# TEMA 13: CONTROL MEDIANTE CIRCUITOS LÓGICOS

1. Introducción
2. Sistemas de numeración y códigos
   1. Sistema binario. Código natural
   2. Sistema hexadecimal
   3. Códigos binarios
3. Álgebra de Boole

* Lógica de niveles
* Operaciones básicas en el álgebra de Boole
* Postulados, propiedades y teoremas del álgebra de Boole
  1. Suma lógica o función unión. Puerta O u OR
  2. Producto lógico o función intersección. Puerta AND
  3. Función igualdad
  4. Complementación o función negación. Puerta NO
  5. Función o puerta NOR
  6. Función o puerta NAND o NO Y
  7. Función o puerta ORex u O exclusiva

1. Obtención de la función lógica a partir de la tabla de verdad
2. Simplificación de funciones
   1. Diagramas de Karnaugh
3. Implementación de funciones con puertas NAND y NOR
4. Circuitos lógicos combinaciones integrados

7.1. Codificadores

7.2. Decodificador

7.3. Demultiplexador

7.4. Multiplexador

7.5. Comparador

## Introducción

Existen dos clases de control, dependiendo del tipo de señal que se transmite:

* El **control analógico** se emplea para señales analógicas, y éstas son aquellas en las que la variable estudiada es una función continua del tiempo.
* El **control digital**  se emplea para señales digitales, que son aquéllas en las que la variable estudiada sólo toma valores discretos, generalmente codificados según un sistema de notación.

1. **Sistemas de numeración y códigos.**

El sistema de numeración empleado habitualmente es el que utiliza la **base 10** o **sistema decimal.**

Los circuitos digitales utilizan para su trabajo el **sistema de numeración binario**, es decir, el que toma como base el número 2.

La representación de un número N en un sistema de base b, puede realizarse mediante el desarrollo en forma polinómica:

N = an . bn + an-1 bn-1 + … + a1 b1 + a0 b0 + an-1 bn-1…

b = base del sistema

ai =coeficientes que representan las cifras del nº.

Por ejemplo:

1. Si b =10, 0 ≤ ai<10

423,52 = 4 102 + 2 101 + 3 100 + 5 10-1 + 2 10-2

1. Si b =2, 0 ≤ ai< 2

1101,101 = 1 23 + 1 22 + 0 21 + 1 20 + 1 2-1 + 0 2-2 + 1 2-3

* 1. ***Sistema binario. Código binario natural***

Este sistema utiliza para su representación dos símbolos: 0 (cero) y 1 (uno). A cada uno de estos símbolos se le denomina bit.

Para pasar un número en sistema binario a su equivalente en sistema decimal, se procede de la siguiente manera: en primer lugar se expresa el número binario en su polinomio equivalente, y a continuación se opera el polinomio equivalente, de forma que el resultado obtenido será el número en base 10.

101,1 = 1 22 + 0 21 + 1 20 + 1 2-1 = 5,5

La operación inversa, es decir, pasar un número decimal entero a binario, se lleva a cabo dividiendo sucesivamente por dos hasta que el último cociente sea inferior a dos. El último cociente será el bit más significativo, seguido de los restos comenzando del último al primero.

Cociente Resto

25 : 2 12 1

12 : 2 6 0

6 : 2 3 0

3 : 2 1 1

bit más significativo 1 0 0 1

Para pasar un nº decimal fraccionado a uno binario, se multiplica éste por dos y se toma la parte entera. La parte decimal del número obtenido se vuelve a multiplicar por dos, y el proceso se repite hasta que el resultado sea cero o lleguemos a la precisión necesaria.

0,63 . 2 = 1,26

0,26 . 2 = 0,52 0,63 = 101(2)

0,52 . 2 = 1,04

* 1. ***Sistema hexadecimal***

Es un sistema de numeración muy empleado en microprocesadores. Es el que tiene base 16. Para su representación se utilizan los diez primeros dígitos decimales ( del 0 al 9) y las letras ( A, B, C, D, E, F).

En la siguiente tabla se muestra la equivalencia entre los sistemas hexadecimal, decimal y binario.

|  |  |  |
| --- | --- | --- |
| D | **H** | **B** |
| 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 |
| 10 | A | 1010 |
| 11 | B | 1011 |
| 12 | C | 1100 |
| 13 | D | 1101 |
| 14 | E | 1110 |
| 15 | F | 1111 |

* 1. ***Códigos binarios***

Un código es una representación de cantidades, de tal forma que a cada una de éstas se le asigna una combinación de símbolos determinada y viceversa.

El código más utilizado es el BCD natural (Decimal Codificado en Binario) que emplea las diez primeras combinaciones en orden creciente. El código Aiken, emplea las 5 primeras combinaciones y las 5 últimas de las 16 que se pueden formar

|  |  |  |  |
| --- | --- | --- | --- |
| Decimal | **BCD natural** | **BCD aiken** | **BCD ex3** |
| 0 | 0000 | 0000 | 0011 |
| 1 | 0001 | 0001 | 0100 |
| 2 | 0010 | 0010 | 0101 |
| 3 | 0011 | 0011 | 0110 |
| 4 | 0100 | 0100 | 0111 |
| 5 | 0101 | 0111 | 1000 |
| 6 | 0110 | 1100 | 1001 |
| 7 | 0111 | 1101 | 1010 |
| 8 | 1000 | 1110 | 1011 |
| 9 | 1001 | 1111 | 1100 |

1. **Álgebra de Boole**

Un álgebra de Boole es la estructura algebraica que corresponde a un conjunto de elementos, que pueden tomar los valores de 0 y 1.

* *Lógica de niveles*

La característica fundamental del control digital es que la magnitud que varía lo hace en torno a dos estados.

Estos dos estados, reciben varias denominaciones.

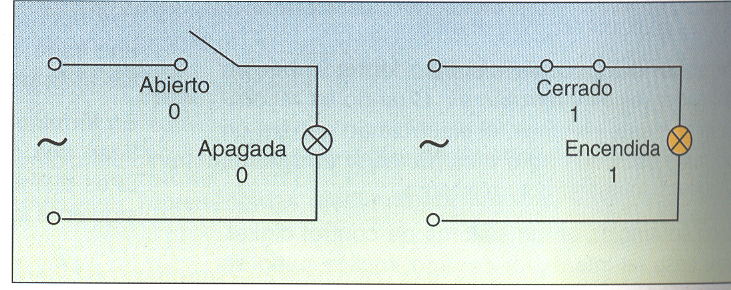
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 0 | No activo | OFF | L (Low) | Bajo |
| 1 | Activo | ON | H (High) | Alto |

Las cifras 0 y 1 representan un estado lógico, no un valor de la variable.

EJEMPLOS:

* Un circuito dispone de una lámpara gobernada por medio de un interruptor. Analiza los estados de ambos componentes según el circuito esté activado o desactivado.
* Cuando el circuito esté activado, el interruptor deberá estar cerrado y la lámpara, encendida. Cuando el circuito esté desactivado, el interruptor deberá estar abierto y la lámpara, apagada.

El siguiente esquema muestra los dos estados posibles del circuito y de sus elementos componentes.



Estos dos estados pueden resumirse en la tabla siguiente:

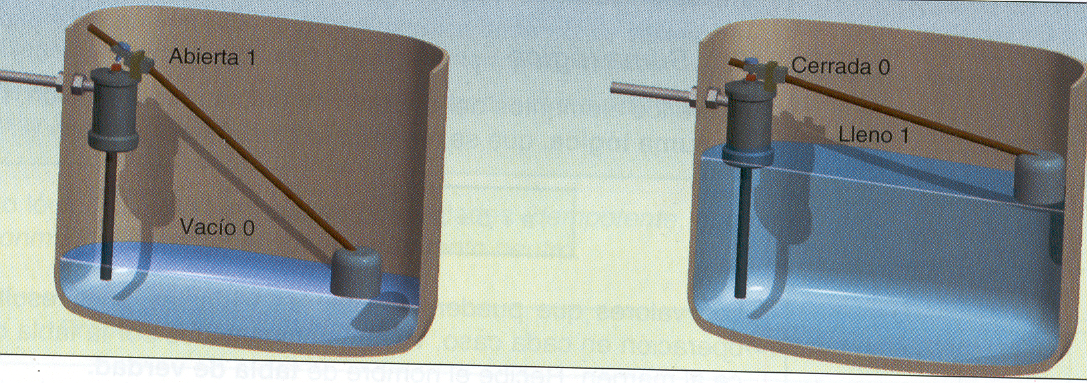
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Circuito | Interruptor | ***Estado*** | ***Lámpara*** | ***Estado*** |
| Activado | Cerrado | 1 | Encendida | 1 |
| Desactivado | Abierto | 0 | Apagada | 0 |

No existe ningún tipo de contraposición entre el estado del interruptor y el de la lámpara.

* El llenado de una cisterna de un inodoro está controlado por medio de una boya, de forma que han de cumplirse las condiciones siguientes:
* Mientras la cisterna está llena, la boya no actúa sobre la llave de paso, por lo que ésta permanece abierta y permite la entrada de agua.
* Cuando se alcanza el nivel máximo de llenado, la boya actúa sobre la llave de paso y ésta cierra el conducto, por lo que el agua deja de entrar.

Analizaremos comparativamente los estados de la cisterna y de la llave:

La cisterna vacía recibe el valor 0 y, cuando está llena el valor 1. A la llave le asignaremos el valor 0 cuando cierra el paso del agua, y el valor 1 en el momento que lo permite.



Si resumimos los estados en una tabla, obtenemos lo siguiente:

|  |  |  |  |
| --- | --- | --- | --- |
| Depósito | ***Estado*** | Llave de paso | ***Estado*** |
| Vacío | 0 | Abierta | 1 |
| Lleno | 1 | Cerrada | 0 |

En este caso, existe contraposición entre los estados de las variables, lo que nos lleva a concluir que la asignación de los valores 0 y 1 a un estado es un proceso totalmente arbitrario: se podría haber establecido el criterio a la inversa.

* *Operaciones básicas en el álgebra de Boole*

Se llama **función lógica** a toda variable binaria cuyo valor depende de una expresión algebraica formada por otras variables binarias que están relacionadas entre sí por las operaciones más y por. Ej:

S = a + b . c

Una **tabla de verdad** se utiliza para reflejar la ecuación y el comportamiento de las distintas operaciones y circuitos lógicos. El número de combinaciones será 2n, siendo n, el número de variables de entrada.

* *Postulados, propiedades y teoremas del álgebra de Boole*

**Propiedades**

*Propiedad conmutativa:*

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

*Propiedad asociativa:*

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

*Propiedad distributive:*

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

**Postulados**

a + 1 = 1 a . 1 = a

a + 0 = a a . 0 = 0

a + a = a a . a = a

a + ‾a = 1 a . ‾a = 0

a = a

**Teoremas**

* a + a . b = a
* a . ( a + b) = a
* a + ‾a . b = a + b
* b . ( a + ‾b ) = a . b

**Leyes De Morgan**

Son dos teoremas que se cumplen en los conjuntos con estructura de álgebra de Boole.

El **complementario o la negación de la suma lógica** de dos elementos es igual al **producto lógico** de los **complementarios o las negaciones** de los elementos considerados.

**a+b = a \* b**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **a** | **b** | **a+b** | **a+b** | **a** | **b** | **a \* b** |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 |

**El complementario o la negación del producto lógico** de dos elementos es igual a **la suma lógica de los complementarios o las negaciones** de los elementos considerados.

**a \* b = a + b**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **a** | **b** | **a\*b** | **a\*b** | **a** | **b** | **a + b** |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 |

* 1. ***Suma lógica o función unión.* PUERTA O (OR)**

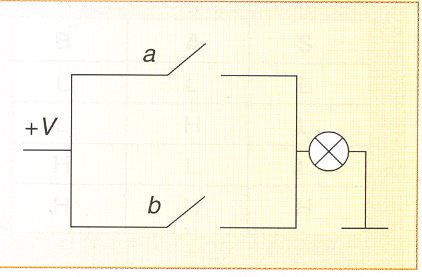
Sobre los elementos del conjunto se define una operación, denominada **suma lógica,** que se representa por el símbolo **+**.

**a + b = S** donde a,b = variables S= suma lógica

Los valores que pueden adoptar las variables, y el resultado de la operación en cada caso, vienen determinados por una tabla, que recibe el nombre de **tabla de verdad.**

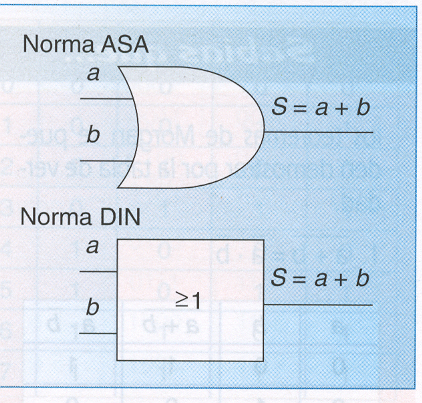
|  |  |  |
| --- | --- | --- |
| ***b*** | ***a*** | ***a+b = S*** |
| ***0*** | ***0*** | ***0*** |
| ***0*** | ***1*** | ***1*** |
| ***1*** | ***0*** | ***1*** |
| ***1*** | ***1*** | ***1*** |

Esta operación es equivalente a un circuito eléctrico provisto de dos interruptores en paralelo.



El valor 0 significa, en este caso, ausencia de señal (interruptor abierto) y el valor 1, presencia de señal (interruptor cerrado).

Puede comprobarse fácilmente que, para que el circuito permita el paso de la señal, basta con que uno de los interruptores esté cerrado.



Los símbolos que se utilizan en su representación pueden ser, según las normas empleadas, los siguientes:

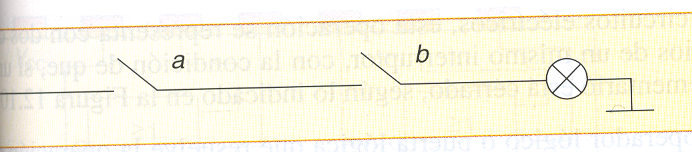
* 1. ***Producto lógico o función intersección.* PUERTA AND**

Se representa mediante el símbolo **\*** o simplemente escribiendo una variable junto a la otra.

**a \* b = a b = P** a,b = variables P = producto lógico

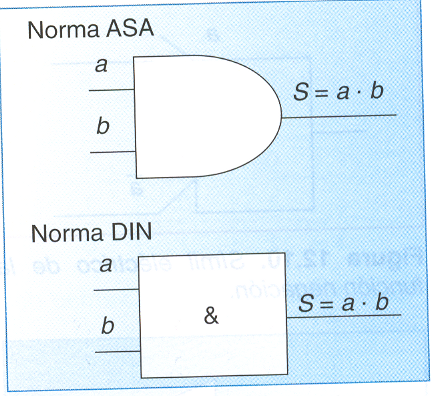
Tanto los valores de las variables como los resultados de la operación se determinan por medio de la **tabla de verdad**

|  |  |  |
| --- | --- | --- |
| ***b*** | ***a*** | ***a\* b = P*** |
| ***0*** | ***0*** | ***0*** |
| ***0*** | ***1*** | ***0*** |
| ***1*** | ***0*** | ***0*** |
| ***1*** | ***1*** | ***1*** |

 Esta operación es equivalente a un circuito eléctrico provisto de dos interruptores en serie.

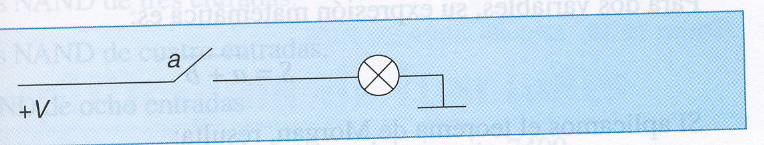
Al igual que en la suma lógica, el valor 0 significa ausencia de señal y el

valor 1, presencia de señal.

Para que el circuito permita el paso de la señal, han de estar cerrados los dos interruptores.

* 1. ***Función igualdad***

Es la más sencilla de todas, pues sólo interviene una variable. Su expresión matemática es :



S = a

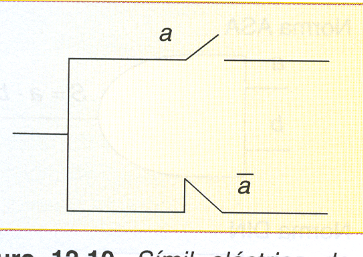
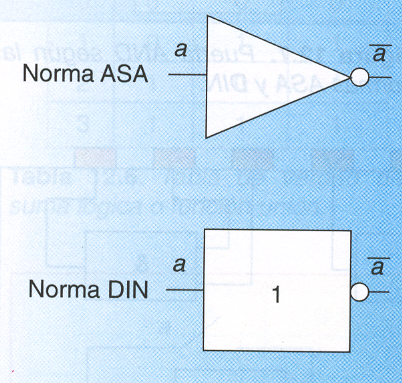
* 1. ***Complementación o función negación.* FUNCIÓN NO**

Todo elemento del conjunto, a, posee un elemento simétrico a de tal forma que la suma y el producto lógicos de cada elemento con su simétrico determinan, respectivamente, los valores 1 y 0. El simétrico se indica colocando una rayita sobre el símbolo del elemento.

**S = a**

La tabla de verdad siguiente, permite establecer la relación que existe entre los valores de a y su simétrico, así como la comprobación de la validez de la definición dada.

|  |  |  |  |
| --- | --- | --- | --- |
| **a** | **a** | **a + a** | **a\* a** |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |

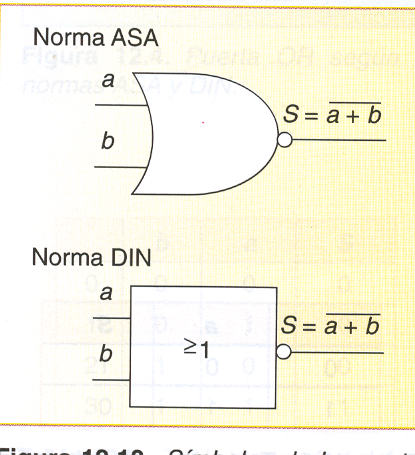


***3.5. Función o* PUERTA NOR**

Es la función complementaria o negación de la función OR. Su símbolo algebraico se obtiene añadiendo una rayita horizontal en la parte superior de la expresión de la función. Su ecuación lógica será, pues:

**F = a + b**

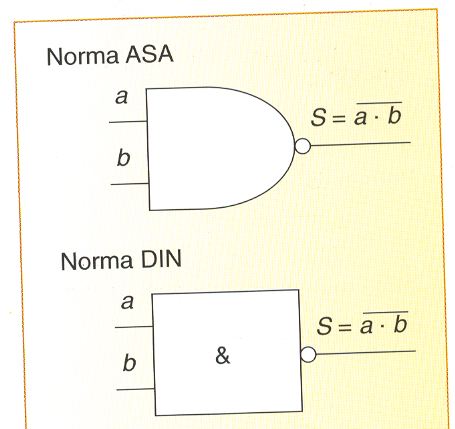
Sus posibles representaciones lógicas, así como su tabla de verdad, están reflejadas a continuación:



|  |  |  |
| --- | --- | --- |
| ***b*** | ***a*** | ***F*** |
| ***0*** | ***0*** | ***1*** |
| ***0*** | ***1*** | ***0*** |
| ***1*** | ***0*** | ***0*** |
| ***1*** | ***1*** | ***0*** |

* 1. ***Función o* PUERTA NAND o NO Y**

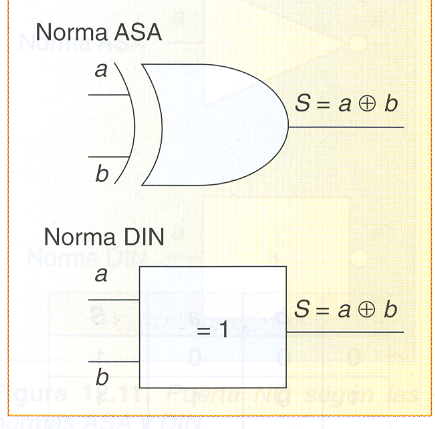
Es la función complementaria o negación de la función AND. Su símbolo algebraico se obtiene añadiendo una rayita horizontal en la parte superior de la expresión de la función. Su ecuación lógica será por tanto:



**F = a \* b = a b**

Las posibles representaciones gráficas de la función pueden observarse en la siguiente figura, así como la tabla de verdad, que es contraria de la función AND:

|  |  |  |
| --- | --- | --- |
| ***b*** | ***a*** | ***F*** |
| ***0*** | ***0*** | ***1*** |
| ***0*** | ***1*** | ***1*** |
| ***1*** | ***0*** | ***1*** |
| ***1*** | ***1*** | ***0*** |

* 1. ***Función o* PUERTA Orex u O exclusiva**

Se denomina función dilema. Su símbolo algebraico es ⊕. Su ecuación lógica se deriva de una combinación de las funciones AND, OR y NOT, aunque se considera una función elemental:

**F =** a b + a b = **a** ⊕ **b**

Las posibles representaciones gráficas de la función se muestran en la siguiente figura. Su tabla de verdad puede deducirse combinando adecuadamente las funciones elementales que la forman, es decir:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| ***b*** | ***a*** | ***b*** | ***a*** | ***a b*** | ***a b*** | ***F*** |
| ***0*** | ***0*** | ***1*** | ***1*** | ***0*** | ***0*** | ***0*** |
| ***0*** | ***1*** | ***1*** | ***0*** | ***0*** | ***1*** | ***1*** |
| ***1*** | ***0*** | ***0*** | ***1*** | ***1*** | ***0*** | ***1*** |
| ***1*** | ***1*** | ***0*** | ***0*** | ***0*** | ***0*** | ***0*** |

1. **Obtención de la función lógica a partir de la tabla de verdad**

A partir de la tabla de verdad podemos obtener la función lógica de dos maneras distintas, utilizando la primera y segunda forma canónica.

La **forma canónica** de una función es todo producto de sumas o toda suma de productos en las que aparecen todas las variables, bien en forma directa, bien en forma complementada.

* ***La primera forma canónica o suma de productos o minterms,*** se obtiene sumando todos los productos lógicos que dan salida 1, asignando al estado 0 (cero) la variable inversa y al estado 1 la variable directa.
* ***La segunda forma canónica o producto de sumas o maxterms,*** se obtiene con este razonamiento:

La salida en forma de productos de sumas ( S = ) será igual a 2n – 1 – i , siendo i el lugar que ocupan los términos que dan salida 0.

También se puede obtener directamente de la tabla de verdad observando que combinaciones hacen S = 0, y sustituyendo en cada una de ellas el valor 0 por una variable directa y el valor 1 por su expresión inversa.

Ejemplo:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **c** | **b** | **a** | **S** |
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 2 | 0 | 1 | 0 | 0 |
| 3 | 0 | 1 | 1 | 1 |
| 4 | 1 | 0 | 0 | 1 |
| 5 | 1 | 0 | 1 | 0 |
| 6 | 1 | 1 | 0 | 1 |
| 7 | 1 | 1 | 1 | 1 |

***Primera forma canónica:***



***Segunda forma canónica***

i = 0 7 – 0 = 7

i = 2 7 – 2 = 5

i = 5 7 – 5 = 2



1. **Simplificación de funciones**

***Simplificar*** una función lógica es hallar una nueva función equivalente a la primera, cuyo logigrama resulte más simplificado que el del circuito inicial.

Para obtener la simplificación de una función de varias variables, podemos aplicar las propiedades de las operaciones lógicas y las características de la estructura de álgebra de Boole.

Ejemplo: simplificar la función F = A B C + A B C

* Como la suma y el producto lógicos son distributivos, podemos sacar factor común, con lo que resulta:

F = AB ( C + C )

* Si cosideramos los respectivos elementos neutros de cada operación, tendremos:

F = AB ( C + C ) = A B \* 1 = A B

* 1. ***Diagramas de Karnaugh***

Se utilizan para simplificar funciones. Son cuadros de doble entrada en los que aparecen tantas casillas como posibles términos canónicos tenga la función que se desea representar.

El número dentro del recuadro indica el equivalente decimal de la combinación correspondiente.

En este método, los cuadrados correspondientes a los términos canónicos que forman parte de la función se indican mediante un 1, y los correspondientes a los términos que no forman parte de ella se dejan en blanco. En el caso de que existan combinaciones con términos indefinidos, éstos se representarán como más interese, o 0 o 1.

Para obtener la expresión más sencilla, es necesario realizar el mínimo número de agrupaciones con el mayor número de 1 posibles, que formen parte de cuadros adyacentes. La tabla es cerrada, es decir, la columna ( 1, 9, 13, 5) es adyacente a la (0, 8, 12, 4) y la primera fila es adyacente a la última.

El procedimiento para agrupar los 1 será:

* Se toman todos los unos que no pueden formar parte de un grupo de dos por no ser adyacentes con ninguno.
* Se forman los grupos de dos unos que no puedan formar parte de un grupo de cuatro.
* Se forman los grupos de cuatro que no puedan formar parte de un grupo de ocho.
* Cuando se cubran todos los 1, el proceso se detiene.
* Se ha de tener en cuenta que un 1 puede estar incluido en tantos grupos como sea necesario.

***Tablas de Karnaugh para dos, tres y cuatro variables***

|  |  |  |
| --- | --- | --- |
| b a | 0 | 1 |
| 0 | 0 | 1 |
| 1 | 2 | 3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| c a b | 00 | 01 | 11 | 10 |
| 0 | **0** | **2** | **3** | **1** |
| 1 | **4** | **6** | **7** | **5** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| cd ab | 00 | 01 | 11 | 10 |
| 00 | **0** | **2** | **3** | **1** |
| 101 | **8** | **10** | **11** | **9** |
| 11 | **12** | **14** | **15** | **13** |
| 10 | **4** | **6** | **7** | **5** |

Ejemplo: simplifica la función: 

* En primer lugar, procederemos a confeccionar el diagrama de Karnaugh correspondiente a una función de tres variables y asignaremos 1 a las casillas que representan los términos canónicos presentes.
* A continuación, agrupamos las casillas marcadas con un 1. Es posible formar un grupo horizontal de cuatro y otro vertical de dos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ab**  **c** | 0 0 | 0 1 | 1 1 | 1 0 |
| 0 | 1 0 | 1 2 | 1  3 | 1 1 |
| 1 | 4 | 1 6 | 7 | 5 |

- Grupo 1: celdas 0,2,3,4 → c

**-** Grupo 2: celdas 2,6 → a b

La función simplificada será pues:



1. **Implementación de funciones con puertas NAND y NOR**

Las puertas NAND y NOR se conocen también como puertas universales debido a que todas las funciones se pueden construir con ellas.

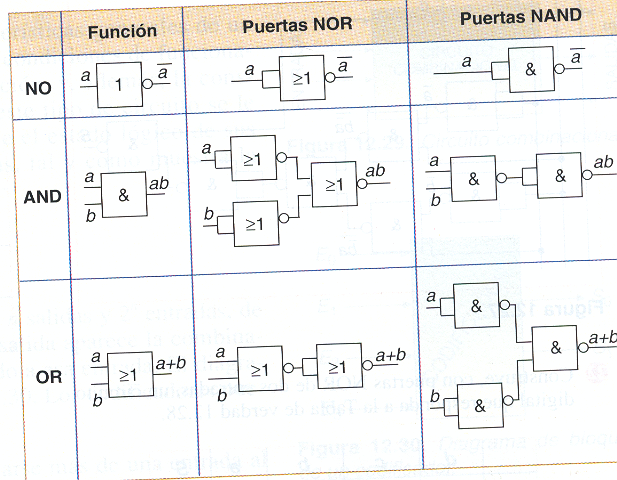
Para poder realizar una función determinada o un circuito digital utilizando sólo puertas NAND o NOR, debemos aplicar los teoremas de Morgan tantas veces como sea necesario, hasta que la función se exprese en forma de productos o sumas negadas respectivamente.

Las tres funciones básicas con puertas NAND y NOR de dos entradas, son:

* ***Función inversión***

Para realizar la función inversión con puertas NAND o NOR de dos entradas debemos recordar la tabla de verdad de este tipo de puertas. En ambas, cuando las dos entradas valen lo mismo, 1 o 0, la salida es 0 o 1 respectivamente.

* ***Función AND***

* Puertas NAND*



*Puertas NOR*



* ***Función OR***

*Puertas NAND*



*Puertas NOR*

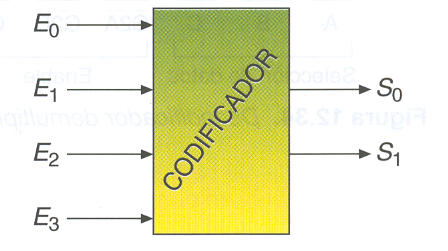


1. **Circuitos lógicos combinacionales integrados**



La característica fundamental de los circuitos combinacionales es que las salidas dependen del estado de las entradas en cada instante. Es decir, las salidas son independientes del tiempo.

* 1. ***Codificadores***

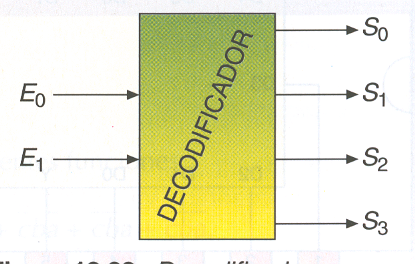


Un codificador es un circuito combinacional que posee n salidas y 2n entradas, de forma que al accionarse una de sus entradas, en la salida aparece la combinación binaria correspondiente al número decimal asignado a esa entrada.

En los **codificadores sin prioridad** no puede activarse más de una entrada al mismo tiempo y normalmente no se emplean.

Los **codificadores con prioridad** se produce una acción simultánea de varias de sus entradas, en la salida se presentará el código de aquella entrada que tenga asignada mayor peso significativo, normalmente es la de mayor valor decimal.

* 1. ***Decodificadores***

******Son circuitos combinacionales cuya misión es convertir todas las combinaciones binarias pertenecientes a un código determinado en su correspondiente equivalencia en el sistema decimal.

Los decodificadores disponen de un cierto número de entradas, **n,** y de otro número de salidas menor o igual a **2n.**

Su funcionamiento es el siguiente: cuando se presenta una determinada combinación binaria a la entrada, se activa una de las salidas (las restantes quedan desactivadas). Cada una de ellas corresponde a la expresión decimal equivalente a la combinación binaria de entrada.

Los decodificadores se utilizan también para dos aplicaciones:

Los demultiplexadores

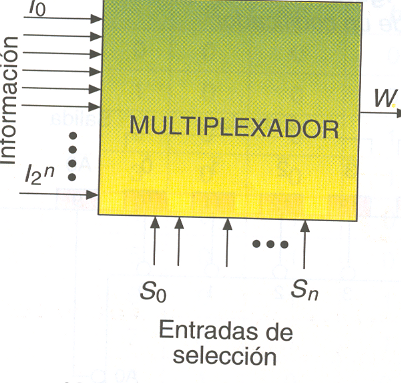
La generación de funciones lógicas.

* 1. ***Demultiplexador***

Los demultiplexadores son circuitos lógicos combinacionales con una sola entrada, N salidas y n entradas de control.

Su labor consiste en transmitir la información desde la entrada a la salida seleccionada mediante las entradas de control.

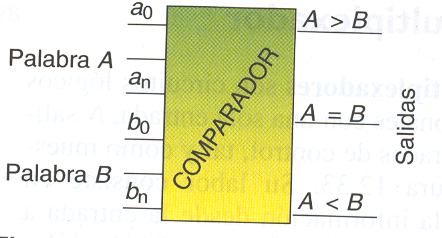
***7.4. Multiplexador***

El **multiplexor** es un circuito lógico combinacional cuya labor consiste en canalizar varias fuentes de información binaria hacia una línea común de salida.

En general, un multiplexor posee 2n entradas de información, denominadas I0 a In, n entradas de selección, conocidas como S0 a Sn, y una sola salida de información W.

El funcionamiento de un multiplexor es el siguiente: cuando se presenta una combinación binaria en las entradas de selección, en la salida aparece un solo dato, correspondiente a la entrada que lleve asignada esta combinación binaria.

***7.5. Comparador***

Un comparador digital es un circuito lógico combinacional capaz de detectar las relaciones mayor, igual y menor entre dos configuraciones binarias.

En esencia, una comparación digital presenta:

* Dos grupos de n líneas de entrada ( A y B). Cada grupo de líneas canaliza hacia la entrada del comparador una palabra binaria de n bits.
* Tres líneas de salida. Al comparar las dos palabras binarias introducidas en el comparador, el sistema combinacional responderá activando una de las tres salidas siguientes.

**A > B:** cuando la palabra binaria A sea de magnitud superior a B

**A = B:** cuando la palabra binaria A sea igual a la B

**A < B:** cuando la palabra binaria A sea de menor magnitud que la B