

CAPÍTULO

V

Álgebra booleana

Álgebra de Boole es un capítulo de *Matemáticas para la Computación* de José Jiménez Murillo, quien amablemente nos autorizó a incluirlo como lectura complementaria de *Arquitectura de Computadoras* de Patricia Quiroga.

C	D	A'	A	B	D'	AC	D	AC'D'	BC	AC'D' + BC	A + C	B
0	0	1	1	1	1	0	0	0	0	0	0	1
0	1	1	1	1	0	0	0	0	0	0	0	1
1	0	1	1	0	1	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0	1	0
0	0	1	0	1	1	0	0	0	0	0	0	1
0	1	1	0	1	0	0	0	0	0	0	0	1
1	0	1	0	1	0	0	0	0	1	1	1	0
1	1	1	0	0	0	0	0	0	1	1	1	0
0	0	0	1	1	1	1	1	1	0	0	1	1
0	1	0	1	1	0	1	0	0	0	0	1	1
1	0	0	1	0	1	0	0	0	0	0	1	0
1	1	0	1	0	0	0	0	0	0	0	1	0
0	0	0	0	1	1	1	1	1	0	1	1	1
0	1	0	0	1	0	1	0	0	0	0	1	1
1	0	0	0	0	1	0	0	0	1	1	1	0
1	1	0	0	0	0	0	0	0	1	1	1	0

- 5.1** Introducción
- 5.2** Expresiones booleanas
- 5.3** Propiedades de las expresiones booleanas
- 5.4** Optimización de expresiones booleanas
- 5.5** Compuertas lógicas
- 5.6** Aplicaciones del álgebra booleana
- 5.7** Resumen
- 5.8** Problemas

C'	C+D'	F	A	B	C	D	A'	B'	C'	D'	AC'	AC'D'	BC	A
	1	0	0	0	0	0	1	1	1	1	0	0	0	
	0	0	0	0	0	0	1	1	1	1	0	0	0	
	1	0	0	0	1	0	1	1	0	1	0	0	0	
	1	0	0	0	0	0	1	1	1	1	0	0	0	
	1	0	0	1	0	0	1	0	1	1	0	0	0	
	0	0	0	1	0	1	1	0	1	0	0	0	0	
	1	1	0	1	1	0	1	0	0	1	0	0	1	
	1	1	0	1	1	1	1	0	0	0	0	0	1	
AC'	1	1	1	0	0	0	0	1	1	1	1	1	0	
	0	0	1	0	0	1	0	1	1	0	1	0	0	
	1	0	1	0	1	0	0	1	0	1	0	0	0	
	1	0	1	0	1	1	0	1	0	0	0	0	0	
(A'D)'	1	1	1	0	0	0	0	1	1	1	1	1	0	
	0	0	1	1	0	1	0	0	1	0	1	0	0	
	1	1	1	1	1	0	0	0	0	1	0	0	1	
	1	1	1	1	1	1	0	0	0	0	0	0	1	

Boole interpretó su sistema a la manera aristotélica, como un álgebra de clases y de sus propiedades, y al hacerlo amplió la antigua lógica de clases y la liberó de los límites del silogismo.

Martin Gardner

Objetivos

- Aprender a simplificar expresiones booleanas usando teoremas del álgebra booleana.
- Aprender a simplificar expresiones booleanas por medio de mapas de Karnaugh.
- Representar expresiones booleanas por medio de bloques lógicos.

5.1 Introducción

George Boole

(1815-1864)

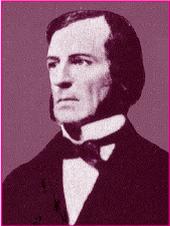
Fue un matemático británico que es considerado como uno de los fundadores de las ciencias de la computación debido a su creación del álgebra booleana, la cual es la base de la aritmética computacional moderna.

Con una formación autodidacta, Boole fue profesor a la edad de 16 años y a partir de 1835 comenzó a aprender matemáticas por sí mismo. En este periodo estudió los trabajos de Laplace y Lagrange, comenzó a estudiar álgebra y en el *Transaction of the Royal Society* publicó *Aplicación de métodos algebraicos para la solución de ecuaciones diferenciales*, trabajo por el cual recibió la medalla de la Real Sociedad.

En 1849 Boole ocupó una cátedra de matemáticas en el Queens College, y permaneció en este puesto por el resto de su vida. En 1854 publicó *Las leyes del pensamiento*, obra en la que plantea la lógica en términos de un álgebra simple que se conoce como álgebra booleana y que se aplica en la ciencia de la computación y en el análisis de circuitos.

Otras áreas de interés de Boole fueron las ecuaciones diferenciales en relación con las cuales escribió *Tratado en ecuaciones diferenciales*

que publicó en 1859, el cálculo de las diferencias finitas que expuso en *Tratado sobre el cálculo de las diferencias finitas* publicado en 1860, y los métodos generales en probabilidad.



El álgebra booleana fue desarrollada por George Boole y en su libro *An Investigation of the Laws of Thought*, publicado en 1854, muestra las herramientas para que las proposiciones lógicas sean manipuladas en forma algebraica. Debido al carácter abstracto de sus principios no tuvo una aplicación directa sino hasta 1938 en que la compañía de teléfonos Bell de Estados Unidos la utilizó para realizar un análisis de los circuitos de su red telefónica. En ese mismo año Claude E. Shannon, entonces estudiante de postgrado del Instituto Tecnológico de Massachussets, a partir del álgebra de Boole creó la llamada álgebra de conmutación para representar las propiedades de conmutación eléctrica biestables, demostrando con esto que el álgebra booleana se adapta perfectamente al diseño y representación de circuitos lógicos de control basados en relés e interruptores.

Los circuitos lógicos de control tienen una gran importancia ya que las computadoras, los sistemas telefónicos, los robots y cualquier operación automatizada en una empresa, son algunos de los ejemplos de la aplicación de éstos y del álgebra booleana.

Una señal es la representación de información, y puede aparecer en forma de valor o de una cadena de valores de una magnitud física. Existen principalmente dos clases de señales: analógicas y digitales.

La señal analógica tiene como característica principal el continuo cambio de magnitud, de la misma manera que una corriente eléctrica y una presión de gas.

En la señal digital los posibles valores de tensión están divididos en un número infinito de intervalos, a cada uno de los cuales está asignado un valor o una cadena de valores como información. Una señal digital puede obtenerse de una manera analógica asignando ciertos umbrales de sensibilidad.

La señal binaria es una señal digital con sólo dos valores posibles: conectado-desconectado, verdadero-falso, 1-0.

5.2 Expresiones booleanas

El álgebra booleana trabaja con señales binarias. Al mismo tiempo una gran cantidad de sistemas de control, también conocidos como digitales, usan señales binarias y éstas son un falso o un verdadero que proviene de sensores que mandan la información al circuito de control, mismo que lleva a cabo la evaluación para obtener un valor que indicará si se lleva a cabo o no una determinada actividad, como encender un foco, arrancar un equipo de ventilación en un cine o ejecutar una operación matemática en una computadora.

Los sensores pueden ser “ópticos”, como los que se usan en tiendas departamentales (de proximidad); “magnéticos”, como los que permiten detectar armas en aeropuertos; de “temperatura”, como los que utiliza un sistema de calefacción, los refrigeradores o bien el mismo termostato que controla el sistema de enfriamiento del motor de un vehículo; de “nivel”, ya que un flotador como el que tiene un tinaco o una cisterna para controlar la cantidad de agua, es un sensor que puede mandar información a un circuito de control.

En cada uno de estos grupos de sensores existen tipos, tamaños y modelos, de acuerdo con el uso y funcionamiento, de forma que existen infrarrojos, láser, fotoeléctricos y de ultrasonido, entre otros.

Para resolver un problema práctico en el cual se desea automatizar un proceso, es necesario realizar un análisis detallado de lo que se quiere lograr así como de los tipos de sensores necesarios para obtener las señales. Una vez que se conoce esto se plantea el funcionamiento del circuito lógico en una expresión matemática, la cual recibe el nombre de función booleana, y cada una de las variables de que está integrada esta función representa un sensor que provee al circuito de una señal de entrada.

Ejemplo 5.1. Supóngase que en una industria refresquera se desea que un sistema automático saque de la banda de transportación un refresco que no cumple con los requisitos mínimos de calidad, y que para esto se cuenta con cuatro sensores en diferentes puntos del sistema de transportación para revisar aspectos importantes de calidad. Supóngase además que los sensores son A, B, C y D y que el sistema F sacará al refresco si los sensores emiten el siguiente grupo de señales:

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



Claude Elwood Shannon

(1916-2001)

Ingeniero eléctrico y matemático estadounidense, es considerado como el fundador de la teoría de la información.

En 1936 obtuvo los títulos de ingeniero electricista y matemático, y ese mismo año comenzó a desempeñarse como asistente de investigación en el departamento de ingeniería eléctrica en el Instituto de Tecnología de Massachusetts (MIT), en donde trabajó en el computador analógico más avanzado de ese tiempo (Vannevar Bush's Differential Analyzer).

En esta época surgió su interés por los circuitos de relevadores complejos e intentando simplificar sistemas telefónicos de relés se dio cuenta de que éstos podían usarse para hacer cálculos. Combinando esto con su gusto por la lógica y el álgebra booleana pudo desarrollar esta idea durante el verano de 1937, que pasó en los laboratorios Bell en la ciudad de Nueva York.

En su tesis de maestría demostró que el álgebra booleana se podía utilizar en el análisis y la síntesis de la conmutación de los circuitos digitales. La tesis despertó mucho interés cuando apareció en 1938 en las publicaciones especializadas, y un cuarto de siglo más tarde H. H. Goldstine la citó en su libro “Las computadoras desde Pascal hasta Von Neumann” y la calificó como una de las aportaciones teóricas fundamentales que ayudó a cambiar el diseño de los circuitos digitales.

Shannon pasó quince años en los laboratorios Bell y durante este periodo trabajó en muchas áreas, siendo lo más notable todo lo referente a la teoría de la información, un desarrollo que fue publicado en 1948 bajo el nombre de “Una Teoría Matemática de la Comunicación”. En este trabajo demostró que todas las fuentes de información (telégrafo eléctrico, teléfono, radio, la gente que habla, las cámaras de televisión, etc.) se pueden medir y que los canales de comunicación tienen una unidad de medida similar. Mostró también que la información se puede transmitir sobre un canal si, y solamente si, la magnitud de la fuente no excede la capacidad de transmisión del canal que la conduce, y sentó las bases para la corrección de errores, supresión de ruidos y redundancia.

En el área de las computadoras y de la inteligencia artificial, en 1950 publicó un trabajo que describía la programación de una computadora para jugar al ajedrez, convirtiéndose en la base de posteriores desarrollos.

Claude Elwood Shannon falleció el 24 de febrero del año 2001, a la edad de 84 años, después de una larga lucha en contra de la enfermedad de Alzheimer.

Álgebra booleana

El álgebra booleana es un sistema algebraico que consiste en un conjunto B que contiene dos o más elementos y en el que están definidas dos operaciones, denominadas respectivamente “suma u operación OR” (+) y “producto u operación AND” (\cdot), las cuales satisfacen las siguientes propiedades:

1) Existencia de neutros. En B existen el elemento neutro de la suma (0) y el elemento neutro del producto (1), tales que para cualquier elemento x de B :

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

2) Conmutatividad. Para cada x, y en B :

$$x + y = y + x \quad x \cdot y = y \cdot x$$

3) Asociatividad. Para cada x, y, z en B :

$$x + (y + z) = (x + y) + z \\ x \cdot (y \cdot z) = (x \cdot y) \cdot z$$

4) Distributividad. Para cada x, y, z en B :

$$x + (y \cdot z) = (x + y) \cdot (x + z) \\ x \cdot (y + z) = (x \cdot y) + (x \cdot z)$$

5) Existencia de complementos. Para cada x en B existe un elemento x' , llamado complemento de x , tal que:

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

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

La función booleana que equivale a la tabla de verdad anterior es:

$$F = A'B'C'D + A'B'CD + AB'C'D + AB'CD + AB'CD'$$

Esto implica que el refresco será extraído de la banda de transportación en cualquiera de los siguientes casos, ya que para cualquiera de ellos se tiene que $F = 1$:

$$A = 0, \quad B = 0, \quad C = 0, \quad D = 1$$

$$A = 0, \quad B = 0, \quad C = 1, \quad D = 1$$

$$A = 1, \quad B = 0, \quad C = 0, \quad D = 1$$

$$A = 1, \quad B = 0, \quad C = 1, \quad D = 1$$

$$A = 1, \quad B = 0, \quad C = 1, \quad D = 0$$

La función booleana indica solamente los casos en donde el refresco será extraído, pero existen varios casos más en donde se dejará pasar porque cumple con los requisitos mínimos de calidad.

Se puede decir que en general una expresión booleana es un sistema de símbolos que incluyen 0, 1, algunas variables y las operaciones lógicas.

5.3 Propiedades de las expresiones booleanas

Las expresiones booleanas poseen las siguientes propiedades:

- Están compuestas de literales (A, B, C, \dots) y cada una de ellas representa la señal de un sensor. Un ejemplo es $F = A'BD + AB'CD$.
- El valor de las señales o de la función sólo puede ser 0 o 1, falso o verdadero.
- Además de literales, en la expresión booleana se puede tener el valor de 0 o 1. Por ejemplo: $F = A'BD1 + AB'CD + 0$.

d) Las literales de las expresiones booleanas pueden estar conectadas por medio de los operadores lógicos And (\wedge), Or (\vee) y Not ($'$). El operador And es una multiplicación lógica que se indica por medio de un paréntesis, un punto o simplemente poniendo juntas las variables que se multiplican, por ejemplo el producto de A y B se expresa como $(A)(B) = A \cdot B = AB$; el Or es una suma lógica que se indica con el signo +; y el operador Not es el complemento o negación de una señal que se indica por un apóstrofo ($'$). En la siguiente expresión se muestra la forma en que se representan los operadores:

$$F = A'BD1 + AB'CD + 0$$

$$= A' \wedge B \wedge D \wedge 1 \vee A \wedge B' \wedge C \wedge D \vee 0$$

e) Es posible obtener el valor de una expresión booleana sustituyendo en cada una de las literales el valor de 0 o 1, teniendo en cuenta el comportamiento de los operadores lógicos. En las siguientes tablas se muestra la manera en la que se aplica esta propiedad:

<i>And</i>		
A	B	$A \wedge B = AB$
1	1	1
1	0	0
0	1	0
0	0	0

<i>Or</i>		
A	B	$(A \vee B) = A + B$
1	1	1
1	0	1
0	1	1
0	0	0

<i>Not</i>	
A	A'
1	0
0	1

Hay que tener presente que en álgebra booleana:

$$1 + 1 = 1$$

$$1 + 1 + 1 = 1$$

$$0 + 1 = 1$$

$$0 + 0 = 0$$

ya que el valor máximo es 1.

f) Además de las operaciones básicas, también es posible aplicar la ley de De Morgan de forma semejante a como se aplica en teoría de conjuntos. El siguiente ejemplo muestra la aplicación de esta propiedad:

$$(ABCD)' = A' + B' + C' + D'$$

$$(A + B + C + D)' = A' B' C' D'$$

Teoremas del álgebra booleana

A partir de las propiedades de las operaciones del álgebra booleana se pueden demostrar los siguientes teoremas.

1) Teorema 1. Idempotencia.

$$x + x = x \quad x \cdot x = x$$

2) Teorema 2. Identidad de los elementos 0 y 1.

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

3) Teorema 3. Absorción.

$$x + (x \cdot y) = x$$

$$x \cdot (x + y) = x$$

4) Teorema 4. Complemento de 0 y 1.

$$0' = 1 \quad 1' = 0$$

5) Teorema 5. Involución.

$$(x')' = x$$

6) Teorema 6. Leyes de Morgan.

$$(x + y)' = x' \cdot y' \quad (x \cdot y)' = x' + y'$$

5.4 Optimización de expresiones booleanas

Cuando se plantea un problema, en general la expresión booleana obtenida no necesariamente es la óptima, esto es, la más fácil, clara y sencilla de implementar utilizando compuertas lógicas. La expresión que resulta del planteamiento del problema puede ser simplificada empleando para ello teoremas y postulados del álgebra booleana o bien mapas de Karnaugh.

5.4.1 Simplificación de expresiones booleanas mediante teoremas del álgebra de Boole

Los teoremas que se van a utilizar se derivan de los postulados del álgebra booleana, y permiten simplificar las expresiones lógicas o transformarlas en otras que son equivalentes. Una expresión simplificada se puede implementar con menos equipo y su circuito es más claro que el que corresponde a la expresión no simplificada.

A continuación se presenta una lista de teoremas, cada uno con su "dual".

Tabla 5.1 Teoremas del álgebra de Boole

Número	Teorema	Dual
1a.	$0A = 0$	$1 + A = 1$
2a.	$1A = A$	$0 + A = A$
3a.	$AA = A$	$A + A = A$
4a.	$AA' = 0$	$A + A' = 1$
5a.	$AB = BA$	$A + B = B + A$
6a.	$ABC = A(BC)$	$A + B + C = A + (B + C)$
7a.	$(AB\dots Z)' = A' + B' + \dots + Z'$	$(A + B + \dots + Z)' = A'B'\dots Z'$
8a.	$AB + AC = A(B + C)$	$(A + B)(A + C) = A + BC$
9a.	$AB + AB' = A$	$(A + B)(A + B') = A$
10a.	$A + AB = A$	$A(A + B) = A$
11a.	$A + A'B = A + B$	$A(A' + B) = AB$
12a.	$CA + CA'B = CA + CB$	$(C + A)(C + A' + B) = (C + A)(C + B)$
13a.	$AB + A'C + BC = AB + A'C$	$(A + B)(A' + C)(B + C) = (A + B)(A' + C)$

En esta tabla A representa no sólo una variable, sino también un término o factor, o bien una expresión.

Para obtener el "dual" de un teorema se convierte cada 0 (cero) en 1 (uno) y cada 1 (uno) en 0 (cero), los signos más (+) se convierten en paréntesis,

puntos o simplemente no se ponen, y los puntos en signos más (+). Además de esto, las variables no se complementan ya que al hacerlo se obtendría el complemento en lugar del dual.

Por otro lado, los teoremas 1 a 4 se aplican en cualquier caso y los teoremas 5 a 9 son propiedades que tiene el álgebra booleana, semejantes a las reglas de conjuntos correspondientes a las propiedades conmutativa, asociativa y de De Morgan. Por lo general los teoremas 11 a 13 se aplican en combinación, dependiendo de la expresión booleana.

La aplicación de los teoremas es muy sencilla: simplemente se comparan partes de la expresión con los teoremas que permitan hacer más simple la expresión, y esto se realiza hasta que ya no sea posible simplificar.

Ejemplo 5.2. Para simplificar la expresión booleana

$$F = A'B + (ABC)' + C(B' + A)$$

los teoremas de la tabla 5.1 se aplican de la siguiente manera:

$F = A'B + (ABC)' + C(B' + A)$	
$F = A'B + A' + B' + C' + C(B' + A)$	Después de aplicar 7a.
$F = A'B + A' + B' + C' + CB' + CA$	Por 8a a la inversa
$F = A'B + A' + B' + CB' + C' + CA$	Por 5a.
$F = A'(B + 1) + B'(1 + C) + C' + CA$	Por 8a.
$F = A'1 + B'1 + C' + CA$	Por 1b.
$F = A' + B' + C' + CA$	Por 2a.
$F = A' + B' + C' + A$	Por 11a.
$F = (A + A') + B' + C'$	Por 5a.
$F = (1 + B') + C'$	Por 4b.
$F = 1 + C'$	Por 1b.
$F = 1$	Por 1b.

La expresión booleana en su forma más simple es $F = 1$, y este resultado indica que si se sustituyen las diferentes combinaciones con los valores binarios 0 o 1 de las variables A, B y C en la expresión inicial, entonces el resultado será siempre igual a 1 (lo que se conoce en lógica matemática como tautología).

En general luego de un proceso de simplificación el resultado no siempre es 1, en cambio lo que se espera es obtener una expresión más simple conformada por menos variables.

Ejemplo 5.3. La simplificación de la expresión booleana

$$F = Z'X + XY'Z + X'Z'W$$

es la siguiente:

$$F = Z'X + XY'Z + X'Z'W$$

$$F = Z'(X + X'W) + XY'Z$$

Por 8a

$$F = Z'(X + W) + XY'Z$$

Por 11a

$$F = Z'X + Z'W + XY'Z$$

Por 8a, a la inversa

$$F = X(ZY' + Z') + Z'W$$

Por 8a

$$F = X(Z' + Y') + Z'W$$

Por 11a

$$F = XZ' + XY' + Z'W$$

Por 8a, a la inversa

En los ejemplos anteriores se utilizó un teorema a la vez, y esto se hizo para que no haya confusión en la aplicación de los mismos. Obviamente que cuando ya se tiene suficiente práctica, se pueden aplicar varios teoremas a la vez. Tampoco es necesario indicar qué teorema se usa, sin embargo aquí se hace para ilustrar la simplificación.

Comprensiblemente las expresiones booleanas a simplificar son el resultado del planteamiento de un problema que se busca resolver, tal y como se ilustró al inicio del capítulo con la función booleana

$$F = A'B'C'D + A'B'CD + AB'C'D + AB'CD + AB'CD'$$

Comúnmente este tipo de expresiones booleanas son factibles de ser simplificadas, como se muestra a continuación:

$$F = A'B'C'D + A'B'CD + AB'C'D + AB'CD + AB'CD'$$

$$F = A'B'D(C' + C) + AB'D(C' + C) + AB'CD'$$

$$F = A'B'D + AB'D + AB'CD'$$

$$F = B'D(A' + A) + AB'CD'$$

$$F = B'D + AB'CD'$$

$$F = B'(D + D'AC)$$

$$F = B'(D + AC)$$

$$F = B'D + AB'C$$

Es conveniente mencionar que con las funciones booleanas se pueden elaborar circuitos equivalentes tanto con la función booleana simplificada como con la que se obtuvo inicialmente, sin embargo el circuito lógico de la función booleana sin simplificar será más grande, complejo y usará más equipo electrónico en su implementación.

5.4.2 Simplificación de expresiones booleanas usando mapas de Karnaugh

El método del mapa de Karnaugh es un procedimiento simple y directo para minimizar las expresiones booleanas, y fue propuesto por Edward W. Veitch y modificado ligeramente por Maurice Karnaugh.

El mapa representa un diagrama visual de todas las formas posibles en que se puede plantear una expresión booleana en forma normalizada. Al reconocer varios patrones se pueden obtener expresiones algebraicas alternas para la misma expresión, y de éstas se puede escoger la más simple, la cual en general es la que tiene el menor número de variables además de que esta expresión posiblemente no sea única.

Las tablas o mapas se dividen en cierto número de casillas, dependiendo de la cantidad de variables que intervengan en la expresión. El número de casillas se puede calcular con la fórmula

$$\text{número de casillas} = 2^n$$

en donde n es el número de variables. Así a una expresión de 2 variables le corresponderá un mapa de 4 casillas, a una de 3 variables un mapa de 8 casillas y así sucesivamente.

Un minitérmino es aquel que forma parte de la expresión y que se puede escribir de la manera más simple formando lo que se conoce en álgebra elemental como un monomio.

Por ejemplo, la expresión

$$F = X'Y + XY$$

consta de dos minitérminos, $X'Y$ y XY , y como se muestra a continuación en las casillas respectivas de la tabla correspondiente se pone un 1 si el minitérmino se encuentra en la expresión o un 0 si no está:



Maurice Karnaugh

Fue Ingeniero de telecomunicaciones estadounidense graduado en la universidad de Yale en 1952 y director emérito del ICCO (International Council for Computer Communication). Trabajó como investigador en los Laboratorios Bell desde 1952 a 1966 y en el centro de investigación de IBM de 1966 a 1993, desde 1975 es miembro del IEEE (Institute of Electrical and Electronics Engineers) por sus aportaciones sobre la utilización de métodos numéricos en las telecomunicaciones y es el creador del método tabular o mapa de Karnaugh.

Un mapa de Karnaugh (también conocido como tabla de Karnaugh o diagrama de Veitch, abreviado como K-Mapa o KV-Mapa) es un diagrama utilizado para la minimización de funciones algebraicas booleanas y consiste en una serie de cuadrados cada uno de los cuales representa una línea de la tabla de verdad. Puesto que la tabla de verdad de una función de N variables posee $2N$ filas, el mapa K correspondiente debe poseer también $2N$ cuadrados. Cada cuadrado contiene un 0 o un 1, dependiendo del valor que toma la función en cada fila. Las tablas de Karnaugh se pueden utilizar para funciones de hasta 6 variables.

	Y	
X	0	1
0	0	1
1	0	1

Para simplificar la expresión, en la tabla se agrupan los 1 de casillas adyacentes en bloques cuadrados o rectangulares de 2, 4, 8, 16, ..., 2^n y se descartan las variables cuyo valor, 1 o 0, cambia de una casilla a otra. La regla es agrupar la información con el menor número posible de bloques ya que de cada uno de éstos se obtiene cuando menos una literal, y los bloques deben estar conformados por el mayor número de casillas porque entre más grande sea el número de casillas agrupadas por bloque, más simple será la expresión booleana resultante.

En el mapa anterior la variable X no conserva su valor ya que en la primera línea vale 0 y en la segunda 1, por lo tanto se elimina. Sin embargo, Y mantiene el valor de 1 en ambas casillas, ya que en este caso el bloque que agrupa la información se encuentra solamente en la columna de la derecha. De esta forma se obtiene que la expresión simplificada del mapa de Karnaugh es $F = Y$.

Como se ve, la simplificación anterior consiste en la aplicación de los postulados del álgebra booleana, pero de manera gráfica.

Para simplificar una expresión que incluye tres variables se tiene que el mapa consta de 8 casillas. Hay que observar que la secuencia en que se coloca la expresión en la tabla no es la binaria ascendente, sino una de forma que solamente exista un cambio de 0 a 1 o de 1 a 0 a la vez, esto es, una en la que no debe cambiar más que un bit en cada paso. A esta forma de arreglar los bits se le llama código reflejado.

Ejemplo 5.4. Representar en un mapa de Karnaugh y determinar la expresión booleana simplificada de:

$$F = XY'Z' + XY'Z + XYZ' + X'YZ'$$

La solución es la siguiente:

	YZ			
X	00	01	11	10
0				1
1	1	1		1

En este caso se forman dos bloques, mismos que permiten eliminar una variable en cada uno de ellos de forma que la expresión simplificada es:

$$F = XY' + YZ'$$

En general se tiene que cuando el número de variables que integran la expresión booleana es impar, el número de filas del mapa es menor que el número de columnas. También es conveniente ordenar las variables alfabéticamente colocando las primeras variables como filas y las restantes como columnas.

Ejemplo 5.5. Como se muestra en el siguiente mapa, un 1 de una celda puede estar contenido en más de un bloque.

	YZ			
X	00	01	11	10
0		1	1	
1		1	1	1

En el caso de esta tabla se tiene que la expresión booleana sin simplificar es:

$$F = X'Y'Z + X'YZ + XY'Z + XYZ + XYZ'$$

la cual ya simplificada queda como:

$$F = Z + XY$$

En el ejemplo anterior se formaron dos bloques, y en el mayor se eliminaron las variables X, Y debido a que de una casilla a otra cambian de valor. Además se observa que entre más grande sea el bloque, la expresión resultante es menor.

Si en un mapa de Karnaugh se unen los dos extremos, ya sea horizontal o verticalmente, entonces las celdas de las esquinas del mismo quedarán juntas y por lo tanto se considerarán como celdas adyacentes. Esto permite realizar una mejor simplificación.

Ejemplo 5.6. Simplificar la siguiente expresión booleana:

$$F = W'X' + W'XY'Z + W'XYZ + WXY'Z' + WX'Y'Z' + WX'YZ'$$

Como se ve, no siempre la expresión original tiene todas las variables en cada uno de sus minitérminos. En donde es así, el minitérmino equivale a las variables que se dan inicialmente, en este caso $W'X'$ juntamente con todas las posibles combinaciones de las variables faltantes:

$$W'X' = W'X'YZ + W'X'Y'Z + W'X'Y'Z' + W'X'YZ'$$

Después se colocan los unos en las celdas correspondientes y se procede a realizar la agrupación y simplificación de los bloques.

	YZ			
WX	00	01	11	10
00	1	1	1	1
01		1	1	
11	1			
10	1			1

Hay que observar cómo cada uno de los bloques tiene cuando menos un 1 que es exclusivo de él. Además se tienen dos bloques de 4 celdas adyacentes, uno de ellos enmarcado en un cuadrado mientras que al otro lo conforman las esquinas del mapa, y en cada uno de ellos se eliminan 2 variables. Aparte de esto, se tiene un pequeño bloque de dos celdas.

La función simplificada queda como sigue:

$$F = \underline{X'Z'} + \underline{W'Z} + \underline{WY'Z'}$$

Del bloque de las esquinas
Del cuadrado de 4 celdas
Del bloque de 2 celdas

Ejemplo 5.7. Usando mapas de Karnaugh es posible simplificar la expresión booleana

$$F = A'B'C'D + A'B'CD + AB'C'D + AB'CD + AB'CD'$$

que resultó del problema de la embotelladora planteado al principio del capítulo.

En este caso se tiene la siguiente tabla:

	CD			
AB	00	01	11	10
00		1	1	
01				
11				
10		1	1	1

La expresión simplificada es:

$$F = B'D + AB'C$$

Ejemplo 5.8. Simplificar la expresión booleana

$$F = A'B'C'D + A'B'C + CD + AB'CD + AB'CD'$$

y obtener la expresión simplificada en sumas de productos y en productos de sumas.

Primero que nada se sabe que:

$$A'B'C = A'B'CD' + A'B'CD$$

$$CD = A'B'CD + A'BCD + ABCD + AB'CD$$

Usando la información, tanto los minitérminos que se complementaron con variables como los inicialmente completos, se tiene el siguiente mapa de Karnaugh:

	CD			
AB	00	01	11	10
00		1	1	1
01			1	
11			1	
10			1	1

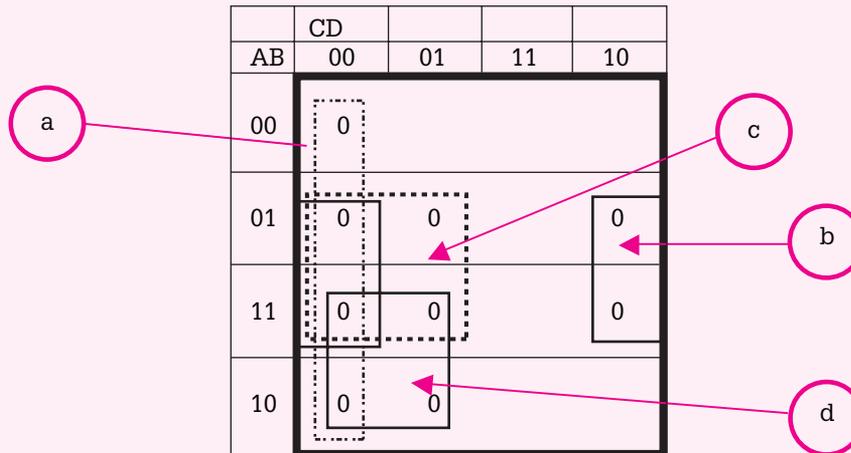
Hay que observar cómo un 1 puede estar considerado en diferentes bloques, como ocurre con el que está en la posición 0011.

En este mapa se tienen nuevamente 3 bloques, 2 de cuatro celdas y 1 de dos. Eliminando los que cambian de valor de una celda a otra se tiene que:

$$F = B'C + CD + A'B'D$$

Ésta es la expresión booleana simplificada en sumas de productos.

En el caso del “producto de sumas” se utiliza el mismo mapa de Karnaugh, pero en las celdas vacías se colocan ceros y se agrupa la información de manera semejante a cuando se tienen unos, como se muestra en el siguiente mapa:



La información se agrupó en este caso en cuatro bloques de 4 celdas cada uno de ellos, y para evitar confusiones en su lectura se le asignó una letra a cada bloque de tal forma que se obtiene la siguiente expresión complemento debido a que se usaron las celdas de ceros y no las de unos:

$$F' = C'D' + BD' + BC' + AC'$$

El asignarle una letra o número a un bloque permite ordenar mejor el resultado de tal forma que el primer término $C'D'$ es la lectura del bloque “a”, BD' lo es del bloque “b” y así sucesivamente. El orden en que se asigne la letra no es importante, ya que puede variar de persona a persona.

Complementando ambos miembros de la expresión booleana resulta que:

$$(F')' = (C'D' + BD' + BC' + AC')'$$

Aplicando ahora la ley de De Morgan:

$$F = (C + D) (B' + D) (B' + C) (A' + C)$$

Ésta es la expresión booleana simplificada en productos de sumas.

Hay que observar que no es igual la expresión booleana simplificada en sumas de productos que la que se obtuvo en productos de sumas, sin embargo se puede decir que son lógicamente equivalentes. Esto se puede demostrar usando teoremas del álgebra booleana o bien elaborando las tablas de verdad correspondientes.

A medida que crece el número de variables de la expresión booleana, se hace más complicado el mapa de Karnaugh ya que el número de celdas está dado por 2^n . Un mapa de 5 variables es equivalente a dos mapas de 4, como se muestra a continuación.

	CDE							
AB	000	001	011	010	110	111	101	100
00	4							
01								
11	1							
10	X	2		3				5

Cuanto crece el mapa, también se ve incrementada la cantidad de celdas adyacentes para agrupar la información. Por ejemplo, en un mapa de 4 variables una celda es adyacente a 4 celdas, mientras que en un mapa de 5 variables cada celda tiene 5 celdas adyacentes y así sucesivamente. En el mapa anterior la celda con sombreado oscuro es adyacente a las 5 celdas con sombreado más claro, la celda con la letra X es adyacente a las celdas numeradas con 1, 2, 3, 4, 5, de tal manera que cada celda se puede agrupar para formar un bloque de dos casillas, con cinco celdas más.

	CDE							
AB	000	001	011	010	110	111	101	100
00								
01								
11								
10								

En el mapa anterior el par de celdas con sombreado oscuro se pueden agrupar con las celdas de sombreado claro para formar un bloque de 4 casillas. Obsérvese cómo la frontera entre los dos mapas de 4 actúa como espejo.

Ejemplo 5.9. Considérese el siguiente mapa de Karnau, y a partir de él determínese la expresión booleana simplificada en sumas de productos y productos de sumas.

AB	CDE	001	011	010	110	111	101	100
00	1	1	1	1				1
01	1	1	1	1		1		
11			1			1		
10	1		1			1		1

Primero se tiene que la expresión booleana simplificada en sumas de productos es:

$$F = BDE + ADE + B'D'E' + A'C'$$

Para obtener la expresión booleana simplificada en productos de sumas se ponen ceros en las celdas vacías, se agrupa la información en bloque y se hace la lectura correspondiente.

	CDE							
AB	000	001	011	010	110	111	101	100
00					0	0	0	
01					0		0	0
11	0	0		0	0		0	0
10		0		0	0		0	

La expresión booleana que se lee a partir de la tabla es:

$$F' = ABD' + AD'E + ADE' + CDE' + A'B'CE + BCD'$$

Complementando ambos miembros de la igualdad y aplicando la ley de De Morgan se tiene finalmente que:

$$F = (A' + B' + D)(A' + D + E')(A' + D' + E)(C' + D' + E)(A + B + C' + E')(B' + C' + D)$$

El mapa de seis variables se divide en 4 mapas de cuatro variables. Cada una de las celdas es adyacente a 6 casillas con las mismas reglas ya conocidas.

Ejemplo 5.10. Considérese el siguiente mapa de Karnaugh y determínese la expresión booleana más simple en sumas de productos y productos de sumas.

	DEF							
ABC	000	001	011	010	110	111	101	100
000	1			1				
001		1	1			1	1	
011		1	1			1	1	
010	1			1	1	1	1	1
110					1	1	1	1
111	1	1	1	1		1	1	
101	1	1	1	1		1	1	
100	1			1				

Diagram illustrating the Karnaugh map with groupings (a, b, c, d, e) and arrows pointing to the corresponding simplified terms in the text below.

Se tiene que la expresión booleana simplificada en sumas de productos es:

$$F = A'C'D'F' + ACD' + B'C'D'F' + BC'D + CF$$

Para obtener la expresión de productos de sumas se tiene la tabla siguiente:

	DEF							
ABC	000	001	011	010	110	111	101	100
000		0	0		0	0	0	0
001	0			0	0			0
011	0			0	0			0
010		0	0					
110	0	0	0	0				
111					0			0
101					0			0
100		0	0		0	0	0	0

Diagram illustrating a Karnaugh map for a function F'. The map is a 4x4 grid with rows labeled ABC and columns labeled DEF. The cells contain 0s. The map is partitioned into groups by thick black lines. Seven pink circles (a-g) with arrows point to specific cells or groups:

- a: points to the cell (001, 000)
- b: points to the cell (010, 010)
- c: points to the cell (110, 110)
- d: points to the cell (100, 001)
- e: points to the cell (000, 100)
- f: points to the cell (011, 110)
- g: points to the cell (111, 111)

A partir del mapa de Karnaugh se tiene la expresión

$$F' = A'CE'F' + A'C'D'F + ABC'D' + AB'C'F + B'C'D + A'CEF' + ACDF'$$

Complementando ambos miembros de la igualdad y aplicando la ley de De Morgan resulta que

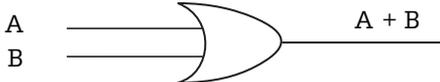
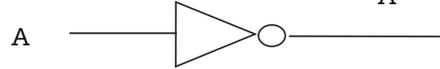
$$F = (A + C' + E + F)(A + C + D + F')(A' + B' + C + D)(A' + B + C + F') \\ (B + C + D')(A + C' + E' + F)(A' + C' + D' + F)$$

En algunos mapas de Karnaugh la solución no es única, ya que a veces la información se puede agrupar de manera diferente. Lo que importa al simplificar es obtener la expresión booleana simplificada óptima, independientemente de qué variables son eliminadas. Esto mismo puede suceder con los teoremas del álgebra booleana.

5.5 Compuertas lógicas

Un bloque lógico es una representación simbólica gráfica de una o más variables de entrada a un operador lógico, para obtener una señal determinada o resultado. Los símbolos varían de acuerdo con la rama donde se utilizan, o bien del fabricante. Cada bloque lógico representa un dispositivo que permite manipular la señal según el campo de acción: en mecánica se les llama válvulas (paso del aire o aceite); en electricidad apagadores, contactos (paso de corriente eléctrica); y en electrónica puertas o compuertas (paso de pulsos eléctricos). En este libro sólo se abordarán los símbolos usados en electrónica para la representación de las compuertas, ya que son los que interesan al área de la computación, sin embargo el tratamiento teórico por medio del álgebra booleana es válido para todos ellos independientemente del área.

Tabla 5.2 Compuertas básicas

Compuerta	Símbolo
O (Or)	
Y (And)	
No (Not)	
Or-exclusivo (Xor)	

Las compuertas pueden recibir una o más señales de entrada. En la tabla 5.2, A y B son señales que entran a la compuerta y pueden tener un valor de 1 o 0 dependiendo de si existe o no la señal, la cual procede de un sensor o bien de la salida de una compuerta anterior. Esos valores de entrada generan una sola salida, que a su vez también es 0 o 1 dependiendo de la compuerta de que se trate y de los valores de las señales de entrada.

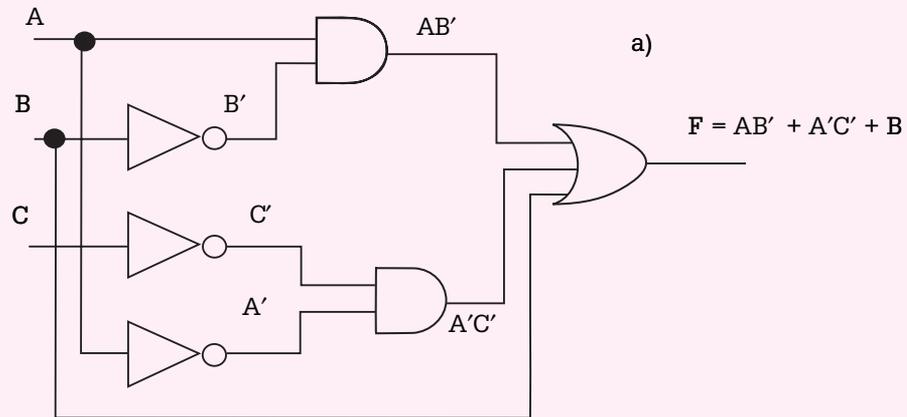
Para representar expresiones booleanas mediante compuertas lógicas es conveniente tener en cuenta las tablas de verdad de las compuertas básicas (operadores lógicos) Or, And y Not vistas en el capítulo de lógica matemática.

Ejemplo 5.11. Representar las siguientes expresiones booleanas usando compuertas lógicas básicas:

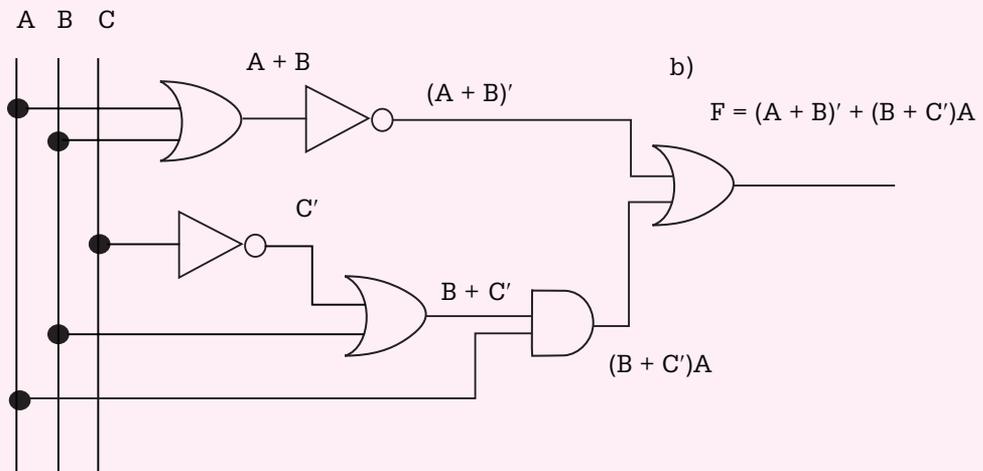
a) $F = AB' + A'C' + B$

b) $F = (A + B)' + (B + C')A$

La representación de (a) es:

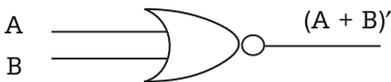


La representación de (b) es:



También existen compuertas lógicas compuestas como Nand y Nor, que son una combinación de los operadores Not y And para la primera y Not y Or para la segunda. En la tabla 5.3 se muestran los símbolos correspondientes.

Tabla 5.3 Compuertas compuestas

Compuerta	Símbolo
Nor	
Nand	
Xnor	

Generalmente los circuitos digitales se construyen con compuertas Nand y Nor, ya que son más fáciles de encontrar en el mercado, son más comunes desde el punto de vista del hardware y están disponibles en la forma de circuitos integrados. Debido a la preferencia de uso de estas compuertas en el diseño de los circuitos, es importante reconocer la relación que existe entre los circuitos construidos con compuertas And, Or y Not y su diagrama equivalente Nand o Nor.

Cuando se simplifica una función el resultado se puede presentar en “sumas de productos” o en “productos de sumas”, y en forma natural la presentación en suma de productos permite una implementación usando compuertas Nand mientras que el producto de sumas se puede representar más fácilmente con compuertas Nor, sin embargo es posible implementar cualquier expresión booleana sólo con compuertas Nand o sólo con compuertas Nor.

Ejemplo 5.12. ¿Cuál es el circuito de la expresión booleana

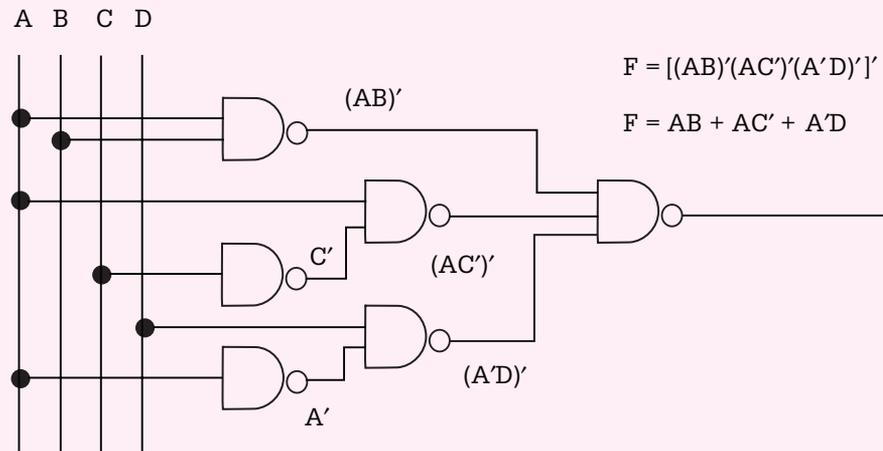
$$F = A(B + C') + A'D$$

hecho sólo con compuertas Nand?

Para obtener el circuito pedido es recomendable llevar la expresión dada a suma de productos:

$$F = AB + AC' + A'D$$

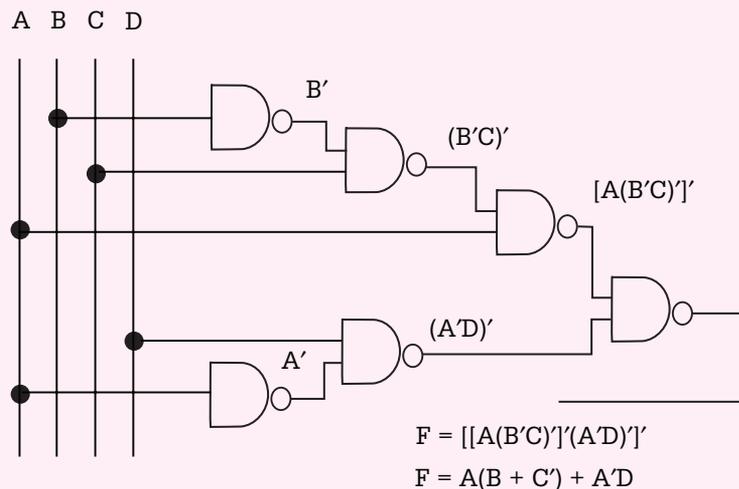
Por lo tanto, el circuito es el siguiente:



Hay que observar que al final se aplicó la ley de De Morgan para quitar la complementación del corchete y obtener el resultado. También se debe destacar que cuando entran dos o más señales a una compuerta Nand primero las multiplica y después complementa dicha multiplicación, pero cuando entra una señal sólo la complementa.

Por otro lado, si no se hubieran hecho las operaciones necesarias para quitar el paréntesis y tener la expresión en sumas de productos, también se podría representar únicamente con compuertas Nand aunque esto algunas veces es un poco más complicado:

$$F = A(B + C') + A'D$$



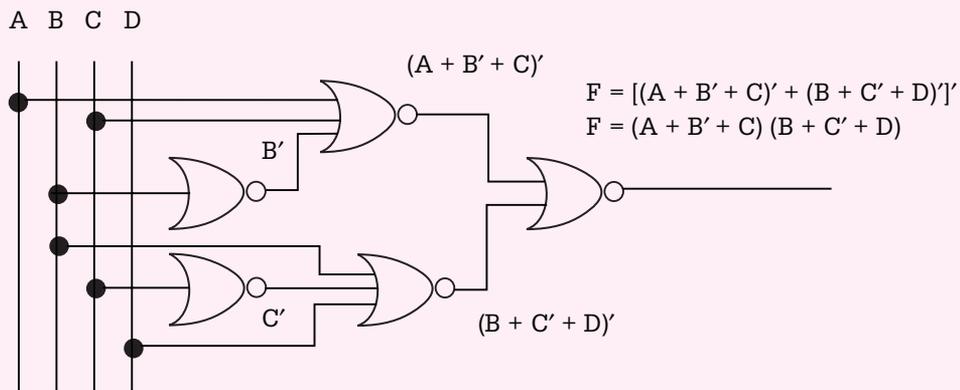
De la misma manera, el bloque lógico Nor facilita su uso cuando la expresión se encuentra dada en productos de sumas.

Ejemplo 5.13. Representar la expresión booleana

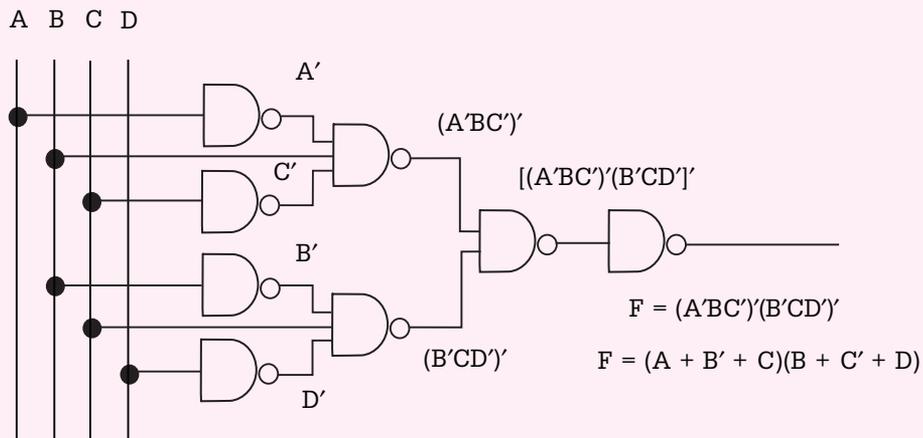
$$F = (A + B' + C)(B + C' + D)$$

usando sólo compuertas Nor.

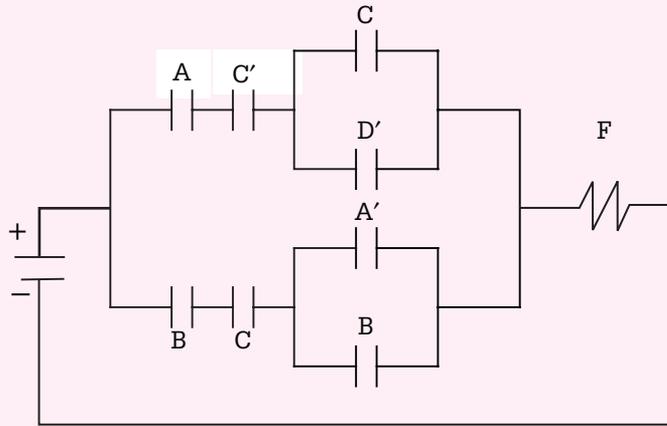
En este caso se tiene el siguiente esquema



La misma expresión booleana representada con compuertas Nand quedaría de la siguiente manera:



Ejemplo 5.14. Considérese el siguiente circuito:



- ¿Cuál es la expresión booleana sin simplificar que representa dicho circuito?
- Simplificar la expresión booleana usando teoremas del álgebra booleana.
- Por medio del mapa de Karnaugh simplificar la expresión del inciso (a) y expresar el resultado en sumas de productos.
- ¿Cuál es la expresión simplificada en productos de sumas?
- Comprobar, por medio de una tabla de verdad, que la expresión booleana obtenida en el inciso (c) es lógicamente equivalente a la obtenida en el inciso (d).
- Representar el resultado del inciso (c) en un circuito lógico, usando para ello compuertas básicas.
- ¿Cuál es el circuito del inciso (c) basado en compuertas Nand exclusivamente?
- ¿Cuál es el circuito lógico del inciso (c) basado en compuertas Nor exclusivamente?

La solución de cada inciso es la siguiente:

- a) La expresión booleana es

$$F = AC'(C + D') + BC(A' + B)$$

b) Simplificando mediante teoremas resulta que

$$F = AC'C + AC'D' + A'BC + BBC$$

$$F = 0 + AC'D' + A'BC + BC$$

$$F = AC'D' + BC(A' + 1)$$

$$F = AC'D' + BC$$

c) Se sabe que $AC'C = 0$ y $BBC = BC$, y sustituyendo esto en la expresión $F = AC'C + AC'D' + A'BC + BBC$ resulta que la expresión booleana a representar en el mapa es $F = AC'D' + A'BC + BC$. Aplicando la condición de que para representar un minitérmino en el mapa de Karnaugh éste debe contener todas las letras, a continuación se agregan las variables faltantes con sus posibles combinaciones:

$$AC'D' = AB'C'D' + ABC'D'$$

$$A'BC = A'BCD' + A'BCD$$

$$BC = A'BCD' + A'BCD + ABCD' + ABCD$$

A partir de la información se obtiene el siguiente mapa:

	CD			
AB	00	01	11	10
00				
01			1	1
11	1		1	1
10	1			

Del mapa se obtiene que la expresión booleana en sumas de productos es:

$$F = AC'D' + BC$$

lo cual concuerda con el resultado obtenido usando teoremas.

d) Para obtener el producto de sumas se colocan ceros en las casillas vacías y se agrupa la información:

	CD			
AB	00	01	11	10
00	0	0	0	0
01	0	0		
11		0		
10		0	0	0

A partir del mapa se puede leer que:

$$F' = A'C' + B'C + C'D$$

Complementado y aplicando leyes de De Morgan resulta que:

$$(F)' = (A'C' + B'C + C'D)'$$

$$F = (A + C)(B + C')(C + D')$$

e) Las expresiones booleanas obtenidas en los incisos (c) y (d) son

$$F = AC'D' + BC$$

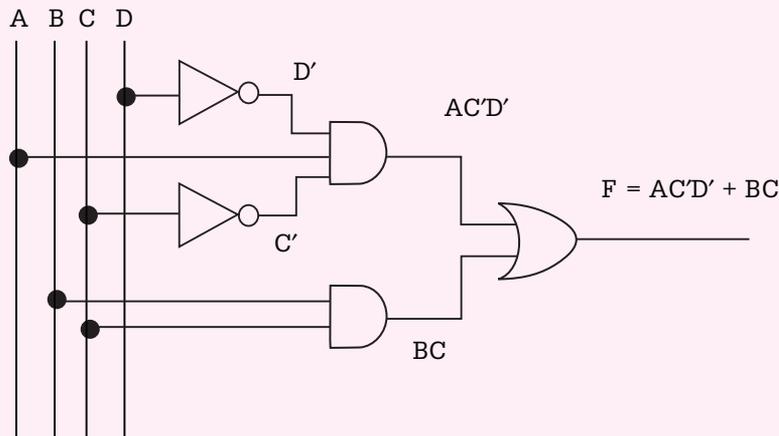
$$F = (A + C)(B + C')(C + D')$$

y a partir de éstas se tiene la siguiente tabla:

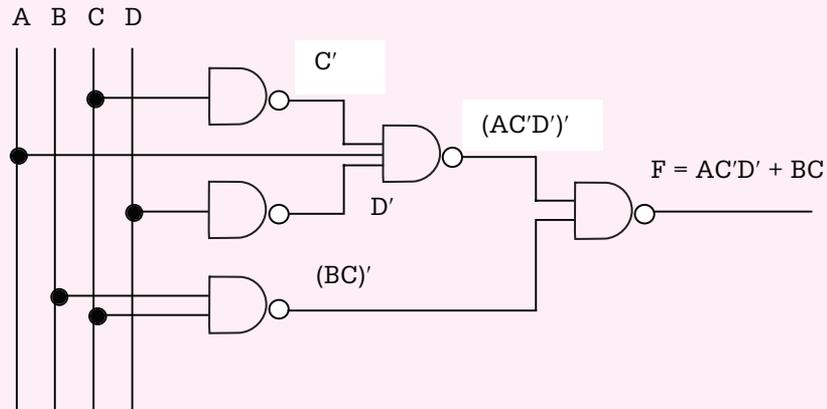
A	B	C	D	A'	B'	C'	D'	AC'	AC'D'	BC	AC'D' + BC	A + C	B + C'	C + D'	F
0	0	0	0	1	1	1	1	0	0	0	0	0	1	1	0
0	0	0	1	1	1	1	0	0	0	0	0	0	1	0	0
0	0	1	0	1	1	0	1	0	0	0	0	1	0	1	0
0	0	1	1	1	1	0	0	0	0	0	0	1	0	1	0
0	1	0	0	1	0	1	1	0	0	0	0	0	1	1	0
0	1	0	1	1	0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	1	0	0	1	0	0	1	1	1	1	1	1
0	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1
1	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1
1	0	0	1	0	1	1	0	1	0	0	0	1	1	0	0
1	0	1	0	0	1	0	1	0	0	0	0	1	0	1	0
1	0	1	1	0	1	0	0	0	0	0	0	1	0	1	0
1	1	0	0	0	0	1	1	1	1	0	1	1	1	1	1
1	1	0	1	0	0	1	0	1	0	0	0	1	1	0	0
1	1	1	0	0	0	0	1	0	0	1	1	1	1	1	1
1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1

Aquí se observa que las columnas sombreadas concuerdan en todas sus líneas, por lo tanto esto demuestra que $F = AC'D' + BC$ es lógicamente equivalente a $F = (A + C)(B + C')(C + D')$.

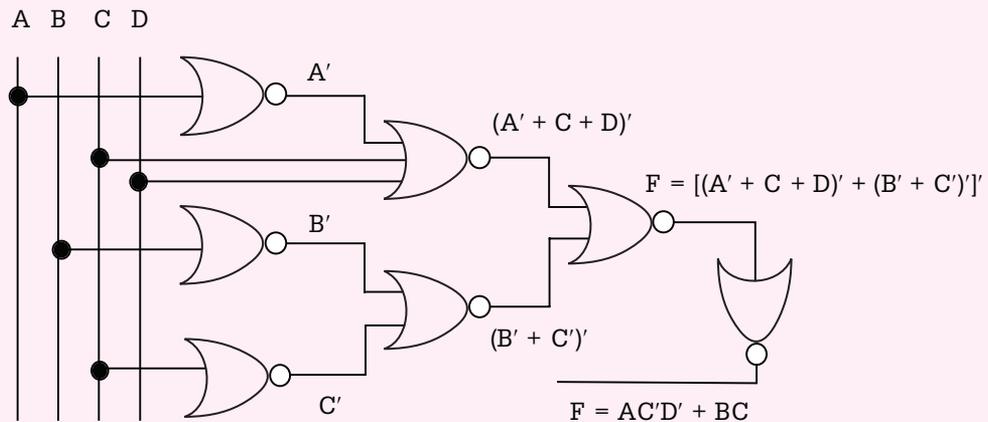
- f) La expresión obtenida en el inciso (c) es $F = AC'D' + BC$, y su representación con compuertas básicas es:



- g) La representación de la expresión booleana $F = AC'D' + BC$, sólo con compuertas Nand es la siguiente:



- h) La representación de la expresión booleana $F = AC'D' + BC$, sólo con compuertas Nor es la siguiente:



5.6 Aplicaciones del álgebra booleana

El álgebra booleana es una extensión de la lógica matemática, ya que utiliza los mismos principios y operadores lógicos (and, or, not, xor, nand, nor) así como los mismos valores, y gracias a esto John Von Neuman pudo crear la computadora de la primera generación.

Los dispositivos con los que se implementan las funciones booleanas se llaman “compuertas”, y al combinarse han permitido inicialmente la creación del “bulbo”, posteriormente la del “transistor” y actualmente la del “chip”, elementos con los cuales se construye todo tipo de aparato electrónico digital.

La electrónica digital es una parte de la electrónica que maneja información codificada en dos únicos estados: “falso” y “verdadero”, o más comúnmente 0 y 1. Electrónicamente se asigna a cada uno un voltaje o rango de voltaje determinado. Esta particularidad permite que, usando el álgebra booleana y con un sistema de numeración binario, se puedan realizar complejas operaciones lógicas o aritméticas sobre señales de entrada. La electrónica digital ha alcanzado una gran importancia debido a que se utiliza en el diseño de sistemas de automatización, robótica, etc., además de que constituye la piedra angular de las computadoras.

Las computadoras llevan a cabo su trabajo por medio de un microprocesador, el cual es un circuito de alta escala de integración (LSI) compuesto por muchos circuitos simples como flip-flops, contadores, decodificadores, comparadores, etc., todos en una misma pastilla de silicio en donde se utilizan compuertas del álgebra booleana para llevar a cabo las operaciones lógicas.

Las microoperaciones que lleva a cabo el microprocesador se realizan en lenguaje binario a nivel bit. Por ejemplo, si $A = 110010$, $B = 011011$ entonces el resultado de llevar a cabo las siguientes operaciones en donde intervienen los operadores lógicos (\wedge , \vee , \oplus , $'$) es:

$$A \wedge B = 110010 \wedge 011011 = 010010$$

$$A \vee B = 110010 \vee 011011 = 111011$$

$$A \oplus B = 110010 \oplus 011011 = 101001$$

$$A' = (110010)' = 001101$$

Basada en el álgebra booleana, la unidad lógica aritmética (ALU: Arithmetic Logic Unit) es la parte del microprocesador que realiza las operaciones aritméticas y lógicas en los datos.

Se sabe que toda computadora está integrada por las memorias ROM (Read Only Memory: Memoria de sólo lectura) y RAM (Random Access Memory: Memoria de acceso aleatorio). Cuando arranca una computadora, ésta debe saber qué hacer, lo cual implica que pueda correr un pequeño programa que le indique lo que debe realizar, qué programas debe ejecutar y en qué lugar debe comenzar. Esta información se guarda en un pequeño programa de sólo lectura que recibe el nombre de ROM, el cual está en lenguaje binario y utiliza operadores lógicos del álgebra booleana para la manipulación de la información. La información en este caso se graba eléctricamente y se borra también de la misma manera. Este tipo de memoria se



John von Neuman

(1903-1957)

Fue un matemático húngaro-estadounidense que realizó contribuciones importantes en física cuántica, análisis funcional, teoría de conjuntos, informática, economía, análisis numérico, hidrodinámica, estadística y muchos otros campos de la matemática.

Fue pionero de la computadora digital moderna, trabajó con Eckert y Mauchly en la Universidad de Pennsylvania y publicó un artículo acerca del almacenamiento de programas. El concepto de programa almacenado permitió la lectura de un programa dentro de la memoria de la computadora, y después la ejecución de las instrucciones del mismo sin tener que volverlas a escribir. La primera computadora en usar el citado concepto fue la llamada EDVAC (Electronic Discrete Variable Automatic Computer), desa-



rollada por Von Neumann, Eckert y Mauchly. Los programas almacenados dieron a las computadoras flexibilidad y confiabilidad, haciéndolas más rápidas y menos sujetas a errores que los programas mecánicos.

llama Memoria ROM programable eléctricamente (EEPROM). En las computadoras ésta se encuentra en lo que se llama BIOS, la cual es una memoria donde se guarda información de la “tarjeta madre” de los conectores y dispositivos de la PC.

La RAM puede borrarse y grabarse las veces que se desee, la desventaja es que la información grabada en ella sólo se puede utilizar mientras se tenga energía, y se usa como almacenamiento temporal. Existen dos variantes para la memoria RAM: SRAM y DRAM. La SRAM es conocida como memoria estática y en ella los valores binarios o información que se almacena utilizan compuertas del álgebra booleana, por lo que mientras se tenga energía la información en ella se mantendrá intacta. La DRAM es conocida como memoria dinámica y está hecha con celdas que almacenan los datos como cargas en condensadores; la presencia o ausencia de carga en el condensador se interpreta como 1 o 0 binarios, manipulados mediante álgebra booleana. La DRAM es una memoria que requiere refrescarse periódicamente para mantener memorizados los datos, de ahí el nombre de memoria dinámica.

Como se puede ver, la computadora está integrada por elementos que utilizan el álgebra booleana para su desarrollo y funcionamiento. Sin embargo, no es para lo único que se utiliza el álgebra booleana, ya que otra de sus aplicaciones que actualmente está teniendo mucho éxito es la relacionada con la construcción de robots.

Un robot está integrado por elementos mecánicos, eléctricos y electrónicos, y el área de conocimiento en este caso es la “mecatrónica”. El motor eléctrico es un dispositivo que convierte la energía eléctrica en energía mecánica rotacional, que se utiliza para darle movimiento a los medios de locomoción del robot como son ruedas, brazos y tenazas. El motor puede ser de corriente continua o motor de pasos.

Los medios de locomoción permiten al robot desplazarse de un lugar a otro por medio de ruedas, barras u orugas. Algunos robots deben sostener o manejar objetos y para ello se utilizan tenazas. Algunas veces el movimiento no se proporciona directamente a los medios de locomoción, sino que es necesaria una interfase de transmisión para aumentar la fuerza, reducir la intensidad de giro o cambiar la naturaleza del movimiento (de circular a lineal) por medio de pistones, engranes, levas o poleas.

El funcionamiento de los distintos elementos del robot depende de la señal que se mande de los distintos sensores. Los sensores permiten al robot manejarse con cierta inteligencia al interactuar con el medio, ya que detectan situaciones en las cuales el robot debe llevar a cabo la actividad programada. Entre los diferentes sensores que se utilizan con frecuencia en robots están los sensores ópticos, magnéticos, de ultrasonido, presión, temperatura, nivel e incluso cámaras de video.

Para que el robot lleve a cabo todas las actividades, es necesario el circuito de control (cerebro del robot) que le permita decidir qué hacer cuando se presente una determinada situación. Por ejemplo, qué debe hacer el robot si a su paso se interpone una barrera (girar 90° a la izquierda y avanzar, girar 180° y avanzar, detenerse, etc.). ¿Qué hacer si detecta temperaturas altas (emitir un sonido, parar)? ¿Qué hacer si encuentra un objeto de cierto color (tomarlo y transportarlo)? En fin, todas esas actividades que puede llevar a cabo el robot y para lo cual fue creado, deben estar programadas en el circuito de control y nuevamente el álgebra booleana es la base para el diseño de dicho circuito, el cual se representa inicialmente por medio de una expresión booleana que se simplifica por medio de teoremas del álgebra booleana o mapas de Karnaugh y se implementa usando las compuertas lógicas.

5.7 Resumen

El álgebra es un área de las matemáticas que ocupa un lugar privilegiado, sobre todo por la aplicación de la misma a la computación. Por medio del álgebra booleana es posible diseñar hardware que es la parte fundamental de las computadoras, los robots y todos los sistemas de funcionamiento automático.

Los robots, computadoras o cualquier sistema de funcionamiento automático requieren del uso de elementos mecánicos, eléctricos y electrónicos para llevar a cabo alguna actividad. La forma ordenada en que deben trabajar dichos elementos se controla por medio de un circuito implementado a base de compuertas lógicas.

Cuando se desea que un sistema trabaje de manera automática, primero se representa el funcionamiento de dicho sistema por medio de una expresión booleana. Esta expresión booleana está integrada por variables y cada una de éstas representa la señal de un sensor, la cual puede ser falso o verdadero.

Por lo general la expresión booleana resultante del planteamiento de un problema no es la más simple, sino que tiene variables redundantes que pueden ser eliminadas por medio de:

- a) Teoremas del álgebra booleana.
- b) Mapas de Karnaugh.

El método para simplificar expresiones booleanas usando teoremas del álgebra booleana consiste en usar éstos para eliminar las variables redundantes hasta obtener una expresión simplificada que realice lo mismo que la expresión inicial que tenía las variables redundantes, pero que al ser más simple el circuito de control es por lo tanto más rápido, económico y eficaz.

El método para simplificar expresiones booleanas mediante mapas de Karnaugh consiste en representar la expresión booleana con n variables diferentes en una tabla de forma cuadrada o rectangular que tiene 2^n celdas y que recibe el nombre de mapa de Karnaugh. La expresión booleana simplificada es el resultado de agrupar la información de celdas adyacentes en bloques rectangulares o cuadrados de 1, 2, 4, 8, ..., 2^n , y después leer la expresión conservando las variables que no cambian de valor de un reglón con respecto a otro o de una columna con relación a otra en cada uno de los bloques en que fue agrupada la información y eliminando las variables que sí sufren un cambio de valor de un renglón con respecto a otro o de una columna con relación a otra.

Por último, esta función booleana simplificada, ya sea por teoremas o mapas de Karnaugh, se representa por medio de símbolos gráficos (bloques lógicos) de cada uno de los operadores lógicos and, or, not, xor, nand, nor y xnor, considerando que las compuertas más comunes son las nand y las nor, mismas que al combinarse permiten suplir las demás compuertas.

5.8 Problemas

5.1. Obtener la tabla de verdad para la siguiente expresión booleana:

$$F = A'B'C' + A'B'CD + A'BC + A'BC'D + ABC' + ABC + AB'D + AB'C'D'$$

5.2. Obtener la tabla de verdad para cada una de las siguientes expresiones booleanas:

a) $F = A'B'C'D' + A'B'CD' + A'BC'D + A'BCD + ABC'D' + ABCD' + AB'C'D' + AB'CD'$

b) $F = (A + BD')(C'DB + AB' + DA)'$

c) $F = [A'(BC + D)' + B'A]$

5.3. Simplificar las siguientes expresiones booleanas usando los teoremas del álgebra booleana, y verificar los resultados por medio de mapas de Karnaugh.

a) $F = A'B'D' + A'BD' + A'BD + ABD$

b) $F = A'CD + ACD + A'B'D + A'B'C + AB'D + AB'CD'$

c) $F = A'B'C'D' + A'B'CD' + A'BC'D + A'BCD + ABC'D' + ABCD' + AB'C'D' + AB'CD'$

- d) $F = A'B'C'D'E + A'B'C'DE + A'B'C'DE' + A'BC + ABC + ABC'D'E' + ABC'D'E + ABC'DE + AB'C'D'E + AB'C'DE + AB'CDE + AB'CD'E$
- e) $F = ((A+B)' + C' + D')((AC)' + (A + (BC)')' + D)$
- f) $F = A'B'C'D + A'B'CD + A'B'CD' + A'BCD + ABCD' + AB'C'D + AB'CD + AB'CD'$
- g) $F = A'B'C'D' + A'B'CD + A'B'CD' + ABC'D + ABCD + ABCD' + AB'C'D' + AB'CD + AB'CD'$

5.4. Simplificar las siguientes expresiones booleanas usando los teoremas del álgebra booleana y verificar los resultados por medio de mapas de Karnaugh.

- a) $F = A'B'C'D' + A'B'CD + A'B'CD' + A'BC'D + A'BCD + A'BCD' + ABCD + ABCD' + AB'C'D' + AB'CD'$
- b) $F = W'X'Y'Z' + W'X'YZ + WXY'Z + WXYZ + WX'Y'Z' + WX'YZ + WX'YZ + WX'YZ' + W'XY'Z'$
- c) $F = W'X'Y'Z + W'XY'Z' + W'XYZ + W'XYZ' + WXY'Z + WXYZ + WXYZ'$
- d) $F = A'B'C'D' + A'B'CD' + A'BC'D' + A'BCD' + ABC'D' + ABCD + ABCD' + AB'CD + AB'CD'$
- e) $F = A'B'C'D' + A'B'CD + A'B'CD' + A'BC'D + A'BCD + A'BCD' + ABCD + ABCD' + AB'C'D' + AB'CD'$
- f) $F = B'CD + ABC + A'BD' + ABC'D' + AB'CD'$
- g) $F = A'BC + BC'D' + ABC + AB'C'D' + AB'CD'$

5.5. En cada uno de los siguientes incisos obtener la expresión booleana simplificada en sumas de productos y en productos de sumas. Plantear el mapa y la agrupación correspondiente.

a)

	CDE							
AB	000	001	011	010	110	111	101	100
00	1	1						
01	1	1			1	1	1	
11		1	1			1	1	1
10							1	1

b)

	CDE							
AB	000	001	011	010	110	111	101	100
00	1	1	1	1	1	1	1	
01	1						1	1
11	1	1	1				1	1
10		1			1	1		

c)

	CDE							
AB	000	001	011	010	110	111	101	100
00	1	1					1	
01			1	1	1	1		1
11	1	1	1	1	1		1	1
10			1				1	

d)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1			1	1
01	1			1				
11	1	1		1			1	1
10	1	1					1	1

e)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1	1			
01		1	1			1		1
11	1		1			1	1	1
10				1	1	1	1	

5.6. En cada uno de los siguientes incisos obtener la expresión booleana simplificada en sumas de productos y en productos de sumas. Plantear el mapa y la agrupación correspondiente.

a)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1				1
01		1	1	1	1	1		
11	1		1	1	1	1	1	1
10				1			1	1

b)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1	1		1	1
01	1			1	1		1	1
11	1		1	1				
10	1			1			1	1

c)

	CDE							
AB	000	001	011	010	110	111	101	100
00			1	1	1	1	1	1
01	1		1	1				
11	1			1				1
10			1	1	1	1		1

d)

	CDE							
AB	000	001	011	010	110	111	101	100
00			1	1			1	1
01		1	1				1	1
11	1				1		1	1
10	1				1		1	1

e)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1		1		1
01	1		1		1		1	
11		1		1		1		1
10	1		1		1		1	

f)

	CDE							
AB	000	001	011	010	110	111	101	100
00	1	1	1	1	1	1	1	1
01								
11								
10	1	1	1	1	1	1	1	1

- 5.7. Representar con compuertas básicas (And, Or y Not), con compuertas Nand (exclusivamente) y con compuertas Nor (exclusivamente), la expresión lógica:

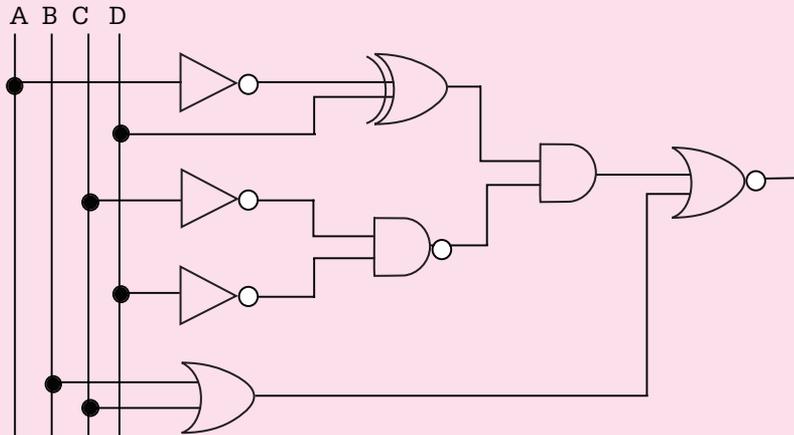
$$F = AB'C + A'B'D' + AD$$

- 5.8. Representar con compuertas básicas (And, Or y Not), con compuertas Nand (exclusivamente) y con compuertas Nor (exclusivamente), la expresión lógica:

$$F = (B' + C + D')(A + C' + D)B'$$

- 5.9. Obtener las compuertas Not, And, Or, Nor, X-or y X-nor con base en compuertas Nand exclusivamente.
- 5.10. Obtener las compuertas Not, And, Or, Nand, X-or y X-nor con base en compuertas Nor exclusivamente.

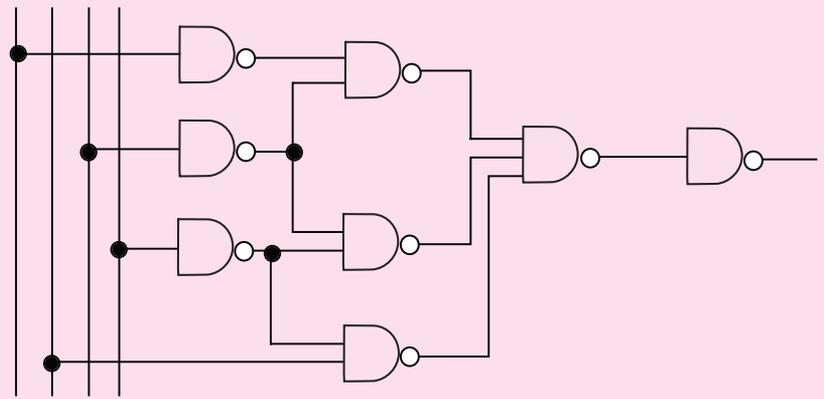
5.11. Considérese el siguiente circuito lógico:



- Obtener la función booleana de salida (sin simplificar).
 - Obtener la función booleana simplificada en sumas de productos.
 - Obtener la función booleana simplificada en productos de sumas.
 - Elaborar la tabla de verdad que muestre que las expresiones booleanas obtenidas en los incisos (a) y (b) son lógicamente equivalentes.
 - Implementar el diagrama de la expresión booleana obtenida en el inciso (b) usando exclusivamente compuertas Nand.
 - Hacer el diagrama correspondiente de la expresión booleana obtenida en el inciso (c) usando exclusivamente compuertas Nor.
- 5.12. En relación con los circuitos de cada uno de los incisos (i) a (v) obtener:
- La función booleana de salida.
 - La función booleana más simple en sumas de productos.
 - La función booleana simplificada en productos de sumas.
 - La tabla de verdad que muestre que las expresiones booleanas obtenidas en los incisos (a), (b) y (c) son lógicamente equivalentes.
 - El circuito lógico de la expresión booleana del inciso (b), con base en compuertas Nand exclusivamente.
 - El circuito lógico de la expresión booleana obtenida en el inciso (c), a base de compuertas Nor exclusivamente.

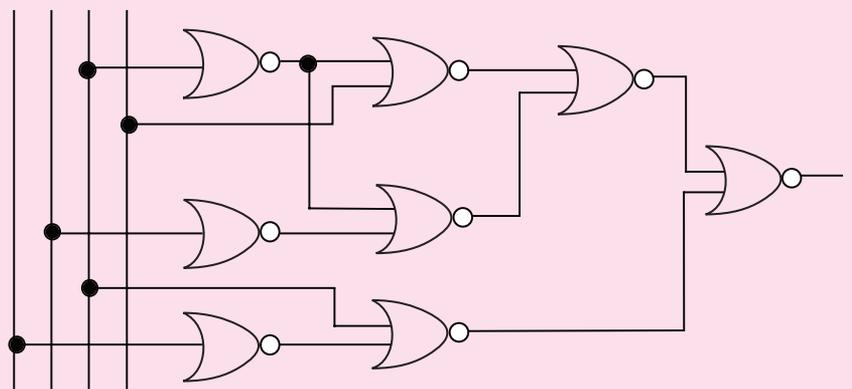
i)

A B C D



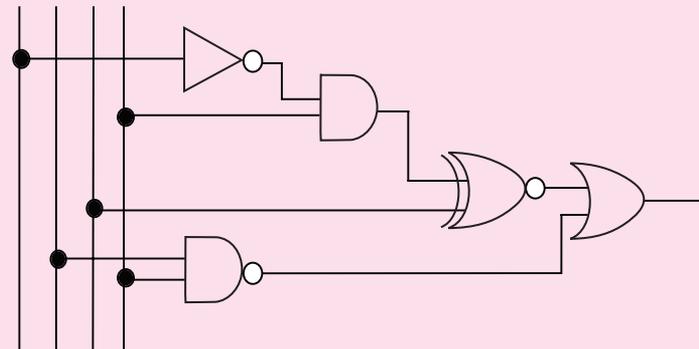
ii)

A B C D

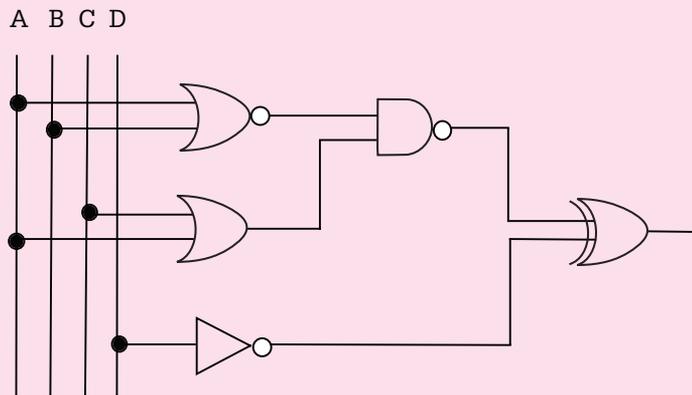


iii)

A B C D



iv)



v)

