Circuitos Digitales

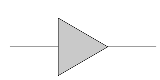
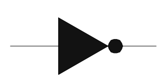
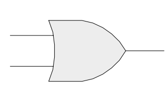
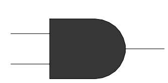
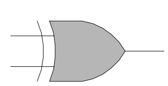
Ing. Javier Barriga Hoyle

Unidad 2

Compuertas Lógicas

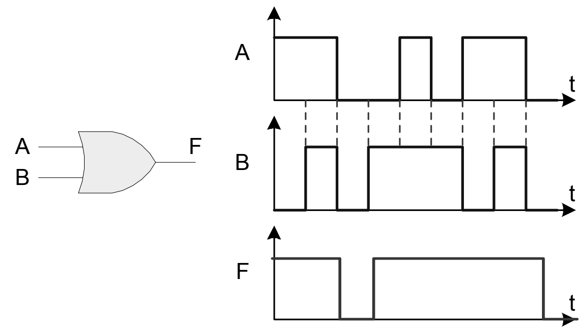
Contenido

1. Simbología lógica

- IGUALDAD: 
- NOT: 
- OR: 
- AND: 
- XOR: 

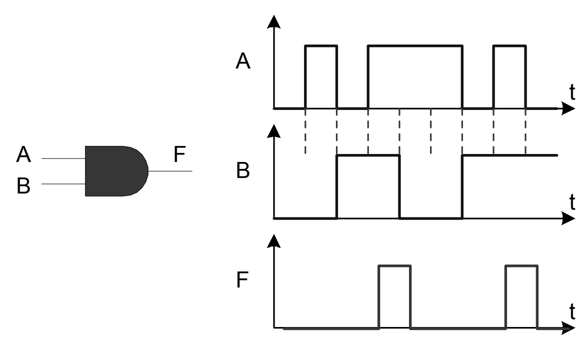
1.1 Aplicaciones de compuertas lógicas

- OR:



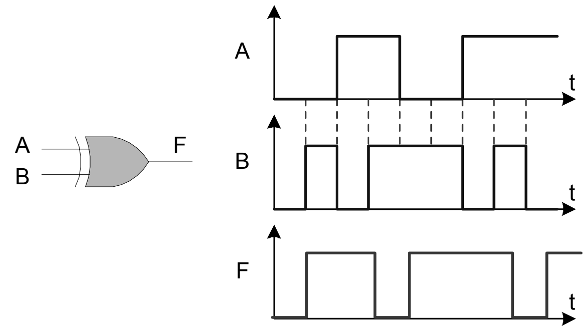
1.1 Aplicaciones de compuertas lógicas

- AND:



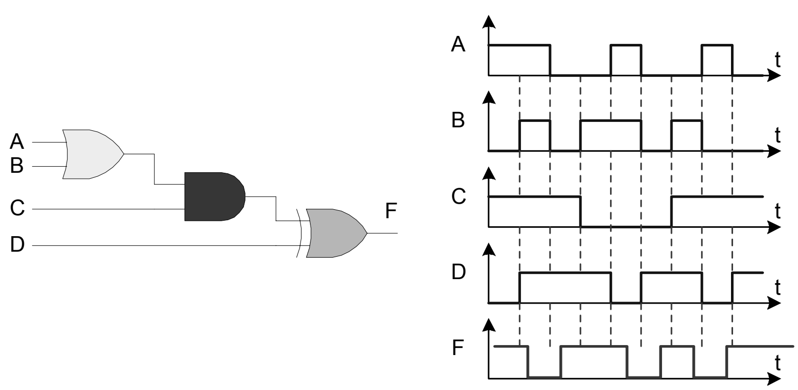
1.1 Aplicaciones de compuertas lógicas

- XOR:



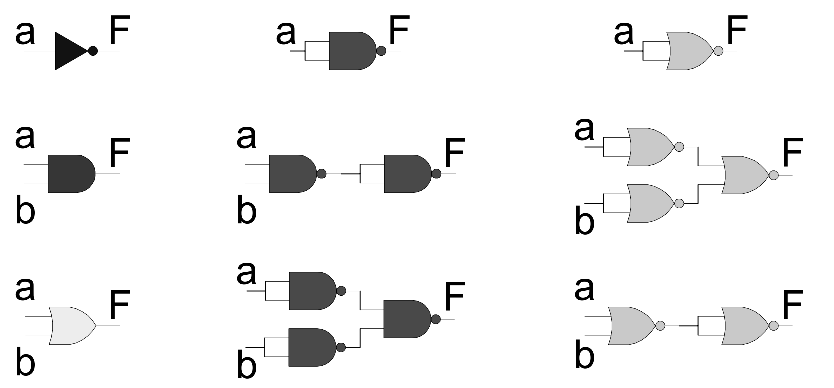
1.1 Aplicaciones de compuertas lógicas

- Combinada:



2. Funciones NAND y NOR

Función NAND NOR



2.1 Funciones lógicas con sólo NAND

- Para transformar cualquier función lógica a solo NAND se debe seguir los siguientes pasos:
  - Aplicar a la expresión en su conjunto una doble inversión.
  - Si la función es un producto, las dos negaciones quedan tal cual.
  - Si es una suma, se elimina una de ellas mediante la aplicación del teorema de D'Morgan.

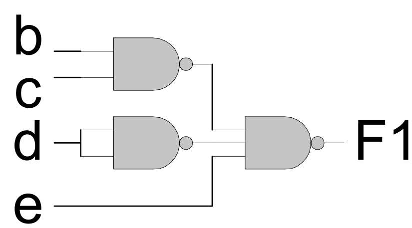
2.1 Funciones lógicas con sólo NAND

- Se continúa invirtiendo doblemente los términos o partes de la función hasta que todas las sumas y productos se conviertan en productos negados.

2.2 Aplicaciones con sólo NAND

1) $F_1 = b.c + d + \overline{e}$

- ➔ $F_1 = \overline{\overline{(b.c + d + \overline{e})}}$
- ➔ $F_1 = \overline{\overline{(b.c)}.\overline{d}.e}$

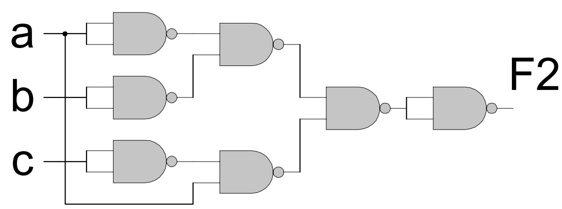


2.2 Aplicaciones con sólo NAND

2) $F_2 = (a + b).(c + a)$

➔ $F_2 = \overline{\overline{(a + b).(c + a)}}$

➔ $F_2 = \overline{\overline{\overline{\overline{(a + b)}}.\overline{\overline{(c + a)}}}} \Rightarrow F_2 = \overline{\overline{\bar{a}.\bar{b}}.\overline{\bar{c}.\bar{a}}}$



2.3 Funciones lógicas con sólo NOR

- Para transformar cualquier función lógica a solo NOR se debe seguir los siguientes pasos:
  - Se debe aplicar una doble inversión.
  - Si la función es una suma lógica, no se opera ninguna inversión.
  - Si es un producto, se elimina una de ellas por aplicación del teorema de D’Morgan.

2.3 Funciones lógicas con sólo NOR

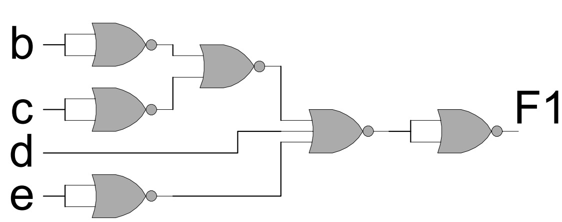
- Se continúa invirtiendo doblemente los términos hasta que todas las sumas y productos se hayan convertido en sumas negadas.

2.4 Aplicaciones con sólo NOR

1) $F_1 = b.c + d + \overline{e}$

➔ $F_1 = \overline{\overline{b.c + d + \overline{e}}} \Rightarrow F_1 = \overline{\overline{\overline{\overline{b.c}} + d + \overline{e}}}$

➔ $F_1 = \overline{\overline{\overline{\overline{\overline{b} + \overline{c}}} + d + \overline{e}}}$

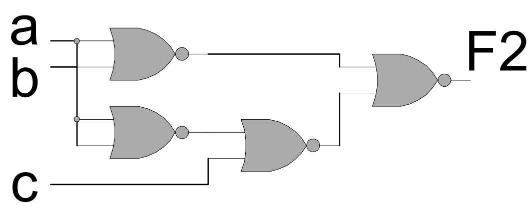


2.4 Aplicaciones con sólo NOR

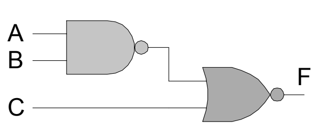
2) $F_2 = (a + b).(c + a)$

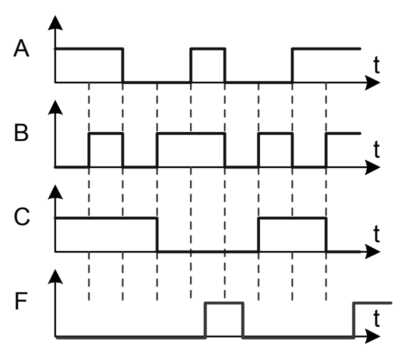
➔ $F_2 = \overline{\overline{(a + b).(c + a)}}$

➔ $F_2 = \overline{\overline{(a + b)} + \overline{(c + a)}}$



2.5 Aplicaciones con NAND y NOR





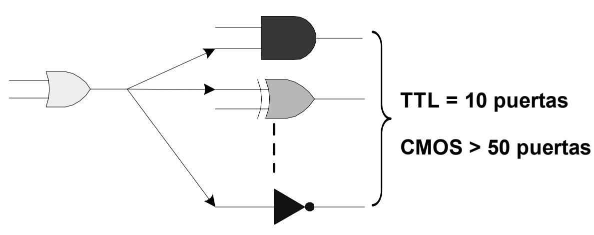
3. Características de los CI.

Las principales características son:

- La tensión de alimentación y su tolerancia.
- La temperatura máxima de trabajo.
- Mínimo consumo.
- Bajo costo.
- Compatibilidad con otras familias lógicas.

3. Características de los CI.

- Fan-out: Es el número máximo de entradas de otras puertas que puede alimentar una salida.

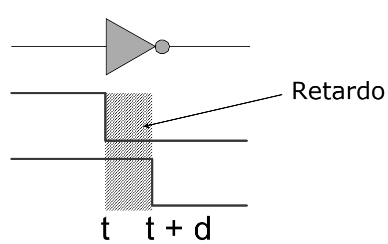


3. Características de los CI.

- Niveles de entrada y salida: Indican los valores de tensión de los estados lógicos 1 y 0.
- Margen de ruido: Son las variaciones máximas que se pueden producir a la entrada sin que la salida varíe su estado lógico (1 o 0).

3. Características de los CI.

- Tiempo de propagación medio: Es el tiempo que transcurre desde que se produce el cambio lógico en la entrada, hasta que lo hace en la salida.



3.1 Familia Lógica TTL

- TTL son las iniciales de Transistor-Transistor-Logic.
  - Construidas con resistencias, diodos y transistores bipolares.
- Sus principales características son:
  - Tensión de alimentación comprendida entre 4.5 y 5.5 V (+5V nominales).
  - Temperatura de trabajo de 0 a 70ºC.
  - Fan-out igual a 10.

3.1 Familia Lógica TTL

- El margen de ruido en ambos niveles, es de 0.4 V.
- Tiempo de propagación medio, 10 ns.
- Disipación de potencia, 10 mW por chip.
- Niveles de tensión:
  - VIH máx = 5.0 V. ; VIH min = 2.0 V.
  - VIL máx = 0.8 V. ; VIL min = 0.0 V.
  - VOH máx = 5.0 V. ; VOH min = 2.4 V.
  - VOL máx = 0.4 V. ; VOL min = 0.0 V.

3.1 Familia Lógica TTL

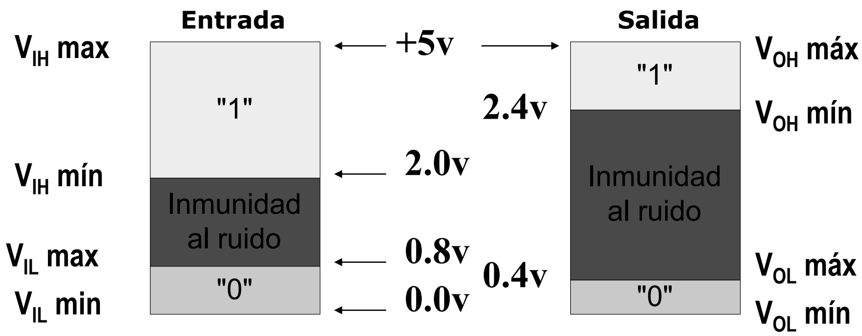


Figura 1. Características de E/S de la familia TTL.

3.2 Familia Lógica CMOS

- Las características más significativas de esta familia son:
  - Tensión de alimentación variable entre 3 y 18 V.
  - Rango de temperatura comprendido entre -40ºC y 85ºC.
  - Fan-out generalmente superior a 50.
  - Gran inmunidad al ruido; soporta hasta el 30% de la tensión de alimentación.

3.2 Familia Lógica CMOS

- La potencia disipada por puerta es de 10 nW.
- Niveles de tensión (para una tensión de alimentación de 5V):
  - VIH máx = 5.0 V; VIH min = 3.5 V.
  - VIL máx = 1.5 V; VIL min = 0.0 V.
  - VOH máx = 5.0 V; VOH min = 4.95 V.
  - VOL máx = 0.05 V; VOL min = 0.0 V.

3.2 Familia Lógica CMOS

- Los tiempos de propagación varían en forma inversa a la tensión de alimentación.
  - De 125 ns para 5V.
  - De 45 ns para 15V.

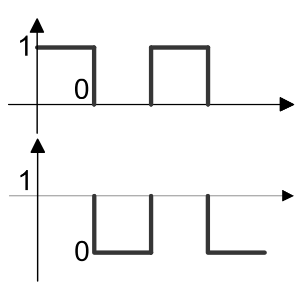
3.3 Diferencias entre TTL y CMOS

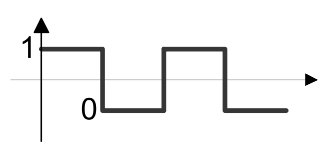
Serie	Potencia disipada	Tiempo de commutación
TTL	10 mW	10 ns
TTL S	19 mW	3 ns
CMOS	10 nW	125 ns
H-S CMOS	2.5 nW	7 ns

Cuadro 1. Diferencias entre familia TTL y CMOS.

4. Lógica positiva y Lógica negativa

- Lógica positiva: El nivel de tensión para el estado lógico "1" es mayor que para el estado cero.



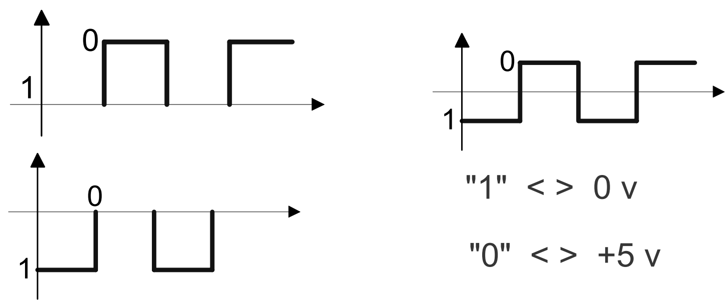


"1" < > +5 v

"0" < > 0 v

4. Lógica positiva y Lógica negativa

- Lógica negativa: El nivel de tensión correspondiente al estado lógico “1” es menor que el del estado cero.



5. Simplificación de funciones

- Para reducir o simplificar funciones se utilizará el álgebra de Boole y el método gráfico de Karnaugh.
- Cuanto más simplificadas resulten, será menor el costo y el número de puertas para la implementación del circuito lógico.

5.1 Método algebraico

- No existe una regla fija para realizar la simplificación por éste método.
- Para reducir una función lógica se tiene que recurrir, en la medida de lo posible al Algebra de Boole.

5.1 Método algebraico

Ejemplo 1:

- Simplificar: $f = a + b.(a'.c')'$

Solución

➔ T. De Morgan: $(a'.c')' = a + c$

➔ $f = a + b.(a + c)$

➔ $f = a + a.b + b.c$

➔ teorema de absorción: $a + a.b = a$

➔ $f = a + b.c$

5.2 Forma canónica de una función

- Cualquier función lógica, sea cual sea la forma en que esté expresada, se puede transformar a su forma canónica, bien en suma de productos o productos de sumas.
- Completar términos de las variables que faltan sumando CERO (x.x’) o multiplicando por UNO (x + x’).

5.2 Forma canónica de una función

Ejemplo 2:

- Expresar en forma canónica de suma de productos la función: f = a.(b+c)

<u>Solución</u>

➔ f = a.(b + c) = a.b + a.c

➔ f = a.b.(c+c’) + a.c.(b+b’)

➔ f = a.b.c + a.b.c’ + a.b.c + a.b’.c

➔ f = a.b.c + a.b.c’ + a.b’.c

5.2 Forma canónica de una función

Ejemplo 3:

- Pasar a forma canónica de productos de suma la función: $f = a.(b+c)$

Solución

- ➔ $f = [a+(b.b')+(c.c')].[b+c+(a.a')]$
- ➔ $f = [(a+b).(a+b')+(c.c')].(a+b+c)$
  $(a'+b+c)$
- ➔ $f = (a+b+c.c').(a+b'+c.c').$
  $(a+b+c).(a'+b+c)$

5.2 Forma canónica de una función

- ➔ $f = (a+b+c).(a+b+c').(a+b'+c).$
  $(a+b'+c').(a+b+c).(a'+b+c)$
- ➔ eliminamos los términos repetidos haciendo uso de que: $(x + x = x)$
- ➔ $f = (a+b+c).(a+b+c').(a+b'+c).$
  $(a+b'+c').(a'+b+c)$

5.3 Forma normalizada de una función

- Los términos mínimos que forman la función canónica se reemplaza con el signo "Σ". el cual tendrá los valores de las combinaciones donde la función es UNO.
- Los términos máximos se reemplaza con el signo "Π", y tendrá los valores de las combinaciones donde la función es CERO.

5.3 Forma normalizada de una función

Ejemplo 4:

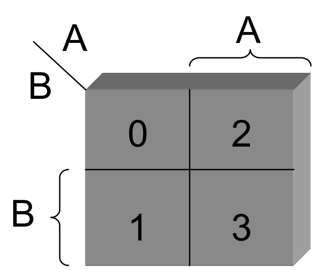
- Pasar a su forma normalizada las funciones:
  - ➔ F(a,b,c) = a'.b.c + a.b'.c' + a.b.c' + a.b.c
    - ➔ F = Σ_3(3,4,6,7)
  - ➔ F(a,b,c) =(a+b+c).(a+b'+c).(a'+b+c). (a'+b+c').(a'+b'+c')
    - ➔ F = Π_3(3,4,6,7)

6. Método gráfico de Karnaugh

- Este método consiste en un sistema de celdas y casillas que contienen toda la información presente en la tabla de verdad.
- Permite una rápida simplificación visual en la ecuación lógica de acuerdo con algunas reglas simples.

6.1 Karnaugh: 2 variables F(A,B)

	A	B
0	0	0
1	0	1
2	1	0
3	1	1



6.1 Karnaugh: 2 variables F(A,B)

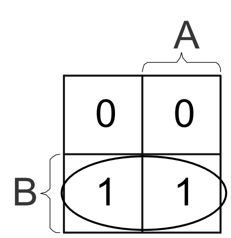
Reglas para simplificar:

- Se combina un agrupamiento de DOS celdas adyacentes para producir una sola variable.
- La celda que no se puede combinar, representa un término de 2 variables

6.1 Karnaugh: 2 variables F(A,B)

Ejemplo 5: Simplificar la función:

A	B	F
0	0	0
0	1	1
1	0	0
1	1	1



Del mapa se tiene: F = B

6.1 Karnaugh: 2 variables F(A,B)

Ejemplo 6: Simplificar la función:

A	B	F
0	0	1
0	1	1
1	0	0
1	1	1

		A
	1	0
B	1	1

➔ F = A’ + B

6.1 Karnaugh: 2 variables F(A,B)

Ejemplo 7: Simplificar la función:

A	B	F
0	0	1
0	1	0
1	0	0
1	1	0

		A
	1	0
B	0	0

➔ F = A’. B’

6.2 Karnaugh: 3 variables F(A,B,C)

A	B	C	F
0	0	0	m0
0	0	1	m1
0	1	0	m2
0	1	1	m3
1	0	0	m4
1	0	1	m5
1	1	0	m6
1	1	1	m7

C \ AB (A over the last two columns, B over the middle two columns)

C \ AB				
	m0	m2	m6	m4
C	m1	m3	m7	m5

6.2 Karnaugh: 3 variables F(A,B,C)

Reglas para simplificar:

- Cuatro celdas adyacentes (en línea o cuadrado) producen una variable.
- Dos celdas adyacentes producen un término de 2 variables.
- La celda que no se puede combinar, representa un término de 3 variables

6.2 Karnaugh: 3 variables F(A,B,C)

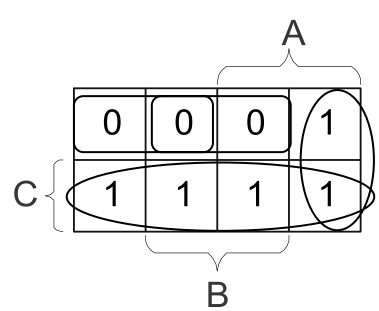
Nota:

- Proximidad: Cuando las celdas en el gráfico están una al costado del otro.
- Simetría: Cuando las celdas en el gráfico están equidistantes de la línea divisoria (eje principal y eje secundario) entre el campo de una variable y el de su complemento.

6.2 Karnaugh: 3 variables F(A,B,C)

Ejemplo 8: Simplificar la función.

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



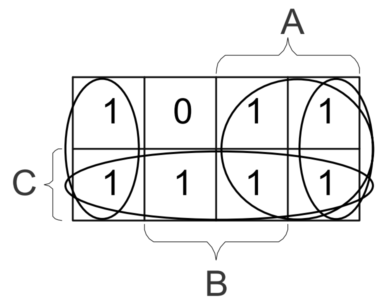
(1s) ➔ F = C + A.B’

(0s) ➔ F = (A+C).(B’+C)

6.2 Karnaugh: 3 variables F(A,B,C)

Ejemplo 9: Simplificar la función.

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

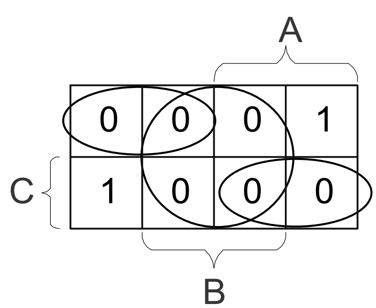


(1s) ➔ F = C + A + B’

6.2 Karnaugh: 3 variables F(A,B,C)

Ejemplo 10: Simplificar la función.

$$F = \Pi_3\ (0,2,3,5,6,7)$$



➔ F = (B’).(A’+C’).(A+C)

6.2 Karnaugh: 3 variables F(A,B,C)

Ejemplo 10: Simplificar la función.

$$F = \Pi_3\ (0,1,2,5,6)$$

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

➔ F = (A'+C').(A+C).(A+B)

6.3 Karnaugh: 4 variables F(A,B,C,D)

			A	A	
	0	4	12	8	
	1	5	13	9	D
C	3	7	15	11	D
C	2	6	14	10	
		B	B		

6.3 Karnaugh: 4 variables F(A,B,C,D)

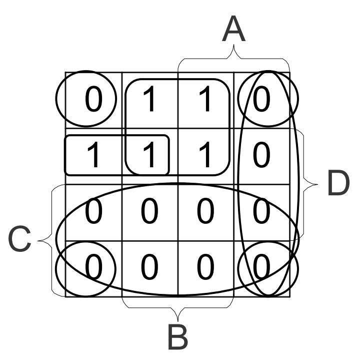
Reglas para simplificar:

- Ocho celdas adyacentes producen una sola variable.
- Cuatro celdas adyacentes producen un término de 2 variables.
- Dos celdas adyacentes producen un término de 3 variables.
- Las celdas individuales representan términos de 4 variables.

6.3 Karnaugh: 4 variables F(A,B,C,D)

Ejemplo 11: Simplificar la función.

$F = \Pi_4\ (0,2,3,6,7,8,9,10,11,14,15)$



Ceros:

➔F = (C’).(A’+B).(B+D)

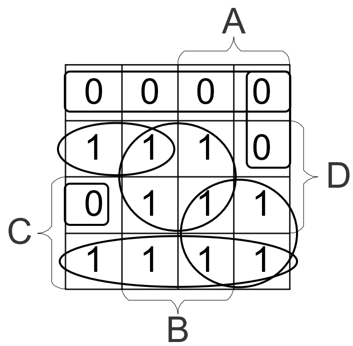
Unos:

➔F = B.C’ + A’.C’.D

6.3 Karnaugh: 4 variables F(A,B,C,D)

Ejemplo 12: Simplificar la función.

$$F = \Pi_4\ (0,3,4,8,9,12)$$



Ceros:

➔F = (C+D).(A'+B+C).
(A+B+C'+D')

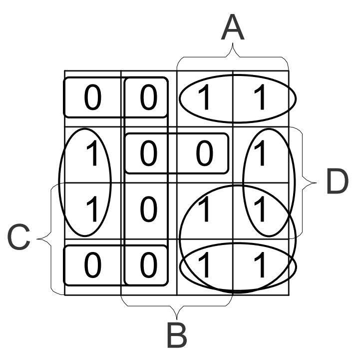
Unos:

➔F = A.C + B.D +
C.D' + A'.C'.D

6.3 Karnaugh: 4 variables F(A,B,C,D)

Ejemplo 13: Simplificar la función.

$$F = \Sigma_4\ (1,3,8,9,10,11,12,14,15)$$



Unos:

➔F = A.C + B'.D +
A.D'

Ceros:

➔F = (A+B').(A+D).
(B'+C+D')

7. Condiciones indiferentes

- Son combinaciones de las variables de entrada que no están permitidas. Esto es, pueden considerarse como términos indiferentes con respecto a su efecto en la salida.
- Estos términos pueden utilizarse para la simplificación en el mapa de Karnaugh. Para diferenciarse de los **unos** y **ceros** se usará la letra **X**.

7. Condiciones indiferentes

- Cuando se agrupan los **unos**:
  - Las **X** pueden considerarse como **unos** para agrandar los grupos, ó como **ceros** si no se obtiene ninguna ventaja.
- Las condiciones indiferentes (las X) se agrupan dentro de una función:

  $F = \Sigma_3 (1,4,5,7) + d\ (0,6)$

  $F = \Pi_4 (0,3,4,8,12,15).\ D(1,5,10,14)$

7. Condiciones indiferentes

Ejemplo 14: Implemente un circuito que funcione con el código BCD. "F" será UNO cuando en la entrada se tenga los números 7, 8 y 9.

Solución

- ➔En el código BCD los números válidos están comprendidos entre el 0 y 9.
- ➔Los números indiferentes están entre el 10 y 15.

7. Condiciones indiferentes

➔ Circuito lógico: