Unidad 3.2 – CIRCUITOS LÓGICOS COMBINACIONALES

CIRCUITOS COMBINACIONALES ESTÁNDARES

Contenido

- Circuitos Aritméticos
  - Sumadores
  - Restadores
  - ALU
- De transferencia de datos
  - Multiplexores
  - Demultiplexores
  - Comparadores
- De codificación
  - Codificadores
  - Decodificadores
  - Conversores
- Ejemplos

Referencias (Aula Virtual)

> Floyd T. (2006). FUNDAMENTOS DE SISTEMAS DIGITALES. Capítulo 3: Puertas lógicas; capítulo 5: Análisis de lógica combinacional.

> Ramos J. (2012). SISTEMAS DIGITALES. Capítulo 3: Lógica combinacional.

> Mano M. (2003). DISEÑO DIGITAL. Capítulo 4: Lógica combinacional.

> Brown S. (2006). FUNDAMENTOS DE LÓGICA DIGITAL CON DISEÑO VHDL. Capítulo 2: Introducción a los circuitos lógicos.

> Tocci R. (2007). SISTEMAS DIGITALES PRINCIPIOS Y APLICACIONES. Capítulo 4: Circuitos lógicos combinacionales.

CIRCUITOS LÓGICOS COMBINACIONALES

Definición

Son aquellas configuraciones de *n* entradas y *m* salidas, tal que su salida o salidas dependen exclusivamente del estado *actual* de sus variables de entrada, independientemente del factor *tiempo*.

Todo circuito lógico combinacional representa a una determinada función lógica que no contiene componentes de realimentación.

El flujo de señales es unidireccional, de la entrada hacia la salida.

COMBINACIONALES ESTÁNDARES

Concepto

Son estructuras combinacionales de uso frecuente y muy difundido, por lo que existen disponibles comer-cialmente como circuitos integrados listos para ser utilizados.

- **Aritméticos**
  - **Sumadores**
    - Semisumador
    - Sumador total
    - Sumador BCD
  - Restadores
  - Sumador / restador por complemento
  - Unidad Aritmético - lógica
  - Otras operaciones
- **De transferencia**
  - Multiplexores
  - Demultiplexores
  - Comparadores
- **De codificación**
  - **Codificadores**
    - Sin prioridad
    - Con prioridad a mayor
    - Con prioridad a menor
  - **Decodificadores**
    - No excitadores
    - **Excitadores**
      - 7 segmentos
      - 14 segmentos
      - Matríz de puntos
  - Conversores de código

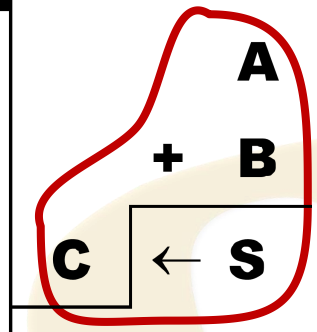
SUMADORES

COMBINACIONALES ESTÁNDARES - ARITMÉTICOS

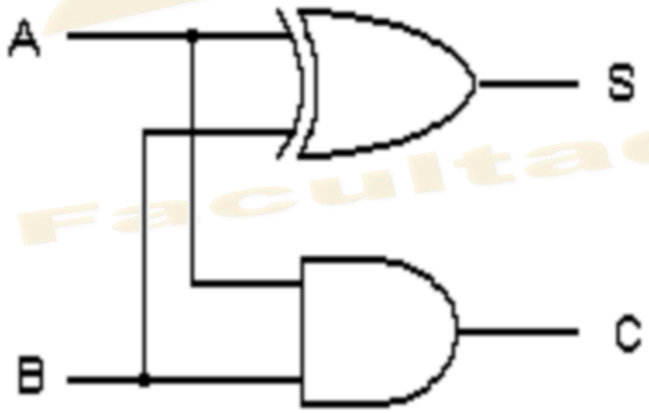
Sumadores

Floyd, pg. 328
Mano, pg. 119
Dormido, pg. 182

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

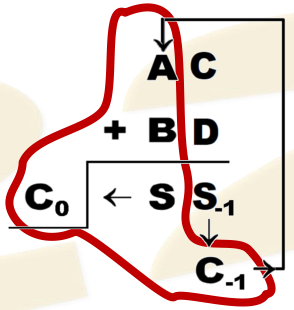


Semisumador
suma la primera columna

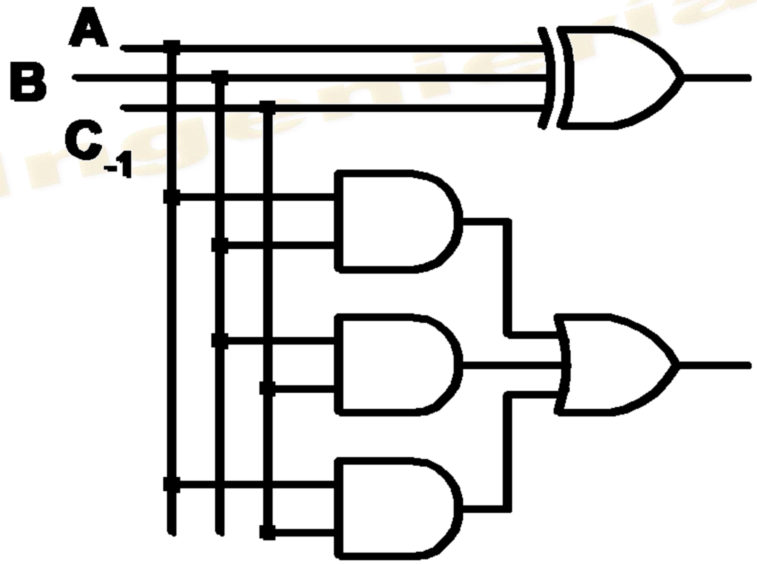


$$S = A \oplus B$$
$$C = A \cdot B$$

A	B	C_{-1}	S	C_0
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



Sumador total
suma la 2º columna en adelante

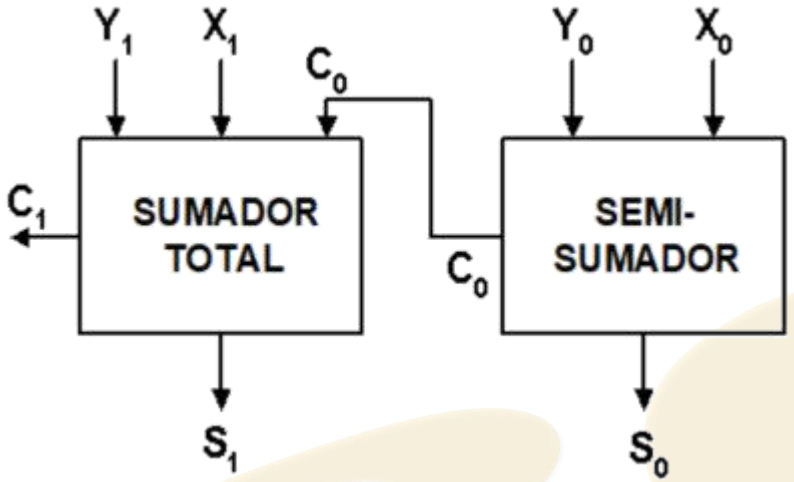


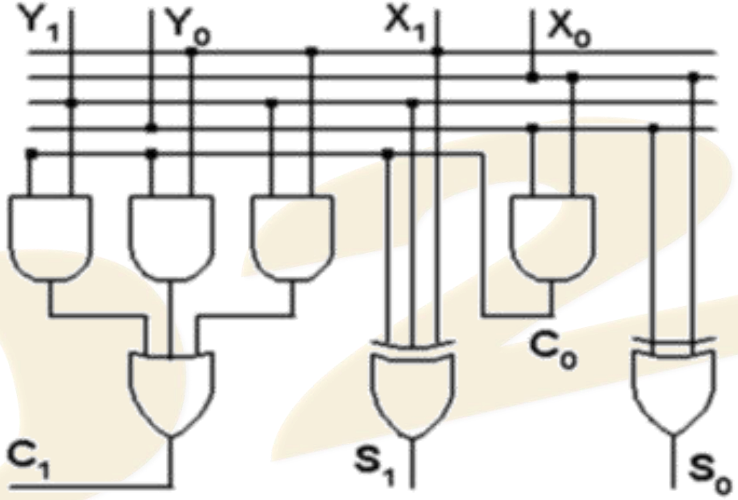
$$S = A \oplus B \oplus C_{-1}$$
$$C_0 = A.B + B.C_{-1} + A.C_{-1}$$

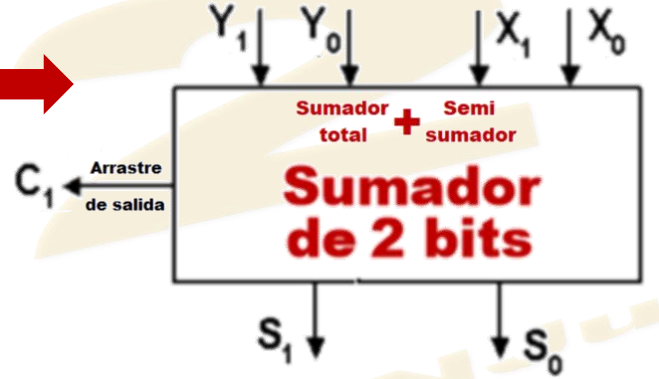
COMBINACIONALES ESTÁNDARES - ARITMÉTICOS

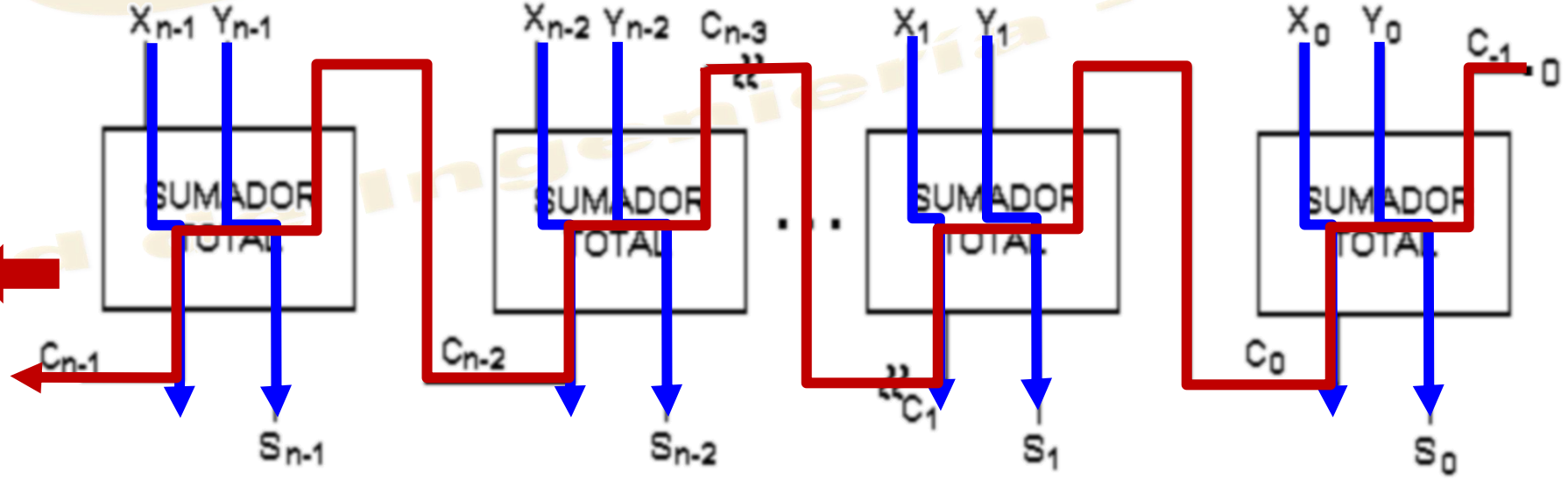
Sumador paralelo con arrastre propagado

Dormido, pg. 188





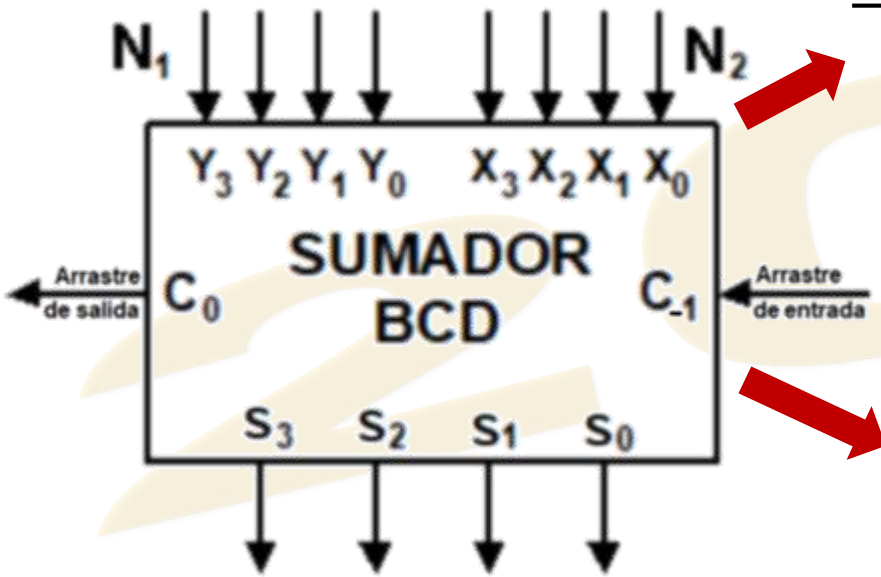




Suma Arrastre

COMBINACIONALES ESTÁNDARES - ARITMÉTICOS

Sumador BCD



Suma de 2 dígitos

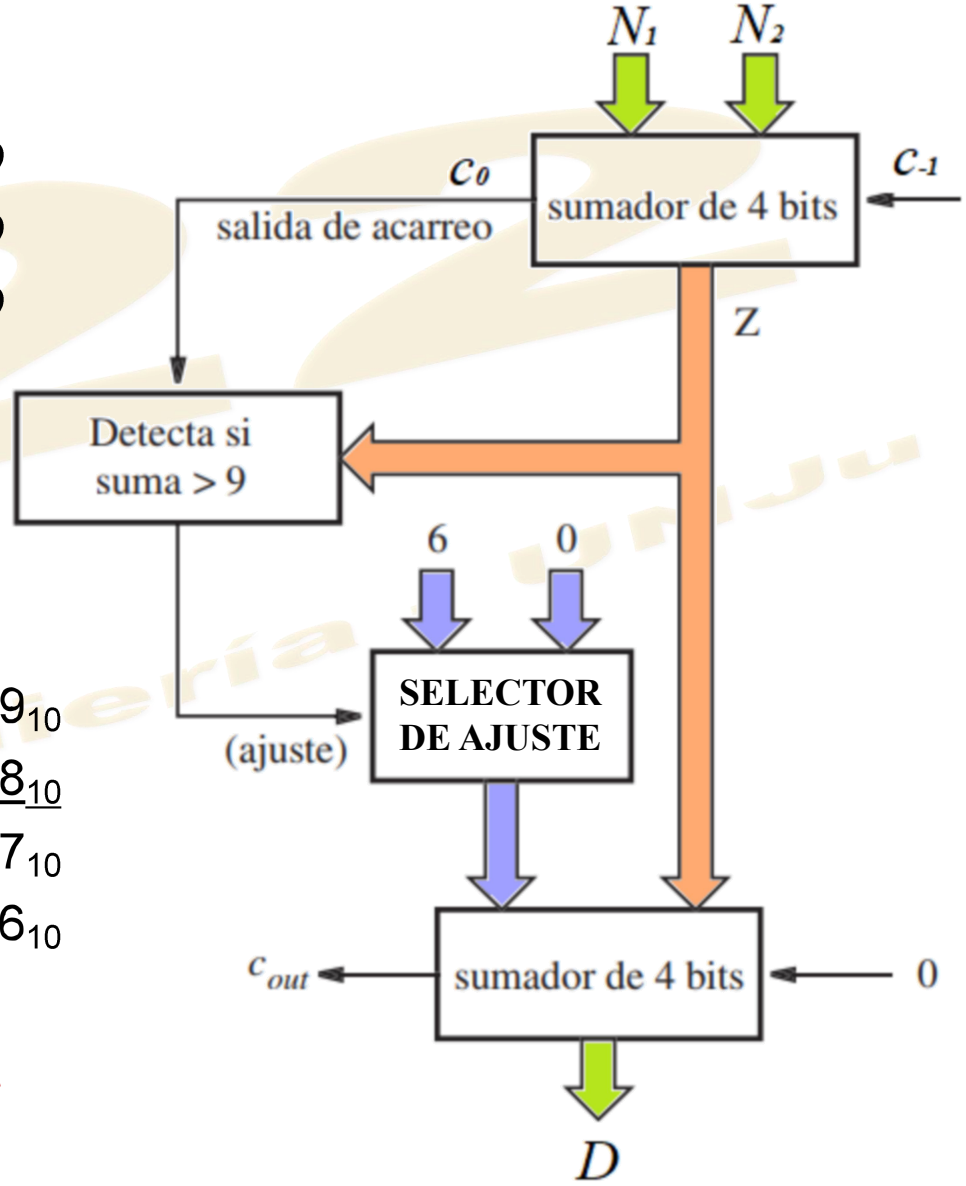
3_{10}	0011_2	◅ *en BCD*
$+\ 4_{10}$	$+\ 0100_2$	◅ *en BCD*
7_{10}	0111_2	◅ *en BCD*

Resultado en un dígito (NO necesita ajuste)

Suma de dos dígitos

en BCD ➢	1001_2	9_{10}
en BCD ➢	$+\ 1000_2$	$+\ 8_{10}$
no en BCD ➢	$1\ 0001_2$	17_{10}
ajuste BCD ➢	$+\ 0110_2$	6_{10}
en BCD ➢	$1\ 0111_2$	

Resultado en dos dígitos (SÍ necesita ajuste)



Mano, pg. 129
Dormido, pg. 182

COMBINACIONALES ESTÁNDARES - ARITMÉTICOS

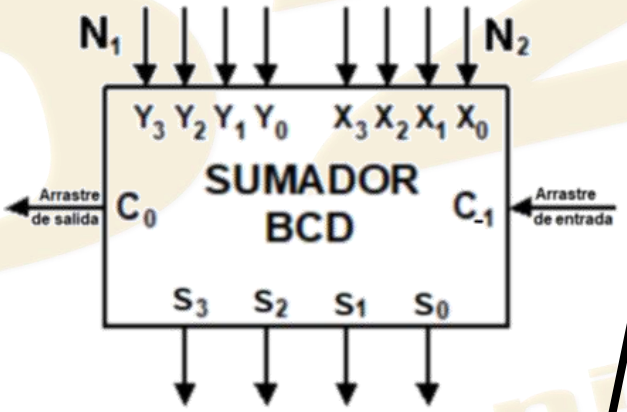
Sumador BCD - AJUSTE 1

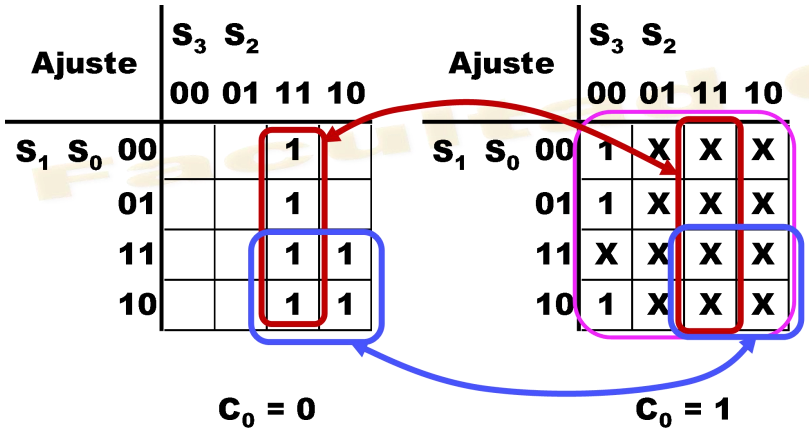
Mano, pg. 129

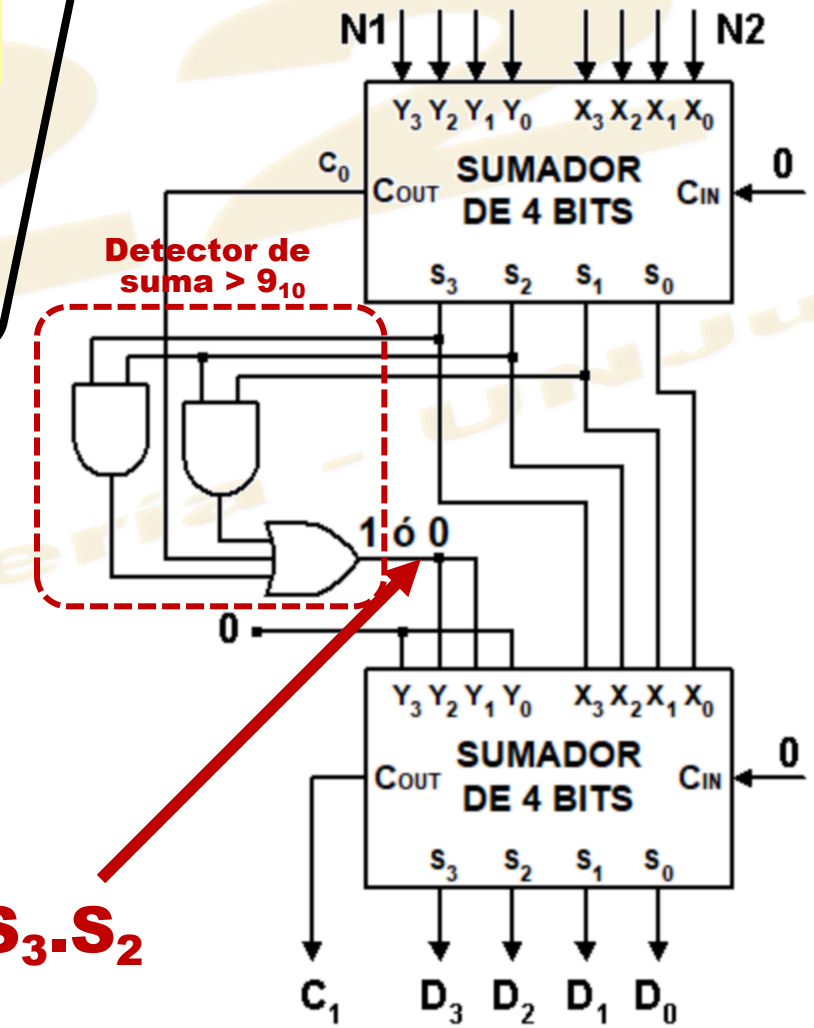
Suma	C_0	S_3	S_2	S_1	S_0	Ajuste
0	0	0	0	0	0	0
1	0	0	0	0	1	0
2	0	0	0	1	0	0
3	0	0	0	1	1	0
4	0	0	1	0	0	0
5	0	0	1	0	1	0
6	0	0	1	1	0	0
7	0	0	1	1	1	0
8	0	1	0	0	0	0
9	0	1	0	0	1	0
10	0	1	0	1	0	1
11	0	1	0	1	1	1
12	0	1	1	0	0	1
13	0	1	1	0	1	1
14	0	1	1	1	0	1
15	0	1	1	1	1	1

Suma	C_0	S_3	S_2	S_1	S_0	Ajuste
16	1	0	0	0	0	1
17	1	0	0	0	1	1
18	1	0	0	1	0	1
	1	0	0	1	1	X
	1	0	1	0	0	X
	1	0	1	0	1	X
	1	0	1	1	0	X
	1	0	1	1	1	X
	1	1	0	0	0	X
	1	1	0	0	1	X
	1	1	0	1	0	X
	1	1	0	1	1	X
	1	1	1	0	0	X
	1	1	1	0	1	X
	1	1	1	1	0	X
	1	1	1	1	1	X

Diseño 1
Por tabla de verdad







$$Ajuste = C_0 + S_3.S_1 + S_3.S_2$$

COMBINACIONALES ESTÁNDARES - ARITMÉTICOS

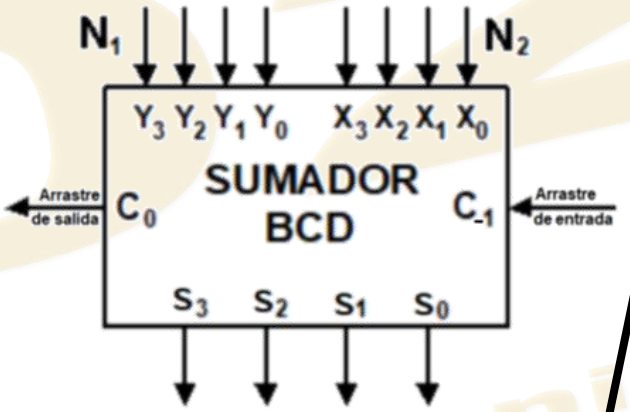
Sumador BCD - AJUSTE 2

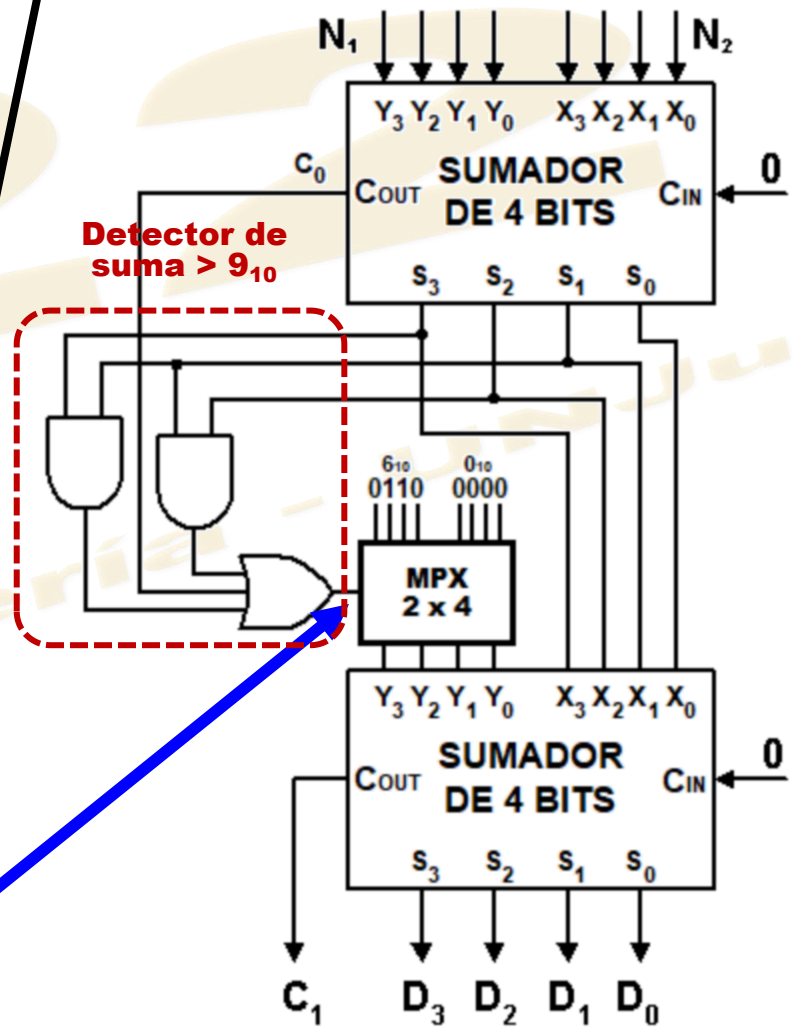
Mano, pg. 129

Suma	C_0	S_3	S_2	S_1	S_0	Ajuste
0	0	0	0	0	0	0
1	0	0	0	0	1	0
2	0	0	0	1	0	0
3	0	0	0	1	1	0
4	0	0	1	0	0	0
5	0	0	1	0	1	0
6	0	0	1	1	0	0
7	0	0	1	1	1	0
8	0	1	0	0	0	0
9	0	1	0	0	1	0
10	0	1	0	1	0	1
11	0	1	0	1	1	1
12	0	1	1	0	0	1
13	0	1	1	0	1	1
14	0	1	1	1	0	1
15	0	1	1	1	1	1

Suma	C_0	S_3	S_2	S_1	S_0	Ajuste
16	1	0	0	0	0	1
17	1	0	0	0	1	1
18	1	0	0	1	0	1
	1	0	0	1	1	X
	1	0	1	0	0	X
	1	0	1	0	1	X
	1	0	1	1	0	X
	1	0	1	1	1	X
	1	1	0	0	0	X
	1	1	0	0	1	X
	1	1	0	1	0	X
	1	1	0	1	1	X
	1	1	1	0	0	X
	1	1	1	0	1	X
	1	1	1	1	0	X
	1	1	1	1	1	X

Diseño 2
Directo con MPX





Detector marca 1 cuando hay arrastre (línea 16, 17 y 18) → 1° término

Detector marca 1 para S_3 y S_1 = 1 (líneas 10 y 11) → 2° término

Detector marca 1 para S_3 y S_2 = 1 (líneas 12 a 15) → 3° término

Ajuste = $C_0+S_3.S_1+S_3.S_2$

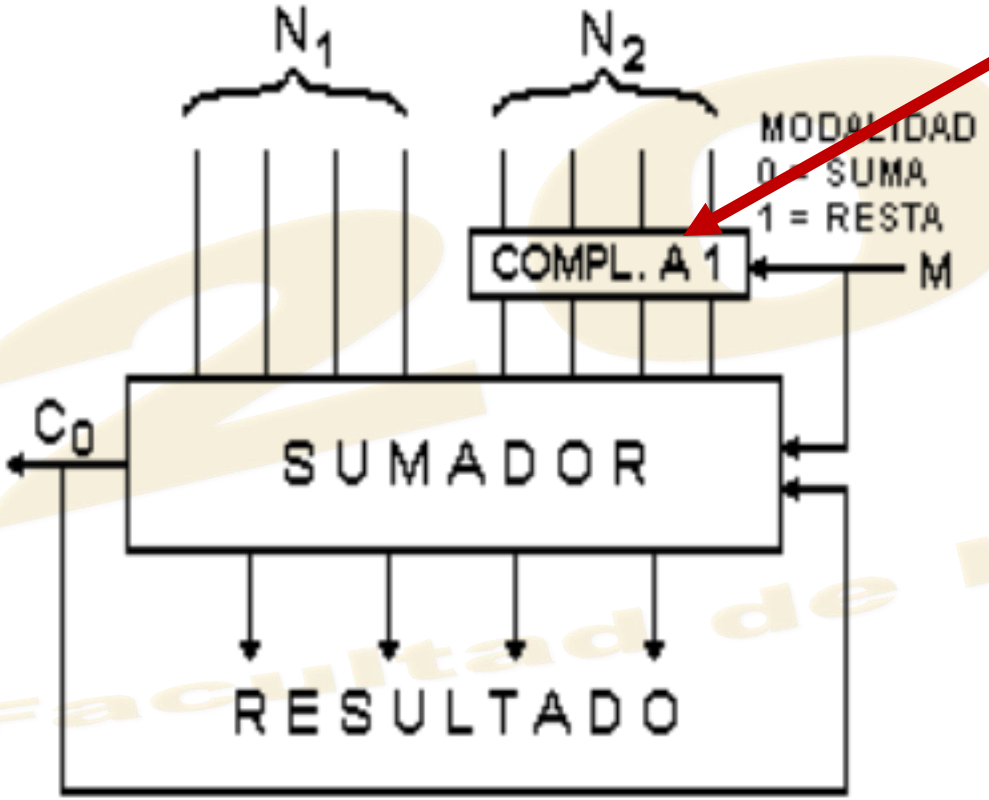
COMBINACIONALES ESTÁNDARES - ARITMÉTICOS

Sumador/restador por complemento

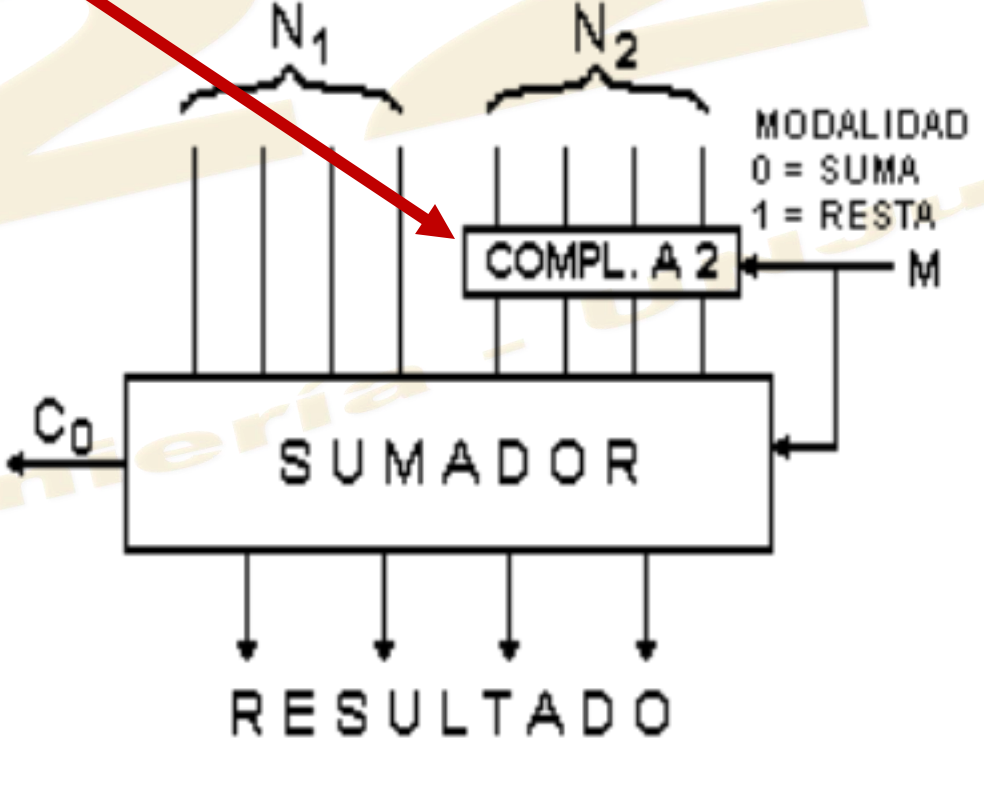
Ver Dormido, cap 4

$$\text{Suma} \rightarrow N_1 + N_2$$

$$\text{Resta} \rightarrow N_1 + (-N_2)$$



Sumador / restador por complemento a 1



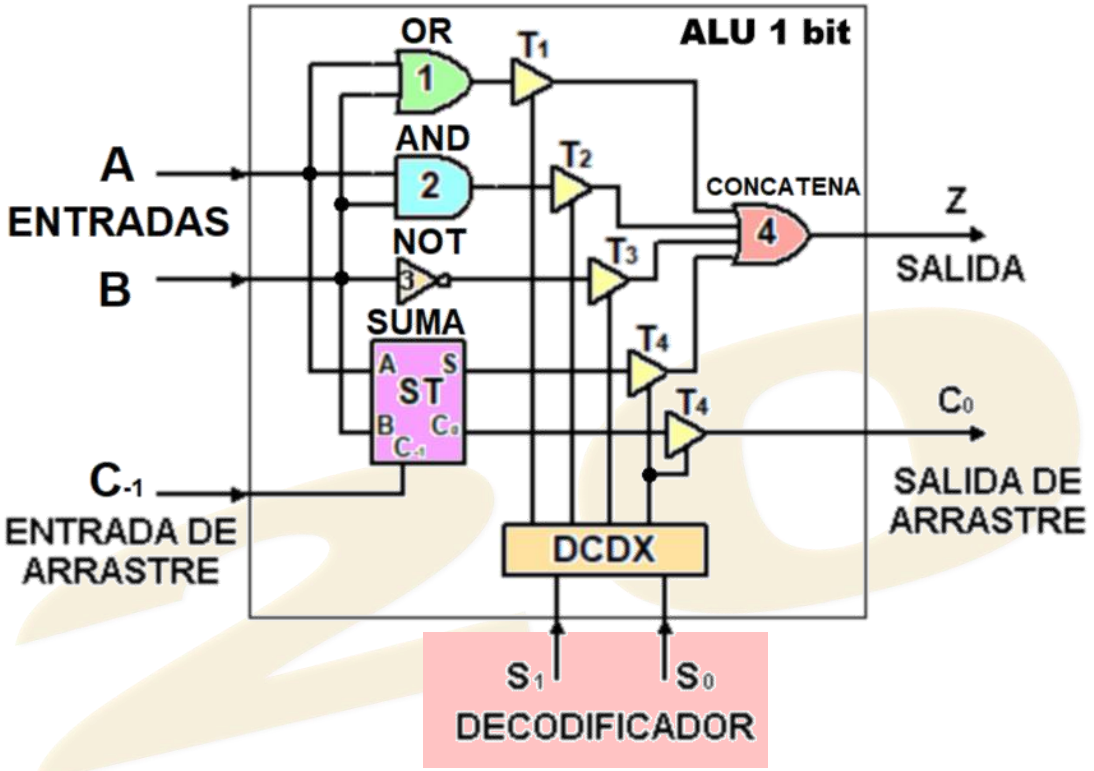
Sumador / restador por complemento a 2

COMBINACIONALES ESTÁNDARES - ARITMÉTICOS

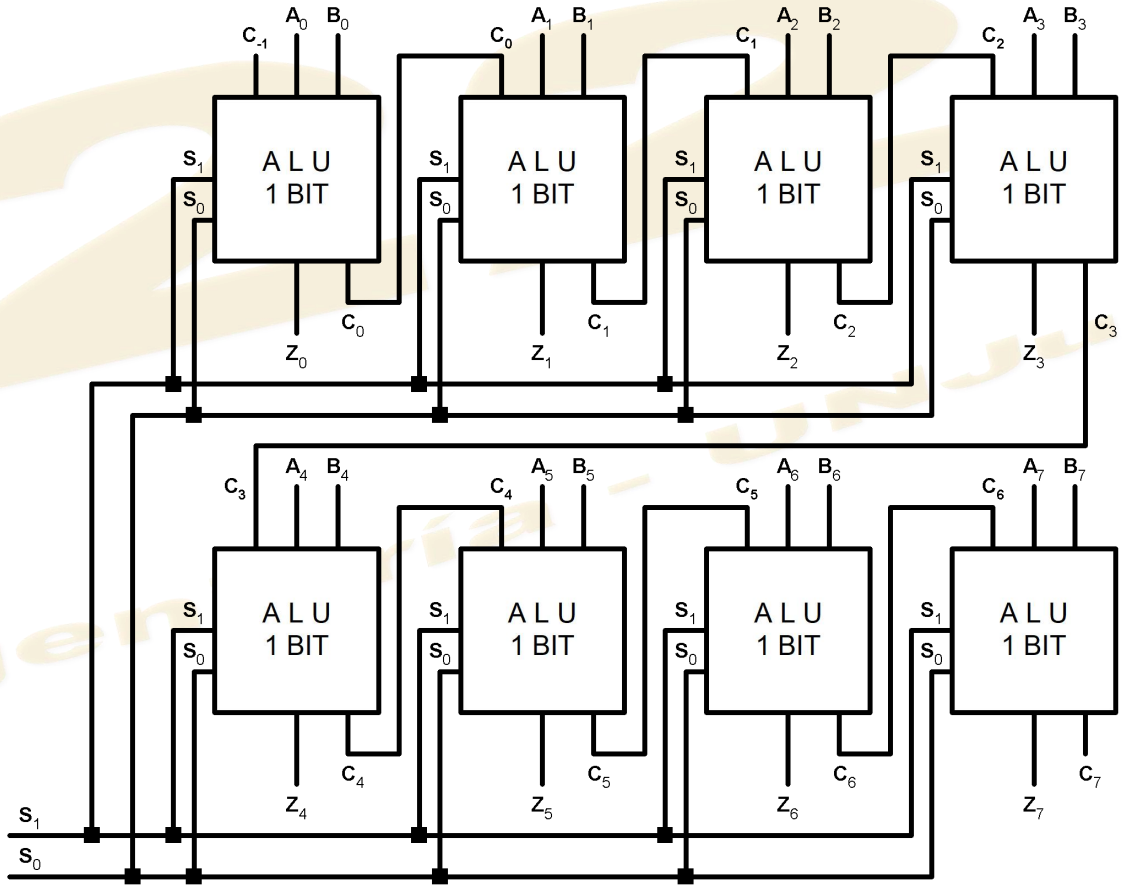
A.L.U.
UNIDAD
ARITMÉTICO - LÓGICA

COMBINACIONALES ESTÁNDARES - ARITMÉTICOS

Unidad aritmético-lógica



código		tri-state que se activa					
S1	S0	T1	T2	T3	T4	T4	Operación
0	0	1	0	0	0	0	A or B
0	1	0	1	0	0	0	A and B
1	0	0	0	1	0	0	B negado
1	1	0	0	0	1	1	A + B con arrastre C



ALU 8 bits

Dormido, pg. 234

COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

MPX
MULTIPLEXORES

COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

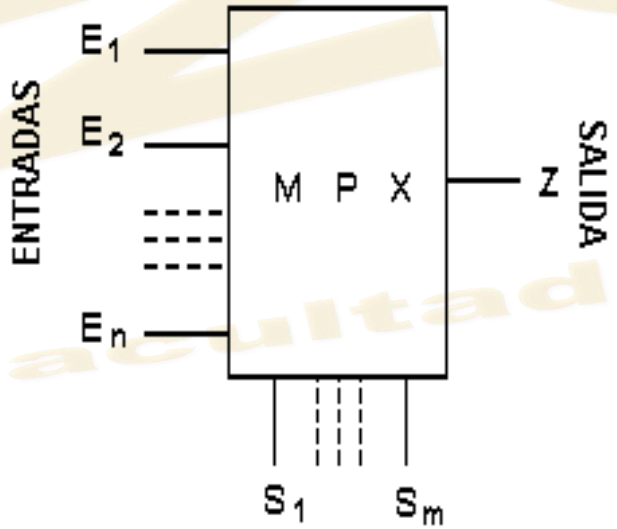
Multiplexores (MPX)

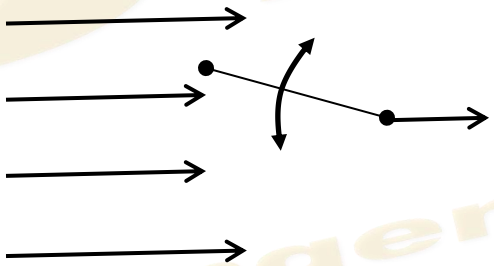
Floyd, pg. 367 - Mano, pg. 141 - Dormido, pg. 451

Un multiplexor puede definirse como un circuito lógico combinacional de varias entradas y una salida de tal forma que es posible "transferir" (reconstruir) una de las entradas a la salida mediante una codificación preestablecida.

MPX n × 1

n entradas/canales
× 1 bit/canal

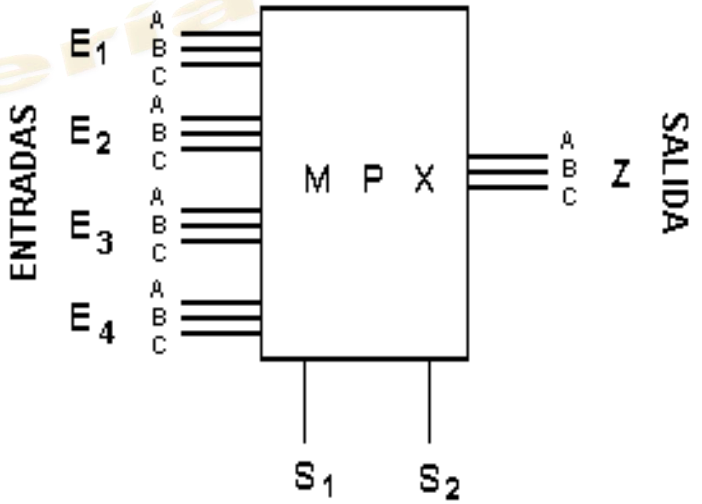




analogía eléctrica

MPX 4 × 3

4 entradas/canales
× 3 bits/canal

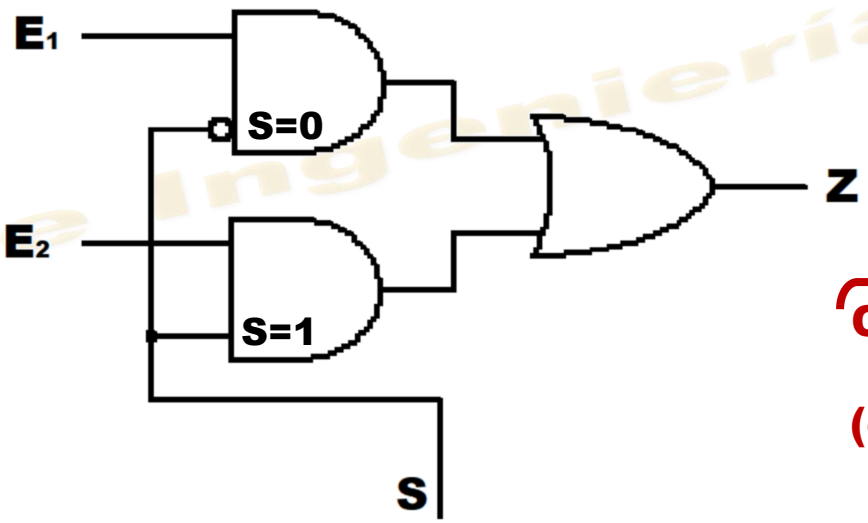


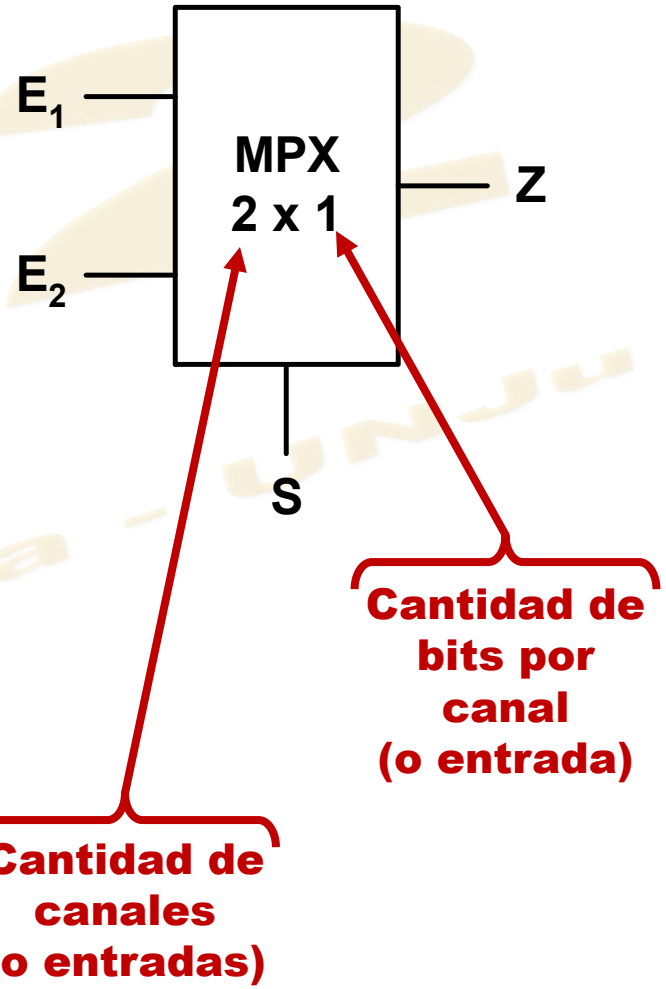
COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

Ejemplo: Multiplexor de 2x1

$$Z = E_1 . \overline{S} + E_2 . S$$

S	E_1	E_2	Z
0	0	X	0
0	1	X	1
1	X	0	0
1	X	1	1

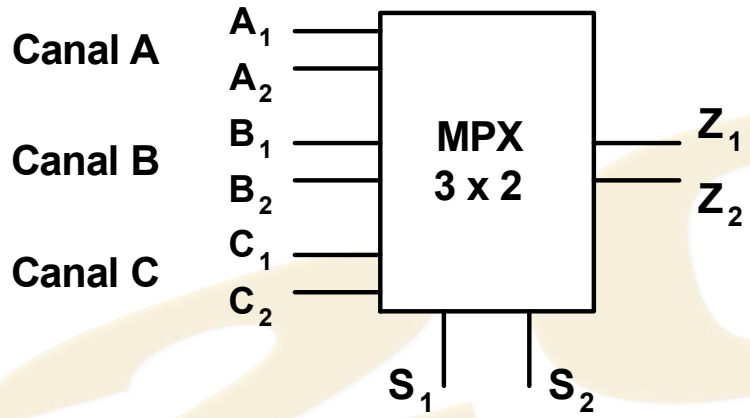


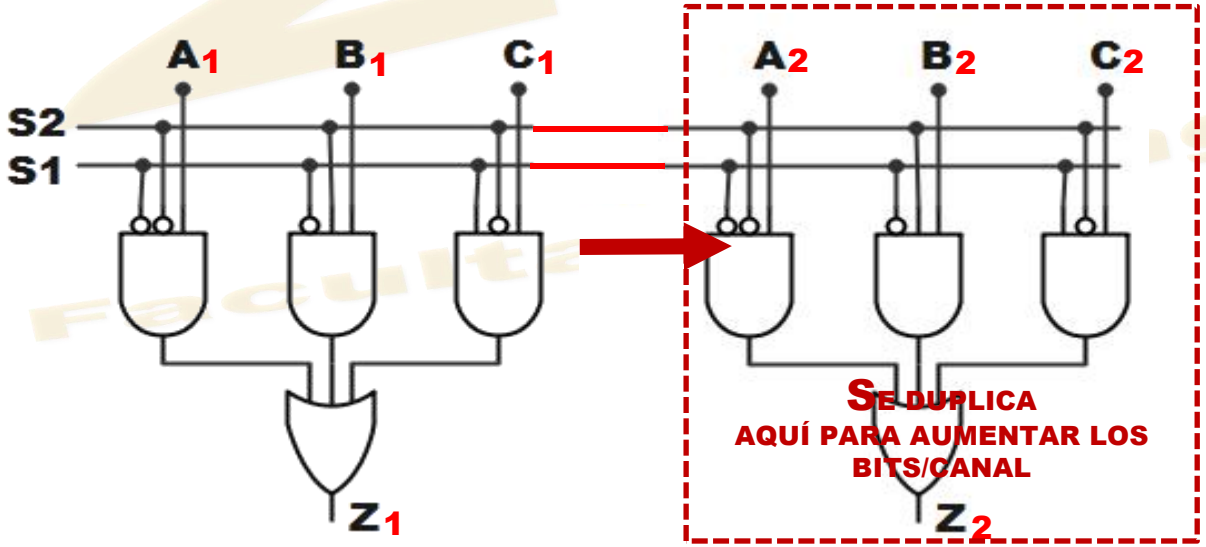


COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

Multiplexor (MPX)

Ejemplo: Multiplexor de 3x2





Conviene siempre iniciar con un diseño MPX 2^N x 1, luego reproducir la estructura hasta conseguir los bits/canal requeridos.

Selección		Datos				Salida
S1	S2	A	B	C	D	Z
0	0	0	X	X	X	0
0	0	1	X	X	X	1
0	1	X	0	X	X	0
0	1	X	1	X	X	1
1	0	X	X	0	X	0
1	0	X	X	1	X	1
1	1	X	X	X	0	0
1	1	X	X	X	1	1

MPX 4X1

$$Z = \overline{S1}.\overline{S2}.A + \overline{S1}.S2.B + S1.\overline{S2}.C + S1.S2.D$$

MPX 3X1

$$Z = \overline{S1}.\overline{S2}.A + \overline{S1}.S2.B + S1.\overline{S2}.C$$

Multiplexores (MPX) - Asociación

Combinar o asociar multiplexores permite crear estructuras más complejas, utilizando bloques menores, con las que se puede multiplexar datos para (relativamente) cualquier número de canales y de bits por canal.

Los criterios a seguir son simples:

> Al aumentar la cantidad de canales, generalmente debe aumentar la cantidad de líneas de selección, tal que

$$\text{Nº canales} \leq 2^{(\text{Nº líneas selección})}$$

significa: por lo menos una palabra de código de selección para cada canal.

> Al aumentar la cantidad de bits por canal, no aumentan las líneas de selección. La nueva estructura debe conectar sus líneas de selección 'en bus' con las ya existentes.

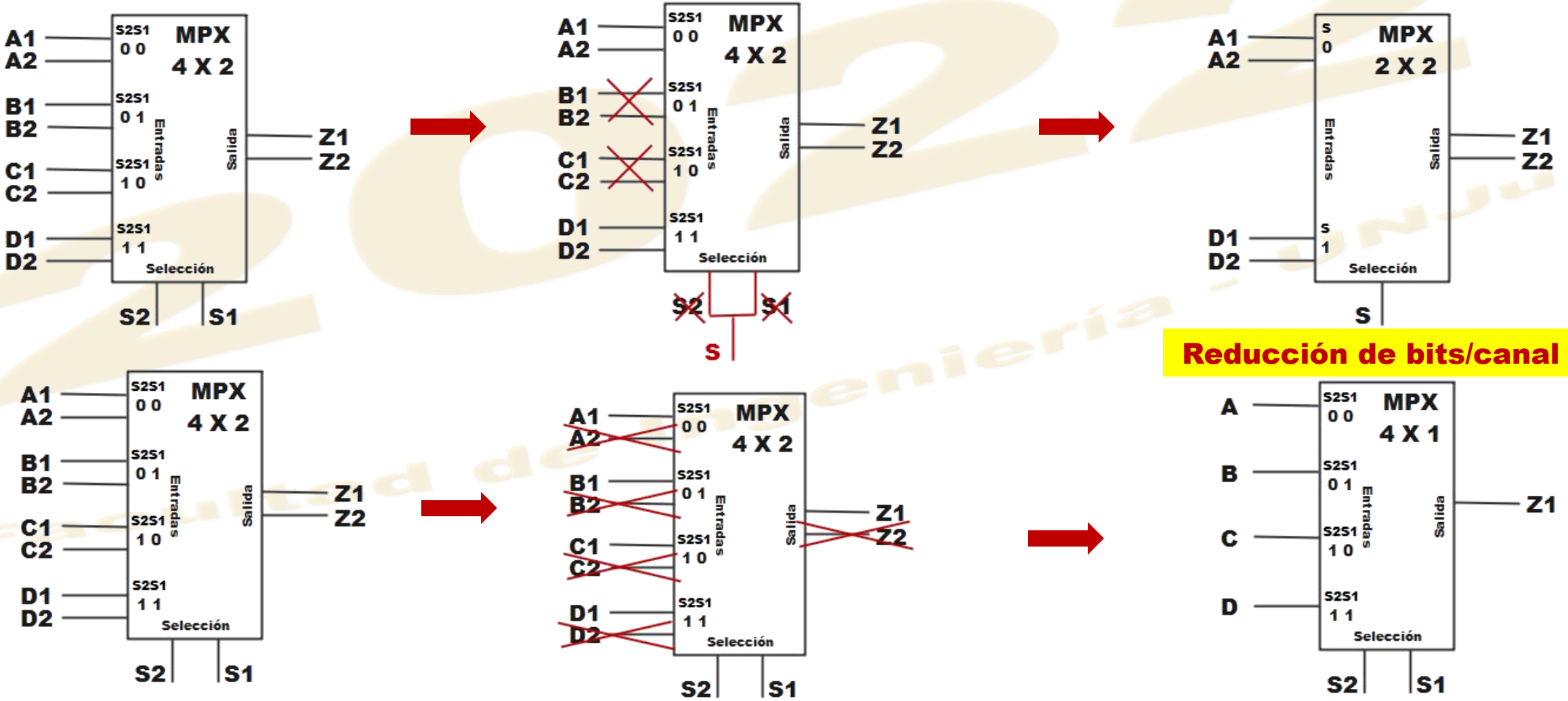
> Para el diseño de estructuras de NxM (N=canales; M=bits), es buena práctica iniciar con un bloque de Nx1, multiplicar este formato y conectarlos, hasta alcanzar los M bits por canal.

> Multiplexores de NxM se pueden reducir en cantidad de canales, omitiendo los que no se necesitan. Idem para los bits/canal.

COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

Multiplexor (MPX) - Asociación

Ejemplo: Reducción de multiplexores

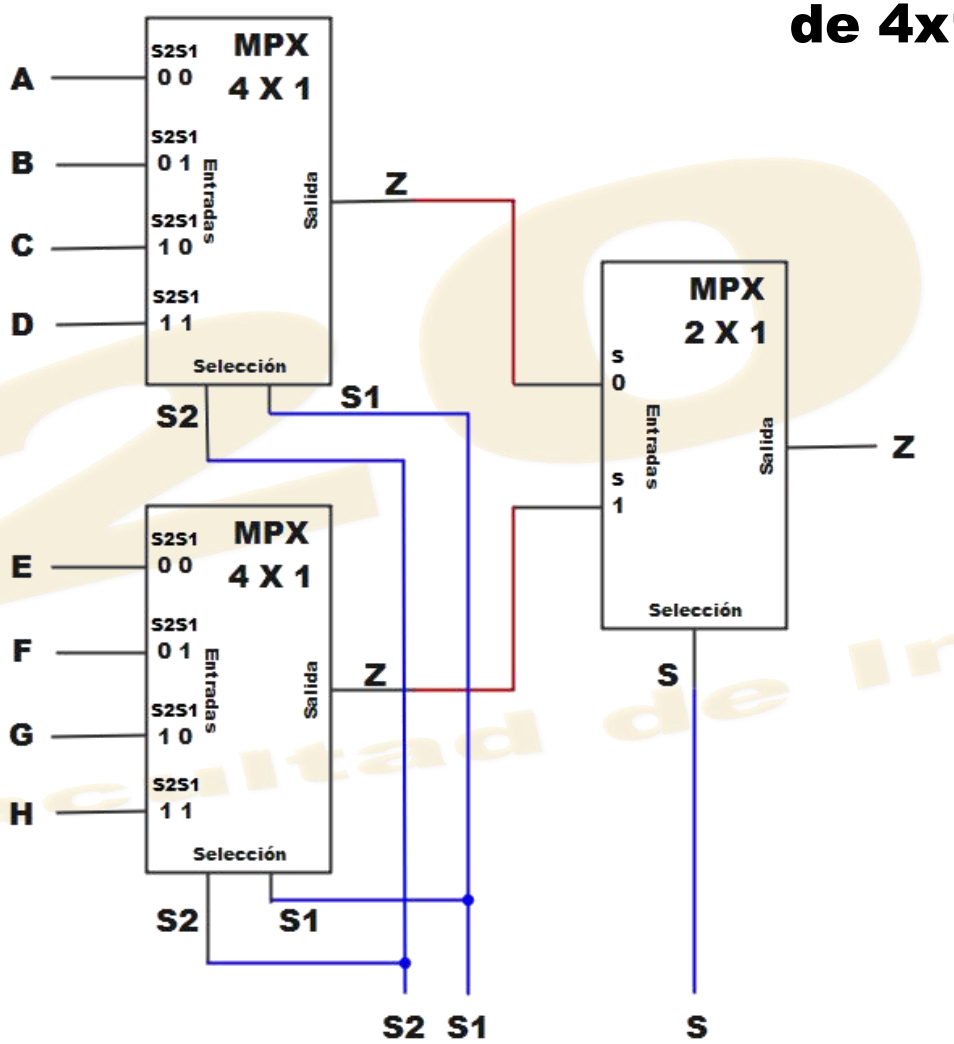


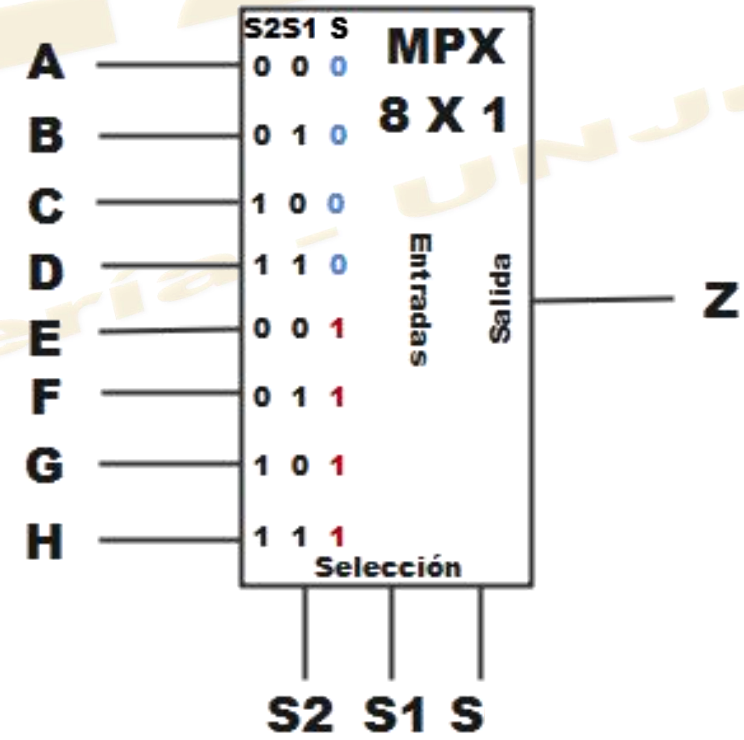
COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

Multiplexor (MPX) - Asociación

Ejemplo: Aumentar la cantidad de canales de 4x1 a 8x1

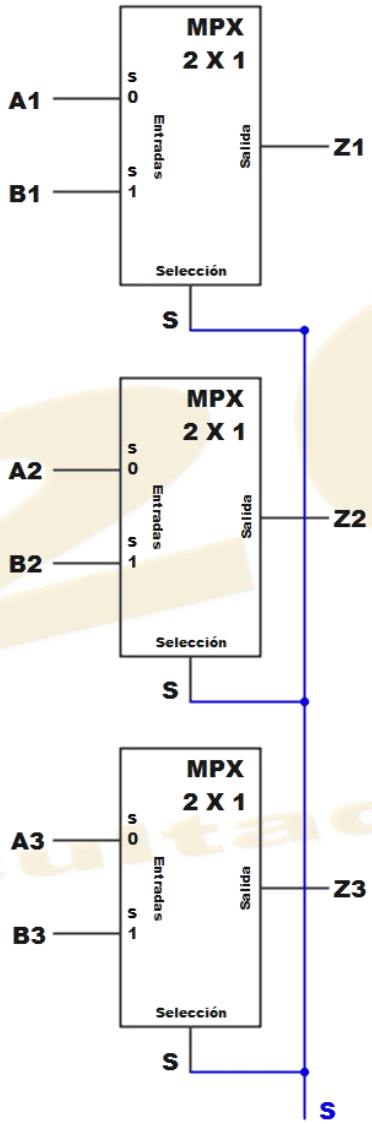
Para aumentar una línea de selección se agrega una nueva instancia de decisión (MPX 2x1) que seleccione las salidas del primer grupo.



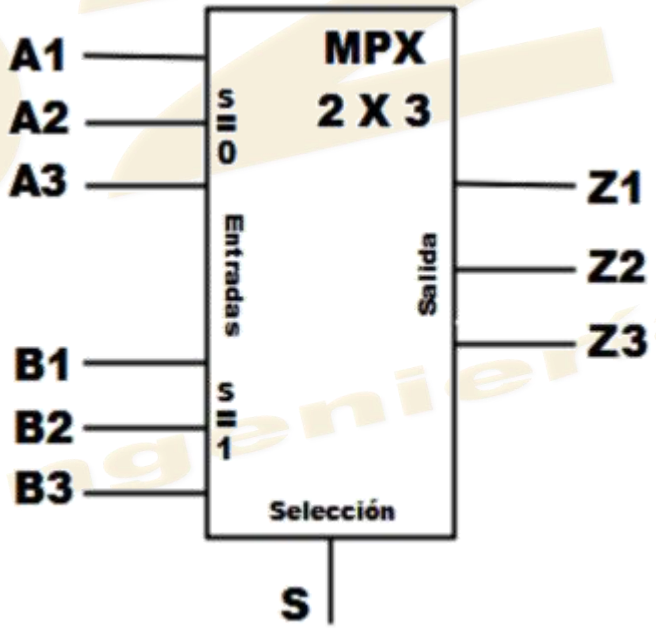


COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

Multiplexores (MPX) - Asociación



Ejemplo: Aumentar la cantidad de bits por canal, de 2x1 a 2x3.



Al aumentar la cantidad de bits por canal, no se modifica la cantidad de líneas de selección. La estrategia es conectar en bus, unidades con la misma cantidad de canales.

DMPX
DEMULTIPLEXORES

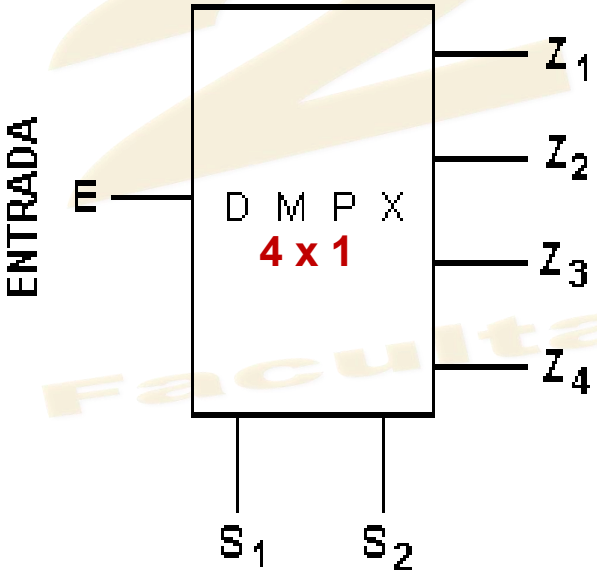
COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

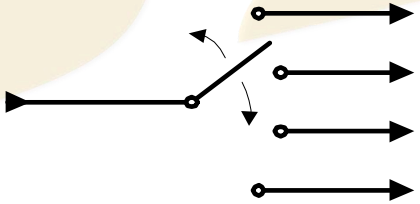
Demultiplexores (DMPX)

Floyd, pg. 377 - Dormido, pg. 454

Es un circuito combinacional de una entrada y varias salidas, que puede "transferir" (reconstruir) la entrada a una de las salidas, mediante una codificación predefinida.

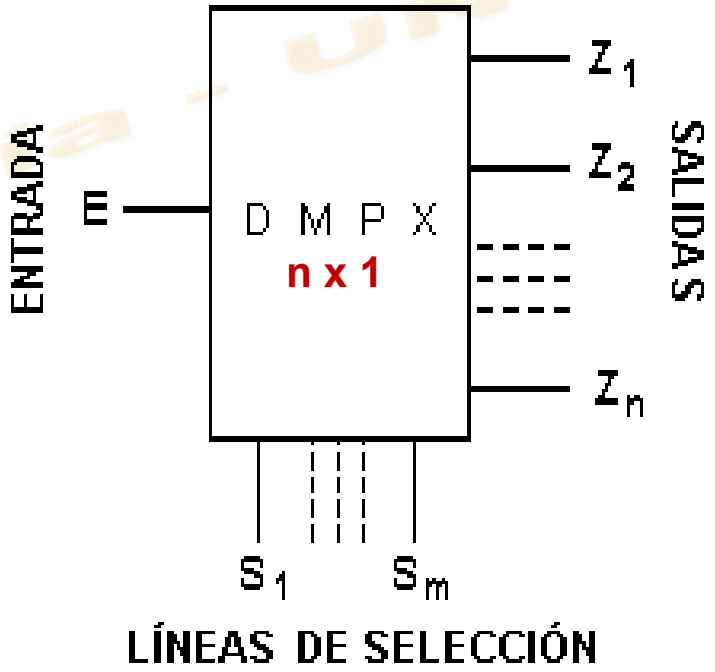
4 salidas/canales × 1 bit/canal





analogía eléctrica

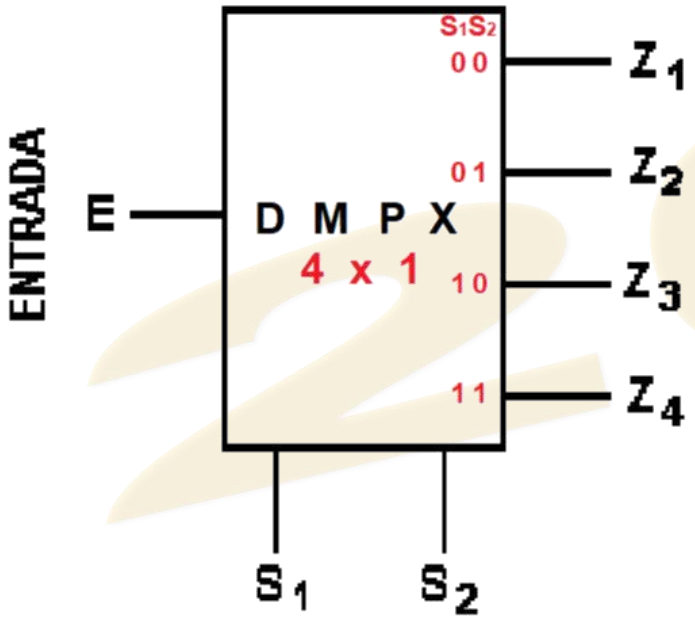
n salidas/canales × 1 bit/canal



COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

Demultiplexores (DMPX)

Ejemplo: Demultiplexor de 4x1 – Código binario natural



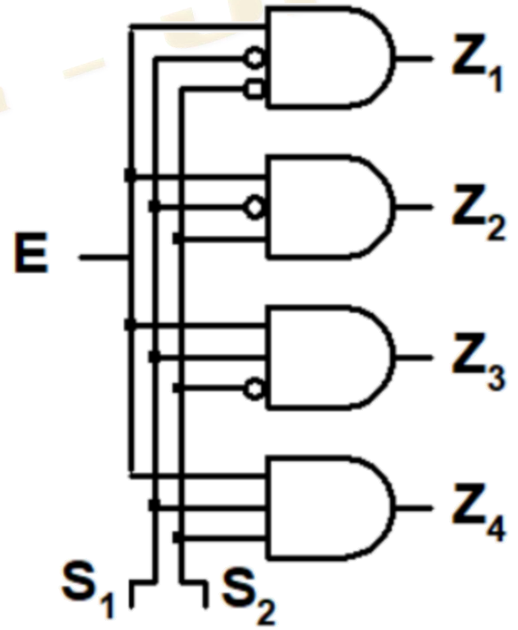
S_1	S_2	E	Z_1	Z_2	Z_3	Z_4
0	0	0	0	X	X	X
0	0	1	1	X	X	X
0	1	0	X	0	X	X
0	1	1	X	1	X	X
1	0	0	X	X	0	X
1	0	1	X	X	1	X
1	1	0	X	X	X	0
1	1	1	X	X	X	1

$$Z_1 = \overline{S_1} \cdot \overline{S_2} \cdot E$$

$$Z_2 = \overline{S_1} \cdot S_1 \cdot E$$

$$Z_3 = S_1 \cdot \overline{S_2} \cdot E$$

$$Z_4 = S_1 \cdot S_2 \cdot E$$



La síntesis directa es simple. Tantas AND como salidas. A cada AND ingresa la entrada E y las líneas de selección codificadas.

COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

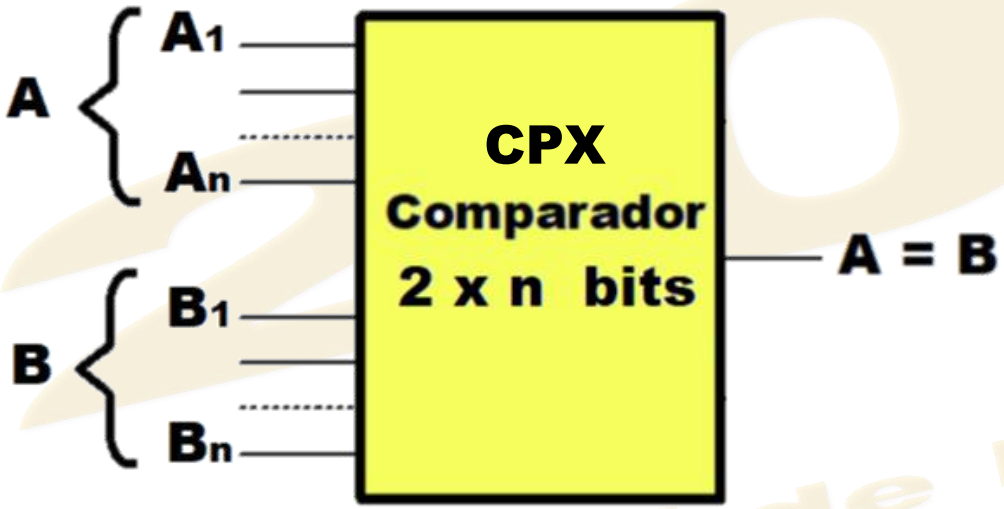
CMP
COMPARADORES

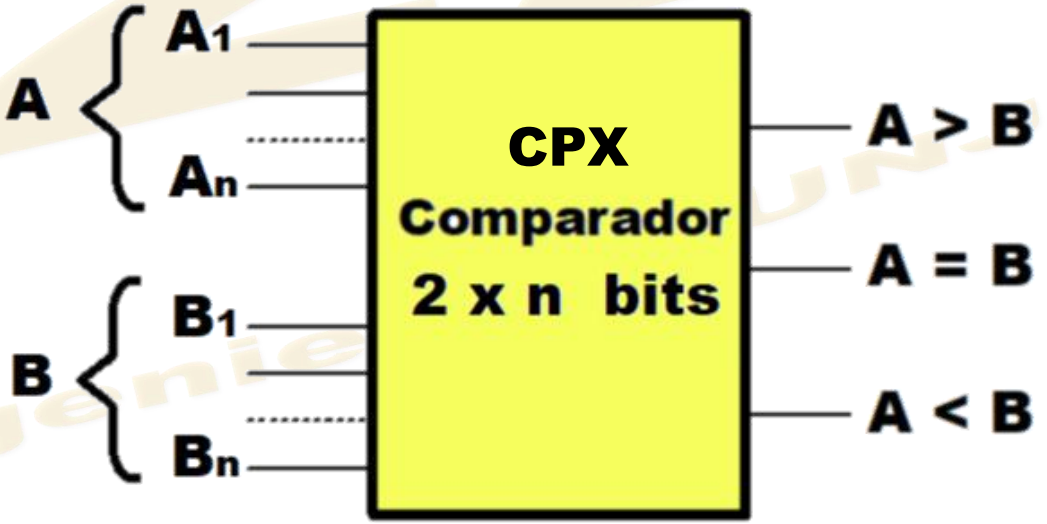
COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

Comparadores (CPX)

Floyd, pg. 344

Circuito combinacional de 2 x n entradas y una salida o más salidas, tal que produce resultado 1 en la o las salidas cuando las condiciones de comparación en las dos palabras binarias de n bits son verdaderas.



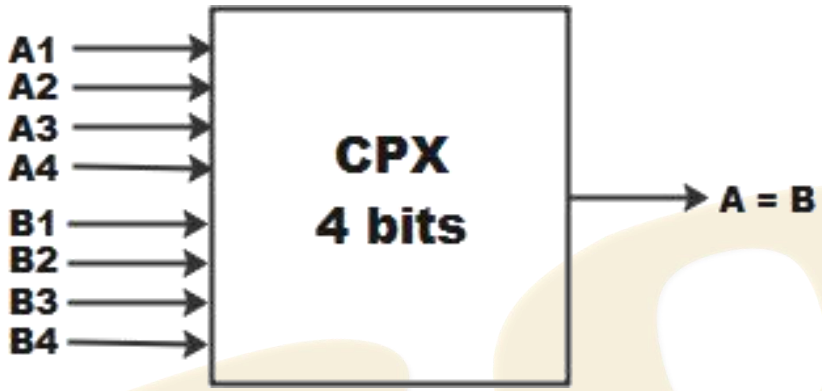


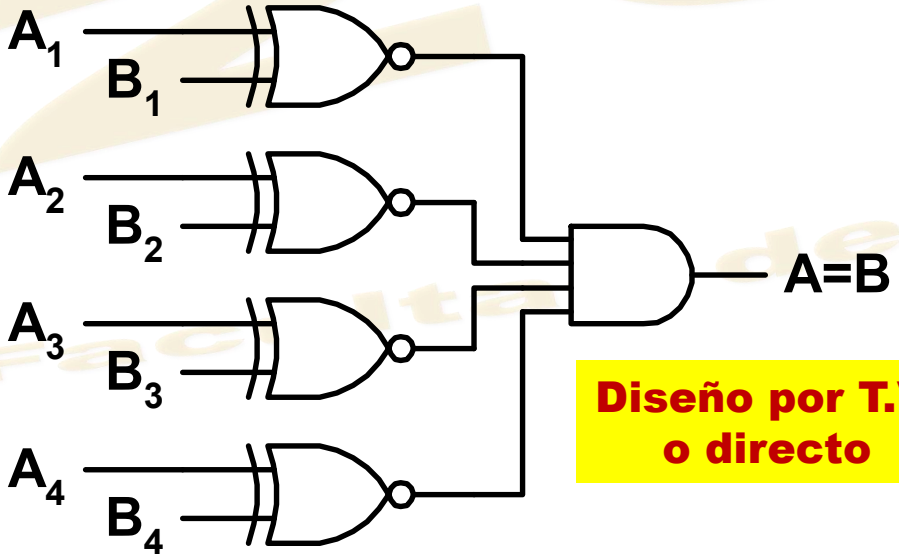
Bloques genéricos

La estructura se puede ampliar para comparar más de dos palabras.
La comparación puede ser a igualdad, a mayor o a menor.

COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

Comparadores (CPX)





Diseño por T.V. o directo

Ejemplo: Diseño de un comparador de 4 bits a igualdad.

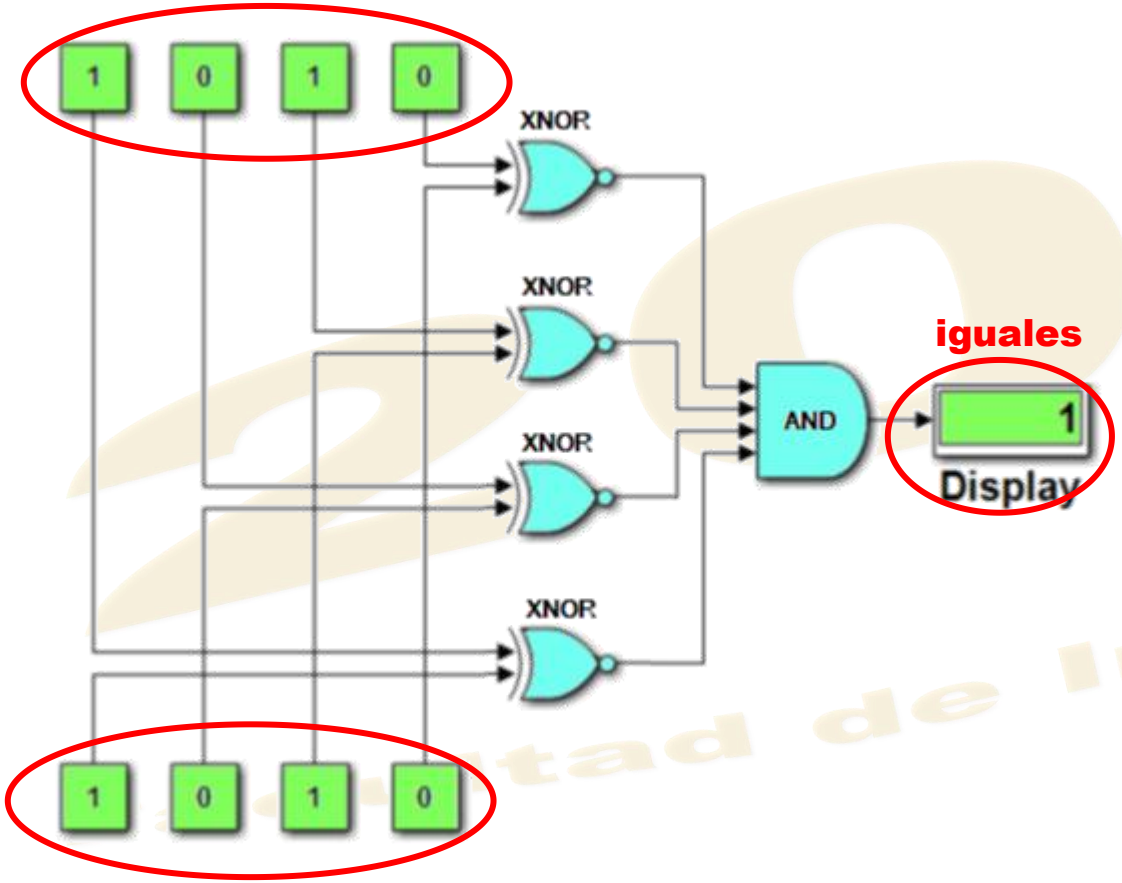
Dec	A1	A2	A3	A4	B1	B2	B3	B4	Z1
0_{10}	0	0	0	0	0	0	0	0	1
1_{10}	0	0	0	1	0	0	0	1	1
2_{10}	0	0	1	0	0	0	1	0	1
3_{10}	0	0	1	1	0	0	1	1	1
4_{10}	0	1	0	0	0	1	0	0	1
5_{10}	0	1	0	1	0	1	0	1	1
6_{10}	0	1	1	0	0	1	1	0	1
7_{10}	0	1	1	1	0	1	1	1	1
8_{10}	1	0	0	0	1	0	0	0	1
9_{10}	1	0	0	1	1	0	0	1	1
10_{10}	1	0	1	0	1	0	1	0	1
11_{10}	1	0	1	1	1	0	1	1	1
12_{10}	1	1	0	0	1	1	0	0	1
13_{10}	1	1	0	1	1	1	0	1	1
14_{10}	1	1	1	0	1	1	1	0	1
15_{10}	1	1	1	1	1	1	1	1	1

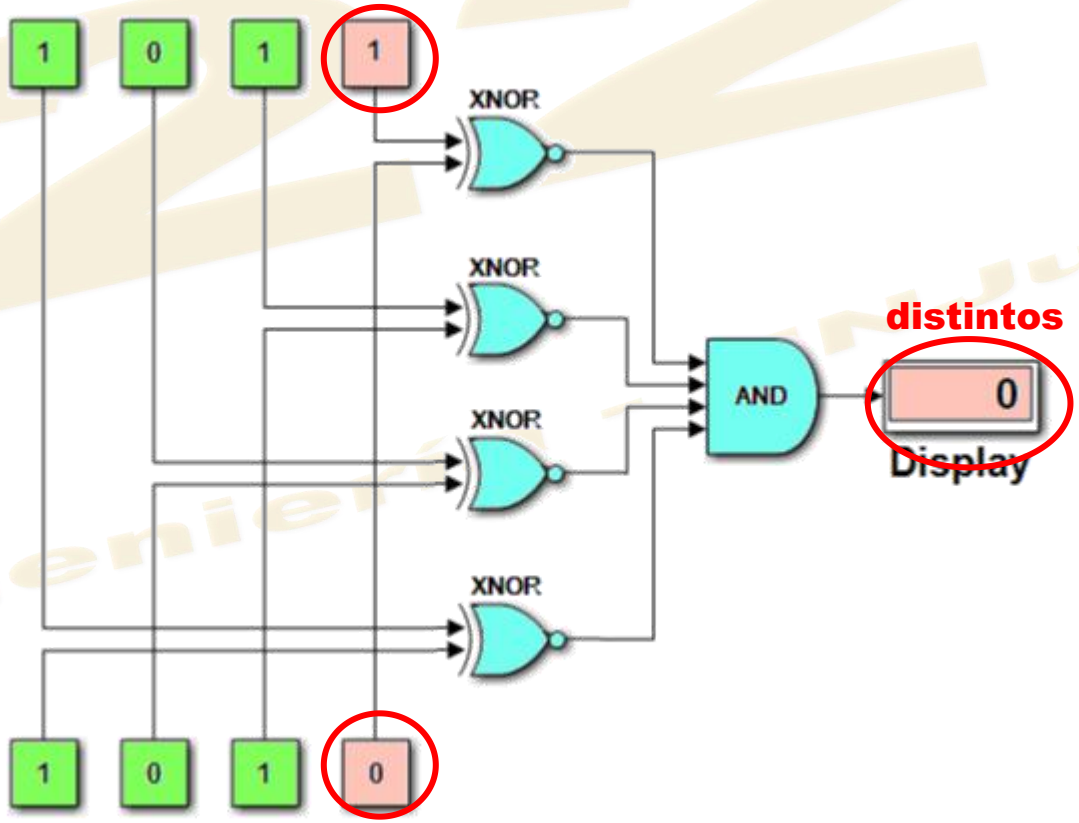
Las restantes combinaciones toman valor 0

COMBINACIONALES ESTÁNDARES - TRANSFERENCIA

Comparadores (CPX)

Ejemplo: Diseño de un comparador de 4 bits a igualdad.





Simulación sobre Simulink de Matlab

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

CDX
CODIFICADORES

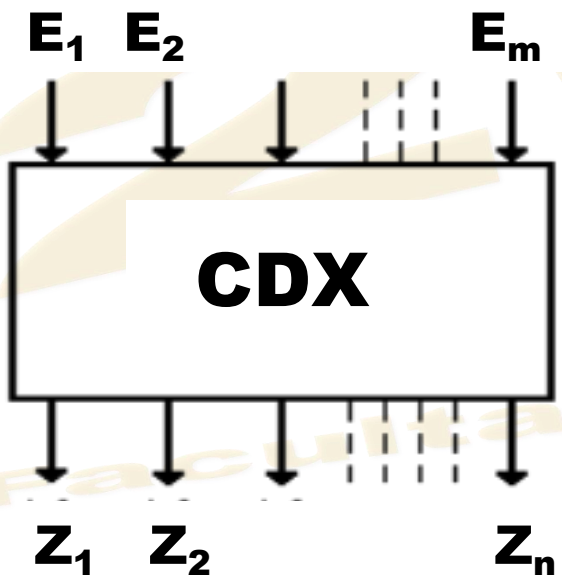
COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Codificadores (CDX)

Floyd, pg. 359 - Mano, pg. 139 - Dormido, pg. 441

Son circuitos combinacionales de m entradas y n salidas, que al recibir por las entradas una palabra binaria no codificado (ej. código posicional), produce a la salida la palabra equivalente codificada.

Bloque genérico



El proceso de diseño consiste básicamente en diseñar un circuito combinacional que asocie cada palabra de entrada con la correspondiente palabra de salida codificada

Código posicional decimal (1 de 10)

E9	E8	E7	E6	E5	E4	E3	E2	E1	E0	Dec
0	0	0	0	0	0	0	0	0	1	0_{10}
0	0	0	0	0	0	0	0	1	0	1_{10}
0	0	0	0	0	0	0	1	0	0	2_{10}
0	0	0	0	0	0	1	0	0	0	3_{10}
...	...	...	...	...	...	...	...	...	...	...
0	0	1	0	0	0	0	0	0	0	7_{10}
0	1	0	0	0	0	0	0	0	0	8_{10}
1	0	0	0	0	0	0	0	0	0	9_{10}

Código posicional octal (1 de 8)

E7	E6	E5	E4	E3	E2	E1	E0	Oct
0	0	0	0	0	0	0	1	0_8
0	0	0	0	0	0	1	0	1_8
0	0	0	0	0	1	0	0	2_8
0	0	0	0	1	0	0	0	3_8
0	0	0	1	0	0	0	0	4_8
0	0	1	0	0	0	0	0	5_8
0	1	0	0	0	0	0	0	6_8
1	0	0	0	0	0	0	0	7_8

Codificadores
- Sin prioridad
- Con prioridad a mayor
- Con prioridad a menor

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Codificadores (CDX)

Ejemplo: Codificador sin prioridad de cuaternario $(4,1)_{ss}$ a binario natural $(2,2)_{ss}$

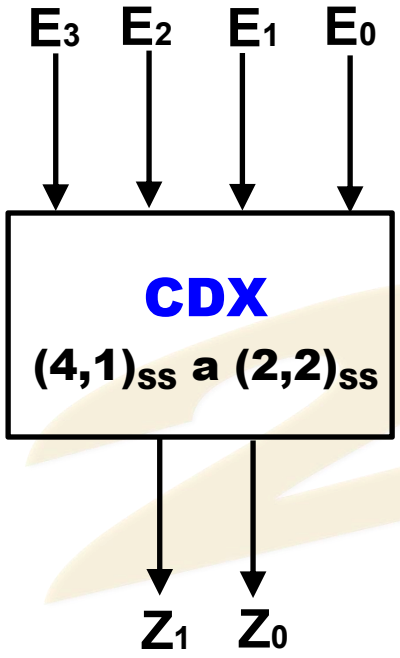


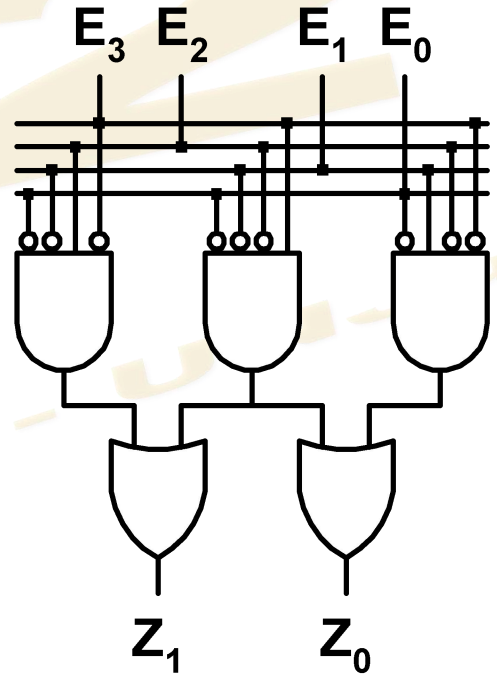
Diagrama de bloque

Diseño por T.V.

Posicional				Binario	
E_3	E_2	E_1	E_0	Z_1	Z_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1
Restantes combinaciones				X	X

Tabla de verdad reducida
SIN PRIORIDAD

Implementación canónica



Circuito lógico

Nótese que en este caso que la implementación de variables, como la asignación de estados en las entradas, está implícita.

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Codificadores (CDX)

Ejemplo: Codificadores con prioridad de cuaternario $(4,1)_{ss}$ a binario natural $(2,2)_{ss}$

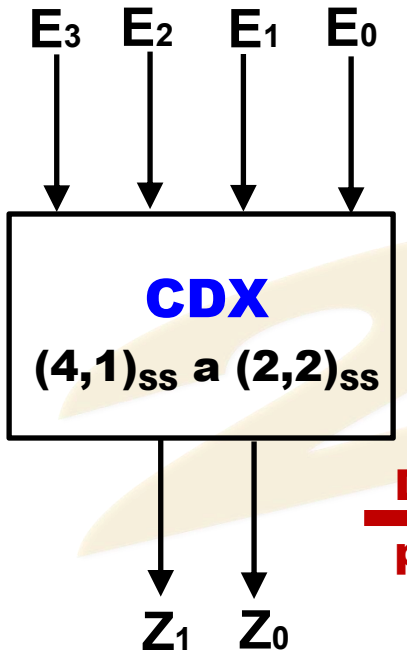


Diagrama de bloque

Diseño por T.V.

Prioridad a menor $(4,1)_{ss}$

E0	E1	E2	E3	Z1	Z0	Dec
1	X	X	X	0	0	0_4
0	1	X	X	0	1	1_4
0	0	1	X	1	0	2_4
0	0	0	1	1	1	3_4

X = representa 0 ó 1

Prioridad a mayor

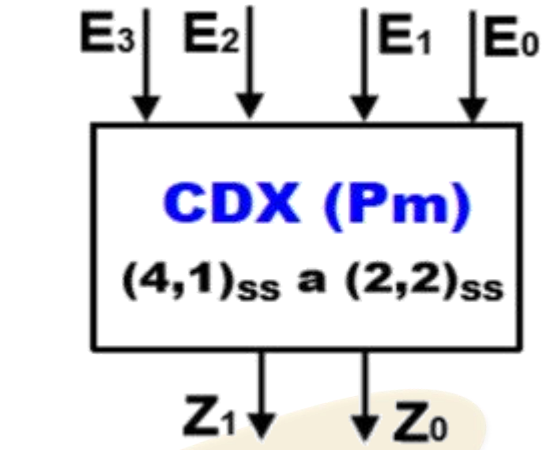
E0	E1	E2	E3	Z1	Z0
1	0	0	0	0	0
X	1	0	0	0	1
X	X	1	0	1	0
X	X	X	1	1	1

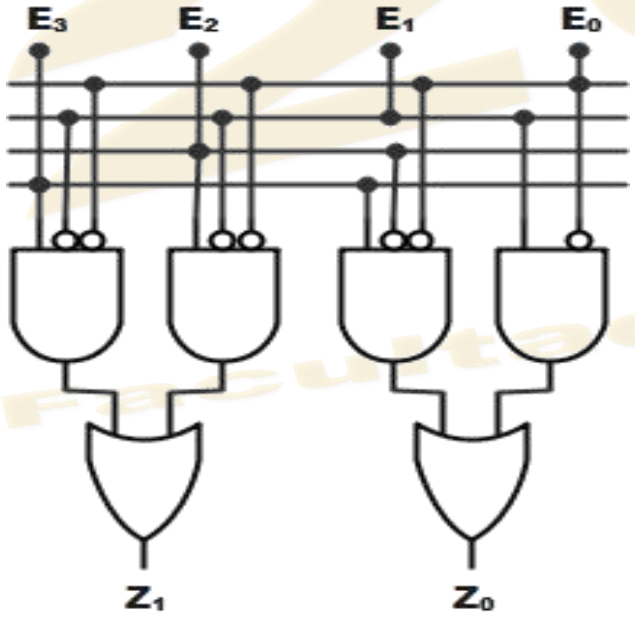
X = representa 0 ó 1

Tablas de verdad reducidas

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Codificadores (CDX)





Ejemplo: Codificador con prioridad a menor de cuaternario $(4,1)_{ss}$ a binario natural $(2,2)_{ss}$

Diseño por T.V.

Prioridad a menor

E0	E1	E2	E3	Z1	Z0	(4,1)ss
1	X	X	X	0	0	0_4
0	1	X	X	0	1	1_4
0	0	1	X	1	0	2_4
0	0	0	1	1	1	3_4

Implementación

Z_1	E_0E_1			
E_2E_3	00	01	11	10
00				
01	1			
11	1			
10	1			

$$Z_1 = \overline{E_0}.\overline{E_1}.E_3 + E_0.\overline{E_1}.E_2$$

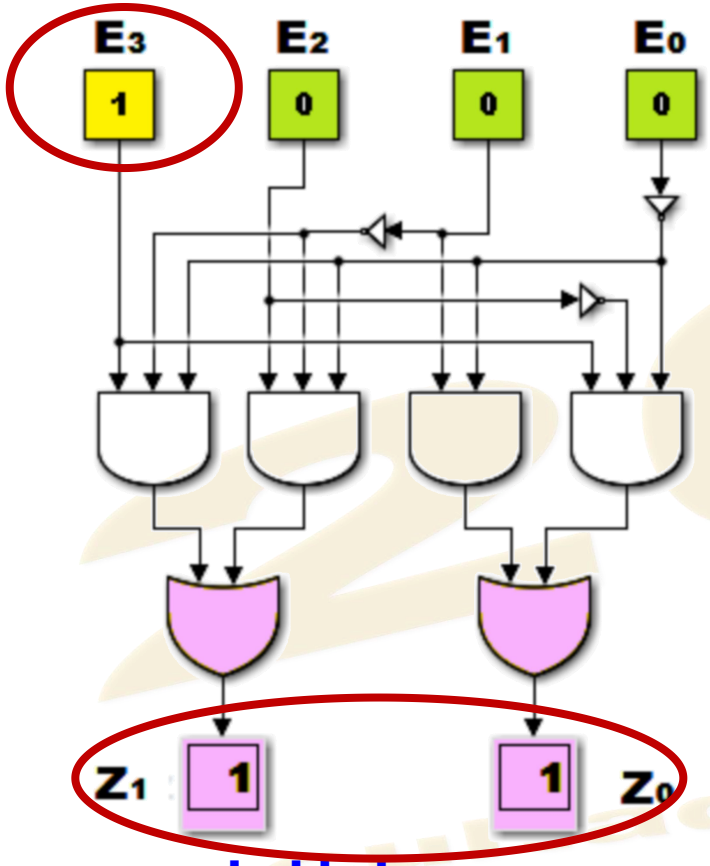
Z_0	E_0E_1			
E_2E_3	00	01	11	10
00		1		
01	1	1		
11		1		
10		1		

$$Z_0 = \overline{E_0}.E_2.E_3 + E_0.\overline{E_1}$$

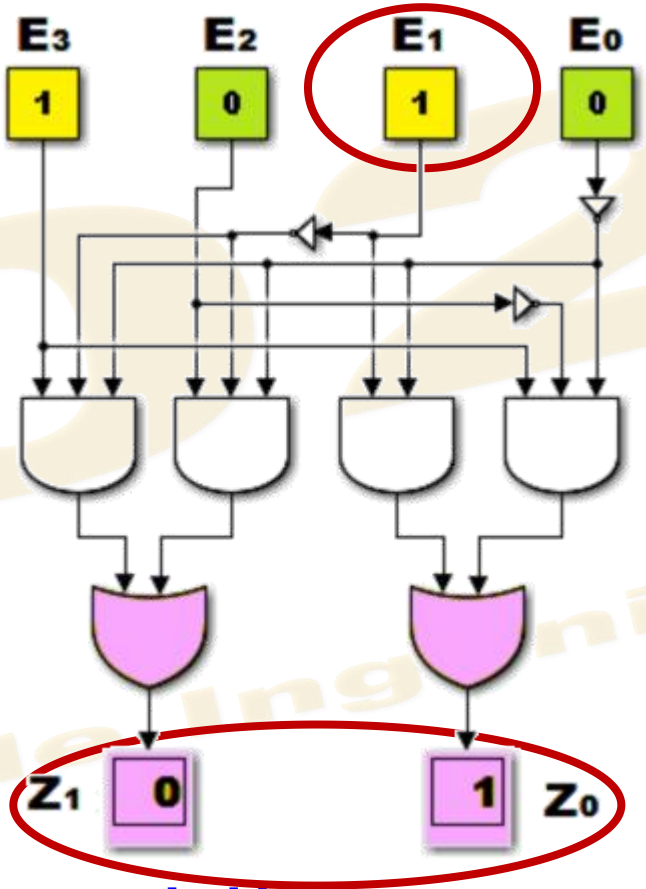
COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Codificadores (CDX)

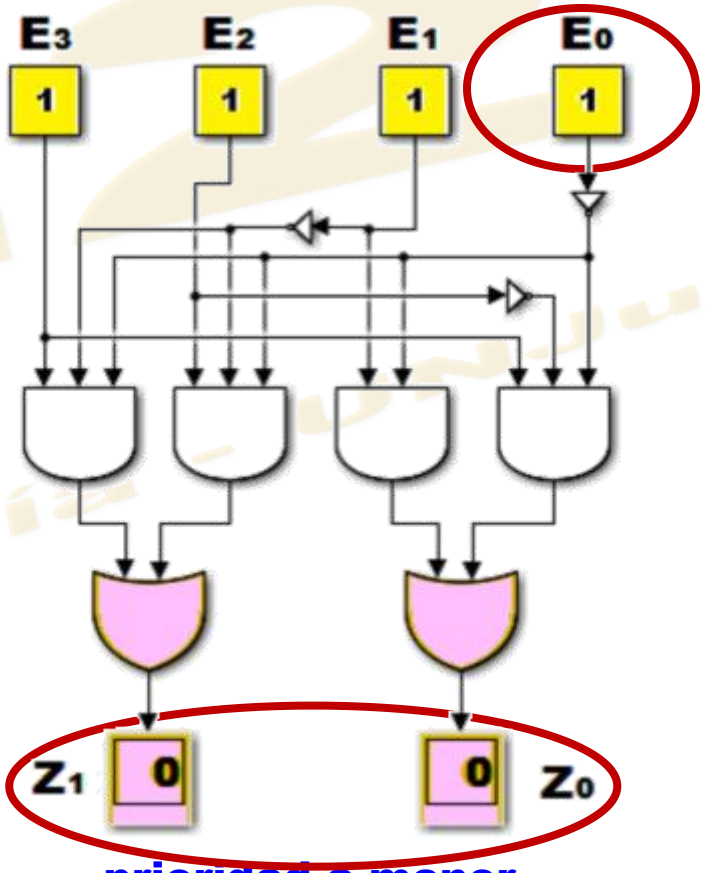
Ejemplo: Codificador con prioridad a menor de cuaternario $(4,1)_{SS}$ a binario natural $(2,2)_{SS}$



prioridad a menor
codifica 3_4



prioridad a menor
codifica 1_4

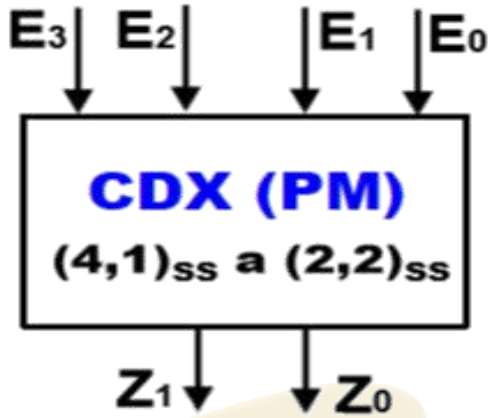


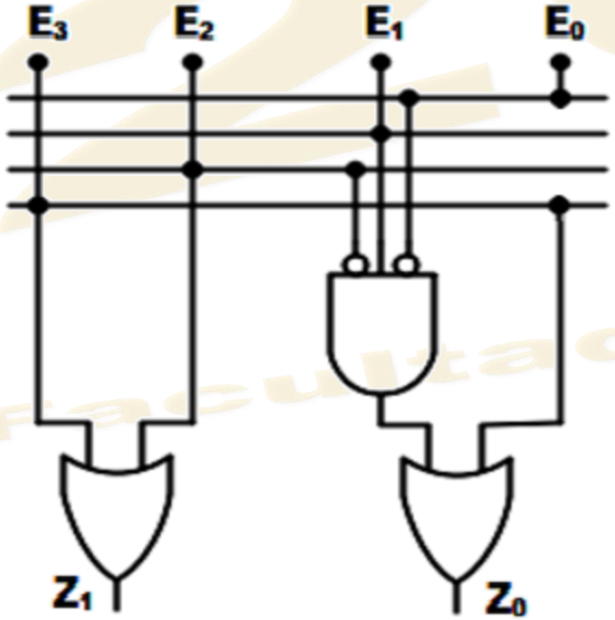
prioridad a menor
codifica 0_4

Simulación en Matlab

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Codificadores (CDX)





Ejemplo: Codificador con prioridad a mayor de cuaternario $(4,1)_{ss}$ a binario natural $(2,2)_{ss}$

Diseño por T.V.

Prioridad a mayor

(4,1)ss	E0	E1	E2	E3	Z1	Z0
0_4	1	0	0	0	0	0
1_4	X	1	0	0	0	1
2_4	X	X	1	0	1	0
3_4	X	X	X	1	1	1

Z_1	E_0E_1			
E_2E_3	00	01	11	10
00				
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$Z_1 = E_2 + E_3$$

Z_0	E_0E_1			
E_2E_3	00	01	11	10
00		1		
01	1	1	1	1
11	1	1	1	1
10				

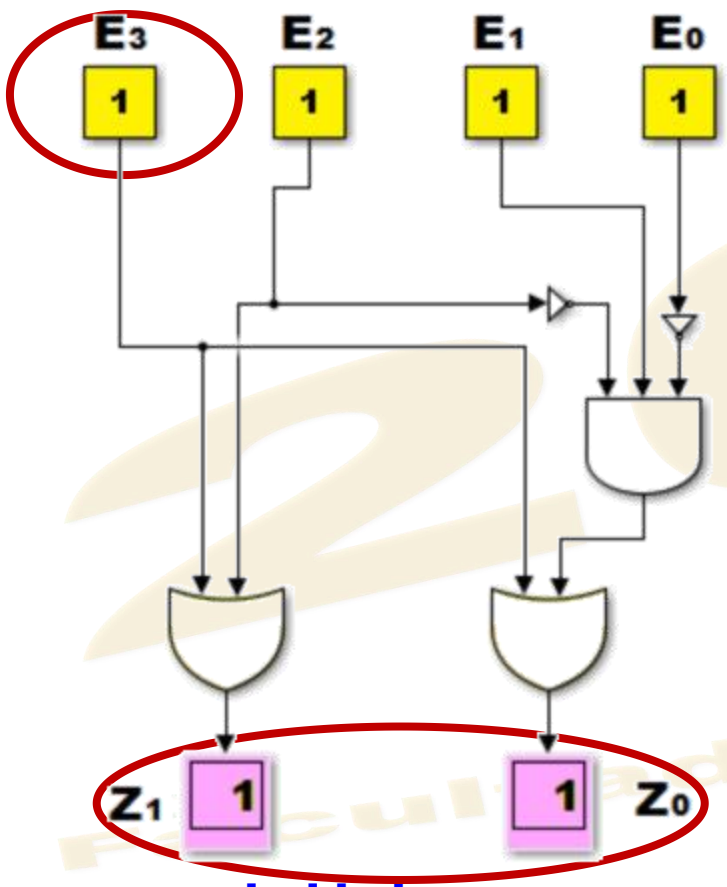
$$Z_0 = \overline{E_0}.E_1.\overline{E_2} + E_3$$

Implementación

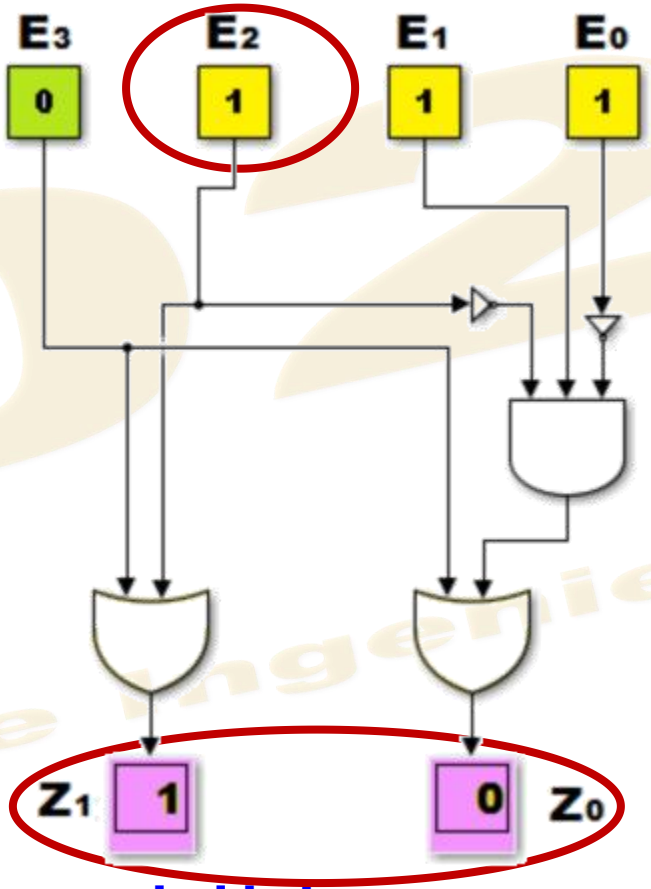
COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Codificadores (CDX)

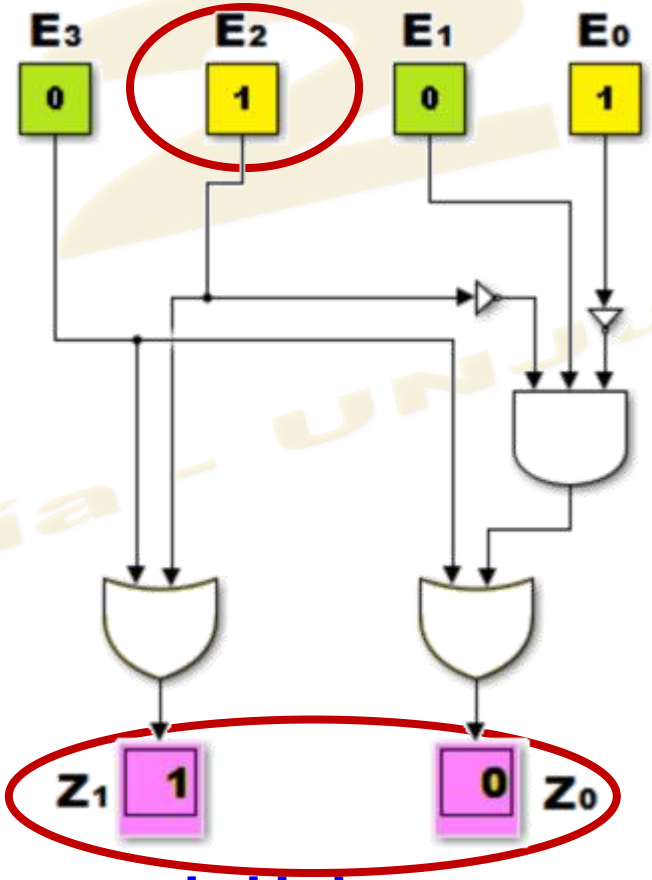
Ejemplo: Codificador con prioridad a mayor de cuaternario $(4,1)_{SS}$ a binario natural $(2,2)_{SS}$



prioridad a mayor
codifica 3_4



prioridad a mayor
codifica 2_4



prioridad a mayor
codifica 2_4

Simulación en Matlab

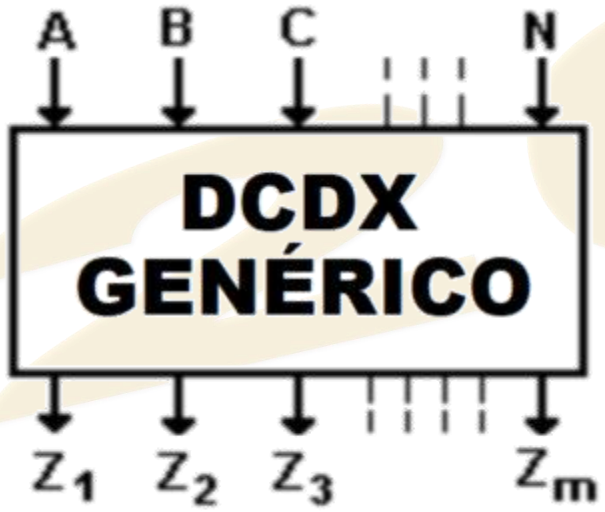
DCDX
DECODIFICADORES

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

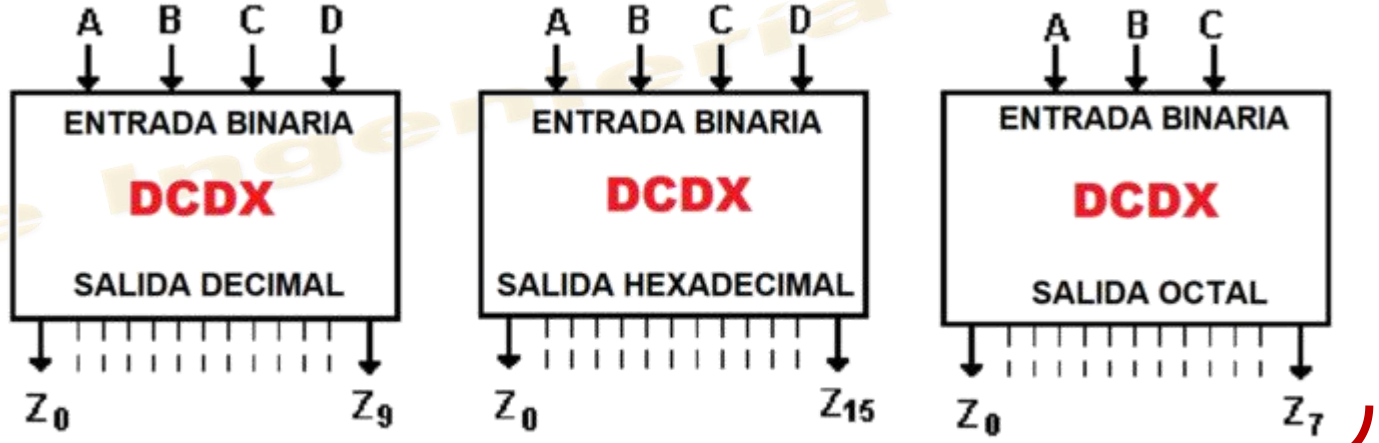
Decodificadores (DCDX)

Mano, pg. 134 - Dormido, pg. 446

Son circuitos combinacionales de N líneas de entradas y m líneas de salida, que al recibir una secuencia binaria en un código preestablecido, produce a la salida un valor equivalente sin codificar (código posicional).



- Decodificadores
  - No excitadores - salida binaria posicional
  - Excitadores
    - 7 segmentos
    - 14 segmentos
    - matriz de puntos
    - otros visualizadores

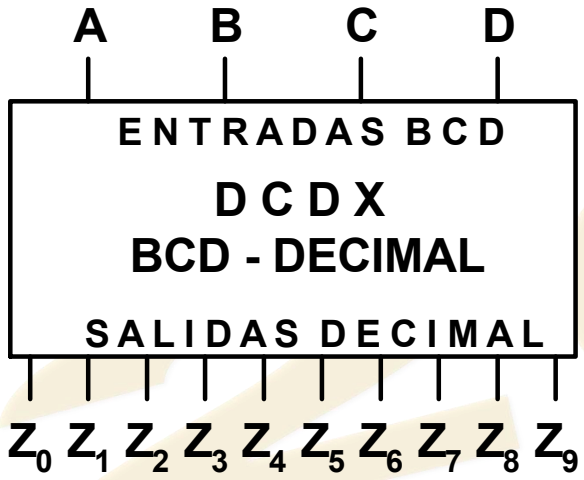


No excitadores - código posicional

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

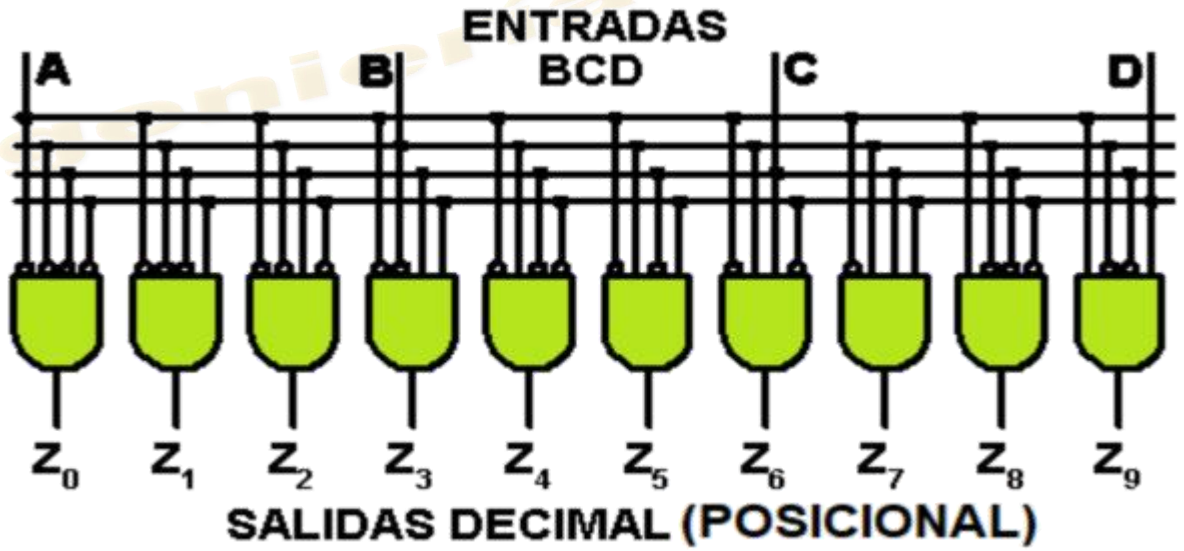
Decodificadores (DCDX)

Ejemplo: **Decodificador BCD natural a decimal no excitador.**



#	A B C D	Z_0	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7	Z_8	Z_9
0	0 0 0 0	**1**	0	0	0	0	0	0	0	0	0
1	0 0 0 1	0	**1**	0	0	0	0	0	0	0	0
2	0 0 1 0	0	0	**1**	0	0	0	0	0	0	0
3	0 0 1 1	0	0	0	**1**	0	0	0	0	0	0
4	0 1 0 0	0	0	0	0	**1**	0	0	0	0	0
5	0 1 0 1	0	0	0	0	0	**1**	0	0	0	0
6	0 1 1 0	0	0	0	0	0	0	**1**	0	0	0
7	0 1 1 1	0	0	0	0	0	0	0	**1**	0	0
8	1 0 0 0	0	0	0	0	0	0	0	0	**1**	0
9	1 0 0 1	0	0	0	0	0	0	0	0	0	**1**
Combinaciones restantes no utilizadas											

Nótese que en este caso el diseño directo es simple: una compuerta AND por salida de codigo y cada una recibe a la entrada las líneas de datos codificadas con negadores.



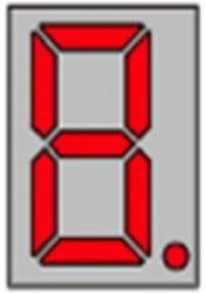
COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Decodificadores (DCDX) excitadores

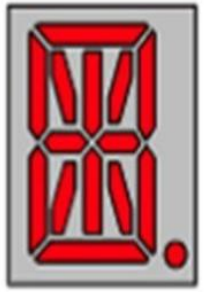
Los decodificadores identificados como excitadores, en vez de dar salida en un código posicional, se diseñan para generar un código especial que sea capaz de activar algún dispositivo visualizador, interpretable por un usuario.

Los dispositivos visualizadores (displays) más comunes se muestran abajo, siendo el de 7 segmentos el más usual.

7 segmentos

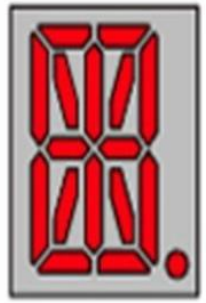


14 segmentos

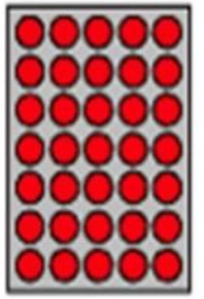


tubo nixie

16 segmentos alfanumérico



matriz de puntos alfanumérico



Dígito 7 segmentos: **Son segmentos independientes de diodos led, que se disponen como en la figura. Cada segmento/diodo, al recibir un '1' lógico se enciende.**

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

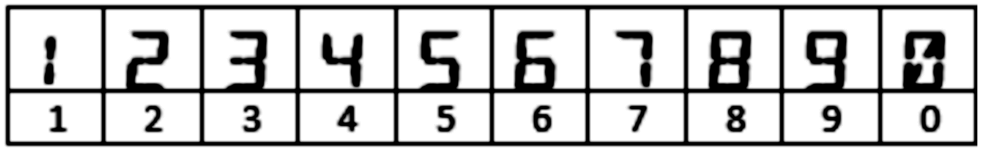
Decodificadores (DCDX)

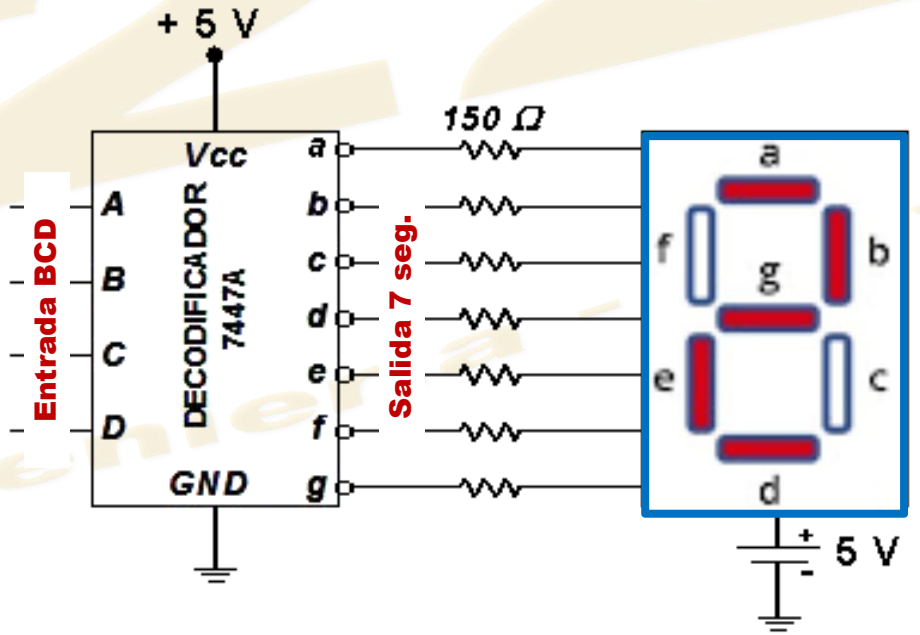
Floyd, pg. 356

Ejemplo: Decodificador excitador de BCD a 7 segmentos

Cada segmento del display se identifica con una letra (de a hasta g). La estrategia para construir la TV es marcar con '1' aquellos segmentos que deban formar el dígito.

	Entrada BCD				Salida 7 segmentos						
dec	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1			0	1	0	1	1	0	0	0	0
2			1	0	1	1	0	1	1	0	1
3			1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5			0	1	1	0	1	1	0	1	1
6			1	0	1	0	1	1	1	1	1
7			1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9			0	1	1	1	1	1	0	1	1



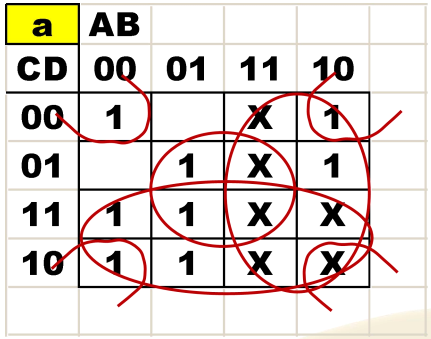


Un procedimiento similar se puede seguir para configurar el código de salida en los otros displays, sólo tener en cuenta la cantidad de segmentos y el diseño de los caracteres.

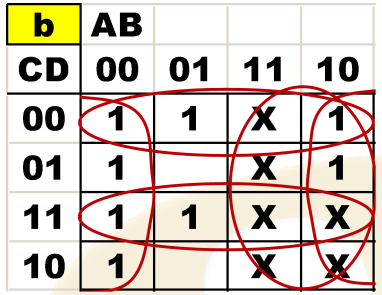
COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Decodificadores (DCDX)

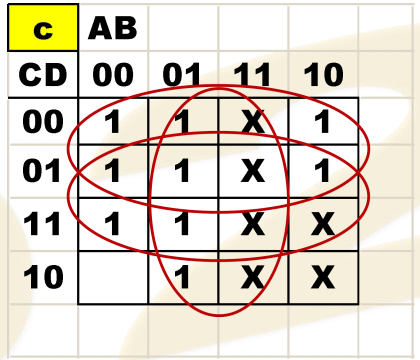
Ejemplo: Decodificador excitador de BCD a 7 segmentos *(continuación)*



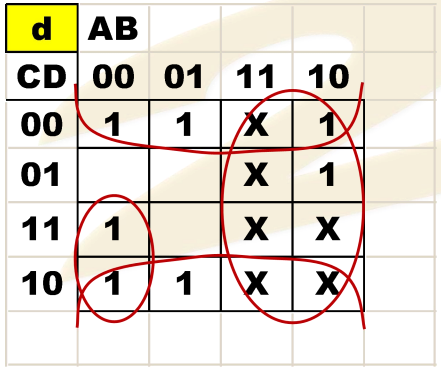
a	AB			
CD	00	01	11	10
00	1		X	1
01		1	X	1
11	1	1	X	X
10	1	1	X	X



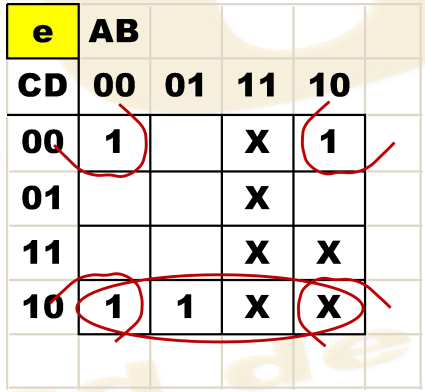
b	AB			
CD	00	01	11	10
00	1	1	X	1
01	1		X	1
11	1	1	X	X
10	1		X	X



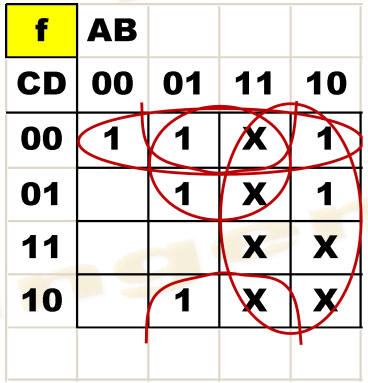
c	AB			
CD	00	01	11	10
00	1	1	X	1
01	1	1	X	1
11	1	1	X	X
10		1	X	X



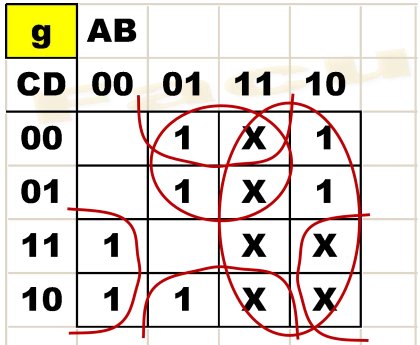
d	AB			
CD	00	01	11	10
00	1	1	X	1
01			X	1
11	1		X	X
10	1	1	X	X



e	AB			
CD	00	01	11	10
00	1		X	1
01			X	
11			X	X
10	1	1	X	X



f	AB			
CD	00	01	11	10
00	1	1	X	1
01		1	X	1
11			X	X
10		1	X	X



g	AB			
CD	00	01	11	10
00		1	X	1
01		1	X	1
11	1		X	X
10	1	1	X	X

$$a = A + C + B.D + \bar{B}.\bar{D}$$

$$b = A + \bar{B} + \bar{C}.\bar{D} + C.D$$

$$c = B + \bar{C} + D$$

$$d = A + \bar{D} + \bar{A}.B.C$$

$$e = \bar{B}.\bar{D} + C.\bar{D}$$

$$f = A + B.\bar{C} + B.\bar{D} + \bar{C}.\bar{D}$$

$$g = A + B.\bar{C} + B.\bar{D} + \bar{B}.C$$

En este caso, se han utilizado las combinaciones mayores a 9_{10}, como salidas indefinidas, aprovechables por Karnaugh.

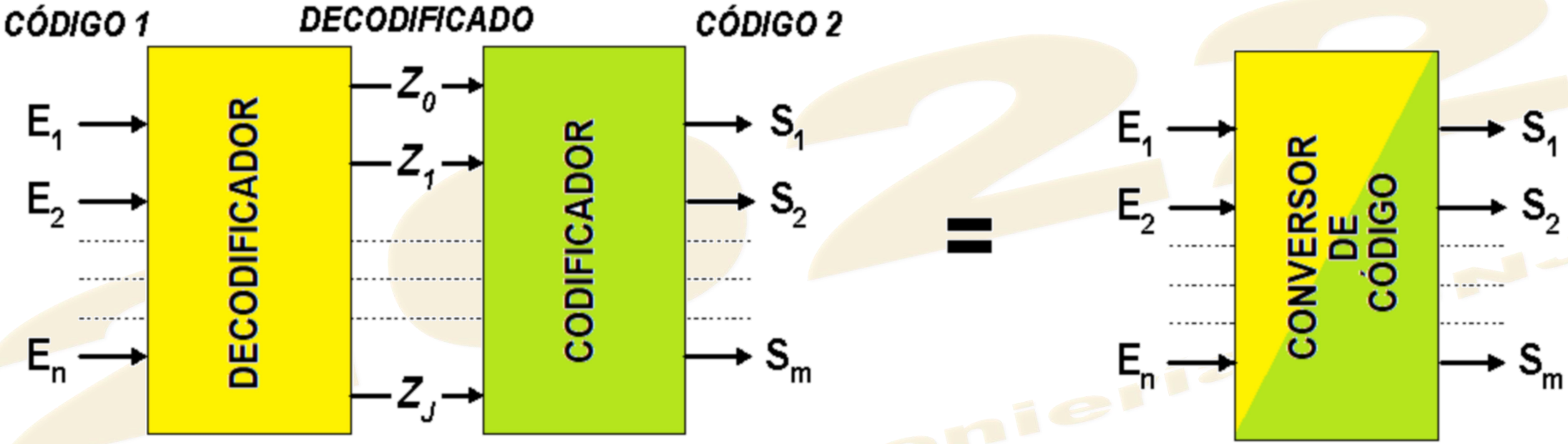
COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

CCDX
CONVERSORES
DE CÓDIGO

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Conversores de código

Floyd, pg. 364



El conversor de código puede verse como la asociación de un decodificador y un codificador.

El diseño se realiza directamente desde el código 1 (como entrada) al código 2 (como salida).

Eventualmente se pueden diseñar por separado y acoplar.

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Conversor de código

Ejemplo: **Conversor de código de BCD 2 de 5 a BCD 2 de 7.**

DEC.	A	B	C	D	E	T	U	V	W	X	Y	Z
	Pesos (2 de 5)					*Pesos (2 de 7)*						
	0	1	2	3	6	5	0	4	3	2	1	0
0	0	1	1	0	0	0	1	0	0	0	0	1
1	1	1	0	0	0	0	1	0	0	0	1	0
2	1	0	1	0	0	0	1	0	0	1	0	0
3	1	0	0	1	0	0	1	0	1	0	0	0
4	0	1	0	1	0	0	1	1	0	0	0	0
5	0	0	1	1	0	1	0	0	0	0	0	1
6	1	0	0	0	1	1	0	0	0	0	1	0
7	0	1	0	0	1	1	0	0	0	1	0	0
8	0	0	1	0	1	1	0	0	1	0	0	0
9	0	0	0	1	1	1	0	1	0	0	0	0

$T = \bar{A}\bar{B}.C.D.\bar{E} + A\bar{B}.\bar{C}.\bar{D}.E + \bar{A}B.\bar{C}.\bar{D}.E + \bar{A}\bar{B}.C.\bar{D}.E + \bar{A}\bar{B}.\bar{C}.D.E$

$U = \bar{A}B.C.\bar{D}.\bar{E} + AB.\bar{C}.\bar{D}.\bar{E} + A\bar{B}.C.\bar{D}.\bar{E} + A\bar{B}.\bar{C}.D.\bar{E} + \bar{A}B.\bar{C}.D.\bar{E}$

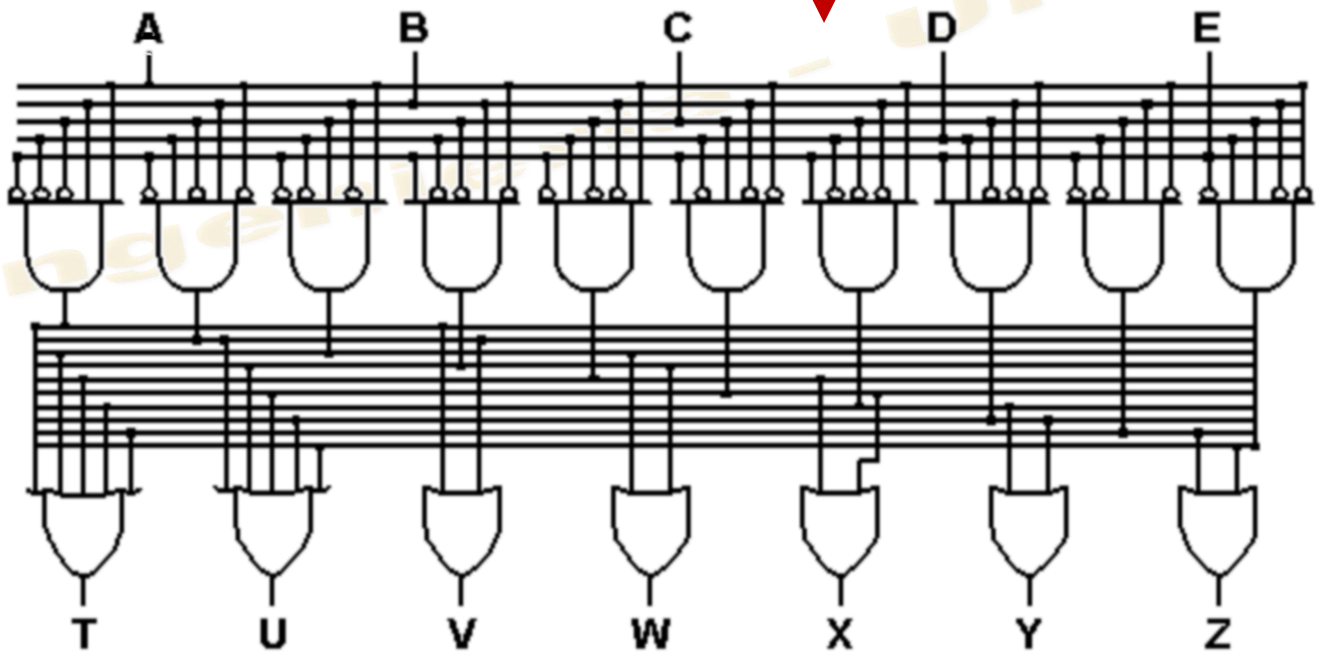
$V = \bar{A}B.\bar{C}.D.\bar{E} + \bar{A}\bar{B}.\bar{C}.D.E$

$W = A\bar{B}.C.D.\bar{E} + \bar{A}B.C.\bar{D}.E$

$X = A\bar{B}.C.\bar{D}.\bar{E} + \bar{A}B.\bar{C}.\bar{D}.E$

$Y = AB.\bar{C}.\bar{D}.\bar{E} + A\bar{B}.\bar{C}.\bar{D}.E$

$Z = \bar{A}B.C.\bar{D}.\bar{E} + \bar{A}\bar{B}.C.D.\bar{E}$



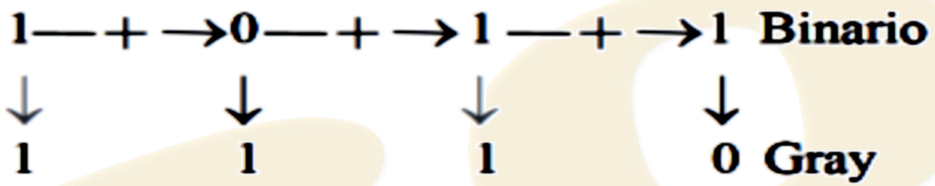
Mano, pg. 116

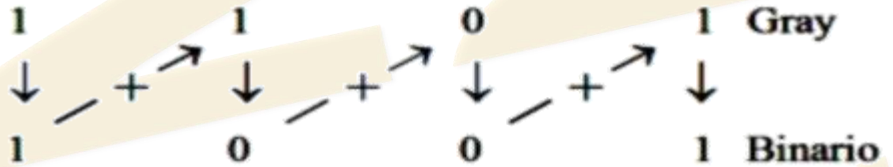
Conversor de código

Floyd, pg. 366

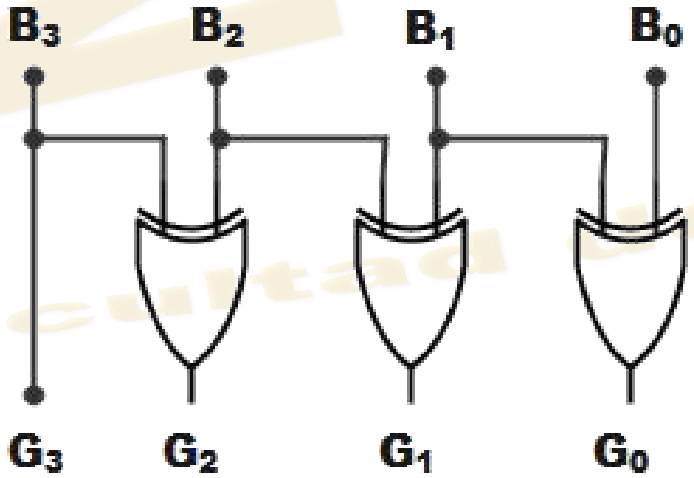
Ejemplo: Conversor de código de binario natural a Gray y viceversa.

Este modelo se puede realizar mediante el procedimiento estándar con la tabla de verdad. Aunque es más sencillo (y es equivalente) utilizar el algoritmo de conversión, como se muestra.



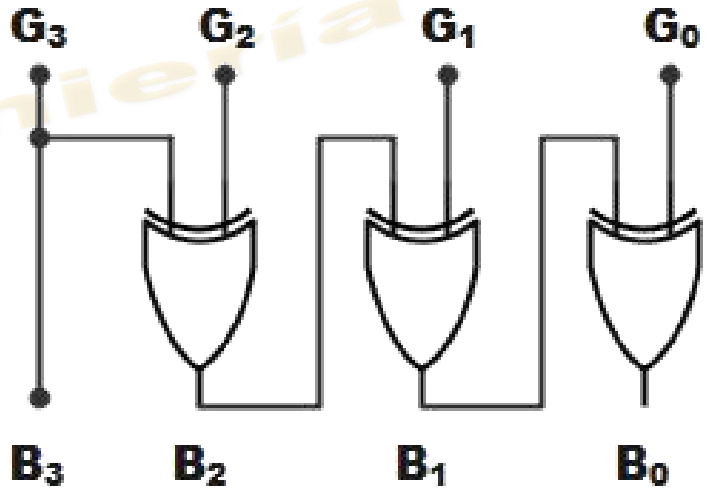


Entrada binaria



Salida Gray

Entrada Gray



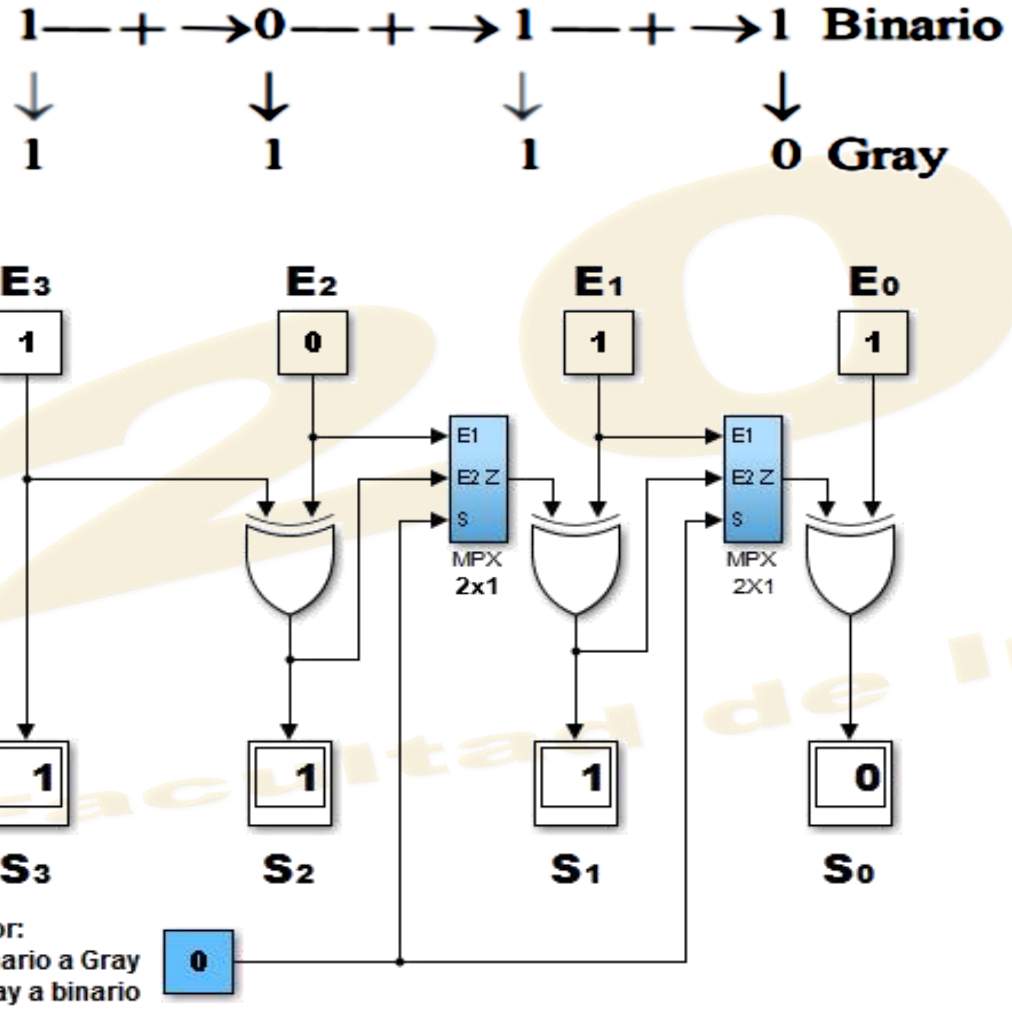
Salida binaria

COMBINACIONALES ESTÁNDARES - CODIFICACIÓN

Conversor de código

Floyd, pg. 366

Ejemplo: Conversor de código de binario natural a Gray y viceversa (continuación)



Simulación en Matlab