



Tema 2:

# Especificación de sistemas combinacionales

## Fundamentos de computadores I

**José Manuel Mendías Cuadros**

*Dpto. Arquitectura de Computadores y Automática  
Universidad Complutense de Madrid*





# Contenidos

- ✓ Especificación de alto nivel / binaria.
- ✓ Codificación.
- ✓ Funciones de conmutación. Tablas de verdad.
- ✓ Expresiones de conmutación.
- ✓ Algebra de Boole. Transformaciones algebraicas.
- ✓ Forma canónica. Suma de productos.
- ✓ Mapas de Karnaugh. Simplificación.

Transparencias basadas en los libros:

- R. Hermida, F. Sánchez y E. del Corral. *Fundamentos de computadores.*
- D. Gajsky. *Principios de diseño digital.*



# Sistemas combinacionales

- La salida en cada instante depende exclusivamente del valor de la entrada en ese instante.
  - En todo momento, a misma entrada, misma salida.



$$z(t_i) = F(x(t_i)), \text{ con } x(t_i) \in E, z(t_i) \in S$$

- Para especificar su comportamiento deberán definirse:
  - Los conjuntos discretos de valores de entrada/salida: E, S
  - La función  $F: E \rightarrow S$



# Sistemas combinacionales



$$x(t) \in E = \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$$

$$z(t) \in S = \{ 0, 1, 2 \}$$

$$F: E \rightarrow S / z(t) = f( x(t) ) = x(t) \bmod 3$$

Simulación de su comportamiento:

<b>x(t)</b>	0	1	5	1	1	2	8	1	9	0
<b>z(t)</b>	0	1	2	1	1	2	2	1	0	0

—————→ tiempo



# Especificación de alto nivel

- **Especificación del dominio:**
  - Conjunto discreto de valores que puede tomar la entrada.
- **Especificación del codominio:**
  - Conjunto discreto de valores que puede tomar la salida.
- **Función de entrada/salida:**
  - Definición del comportamiento del sistema: qué valor toma la salida para cada posible valor de la entrada
  - Mediante tabla, expresión aritmética, condicional, lógica...  
o una composición de todas ellas.

**Sin embargo, la información debe estar codificada en binario para que sea implementable en un sistema digital**



# Especificación binaria

- La entrada es un vector de  $n$  bits
  - $\underline{x} \in \{0, 1\}^n$  es decir,  $\underline{x} = (x_{n-1} \dots x_0)$  con  $x_i \in \{0, 1\}$
- La salida es un vector de  $m$  bits
  - $\underline{z} \in \{0, 1\}^m$  es decir,  $\underline{z} = (z_{m-1} \dots z_0)$  con  $z_i \in \{0, 1\}$
- Función de entrada/salida
  - $m$  **funciones de conmutación** de  $n$  variables definiendo cada una el comportamiento de un bit de la salida
  - $\underline{F} = \{ f_i : \{0, 1\}^n \rightarrow \{0, 1\} / z_i = f_i(\underline{x}), \text{ con } 0 \leq i \leq m-1 \}$





# Descripción binaria

- Proceso de obtener una **especificación binaria** partiendo de una **especificación de alto nivel**:
  1. Codificar el dominio (elegir una representación binaria de cada elemento).
  2. Codificar el codominio.
  3. Traducir la función de E/S.
- Para una misma especificación de alto nivel existen **infinidad** de especificaciones binarias válidas.
  - Cada una **con distinta codificación** del dominio/codominio
- La cardinalidad del dominio/codominio determina la longitud mínima del vector de bits  $\underline{x} / \underline{z}$ 
  - Para que todos los puntos del dominio/codominio puedan estar representados por una cadena de bits distinta:
    - $n \geq \log_2( |E| )$  y  $m \geq \log_2( |S| )$   $[ \log_2(x) = \ln(x) / \ln(2) ]$
    - casi siempre quedarán codificaciones sin usar



# Descripción binaria

- Codificación domino: BCD (4 bits) – usando solo 10 códigos
- Codificación codominio: one-hot (3 bits)
  - { 0 → (001), 1 → (010), 2 → (100) }
- Traducción de la función de E/S
  - $F = \{ (0000) \rightarrow (001), (0001) \rightarrow (010), (0010) \rightarrow (100), (0011) \rightarrow (001), (0100) \rightarrow (010), (0101) \rightarrow (100), (0110) \rightarrow (001), (0111) \rightarrow (010), (1000) \rightarrow (100), (1001) \rightarrow (001) \}$

## Simulación de su comportamiento:

<b><math>\underline{x}(t)</math></b>	0000	0001	0101	0001	0001	0010	1000	0001	1001	0000
<b><math>\underline{z}(t)</math></b>	001	010	100	010	010	100	100	010	001	001

—————→ tiempo





# Funciones de conmutación (FC)

- Una **función de conmutación** de  $n$  variables es una aplicación

$$f: \{0, 1\}^n \rightarrow \{0, 1\}$$

- Cuando es **total** (todo punto del dominio está asociado a uno del codominio) se dice que está **completamente especificada**

- Se suele definir mediante una **tabla de verdad** que indica el valor que toma la función en cada punto del dominio.

	$x_2$	$x_1$	$x_0$	$f(x_2, x_1, x_0)$
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	0
7	1	1	1	1



# Funciones de conmutación (FC)

- El número de funciones de conmutación distintas de  $n$  variables es finito:  $2^{2^n}$ 
  - Para 2 variables existen únicamente 16 distintas

$x_1$	$x_0$	$f_0$	$f_1$	$f_2$	$f_3$	$f_4$	$f_5$	$f_6$	$f_7$	$f_8$	$f_9$	$f_{10}$	$f_{11}$	$f_{12}$	$f_{13}$	$f_{14}$	$f_{15}$
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
		nula	and	$x_1$	$x_0$	XOR	OR	NOR	XNOR	not $x_0$		not $x_1$		nand		unidad	



# Funciones de conmutación (FC)

- A veces las funciones de conmutación son **parciales** (no están definidas para ciertos puntos del dominio).
  - Típicamente porque existen códigos que no representan ningún valor de alto nivel.
- Una **función de conmutación incompletamente especificada** de  $n$  variables es una aplicación:

$$f: \{0, 1\}^n \rightarrow \{0, 1, -\}$$

- Donde '-' (don't care) denota **indiferencia**: da igual que la función valga 0 ó 1 en aquellos puntos del dominio asociados a este valor.



# Funciones de conmutación (FC)

	$x_3$	$x_2$	$x_1$	$x_0$	$z_2$	$z_1$	$z_0$
0	0	0	0	0	0	0	1
1	0	0	0	1	0	1	0
2	0	0	1	0	1	0	0
3	0	0	1	1	0	0	1
4	0	1	0	0	0	1	0
5	0	1	0	1	1	0	0
6	0	1	1	0	0	0	1
7	0	1	1	1	0	1	0
8	1	0	0	0	1	0	0
9	1	0	0	1	0	0	1
10	1	0	1	0	-	-	-
11	1	0	1	1	-	-	-
12	1	1	0	0	-	-	-
13	1	1	0	1	-	-	-
14	1	1	1	0	-	-	-
15	1	1	1	1	-	-	-



$E = \{ 0, \dots, 9 \}$   
la codificación es BCD  
nunca aparecerán estos  
códigos



# Expresiones de conmutación (EC)

- Forma alternativa de definir FC completamente especificadas
  - Compacta, manipulable y **directamente sintetizable**.
- **Alfabeto:**  $\{ x_i, 0, 1, +, \cdot, -, (, ) \}$ 
  - Variables lógicas:  $x_i$  (*puede usarse cualquier letra con o sin subíndice*)
  - Constantes: 0, 1
  - Operadores : +,  $\cdot$ , -
  - Símbolos auxiliares: (, )
- **Reglas de generación:**
  1. Toda variable lógica es una EC válida.
  2. 0 y 1 son EC válidas.
  3. Si A es una EC válida,  $\bar{A}$  también lo es.
  4. Si A y B son EC válidas,  $(A)$ ,  $A+B$  y  $A \cdot B$  también lo son.
  5. Solo son EC válidas las generadas usando las reglas 1 a 4.



# Expresiones de conmutación (EC)

- **Semántica:** el álgebra de conmutación  $\{ \{0,1\}, \text{and, or, not} \}$

operador **and**

$x$	$y$	$x \cdot y$
0	0	0
0	1	0
1	0	0
1	1	1

operador **or**

$x$	$y$	$x + y$
0	0	0
0	1	1
1	0	1
1	1	1

operador **not**

$x$	$\bar{x}$
0	1
1	0

- **Valor de una EC, E, para una asignación,  $\underline{a}$ :**  $v(E, \underline{a})$ 
  - Resultado de sustituir las variables de E por los valores indicados en  $\underline{a}$  y realizar las operaciones de acuerdo con el álgebra de conmutación.

$$v(x_2 + \bar{x}_2 \cdot x_1 + x_1 \cdot x_0, (0, 1, 0)) = 0 + \bar{0} \cdot 1 + 1 \cdot 0 = 0 + 1 \cdot 1 + 1 \cdot 0 = 1$$



# Expresiones de conmutación (EC)

- Para una expresión de conmutación dada, el conjunto de todos los pares

$$f = \{ (\underline{a}, v(E, \underline{a})) / \underline{a} \in \{0,1\}^n \}$$

es una función de conmutación.

- En ese caso diremos que  $E$  representa a  $f$
- Dos EC son equivalentes si representan a la misma función de conmutación.
  - Toda FC tiene infinitas EC equivalentes que la representan.
  - Habrá unas más convenientes que otras, en particular las más simples.

# Expresiones de conmutación (EC)



$$\overline{x_1} + x_1 \cdot x_0$$

$$v(\overline{x_1} + x_1 \cdot x_0, (0,0)) = \overline{0} + 0 \cdot 0 = 1$$

$$v(\overline{x_1} + x_1 \cdot x_0, (0,1)) = \overline{0} + 0 \cdot 1 = 1$$

$$v(\overline{x_1} + x_1 \cdot x_0, (1,0)) = \overline{1} + 1 \cdot 0 = 0$$

$$v(\overline{x_1} + x_1 \cdot x_0, (1,1)) = \overline{1} + 1 \cdot 1 = 1$$

	$x_1$	$x_0$	$f(x_1, x_0)$
0	0	0	1
1	0	1	1
2	1	0	0
3	1	1	1

**SON EQUIVALENTES**

$$\overline{x_1} + x_0$$

$$v(\overline{x_1} + x_0, (0,0)) = \overline{0} + 0 = 1$$

$$v(\overline{x_1} + x_0, (0,1)) = \overline{0} + 1 = 1$$

$$v(\overline{x_1} + x_0, (1,0)) = \overline{1} + 0 = 0$$

$$v(\overline{x_1} + x_0, (1,1)) = \overline{1} + 1 = 1$$

	$x_1$	$x_0$	$f(x_1, x_0)$
0	0	0	1
1	0	1	1
2	1	0	0
3	1	1	1





# Expresiones de conmutación (EC)

- El álgebra de conmutación es un **álgebra de Boole** por lo que dadas 2 EC, A y B, se cumplen las siguientes propiedades:

Propiedad	Versión “+”	Versión “.”
Conmutativa	$A + B = B + A$	$A \cdot B = B \cdot A$
Distributiva	$A + (B \cdot C) = (A + B) \cdot (A + C)$	$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$
Elemento neutro	$0 + A = A$	$1 \cdot A = A$
Elem. complementario	$A + \bar{A} = 1$	$A \cdot \bar{A} = 0$
Idempotencia	$A + A = A$	$A \cdot A = A$
Asociativa	$A + (B + C) = (A + B) + C$	$A \cdot (B \cdot C) = (A \cdot B) \cdot C$
Elemento dominante	$1 + A = 1$	$0 \cdot A = 0$
Involución	$\bar{\bar{A}} = A$	
Absorción	$A + (A \cdot B) = A$	$A \cdot (A + B) = A$
Leyes de Morgan	$\overline{A + B} = \bar{A} \cdot \bar{B}$	$\overline{A \cdot B} = \bar{A} + \bar{B}$



# Expresiones de conmutación (EC)

- Las anteriores propiedades permiten transformar algebraicamente una EC en otra/s equivalente/s.

$$x_2 \cdot x_1 + x_2 \cdot \overline{x_1} + x_2 \cdot x_0$$

$$= x_2 \cdot (x_1 + \overline{x_1}) + x_2 \cdot x_0$$

$$= x_2 \cdot 1 + x_2 \cdot x_0$$

$$= x_2 \cdot (1 + x_0)$$

$$= x_2 \cdot 1$$

$$= x_2$$

*distributiva*

*elem. complementario*

*distributiva*

*elem. dominante*

*elem. neutro*



# Otras operaciones lógicas

- Además de los operadores primitivos del álgebra de conmutación es muy común referirse a otros operadores derivados:

operador **nand**

$x$	$y$	$\frac{x \uparrow y}{(x \cdot y)}$
0	0	1
0	1	1
1	0	1
1	1	0

operador **nor**

$x$	$y$	$\frac{x \downarrow y}{(x + y)}$
0	0	1
0	1	0
1	0	0
1	1	0

operador **xor**

$x$	$y$	$\frac{x \oplus y}{x \cdot \bar{y} + \bar{x} \cdot y}$
0	0	0
0	1	1
1	0	1
1	1	0

operador **xnor**

$x$	$y$	$\frac{\overline{(x \oplus y)}}{x \cdot y + \bar{x} \cdot \bar{y}}$
0	0	1
0	1	0
1	0	0
1	1	1

- Todos ellos son conmutativos.
- NAND y NOR no son asociativos.** XOR y XNOR sí lo son.



# EC vs. lenguaje natural

- En muchos casos es posible obtener directamente una EC desde un enunciado en lenguaje natural

*La barrera debe abrirse si es de día y hay un coche esperando o si el vigilante presiona un pulsador*



- Codificando los sucesos en "lógica directa"
  - $L=1 \Leftrightarrow$  Se detecta luz (es de día)
  - $P=1 \Leftrightarrow$  Se detecta coche
  - $A=1 \Leftrightarrow$  Se ha presionado el pulsador
  - $M=1 \Leftrightarrow$  Se activa el motor que abre la barrera
- La formulación del enunciado queda:

$$M = \begin{cases} 1 & \text{si } L=1 \text{ y } P=1 \text{ o } A=1 \\ 0 & \text{en caso contrario} \end{cases} \Leftrightarrow M = L \cdot P + A$$



# Recapitulación

- Hasta el momento tenemos:
  - Dada una FC, **existen infinidad de EC** que la representan.
  - Dada una FC, **no sabemos cómo obtener una EC** que la represente.
  - Dada una EC, **es tedioso obtener la tabla de verdad** de la FC que representa.
  - Dada una EC, **es complejo obtener una EC simplificada** equivalente.
- La definición de una **forma canónica** permitirá:
  - Que toda FC tenga asociada una única EC normalizada.
  - Que ésta pueda obtenerse fácilmente a partir de una tabla de verdad.
  - Que el mecanismo de obtención de la tabla de verdad de la FC que representa una cierta EC sea más simple.
  - Abrir las puertas a un mecanismo de simplificación de EC.



# Suma de productos canónica

- **Literal:** EC compuesta por una única variable natural o complementada.

$$\overline{x_0} \quad x_1$$

- **Término producto:** EC compuesta únicamente por un producto de literales.

$$x_1 \cdot x_0 \quad \overline{x_1} \cdot x_0 \cdot x_1 \cdot x_2$$

- **Mintérmino de n variables:** término producto de n literales, en donde cada variable aparece una y solo una vez.

$$\overline{x_1} \cdot x_0 \quad \overline{x_2} \cdot x_1 \cdot x_0$$

- **Suma de productos:** EC compuesta únicamente por sumas de términos producto.

$$x_1 \cdot \overline{x_0} \quad x_2 \cdot \overline{x_1} + x_2 \cdot \overline{x_1} \cdot \overline{x_0}$$

# Suma de productos canónica



- **Notación:** Un **mintérmino de n variables** se representará por  $m_i$  o  $m(i)$ , siendo  $i$  el número cuya representación binaria se obtiene sustituyendo en el mintérmino ordenado (variables de mayor a menor peso):
  - Cada variable complementada por un 0.
  - Cada variable sin complementar por un 1

$$e(x_3, x_2, x_1, x_0) = \overline{x_3} \cdot x_2 \cdot \overline{x_1} \cdot x_0 = m_5 = m(5)$$
$$(0 \ 1 \ 0 \ 1)_2 = 5_{10}$$

$$e(x_3, x_2, x_1, x_0) = \overline{x_3} \cdot x_2 \cdot x_1 \cdot x_0 = m_7 = m(7)$$
$$(0 \ 1 \ 1 \ 1)_2 = 7_{10}$$



# Suma de productos canónica

- **Propiedad:** El valor de un mintermino para una asignación dada es:

$$v( m_i, \underline{a} ) = \begin{cases} 1 & \text{si } i = (\underline{a})_{10} \\ 0 & \text{en otro caso} \end{cases}$$

- es decir, el mintermino  $m_i$  representa a una FC que vale 0 en todos sus puntos del dominio excepto en el  $i$ , en donde vale 1.

$$e( x_1, x_0 ) = \overline{x_1} \cdot x_0 = m_1$$

$$v( \overline{x_1} \cdot x_0, (0,0) ) = \overline{0} \cdot 0 = 0$$

$$v( \overline{x_1} \cdot x_0, (0,1) ) = \overline{0} \cdot 1 = 1$$

$$v( \overline{x_1} \cdot x_0, (1,0) ) = \overline{1} \cdot 0 = 0$$

$$v( \overline{x_1} \cdot x_0, (1,1) ) = \overline{1} \cdot 1 = 0$$

	$x_1$	$x_0$	$f(x_1, x_0)$
0	0	0	0
1	0	1	1
2	1	0	0
3	1	1	0





# Suma de productos canónica

- **Suma de productos canónica (SPC):** EC compuesta únicamente por sumas de mintérminos en la que no hay mintérminos repetidos.

$$\begin{aligned} e(x_2, x_1, x_0) &= x_2 \cdot x_1 \cdot x_0 + \overline{x_2} \cdot x_1 \cdot x_0 + \overline{x_2} \cdot \overline{x_1} \cdot x_0 \\ &= m_7 + m_3 + m_1 = \sum m(7,3,1) \end{aligned}$$

- **Propiedad:** Toda SPC representa a una FC que vale 1 en cada uno de los puntos del dominio asociados a cada uno de los mintérminos que forman la SPC y 0 en el resto.
  - Y viceversa, toda FC de n variables puede representarse como una SPC compuesta por la suma de todos los mintérminos de n variables asociados a cada uno de los puntos del dominio para los cuales la FC vale 1.
  - Además, **toda FC, tiene una y solo una representación como SPC** (por eso se llama canónica).



# Suma de productos canónica

$$e(x_2, x_1, x_0) = \sum m(7,3,1)$$

$$x_2 \cdot x_1 \cdot x_0 + \overline{x_2} \cdot x_1 \cdot x_0 + \overline{x_2} \cdot \overline{x_1} \cdot x_0$$

	$x_2$	$x_1$	$x_0$	$m_7$	$m_3$	$m_1$	$m_7 + m_3 + m_1$
<b>0</b>	0	0	0	0	0	0	0
<b>1</b>	0	0	1	0	0	1	1
<b>2</b>	0	1	0	0	0	0	0
<b>3</b>	0	1	1	0	1	0	1
<b>4</b>	1	0	0	0	0	0	0
<b>5</b>	1	0	1	0	0	0	0
<b>6</b>	1	1	0	0	0	0	0
<b>7</b>	1	1	1	1	0	0	1



# Suma de productos canónica

- **Notación:** La comodidad de la notación compacta de una SPC como sumatorio de minterminos suele usarse para describir FC incompletamente especificadas.

Téngase en cuenta que es un **abuso de notación**, ya que las EC solo pueden representar FC completamente especificadas.

	$x_2$	$x_1$	$x_0$	$f(x_2, x_1, x_0)$
<b>0</b>	0	0	0	0
<b>1</b>	0	0	1	1
<b>2</b>	0	1	0	0
<b>3</b>	0	1	1	1
<b>4</b>	1	0	0	0
<b>5</b>	1	0	1	-
<b>6</b>	1	1	0	-
<b>7</b>	1	1	1	1

$$e(x_2, x_1, x_0) = \sum m(7, 3, 1) + \sum d(5, 6)$$



# Conversión de una EC a su SPC

- Dos EC son equivalentes si representan a la misma FC.
  - Dado que toda FC tiene una única SPC que la representa: dos EC son equivalentes si ambas son equivalentes a una misma SPC.
  
- Método 1:
  - Evaluando la EC punto a punto hasta obtener la tabla de verdad de la FC que representa.
  
- Método 2:
  - Transformando la EC en una suma de productos:
    - Aplicando ley de Morgan
    - Aplicando la distributividad del producto
  - Multiplicando cada término producto que no contenga una cierta variable  $x_i$  por  $(x_i + \bar{x}_i) = 1$  y aplicando distributividad.
  - Eliminando los mintérminos repetidos.



# Conversión de una EC a su SPC

$$e(x_2, x_1, x_0) = x_2 \overline{(x_1 x_0)} + x_1 x_0$$

	$x_2$	$x_1$	$x_0$	$x_1 x_0$	$\overline{(x_1 x_0)}$	$x_2 \overline{(x_1 x_0)}$	$x_2 \overline{(x_1 x_0)} + x_1 x_0$
<b>0</b>	0	0	0	0	1	0	0
<b>1</b>	0	0	1	0	1	0	0
<b>2</b>	0	1	0	0	1	0	0
<b>3</b>	0	1	1	1	0	0	1
<b>4</b>	1	0	0	0	1	1	1
<b>5</b>	1	0	1	0	1	1	1
<b>6</b>	1	1	0	0	1	1	1
<b>7</b>	1	1	1	1	0	0	1

$$\sum m(3, 4, 5, 6, 7)$$



# Conversión de una EC a su SPC

$$x_2 \overline{(x_1 x_0)} + x_1 x_0$$

*ley de Morgan*

$$= x_2 (\overline{x_1} + \overline{x_0}) + x_1 x_0$$

*distributiva*

$$= x_2 \overline{x_1} + x_2 \overline{x_0} + x_1 x_0$$

*elem. neutro e idempotencia*

$$= x_2 \overline{x_1} (x_0 + \overline{x_0}) + x_2 (x_1 + \overline{x_1}) \overline{x_0} +$$

$$+ (x_2 + \overline{x_2}) x_1 x_0 \quad \textit{distributiva}$$

$$= x_2 \overline{x_1} x_0 + x_2 \overline{x_1} \overline{x_0} + x_2 x_1 \overline{x_0} + x_2 \overline{x_1} \overline{x_0}$$

$$+ x_2 x_1 x_0 + \overline{x_2} x_1 x_0$$

$$= m_5 + m_4 + m_6 + \cancel{m_4} + m_7 + m_3 \quad \textit{eliminación de repetidos}$$

$$= \sum m(3, 4, 5, 6, 7)$$



# Mapas de Karnaugh

- Mapa de Karnaugh: tabla de verdad de doble entrada que permite obtener de manera gráfica una **EC mínima en forma de suma de productos** que la represente.
  - **EC mínima** que tenga el menor número de términos producto y éstos el menor número de literales.
- Un mapa de **Karnaugh de n variables** tiene las siguientes propiedades:
  - Como la tabla de verdad que es, tiene  **$2^n$  casillas** cada una de ellas asociada a un mintérmino.
  - Los mintérminos asociados a **casillas adyacentes** solo se **diferencian en la polaridad de una de las variables**.
    - Dos mintérminos adyacentes pueden representarse por un término producto en donde no aparece la variable con diferente polaridad.



# Mapas de Karnaugh

	$x_0$	0	1
$x_1$	0	0 (00)	1 (01)
	1	2 (10)	3 (11)

$x_0$

**2 variables**

		$x_1 x_0$	00	01	11	10
$x_2$	0		0	1	3	2
	1		4	5	7	6

$x_0$        $x_1$

$x_2$

**3 variables**





# Mapas de Karnaugh

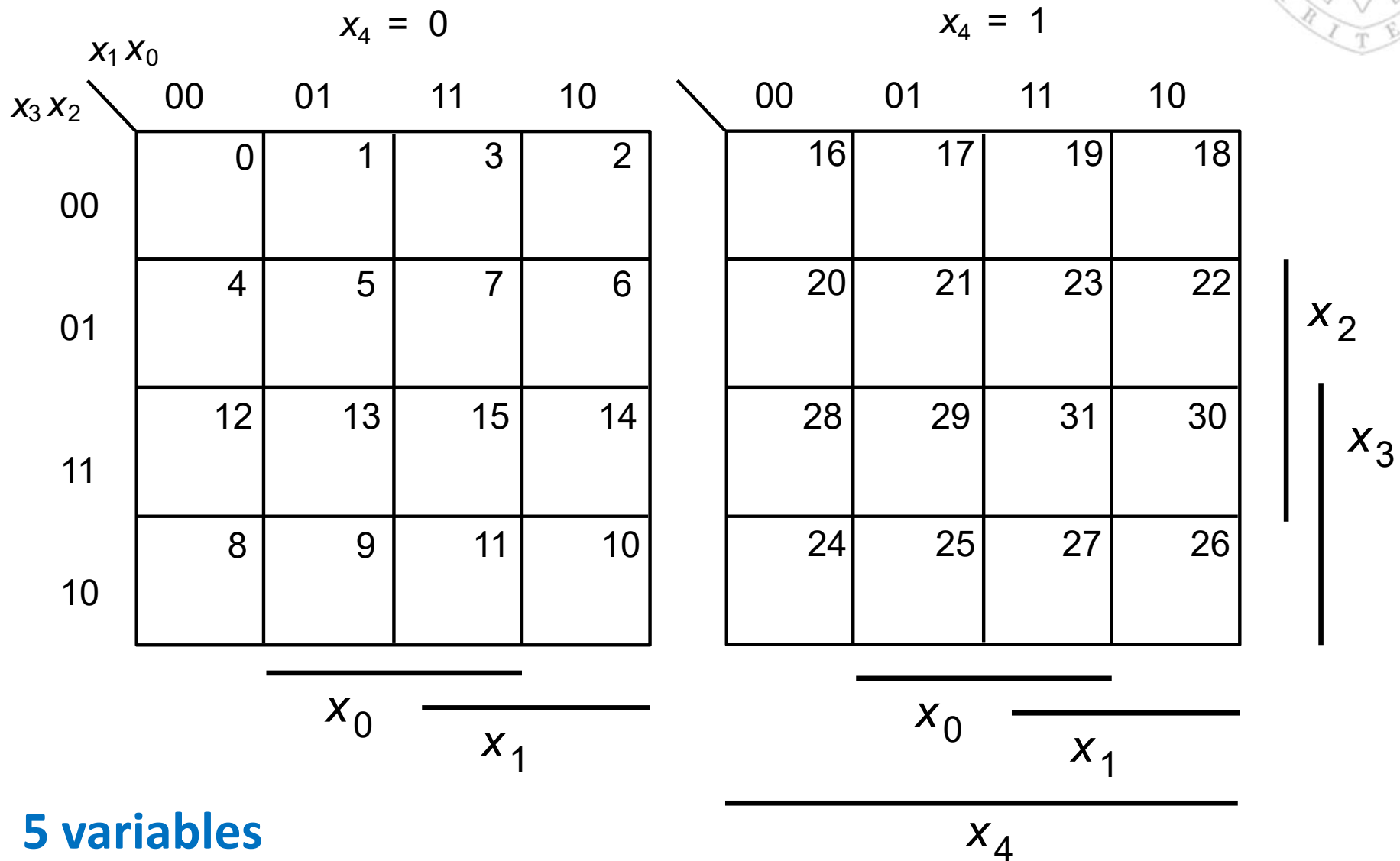
		$x_1 x_0$			
		00	01	11	10
$x_3 x_2$	00	0	1	3	2
	01	4	5	7	6
11	12	13	15	14	
10	8	9	11	10	

Diagram illustrating a 4-variable Karnaugh map. The map is a 4x4 grid with rows labeled  $x_3 x_2$  (00, 01, 11, 10) and columns labeled  $x_1 x_0$  (00, 01, 11, 10). The cells contain the decimal values 0 through 15. The map is annotated with variable labels:  $x_0$  and  $x_1$  are indicated by horizontal lines below the columns,  $x_2$  by a vertical line to the right of the first two rows, and  $x_3$  by a vertical line to the right of the last two rows.

4 variables



# Mapas de Karnaugh



5 variables



# Mapas de Karnaugh

	$x_1x_0$			
$x_3x_2$	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

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

$x_5 = 0$

	00	01	11	10
	32	33	35	34
	36	37	39	38
	44	45	47	46
	40	41	43	42

	00	01	11	10
	48	49	51	50
	52	53	55	54
	60	61	63	62
	56	57	59	58

$x_5 = 1$

$x_4 = 0$

$x_4 = 1$

¡6 variables!

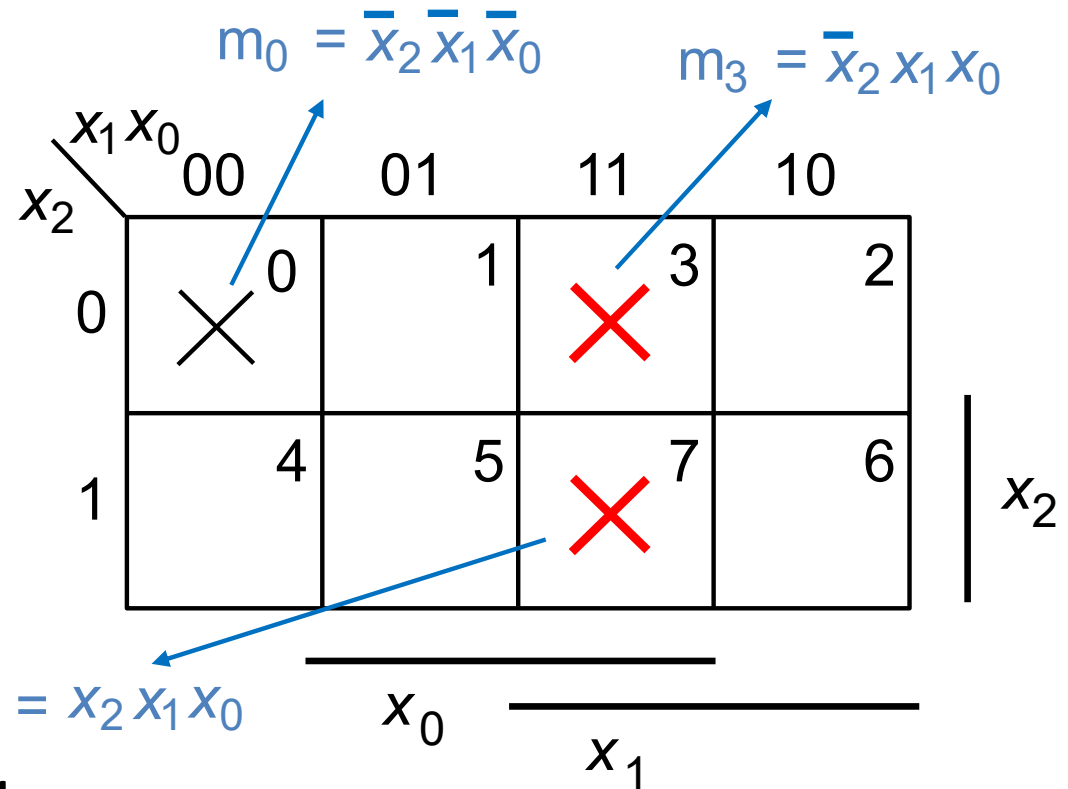
¡6 variables!



# Mapas de Karnaugh

- Para obtener el mapa de Karnaugh de una SPC basta con marcar los mintérminos que la forman.

$$f(x_2, x_1, x_0) = \sum m(0, 3, 7)$$



- $m_3$  y  $m_7$  son adyacentes luego:

$$m_3 + m_7 = \overline{x_2}x_1x_0 + x_2x_1x_0 = (\overline{x_2} + x_2)x_1x_0 = x_1x_0$$



# Simplificación por MK

- Procedimiento de simplificación:
  - Construir el mapa de Karnaugh de la FC
  - Cubrir todos los minterminos con el menor número posible de rectángulos de tamaño en casillas múltiplo de 2 (1, 2, 4, 8, 16...)
    - Cada rectángulo se corresponde con un término producto, más simple conforme mayor es el rectángulo.
    - La EC simplificada será la suma de los términos producto obtenidos.
  - Si hay *don't cares*, pueden tomarse como 0 ó 1 según convenga



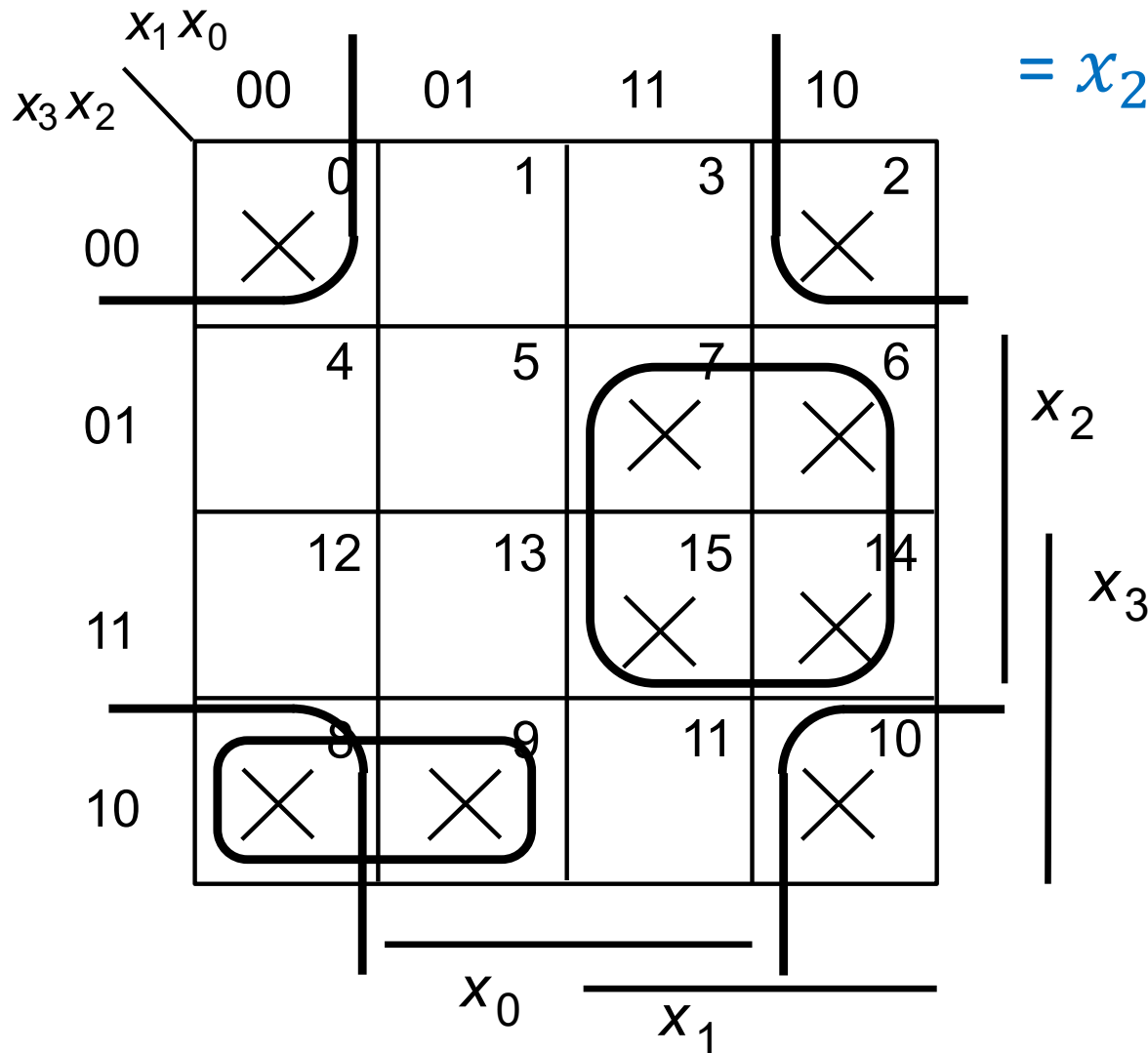
# Simplificación por MK

- Estrategias:
  - Los rectángulos deberán ser lo mayor posible, así los términos producto tendrán un menor número de literales.
  - Si es necesario, una misma casilla puede ser cubierta varias veces por distintos rectángulos (para que éstos puedan ser más grandes).
  - Si una casilla puede cubrirse de distintos modos, empezar cubriendo aquellas que solo puedan hacerlo de una manera.
  - Las casillas frontera pueden cubrirse junto con las del otro extremo.
  - Las casillas de las esquinas pueden cubrirse todas juntas.



# Simplificación por MK

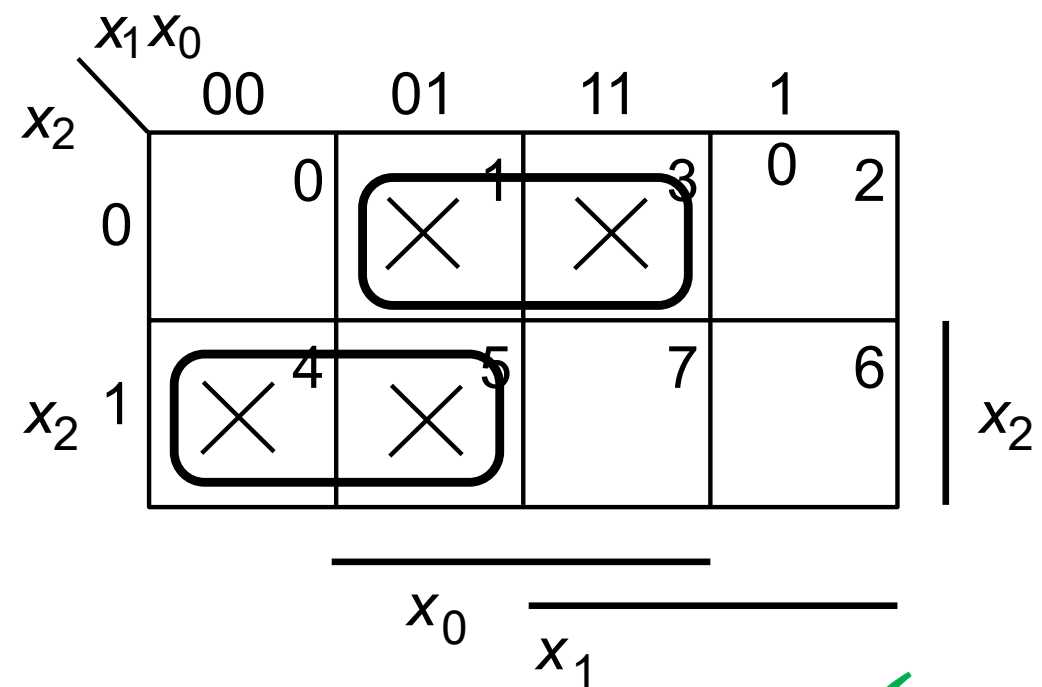
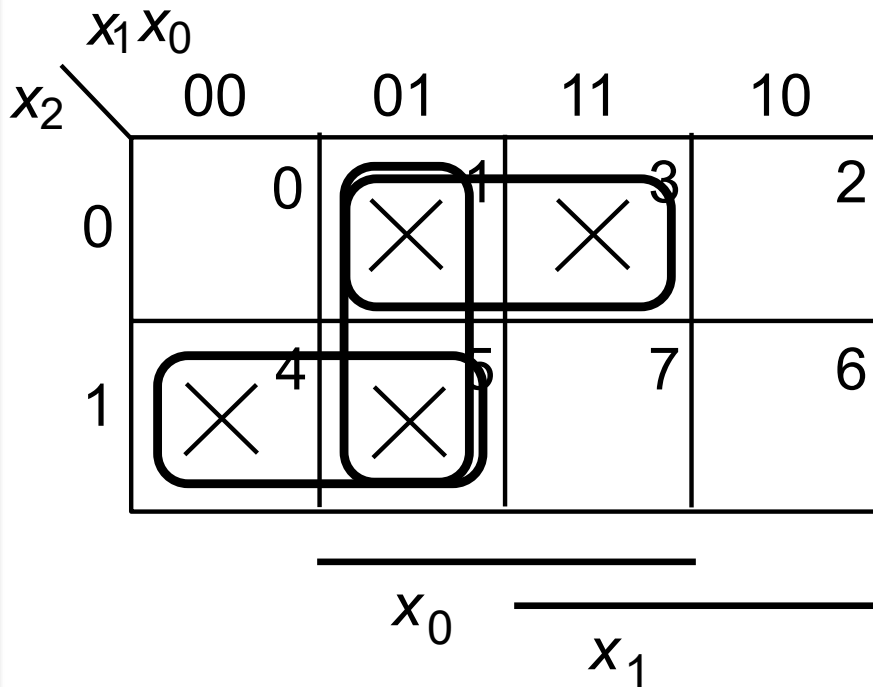
$$f(x_3, x_2, x_1, x_0) = \sum m(0, 2, 6, 7, 8, 9, 10, 14, 15)$$
$$= x_2 x_1 + \overline{x_2} \overline{x_0} + x_3 \overline{x_2} \overline{x_1}$$





# Simplificación por MK

$$f(x_2, x_1, x_0) = \Sigma m(1, 3, 4, 5)$$



$$= \overline{x_1}x_0 + x_2\overline{x_1} + \overline{x_2}x_0 \quad \times$$

$$= x_2\overline{x_1} + \overline{x_2}x_0 \quad \checkmark$$



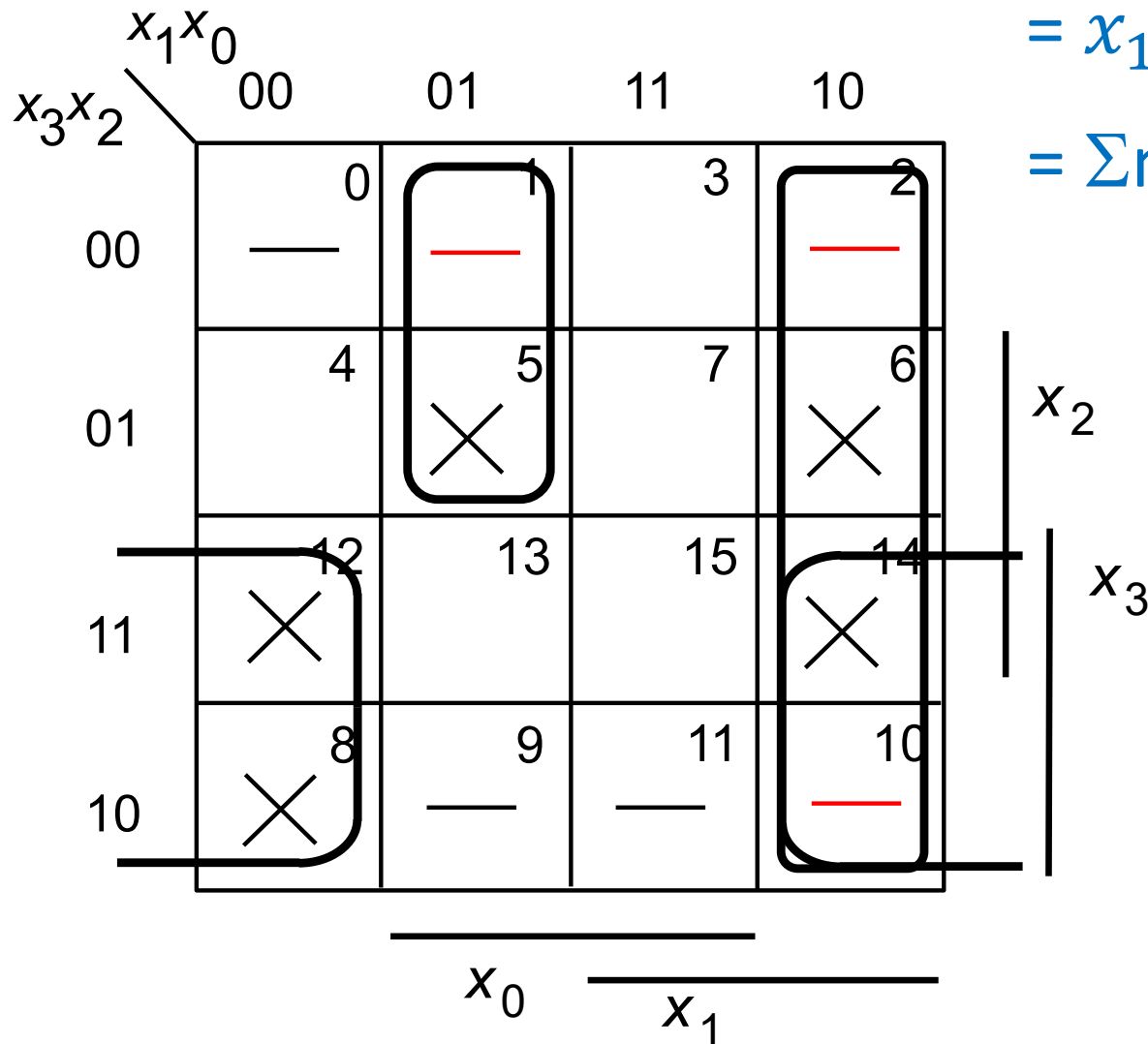


# Simplificación por MK

$$f(x_3, x_2, x_1, x_0) = \sum m (5, 6, 8, 12, 14) + \sum d (0, 1, 2, 9, 10, 11)$$

$$= x_1 \overline{x_0} + x_3 \overline{x_0} + \overline{x_3} \overline{x_1} x_0$$

$$= \sum m (1, 2, 5, 6, 8, 10, 12, 14)$$

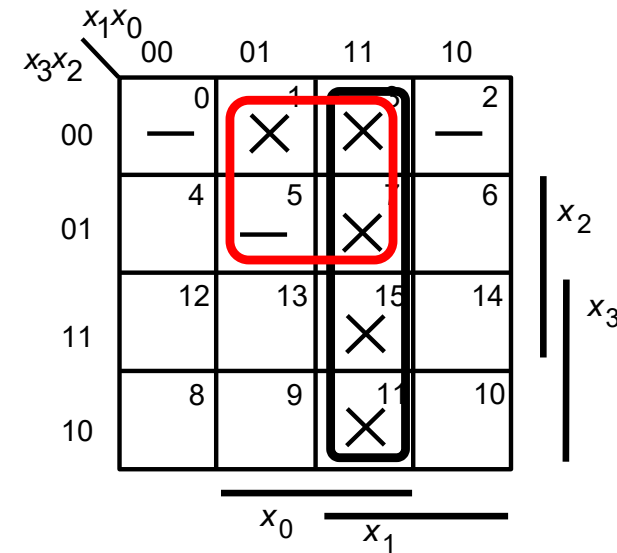
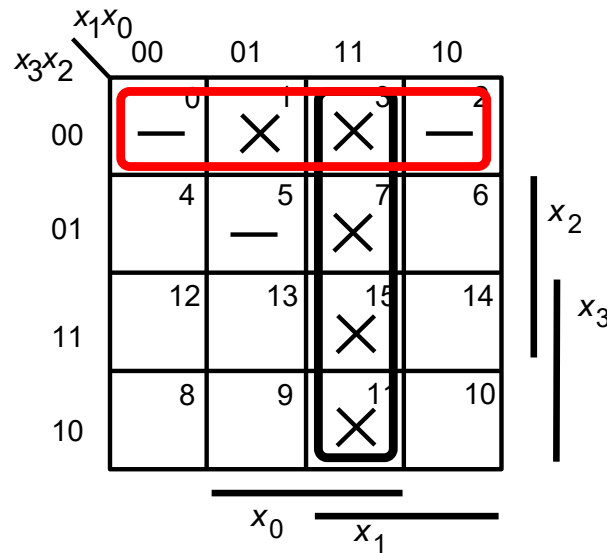




# Equivalencia lógica vs. algebraica

- Las distintas EC obtenidas al simplificar una misma FC incompletamente especificada pueden no ser equivalentes entre sí.

$$f(x_3, x_2, x_1, x_0) = \sum m (1, 3, 7, 11, 15) + \sum d (0, 2, 5)$$



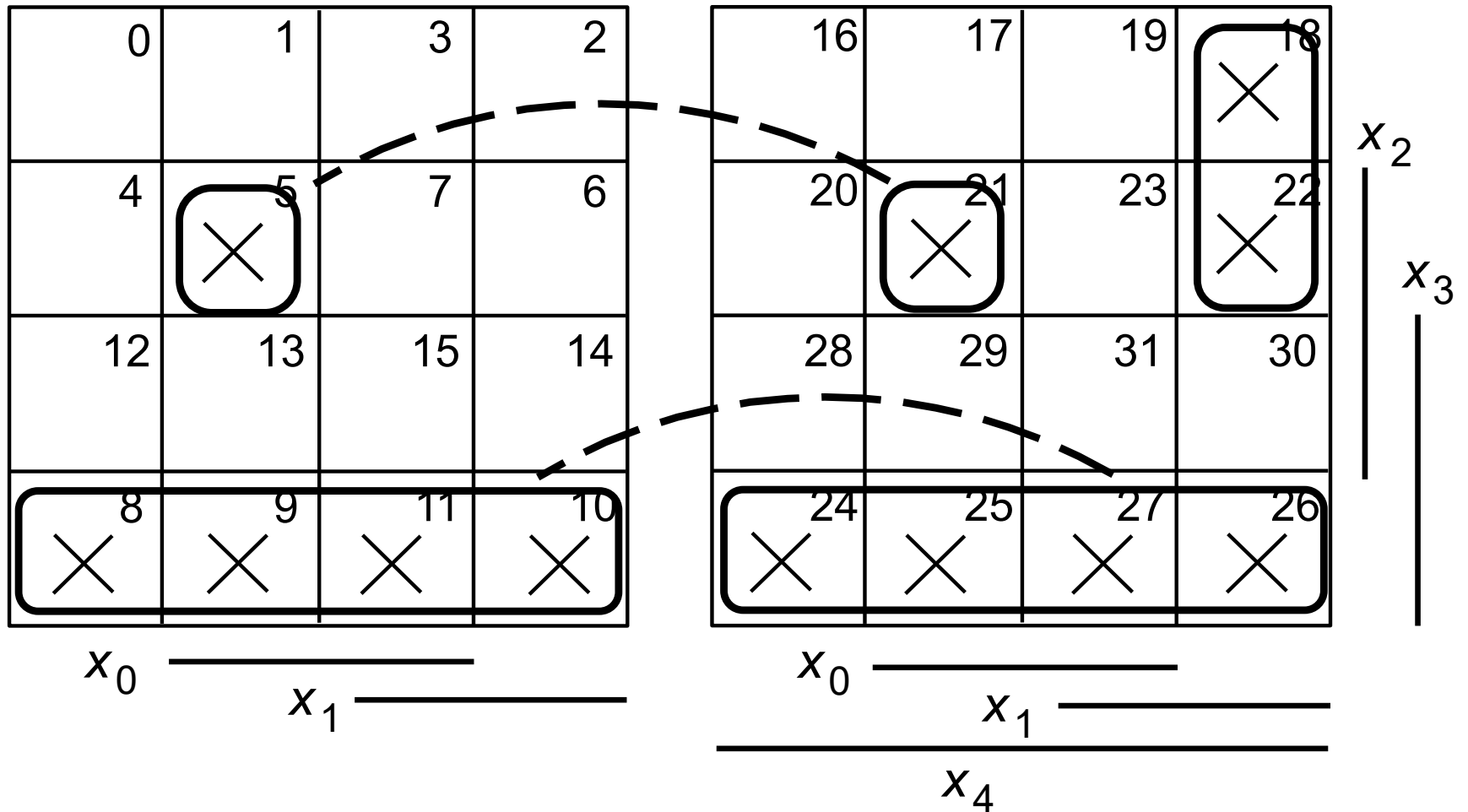
$$f_A = x_1x_0 + \overline{x_3}\overline{x_2} = \sum m (0, 1, 2, 3, 7, 11, 15) \quad f_B = x_1x_0 + \overline{x_3}x_0 = \sum m (1, 3, 5, 7, 11, 15)$$

- Dos EC son equivalentes algebraicamente si representan a la misma FC en todos los puntos del dominio.
- Dos EC son equivalentes lógicamente si representan a la misma FC en todos los puntos del dominio para los que está definida.



# Simplificación por MK

$$f(x_4, x_3, x_2, x_1, x_0) = \sum m (5, 8, 9, 10, 11, 18, 21, 22, 24, 25, 26, 27)$$
$$= x_3 \overline{x_2} + \overline{x_3} x_2 \overline{x_1} x_0 + x_4 \overline{x_3} x_1 \overline{x_0}$$

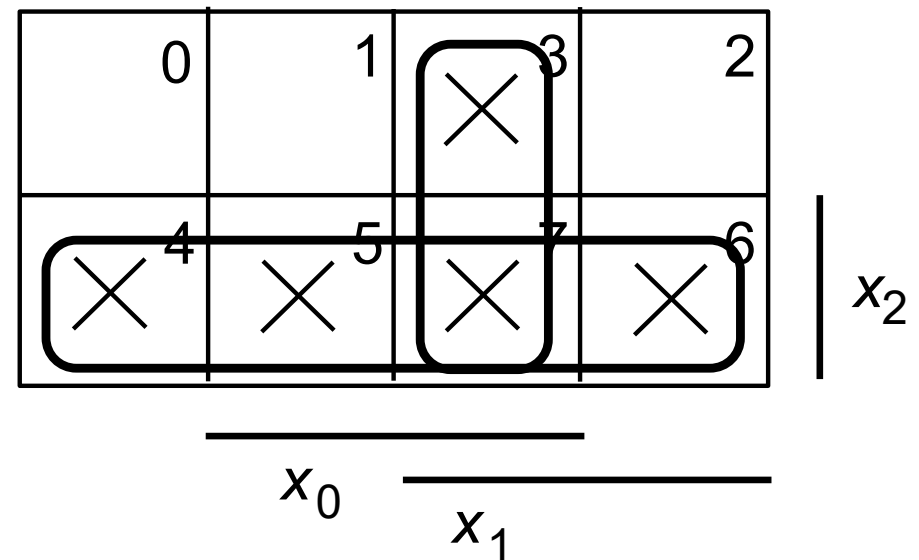




# Otros usos de los MK

- Adicionalmente los mapas de Karnaugh pueden usarse para obtener:
  - La SPC de una EC (en forma de suma de productos).
  - Una EC mínima equivalente a una EC dada.

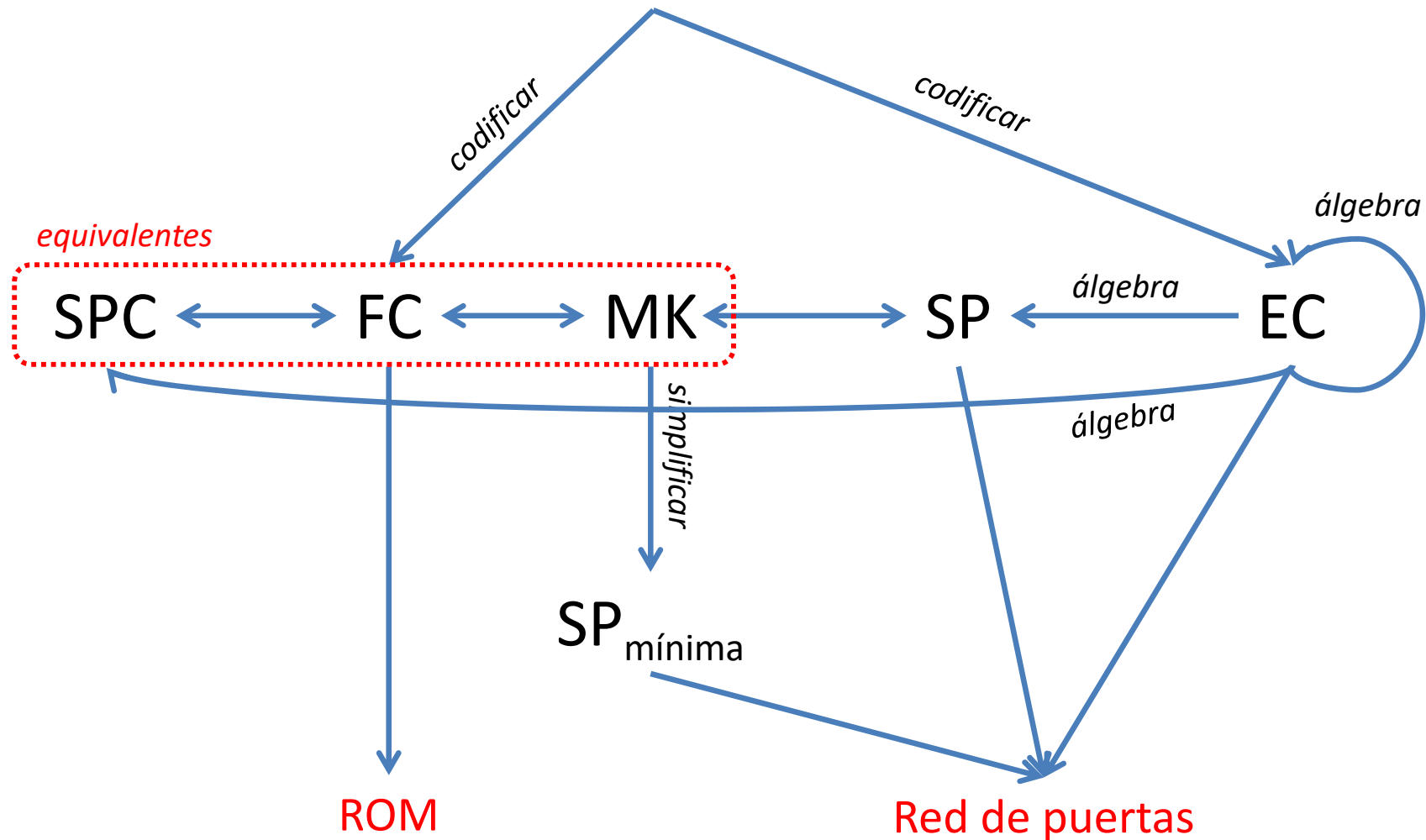
$$\begin{aligned} & x_2 \overline{(x_1 x_0)} + x_1 x_0 \\ &= x_2 (\overline{x_1} + \overline{x_0}) + x_1 x_0 \\ &= x_2 \overline{x_1} + x_2 \overline{x_0} + x_1 x_0 \\ &= \sum m(3, 4, 5, 6, 7) \\ &= x_2 + x_1 x_0 \end{aligned}$$





# Rescapitulación

Especificación de alto nivel



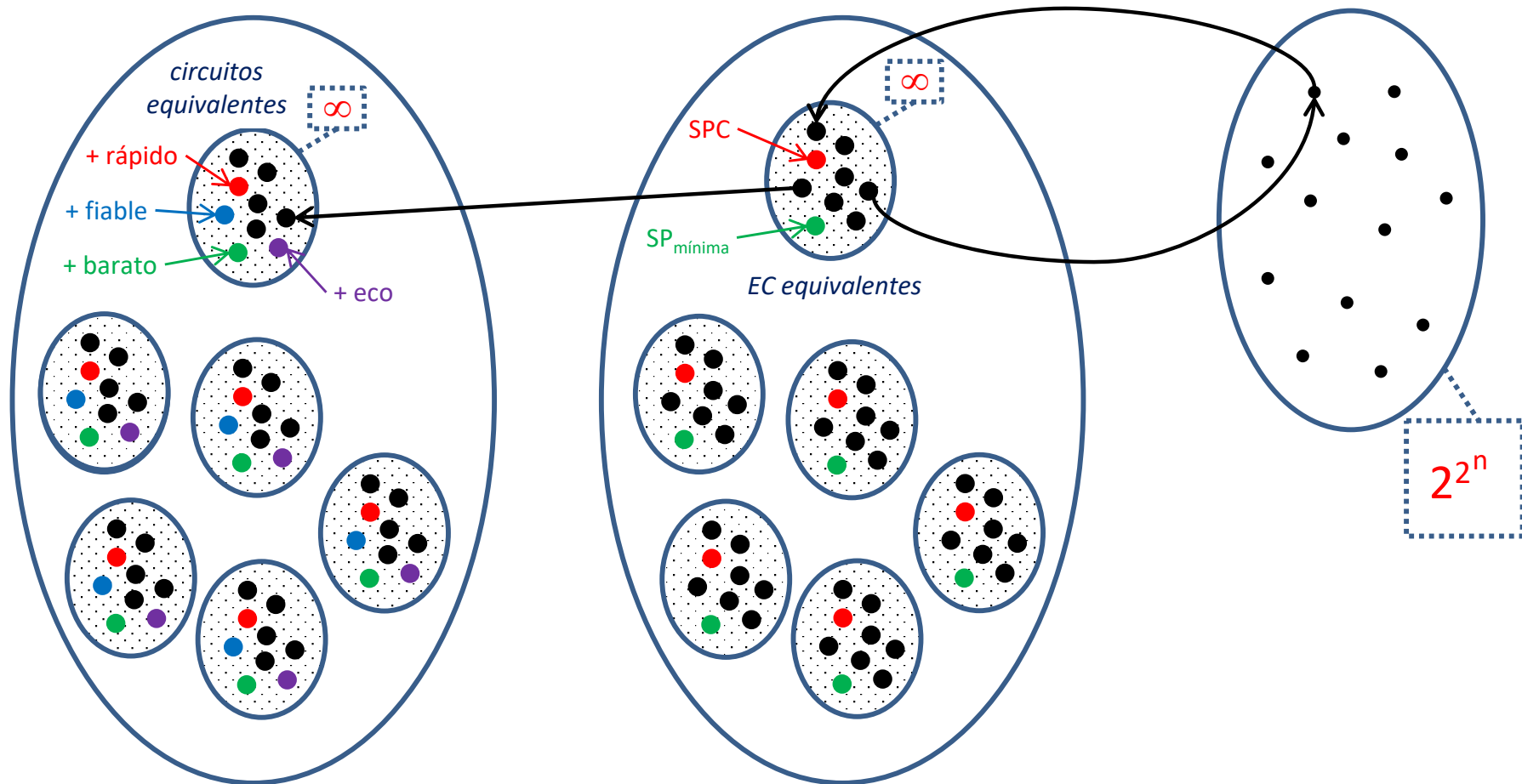


# Panorama en abstracto

Circuitos de  $n$  entradas

EC de  $n$  variables

FC de  $n$  variables



# Acerca de *Creative Commons*



## ■ Licencia CC (**Creative Commons**)

- Ofrece algunos derechos a terceras personas bajo ciertas condiciones. Este documento tiene establecidas las siguientes:



**Reconocimiento** (*Attribution*):

En cualquier explotación de la obra autorizada por la licencia hará falta reconocer la autoría.



**No comercial** (*Non commercial*):

La explotación de la obra queda limitada a usos no comerciales.



**Compartir igual** (*Share alike*):

La explotación autorizada incluye la creación de obras derivadas siempre que mantengan la misma licencia al ser divulgadas.

**Más información:** <https://creativecommons.org/licenses/by-nc-sa/4.0/>