Tema 2. Funciones Lógicas

- Algebra de Conmutación.
- Minimización de funciones Lógicas.
- Introducción al VHDL.

Minimización de Funciones Lógicas

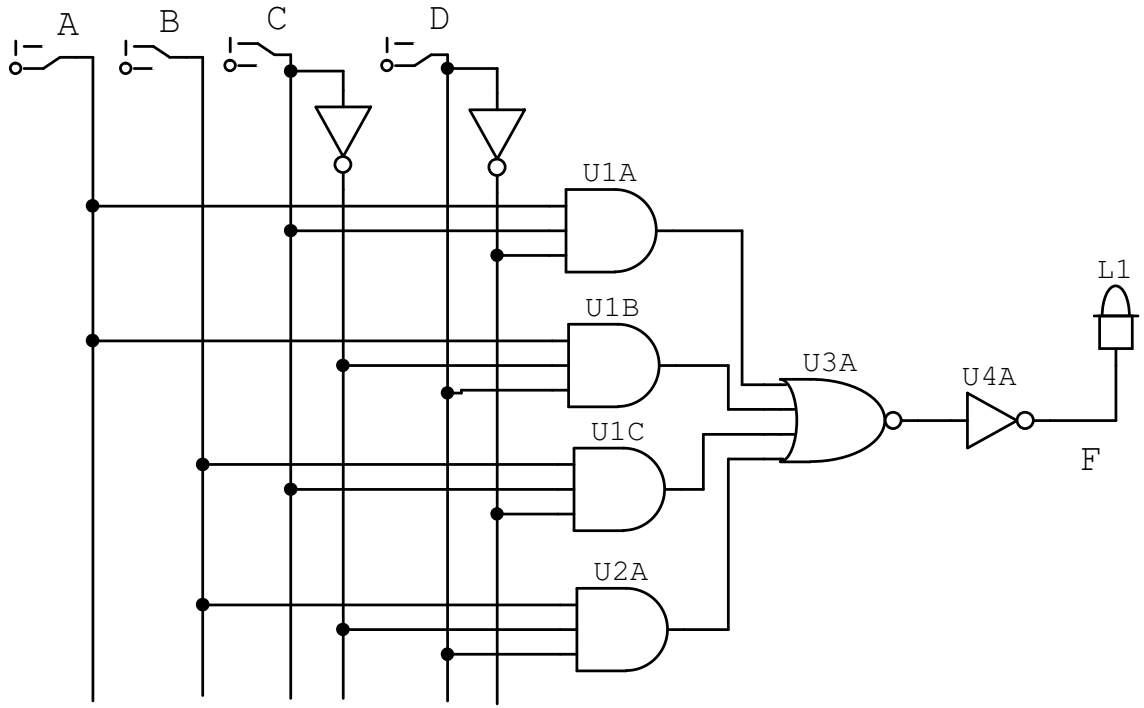
- Minimización en dos niveles. Mapas de Karnaugh de 3 y 4 variables. K-cubos. Definición de una función mínima en dos niveles.
- Implicantes primos. Implicantes primos esenciales. Minimización en dos niveles mediante el mapa de Karnaugh en problemas lógicos completa e incompletamente especificados.
- Mapas de Karnaugh de 5 y 6 variables. Minimización multifunción. Minimización en dos niveles mediante el mapa de Karnaugh en problemas lógicos completa e incompletamente especificados.
- Minimización algorítmica en dos niveles (una y varias salidas) y multinivel.

- Minimización en dos niveles de una función lógica. Encontrar una forma SOP o POS mínima.

  Extensión a problemas de varias funciones

  Objetivo básico: encontrar formas lógicas con el menor número de términos productos (sumas) y el menor número de literales por término producto (sumas).

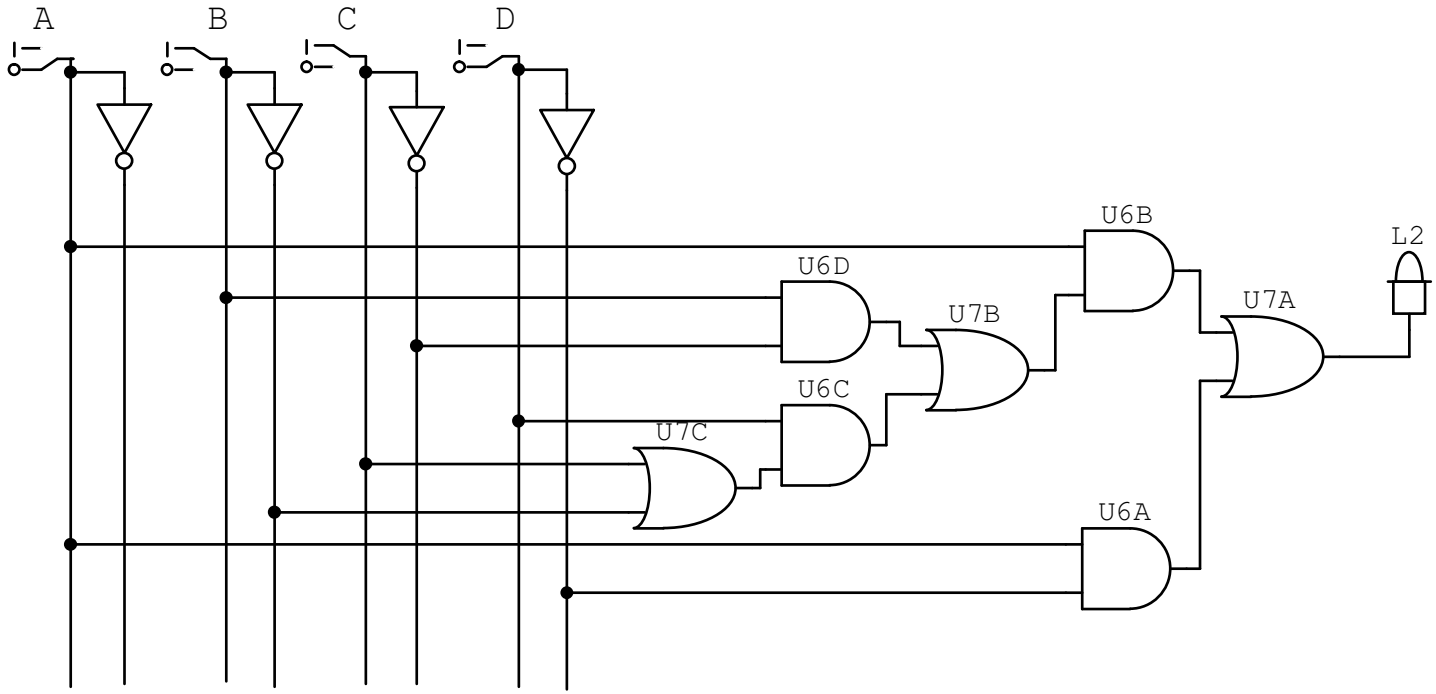
  $F(A, B, C, D) = A\,C\,\overline{D} + A\,\overline{C}\,D + B\,C\,\overline{D} + B\,\overline{C}\,D$



- Síntesis multinivel. Realizar una serie de operaciones sobre funciones lógicas que encuentren una buena forma factorizada (varios niveles AND/OR/AND/OR…).

  Objetivo básico: reducir el número de literales de la expresión lógica.

  $F(A, B, C, D) = \overline{A} [B \overline{C} + D (C + \overline{B})] + A \overline{D}$



Minimización en dos niveles

- El paso de funciones canónicas a funciones estándar mediante álgebra de conmutación no garantiza encontrar una solución mínima si no se usa un método algorítmico.

  $F(x,y,z) = x\,\overline{z} + x\,\overline{y}\,z + \overline{x}\,z + \overline{x}\,y\,\overline{z} = x\,\overline{z} + x\,\overline{y} + \overline{x}\,z + \overline{x}\,y$

  Sin embargo una solución mínima es:

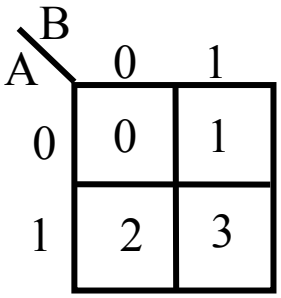
  $F(x,y,z) = y\,\overline{z} + x\,\overline{y} + \overline{x}\,z$

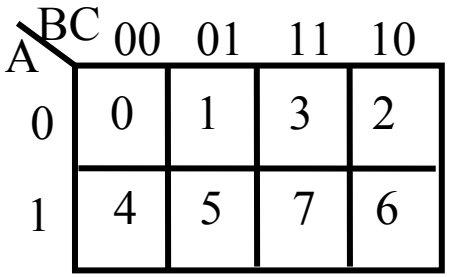
  La aplicación de los teoremas “a mano” no permite ver relaciones de simplificación “ocultas”. A veces sería necesario expandir la función para luego simplificarla. Para ver bien las relaciones se usan métodos gráficos.

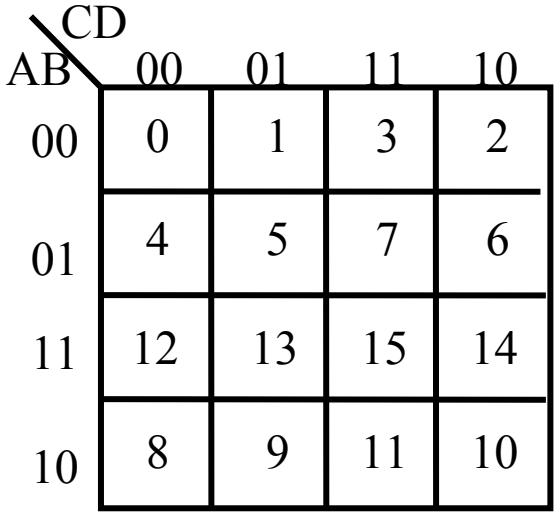
Mapa de Karnaugh

- El Mapa de Karnaugh es un método para observar una tabla de verdad de forma gráfica y observar la relaciones de adyacencia entre los 1s ó 0s de la tabla. Cada grupo de 1, 2, 4, 8, 16, …, 1s (0s) de la tabla que formen un cuadrado o un rectángulo en el Mapa son un **cubo** de la función y corresponden a un término producto (término suma).

Cada casilla está marcada en notación decimal.





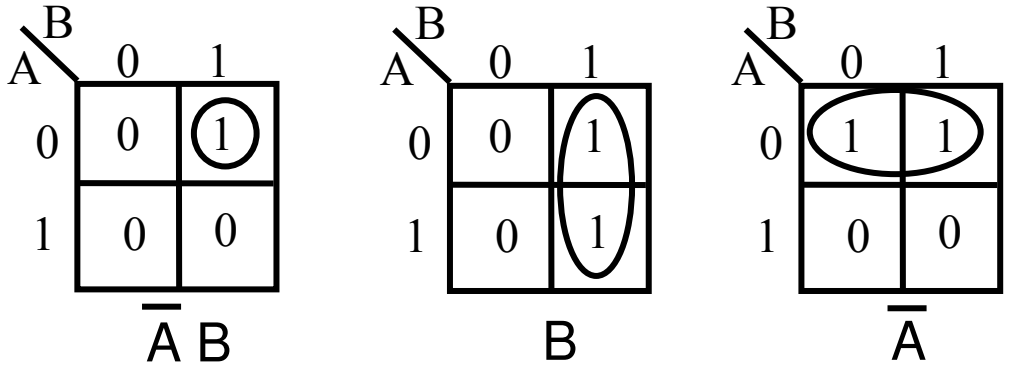


- Los valores en los entradas de los Mapas de Karnaugh se sitúan de forma que entre cada casilla adyacente del Mapa de Karnaugh (izquierda, derecha, arriba, abajo) tengan distancia de Hamming 1. Hay que considerar que los bordes están unidos y que hay adyacencia entre las filas de abajo y arriba, y las columnas derecha e izquierda.
- Cada casilla es adyacente a tantas casillas como entradas haya en la función.
- Los cubos o agrupaciones de 1s ó 0s de la función son de un orden determinado (k-cubos):

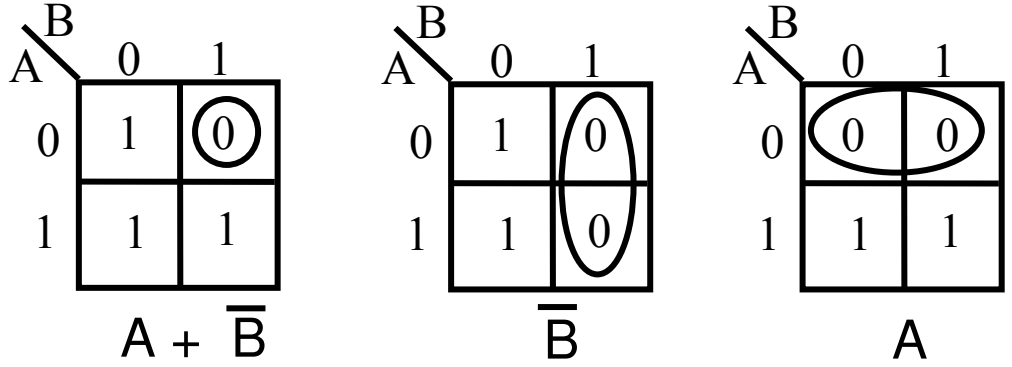
  1 casilla: 0-cubo; 2 casillas 1-cubo; 4 casillas 2-cubo; etc.

  El número de literales de un k-cubo en función de N entradas es N-k.

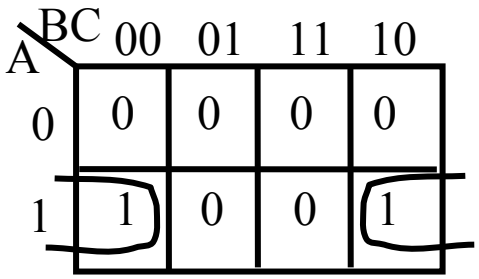
Siguiendo la notación de las formas canónicas, al tomar los 1s de se forman términos productos:
Si X está siempre a 1 => literal X; si X está siempre a 0 => literal $\overline{X}$
Si X toma valores 0 y 1 => no se utiliza X



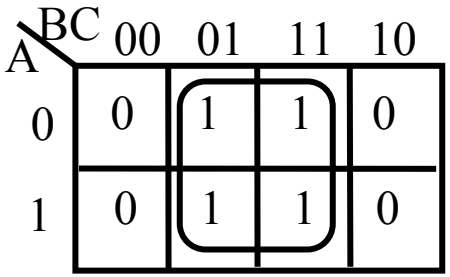
Siguiendo la notación de las formas canónicas, al tomar los 0s de se forman términos sumas:
Si X está siempre a 1 => literal $\overline{X}$; si X está siempre a 0 => literal X
Si X toma valores 0 y 1 => no se utiliza X



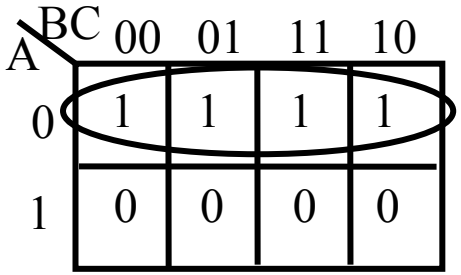
En Mapas de 3 ó 4 entradas los bordes están unidos y se pueden formar cubos entre las filas de abajo y arriba, y las columnas derecha e izquierda



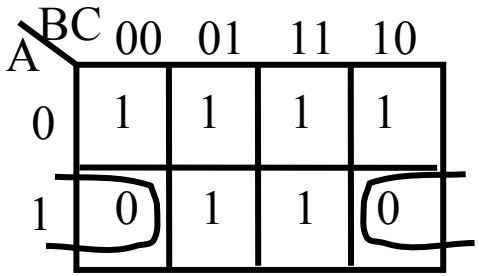
$A\,\overline{C}$



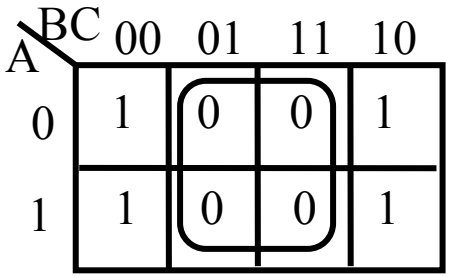
C



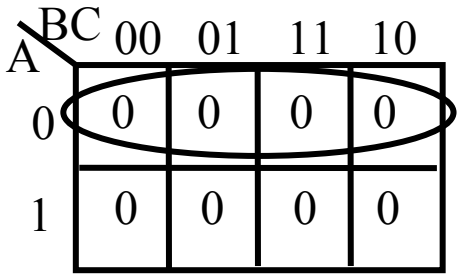
$\overline{A}$



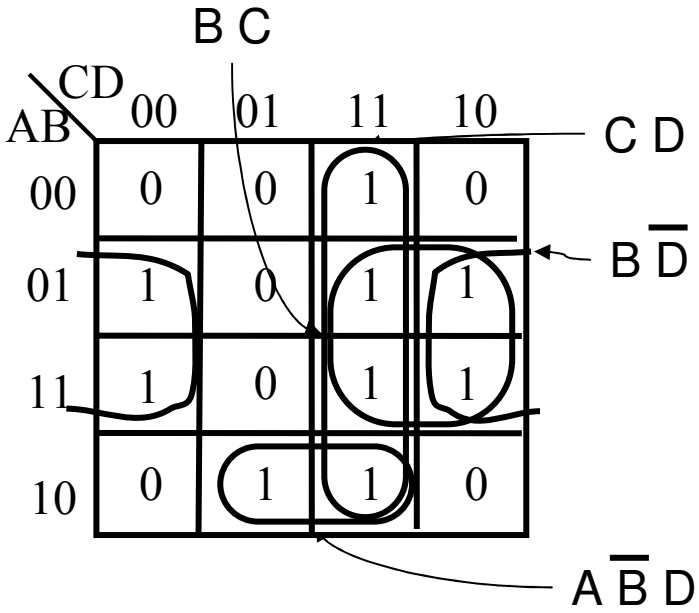
$\overline{A} + C$

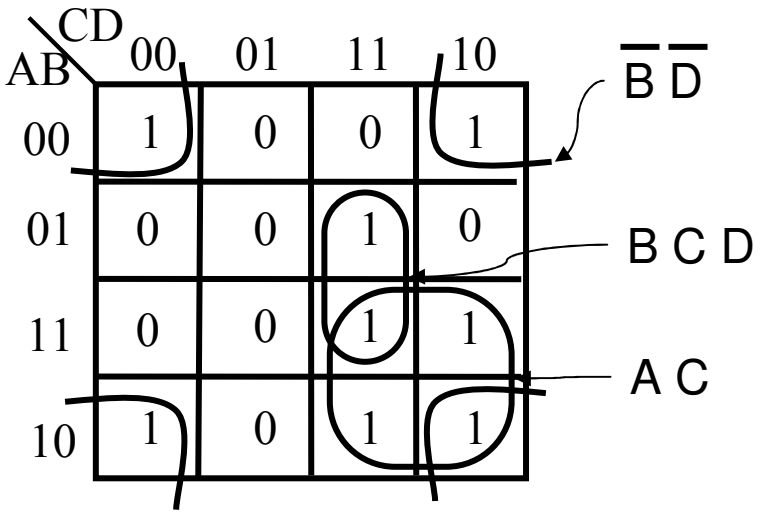


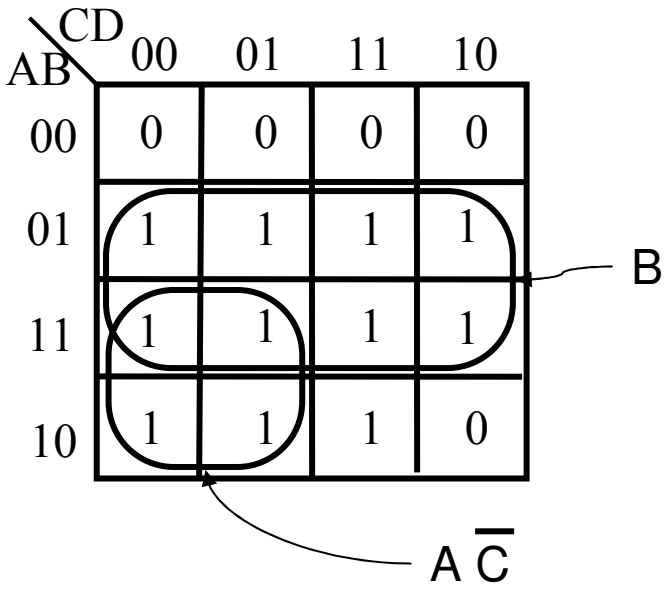
$\overline{C}$

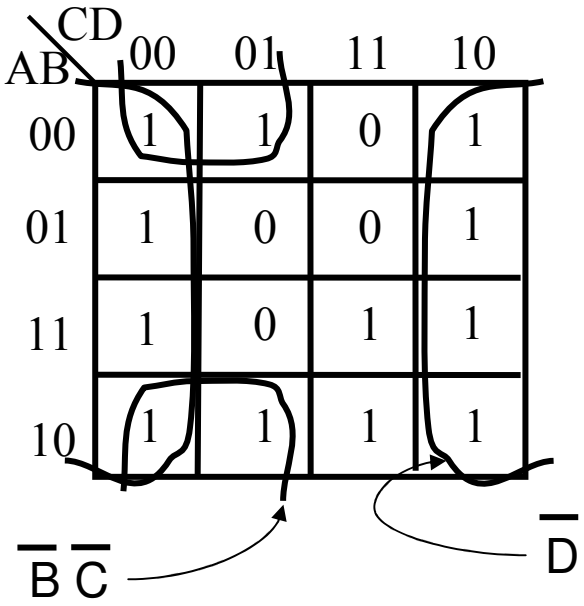


A


B C
CD
AB
00 01 11 10
0 0 1 0
1 0 1 1
1 0 1 1
0 1 1 0
C D
B $\overline{D}$
A $\overline{B}$ D


CD
AB
00 01 11 10
1 0 0 1
0 0 1 0
0 0 1 1
1 0 1 1
$\overline{B}$ $\overline{D}$
B C D
A C


CD
AB
00 01 11 10
0 0 0 0
1 1 1 1
1 1 1 1
1 1 1 0
B
A $\overline{C}$


CD
AB
00 01 11 10
1 1 0 1
1 0 0 1
1 0 1 1
1 1 1 1
$\overline{B}$ $\overline{C}$
$\overline{D}$

Minimización mediante el Mapa de Karnaugh

- Objetivo: generar una expresión mínima en dos niveles para una función lógica.

- Una expresión de tipo SOP (POS) es mínima cuando:

  1. No existe ninguna otra expresión equivalente que incluya menos términos productos (sumas).

  2. No hay otra expresión equivalente que conste del mismo número de términos productos (sumas) y que tenga un menor número de literales.

Minimización mediante el Mapa de Karnaugh

- Implicante primo de una función F es un cubo de F que no está incluido en ningún cubo de mayor orden. El uso de implicantes primos en lugar de cubos garantiza que se cumple la condición 2.

- Un proceso de minimización en dos niveles incluye estos dos procesos:

  1. Determinación de todos los implicantes primos (no es absolutamente necesario).

  2. Selección del menor número de implicantes primos que cubre al menos una vez a todos los 1s (o 0s) de la función F. Si hay varias posibilidades tomar los que sean cubos de mayor orden.

Minimización mediante el Mapa de Karnaugh

- Implicante primo esencial: único implicante primo P que cubre a un minterm (o maxterm) determinado de la función.

  Por ello, la expresión de F en forma SOP (POS) debe incluir obligatoriamente a P como término producto (suma).

  1ª regla para minimización: Localizar las casillas cubiertas por un único implicante primo P. P es esencial, forma parte de la función mínima, y todas las casillas de P quedan cubiertas, por lo que no hay que preocuparse más de ellas.

Minimización mediante el Mapa de Karnaugh

2ª. Cuando una casilla puede ser cubierta por varios implicantes primos, escoger un implicante primo de orden k máximo entre ellos, que cubra las mismas casillas que quedan sin cubrir (y si se puede alguna más), que todos los otros implicantes. Si hay dos implicantes del mismo tamaño que cumplan esta condición da igual escoger uno u otro.

3ª regla para minimización: Buscar la casilla cubierta por menos implicantes primos y probar con cada uno de ellos aplicando de nuevo las tres reglas. Quedarse con la solución más pequeña generada.

Minimización mediante el Mapa de Karnaugh

- Funciones incompletamente especificados

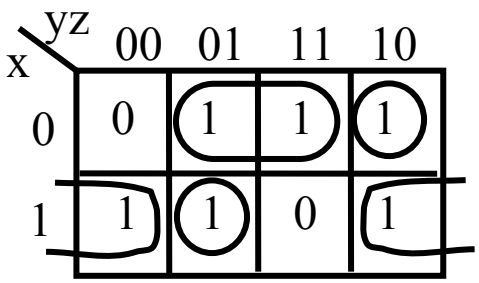
  Generación de implicantes primos: utilizar los "don't care" como si fuesen 1's en la forma SOP y como si fuesen 0's en la forma POS para encontrar implicantes primos.

  Selección de implicantes primos: cubrir todos los 1’s (SOP) o los 0’s (POS) de la función. No hace falta cubrir los "don't care".

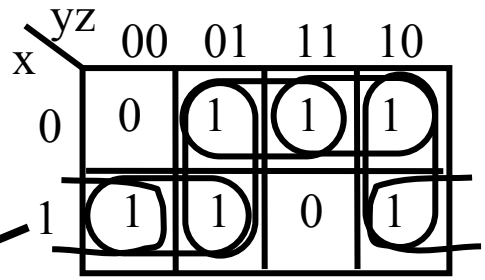
Minimización de

$F(x,y,z) = x\,\overline{z} + x\,\overline{y}\,z + \overline{x}\,z + \overline{x}\,y\,\overline{z}$

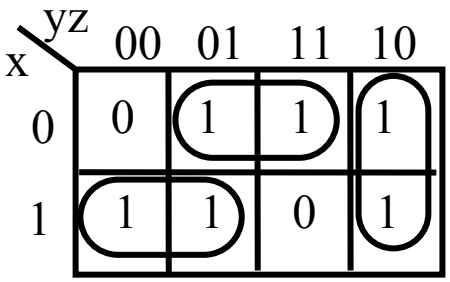
4 Cubos



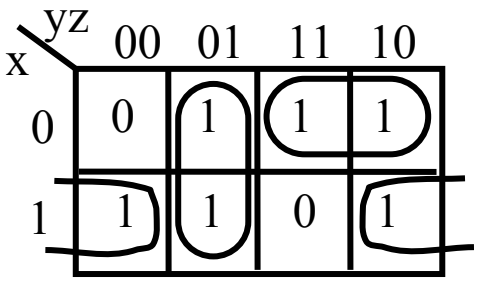
6 Implicantes primos



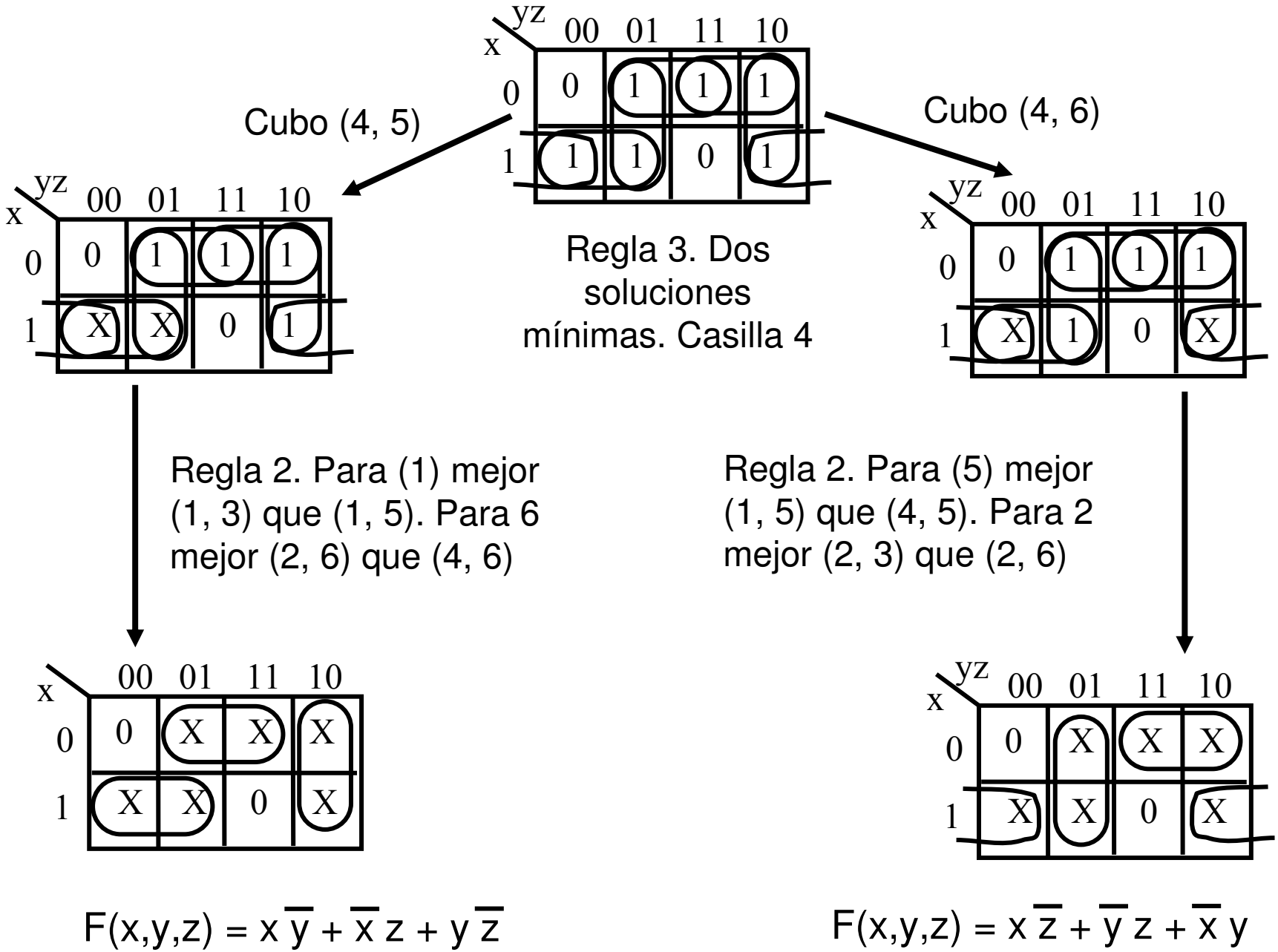
Regla 3. Dos soluciones mínimas



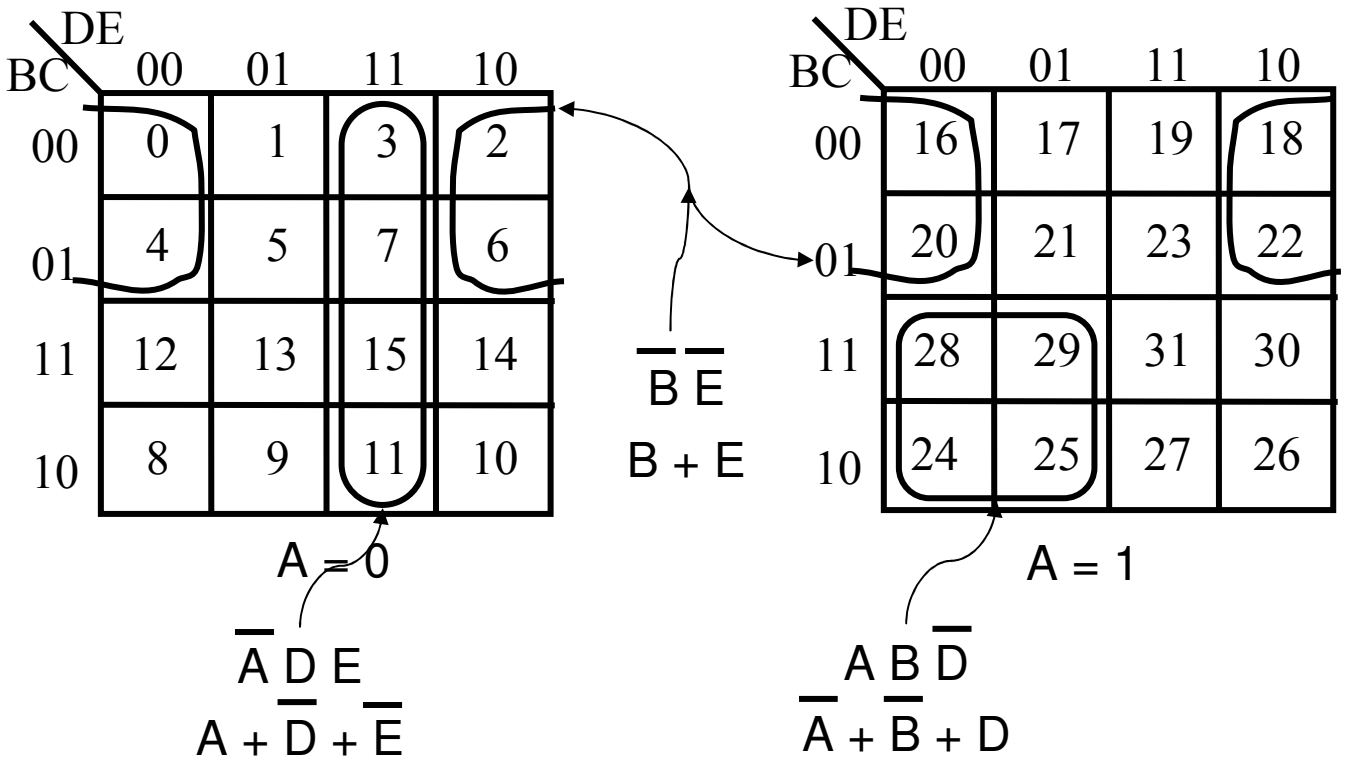
$F(x,y,z) = x\,\overline{y} + \overline{x}\,z + y\,\overline{z}$



$F(x,y,z) = x\,\overline{z} + \overline{y}\,z + \overline{x}\,y$


yz
x
00 01 11 10
0 0 1 1 1
1 1 1 0 1
Cubo (4, 5)
Cubo (4, 6)
Regla 3. Dos soluciones mínimas. Casilla 4
0 0 1 1 1
1 X X 0 1
0 0 1 1 1
1 X 1 0 X
Regla 2. Para (1) mejor (1, 3) que (1, 5). Para 6 mejor (2, 6) que (4, 6)
Regla 2. Para (5) mejor (1, 5) que (4, 5). Para 2 mejor (2, 3) que (2, 6)
0 0 X X X
1 X X 0 X
0 0 X X X
1 X X 0 X
F(x,y,z) = x $\overline{y}$ + $\overline{x}$ z + y $\overline{z}$
F(x,y,z) = x $\overline{z}$ + $\overline{y}$ z + $\overline{x}$ y

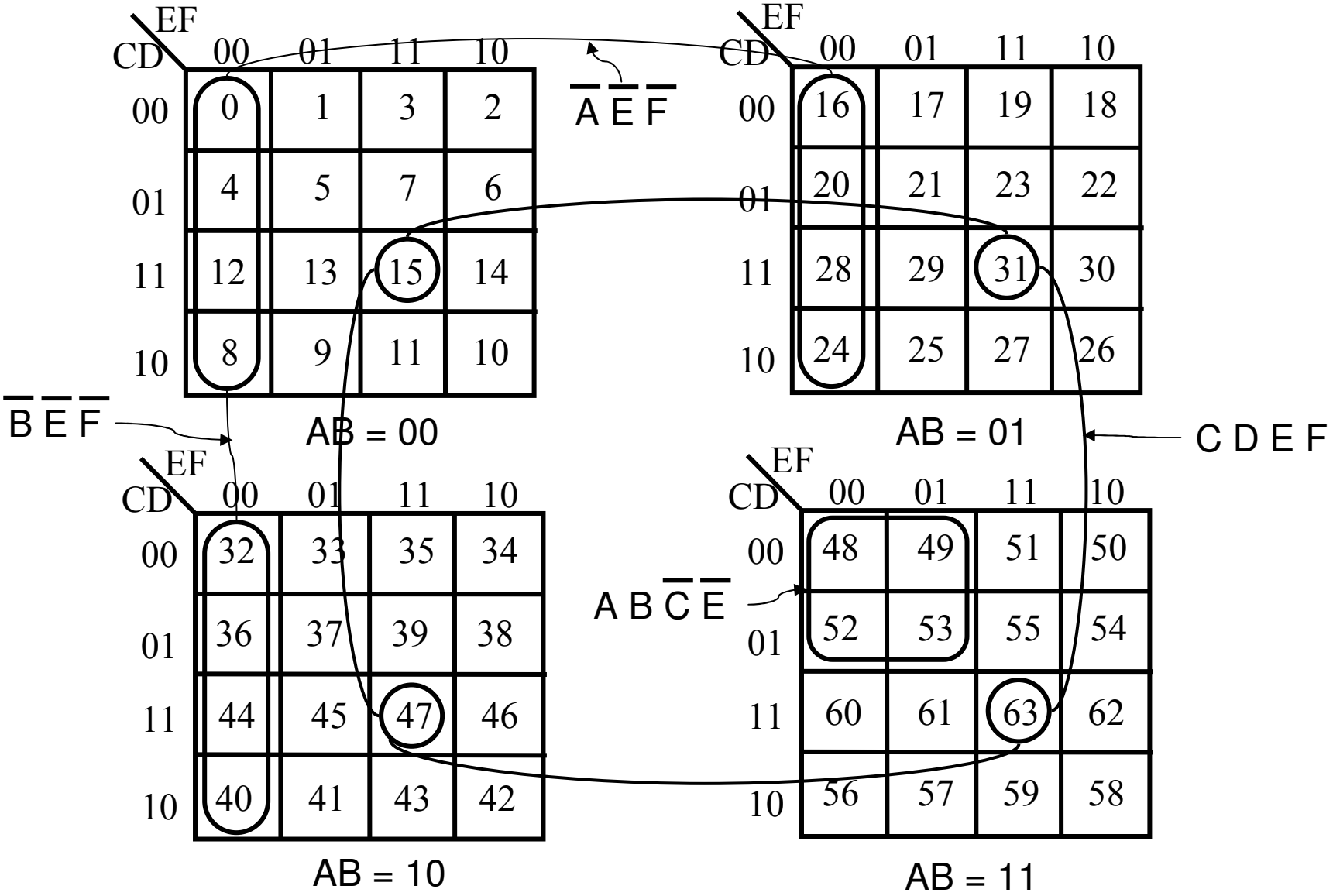
Mapas de Karnaugh de 5 variables F(A,B,C,D,E)



Tres tipos de implicantes primos:

- Sólo en el Mapa A = 0,
- Sólo en el Mapa A = 1
- En los dos Mapas. La expresión lógica no depende de A.

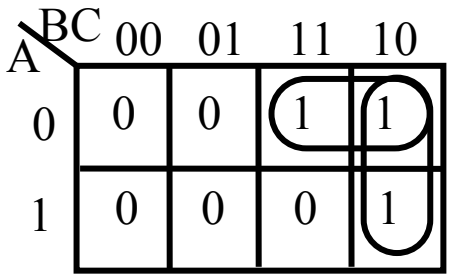
Mapas de Karnaugh de 6 variables F(A,B,C,D,E,F)



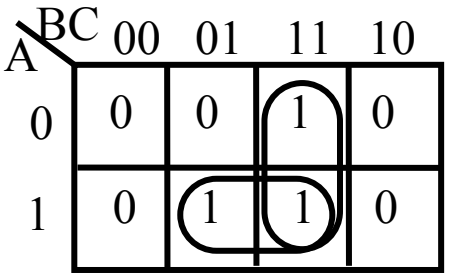
Minimización de varias funciones lógicas

F1 (A, B, C) = $\sum$ (2, 3, 6)
F2 (A, B, C) = $\sum$ (3, 5, 7)

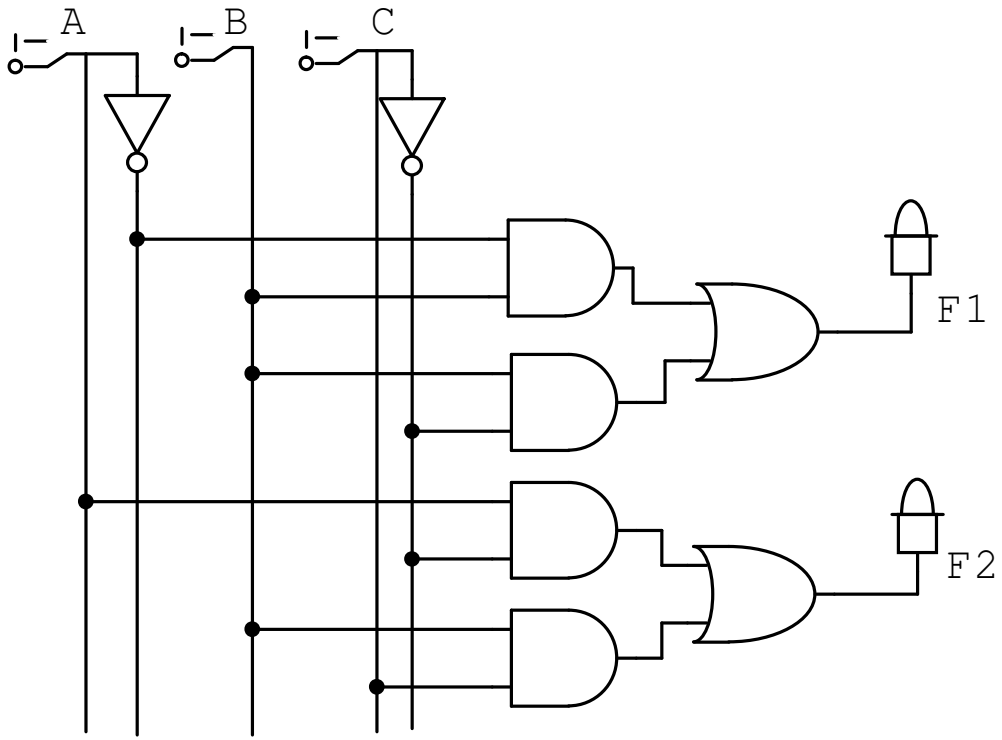
Minimización por separado



$F1 = \overline{A}\,B + B\,\overline{C}$



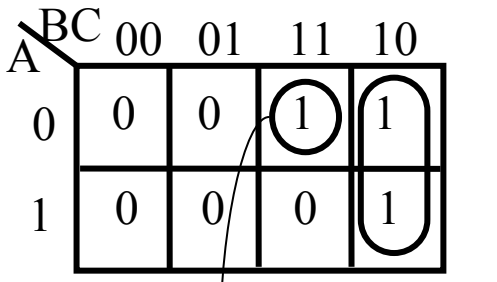
F2 = A C + B C



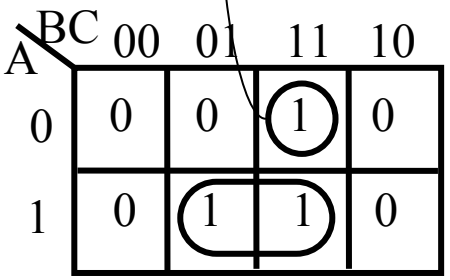
Minimización de varias funciones lógicas

F1 (A, B, C) = $\sum$ (2, 3, 6)
F2(A, B, C) = $\sum$ (3, 6, 7)

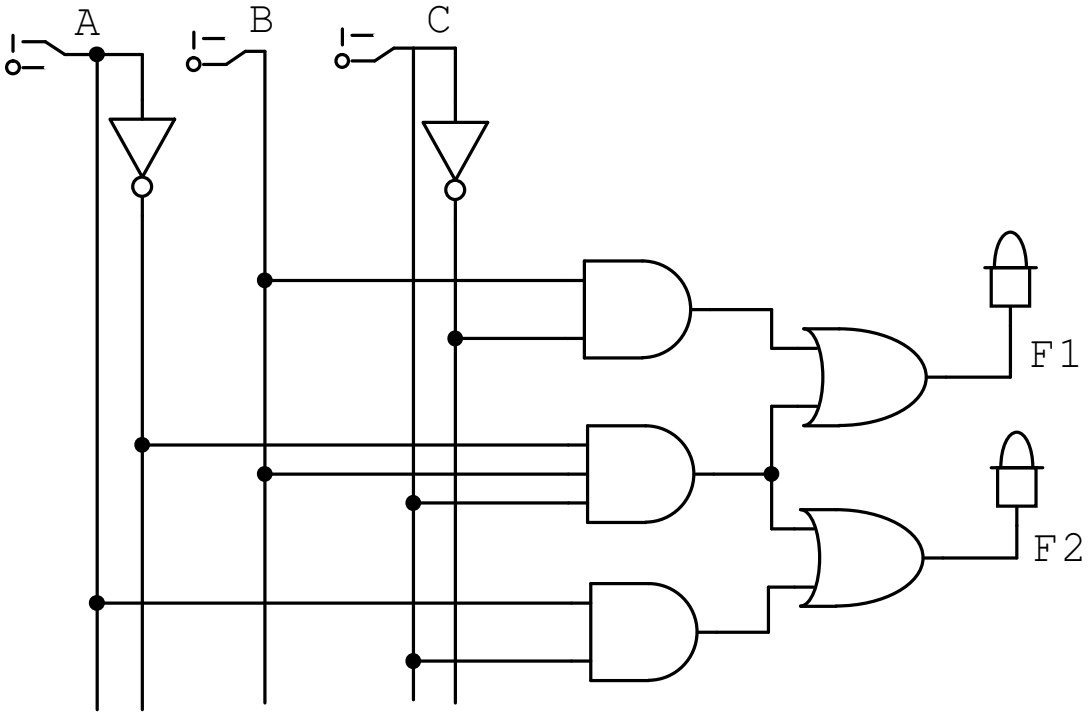
Minimización conjunta. El circuito se reduce. Hay que reducir el número de implicantes de todas las funciones a la vez.



$F1 = \overline{A}\, B\, C + B\, \overline{C}$



$F2 = \overline{A}\, B\, C + A\, C$



Métodos algorítmicos para síntesis lógica

- Cuando se plantean problemas lógicos muy complejos no sólo la minimización sino casi la resolución a mano se hace inviable. Se requieren herramientas de ayuda al diseño (CAD) que generen automáticamente los resultados de síntesis de las funciones lógicas.
- Existen algoritmos aplicados a la síntesis de funciones en dos niveles de una y varias salidas, y a la síntesis multinivel, la mayoría de ellos demasiado complejos para ser explicados en un tiempo razonable.

Método Quine-McCluskey

- Algoritmo de minimización en dos niveles. Las variantes de este algoritmo tiene un límite de unas 20 entradas y/o salidas para un tiempo de cómputo razonable.
- Se divide en dos partes generación de implicantes primos y selección de implicantes primos. La descripción corresponde a una función de N entradas y una salida aunque puede adaptarse para varias salidas.

- Generación de implicantes primos: se parte de los 0-cubos (1s ó 0s, y “don’t cares”) representados en binario y ordenados por el número i de 1s en su codificación binaria G0[i]. Dentro del procedimiento se van generando k-cubos que se guardan en los G[k][i]. Los k-cubos que no generan ningún (k+1)-cubo son implicantes primos.

Generación de implicantes primos

1. k = 0. Agrupar los 0-cubos en grupos G[0][i]. i = 0.
2. Comparar cada elemento A del grupo G[k][i] con todos los elementos B del grupo G[k][i+1]. Si A y B difieren en una única posición P, y (A es 0 y B es 1) ó, (A es 1 y B es 0), se genera un elemento C en el grupo G[k+1][i]. C es como A con el valor de la posición P fijado a -. A y B se marcan, no pueden ser implicantes primos.
3. Si i ≠ N-K-1, incrementar i, volver a 2; si no, incrementar k, si k = N ó no se ha generado ningún elemento C para el anterior k pasar a 4, si no hacer i = 0, volver a 2.
4. Los elementos no marcados son implicantes primos. Para su forma lógica SOP (POS): valor a 0 entrada complementada (sin complementar), valor a 1 entrada sin complementar (complementada), valor a - no depende de la entrada. Fin del algoritmo.

$$F(A, B, C, D) = \sum(1, 2, 3, 5, 11, 12, 15) + \sum_{\varnothing}(6, 10, 13)$$

Nº 1's	0-cubos	A	B	C	D	
1	1	0	0	0	1	✗
	2	0	0	1	0	✗
2	3	0	0	1	1	✗
	5	0	1	0	1	✗
	6	0	1	1	0	✗
	10	1	0	1	0	✗
	12	1	1	0	0	✗
3	11	1	0	1	1	✗
	13	1	1	0	1	✗
4	15	1	1	1	1	✗

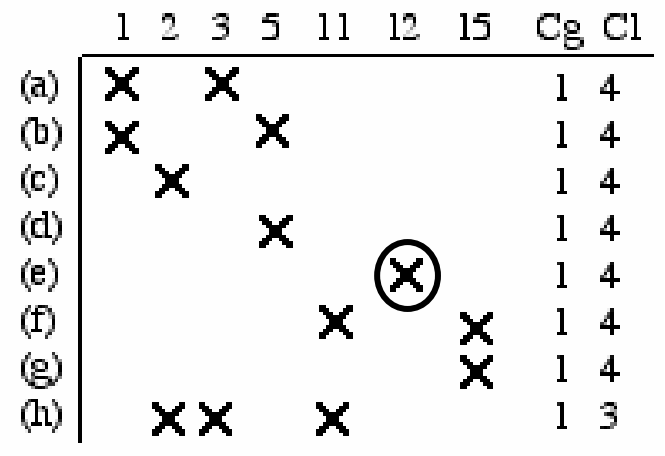
Nº 1's	1-cubos	A	B	C	D		
1	(1, 3)	0	0	-	1		$\overline{A}\,\overline{B}\,D$ (a)
	(1, 5)	0	-	0	1		$\overline{A}\,\overline{C}\,D$ (b)
	(2, 3)	0	0	1	-	✗	
	(2, 6)	0	-	1	0		$\overline{A}\,C\,\overline{D}$ (c)
	(2, 10)	-	0	1	0	✗	
2	(3, 11)	-	0	1	1	✗	
	(5, 13)	-	1	0	1		$B\,\overline{C}\,D$ (d)
	(10, 11)	1	0	1	-	✗	
	(12, 13)	1	1	0	-		$A\,B\,\overline{C}$ (e)
3	(11, 15)	1	-	1	1		$A\,C\,D$ (f)
	(13, 15)	1	1	-	1		$A\,B\,D$ (g)

Nº 1's	2-cubos	A	B	C	D	
1	(2, 3, 10, 11)	-	0	1	-	$\overline{B}\,C$ (h)

Selección de implicantes primos

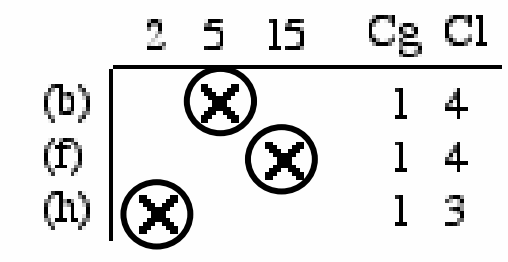
1. Formar una tabla de implicantes primos (filas) frente a 1s (SOP) ó 0s (POS). No usar los “don’t cares”. Marcar los 0-cubos cubiertos por cada implicante primo.
2. Realizar los pasos 3, 4, 5 mientras queden columnas en la tabla se produzcan modificaciones en la tabla. Si la tabla queda vacía generar la función y finalizar, si no ir a 6.
3. Buscar columnas con una única marca. Incluir el implicante primo correspondiente en la función y eliminar de la tabla las columnas cubiertas por el implicante.
4. Si la fila A incluye a la fila B (mismas marcas en A que en B y alguna más en A) y el coste de A es menor o igual que el coste de B, eliminar la fila B.
5. Si la columna A incluye a la columna B, eliminar la columna A.
6. La tabla es cíclica. Seleccionar un implicante primo IP y hacer dos pruebas, quedarse con la genere una función más pequeña:
   - Incluir IP en la función, eliminando las columnas de la tabla. Volver a 2.
   - No incluir IP en la función, eliminar la fila de IP. Volver a 2.

Criterio de Coste. Número de puertas y líneas para cada implicante



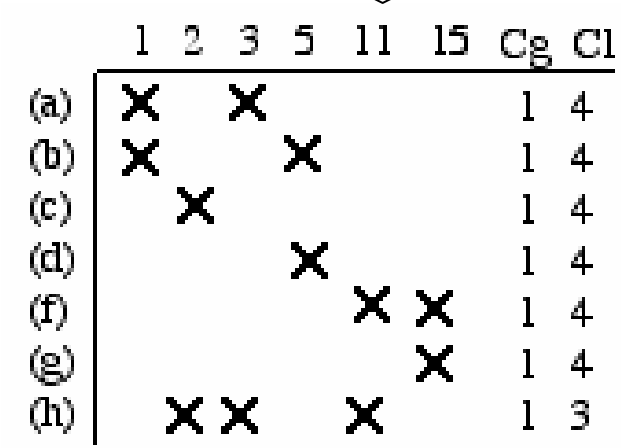
	1	2	3	5	11	12	15	Cg	Cl
(a)	X		X					1	4
(b)	X			X				1	4
(c)		X						1	4
(d)				X				1	4
(e)						⊗		1	4
(f)					X		X	1	4
(g)							X	1	4
(h)		X	X		X			1	3

Columna 12 con una marca. Incluir (e). En la función F = e + …



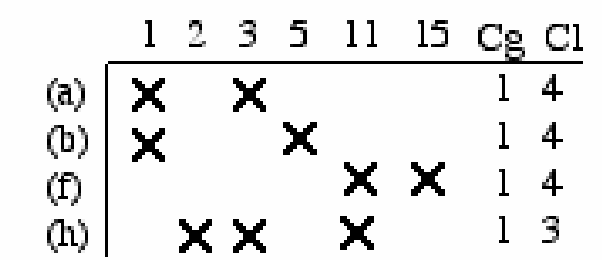
	2	5	15	Cg	Cl
(b)	⊗			1	4
(f)		⊗		1	4
(h)	⊗			1	3

Todas las columnas con una marca. La función F = e + b + f + h



	1	2	3	5	11	15	Cg	Cl
(a)	X		X				1	4
(b)	X			X			1	4
(c)		X					1	4
(d)				X			1	4
(f)					X	X	1	4
(g)						X	1	4
(h)		X	X		X		1	3

(h) incluye a (c)
(b) incluye a (d)
(f) incluye a (g)
con menor coste.
Se eliminan (c), (d) y (g)



	1	2	3	5	11	15	Cg	Cl
(a)	X		X				1	4
(b)	X			X			1	4
(f)					X	X	1	4
(h)		X	X		X		1	3

(3) incluye a (2)
(1) incluye a (5)
(11) incluye a (15)
Se eliminan (3), (1) y (13)

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

Espresso

- Algoritmo estándar de minimización en dos niveles. Opera con problemas de más de 100 variables de entrada y/o salida de los que no se conoce cuál es la solución óptima.
- Utiliza un método de aproximaciones sucesivas basado en tres procedimientos.

  1. Generación de implicantes primos: se parte de un conjunto de cubos que se expanden en determinadas variables para configurar un subconjunto de todos los implicantes primos que pueden cubrir las funciones.

  2. Se selecciona un conjunto mínimo de implicantes primos del conjunto anterior que cubra a todas las funciones. En las iteraciones que se realizan se comprueba que esta solución encontrada es mejor que la solución anterior. Si no lo es finalizar.

  3. El conjunto anterior se rehace reduciendo algunos implicantes primos a cubos en las zonas cubiertas por más de 1 implicante. Se vuelve al paso 1.
- Incluye además procedimientos extra como el cálculo de implicantes primos esenciales.

mi_fich

```
.i 3
.o 3
.ilb a b c
.ob f1 f2 f3
.type fd
.p 9
-00 100
101 110
01- 100
110 -10
001 0--
010 0-0
100 001
011 101
00- 00-
.e
```

número de entradas

número de salidas

nombre de cada entrada

nombre de cada salida

Indica el tipo de cubos activos f (ON-SET, 1s), d (DC-SET), r (OFF-SET, 0s) Por defecto fd. No se leen los valores no activos (0s, por defecto)

número de cubos

fin de cubos

Descripción por cubos: 01- 100
Para cada entrada:
0 Literal complementado
1 Literal sin complementar
- No depende de la entrada
Para cada salida activada por .type:
1 El cubo pertenece al ON-set
0 El cubo pertenece al OFF-set
- El cubo pertenece al DC-set

espresso mi_fich

-Dexact: minimización exacta conjunta.
-Dopo: minimización conjunta pero elige F ó $\overline{F}$ para cada función.
-Dso: minimización de cada salida por separado.
-Dso_both: minimización de cada salida por separado, elige F ó $\overline{F}$.

```
.i 3
.o 3
.ilb a b c
.ob f1 f2 f3
.phase 111
.p 4
101 110
011 101
-10 110
-00 101
.e
```

Indica las salidas en las que se obtiene F (1) y en las que se obtiene $\overline{F}$ (0).

Descripción por cubos: 01- 100
Para cada entrada como en el fichero de entrada
Para cada salida:
1 El cubo pertenece a la salida
0 El cubo no pertenece a la salida

Síntesis multinivel

- En síntesis multinivel no se aplica el término minimización ya que no existe ningún método que garantice unas funciones lógicas mínimas. Se realizan una operaciones que mejoran de alguna manera el coste final de las expresiones, en principio reduciendo el número de literales del conjunto de las expresiones.
- La síntesis multinivel suele comenzar con una minimización en dos niveles. Al resultado de la minimización se le aplican operaciones típicas de síntesis multinivel y algoritmos complejos para llevarlas a cabo. Entre estas operaciones :

  - Factorización: pasar de dos niveles a forma factorizada.

  - Descomposición: reemplazar una expresión lógica por un conjunto de expresiones lógicas.

  - Substitución: expresar una función en términos de otra función.

  - Colapsado: la operación contraria a la substitución

  - Extracción: identifica términos comunes en varias expresiones.

Factorización

- Una operación típica de la síntesis multinivel es la factorización: el paso de una expresión en dos niveles a una expresión factorizada.
- Esta operación es un punto de partida importante de la síntesis multinivel y es interesante encontrar una buena (si no la mínima) factorización. Dada una expresión lógica F en dos niveles puede conseguirse una factorización aplicando recursivamente este procedimiento:

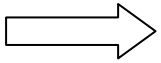
  - Localizar el literal X que más veces aparece en la expresión en dos niveles F y generar dos subexpresiones F1 y F2 de forma que F = X F1 + F2. Repetir recursivamente el procedimiento para F1 y para F2, hasta que en las subexpresiones generadas en cada paso no aparezca ningún literal más de 1 vez.

Extracción

- Identifica términos comunes en varias expresiones lógicas. Es la operación multinivel más difícil de implementar ya que exige encontrar los factores de las expresiones.
- Se basa en las operaciones de división algebraica (o booleana más compleja) que reexpresa una expresión lógica en términos de divisores (cokernels) y de cocientes (kernels)

$F1 = \underline{AB}C\ (1) + A\overline{C}\underline{G}\ (2) + \overline{B}DF\ (3) + \underline{CD}E\ (4)$
$F2 = \overline{A}\overline{B}\overline{D}\ (5) + BC\overline{E}\ (6) + \overline{B}DE\ (7) + \overline{B}\overline{G}\ (8) + \overline{C}\overline{E}G\ (9)$

Función	Co-kernel	Kernel
F1	A	$BC + \overline{C}G$
F1	C	$\underline{A}B + DE$
F1	D	$\overline{B}F + CE$
F2	$\underline{B}$	$\overline{A}\overline{D} + C\overline{E}$
F2	$\overline{B}$	$DE + \overline{G}$
F2	$\overline{E}$	$BC + \overline{C}G$



$F1 = \underline{A}X + DY$
$F2 = \overline{E}X + \overline{B}Z + \overline{A}\overline{B}\overline{D}$
$X = \underline{B}C + \overline{C}G$
$Y = \overline{B}F + CE$
$Z = DE + \overline{G}$