Circuitos MSI

Circuitos Digitales EC1723

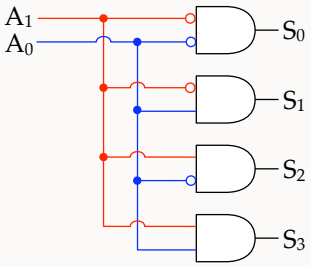
Universidad Simón Bolívar
Departamento de Electrónica y Circuitos
Prof. Juan. C. Regidor

Decodificadores

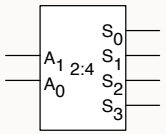
- Un decodificador N:2^N es un circuito combinatorio con N entradas y 2^N salidas. Cada salida "se activa" cuando las entradas, interpretadas como un número binario de N bits, coinciden con su número de orden.
- La tabla de verdad de un decodificador 2:4 con salidas activas en nivel alto es:
- Las salidas son los mintérminos de una función de N variables!

A_1	A_0	S_0	S_1	S_2	S_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**

Decodificadores



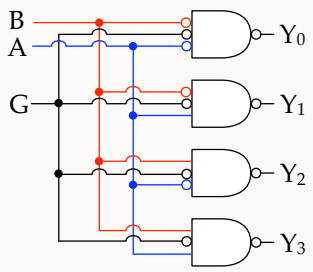
A_1	A_0	S_0	S_1	S_2	S_3
0	0	**1**	0	0	0
0	1	0	**1**	0	0
1	0	0	0	**1**	0
1	1	0	0	0	**1**

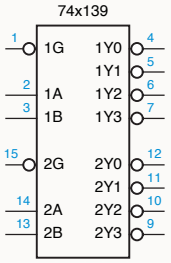


Decodificadores

- Su uso principal es para la selección de memorias o dispositivos que deben activarse en base a un código binario ("dirección").
  - Pueden usarse también para implementar funciones de forma rápida.
- Un decodificador puede tener una o más entradas de "habilitación" que fuerzan las salidas al estado inactivo, sin importar el valor de las entradas A_k.

Decodificador 74139

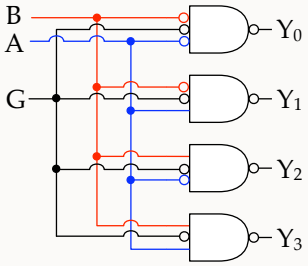


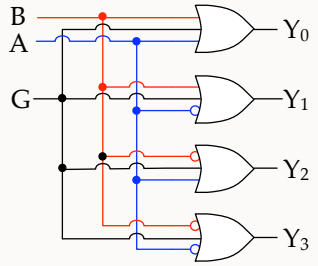


G	B	A	Y_0	Y_1	Y_2	Y_3
0	0	0	**0**	1	1	1
0	0	1	1	**0**	1	1
0	1	0	1	1	**0**	1
0	1	1	1	1	1	**0**
1	X	X	1	1	1	1

Decodificador 74139

- Con salidas activas en nivel bajo, las salidas de este decodificador son los maxtérminos de una función de 2 variables.
- Por De Morgan, podemos representar el circuito:

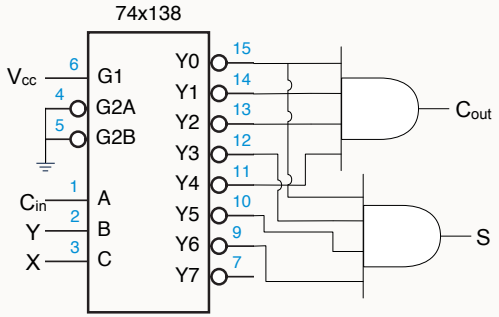




Implementación de funciones con decodificadores

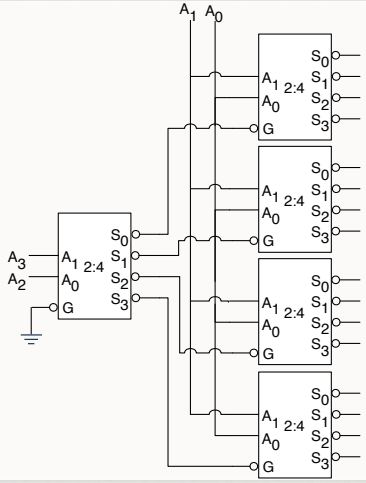
- Sumador completo:

X	Y	C_{in}	C_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



Decodificadores

- Construcción de un decodificador 4:16 con decodificadores 2:4

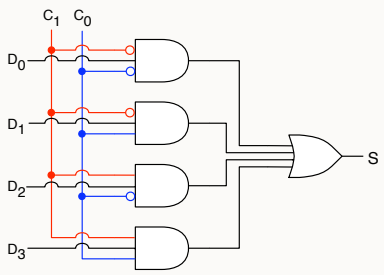


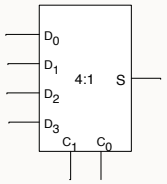
Multiplexores

- Un multiplexor o selector de datos 2^N:1 es un circuito combinatorio con 2^N entradas de datos, N entradas de control y una salida. La salida toma el valor de la entrada cuyo número de orden coincida con el número binario dado por las entradas de control.
- La tabla de verdad de un selector 2:1 es:
  - C: entrada de control
  - A_1, A_0: entradas de datos

C	A_1	A_0	S
0	X	0	0
0	X	1	1
1	0	X	0
1	1	X	1

Multiplexores

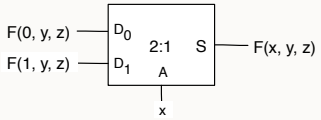


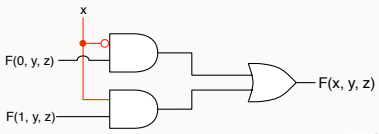


C_1	C_0	S
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

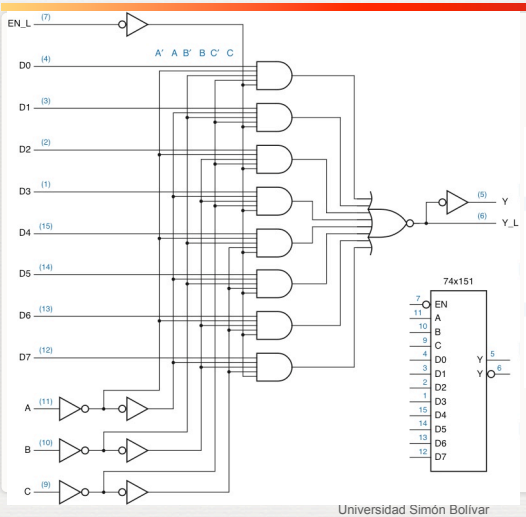
Multiplexores

- Los multiplexores se emplean en aplicaciones en las que es necesario seleccionar un dato de entre varias fuentes para su transmisión (la palabra es tomada de los sistemas telefónicos).
- Se pueden usar también en la implementación de funciones lógicas, aplicando el teorema de expansión de Shannon.



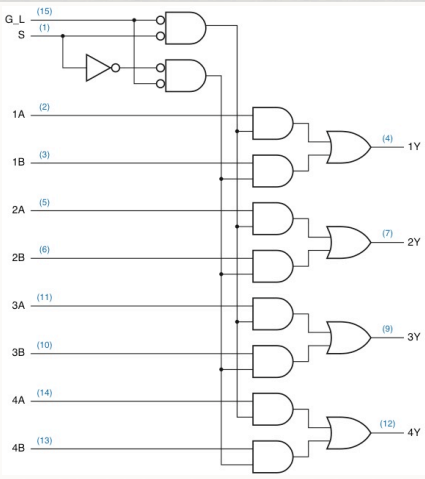


Multiplexor 74151



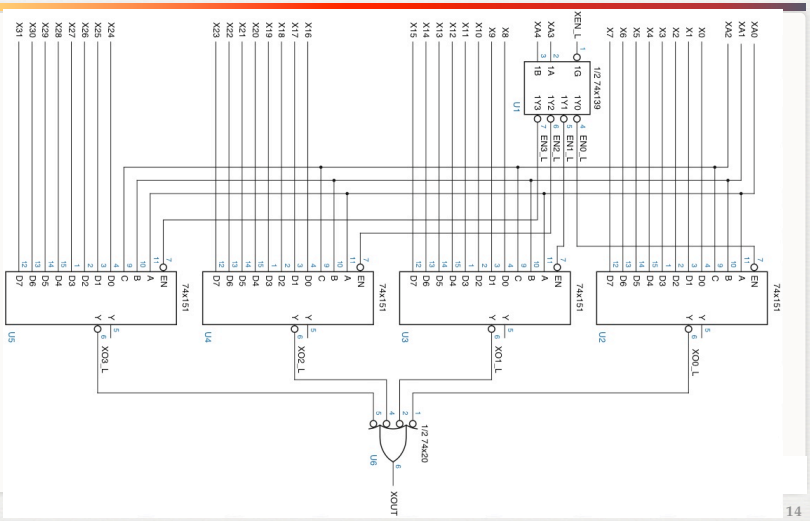
EN_L	C	B	A	Y	Y_L
1	x	x	x	0	1
0	0	0	0	D0	D0′
0	0	0	1	D1	D1′
0	0	1	0	D2	D2′
0	0	1	1	D3	D3′
0	1	0	0	D4	D4′
0	1	0	1	D5	D5′
0	1	1	0	D6	D6′
0	1	1	1	D7	D7′

Multiplexor 74157



G_L	S	1Y	2Y	3Y	4Y
1	x	0	0	0	0
0	0	1A	2A	3A	4A
0	1	1B	2B	3B	4B

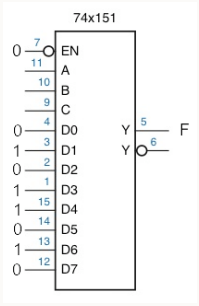
Multiplexor 32:1



Implementación de funciones con multiplexores (1)

- Colocando en las entradas del selector los valores correspondientes de la tabla de verdad:

C	B	A	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X



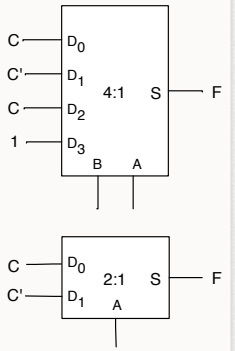
Implementación de funciones con multiplexores (2)

- Se puede reducir el tamaño del mux usando una de las variables en las entradas de datos:

C	B	A	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	X

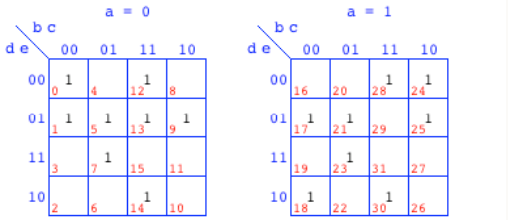
B	A	F
0	0	C
0	1	C’
1	0	C
1	1	1

A	F
0	C
1	C’



Implementación de funciones con multiplexores (3)

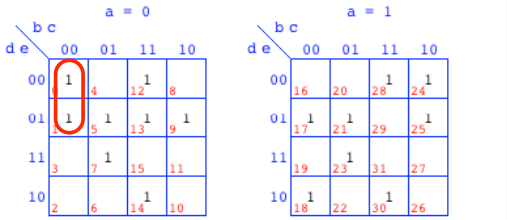
- Para funciones con más variables, se pueden usar funciones de algunas de ellas como entradas del mux, y se pueden implementar estas funciones con otros mux.
- $f(a,b,c,d,e) = \sum(0, 1, 5, 7, 9, 12, 13, 14, 17, 18, 21, 23, 24, 25, 28, 30)$



a	b	c	f

Implementación de funciones con multiplexores (3)

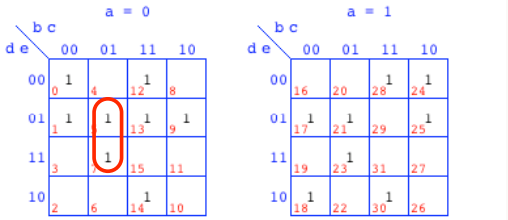
- Para funciones con más variables, se pueden usar funciones de algunas de ellas como entradas del mux, y se pueden implementar estas funciones con otros mux.
- $f(a,b,c,d,e) = \sum(0, 1, 5, 7, 9, 12, 13, 14, 17, 18, 21, 23, 24, 25, 28, 30)$



a	b	c	f
0	0	0	d'

Implementación de funciones con multiplexores (3)

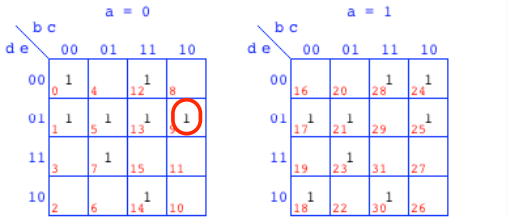
- Para funciones con más variables, se pueden usar funciones de algunas de ellas como entradas del mux, y se pueden implementar estas funciones con otros mux.
- $f(a,b,c,d,e) = \sum(0, 1, 5, 7, 9, 12, 13, 14, 17, 18, 21, 23, 24, 25, 28, 30)$



a	b	c	f
0	0	0	d'
0	0	1	e

Implementación de funciones con multiplexores (3)

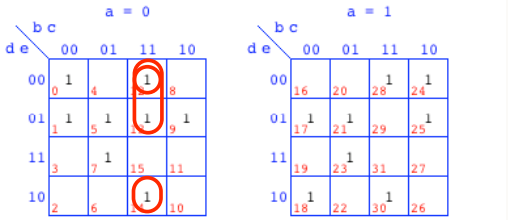
- Para funciones con más variables, se pueden usar funciones de algunas de ellas como entradas del mux, y se pueden implementar estas funciones con otros mux.
- $f(a,b,c,d,e) = \sum(0, 1, 5, 7, 9, 12, 13, 14, 17, 18, 21, 23, 24, 25, 28, 30)$



a	b	c	f
0	0	0	d'
0	0	1	e
0	1	0	d'·e

Implementación de funciones con multiplexores (3)

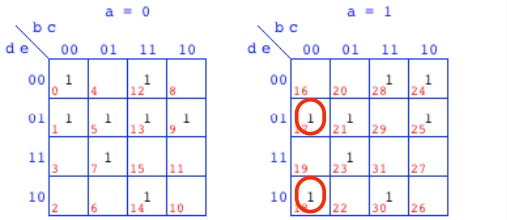
- Para funciones con más variables, se pueden usar funciones de algunas de ellas como entradas del mux, y se pueden implementar estas funciones con otros mux.
- $f(a,b,c,d,e) = \sum(0, 1, 5, 7, 9, 12, 13, 14, 17, 18, 21, 23, 24, 25, 28, 30)$



a	b	c	f
0	0	0	d'
0	0	1	e
0	1	0	d'·e
0	1	1	d'+e'

Implementación de funciones con multiplexores (3)

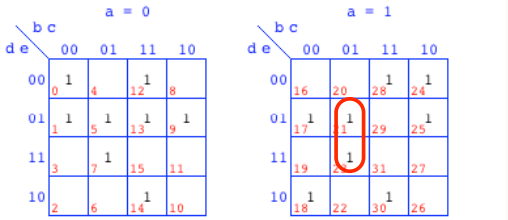
- Para funciones con más variables, se pueden usar funciones de algunas de ellas como entradas del mux, y se pueden implementar estas funciones con otros mux.
- $f(a,b,c,d,e) = \sum(0, 1, 5, 7, 9, 12, 13, 14, 17, 18, 21, 23, 24, 25, 28, 30)$



a	b	c	f
0	0	0	d'
0	0	1	e
0	1	0	d'·e
0	1	1	d'+e'
1	0	0	d⊕e

Implementación de funciones con multiplexores (3)

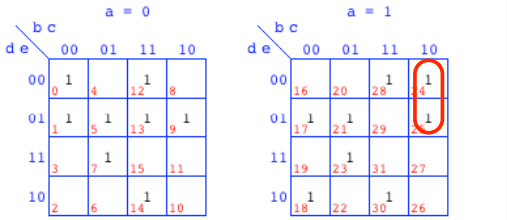
- Para funciones con más variables, se pueden usar funciones de algunas de ellas como entradas del mux, y se pueden implementar estas funciones con otros mux.
- $f(a,b,c,d,e) = \sum(0, 1, 5, 7, 9, 12, 13, 14, 17, 18, 21, 23, 24, 25, 28, 30)$



a	b	c	f
0	0	0	d'
0	0	1	e
0	1	0	d'·e
0	1	1	d'+e'
1	0	0	d⊕e
1	0	1	e

Implementación de funciones con multiplexores (3)

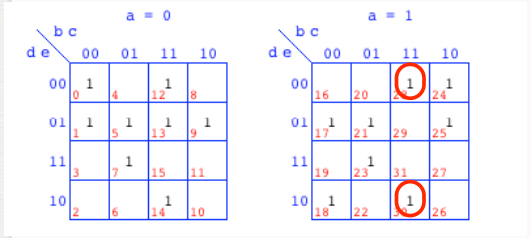
- Para funciones con más variables, se pueden usar funciones de algunas de ellas como entradas del mux, y se pueden implementar estas funciones con otros mux.
- $f(a,b,c,d,e) = \sum(0, 1, 5, 7, 9, 12, 13, 14, 17, 18, 21, 23, 24, 25, 28, 30)$



a	b	c	f
0	0	0	d'
0	0	1	e
0	1	0	d'·e
0	1	1	d'+e'
1	0	0	d⊕e
1	0	1	e
1	1	0	d'

Implementación de funciones con multiplexores (3)

- Para funciones con más variables, se pueden usar funciones de algunas de ellas como entradas del mux, y se pueden implementar estas funciones con otros mux.
- $f(a,b,c,d,e) = \sum(0, 1, 5, 7, 9, 12, 13, 14, 17, 18, 21, 23, 24, 25, 28, 30)$

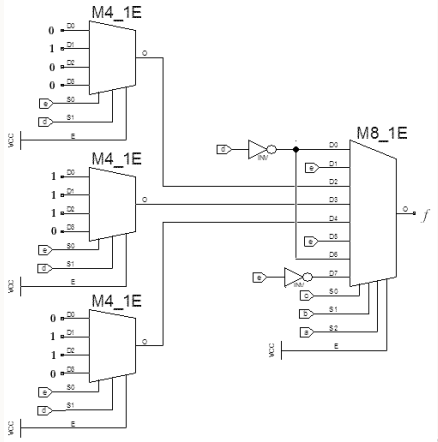


a	b	c	f
0	0	0	d’
0	0	1	e
0	1	0	d’·e
0	1	1	d’+e’
1	0	0	d⊕e
1	0	1	e
1	1	0	d’
1	1	1	e’

Implementación de funciones con multiplexores (4)

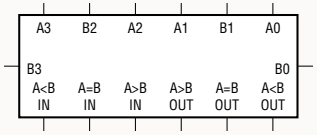
- Suponiendo que sólo se dispone de un mux 8:1, tres mux 4:1, y negadores:

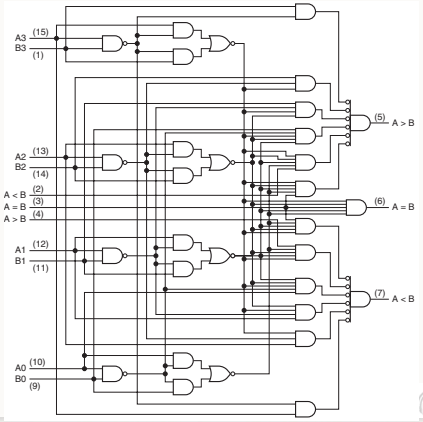
a	b	c	f
0	0	0	d’
0	0	1	e
0	1	0	d’·e
0	1	1	d’+e’
1	0	0	d⊕e
1	0	1	e
1	1	0	d’
1	1	1	e’



Comparadores

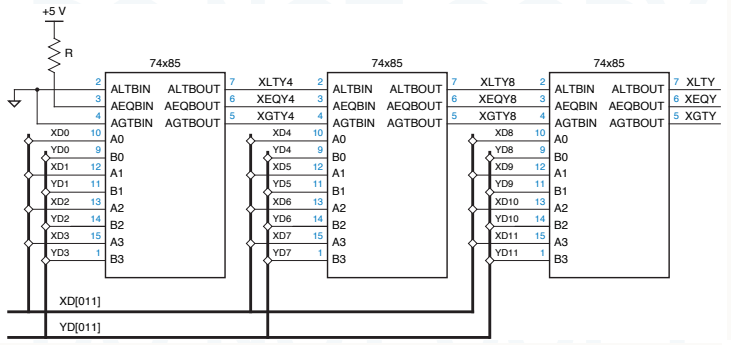
- Comparador de magnitud de 4 bits 7485.





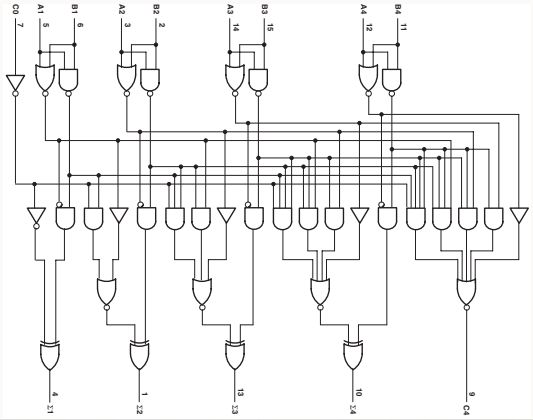
Comparadores

- Las entradas A<B IN, A=B IN y A>B IN permiten conectar varios comparadores en cascada:



Sumadores con generación rápida de acarreo

- Sumador 74283 con *carry look-ahead*.



Unidad Lógica Aritmética

- ULA: Unidad Lógica Aritmética (*Arithmetic Logic Unit, ALU*). La ULA es el componente central de un procesador.
- ULA de 4 bits 74181. Es posible conectar varias en cascada (posiblemente con un generador de acarreo para aceleración) para procesar más bits.