ANTOLOGÍA (SISTEMAS NUMÉRICOS), POR CARLOS CEDILLO NAKAY Y MÓNICA TALIA VIOLETA SIERRA PEÓN

INTRODUCCION

La presente antología fue desarrollada con el propósito de auxiliar a los alumnos que se introducen en el estudio de los temas de Sistemas Combinacionales en el de área de Comunicaciones y Electrónica

El tema abordado dentro de este campo son los Sistemas Numéricos, de los cuales existen una gran variedad y en el presente trabajo
Se resume a los sistemas que tendrán mayor aplicación en su curso de Circuitos Combinacionales.

Sistemas Numéricos
Se denomina sistema de numeración al conjunto de símbolos y reglas que se utilizan para la representación de datos numéricos o cantidades.

Un sistema de numeración se caracteriza fundamentalmente por su *base*, que es el número de símbolos distintos que utiliza, y además es el coeficiente que determina cuál es el valor de cada símbolo dependiendo de la posición que ocupe.

Los sistemas de numeración actuales son sistemas posicionales, en los que el valor relativo que representa cada símbolo o cifra de una determinada cantidad depende de su valor absoluto y de la posición relativa que ocupa dicha cifra con respecto a la coma decimal.

Este sistema cuenta con conjuntos ordenados de símbolos llamados "dígitos", con relaciones definidas para:

- Suma
- Resta
- Multiplicación
- División

La Base (r) del sistema representa el numero total de dígitos permitidos, ejemplo:

- r = 2 Sist. Binario, dígitos: 0,1
- r = 10 Sist. Decimal, dígitos: 0,1,2,3,4,5,6,7,8,9
- r = 16 Sist. Hexadecima1, dígitos: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Notación Posicional
Suponga que pide a su banco local un préstamo por ciento veintitrés yens y treinta y cinco centavos. El cheque que le dan indica la cantidad como Y/.123.35. Al escribir este número, se ha utilizado la notación posicional. El cheque puede cobrarse con un billete de cien yens, dos billetes de diez yens, tres billetes de un yen, tres monedas de diez centavos y cinco monedas de un centavo. Por tanto, la posición de cada dígito indica su peso o significado relativo.
En general, un número positivo *N* se puede escribir en notación posicional como

$$N = \left(a_{n-1}\, a_{n-2} \cdots a_1\, a_0 \cdot a_{-1}\, a_{-2} \cdots a_{-m}\right)_r$$

- Donde: Ejemplos:

. = punto
r = base * $(123.45)_{10}$
n = # dígitos positivos * $(1001.11)_2$
m = # dígitos negativos * $(3A.2F)_{16}$
a_{-1} = dígito más significativo
a_{-m} = dígito menos significativo

Observe que el intervalo de valores para los dígitos a_i es r -1>= a_i >= 0. Con esta notación, la cantidad del préstamo bancario podría escribirse B/.$(123.35)_{10}$. Los paréntesis y el subíndice que denota la base pueden eliminarse sin perder información siempre que la base se conozca por el contexto o se especifique de otra forma.

Notación Polinomial
Podemos escribir la cantidad del préstamo de $(123.35)_{10}$ balboas en forma polinomial como
$N = 1 \times 100 + 2 \times 10 + 3 \times 1 + 3 \times 0.1 + 5 \times 0.01$
$N = 1 \times 10^2 + 2 \times 10^1 + 3 \times 10^0 + 3 \times 10^{-1} + 5 \times 10^{-2}$
Observe que cada dígito está en una posición ponderada y que el peso de cada posición es una potencia de la base 10. En general, cualquier número *N* con base *r* se puede escribir como un polinomio de la forma

$$N = \sum_{i=-m}^{n-1} a_i r^i$$

donde cada símbolo se define como en la ecuación 1.1. Para el préstamo del banco, $r = 10$, $a_2=1$, $a_1=2$, $a_0=3$, $a_{-1}=3$, $a_{-2}=5$ y $a_i=0$, para i >= 3 y para i >= 3.

- Ejemplos:

$(123.45)_{10} = 1 * 10^2 + 2 * 10^1 + 3 * l0^0 + 4 * 10^{-1} + 5 * 10^{-2}$

$(1001.11)_2 = 1 * 2^3 + 0 * 2^2 + 0 * 2^1 + 1 * 2^0 + 1 * 2^{-1} + 1 * 2^{-2}$

$(3A.2F)_{16} = 3 * 16^1 + A * 16^0 + 2 * 16^{-1} + F * 16^{-2}$

Donde: A = 10, B = 11, C = 12, D = 13, E = 14 y F = 15

Sistemas de uso común

Binary	Decimal	Hexadecimal
0	0	0
1	1	1
10	2	2
11	3	3
100	4	4
101	5	5
110	6	6
11	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

Conversión de un sistema de base r a base 10

- Utilizando la notación polinomial:

Ejemplos:

$(10100)_2 = 1 * 2^4 + 0 * 2^3 + 1 * 2^2 + 0 * 2^1 + 0 * 2^0 = (20)_{10}$

$(AF3.15)_{16} = 10 * 16^2 + 15 * 16^1 + 3 * 16^0 + 1 * 16^{-1} + 5 * 16^{-2} =$
$(2803.08203125)_{10}$

Conversión de un sistema de base r a base 10

- Utilizando la noción de los pesos

Ejemplo en el sistema Binario (r = 2):

Peso (2^i): 8 4 2 1

Dígito $(b_i) = b_3\ \ b_2\ \ b_1\ \ b_0$

$(1001)_2 = 8 + 1 = (9)_{10}$

$(0101)_2 = 4 + 1 = (5)_{10}$

Conversión de un sistema de base 10 a base r

- Utilizando divisiones sucesivas por la base

Ejemplos:

$(13)_{10} = (1101)_2$ $(234)_{10} = (EA)_{16}$

```
13   2
 1   6  2
     0  3  2
        1  1  2
           1  0
```

```
234  16
 10  14  16
  A  14   0
      E
```

Conversión de un sistema de base 10 a base r

- Usando la noción de los pesos:

Ejemplo para el sistema Binario (r = 2)

$(38)_{10} = 32 + 4 + 2 = (100110)_2$

$(59)_{10} = 32 + 16 + 8 + 2 + 1 = (111011)_2$

Conversión entre las bases 2 y 16

- $(\underline{1100}\ \underline{0011}\ \underline{1111}\ \underline{1101})_2 = (C3FD)_{16}$

  C 3 F D

- $(0001\ 1000)_2 = (18)_{16}$ (completando con 0's)

- $(4AB)_{16} = (0100\ 1010\ 1011)_2$

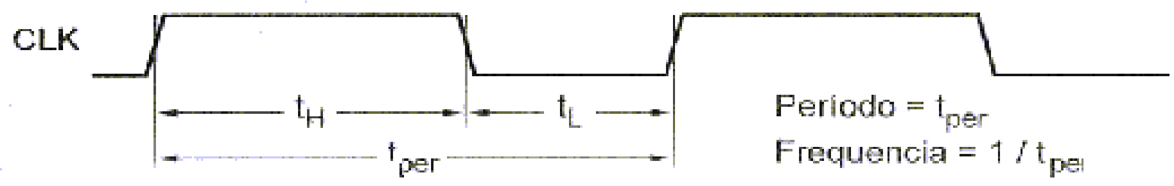
II Aritmética Binaria (SUMA)

II.1.- Suma Binaria

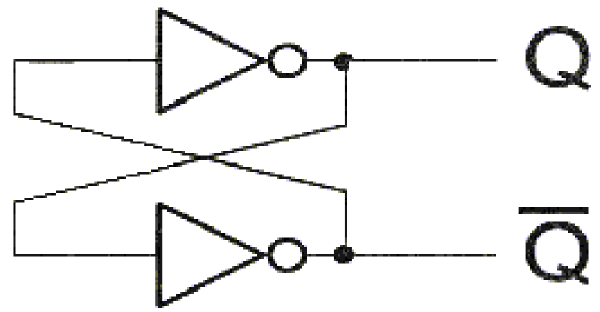
Las tablas 1.1a y b muestran las tablas de suma y multiplicación, respectivamente, para el sistema numérico binario. Las tablas son muy pequeñas ya que sólo hay dos dígitos, o *bits*, en el sistema. En consecuencia, la aritmética binaria es muy sencilla. Observe que la suma 1 + 1 produce un bit se suma de 0 y un bit de acarreo de 1. El acarreo debe sumarse a la siguiente columna de bits para realizar la suma en el patrón normal, de derecha a izquierda.

Memorias

- Reloj



- Biestables



- Latch
- Flip Flop

+	0	1
0	0	1
1	1	10

x	0	1
0	0	0
1	0	1

tabla_1.1a tabla_1.1b

Tabla de Sumar:

Ejemplos:

+	0	1
0	0	1
1	1	10

1 1 1 1 1 1
1 1 1 1 0 1
0 1 0 1 1 1 +
1 0 1 0 1 0 0

1 1 1
1 0 0 1 0 1
0 1 0 1 1 1 +
1 1 1 1 0 0

II.2 **Aritmética Binaria (RESTA)**

Resta Binaria

La resta se puede visualizar como el inverso de la suma. Las reglas para la resta binaria se derivan directamente de la tabla de suma binaria y son:

1 - 0 = 1
1 - 1 = 0
0 - 0 = 0

0 - 1 = 1 tomando prestado 1, o 10 - 1 = 1

La última regla muestra que si se resta un bit 1 de un bit 0, hay que tomar prestado un 1 de la siguiente columna más significativa. Los préstamos se propagan hacia la izquierda de columna en columna, como se ilustra a continuación.

Ejemplo 1.2

Restar los dos números binarios $(1001101)_2$ y $(10111)_2$

	6	5	4	3	2	1	0	*columna*
		1			10			*préstamos*
	0	10	10	0	0	10		*préstamos*
	1	0	0	1	1	0	1	*minuendo*
—			1	0	1	1	1	*sustraendo*
		1	1	0	1	1	0	*diferencia*

Tabla de Restar:

Ejemplos:

10 -	0	1
0	0	-
1	1	0

10 - 1 = 1
100 - 1 = 11
1000 - 1 = 111

0 1 10 0 0 10
1 0 0 1 1 0 1
1 0 1 1 1 -
0 1 1 0 1 1 0

10
0 0 10 0 1 1 10
1 1 0 1 0 0 0
0 1 1 0 1 0 1 -

0 1 1 0 0 1 1

II.3 **Aritmética Binaria (Multiplicación)**

Multiplicación Binaria
La multiplicación binaria se realiza en forma similar a la multiplicación decimal, excepto que las operaciones de multiplicación binaria son mucho más sencilla. No obstante, se debe tener mucho cuidado al sumar los productos parciales, como se ilustra en el siguiente ejemplo.
Ejemplo 1.3
Multiplicar $(10111)_2$ por $(1010)_2$

			1	0	1	1	1	*multiplicando*
x				1	0	1	0	*multiplicador*
			0	0	0	0	0	
		1	0	1	1	1		
	0	0	0	0	0			
1	0	1	1	1				
1	1	1	0	0	1	1	0	*producto*

Observe que hay un producto parcial por cada bit del multiplicador. Este procedimiento puede realizarse con mayor eficiencia si sólo recorremos una columna a la izquierda, en vez de anotar un producto parcial con ceros para un bit 0 del multiplicador. Este ejemplo nos sirve para ver lo sencillo de este procedimiento.

Tabla de Multiplicar:

Ejemplos:

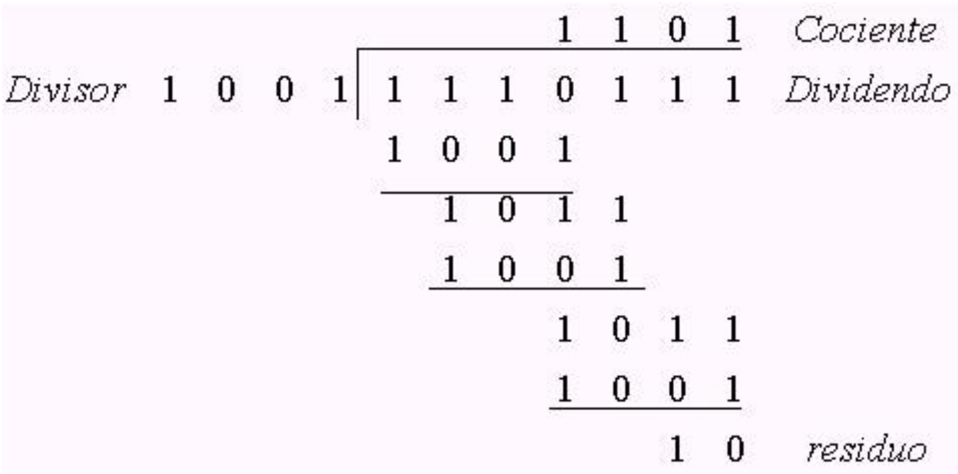
*	0	1
0	0	0
1	0	0

```
   10111
 1010 *
00000 +
  10111
 00000
10111
11100110
```

100111
1010 *
00000 1^{er}.pp.
00000 +
00000 2^{do}.pp.
10111 +
101110 3^{er}.pp.
00000 +
0101110 4^{to}.pp.
10111 +
11100110 Resultado

II.4 Aritmética Binaria (División)

División Binaria
La división binaria se realiza utilizando el mismo procedimiento de prueba y error de la división decimal. Sin embargo, la división binaria es más sencilla pues sólo hay que intentar con dos valores. Se restan del dividendo copias de los términos del divisor, de lo cual se obtienen residuos intermedios positivos. El siguiente ejemplo ilustra la división binaria.
Ejemplo 1.4
Dividir $(1110111)_2$ entre $(1001)_2$




```
1110111     1001
          - 1001        1101   <----   Cociente
            01011
             -1001

001011
            -1001
              0010   <----   Resto
```

III Representación de números binarios con signo

n

S	Rep.de la magnitud

En este sistema de representación, el bit que está situado más a la izquierda representa el signo, y su valor será 0 para el signo + y 1 para el signo -. El resto de bits (n-1) representan el módulo del número. Suponemos en principio que los números no poseen parte decimal, por lo que la coma se supone implícita a la derecha.
Por ejemplo, supongamos que disponemos de 8 bits, y queremos representar los números 10 y –10. Veamos cuales son sus representaciones.

0 (+)	0	0	0	1	0	1	0

representa al número 10,

1 (-)	0	0	0	1	0	1	0

representa al número –10.

Se denomina *rango de representación* en un método determinado al conjunto de número representables en el mismo. Para módulo y signo el rango de representación es, si se disponen de n bits:

$$-2^{n-1} + 1 <= X <= 2^{n-1} - 1$$

Para el caso de n = 8 bits, el rango de representación va desde –127 a 127.
La ventaja que presenta este sistema frente a otros es la de poseer rango simétrico (igual cantidad de números positivos que negativos), mientras que su mayor inconveniente es el de poseer dos representaciones para el número 0. El cual se representa tanto con un signo positivo (0) como con uno negativo (1) y el resto de los bits en 0.

Existe:

- Rep. Signo - Magnitud

- Rep En Complemento

III. 1 Representación de números binarios con signo – magnitud

En esta notación el bit de más de la izquierda en la palabra (bit más significativo [BMS]) representa el signo. Usualmente, 0 denota + (cantidad positiva) y 1 denota – (cantidad negativa). El resto de los bits representa magnitud.

Un número en representación signo –magnitud puede escribirse como:

$$N = (sa_{n-1} \ldots\ldots a_1 a_0)_{2sm}$$

Donde:

s =signo (0 = positivo y 1 = negativo)
n = # de bits para la magnitud
a_{n-1}= bits mas significativo (MSB) para la magnitud

Ejemplos:

- - $(1101)_2$= $(11101)_{2sm}$
- + $(1001)_2$ = $(01001)_{2sm}$

III.2 Representación de números binarios en Complemento

Un número en representación signo – magnitud puede escribirse como:

$$[N]_2 = 2^n - (N)_2$$

N = número binario
[N] = complemento del número N
n = número de bits de N

Rango(n) : $2^{n-1} - 1$
-2^{n-1}

Complemento a 1
Este sistema de representación utiliza el bit de más a la izquierda para el signo, correspondiendo el 0 para el signo + y el 1 para el signo -. Para los números positivos, los n-1 bits de la derecha representan el módulo (igual que en el sistema anterior). El negativo de un número positivo se obtiene complementando todos sus dígitos (cambiando ceros por uno y viceversa) incluido el signo. Veamos la representación en complemento a 1 de los números 10 y –10 para el caso de n = 8 bits.

0 (+)	0	0	0	1	0	1	0

representa al número 10,

1 (-)	1	1	1	0	1	0	1

representa al número –10.

Para el complemento a 1 el rango de representación es, si se disponen de n bits:
$$-2^{n-1} + 1 <= X <= 2^{n-1} - 1$$

Para el caso de n = 8 bits, el rango de representación va desde –127 a 127.
La ventaja que presenta este sistema frente a otros es la de poseer rango simétrico (igual cantidad de números positivos que negativos), mientras que su mayor inconveniente es el de poseer dos representaciones para el número 0. El cual se representa tanto con todos 0 como con todos los bits en uno.

Complemento a 2

Este sistema de representación utiliza el bit de más a la izquierda para el signo, correspondiendo el 0 para el signo + y el 1 para el signo -. Para los números positivos, los n -1 bits de la derecha representan el módulo (igual que en los dos sistemas anteriores). El negativo de un número positivo se obtiene en dos pasos:

- Primer paso: se complementa el número positivo en todos sus bits (cambiando ceros por uno y viceversa), incluido el bit de signo, similar a complemento a 1.
- Segundo paso: al resultado obtenido se el suma 1 (en binario), despreciando el último acarreo si existiera. Veamos la representación en complemento a 2 de los números 10 y –10 para el caso de n=8 bits.

0 (+)	0	0	0	1	0	1	0

representa al número 10,

1 (-)	1	1	1	0	1	1	0

representa al número –10.

Para el complemento a 2 el rango de representación es, si se disponen de n bits:

$$-2^{n-1} <= X <= 2^{n-1} - 1$$

Para el caso de n = 8 bits, el rango de representación va desde –128 a 127.
La principal ventaja es la de tener una única representación para el número 0, ya que el 0 positivo o negativo se representan igual.

Exceso a 2^{n-1}

Este método de representación no utiliza la convención del bit más significativo para identificar el signo, con lo cual todos los bits representan un número o valor. Este valor se corresponde con el número representado más el exceso, que para n bits viene dado por 2n-1. El signo del número resulta de una operación aritmética.
Por ejemplo, para n = 8 bits el exceso será 128, con lo cual para representar un número deberá sumársele dicho exceso. De esta manera el número 10, que veníamos representando, recibirá la adición del número 128, con lo que representaremos el número binario 138. Por otro lado, el número –10, se representará como el 118 (-10+128). De esta forma quedarán:

1	0	0	0	1	0	1	0

representando al número 10,

0	1	1	1	0	1	1	0

representando al número –10.

En este sistema el número 0 tiene una sola representación, la cual consiste en representar el exceso, 128 en este caso.
El rango de representación en exceso a 2n-1 es asimétrico y viene dado por:
$-2^{n-1} <= X <= 2^{n-1} - 1$
Resulta interesante observar que todo número representado en exceso a 2n-1 tiene la misma representación que un complemento a 2 con el bit de signo cambiado. Puede inferirse entonces, que el bit mas significativo representaría el signo de valor opuesto (el 0 un valor '-' y el 1 un valor '+').

Ejemplos:

- Si N = 01100101, entonces $[N]_2$=?

$[N]_2 = 2^8 –(01100101)_2= (100000000)_2- (01100101)_2$

= 10011011

- Si N = 1101100, demuestre que $[N]_2]_2$ $(N)_2$

$[N]_2 = 2^8 - (1101100)_2 = (100000000)_2 - (1101100)_2$
$= (00101100)_2$

$[[N]]_2 = 2^8 - (00101100)_2 = (100000000)_2 - (00101100)_2$
$= (1101100)_2$

$[N]_2$ sirve para representar a $-(N)_2$

IV Algoritmo de conversión

- Algoritmo:

- Reemplazar cada bit (b_i) de $(N)_2$ por su complemento, donde:

* Si bi = 0 su complemento = 1
* Si bi = 1 su complemento = 0

- Luego sumarle 1.

Ejemplos:
$(10100)_2$ => 01011 + 1 = 01110 = $[10100]_2$
$(11010100)_2$ => 00101011 + 1 = 00101100 = $[11010100]_2$

IV. 1 Conversión entre un sistema en complemento y el sistema decimal

- Se utiliza la misma noción, ahora con el peso del MSB como negativo

Ejemplo:

Peso (2^i): -8 4 2 1
Dígito (b_i): b_3 b_2 b_1 b_0 (donde b_3 es el MSB)
$(1001)_2 = -8 + 1 = -(7)_{10}$
$(0101)_2 = 4 + 1 = +(5)_{10}$
$-(21)_{10} = -32 + 8 + 4 = (101100)_2$
$+(16)_{10} = 16 = (010000)_2$

Rango y precisión

- Si n = 5 => $b_4b_3b_2b_1b_0$ (b_4 MSB y b_0 LSB)

Rango (5) = $2^{5-1} - 1 = 15$ (01111)
$-2^{5-1} = 16$ (10000)

- Si n = 8 => $b_7b_6b_5b_4b_3b_2b_1b_0$ (b_7 MSB y b_0 LSB)

Rango (8) = $2^{8-1} - 1 = 127$ (01111111)
$-2^{8-1} = -128$ (10000000)

IV. 2 Aritméticas en Complemento (SUMA)

Ejemplos, con n = 5:

01001	0110	01100	10111	10100
00101 +	00111 +	11011 +	11011 +	11011 +
01110	10011	~~1~~00111	~~1~~10010	~~1~~01111

Se eliminan, pues sobrepasa la precisión

Desborde (el resultado sobrepasa el rango), y se presenta cuando ambos sumados tienen el mismo signo distinto.

Expansión de signo

Ejemplo:

(n = 4) 0011 = (n = 5) 00011 = (n = 8) 00000011

(n = 4) 1101 = (n = 5) 11101 = (n = 8) 11111101

IV.3 Aritmética en complemento (RESTA)

$(A)_r - (B)_r = (A)_r + (-(B)_r) = (A)_r + [B]_r$

Ejemplos con n = 5:

11001	00011	01111
01101 -	11011 -	1001 –
11001	00011	01111
10011 +	00101 +	01111 +
~~1~~01100	01000	11110

Desborde

IV. 4 Aritmética en Complemento (Multiplicación)

Ejemplo:

	0110
1011 +	
	00000 1er. pp.
	00110 +
	000110 2do. pp.
	00110 +

Expansión del Signo

```
 0010010  3er. pp.
 00000      +
00010010  4to. pp.
 11010       +
11100010  Resultado
```

n * n bits = 2n bits

V.-Postulados del álgebra de boole

Postulado 1:

- <u>DEFINICIÓN:</u> Un álgebra booleana es un sistema algebraico cerrado formado por dos elementos 0 y 1 (conjunto K), y operadores • y +; para cada par de elementos a y b ∈ K,

donde: + => or
• => and

a	b	a + b
0	0	0
0	1	1
1	0	1
1	1	1

a	b	a • b
0	0	0
0	1	1
1	0	1
1	1	1

Postulado 2:

- Existen elementos 0 y 1, tal que, para a ∈K:
  a) a + 0 = a (elemento neutro)
  b) a . 1 = 1 (elemento identidad)

Postulado 3: Ley conmutativa::

- Para a y b ∈K:
  a) a + b = b + a
  b) a . b = b . a

Postulado 4: Ley Asociativa:

- Para a, b y c ∈K:

  a) a + (b + c) = (a + b) + c
  b) a . (b . c) = (a . b) . c

Postulado 5: Ley Distributiva:

- Para a, b y c ∈K:
  a) a + (b . c) = (a + b) . (a + c)
  b) a . (b + c) = (a . c) + (a . c)

Postulado 6: Ley de absorción.

- Para $a \in K$:
  a) $a + \bar{a} = 1$
  b) $a \, . \, \bar{a} = 0$

V.1 Principio de Dualidad

Establece que si una expresión es válida en el álgebra de boole, entonces su expresión dual también lo es.

Determinamos la expresión dual remplazando los operadores + por · y viceversa y todos los elementos 0 por 1 y viceversa.

Ejemplo:

$a + (b \cdot c) = 1$, expresión su dual es $a \cdot (b + c) = 0$

V.2 Teoremas

Teorema 1: Idempotencia: Tanto la suma como el producto de una variable booleana consigo misma da como resultado la misma variable.

a) $a + a = a$

b) $a \cdot a = a$

- Demostración:

$a + a =$

$(a + a) \cdot 1 =$

$(a + a) \cdot (a + \bar{a}) =$

$a + a \cdot \bar{a} =$

$a + 0 = a$

- Teorema 2: Elemento neutro para + y ·

a) $a + 1 = 1$
b) $a \cdot 0 = 0$

- Demostración:

$a + 1 =$
$(a + 1) \cdot 1 =$
$1 \cdot (a + 1) =$
$(a + \bar{a}) \cdot (a + 1) =$
$a + \bar{a} \cdot 1 =$
$a + \bar{a} = 1$

Teorema 3: Involución: Una variable booleana negada dos veces, da como resultado la misma variable:

-

$\bar{\bar{a}} = a$

- Demostración:

$a + 1 =$

$\bar{a} \cdot 1 + 0 =$

$\bar{a} \cdot (a + \bar{a}) + a \cdot \bar{a} =$

$\bar{a} \cdot a + \bar{a} \cdot \bar{a} + a \cdot \bar{a} =$

$a \cdot (\bar{a} + \bar{a}) = a$

- **Teorema 4 : Absorción**

a) a + a · b = a

b) a · (a + b) = a

- Demostración:

$a + a \cdot b =$

$a \cdot 1 + a \cdot b =$

$a \cdot (1 + b) =$

$a \cdot 1 = a$

- Teorema 5:

a) $a + \bar{a} \cdot b = a + b$

b) $a \cdot (\bar{a} + b) = a \cdot b$

-
- Demostración :

$a + \bar{a} \cdot b =$

$(a + \bar{a}) \cdot (a + b) =$

$1 \cdot (a + b) =$

$(a + b) \cdot 1 = a + b$

- Teorema 6 :

  a) $a \cdot b + a \cdot b = a$

  b) $(a + b) \cdot (a + b) = a$

- Demostración:

  $a \cdot b + a \cdot b =$

  $a \cdot (b + b) =$

  $a \cdot 1 = a$

$x = a + b \Rightarrow x = a + b$
sabemos:
$x \cdot x = o$
$x + x = I$
asumimos :
$x \cdot y = O$
$x + y = I$
entonces:
$y = x$
asumimos :
$y = ā \cdot b$

$x \cdot y =$
$(a + b) \cdot (ā \cdot b) =$
$(ā \cdot b) \cdot (a + b) =$
$(ā \cdot b) \cdot a + (ā \cdot b) \cdot b =$
$a \cdot (ā \cdot b) + b \cdot (ā \cdot b) =$
$(a \cdot ā) \cdot b + ā \cdot (b \cdot b) =$
$0 \cdot b + ā \cdot 0 =$
$b \cdot 0 + ā \cdot 0 =$
$0 + 0 = 0$
entonces:
$x \cdot y = 0$

$x + y =$
$(a + b) + ā \cdot b =$
$(b + a) + ā \cdot b =$
$b + (a + ā .b) =$
$b + (a + b) =$
$(a + b) + b =$
$a + (b + b) =$
$a + I = I$
entonces :
$x + y = 1$
resulta :
$x = y \Rightarrow a + b = ā + b$

- Teorema 7:

  a) $a \cdot b + a \cdot b \cdot c = a \cdot b + a \cdot c$

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

- Demostración:

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

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

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

-
- Teorema 8: Teorema de Morgan:

  a) $a + b = a \cdot b$
  b) $a \cdot b = ā + b$

- En general:

  $a + b + ... + z = ā \cdot b \cdot c \cdot ... \cdot z$
  $a \cdot b \cdot c \cdot ... \cdot z = ā + b + c + ... + z$

Demostración del Teorema de Morgan

- **Teorema 9: Consenso**
  a) a · b + ā · c + b · c = a · b + ā · c
  b) (a + b) · (ā + c) · (b + c) = (a + b) · (ā + c)

- Demostración:

  a · b + ā · c + b · c =
  a · b + ā · c + 1 · b · c =
  a · b + ā · c + (a + ā) · b · c =
  a · b + ā · c + a · b · c + ā · b · c = a · b + ā · c

VI Funciones de Conmutación

Sean x_1, x_2, ... , x_n símbolos llamados variables, cada uno representa un 0 o un 1, definiremos f(x_1, x_2, ... , x_n) como una función de conmutación de x_1, x_2, ... , x_n f puede tomar el valor de 0 ó 1 según los valores para x_1, x_2, ... , x_n; si existen n variables (x_i), entonces existe 2^n formas de asignar los valores para x_1, x_2, ... , x_n y como f tiene dos posibles valores, existen 2^{2n} diferentes funciones para n variables.

Ejemplos:

- n=0
  f()=0,1
- n=1
  f(x)=0 1,x,x

- n=2
  f(x,y)= 0 , x.y , x.y , x
  x.y , y , x.y+x.y, x+y
  x.y , x.y+x.y, y, x+y
  x, x+y , x+y, 1

Representación de una función de Conmutación

- Tabla de Verdad:
  Evaluamos todos los posibles valores de entrada de la función y los colocamos en una forma ordenada de acuerdo al orden decimal.

Ejemplo: f(x, y) = x+y f(x, y) = x.y

a b a+b a b a+b

0 0	0	0 0	0		
0 1	1	0 1	0		
	1 0	1		1 0	0
	1 1	1		1 1	1

Tabla de Verdad

- Describa una función de conmutación con 3 entradas a, b y c una salida z, que es verdadera (1)cuando al menos 2 de sus entradas son verdaderas (1)

a	b	c	f
0	0	0	0
0	0	1	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

VI.1 Representación de una función de Conmutación

- Formas Algebraicas

- SOP (Suma de Productos):se construye al sumar(or)términos productos (and).

Ejm: f(a, b, c, d) = a · b · c + b · d + ā · c · d

- POS(Producto de sumas):se construye con el producto (and)de términos suma (or).

*Ejemplo: f(a, b, c, d) = (a + b + c) · (ā + d)

Formas Algebraicas:

a	b	c	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Representación de una función de Conmutación

- Formas Canónicas:

Son formas SOP y POS con características especiales. Existe una única forma canónica para cada función de conmutación.

- Minitérmino: es un término producto(and) para una función de n variables, en donde cada una aparece bien sea complementada o sin complementar.

* ejm: f(a,b,c) m = a· b · c, a · b · c, ā · b · c

- Maxtérmino: es un término suma (or) para una función de n variables, en donde cada una aparece bien sea complementada o sin complementar.

* ejemplo: f(a, b, c) M = (a + b + c), (ā + b + c)

Formas Canónicas SOP

f (a,b,c) = ā ·b ·c + ā · b · c + a · b ·c

a	b	c	f	
0	0	0	1	ā · b · c
0	0	1	0	
0	1	0	1	a · b · c
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	0	
1	1	1	1	

Relación con la tabla de verdad:

Cada mintérmino está asociado con la línea de la que:

- Las variables que tiene 1 no están complementadas
- Las variables que tiene 0 aparecen complementadas

Formas Canónicas POS

f (a, b, c) = (ā + b + c) · (a + b + c) · (ā + b + c)

a	b	c	f	
0	0	0	0	a + b + c
0	0	1	1	
0	1	0	1	
0	1	1	0	a + b + c
1	0	0	1	
1	0	1	1	
1	1	0	0	a + b + c
1	1	1	1	

Relación con la tabla de verdad:

Cada maxtérmino está asociado con la línea de la que:

- Las variables que tiene 0 no están complementadas
- Las variables que tiene 1 aparecen complementadas

Representación de una función de Conmutación

- Especificación decimal:

-SOP:

f(a, b, c) = ā · b · c + ā · b · c + ā · b · c + a · b · c

f(a, b, c) = $m_2 \cdot m_3 \cdot m_6 \cdot m_7$

f(a, b, c) = $\sum$ m(2,3, 6, 7)

- POS:

f(a, b, c) = (a + b + c) · (a + b + c) · (ā + b + c) · (ā + b + c)

f(a, b, c) = $M_1 \cdot M_3 \cdot M_5 \cdot M_7$

f(a, b, c) = $\prod$ M (1, 3, 5, 7)

Relación Mintérminos Maxtérminos

$\overline{m_i} = M_i$

$\overline{M_i} = m_i$

$F(a, b, c) = \sum m(2, 3, 6, 7) = \prod M(0, 1, 4, 5)$

Deducción de Formas Canónicas

- Teorema 10: Teorema de desarrollo de Shannon.

a) $f(x_1, x_2,..., x_n) = x_1 \cdot f(1, x_2,\dots, x_n) + \overline{x_1} \cdot f(0, x_2,\dots, x_n)$
b) $f(x_1, x_2,..., x_n) = [x_1 + f(0,x_2,\dots,x_n)]\cdot[\overline{x_1} + f(1,x_2,\dots,x_n)]$

Convertir a SOP Canónica

$f(a, b, c) = a \cdot b + a \cdot c + \bar{a} \cdot c$
$= a \cdot f(l, b, c) + \bar{a} \cdot f(0, b, c)$
$= a \cdot (b + c) + \bar{a} \cdot c$
$= b \cdot f(a, l, c) + b \cdot (a, 0, c)$
$= b \cdot (a + \bar{a} \cdot c) + b \cdot (a \cdot c + \bar{a} \cdot c)$
$= a \cdot b + \bar{a} \cdot b \cdot c + a \cdot b \cdot c + \bar{a} \cdot b \cdot c$
$= c \cdot f(a, b, l) + c \cdot (a, b, 0)$
$= c \cdot (a \cdot b + \bar{a} \cdot b + \bar{a} \cdot b) + c \cdot (a \cdot b + a \cdot b)$
$= a \cdot b \cdot c + \bar{a} \cdot b \cdot c + \bar{a} \cdot b \cdot c + a \cdot b \cdot c + a \cdot b \cdot c$
= L m(1, 3, 4, 6, 7)

~

Convertir a SOP Canónica

T6: $a \cdot b + a \cdot \bar{b} = a$
$f(a, b, c) = a \cdot b + a \cdot c + \bar{a} \cdot c$
$a \cdot b = a \cdot b \cdot c + a \cdot b \cdot \bar{c} = \mathbf{m_7} + \mathbf{m_6}$
$a \cdot c = a \cdot b \cdot c + a \cdot \bar{b} \cdot c = \mathbf{m_6} + \mathbf{m_4}$
$\bar{a} \cdot c = \bar{a} \cdot b \cdot c + \bar{a} \cdot \bar{b} \cdot c = \mathbf{m_3} + \mathbf{m_1}$
$f(a, b, c) = \sum m(1, 3, 4, 6, 7)$

Convertir a POS Canónica

$f(a, b, c) = a \cdot (\bar{a} + c)$
$= (a + b \cdot \bar{b} + c \cdot \bar{c}) \cdot (\bar{a} + b \cdot \bar{b} + c)$
$= ((a + b) \cdot (a + \bar{b}) + c \cdot \bar{c}) \cdot ((\bar{a} + b) \cdot (\bar{a} + \bar{b}) + c)$
$= (a + b + c \cdot \bar{c}) \cdot (a + \bar{b} + c \cdot \bar{c}) \cdot (\bar{a} + b + c) \cdot (\bar{a} + \bar{b} + c)$
$= (a + b + c) \cdot (a + b + \bar{c}) \cdot (a + \bar{b} + c)(a + \bar{b} + \bar{c}) \cdot (\bar{a} + b + c) \cdot (\bar{a} + \bar{b} + c)$
$= \prod M(0, 1, 2, 3)$

VII Minimización

″ En general al minimizar un sistema digital para su implementación con compuertas ofrece:

- ❑ Menor costo, consumo de potencia, espacio físico, tiempo de respuesta.

″ Técnicas:

- ❑ Minimización Algebraica
- ❑ Minimización a través de Mapas de Karnuagh,
- ❑ Minimización Tabular

Minimización Algebraica

- Usa los teoremas del álgebra de Boole, para minimizar la función.
- No existe una técnica o método que indique cuales teoremas usar, en general se recomienda:

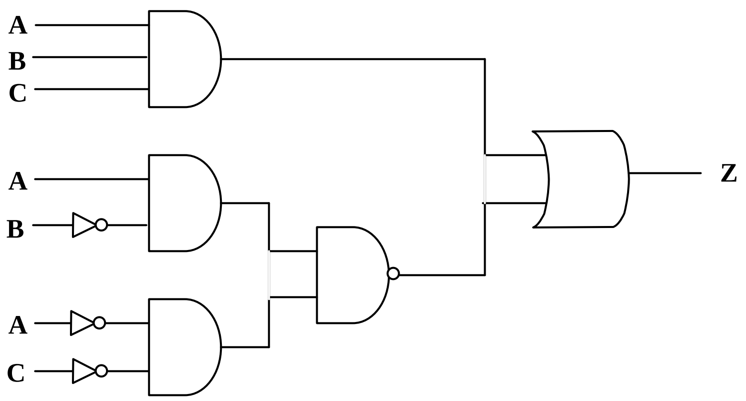
- Expresar la función en forma de SOP o POS.
- Utilizar el teorema 6, para eliminar variables, duplicando términos que puedan agruparse,
-Aplicar la ley distributiva

Minimización Algebraica

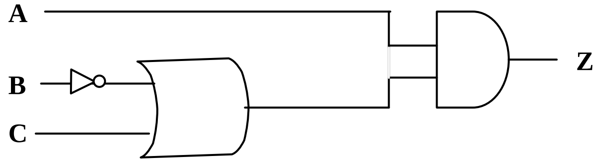
Ejemplo: z = a · b · c + a · b · (ā · c)
Paso1:
Z = a · b · c + a · b ·(a + c)
Z = a · b · c + a · b + a · b · c
Paso 2:
Z = a · b · c + a · b · c + a · b · c
Z = a · b · c + a · b + a · b · c
Z = a · c · (b + b) + a · b · (1 + c)
Z = a · c + · a · b
Paso 3:
Z = a · (c + b)

Minimización Algebraica

Implementación original:



Implementación minimizada:

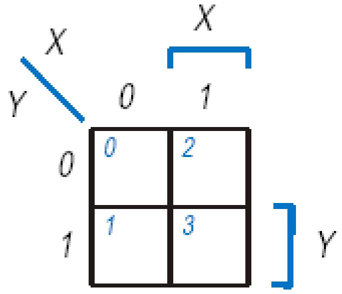


Minimización por Mapas de Karnaugh

- Un mapa de Karnaugh es una representación gráfica de la tabla de verdad de una función de conmutación.

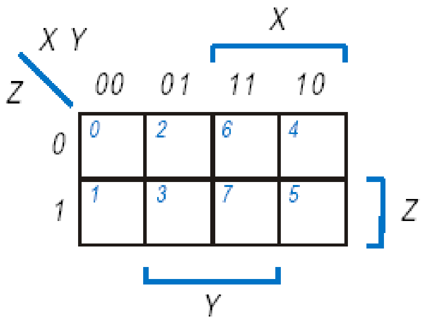
- Para 2 variables:

X	Y	Minter
0	0	0
0	1	1
1	0	2
1	1	3



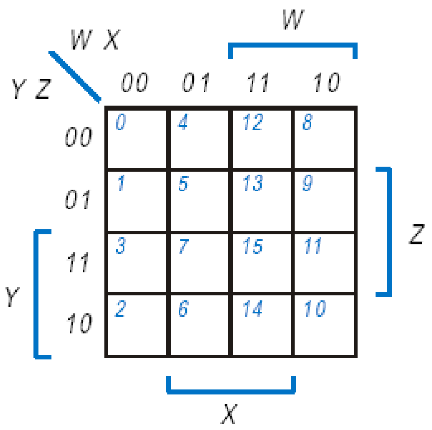
- Para 3 variables

X	Y	Z	Minter
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	**3**
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7



- Para 4 variables

W	X	Y	Z	Minter
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	**3**
0	1	0	0	4
⋮	⋮	⋮	⋮	⋮
1	1	1	1	15



Minimización por mapas de Karnaugh

- Coloque 1’s en las celdas correspondientes a los mintérminos de la función.
- Agrupe en un elipse lo mas grande posible, en conjuntos rectangulares de 1’s,

- # de 1’s en cada conjuntos debe ser potencia de 2,
- Se permite cursar elipses.

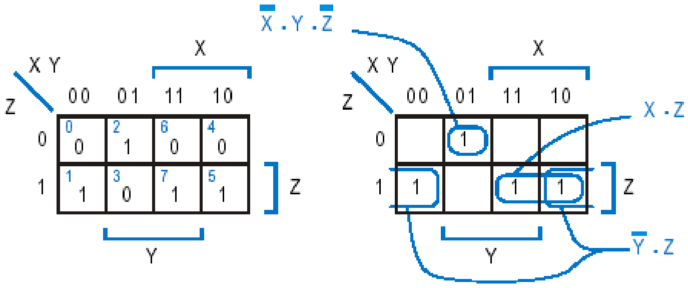
- El térmico producto resultante tendrá:

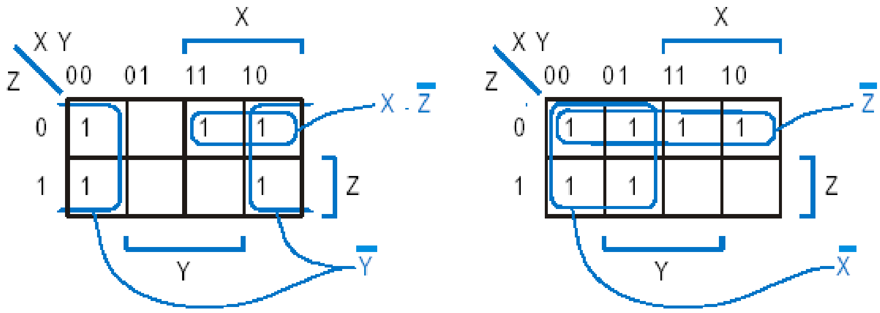
-Si la variable es 1 => incluya la variable,
- Si la variable es 0 => incluya la variable complementada
- Si la variable es tanto 0 y 1 => no incluya la variable.

- Las elipses correspondientes a los términos productos se llaman “implicantes primos”.

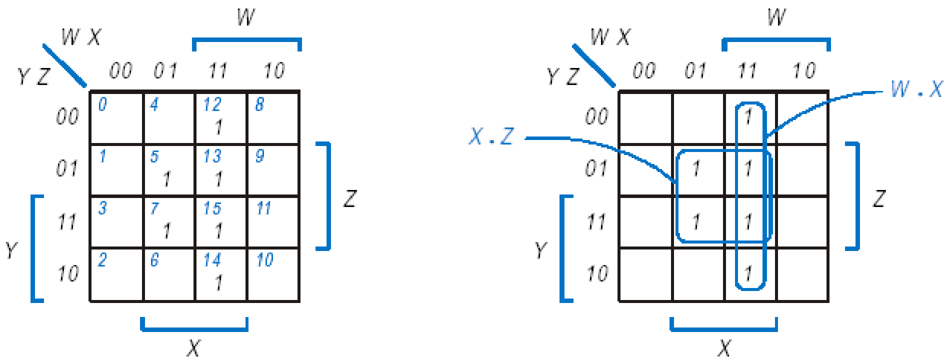
- Ejemplos

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

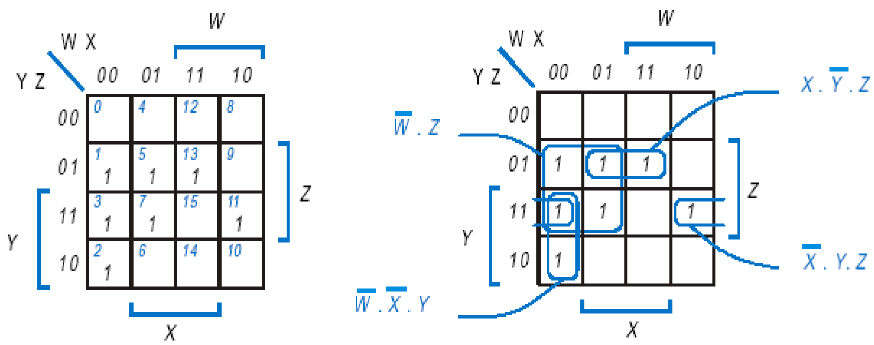




Minimización por mapas de Karnaugh

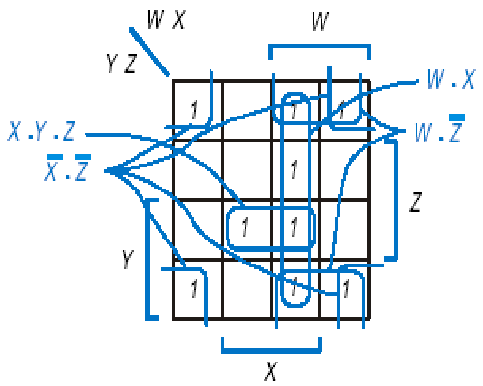


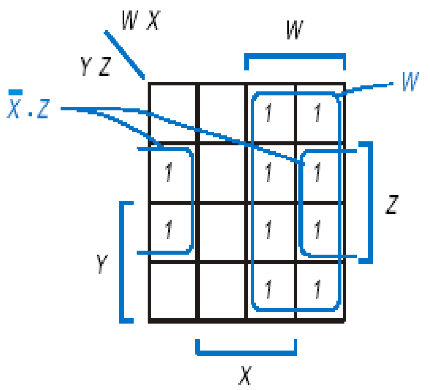
F (W,X,Y,Z) = Σm(5,7,12,13,14,15)



F(W,X,Y,Z) = Σm(1,2,3,5,7,11,13)

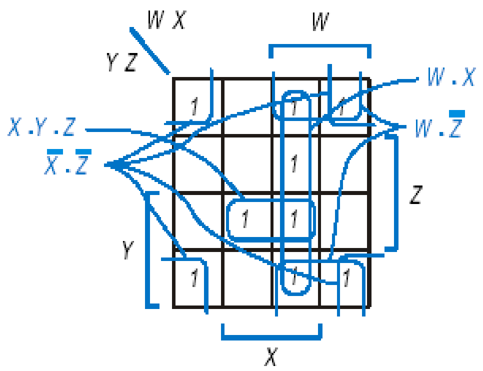
Minimización por mapas de Karnaugh

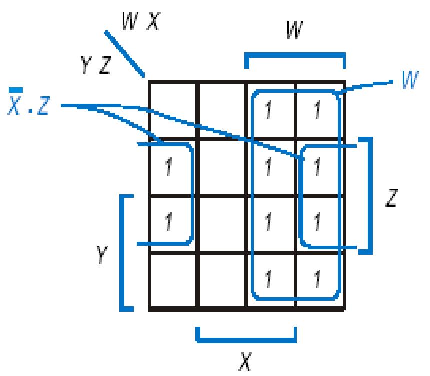




Minimización por mapas de Karnaug

<u>Suma total:</u> Suma de los implicantes primos



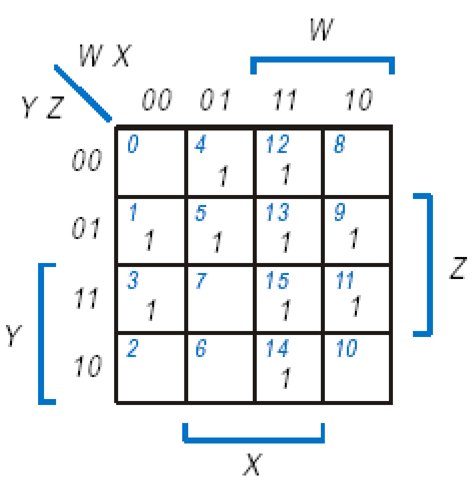


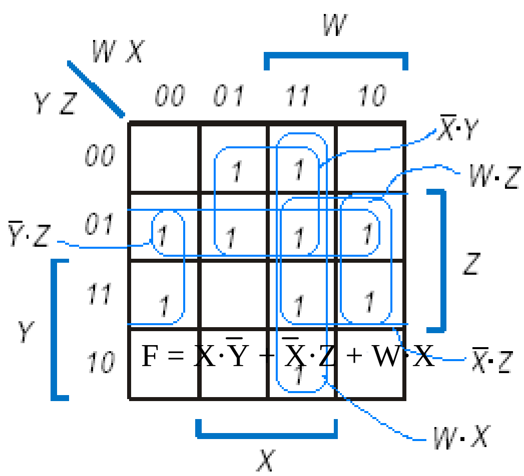
$f(w,x,y,z) = x\cdot y\cdot z + \overline{x}\cdot\overline{z} + w\cdot x + w\cdot\overline{z}$ $\qquad$ $f(w,x,y,z) = w + x \cdot \overline{z}$

Minimización por mapas de Karnaugh

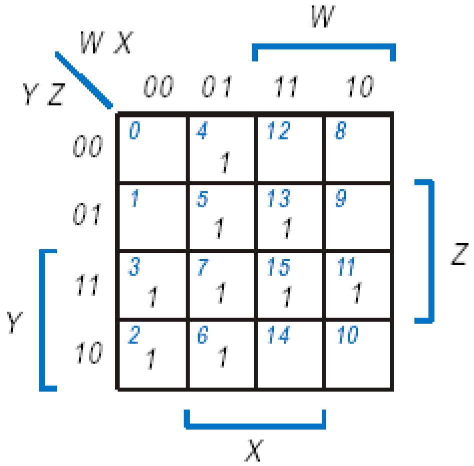
- <u>Celdas 1 distinguidas:</u> celdas 1 que están cubiertas por un único implicante primo.
- <u>Implicante primo esencial (IPE):</u> implicante que contenga al menos una celda 1 distinguida.

- <u>Suma mínima:</u> Suma de los IPE.

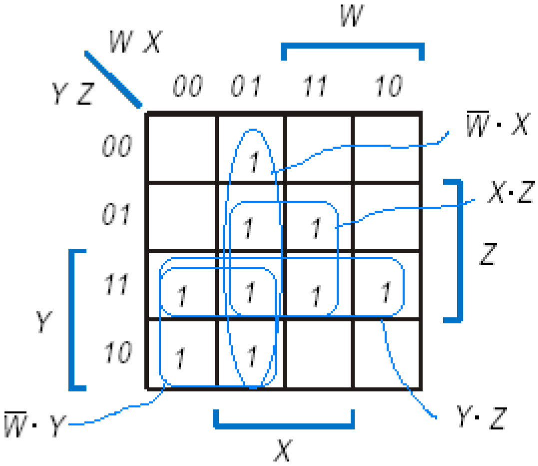




Minimización por mapas de Karnaugh



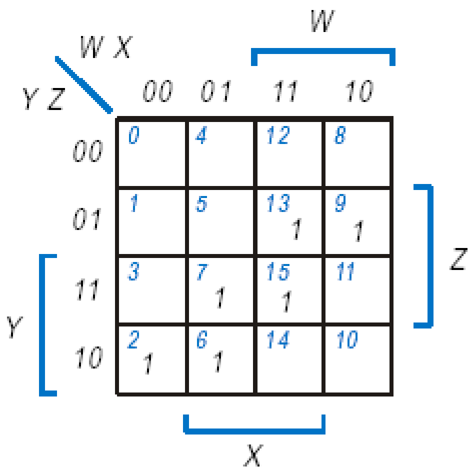
$F(W,X,Y,Z) = \Sigma m(2,3,4,5,6,7,11,13,1$



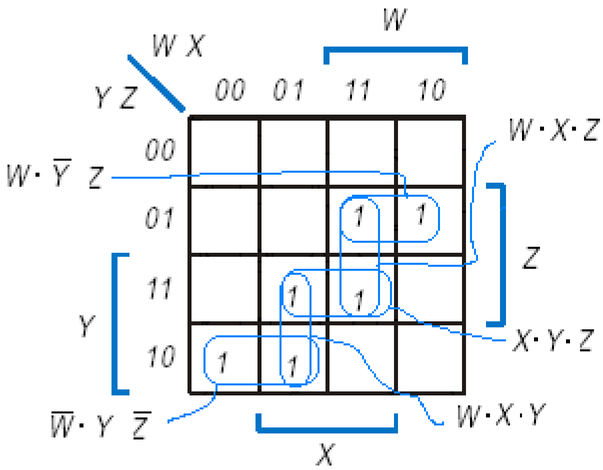
$F(W,X,Y,Z) = \Sigma m(\overline{W}\cdot Y + \overline{W}\cdot X + X\cdot Z + Y\cdot Z)$

Minimización por mapas de Karnaugh

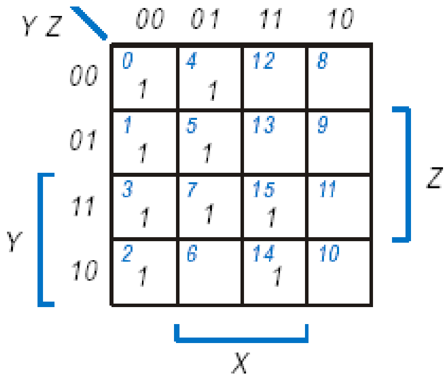
- Implicantes primos esenciales secundarios (IPES),
- Suma Mínima = IPE + IPES



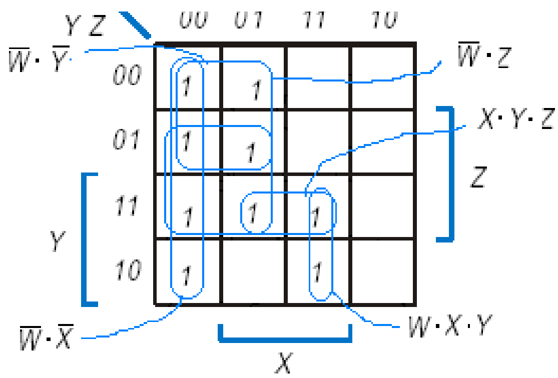
$F(W,X,Y,Z) = \Sigma m(2,6,7,9,13,15)$



$F(W,X,Y,Z) = \Sigma m(W\cdot\overline{Y}\cdot Z + \overline{W}\cdot Y\cdot\overline{Z} + X\cdot Y\cdot Z)$

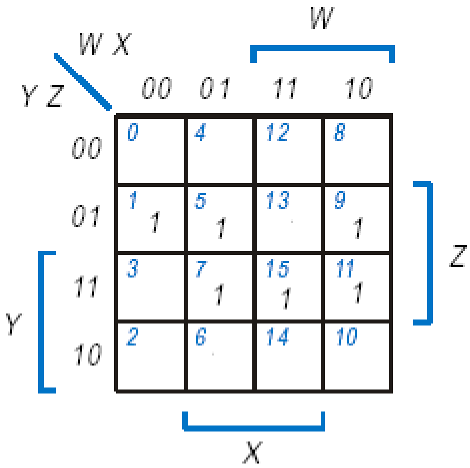


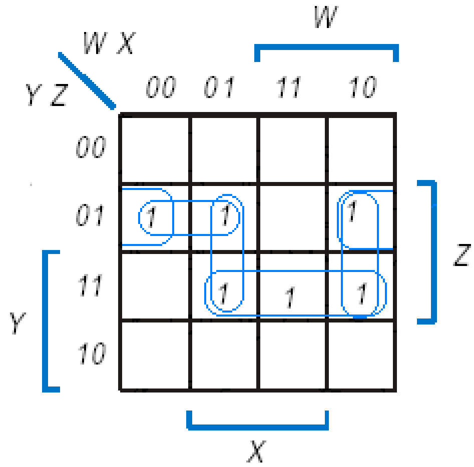
$F(W,X,Y,Z) = \Sigma m(0,1,2,3,4,5,7,14,15)$

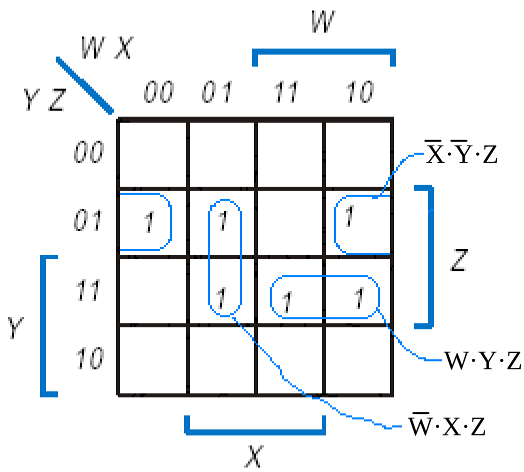


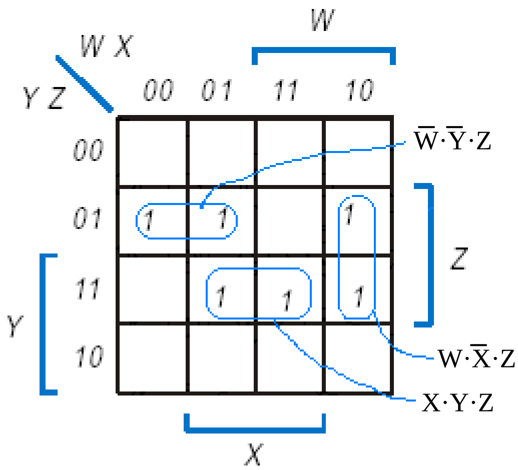
$F(W,X,Y,Z) = \Sigma m(\overline{W}\cdot\overline{Y} + \overline{W}\cdot\overline{X} + W\cdot X\cdot Y + \overline{W}\cdot Z)$

Minimizaciones por mapas de Karnaugh

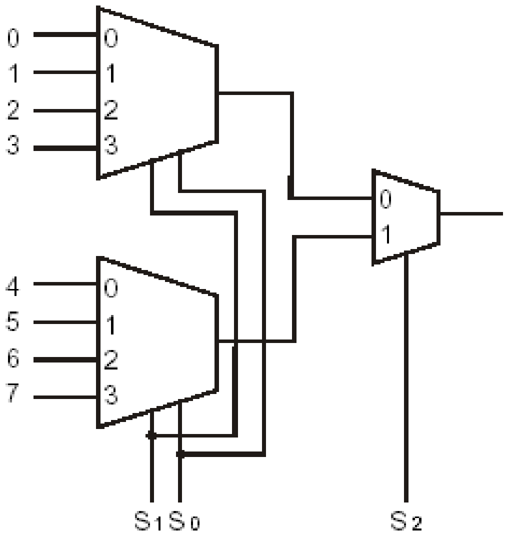






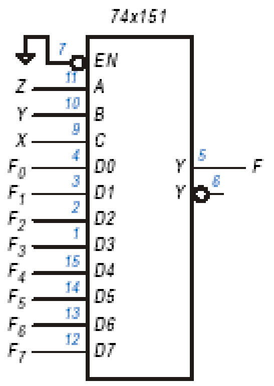


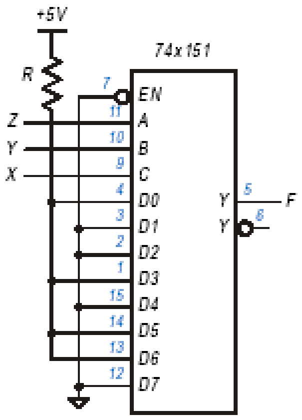
Expansión con Multiplexores



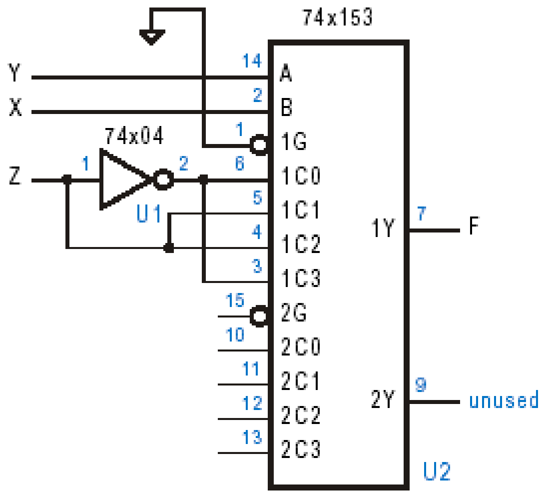
Funciones con Multiplexores

	X	Y	Z	F
F0	0	0	0	1
F1	0	0	1	0
F2	0	1	0	0
F3	0	1	1	1
F4	1	0	0	0
F5	1	0	1	1
F6	1	1	0	1
F7	1	1	1	0





Row	X	Y	Z	F	
0	0	0	0	1	Z¢
1	0	0	1	0	
2	0	1	0	0	Z
3	0	1	1	1	
4	1	0	0	0	Z
5	1	0	1	1	
6	1	1	0	1	Z¢
7	1	1	1	0	



ANTOLOGÍA (SISTEMAS NUMÉRICOS), POR CARLOS CEDILLO NAKAY Y MÓNICA TALIA VIOLETA SIERRA PEÓN

Enviado por:
Ing.+Lic. Yunior Andrés Castillo S.
“NO A LA CULTURA DEL SECRETO, SI A LA LIBERTAD DE INFORMACION”®
www.monografias.com/usuario/perfiles/ing_lic_yunior_andra_s_castillo_s/monografias
Página Web: yuniorandrescastillo.galeon.com
Correo: yuniorcastillo@yahoo.com
yuniorandrescastillosilverio@facebook.com
Twitter: @yuniorcastillos
Celular: 1-829-725-8571
Santiago de los Caballeros,
República Dominicana,
2015.
“DIOS, JUAN PABLO DUARTE Y JUAN BOSCH – POR SIEMPRE”®