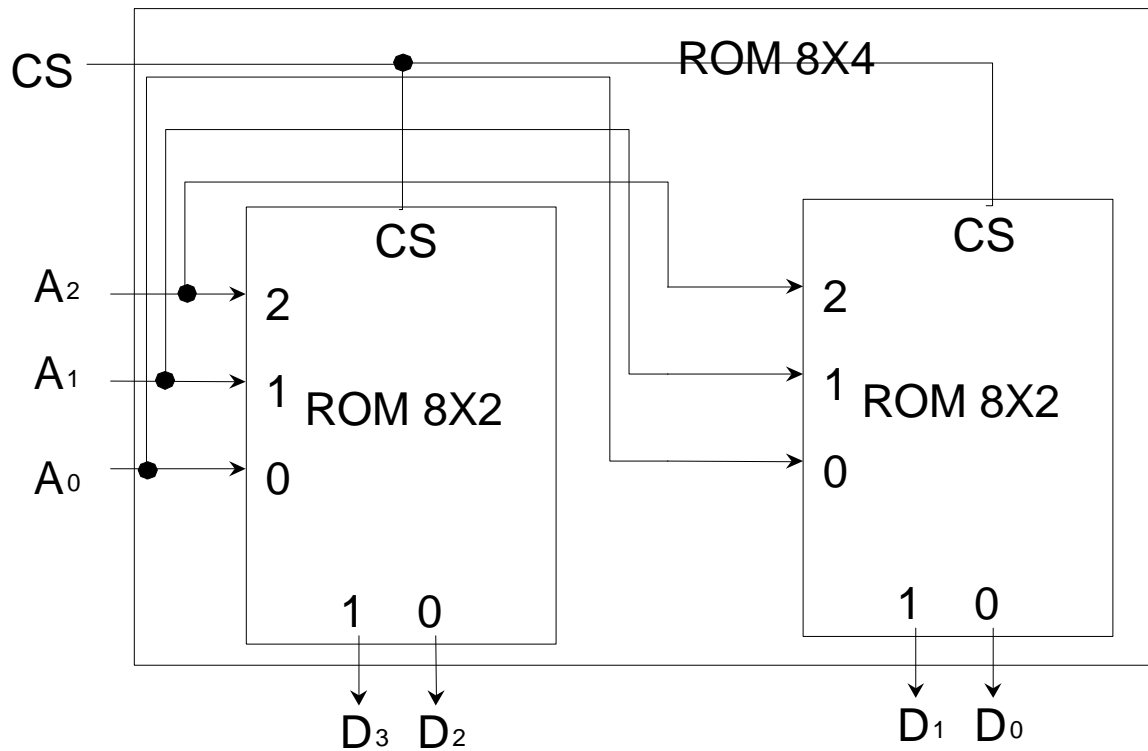
Las salidas de datos disponen de buffers triestado



3.2.2 Asociación de ROM

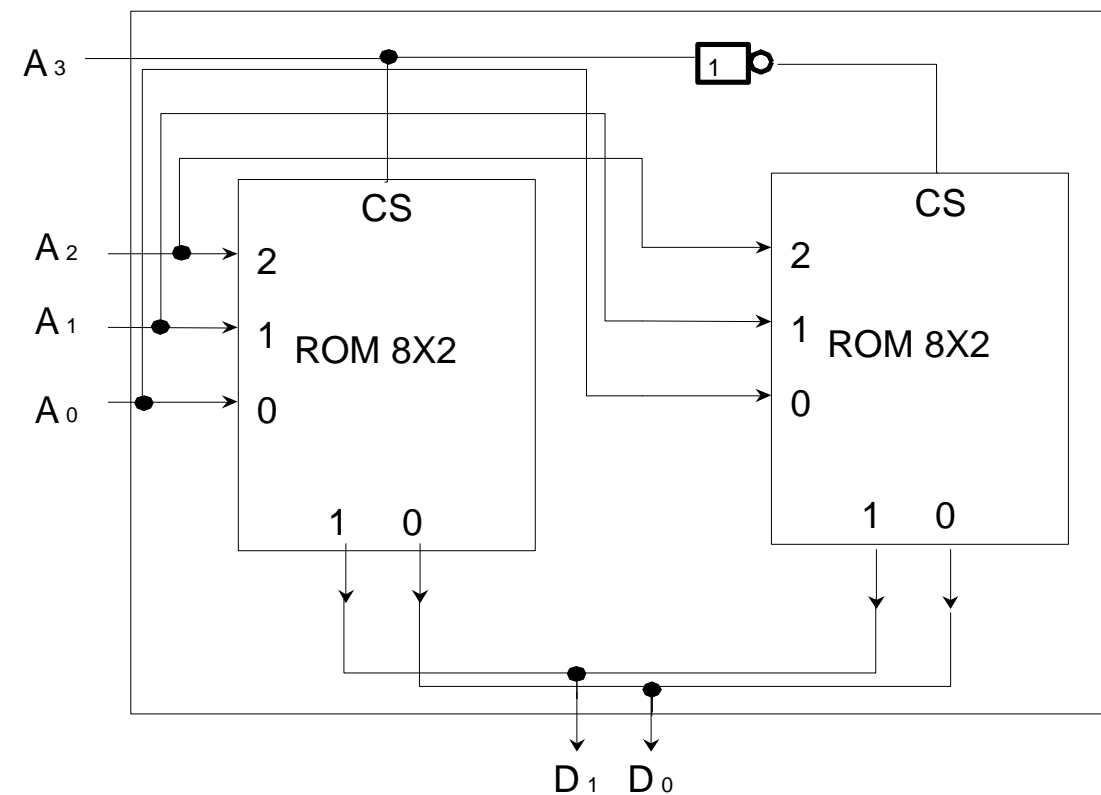
Ejemplo 1

Se dispone de ROM de 3X2 y se desea construir una ROM de 3x4.



Ejemplo 2

Se dispone de ROMs de 8x2, y se desea construir una ROM de 16x2.



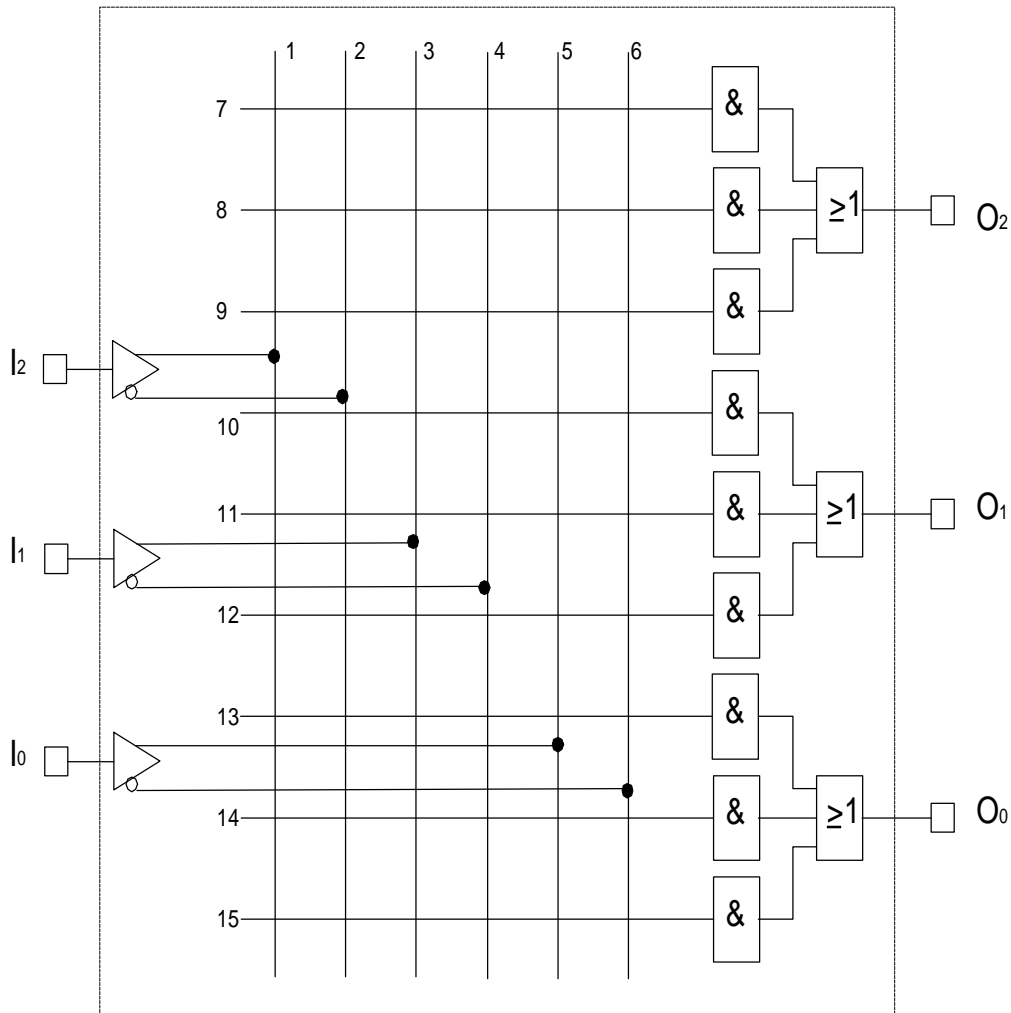
3.3 PLD (Programmable Logic Devices)

En estos dispositivos podemos encontrar dos planos o matrices (arrays): el plano AND, que genera los términos productos necesarios; y el plano OR que genera la suma de los términos productos resultantes del plano anterior..

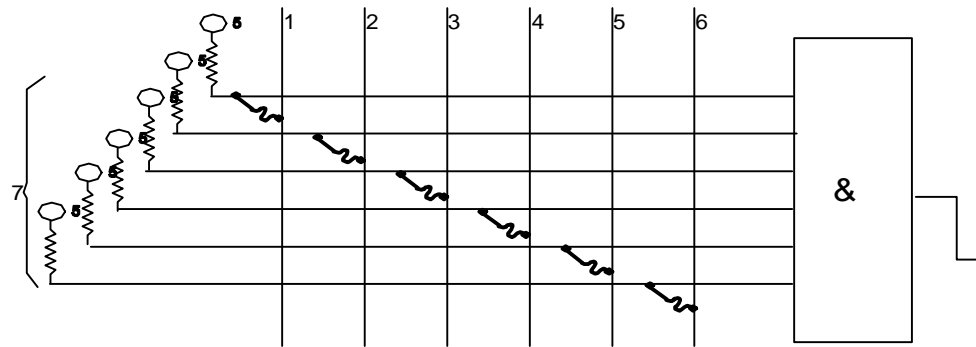
3.3.1 PAL (Programmable Array Logic)

Los dispositivos PAL son unos circuitos programables que poseen el plano AND programable, y el plano OR fijo.

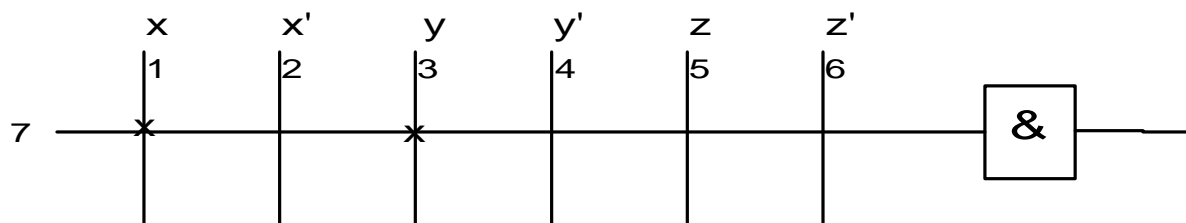
Estructura interna



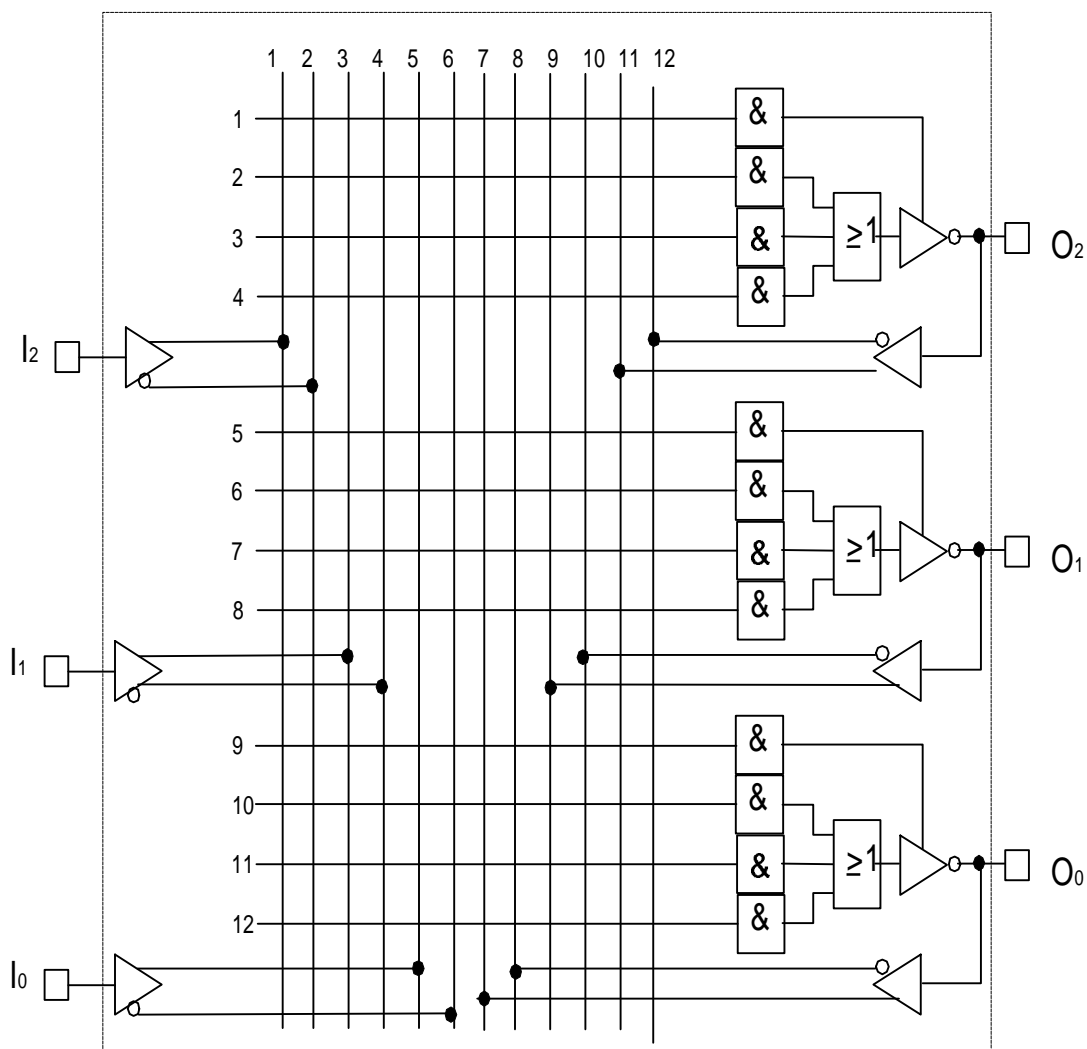
Detalle de la generación de un término producto



Representación simplificada del término producto



Estructura de una PAL más compleja



Ejemplo: Usando la primera estructura PAL, implementar las funciones de conmutación siguientes $f=\Sigma(0,2,4,7)$, $g=\Sigma(0,1,2,6,7)$, $h=\Sigma(6,7)$.

$$H(x,y,z) = xyz' + xyz$$

z \ xy	00		01	11	10
	0	1			
0	1	1			1
1				1	

f

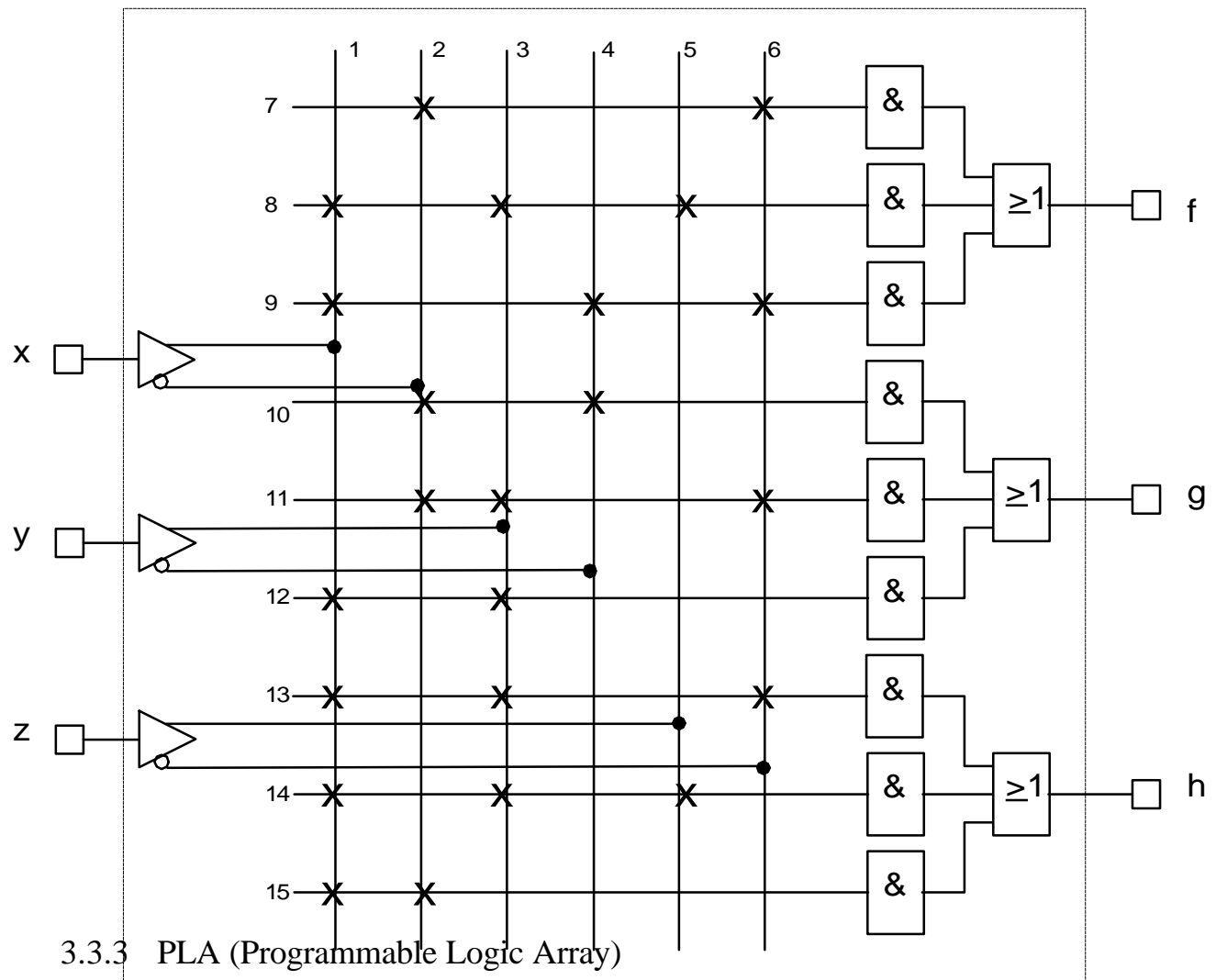
z \ xy	00		01	11	10
	0	1			
0	1	1	1		
1	1			1	

g

Las expresiones para f y g son:

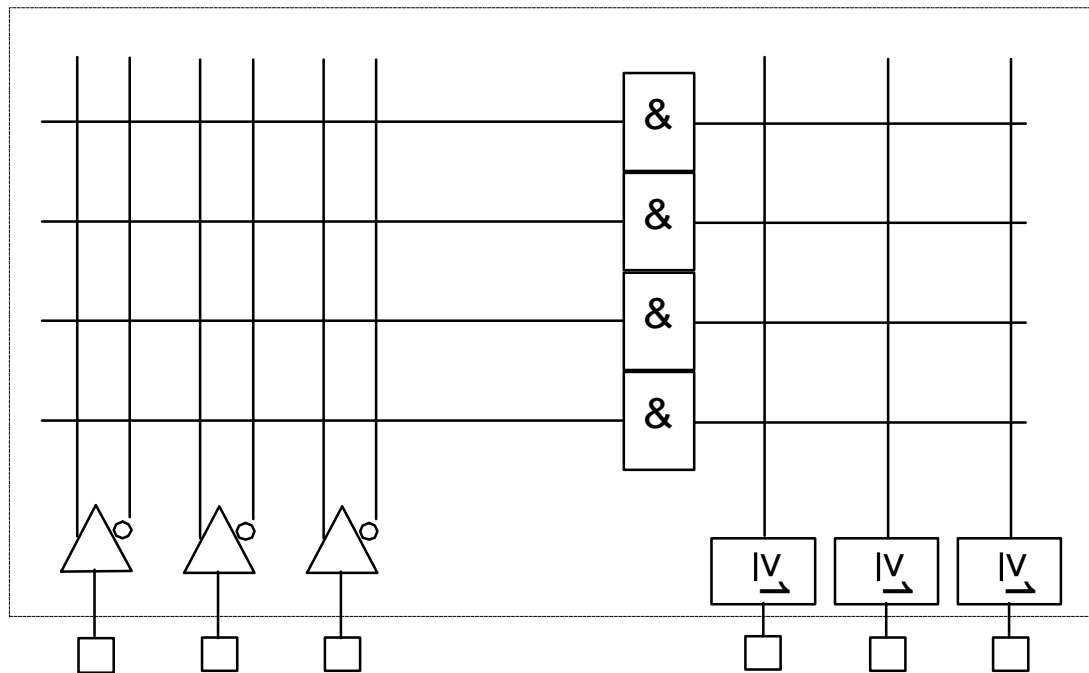
$$F(x,y,z) = x'z' + xyz + xy'z'$$

$$G(x,y,z) = x'y' + x'yz' + xy$$



3.3.3 PLA (Programmable Logic Array)

Este dispositivo tiene tanto el plano AND como el OR totalmente programables. La siguiente figura muestra la estructura de una PLA de 3 entradas y 3 salidas.



Ejemplo. Implementar las siguientes funciones en la PLA anterior

$$F = x' + y$$

$$G = x'y + xz'$$

$$H = xz' + x' + y$$

