Principios y características del mundo Digital

Jorge Luis Yanza

CONTENIDO

Prefacio

Es difícil creer que hoy en día la gente diga que la electrónica es bastante compleja de entender, aunque tienen algo de razón en eso, ya que para ello es realmente obligatorio contar con varias capacidades analíticas junto con vastos conocimientos en matemática, pero en realidad esta especialidad no es del todo imposible de comprender como para no saber diferenciar entre un artefacto digital y otro que no lo es. Lo que trato de decir es que las personas posiblemente no puedan convertirse en ingenieros profesionales de un día para otro, pero al menos pueden adquirir conocimientos básicos. El objetivo de este módulo es, por tanto, guiar a esas personas que recién escuchan y desconocen por completo el término *"electrónica digital"*.
Dentro de este documento literario se ha recopilado información, conceptos básicos e ilustraciones relacionadas a los sistemas digitales y su contraparte que vienen siendo los sistemas analógicos. Nuestra mayor prioridad será hablar sobre los dispositivos electrónicos que almacenan información digitalizada y que ya todos conocemos como los teléfonos inteligentes (*Smartphone*), tabletas, laptops, etc. La realización de este libro fue posible gracias a la información recogida de otras grandes obras, que hablan de teoría y práctica de la especialidad de electrónica, estos otros libros fueron las bases fundamentales para comenzar a escribir este texto. Por eso quiero dar un enorme y merecido agradecimiento a las siguientes obras:

-El libro titulado *"Sistemas digitales": Principios y aplicaciones (Octava edición)* de ***Ronald J. Tocci*** (Monroe Community College) y ***Neal S. Widmer*** (Purdue University)
-El libro de Equipos Electrónicos de Consumo titulado *"Electrónica Digital y Microprogramable"* de ***Fernando Blanco Flores*** y ***Santiago Olvera Peralta***

También quisiera agradecer a aquellas personas que me apoyaron al momento de realizar este proyecto:
-Un agradecimiento a los miembros de mi familia, en especial a mis padres, que me apoyaron y me brindaron todo el tiempo necesario para poder dedicarme a escribir este libro sin muchas interrupciones.

-Finalmente, quiero dar un reconocimiento especial para la Ingeniera ***Rosario Llivicura***, de *"L y L Servicios Eléctricos"*, que me motivó en todo momento, me aconsejó durante la realización de este módulo y me otorgó todos los medios necesarios para que este documento se haya vuelto realidad. Espero algún día poder compensar toda la ayuda que me ha brindado desde el principio.

El Autor Jorge Luis Yanza

CAPITULO N°1

Conceptos básicos

♦ TEMARIO:

1.1 Representación de cantidades

1.2 Definición de un sistema digital y analógico

1.3 Sistemas digitales de numeración

1.4 Representación de valores en forma binaria

1.5 Circuitos digitales y Circuitos lógicos digitales

1.6 Transmisión paralela y transmisión serial

1.7 La Memoria

1.8 Computadoras Digitales

♦OBJETIVOS:

El objetivo principal de este libro es la de proporcionar la información necesaria a todos los estudiantes, que estén iniciando en la especialidad de electrónica, para que sean capaces de:

♦ Diferenciar entre un sistema analógico y digital.

♦Conocer las ventajas de las técnicas digitales.

♦Comprender el uso de convertidores analógico-digital y viceversa.

♦ Distinguir los diferentes tipos de numeración digital.

♦Convertir un valor binario a decimal.

♦Utilizar el sistema binario para realizar conteos.

♦Mencionar las diferencias entre una transmisión paralela y serial.

♦INTRODUCCCION:

La electrónica se enfoca en cómo se comportan y se procesan las corrientes eléctricas. Es por esa razón que esta especialidad es mayormente conocida como el estudio y análisis de los circuitos lógicos que emplean dichas corrientes para lograr su funcionamiento. La electrónica en general se divide en dos partes, las cuales serán explicadas en este primer capítulo, la parte que corresponde a sistemas analógicos y la otra que tiene que ver con sistemas digitales; esta última será la más estudiada en este libro.

Si en este módulo nos vamos a concentrar principalmente en la parte digital debemos tener en cuenta que, en la actualidad, la palabra “digital” y todo lo relacionado a ella

se ha convertido en algo bastante común con respecto a todo tipo de tecnologías que se están desarrollando o que ya tenemos a disposición, tales como las computadoras, calculadoras, robótica, domótica, internet, y todo lo que nos permite transmitir información como los sistemas de telefonía, entre otros. A continuación, en este capítulo se introducirán las características y los conocimientos básicos, que son indispensables para que ustedes comiencen a hacerse una idea sobre cómo es el funcionamiento de los circuitos y las técnicas digitales.

1.1 REPRESENTACIÓN DE CANTIDADES

Dentro del campo de la tecnología se trabaja de manera constante con cantidades decimales, en binario, octal y hexadecimal, las cuales son indispensables en la mayoría de sistemas físicos.
Es importante saber cómo representar estos valores de la manera más precisa posible. Para ello en este módulo aprenderemos a representarlas de manera **analógica** y **digital**.

1.1.1 Representación de cantidades analógicas

Las cantidades analógicas son representadas mediante diferentes niveles de voltaje o por el movimiento de cualquier instrumento de medición que es proporcional al valor de esa cantidad. Por ejemplo, la carátula de una balanza, en la cual el movimiento angular que realiza la aguja representa el peso de la masa que se encuentre encima de la balanza.
Una característica relevante que presentan las cantidades analógicas es que pueden tener variaciones en su valor, es por eso que se dice que estas cantidades varían en un *rango continuo de valores*.

Figura 1.1

Balanza analógica que representa el peso mediante el movimiento de su aguja. [1]

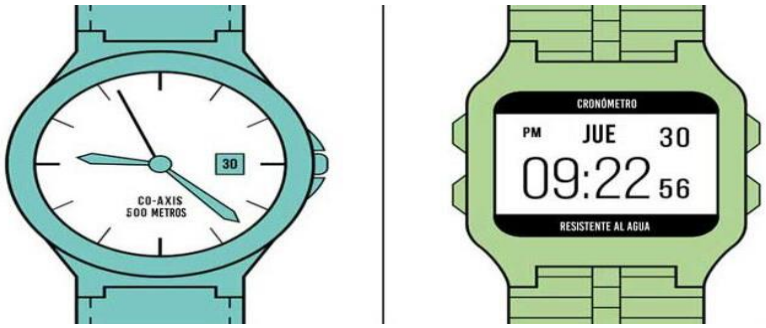
[1]https://ottoyanna.com/7471-large_01oslo/balanza-cocina-retro-5kg-verde- agua.jpg

1.1.2 Representación de cantidades digitales

En este caso las cantidades digitales *sólo pueden ser representadas con* ***dígitos*****.** Tomemos como ejemplo un reloj digital, el cual indica la hora mediante dígitos decimales que representan horas, minutos y segundos. Es un hecho que el tiempo corre continuamente, pero la lectura del reloj digital no cambia constantemente, sino más bien cambia minuto tras minuto. Dicho de otra forma, la representación digital de la hora cambia en escalones *discretos,* en comparación con un reloj analógico, en el que la lectura de su carátula cambia continuamente.

Figura 1.2

Comparación de un reloj analógico con un reloj digital. [2]



De acuerdo a todo lo dicho anteriormente, podemos deducir que existe una diferencia principal entre estos dos tipos de cantidades, esta diferencia se puede ilustrar de la siguiente manera:

Analógico = continuo

Digital = discreto

[2]http://www.marjoya.com/blog/wp-content/uploads/2016/09/reloj-anal%C3%B3gico-vs-digital.jpg

Ejercicios de repaso 1.1

1.-Clasifique los siguientes enunciados como cantidades analógicas o cantidades digitales.
a) Reloj de manecillas.
b) Medir temperatura con un termómetro de mercurio.
c) Cantidad de estrellas en el cielo.
d) Imágenes de un celular.
2.-Establezca la diferencia entre cantidades analógicas y digitales.
3.-Conteste: ¿Qué son los dígitos y para qué sirven?

1.2 DEFINICIÓN DE UN SISTEMA DIGITAL Y ANALÓGICO

-Un **sistema digital** es un conjunto de dispositivos capaces de manejar cualquier tipo información digitalizada. Casi todos estos dispositivos son electrónicos, pero también pueden ser mecánicos o magnéticos. Para realizar el análisis de los sistemas digitales se usa como herramienta principal el *álgebra de Boole* (hablaremos más de eso en el capítulo 3). Algunos de estos sistemas más comunes son las computadoras, calculadoras digitales y también el sistema telefónico, que es el sistema más grande que existe.

Figura 1.3

Calculadora científica (ejemplo de sistema digital). [3]

-Un **sistema analógico** se compone de dispositivos que manejan cantidades físicas representadas analógicamente. En este tipo sistemas las cantidades varían en un rango continuo de valores. Algunos ejemplos de estos sistemas analógicos son los multímetros analógicos (con aguja), los amplificadores de audio, entre otros.

Figura 1.4

Multímetro de aguja (ejemplo de sistema analógico). [4]

Tanto los sistemas analógicos como los sistemas digitales tienen su propia lista de aplicaciones, aunque cabe mencionar que los sistemas digitales son los que se están introduciendo más, día a día, en el campo de la electrónica analógica, para en ciertos casos mejorarla o para incluso llegar a sustituirla por completo.

[3]https://encryptedtbn0.gstatic.com/images?q=tbn:ANd9GcRqdM4yvLvzdcqnfC76-Wema0Bnln8zgDJRv-RzYxyL6yVsN9_M_g

[4]https://http2.mlstatic.com/multimetro-analogico-tekpower-con-el-valor-medio-de-la-D_NQ_NP_697512-MCO26869692376_022018-F.jpg

1.2.1 Ventajas de emplear técnicas digitales

La mayoría de las tecnologías analógicas han optado por el uso de técnicas digitales para realizar operaciones, las cuales antes eran realizadas de manera analógica. Todo este cambio se debe a que la tecnología digital nos ofrece las siguientes ventajas:

***-Son fáciles de diseñar*:**
Esto es debido a que usan circuitos de conmutación para representar el intervalo en el que se encuentre (ALTO o BAJO).

***-Facilidad al momento de almacenar información*:**
Esto se logra mediante el uso de dispositivos capaces de manipular y retener toda la información, además su capacidad de almacenamiento es extensa, lo que le permite guardar millones de datos en un espacio físico muy reducido.

-Mayor precisión:
Los sistemas digitales pueden manipular la cantidad de dígitos de precisión necesaria con simplemente adicionar más circuitos de conmutación. En cambio, en los sistemas analógicos la precisión se limita a tres o cuatro dígitos, debido a que los valores de voltajes dependen de los componentes de los circuitos que son afectados por el ruido (fluctuaciones aleatorias del voltaje).

***-La operación es programable*:**
La operación de los sistemas digitales se puede gestionar mediante programas (conjunto de instrucciones). Los sistemas analógicos también son programables, pero cabe mencionar que la complejidad de las operaciones está limitada.

***-Menor susceptibilidad al ruido*:**
Esto es debido a que el valor exacto de los voltajes no son muy relevantes, siempre y cuando el ruido no nos impida distinguir entre un ALTO (1) o un BAJO (0).

***-Mayor fabricación de circuitería digital en los chips de los circuitos integrados*:**
Los circuitos analógicos poseen una desventaja relacionada con el uso de dispositivos que no se pueden integrar económicamente (condensadores, resistencias, transformadores, entre otros). Este tipo de inconveniente hace que los sistemas analógicos se vean inferiores ante los sistemas digitales.

1.2.2 Restricciones de las técnicas digitales

La única desventaja que presentan las técnicas digitales es enuncia de la siguiente forma:

“El mundo real está mayormente compuesto de variables analógicas”
Casi todas las cantidades físicas son originalmente analógicas, y con frecuencia son gestionadas mediante sistemas de control. Algunos ejemplos más conocidos son la temperatura, presión,

velocidad, posición, etcétera. Estas cantidades las podemos expresar digitalmente, por ejemplo cuando decimos que un vehículo viaja a 15km/h, pero esto sólo es una aproximación digital que hacemos de una cantidad analógica.
Para conseguir un máximo rendimiento de las técnicas digitales cuando se dispone de entradas y salidas analógicas, se siguen estos pasos:

Paso 1. Digitalizar las entradas analógicas del mundo real.
Paso 2. Procesar la información digital.
Paso 3. Las salidas digitales deben volver a su forma analógica.

En el siguiente diagrama de bloques se presenta un sistema de control de temperatura. Aquí se muestra la temperatura analógica siendo medida y el valor que obtenemos se digitaliza gracias a un convertidor analógico-digital. Luego la circuitería procesa la cantidad digital. Su salida digital se vuelve analógica mediante un convertidor digital-analógico. Esta salida alimenta a un controlador que se encarga del ajuste de la temperatura.

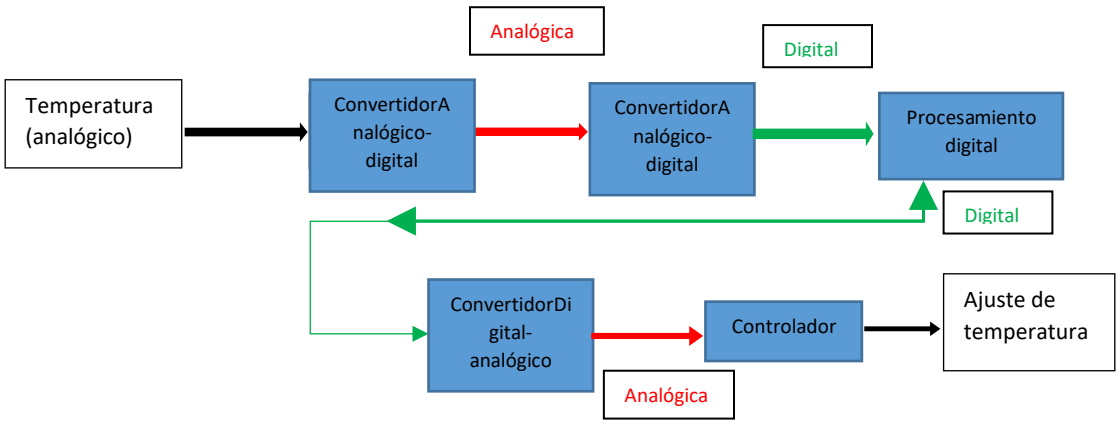


Figura 1.5

"Diagrama de bloques de un sistema de control de temperatura que requiere conversiones analógica-digital con objeto de permitir el uso de técnicas de procesamiento digital."[5]

[5] Diagrama tomado de la figura 1.1 del libro "Sistemas digitales-Principios y aplicaciones" Octava edición.

1.2.3La evolución de la tecnología digital

La tecnología digital está avanzando a pasos tan acelerados que dentro de poco se encontraran a disposición una gran variedad de complejos dispositivos que serán parte de nuestra vida diaria, y lograrán que la calidad de vida de los seres humanos mejore radicalmente. Todo esto será posible gracias a que la tecnología está destinada principalmente a mejorar aquellos aspectos que son esenciales para nuestra existencia como el campo de la medicina, la comunicación a distancia e incluso del entretenimiento personal.

En caso de que este tipo de dispositivos no estén desarrollados aún, no debemos desilusionarnos, es sólo cuestión de tiempo para que estas innovaciones sean una realidad; Hay que recordar que la tecnología evoluciona constantemente, así que solo debemos ser pacientes y observar como algunos de estos dispositivos mejoran poco a poco a nuestro alrededor.

Ejercicios de repaso 1.2

1.-Defina que es un sistema digital y un sistema analógico.
2.-Mencione 3 ventajas de las técnicas digitales sobre las analógicas.
3.-Escriba la única desventaja que presentan las técnicas digitales.

1.3 Sistemas de numeración digital

La tecnología digital siempre trabaja usando sistemas de numeración. De los cuales destacan los sistemas: decimal, binario, octal y hexadecimal. Ahora nos enfocaremos en el sistema decimal, el cual es el más utilizado por nosotros y que si hacemos un análisis de sus características seremos capaces de entender mejor a los demás sistemas.

1.3.1 Sistema de numeración decimal

Este sistema se compone de 10 numerales o dígitos que son: 0, 1, 2, 3, 4, 5, 6, 7, 8 y 9; estos dígitos nos permiten expresar distintas cantidades. También es conocido como sistema de base 10 porque consta de 10 dígitos. Además, es un sistema de valor posicional, lo que significa que el valor de un dígito dependerá de la posición en la que se encuentre (unidades, decenas o centenas).
Por ejemplo consideremos el número 748.63, este número representa 7 *centenas*, 4 *decenas y* 8 *unidades* más 6 *décimos* y 3 *centésimos*, todo esto se representa de la siguiente forma:

$$(7x100) + (4x10) + (8x1) + (6x0.1) + (3x0.01)$$

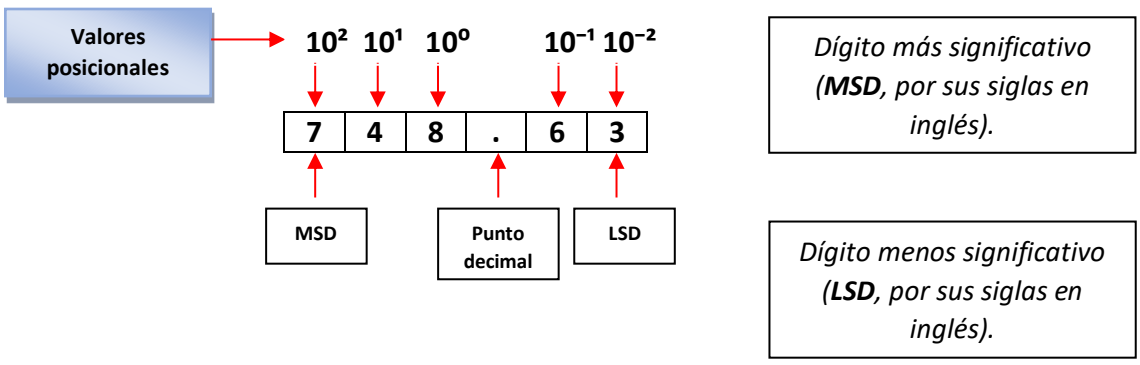
O también, se puede expresar así:

$$(7x10^{2}) + (4x10^{1}) + (8x10^{0}) + (6x10^{-1}) + (3x10^{-2})$$

En resumen, decimos que cualquier número es la suma de los productos del valor de cada dígito y su valor posicional (peso).

Figura 1.6

Valores de posición decimal como potencias de base 10.



1.3.2 Conteo en Decimal

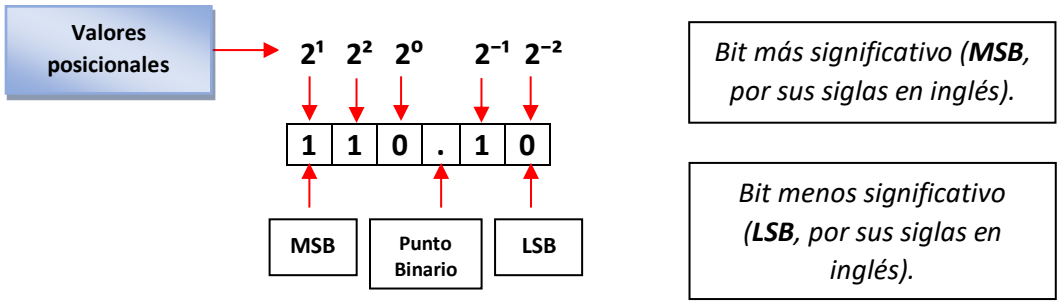
Recordemos que el sistema decimal sólo cuenta con 10 dígitos (Del 0 al 9), pero luego nos damos cuenta que no existe un dígito para el "10", ya que en realidad este es un número conformado de otros dos números juntos, el "0" y el "1".
Cuando realizamos un conteo 0, 1, 2, 3, 4, 5, 6, 7, 8, 9,… y nos toca colocar el 10 sólo es cuestión de iniciar de nuevo con "0" luego agregar un "1" a la izquierda.
Lo mismo sucede con el número "100", sólo que en este caso se inicia con ***dos*** ceros y después se le agrega un "1" a la izquierda. Este proceso también se aplica con 1000, 10000, etc.

1.3.3 Sistema de numeración binaria

En los sistemas digitales no se utiliza el sistema numérico decimal debido a la difícil tarea que sería diseñar equipos electrónicos que operen 10 valores distintos de voltaje. Es por esta razón, que en la mayoría de sistemas digitales se emplea frecuentemente la numeración en binario como sistema de numeración definitivo.
Este sistema consta solamente de dos dígitos, que son el "0" y el "1", y por ese motivo es que este sistema también es conocido como sistema de base 2. Mediante sólo esos dos símbolos uno es capaz de representar cualquier cantidad de cualquier sistema, aunque se requieren muchos dígitos binarios para expresar una cantidad específica. En este sistema se utiliza muy frecuente el término bit, que es la abreviación de "**dígito binario**" (por sus siglas en inglés).
Al igual que el sistema decimal, el sistema binario también es un sistema de valor posicional, en el que cada digito tiene su valor posicional (peso) expresado como una potencia de 2.

Figura 1.7

Valores de posición binaria como potencias de base 2.



1.3.4 Conteo en Binario

A la hora de trabajar con números binarios estaremos limitados a usar solamente un número determinado de bits. Esta restricción se basa en la circuitería usada para representar valores binarios. Por lo tanto, usaremos números binarios de cuatro dígitos para ilustrar el formato del conteo en binario.
Realizamos el *conteo en cero*, que consiste en iniciar la secuencia con todos los bits en cero. En cada conteo sucesivo la posición de las unidades se conmuta, es decir, cambia su valor binario a otro. Siempre que el bit de las unidades pasa de 1 a 0, la posición de los 2^1se conmuta. Cada vez que la posición 2^1 pasa de 1 a 0, la posición 2^2 se conmuta. De igual forma, cuando la posición 2^2 va de 1 a 0, la posición 2^3 se conmuta. Este proceso seguirá y seguirá para las demás posiciones en caso de que el número binario esté conformado de más de cuatro números.

El conteo binario posee una característica importante. Como se muestra en la siguiente figura. El bit de las unidades (LSB) cambia ya sea de 0 a 1 o viceversa por cada conteo. El segundo bit (posición 2^1) permanece en **0** durante dos conteos, luego en **1** en dos conteos, luego en **0** en dos conteos, etc. El tercer bit (posición 2^2) se mantiene en **0** en cuatro conteos, luego en **1** en cuatros conteos, etc. El cuarto bit (posición 2^3) se mantiene en **0** durante ocho conteos y luego en **1** en ocho conteos. Si deseáramos contar más, sólo debemos agregar más espacios para que este patrón continúe con los ceros y unos alternando en los grupos de 2^n-1.
Al igual que el sistema decimal, el sistema binario también es capaz de usar ***n*** bits para realizar 2^n conteos.

Figura 1.8

Secuencia de conteo binario. (Posición de los 2^0 = **LSB**)

2^3=8	2^2=4	2^1=2	2^0=1		Equivalente decimal
0	0	0	0	=	**0**
0	0	0	1	=	**1**
0	0	1	0	=	**2**
0	0	1	1	=	**3**
0	1	0	0	=	**4**
0	1	0	1	=	**5**
0	1	1	0	=	**6**
0	1	1	1	=	**7**
1	0	0	0	=	**8**
1	0	0	1	=	**9**
1	0	1	0	=	**10**
1	0	1	1	=	**11**
1	1	0	0	=	**12**
1	1	0	1	=	**13**
1	1	1	0	=	**14**
1	1	1	1	=	**15**

Ejercicios de repaso 1.3
1.-Escriba el número mayor que se puede representar con 6 bits.
2.- ¿Cuál es el equivalente decimal de 10110?
3.- ¿Cuál es el siguiente número binario que sigue a 0101 en la secuencia de conteo?

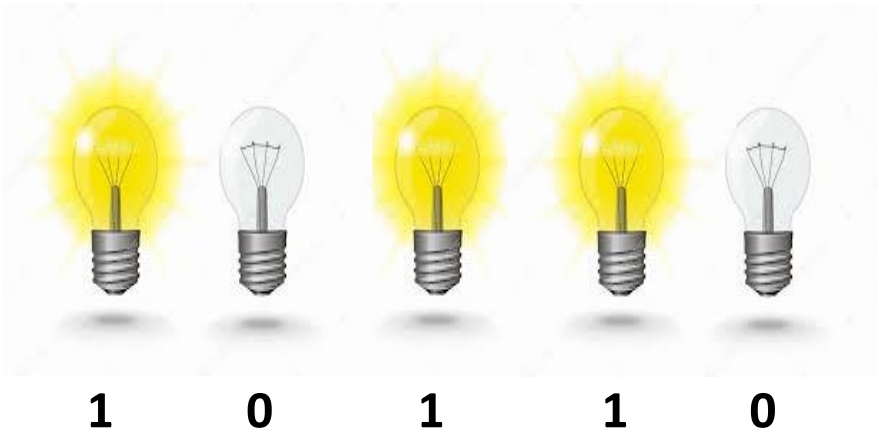
1.4 Representación de valores binarios

Por lo general, toda información digital es presentada en forma binaria. Estas cantidades se pueden representar mediante sólo 2 estados o condiciones de un dispositivo. Por ejemplo, los 2 estados de un pulsante: abierto (representa un 0 binario) y cerrado (representa un 1 binario).

En la siguiente figura se muestra otro ejemplo con los estados de un foco, donde encendido es un 1 y apagado es un 0.

Figura 1.9

Focos encendidos y apagados que representan un 1 y un 0 binario. [5]



[5]https://st2.depositphotos.com/2643781/9651/v/950/depositphotos_96519312-stock-illustration-light-bulb-onoff.jpg

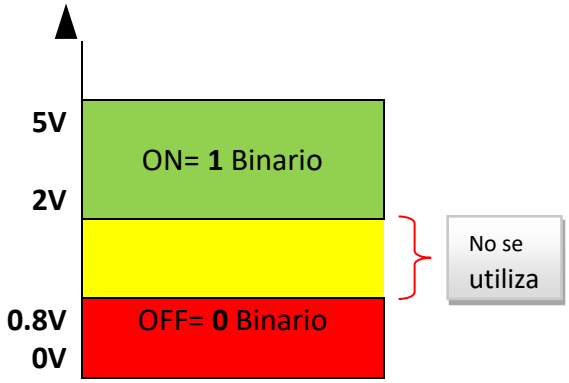
En los sistemas electrónicos, la representación es realizada con dos niveles distintos de voltaje que son 0V (0 binario) y 5V (1 binario), pero debido a que existen variaciones de corriente estos valores se representan con ***intervalos de voltaje***, en donde:

A) Un voltaje entre 0V y 0.8V representa un **0**.

B) Un voltaje entre 2V y 5V representa un **1**.

Todo esto se ilustra en la siguiente figura:

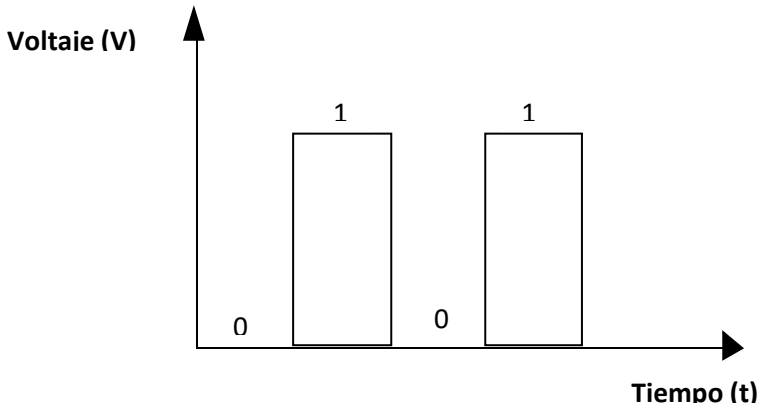
Figura 1.10 Intervalos de voltaje comunes en un sistema digital.



1.4.1 Señales digitales representadas en diagramas de temporización y sincronización

Figura 1.11

Diagrama de temporización.



La anterior figura 1.11 nos presenta un **diagrama de temporización** que representa una gráfica de voltaje (*V*) contra tiempo (*t*), en donde las transiciones son dibujadas como líneas verticales, pero esto hace la ilusión de que estas líneas son instantáneas, cuando realmente no son así.
Para mostrar transiciones más exactas es necesaria una escala de tiempo más extendida, ya que los tiempos de transición son demasiado cortos si los comparamos con los de la **figura 1.11**.
Los **diagramas de sincronización** sirven para indicar el cambio que sufren las señales digitales con respecto al tiempo, también es usada para mostrar su relación en un mismo sistema. La representación de varias señales mediante un osciloscopio (instrumento de medición de ondas) nos permite realizar una comparación, facilitándonos el proceso de detección de fallas.

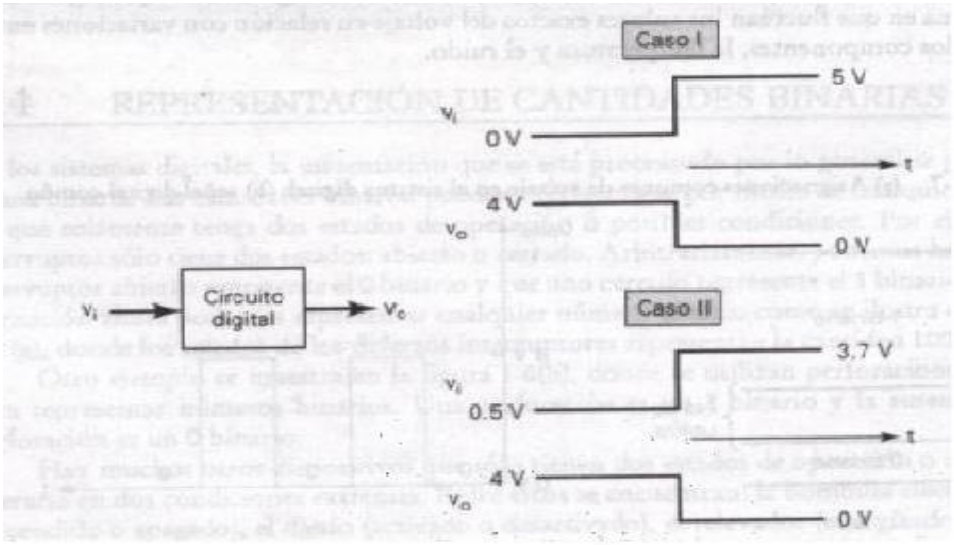
Ejercicios de repaso 1.4
1.-Cierto o Falso: la información digital es, generalmente, presentada en forma decimal.
2.- ¿Cómo es representada la información en los sistemas electrónicos?
3.-Un voltaje entre 2V y 5V representa un _____
4.- ¿Para qué sirve un diagrama de sincronización?

1.5 Circuitos digitales y circuitos lógicos digitales

Los circuitos digitales son aquellos que proporcionan voltajes de salida que puedan representar a los intervalos de voltaje prescritos: 0 y 1. De igual manera, los circuitos digitales se diseñan para dar una respuesta anticipada a voltajes de entrada que estén dentro de los intervalos permitidos.
Para explicar esto de mejor forma a continuación se presentará un circuito digital (Figura 1.11) con voltajes de E/S (entrada y salida). La salida se muestra para dos señales de onda de entrada distintas.

Figura 1.12

Circuito digital que responde a un nivel binario de entrada y no a su voltaje real. [6]



[6]http://4.bp.blogspot.com/_RzSxi31-B6U/TLYda1v9DJI/AAAAAAAAABY/gL5Hb-qC-gE/s1600/hh.JPG

1.5.1 Circuitos lógicos

Llamamos ***lógica*** del circuito a la manera en que reacciona un circuito digital ante una entrada. Existen reglas lógicas para cada tipo de circuitos digitales, por esa razón también son conocidos como ***circuitos lógicos***. Estos dos términos son fundamentales, pero por ahora nuestro principal enfoque son las distintas operaciones lógicas.

1.5.2 Circuitos integrados (CI.)

Son los más utilizados y su alta velocidad ha permitido el desarrollo de complejos sistemas digitales con espacios físicos pequeños, además son más confiables que sus contrapartes que usan componentes discretos.

Se han fabricado distintos CI, de los cuales destacan las siguientes familias:

-TTL: Sus siglas en español significan "*Lógica de transistor-transistor"*, esta familia usa el *transistor bipolar* (*"dispositivo electrónico de estado sólido consistente en dos uniones PN muy cercanas entre sí, que permite aumentar la corriente y disminuir el voltaje, además de controlar el paso de la corriente a través de sus terminales."*[A]) como principal elemento del circuito.

[A]https://es.m.wikipedia.org/wiki/Transistor_de_unión_bipolar

-CMOS: Sus siglas en español significan "*Semiconductor complementario de óxido metálico"*, esta familia usa el *MOSFET* (*"transistor utilizado para amplificar conmutar señales electrónicas."*[B]) de modo de enriquecimiento como elemento principal.

[B]https://es.m.wikipedia.org/wiki/Transistor_de_efecto_de_campo_metal-óxido-semiconductor

Una vez que ya reconocemos los tipos de CI, proseguiremos a estudiar acerca de las características de las diferentes tecnologías de CI.

Ejercicios de repaso 1.5
1.- ¿Qué es la lógica de un circuito?
2.-Los circuitos digitales también son conocidos como circuitos_____
3.-Anote las dos familias de circuitos integrados más comunes.

1.6 Transmisión paralela y transmisión serial

La principal operación de un sistema digital es la transmisión de información de un punto remoto a otro. La información se puede transmitir a distancias cortas y largas, por ejemplo cuando varios usuarios, de distintas zonas geográficas, se comunican entre sí utilizando sus computadoras con acceso a internet. La información se transmite en binario y se representa como un voltaje en las salidas de un dispositivo emisor conectadas a las entradas de un dispositivo receptor. En las siguientes figuras se muestra la comparación entre una transmisión en serie y paralelo de información entre un circuito emisor con un circuito receptor.

Figura 1.13

En la transmisión paralela se usa una línea de conexión por cada bit (en este caso se usan 8 líneas de conexión), permitiendo que todos los bits puedan transmitirse simultáneamente.

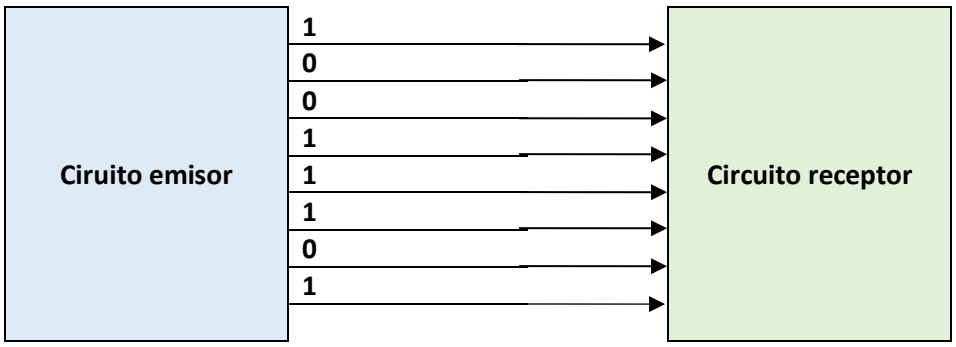
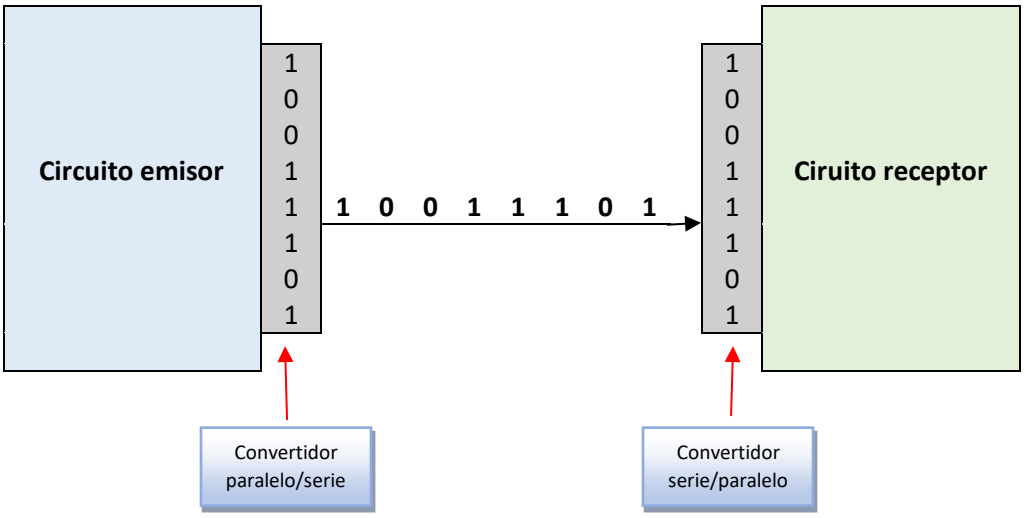


Figura 1.14

En la transmisión serial se usa una sola línea de conexión, esto significa que los bits se transmitirán uno a la vez.



A partir de esta comparación, podemos definir que la transmisión en paralelo es más rápida que la serial, ya que en la transmisión en paralelo todos los bits viajan al mismo tiempo (lo que significa menos tiempo de transmisión), pero para ello se requiere conectar varias líneas de conexión entre el emisor y el receptor.

Ejercicios de repaso 1.6
1.-Establezca la diferencia entre la transmisión paralela y serial.
2.-Escriba la ventaja de usar la transmisión en paralelo.
3.-Mencione la pricipal característica de la transmisión en serie.

1.7 La Memoria

Llamamos ***memoria*** a la capacidad de retener la respuesta que tiene un cierto dispositivo o circuito ante una entrada momentánea.
La memoria es fundamental en los sistemas digitales, ya que proporcionan una forma de almacenamiento binario, con la capacidad de modificar la información almacenada en cualquier momento que se desee.

1.8 Computadoras Digitales

Hoy en día, todo el mundo sabe lo que es una computadora, pero es muy propable que nadie tenga idea de como funciona exactamente una computadora. Explicado en términos más simples: *"una computadora es un dispositivo electrónico que tiene la capacidad de manejar, procesar y acumular datos en forma binaria."*
Nosotros los seres humanos somos capaces de realizar cualquier cosa que haga una computadora, pero con la clara diferencia de que las computadoras son mucho más rápidas y exactas que los humanos, aunque para ello las computadoras requieren que se les proporcione un exuberante conjunto de instrucciones que son las encargadas de decirle qué hacer en cada paso de su operación. A este conjunto de operaciones se lo conoce como *programa*, el cual es creado por una o varias personas para cada tarea que deba realizar la computadora.
Los programas son agregados a la unidad de memoria de la computadora codificada en binario y cada orden tiene un código único. La computadora toma estos códigos de la memoria *uno por uno* y realiza la operación que requiera el código.

1.8.1 Partes de una computadora

Todos los sistemas de cómputo se dividen en las mismas unidades funcionales. Cada unidad realiza funciones determinadas y todas funcionan en conjunto para llevar a cabo las instrucciones que proporciona el programa.

-Las funciones principales de cada unidad son las siguientes:

1ºUnidad de entrada:
Son aquellos dispositivos que ingresan la información a la unidad de procesamiento. Por ejemplo la información puede ingresar mediante un teclado, el mouse o ratón, el micrófono, etc.

2ºUnidad de memoria:
En ella se almacenan, de manera temporal, todas las instrucciones de un programa y los datos de E/S (entrada y salida) que éste maneje. También es llamada *memoria principal* o solamente *memoria*.

3ºUnidad de control:
Esta es la parte que realmente hace que las cosas ocurran. Esta unidad es capaz de emitir órdenes externas a la unidad central de procesamiento (CPU) para posteriormente realizar el intercambio de datos con la memoria y la unidad de E/S.

4ºUnidad Aritmética Lógica (ALU):
Se trata de un contador digital que realiza operaciones lógicas y aritméticas entre los datos de un circuito; suma, resta, multiplicación y división, así como también establece comparaciones lógicas usando condicionales lógicas (“si”, “no” y “o”).

5ºUnidad de salida:
Esta unidad se encarga de tomar datos pertenecientes de la unidad de memoria para posteriormente imprimir o presentar la información al operador.

1.8.2 Unidad Central de Procesamiento o CPU

Es el componente fundamental de la computadora, capaz de procesar la información y proporcionar los puertos necesarios para instalar los distintos periféricos (dispositivos de E/S). Sus prestaciones internas y sus conexiones son dos de sus características más sobresalientes.
En el interior de la CPU se encuentra toda la circuitería para buscar, cargar y realizar instrucciones para ejecutar las operaciones que se requieren mediante las órdenes que proporcionan los programas.

TIPOS DE COMPUTADORAS:
Todas están conformadas por unidades básicas, pero difieren según el tamaño físico, la velocidad, la capacidad de almacenamiento, entre otros aspectos. Las computadoras suelen clasificarse, de la más pequeña a la más grande, pero sus tres clasificaciones más básicas son: *microcomputadora, minicomputadora y mainframe*.
La evolución de las computadoras ha provocado muchas confusiones a la hora de querer distinguir entre microcomputadoras y minicomputadoras, haciendo que se comience a distinguir sólo computadoras pequeñas (de oficina o escritorio), y grandes que poseen un espacio físico muy grande, volviendo imposible poder ubicarlas en una oficina o espacio que no sea lo suficientemente amplio.

Ahora nos centraremos en el estudio de la microcomputadoras:

Las **microcomputadoras** son el tipo de computadora más pequeño, generalmente cuentan con varios CIs, incluyendo un **microprocesador**, memoria y circuitos de interfaz de E/S, junto con dispositivos de E/S como el teclado, el micrófono, los parlantes, la impresora, etc. Estas microcomputadoras se desarrollaron como el producto de los avances de la tecnología de fabricación de los CIs, que permitieron la encapsulación de circuitos digitales en pequeños chips. Por ejemplo: *los chips de memoria* que pueden retener programas de manera temporal y permanente. Otro ejemplo es el *chip microprocesador*, que resumido en pocas palabras es una CPU dentro de un chip. También existe otro tipo de microcomputadora denominada **microcontrolador,** que es un circuito diseñado para ser utilizado para diversas aplicaciones, debido a que tiene la propiedad de ser *programable*. Además están compuestas por una CPU, memorias y por periféricos.

Algunas de las aplicaciones de los microcontroladores son: calculadoras, juegos, alarmas, robots, cerraduras electrónicas, entre muchos otros. Así que pueden dejar correr su creatividad a la hora de trabajar con ellos para realizar todo tipo de proyectos.

Ejercicios de repaso 1.7 y 1.8

1.-Explique qué es una memoria
2.-Defina qué es una computadora
3.-Anote las cinco unidades principales de la computadora
4.- ¿Qué significan las siglas de CPU?

SOLUCIONARIO

Ejercicios de repaso 1.1

1.-

a) Analógica

b) Analógica

c) Digital, puesto que el número de estrellas sólo puede tener determinados valores discretos, y no cualquier valor posible sobre un rango continuo.

d) Digital

2.- Analógicas: se representan mediante un voltaje

Digital: se representan mediante dígitos

3.- Son símbolos que sirven para representar cantidades digitales

Ejercicios de repaso 1.2

1.- Un sistema digital es un conjunto de dispositivos capaces de manejar todo tipo de información digital.

Un sistema analógico se compone de dispositivos que manipulan cantidades físicas representadas analógicamente.

2.- Son fáciles de diseñar.

Almacenamiento sencillo.

Mayor precisión.

3.- El mundo real está compuesto de variables analógicas

Ejercicios de repaso 1.3

1.- $2^n-1 = 2^6 - 1 = 63_{10} = 111111_2$

2.- $(1x2^4) + (0) + (1x2^2) + (1x2^1) + (0) = 22$

3.-0101 = 5 0110 = 6

Ejercicios de repaso 1.4

1.-Falso.

2.- En los sistemas electrónicos, la representación es realizada con 2 niveles distintos de voltaje que son 0V que indican un cero y 5V que indican un uno.

3.- uno.

4.-Los diagramas de sincronización sirven para indicar el cambio que tienen las señales digitales con respecto al tiempo y también para mostrar su relación en un mismo sistema.

Ejercicios de repaso 1.5

1.- Llamamos lógica del circuito a la manera en que reacciona un circuito digital ante una entrada.

2.- lógicos

3.- TTL y CMOS

Ejercicios de repaso 1.6

1.-En la transmisión paralela todos los bits pueden transmitirse simultáneamente.

En la transmisión serial los bits se transmiten uno a la vez.

2.-la transmisión en paralelo es más rápida que la serial, ya que la transmisión en paralelo todos los bits se transmiten simultáneamente.

3.-la transmisión serial usa una sola línea de conexión.

Ejercicios de repaso 1.7 y 1.8

1.- Llamamos memoria a la capacidad de retención de la respuesta que tiene un dispositivo o circuito ante una entrada momentánea.

2.- una computadora es un dispositivo electrónico que tiene la capacidad de manejar, procesar y acumular datos en forma binaria.

3.-

Unidad de entrada.

Unidad de memoria.

Unidad de control.

Unidad Aritmética Lógica (ALU).

Unidad de salida.

4.-Unidad Central de Procesamiento

CAPITULO N°2

Sistemas de numeración y Códigos

♦ TEMARIO:

2.1 Conversión de binario a un número decimal
2.2 Conversión de un numero decimal a binario
2.3 Sistema de numeración Octal
2.4 Sistema de numeración Hexadecimal
2.5Código BCD
2.6 Integración de los sistemas
2.7 El byte
2.8Códigos alfanuméricos
2.9Método paridad para la detección de fallas

♦OBJETIVOS:

El objetivo principal de este libro es la de proporcionar la información necesaria a todos los estudiantes, que estén iniciando en la especialidad de electrónica, para que sean capaces de:

♦ Convertir cualquier número de un sistema numérico a su equivalente en cualquier otro sistema.

♦Conocer las ventajas del sistema Octal y hexadecimal.

♦Contar en forma Octal y Hexadecimal.

♦ Representar valores decimales mediante el uso del código BCD.

♦Comentar acerca de las ventajas y desventajas del uso de BCD.

♦Establecer la diferencia entre binario directo y BCD.

♦Comprender el objetivo de los códigos alfanuméricos (código ANSII).

♦Definir el método de paridad para la detección de fallas.

♦Determinar el bit de paridad que debe ser agregado a una serie de datos digitalizados.

♦INTRODUCCCION:

Puede que el sistema binario sea el sistema más relevante para los sistemas digitales, pero existen otros sistemas que también son importantes. La gran importancia del sistema decimal radica en que es el sistema más utilizado alrededor del mundo para representar varias cantidades que no tienen ninguna relación con los sistemas digitales. Esto implica que existirán distintas situaciones en las que

debemos convertir valores decimales a valores binarios antes de que estos entren a un dispositivo digital.

Por ejemplo, cuando se presiona algún un número decimal del control de una televisión la circuitería interna del dispositivo transforma el número decimal a un valor en binario.

Igualmente, existirán situaciones en las que los valores binarios de las salidas de algún sistema digital deban ser transformados en valores decimales para su respectiva interpretación al mundo exterior. Por ejemplo, la computadora utiliza cantidades binarias para procesar las repuestas a uno o varios problemas y luego convierte las respuestas a un valor decimal antes de manifestarlos visualmente.

Dejando a un lado el sistema binario y decimal, hay que mencionar que existen otros dos sistemas de numeración que tienen una larga lista de aplicaciones dentro de los sistemas digitales, los cuales son: el sistema numérico "Octal" (conocido también como sistema de base 8) y el sistema "Hexadecimal" (conocido también como sistema de base 16) estos dos últimos son usados para el mismo propósito, ya que ambos tienen la ventaja de convertirse fácilmente a su equivalente binario.

En los sistemas digitales se pueden usar simultáneamente tres o cuatro de estos sistemas de numeración, por lo que la comprensión sobre la operación del sistema requiere la capacidad para convertir de un sistema numérico a otro. En este segundo capítulo veremos explicaciones sobre cómo realizar estas conversiones. Sin embargo, alguna de estas

conversiones no serán de uso instantáneo en el estudio de los sistemas digitales, todo esto lo necesitarán cuando inicien en el estudio de los microprocesadores.

2.1 Conversión de binario a un número decimal

Como ya es sabido, el sistema binario es un sistema de valor posicional, en donde cada bit soporta un determinado peso relacionado a la posición en la que se encuentre. Todo valor binario puede convertirse en su equivalente decimal con simplemente sumar los pesos de las diferentes posiciones en el número binario que contenga un "1". A continuación, se ilustrará un ejemplo, para ello consideraremos el número110010_2, el cual convertiremos a su equivalente decimal.

$$\begin{array}{cccccccccccccc} \mathbf{1} & & \mathbf{1} & & \mathbf{0} & & \mathbf{0} & & \mathbf{1} & & \mathbf{0} & & & \\ 2^5 & + & 2^4 & + & 2^3 & + & 2^2 & + & 2^1 & + & 2^0 & = & (32+16+0+0+2+0) & = & 50_{10} \end{array}$$

Como se ve, el procedimiento consiste en encontrar los pesos de cada posición del bit que contenga un uno, y después se suman. Asimismo, observe que el MSB posee un peso de 2^5, a pesar de que es el sexto bit, esto se debe a que el LSB siempre será 2^0.

Ejercicios de repaso 2.1

1.-Realice la conversión de 10110001_2 a su equivalente decimal.
2.-Explique cómo un valor binario se convierte en decimal.
3.- ¿Cuál sería el peso del LSB y del MSB de un número que contenga 20 bits?

2.2 Conversión de un numero decimal a binario

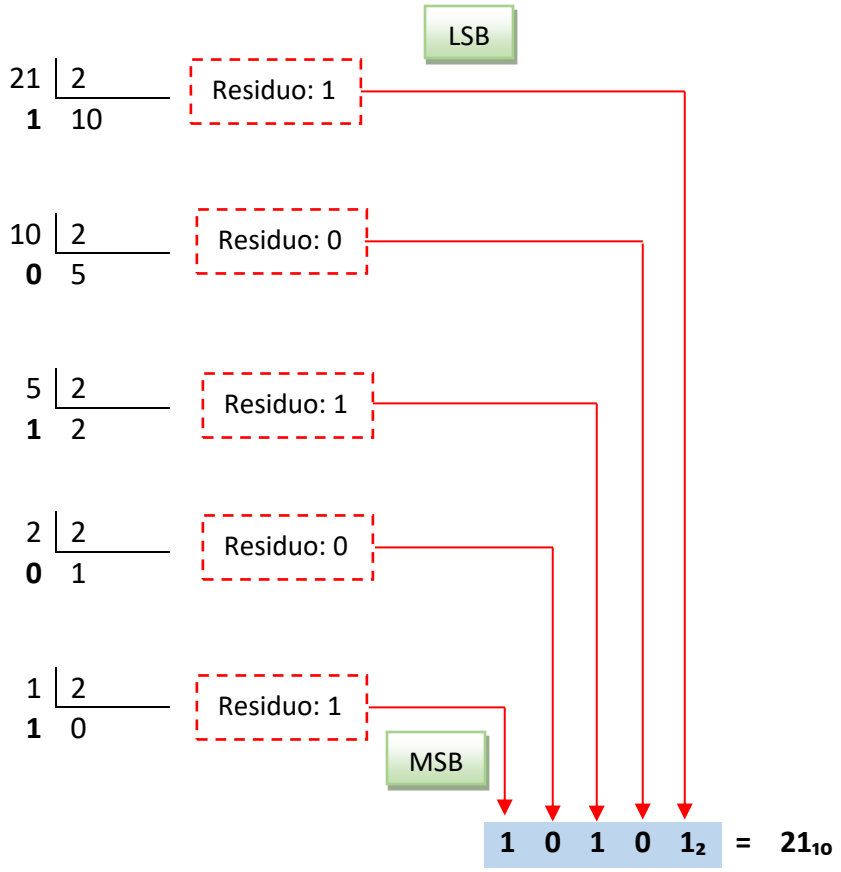
Existen dos métodos para convertir un número decimal *entero* a su equivalente binario. Ahora proseguiremos a explicar cada uno. El primer método consiste en la inversa del anterior proceso demostrado en el tema 2.1. El número decimal se expresa como una suma de pesos o potencias de base 2, y posteriormente se escriben los "1" y los "0" en las posiciones correctas del bit. Todo lo dicho se ilustra de la siguiente forma:

$$\begin{array}{lcccccccccccccccc} 70_{10} & = & 64 & + & 4 & + & 2 & = & 2^6 & + & 0 & + & 0 & + & 0 & + & 2^2 & + & 2^1 & + & 0 \\ & & & & & & & = & 1 & & 0 & & 0 & & 0 & & 1 & & 1 & & 0_2 \end{array}$$

Fíjese que se coloca un cero en las posiciones 2^5, 2^4, 2^3 y 2^0, ya que se deben tener en cuenta todos los pesos.
Una vez que haya quedado claro todo acerca del primer método, proseguiremos a explicar cómo funciona el segundo método:

División repetida:

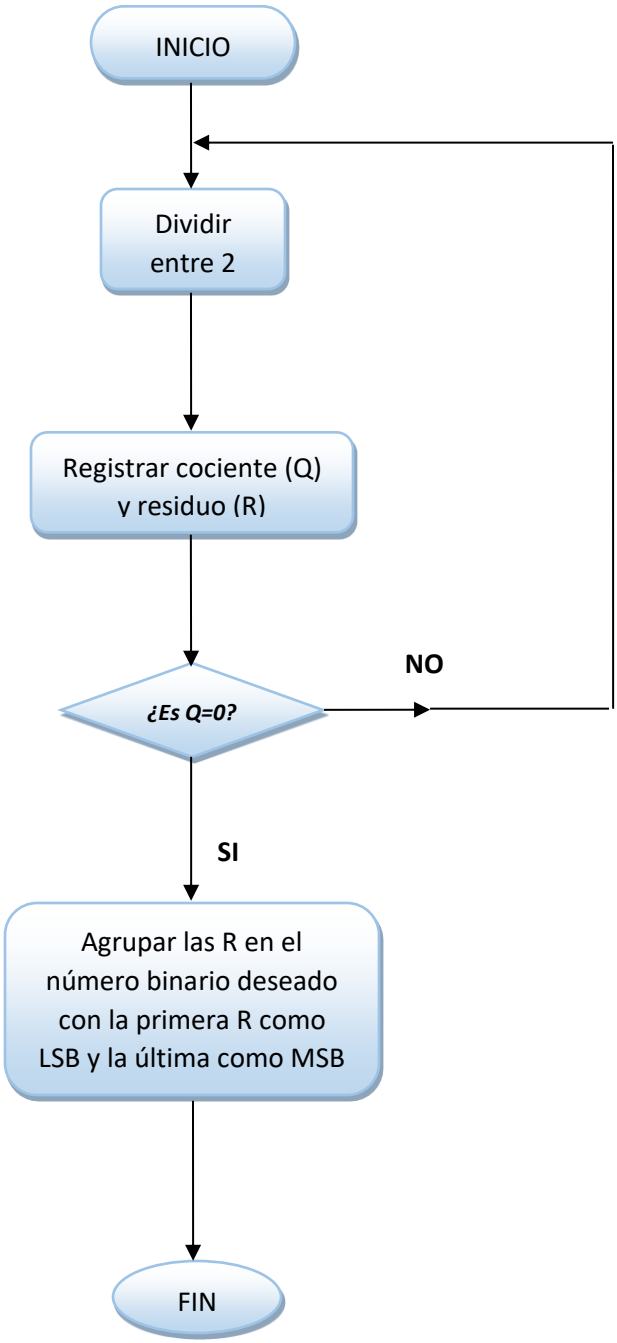
Este método consiste en dividir el número decimal entero para 2 repetidamente. El siguiente ejemplo muestra la conversión del número 21_{10}, la cual dicha conversión requiere dividir el número decimal para 2 y anotar el residuo luego de cada división hasta que el cociente llegue a cero. Fíjese que el resultado binario se obtiene escribiendo el primer residuo como el LSB y después el último residuo como el MSB.



Este otro proceso, que se presenta en la siguiente figura 2.1, también se puede usar para realizar la conversión de un número decimal a otro sistema numérico.

Figura 2.1

"Diagrama de flujo para el método de división repetida en la conversión decimal a binario con dos números enteros. El mismo proceso se puede usar para convertir un entero decimal a cualquier otro sistema numérico". [1]



[1] Diagrama tomado de la figura 2.1 del libro "Sistemas digitales-Principios y aplicaciones" Octava edición.

Ejercicios de repaso 2.2

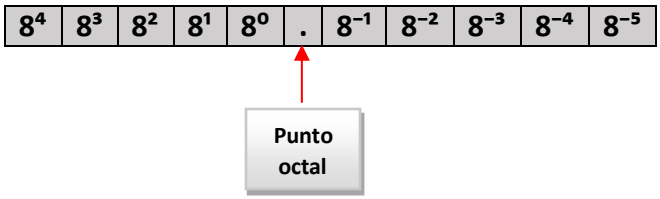
1.- ¿Cuáles son los dos métodos que existen para convertir un número decimal a su equivalente decimal?

2.- Explique en qué consiste el método de división repetida.

3.- Convierta el número 47_{10} de decimal a su equivalente binario.

2.3 Sistema de numeración Octal

El sistema octal (base 8) es usado mayormente para el trabajo con computadoras digitales. Este sistema es de base 8, por lo tanto significa que consta de ocho dígitos (del 0 al 7), en el que cada dígito de un valor octal puede ser cualquier número del 0 al 7. Los pesos de un número octal se muestran a continuación:



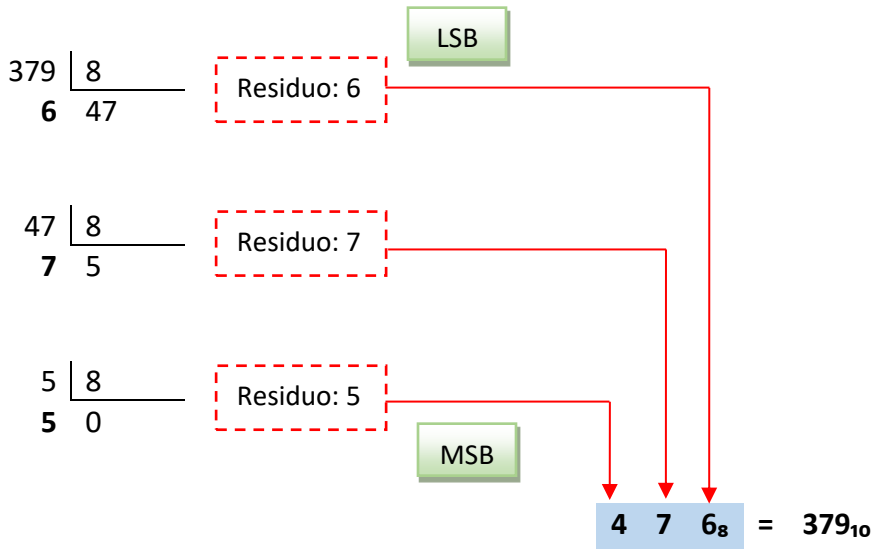
2.3.1 Conversión de un valor octal a decimal

Todo número octal se convierte a su equivalente decimal con simplemente multiplicar cada dígito octal por su peso. Un ejemplo:

$$255.82_8 = (2x8^2) + (5x8^1) + (5x8^0) + (8x8^{-1}) + (2x8^{-2})$$
$$= (128) + (40) + (5) + (1) + (0.03)$$
$$= 174.03_{10}$$

2.3.2Conversión de un valor decimal a octal

Un número decimal *entero* se convierte a su equivalente octal mediante el mismo método de la división repetida pero con la única diferencia de que ahora el factor de división será de 8 y no de 2. Por ejemplo consideraremos al número 379_{10} para ilustrar todo lo dicho anteriormente:



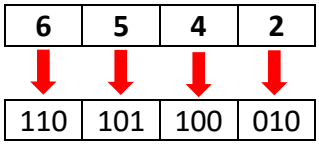
-Fíjense que el primer residuo se vuelve el LSB y el último residuo se vuelve el MSB de todo el número octal.

2.3.3 Conversión de un valor octal a binario

La ventaja de usar el sistema octal es la facilidad al momento de realizar las conversiones entre números binarios y números octales. El proceso de conversión de un numero decimal a octal consiste en convertir ***cada*** digito octal a su equivalente binario conformado de tres dígitos. Para ilustrar lo anterior:

Digito octal	0	1	2	3	4	5	6	7
Equivalente binario	000	001	010	011	100	101	110	111

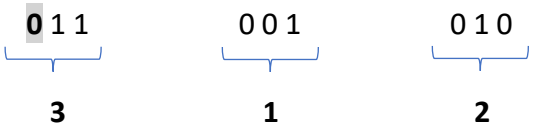
Mediante estas conversiones podemos convertir cualquier número octal a binario transformando cada dígito, como ejemplo convertiremos el número decimal 6542_8 a binario:



Por lo tanto, el equivalente binario de 6542_8 es 110101100010_2.

2.3.4 Conversión de un valor binario a octal

Para realizar esta conversión simplemente tenemos que aplicar el proceso anterior, pero a la inversa. Este proceso se ilustra en el siguiente ejemplo en donde realizaremos la conversión de 11001010_8, y para ello agrupamos los bits del número binario en grupos de tres bits, después convertimos cada grupo en su equivalente decimal.



Observen que se ha colocado un cero a la izquierda para completar los grupos de tres bits, debido a que existirán momentos en el que el número binario no disponga de todos los grupos de tres bits completos, para esos casos existe la opción de adicionar los ceros que sean necesarios a la izquierda del MSB.

2.3.5 Conteo en octal

En el conteo: 0, 1, 2, 3, 4, 5, 6 y 7; el mayor dígito es el "7", por lo tanto, cuando se realiza el conteo se aumenta una posición de un dígito hacía arriba de 0 a 7. En el momento en el que se llega a 7, se recicla a 0 en el siguiente conteo y esto provoca que la posición mayor del dígito aumente. Para ilustrar lo anterior se muestra la siguiente secuencia de conteo octal:

a)…, 65, 66, 67, 68, 69, 70, 71,…

b)…, 275, 276, 277, 300,…

-Con ***n*** posiciones podemos contar de cero a 8^n- 1, para obtener un total de 8^n conteos.

Ejercicios de repaso 2.3

1.-Convierta el número octal 556_8 a decimal.
2.- Convierta el número decimal 120_{10} a octal.
3.- Convierta el número octal 125_8 a binario.
4.- Convierta el numero binario 1001101_2 a octal.

2.4 Sistema de numeración Hexadecimal

En este sistema se emplea la base 16, lo que significa que consta de 16 dígitos, los cuales son los números del 0 al 9 más las letras mayúsculas: A, B, C, D, E y F. En la siguiente tabla se muestra la relación que existe entre los sistemas hexadecimal, decimal y binario, en donde los dígitos de la "A" hasta la "F" representan los valores del 10 al 15.

HEXADECIMAL	*DECIMAL*	*BINARIO*
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

2.4.1 Conversión de un valor hexadecimal a decimal

Para realizar esta conversión partimos de cada peso de los dígitos que son de base 16, en donde comenzamos con el LSD que es 16^o. Veamos el siguiente ejemplo para entender mejor este procedimiento:

$$6BD_{16} = (6x16^2) + (11x16^1) + (13x16^o)$$

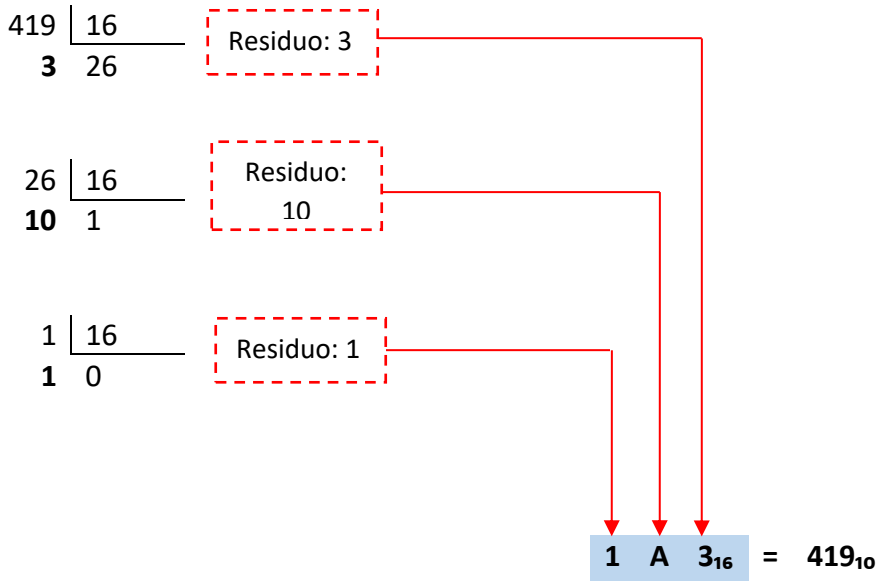
$$= (1536) + (176) + (13)$$

$$= 1725_{10}$$

-Noten que, durante el proceso de conversión, el valor de 11 fue reemplazado por “B” y el valor de 13 fue reemplazado por “D”.

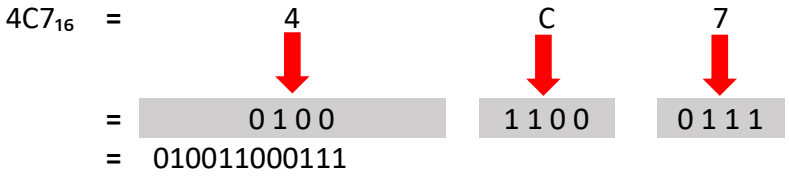
2.4.2 Conversión de un valor decimal a hexadecimal

Al igual que en las anteriores conversiones, esta conversión también emplea el método de la división repetida para convertir un valor decimal a hexadecimal. El proceso se ilustra en el siguiente ejemplo, donde transformaremos al número decimal 419_{10} a ***hex*** (su valor equivalente en el sistema hexadecimal):



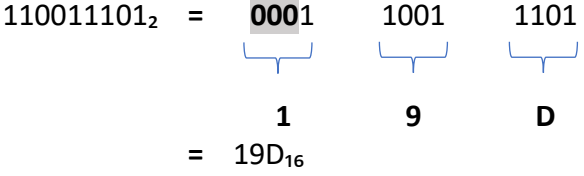
2.4.3 Conversión de un valor hexadecimal a binario

El procedimiento es relativamente fácil, solo es cuestión de convertir cada dígito hex a su equivalente binario conformado de cuatro dígitos. El siguiente ejemplo ilustra el proceso:



2.4.4 Conversión de un valor binario a hexadecimal

Esta conversión, no es nada más ni nada menos, que la inversa del proceso explicado anteriormente. El valor binario se agrupa en conjuntos de cuatro dígitos y cada conjunto se transforma en ***hex***. Para ilustrar lo dicho anteriormente se muestra el siguiente ejemplo:



-Noten los ceros que se han agregado, los cuales son necesarios para completar los conjuntos de cuatro bits.

2.4.5 Conteo en forma hexadecimal

Al realizar el conteo en hexadecimal cada posición de los dígitos puede aumentar de 0 a F. En el momento en el que un dígito alcanza el valor F, se vuelve a comenzar de cero y se le aumenta la siguiente posición del dígito. Las siguientes secuencias hex ilustran todo lo dicho anteriormente:

a)…, 66, 67, 68, 69, 6A, 6B, 6C, 6D, 6E, 6F, 70, 71, 72…
b)..., 3F7, 3F8, 3F9, 3FA, 3FB, 3FC, 3FD, 3FE, 3FF, 400…

2.4.6 Usos del sistema hex y sistema octal

Ambos sistemas se utilizan en los sistemas digitales como una especie de *taquigrafía* para representar series de bits. Cuando se realizan trabajos de cómputo casualmente las series pueden ser de hasta 64 bits. Estas series binarias no siempre presentan valores numéricas, estas también pueden presentar códigos con información que no es nada numérica. Se ha visto anteriormente que al momento de tratar con una gran cantidad de bits, es preferible escribir los números en binario, en hex u octal para así facilitarnos el trabajo. Sin embargo, deben recordar que todos los sistemas digitales trabajan principalmente con valores binarios, por lo tanto los números hex y octal son solamente usados por conveniencia de ciertos usuarios.

En resumen: Después de haber estudiado acerca de las distintas conversiones, sus mentes deben estar algo revueltas debido a las diferentes características que posee cada conversión, las cuales hacen que cada método y proceso de conversión sea único. Probablemente aún no comprendan estas conversiones, pero es necesario que las dominen lo más pronto posible, es por eso que se ha hecho el siguiente resumen, recopilando todo lo visto hasta ahora en este segundo capítulo, aunque creo que es más recomendable que practiquen hasta dominar todas estas conversiones, ya que recordemos que la práctica hace al maestro:

-Al momento de realizar la conversión de un número hex, octal o binario a su equivalente decimal, empleen el método de tomar la suma ponderada (peso) de cada posición del digito.

-Al momento de realizar la conversión de un número decimal a binario, octal o hex, empleen el método de la división repetida.

-Al momento de realizar la conversión de un número binario a octal o hex, agrupen los bits en conjuntos de tres (para sacar el equivalente octal) o cuatro (para sacar el equivalente hex) y transformen cada agrupación al dígito apropiado octal o hex.

-Al momento de realizar la conversión de un número octal o hex a su equivalente binario cambien cada dígito a su equivalente de tres bits (para octal) o cuatro bits (para hex).

Ejercicios de repaso 2.4

1.- Convierta 112_{10} a hex.
2.-Convierta $8A5_{16}$ a binario
3.- Convierta $2D4_{16}$ a octal.

2.5 Código BCD

Un ***código*** es un conjunto de símbolos representados por números, letras o palabras. Un ejemplo ya conocido por ustedes es el sistema binario, puesto que el grupo de ceros y unos es considerado como un código que representa números decimales. Esta representación binaria de un valor decimal, se denomina ***codificación binaria directa*.**
Como ya es sabido, los sistemas digitales trabajan con números binarios, pero cabe recalcar que el mundo real es fundamentalmente decimal. Por lo tanto, la conversión de decimal a binario es algo frecuente, pero debido a que hay situaciones en las que se deben realizar conversiones largas, como en el caso de un número grande. Es por esta razón que se ha optado por el uso de varios dispositivos de codificación de números decimales.

2.5.1 Código decimal codificado en binario

La representación binaria de cada dígito perteneciente a un número decimal, es considerada un código denominado ***decimal codificado en binario*** (BCD, por sus siglas en inglés). Para representar en forma binaria cualquier valor decimal, se requieren cuatro bits para codificar cada dígito.
Para ilustrar el código BCD, consideremos el número decimal 296_{10}. Cada dígito es transformado a su equivalente binario como se muestra a continuación:

Decimal	2	9	6
BCD	0010	1001	0110

En resumen, el código BCD consiste en transformar cada dígito decimal en un grupo de cuatro dígitos binarios o *cuarteto*; así, por cada cifra del número decimal, existirá un cuarteto en BCD. Cuando usemos este código debemos tener en cuenta que estamos limitados a usar sólo números binarios de 0000 (0 decimal) a 1001 (9 decimal), esto quiere decir que si aparece otro número binario fuera de los que están permitidos en alguna máquina, ésta generalmente nos avisará que hemos hecho algo erróneo.

2.5.2Diferencias entre BCD y binario

Hay que tener en cuenta que el BCD no es un sistema de numeración, sino más bien, es el sistema decimal con cada dígito codificado en binario. Al contrario del sistema binario que representa en forma binaria el número decimal completo, el código BCD convierte cada dígito decimal individualmente. El siguiente ejemplo ilustra lo dicho anteriormente:

Binario	$364_{10} = 101101100_2$
BCD	364_{10} = 0011 0110 0100

Como se ve en el ejemplo, el código BCD representa de manera individual el número decimal utilizando cuatro bits para cada dígito (en este caso requiere de 12 bits), mientras que el código binario representa todo el número utilizando 9 bits en total.
El código BCD posee una gran ventaja, que es la facilidad de realizar la conversión a decimal y desde decimal. Para ello simplemente es necesario recordar los grupos los códigos de cuatro bits para los números del 0 al 9. Esta ventaja es importante para el hardware porque en los sistemas digitales los circuitos lógicos realizan estas conversiones a y desde decimal.

Ejercicios de repaso 2.5
1.- Cambie el número decimal 249 a su representación en código BCD.

2.- Transforme el número BCD 011000100101 a su equivalente decimal.
3.- Represente el número 848_{10} en código binario y BCD.
4.- Conteste: ¿Cuántos bits necesita el número613_{10} para su representación en el código BCD?

2.6 Integración de los sistemas

A continuación, verán la siguiente tabla que muestra los números decimales del 1 al 15 representados en los diferentes sistemas de numeración que ya hemos explicado anteriormente. Obsérvenla detenidamente y traten de comprenderla, ya que esta es prácticamente como un resumen general de todo lo que ya vieron:

Decimal	*Binario*	*Octal*	*Hex*	*BCD*
0	0	0	0	0000
1	1	1	1	0001
2	10	2	2	0010
3	11	3	3	0011
4	100	4	4	0100
5	101	5	5	0101
6	110	6	6	0110
7	111	7	7	0111
8	1000	10	8	1000
9	1001	11	9	1001
10	1010	12	A	0001 0000
11	1011	13	B	0001 0001
12	1100	14	C	0001 0010
13	1101	15	D	0001 0011
14	1110	16	E	0001 0100
15	1111	17	F	0001 0101

-Fíjense que la representación en código BCD siempre utiliza *cuatro* bits para representar cada dígito del número decimal.

2.7 El byte

El ***Byte*** es el conjunto de ocho bits que representa la información que es almacenada por la microcomputadora. Por ejemplo:
Una serie o conjunto de 64 bits representa 8 bytes, ya que 8x8 = 64.

Ejercicios de repaso 2.7
1.- Responda: ¿una serie de 40 bits cuántos bytes representa?
2.- ¿Cuántos bits hay en 7 bytes?
3.- ¿Cuántos bytes se requieren para representar el número 4571_{10} en código BCD?

2.8 Códigos alfanuméricos

Una computadora, aparte de trabajar con datos numéricos, también debe ser capaz de reconocer códigos que representan letras, signos de puntuación y otros símbolos. Este tipo de códigos no numéricos son denominados ***códigos alfanuméricos.*** Un código alfanumérico completo incluye 26 letras mayúsculas, 26 minúsculas, 10 dígitos numéricos, 7 signos de puntuación, y entre 20 y 40 símbolos como +, -, /, *, etc. En resumen, un código alfanumérico representa todos los caracteres de un teclado.

2.8.1 El Código ASCII

(Su nombre se pronuncia "asquii"). Sus siglas en español significan *Código Internacional Estándar para Intercambio de Información.* Es un código muy utilizado que posee 7 dígitos y por lo tanto significa que tiene $2^7 = 128$ grupos de códigos posibles. Los suficientes para representar casi todos los caracteres de un teclado. En la siguiente tabla se muestra una lista de caracteres del código ASCII:

Carácter	ASCII (7bits)	Octal	Hex	Carácter	ASCII (7 bits)	Octal	Hex
A	100 0001	101	41	Y	101 1001	131	59
B	100 0010	102	42	Z	101 1010	132	5A
C	100 011	103	43	0	011 0000	060	30
D	100 0100	104	44	1	011 0001	061	31
E	100 0101	105	45	2	011 0010	062	32
F	100 0110	106	46	3	011 0011	063	33
G	100 0111	107	47	4	011 0100	064	34
H	100 1000	110	48	5	011 0101	065	35
I	100 1001	111	49	6	011 0110	066	36
J	100 1010	112	4A	7	011 0111	067	37
K	100 1011	113	4B	8	011 1000	070	38
L	100 1100	114	4C	9	011 1001	071	39
M	100 1101	115	4D	Space	010 0000	040	20
N	100 1110	116	4E	.	010 1110	056	2E
O	100 1111	117	4F	(	010 1000	050	28
P	101 0000	120	50	+	010 1011	053	2B
Q	101 0001	121	51	$	010 0100	044	24
R	101 0010	122	52	*	010 1010	052	2A
S	101 0011	123	53	)	010 1001	051	29
T	101 0100	124	54	-	010 1101	055	2D
U	101 0101	125	55	/	010 1111	057	2F
V	101 0110	126	56	,	010 1100	054	2C
W	101 0111	127	57	=	011 1101	075	3D
X	101 1000	130	58	(return)	000 1101	015	0D
				(linfeed)	000 1010	012	0A

Ahora veremos un ejemplo que muestra cómo se utiliza este código, para ello vamos a descifrar el siguiente mensaje codificado en código ASCII:

1001000 1000101 1001100 1001100 1001111

Primero debemos convertir cada código su equivalente hex, por lo que nos quedaría así:

48 45 4C 4C 4F

Y por último buscamos los resultados obtenidos en la tabla anterior y determinamos que tipo de carácter representa cada valor:

H E L L O

Este proceso también puede realizarse a la inversa, por ejemplo trataremos de codificar el mensaje "SOS":

S = 01010011
O =01001111
S = 01010011

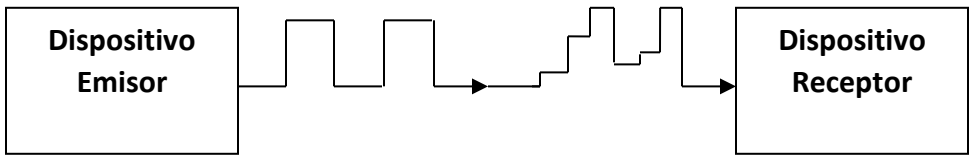
-Noten los ceros agregados, estos son necesarios debido a que los códigos deben almacenarse como bytes. Esta acción de adición es conocida como *relleno de ceros*.

Ejercicios de repaso 2.8
1.- ¿Qué dice el siguiente mensaje?
1010111 1000001 1010010 1001110 1001001 100 1110 1000111
2.- Codifique el siguiente mensaje en código ASCII usando la representación hex: "NO ENTER".
3.- ¿Qué significan las siglas "ASCII"?
4.- ¿en qué consiste el relleno de ceros?
5.- ¿Cómo se representa el carácter "space" en el código ASCII?

2.9 Método paridad para la detección de fallas

La principal operación que realizan los sistemas digitales es la transmisión de datos binarios de un punto a otro.
Existen situaciones en las que cuando se esté realizando la transmisión de información, entre un emisor y un receptor, se puedan producir errores que impliquen el deterioro de las señales durante el viaje. La principal causa de este deterioro es el *ruido eléctrico* que se presenta en los sistemas electrónicos con grados variantes. Para ilustrar lo anterior:

Figura 2.2 Representación gráfica de un error de transmisión causada por el ruido.

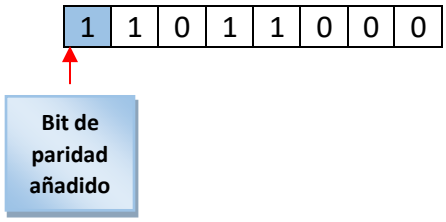


Como se ve en la figura el emisor transmite una señal libre de ruido a través de una línea de conexión al receptor. Sin embargo, cuando la señal llega al receptor ésta se ve afectado por el ruido haciendo que el receptor interprete de manera incorrecta los bits enviados.
Para evitar esto la mayoría de equipos digitales se diseñan de manera que sean inmunes a errores, aunque hay que reconocer que en ocasiones los sistemas digitales transmiten millones de bits por segundo, de tal forma que inclusive un índice muy bajo de ocurrencia de fallas puede producir un error ocasional que puede llegar a ser molesto, y en algunos casos hasta perjudicial. Es por esa razón que los sistemas digitales aplican métodos y técnicas de detección de fallas. Uno de los esquemas más utilizados y simples que existe es el ***método de paridad.***

2.9.1 El Bit de paridad

Llamamos **bit de paridad** a un bit adicional que se añade a un grupo de código que se desplaza de un punto a otro. Este bit se conforma de un cero o un uno, con respecto al número de contenidos en el grupo del código. Para todo lo anterior se emplean los siguientes dos métodos:

-Método de paridad par: En este método el valor que tendrá el bit de paridad se escoge siempre y cuando la cantidad de "1" en el grupo de código sea un número ***par***, incluyendo el bit de paridad. Como ejemplo, consideremos que el grupo sea 1011000 (carácter "X" en ASCII). Este grupo de código se conforma de *tres* números (impar), por lo que se añadirá un bit de paridad 1 para que la cantidad total de unos sea par. Por lo que el *nuevo* grupo de código queda de la siguiente forma:



Si el grupo de código posee una cantidad par de unos, el bit de paridad adoptará el valor de cero. Por ejemplo si tenemos 0101011 (carácter "+" en ASCII), la paridad asignada seria de 0 dejándonos el nuevo grupo de código, incluyendo el bit de paridad, así: **00101011.**

-Método de paridad impar: Se usa igual que el anterior método, pero con la única diferencia de que ahora el bit de paridad se escoge de tal forma que la cantidad total de unos sea ***impar***. Por ejemplo, para el grupo código 1000100, el bit paridad tendrá el valor de 1, pero si el grupo código fuera 0110111, entonces el bit paridad valdría 0.
No importa si se usa el método de paridad par o impar, el bit de paridad se vuelve parte de la palabra código. El bit de paridad ayuda a detectar errores en un solo bit que ocurre durante la transmisión de un código de un lugar a otro. Por ejemplo, consideren la transmisión del carácter "W" y que en este caso se usa paridad ***impar.*** El código que se transmitirá será:

1 1 0 0 1 0 0 0

Al recibir este código, el receptor verificará que la cantidad que hay de unos sea impar, incluyendo el bit de paridad, por lo que asumirá que el código se recibió sin ningún error. En caso de la presencia de alguna falla, provocada por el ruido, haga que realmente se reciba por error este código:

1 1 0 0 0 0 0 0

Lo que hará el receptor es detectar que este código posee un número impar de unos. Esto le avisará que el código tiene un error, puesto que se acordó usar el método paridad impar entre el emisor y el receptor. Sin embargo, es imposible que el receptor identifique el bit erróneo, debido a que no se sabe exactamente qué código es.

Queda bastante claro que el método paridad no funcionará si dos bits son erróneos, porque dos errores no alteran el estado de impar a par de la cantidad de unos del código. Durante la práctica, este método solamente es utilizado en momentos en que la probabilidad de un sólo error es bajísima y la de dos errores es nula.

Al usar el método paridad, debe haber un acuerdo anticipado relacionado que tipo de paridad se usará entre el emisor y el receptor. Entre estos dos métodos no existe ventaja de una sobre la otra, aunque hay que mencionar que el método **par** es el más utilizado. El emisor deberá añadir un bit de paridad correcto para cada unidad de información que se transmite. Por ejemplo, cuando el emisor transmita información codificada en ASCII, añadirá un bit de paridad adecuado a cada grupo ASCII conformado de 7 bits. El momento en el que el receptor recibe la información, la analiza para asegurarse de que la cantidad de *unos* sea la deseada con respecto al método de paridad establecido. Todo ese proceso es conocido como *verificación de la paridad* de la información. Si es que en algún momento se detecta un error, el receptor le envía una orden al emisor para vuelva a transmitir el último conjunto de información.

Ejercicios de repaso 2.9

1.- Añada un bit de paridad par al código ASCII del carácter ",".
2.- ¿Cuál de los dos métodos es el mejor?
3.- Defina: ¿Qué es un bit de paridad?
4.- Coloque un bit de paridad impar al siguiente código:
0110000
5.- Explique: ¿Qué es el método paridad?

SOLUCIONARIO

Ejercicios de repaso 2.1

1.- 1 0 1 10 0 0 1

$2^7\ 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^o = 128+32+16+1$

$= 177_{10}$

2.-Todo valor binario puede convertirse en su equivalente decimal con simplemente sumar los pesos de las diferentes posiciones en el número binario que contenga un "1".

3.- El LSB tendrá un peso de 2^o (1) y el MSB tendrá un peso de 2^{2o} (1048576)**.**

Ejercicios de repaso 2.2

1.-El primer método consiste en la inversa del anterior proceso demostrado en el tema 2.1 y el segundo método es en la que se usa la división repetida.

2.- Consiste en dividir el número decimal entero para 2 repetidamente.

3.- 47/2 = 23 + residuo de 1
23/2 = 11 + residuo de 1
11/2 = 5 + residuo de 1
5/2 = 2 + residuo de 1
2/2 = 1 + residuo de 0
1/2 = 0 + residuo de 1
$47_{10} = 101111_2$

Ejercicios de repaso 2.3

1.- $556_8 = 5x8^2 + 5x8^1 + 6x8^o$
$= 5x64 + 5x8 + 6x1$
$= 366_{10}$

2.- 120/8 = 15 + residuo de 0
15/8 = 1 + residuo de 7
1/8 = 0 + residuo de 1
$120_{10} = 170_8$

3.- 1 2 5
↓ ↓ ↓
001 010 101

4.- **0 0** 1 0 0 1 1 0 1
1 1 5

Ejercicios de repaso 2.4

1.- 112/16 = 7 + residuo 0
7/16 = 0 + residuo 7
$112_{10} = 70_{16}$

2.- $8A5_{16}$ = 8 A 5
= 1000 1010 0101
$= 100010100101_2$

3.- $2D4_{16}$ = 0010 1111 0100
= 001 011 110 100
= 1 3 6 4_8

Ejercicios de repaso 2.5

1.-

Decimal	2	4	9
BCD	0010	0100	1001

2.- 011000100101
6 2 5

3.- $848_{10} = 01101010000_2$(**Binario**)
848_{10} = 1000 0100 1000(**BCD**)

4.-Se requieren 12 bits, debido a que son 3 dígitos representados por un grupo de 4 bits cada uno.

Ejercicios de repaso 2.7

1.- 40/8 = 5 bytes

2.- 7x8 = 56 bits

3.- El número 4571_{10} en código BCD se representa con 16 bits, por lo tanto requiere de 2 bytes.

Ejercicios de repaso 2.8

1.-W A R N I N G

2.-"NO ENTER" = 1001110 1001111 0100000 1000101 1001110 1010100 1000101 1010010

3.- *"Código Internacional Estándar para Intercambio de Información".*

4.- Los ceros necesarios que deben ser agregados debido a que los códigos deben almacenarse como bytes (8 bits).

5.- 010 000

Ejercicios de repaso 2.9

1.- 10 1 0 1 1 0 0

2.- Entre estos dos métodos no existe ventaja de una sobre la otra, aunque hay que mencionar que el método par es el más utilizado.

3.- Llamamos bit de paridad a un bit adicional que se añade a un grupo de código que se desplaza de un punto a otro.

4.- 10 1 1 0 0 0 0

5.- Es un método empleado por los sistemas digitales para la detección de fallas.

CAPITULO N°3

Algebra de Boole y Compuertas Lógicas

♦ TEMARIO:

3.1 Constantes y variables Booleanas
3.2 Tabla de Verdad
3.3 Operaciones con compuertas OR
3.4 Operaciones con compuertas AND
3.5 Operación NOT
3.6 Descripción algebraica de los Circuitos lógicos
3.7 Evaluación de salidas de los Circuitos lógicos
3.8 Implementación de circuitos mediante expresiones booleanas
3.9 Compuertas NOR y NAND
3.10 Teoremas Booleanos
3.11 Teoremas de DeMorgan
3.12 Universalidad de la compuerta NOR y compuerta NAND
3.13 Representaciones alternas de las compuertas lógicas
3.14 Qué tipo de representación de compuertas se debe usar
3.15 Simbología lógica estándar IEEE/ANSI

♦OBJETIVOS:

El objetivo principal de este libro es la de proporcionar la información necesaria a todos los estudiantes, que estén iniciando en la especialidad de electrónica, para que sean capaces de:

♦Desarrollar operaciones lógicas básicas.

♦Definir la operación y realizar tablas de verdad para las compuertas AND, NAND, OR, NOR y para el circuito NOT.

♦Graficar diagramas de sincronización para las compuertas de circuitos lógicos.

♦Expresar la forma booleana de las compuertas lógicas y combinaciones de compuertas lógicas.

♦Ejecutar circuitos lógicos usando las compuertas AND, OR y NOT.

♦Conocer la importancia del algebra de Boole para simplificar expresiones lógicas.

♦Emplear el teorema de DeMorgan para realizar simplificaciones de expresiones lógicas.

♦Saber cómo utilizar las compuertas lógicas (NAND y OR) para implementar un circuito representado mediante una expresión booleana.

♦Mencionar las ventajas de diseñar diagramas de circuitos lógicos usando los símbolos de compuertas alternos contra los símbolos estándar de compuertas.

♦Explicar los conceptos de las señales activa BAJA y ALTA.

♦Graficar y distinguir los símbolos estándar de compuertas lógicas IEEE/ANSI.

♦INTRODUCCCION:

Como ya hemos visto en el capítulo 1, los circuitos digitales conocidos también como circuitos lógicos trabajan principalmente con el sistema binario donde los voltajes de entrada y salida pueden representar un BAJO o un ALTO; los valores binarios de 0 y 1 representan intervalos de voltaje prescritos. Gracias a esta característica que poseen los circuitos lógicos podemos emplear el "álgebra de Boole" para realizar análisis y diseño de sistemas digitales. El álgebra de Boole es un emblemático sistema matemático, algo simple, que nos permite manejar ecuaciones, las cuales pueden ser simplificadas y convertidas en o desde un sistema físico de compuertas lógicas (las cuales realizan esa misma función). Es decir, que mediante las matemáticas podemos hacer que un sistema de control complejo se simplifique. Ustedes constatarán cómo la operación de las compuertas lógicas y los circuitos más complejos pueden ser analizados con el uso del álgebra booleana. De igual manera, verán una introducción sobre cómo el álgebra de Boole se emplea para realizar simplificaciones de expresiones de un circuito, permitiendo que éste se pueda reconstruir sin la necesidad de usar muchas compuertas lógicas. En el capítulo 4, se estudiará más a detalle la simplificación de circuitos. El álgebra de Boole nos ayuda también a plantear circuitos lógicos que produzcan la relación entrada/salida deseada. En este capítulo 3 sólo introduciremos lo básico, para luego estudiar esto más a fondo en el capítulo 4. El álgebra de Boole expresa algebraicamente las operaciones de un circuito, por lo que es perfecto para introducir la operación a una computadora,

la cual funciona con un software que necesita para saber cómo es el circuito. El software puede funcionar como una rutina de simplificación del circuito que simplifique la ecuación de entrada del álgebra booleana y haga una nueva versión simplificada del circuito original. Otra aplicación podría ser el software que proporciona mapas de fusibles requeridos para programar un PLD (Dispositivo lógico programable). El que realiza la operación ingresa las ecuaciones booleanas para la operación del circuito que se desee y el software se encarga de convertirla en un mapa de fusibles. Hablaremos más acerca de este proceso en el capítulo 4. El álgebra de Boole es una asombrosa herramienta de descripción, análisis y diseño de circuitos digitales. Los estudiantes que en el futuro trabajaran en todo lo relacionado a digital deben esforzarse para alcanzar una comprensión definitiva del álgebra booleana. Traten de concentrarse y entender las explicaciones con sus respectivos ejemplos para que sean capaces de efectuar los ejercicios que se plantean en cada tema. Y recuerden siempre practicar o dense el tiempo para una segunda leída a los temas que no entienden por completo.

3.1 Constantes y variables Booleanas

Denominamos variable Booleana a una cantidad que puede ser ocasionalmente igual a 0 o 1. Estas variables suelen ser usadas para representar el nivel de voltaje en un cable. Por ejemplo, en un sistema digital el valor booleano de 0 puede ser asignado a cualquier intervalo de voltaje de entre 0V a 0.8V, mientras que el valor booleano de 1 puede ser asignado a cualquier intervalo de voltaje de entre 2V a 5V.
Por lo tanto, el 0 y 1 booleanos no presentan números reales, sino más bien el estado de una variable de voltaje, o lo que conocemos como *nivel lógico.* Se dice que el voltaje de un circuito está en el nivel lógico 0 o 1, dependiendo de su valor numérico real. Dentro de la lógica digital empleamos sinónimos que puedan representar el 0 y 1. Estos sinónimos se muestran a continuación en la siguiente tabla:

0 lógico	*1 lógico*
Bajo	Alto
OFF	ON
Desactivado	Activado
Apagado	Encendido
NO	SI
Falso	Verdadero
Interruptor Abierto	Interruptor Cerrado

Como ya se ha dicho en la introducción, el álgebra booleana expresa la relación entre las entradas y salidas de un circuito lógico. Las entradas son consideradas variables, cuyos niveles lógicos pueden determinar los niveles de salida. Posterior a eso, utilizaremos literales para la representación de variables lógicas. Por ejemplo, podríamos usar el literal "X" para representar una entrada o salida de un circuito lógico y en cualquier momento se debe tener X = 0 o X = 1: entonces el valor que tome "X" si no es cero entonces es uno y viceversa.

El álgebra booleana es más fácil si lo comparamos con el álgebra común, debido a que éste sólo consta de dos variables. En el álgebra de Boole no existen valores negativos, quebrados, decimales, raíces, etc. Técnicamente en el álgebra booleana sólo existen *tres* tipos de operaciones fundamentales que son: Suma (se usa la puerta OR), Multiplicación (se usa la puerta AND) e Inversión (se usa la puerta NOT).

Ejercicios de repaso 3.1
1.- ¿Qué es una variable Booleana?
2.- Defina qué es un "nivel lógico".
3.- ¿Cuál de los siguientes términos corresponden a sinónimos comunes que pueden representar un 0 o un 1?
a) Tibio y Caliente.
b) ON y OFF.
c) HIGH y LOW.
d) Crudo y Cocinado.

3.2 Tabla de Verdad

Una **tabla de verdad** es un medio para explicar el funcionamiento de un sistema digital; en ella se presenta el estado de las entradas y salidas para cada combinación posible que se pueda dar en el circuito.
En la siguiente figura se muestra una tabla de verdad de un sistema de tres entradas en el cual el cero representa la ausencia de corriente y el 1 representa la presencia de tensión. También se puede representar por literales donde L (LOW) representa 0 y H (HIGH) representa 1.

Figura 3.1

Ejemplo de tabla de verdad de tres entradas (A, B, y C).

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

-Noten que esta tabla de tres entradas posee 16 anotaciones, por lo tanto, deducimos que el número de combinaciones de entrada es igual a 2^n para una tabla de verdad de ***n*** entradas.

Ejercicios de repaso 3.2

1.- ¿Para qué sirve la tabla de verdad?
2.- Defina que representa el 0 y 1 en una tabla de verdad.
3.- ¿En qué estado se encuentra la salida de la tabla de la figura 3.1 cuando todas sus entradas, excepto A son 0?
4.- ¿Cuántas anotaciones puede tener un circuito de 6 entradas?

3.3 Operaciones con compuertas OR

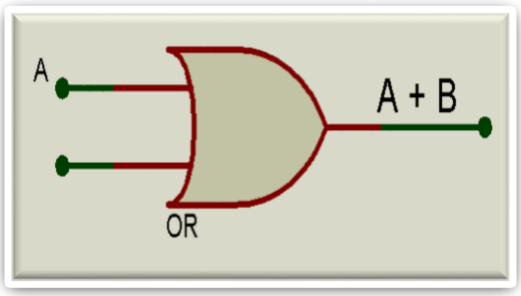
La ***operación OR*** es la operación booleana básica que deben aprender primero. Para ello, se presentará la siguiente tabla (**Figura 3.2**) que muestra qué ocurre cuando dos entradas se combinan usando la operación OR para producir la salida **Q.**
-La operación OR se basa en una operación de *suma*, por lo tanto se expresa de la siguiente forma booleana:

$$A + B = Q$$

-Cabe mencionar que al ser una operación de adición OR implica que cuando las entradas A y B son 1, ***Q*** será igual a 1 y no 2; en el caso de tres entradas "Q" tampoco podrá ser igual a 3, sino también seguirá siendo 1. En el álgebra de Boole se indica que "1" es el máximo valor, por lo que nunca se podrá obtener un resultado mayor a 1.

Figura 3.2

Tabla de verdad que representa la operación OR y la simbología de una compuerta lógica OR de dos entradas.



A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

-La regla general de la operación OR se enuncia de la siguiente manera:

"Basta con que una de sus salidas sea igual a uno para que su salida también sea igual a uno".

3.3.1 Compuerta lógica OR
En un circuito digital la compuerta lógica OR es aquella cuya salida es igual a la combinación OR de sus dos o tres entradas.

Figura 3.3

Símbolo **MIL** (aplicaciones de lógica militar) y símbolo **CEI** (Comisión Electrotécnica Internacional) de una compuerta OR.

Ejercicios de repaso 3.3
1.-La operación booleana de la operación OR es:
a) Q = A/B
b) Q = A .B
c) Q = A+B
d) Q = A-B
2.-Cierto o falso: La operación OR produce 1 + 1 = 2.
3.- ¿Cuál es la regla general de la operación con compuertas OR?
4.- ¿Qué es una compuerta OR?

3.4 Operaciones con compuertas AND

La ***operación AND*** es la segunda operación básica booleana que deben aprender. Para ello, se presentará la siguiente tabla (Figura 3.4) que muestra qué ocurre cuando dos entradas se combinan usando la operación NAND para producir la salida **Q.**
-La operación AND se basa en una operación de *multiplicación*, por lo tanto se expresa de la siguiente forma booleana:

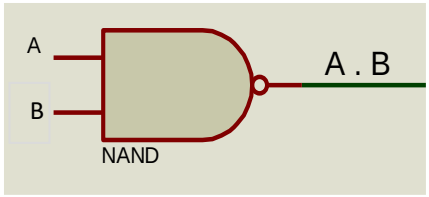
$$A.B = Q$$

-En esta expresión el punto (“**.**”) es el signo que representa la multiplicación.

Figura 3.4

Tabla de verdad que representa la operación AND y la simbología de una compuerta lógica AND de dos entradas.

A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1



-La regla general de la operación AND se enuncia de la siguiente manera:

“Basta con que una de sus salidas sea igual a cero para que su salida también sea igual a cero”.

3.4.1 Compuerta lógica AND

En un circuito digital la compuerta lógica AND es aquella cuya salida es igual al producto AND de sus dos o tres entradas.

Figura 3.5

Símbolo **MIL** (aplicaciones de lógica militar) y símbolo **CEI** (Comisión Electrotécnica Internacional) de una compuerta AND.

<u>Ejercicios de repaso 3.4</u>

1.-La operación booleana de la operación AND es:
a) Q = A/B
b) Q = A .B
c) Q = A+B
d) Q = A-B
2.-Cierto o falso: La operación AND produce 1. 0 = 1.
3.- ¿Cuál es la regla general de la operación con compuertas AND?
4.- ¿Qué es una compuerta AND?

3.5 Operación NOT

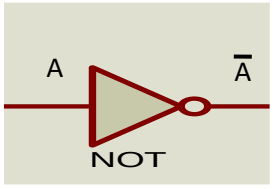
La ***operación NOT*** se distingue de las operaciones OR y NAND porque éste puede ser realizado en una sola variable de entrada. Por ejemplo, supongamos que se aplica la operación NOT a la variable de entrada "A", entonces el resultado de la salida **Q** se expresaría de la siguiente manera:

$$\bar{A} = Q$$

-Observen la barra que esta sobrepuesta en la variable "A", ésta sirve para representar la operación NOT. Esta expresión indica que "*Q* es igual al valor inverso de A". Para explicar mejor esto observen la siguiente tabla (**Figura 3.6**) que muestra que esta operación siempre tendrá una sola entrada y su salida es igual a la inversa del nivel lógico de esta entrada.

Figura 3.6

Tabla de verdad que representa la operación NOT y la simbología de un circuito NOT (inversor).



A	$Q = \bar{A}$
0	1
1	0

3.5.1 Resumen de las operaciones Booleanas

Las reglas para las distintas operaciones se pueden resumir a través de las siguientes expresiones:

OR	AND	NOT
0 + 0 = 0	0 . 0 = 0	$\bar{0} = 1$
0 + 1 = 1	0 . 1 = 0	$\bar{1} = 0$
1 + 0 = 1	1 . 0 = 0	
1 + 1 = 1	1 . 1 = 1	

Ejercicios de repaso 3.5

1.-La operación booleana de la operación NOT es:
a) Q = A/B
b) Q = A .B
c) Q = A+B
d) Ninguna de las anteriores.
2.-Cierto o falso: La operación NOT produce 0.1 = 0
3.- ¿Para qué sirve la barra sobrepuesta en una variable?

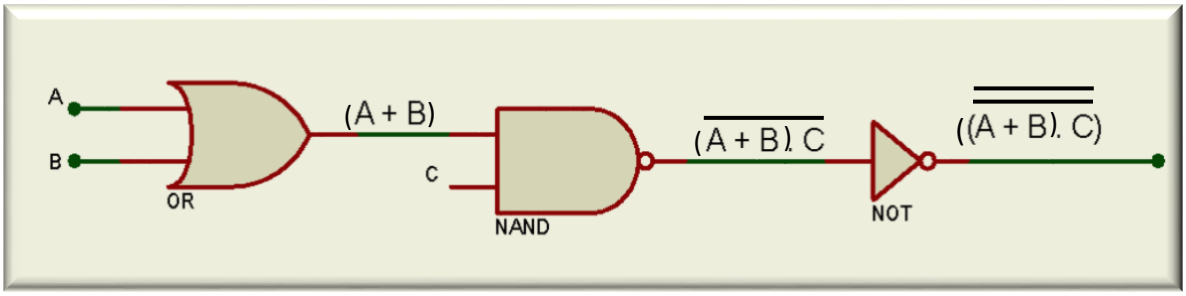
3.6 Descripción algebraica de los Circuitos lógicos

Cualquier tipo de circuito lógico puede ser descrito empleando el uso de las operaciones booleanas que ya hemos visto, ya que la compuerta OR, compuerta AND y el circuito NOT son las principales

bases para la construcción de los sistemas digitales. Por ejemplo, observemos el siguiente circuito (figura 3.7) de tres entradas y una salida con Inversor NOT. Expresando cada compuerta de forma booleana, podemos determinar la expresión de la salida.

Figura 3.7

Circuito lógico con su salida expresada en forma booleana.



Como se ve el resultado que obtenemos del circuito, conformado de una compuerta OR de dos entradas que tiene su salida conectada a una de las entradas de la compuerta AND, y esta a su vez tiene su salida conectada a un Inversor (NOT), expresa la salida como: (A+B).C = Q (si eliminamos la doble barra), el momento de efectuar puede llegar a ser confuso debido a que no sabemos si realizar primero la suma o la multiplicación. Para evitar este tipo de confusiones se establece que las operaciones AND se realizan primero, pero en caso de que existan paréntesis en la expresión, se debe asumir que la operación que está dentro del paréntesis es la que se realizará primero.

Ejercicios de repaso 3.6

1.- ¿Cómo queda expresada la salida del circuito de la figura 3.7 si conectamos una compuerta AND a la salida del Inversor (NOT)?

2.- En el circuito de la figura 3.7 cambie la compuerta AND por una compuerta OR y escriba la expresión de la salida.

3.7 Evaluación de salidas de los Circuitos lógicos

Cuando ya determinemos la expresión booleana de la salida de un circuito podemos determinar el nivel lógico de la salida para cualquier grupo de niveles de entrada. A manera de ejemplo, supongamos que queremos obtener el nivel lógico de la salida para el anterior circuito de la figura 3.7 en donde los valores de las entradas son A=0, B=0 y C=1. Al sustituir los valores de las variables en la expresión y realizando la operación podemos obtener el siguiente valor que corresponde a la salida:

$$\overline{\overline{(A+B.C)}} = \mathbf{Q}$$
$$\overline{(0 + \overline{0 \,.\, 1})} = \mathbf{Q}$$
$$\overline{(0 + \overline{0})} = \mathbf{Q}$$
$$\overline{0} = \mathbf{Q}$$
$$1 = \mathbf{Q}$$

-Generalmente, cuando queramos obtener el nivel lógico de la salida de algún circuito se deben tener en cuenta las siguientes reglas:

a) Primeramente, realizar las inversiones de términos simples como: $\overline{0} = 1$ y $\overline{1} = 0$.
b) Después de realizar las inversiones, proseguimos a efectuar todas las operaciones dentro de paréntesis.
c) Efectuar la operación AND antes que la operación OR, a menos que haya la presencia de paréntesis que diga lo contrario.
d) Si una barra de inversión esta sobrepuesta en una variable primero se deben realizar las operaciones dentro de la expresión, y por último invertir ese resultado.

Ejercicios de repaso 3.7
1.- Determine el valor lógico de la salida de la figura 3.7 cuando el valor de las entradas son A=1, B=0 y C=0.
2.- ¿Cómo se obtiene el valor de una salida cuando se tiene a disposición los valores de las entradas?
3.- Cierto o falso: En caso de la ausencia de paréntesis, se realiza la operación AND antes que la operación OR.

3.8 Implementación de circuitos mediante expresiones booleanas

Cuando una expresión Booleana define la operación de un circuito, se puede graficar un circuito lógico directamente mediante esa expresión. Por ejemplo, si necesitáramos de un circuito que fuese definido por la expresión Q = A + B + C + D, rápidamente nos daríamos cuenta de que requerimos una compuerta OR de tres entradas y una compuerta OR de dos entradas o sino más sencillo dos compuertas OR de dos entradas. Otro ejemplo es que si necesitáramos de un circuito que estuviese definido por la expresión Q = A . B + C, entonces usaríamos una compuerta AND de dos entradas con una compuerta OR de dos entradas, conectada en la salida AND.

Ejercicios de repaso 3.8
1.- ¿Qué es lo que necesitaría un circuito que estuviera definido por la siguiente expresión Q = A .B . C?
2.- ¿Cuál es la expresión que define a un circuito que está compuesto por una compuerta OR de 3 entradas?

3.9 Compuertas NOR y NAND

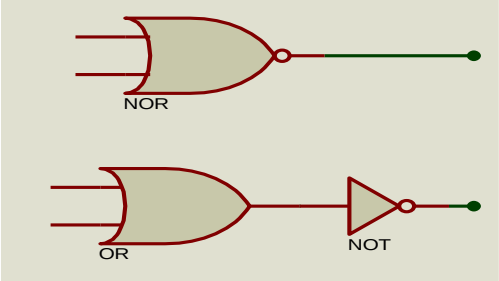
Las compuertas NOR y NAND son extensamente usadas en los circuitos digitales, debido a que estos dos tipos de compuertas combinan las operaciones OR, AND y OR, por lo que hace fácil escribir sus expresiones booleanas.

3.9.1 La Compuerta NOR
El símbolo de una compuerta NOR se parece al símbolo de una compuerta OR, pero con la pequeña diferencia de que la compuerta NOR tiene un pequeño círculo en su salida, el cual representa la operación de inversión. En otras palabras, una compuerta NOR es como un circuito compacto

compuesto de una compuerta OR seguida de una NOT en su salida. Para ilustrar lo dicho anteriormente se presenta la siguiente figura:

Figura 3.8

Simbología y tabla de verdad de una compuerta NOR



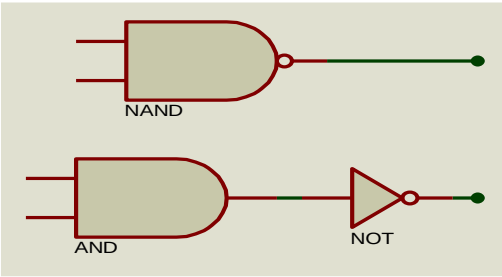
Entradas		OR	NOR
A	B	A + B	A + B
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

3.9.2 La Compuerta NAND

En la siguiente figura (Figura 3.9) se muestra el símbolo de la compuerta NAND, el cual se parece al símbolo de la compuerta AND, y al igual que la NOR ésta también posee un pequeño círculo que representa la operación de inversión. En otras palabras, una compuerta AND es como un circuito compacto compuesto de una compuerta AND seguida de una NOT en su salida. Para ilustrar lo dicho anteriormente se presenta la siguiente figura:

Figura 3.9

Simbología y tabla de verdad de una compuerta NAND.



Entradas		OR	NAND
A	B	A . B	A . B
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Ejercicios de repaso 3.9

1.- Mencione una característica de la compuerta NOR.
2.- Mencione una característica de la compuerta NAND.

3.10 Teoremas Booleanos

Hasta ahora hemos aprendido como el álgebra de Boole es una herramienta útil para analizar un circuito lógico y expresar matemáticamente su operación. A partir de este punto proseguiremos con el estudio del álgebra booleana analizando los teoremas booleanos, es decir, las reglas que nos ayudarán en la simplificación de expresiones y circuitos lógicos. A continuación, veremos los enunciados de los ocho teoremas, en los cuales se usará "x" como una variable lógica que puede ser 0 o 1.

-El **primer teorema** nos dice que si cualquier variable se opera con AND y con un 0 el resultado deberá ser 0 debido a que todo número multiplicado por 0 es 0.

1) x . 0 = 0

-El **segundo teorema** también es muy obvio en comparación con la multiplicación.

2) x . 1 = x

-El **tercer teorema** se puede demostrar ensayando los anteriores casos. Si la variable "x" es igual a 0, entonces 0 . 0 = 0, pero si la variable "x" es igual a 1, entonces 1 . 1 = 1. Por lo tanto este teorema se expresa de la siguiente manera:

3) x . x = x

-El **cuarto teorema** se puede demostrar de igual manera o se puede decir que en cualquier momento "x" o su inverso tiene que estar a nivel 0 y por lo tanto su producto AND deberá ser siempre 0.

4) $x \cdot \bar{x} = 0$

-El **quinto teorema** dice que cualquier número que se le sume 0 no altera su valor.

5) x + 0 = x

-El **sexto teorema** establece que cualquier variable que se opera con OR con 1, el resultado será siempre igual a 1. Ya que recordemos que en una operación OR *basta con que una entrada sea 1 para que su salida también sea 1.*

6) x + 1 = 1

-El **séptimo teorema** puede ser demostrado comprobando ambos valores de "x": 0 + 0 = 0 y 1 + 1 = 1.

7) x + x = x

-El **octavo teorema** puede ser demostrado de igual forma o se puede decir que en cualquier momento x o $\bar{x}$ tiene que estar a nivel 1, de modo que siempre se opere con OR un 0 y 1, lo cual nos da un 1 como resultado.

8) $x + \bar{x} = 1$

3.10.1 Teoremas con más de una variable

Acabamos de ver los ocho teoremas que implicaban una variable, a partir de ahora veremos los demás teoremas que implican variables múltiples:

-El **noveno** y **décimo teorema** son denominados *leyes conmutativas.* Estas leyes consisten en que el orden de los factores no afecta al producto.

9) x + y = y + x

10) x . y = y .x

- El **onceavo** y **doceavo teorema** son conocidos como las *leyes asociativas,* las cuales estipulan que se pueden agrupar las variables en una expresión AND o en una expresión OR en cualquier forma que queramos.

11) x + (y + z) = (x + y) + z = x + y + z

12) x(yz) = (xy)z = xyz

-El **treceavo teorema** es la *ley distributiva,* la cual establece que una expresión puede desarrollarse multiplicando término por término, también indica que podemos factorizar las expresiones, como en el álgebra común.

13a) x(y + z) = xy + xz

13b)(w + x)(y + z) = wy + xy + wz +xz

Todos estos teoremas que acabamos de ver son fáciles de utilizar, pero por otra parte, el *catorceavo* y *quinceavo* teorema pueden ser probados ensayando todos los casos posibles de "*x*" y "*y*".

14) x + xy = x

15a) x + xy = x

15b) $\bar{x}$ + xy = $\bar{x}$

-El **catorceavo** teorema puede ser aplicado para diferentes casos:

Caso 1: Cuando x=0 y=0

x + xy = x

0 + 0 . 0 = 0

0 = 0

Caso 2: Cuando x=0 y=1

x + xy = x

0 + 0 . 1 = 0

$$0 + 0 = 0$$

$$0 = 0$$

Caso 3: Cuando x=1 y=0

$$\mathbf{x + xy = x}$$

$$1 + 1 \,.\, 0 = 1$$

$$1 + 0 = 1$$

$$1 = 1$$

Caso 4: Cuando x=1 y=1

$$\mathbf{x + xy = x}$$

$$1 + 1 \,.\, 1 = 1$$

$$1 + 1 = 1$$

$$1 = 1$$

En conclusión, todos estos teoremas son muy útiles para reducir el número de términos en una expresión lógica. Cuando hacemos esto la expresión reducida nos dará un circuito menos complejo que el original.

Ejercicios de repaso 3.10
1.- ¿Cuál es la importancia de los teoremas booleanos?
2.- ¿en qué consiste el primer teorema?
3.- Cierto o falso el tercer teorema se expresa así: "x . x = x".
4.- ¿Cómo es la expresión del catorceavo teorema?

3.11Teoremas de DeMorgan

Los teoremas de *DeMorgan* son muy útiles para pasar de un tipo de compuerta a otro. Sus dos teoremas son:

-Primer teorema de DeMorgan: Este teorema nos dice que la inversa de una suma OR es equivalente al producto de sus inversas:

$$\mathbf{16)}\ \overline{x + y} = \bar{x} \,.\, \bar{y}$$

-Segundo teorema de DeMorgan: Este segundo teorema nos dice que el inverso de una multiplicación AND es equivalente a la suma de sus inversas:

$$\mathbf{17)}\ \overline{x \,.\, y} = \bar{x} + \bar{y}$$

Ejercicios de repaso 3.11
1.- Defina el primer teorema de Demorgan.

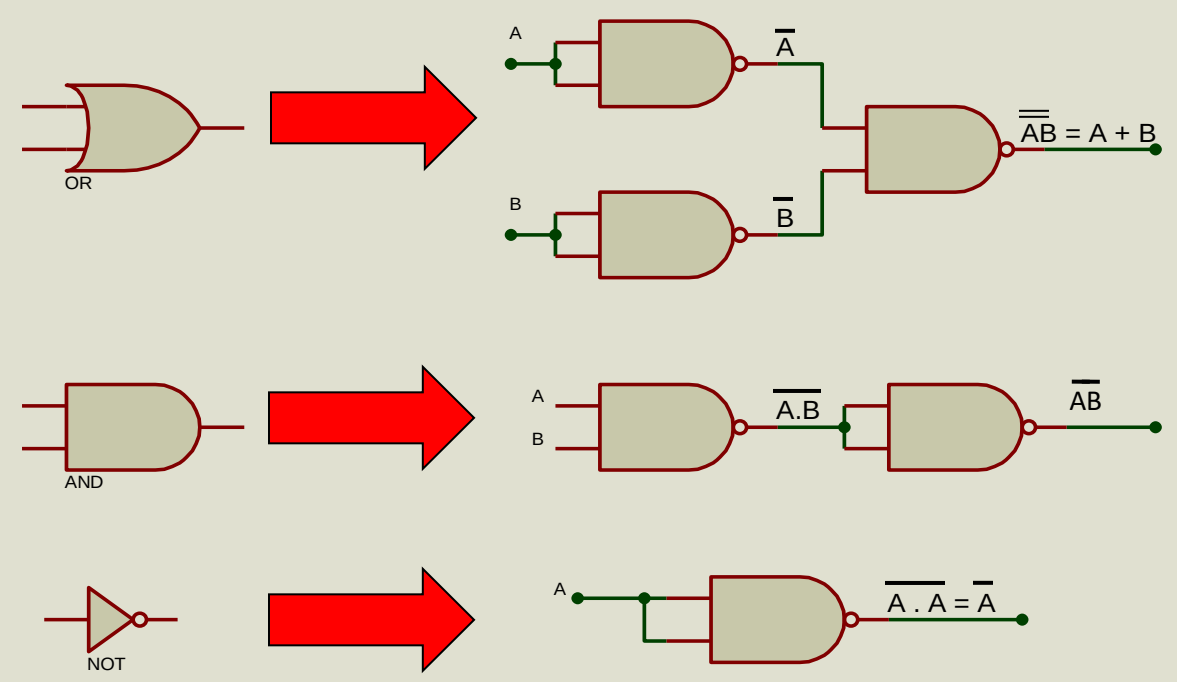
2.- Defina el segundo teorema de Demorgan.

3.12 Universalidad de la compuerta NOR y compuerta NAND

Las expresiones booleanas constan de distintas formas de combinación de las operaciones OR, AND e INVERSION y por lo tanto toda expresión se puede llevar a cabo mediante el uso de las compuertas lógicas, aunque cabe mencionar que usando solamente compuertas NAND uno es capaz de implementar cualquier expresión lógica. Debido a que las compuertas NAND, empleadas en la combinación correcta, se pueden utilizar para realizar cada operación booleana. Para demostrar lo anterior se presenta la figura 3.10.

Figura 3.10

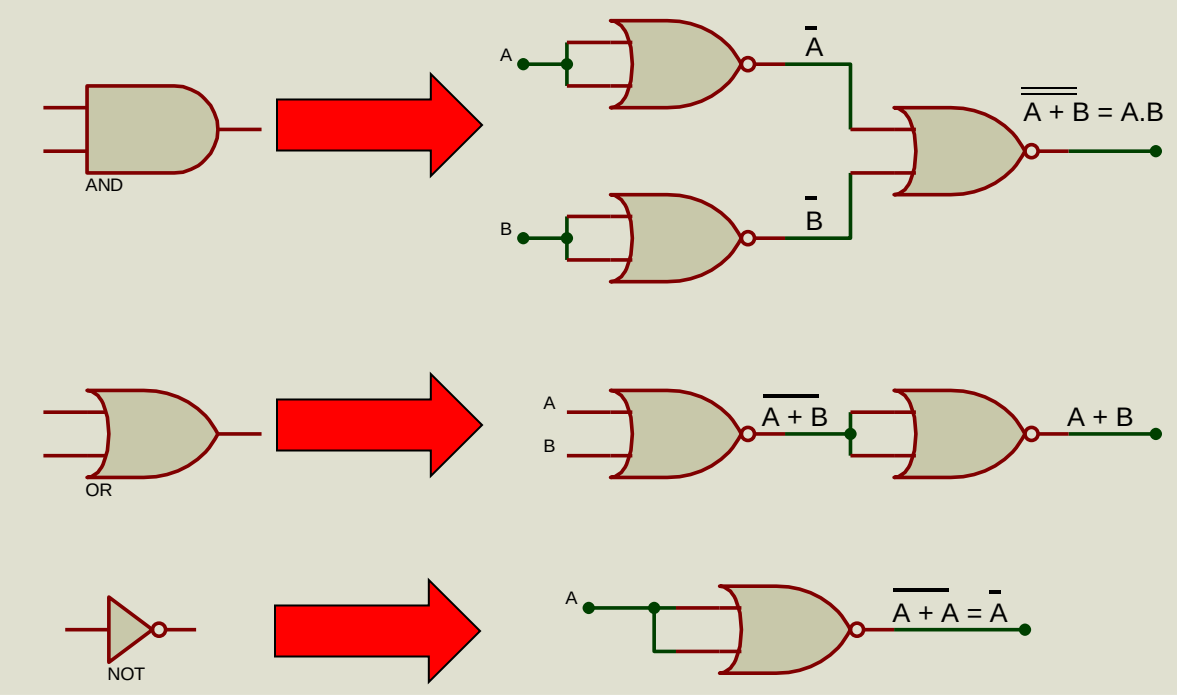
Compuertas AND usadas para implementar cualquier expresión booleana.



En cambio en la **figura 3.11** se presenta lo mismo, pero aplicando solamente compuertas NOR, ya que éstas también se pueden usar para implementar cualquier operación básica:

Figura 3.11

Compuertas NOR usadas para implementar cualquier expresión booleana.

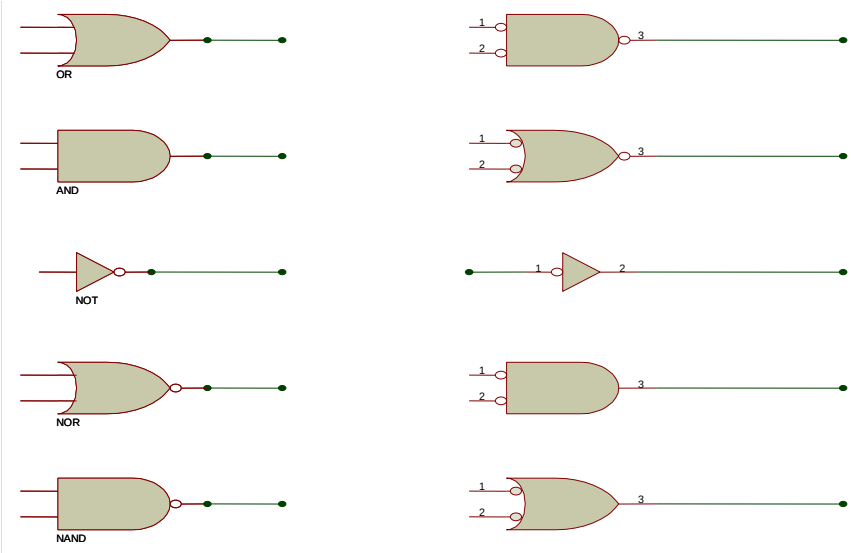


3.13Representaciones alternas de las compuertas lógicas

Ya hemos visto las cinco compuertas lógicas básicas y sus símbolos lógicos estándar que se usan para representarlas en diagramas de circuitos, pero también existen diagramas en los que se utilizan símbolos lógicos alternos, en lugar de los símbolos estándar. En la siguiente figura se muestran los símbolos alternos de cada compuerta:

Figura 3.12

Símbolos estándar y símbolos alternos.



El símbolo alterno para una compuerta se obtiene a partir de su símbolo estándar y realizando los siguientes pasos:

a) Invertir las entradas y salidas del símbolo estándar mediante la agregación de círculos pequeños en las líneas de entrada y salida sin círculos o *burbujas*, y se eliminan las que si los tengan.
b) Cambiamos el símbolo de la operación AND por OR y viceversa. En el caso del inversor NOT, este mantiene su símbolo igual.

3.13.1 Interpretación de los símbolos lógicos
Los símbolos de las compuertas lógicas nos dan una interpretación única de cómo opera cada compuerta. Antes de analizar estas interpretaciones es necesario establecer el concepto de ***niveles lógicos activos***.

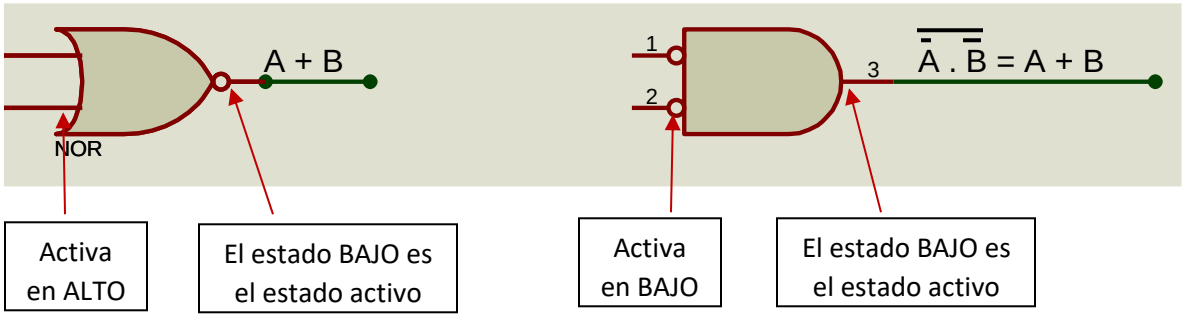
-Una línea es **activa en ALTO** cuando una línea de entrada o salida en un símbolo de un circuito no tiene *burbuja*.

-Una línea es **activa en BAJO** cuando una línea de entrada o salida en un símbolo de un circuito tiene *burbuja*.

En conclusión, la ausencia o presencia de una burbuja establece el estado activo ALTO o BAJO. Para ilustrar esto se presenta la siguiente figura:

Figura 3.13

Interpretación de símbolos de la compuerta NOR.



RESUMEN: Una vez llegado a este punto puede que se estén preguntando *"¿Cuál es la importancia de tener dos símbolos e interpretaciones distintas para cada compuerta lógica?"*. Espero que esta duda sea aclarada al leer el siguiente tema. Pero por ahora, se mostrará un resumen de los aspectos más relevantes respecto a las interpretaciones de las compuertas lógicas:
Cuando queramos determinar el símbolo alterno de una compuerta lógica, tomamos el símbolo estándar y cambiamos su símbolo de operación (de OR a AND o viceversa), y luego removemos las burbujas presentes y agregamos otras en donde no existen.
Para la interpretación de la operación de una compuerta lógica, primeramente determinamos cuál estado lógico (0 o 1) es el activo para las entradas y cuál para la salida. Después, se toma en cuenta que el estado activo se produce cuando *todas* las entradas están en su estado activo (al usar un símbolo AND) o cuando se tiene *cualquier* entrada en su estado activo (al usar un símbolo OR).

Ejercicios de repaso 3.13

1.- Cierto o falso: para determinar el símbolo alterno de una compuerta se cambia el símbolo de la operación OR a AND y viceversa.

2.- ¿Qué es un nivel activo ALTO?

3.- ¿Qué es un nivel activo BAJO?

3.14Qué tipo de representación de compuertas se debe usar

Mayormente se usa la representación estándar para realizar esquemas de circuitos. Aunque es muy utilizada, no ayuda mucho en lo que viene siendo el seguimiento de los circuitos. El correcto uso de los símbolos alternos de compuertas en el diagrama del circuito puede hacer que la operación del circuito sea menos compleja. Para ilustrar esto se presenta la figura 3.14.

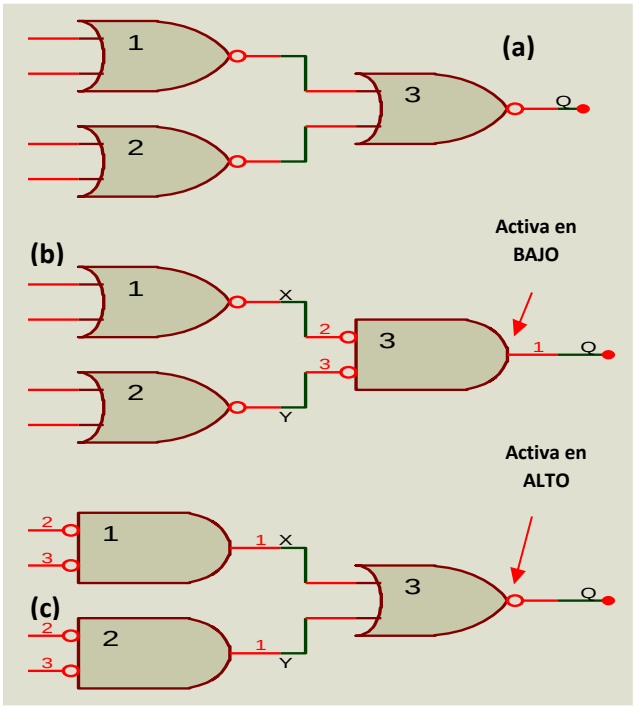
El circuito de la figura 3.14a) contiene tres compuertas NOR conectadas para producir una salida "*Q*" que corresponde a las entradas. El diagrama de este circuito usa símbolos estándar para todas las compuertas OR. A pesar de que este diagrama es correcto, no facilita nada en la comprensión del funcionamiento del circuito. Sin embargo, las representaciones del circuito de las figuras 3.14b) y c) pueden ser analizadas más sencillamente para determinar su operación.

La representación de la figura 3.14b) se obtiene a partir del diagrama original del circuito reemplazando la tercer compuerta NOR Nº3 por su equivalente alterno. En el diagrama la salida *Q* se toma de un símbolo OR que posee una salida activa en ALTO. Así, establecemos que *Q* pasa a estado **ON (ALTO)** cuando X o Y sea **0(o ambas).** Ya que X y Y están presentes en la salida de los símbolos NOR teniendo salidas activas en BAJA, podemos decir que X pasará a estado **OFF (BAJO)** sólo cuando la entrada A o B sea **1 (o ambas)**, y Y pasará a estado **OFF** sólo cuando la entrada C o D sea **1 (o ambas)**. Deduciendo todo lo dicho podemos describir la operación del circuito de la siguiente forma:

"La salida Q pasará a estado ON siempre y cuando A = B = 1 y C = D = 1". Esta descripción la podemos interpretar mediante una tabla de verdad haciendo *Q*=1 (**ON**) en los casos donde las entradas sean A = B = C = D = 1. Para los otros casos *Q* se vuelve 0. En la figura 3.14d) se presenta la tabla resultante:

Figura 3.14

a) Circuito original con símbolos estándar NOR; **b)** representación equivalente donde Q es activa en ALTO; **c)** representación equivalente donde Q es activa en BAJO; **(d)** Tabla de verdad resultante, donde 0=OFF y 1=ON.



(d)

A	*B*	*C*	*D*	*Q*
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
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	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

La figura 3.14c) se obtiene con el diagrama original reemplazando las compuertas NOR Nº1 y Nº2 por sus equivalentes alternos. En esta representación equivalente *Q* se toma de una compuerta NOR con su salida en activa BAJA. Por lo que decimos que *Q* pasa a estado **OFF** cuando X = **1**o Y = **1(o ambas).** Debido a que X y Y son salidas en activo ALTO, podemos establecer que X pasa al estado **ON** sólo cuando A o B sean **0 (o ambas)**, y Y pasa al estado **ON** sólo cuando C o D sea **0(o ambas).**Deduciendo todo lo dicho podemos describir la operación del circuito de la siguiente forma:

"La salida Q pasará a estado OFF siempre y cuando A, B, C y D no sean 1".Esta descripción la podemos interpretar mediante una tabla de verdad haciendo *Q*=0 (**OFF**) en los casos donde las entradas no sean A = B = C = D = 1. En la figura 3.14d) se presenta la tabla resultante que es la misma para el circuito de la figura 3.14b).

3.14.1 ¿Qué diagrama de circuito lógico debo usar?

La repuesta depende del tipo de función que se realice a través de la salida del circuito lógico. Si el circuito provoca alguna acción (como el encender un motor o foco, etc.) cuando Q= 1, entonces decimos que Q es activa en ALTO (ON) y se deberá optar por el uso del diagrama del circuito de la figura 3.14b). Pero si en caso de que el propósito del circuito sea en realidad el de provocar una acción cuando Q=0, entonces Q será activa en BAJO (OFF) y por lo tanto se deberá usar el diagrama de la figura 3.14c).
Cabe recalcar que existirán momentos en las que ambos estados de salida sean usados para provocar distintas acciones y cualquiera pueda ser considerado como el estado activo. Para este tipo de situaciones se puede elegir libremente cualquier representación para el circuito.

3.14.2 Colocación de círculos (burbujas)

Para esto consideren la figura 3.14b) y fíjense que se estableció que los símbolos para las compuertas NOR Nº1 y Nº2 tuvieran salidas activas en BAJO, con el propósito de se acomodarán a las entradas activas en BAJO de la NOR Nº3. Ahora consideren la figura 3.14c), fíjense que se determinó que los símbolos para las compuertas NOR Nº1 y Nº2 tuvieran salidas activas en ALTO, con el propósito de se acomodarán a las entradas activas en ALTO de la NOR Nº3. A partir de todo esto podemos establecer la siguiente regla fundamental para la realización de esquemas de circuitos lógicos:

Siempre que se pueda, se deben elegir símbolos de compuertas de modo que las salidas estén conectadas a entradas con círculos, y las salidas sin círculos estén conectadas a entradas sin círculos.

3.14.3 Análisis de circuitos lógicos

Cuando se realiza un diagrama aplicando reglas que ya conocemos, es mucho más fácil para un ingeniero profesional o un novato, seguir la señal a través de todo el circuito y determinar las condiciones de entrada que se requieren para accionar la salida.

3.14.4 Niveles asegurados

Cuando una señal lógica está en estado activo ALTO se puede considerar que está *asegurada*. En cambio, cuando una señal lógica está en su estado activo en BAJO se considera que *no está asegurada*. En resumen, estos términos no son nada más y nada menos que sinónimos de activo e inactivo. Todo esto se puede enunciar de la siguiente forma:

Asegurado = activo (ALTO)

No asegurado = inactivo (BAJO)

3.14.5 Etiquetación de señales lógicas inactivas (BAJO)

Este tema habla de que es bastante común el uso de una barra sobrepuesta para etiquetar señales activas en BAJO. La barra sirve para indicar que una señal es activa en BAJO.

Ejercicios de repaso 3.14

1.- ¿Qué tienen que ver los símbolos alternos con la operación de un circuito?
2.- Cierto o falso: el tipo de representación de compuerta que se debe usar depende del tipo de función que se realice a través de la salida del circuito lógico.
3.-Explique en pocas palabras qué significa "asegurado" y "no asegurado"
4.- ¿Qué se utiliza para indicar el estado en BAJO de una señal?

3.15 Simbología lógica estándar IEEE/ANSI

El tipo de simbología que hemos utilizado a lo largo de este capítulo es la *tradicional* o estándar. Para estos símbolos tradicionales se usa una forma singular para cada compuerta. En el año de 1984 se realizó la introducción de una nueva norma para símbolos lógicos llamada ***Norma IEEE/ANSI 91-1984*** para símbolos lógicos. Esta variedad de nuevos símbolos están siendo aceptados por varias compañías de electrónica y diseñadores de CIs, también han empezado a hacer apariciones en las obras literarias que estos publican. La diferencia primordial en el nuevo estándar es que envés de

emplear distintos símbolos acude a símbolos con forma rectangular para todos los dispositivos. Se utiliza un sistema de notación especial para mostrar cómo las salidas dependen de las entradas.
-Las características de estos símbolos son:

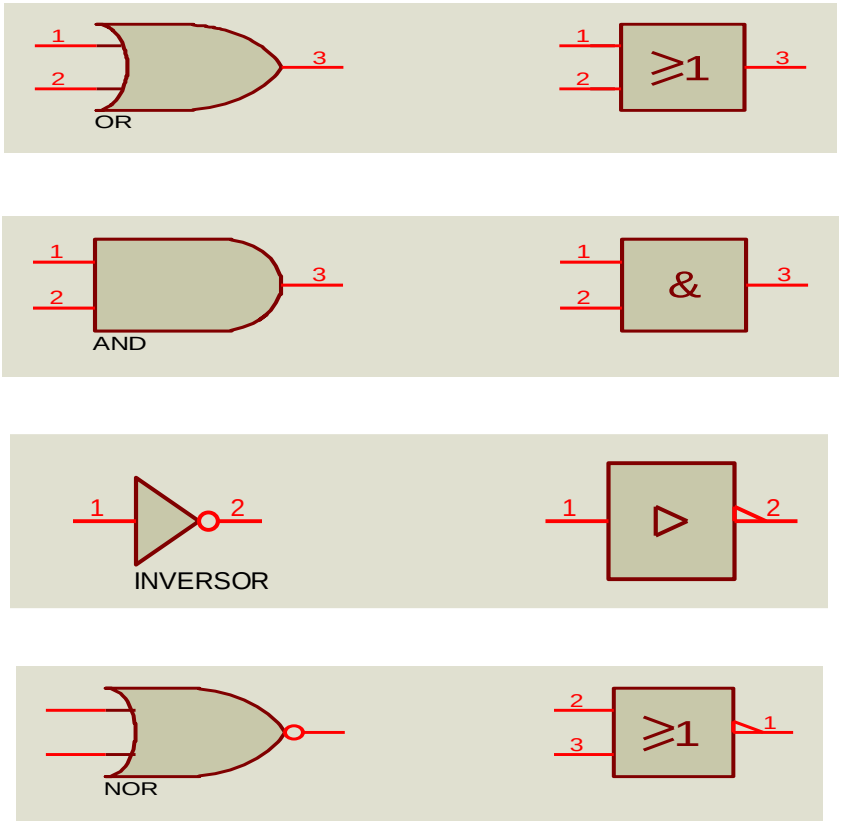
1) Estos nuevos símbolos usan un pequeño triangulo recto envés de una burbuja. Al igual que la burbuja, el triángulo denota una inversión. La presencia del triángulo también denota si una entrada o salida es activa en ALTO o en BAJO.

2) La relación entre las entradas y la salida es definida por una notación única dentro d cada símbolo rectangular. El número "1" dentro del símbolo del INVERSOR (NOT) representa un dispositivo de una sola entrada; el triángulo de la salida denota que éste irá al estado activo en BAJO si la entrada está en activo ALTO. El símbolo "&" dentro del símbolo AND denota que la salida estará en estado activo en ALTO si todas las entradas también están en activo ALTO. El símbolo "≥" dentro de la compuerta OR denota que la salida pasará a su estado activo (ALTO) si una o varias entradas están en estado activo (ALTO).

3) Los símbolos con forma rectangular para las compuertas NAND y NOR son iguales para las compuertas AND y OR, respectivamente, con la agregación de un pequeño triangulo para la inversión en la salida.
-Observe los ejemplos que se muestran en la siguiente figura:

Figura 3.15

Simbología estándar: tradicional y Norma IEEE/ANSI



3.15.1 Simbología tradicional o IEEE/ANSI ¿Cuál es mejor?

La Norma IEEE/ANSI sigue sin ser aceptado completamente, pero es probable que estén presentes en algunos esquemas de equipos de fabricación actuales. La mayoría de libros con información de CIs incluyen simbología tradicional o IEEE/ANSI (o ambos), y es existe la posibilidad de que algún día la nueva norma pueda tener un uso extenso. Por lo tanto, a pesar de que en este libro usaremos símbolos tradicionales en casi todos los diagramas de circuitos, se mostrará y determinará el símbolo IEEE/ANSI para cada nuevo dispositivo lógico a medida que sea introducido. De esta manera, se logrará que ustedes se acostumbren al uso de esta nueva norma.

Ejercicios de repaso 3.15

1.- ¿En qué año se introdujo la Norma IEEE/ANSI?
2.- ¿Qué es lo que diferencia a este nuevo estándar de los símbolos tradicionales?
3.- Escriba una característica de la Norma IEEE/ANSI.
4.- Cierto o falso: El símbolo "&" dentro del símbolo AND denota que la salida pasará a su estado activo (ALTO) si una o varias entradas están en estado activo (ALTO).

SOLUCIONARIO

Ejercicios de repaso 3.1

1.-Denominamos variable Booleana a una cantidad que puede ser ocasionalmente igual a 0 o 1.
2.-El 0 y 1 booleanos no presentan números reales, sino más bien el estado de una variable de voltaje, o lo que conocemos como nivel lógico.
3.-
a) No es.
b) Sí es.
c) Sí es.
d) No es.

Ejercicios de repaso 3.2

1.-Una tabla de verdad es un medio para explicar el funcionamiento de un sistema digital
2.-El cero representa la ausencia de corriente y el 1 representa la presencia de tensión.
3.-La salida es 0.
4.- 2^6 = 64 entradas (32 ceros y 32 unos)

Ejercicios de repaso 3.3

1.-c) Q = A+B
2.-Falso.
3.-*"Basta con que una de sus salidas sea igual a uno para que su salida también será igual a uno".*
4.-En un circuito digital la compuerta lógica OR es aquella cuya salida es igual a la combinación OR de sus dos o tres entradas.

Ejercicios de repaso 3.4

1.-b) Q = A . B
2.-Falso.
3.-*"Basta con que una de sus salidas sea igual a cero para que su salida también será igual a cero".*
4.-En un circuito digital la compuerta lógica AND es aquella cuya salida es igual al producto AND de sus dos o tres entradas.

Ejercicios de repaso 3.5

1.-d) Ninguna de las anteriores.
2.- Falso
3.-Sirve para representar la operación NOT.

Ejercicios de repaso 3.6

1.-$\overline{\overline{(A+B)(C).(D)}} = Q$

2.- $\overline{\overline{A+B+C}} = Q$

Ejercicios de repaso 3.7

1.-A=1, B=0 y C=0.

$\overline{\overline{(A+B).C}} = Q$ (se eliminan las barras dobles)
(1+0).0 = Q
(1)(0) = Q
0 = Q
2.-Se puede obtener el valor que corresponde a la salida si sustituimos los valores de las variables en la expresión y realizamos la operación.
3.- Cierto.

Ejercicios de repaso 3.8

1.-Necesitaría una compuerta AND de tres entradas.
2.-A + B + C

Ejercicios de repaso 3.9

1.-Una compuerta NOR es como un circuito compacto compuesto de una compuerta OR seguida de una NOT en su salida.
2.-Una compuerta AND es como un circuito compacto compuesto de una compuerta AND seguida de una NOT en su salida.

Ejercicios de repaso 3.10

1.-Son las reglas que nos ayudarán en la simplificación de expresiones y circuitos lógicos.
2.-El **primer teorema** nos dice que si cualquier variable se opera con AND y con un 0 el resultado deberá ser 0 debido a que todo número multiplicado por 0 es 0. **(x . 0 = 0)**
3.- Cierto.
4.-x + xy = x

Ejercicios de repaso 3.11

1.-El primer teorema nos dice que la inversa de una suma OR es equivalente al producto de sus inversas.
2.-El segundo teorema nos dice que el inverso de una multiplicación AND es equivalente a la suma de sus inversas.

Ejercicios de repaso 3.13

1.- Cierto.
2.-Una línea es activa en ALTO cuando una línea de entrada o salida en un símbolo de un circuito no tiene burbuja.
3.-Una línea es activa en BAJO cuando una línea de entrada o salida en un símbolo de un circuito tiene burbuja.

Ejercicios de repaso 3.14

1.-En que el correcto uso de los símbolos alternos de compuertas en el diagrama del circuito puede hacer que la operación del circuito sea menos compleja.
2.- Cierto.
3.-Asegurado = activo (ALTO)
No asegurado = inactivo (BAJO)

4.-La barra sirve para indicar que una señal es activa en BAJO.

Ejercicios de repaso 3.15

1.-En el año 1984
2.-La diferencia primordial en el nuevo estándar es que envés de emplear distintos símbolos acude a símbolos con forma rectangular para todos los dispositivos.
3.-Estos nuevos símbolos usan un pequeño triangulo recto envés de una burbuja.
4.-Falso.

CAPITULO N°4

Circuitos lógicos Combinacionales

♦ TEMARIO:

4.1 Suma de productos.
4.2 Circuitos lógicos simplificados.
4.3 Simplificación Algebraica.
4.4 Diseño de Circuitos lógicos combinacionales.
4.5 Mapa de Karnaugh.

♦OBJETIVOS:

El objetivo principal de este libro es la de proporcionar la información necesaria a todos los estudiantes, que estén iniciando en la especialidad de electrónica, para que sean capaces de:

♦Transformar una expresión lógica en una suma de productos.

♦Aplicar el procedimiento para simplificar una expresión de suma de productos a su forma más sencilla.

♦Manejar el álgebra de Boole y el Mapa de Karnaugh como principal herramienta en la simplificación de circuitos lógicos.

♦Explicar la operación de los circuitos lógicos exclusivos OR y NOR.

♦Diseñar circuitos sencillos sin la necesidad de una tabla de Verdad.

♦Accionar circuitos de ENABLE.

♦Mencionar las características de las familias de CIs TTL y CMOS.

♦Utilizar las reglas básicas para la localización de fallas en sistemas digitales.

♦Citar la idea general de los dispositivos lógicos (PLD).

♦Describir el procedimiento que implica la programación de un PLD para realizar una función lógica combinatoria menos compleja.

♦Acudir al manual de usuario de CPUL para conseguir información necesaria para realizar un experimento de programación sencilla en el laboratorio.

♦INTRODUCCCION:

En el capítulo anterior nos enfocamos en el estudio y el análisis de las distintas operaciones de cada compuerta

lógica básica y utilizamos el álgebra booleana para describir circuitos constituidos por combinaciones de compuertas lógicas. A esta clase de circuitos la podemos clasificar como circuitos lógicos *combinacionales* porque, en cualquier ocasión, el nivel lógico de la salida depende de la combinación que se dé entre los niveles lógicos presentes en las entradas.

En este capítulo 4 seguiremos estudiando los circuitos combinacionales. Para empezar, nos centraremos en la simplificación de circuitos lógicos, para ello utilizaremos dos métodos: los teoremas del álgebra booleana y una técnica de mapeo. También, analizaremos las técnicas simples para el diseño de circuitos lógicos combinacionales y para complacer un conjunto de requisitos. Un estudio a fondo y completo del diseño de circuitos lógicos no forma parte de nuestros objetivos, pero los métodos que se muestran nos darán una idea sobre el diseño lógico.

Este capítulo estará dedicado principalmente a la localización y detección de fallas que se presenten en los circuitos combinacionales. La primera explicación sobre la localización de fallas comenzará a desarrollar el tipo de capacidades deductivas que son requeridas para que la localización de fallas sea realizada efectivamente. Para que este material sea lo más práctico posible, primeramente veremos algunas características básicas de los CIs de compuertas lógicas en las familias TTL y CMOS, acompañados de una descripción de los tipos de fallas en CIs digitales más comunes.

En el último tema de este capítulo incluiremos los conceptos básicos de los dispositivos lógicos programables;

es decir, CI cuya circuitería interna puede ser alterada por el usuario con el fin de realizar distintas operaciones lógicas. Este material opcional tendrá el propósito de ofrecer a todos los lectores un buen comienzo, de modo que todos puedan dirigirse al laboratorio para hacer experimentos basados en la programación, siempre y cuando dispongan de equipo con software.

4.1 Suma de productos

Los métodos de simplificación y diseño de circuitos que vamos estudiar necesitan que la expresión lógica deba estar en una forma de *suma de productos*. Algunos ejemplos de suma de productos son:

a) $XYZ+\bar{X}Y\bar{Z}$
b) $XY+X\bar{Y}Z+\bar{A}B + C$
c) $\bar{A}B + C\bar{D} + EFG + GK + \bar{H}I + X$

Cada expresión en forma de suma de productos cuenta con dos o varios productos (AND) que son operados con OR. Cada producto cuenta cuneta con una o más variables que se presentan de manera individual, tanto en forma complementada como no complementaria. Por ejemplo, en la expresión en forma de suma de productos ABC +$\bar{X}Y\bar{Z}$, el 1er producto se conforma de las variables A, B, y C en sus formas no complementadas (sin inversor). El 2do producto contiene X y Z en sus formas complementadas (con inversor). Observen que en una expresión de suma de productos una barra inversora no puede estar sobrepuesta en más de una variable en un término, eso significa que nunca se podrá tener $\overline{XYZ}$ o $\overline{ABC}$.

4.1.1 Producto de sumas

El **producto de sumas** otra forma de expresiones lógicas para el diseño de circuitos lógicos, esta consiste en dos o más sumas (OR) que son operados con AND. Cada suma se conforma de una o más variables en forma complementada o sin complementar. Algunos ejemplos de expresiones de suma de productos son:

a)(A + B + C).(X + Z)
b) (A + B).(C + D)Z
c) (A + D + C).(B + C).(A + B)

En este libro haremos uso de métodos de simplificación y diseño de circuitos, los cuales se basan en la forma de suma de productos o SOP (por sus siglas en inglés), eso significa que no trabajaremos mucho con el producto de sumas o POS (por sus siglas en inglés). Aunque habrá veces en las que se presentarán circuitos lógicos con esta estructura peculiar.

Ejercicios de repaso 4.1

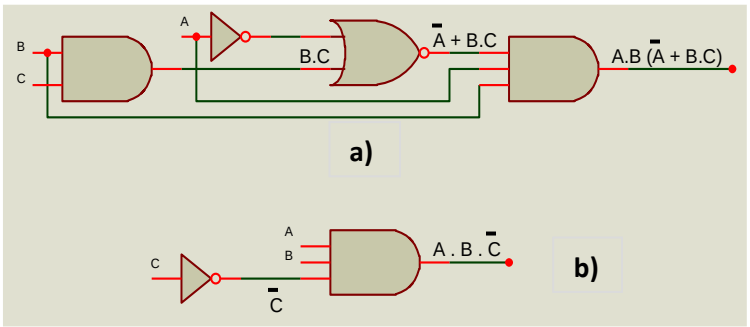
1.-Indique cuál de las siguientes expresiones está en forma POS:
a) (A + B + D)C
b) A + XY + Z
c) (B + C).(A + D)
d)BDC + $\bar{A}$B
2.- Indique cuál de las siguientes expresiones está en forma SOP:
a) (X + Y)Z
b) STR + AB + C
c) PS + Q + ZXY
d)CB(A +D)

4.2 Circuitos lógicos simplificados

Cuando ya tengamos la expresión para un circuito lógico, podemos simplificarla con el fin de que contenga menos términos. La nueva expresión puede usarse para ejecutar un circuito que sea igual al circuito original, pero que contenga menos compuertas.
Para ilustrar lo dicho anteriormente, se presenta el circuito de la figura 4.1a), el cual se puede reducir con el objetivo de crear el circuito de la figura 4.1b). Como los dos circuitos realizan la misma lógica, es un hecho que el circuito menos complejo es el mejor porque no contiene muchas compuertas, por lo tanto, tendrá un espacio físico más pequeño y será más barato que el original. Además, la fidelidad del circuito mejorará debido a la presencia de menos interconexiones que puedan ser fallas de circuitos.
En los próximos temas estudiaremos dos métodos de simplificación de circuitos lógicos. Uno de los métodos implica el uso de teoremas del álgebra de Boole y, como se verá, depende principalmente de la creatividad y los conocimientos del diseñador. El otro método el del *mapa de Karnaugh* es una aproximación sistemática, paso a paso. Varios profesores tal vez quieran saltarse este último método porque es algo mecánico y es probable que no aporte nada para mejorar la comprensión del álgebra de Boole.

Figura 4.1

*"A menudo es posible simplificar un circuito lógico como el de la parte **a)** para producir un funcionamiento más eficiente, como se muestra en **b)**". [1]*



[1] Diagrama tomado de la figura 4.1 del libro "Sistemas digitales-Principios y aplicaciones" Octava edición.

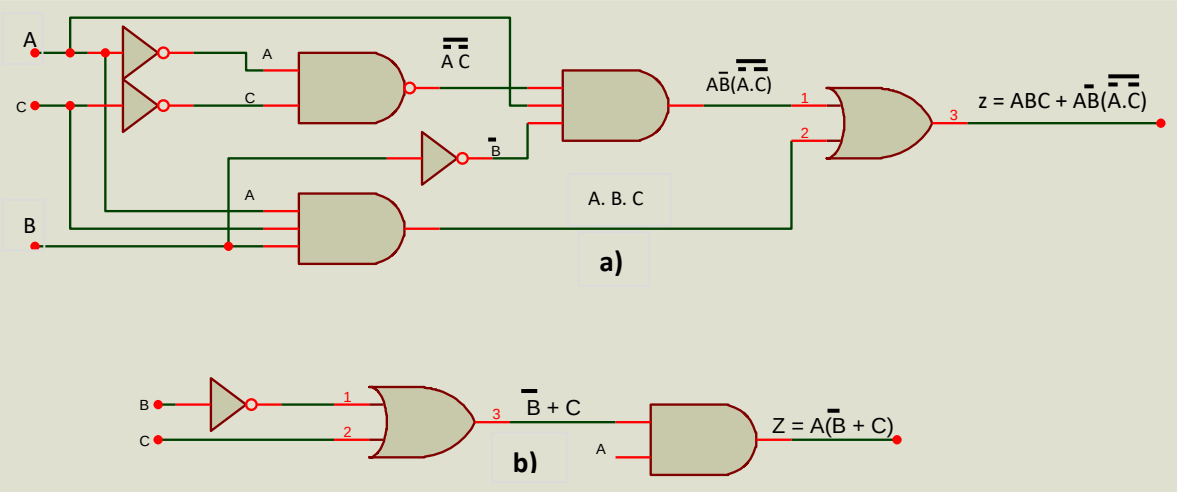
4.3 Simplificación Algebraica

Los teoremas que estudiamos en el capítulo 3 pueden ayudarnos a simplificar la expresión de un circuito lógico. Pero hay veces en las que no se sabe que teorema es el que se debe emplear para conseguir el resultado menos complejo. Además, no existe una forma factible que garantice si la expresión simplificada está en su forma más sencilla o si hay la posibilidad de simplificar aún más. De esta manera, frecuentemente la simplificación se vuelve un proceso de intento y fracaso. Sin embargo, la práctica puede hacer que cualquiera obtenga mejores resultados.
Los ejemplos que se presentarán a continuación ilustran muchas de las formas en que los teoremas booleanos pueden ser aplicados para reducir una expresión. Todos los ejemplos comparten una característica en común, y es que todos contienen dos pasos importantes:

1. *La expresión original debe estar en forma SOP mediante el uso de los teoremas DeMorgan y la multiplicación de términos.*
2. *Cuando la expresión original esté en forma SOP, los términos del producto se revisan para comprobar si existen factores semejantes, y factorizamos donde sea posible. Es probable que ésta nos dé como resultado la erradicación de varios términos.*

Figura 4.2

Ejemplo 4.1:*"Simplifique el circuito lógico que se muestra en la figura 4.2a)". [2]*



[2] Ejemplo tomado de la figura 4.2 del libro "Sistemas digitales-Principios y aplicaciones" Octava edición.

Procedimiento*: Primeramente determinamos la expresión de la salida empleando el método del tema 3.6 (Descripción algebraica de los circuitos lógicos). Obtenemos el siguiente resultado:*

$$z = ABC + A\bar{B}\,(\overline{\bar{A}.\bar{C}})$$

-Cuando ya tengamos determinada la expresión, es recomendable descomponer todas las barras inversoras, que sean posibles, con la ayuda de los teoremas de DeMorgan y después proseguir con la multiplicación de todos los términos.

$$\begin{aligned} z &= ABC + A\bar{B}(\bar{A} + \bar{\bar{C}}) \longrightarrow \textbf{(Teorema Nº 17)} \\ &= ABC + A\bar{B}(A + C) \longrightarrow \textbf{(Eliminar doble inversión)} \\ &= ABC + A\bar{B}A + A\bar{B}C \longrightarrow \textbf{(Multiplicar)} \\ &= ABC + A\bar{B} + A\bar{B}C \longrightarrow \textbf{(A . A = A)} \end{aligned}$$

-Una vez que tengamos la expresión en forma SOP, buscamos variables semejantes para simplificarlas. El 1er y 3er términos del anterior párrafo tienen AC como factor común, los cuales podemos simplificar:

$$z = AC(B + \bar{B}) + A\bar{B}$$

-Sabemos que $B + \bar{B} = 1$, entonces:

$$z = AC(1) + A\bar{B}$$

$$= AC + A\bar{B}$$

-Finalmente podemos factorizar **A**, lo que resulta en:

$$z = A(C + \bar{B})$$

Como se ve, este resultado ya no se puede simplificar más. Su implementación en circuito está presentado en la **figura 4.2b)**. Obviamente el circuito **b)** es más reducido y simple que el original en **a)**.

Ejercicios de repaso 4.3

1.- Escriba los pasos para realizar una simplificación algebraica.

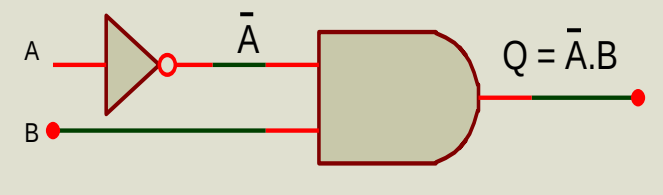
4.4 Diseño de Circuitos lógicos combinacionales

Cuando el valor de la salida está determinado para todas las condiciones existentes de entrada, los resultados se pueden representar en una tabla de verdad. Eso significa que la expresión booleana para cualquier circuito se puede derivar a partir de una tabla de verdad. A modo de ejemplo, consideren la **figura 4.3b)**, ahí se puede observar una tabla de verdad

para un circuito de dos entradas y una salida. En la tabla se muestra que la salida está en 1 sólo cuando *A*=0 y *B*=1. Ahora, debemos encontrar que tipo de circuito lógico es la que produce la operación deseada. Es obvio que una posible solución es la que se muestra en la **figura 4.3a)**. Aquí se utiliza la compuerta AND con la entrada ***A*** negada, de manera que su expresión es ***Q* = *A.B*.** Resulta claro que *Q* es igual a 1 solamente cuando las entradas de la compuerta AND sean 1. En cambio para los otros valores de *A* y *B* la salida Q será igual a 0.

Figura 4.3

"Circuito que produce una salida 1 sólo para la condición A=0 y B=1". [3]



A	*B*	*Q*
0	0	**0**
0	1	**1**
1	0	**0**
1	1	**0**

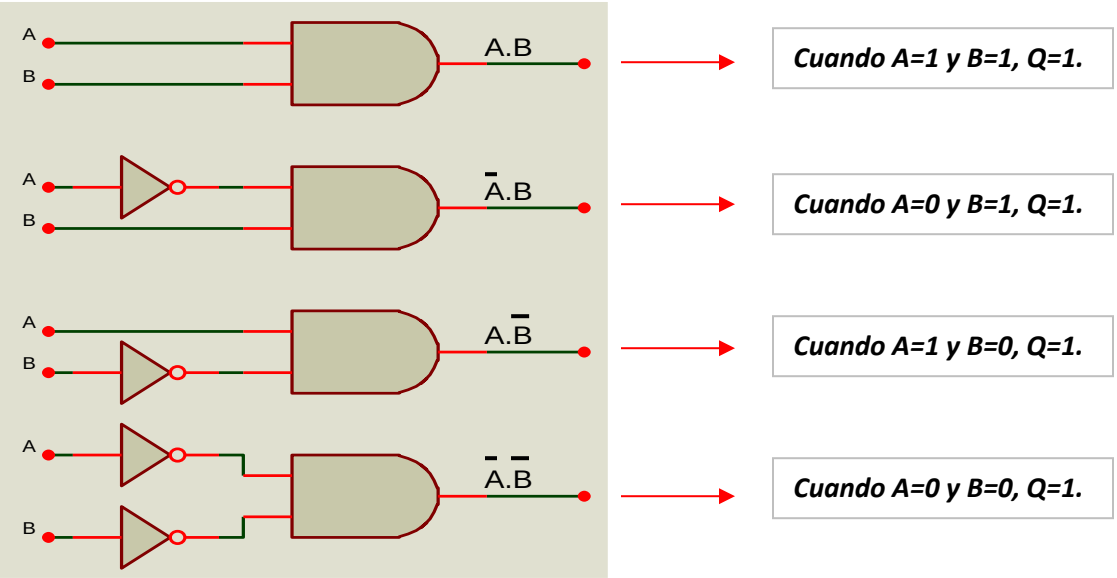
a) **b)**

[3] Ejemplo y figura tomados de la sección 4.4 del libro "Sistemas digitales-Principios y aplicaciones" Octava edición.

Podemos hacer una aproximación semejante para las demás condiciones de entrada. Por ejemplo si *Q* fuese 1 solamente cuando A=1 y B=0, el resultado sería un circuito conformado de una compuerta AND con la entrada ***B*** negada. Dicho de otra forma, para cualquiera de las cuatro posibles condiciones de entrada se puede producir una salida Q=1, usando una AND con entradas que puedan generar el producto deseado. En la **figura 4.4** se muestran los cuatro casos diferentes. Cada compuerta AND produce una salida igual a 1 sólo para una determinada condición de entrada, y la salida es igual a 0 para las demás condiciones. Es necesario fijarse en las salidas de la compuerta AND que pueden estar invertidas o no invertidas dependiendo del valor que tomen las variables para la condición determinada. Si una variable es cero para la condición específica, se niega (invierte) antes de ingresar a la compuerta AND.

Figura 4.4

Aplicación de compuertas AND con diferentes niveles de entradas para producir Q=1.



*Ahora observemos el siguiente caso presente en la **figura 4.5a),** aquí podemos ver una tabla de verdad que establece que Q=1 en dos casos distintos, donde A=0, B=1 y A=1, B=0. Para hacer esto primero recordemos que el producto $\bar{A}$.B producirá un 1 sólo cuando A=0, B=1, y el producto $\bar{A}$.B producirá un 1 cuando A=1, B=0. Ya que Q debe ser ON para toda condición, se deduce que los términos deben ser operados con una compuerta OR para generar una determinada salida. En la **figura 4.5b)** se muestra esta implementación, donde la salida se expresa de la siguiente forma: Q= $\bar{A}$.B + A.$\bar{B}$.*

*Veremos que en este ejemplo, se genera una multiplicación para cada caso en la tabla donde Q será igual a 1. Después las salidas de la compuerta AND son operadas con una compuerta OR para generar la salida total Q que será un 1 cuando cualquier término AND sea 1. Este procedimiento también puede ser aplicado en ejemplos de dos o varias entradas. Observen la siguiente **tabla** para un circuito de tres variables que presenta tres casos en los que Q es 1. Se muestra la multiplicación necesaria para cada caso. La expresión de la suma de productos para la salida Q la obtenemos operando con OR los tres términos AND.*

A	B	C	Q	
0	0	0	0	
0	0	1	0	
0	1	0	1	$=\bar{A}B\bar{C}$
0	1	1	1	$=\bar{A}BC$
1	0	0	0	
1	0	1	0	
1	1	0	0	
1	1	1	1	$= ABC$

$$Q = \bar{A}B\bar{C} + \bar{A}BC + ABC$$

Términos AND

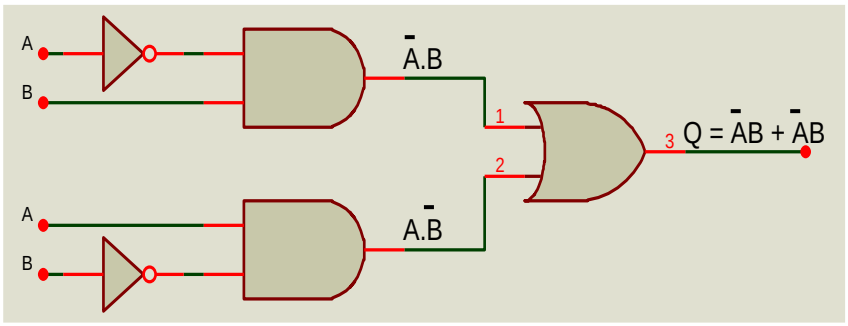
4.4.1 Procedimiento para el diseño de un circuito lógico

Cuando ya tengamos determinada la expresión de la salida a partir de la tabla de verdad en forma de suma de productos, puede ser sencillamente implementada empleando las compuertas OR, AND y NOT. Aunque, generalmente la expresión tiene la posibilidad de ser simplificada y por eso el circuito resulta ser más eficiente. El siguiente ejemplo que se muestra a continuación ilustra el procedimiento para realizar el diseño de un circuito lógico:

Figura 4.5

Todas las condiciones de las variables que genera una salida ON la implementa una AND aparte. Las salidas de las dos compuertas AND son operadas con OR para generar la salida definitiva.

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0



a) **b)**

EJEMPLO *[4]*: Diseñar un circuito lógico de tres entradas, cuya salida sea 1 sólo cuando casi todas las entradas sean 1.

1º Establecemos la tabla de verdad:

Según el enunciado Q será igual a 1 sólo cuando dos o más entradas sean 1: para los demás casos Q será igual a 0. Por lo tanto la tabla resultante será:

A	B	C	Q	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	= $\bar{A}BC$
1	0	0	0	
1	0	1	1	= $A\bar{B}C$
1	1	0	1	= $AB\bar{C}$
1	1	1	1	= ABC

2º Escribimos los términos AND en todos los casos donde Q=1:

Se puede observar que solamente hay cuatro casos. Los términos AND están presentes al lado derecho de la tabla. Fíjense que cada término AND está conformado de una variable de entrada, ya sea con signo de inversión o sin él.

3º Determinamos la expresión en forma de suma de productos para la salida:

$$Q = \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC$$

4º Simplificamos la expresión:

Podemos realizar la simplificación de varias maneras. Posiblemente el método más rápido sea notar que el último término (*ABC*) tiene dos variables en común con cada uno de los otros términos. Por lo que podemos usar *ABC* para factorizarlo con cada uno de los otros.

-La expresión es restablecida con el término *ABC* haciendo presencia tres veces:

$$\mathbf{Q =} \bar{A}BC \mathbf{+ ABC +} A\bar{B}C \mathbf{+ ABC +} AB\bar{C} \mathbf{+ ABC}$$

-Factorizamos los pares de términos apropiados

$$\mathbf{Q =} BC(\bar{A} + A) + AC(\bar{B} + B) + AB(\bar{C} + C)$$

- Como $(\bar{x} + x = 1)$, por lo tanto tenemos:

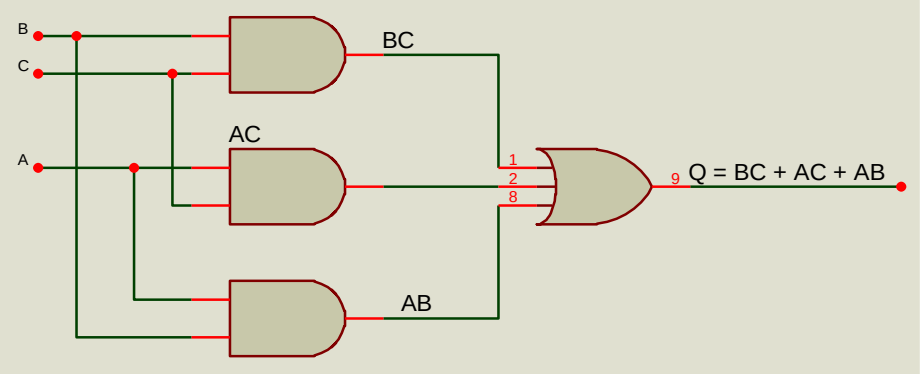
$$\mathbf{Q =} BC + AC + AB$$

5º Accionamos el circuito para la expresión definitiva:

La expresión se aplica en el siguiente circuito de la figura 4.6. Ya que la expresión está en forma SOP, el circuito lógico estará conformado de tres compuertas AND operando en una sola compuerta OR de tres entradas (porque son tres términos AND).

Figura 4.6

Ejemplo 1-4. [4]



[4] Ejemplo y figura tomados de la sección4.4 del libro "Sistemas digitales-Principios y aplicaciones" Octava edición.

4.5 Mapa de Karnaugh

El ***mapa de Karnaugh*** (conocido también como diagrama de ***Vetch***, abreviado como Mapa K o Mapa KV) es una herramienta útil para la simplificación de ecuaciones lógicas. Aunque este mapa es usado en problemas que implican cualquier cantidad de variables de entrada, su utilidad manual está restringida a funciones de hasta seis variables, para funciones con mayor número de variables es recomendable usar un programa especializado.

4.5.1 Representación del método de Mapa de Karnaugh

El ***mapa K*** es similar a una tabla de verdad, ya que presenta todos los valores posibles de las entradas y la salida resultante para cada valor. A continuación, en la figura 4.7 se mostrarán tres ejemplos de mapas K para dos, tres y cuatro entradas, junto con su respectiva tabla de verdad. Mediante estos ejemplos se ilustran los siguientes aspectos fundamentales:

a) La tabla de verdad proporciona el valor de la salida para cada combinación posible de las variables. El mapa K nos da una información similar pero en un formato distinto. Cada caso en la tabla de verdad corresponde a una casilla en el mapa K. Por ejemplo, en la tabla de verdad de la **figura 4.7a)** la condición A=0, B=0 corresponde a la casilla $\bar{A}\bar{B}$ en el mapa. Como la tabla de verdad indica que Q=1 para este caso, se agrega un 1 a la casilla $\bar{A}\bar{B}$ en el mapa K. De igual manera, la condición A=1, B=1 en la tabla corresponde a la casilla **AB**. Debido a que *Q* es igual a 1 para este caso, se agrega un 1 en la casilla **AB**. Las otras casillas faltantes son llenadas con ceros.

b) Las casillas del mapa K son marcadas de modo que las casillas horizontales adyacentes difieren solamente en una variable. Por ejemplo, la casilla superior izquierda en el mapa de cuatro variables es $\bar{A}\bar{B}\bar{C}\bar{D}$, mientras que la casilla siguiente a su derecha es $\bar{A}\bar{B}\bar{C}D$. De igual manera, las casillas verticales adyacentes sólo difieren en una variable. Por ejemplo, la casilla superior izquierda es $\bar{A}\bar{B}\bar{C}\bar{D}$, mientras que la casilla justo debajo de ésta es $\bar{A}B\bar{C}\bar{D}$**.** Fíjense que cada casilla en la fila superior se considera próxima a una casilla correspondiente en la fila inferior. Por ejemplo, la casilla $\bar{A}\bar{B}CD$ en la fila superior anexa a la casilla $A\bar{B}CD$ en la fila inferior, puesto que solamente difieren en la entrada "*A*". Ustedes podrían llegar a pensar que la parte superior del mapa fue enrollada para que de ese modo haga contacto con la parte inferior. De igual manera, todas las celdas de la columna de la extrema izquierda son cercanas a las casillas de la columna extrema derecha.

c) Con el objetivo de que las casillas verticales y horizontales cercanas difieran sólo en una variable, la marcación de arriba hacia abajo debe ser realizada siguiendo la siguiente orden: $\bar{A}\bar{B}$, $\bar{A}B$, **AB**, $A\bar{B}$. Lo mismo sucede con la marcación de izquierda a derecha: $\bar{C}\bar{D}$, $\bar{C}D$, **CD**, $C\bar{D}$.

d) Cuando un mapa K es finalmente llenado de ceros y unos se puede obtener la expresión de la suma de productos para la salida, trabajando con OR las casillas con unos. En el mapa de tres entradas (**Figura 4.7b)**) las casillas $\bar{A}\bar{B}\bar{C}$, $\bar{A}\bar{B}C$, $\bar{A}B\bar{C}$ y $AB\bar{C}$ contienen un 1, de modo que Q = $\bar{A}\bar{B}\bar{C}$+ $\bar{A}\bar{B}C$ + $\bar{A}B\bar{C}$ + **ABC.**

Figura 4.7

*"Mapa de Karnaugh y tablas de verdad para **a)** dos, **b)** tres y cuatro **c)** entradas". [5]*

a)

A	B	Q	
0	0	**1**	= $\bar{A}\bar{B}$
0	1	**0**	
1	0	**0**	
1	1	**1**	= AB

Q = AB + AB

	$\bar{B}$	B
$\bar{A}$	**1**	**0**
A	**0**	**1**

b)

A	B	C	Q	
0	0	0	**1**	= $\bar{A}\bar{B}\bar{C}$
0	0	1	**1**	= $\bar{A}\bar{B}C$
0	1	0	**1**	= $\bar{A}B\bar{C}$
0	1	1	**0**	
1	0	0	**0**	
1	0	1	**0**	
1	1	0	**1**	= $AB\bar{C}$
1	1	1	**0**	

Q = ABC + ABC + ABC +ABC

	$\bar{C}$	C
$\bar{A}\bar{B}$	**1**	**1**
$\bar{A}B$	**1**	**0**
AB	**1**	**0**
$A\bar{B}$	**0**	**0**

c)

A	B	C	D	Q	
0	0	0	0	**0**	
0	0	0	1	**1**	= $\bar{A}\bar{B}\bar{C}D$
0	0	1	0	**0**	
0	0	1	1	**0**	
0	1	0	0	**0**	
0	1	0	1	**1**	= $\bar{A}B\bar{C}D$
0	1	1	0	**0**	
0	1	1	1	**0**	
1	0	0	0	**0**	
1	0	0	1	**0**	
1	0	1	0	**0**	
1	0	1	1	**0**	
1	1	0	0	**0**	
1	1	0	1	**1**	= $AB\bar{C}D$
1	1	1	0	**0**	
1	1	1	1	**1**	= ABCD

Q = ABCD + ABCD
+ ABCD + ABCD

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	**0**	**1**	**0**	**0**
$\bar{A}B$	**0**	**1**	**0**	**0**
AB	**0**	**1**	**1**	**0**
$A\bar{B}$	**0**	**0**	**0**	**0**

[5] Mapas y tablas de verdad tomadas de la figura 4.11sección4.5 del libro "Sistemas digitales-Principios y aplicaciones" Octava edición.

4.5.2 Agrupamiento

La ecuación de la salida *Q* puede ser simplificada combinando apropiadamente en el mapa K aquellas casillas que contengan un 1. A este proceso para combinar estos unos lo conocemos como ***agrupamiento.***

Ejercicios de repaso 4.5

1.-Defina qué es el mapa de Karnaugh.

2.-Anote una característica del mapa de Karnaugh.

3.-Cierto o falso: el mapa de Karnaugh es conocido también como diagrama de barras.

SOLUCIONARIO

Ejercicios de repaso 4.1

1.-

a) Sí es.
b) No es.
c) Sí es.
d) No es.

2.-

a) No es.
b) Sí es.
c) Sí es.
d) No es.

Ejercicios de repaso 4.3

Paso 1. *La expresión original debe estar en forma SOP mediante el uso de los teoremas DeMorgan y la multiplicación de términos.*

Paso 2. *Cuando la expresión original esté en forma SOP, los términos del producto se revisan para comprobar si existen factores semejantes, y factorizamos donde sea posible. Es probable que ésta nos dé como resultado la erradicación de varios términos.*

Ejercicios de repaso 4.5

1.- El ***mapa de Karnaugh*** es una herramienta útil para la simplificación de ecuaciones lógicas.

2.- El ***mapa K*** es similar a una tabla de verdad, ya que presenta todos los valores posibles de las entradas y la salida resultante para cada valor.

3.-Falso