

# 5

## Álgebra de Boole

### Contenido

5.1 Introducción .....	88
5.2 Álgebra de los circuitos digitales .....	88
5.3 Álgebra de Boole .....	89
5.4 Función booleana.....	92
5.5 Compuertas lógicas o <i>gates</i> .....	93
5.6 Circuito lógico.....	99
5.7 Circuito sumador-binario en paralelo .....	102
5.8 Formas normales o canónicas de una función.....	104
5.9 Circuitos equivalentes.....	107
5.10 Minimización de circuitos.....	108
5.11 Resumen.....	110
5.12 Ejercicios propuestos.....	111
5.13 Contenido de la página Web de apoyo.....	112

### Objetivos

- Brindar una introducción al Álgebra proposicional y al Álgebra de Boole.
- Formular, resolver y minimizar en la confección de los circuitos digitales que intervienen en la implementación de una computadora.



En la página Web de apoyo encontrará un breve comentario de la autora sobre este capítulo.



La palabra "digital" describe la tecnología que genera, almacena y procesa datos en términos de 0 y 1, denominados bits. Antes, la transmisión electrónica se denominaba analógica; en esta tecnología los datos se convertían en una señal, de frecuencia o amplitud, variable sobre una onda portadora de una frecuencia determinada. La radio y el teléfono utilizaban tecnología analógica.

## 5.1 Introducción

En este capítulo se presentan los temas del Álgebra de Boole requeridos en el diseño de circuitos lógicos del capítulo Lógica Digital en el que se analizarán y diseñarán otros sistemas digitales sencillos. Se presentan los operadores lógicos básicos y los dispositivos que los implementan, las compuertas lógicas. Los circuitos que se presentan en este capítulo son "combinacionales", o sea que el valor de las salidas sólo depende del valor de las entradas. Para representar un circuito se utiliza el diagrama de bloque, que sólo indica las entradas y las salidas del circuito, y el diagrama de lógica. Luego se explica el método basado en mapas de Karnaugh para simplificación de funciones. Para la representación de circuitos se utiliza la norma establecida por el ANSI (*American National Standard Institute*).

## 5.2 Álgebra de los circuitos digitales

### 5.2.1 Elementos de Álgebra proposicional

El Álgebra proposicional estudia la operación entre elementos denominados **proposiciones**. Los siguientes enunciados son ejemplos de ellos:

"Los gatos son animales"

"Los perros son animales"

Una proposición es una sentencia declarativa de la que tiene sentido decir si es verdadera o falsa. Esto implica que es posible dar a una proposición un **valor de verdad**. Sin embargo, el uso del lenguaje puede llevar a conclusiones absurdas. Si se reemplaza "son" por "=", "**Los gatos**" por "**a**", "**Los animales**" por "**b**" y "**Los perros**" por "**c**", se puede escribir:  $a = b$ ,  $c = b$  y, por lo tanto, se puede concluir erróneamente que  $a = c$ , es decir que los gatos son perros.

Por esta razón, el Álgebra proposicional vacía las proposiciones de contenidos semánticos y sólo estudia su relación utilizando una serie de operadores denominados **operadores lógicos**.

Una proposición constituida por dos o más proposiciones simples relacionadas por operadores lógicos se denomina **proposición compuesta**, y se le puede adjudicar un valor de verdad. Algunos operadores lógicos son:

- La conjunción ( $\wedge$ ).
- La disyunción inclusiva ( $\vee$ ).
- La disyunción exclusiva ( $\vee\!\!\!/\!$ ).

Dadas dos proposiciones, vacías de contenido semántico y simbolizadas como  $p$  y  $q$ , se puede confeccionar una **tabla de verdad** que represente las cuatro combinaciones de valores de verdad:

- Si ambas son falsas.
- Si  $p$  es falsa y  $q$  verdadera.
- Si  $p$  es verdadera y  $q$  es falsa.
- Si ambas son verdaderas.

Para cada una de estas combinaciones se describe un valor de verdad para cada operador. Considérese a  $F$  como el literal que representa el valor de verdad falso y a  $V$  como el literal que representa el valor de verdad verdadero. En la tabla 5-1 se muestra el valor de verdad de cada proposición compuesta para los operadores mencionados:

**Tabla 5-1. Tabla de verdad para dos proposiciones.**

$p$	$q$	$p \wedge q$	$p \vee q$	$p \vee \neg q$
F	F	F	F	F
F	V	F	V	V
V	F	F	V	V
V	V	V	V	F

El operador **negación** se aplica a una sola proposición, simple o compuesta. El valor de verdad  $F$ (falso) es la negación de  $V$ (verdadero) y el valor de verdad  $V$ (verdadero) es la negación de  $F$ (falso). Como se observa en la tabla 5-2 el símbolo  $\sim p$  representa la negación de la proposición  $p$ :

**Tabla 5-2. Tabla de verdad de negación.**

$p$	$\sim p$
F	V
V	F

## 5.3 Álgebra de Boole

George Boole, en el siglo XIX, se dedicó a formalizar y mecanizar el proceso del pensamiento lógico y desarrolló una teoría lógica que utilizaba variables en lugar de proposiciones. Los valores que puede tomar una variable “booleana” son  $0$  o  $1$ .

En 1938, Claude Shannon propuso la utilización del Álgebra de Boole en el análisis de circuitos eléctricos, lo que años más tarde permitió profundizar en el análisis y el desarrollo de computadoras electrónicas.

El Álgebra de Boole es un ente matemático constituido por los elementos  $0$  y  $1$  y las operaciones suma lógica, producto lógico y complemento.

Relacionando lo expresado antes, las proposiciones del Álgebra proposicional son variables en el Álgebra de Boole y los valores de verdad que se asignaban a proposiciones son los que pueden tomar las variables. El  $0$  se relaciona con el falso y el  $1$  con el verdadero. El Álgebra de Boole constituye el fundamento teórico para el diseño de circuitos digitales. El valor del Álgebra de Boole, plasmado en su “Investigación sobre las leyes del pensamiento”, consiste en lograr un análisis lógico con un soporte matemático desconocido para su época. Su obra fue interpretada como base teórica del desarrollo de la Cibernetica.

### 5.3.1 Operadores

Los operadores “booleanos” son:

- Producto lógico ( $\cdot$ ).
- Suma lógica ( $+$ ).
- Complemento ( $\sim$ ).

Otra vez, haciendo referencia al Álgebra proposicional podemos relacionar los operadores “booleanos” con los operadores lógicos, ya vistos en la sección anterior.

Desde el punto de vista semántico, los nombres de las variables utilizan las primeras letras del abecedario en minúscula, por eso, reemplazamos “ $p$ ” por “ $a$ ” y “ $q$ ” por “ $b$ ”.

El “producto lógico” o “booleano” está relacionado con la tabla de verdad de la conjunción y se simboliza con un punto ( $\cdot$ ).

La “suma lógica” o “booleana” está relacionada con la tabla de verdad de la disyunción inclusiva y se simboliza con un signo más ( $+$ ).

**Boole** (1815-1864). Matemático británico que es considerado uno de los fundadores de las Ciencias de la Computación, debido a su creación del Álgebra booleana, la cual es la base de la Aritmética computacional moderna.

**Shannon** (1916-2001). Ingeniero eléctrico y matemático estadounidense, considerado el fundador de la teoría de la información.

Demostró que el Álgebra booleana se podía utilizar en el análisis y la síntesis de la conmutación de los circuitos digitales, idea que fue calificada como una de las aportaciones teóricas fundamentales que ayudó a cambiar el diseño de los circuitos digitales.

El “complemento” es un operador equivalente a la negación. En las tablas 5-3 y 5-4 se representan los valores de verdad para cada operador en términos “booleanos”.

**Tabla 5-3. Tabla para dos variables.**

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

**Tabla 5-4. Tabla de verdad.**

a	$\bar{a}$
0	1
1	0

### 5.3.2 Tablas de verdad

Una tabla de verdad es: “Una lista ordenada de las  $2^n$  combinaciones distintas de ceros y unos, que se pueden obtener de la combinación del valor de n variables binarias”.

Para 3 variables, y considerando que los valores que puede tomar cada una son sólo 0 o 1, la cantidad de combinaciones binarias distintas es de  $2^3 = 8$ , para 4 variables la cantidad de combinaciones es, entonces,  $2^4 = 16$ . En términos generales, con n variables se pueden obtener  $2^n$  combinaciones diferentes.

### 5.3.3 Propiedades del Álgebra de Boole

Los enunciados que se expresan a continuación pertenecen a las propiedades que todo elemento o toda operación del Álgebra de Boole deben cumplir.

#### Leyes comutativas

Ambas operaciones binarias (suma y producto) son comutativas, esto es que si  $a$  y  $b$  son elementos del Álgebra se verifica que:

$$a + b = b + a$$

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

#### Leyes Distributivas

Cada operación binaria (suma y producto) es distributiva respecto de la otra:

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

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

#### Leyes de Identidad

Dentro del Álgebra existen dos elementos neutros, el 0 y el 1, que cumplen la propiedad de identidad con respecto a cada una de las operaciones binarias:

$$a + 0 = a$$

$$a \cdot 1 = a$$



Encuentre un simulador que permite crear el diagrama lógico de una expresión booleana en la Web de apoyo.

### Leyes De Complementación

Para cada elemento  $a$  del Álgebra, existe un elemento denominado a negada, tal que:

$$a + \bar{a} = 1 \quad a \text{ y } \bar{a} \text{ no pueden ser cero al mismo tiempo,}$$

$$a \cdot \bar{a} = 0 \quad a \text{ y } \bar{a} \text{ no pueden ser uno al mismo tiempo.}$$

Estas dos leyes definen el **complemento** de una variable.

### 5.3.4 Teoremas del Álgebra de Boole

Sobre la base de los postulados anteriores, se deduce una serie de teoremas. La demostración de cada teorema se puede realizar en forma algebraica o mediante la tabla de verdad.

#### Teorema 1

De las expresiones anteriores se deduce el “principio de dualidad”, que determina que estas expresiones permanecen válidas si se intercambian las operaciones  $+$  por  $\cdot$  y los elementos 0 por 1.

#### Teorema 2

Para cada elemento del Álgebra de Boole se verifica que:

$$a + 1 = 1$$

$$a \cdot 0 = 0$$

Según este teorema y las “leyes de identidad” citadas anteriormente, se deduce que:

$$0 + 0 = 0 \quad 0 \cdot 0 = 0$$

$$0 + 1 = 1 \quad 0 \cdot 1 = 0$$

$$1 + 1 = 1 \quad 1 \cdot 1 = 1$$

#### Teorema 3

Para cada elemento  $a$  de un Álgebra de Boole se verifica que:

$$a + a = a \quad \text{Ley de idempotencia}$$

$$a \cdot a = a$$

#### Teorema 4

Para cada par de elementos del Álgebra de Boole,  $a$  y  $b$ , se verifica que:

$$a + (a \cdot b) = a \quad y \quad \text{Ley de absorción}$$

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

Confeccionando la tabla de verdad se puede demostrar, por ejemplo, para la primera igualdad (tabla 5-5):

<b>Tabla 5-5</b>			
$a$	$b$	$a \cdot b$	$a + a \cdot b$
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

### Teorema 5

En el Álgebra de Boole la suma y el producto son asociativos:

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

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

### Teorema 6

Para todo elemento a de un Álgebra de Boole, se verifica que:

$$\bar{\bar{a}} = a \quad \text{Ley de involución}$$

Comprobación:



**De Morgan** (1806-1871). Profesor de Matemáticas en el Colegio Universitario de Londres entre 1828 y 1866 y primer Presidente de la Sociedad de Matemáticas de Londres. Se interesó especialmente por el Álgebra y escribió varias obras de Lógica.

En la moderna Lógica Matemática, llevan el nombre de De Morgan las siguientes leyes fundamentales del Álgebra de la Lógica: «la negación de la conjunción es equivalente a la disyunción de las negaciones»; «la negación de la disyunción es equivalente a la conjunción de las negaciones».

### Teorema 7

En todo Álgebra de Boole se verifican las siguientes igualdades, conocidas como “Leyes de De Morgan”, que permiten transformar sumas en productos y productos en sumas:

$$\begin{array}{c} a + b + c + \dots = \bar{a} \cdot \bar{b} \cdot \bar{c} \cdot \dots \\ a \cdot b \cdot c \cdot \dots = \bar{a} + \bar{b} + \bar{c} + \dots \end{array}$$

## 5.4 Función booleana

Una función “booleana” es: “*Una expresión algebraica constituida por variables binarias, los operadores binarios suma lógica y producto lógico, el operador complemento, el paréntesis y el signo igual*”.

La descripción del valor que asume una función se representa en una tabla de verdad y se denomina representación estándar; por ejemplo, en la tabla 5-6 se describe el valor de las funciones  $f_2 = a \cdot \bar{b}$ ;  $f_4 = \bar{a} \cdot b$ ;  $f_{12} = a + \bar{b}$  y  $f_{15} = 1$ .

**Tabla 5-6. Descripción del valor que asume una función.**

a	b	$f_0$	$f_1$	$f_2$	$f_3$	$f_4$	$f_5$	$f_6$	$f_7$	$f_8$	$f_9$	$f_{10}$	$f_{11}$	$f_{12}$	$f_{13}$	$f_{14}$	$f_{15}$
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	1	
		0	and			a			b	xor	or	nor	xnor	$\bar{b}$	$\bar{a}$	nan	1

“*Para n variables se pueden definir  $2^n$  funciones distintas*”. En la tabla 5-7 se describen las diecisésis funciones que se pueden definir para dos variables.

**Tabla 5-7. Funciones que se pueden definir para dos variables.**

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



Cada una de estas funciones se puede expresar algebraicamente así:

$$\begin{aligned}
 f_0 &= 0 \\
 f_1 &= a \cdot b \\
 f_2 &= a \cdot \bar{b} \\
 f_3 &= a \\
 f_4 &= \bar{a} \cdot b \\
 f_5 &= b \\
 f_6 &= a \cdot \bar{b} + \bar{a} \cdot b \\
 f_7 &= \overline{a + b} \\
 f_8 &= a + b \\
 f_9 &= a \cdot b + \bar{a} \cdot \bar{b} \\
 f_{10} &= \bar{b} \\
 f_{11} &= a + \bar{b} \\
 f_{12} &= \bar{a} \\
 f_{13} &= \bar{a} + b \\
 f_{14} &= a \cdot \bar{b} \\
 f_{15} &= 1
 \end{aligned}$$

## 5.5 Compuertas lógicas o gates

Una compuerta lógica es la “representación” de una red de conmutadores. Cada conmutador se controla con una señal binaria identificada como  $a, b, c, \dots, n$ . Cada una de estas señales constituye una “entrada” de la compuerta o “terminal de entrada”. La estructura de la red de conmutadores genera una señal binaria de salida, o “terminal de salida”, identificada como  $f$ , que es una función de  $a, b, c, \dots, n$ . Para cada combinación de entradas, la función  $f(a, b, c, \dots, n)$  asume un valor 0 o 1. La tabla de verdad de la función  $f$  representa el comportamiento de la compuerta para cada combinación de valores.

El Álgebra de Boole enunciada en este libro en su forma binaria es equivalente, por definición, al Álgebra de conmutadores.



Una compuerta lógica es el bloque elemental que permite la implementación de circuito digital. La mayoría tiene dos entradas y una salida, que también se denominan “terminales”; cada terminal puede estar en uno de dos estados, 0 o 1, representados por diferentes niveles de voltajes.

### Comutador

Es un dispositivo físico que permite controlar el flujo de un elemento (corriente eléctrica, agua, etc.). En la vida cotidiana vemos innumerables ejemplos de conmutadores, por ejemplo, en una habitación la luz se prende o apaga con un conmutador manual.

### Comutador ideal

Es un modelo teórico que representa el comportamiento de un modelo real. El estudio del funcionamiento de un conmutador ideal es independiente de la tecnología que se utilice para su implementación. En un conmutador ideal se utilizan los dígitos 0 y 1 para representar el estado de una entrada o una salida y, también, para representar el estado del conmutador abierto o cerrado.



Encuentre una animación sobre cómo trabajan los interruptores no mecánicos en la Web de apoyo.



Los terminales de entrada de un circuito aceptan señales binarias, dentro de tolerancias de voltajes permitidas, y generan en los terminales de salida señales binarias que cumplen también el convenio de voltaje establecido para ese circuito.



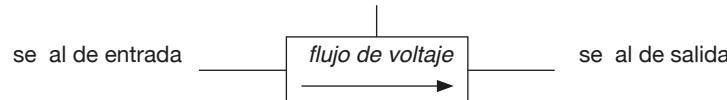
La información binaria se representa en un sistema digital por cantidades físicas denominadas señales.

## Circuitos de conmutadores

A una agrupación de varios conmutadores relacionados entre sí se la denomina circuito de conmutación. El Álgebra de Boole constituye el fundamento teórico para su diseño. Los primeros circuitos de conmutación se diseñaron con contactos, el elemento 0 representa un contacto que está abierto y el elemento 1 representa un contacto que está cerrado.

Señal

En lugar de cambiar el estado de un conmutador manualmente, como ocurre con el interruptor de la luz de una habitación, en un conmutador electrónico se utiliza una señal binaria, que se denomina señal de control o **señal de entrada**. Esta señal regula el flujo de un voltaje por medio del conmutador, que es, a su vez, otra señal binaria que se denomina señal de dato o **señal de salida** del conmutador.

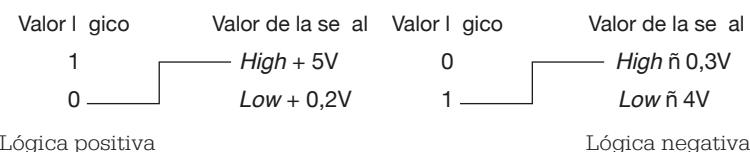


En la figura anterior se muestra un típico símbolo gráfico de conmutador. La flecha muestra el sentido de circulación de las señales binarias y las líneas más gruesas representan los **terminales** de acceso o egreso de las señales.

## Lógica positiva

Los componentes electrónicos binarios operan solamente con dos niveles de tensión de señales, llamados nivel lógico “1” y nivel lógico “0”.

Cuando se indica que un conmutador opera con **lógica positiva**, la tensión lógica "1" es más positiva que la tensión lógica "0", lo que significa que la tensión lógica "0" está próxima a la tensión real cero y la tensión lógica "1" se sitúa en un cierto nivel de tensión positiva. Si, por el contrario, el conmutador trabaja con lógica negativa, la tensión lógica "1" es menos positiva que la tensión lógica "0". Lo que determina el tipo de lógica es la asignación de los valores lógicos de acuerdo con las amplitudes relativas de las señales.



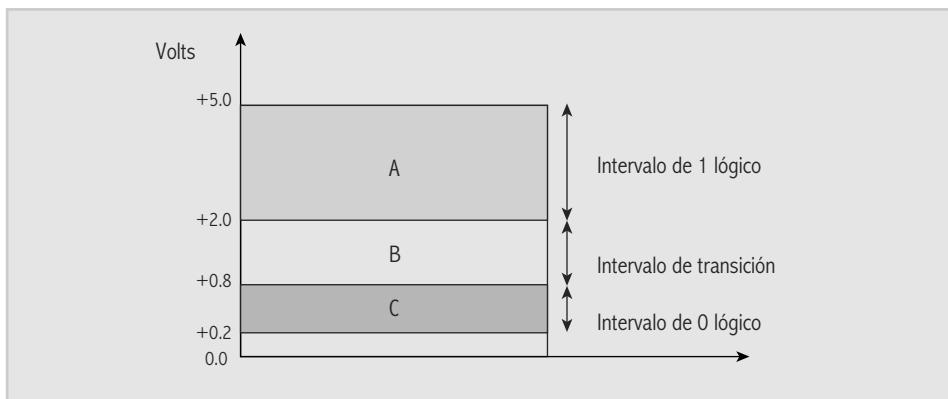
## Convenios de voltaje

Como la tensión es una magnitud continua sujeta a fluctuaciones, se establece un rango de variabilidad para cada estado lógico. Cuando una señal cambia de uno a otro estado se encuentra, en determinado momento, fuera de los rangos permitidos; esta etapa se denomina período de transición y no es tomada en cuenta como valor lógico para la señal.

En la figura 5.1 se observa un ejemplo de señal binaria en un convenio de voltaje que utiliza muchos **circuitos integrados**.

- Las zonas A y C corresponden a los intervalos considerados aceptables para una señal de salida

- La zona B corresponde al intervalo de transición, o sea, cierto intervalo en el que el valor de la señal pasa de un estado lógico al otro y no es tomado en cuenta.



**Fig. 5.1.** Convenio de voltaje.

Este diagrama pertenece a las llamadas señales de “lógica positiva”, donde la tensión lógica “1” es más positiva que la tensión lógica “0”.

#### 5.5.1 “Compuerta AND”, “compuerta y” o “compuerta producto lógico”

La tabla de verdad de la función producto lógico representa el comportamiento de la salida de esta compuerta:

##### Compuerta AND

*La salida de la compuerta (función AND) sólo toma el valor 1 lógico cuando todas las entradas son 1 lógico.*

Si la compuerta tiene, por ejemplo, 2 entradas será (figura 5.2):



**Fig. 5.2.** Compuerta AND.

*La salida de esta compuerta responde a la tabla de verdad del producto lógico, para cada par de valores correspondientes a las cuatro combinaciones posibles de dos entradas.*

#### 5.5.2 “Compuerta OR”, “compuerta +” o “compuerta suma lógica”

La tabla de verdad de la función suma lógica representa el comportamiento de la salida de esta compuerta:

##### Compuerta OR

*La salida de la compuerta (función OR) sólo toma el valor 0 lógico cuando todas las entradas son 0 lógico.*

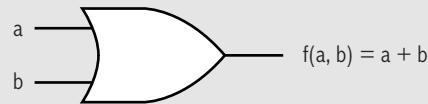


La región intermedia entre las dos permitidas se cruza solamente durante la transición de estado de 0 a 1 y de 1 a 0.



Un circuito digital particular puede emplear una señal de +5 Volts, para representar el binario “1”, y +0.8 Volts, para el binario “0”

Si la compuerta tiene solo dos entradas, será (fig. 5.3):



**Fig. 5.3.** Compuerta OR.

La salida de esta compuerta responde a la tabla de verdad de la suma lógica, para cada par de valores correspondientes a las cuatro combinaciones posibles de dos entradas.

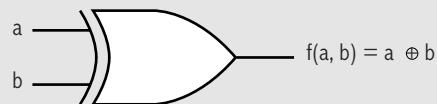
#### 5.5.3 “Compuerta OR EXCLUSIVE” o “compuerta exclusiva”

En realidad, el comportamiento de esta compuerta no responde a ningún operador “booleano”. Sin embargo, su comportamiento se puede comprender a partir de la función “disyunción exclusiva” del Álgebra proposicional. Recuerde que esa función sólo es falsa cuando el valor de verdad de todas las proposiciones es el mismo; en el lenguaje de lógica de compuertas esto se declara como:

#### Compuerta XOR

*La salida de la compuerta (función XOR) sólo toma el valor 0 lógico cuando **todas** las entradas son 0 lógico, o bien cuando una cantidad par de entradas son 1 lógico.*

Si la compuerta tiene, por ejemplo, dos entradas, será (fig. 5.4):



**Fig. 5.4.** Compuerta XOR.

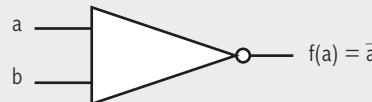
La salida de esta compuerta responde a la tabla de verdad de la suma “aritmética”, para cada par de valores correspondientes a las cuatro combinaciones posibles de dos entradas.

#### 5.5.4 “Compuerta NOT” o “inversión”

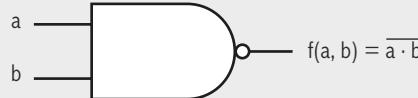
La tabla de verdad de la función complemento representa el comportamiento de la salida de esta compuerta (fig. 5.5):

#### Compuerta NOT

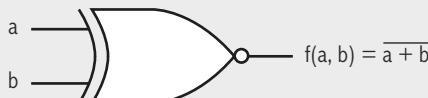
*Siempre tiene una sola entrada y una sola salida, siendo el valor de la salida el complemento del valor de la entrada.*

**Fig. 5.5.** Compuerta NOT.**5.5.5 “Compuertas con funciones negadas”**

Se pueden definir dos nuevas compuertas, que son las más utilizadas como elementos básicos para la realización de los circuitos actuales. Estas compuertas se denominan: NOR (NO-OR) y NAND (NO-AND).

**Compuerta NAND** (fig. 5.6)**Fig. 5.6.** Compuerta NAND.

*La salida de la compuerta (función NAND) sólo toma el valor 0 lógico cuando todas las entradas son 1 lógico.*

**Compuerta NOR** (fig. 5.7)**Fig. 5.7.** Compuerta NOR.

*La salida de la compuerta (función NOR) sólo toma el valor 1 lógico cuando todas las entradas son 0 lógico.*

Las tres funciones elementales, suma, producto e inversión lógica, pueden ser representadas con compuertas NOR y NAND. En efecto, aplicando el teorema de De Morgan se tiene:

**La negación de la suma de dos variables puede expresarse como el producto de las variables negadas.**

Negar la función suma dos veces. Por teorema 6 (involución), la doble negación de una función es igual a la función.

Aplicar el teorema enunciado a una de las negaciones.

La función queda expresada con un producto y tres negaciones.

Desde el punto de vista del circuito la compuerta OR fue reemplazada por compuertas NOT y NAND.

Veamos cómo aplicamos este procedimiento (fig. 5.8):

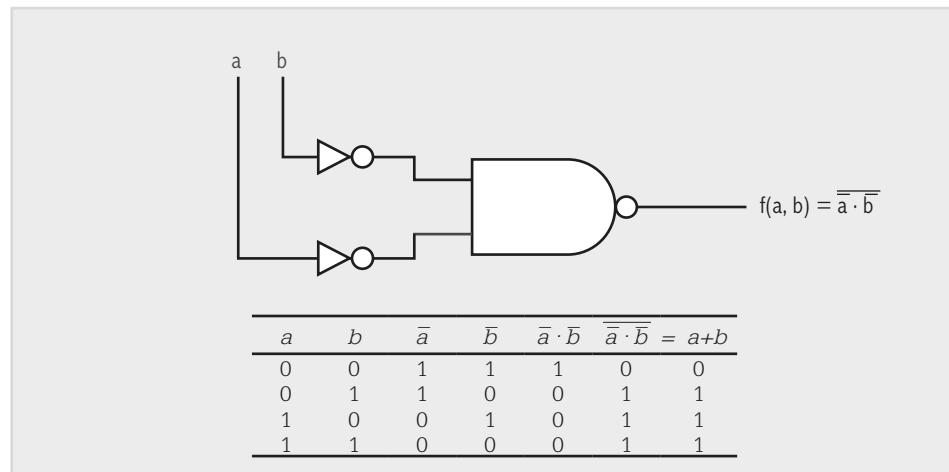


Fig. 5.8. Teorema de De Morgan aplicado a la suma lógica.

**La negación del producto de dos variables puede expresarse como la suma de las variables negadas.**

Negar la función producto dos veces. Por teorema 6 (involución) la doble negación de una función es igual a la función.

Aplicar el teorema enunciado a una de las negaciones.

La función queda expresada con una suma y tres negaciones.

Desde el punto de vista del circuito la compuerta AND fue reemplazada por compuertas NOT y NOR.

Veamos cómo aplicamos este procedimiento (fig. 5.9):

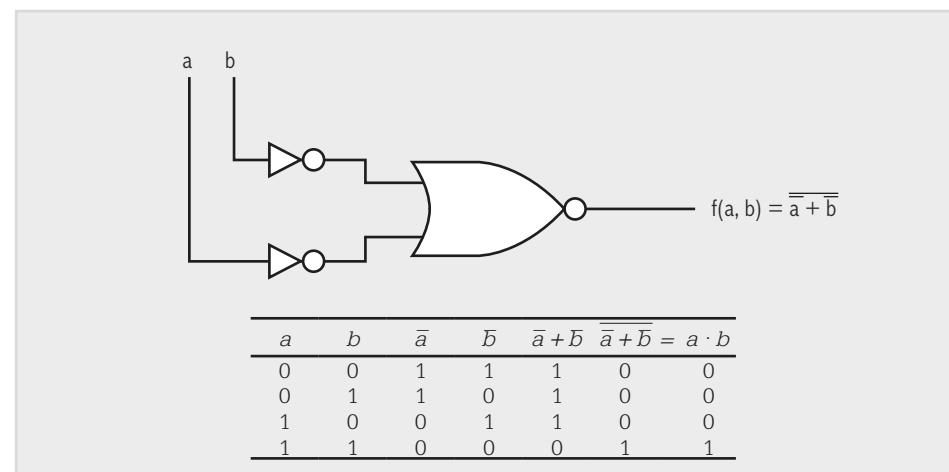
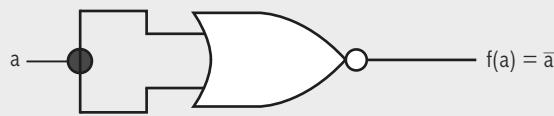


Fig. 5.9. Teorema de De Morgan aplicado al producto lógico.

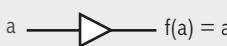
La inversión se realiza con una función NOR o NAND con sus dos entradas iguales. Veamos cómo se representa gráficamente para con una compuerta NOR (fig. 5.10):



**Fig. 5.10.** Implementación de la negación con compuertas negadas.

### Compuerta YES

Un símbolo triángulo por sí solo designa un circuito separador, que no produce ninguna función lógica particular, puesto que el valor binario de la salida es el mismo que el de la entrada. Este circuito se utiliza simplemente para amplificación de la señal. Por ejemplo, un separador que utiliza 5 voltos para el binario 1 producirá una salida de 5 voltos cuando la entrada es 5 voltos. Sin embargo, la corriente producida a la salida es muy superior a la corriente suministrada a la entrada de la misma. De esta manera, un separador puede excitar muchas otras compuertas que requieren una cantidad mayor de corriente, que no se encontraría en la pequeña cantidad de corriente aplicada a la entrada del separador. Se podría pensar que una compuerta YES es inútil, si se considera que el mismo valor lógico de la entrada se establece en la salida. Sin embargo, los ceros y unos son corriente eléctrica a cierto voltaje. La corriente no es suficiente si el abanico de salida es grande. Se denomina "abanico de salida" a la cantidad de dispositivos a la que el terminal de salida de una compuerta lógica puede abastecer.



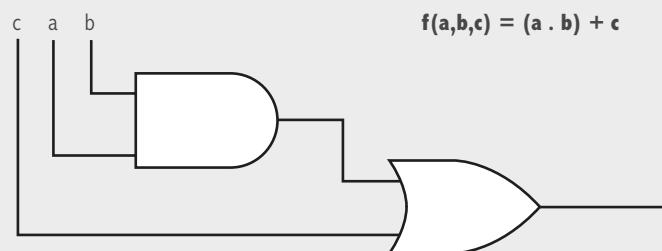
**Fig. 5.11.** Compuerta lógica YES.

## 5.6 Circuito lógico

Un circuito lógico es la representación de la estructura de una red de compuertas y su comportamiento se expresa algebraicamente por una o varias funciones booleanas; las variables constituyen las entradas y las funciones constituyen las salidas del circuito.

La expresión algebraica de la función relaciona las variables con los operadores suma lógica, producto lógico y complemento, que se representan en el circuito, también denominado **diagrama de lógica**, con las compuertas que representan a estos operadores.

Así, la función  $f = (a \cdot b) + c$  está definiendo un circuito de tres entradas y una salida, cuyo valor representa el que la función asuma en cada caso según la tabla de verdad adjunta (fig. 5.12):



**Fig. 5.12.** Diagrama de lógica.

El dibujo del circuito se denomina **diagrama de lógica** (tabla 5-8).

**Tabla 5-8. Diagrama de lógica**

a	b	c	f(a,b,c)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

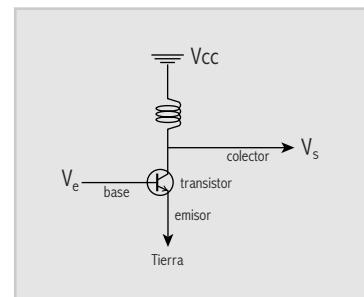
### 5.6.1 Transistor

Es un conmutador electrónico que controla el flujo de un nivel de tensión, con la particularidad de que puede conmutar (cambiar de estado) muy rápidamente.

Las compuertas se sustituyen por dispositivos electrónicos que, como explicaremos, se implementan con redes de transistores y permiten calcular una función de una, dos o más señales binarias. Estas funciones tienen relación directa con los operadores booleanos e incluso asumen nombres semejantes a ellos. Sin intención de incluir en esta unidad un tratado de técnicas digitales, observemos algunos conceptos que explican cómo se puede transferir, transformar y almacenar información binaria en una computadora.

Consideremos al transistor como el elemento básico en la construcción de una compuerta. Un transistor puede operar como si fuera un conmutador abierto (bit 0) o cerrado (bit 1), según el comportamiento de las señales binarias que actúen sobre él.

Obsérvese la figura 5.13:

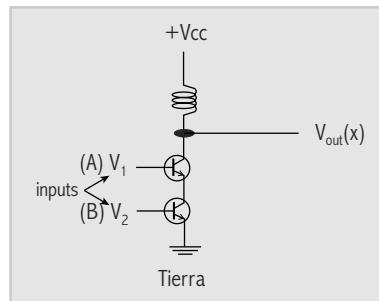


**Fig. 5.13.** Esquema de un transistor.

La tensión de alimentación se regula con la resistencia, cuando el voltaje de entrada está en un nivel alto, lo que permite que el transistor se encuentre en estado de conducción; la tensión en la salida cae, asume el valor de tierra, que sería 0 volts.

Cuando la entrada es igual a 1 lógico la salida es igual a 0 lógico en el caso inverso, cuando la entrada es igual a 0 lógico el transistor se corta y actúa como un interruptor abierto, la salida entonces asume el valor retenido de la tensión de alimentación equivalente a 1 lógico. Entonces, en el ejemplo mostrado, el transistor reproduce la función complemento del Álgebra de Boole.

Así, con distintas estructuras de transistores se pueden armar otras funciones booleanas; la más simple es la compuerta NAND, ejemplificada en la figura 5.14:



**Fig. 5.14.** Esquema de una compuerta NAND.

Con el mismo criterio, sólo se produce la caída de la salida cuando ambos transistores se encuentran en estado de “interruptor cerrado”, esto es, cuando las entradas son iguales a 1.

En el caso de una compuerta NOR es similar, sólo que los transistores están montados en paralelo, por lo tanto, basta que uno de ellos, o ambos, estén cerrados para que la salida asuma el valor 0. La única posibilidad de que la salida mantenga un valor semejante a la tensión de alimentación es que ambos transistores estén abiertos; por lo tanto, las dos señales de entrada deben estar en 0 lógico.

Para lograr las compuertas AND y OR se debe acoplar a la tensión de la salida de un circuito inversor. Por lo tanto, las compuertas AND y OR son estructuras más complejas que las anteriores. Así, se demuestra que la implementación de circuitos con compuertas NAND, o bien NOR, son estructuras más simples que sus equivalentes con AND y OR, y éste es el motivo por el cual las compuertas NAND y NOR se utilizan como “compuertas básicas” en el diseño de circuitos.

De lo expuesto, surge la razón tecnológica por la que se utiliza el sistema binario como elemento representativo de la información en una computadora, dado que, en la práctica, establecer sólo dos niveles de tensión es más confiable que establecer varios niveles que respondan sin ambigüedad y con toda precisión a la información que se quiera representar.

### 5.6.2 Compuerta triestado

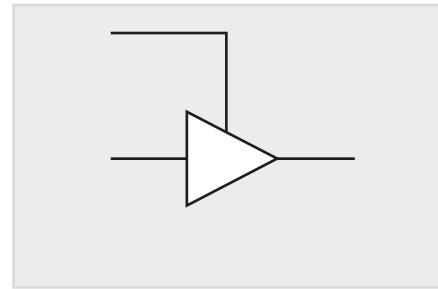
Aparece aquí un nuevo estado lógico que es el tercer estado posible en la salida de una compuerta. Éste se denomina de **alta impedancia** y significa que la salida se **desconecta** o asume un **estado flotante**. En algunos casos puede ser útil que una salida conectada a otros componentes aparezca como si no lo estuviese, el procedimiento resulta análogo a cuando en la vida cotidiana se “desenchufa” un dispositivo eléctrico, separándolo de la red. El estado de alta impedancia en un circuito con lógica triestado (de tres estados), se puede controlar con una señal adicional denominada **entrada de habilitación**. La lógica triestado es muy útil para transferir datos entre varios registros relacionados por un “bus” único.

#### 5.6.2.1 Buffer triestado

La compuerta *buffer* triestado tiene una línea de entrada adicional **d** (desconectado), que opera como entrada de habilitación; con un 1 lógico habilita el estado de alta impedancia (fig. 5.15 y tabla 5-9).



Una compuerta de tres estados es un circuito digital que permite los estados 0 lógico, 1 lógico y, además, el estado de “alta impedancia”.



**Fig. 5.15.** Buffer triestado.

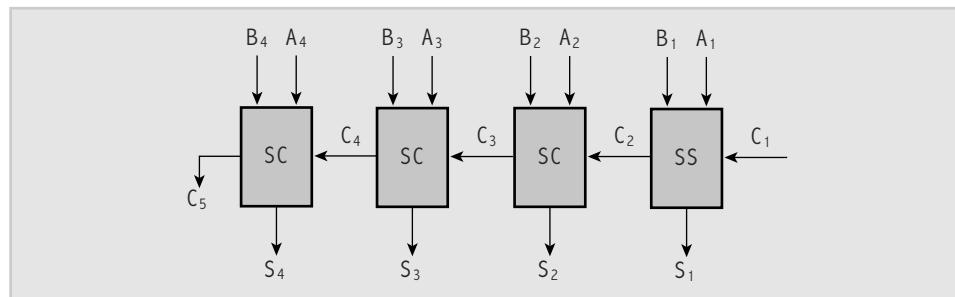
### 5.7 Circuito sumador-binario en paralelo

Ahora analizaremos cómo a partir de la expresión de una función booleana se puede confeccionar el circuito que lleve a cabo la operación que describe.

Supongamos que queremos sumar dos números de 8 dígitos cada uno:

$$\begin{array}{r}
 \begin{array}{c} Cy \\ A \\ + B \\ \hline S \end{array} \\
 \begin{array}{cccccccc} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ \hline 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}
 \end{array}$$

Se puede confeccionar un circuito sumador binario de 4 bits, cuya operación permita llevar a cabo la suma de los primeros cuatro bits de los operandos  $A$  y  $B$ . En la figura 5.16 el circuito se representa con cuatro módulos; el primero será un circuito semisumador (SS) y los restantes serán sumadores completos (SC).

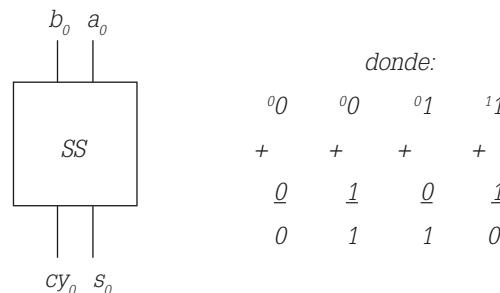


**Fig. 5.16.** Sumador binario paralelo con acarreo serie.

Los bits de ambos operandos constituyen la entradas del circuito sumador e ingresan al mismo tiempo (en paralelo); sin embargo, la suma de los dos primeros bits genera el acarreo después de cierto tiempo de operación del bloque semisumador, lo que produce un acarreo que afectará al bloque siguiente; esto ocurre así en todos los bloques. Las salidas del circuito son funciones que dependen de las entradas y representan los 8 bits del resultado y los sucesivos acarreos. Veamos cómo cumple su función cada bloque.

### 5.7.1 Circuito semisumador (SS) o Half Adder (HA)

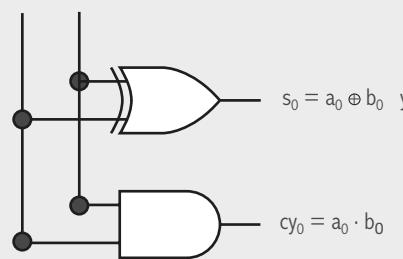
Es un circuito cuya función es sumar 2 bits sin tener en cuenta el acarreo anterior. Tiene dos salidas, una representa la suma y la otra, el valor del acarreo (*carry*).



Estas igualdades pertenecientes al resultado de la suma ( $s_0$ ) y al acarreo ( $cy_0$ ), las podemos expresar con una tabla de verdad (tabla 5-9):

<b>Tabla 5-9. Tabla para la suma de 2 bits.</b>			
$a_0$	$b_0$	$s_0$	$c_0$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Si se observan los valores representados en la tabla de verdad, se puede deducir que la función  $s_0$  responde a la compuerta XOR (suma aritmética), mientras que los valores de la función  $cy_0$  corresponden a una compuerta AND (fig. 5.17):

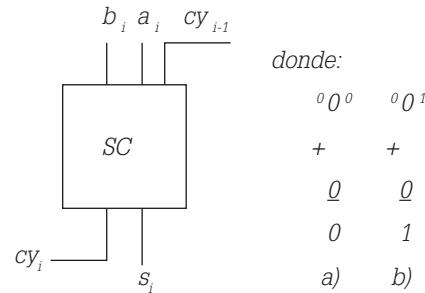


**Fig. 5-17.** Circuito lógico del semisumador.

### 5.7.2 Circuito sumador completo (SC) o Full Adder (FA)

Este circuito tiene como finalidad sumar 2 bits y el acarreo anterior, generando como salidas el resultado de la suma y el nuevo acarreo.

Genéricamente:



En el esquema anterior se observa un ejemplo de suma considerando acarreo anterior. En a) la suma de las variables con acarreo anterior en 0 da un resultado 0 y un nuevo acarreo 0; en b) la suma de las variables con acarreo anterior en 1 da un resultado 1 y un nuevo acarreo 0. En la tabla de verdad se deben representar las ocho combinaciones posibles de las tres variables de entrada al circuito, y las funciones suma ( $s_i$ ) y nuevo acarreo ( $cy_i$ ) con los valores que correspondan en cada caso.

Para no tener problemas de subíndices trabajaremos con:

$$b_i = b \text{ (variable que se ha de sumar)}$$

$$a_i = a \text{ (variable que se ha de sumar)}$$

$$cy_{i-1} = c \text{ (acarreo anterior)}$$

$$s_i = S \text{ (resultado de la suma)}$$

$$cy_i = C \text{ (nuevo acarreo)}$$

La tabla de verdad del circuito semisumador representa la función correspondiente a la suma  $s_i$  y al acarreo  $cy_i$ . Si observamos la tabla 5-10, notamos que al utilizar el mismo procedimiento para las funciones  $s_i$  y  $cy_i$  del circuito sumador completo se hace complejo determinar la estructura de compuertas del circuito. Para salvar esta dificultad recurriremos a la utilización de las formas normales o canónicas de una función.

**Tabla 5-10**

$a_i$	$b_i$	$cy_{i-1}$	$S_i$	$Cy_i$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

## 5.8 Formas normales o canónicas de una función

Una función se puede representar por una serie de términos canónicos

Se llama **término canónico** de una función lógica a “*todo producto o toda suma en los que aparecen todas las variables que componen una función, en su forma directa o inversa*”. O sea que hay **productos canónicos** y **sumas canónicas**.

Un **minitérmino** o producto canónico es: “*el producto de las variables en juego, o sus negaciones individuales que hacen que el producto valga 1 (uno)*”, o, expresado de otro modo, “*es la representación de una de las combinaciones de la tabla de verdad por medio del producto lógico*”.

En la tabla 5-11 se observa su representación.

**Tabla 5-11**

<i>a</i>	<i>b</i>	Minitérminos
0	0	$\bar{a} \cdot \bar{b}$
0	1	$\bar{a} \cdot b$
1	0	$a \cdot \bar{b}$
1	1	$a \cdot b$

En una función se pueden definir  $2^n$  minitérminos, siendo n la cantidad de variables en juego.

### 5.8.1 Forma normal disyuntiva

*“Si se expresa una función como la suma lógica de aquellos minitérminos que en su tabla de verdad tengan valor 1, se obtiene la expresión de su forma normal disyuntiva (FND)”.*

Un **maxitérmino** o suma canónica es: “la suma de las variables en juego, o sus negaciones individuales que hacen que la suma valga 0 (cero)”, o, expresado de otro modo, “es la representación de una de las combinaciones de la tabla de verdad por medio de la suma lógica, pero teniendo en cuenta que los 0 en la combinación se expresan con la variable correspondiente sin negar y los 1, con la variable correspondiente negada”.

En la tabla 5-12 se observa su representación.

**Tabla 5-12**

<i>a</i>	<i>b</i>	Maxitérminos
0	0	$a + b$
0	1	$a + \bar{b}$
1	0	$\bar{a} + b$
1	1	$\bar{a} + \bar{b}$

En una función se pueden definir  $2^n$  maxitérminos, siendo n la cantidad de variables en juego.

### 5.8.2 Forma normal conjuntiva

*“Si se expresa una función como el producto lógico de aquellos maxitérminos que en su tabla de verdad tengan valor 0, se obtiene la expresión de su forma normal conjuntiva (FNC)”.*

La FND y la FNC se denominan **formas normales o canónicas** de una función.

Ejemplos:

Sea un circuito semisumador, a partir de su tabla de verdad podremos obtener en primer lugar los minitérminos y los maxitérminos de la función y luego expresar la FND y la FNC de ésta.

Anteriormente dedujimos que:

$$S_0 = a_0 \oplus b_0 \quad y \quad C_0 = a_0 \cdot b_0$$

Si recuerda la tabla de verdad de la tabla 5-9, podrá hallar las formas canónicas de las funciones.

**Tabla 5-13**

<i>a<sub>0</sub></i>	<i>b<sub>0</sub></i>	<i>s<sub>0</sub></i>	<i>c<sub>0</sub></i>
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Para  $s_0$ :

$$FND(s_0) = (\bar{a}_0 \cdot b_0) + (a_0 \cdot \bar{b}_0)$$

$$FNC(s_0) = (a_0 + b_0) \cdot (\bar{a}_0 + \bar{b}_0)$$

y para  $cy_0$ :

$$FND(cy_0) = a_0 \cdot b_0$$

$$FNC(cy_0) = (a_0 + b_0) \cdot (a_0 + \bar{b}_0) \cdot (\bar{a}_0 + b_0)$$

En este momento podremos diseñar el gráfico del circuito a partir de las FND correspondientes a cada una de las funciones (fig. 5.18):

Luego, observando de la tabla de verdad 5-14 el circuito sumador-completo, podemos deducir las funciones de suma y acarreo correspondientes, ya sea por intermedio de la forma normal disyuntiva (FND) o por la forma normal conjuntiva (FNC). Elegiremos siempre la que más nos convenga, teniendo en cuenta la menor cantidad de términos que se han de representar, para favorecer, así, la implementación del circuito.

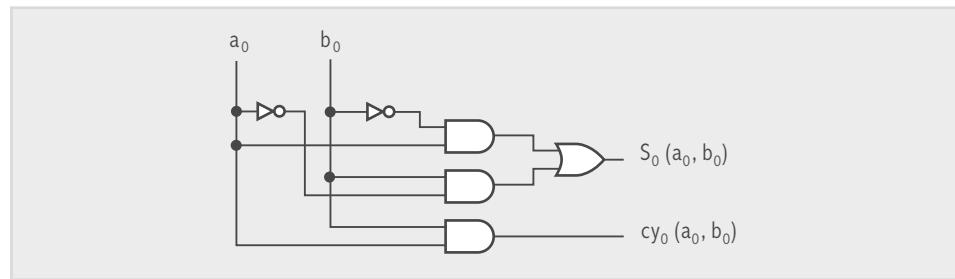


Fig. 5.18. Circuito lógico con las FND del semisumador.

Ahora hallamos las formas normales para el sumador completo.

Recordemos que:

$$b_i = b \text{ (variable que se ha de sumar)} \quad s_i = S \text{ (resultado de la suma)}$$

$$a_i = a \text{ (variable que se ha de sumar)} \quad cy_i = C \text{ (nuevo acarreo)}$$

$$cy_{i-1} = c \text{ (acarreo anterior)}$$

Tabla 5-14

a	b	c	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

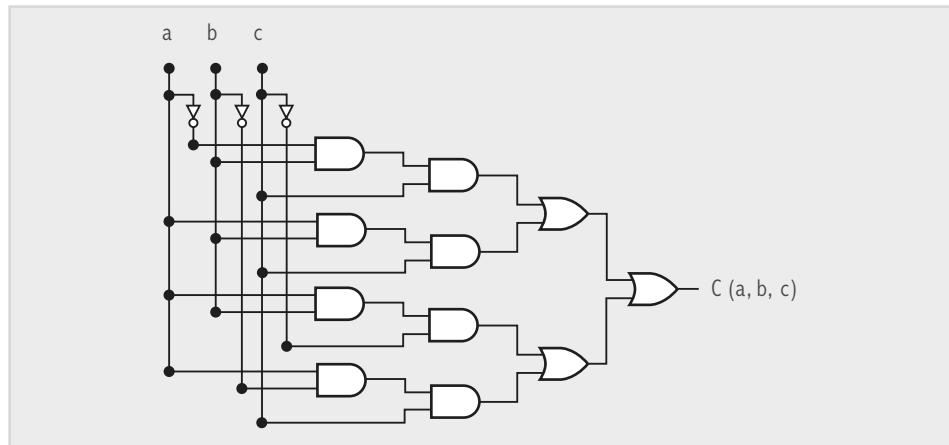
$$FND_{(S)} = (\bar{a} \cdot \bar{b} \cdot c) + (\bar{a} \cdot b \cdot \bar{c}) + (a \cdot \bar{b} \cdot \bar{c}) + (a \cdot b \cdot c)$$

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

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

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

Si graficamos el circuito con las formas normales disyuntivas de  $S$  y  $C$ , obtendremos el diagrama de lógica del circuito sumador-completo o *full adder* (fig. 5.19).



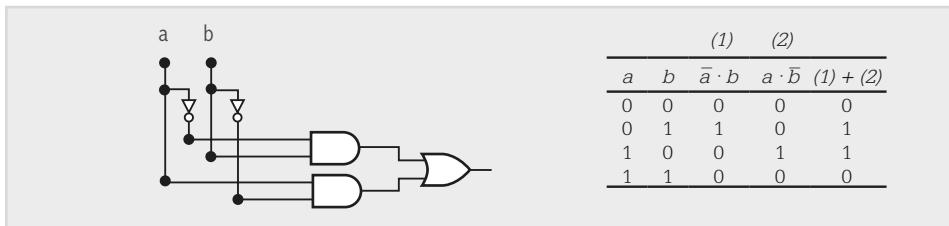
**Fig. 5.19.** Circuito lógico con las FND del sumador completo.

Estos diagramas están representados con compuertas de dos entradas. Recuérdese que el producto lógico (así como la suma lógica) es una operación asociativa, por lo tanto,  $(a \cdot b) \cdot c$  es igual a  $a \cdot (b \cdot c)$ , o sea que el último producto expresado entre paréntesis se puede representar con una compuerta AND de tres entradas.

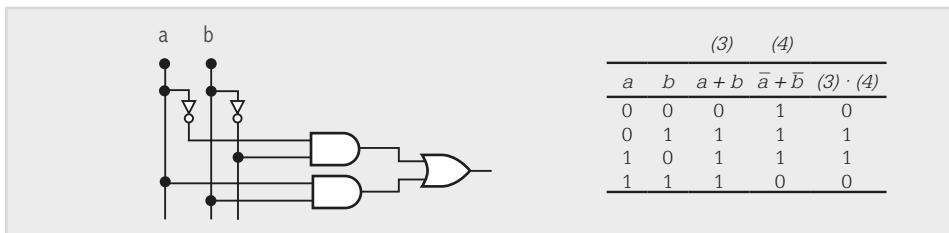
## 5.9 Circuitos equivalentes

Dos circuitos son equivalentes cuando son representados con distintas formas algebraicas pero responden a la misma tabla de verdad.

En las figuras 5.20 y 5.21 se observan ejemplos:



**Fig. 5.20.** Circuito A.



**Fig. 5.21.** Circuito B.



**Karnaugh:** Ingeniero de Telecomunicaciones estadounidense y director emérito del ICCC (*International Council for Computer Communication*). Hizo grandes aportes sobre la utilización de métodos numéricos en las Telecomunicaciones. Es el creador del método tabular o mapa de Karnaugh.

Los circuitos  $A$  y  $B$  representan las formas normales de la función  $a \oplus b$ . Como ambas son expresiones diferentes obtenidas de la misma tabla, generan circuitos equivalentes.

Al circuito  $A$  le corresponde la función FND =  $(\bar{a} \cdot b + a \cdot \bar{b})$ , y al circuito  $B$  le corresponde la función FNC =  $(a + b) \cdot (\bar{a} + \bar{b})$ .

## 5.10 Minimización de circuitos

La minimización de circuitos se realiza para que el diseño de un circuito sea lo más simple posible, que cumpla la misma función con la menor cantidad de compuertas posibles, la menor cantidad de entradas a ellas y en dos niveles. Para eso se debe minimizar la función. Uno de los métodos más utilizados para la representación de funciones y su posterior reducción es la minimización por medio de un mapa de "Veitch-Karnaugh". Éste consiste en representar la matriz que contempla todas las combinaciones posibles del juego de variables de las funciones. Por lo tanto, para 2 variables que implican 4 combinaciones, tendremos que utilizar un mapa o matriz de  $2 \times 2$  (4 casillas); para 3 variables, una matriz de  $4 \times 2$  (8 casillas) y para 4 variables, una matriz de  $4 \times 4$  (16 casillas). La minimización por este método para funciones de más de 4 variables de entrada se toma complicada.

Así, las líneas interceptadas por las columnas forman cubos que se completarán con un "1" o un "0" según la tabla de verdad (los 0 pueden no ponerse). Luego de este paso, se deberá agrupar la mayor cantidad de 1 adyacentes posibles, siempre y cuando esta cantidad sea una potencia exacta de 2. O sea que, en primer lugar debemos intentar hacer grupos de 8 "1" adyacentes, luego de 4 y después de 2, teniendo en cuenta que las adyacencias se producen con las aristas de los cubos y no con los vértices y que el borde superior de la matriz es adyacente con el inferior, de forma que si los unimos obtenemos un cilindro y, de idéntica manera, el borde derecho de la matriz se une con el izquierdo para posibilitar nuevas adyacencias. Agrupar la mayor cantidad de 1 posibles evita la redundancia de variables en la función final, o sea que permite hallar variables que no tienen sentido de estar porque no alterarían el funcionamiento del circuito.

Cada grupo queda definido, entonces, como la intersección de las adyacencias definidas (o sea, el producto de las zonas adyacentes). Luego, la función mínima será la unión (suma lógica) de todos los grupos definidos antes.

Ahora trataremos de minimizar con el método de Veitch-Karnaugh la función suma del semisumador, a partir de su tabla de verdad 5-13 (que repetimos abajo). Para ello confeccionaremos el mapa correspondiente.

**Tabla 5-13**

$a_0$	$b_0$	$S_0$	$C_0$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

**Función suma del semisumador.** Si observamos el mapa correspondiente a la función suma ( $S_0$ ), vemos que por tratarse de "1" aislados no es posible agruparlos, resultando la suma de minitérminos, o sea, la suma mínima ( $S_m$ ) (tabla 5-14).

**Tabla 5-15**

	$\bar{b}$	$b$
$\bar{a}$	0	1
$a$	1	1

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

### 5.10.1 Ejemplos de minimización a partir de distintos mapas de Karnaugh de 2, 3 y 4 variables

En este ejemplo los dos únicos "1" son adyacentes y se agrupan, ambos cubren toda la zona de  $\bar{b}$ ; esto significa que el valor que asume la variable  $a$  no tiene significado en el valor de la función  $y$ , por lo tanto, no se incluye en el circuito.

	$\bar{b}$	$b$
$\bar{a}$	0	1
$a$	1	1

Esta situación se puede demostrar fácilmente si se aplica a la expresión de la función una minimización algebraica:

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

$f = \bar{b} \cdot (\bar{a} + a)$  de acuerdo con las leyes de complemento

$$f_m = \bar{b} \cdot 1 = \bar{b}$$

	$\bar{b}$	$b$
$\bar{a}$	0	1
$a$	1	

En este ejemplo no hay "1" adyacentes, por lo tanto, no es posible la minimización. La función se expresa como:

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

	$c$	$\bar{c}$	$c$
$a$	$b$	0	1
0	0		
0	1		(1)
1	1		
1	0		

En este ejemplo hay un solo "1", por lo tanto, no es posible la minimización y la función se representa

$$f_m = \bar{a} \cdot b \cdot c$$

	$c$	$\bar{c}$	$c$
$a$	$b$	0	1
0	0	(1)	(1)
0	1		
1	1	(1)	(1)
1	0		

En este ejemplo se agrupan los 2 únicos pares de "1" adyacentes. El grupo superior indica que las variables  $a$  y  $b$  deben estar negadas para que la función valga 1, sin importar el valor que asume la variable  $c$ . El grupo inferior indica que la función vale 1 si  $a$  y  $b$  no están negadas, sin importar el valor de la variable  $c$ . La función se expresa como la suma de los productos que definen ambos grupos:  $f_m = \bar{a} \cdot \bar{b} + a \cdot b$

	$c$	$\bar{c}$	$c$
$a$	$b$	0	1
0	0	(1)	(1)
0	1	(1)	
1	1		
1	0		

En este ejemplo se deben agrupar los "1" adyacentes horizontales y luego los dos "1" adyacentes verticales. El "1" definido por  $a \cdot b \cdot \bar{c}$  se agrupa nuevamente, por lo que este procedimiento resulta óptimo para la minimización, dado que el grupo vertical permite la eliminación de la variable  $b$ .

La función mínima es  $f_m = \bar{a} \cdot \bar{c} + \bar{a} \cdot b$

	$c$	$\bar{c}$	$c$
$a$	$b$	0	1
0	0	(1)	(1)
0	1	(1)	(1)
1	1		
1	0		

En este caso se observan cuatro "1" adyacentes que se deben tomar formando un solo grupo. Esta agrupación indica que sólo tiene significado para el valor de la función la variable  $\bar{a}$ , eliminando las variables  $b$  y  $c$ . La función mínima resulta  $f_m = \bar{a}$

	$c$	$\bar{c}$	$c$
$a$	$b$	0	1
0	0	1	1
0	1		
1	1		
1	0	1	1

Esta matriz aparenta contener dos grupos de "1" adyacentes (esta vez identificados por la zona marcada). Sin embargo, esto no es así; si se observa con detenimiento, ambos grupos pertenecen a la zona  $\bar{b}$ , por lo tanto, los cuatro "1" deben considerarse adyacentes, aun cuando la limitación de la figura plana (los "1" de los extremos no se tocan) pueda confundir. De este modo la función queda definida como  $f_m = \bar{b}$ , eliminando las variables  $a$  y  $c$  de la función.

	$c$	$\bar{c}$	$c$
$a$	$b$	0	1
0	0	1	1
0	1		
1	1		(1)
1	0	(1)	1

Éste es un ejemplo aún más complejo; el primero es un grupo horizontal de dos "1" adyacentes. Luego el segundo grupo es vertical, constituido por dos "1" en apariencia no adyacentes, y, además, reagrupa un "1" tomado anteriormente. Por último, el "1" definido por  $a \cdot b \cdot c$  queda solo sin posibilidad de agruparse. La función queda expresada como:  $f_m = \bar{a} \cdot \bar{b} + \bar{b} \cdot \bar{c} + a \cdot b \cdot c$

## 5.11 Resumen

Podemos considerar de interés para la comprensión de la representación de datos, por lo menos dos formas de representación de cantidades: la digital, que es la que nos compete, y la analógica, mencionada en el capítulo Evolución del procesamiento de datos. En el caso de las representaciones analógicas, podemos mencionar que una cantidad se representa por su relación proporcional con otra magnitud, por ejemplo, el voltaje de una tensión o el movimiento a escala de una sustancia (el caso típico y el que primero abordamos cuando somos pequeños es el termómetro, que cuantifica la temperatura del cuerpo en relación con el movimiento del mercurio en el tubo de vidrio). Lo más importante para indicar de las representaciones analógicas es que varían en forma continua dentro de un rango predeterminado.

Al "marcar" el tubo de vidrio del termómetro con pequeñas líneas estamos haciendo que la información sea discreta, porque como la temperatura del cuerpo humano es un valor que fluctúa en forma "continua" y el mercurio es un elemento que también lo hace, preferimos saber si nuestra temperatura es mayor que  $37,5^\circ\text{C}$ , además, "cuantas líneas", para decidir si sólo nos damos un baño o es preciso tomar un medicamento y llamar al médico. Las mediciones " $38$ " o " $39$ " se denominan "digitales", si consideramos el término desde el punto de vista de los dígitos  $3, 8$  y  $9$ .

O sea que los sistemas como el decimal, el octal, el hexadecimal y el binario, explicados en el capítulo Sistemas numéricos se utilizan para representar datos "discretos". En un avión habrá sistemas analógicos para "medir" altitud, presión y temperatura, y sistemas digitales para realizar cálculos con ellos, que pueden contemplar un margen de error predecible cuando una cantidad analógica se procesa de manera digital.

De todos los sistemas, el que mejor se adapta a la operación de componentes electrónicos es el binario. Por esta razón, nuestro mundo informático tiene millones de bits. Éstos se representan en el interior de las computadoras con dos rangos de voltaje, uno para el  $0$  y otro para el  $1$ . Por ejemplo, se puede representar un "1" con un voltaje fluctuante en el entorno de  $3,5$  voltios y el "0" con un voltaje en el entorno de  $0$  voltios, o se puede representar un "1" con un voltaje fluctuante en el entorno  $4,5$  voltios y otro para el "0". En realidad, en el estudio de los sistemas digitales no interesa el valor de voltaje que representan los niveles lógicos "1" o "0", sino que lo que importa es si se interpreta como "0" o como "1". Es como marcar en el termó-

metro tres zonas por debajo de  $3,4^{\circ}$  ("no hay fiebre") y por encima de  $3,5^{\circ}$  ("hay fiebre"); la zona continua entre  $3,4^{\circ}$  y  $3,5^{\circ}$  es de transición y no se utiliza para la determinación de "no hay" y "sí hay", que por analogía serían el 0 o el 1, respectivamente.

En un circuito digital o circuito lógico, así lo llamaremos en adelante, las entradas se interpretan como niveles lógicos, esto es, como 0 y 1. Asimismo, las salidas también son niveles lógicos "0" o "1", o sea que "entran" bits y "salen" bits. La relación que hay entre las entradas y las salidas constituye la "lógica" del circuito, de ahí los nombres de lógica digital, diagrama de lógica, lógica de circuitos y tantos otros. La lógica digital halla una forma de representación matemática en el Álgebra de Boole, donde las variables booleanas son las "entradas" del circuito y las funciones booleanas sus "salidas"; estas últimas no son más que "variables booleanas que dependen del valor de otras variables booleanas". Las formas normales de una función permiten la expresión algebraica que "manifiesta" la forma en que las variables se relacionan entre sí, utilizando los operadores booleanos, de manera que se pueda "dibujar el comportamiento del circuito" con compuertas lógicas y armar así el "diagrama de lógica" o "circuito lógico".

Por último, el método de mapas de Karnaugh permite simplificar las funciones utilizando un método visual, que es equivalente a la simplificación que se puede lograr al aplicar los teoremas y los postulados del Álgebra de Boole a la expresión algebraica. Esta simplificación permite reducir el número de compuertas lógicas y, eventualmente, la cantidad de entradas a ellas o al circuito. Esto se traduce en un menor número de componentes electrónicos, que "mejoran" la eficacia de su comportamiento al minimizar su complejidad.

## 5.12 Ejercicios propuestos

1) Represente la tabla de verdad de la siguiente función:

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

2) Represente el diagrama lógico de la función f del enunciado anterior.

3) Dada la función  $f = a \cdot b + \bar{a} \cdot \bar{b}$ :

a) Representar el diagrama de lógica con compuertas AND, OR y NOT.

b) Representar el diagrama de lógica sólo con compuertas OR y NOT.

c) Representar el diagrama de lógica sólo con compuertas AND y NOT.

4) Dada la siguiente tabla de verdad represente la forma normal más conveniente para cada función:

$a$	$b$	$c$	$f_1$	$f_2$
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

5) Considerando  $n = 3$  verifique que la suma de los minitérminos de una función de Boole para  $n$  variables es = 0.

6) Considerando  $n = 3$  verifique que el producto de los máxiterminos de una función de Boole para  $n$  variables es = 1.

7) Infiera un procedimiento que generalice los enunciados de los dos últimos ejercicios.

8) Dada la tabla de verdad de las funciones  $f_1$  y  $f_2$ :

a) Represente la forma normal disyuntiva y la forma normal conjuntiva para cada una de ellas.

b) Represente los cuatro diagramas de lógica con compuertas NAND o NOR, según corresponda.

$a$	$b$	$c$	$f_1$	$f_2$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



### 5.13 Contenido de la página Web de apoyo

El material marcado con asterisco (\*) sólo está disponible para docentes.

#### Resumen gráfico del capítulo

#### Simulación

Herramienta interactiva que permite crear el diagrama lógico de una expresión booleana.

#### Animación

Cómo trabajan los interruptores no mecánicos.

#### Autoevaluación

#### Lecturas adicionales:

Álgebra booleana de José A. Jiménez Murillo, es parte del libro “Matemáticas para la Computación” de Alfaomega Grupo Editor (42 páginas). Agradecemos a su autor por permitir que su escrito sea parte de las lecturas complementarias de esta obra.

**Video explicativo (01:44 minutos aprox.)**

**Audio explicativo (01:44 minutos aprox.)**

**Evaluaciones Propuestas\***

**Presentaciones\***