TEMA 9: *Lógica De Circuitos. Circuitos Combinacionales Y Secuenciales*

Autor: Javier Sánchez Fernández

Esquema:

1.- INTRODUCCION

A finales del siglo XIX un matemático inglés llamado George Boole desarrolló unas reglas de cálculo para aplicar a la lógica deductiva. En 1939, Claude Shanon estableció la interrelación entre el álgebra de Boole y los circuitos de conmutación, desarrolló circuitos que ejecutaban las operaciones booleanas elementales, y demostró que la combinación de circuitos podía representar operaciones aritméticas y lógicas complejas. Además demostró que el álgebra de Boole se podía utilizar para simplificar circuitos conmutadores. A partir de entonces el álgebra de Boole se ha utilizado para el diseño de los circuitos digitales de los ordenadores reduciendo el tamaño de estos.

Los circuitos digitales se pueden clasificar en circuitos combinacionales y circuitos secuenciales. En el tema se realiza una pequeña descripción de los principales circuitos elementales, que se utilizaran posteriormente

para el desarrollo de circuitos digitales más complejos, como pueden ser una Unidad Aritmética Lógica o una Unidad de control.

2.- ÁLGEBRA DE BOOLE

El álgebra de Boole es un conjunto cualquiera A en el que se han definido dos operaciones binarias denominadas suma lógica (+), también conocida como **OR**, y producto lógico (·), también conocido como **AND**, y una operación unitaria denominada complemento ($a \rightarrow \bar{a}$), también conocida como **NOT**. Se dice que $(A, +, \cdot, \bar{})$ es un álgebra de Boole si cumple las siguientes propiedades:

- El conjunto **A es cerrado para las dos operaciones**, es decir, $\forall$ a, b $\in$ A:

  $a + b \in A$
  $a \cdot b \in A$

- **Conmutativa:** $\forall$ a, b $\in$ A:

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

- **Identidad:** Existencia el elemento identidad para las dos operaciones (para la + el elemento identidad es el 0 y para el · es el 1), es decir, $\forall$ a $\in$ A:

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

- **Distributiva:** Cada una de las operaciones es distributiva en relación con la otra, es decir, $\forall$ a, b, c $\in$ A:

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

- **Complementario:** Todo elemento de A tiene su complementario que verifica:

  $\forall a \in A \quad \exists \bar{a} \qquad a + \bar{a} = 1$
  $\forall a \in A \quad \exists \bar{a} \qquad a \cdot \bar{a} = 0$

2.1.-Teoremas

1. El teorema de la dualidad

Dada una expresión booleana podemos pasar a su dual simplemente intercambiando entre sí el (+) por (·) y el (0) por el (1).

2. El teorema de idempotencia

$\forall$ a $\in$ A **a + a = a**
$\forall$ a $\in$ A **a · a = a** Propiedad dual

3. Teorema de los elementos dominantes

$\forall$ a $\in$ A **a + 1 = 1**
$\forall$ a $\in$ A **a · 0 = 0** Propiedad dual

4. Teorema de Unicidad

$\forall$ a $\in$ A $\exists$ un único ā tal que **a + ā = 1**
$\forall$ a $\in$ A $\exists$ un único ā tal que **a · ā = 0** Propiedad dual

Si a + b = 1 entonces b = ā
Si a · b = 0 entonces b = ā

4. Teorema de absorción

$\forall$ a, b $\in$ A **a + a · b = a**
$\forall$ a, b $\in$ A **a (a + b) = a** Propiedad dual

5. La propiedad asociativa

$\forall$ a, b, c $\in$ A **(a + b) + c = a + (b + c)**
$\forall$ a, b, c $\in$ A **(a · b) c = a (b · c)** Propiedad dual

7. Teorema de la involución

Cualquier cosa doblemente negada es esa misma cosa.

$\forall$ a $\in$ A $\quad \overline{\overline{a}} = a$

8. Leyes de Morgan

Sirven para romper las negaciones que afectan a varios términos y nos permiten relacionar la operación lógica de la suma con el producto:

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

De todos los conjuntos que tienen estructura de álgebra de Boole, nos restringimos a aquel que tienen dos elementos {0,1}, esta álgebra recibe el nombre de **álgebra de conmutación.** Las tablas de verdad de los operadores:

Variables		Funciones	
a	b	a + b	a · b
0	0	0	0
0	1	1	0
1	0	1	0
1	1	1	1

a	ā
0	1
1	0

2.2.-Funciones Lógicas

Sea B un álgebra de conmutación, definimos una función lógica f:$B^n \rightarrow B$ como toda aplicación que hace corresponder a una n-tupla de elementos B^n un elemento de B. Una función booleana se puede representar indistintamente mediante su expresión algebraica o mediante su tabla de verdad.
Por ejemplo:

$$f(a,b,c) = a \cdot c + \overline{b} \cdot c$$

a	b	c	ac	$\overline{b}$c	f
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	0	0
0	1	1	0	0	0
1	0	0	0	0	0
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	1	0	1

El valor de una función lógica se haya mediante una tabla de verdad en la que aparecen los resultados de aplicar la función a todas las posibles combinaciones de las variables binarias de entrada.

2.2.a.- Funciones de Boole

Una función de Boole se puede representar mediante diversas expresiones equivalentes. Entre las múltiples formas que tenemos para representar una función destacan dos:

- **Forma disyuntiva o suma de productos:**

Formada por una suma de términos. Cada término compuesto por un producto de variables, complementadas o no, y en el que no tienen que aparecer todas las variables.

$$f(a,b,c) = \overline{a}\cdot b + a\cdot b\cdot\overline{c} + \overline{a}\cdot\overline{c}$$

- **Forma conjuntiva o producto de sumas:**

Formada por varios términos multiplicados, las variables dentro de cada término, complementadas o no, están sumadas. En cada término no tienen por qué aparecer todas las variables.

$$f(a,b,c) = (a+b)\cdot(a+c)\cdot(\overline{a}+b+c)$$

2.2.b.- Formas canónicas de una función lógica

Se define término canónico de una función lógica a todo producto (o suma) que contiene todas las variables lógicas, complementadas o no, de la función. Una función lógica está expresada en su forma canónica, si todos los términos que la componen son canónicos. Por ejemplo:

$$f(a,b,c) = a\cdot b\cdot\overline{c} + \overline{a}\cdot b\cdot\overline{c}$$

A los términos que son sumas de variables reciben el nombre de **Maxterms** y a los términos que son productos de variables reciben el nombre de **minterms**.

- **La forma canónica disyuntiva**

Es igual que la forma disyuntiva pero cada uno de los términos es un **minterm**, es decir, deben aparecer todas las variables de la función.

Ej. $f(a,b,c) = a\cdot b\cdot\overline{c} + \overline{a}\cdot b\cdot c + \overline{a}\cdot\overline{b}\cdot\overline{c}$

- **La forma canónica conjuntiva**

Es igual que la forma conjuntiva pero cada uno de los términos es un **Maxterm**, es decir, tienen que aparecer las tres variables.

$$\text{Ej. } f(a,b,c) = (a+b+\overline{c})\cdot(\overline{a}+b+\overline{c})\cdot(\overline{a}+\overline{b}+\overline{c})$$

2.2.c.- Teorema de Shannon

Cualquier función lógica de n variables se puede expresar como suma de **minterms**, siendo esta expresión única. Análogamente este teorema tiene su correspondiente dual, es decir, toda función lógica de n variables se puede expresar como producto de **Maxterms**.

$$f(a_0, a_1 \ldots, a_n) = \overline{a}_0\cdot\overline{a}_1\cdot\ldots\cdot\overline{a}_n f_0(0,0,..,0) + a_0\cdot\overline{a}_1\cdot\ldots\cdot\overline{a}_n f_1(1,0,\ldots,0) + \ldots + a_0\cdot a_1\cdot\ldots\cdot a_n f_n(1,1,\ldots,1)$$

Existe otra forma de representar esta función, si la variable a_i está complementada se le asigna un valor 0 y si no está complementada se le asigna un valor 1. Cada $a_0\cdot a_1\cdot\ldots a_n$ se sustituye por su valor y obtenemos un número decimal representado en binario que denotaremos por m_i. Además como cada f_0 (0,0, …, 0), f_1 (1,0…,0),…, f_n (1,…, 1) solo puede tomar dos valores (0 ó 1), la expresión lógica f (a_0, …, a_n) se puede expresar como suma de aquellos términos f_i cuyo valor sea 0

$$f(a_0,\ldots a_n) = \sum_{i=0}^{2^n-1} m_i\cdot f_i$$

El teorema dual se representaría de la siguiente forma:

$$f(a_0,\ldots a_n) = \prod_{i=0}^{2^n-1} (M_{2^n-1-i} + f_i)$$

Sea la siguiente función lógica representada por su tabla de verdad

abc	f
000	0
001	0
010	0
011	1
100	0
101	0
110	1
111	0

A partir de la tabla de verdad podemos obtener la representación en su forma canónica, tanto en **minterms** como en **Maxterms**:

- Con **minterms** solo aparecen aquellos términos en los que la función vale 1: $f = m_3 + m_6 = \Sigma\,(3, 6)$
- Con **Maxterms** solo aparecen aquellos términos en los que la función vale 0: $f = M_0 + M_1 + M_2 + M_4 + M_5 + M_7 = \Pi\,(0,1,2,4,5,7)$

Una función lógica expresada como suma de minterms se puede transformar en otra expresión lógica equivalente expresada como producto de Maxterms simplemente aplicando las leyes de Morgan. Cada minterm, tiene el siguiente Maxterm equivalente:

$$\overline{m_i} = M_{2^n-1-i}$$

Y análogamente, cada Maxterm tiene el siguiente minterm equivalente:

$$\overline{M_i} = m_{2^n-1-i}$$

Por ejemplo para n=3

$$\overline{m_6} = M_{2^3-1-6}$$
$$\overline{m_6} = M_1$$

Dada la función expresada como suma de minterms

$$f(a,b,c) = m_0 + m_1 + m_4 + m_6 = \Sigma\,(0,1,4,6)$$

La función complemento de esta sería aquella en que los términos que aparecen valen cero

$$\overline{f(a,b,c)} = m_2 + m_3 + m_5 + m_7$$

Aplicando las leyes de involución y de Morgan tendríamos:

$$\overline{\overline{f(a,b,c)}} = \overline{m_2 + m_3 + m_5 + m_7} = \overline{m_2} \cdot \overline{m_3} \cdot \overline{m_5} \cdot \overline{m_7} = M_5 \cdot M_4 \cdot M_2 \cdot M_0$$

$$f(a,b,c) = \Pi\,(0,2,4,5)$$

2.3.-Simplificación de las funciones lógicas

Las funciones lógicas como suma de minterms o productos de Maxterms se pueden simplificar de tal forma que aparezca el menor número de términos en la expresión y con el menor número de variables en ellos. Existen dos métodos para simplificar una función:

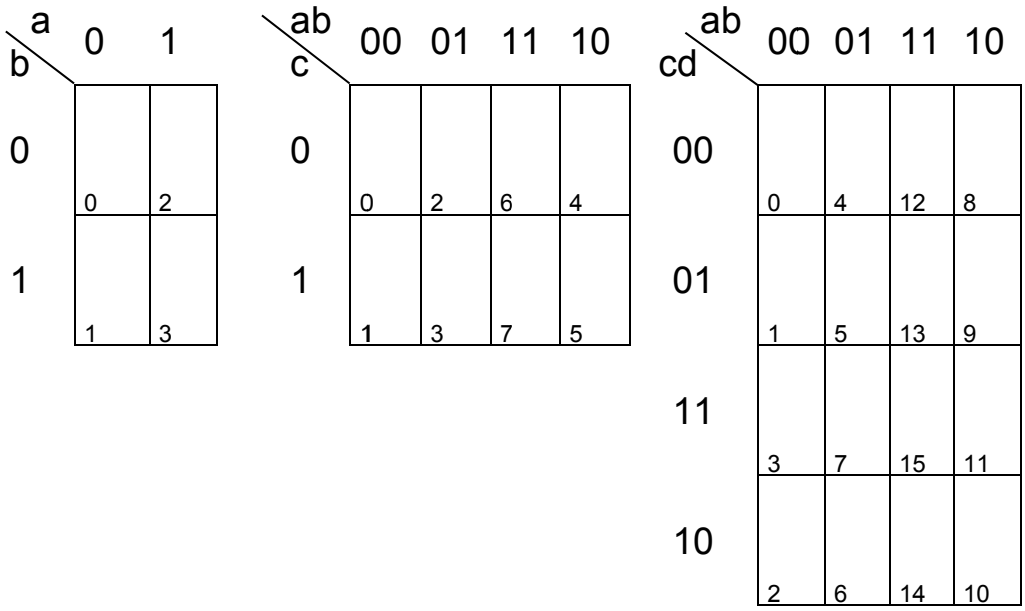
- Aplicando los teoremas del álgebra de Boole de forma sistemática.
- Mediante mapas de karnaugh

2.3.a.- Mapas de karnaugh

Son un método gráfico para simplificar funciones booleanas de hasta seis variables. Un mapa de Karnaugh es un mapa formado por una serie de casillas cuya propiedad más importante que tienen es que dos casillas en horizontal o en vertical difieren en una única variable.

El número de casillas de un mapa de Karnaugh es 2^n siendo n el número de variables.

Ej. De 2 variables. De 3 variables De 4 variables



Regla para numerar las celdas laterales:

- En dos celdas consecutivas sólo puede variar una variable. Por este motivo en lugar de numerarse así 00 01 10 11 se numera de la siguiente manera: 00 01 11 10

Regla para numerar las celdas interiores:

- Para numerar estas celdas cogemos las variables en el orden en que están y convertimos de binario a decimal.

a, b, c	decimal
000	0
010	2
110	6
100	4

a, b, c, d	decimal
0000	0
0100	4
1100	12
1000	8

Dentro de cada casilla del mapa se coloca el valor de la función que será 0 o 1. El número de cada celda coincide con el número de minterm y de Maxterm de la función. Un minterm y un Maxterm con el mismo número corresponden a la misma casilla (En la casilla 11 hay un m_{11} y un M_{11}).

Vamos a representar en un mapa de Karnaugh la siguiente función:

Ej. $f(a,b,c) = \overline{a}\cdot b\cdot\overline{c} + a\cdot b\cdot\overline{c} + \overline{a}\cdot b$

A esta función le falta en el último término una variable por lo que no está en forma canónica. Lo primero que debemos de hacer es pasarla a forma canónica.

$$f(a,b,c) = \overline{a}\cdot b\cdot\overline{c} + a\cdot b\cdot\overline{c} + \overline{a}\cdot b\cdot(c+\overline{c})$$
$$f(a,b,c) = \overline{a}\cdot b\cdot\overline{c} + a\cdot b\cdot\overline{c} + \overline{a}\cdot b\cdot c + \overline{a}\cdot b\cdot\overline{c}$$
$$f(a,b,c) = \overline{a}\cdot b\cdot\overline{c} + a\cdot b\cdot\overline{c} + \overline{a}\cdot b\cdot c$$

Después sustituimos cada variable por su valor y obtenemos la función como suma de minterm

$$f(a,b,c) = m_2 + m_3 + m_6 = \sum m(2,3,6)$$

El mapa de Karnaugh de esta función seria:

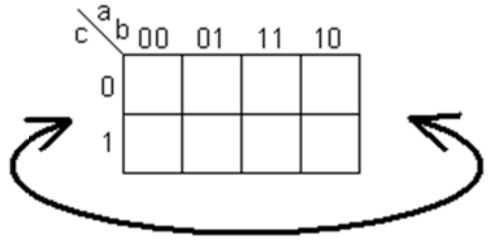
ab / c	00	01	11	10
0	0 0	1 2	1 6	0 4
1	0 1	1 3	0 7	0 5

2.3.b.- Algoritmo de simplificación

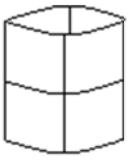
Para simplificar una función lógica debemos aplicar una serie de reglas:

- Paso 1. Hay que elegir si vamos a simplificar por minterms, es decir, (agrupando unos), o por Maxterms (agrupando ceros).
- Paso 2. Agrupamos las casillas, en grupos de 2^m casillas (siendo m = 0, 1, 2, 3, 4) lo más grande posible. Solo se pueden agrupar casillas si son contiguas tanto en horizontal, como en vertical pues son las que difieren en una única variable, pero no se pueden agrupar en diagonal. Hay que tener en cuenta que las casillas de la última fila son contiguas a las casillas de la primera fila y que las casillas de la última columna son contiguas a las de la primera columna. Gráficamente:

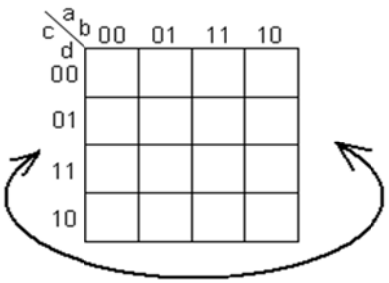
- Para tres variables:



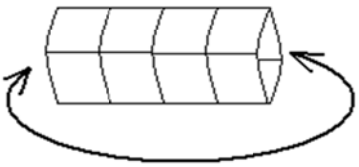
Doblar el mapa hasta formar un cilindro



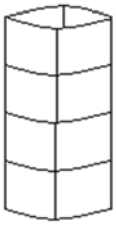
- Para cuatro variables:

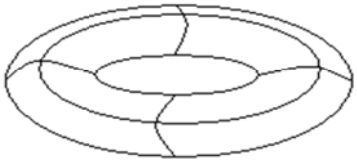


Primero doblar formando un cilindro



Segundo doblar formando un toroide (donut)



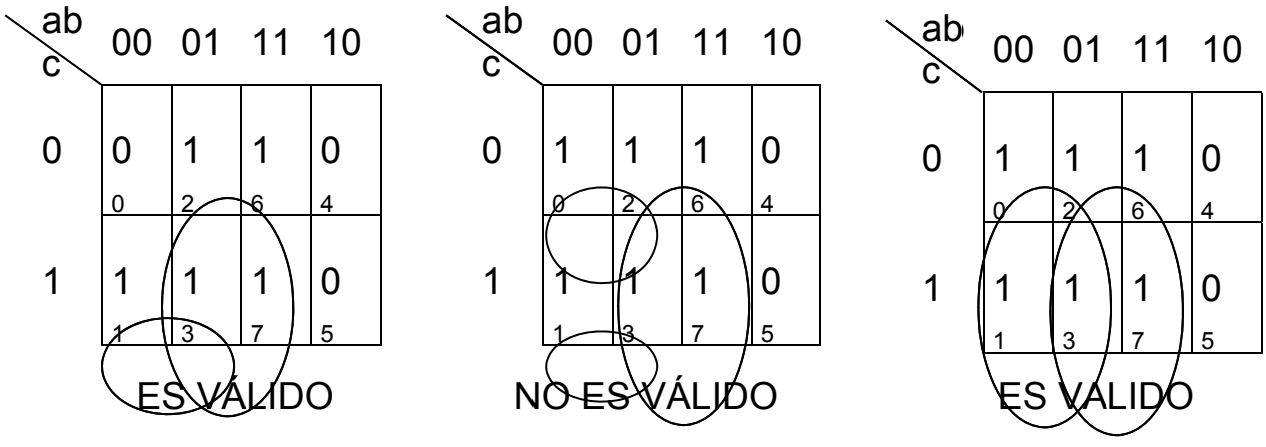


Nota: No se pueden hacer agrupaciones de 3 casillas.

- Paso 3. Una casilla puede estar en distintas agrupaciones (una de 8 por un lado y otra de 2 por otro).
- Paso 4. No puede quedar ningún término sin agrupar, pero hay que intentar hacer el menor número de agrupaciones posibles y que sean lo más grande posible.

Las casillas en un mapa de Karnaught se pueden agrupar de maneras diferentes y obtendremos resultados distintos. Para obtener un resultado mínimo, es necesario aplicar de manera sistemática las siguientes reglas:

- Paso 1: Empezar por aquellas agrupaciones de una casilla que no se puedan agrupar por otras.
- Paso 2: Buscar las casillas que sólo se pueden agrupar en grupos de dos y además de forma única examinando una por una las casillas no agrupadas. Si existen varias soluciones para una casilla dejarla pendiente (por ahora).
- Paso 3. Buscar aquellas casillas que sólo se pueden agrupar en grupos de cuatro y de forma única examinando una a una las casillas no agrupadas. Si hay varias soluciones para una casilla reservarla (esperar).
- Paso 4: Lo mismo pero para grupos de 8; grupos de 16; grupos de 32;
- Paso 5: Si quedan casillas sin agrupar combinarlas con otras agrupadas o no.
- Cada grupo representa un producto. Este está formado por las variables que no cambian de valor en dicho grupo. Si está a 1 la variable se escribe como esta y si está a 0 se complementa



No olvidar que hay que hacer siempre el menor número de agrupaciones y que sean éstas lo más grandes posibles.

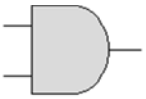
2.4.-Puertas lógicas

Los sistemas digitales solo tratan con funciones lógicas de dos variables, con n=2 existen 2^4 funciones lógicas diferentes. Las funciones lógicas más importante son:

- AND
- OR
- NOT
- NAND
- NOR
- X-OR

Función AND

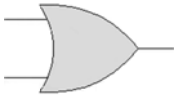
La función AND corresponde con la operación lógica producto. Solo es cierta cuando están activas las dos entradas de la puerta lógica. Gráficamente se representa:



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

Función OR

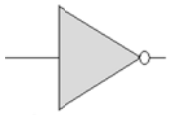
La función OR corresponde con la operación lógica suma y la función solo es falsa cuando las dos entradas de la puerta están inactivas. Gráficamente se representa:



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

Función NOT

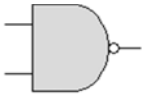
La función Not invierte la entrada. Gráficamente se representa:



a	f(a,b)
0	1
1	0

Función NAND

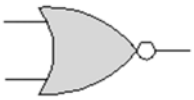
La función NAND es la función AND negada, la salida está activa si todas las entradas están activas. La tabla de verdad de esta función lógica y su representación gráfica es:



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

Función NOR

La función NOR es la función OR negada, la salida está activa si todas las entradas están inactivas. A continuación tenemos la tabla de verdad y su representación gráfica.

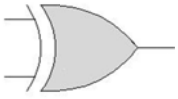


a	b	f(a,b)
0	0	1
0	1	0
1	0	0
1	1	0

Función OR-Exclusiva

La función OR-Exclusiva o función X-OR tiene activa la salida, siempre que una y solo una de las dos entradas este activa.

$$X = A \oplus B = A\overline{B} + \overline{A}B$$

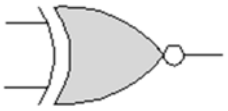


a	b	f(a,b)
0	0	0
0	1	1
1	0	1
1	1	0

Función XNOR

La función XNOR es la negación de la función X-OR, por tanto tiene activa la salida siempre que las dos entradas sean iguales, o las dos activas o las dos inactivas

$$X = A \otimes B = AB + \overline{A}\,\overline{B}$$



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

Definición

Un conjunto *C* de puertas lógicas se considera completo si cualquier función lógica de n variables se puede representar mediante puertas lógicas de este conjunto *C* y esto para todo n≥1

Teorema

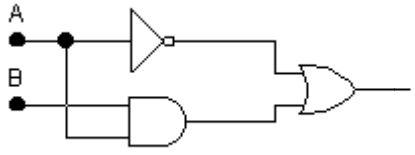
Cada uno de los siguientes conjuntos de puertas lógicas es completo

- NOT, AND
- NOT, OR
- NAND
- NOR

Por ejemplo, sea la siguiente función:

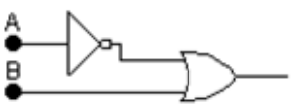
$$\overline{A} + A \cdot B$$

Se puede representar mediante las puertas lógicas NOT, AND y OR

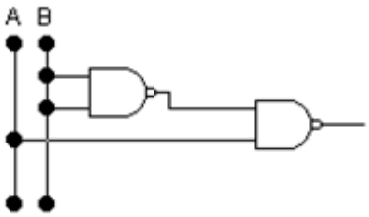


Pero si simplificamos la función, aplicando los teoremas del álgebra de Boole se puede representar utilizando solo puertas NOT y OR. Aplicando el principio de la dualidad también se representara mediante puertas NOT y AND

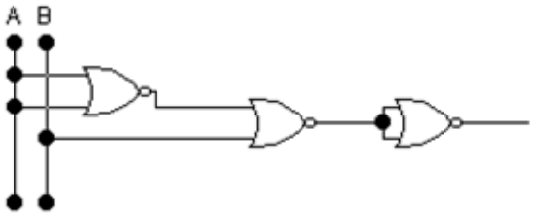
$$\overline{\overline{\overline{A} + AB}} = \overline{A \circ (\overline{A} + \overline{B})} = \overline{A\overline{A} + A\overline{B}} = \overline{A} + B$$



Y esta función se puede representar utilizando únicamente puertas lógicas NAND y NOR



NAND

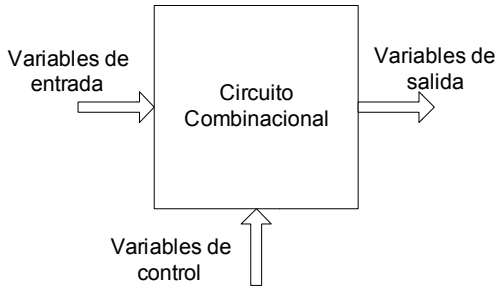


NOR

3.- CIRCUITOS COMBINACIONALES

Un circuito combinacional es todo circuito lógico cuyos valores de salida están completamente determinados por los valores de entrada. Los valores anteriores de la entrada no determinan en ningún caso el valor actual de la salida.

Lo más importante de un circuito combinacional es la función lógica que realiza, algunas de estas se implementan en circuitos integrados, que se utilizarán posteriormente para el desarrollo de sistemas digitales. Estos sistemas se utilizan principalmente para la transferencia de datos y para el procesamiento de datos.
Un circuito combinacional se puede representar de la siguiente forma:



Los elementos que componen un circuito son:

- Variables de entrada: Son las entradas al circuito combinacional, estas pueden estar en dos estados, activo e inactivo que asociaremos respectivamente con el 1 y con el 0 lógico.

- Variables de salida: Son las salidas del circuito combinacional, en ellas aparece el resultado de la operación lógica que realiza el circuito sobre las variables de entrada.
- Variables de control: Permiten modificar el funcionamiento de un circuito combinacional. Entre ellas podemos destacar:

    - De habilitación: Son aquellas que habilitan un circuito integrado. Si están inactivas el circuito no realizara su función
    - De selección: Son aquellas que determinan el modo de operación de un circuito combinacional, por ejemplo, en un circuito operacional diádico de tipo genérico, la operación a realizar se indica a través de estas variables de control.

Los principales circuitos combinacionales son:

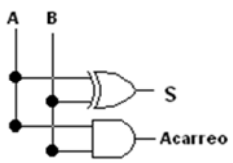
- Semisumador
- Codificador
- Decodificador
- Multiplexor
- Demultiplexor
- Comparadores

3.1.-Semisumador

Es un circuito combinacional que realiza la suma binaria, cuya tabla es la siguiente:

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 0 con 1 de acarreo

Este circuito tiene dos entradas y dos salidas, en una de las salidas obtenemos la suma de las variables de entrada y en la otra, el acarreo en caso de existir.



3.2.-Codificador

Un codificador es un circuito combinacional que tiene 2^n entradas y n salidas. Una única entrada determina los elementos de la salida. Se utiliza en los teclados donde al pulsar una tecla se activa una entrada y devuelve el código de la tecla pulsada. También este tipo de circuitos se utiliza como conversores de un código decimal a código binario natural o BCD.

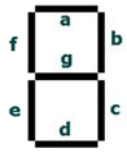
Otra aplicación son los codificadores con prioridad, donde unas entradas son más prioritarias que otras y cuando se activan varias entradas de manera simultánea atiende a la más prioritaria. Sirve por ejemplo, para conectar los diferentes elementos de un bus.

Un problema que plantea el codificador, es la ambigüedad que existe cuando todas las variables de salida son cero. Este estado puede venir determinado por dos situaciones, la entrada del elemento nulo o que el codificador este deshabilitado. Para evitar este problema se activan dos tipos de señales de control:

- Señal de salida de grupo Gs: El codificador está habilitado y hay señal de entrada
- Habilitación de salida Eo: El codificador está habilitado y no hay señal de entrada

3.3.-Decodificador

Un decodificador es un circuito combinacional con n entradas y 2^n salidas. Un ejemplo es el decodificador de siete segmentos, que sirve para representar un digito en una calculadora. El decodificador es un circuito que tiene cuatro entradas (representación de un nº decimal en BCD) y siete salidas, cada una determina cuál de los siete segmentos se deben iluminar para representar la respectiva entrada. En la siguiente tabla se puede ver qué segmentos tienen iluminarse para representar un número decimal



	A	B	C	D	E	F	G
0	1	1	1	1	1	1	0
1	0	1	1	0	0	0	0
2	1	1	0	1	1	0	1
3	1	1	1	1	0	0	1
4	0	1	1	0	0	1	1
5	1	0	1	1	0	1	1
6	1	0	1	1	1	1	1
7	1	1	1	0	0	0	0
8	1	1	1	1	1	1	1
9	1	1	1	1	0	0	1

Los decodificadores también se utilizan para poder direccionar una palabra en una memoria.

3.4.-Multiplexores

Son circuitos combinacionales que tienen n entradas, una única salida y varias entradas de selección. Sirven para redirigir una entrada procedente de varios sitios a un lugar común. Con las señales de control podemos seleccionar cualquier entrada y dirigirla hacia la salida.

Una de las aplicaciones típicas de un multiplexor es la conversión serie/paralelo. También sirve para implementar funciones lógicas, asignando a cada combinación de entrada, la correspondiente salida.

3.5.-Demultiplexor

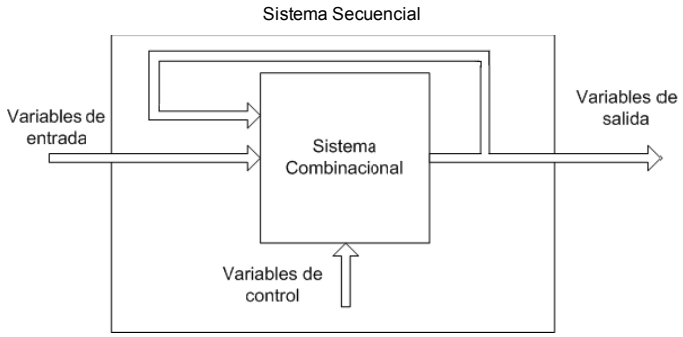
Son circuitos combinacionales que tienen una única entrada, m salidas y n entradas de selección. Hacen la operación inversa que un multiplexor, dada una entrada, el demultiplexor la envía a múltiples lugares. Su diseño se realiza a partir de decodificadores, pero variando el significado de las variables de entrada y de control. Las entradas del decodificador se utilizan como entradas de control en el demultiplexor y las entradas de control como señales de entrada.

3.6.-Comparadores

Son circuitos combinacionales que determinan la relación de igualdad, o desigualdad entre dos cantidades. La forma que el comparador realiza la operación es bit a bit desde el bit de mayor peso. Algunos circuitos comparadores tienen entradas auxiliares que permiten realizar comparaciones sucesivas y así poder comparar palabras de mayor tamaño.

4.- CIRCUITOS SECUENCIALES

Un circuito secuencial es un circuito lógico, en el que el estado de las variables de salida depende del estado de las variables de entrada y de los valores previos, se dice que el circuito secuencial tiene “memoria”. Un sistema secuencial se puede construir a partir de un sistema combinacional añadiéndole realimentación. Gráficamente se puede representar:



Los sistemas secuenciales pueden ser síncronos o asíncronos. En un *sistema asíncrono* el cambio de estado se puede producir en cualquier instante de tiempo, solo tiene en cuenta los cambios de estado de la entrada, mientras que en un *sistema síncrono* este cambio solo puede ocurrir en unos instantes de tiempo determinados por un reloj que emite una señal de sincronismo. En los ordenadores los circuitos secuenciales que se construyen son síncronos, ya que son más fiables y fáciles de construir que los asíncronos. Los circuitos secuenciales se utilizan principalmente en la unidad de control y en la unidad de memoria de cualquier ordenador.

Los principales circuitos secuenciales son:
- Biestables
- Registros
- Contadores

4.1.-Biestables

Un biestable es el elemento básico de la memoria, puede estar en dos estados que corresponden con el "0" y el "1" lógico. La característica principal de este dispositivo es que después de adquirir un estado, es capaz de mantenerlo de manera indefinida.

Los diferentes biestables que existen se pueden agrupar en síncronos y asíncronos, en función si son activados por una señal de reloj. Esta señal de reloj es prioritaria sobre las demás entradas síncronas. Aunque se produzca una alteración en dichas entradas, el biestable, no modificara su salida mientras que no sea activado por la entrada de reloj En un ordenador todo el sistema está gobernado por un reloj por lo que se usan biestables asíncronos.

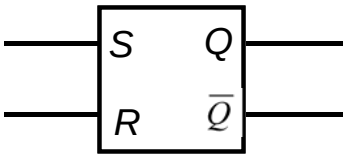
.

Existen diversos dispositivos de memoria:

- Biestable RS
- Biestable JK
- Biestable T
- Biestable D

4.1.a.- Biestable RS

Tiene dos entradas de control y dos salidas



- Entrada Reset: Cuando se activa esta entrada, la salida se pone a 0.
- Entrada de Set: Si se activa esta entrada, la salida se pone a 1.
- En un biestable RS si ninguna de las dos entradas está activa, se mantiene la salida. La situación en que las dos entradas están activas produce una salida indeterminada.

La tabla de verdad de este biestable sería:

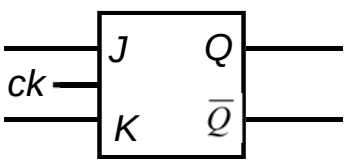
R	S	Q
0	0	Q
0	1	1
1	0	0
1	1	X

Los biestables "R-S" se construyen de dos formas:

- Siempre que aparezca un "11" que provoque un "1" será un biestable de inscripción prioritaria.
- Siempre que aparezca la combinación "11" que provoque un "0" será un biestable de borrado prioritario.

Este biestable es de tipo asíncrono

4.1.b.- Biestable JK



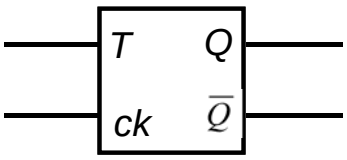
En el biestable anterior existe un caso indeterminado cuando las dos entradas están activas, este biestable soluciona este problema conmutando la salida.

La tabla de verdad sería:

J	K	Q
0	0	Q
0	1	1
1	0	0
1	1	$\overline{Q}$

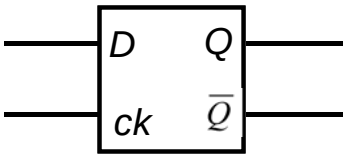
Este biestable normalmente es sincrono y por tanto tiene una entrada de reloj

4.1.c.- Biestable T



El biestable T tiene una entrada de datos y una entrada de reloj. Si la entrada es 0 la salida mantiene su valor, pero si la entrada es 1, la salida cambia cuando se produce un impulso de reloj

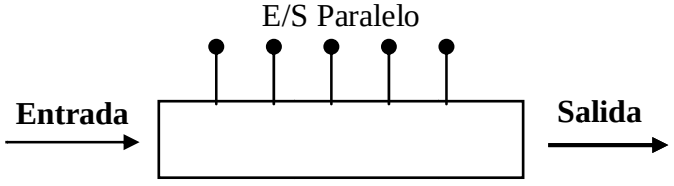
4.1.d.- Biestable D



El biestable D como el caso anterior tiene una entrada de datos y una entrada de reloj. Si hay un impulso de reloj, la salida tendrá el valor de la entrada D. Este biestable es capaz de almacenar un bit de información un tiempo igual al ciclo de reloj

4.2.-Registros de desplazamiento

Los registros de desplazamiento almacenan una palabra binaria de n bits. Están compuestos con n biestables encadenados, donde cada salida de cada biestable se conecta a la entrada del siguiente. Todos están conectados a la misma línea de reloj, lo que sincroniza las acciones.



Los registros de desplazamiento pueden ser:

- Entrada serie, salida serie: La entrada y salida de datos se produce uno a uno con cada impulso del reloj.
- Entrada serie, salida paralelo: La entrada se realiza uno a uno con cada impulso del reloj, pero en la salida se disponen de todos los bits.
- Entrada paralelo, salida serie: Todos los bits se cargan en los biestables con un único impulso, pero se recuperan uno a uno.
- Entrada paralelo, salida paralelo: Tanto la entrada como la salida de todos los bits se producen en un único impulso de reloj.

En las operaciones de "E/S" en "paralelo" tendré acceso simultáneo a todos los bits de los biestables.

En el caso de trabajar en "serie", sólo tendré acceso a un biestable. Por lo tanto para poder cargar un registro de desplazamiento de información 8 bits, necesitaré 8 pulsos de reloj tanto para meter como para sacar la información.

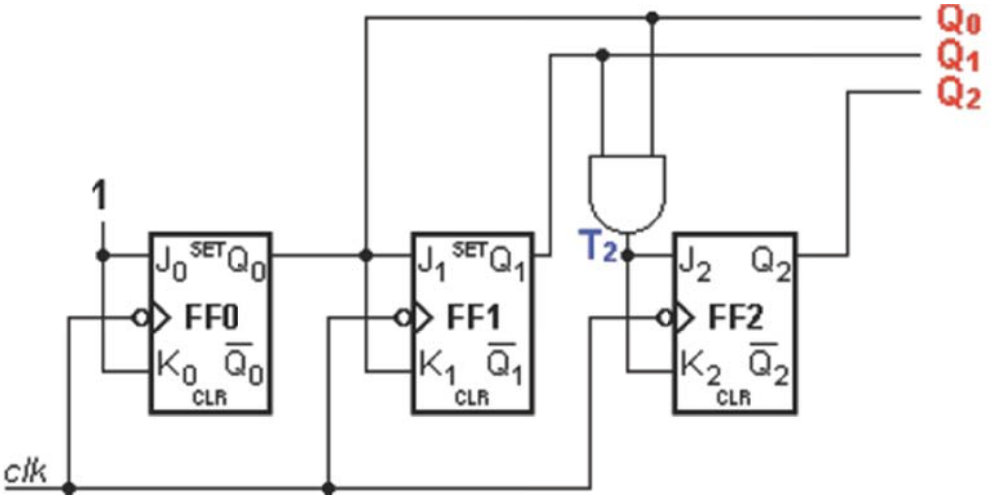
4.3.-Contadores

Los contadores son circuitos secuenciales que son capaces de almacenar en un determinado código (binario, decimal,…) y sirven para contar el número de ocurrencias de un evento. Los contadores se encuentran en casi todo sistema digital. Los contadores tienen varias líneas como entrada de datos:

1. Clk (Clock): Entrada de reloj
2. Rst (Reset): Inicializa el contador a cero
3. Entrada de habilitación: Esta señal si está activa hace que el contador funcione con normalidad; si esta inactiva no funciona el contador y guarda el último valor del contador

Pueden existir contadores tanto síncronos como asíncronos.

- En los sistemas asíncronos los FF no están conectados al mismo reloj, por lo que no cambian simultáneamente. La señal de reloj sólo está conectada al flip-flop que representa al bit menos significativo. Los otros FF se conectan en cascada sirviendo su salida de reloj para el siguiente, hasta llegar al bit más significativo
- En los contadores síncronos las entradas de reloj de todos los flip flops se conectan juntas a un reloj común. De esta manera todos los FF cambian de estado simultáneamente (en paralelo). El siguiente circuito muestra un contador síncrono de 3 bits



5.- CONCLUSIÓN

El tema se compone de tres apartados claramente diferenciados, la lógica de circuitos, en este apartado tratamos el álgebra de Boole que es el fundamento matemático en que se basa la lógica de circuitos. En este punto trataremos lo que es una función lógica, la representación de funciones lógicas en minterm y Maxterm, la simplificación de funciones lógicas y por último la representación de las funciones lógicas más importantes mediante componentes electrónicos como son las puertas lógicas.

El segundo apartado del tema son los circuitos combinacionales en los que explicamos que son los circuitos combinacionales y explicaremos los circuitos más importantes.

En el último punto trataremos los circuitos secuenciales, explicando los circuitos más representativos, biestables, contadores y registros de desplazamiento, también conocidos simplemente como registros.

6.- BIBLIOGRAFIA

Estructura y Tecnología de computadores I Aut. Carlos de Mora, Manuel Castro Ed. UNED
Estructura y Tecnología de computadores II Aut. Sebastián Dormido, Mª Antonia Canto Ed. UNED
Técnica y Procesos en las instalaciones automatizadas en los Edificios Aut. Juan Millán Esteller Ed. Paraninfo