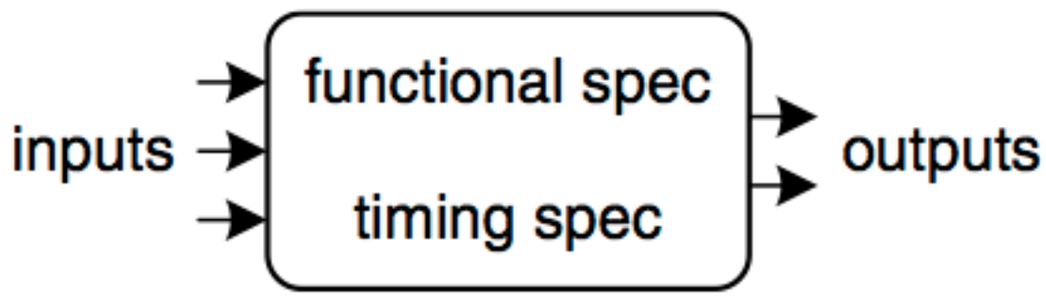
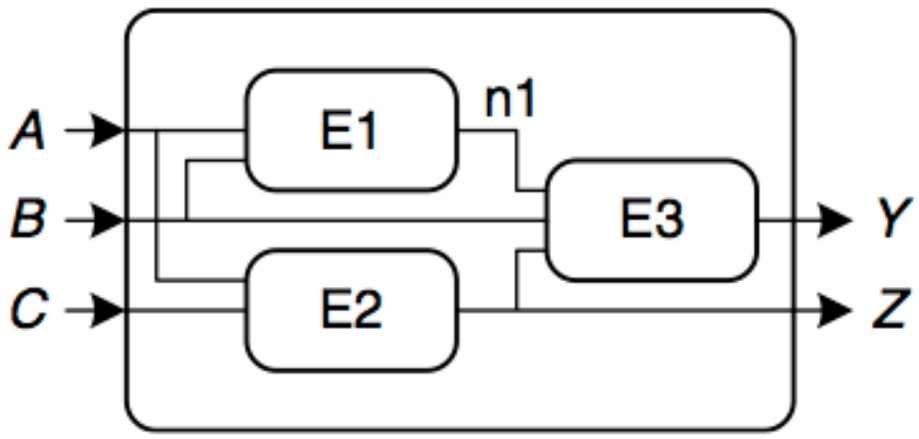
Introducción

- En electrónica digital, un *circuito* es una red que procesa variables de valores discretos.
- Un circuito puede considerarse como una caja negra que tiene:
  - Uno o más terminales de entrada
  - Uno o más terminales de salida
  - Una especificación funcional que describe la relación entre las entradas y las salidas
  - Una especificación temporal que describe el retardo entre un cambio en la entrada y la respuesta de la salida.



Introducción

- Mirando dentro de la caja negra, los circuitos están compuestos de *nodos* y *elementos*
- Un *elemento* es en sí un circuito, con entradas, salidas y especificaciones.
- Un nodo es un alambre cuyo voltaje corresponde a una variable discreta
  - Los nodos se clasifican como *entradas*, *salidas* e *internos*

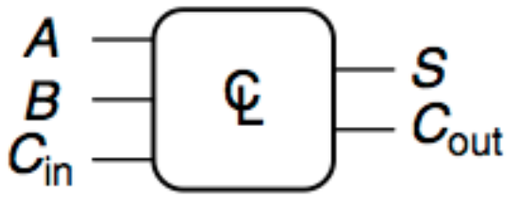


Introducción

Circuito combinacional

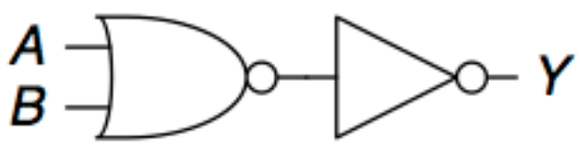
$$Y = F(A, B) = A + B$$

Circuito combinacional de múltiples salidas



$$S = A \oplus B \oplus C_{in}$$
$$C_{out} = AB + AC_{in} + BC_{in}$$

Dos posibles implementaciones



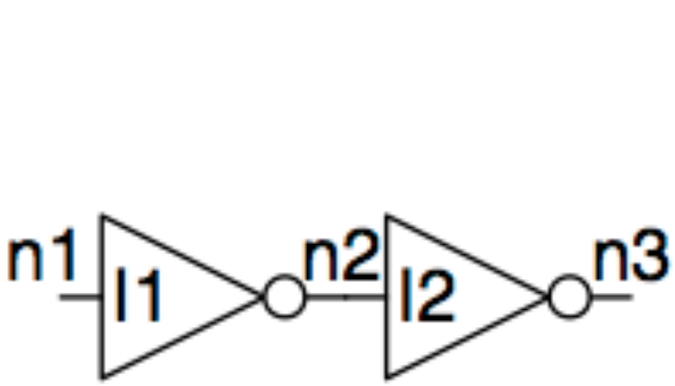
Notación de buses

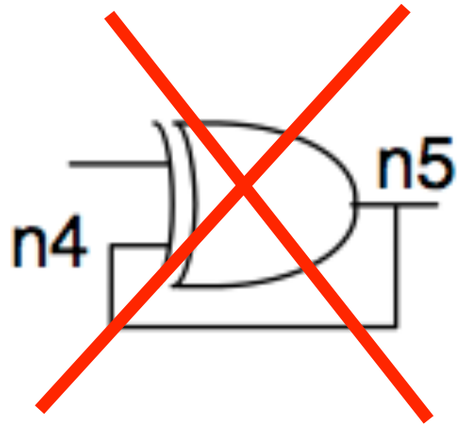
Introducción

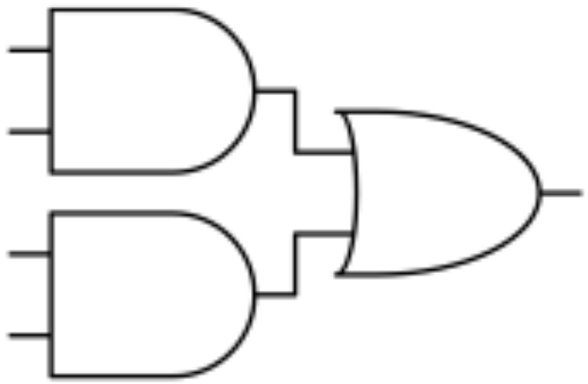
- Un circuito es combinacional si:
- Cada elemento que lo compone es en sí mismo combinacional
- Cada nodo del circuito es designado como entrada del circuito o conecta exactamente a un terminal de salida de un elemento de circuito
- El circuito no contiene caminos cíclicos: cada camino a través del circuito visita a cada nodo del circuito a lo más una sola vez.

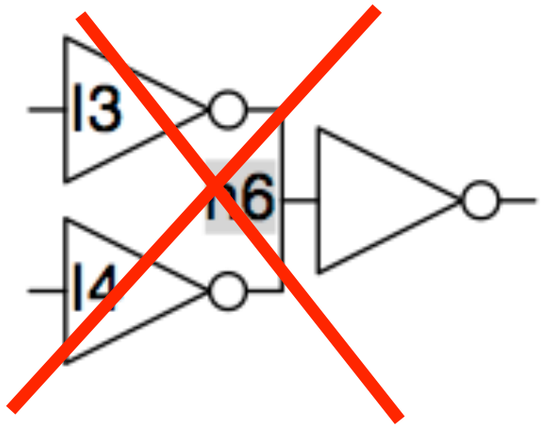
Introducción

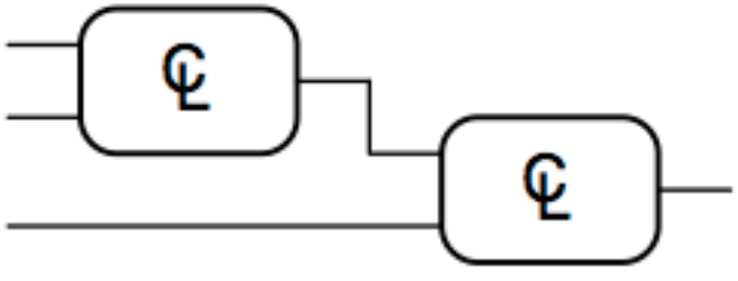
¿ Cuáles son combinacionales?

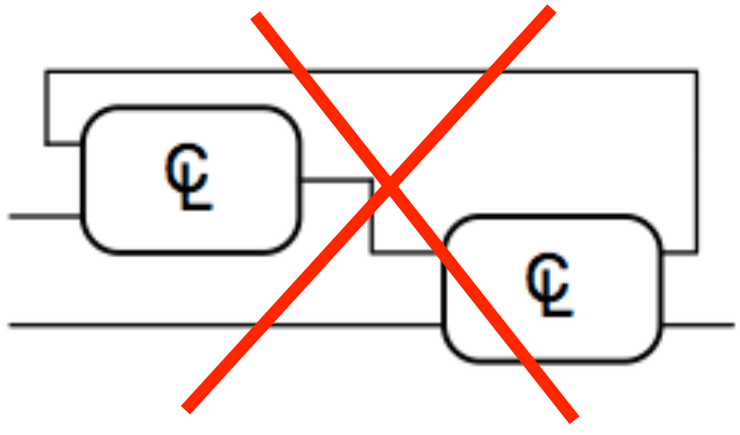












Ecuaciones Booleanas

Terminología

- El complemento de una variable, A, es su inverso, $\bar{A}$.
- La variable o su complemento se llama literal.
- Llamamos a A la forma verdadera de la variable y $\bar{A}$ la forma complementaria.
- Al AND de uno o más literales se le llama un producto o un implicante. $\bar{A}B$, $A\bar{B}\bar{C}$ y B son todos implicantes para una función de 3 variables.
- Un minitérmino es un producto de todas las variables de una función, en forma verdadera o en forma complementaria. Ej. $A\bar{B}\bar{C}$ es un minitérmino de una función de 3 variables A, B y C. $\bar{A}B$ no lo es.

Ecuaciones Booleanas

Terminología

- En forma similar al OR de uno o más literales se le llama suma.
- Un maxitérmino es una suma de todas las variables de entrada de una función. Ej. $A+B+C$ es un maxitérmino para una función de tres variables A, B y C
- El orden de las operaciones es importante. NOT tiene la precedencia más alta, después AND y finalmente OR. Ej. $Y = A + B\bar{C}$

Ecuaciones Booleanas

Suma de productos

minitérmino

A	B	Y	
0	0	0	$\overline{A}\ \overline{B}$
0	1	1	$\overline{A}\ B$
1	0	0	$A\ \overline{B}$
1	1	1	$A\ B$

$$Y = \overline{A}B + AB$$

Forma canónica de suma de productos

Ecuaciones Booleanas

Suma de productos

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

Forma canónica de suma de productos

$$Y = \overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C} + A\overline{B}C$$

Ecuaciones Booleanas

Producto de sumas

A	B	Y	maxitérmino
0	0	0	$A + B$
0	1	1	$A + \overline{B}$
1	0	0	$\overline{A} + B$
1	1	1	$\overline{A} + \overline{B}$

$$Y = (A + B)(\overline{A} + B)$$

Forma canónica de producto de sumas

Álgebra Booleana

AXIOMAS

	Axiom		Dual	Name
A1	$B = 0$ if $B \neq 1$	A1′	$B = 1$ if $B \neq 0$	Binary field
A2	$\overline{0} = 1$	A2′	$\overline{1} = 0$	NOT
A3	$0 \bullet 0 = 0$	A3′	$1 + 1 = 1$	AND/OR
A4	$1 \bullet 1 = 1$	A4′	$0 + 0 = 0$	AND/OR
A5	$0 \bullet 1 = 1 \bullet 0 = 0$	A5′	$1 + 0 = 0 + 1 = 1$	AND/OR

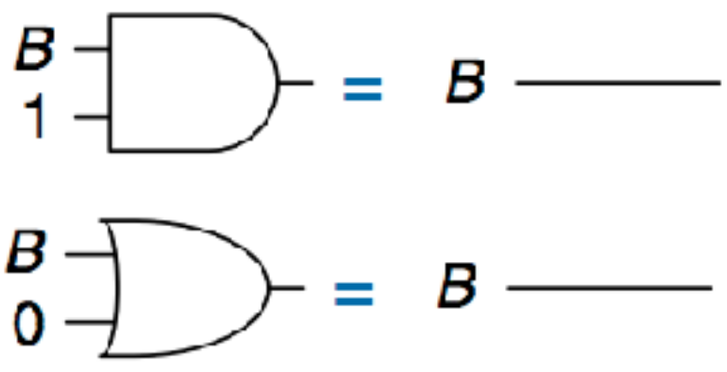
Estos 5 axiomas con sus duales definen las variables Booleanans y el significado de NOT, AND y OR.

Álgebra Booleana

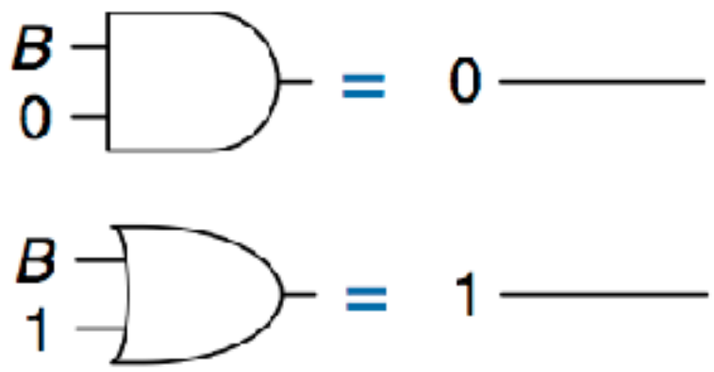
Teoremas de una variable

	Theorem		Dual	Name
T1	$B \bullet 1 = B$	T1′	$B + 0 = B$	Identity
T2	$B \bullet 0 = 0$	T2′	$B + 1 = 1$	Null Element
T3	$B \bullet B = B$	T3′	$B + B = B$	Idempotency
T4	$\overline{\overline{B}} = B$			Involution
T5	$B \bullet \overline{B} = 0$	T5′	$B + \overline{B} = 1$	Complements

Identidad en hardware

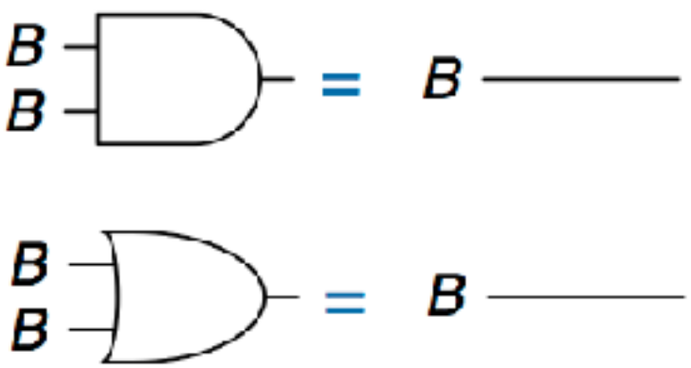


Elemento nulo en hardware

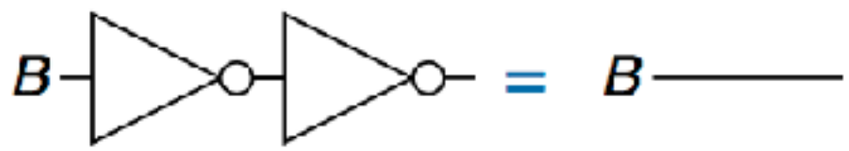


Álgebra Booleana

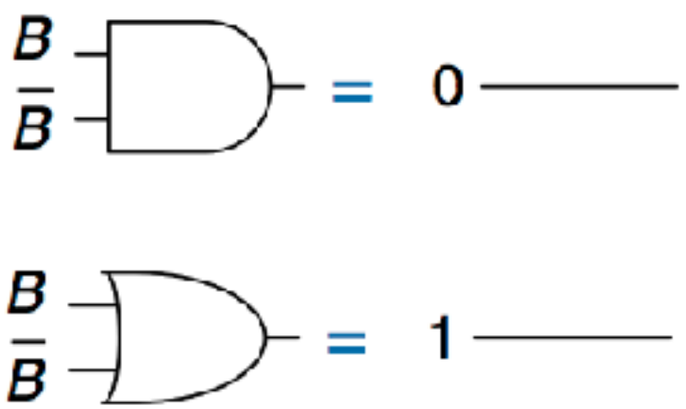
Idempotencia



Involución



Complemento



Álgebra Booleana

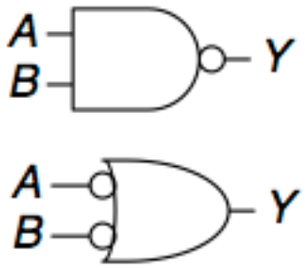
Teoremas de varias variables

	Theorem		Dual	Name
T6	$B \bullet C = C \bullet B$	T6′	$B + C = C + B$	Commutativity
T7	$(B \bullet C) \bullet D = B \bullet (C \bullet D)$	T7′	$(B + C) + D = B + (C + D)$	Associativity
T8	$(B \bullet C) + (B \bullet D) = B \bullet (C + D)$	T8′	$(B + C) \bullet (B + D) = B + (C \bullet D)$	Distributivity
T9	$B \bullet (B + C) = B$	T9′	$B + (B \bullet C) = B$	Covering
T10	$(B \bullet C) + (B \bullet \overline{C}) = B$	T10′	$(B + C) \bullet (B + \overline{C}) = B$	Combining
T11	$(B \bullet C) + (\overline{B} \bullet D) + (C \bullet D)$ $= B \bullet C + \overline{B} \bullet D$	T11′	$(B + C) \bullet (\overline{B} + D) \bullet (C + D)$ $= (B + C) \bullet (\overline{B} + D)$	Consensus
T12	$\overline{B_0 \bullet B_1 \bullet B_2 \ldots}$ $= (\overline{B_0} + \overline{B_1} + \overline{B_2} \ldots)$	T12′	$\overline{B_0 + B_1 + B_2 \ldots}$ $= (\overline{B_0} \bullet \overline{B_1} \bullet \overline{B_2})$	De Morgan's Theorem

Álgebra Booleana

Equivalente en hardware del Teorema de De Morgan

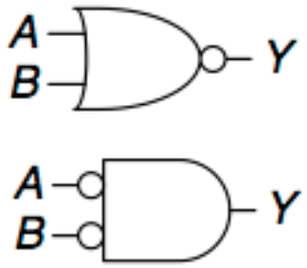
NAND



$Y = \overline{AB} = \bar{A} + \bar{B}$

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

NOR



$Y = \overline{A + B} = \bar{A}\,\bar{B}$

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

Álgebra Booleana

Para la tabla de verdad de la función Booleana, *Y*, y su complemento $\bar{Y}$, encontrar la forma canónica de producto de sumas de *Y*, a partir de la forma canónica de suma de productos que representa a *Y*

A	*B*	*Y*	$\bar{Y}$
0	0	0	1
0	1	0	1
1	0	1	0
1	1	1	0

A	*B*	*Y*	$\bar{Y}$	minterm
0	0	0	**1**	$\bar{A}\,\bar{B}$
0	1	0	**1**	$\bar{A}\,B$
1	0	1	**0**	$A\,\bar{B}$
1	1	1	**0**	$A\,B$

Álgebra Booleana

Demostración el Teorema del Consenso

B	C	D	$BC+\bar{B}D+CD$	$BC+\bar{B}D$
0	0	0	0	0
0	0	1	1	1
0	1	0	0	0
0	1	1	1	1
1	0	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	1	1

Álgebra Booleana

Simplificación o minimización de ecuaciones

Step	Equation	Justification
	$\overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C} + A\overline{B}C$	
1	$\overline{B}\overline{C}\,(\overline{A} + A) + A\overline{B}C$	T8: Distributivity
2	$\overline{B}\overline{C}\,(1) + A\overline{B}C$	T5: Complements
3	$\overline{B}\overline{C} + A\overline{B}C$	T1: Identity

Álgebra Booleana

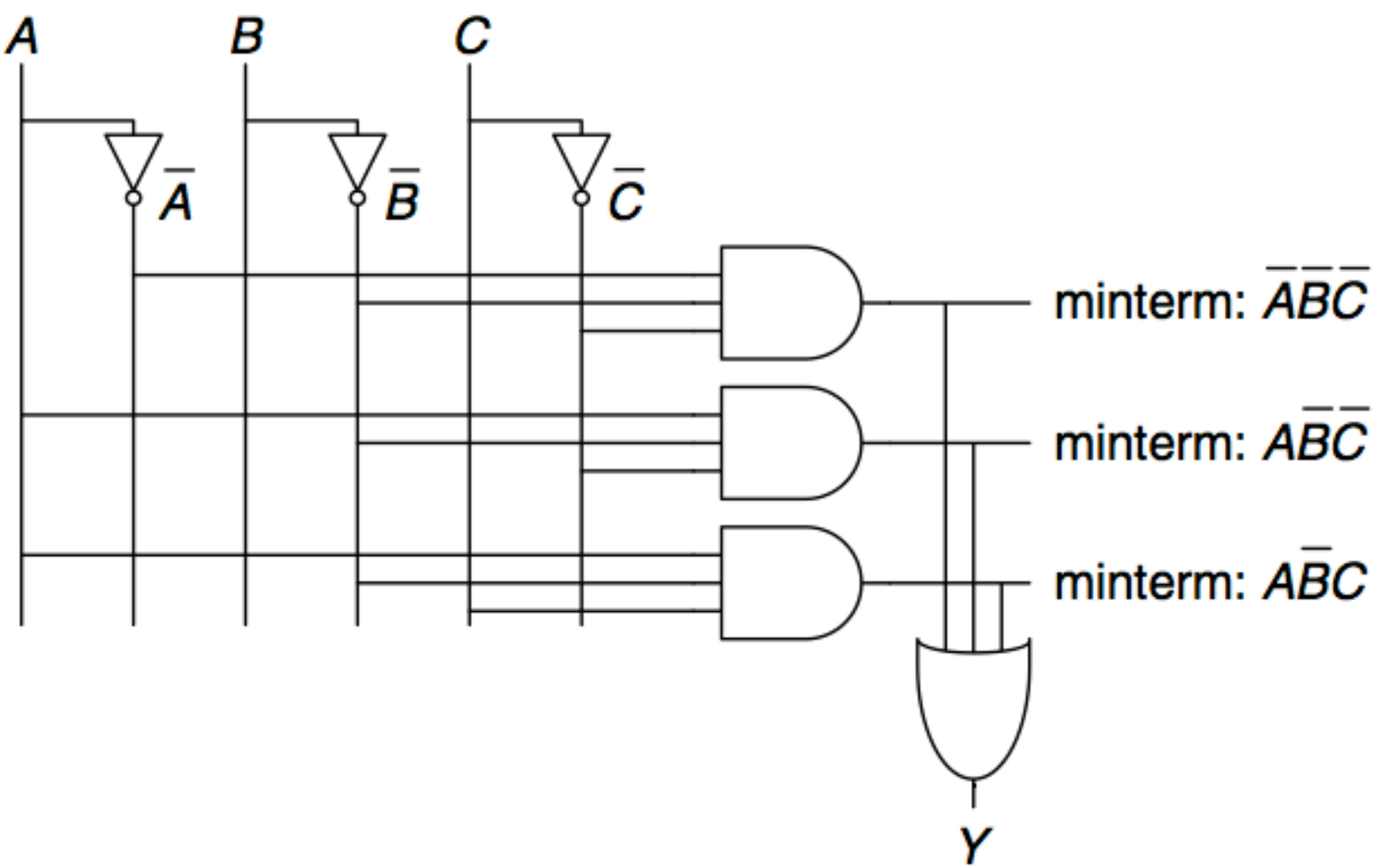
Mejor aún

Step	Equation	Justification
	$\overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C} + A\overline{B}C$	
1	$\overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C} + A\overline{B}\overline{C} + A\overline{B}C$	T3: Idempotency
2	$\overline{B}\overline{C}(\overline{A} + A) + A\overline{B}(\overline{C} + C)$	T8: Distributivity
3	$\overline{B}\overline{C}(1) + A\overline{B}(1)$	T5: Complements
4	$\overline{B}\overline{C} + A\overline{B}$	T1: Identity

De las ecuaciones al circuito

- Un esquemático es un diagrama de un circuito digital que muestra los elementos que lo componen y las interconexiones entre ellos. Ejemplo

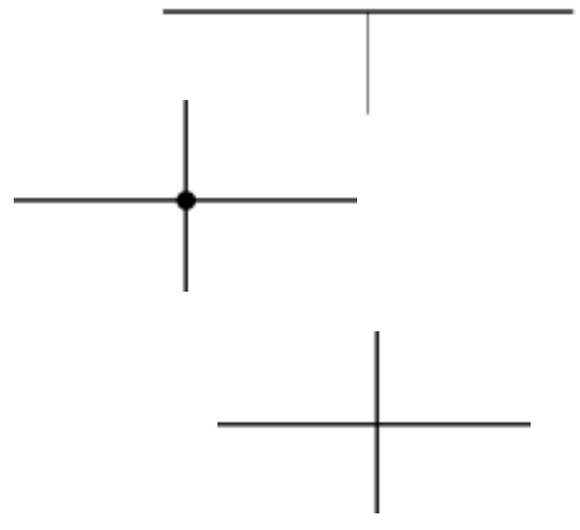
$$Y = \overline{A}\overline{B}\overline{C} + A\overline{B}\overline{C} + A\overline{B}C.$$



De las ecuaciones al circuito

Dibujar los esquemáticos en forma consistente y obedeciendo ciertas reglas, los hace más legibles y fáciles de interpretar

- Las entradas se ubican en la izquierda o arriba
- Las salidas se ubican a la derecha o abajo
- Siempre que sea posible, las compuertas deben fluir de izquierda a derecha
- Conviene dibujar alambres rectos que alambres con esquinas.
- Los alambres siempre conectan en una juntura T
- Un punto donde cruzan dos alambres significa que están conectados
- Dos alambres que se cruzan sin un punto, no están conectados

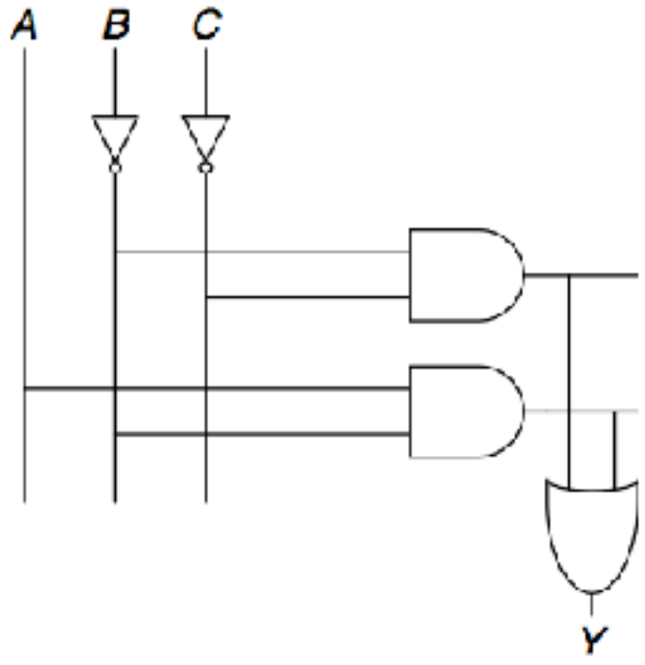


De las ecuaciones al circuito

Para dibujar suma de productos:

- Poner los inversores en columnas adjacentes, a la izquierda para proveer las entradas complementarias si es que se requieren.
- Dibujar una fila de ANDs, una para cada producto.
- Luego para cada salida, dibujar un OR que sume los productos involucrados en la salida correspondiente

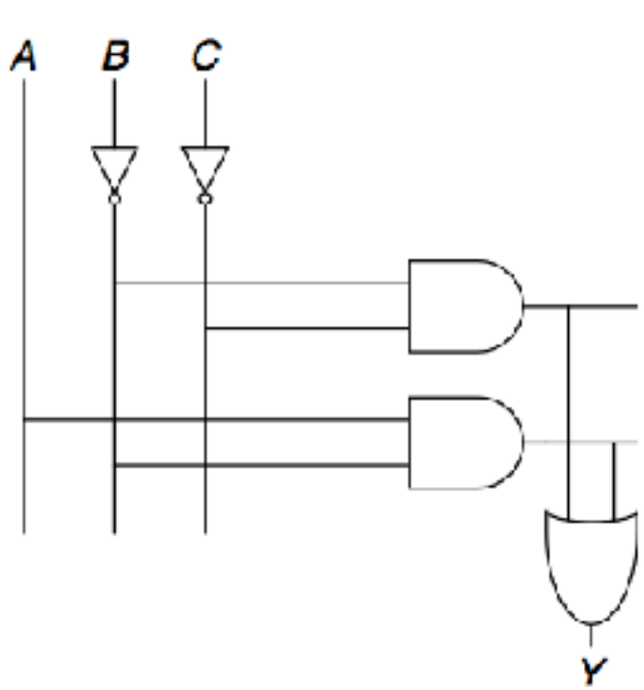
Ejemplo para $Y = \bar{B}\bar{C} + A\bar{B}$

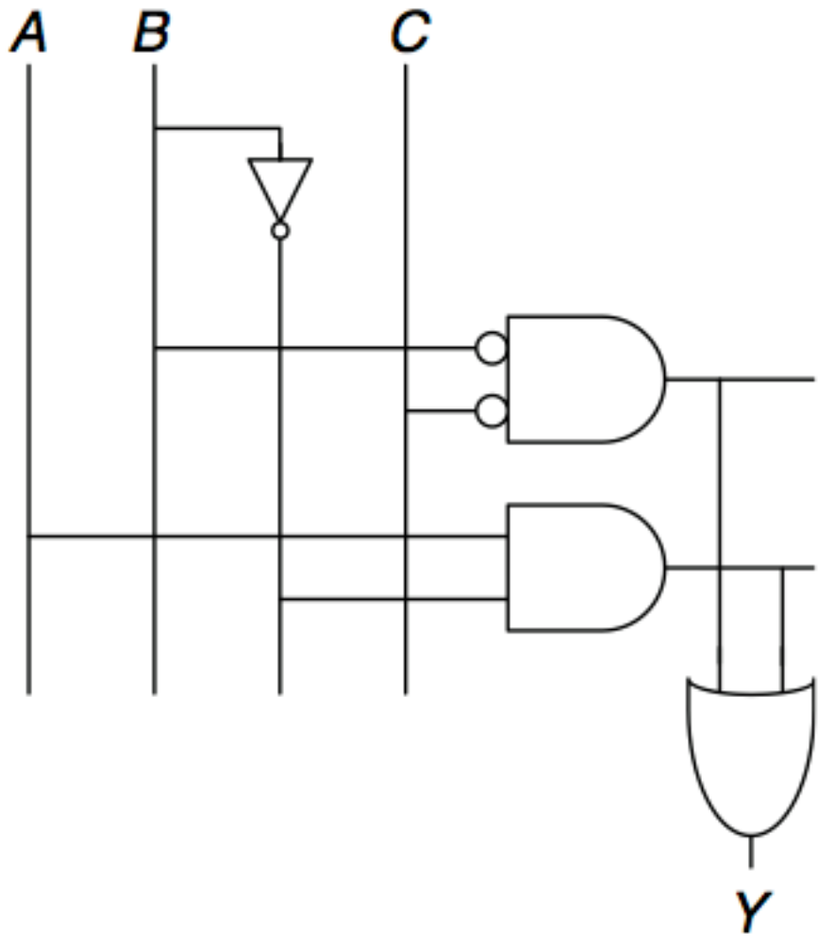


De las ecuaciones al circuito

Otra forma que ahorra un negador es cambiando una compuerta AND por una NOR

$$Y = \overline{B}\,\overline{C} + A\overline{B}$$

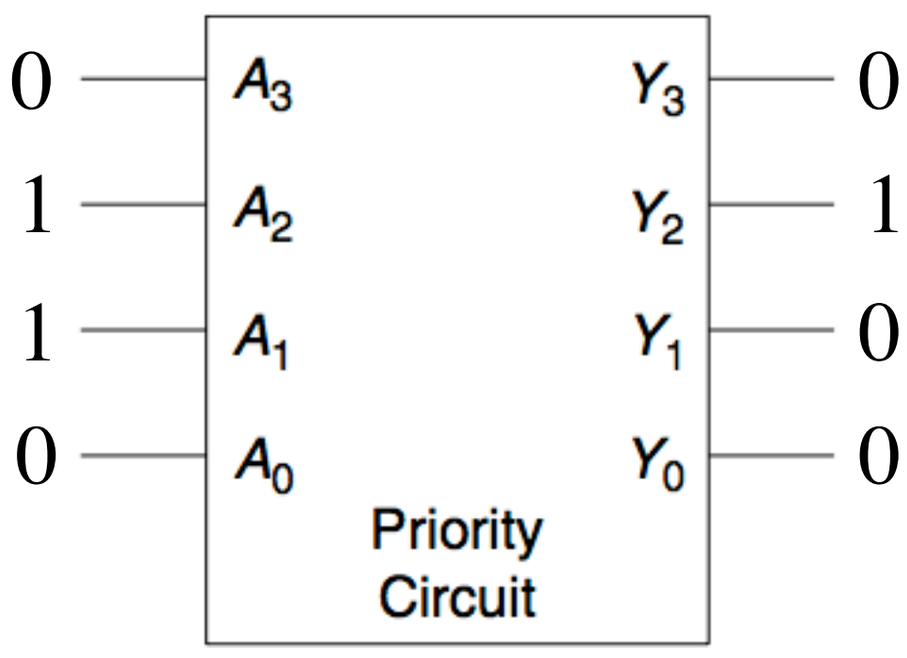


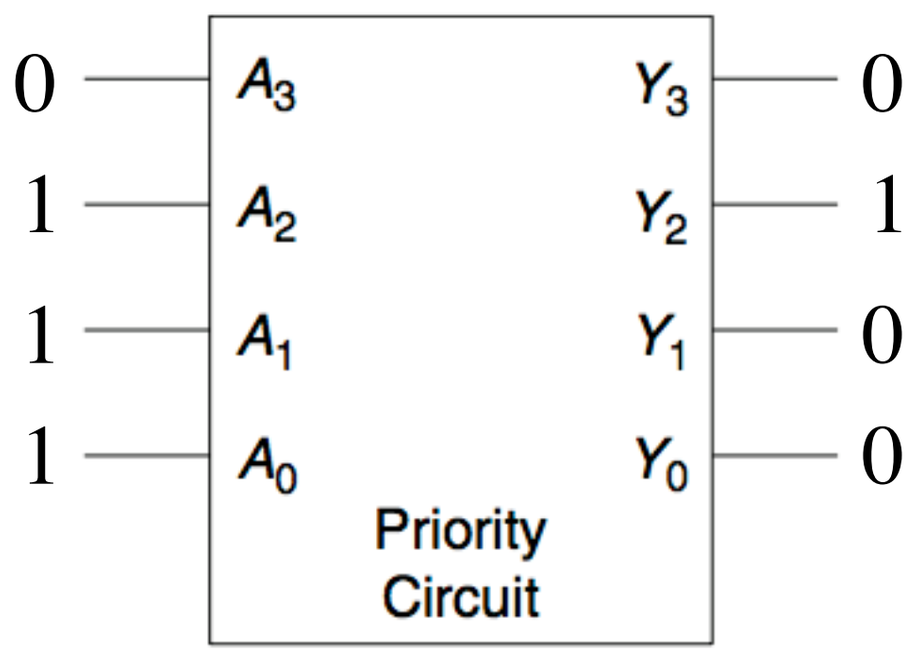


De las ecuaciones al circuito

Circuitos con salidas múltiples (lo veremos con un ejemplo)

El circuito *Prioridad* consiste de n entradas, $A_{n-1}, \ldots, A_0$ y n salidas, $Y_{n-1}, \ldots, Y_0$. Sólo una salida puede estar en 1 a la vez y todas las demás en 0. La salida que se pone en 1 corresponde a la entrada 1 que tiene el mismo subíndice siempre que no haya otra que esté en 1 y tenga un subíndice mayor. Veamos el caso de $n = 4$. En ambos casos la salida Y_2 está en 1 ya que A_2 es la más alta que está en 1. Las salidas de menor prioridad no valen.





De las ecuaciones al circuito

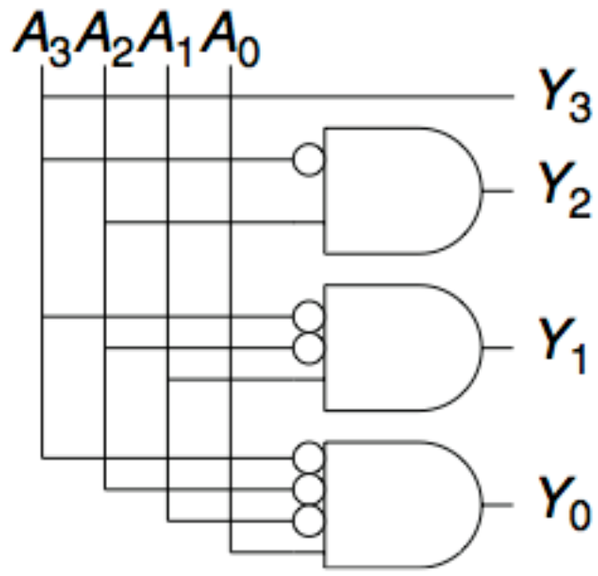
Tabla de verdad del circuito prioridad

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

Tabla de verdad del circuito prioridad utilizando don't cares

A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	X	0	0	1	0
0	1	X	X	0	1	0	0
1	X	X	X	1	0	0	0

Esquemático del circuito prioridad



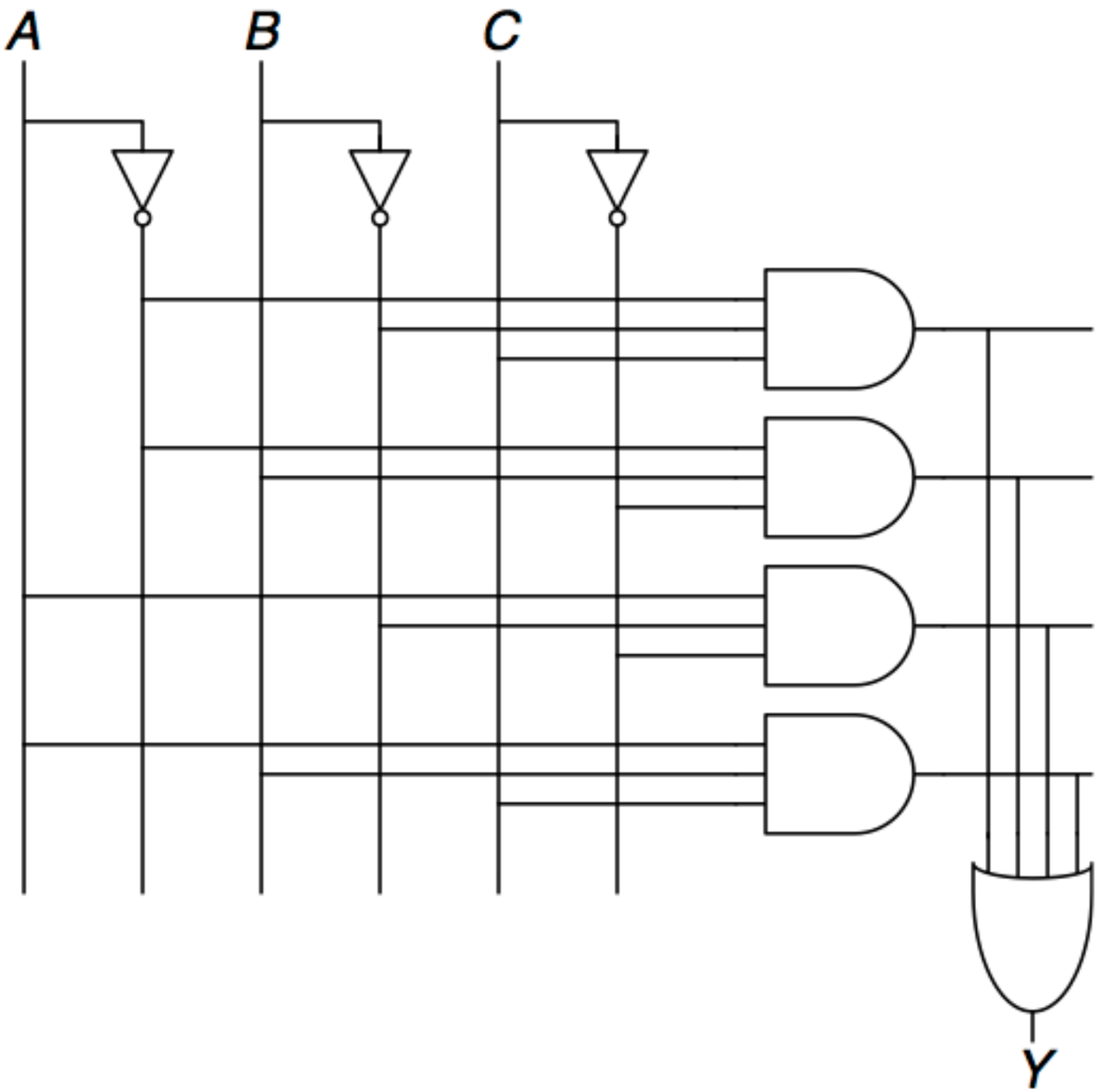
Reducción de funciones binarias

- A la lógica en forma de suma de productos se le llama lógica de dos niveles puesto que consiste de un nivel de compuertas AND y luego un nivel de compuertas OR.
- Algunas funciones lógicas requieren una enorme cantidad de hardware si se implementan en forma canónica de suma de productos o producto de sumas.
- Existen métodos sistemáticos para lograr una expresión mínima, ya sea de suma de productos o

Reducción de funciones binarias

Ejemplo $$Y = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$$

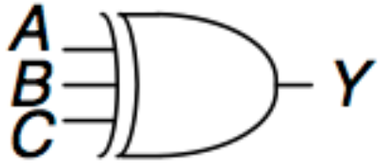
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

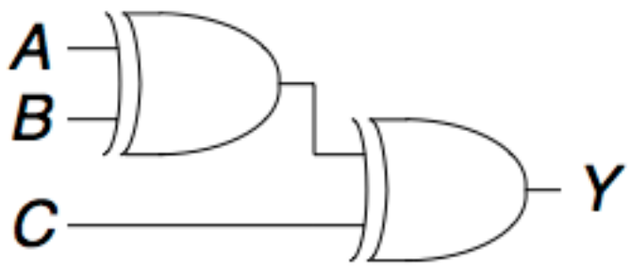


Reducción de funciones binarias

$$Y = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$$

$$Y = A \oplus B \oplus C = (A \oplus B) \oplus C$$





Reducción de funciones binarias

La idea es encontrar alguna expresión equivalente que minimice algún criterio.
Criterios:

- Mínimo número de literales.
- Mínimo número de literales en una expresión de suma de productos (o de producto de sumas)
- Mínimo número de términos en una expresión de suma de productos, siempre y cuando no exista otra expresión con el mismo número de términos y con menos literales.

De acuerdo al tercer criterio $XY + XZ + \bar{X}\bar{Y}$

es menor que $X(Y + Z) + \bar{X}\bar{Y}$

Reducción de funciones binarias

Ejemplos de minimización para la misma función

$$f(x,y,z) = \bar{x}y\bar{z} + \bar{x}\bar{y}\bar{z} + x\bar{y}\bar{z} + \bar{x}yz + xyz + x\bar{y}z$$

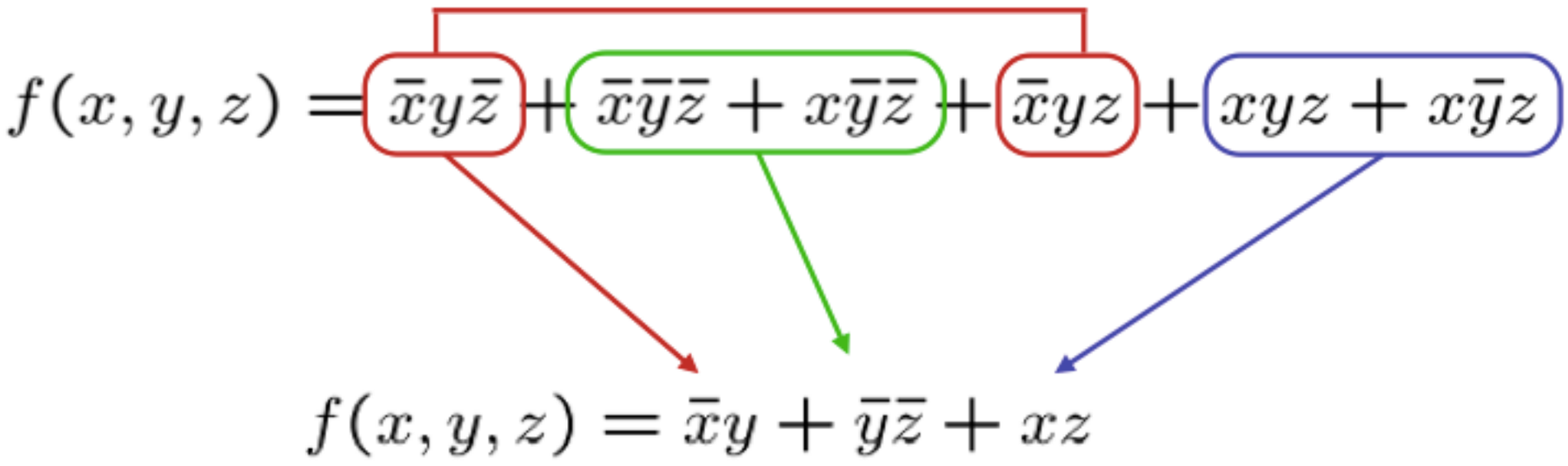
$$f(x,y,z) = \bar{x}\bar{z} + \bar{y}\bar{z} + yz + xz$$

$$f(x,y,z) = \bar{x}y\bar{z} + \bar{x}\bar{y}\bar{z} + x\bar{y}\bar{z} + \bar{x}yz + xyz + x\bar{y}z$$

$$f(x,y,z) = \bar{x}\bar{z} + x\bar{y} + yz$$

Reducción de funciones binarias

Aún otra forma



Claramente, necesitamos una forma más sistemática

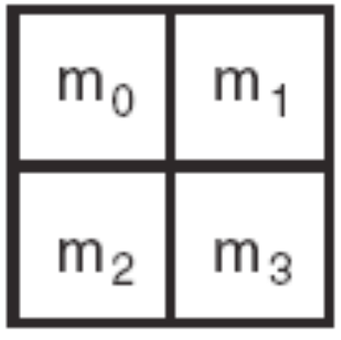
Reducción de funciones binarias

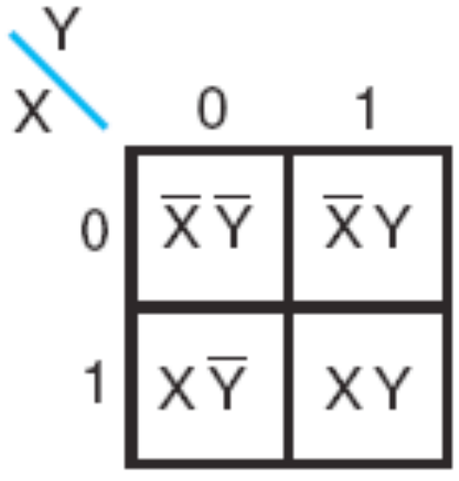
Mapas de Karnaugh

Corresponden a una agrupación de la tabla de verdad en una disposición geométrica que permite aplicar en forma simple y sistemática las reglas de la simplificación

$$Ax + A\bar{x} = A$$

Mapa de dos variables



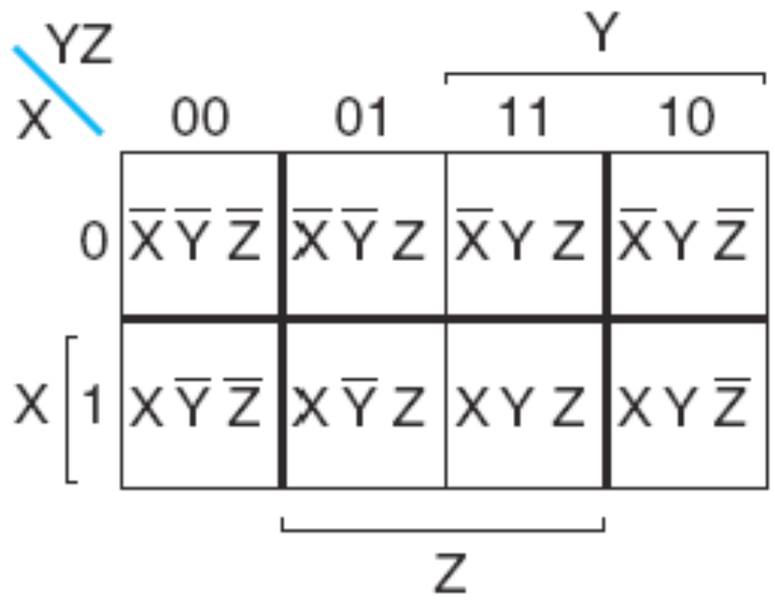


Reducción de funciones binarias

Mapas de Karnaugh

Mapa de tres variables

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

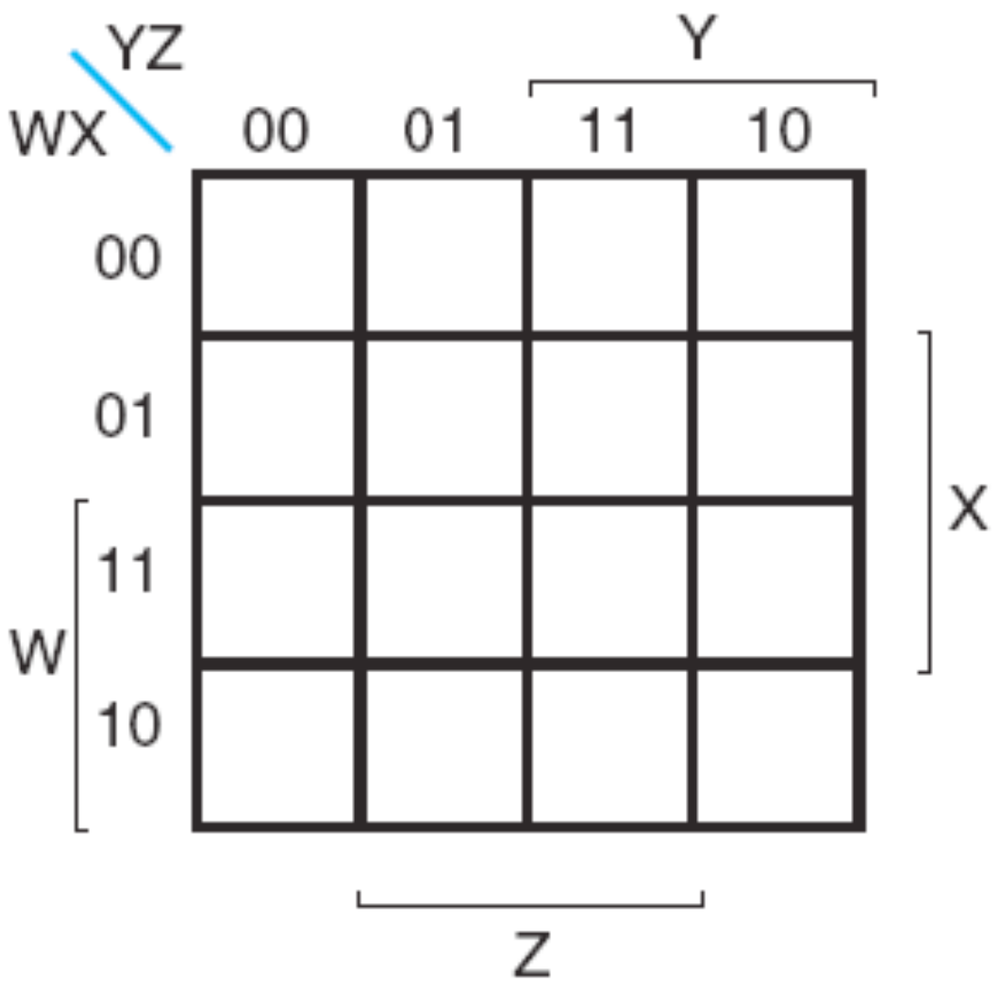


Reducción de funciones binarias

Mapas de Karnaugh

Mapa de cuatro variables

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}



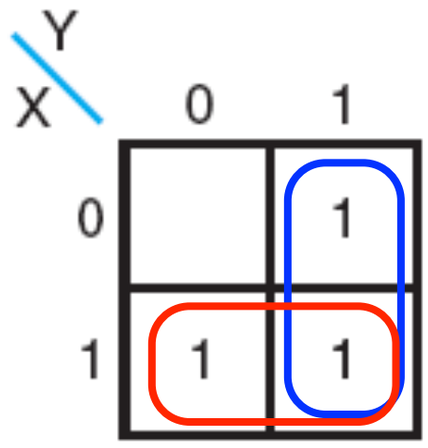
Reducción de funciones binarias

Mapas de Karnaugh

La expresión mínima se logra agrupando las celdas (minitérminos) en subcubos lo más grande posible y con el mínimo número de ellos.

Ejemplo de dos variables

$$F(X,Y) = \bar{X}Y + X\bar{Y} + XY$$



$$F(X,Y) = X + Y$$

Reducción de funciones binarias

Mapas de Karnaugh

Ejemplo: implementación del OR exclusivo

Tabla de verdad

X	Y	F
0	0	0
0	1	1
1	0	1
1	1	0

Función binaria

$$F = X\bar{Y} + \bar{X}Y$$

Mapa de Karnaugh

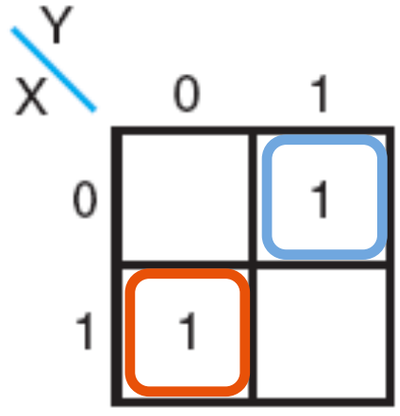
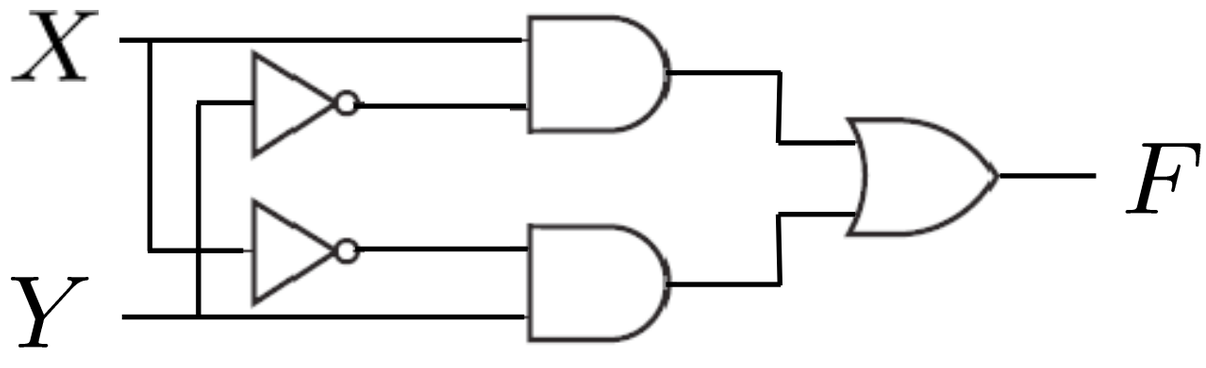


Diagrama lógico



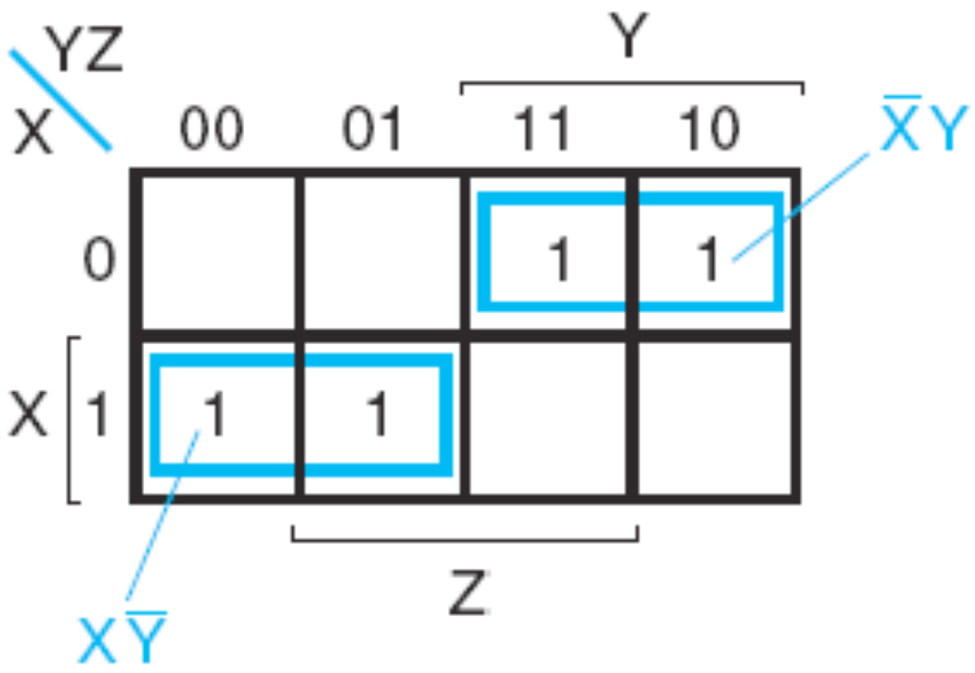
Reducción de funciones binarias

Mapas de Karnaugh

Ejemplo de tres variables

$$F(X,Y,Z) = \sum m(2,3,4,5)$$

$$= \bar{X}Y\bar{Z} + \bar{X}YZ + X\bar{Y}\bar{Z} + X\bar{Y}Z$$



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

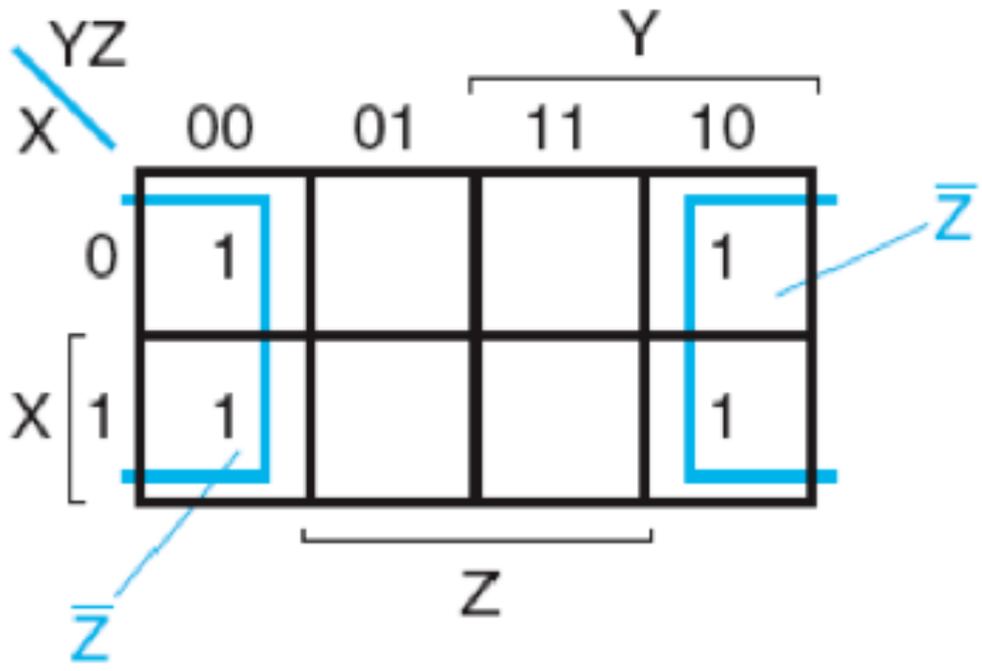
Reducción de funciones binarias

Mapas de Karnaugh

Otro ejemplo de tres variables

$$F(X,Y,Z) = \sum m(0,2,4,6)$$

$$= \bar{X}\bar{Y}\bar{Z} + \bar{X}Y\bar{Z} + X\bar{Y}\bar{Z} + XY\bar{Z}$$



$$F(X,Y,Z) = \bar{Z}$$

Reducción de funciones binarias

Mapas de Karnaugh

Otros ejemplos de tres variables

$$F(X,Y,Z) = \sum m(3,4,6,7)$$

$$F(X,Y,Z) = \sum m(0,2,4,5,6)$$

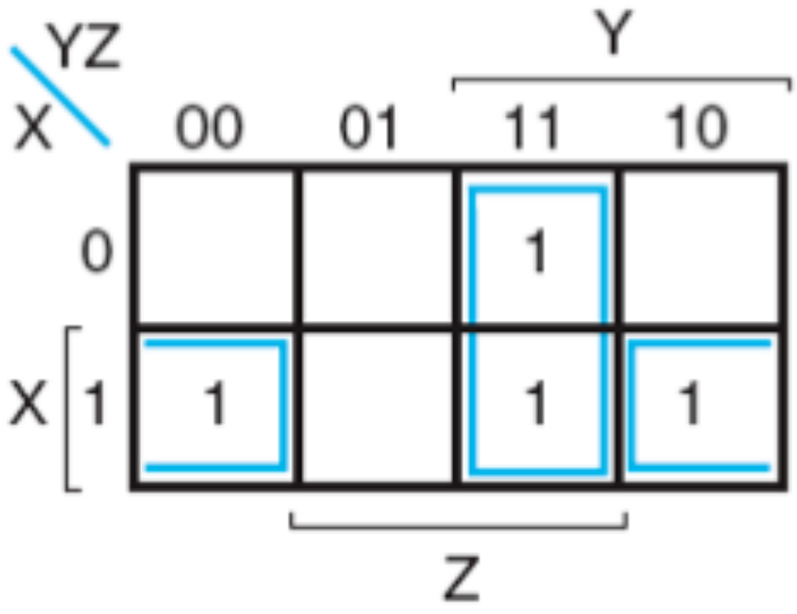
$$F(X,Y,Z) = \sum m(1,3,4,5,6)$$

$$F(X,Y,Z) = \bar{X}Z + \bar{X}Y + X\bar{Y}Z + YZ$$

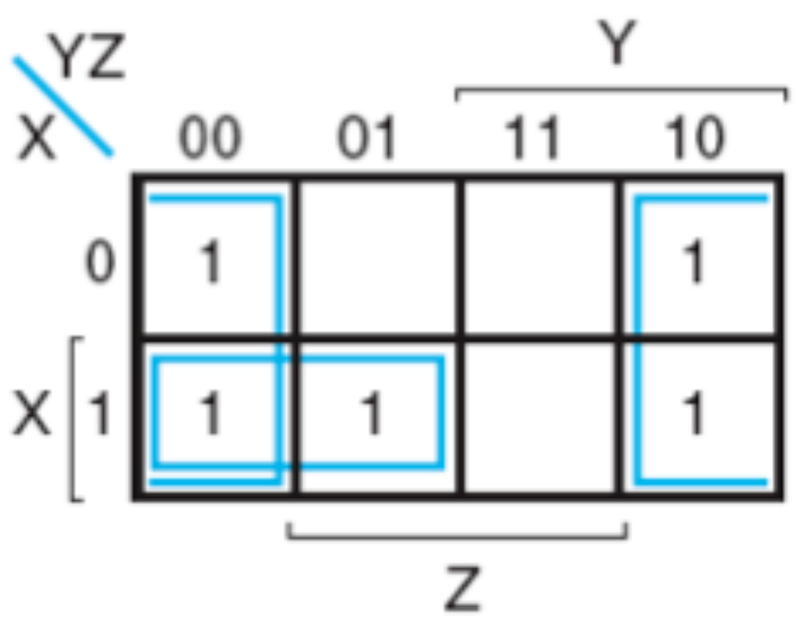
Reducción de funciones binarias

Mapas de Karnaugh

Otro ejemplo de tres variables



(a) $F_1(X, Y, Z) = \Sigma m(3, 4, 6, 7)$
$= YZ + X\overline{Z}$



(b) $F_2(X, Y, Z) = \Sigma m(0, 2, 4, 5, 6)$
$= \overline{Z} + X\overline{Y}$

Reducción de funciones binarias

Mapas de Karnaugh

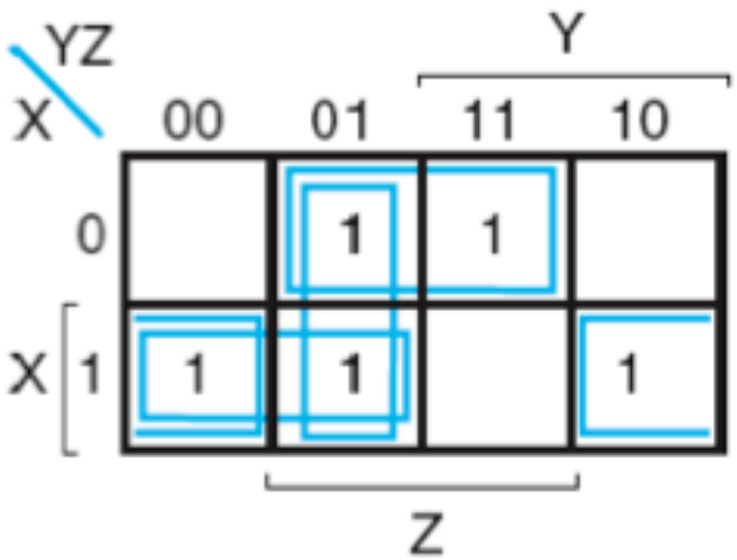


Fig. 2-15 $F(X, Y, Z) = \Sigma m(1, 3, 4, 5, 6)$

$$= \bar{X}Z + X\bar{Z} + X\bar{Y}$$

$$= \bar{X}Z + X\bar{Z} + \bar{Y}Z$$

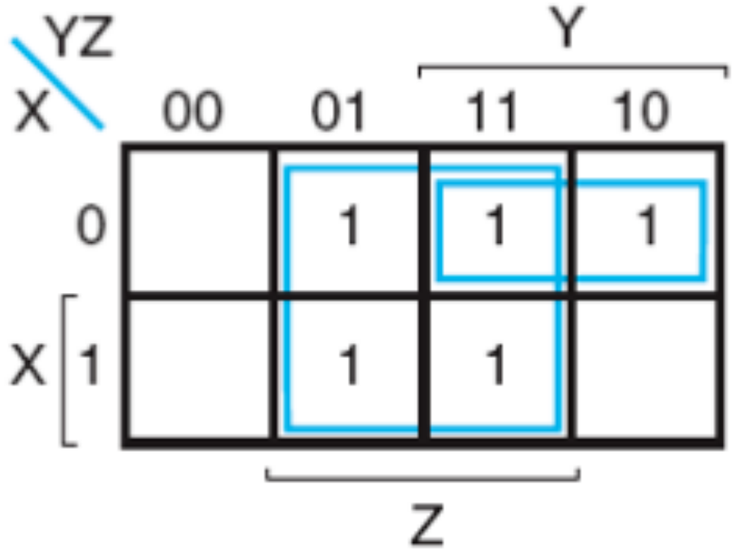


Fig. 2-16 $F(X, Y, Z) = \Sigma m\ (1,2,3,5,7)$

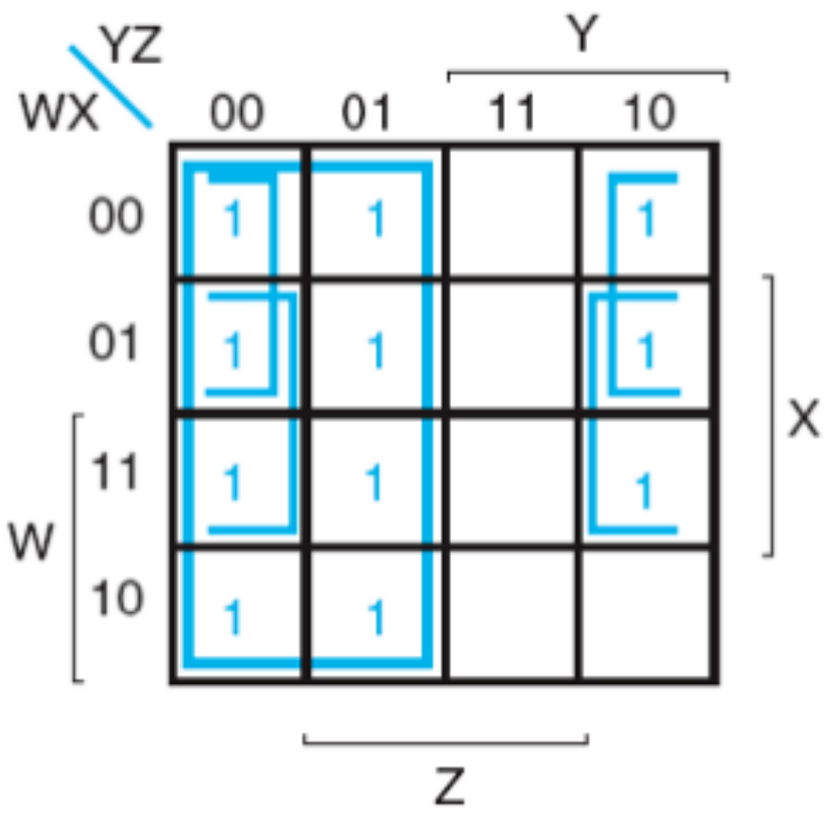
$$= Z + \bar{X}Y$$

Reducción de funciones binarias

Mapas de Karnaugh

Ejemplo de cuatro variables

$$F(W, X, Y, Z) = \sum m(0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$$



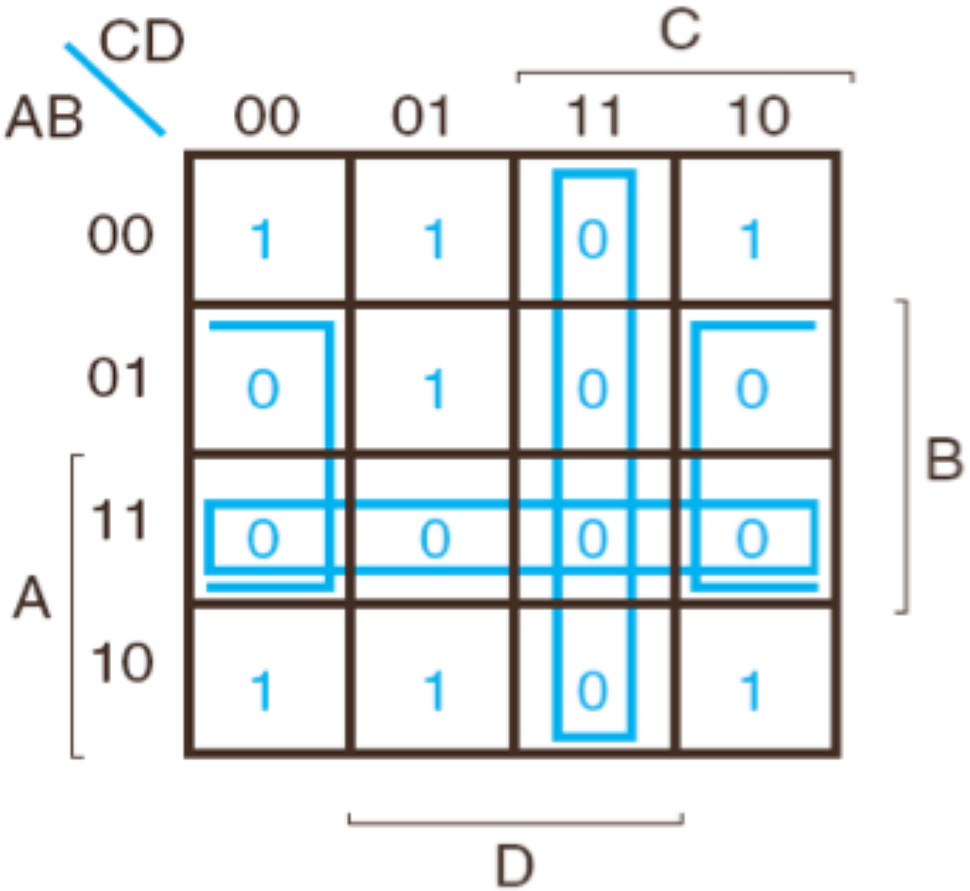
$$F(W, X, Y, Z) = \bar{Y} + \bar{W}\bar{Z} + X\bar{Z}$$

Reducción de funciones binarias

Mapas de Karnaugh

Minimización utilizando el producto de sumas

$$F(A,B,C,D) = \sum m(0,1,2,5,8,9,10)$$



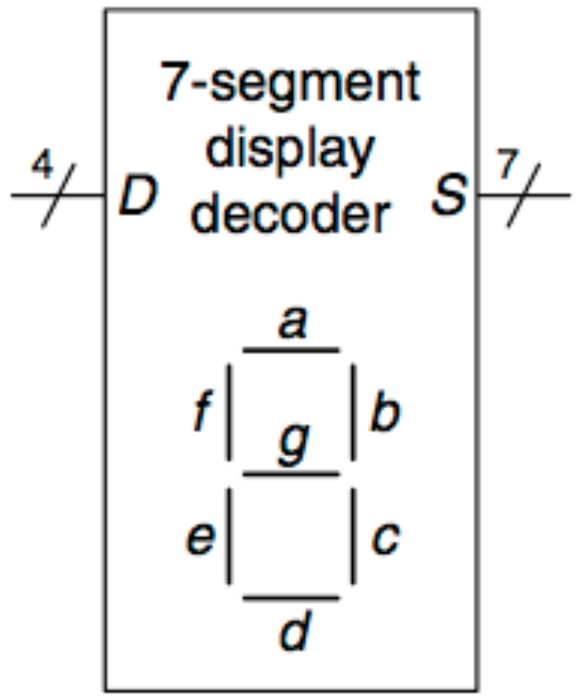
$$\bar{F} = AB + CD + B\bar{D}$$

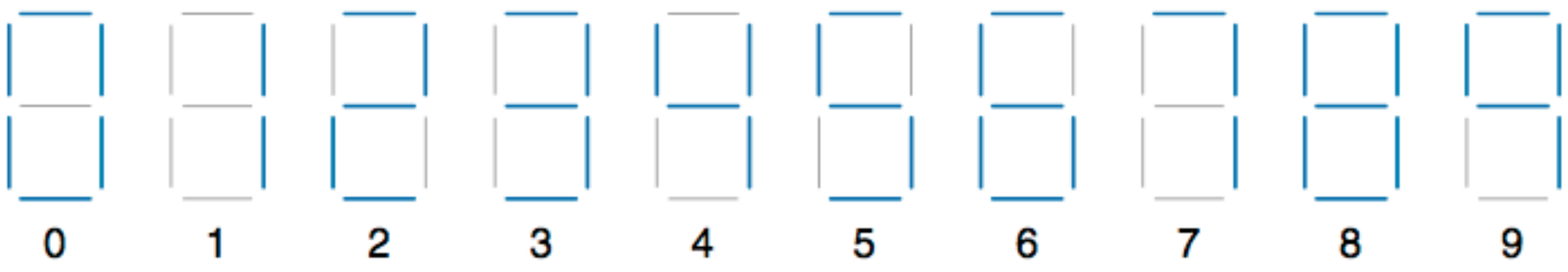
$$F = (\bar{A} + \bar{B})(\bar{C} + \bar{D})(\bar{B} + D)$$

Reducción de funciones binarias

Mapas de Karnaugh

Ejemplo de aplicación: Diseño de un circuito decodificador de BCD a 7 segmentos





Reducción de funciones binarias

Mapas de Karnaugh

Tabla de verdad

$D_{3:0}$	S_a	S_b	S_c	S_d	S_e	S_f	S_g
0000	1	1	1	1	1	1	0
0001	0	1	1	0	0	0	0
0010	1	1	0	1	1	0	1
0011	1	1	1	1	0	0	1
0100	0	1	1	0	0	1	1
0101	1	0	1	1	0	1	1
0110	1	0	1	1	1	1	1
0111	1	1	1	0	0	0	0
1000	1	1	1	1	1	1	1
1001	1	1	1	0	0	1	1
others	0	0	0	0	0	0	0

Reducción de funciones binarias

Mapas de Karnaugh

Decodificador de BCD a 7 segmentos

$D_{3:0}$	S_a	S_b
0000	1	1
0001	0	1
0010	1	1
0011	1	1
0100	0	1
0101	1	0
0110	1	0
0111	1	1
1000	1	1
1001	1	1
others	0	0

S_a

$D_{1:0}$ \ $D_{3:2}$	00	01	11	10
00	1	0	0	1
01	0	1	0	1
11	1	1	0	0
10	1	1	0	0

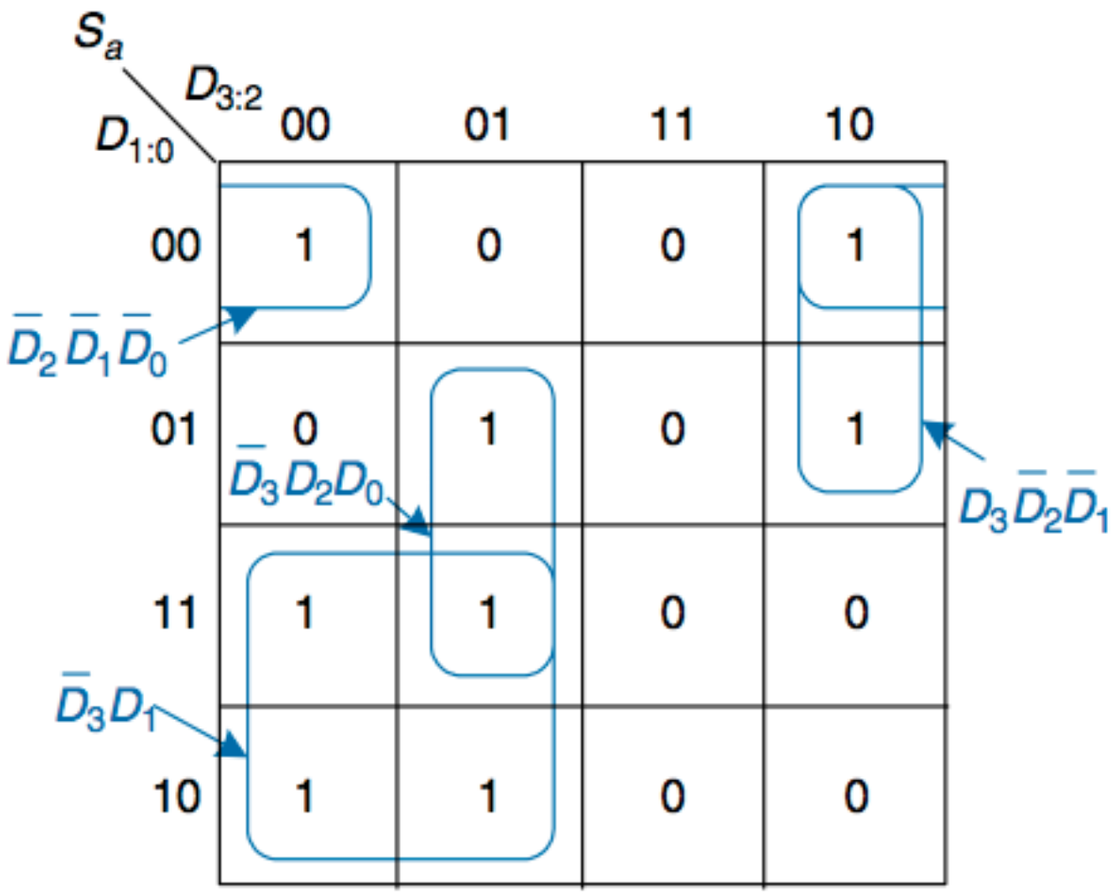
S_b

$D_{1:0}$ \ $D_{3:2}$	00	01	11	10
00	1	1	0	1
01	1	0	0	1
11	1	1	0	0
10	1	0	0	0

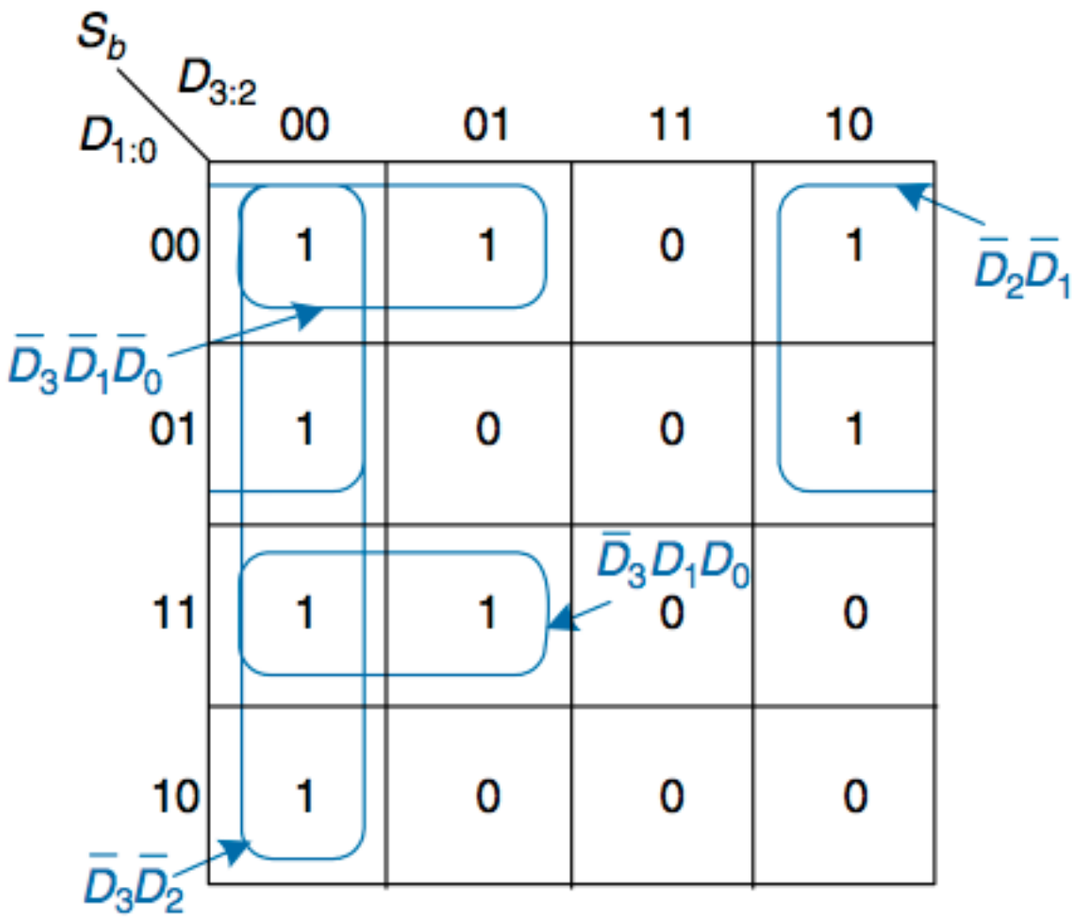
Reducción de funciones binarias

Mapas de Karnaugh

Decodificador de BCD a 7 segmentos



$$S_a = \bar{D}_3D_1 + \bar{D}_3D_2D_0 + D_3\bar{D}_2\bar{D}_1 + \bar{D}_2\bar{D}_1\bar{D}_0$$

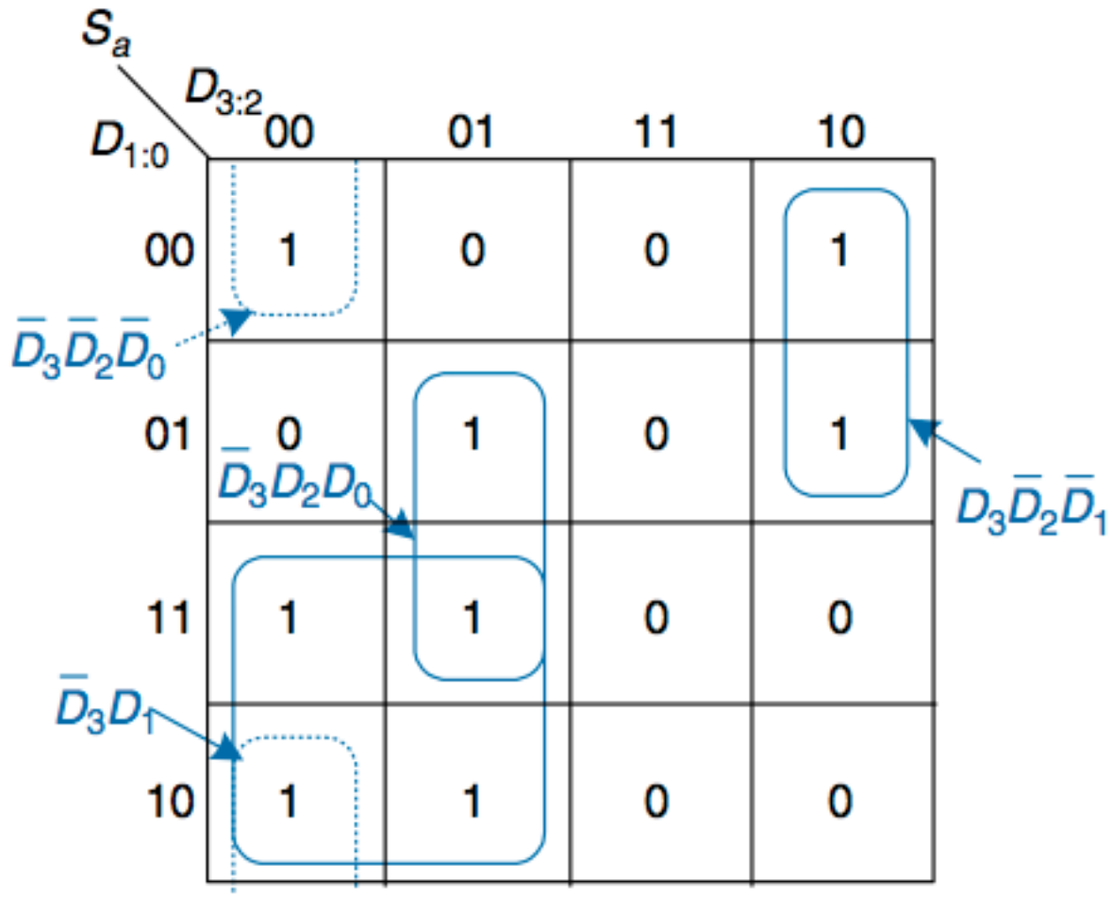


$$S_b = \bar{D}_3\bar{D}_2 + \bar{D}_2\bar{D}_1 + \bar{D}_3D_1D_0 + \bar{D}_3\bar{D}_1\bar{D}_0$$

Reducción de funciones binarias

Mapas de Karnaugh

Decodificador de BCD a 7 segmentos

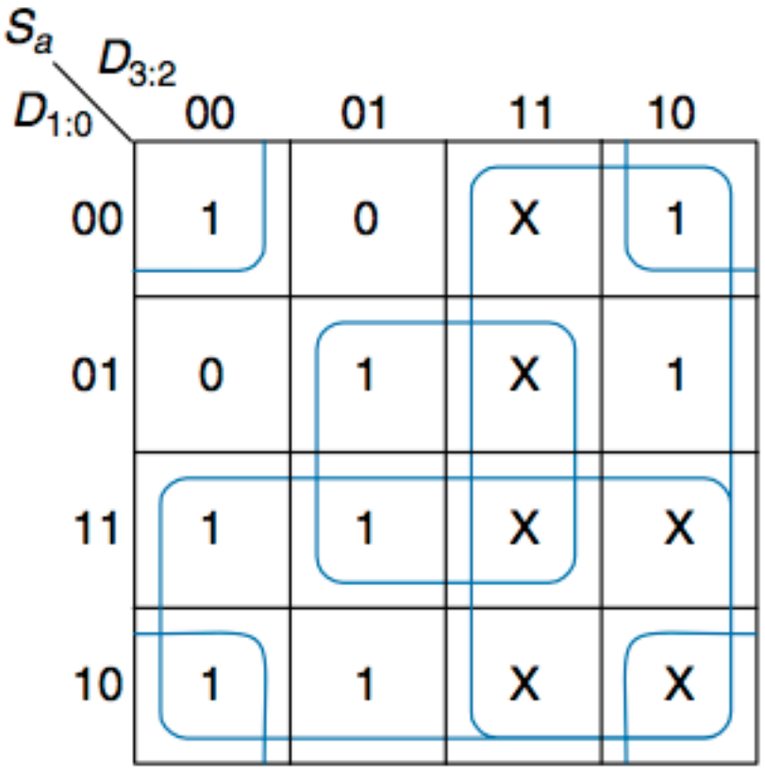


$$S_a = \bar{D}_3 D_1 + \bar{D}_3 D_2 D_0 + D_3 \bar{D}_2 \bar{D}_1 + \bar{D}_3 \bar{D}_2 \bar{D}_0$$

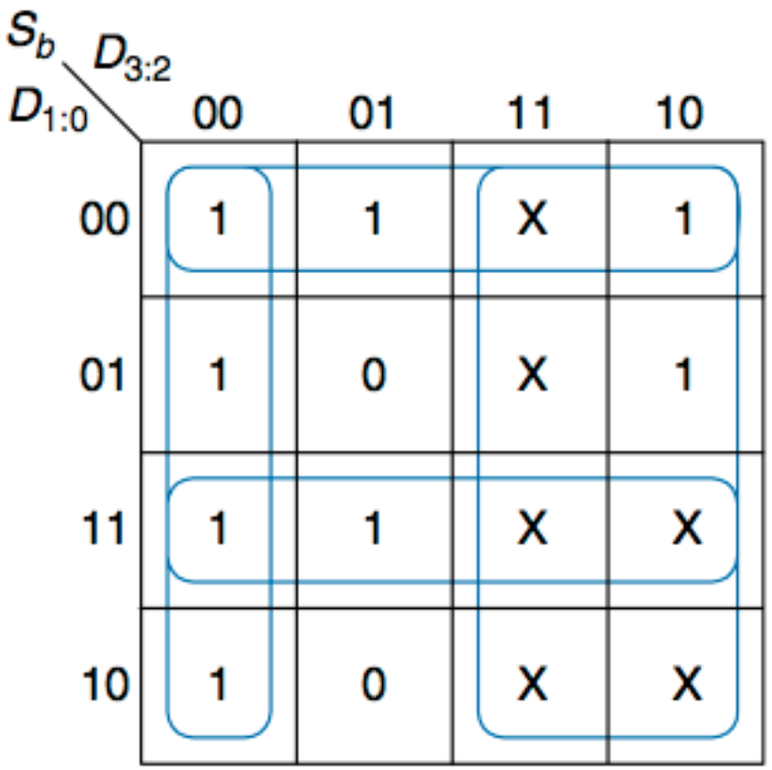
Reducción de funciones binarias

Mapas de Karnaugh

Decodificador de BCD a 7 segmentos utilizando ***don't cares***



$$S_a = D_1 + D_3 + D_2D_0 + \bar{D}_2\bar{D}_0$$



$$S_b = D_3 + \bar{D}_3\bar{D}_2 + D_1D_0 + \bar{D}_1\bar{D}_0$$

Bloques combinacionales

- Los circuitos combinacionales generalmente se agrupan en bloques mayores para formar sistemas más complejos
- Esta es una aplicación del principio de abstracción, esconder los detalles a nivel de compuertas, para enfatizar la función del bloque
- Hemos visto dos bloques: circuito de prioridad y decodificador de BCD a 7 segmentos.
- En esta sección introduciremos dos bloques más que tienen un amplio uso.

Bloques combinacionales

Multiplexor 2:1

Símbolo

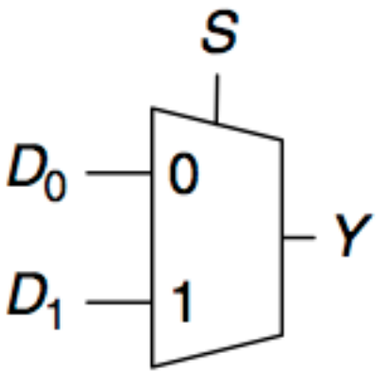
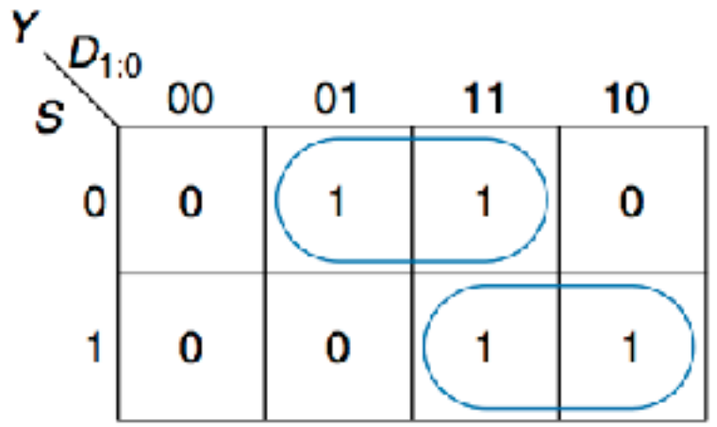


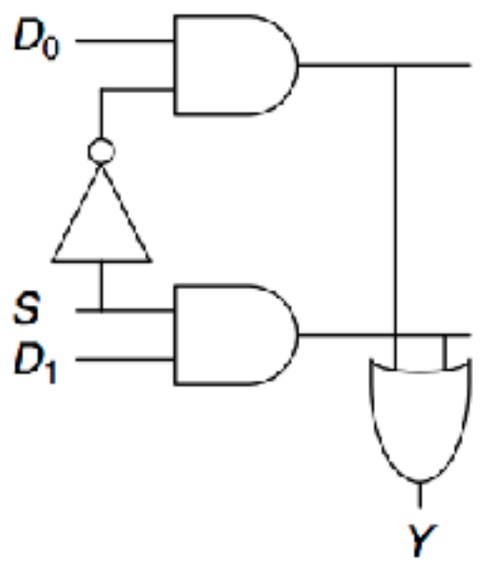
Tabla de verdad

S	D_1	D_0	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

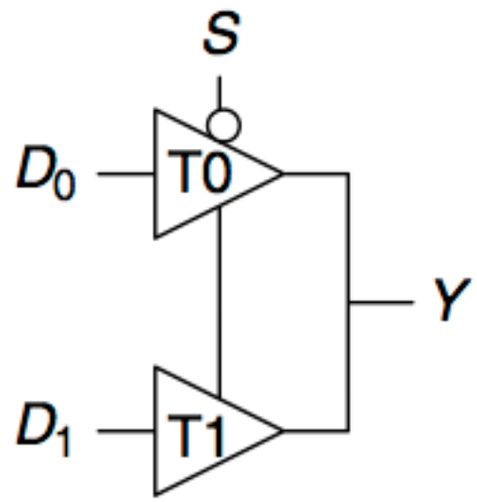
Implementación



$$Y = D_0\bar{S} + D_1S$$



Implementación utilizando compuertas con control de tercer estado

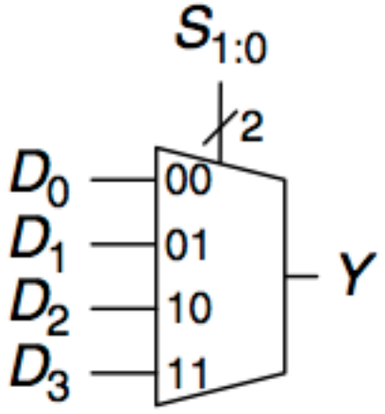


$$Y = D_0\bar{S} + D_1S$$

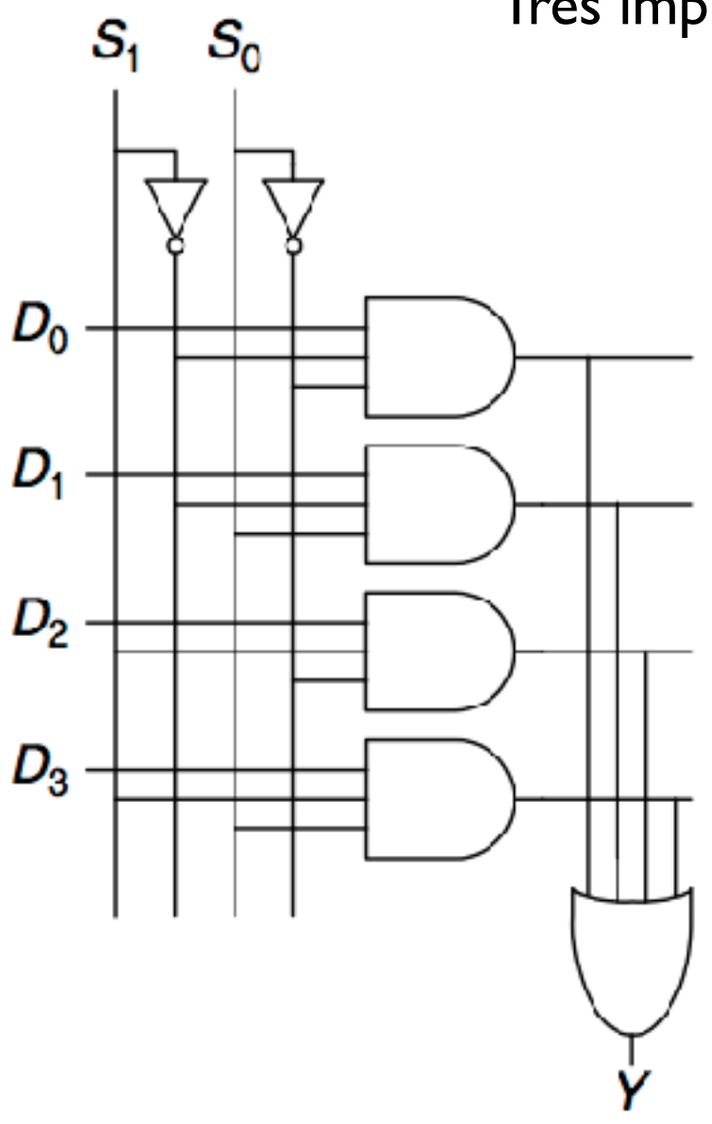
Bloques combinacionales

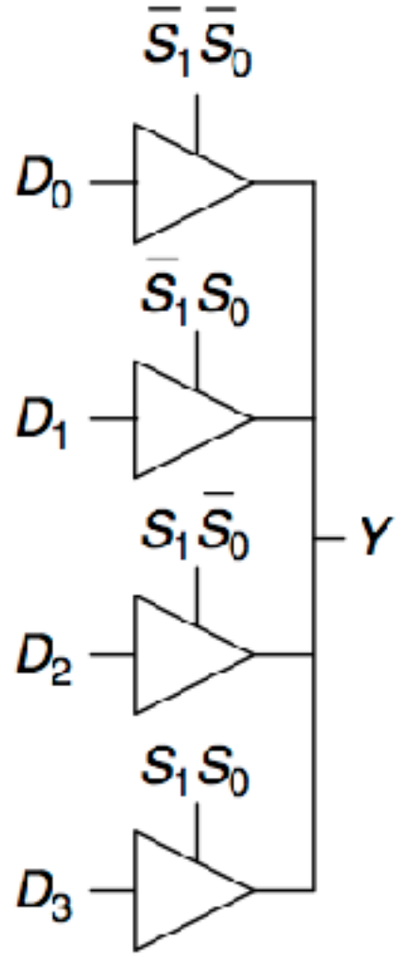
Multiplexor 4:1

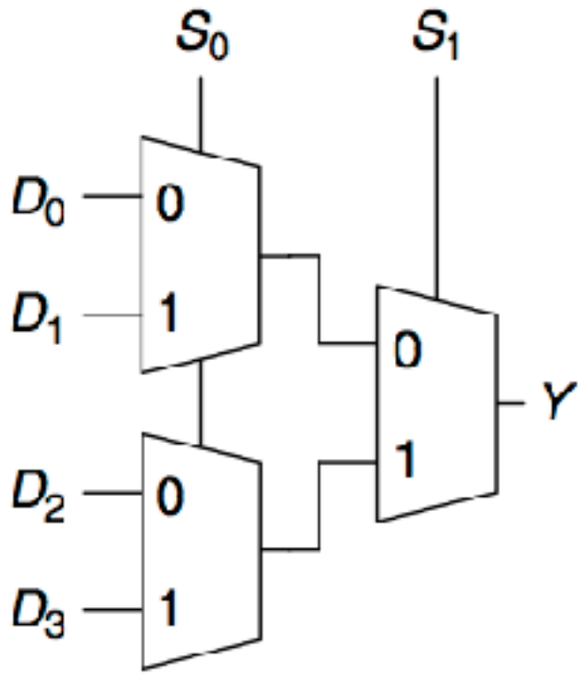
Símbolo



Tres implementaciones diferentes







Bloques combinacionales

Multiplexores

- Multiplexores más ámplios que 4:1, por ejemplo 8:1 o 16:1 se pueen construír expandiendo el método visto para desarrollar el multiplexor 4:1.
- En general un multiplexor N:1 necesita $\log_2 N$ líneas de selección.
- La mejor implementación depende de la tecnología utilizada.

Bloques combinacionales

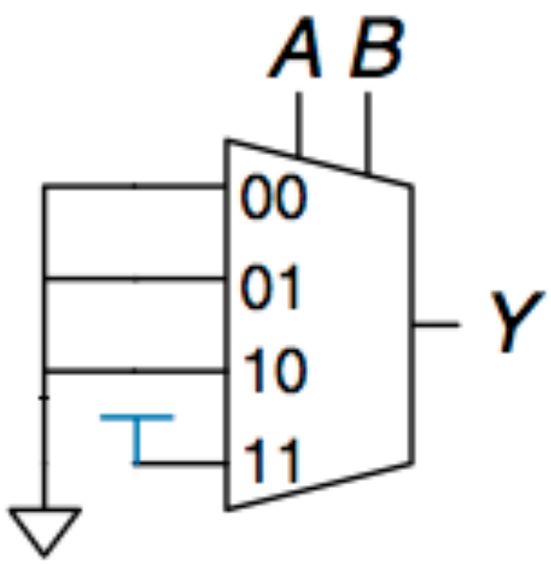
Multiplexores

Circuitos combinacionales basados en multiplexores

- Los multiplexores pueden ser utilizados como *lookup tables* para implementar funciones lógicas.
- Ejemplo: uso de un multiplexor 4:1 para implementar la función AND

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

$Y = AB$



Bloques combinacionales

Multiplexores

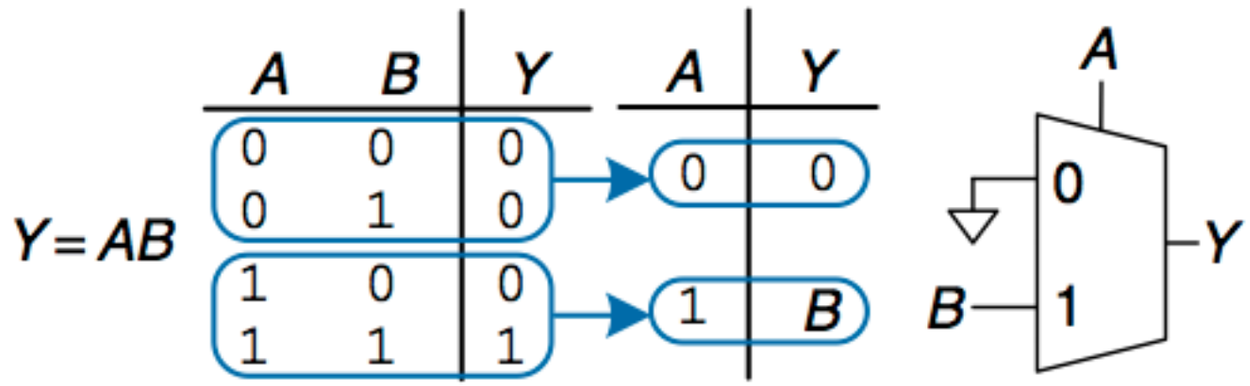
Circuitos combinacionales basados en multiplexores

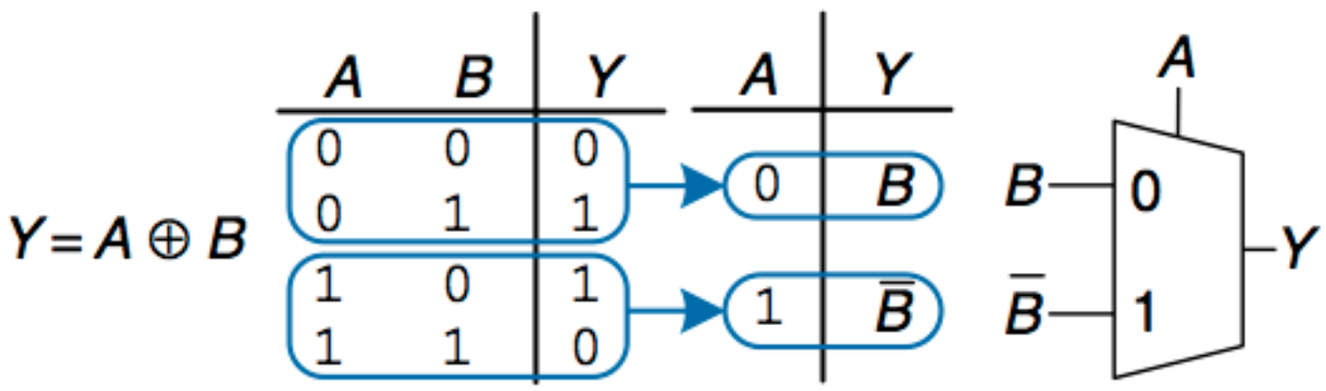
- En general un multiplexor de 2^N entradas puede programarse para implementar cualquier función lógica de N entradas, aplicando 0s y 1s en las entradas apropiadas.
- Cambiando los valores 0s y 1s de las entradas es posible reprogramar el multiplexor para realizar una función distinta.
- Con un poco de intuición, es posible utilizar un multiplexor de menos entradas, $N/2$ por ejemplo, para implementar una función de N entradas.

Bloques combinacionales

Multiplexores

Ejemplos para ilustrar el último punto de la transparencia anterior





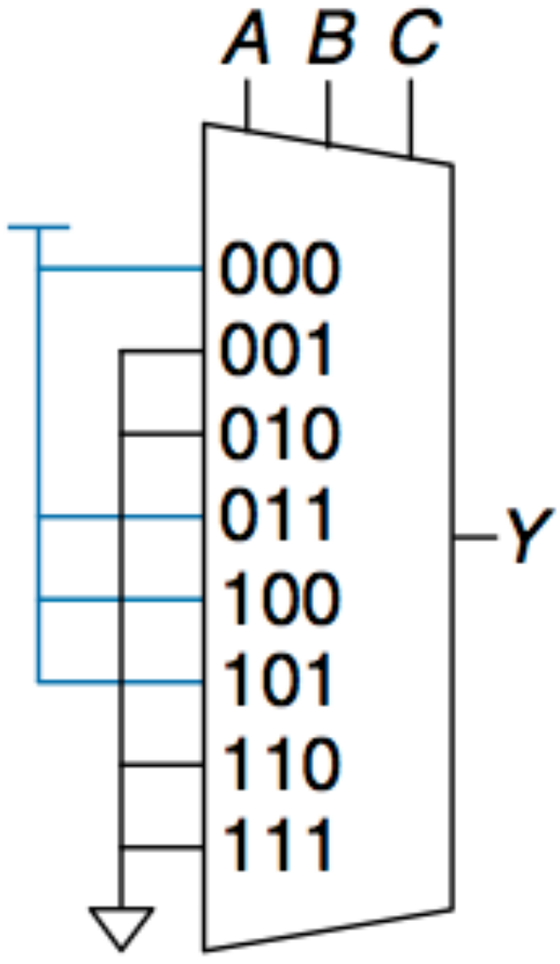
Bloques combinacionales

Multiplexores

Ejemplo: Implemetar la función $Y = A\bar{B} + \bar{B}\bar{C} + \bar{A}BC$ utilizando un multiplexor 8:1

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

$Y = A\overline{B} + \overline{B}\overline{C} + \overline{A}BC$



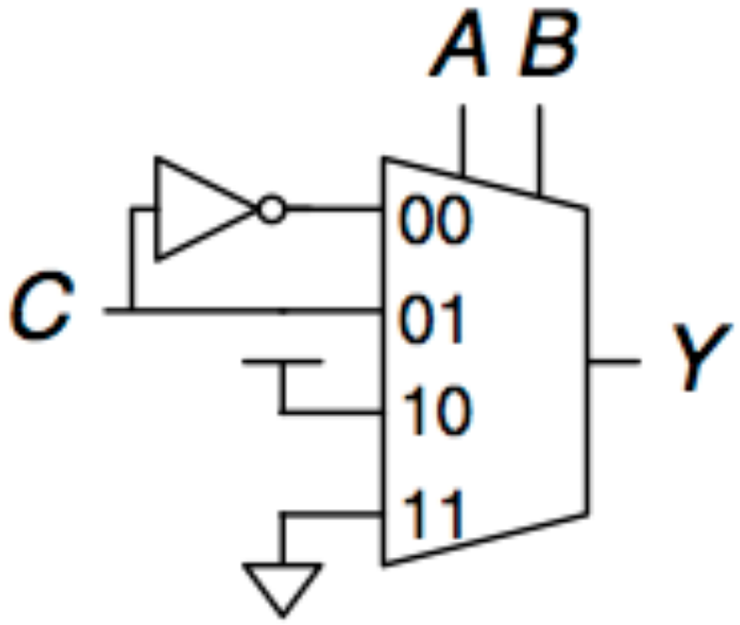
Bloques combinacionales

Multiplexores

Es posible implentar la función $Y = A\bar{B} + \bar{B}\bar{C} + \bar{A}BC$ utilizando un multiplexor 4:1 ?

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

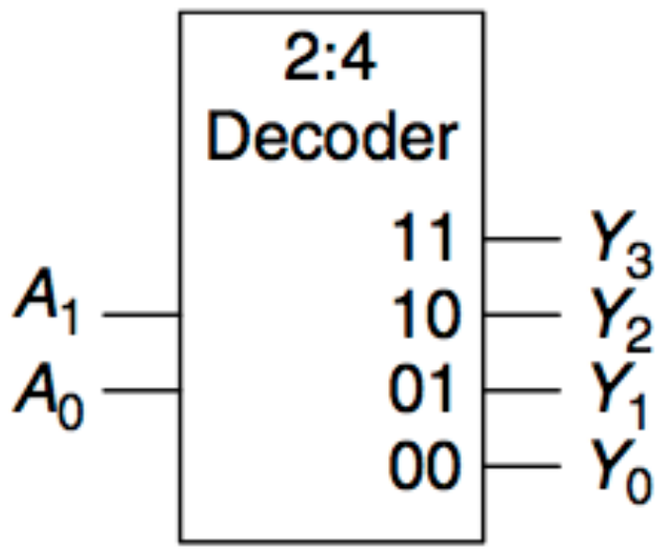
A	B	Y
0	0	$\bar{C}$
0	1	C
1	0	1
1	1	0



Bloques combinacionales

Decodificadores

- Un decodificador tiene N entradas y 2^N salidas
- Sólo una salida puede valer 1 y depende de la combinación de las entradas.
- La siguiente figura muestra un decodificador de 2:4 y su tabla de funcionamiento

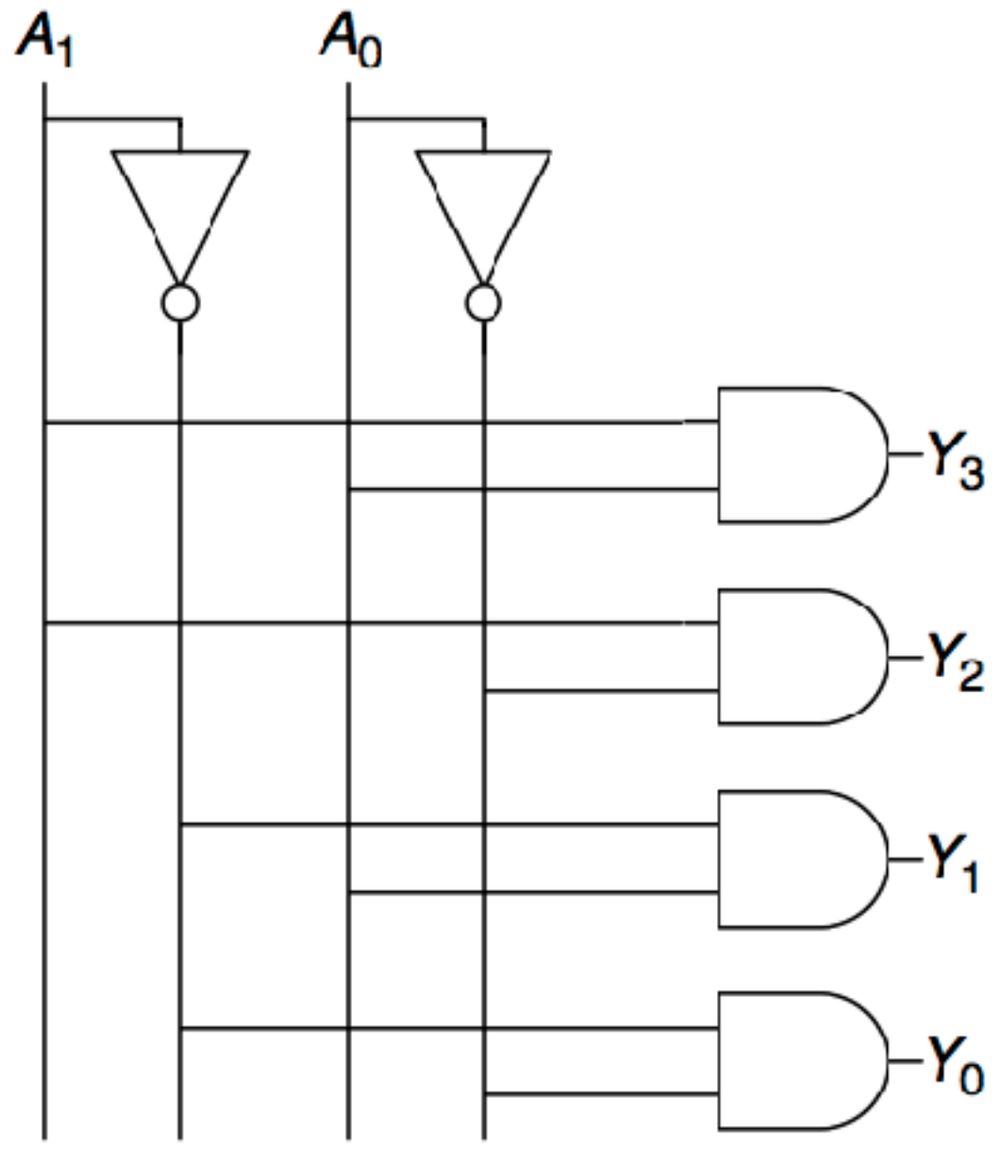


A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Bloques combinacionales

Decodificadores

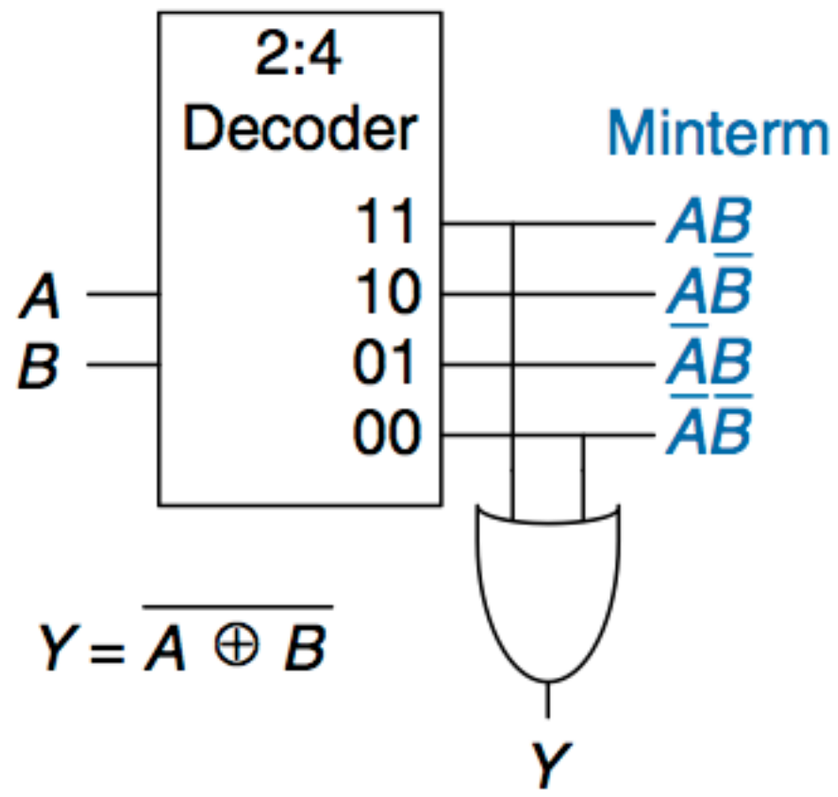
Implementación de un deodificador



Bloques combinacionales

Decodificadores

- Los decodificadores pueden ser combinados con compuertas OR para implementar funciones lógicas.
- La figura muestra la implementación de la función XNOR utilizando un decodificador 2:4 y una

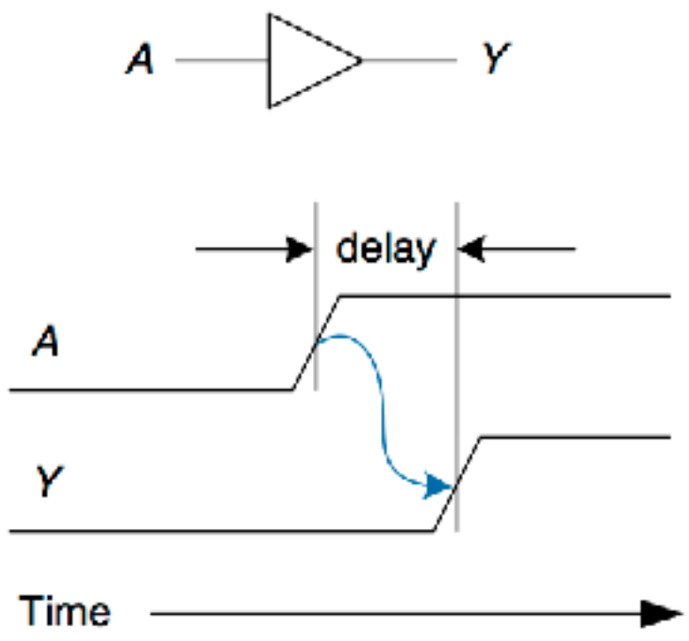


Temporización

- Hasta ahora nos hemos preocupado principalmente de diseñar circuitos que funcionen, idealmente utilizando el menor número de compuertas posible.
- Sin embargo, cualquier diseñador experimentado puede asegurar que uno de los temas más desafiantes en el diseño de circuitos digitales es la temporización (hacer que el circuito funcione rápido)
- La salida de un circuito toma un tiempo en responder al cambio en una entrada.

Temporización

- La figura muestra el retardo entre un cambio en la entrada y la salida correspondiente para un *buffer*.
- La figura recibe el nombre de diagrama de tiempos o diagrama temporal. Muestra la respuesta transitoria.
- La transición de *LOW* a *HIGH* se llama flanco de subida.
- La transición de *HIGH* a *LOW* se llama flanco de bajada.
- La flecha azul indica que el flanco de subida de *Y* es causado por el flanco de subida de *A*.
- El retardo se mide desde el punto correspondiente al 50% de la señal *A* hasta el punto de 50% de la señal *Y*.



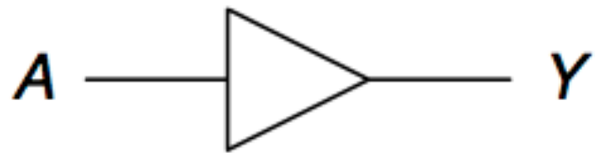
Temporización

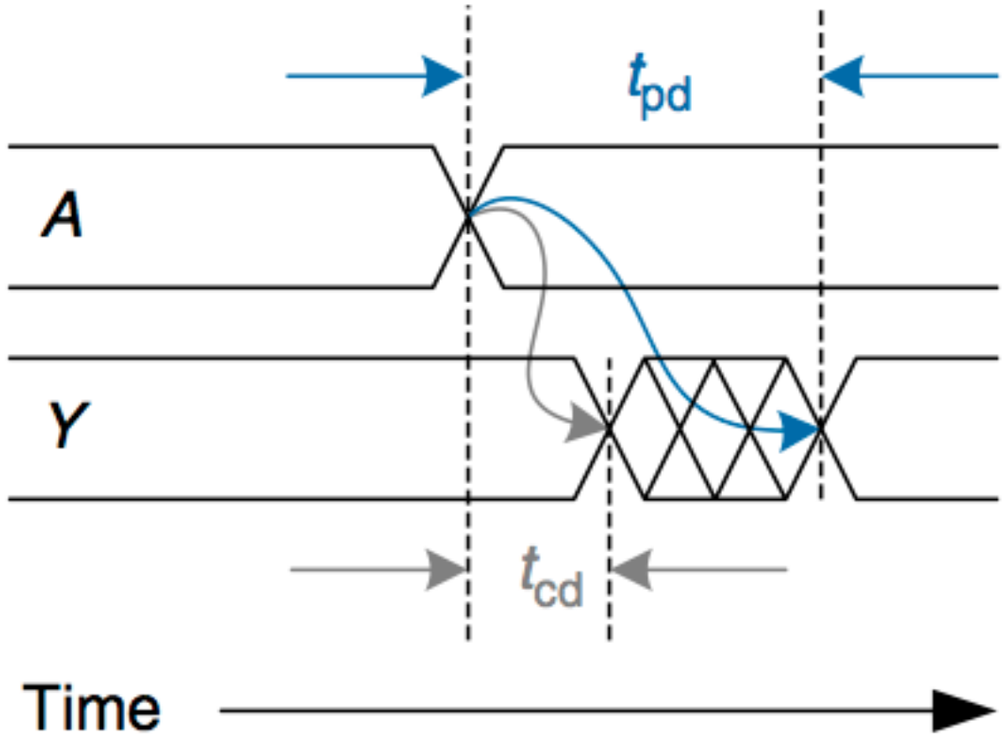
Retardo de propagación y de contaminación

- La lógica combinacional se caracteriza por su retardo de propagación y su retardo de contaminación.
- El retardo de propagación, t_{pd}, es el máximo tiempo desde que una entrada cambia hasta que la señal de salida alcanza su valor final.
- El retardo de contaminación, t_{cd}, es el mínimo tiempo desde que una entrada cambia hasta que cualquiera de las salidas empieza a cambiar su valor.

Temporización

Retardo de propagación y de contaminación para un *buffer*.





Temporización

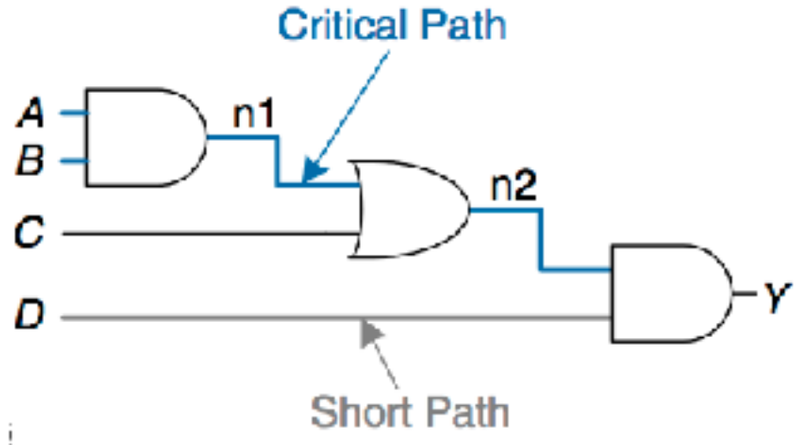
- Las causas subyacentes de los retardos en los circuitos incluyen por una parte el tiempo requerido para cargar las capacidades dentro del circuito y la velocidad de la luz.
- t_{pd} y t_{cd} pueden ser diferentes por diversas razones, incluyendo
  - diferentes tiempos de retardo de subida y de bajada
  - entradas y salidas múltiples, alguna de las cuales son más rápidas que otras
  - Circuitos más lentos cuando suben su temperatura y más rápidos cuando se enfrían.

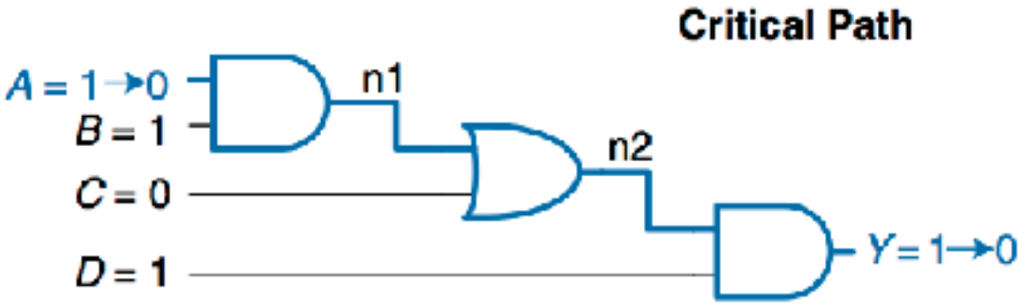
Temporización

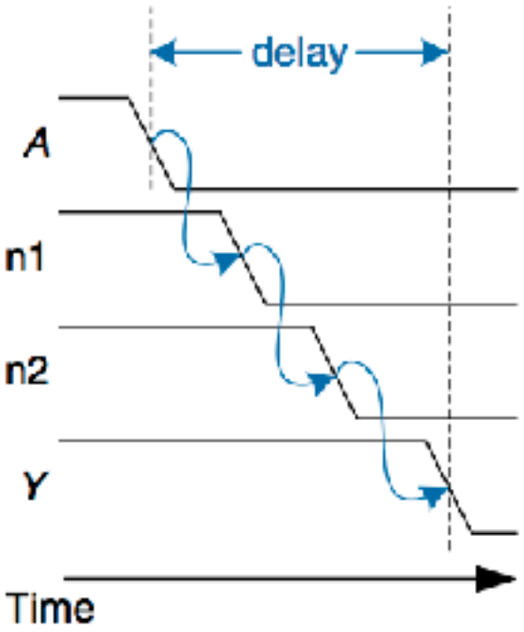
- El cálculo de t_{pd} y de t_{cd} requiere bajar a los menores niveles de abstracción, más allá de lo que veremos en este curso.
- En general los fabricantes proveen hojas de datos especificando los retardos para cada compuerta.
- Con estos datos se puede estudiar bastante bien el retardo de nuestros circuitos

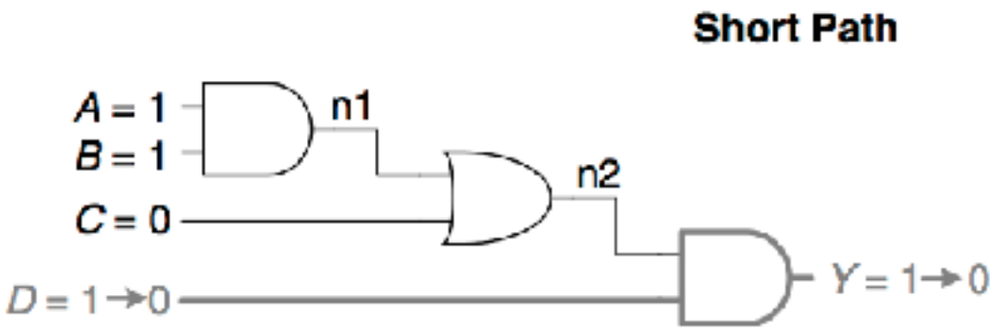
Temporización

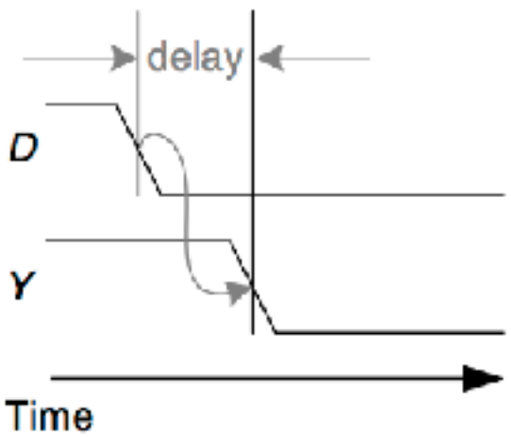
Camino más corto y camino crítico







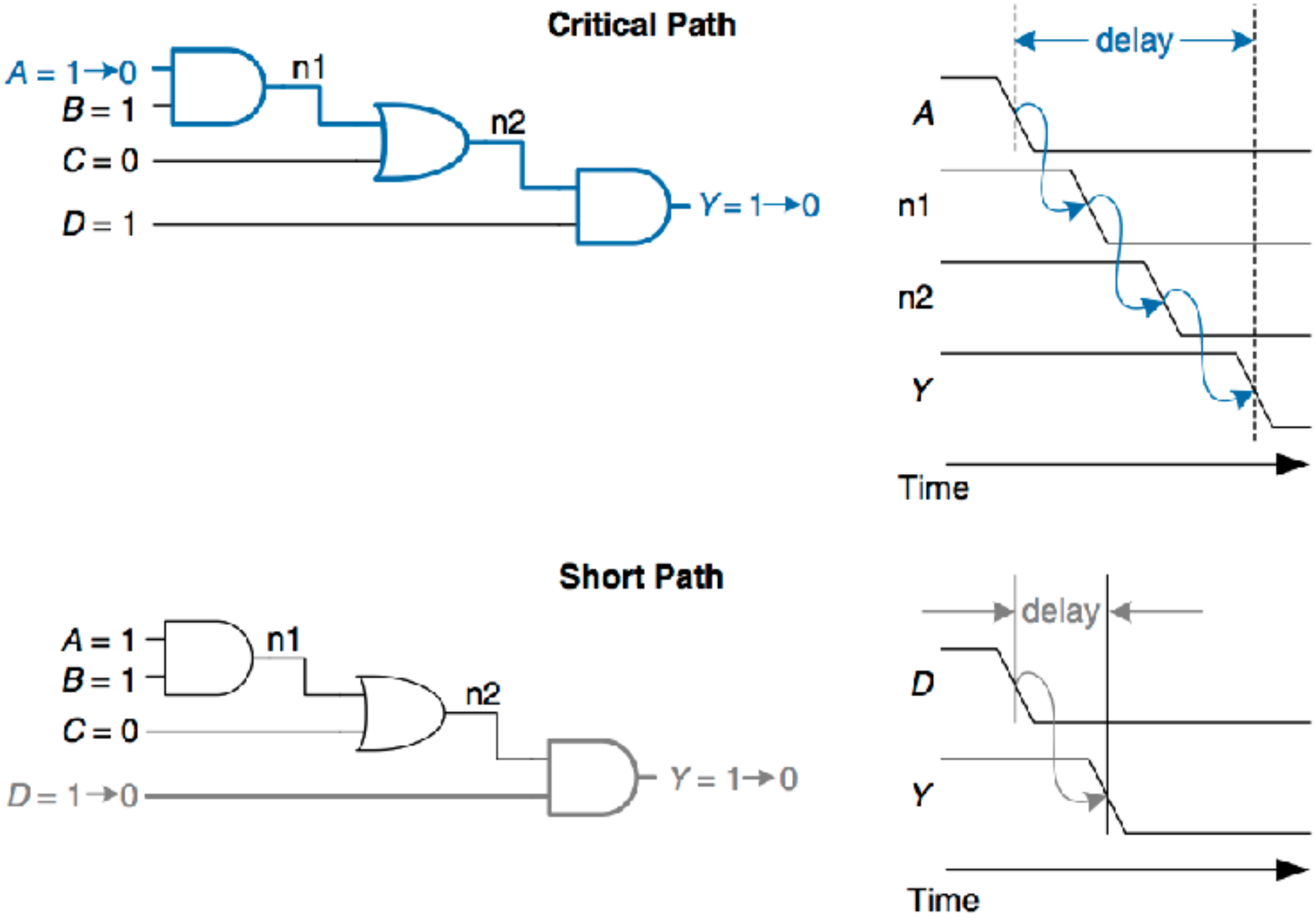




Temporización

- El retardo de propagación de un circuito combinacional es la suma del retardo de propagación a través de cada elemento en el camino crítico
- El retardo de contaminación es la suma de los retardos de contaminación a través de cada elemento en el camino más corto.

Temporización



$$t_{pd} = 2t_{pd\_\mathrm{AND}} + t_{pd\_\mathrm{OR}}$$

$$t_{cd} = t_{cd\_\mathrm{AND}}$$

Temporización

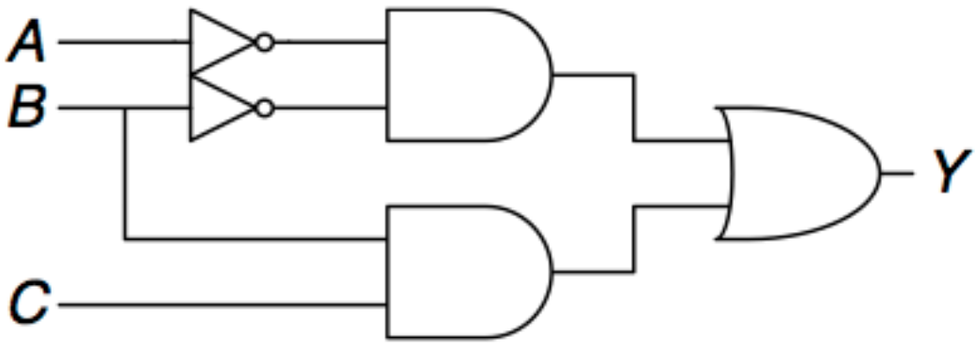
Glitches o hazards

- Hasta ahora hemos pensado sólo en casos donde una transición de entrada causa una transición de salida.
- Es posible que sólo una transición de entrada cause múltiples transiciones de salida.
- Éstas se conocen como glitches o hazards.
- Aunque muchas veces no causan problemas, en algunas aplicaciones producen problemas serios.

Temporización

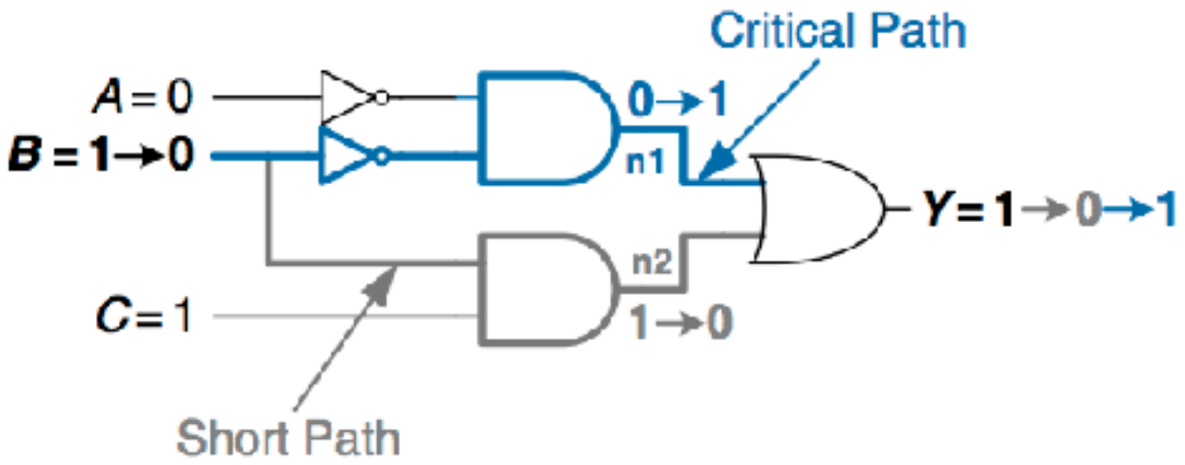
Glitches o hazards

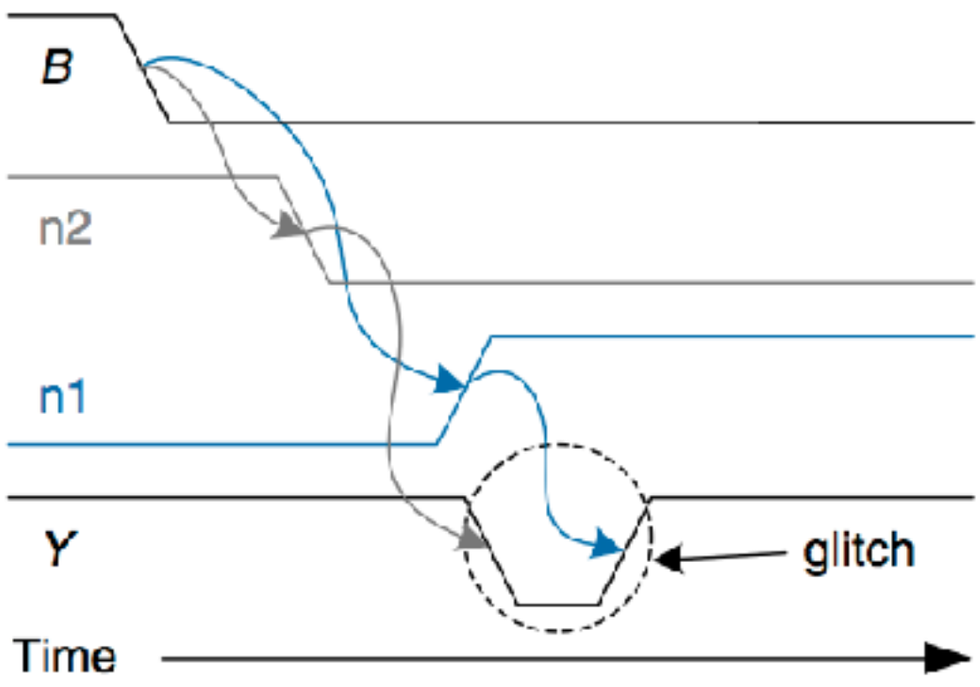
Circuito con *glitch*



Y	AB = 00	01	11	10
C = 0	1	0	0	0
C = 1	1	1	1	0

$$Y = \overline{A}\,\overline{B} + BC$$

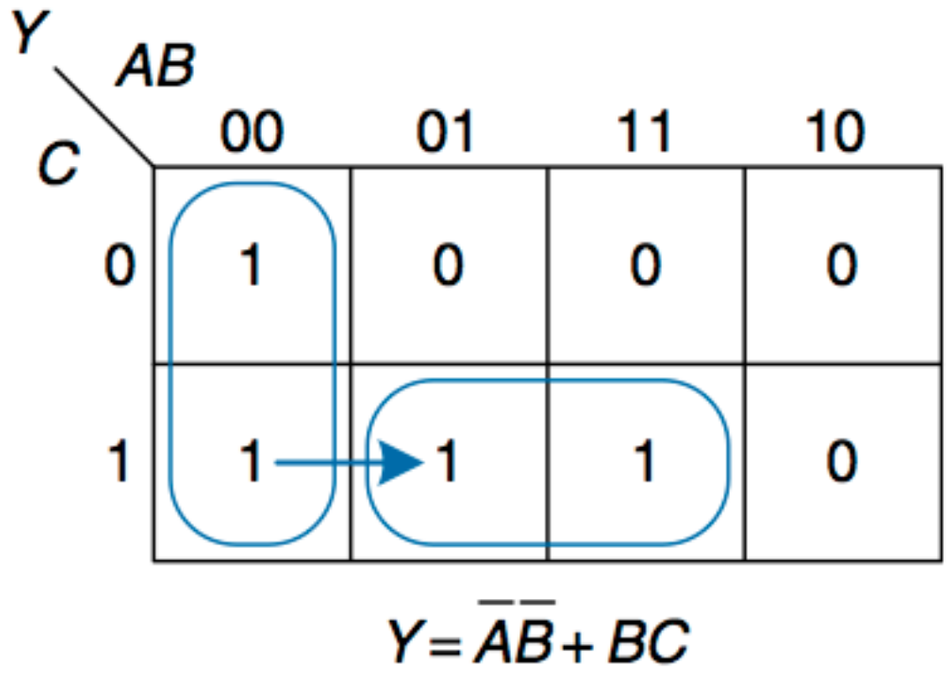




Temporización

Glitches o hazards

- Podemos eliminar el glitch agregando una compuerta.
- Esto queda claro estudiando el mapa de Karnaugh del circuito.
- La figura muestra que la transición de entrada de ABC=001 a ABC=011 produce un cambio de implicante primario (subcubo)



Temporización

Glitches o hazards

- Agregando un subcubo que cubra la frontera entre ambos implicantes soluciona el problema