Tema 1: Circuitos Digitales

1.2 Circuitos Combinacionales

Miguel Ángel Otaduy

Universidad
Rey Juan Carlos

Circuitos Combinacionales

- Tipología
  - Salidas = función (entradas)
- Ejemplo: Sumador.
  - Entradas: Número 1, Número 2
  - Salidas: Resultado

- Tecnología de Desarrollo
  - Puertas lógicas
- Herramientas de Diseño
  - Álgebra de Boole, tabla de verdad, mapas de Karnaugh…

Circuitos Secuenciales

- Tipología
  - Salidas = función (entradas, estado)
- Ejemplo: Ascensor
  - Entradas: Botones de pisos, botones de llamada, detectores de pisos
  - Salidas: Motor (subir, bajar, parado)

- Tecnología de Desarrollo
  - Puertas lógicas y biestables
- Herramientas de Diseño
  - Máquinas de estados

Álgebra de Boole: Definición

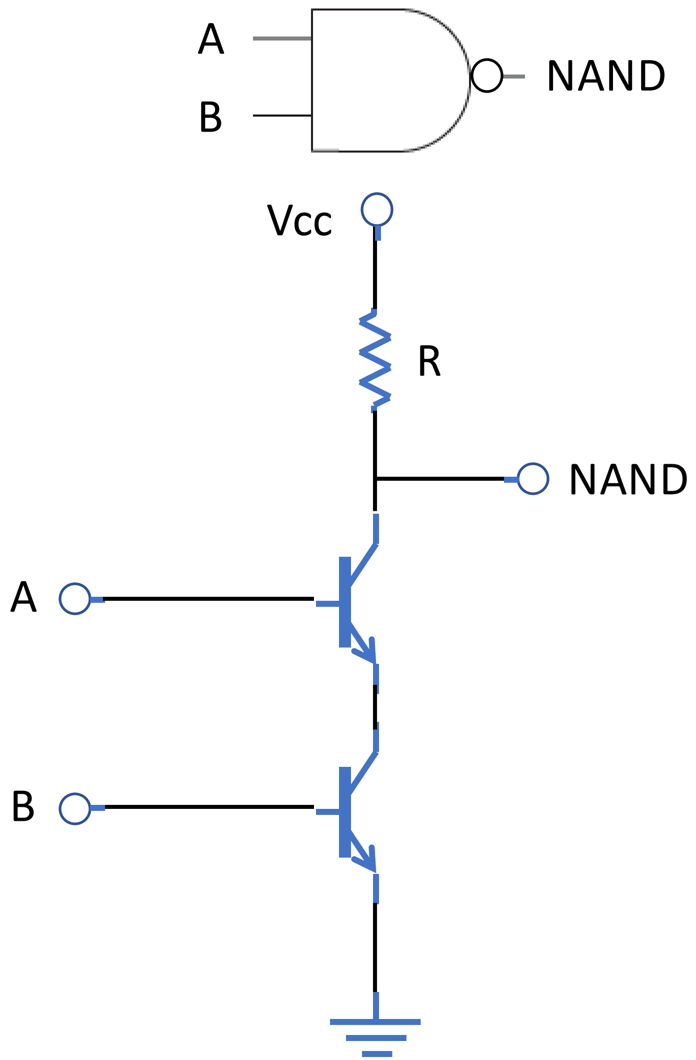
- Cada variable x puede tomar 2 valores {0, 1}
- Valor complementario
  NOT(x), x’, x
- Función producto Y lógico
  x AND y, x * y
- Función suma O lógica
  x OR y, x + y

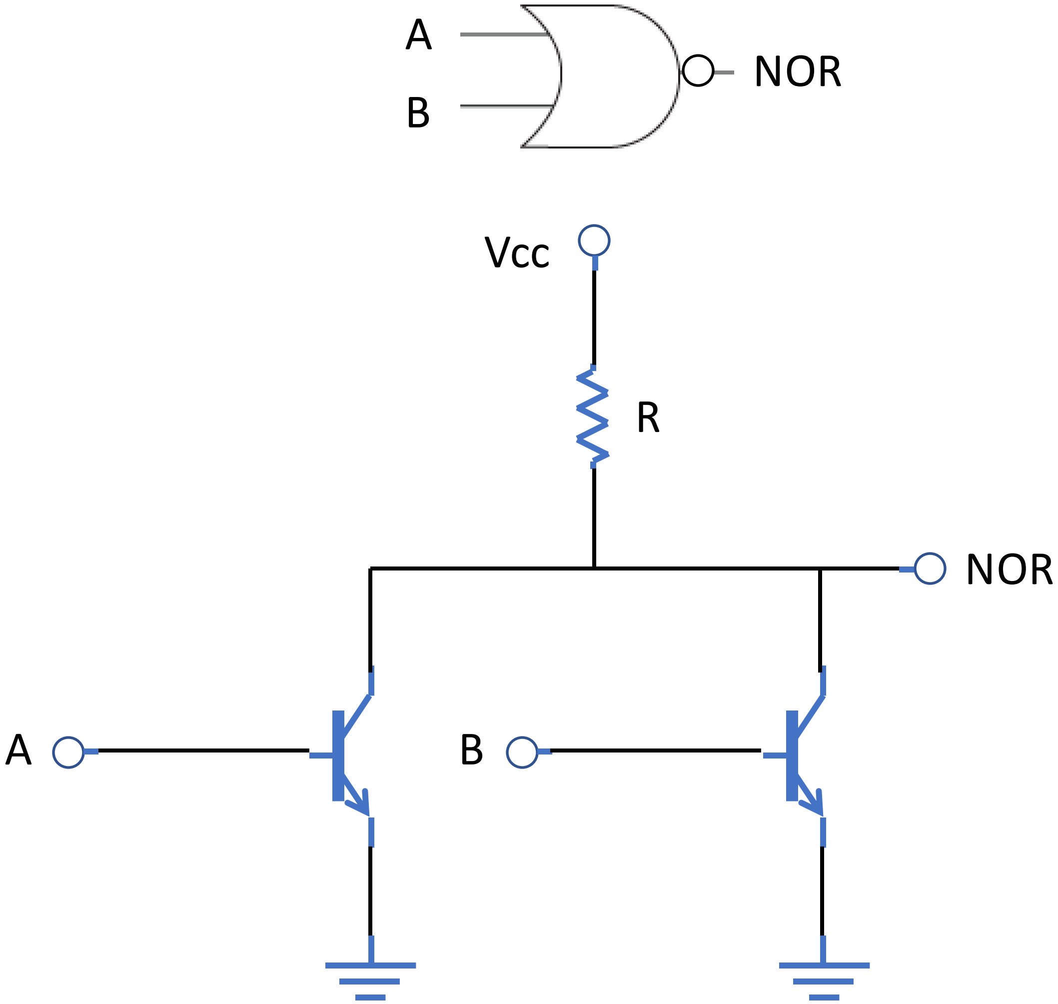
x	y	AND
0	0	0
0	1	0
1	0	0
1	1	1

x	y	OR
0	0	0
0	1	1
1	0	1
1	1	1

x	NOT
0	1
1	0

Puertas Lógicas





Álgebra de Boole: Teoremas y Propiedades

Propiedad asociativa	a+(b+c) = (a+b)+c = a+b+c	a·(b·c) = (a·b)·c = a·b·c
Propiedad conmutativa	a+b = b+a	a·b = b·a
Propiedad distributiva	a+(b·c) = (a+b) · (a+c)	a·(b+c) = a·b + a·c
Elemento neutro	0+a=a	1·a=a
	1+a=1	0·a=0
Teoremas de identidad	a+a'=1	a·a'=0
Teoremas de idempotencia	a+a=a	a·a=a
Teorema de involución	(a')'=a	
Teoremas de absorción	a+a·b = a	a·(a+b) = a
	a+a'·b=a+b	a·(a'+b)=a·b
	a·b+a·b' = a	(a'+b') ·(a'+b) = a'
Teoremas del consenso	a·b+a'·c+b·c = a·b+a'·c	(a+b)·(a'+c)·(b+c) = (a+b)·(a'+c)
Teoremas de De Morgan	(a+b)' = a'·b'	(a·b)' = a'+b'
Teorema de expansión	f(a,b) = a'·f(0,b) + a·f(1,b)	f(a,b) = [a+f(0,b)] · [a'+f(1,b)]

Funciones de Conmutación

- Cada salida de un circuito combinacional se describe mediante una función de conmutación
- Dadas n variables de entrada, una función de conmutación describe el valor (0,1) para cada una de las 2^n combinaciones de valores de entrada
- Representaciones:
  - Tabla de verdad
  - Expresiones de conmutación
  - Mapa de Karnaugh
- Orden de precedencia: NOT > AND > OR

Diseño de un Circuito Combinacional

- Definir la función de conmutación de cada salida
- Obtener su representación mediante una expresión de conmutación concisa
  - Tabla de verdad + teoremas álgebra Boole
  - Mapa de Karnaugh
- Implementar la expresión de conmutación mediante puertas lógicas

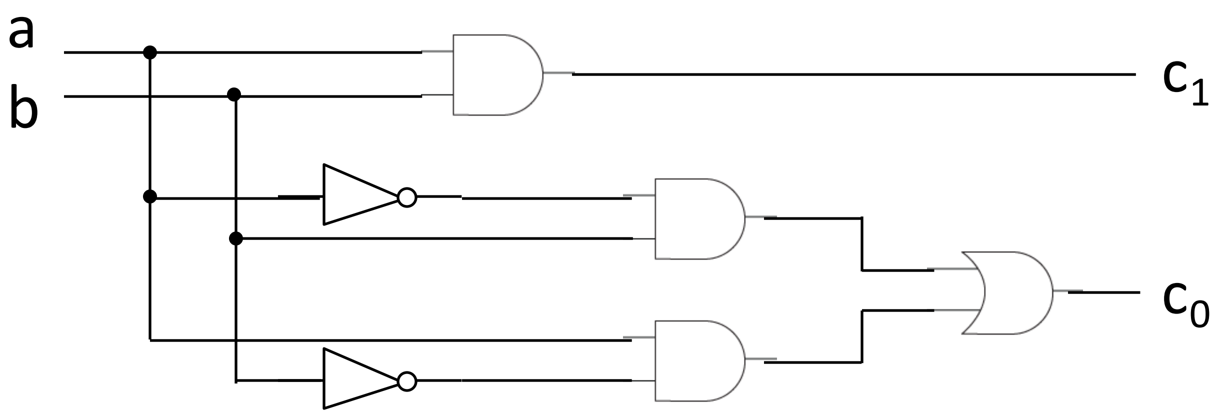
Ejemplo: Sumador de 1 bit

Sumador: $a + b = c_1c_0$

$c_0 = a' \cdot b + a \cdot b'$

$c_1 = a \cdot b$

a	**b**	**c_1c_0**
0	0	00
0	1	01
1	0	01
1	1	10



Suma de Productos Canónica

a	b	c	out
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$\rightarrow a' \cdot b' \cdot c'$

$\rightarrow a' \cdot b \cdot c$

Minterms

$\rightarrow a \cdot b \cdot c$

$$out = a' \cdot b' \cdot c' + a' \cdot b \cdot c + a \cdot b \cdot c$$

Producto de Sumas Canónico

a	b	c	out	
0	0	0	1	
0	0	1	0	→ $a+b+c'$
0	1	0	0	→ $a+b'+c$
0	1	1	1	
1	0	0	0	→ $a'+b+c$
1	0	1	0	→ $a'+b+c'$
1	1	0	0	→ $a'+b'+c$
1	1	1	1	

Maxterms de complementos

$$out = (a+b+c') \cdot (a+b'+c) \cdot (a'+b+c) \cdot (a'+b+c') \cdot (a'+b'+c)$$

Lógica Inversa

a	b	c	out	out’
0	0	0	1	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

$\rightarrow a'\cdot b'\cdot c$

$\rightarrow a'\cdot b\cdot c'$

Minterms

$\rightarrow a\cdot b'\cdot c'$

$\rightarrow a\cdot b'\cdot c$

$\rightarrow a\cdot b\cdot c'$

$$out' = a'\cdot b'\cdot c + a'\cdot b\cdot c' + a\cdot b'\cdot c' + a\cdot b'\cdot c + a\cdot b\cdot c'$$

$$out = (a+b+c')\cdot(a+b'+c)\cdot(a'+b+c)\cdot(a'+b+c')\cdot(a'+b'+c)$$

Mapas de Karnaugh

1 variable

X	
0	0
1	1

2 variables

$X_1 \backslash X_0$	0	1
0	0	1
1	2	3

3 variables

$X_2 \backslash X_1X_0$	00	01	11	10
0	0	1	3	2
1	4	5	7	6

4 variables

$X_3X_2 \backslash X_1X_0$	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

5 variables

$X_3X_2 \backslash X_1X_0$	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

$\overline{X_4}$ (X_4=0)

$X_3X_2 \backslash X_1X_0$	00	01	11	10
00	16	17	19	18
01	20	21	23	22
11	28	29	31	30
10	24	25	27	26

X_4 (X_4=1)

Mapas de Karnaugh: Simplificación

X_3X_2 \ X_1X_0	00	01	11	10
00	0	1	3	2
01	4	5 **1**	7 **1**	6
11	12	13	15	14
10	8	9	11	10

X0·X1’·X2·X3’ + X0·X1·X2·X3’
= X0·X2·X3’

X_3X_2 \ X_1X_0	00	01	11	10
00	0	1	3	2
01	4	5 **1**	7 **1**	6
11	12	13 **1**	15 **1**	14
10	8	9	11	10

X0·X2

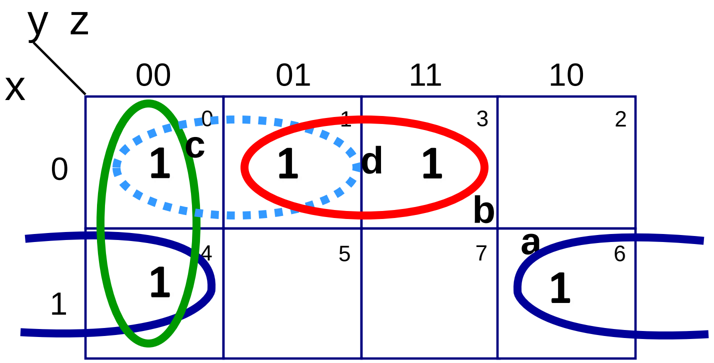
X_3X_2 \ X_1X_0	00	01	11	10
00	0 **1**	1	3	2 **1**
01	4	5	7	6
11	12	13	15	14
10	8 **1**	9	11	10 **1**

X0’·X2’

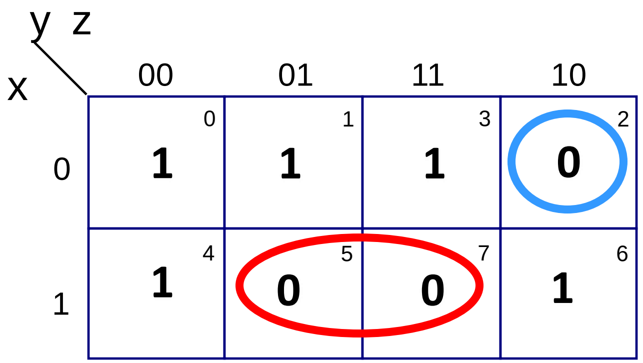
X_3X_2 \ X_1X_0	00	01	11	10
00	0 **1**	1 **1**	3 **1**	2 **1**
01	4	5	7	6
11	12	13	15	14
10	8 **1**	9 **1**	11 **1**	10 **1**

X2’

Mapas de Karnaugh: Solapamientos



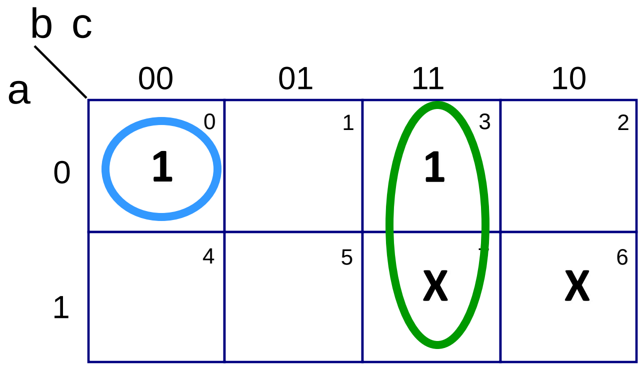
Opción a: x’·z + x·z’ + y’·z’
Opción b: x’·z + x·z’ + x’·y’



Por maxterms: (x+y’+z)·(x’+z’)

Mapas de Karnaugh: “No importa”

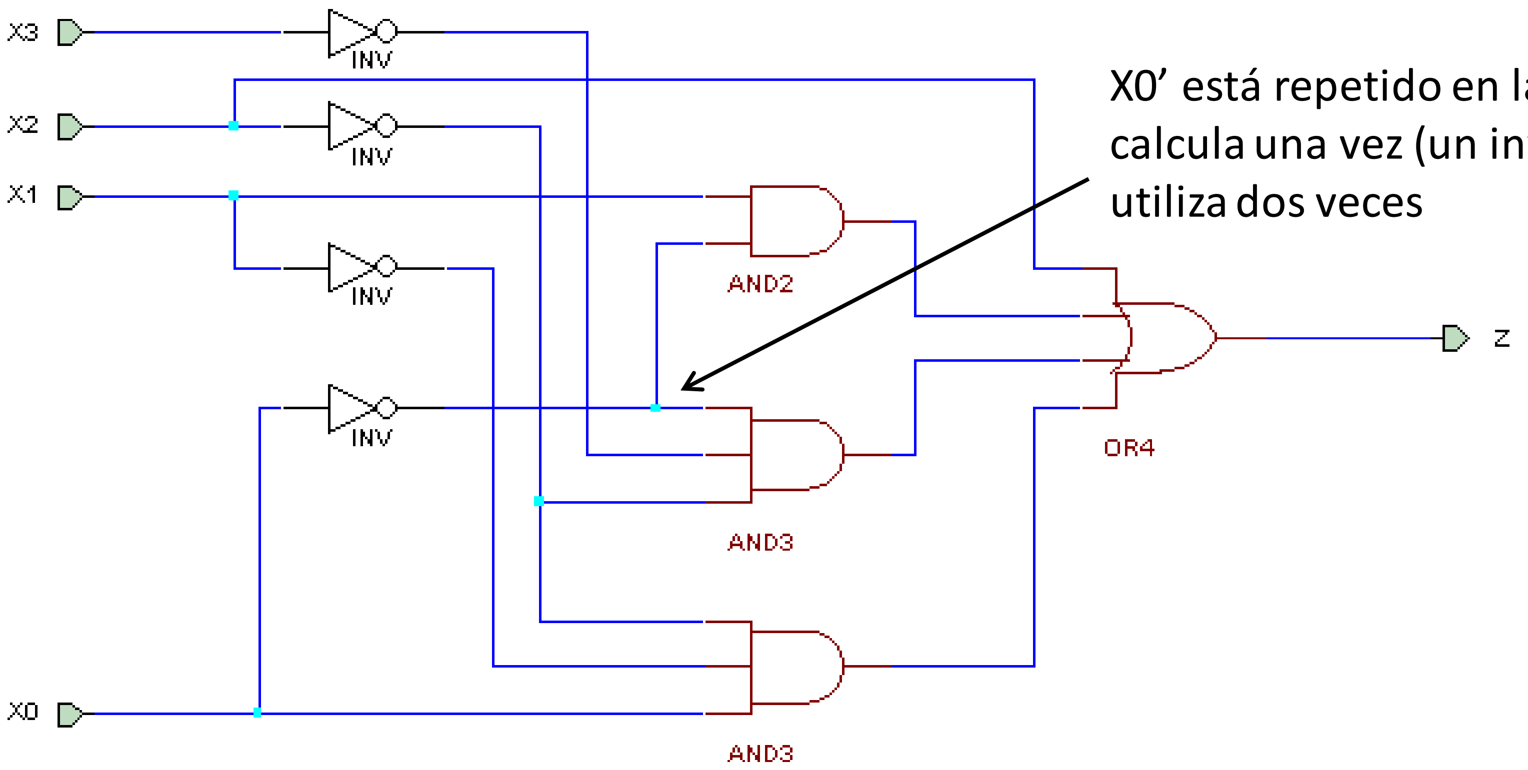
a	b	c	out
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	X
1	1	1	X



a’· b’· c’ + b · c

Implementación mediante Puertas

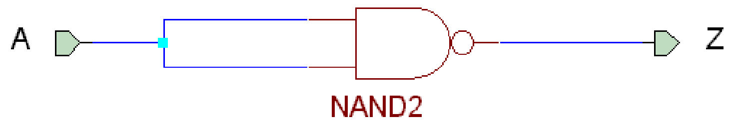
Z = X2 + X1·X0’ + X3’·X2’·X0’+X2’·X1’·X0



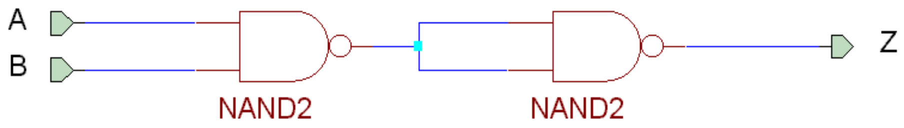
X0’ está repetido en la expresión. Se calcula una vez (un inversor), pero se utiliza dos veces

Puertas Universales NAND y NOR

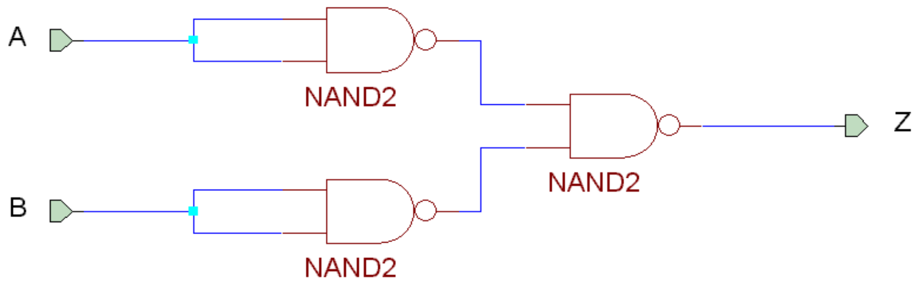
NOT(a) = NOT(a AND a)



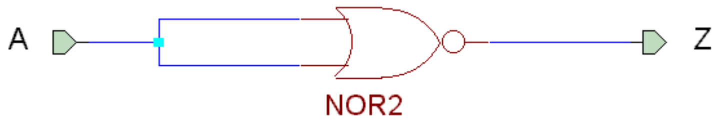
a AND b = NOT(NOT(a AND b))



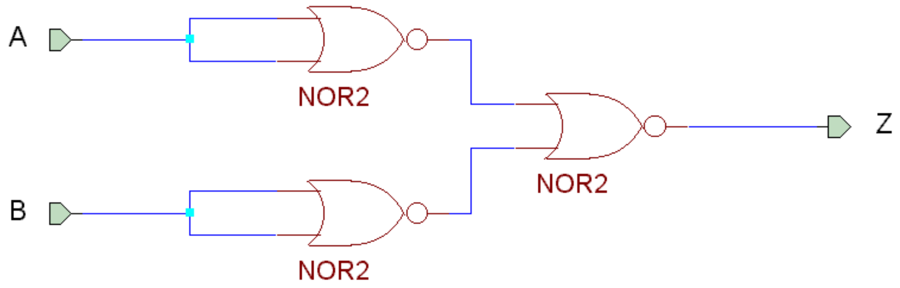
a OR b = NOT(NOT(a) AND NOT(b))



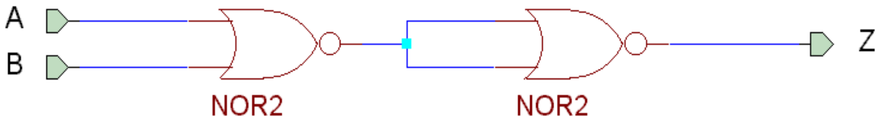
NOT(a) = NOT(a OR a)



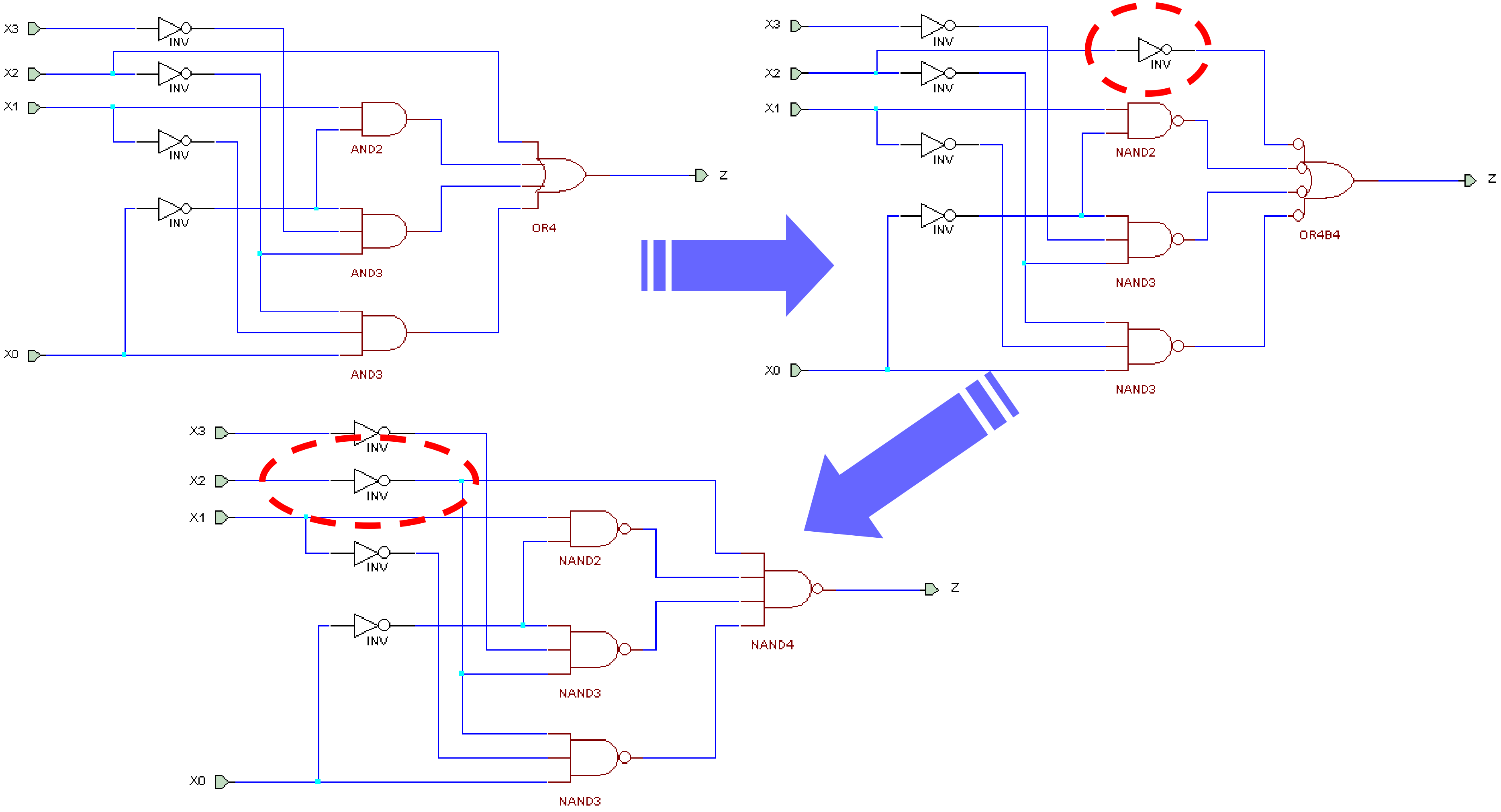
a AND b = NOT(NOT(a) OR NOT(b))



a OR b = NOT(NOT(a OR b))



Puertas Universales NAND



Puertas Universales NOR