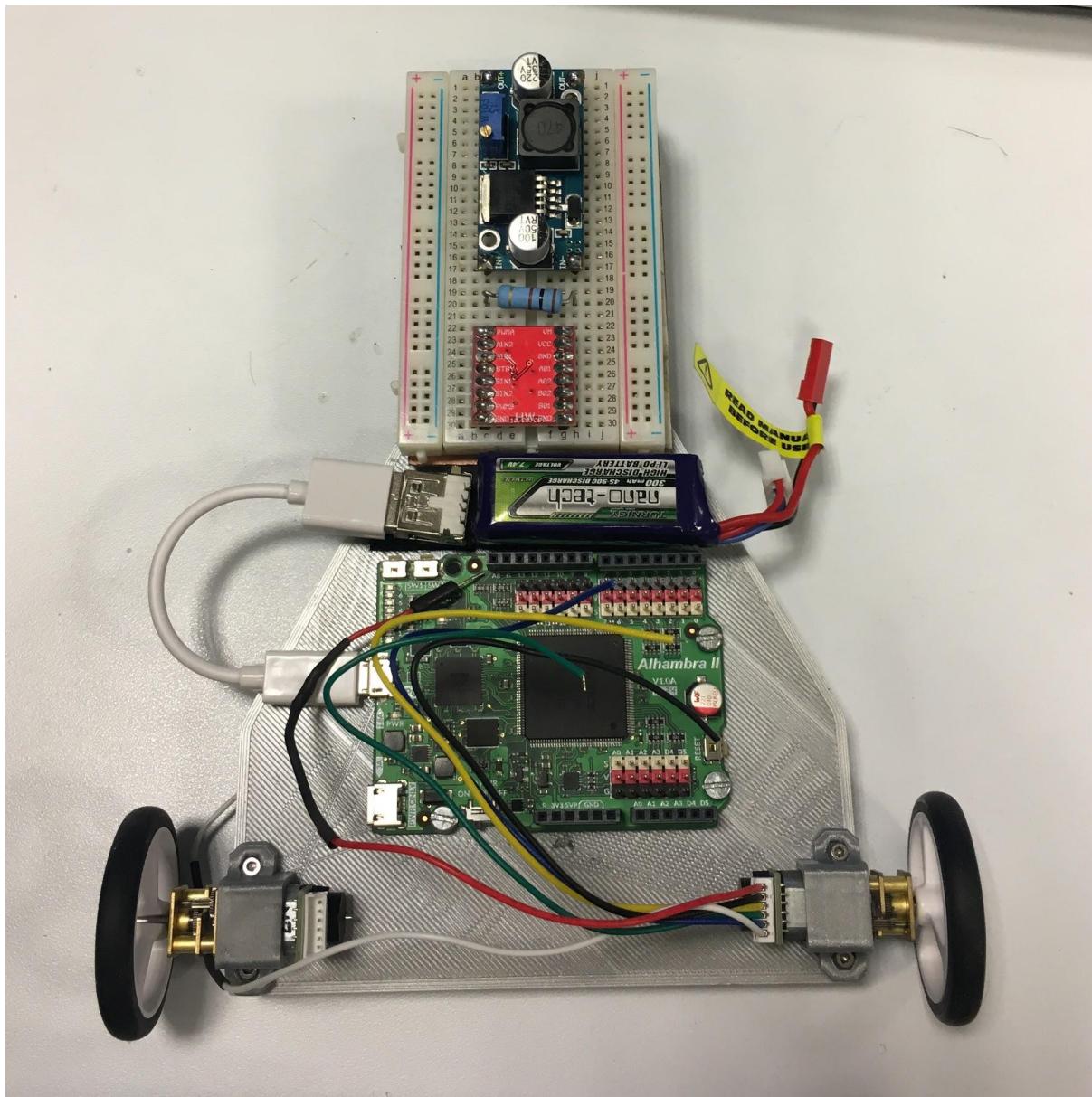


PORTADA



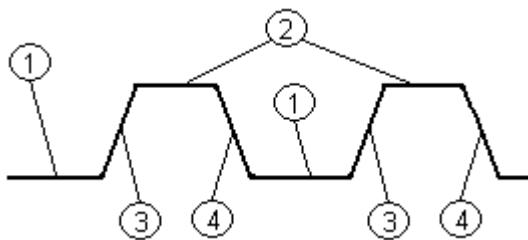
Egileak:

-Unai Cortajarena
-Urki Etxeberria
-Jagoba Fernandez
-Lander Nieto

2. ELECTRÓNICA DIGITAL

2.1 ¿Qué es una señal digital?

Es una parte de la electrónica que se encarga de sistemas electrónicos en los cuales la información está codificada en dos únicos estados. A dichos estados se les puede llamar "verdadero" o "falso", o más comúnmente 1 y 0, refiriéndose a que en un circuito electrónico digital hay dos niveles de tensión.



1. SEÑAL DIGITAL
2. NIVEL BAJO
3. NIVEL ALTO
4. FLANCO DE SUBIDA
5. FLANCO DE BAJADA

2.2 VENTAJAS DE LAS SEÑALES DIGITALES

- Puede ser amplificada y reconstruida al mismo tiempo, gracias a los sistemas de regeneración de señales.

- Cuenta con sistemas de detección y corrección de errores, en la recepción.
- Facilidad para el procesamiento de la señal. Cualquier operación es fácilmente realizable a través de cualquier software de edición o procesamiento de señal.
- Permite la generación infinita con pérdidas mínimas en la calidad. Esta ventaja sólo es aplicable a los formatos de disco óptico; la cinta magnética digital, aunque en menor medida que la analógica (que solo soporta como mucho 4 o 5 generaciones), también va perdiendo información con la multigeneración.
- Las señales digitales se ven menos afectadas a causa del ruido ambiental en comparación con las señales analógicas.

2.3 DESVENTAJAS

- Necesita una conversión analógica-digital previa y una decodificación posterior en el momento de la recepción.
- Requiere una sincronización precisa entre los tiempos del reloj del transmisor con respecto a los del receptor.
- Pérdida de calidad cada vez mayor en el muestreo respecto de la señal original.

2.4 ¿Cuál es la diferencia entre una señal digital y una analógica?

Cuando un equipo electrónico nos muestra una información, puede hacerlo de forma analógica o de forma digital. Analogica quiere decir que la información, la señal, para pasar de un valor a otro pasa por todos los valores intermedios, es continua. La señal digital, en cambio, va “a saltos”, pasa de un valor al siguiente sin poder tomar valores intermedios.

Una señal analoga es continua, y puede tomar infinitos valores.

Una señal digital es discontinua, y sólo puede tomar dos valores o estados: 0 y 1, que pueden ser impulsos eléctricos de baja y alta tensión, interruptores abiertos o cerrados.

3. Puertas lógicas

3.1 ¿Qué es una puerta lógica? ¿Cuáles están ahí y cuál es la "verdad de cada tabla"(egiaren taula)?

Las Compuertas Lógicas son circuitos electrónicos conformados internamente por transistores que se encuentran con arreglos especiales con los que otorgan señales de voltaje como resultado o una salida de forma booleana, están obtenidos por operaciones lógicas binarias (suma, multiplicación).

Existen diferentes tipos de compuertas y algunas de estas son más complejas, con la posibilidad de ser simuladas por compuertas más sencillas. Todas estas tienen tablas de verdad que explican los comportamientos en los resultados que otorga, dependiendo del valor booleano que tenga en cada una de sus entradas.

Trabajan en dos estados, “1” o “0”, los cuales pueden asignarse a la lógica positiva o lógica negativa. El estado 1 tiene un valor de 5v como máximo y el estado 0 tiene un valor de 0v como mínimo y existiendo un umbral entre estos dos estados donde el resultado puede variar sin saber con exactitud la salida que nos entregara. Las lógicas se explican a continuación:

- La lógica positiva es aquella que con una señal en alto se acciona, representando un 1 binario y con una señal en bajo se desactiva. representado un 0 binario.

- La lógica negativa proporciona los resultados inversamente, una señal en alto se representa con un 0 binario y una señal en bajo se representa con un 1 binario.

3.2 De la tabla de verdad a las funciones lógicas del álgebra de Boole ¿De qué sirve el método de Karnaugh? Intenté usar.

El método de Karnaugh es un método gráfico. Se usan unas tablas llamadas tablas o diagramas de Karnaugh. Dichas tablas tienen una casilla por cada combinación de variables de la función, de forma que para 3 variables tendremos $2^3 = 8$ casillas, para cuatro variables tendremos $2^4 = 16$ casillas.

Diagrama de Karnaugh para 4 variables. Es una tabla de 4x4 casillas. Las filas están etiquetadas con las combinaciones de dos variables (ba) y las columnas con las de una variable (dc). Las filas son: 00, 01, 11, 10. Las columnas son: 00, 01, 11, 10. Los cuadros están vacíos.

Diagrama de Karnaugh para 3 variables. Es una tabla de 2x4 casillas. Las filas están etiquetadas con las combinaciones de una variable (c) y las columnas con las de dos variables (ba). Las filas son: 0, 1. Las columnas son: 00, 01, 11, 10. Los cuadros están vacíos.

Diagrama de Karnaugh para 4 variables Diagrama de Karnaugh para 3 variables

El orden de las combinaciones no es binario natural si no que es código Gray (00, 01, 11, 10) esto es debido a que el funcionamiento del método se basa en combinaciones adyacentes.

Una vez dibujado el diagrama, se trasladan a éste las combinaciones de la tabla de la verdad poniendo un 1 en la casilla correspondiente. Ejemplo: sea la función $f = a \cdot b \cdot c + a \cdot b \cdot \bar{c} + a \cdot \bar{b} \cdot c$ que como se ve, vale 1 para las combinaciones $\{c,b, a\} = \{ 0,0,1 \}, \{ 0,1,0 \}, \{ 1,0,0 \}$. Pues en el diagrama de Karnaugh pondremos un 1 en cada una de esas casillas.

	ba 00	01	10
c 0		1	
1	1		

Casillas donde $f = 1$

Ahora es cuando vamos a simplificar. A partir de las posiciones de los unos en la tabla, intentamos formar grupos de unos lo más grandes posibles. Dichos grupos de unos:

Deberán estar constituidos por un número de unos que sea potencia de dos (no valen 3 ni 6 ni 7...). - Deberán ser un conjunto convexo (o sea, no tener esquinas hacia dentro). - No podrán ir en diagonal. - Intentaremos formar el menor número de grupos y éstos deberán ser lo más grandes posible. Uno puede formar parte de tantos grupos como haga falta.

En los grupos que formemos se eliminan las variables que estén presentes en el cero y en el uno. En nuestro diagrama anterior, vemos que podemos hacer dos grupos de dos variables: uno con las casillas

$\{c,b,a\} = \{0,0,1\}, \{1,0,1\}$ y otro con $\{c,b,a\} = \{\}, 1,0,0 \{1,0,1\}$ Vemos que en el primer grupo la variable a aparece con 1 y con 0, por lo que la eliminamos, quedándonos $c=1$ y $b=0$ por lo que el término nos queda $b \cdot c$. En el segundo grupo aparece la c negada y sin negar, por lo que la eliminamos, quedándonos $b=0$ y $a=1$ por lo que el término nos queda $b \cdot a$. Por lo que la función simplificada queda: $f = c \cdot b + b \cdot a = b \cdot (a + c)$.

A continuación se ponen unos cuantos ejemplos de grupos posibles para un diagrama de cuatro variables.

	dc	ba	00	01	11	10
00						
01			1			
11						
10						

No cambia ninguno, por lo que $f = \bar{d} c \bar{b} \cdot a$

	dc	ba	00	01	11	10
00						
01			1	1		
11						
10						

Cambian b y d , por lo que queda $f = a \cdot c$

	dc	ba	00	01	11	10
00			1	1	1	1
01			1	1	1	1
11						
10						

Cambian a , b y c , por lo que queda $f = \bar{d}$

	dc	ba	00	01	11	10
00			1	1		
01			1	1		
11						
10			1	1		

Cambian a , c y d , por lo que queda $f = \bar{b}$

	dc	ba	00	01	11	10
00						
01						
11						
10	1	1				

Cambia a , por lo que queda $f = d \bar{c} \bar{b}$

	dc	ba	00	01	11	10
00						
01						
11						
10	1	1				

Cambia d , por lo que queda $f = c b \cdot a$

	dc	ba	00	01	11	10
00			1	1	1	1
01						
11						
10						

Cambian a y b , por lo que queda $f = \bar{d} \bar{c}$

	dc	ba	00	01	11	10
00			1			
01			1			
11						
10	1					

Cambian d y c , por lo que queda $f = \bar{b} \bar{a}$

	dc	ba	00	01	11	10
00						
01						
11	1					
10				1		

Cambia b , por lo que queda $f = d \bar{c} \bar{a}$

	dc	ba	00	01	11	10
00			1	1		
01						
11						
10	1	1				

Cambian d y b , por lo que queda $f = \bar{c} \bar{a}$

	dc	ba	00	01	11	10
00			1			
01			1			
11						
10	1					

Cambian d y b , por lo que queda $f = \bar{c} \bar{a}$

	dc	ba	00	01	11	10
00			1			
01			1			
11						
10	1					

Cambian b , c y d , por lo que queda $f = \bar{a}$

Observemos que los cuatro últimos ejemplos no parecen cumplir con lo que dijimos acerca de los grupos, que debían ser un conjunto convexo. En realidad sí que lo son. Debemos ver los diagramas de Karnaugh como una superficie continua, algo así como una caja de cartón desmontada que cuando se monta se cierra y se unen los lados. Se pueden coger estos grupos siempre que queramos, sin más condición que ser potencia de 2 y no ir en diagonal.

EJEMPLO:

Diseñar un circuito combinacional que realice la división entre 3 (entera) de un número codificado en BCD.

Como ya se sabe, la división entera de un número tiene dos partes: cociente y resto, ambos enteros. Nuestro circuito hará la división de un número BCD (o

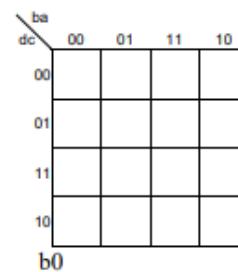
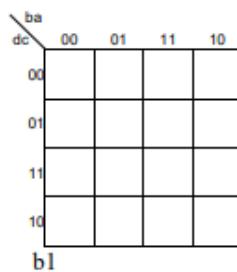
sea, del 0 al 9) entre 3. El cociente más grande será el obtenido al dividir el número más grande, que en BCD es el 9. Como $9:3 = 3$ necesitamos 2 bits para representar este número por lo que nuestro circuito deberá tener dos salidas. Una para el bit de más peso y otra para el de menos peso. A continuación se muestra la salida del circuito y la tabla de verdad.

Número	resultado
0	0
1	0
2	0
3	1
4	1
5	1
6	2
7	2
8	2
9	3

D	C	B	A	b1	b0
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	0	0
0	0	1	1	0	1
0	1	0	0	0	1
0	1	0	1	0	1
0	1	1	0	1	0
0	1	1	1	1	0
1	0	0	0	1	0
1	0	0	1	1	1

División entre 3 de los números del 0 al 9 Tabla de verdad del circuito

Nos van a quedar dos funciones de 4 variables cada una, lo cual es mucho para poder simplificar por el método algebraico, por lo que usaremos el método de Karnaugh. Usaremos dos diagramas de Karnaugh; uno para la variable b1 y otro para b0.



En primer lugar rellenamos los diagramas con unos en los lugares donde corresponda. b1 en las posiciones $\{(0,1,1,0), (0,0,1,1), (1,1,1,0)\}$ y b0 en las posiciones $\{(0,0,0,1), (1,0,0,1), (0,1,1,0)\}$

ba	00	01	11	10
00				
01			1	1
11				
10	1	1		
b1				

ba	00	01	11	10
00			1	
01	1	1		
11				
10		1		
b0				

Ahora buscamos grupos de los más grandes posibles. Vemos que para b1 tendremos dos grupos de dos: uno formado por $\{(,)1,1,1,0\}$ ($0,1,1,0$) y otro por $\{(0,0,0,1), (1,0,0,1)\}$. Mientras que para b0 tendremos un grupo de dos unos formado por $\{(0,0,1,0), (1,0,1,0)\}$ y dos grupos de un solo uno correspondientes a las combinaciones $\{(1,1,0,0)\}$ y $\{(1,0,0,1)\}$.

ba	dc	00	01	11	10
00					
01			1	1	
11					
10	1	1			
b1					

ba	dc	00	01	11	10
00				1	
01	1	1			
11					
10			1		
b0					

A partir de estos diagramas obtenemos directamente las ecuaciones del circuito. Para b1 tendremos dos términos: en el primer grupo (el de arriba) cambia la a, mientras que dc = 011 por lo que nos queda $\bar{d} \cdot c \cdot b$. En el grupo de abajo cambia la a también, por lo que nos queda $d \cdot \bar{c} \cdot \bar{b}$. Para b0 operamos de la misma forma. Observemos que vamos a tener tres términos, dos de los cuales tendrán todas las variables. Cuanto menores sean los grupos, menos variables desaparecen y por tanto más variables aparecerán en la expresión final.

$$b1 = \bar{d} \cdot c \cdot b + d \cdot \bar{c} \cdot \bar{b}$$

$$b0 = \bar{d} \cdot \bar{c} \cdot b \cdot a + d \cdot \bar{c} \cdot \bar{b} \cdot a + \bar{d} \cdot c \cdot \bar{b}$$

A partir de estas ecuaciones ya podemos implementar nuestro circuito. Nótese que mediante el método de Karnaugh lo que obtenemos es la expresión mínima de la función expresada en forma de suma de productos. Esto no quiere decir que la función no se pueda simplificar más. En ocasiones podremos, pero ya no será una expresión en forma de suma de productos. En este caso, podríamos encontrar una expresión más simple para b0:

$$b_0 = \bar{d} \bar{c} b \cdot a + d \bar{c} \bar{b} \cdot a + \bar{d} c \cdot \bar{b} = a \bar{c} \cdot (\bar{d} b + d \bar{b}) + \bar{d} c \cdot \bar{b} = a \bar{c} \cdot (b \oplus d) + \bar{d} c \cdot \bar{b}$$

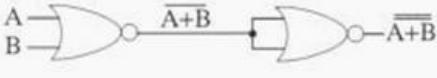
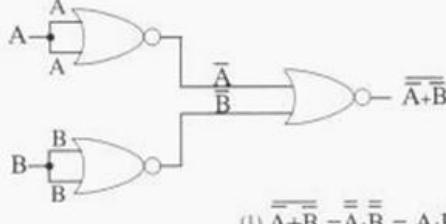
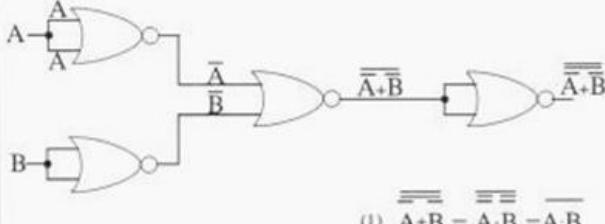
3.3 ¿Cómo se puede implementar un circuito que consta de puertas lógicas?

Para la implementación de circuitos lógicos se pueden utilizar cualquier tipo de puertas. Sin embargo la tendencia más común es implementar un circuito empleando solamente un tipo de puertas. De este modo se abaratan los costes.

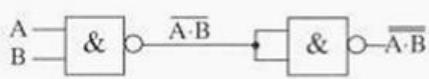
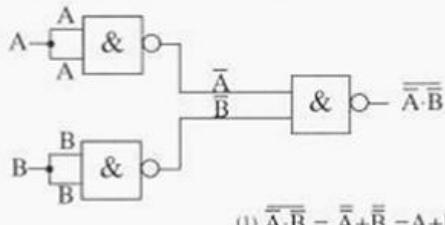
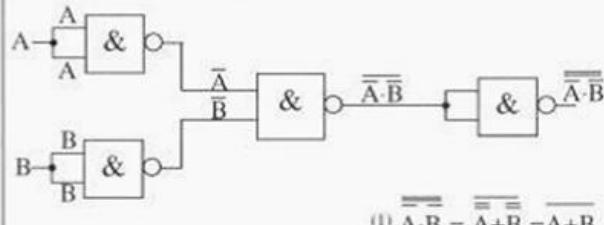
Este método de implementación solo se puede realizar con puertas NAND o NOR, ya que solo estas dos puertas lógicas son universales, es decir se puede realizar cualquier circuito lógico y sustituir cualquier puerta empleando únicamente este tipo de puertas, para ello debemos seguir un cierto protocolo aprovechando que una doble negación es igual a una afirmación ($A = \overline{\overline{A}}$).

Siempre podemos negar una expresión lógica dos veces, tantas veces como necesitemos y ésta quedará inmutable, si luego aplicamos el teorema de DeMorgan a una de las dos negaciones anteriores, conseguimos que un producto negado se convierta en una suma de variables negadas, o bien que una suma negada se convierta en un producto de variables negadas.

Ejemplo de ejecución de cualquier puerta empleando solamente puertas NOR.

puerta <i>NOT</i>	puerta <i>OR</i>
 Nota: $\overline{A+A} = \overline{A}$	 $\overline{\overline{A+B}} = A+B$
puerta <i>AND</i>	puerta <i>NAND</i>
 $(1) \overline{\overline{A+B}} = \overline{\overline{A} \cdot \overline{B}} = A \cdot B$	 $(1) \overline{\overline{A+B}} = \overline{\overline{A} \cdot \overline{B}} = A \cdot B$

De igual forma podemos utilizar puertas lógicas NAND:

puerta <i>NOT</i>	puerta <i>AND</i>
 Nota: $\overline{A \cdot \overline{A}} = \overline{A}$	 $\overline{A \cdot B} = A \cdot B$
puerta <i>OR</i>	puerta <i>NOR</i>
 $(1) \overline{\overline{A+B}} = \overline{\overline{A} + \overline{B}} = A+B$	 $(1) \overline{\overline{A+B}} = \overline{\overline{A} + \overline{B}} = A+B$

3.4 ¿Cómo se pueden crear puertas NAND para crear equivalentes a otras puertas? ¿Para qué es esto? ¿es esto útil?

- Pasar de NAND a NOT:

vemos que la parte $a=0, b=0$ y $a=1, b=1$ es igual que en la tabla del NOT

por lo cual debemos quitar una entrada en NAND, es decir poner la misma entrada, quitando de esta manera el caso $d=0$ y $b=1$ o $d=1$ y $b=0$.

TABLA DE NAND

a	b	f
0	0	1
0	1	1
1	0	1
1	1	0

TABLA DE NOT

d	f
0	1
1	0



3.5 Para crear AND utilizando NAND

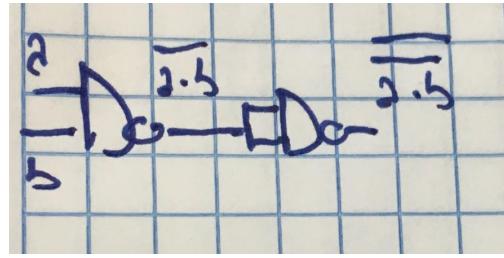
podemos ver que f es la inversa de la x es decir si hacemos un NOT de f , tendremos el resultado de x , y al saber como hacer una NOT con NAND podemos crear una AND utilizando NAND.

TABLA DE NAND

a	b	f
0	0	1
0	1	1
1	0	1
1	1	0

TABLA AND

a	d	f
0	0	0
0	1	0
1	0	0
1	1	1



3.6 OR con una NAND

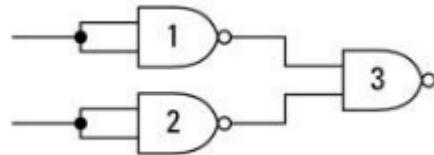
Sabemos que una OR es igual a $a+b$, para confundir esto primero debemos negar esto 2 veces para que siga siendo la misma operación, a continuación y para transformar el $+$ en una multiplicación rompemos una de las negaciones.

TABLA DE NAND

a	b	f
0	0	1
0	1	1
1	0	1
1	1	0

TABLA DE OR

a	d	f
0	0	0
0	1	1
1	0	1
1	1	1



3.6 NOR con NAND

Podemos observar que la NOR es la negada de la OR, y sabiendo cómo representar la OR con NAND solo tendremos que negar esto.

TABLA DE NAND

TABLA DE NOR

a	b	f	a	d	f
0	0	1	0	0	1
1	0	1	1	0	0
0	1	1	0	1	0
1	1	0	1	1	0

Or Con NAND = $\overline{\overline{a+b}}$

Nor Con NAND = $\overline{a \cdot b}$

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

3.6 X or con NAND

En este caso igual que en el caso OR tenemos una suma, y para representarlo con NAND debemos convertirlo en una multiplicación, es decir debemos negarlo 2 veces.

TABLA DE NAND

a	b	f
0	0	1
1	0	1
0	1	1
1	1	0

TABLA XOR

a	d	f
0	0	0
0	1	1
1	0	1
1	1	0

$$XOR = a \oplus b = \bar{a} \cdot b + \bar{b} \cdot a = \overline{\overline{a} \cdot b + \bar{b} \cdot a} = \overline{\overline{a} \cdot b} \cdot \overline{\bar{b} \cdot a}$$

XNor con Nand

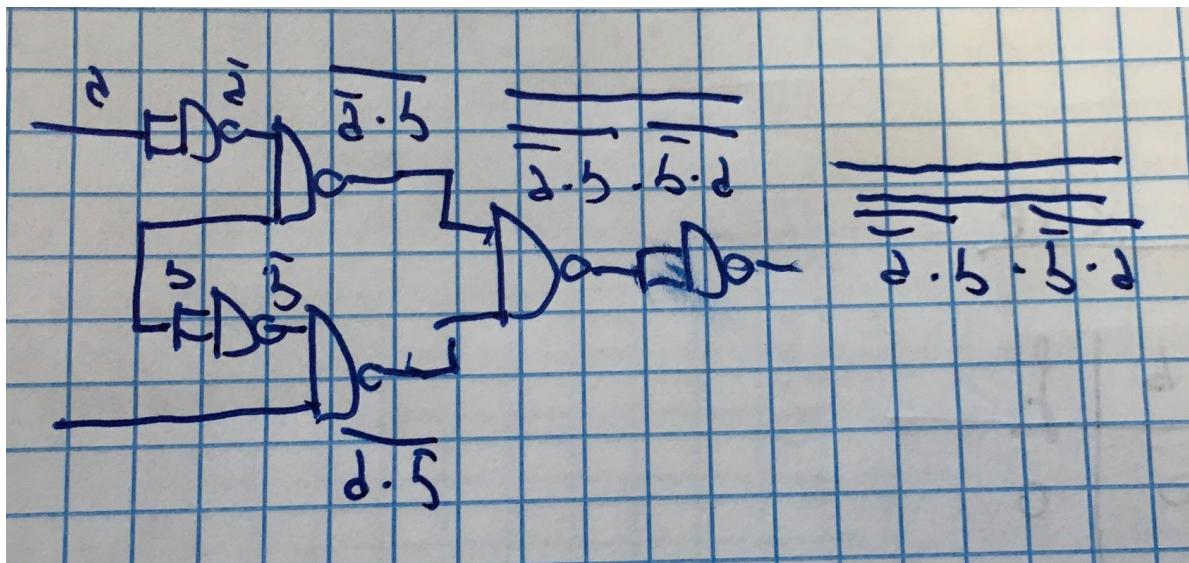
3.7 XNOR con NAND

• en este caso Sabemos que la XNOR es la negación de la XOR, por lo que devaremos negar la fórmula que hemos sacado anteriormente

$$\overline{\overline{a} \cdot b} \cdot \overline{\bar{b} \cdot a}$$

$$XNOR = \overline{\overline{a} \cdot b} \cdot \overline{\bar{b} \cdot a}$$

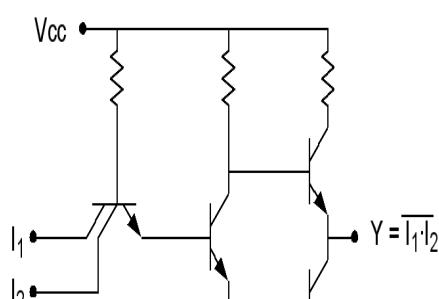
Tabla Nand			Tabla Nor		
a	b	f	a	b	f
0	0	1	0	0	1
0	1	1	0	1	0
1	0	1	1	0	0
1	1	0	1	1	1



4. ¿En qué se diferencian TTL y CMOS?

4.1 TTL (Transistor- Transistor Logic) o "Lógica Transistor a Transistor".

Es una familia lógica o lo que es lo mismo, una tecnología de construcción de circuitos electrónicos digitales. En los componentes fabricados con tecnología TTL los elementos de entrada y salida del dispositivo son transistores bipolares.



4.2 CARACTERÍSTICAS DEL TTL:

- Su tensión de alimentación característica se halla comprendida entre los 4,75v y los 5,25V (como se ve un rango muy estrecho).
- Los niveles lógicos vienen definidos por el rango de tensión comprendida entre 0,2V y 0,8V para el estado L (bajo) y los 2,4V y Vcc para el estado H (alto).
- La velocidad de transmisión entre los estados lógicos es su mejor base, si bien esta característica le hace aumentar su consumo siendo su mayor enemigo. Motivo por el cual han aparecido diferentes versiones de TTL como FAST, LS, S, etc. y últimamente los CMOS: HC, HCT y HCTLS. En algunos casos puede alcanzar poco más de 250 MHz.
- Las señales de salida TTL se degradan rápidamente si no se transmiten a través de circuitos adicionales de transmisión (no pueden viajar más de 2 m por cable sin graves pérdidas).
- Los circuitos de tecnología TTL se prefijan normalmente con el número 74 (54 en las series militares e industriales). A continuación un código de una o varias cifras que representa la familia y posteriormente uno de 2 a 4 con el modelo del circuito.

4.3 CSMO:

La familia lógica de MOS complementarios está caracterizada por su bajo consumo. Es la más reciente de todas las grandes familias y la única cuyos componentes se construyen mediante el proceso MOS. El elemento básico de la CMOS es un inversor.

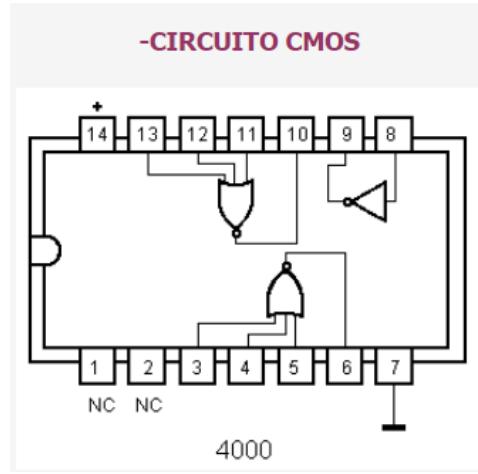
Los transistores CMOS tienen características que los diferencian notablemente de los bipolares:

Bajo consumo, puesto que una puerta CMOS sólo consume 0,01 mW en condiciones estáticas (cuando no cambia el nivel). Si opera con frecuencias elevadas comprendidas entre 5 y 10 MHz, el consumo es de 10 mw.

Los circuitos CMOS poseen una elevada inmunidad al ruido, normalmente sobre el 30 y el 45 % del nivel lógico entre el estado 1 y el 0. Este margen alto sólo es comparable con el de la familia HTL.

Las desventajas que sobresalen en la familia CMOS son su baja velocidad, con un retardo típico de 25 a 50 ns o más, especialmente cuando la puerta tiene como carga un elemento capacitivo; también hay que citar que el proceso de fabricación es más caro y complejo y, finalmente, la dificultad del acoplamiento de esta familia con las restantes.

Una característica muy importante de la familia CMOS es la que se refiere al margen de tensiones de alimentación, que abarca desde los 3 a los 15 V, lo que permite la conexión directa de los componentes de dicha familia con los de la TTL, cuando se alimenta con 5 V a los circuitos integrados CMOS.



4.4 DIFERENCIA DE ENTRE TTL Y CSMO:

Las diferencias más importantes entre ambas familias son: a)

- En la fabricación de los circuitos integrados se usan transistores bipolares para el TTL y transistores MOSFET para la tecnología CMOS.
- Los CMOS requieren de mucho menos espacio (área en el CI) debido a lo compacto de los transistores MOSFET. Además debido a su alta densidad de integración, los CMOS están superando a los CI bipolares en el área de integración a gran escala, en LSI - memorias grandes, CI de calculadora, microprocesadores-, así como VLSI.
- Los circuitos integrados CMOS son de menor consumo de potencia que los TTL.
- Los CMOS son más lentos en cuanto a velocidad de operación que los TTL.
- Los CMOS tienen una mayor inmunidad al ruido que los TTL.
- Los CMOS presentan un mayor intervalo de voltaje y un factor de carga más elevado que los TTL.

En resumen podemos decir que:

- TTL: diseñada para una alta velocidad.
- CMOS: diseñada para un bajo consumo.

Actualmente dentro de estas dos familias se han creado otras, que intentan conseguir lo mejor de ambas: un bajo consumo y una alta velocidad. La familia lógica ECL se encuentra a caballo entre la TTL y la CMOS. Esta familia nació como un intento de conseguir la rapidez de TTL y el bajo consumo de CMOS, pero en raras ocasiones es empleada.

TAMAÑOS:

- SSI (Small Scale Integration) pequeño nivel: de 10 a 100 transistores
- MSI (Medium Scale Integration) medio: 101 a 1000 transistores
- LSI (Large Scale Integration) grande: 1001 a 10 000 transistores
- VLSI (Very Large Scale Integration) muy grande: 10 001 a 100.000 transistores
- ULSI (Ultra Large Scale Integration) ultra grande: 100 001 a 1 000 000 transistores
- GLSI (Giga Large Scale Integration) giga grande: más de un millón de transistores.

ENLACES PARA INFORMACIÓN:

- http://electronica.ugr.es/~amroldan/asignaturas/curso04-05/ftc/pdf/trab_familia_cmos.pdf
- <https://tutorialcid.es.tl/Familia-CMOS.htm>

5. Sistema binario y hexadecimal

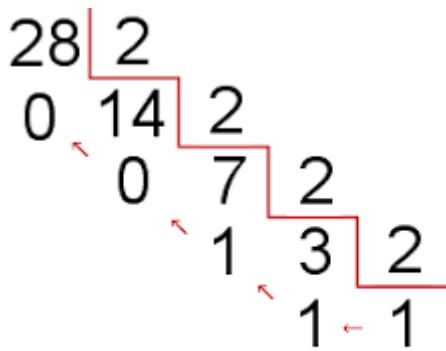
5.1 SISTEMA BINARIO:

“El sistema de numeración Binario o código binario es un sistema numérico que es utilizado para representar textos, datos o simplemente para procesar instrucciones en una computadora o en un dispositivo informático de cualquier tipo. Dicho sistema de numeración como su nombre lo indica se basa en sólo dos dígitos (bits) el cero (0) y el uno (1)”.

El sistema de numeración binario es utilizado básicamente por los microprocesadores de los dispositivos informáticos para detectar la ausencia o presencia de señal o de bits como también se les conoce. La facilidad que tiene el microprocesador de agrupar hasta 8 bits en una sola señal, se denomina velocidad de transferencia de datos y este grupo de bits forman un byte, la unidad base de medida de datos en informática.

El sistema de numeración binario tiene muchos usos, desde la programación de microprocesadores, transferencia de datos, cifrado de información, hasta comunicación digital, electrónica y otras áreas relacionadas.

Para obtener un número binario partiendo desde un número decimal lo único que debemos hacer es dividir este número entre el 2 hasta que nos sea imposible seguir con la división, es decir hacemos la primera división obtenemos el número entero y nos guardamos el resto, después dividiremos el el número entero que hemos obtenido entre 2 y así sucesivamente, a la hora de escribirlo de forma binaria debemos coger el resto de las divisiones hechas empezando por el último



Para hacer el camino inverso debemos coger el número binario y multiplicar cada número por 2 , elevado a la posición en la que se encuentre empezando por n-1 y sumarlo todo

The diagram shows the conversion of the binary number 110101_2 to its decimal equivalent 53_{10} . The binary digits are multiplied by powers of 2 and summed:

$$1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 32 + 16 + 0 + 4 + 0 + 1 = 53$$

$110101_2 = 53_{10}$

En el sistema hexadecimal los números se representan con dieciséis símbolos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E y F. Se utilizan los caracteres A, B, C, D, E y F representando las cantidades decimales 10, 11, 12, 13, 14 y 15 respectivamente, porque no hay dígitos mayores que 9 en el sistema decimal.

El sistema numérico hexadecimal (hex) se usa frecuentemente cuando se trabaja con computadores porque se puede usar para representar números binarios de manera más legible.

Para pasar de sistema decimal a hexadecimal seguiremos la misma fórmula que hemos utilizado para conseguir el binario, pero esta vez en vez de dividirlo por el 2 lo dividiremos por el 16 ya que es la base del sistema hexadecimal

$$\begin{array}{r}
 18541 \quad | \textcolor{green}{16} \\
 \underline{13} \qquad \qquad | \textcolor{green}{16} \\
 \underline{\underline{13}} \qquad \qquad | \textcolor{green}{16} \\
 \qquad \qquad \qquad | \textcolor{green}{16} \\
 \qquad \qquad \qquad \qquad | \textcolor{blue}{4}
 \end{array}$$

Para pasar de sistema hexadecimal a decimal también nos basaremos en lo que hacemos a la hora de pasar de binario a decimal pero esta vez en vez de elevar el 2 al valor de la posición del número elevaremos el número 16

$$\begin{array}{l}
 \text{Hexadecimal: F 1 2 A 4} \\
 \text{Equivalente decimal: } 15 \quad 1 \quad 2 \quad 10 \quad 4 \\
 \text{Potencias: } 16^4 \quad 16^3 \quad 16^2 \quad 16^1 \quad 16^0
 \end{array}$$

Ahora bien si queremos pasar un número binario a hexadecimal lo primero que debemos hacer es separar el número binario en bloques de 4, y escribir estos bloques en hexadecimal

1	0	1	0	1	0	0	1	0	0	1	1	1	0	1
2	A	9	0	F	5									

$$1010101001000011110101_{(2)} = 2A90F5_{(16)}$$

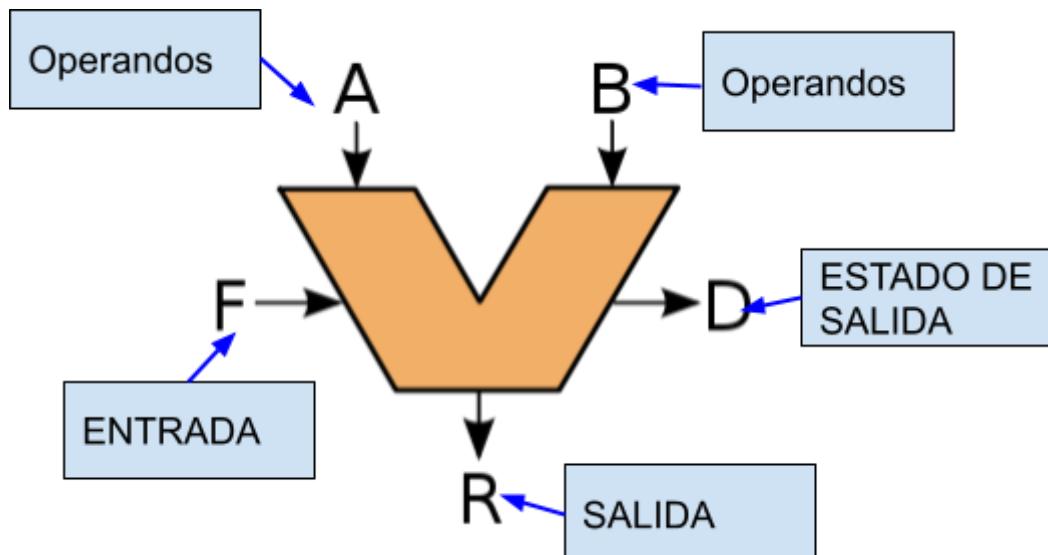
Por último para pasar de hexadecimal a binario debemos escribir cada número hexadecimal en binario, 1 por 1

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

6. ¿Qué es una ALU?

En computación, la unidad aritmética lógica o unidad aritmético-lógica, también conocida como ALU (siglas en inglés de arithmetic logic unit), es un circuito digital que realiza operaciones aritméticas (suma, resta) y operaciones lógicas (SI, Y, O, NO) entre los valores de los argumentos (uno o dos)

Por mucho, los circuitos electrónicos más complejos son los que están construidos dentro de los chips de microprocesadores modernos. Por lo tanto, estos procesadores tienen dentro de ellos un ALU muy complejo y potente. De hecho, un microprocesador moderno puede tener múltiples núcleos, cada núcleo con múltiples unidades de ejecución, cada una de ellas con múltiples ALU.



7. ¿Cuál es la diferencia entre un circuito combinacional y secuencial?

Los sistemas combinacionales están formados por un conjunto de compuertas interconectadas cuya salida, en un momento dado, está únicamente en función de la entrada, en ese mismo instante. Por esto se dice que los sistemas combinacionales no cuentan con memoria.

Los sistemas secuenciales en cambio, son capaces de tener salidas no sólo en función de las entradas actuales, sino que también de entradas o salidas anteriores. Esto se debe a que los sistemas secuenciales tienen memoria y son capaces de almacenar información a través de sus estados internos.

7.1 ¿Cómo conseguir el efecto memoria?

Un sistema secuencial dispone de elementos de memoria cuyo contenido puede cambiar a lo largo del tiempo. El estado de un sistema secuencial viene dado por el contenido de sus elementos de memoria. Es frecuente que en los sistemas secuenciales exista una señal que inicia los elementos de memoria con un valor determinado: señal de inicio (reset). La señal de inicio determina el estado del sistema en el momento del arranque (normalmente pone toda la memoria a cero). La salida en un instante concreto viene dada por la entrada y por el estado anterior del sistema. El estado actual del sistema, junto con la entrada, determinará el estado en el instante siguiente ⇒ realimentación.

7.3 ¿Cuáles son los tipos de biestables más populares?

En electrónica, un biestable, en inglés llamados *flip-flop* y *latch*, es un circuito que tiene dos estados estables y puede almacenar información. Se puede hacer que cambie de estado mediante señales aplicadas a una o más entradas de control y tiene una o dos salidas. Los circuitos biestables son componentes fundamentales de los sistemas electrónicos digitales como las memorias de las computadoras, dispositivos de comunicación digital y muchos otros tipos de sistemas.

Los circuitos biestables tienen la capacidad de permanecer en uno de dos estados posibles durante un tiempo indefinido en ausencia de perturbaciones. El paso de un estado a otro se realiza variando sus entradas. Dependiendo del tipo de dichas entradas los biestables se dividen en:

Asíncronos: solamente tienen entradas de control. El más empleado es el biestable RS.

Síncronos: además de las entradas de control posee una entrada de sincronismo o de reloj.

La entrada de sincronismo puede ser activada por nivel (alto o bajo) o por flanco (de subida o de bajada). Dentro de los biestables síncronos activados por nivel están los tipos RS y D, y dentro de los activos por flancos los tipos JK, T y D.

7.4 BIESTABLE RS

Dispositivo de almacenamiento temporal de 2 estados (alto y bajo), cuyas entradas principales permiten al ser activadas:

- R: el borrado (*reset* en inglés), puesta a 0 o nivel bajo de la salida.
- S: el grabado (*set* en inglés), puesta a 1 o nivel alto de la salida

Si no se activa ninguna de las entradas, el biestable permanece en el estado que poseía tras la última operación de borrado o grabado. En ningún caso deberían activarse ambas entradas a la vez, ya que esto provoca que las salidas directa (Q) y negada (Q') queden con el mismo valor: a bajo, si el flip-flop está construido con puertas NOR, o alto, si está construido con puertas NAND. El problema de que ambas salidas queden al mismo estado está en que al desactivar ambas entradas no se podrá determinar el estado en el que quedaría la salida. Por eso, en las tablas de verdad, la activación de ambas entradas se contempla como caso no deseado (N. D.).

Modo de operación	Entradas		Salidas	
	R	S	Q	Q'
Prohibido	0	0	1	1
Set	0	1	1	0
Reset	1	0	0	1
Mantenimiento	1	1	No cambia	

7.5 BIESTABLE RS (Set Reset) asíncrono:

____ Solo posee las entradas R y S. Se compone internamente de dos puertas lógicas NAND o NOR, según se muestra en la siguiente figura:

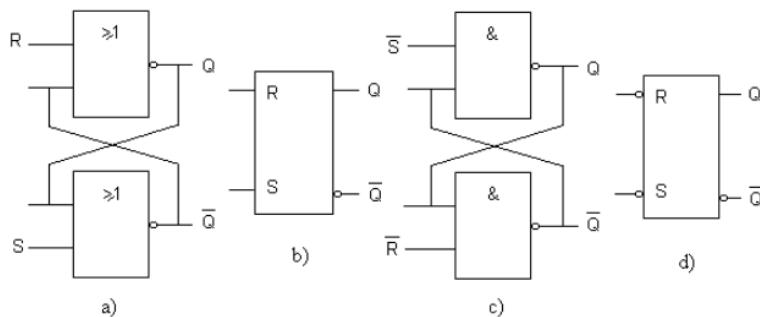


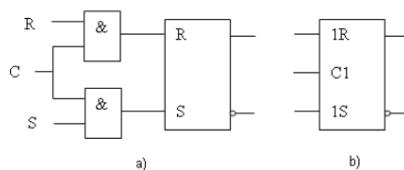
Tabla de verdad biestable RS

R	S	Q (NOR)	Q (NAND)
0	0	q	N. D.
0	1	1	0
1	0	0	1
1	1	N. D.	q

N. D.= Estado no deseado q= Estado de memoria

7.6 BIESTABLE RS SÍNCRONO:

Además de las entradas R y S, posee una entrada C de sincronismo cuya misión es la de permitir o no el cambio de estado del biestable. En la siguiente figura se muestra un ejemplo de un biestable síncrono a partir de una asíncrona, junto con su esquema normalizado:



Su tabla de verdad es la siguiente:

Tabla de verdad
biestable RS

C	R	S	Q (NOR)
0	X	X	q
1	0	0	q
1	0	1	1
1	1	0	0
1	1	1	N. D.

X=no importa

7.7 BIESTABLE D (Data o Delay):

El flip-flop D resulta muy útil cuando se necesita almacenar un único bit de datos (1 o 0). Si se añade un inversor a un flip-flop S-R obtenemos un flip-flop D básico. El funcionamiento de un dispositivo activado por el flanco negativo es, por supuesto, idéntico, excepto que el disparo tiene lugar en el flanco de bajada del impulso del reloj. Recuerde que Q sigue a D en cada flanco del impulso de reloj.

Para ello, el dispositivo de almacenamiento temporal es de dos estados (alto y bajo), cuya salida adquiere el valor de la entrada D cuando se activa la entrada de sincronismo, C. En función del modo de activación de dicha entrada de sincronismo, existen dos tipos:

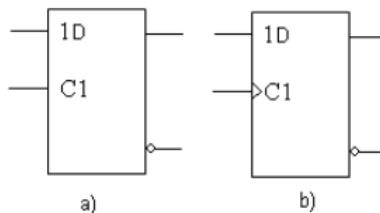
- **Activo por nivel** (alto o bajo), también denominado registro o cerrojo (*latch* en inglés).
- **Activo por flanco** (de subida o de bajada).

La ecuación característica del biestable D que describe su comportamiento es:

$$Q_{\text{siguiente}} = D$$

y su [tabla de verdad](#):

D	Q	$Q_{\text{siguiente}}$
0	X	0
1	X	1
X=no importa		



7.8 BIESTABLE T (Toggle):

Dispositivo de almacenamiento temporal de 2 estados (alto y bajo). El biestable T cambia de estado ("toggle" en inglés) cada vez que la entrada de sincronismo o de reloj se dispara mientras la entrada T está a nivel alto.

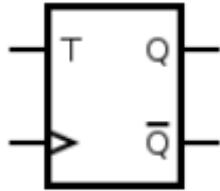
Si la entrada T está a nivel bajo, el biestable retiene el nivel previo. Puede obtenerse al unir las entradas de control de un biestable JK, unión que se corresponde a la entrada T.

La ecuación característica del biestable

T que describe su comportamiento es:

$$Q_{\text{siguiente}} = T \oplus Q$$

y la tabla de verdad:



T	Q	$Q_{\text{siguiente}}$
0	0	0
0	1	1
1	0	1
1	1	0

7.9 BIESTABLE JK:

Es versátil y es uno de los tipos de flip-flop más usados. Su funcionamiento es idéntico al del flip-flop S-R en las condiciones SET, RESET y de permanencia de estado. La diferencia está en que el flip-flop J-K no tiene condiciones no válidas como ocurre en el S-R.

Este dispositivo de almacenamiento es temporal que se encuentra dos estados (alto y bajo), cuyas entradas principales, J y K, a las que debe el nombre, permiten al ser activadas:

- **J**: El grabado (*set* en inglés), puesta a 1 o nivel alto de la salida.
- **K**: El borrado (*reset* en inglés), puesta a 0 o nivel bajo de la salida.

Si no se activa ninguna de las entradas, el biestable permanece en el estado que poseía tras la última operación de borrado o grabado. A diferencia del biestable RS, en el caso de activarse ambas entradas a la vez, la salida adquirirá el estado contrario al que tenía.

La ecuación característica del biestable JK que describe su comportamiento es:

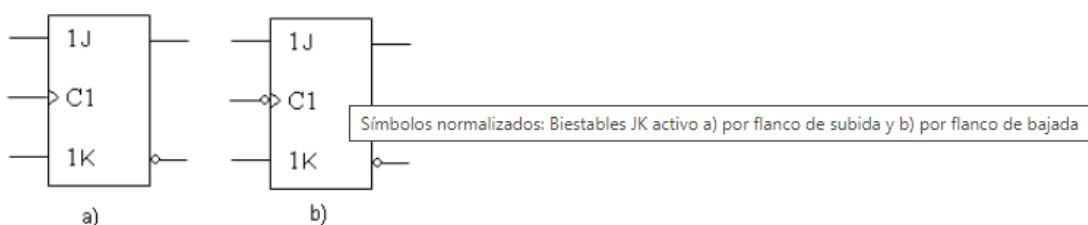
$$Q_{\text{siguiente}} = J\bar{Q} + \bar{K}Q$$

Y su [tabla de verdad](#) es:

J	K	Q	$Q_{\text{siguiente}}$
0	0	0	0
0	0	1	1
0	1	X	0
1	0	X	1
1	1	0	1
1	1	1	0
X=no importa			

7.10 BIESTABLE JK ACTIVO POR FLANCO:

Junto con las entradas J y K existe una entrada C de sincronismo o de reloj cuya misión es la de permitir el cambio de estado del biestable cuando se produce un flanco de subida o de bajada, según sea su diseño. Su denominación en inglés es J-K Flip-Flop Edge-Triggered. De acuerdo con la tabla de verdad, cuando las entradas J y K están a nivel lógico 1, a cada flanco activo en la entrada de reloj, la salida del biestable cambia de estado. A este modo de funcionamiento se le denomina modo de basculación (toggle en inglés).



7.11 BIESTABLE JK MAESTRO-ESCLAVO:

Aunque aún puede encontrarse en algunos equipos, este tipo de biestable, denominado en inglés *J-K Flip-Flop Master-Slave*, ha quedado obsoleto, ya que ha sido reemplazado por el tipo anterior.

Su funcionamiento es similar al JK activo por flanco: en el nivel alto (o bajo) se toman los valores de las entradas J y K y en el flanco de bajada (o de subida) se refleja en la salida.

J	K	Q	Q _{siguiente}
0	X	0	0
1	X	0	1
X	1	1	0
X	0	1	1

X=no importa

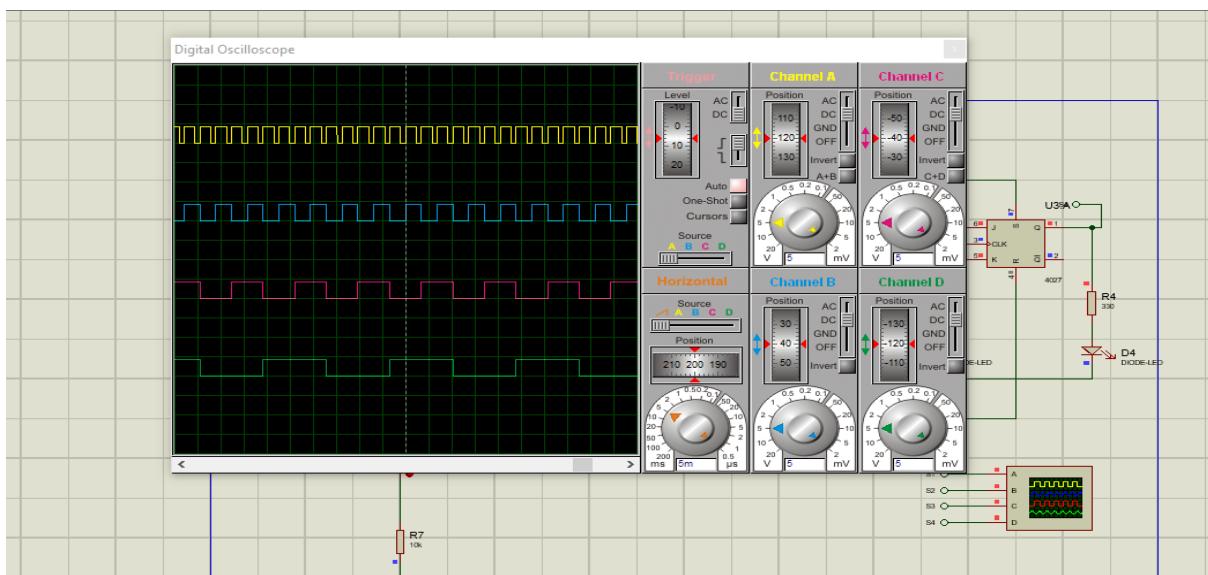
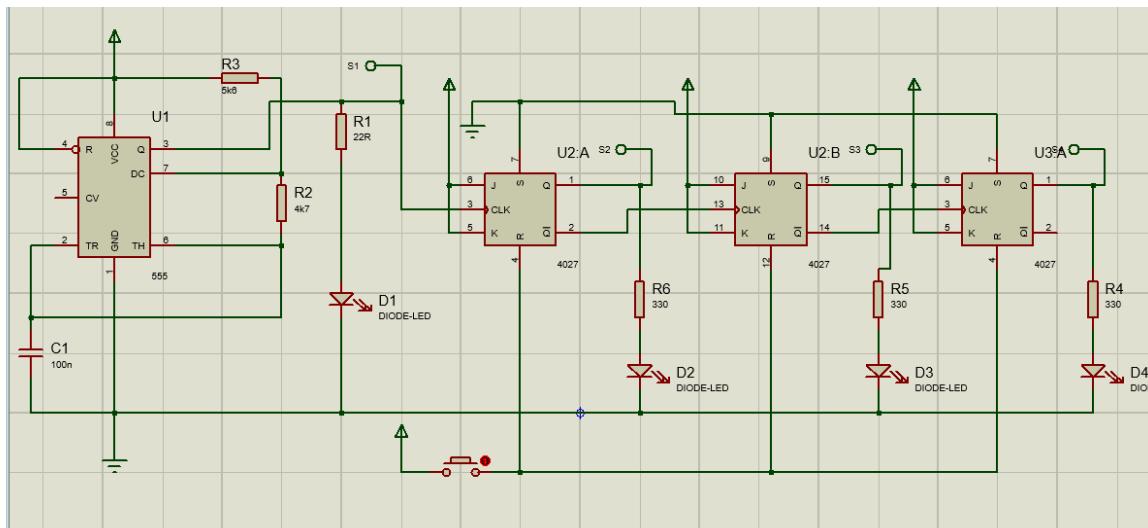
LINKS DE WEB:

- <https://es.wikipedia.org/wiki/Biestable>
- https://ocw.ehu.eus/pluginfile.php/42742/mod_page/content/1/Tema_6/_6_4.pdf

8. ¿Cómo conseguir un divisor de frecuencia con biestables?

Para conseguir un divisor de frecuencia primero necesitamos un aparato que mande una señal que en este caso hemos utilizado el temporizador 555. Este temporizador lo conectamos a unas resistencias y a un condensador y según el condensador que pongamos nos saldrá una señal distinta.

Una vez conectado el temporizador lo conectamos junto a un biestable JK que al unirlo lo que hace el JK es dividir entre 2 la señal que nos dé el Temporizador, es decir, cogiendo la referencia de la tabla de la verdad según el valor de la entrada si es 1 o es 0 nos actuara de distinta manera, en este caso, al estar los dos en 1 la señal cambia y una vez que nos funcione conectamos junto a otro JK que este al estar conectado junto al otro lo divide entre 4 y el siguiente entre 8, así conseguimos un divisor de frecuencia.



YOUTUBE:

- <https://www.youtube.com/watch?v=dMfygQecc48>
- <https://www.youtube.com/watch?v=YMbQ0HNHtd4>

8.1 ¿Cuáles son los puntos?

8.1.1 Combinacionales:

Las salidas en cualquier instante de tiempo dependen del valor de las entradas en ese mismo instante de tiempo (salvo los retardos propios de los dispositivos electrónicos).

Son, por tanto, sistemas sin memoria.

8.1.2 Secuenciales:

La salida del sistema va a depender del valor de las entradas en ese instante de tiempo y del estado del sistema; es decir, de la historia pasada del sistema. Son sistemas con memoria.

8.1.3 Variable binaria:

Es toda variable que solo puede tomar 2 valores, dos dígitos (dígitos=digital) que corresponden a dos estados distintos.

Estas variables las usamos para poner el estado en el que se encuentra un elemento de maniobra o entrada (por ejemplo un interruptor o un pulsador) y el de un receptor (por ejemplo una lámpara o un motor), siendo diferente el criterio que tomamos para cada uno.

8.1.4 Operaciones Lógicas:

Son las operaciones matemáticas que se usan en el sistema de numeración que se une el 0 y el 1.

$$\begin{array}{ll} \mathbf{a + b} & \left\{ \begin{array}{l} 0+0=0 \\ 0+1=1 \\ 1+0=1 \\ 1+1=1 \end{array} \right. \\ \mathbf{a \times b} & \left\{ \begin{array}{l} 0 \times 0=0 \\ 0 \times 1=0 \\ 1 \times 0=0 \\ 1 \times 1=1 \end{array} \right. \end{array}$$

8.1.5 Puertas lógicas:

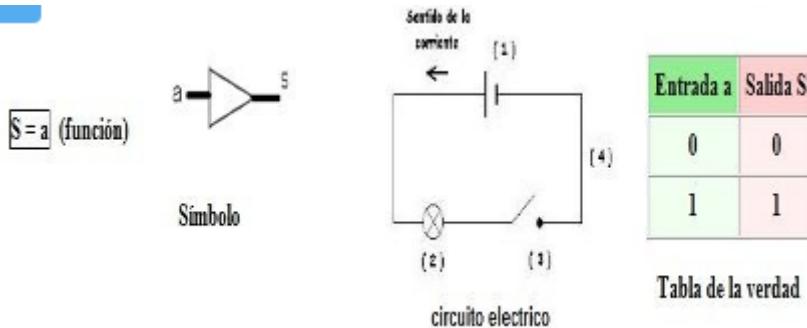
Son componentes electrónicos representados por un símbolo con una o dos entradas y una sola salida que realizan una función y que toman unos valores de salida en función de los que tenga en las entrada.

Las puertas lógicas también representan un circuito electrónico y tienen cada una su propia tabla de verdad, en la que vienen representados todos los posibles valores de entrada que puede tener y los que les corresponden de salida según su función .

8.1.6 Igualdad:

En este caso el valor de salida es siempre igual al del estado de la entrada.

El pulsador en estado 0, la lámpara está apagada, pero si pulsamos el pulsador estando en 1 la lámpara estará encendida

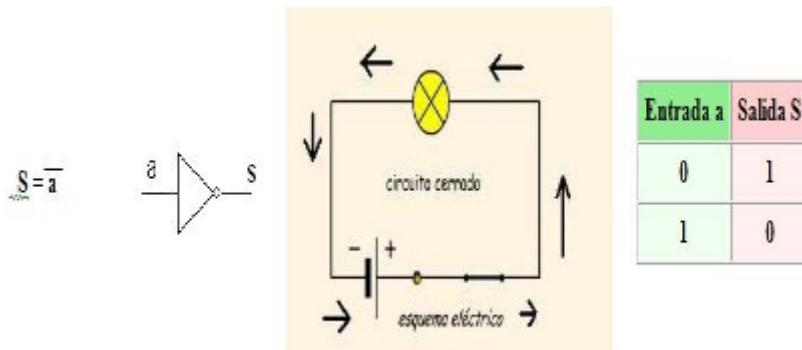


9. PUERTAS LÓGICAS

9.1 Puerta NO O NOT:

En este caso el valor de la entrada siempre es contraria a la del valor de salida.

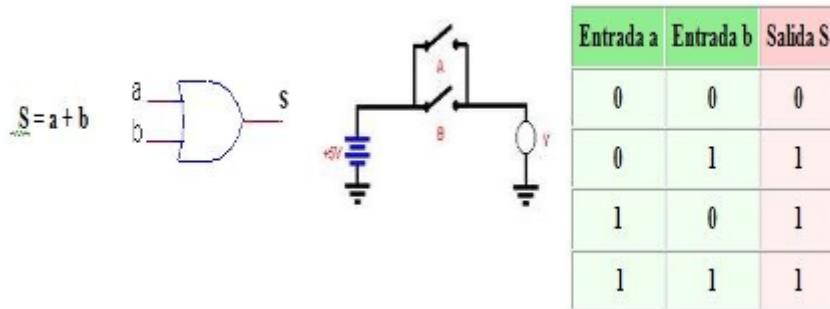
En las funciones, una línea sobre una variable significa que tomará el valor contrario.



En este caso podemos observar que cuando el valor de la entrada es 0 la salida tendrá un valor de 1

9.2 Puerta O u OR:

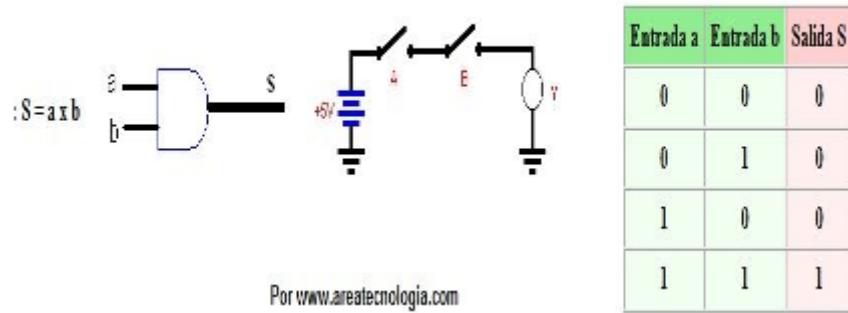
En este caso tenemos dos elementos de entrada, en este caso para que la entrada sea un 1 una de las entradas tendrá que estar pulsada, ya que la suma de las entradas será el valor que tendrá la salida.



9.3 Puerta AND:

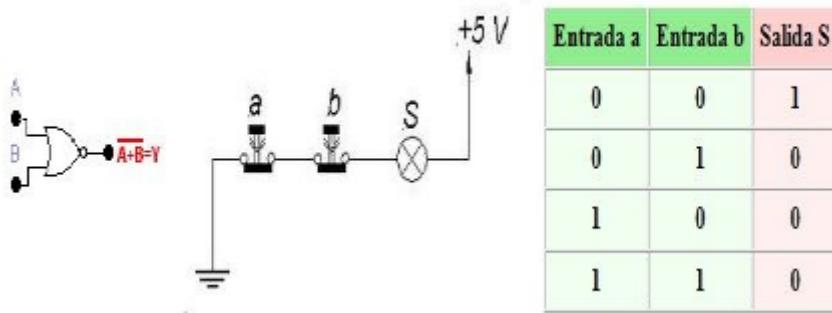
En este caso a caso a diferencia de la anterior debemos tener pulsadas las dos salidas para que la salida esté en 1

Es decir los valores de entrada en vez de sumarse se multiplicarán.



9.4 Puerta NOR:

En este caso tenemos dos entradas, pero la salida nos dará el valor contrario a la de las entradas, es decir para que en la salida nos de el valor 1 las dos entradas deberán estar en 0, ya que en la salida obtendremos el resultado inverso de la suma del valor de las dos entradas.



9.4 Puerta NAND:

Como están en paralelo si una de las entradas está sin accionar es decir en 0, la salida estará encendida o con valor 1, sólo en el caso de que las dos entradas estén accionadas nuestra salida estará apagada, en este caso nuestra salida obtendrá el valor inverso de la multiplicación de las entradas.

