CAPÍTULO I

1. SISTEMAS DE NUMERACIÓN

Un sistema de numeración es el conjunto de símbolos y reglas que se utilizan para la representación de datos numéricos o cantidades. Un sistema de numeración se caracteriza por su base, que es el número de símbolos distintos que utiliza y además es el coeficiente que determina cuál es el valor de cada símbolo dependiendo de la posición que ocupe.

Los actuales sistemas de numeración son netamente posicionales, en los que el valor relativo que representa cada símbolo o cifra depende de su valor absoluto y de la posición que ocupa dicha cifra con respecto a la coma decimal. La coma decimal (,) que separa la parte entera de la parte fraccionaria, en ambientes informáticos, está representada por el punto decimal (.).

En este capítulo se estudiarán los sistemas de numeración decimal, binario, octal y hexadecimal, cómo están conformados y las conversiones de un sistema a otro.

De manera que el sistema binario es el más importante de los sistemas digitales, pero también hay otros que también lo son, por ejemplo, el sistema decimal es el que se utiliza para representar cantidades fuera de un sistema digital y viceversa; hay situaciones donde se deben llevar números decimales a binarios para hacer algún tipo de procesamiento. La computadora debido a su construcción basada en circuitos electrónicos digitales, almacena y maneja la información con el sistema binario. Este el motivo que obliga a transformar internamente todos los datos, a una representación binaria para que la máquina sea capaz de procesarlos. Pero también existen otros dos sistemas con los cuales se pueden realizar aplicaciones en los sistemas digitales; éstos son el sistema octal (Base 8) y el hexadecimal (Base 16), éstos se usan con la finalidad de ofrecer un eficaz medio de representación de números binarios grandes, teniendo la ventaja de poder convertirse fácilmente al y del binario, y ser los más compatibles con éste.

1.1 SISTEMA DE NUMERACIÓN DECIMAL.

El hombre, desde hace tiempo ha utilizado como sistema para contar el sistema decimal, que derivó del sistema indoarábigo, posiblemente se adoptó este sistema por contar con 10 dedos en las manos.

El sistema decimal utiliza un conjunto de símbolos, cuyo significado depende de su posición relativa al punto decimal, que en caso de ausencia se supone colocado implícitamente a la derecha.

El hombre ha utilizado el sistema numérico decimal, basado en diez símbolos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9), que, al combinarlos, permiten representar las cantidades imaginadas; es por esto que se dice que utiliza la base 10.

1.2 SISTEMA DE NUMERACIÓN BINARIA.

Este sistema de base 2 es el más sencillo de todos por poseer sólo dos dígitos, fue introducido por Leibniz en el Siglo XVII, es el sistema que internamente utilizan los circuitos digitales que configuran el hardware de las computadoras actuales.

Los dos dígitos, llamados bits (Contracción de binary digit), son el uno (1) y el cero (0), por lo cual el equivalente decimal se obtendrá al sumar los pesos correspondientes a los bits 1.

En bit más significativo (MSB) es aquel que se ubica más a la izquierda (el que tiene mayor valor). El bit menos significativo (LSB) es aquel que está más a la derecha y que tiene el menor valor.

Para la medida de unidades de información representada en binario, se utilizan una serie de múltiplos de bit que poseen nombre propio:

- **Nibble o Cuarteto**: Es el conjunto de cuatro bits (1001).
- **Byte u Octeto**: Es el conjunto de ocho bits (10101010).
- **Kilobyte (Kb)**: Es el conjunto de 2^10 bits (1.024 * 8 bits)
- **Megabyte (Mb)**: Es el conjunto de 2^20 Kilobytes bits (1.024^2 * 8 bits)
- **Gigabyte (Gb)**: Es el conjunto de 2^30 Megabytes bits (1.024^3 * 8 bits)
- **Terabyte (Tb)**: Es el conjunto de 2^40 Gigabytes bits (1.024^4 * 8 bits)

La razón por la que se utiliza el factor 1.024 en vez de 1.000, es por ser el múltiplo de 2 más próximo a 1000, cuestión importante desde el punto de vista informático (2^{10} = 1.024).

1.3 SISTEMA DE NUMERACIÓN OCTAL.

Se trata de un sistema de numeración en base 8 que utiliza 8 símbolos para la representación de cantidades. Los símbolos utilizados son: 0, 1, 2, 3, 4, 5, 6, 7.

Este sistema también posicional, ya que cada una de sus cifras tiene como posición la relativa al punto decimal que, en caso de no aparecer se supone implícita al lado derecho del número, este proporciona un método conveniente para la representación de códigos y números binarios utilizados en los sistemas digitales.

1.4 SISTEMA DE NUMERACIÓN HEXADECIMAL.

El sistema hexadecimal emplea la base 16. Así, tiene 16 posibles símbolos digitales. Utiliza los dígitos del 0 al 9, más las letras A, B, C, D, E y F como sus 16 símbolos digitales. Cada dígito hexadecimal representa un grupo de cuatro dígitos binarios. Es importante recordar que los dígitos hex (Abreviatura de hexadecimal) de A a F son equivalentes a los valores decimales de 10 a 15.

1.5 REPRESENTACIÓN DE CANTIDADES.

En los sistemas digitales, la información que se está procesando, por lo general, se presenta en forma binaria.

Desafortunadamente, el sistema numérico decimal no se presta para una implantación conveniente en sistemas digitales. Por ejemplo, resulta muy difícil diseñar equipo electrónico para que pueda funcionar con 10 diferentes niveles de voltaje (para que cada uno representara un carácter decimal, de 0 a 9). Por Otro lado, es muy fácil diseñar circuitos electrónicos precisos pero simples que operen con sólo dos niveles de voltaje. Por esta razón, casi todos los sistemas digitales utilizan el sistema numérico binario (base 2) como base de sus operaciones, aunque con frecuencia se emplean otros sistemas junto con el binario.

En el sistema binario solamente hay dos símbolos o posibles valores de dígitos, el 0 y el 1. No obstante, este sistema de base 2 se puede utilizar para representar cualquier cantidad que se denote en sistema decimal o algún otro sistema numérico. En general, se necesitarán muchos dígitos binarios para expresar una cantidad determinada. Este es un sistema de valor posicional, en donde cada dígito binario tiene su valor propio expresado como potencia de 2.

En el sistema binario, el término dígito binario se abrevia a menudo como bit. El bit más significativo (MSB) es aquel que se ubica más a la izquierda (el que tiene el mayor valor). El bit menos significativo (LSB) es aquel que está más a la derecha y que tiene el menor valor.

Las cantidades binarias pueden representarse por medio de cualquier dispositivo que solamente tenga dos estados de operación o posibles condiciones. Por ejemplo, un interruptor sólo tiene dos estados: abierto o cerrado. Arbitrariamente, podemos hacer que un interruptor abierto represente el cero (0) binario y que uno cerrado represente el uno (1) binario.

El sistema de numeración binario es el más importante de los sistemas digitales, pero hay otros que también lo son. La importancia del sistema decimal radica en que se utiliza universalmente para representar cantidades fuera de un sistema digital. Esto significa que habrá situaciones en las cuales los valores decimales tengan que convertirse en valores binarios antes de que se introduzcan al sistema digital. Por ejemplo, cuando se presiona un número decimal en una calculadora portátil (o una computadora), los circuitos que están dentro del dispositivo convierten el número decimal en un valor binario.

De igual manera, habrá situaciones en que los valores binarios de las salidas de un circuito digital tengan que convertirse a valores decimales para presentarse al mundo exterior. Por ejemplo, una calculadora (o computadora) utiliza números binarios para calcular respuestas a un problema, luego las convierte a un valor decimal antes de exhibirías en la pantalla.

Además del binario y el decimal, otros dos sistemas de numeración encuentran amplias aplicaciones en los sistemas digitales. Los sistemas octal (base 8) y hexadecimal (base 16) se usan con la misma finalidad: ofrecer un medio eficaz de representación de números binarios grandes. Como observaremos, ambos sistemas numéricos tienen la ventaja de que pueden convertirse fácilmente al y del binario.

En un sistema digital, se pueden utilizar tres o cuatro de estos sistemas de numeración al mismo tiempo, de modo que un entendimiento de la operación del sistema requiere la facultad de convertir de un sistema numérico a otro.

1.5.1 Conversión de Decimal a Binario.

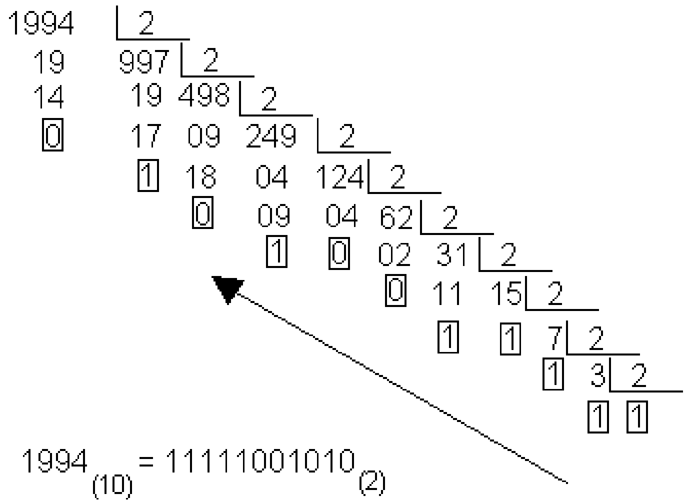
Existen dos maneras de convertir un número decimal a su representación equivalente en el sistema binario. En el primero el número decimal se expresa simplemente como una suma de potencias de 2 y luego los unos y los ceros se escriben en las posiciones adecuadas de bits. Para ilustrar lo anterior, consideremos el siguiente ejemplo:

$$\begin{aligned} 45_{10} = 32 + 8 + 4 + 1 &= 2^5 + 0 + 2^3 + 2^2 + 0 + 2^0 \\ &= 1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1_2 \end{aligned}$$

Obsérvese que se coloca un 0 en las posiciones 2^1 y 2^4, ya que todas las posiciones deben tomarse en cuenta.

El segundo método es llamado, Método de las Divisiones Sucesivas entre Dos. Se trata de dividir sucesivamente el número decimal y los sucesivos cocientes entre dos (2), hasta que el cociente en una de las divisiones tome el valor cero (0). La unión de todos los restos obtenidos, escritos en orden inverso, nos proporciona el número inicial expresado en el sistema binario.

Ejemplo: Convertir el número decimal 1994 en binario.



1.5.2 <u>**Conversión de Binario a Decimal**</u>.

El sistema de numeración binario es un sistema posicional donde cada dígito binario (bit) tiene un valor basado en su posición relativa al LSB. Cualquier número binario puede convertirse a su equivalente decimal, simplemente sumando en el número binario los valores de las diversas posiciones que contenga un 1. Para ilustrar lo anterior consideremos el siguiente ejemplo:

$$\begin{array}{ccccccccc} 1 & & 1 & & 0 & & 1 & & 1_2 \quad (\textit{binario}) \\ 2^4 & + & 2^3 & + & 0 & + & 2^1 & + & 2^0 = 16 + 8 + 2 \;\; 1 \\ & & & & & & & & = 27_{10} \ (\textit{decimal}) \end{array}$$

Nótese que el procedimiento consiste en determinar los valores (es decir, las potencias de 2) de cada posición de bit que contenga un 1 y luego sumarlos. Nótese también que el MSB tiene un valor de 2^4a pesar de que es el quinto bit; esto se debe a que el LSB es el primer bit y tiene un valor de 2^0

1.5.3 Conversión de Decimal a Octal.

Igualmente que en la conversión de decimal a binario, por medio del Método de Divisiones Sucesivas, pero en este caso por ocho (8).

Ejemplo: Convertir el número decimal 1999 a octal.

```
1994 |8
 39 249 |8
 74  09  31 |8
 [2] [1] [7] [B]
        <--------
```

$$1994_{(10)} = 3712_{(8)}$$

1.5.4 Conversión de Octal a Binario.

Para convertir un número octal a binario se sustituye cada dígito octal por sus correspondientes tres dígitos binarios.

TABLA N°.1.1

EQUIVALENCIA OCTAL-BINARIO

DÍGITO OCTAL	DÍGITO BINARIO
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

FUENTE:

Ejemplo: Convertir el número octal 75643.57 a binario:

7	5	6	4	3	.	5	7
111	101	110	100	011	.	101	111

Entonces,

$75643.57_{(8)} = 111101110100011.101111_{(2)}$

1.5.5 Conversión de Binario a Octal.

Para convertir un número binario a octal se realiza un proceso inverso al anterior. Se agrupan los dígitos de 3 en 3 a partir del punto decimal hacia la izquierda y hacia la derecha, sustituyendo cada trío de dígitos binarios por su equivalente dígito octal.

Ejemplo: Convertir el número binario 1100101001001.1011011 en octal.

001	100	101	001	001	.	101	101	100
1	4	5	1	0	.	5	5	4

Luego,
$1100101001001.1011011_{(2)} = 14510.554_{(8)}$

1.5.6 Conversión de Binario a Hexadecimal.

Se realiza un proceso inverso al anterior. Se agrupan los dígitos binarios de 4 en 4 a partir del punto decimal hacia la izquierda y hacia la derecha, sustituyendo cada cuarteto por su correspondiente dígito hexadecimal. Agregando ceros cuando sea necesario para completar un grupo de 4 bits.

1.5.7 Conversión de Octal a Hexadecimal.

Esta conversión realiza un paso intermedio utilizando el sistema binario. Primero se convierte el número octal en binario y éste se pasa a hexadecimal.

Ejemplo: Convertir el número 144 en hexadecimal.

1	4	4
001	100	100

$144_{(8)} = 1100100_{(2)}$

0110	0100
6	4

$1100100_{(2)} = 64_{(16)}$

1.5.8 Conversión de Hexadecimal a Octal.

Se realiza un paso intermedio utilizando el sistema binario. Se convierte en binario y éste en octal.

Ejemplo: Convertir el número hexadecimal 1F4 en octal.

1	F	4
0001	1111	0100

$1F4_{(16)} = 111110100_{(2)}$

111	110	100
7	6	4

$111110100_{(2)} = 764_{(8)}$

1.5.9 Conversión de Decimal a Hexadecimal.

De igual manera, la conversión de decimal a hexadecimal se puede efectuar por medio de la división repetida por 16.

Siguiendo el mismo método utilizado en las conversiones de decimal a binario y de decimal a octal.

Ejemplo: Convertir el número decimal 1994 a hexadecimal:

1994 |16
039 124 |16
074 $\boxed{12}$ $\boxed{7}$
$\boxed{10}$

por lo tanto,

$1994_{(10)} = 7CA_{(16)}$

1.5.10 Conversión de Hexadecimal a Binario.

Se sustituye cada dígito hexadecimal por su representación binaria con cuatro dígitos.

TABLA N°.1.2

EQUIVALENCIA HEXADECIMAL-BINARIO

DÍGITO HEXADECIMAL	DÍGITO BINARIO
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Ejemplo: Convertir el número hexadecimal 7BA3.BC a binario.

7	B	A	3	.	B	C
0111	1011	1010	0011	.	1011	1100

1.5.11 Conversión de Hexadecimal a Decimal.

Un número hex se puede convertir en su equivalente decimal utilizando el hecho de que cada posición de los dígitos hex tiene un valor que es una potencia de 16. El LSD tiene un valor de $16^0 = 1$; el siguiente dígito en secuencia tiene un valor de $16^1 = 16$; el siguiente tiene un valor de $16^2 = 256$ y así sucesivamente El proceso de conversión se demuestra en los ejemplos que siguen

$$
\begin{aligned}
356_{16} &= 3 \times 16^2 + 5 \times 16^1 + 6 \times 16^\circ \\
&= 768 + 80 + 6 \\
&= 854_{10} \\
2AF_{16} &= 2 \times 16^2 + 10 \times 16^1 + 15 \times 16^\circ \\
&= 512 + 160 + 15 \\
&= 687_{10}
\end{aligned}
$$

CAPÍTULO II

2. INTRODUCCIÓN A LOS CIRCUITOS LÓGICOS

Muchos componentes utilizados en sistemas de control presentan dos estados claramente diferenciables. El ejemplo más típico y conocido en el mundo de los automatismos, es el de los contactos y relés.

En los automatismos lógicos se manejan continuamente los conceptos: abierto-cerrado, conduce-no conduce, activado-no activado, tensión alta o baja, mayor que o menor que, etc., siempre haciendo referencia a dos estados posibles.

Para la sistematización del comportamiento de estos elementos, se representan los dos estados por los símbolos 0 y 1. De esta forma se podrá utilizar una serie de leyes y propiedades comunes a todos ellos, teniendo una independencia de la naturaleza física del componente en sí; es decir, bajo este punto de vista se tratará por igual un contacto (0 abierto, 1 cerrado) que un cilindro neumático (0 contraído, 1 extendido) o una electroválvula (0 no pasa, 1 pasa).

2.1 ÁLGEBRA DE BOOLE

2.1.1 <u>Definición</u>:

Álgebra de Boole o álgebra booleana se le denomina a las reglas algebráicas basadas en la teoría de conjuntos para manejar ecuaciones de lógica matemática. La lógica matemática trata de proposiciones, elementos de circuitos de dos estados, etc.; asociados por medio de operadores como Y, O, NO, EXCEPTO, SI… ENTONCES. Y que por lo tanto permite cálculos y demostraciones como cualquier parte de las matemáticas. Es llamado así en honor a George Boole, famoso matemático que la introdujo en 1847.

En otros términos se puede definir el álgebra de Boole como toda clase o conjunto de elementos que pueden tomar dos valores perfectamente diferenciados, que son designados por “0” y “1” y que están relacionados por dos operaciones binarias denominadas suma (+) y producto (**.**) lógicos que cumplen con los postulados siguientes.

2.1.2 Propiedades*:*

1. Ambas operaciones son conmutativas, es decir, si a y b son elementos del álgebra, se verifica:

$$a + b = b + a$$
$$a.b = b.a$$

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

$$0 + a = a$$
$$1.a = a$$

3. Cada operación es distributiva respecto a la otra:

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

4. Para cada elemento “a” del álgebra existe un elemento denominado ā o a’, tal que:

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

Este último postulado define realmente una operación fundamental que es la inversión o complementación de una variable. La variable “*a*” se encuentra siempre en un estado binario contrario al de $\bar{a}$.

Como complemento, se puede decir que el álgebra booleana es relativamente fácil de manejar en comparación con la ordinaria, ya que sólo pueden haber 2 valores. Aquí no hay fracciones, decimales, números negativos, raíces cuadradas, cúbicas, logaritmos o números imaginarios, etc.

2.2 OPERACIONES BÁSICAS DEL ÁLGEBRA BOOLEANA

2.2.1 <u>Suma Lógica</u>:

Llamada también operación «*O*» (*OR* en inglés). Es una operación entre dos variables lógicas *a* y *b*, representadas por el símbolo +, y definida por la siguiente tabla:

TABLA N°.2.1

FUNCIÓN LÓGICA "OR"

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

La regla general que se desprende de la Tabla es la siguiente:

$$0 + a = a$$
$$1 + a = 1$$

2.2.2 <u>Producto Lógico</u>:

Se le llama también operación «*Y*» (*AND* en inglés). Es una operación entre dos variables lógicas *a* y *b*, representadas por el símbolo (**.**), y se define por la Tabla 2.2:

TABLA N°.2.2

FUNCIÓN LÓGICA "AND"

a	***b***	***a . b***
0	0	0
0	1	0
1	0	0
1	1	1

La regla general que se desprende de la Tabla es la siguiente:

$$1 \cdot a = a$$
$$0 \cdot a = 0$$

2.2.3 <u>Complementación o Inversión Lógica</u>:

Llamada también operación «*NO*» (*NOT* en inglés). Es una operación sobre una variable lógica *a*, representada por una barra elevada (¯) o una comilla ('), y definida por la siguiente tabla:

TABLA N°.2.3

FUNCIÓN LÓGICA "NOT"

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

La regla general que se desprende de la Tabla son las siguientes:

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

Cualquier relación entre variables lógicas puede representarse por combinación de éstas tres operaciones básicas.

Se podrá ver el sentido práctico de éstas operaciones en relación al comportamiento de los contactos y relés. Para esto se tomará en cuenta el convenio que se indica a continuación en la Figura 2.1:

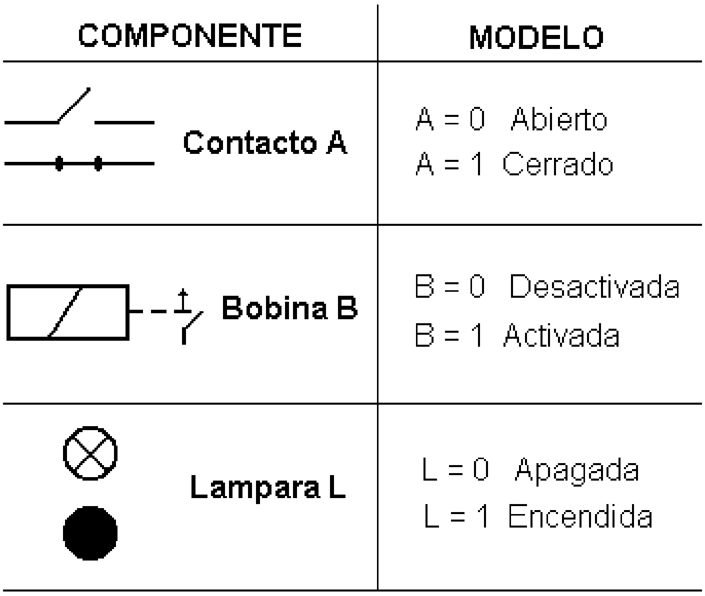


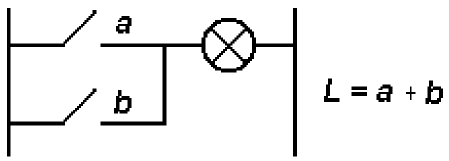
Figura 2.1. Convenios en lógica de redes.

En la Figura 2.2, se muestra un sistema tomado por la lámpara y los dos contactos en paralelo, tiene un comportamiento que resulta ser idéntico al de la Tabla de la operación suma lógica, por lo tanto, se puede escribir:

$$L = a + b$$

En forma general, se indica que colocando contactos en paralelo equivale a efectuar con ellos una operación «*O*» ó suma lógica.

Observando de la Figura 2.2 que al cerrar los dos contactos se enciende la lámpara, pero igual que lo haría uno solo, de ahí que desde un punto de vista lógico es lo mismo (1 + 0) ó (0 + 1) que (1 + 1).



CONTACTO		LAMPARA
a	*b*	*L*
abierto	abierto	apagada
abierto	cerrado	encendida
cerrado	abierto	encendida
cerrado	cerrado	encendida

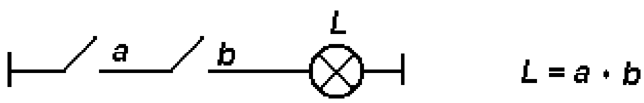
VARIABLES		
a	*b*	*L*
0	0	0
0	1	1
1	0	1
1	1	1

Figura 2.2. Operación Lógica «*O*».

A continuación se muestra en la Figura 2.3, un conjunto formado por dos contactos en serie y una lámpara, teniendo un comportamiento que resulta ser idéntico al de la Tabla de la operación de multiplicación lógica, por lo tanto se puede escribir:

$$L = a \,.\, b$$

En forma general, se puede deducir que contactos en serie equivale a efectuar con ellos una operación «*Y*» ó multiplicación lógica. Se puede observar de la figura que al cerrar los dos contactos es la única manera de que se pueda encender la lámpara.



CONTACTO		LAMPARA
a	*b*	*L*
abierto	abierto	apagada
abierto	abierto	apagada
abierto	cerrado	apagada
cerrado	cerrado	encendida

VARIABLES		
a	*b*	*L*
0	0	0
1	0	0
0	1	0
1	1	1

Figura 2.3. Operación Lógica «*Y*».

Para la esquematización de la operación «*NO*» se puede observar en la Figura 2.4, donde los contactos normalmente cerrados de un relé equivalen a efectuar dicha operación, entonces se cumple que:

$$L = \bar{a}$$

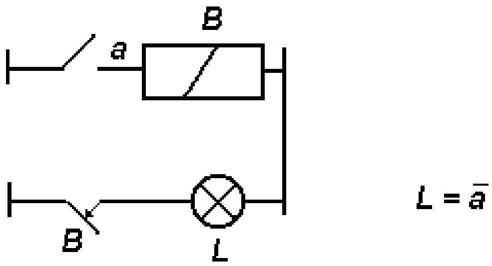


Figura 2.4. Operación Lógica «*NO*».

2.3 COMPONENTES LÓGICOS.

Son circuitos electrónicos que operan con una o más señales de entrada para producir una señal de salida. Existen señales como voltajes o corrientes eléctricas en un sistema digital en uno u otro de dos valores reconocibles. Los circuitos operados por tensión responden a dos niveles de voltajes independientes que representan una variable binaria igual a un "1" lógico ó "0" lógico. Por ejemplo, un sistema digital puede definir como el cero lógico, como una señal igual a 0 voltios y el uno lógico como una señal igual a 5 voltios.

Los símbolos para reconocer a cada una de las compuertas que identifican a las operaciones descritas en el punto anterior se ilustran a continuación.

2.3.1 Compuerta *OR*:

Es un circuito digital que tiene dos o más entradas y cuya salida es igual a la suma *OR* de las entradas. El símbolo correspondiente a una compuerta *OR* de dos entradas es el que se puede observar en la Figura 2.5:

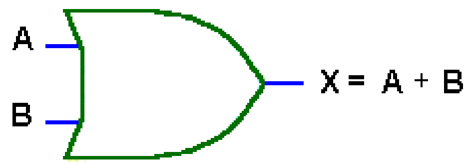


Figura 2.5. Compuerta «*OR*».

En resumen se puede destacar los aspectos más importantes de la operación *OR*:

- La operación *OR* produce un resultado de 1 cuando cualquiera de las variables de entrada es 1.
- La operación *OR* genera un resultado de 0 solamente cuando todas las variables de entrada son 0.
- En la operación *OR*, 1+1 = 1; 1+1+1=1; etc.

La función *OR* también se puede identificar de la siguiente manera:

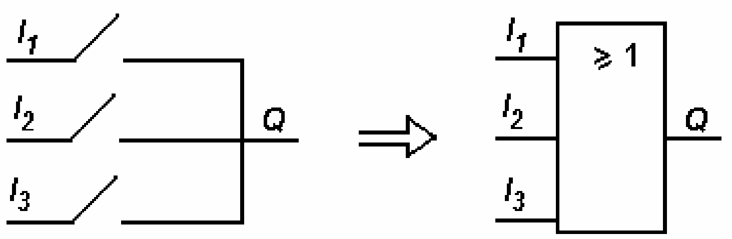


Figura 2.6. Bloque de funciones de la Operación «*OR*».

2.3.2 Compuerta *AND*:

En la Figura 2.7 se muestra en forma simbólica una compuerta *AND* de dos entradas, en este caso la salida de la compuerta *AND* es igual al producto de las entradas lógicas, es decir, $X = A.B$.

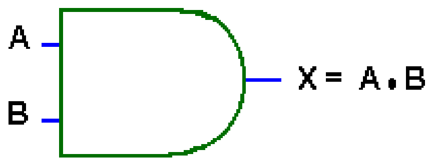


Figura 2.7. Compuerta «*AND*».

Así como la función *OR*, ésta también se puede identificar, como se muestra en la Figura 2.8:

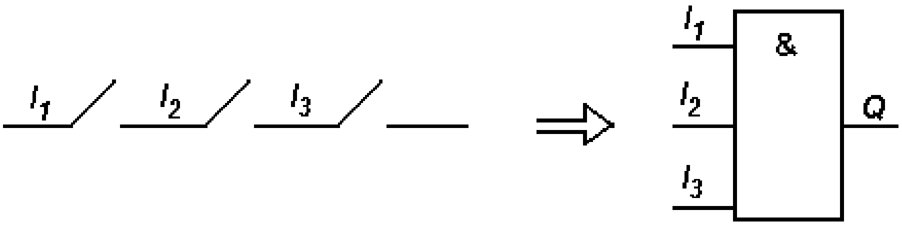


Figura 2.8. Bloque de funciones de la operación «*AND*».

2.3.3 Compuerta *NOT*:

En la Figura 2.9 se muestra en forma simbólica una compuerta *NOT*, la cual tiene una sola entrada y una salida. La salida del inversor se encuentra en estado lógico "1" sí y solo sí, la entrada se encuentra en el estado lógico "0". Esto es, que la salida toma el estado lógico opuesto al de la entrada.

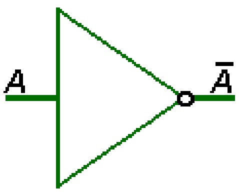


Figura 2.9. Compuerta «*NOT*».

Tal como en los casos anteriores, ésta compuerta también se puede identificar como se muestra en la Figura 2.10:

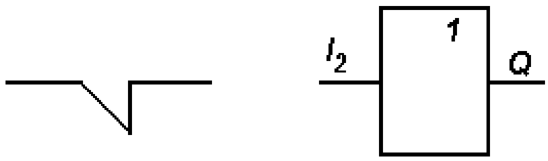


Figura 2.10. Bloque de funciones de la operación «*NOT*».

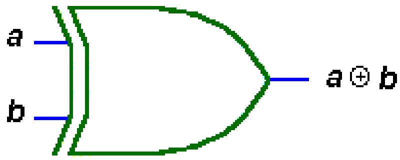
Para las anteriores compuertas se aplica la Tabla de la Verdad correspondiente a cada función, es decir, por ejemplo la función *OR* corresponde a la compuerta *OR*.

Hay otras compuertas que se derivan de las anteriores, las cuales se explicarán brevemente a continuación.

2.3.4 Compuerta *XOR*:

También llamada compuerta *OR exclusivo*, en la Figura 2.11 se puede observar su representación y su Tabla de la Verdad correspondiente.

El símbolo del operador *OR exclusivo* es la suma de un circuito $\oplus$ y está definido como: $a\oplus b = \bar{a}.b + a.\bar{b}$.



a	*b*	$a\oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Figura 2.11. Compuerta y operación *OR Exclusivo*.

La compuerta también se puede identificar de la siguiente manera:

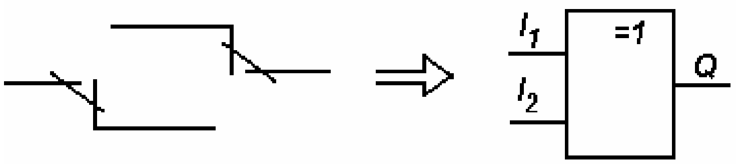


Figura 2.12. Bloques de funciones de la operación *XOR*.

2.3.5 Compuerta *NAND*:

La función u operación de la compuerta le corresponde a:

$$\overline{a.b} = \bar{a} + \bar{b}$$

De lo anterior se deduce que ésta no es más que la operación *AND* complementada, su representación se puede observar en la Figura 2.13.

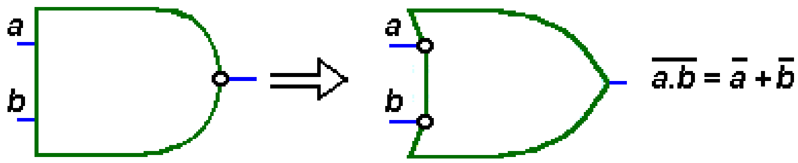


Figura 2.13. Compuertas *NAND*.

También se puede representar como se muestra en la Figura 2.14:

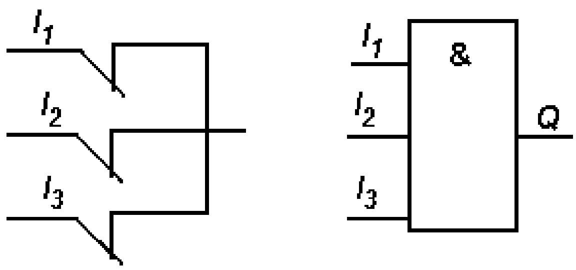


Figura 2.14. Bloques de funciones de la operación *NAND*.

2.3.6 Compuerta *NOR*:

Para esta compuerta la función u operación se establece como:

$$\overline{a+b} = \bar{a}\,.\,\bar{b}$$

De esta función se deduce que ésta representa una operación *OR* complementada, de allí que en la Figura 2.15 se puede observar su representación.

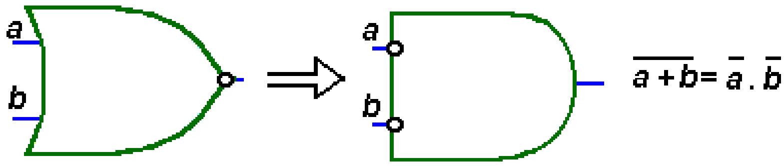


Figura 2.15. Compuertas *NOR*.

Se puede representar como a continuación se muestra en la Figura 2.16:

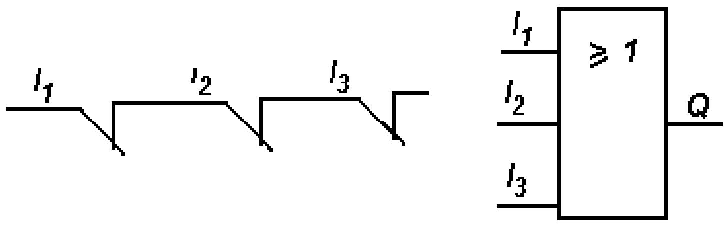


Figura 2.16. Bloques de funciones de la operación *NOR*.

2.4 TEOREMAS DEL ÁLGEBRA DE BOOLE.

Un álgebra de Boole, en virtud de las propiedades que por definición se le exigen, cumplen una serie de teoremas. Estos teoremas son de gran utilidad a la hora de transformar expresiones algebraicas de funciones lógicas en otras equivalentes.

1. **Asociatividad**:

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

2. **Absorción**:

$$a + a.b = a$$

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

3. **Idempotencia**:

$$a + a = a \quad ó \quad a + \bar{a} = 1$$
$$a.a = a \quad ó \quad a.\bar{a} = 0$$

4. **Involución**:

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

5. **Incógnita**:

$$X + 1 = 1$$
$$X.0 = 0$$

6. **Leyes de Morgan**:

$$\overline{a + b} = \bar{a} \, . \, \bar{b}$$
$$\overline{a \, . \, b} = \bar{a} + \bar{b}$$

7. **Dualidad**: Cualquier expresión válida en un álgebra de Boole continua siendo válida si se intercambian entre sí los elementos neutros (0 – 1) y las operaciones (+ ↔.).

Algunos de estos teoremas son generalizables a *n* variables, por ejemplo, las Leyes de Morgan: El complementario de una suma de variables es el producto de los complementarios de las variables, y el comportamiento de un producto de variables es igual a la suma de los complementarios de la suma.

2.5 TABLAS DE VERDAD.

Son una representación gráfica de todos los casos que se pueden dar en una relación algebraica y de sus respectivos resultados.

En cada tabla de verdad, las combinaciones posibles de niveles lógicos 0 y 1 para las entradas se enlistan del lado izquierdo y el nivel lógico resultante para la salida se enlista a la derecha. El número de combinaciones de la entrada será igual a 2^n para una tabla de verdad de n entradas.