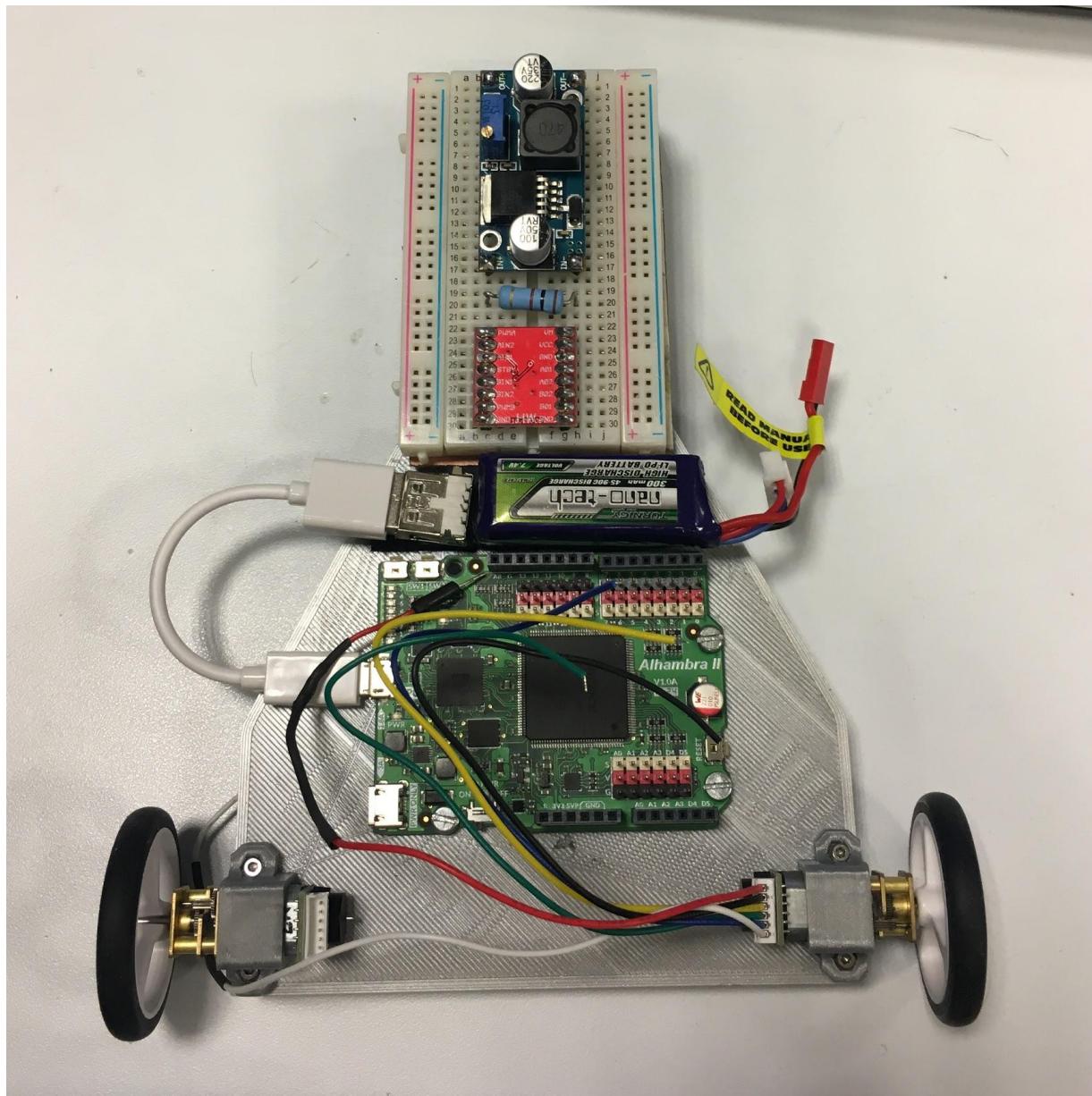


# PORTADA



Egileak:

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

<b>1. MATERIAL QUE HEMOS USADO</b>	<b>3</b>
1.1 PLACA ALHAMBRA II:	3
1.1.1 Características:	4
1.1.2 ENLACE:	5
1.2 kit CHASIS de carro con 2 ruedas:	5
1.3 cable USB:	6
1.4 MOTOR:	7
<b>1.5 CONTROLAR DOS MOTORES DC CON ARDUINO Y DRIVER TB6612FNG:</b>	<b>7</b>
1.6 PROTOBOARD:	9
1.6.1 Especificaciones:	10
1.7 3 Tornillos M3x10 mm:	10
1.7.1 Descripción:	10
1.7.1.1 Especificaciones tuerca:	10
1.8 2 Tuercas M3	10
1.8.1 Descripción:	10
1.8.1.1 Especificaciones tuerca:	10
1.9 SENSOR CNY70	11
1.9.1 ESPECIFICACIONES TÉCNICAS	11
1.10 Cables Dupont Macho Hembra de 20 cm:	11
1.10.1 Especificaciones:	11
1.11 BATERÍA:	12
1.11.1 Características:	12
1.11.1.1 Dimensiones:	12
1.12 PINES:	12
1.13 2 DIODOS ZENER:	13
1.14 FUENTE CONMUTADA LM2596:	13
1.14.1 Características:	14
<b>2. ELECTRÓNICA DIGITAL</b>	<b>15</b>
2.1 ¿Qué es una señal digital?	15
2.2 VENTAJAS DE LAS SEÑALES DIGITALES	15
2.3 DESVENTAJAS	16
2.4 ¿Cuál es la diferencia entre una señal digital y una analógica?	16
<b>3. Puertas lógicas</b>	<b>16</b>
3.1 ¿Qué es una puerta lógica? ¿Cuáles están ahí y cuál es la "verdad de cada tabla"(egiaren taula)?	17
3.2 De la tabla de verdad a las funciones lógicas del álgebra de Boole ¿De qué sirve el método de Karnaugh? Intente usar.	17
3.3 ¿Cómo se puede implementar un circuito que consta de puertas lógicas?	22
3.4 ¿Cómo se pueden crear puertas NAND para crear equivalentes a otras puertas? ¿Para qué es esto? ¿es esto útil?	24
3.5 Para crear AND utilizando NAND	24
3.6 OR con una NAND	25
3.6 NOR con NAND	25
3.6 X or con NAND	26
3.7 XNOR con NAND	27

<b>4. ¿En qué se diferencian TTL y CMOS?</b>	<b>28</b>
4.1 TTL (Transistor- Transistor Logic) o "Lógica Transistor a Transistor".	28
4.2 CARACTERÍSTICAS DEL TTL:	28
4.3 CSMO:	29
4.4 DIFERENCIA DE ENTRE TTL Y CSMO:	30
<b>5. Sistema binario y hexadecimal</b>	<b>32</b>
5.1 SISTEMA BINARIO:	32
<b>6. ¿Qué es una ALU?</b>	<b>35</b>
<b>7. ¿Cuál es la diferencia entre un circuito combinacional y secuencial?</b>	<b>35</b>
7.1 ¿Cómo conseguir el efecto memoria?	36
7.3 ¿Cuáles son los tipos de biestables más populares?	36
7.4 BIESTABLE RS	37
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:	38
7.6 BIESTABLE RS SÍNCRONO:	38
7.7 BIESTABLE D (Data o Delay):	39
7.8 BIESTABLE T (Toggle):	39
7.9 BIESTABLE JK:	40
7.10 BIESTABLE JK ACTIVO POR FLANCO:	41
7.11 BIESTABLE JK MAESTRO-ESCLAVO:	42
<b>8. ¿Cómo conseguir un divisor de frecuencia con biestables?</b>	<b>42</b>
8.1 ¿Cuáles son los puntos?	44
8.1.1 Combinacionales:	44
8.1.2 Secuenciales:	44
8.1.3 Variable binaria:	44
8.1.4 Operaciones Lógicas:	45
8.1.5 Puertas lógicas:	45
8.1.6 Igualdad:	45
<b>9. PUERTAS LÓGICAS</b>	<b>46</b>
9.1 Puerta NO O NOT:	46
9.2 Puerta O u OR:	47
9.3 Puerta AND:	47
9.4 Puerta NOR:	48
9.4 Puerta NAND:	48
<b>10. TABLA DE LA VERDAD:</b>	<b>49</b>
10.1 FUNCIÓN LÓGICA:	50
<b>11. CIRCUITOS COMBINACIONALES</b>	<b>53</b>
11.1 MULTIPLEXORES:	53
11.2 DEMULTIPLEXOR	53
11.3 DECODIFICADOR	54
11.4 CODIFICADOR:	54

<b>12. ¿Qué es FPGA ?</b>	<b>54</b>
12.1 Ze abantaila ditu?	56
<b>13. ¿Qué es Alhambra II-a? ¿Qué características tiene?</b>	<b>56</b>
13.1 Características de la placa Alhambra II :	57
13.2 Zer da PWM-a?	59
<b>14. GitHub</b>	<b>60</b>
14.1 ¿QUÉ ES?	60
14.3 ¿Qué es un repositorio de GITHUB?	61
<b>15. ¿QUÉ ES UN WIKI?</b>	<b>61</b>
<b>16. CONCLUSIONES</b>	<b>62</b>

# 1. MATERIAL QUE HEMOS USADO

## 1.1 PLACA ALHAMBRA II:

### 1.1.1 Características:

- Placa de desarrollo FPGA iCE40HX4K (Lattice) (8K con cadena de herramientas de código abierto)
- Hardware abierto
- De código abierto: Compatible con el código abierto cadena de herramientas Icestorm e Icestudio
- Similar al de Arduino
- Puedes reutilizar la mayoría de los escudos disponibles
- Controla tus robots / printbots desde un FPGA
- El dispositivo USB FTDI 2232H permite la programación FPGA y la interfaz UART a una PC
- Interruptor electrónico de ENCENDIDO / APAGADO de la placa(apague su robot móvil fácilmente, encenderlo no sobrecarga su puerto USB)
- 8 LED de uso general (LED de usuario)
- 2 botones de propósito general
- Memoria flash de 32Mb para hasta 30 flujos de bits o datos de usuario
- 20 pines de entrada / salida de 3.3V (5V tolerante)
- Todos los pines de E / S incluyen resistencias en serie de 200 ohmios para la activación directa de LED
- Convertidor A / D de 4 canales y 12 bits
- Pines de arranque en frío accesibles desde GPIO
- Reguladores de conmutación para procesamiento de alta frecuencia a baja potencia de entrada
- Oscilador MEMS de 12 MHZ
- Botón de reinicio
- Fuente de alimentación USB, dos conectores (hasta 4.8A)
- Los pines de alimentación y los pines de E / S permiten cortocircuito permanente

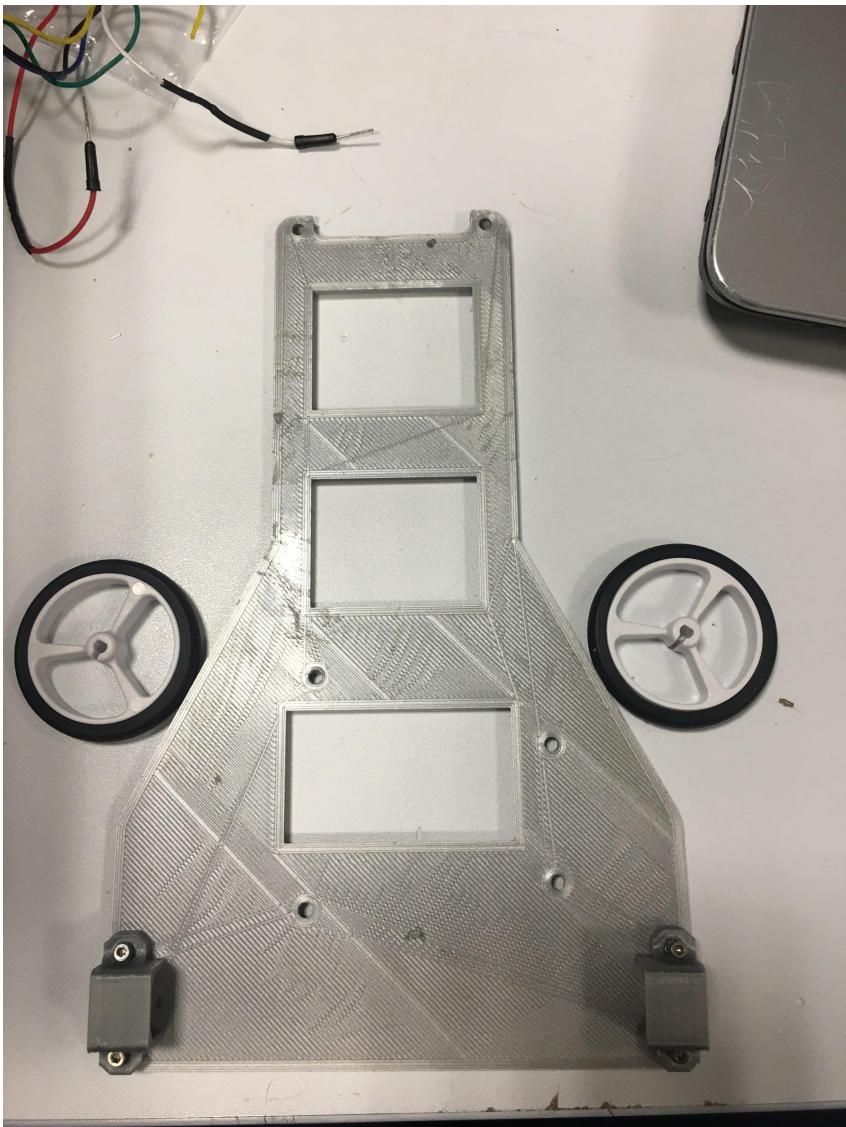


### **1.1.2 ENLACE:**

[https://github.com/FPGAwars/Alhambra-II-FPGA/blob/master/doc/pinout/Alhambra%20II%20V1.0A%20-%20Pinout\\_v1.0\\_rev%202%20-%2020150%20dpi.png](https://github.com/FPGAwars/Alhambra-II-FPGA/blob/master/doc/pinout/Alhambra%20II%20V1.0A%20-%20Pinout_v1.0_rev%202%20-%2020150%20dpi.png)

### **1.2 kit CHASIS de carro con 2 ruedas:**

Este chasis lo hemos usado para la base de los elementos que necesitamos, esto es; para la alhambra II, la batería, la fuente comutada, el controlador de motores,...Y las ruedas, obviamente para que el coche vaya hacia adelante.



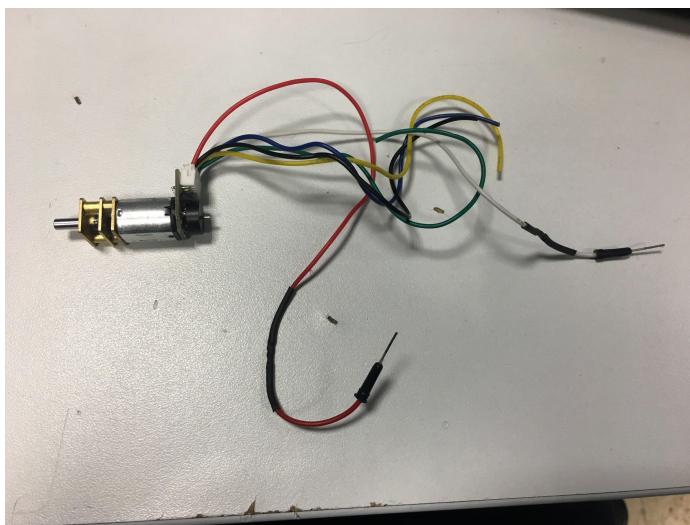
### 1.3 cable USB:

El cable USB lo hemos utilizado para pasar el programa de Icestudio del ordenador a la placa Alhambra II para que el coche haga lo que nosotros queremos que haga. También, donde pone “power only”, eso sirve para encender la alhambra II.



## 1.4 MOTOR:

En la siguiente imagen podemos ver un motor. En este proyecto hemos usado dos motores como los de la imagen, para que el coche siga líneas arranque y le de fuerzas a las ruedas para que el coche siga líneas hacia adelante. Estos dos motores que hemos usado funcionan a 6,2 Voltios.



## 1.5 CONTROLAR DOS MOTORES DC CON ARDUINO Y DRIVER TB6612FNG:

El TB6612FNG es un controlador (driver) de motores que nos permite manejar dos motores de corriente continua desde Arduino, variando tanto la velocidad como el sentido de giro.

Sin embargo, en el caso del TB6612FNG los puentes-H están formados por transistores MOSFET, en lugar de transistores BJT como en el L298N. Esto permite que el TB6612FNG tiene mejor eficiencia y menores dimensiones que el L298N.

El TB6612FNG también permite controlar intensidades de corriente superiores, siendo capaz de suministrar 1.2A por canal de forma continua, y 3.2A de pico. Recordar que el L298N tiene una intensidad máxima teórica de 2A, pero las pérdidas hacen que en la práctica sólo pueda suministrar 0.8-1A.

Además, el TB6612FNG no tiene la caída de tensión que sí tiene el L298N debido a sus transistores BJT, que podían llegar a ser superiores a 3V. En su lugar, el TB6612FNG se comporta como una pequeña resistencia de 0.5 Ohm.

Como puntos negativos, el TB6612FNG puede proporcionar tensiones inferiores, de hasta 13.5V (frente a los 35V del L298N). Además, es algo más difícil de montar, ya que las placas con L298N frecuentemente incorporan placas de conexión, que permiten conectar de forma sencilla el motor.

Finalmente, la protección contra corrientes inducidas son algo más limitadas que las del L298N, por lo que podemos tener algún reinicio de Arduino cuando alimentamos cargas medianas o grandes.

El TB6612FNG dispone de dos canales, por lo que es posible controlar dos motores de corriente continua de forma independiente. También puede controlar un único motor paso a paso aunque, en general, preferimos usar controladores específicos.

En cada canal podemos controlar el sentido de giro y la velocidad, para lo cual admite una señal PWM de frecuencia máxima de 100 kHz (muy por debajo del rango normal de PWM en Arduino)

El TB6612FNG dispone de protecciones térmicas, de inversión de corriente en la fuente de suministro de los motores, condensadores de filtrado en ambas líneas de alimentación, detección de bajo voltaje, y protecciones contra las corrientes inducidas en los motores.

El TB6612FNG también incorpora un modo de Standby, que desactiva por completo el controlador, entrando en un modo de ahorro de energía.

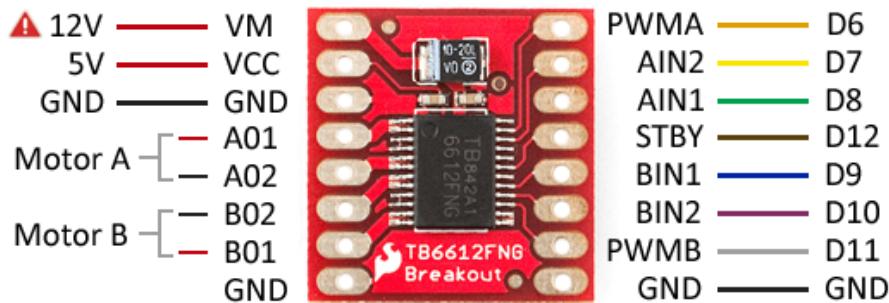
El controlador TB6612FNG puede ser empleado en proyectos de electrónica y robótica. Es ampliamente usado en proyectos electrónicos y robótica, por su sencillez de uso, bajo coste, y buena calidad precio.

Por un lado, suministramos la tensión que alimentará el motor desde una fuente de alimentación externa, mediante el pin VM. La tensión máxima es de 15V.

Además, tenemos que alimentar la electrónica del módulo mediante el pin VCC. El rango de tensión para VCC es 2.7 a 5.5V.

Para el control del módulo Los pines AIN1, AIN2 Y PWMA controlan el canal A, mientras que los pines BIN1, BIN2, y PWMB controlan el canal B.

Finalmente, el pin STBY controla el modo Standby. Debemos ponerlo en HIGH para activar el motor. Podemos conectarlo a un pin digital de Arduino, si queremos poder activar el modo Standby, o conectarlo a VCC si queremos dejarlo permanentemente desconectado.



Vin entre 5 y 15V. Al usar alimentación externa SIEMPRE poner con GND común.

## 1.6 PROTOBOARD:

En este proyecto hemos usado un protoboard pequeño. En este elemento hemos conectado la fuente de alimentación, el controlador de motores y los cables que necesitamos para cada conexión.



### 1.6.1 Especificaciones:

- Matriz: 10 x 17
- Calibre de cable: 29 - 20 AWG
- Color: Blanco, Rojo, Negro, Azul, Verde
- Peso: 13.6 g
- Dimensiones: 48.6 x 36.7 x 9.2 mm

## 1.7 3 Tornillos M3x10 mm:

### 1.7.1 Descripción:

Paquete con 10 tornillos M3 10 mm y 10 tuercas M3.

#### 1.7.1.1 Especificaciones tuerca:

- Rasca: M3
- Material: acero inoxidable.
- Color: Plateado



## 1.8 2 Tuerca M3

### 1.8.1 Descripción:

Paquete con 20 tuercas M3.



#### 1.8.1.1 Especificaciones tuerca:

- Rasca: M3
- Material: Acero inoxidable
- Color: Plateado
- Peso: 6.9 g

## 1.9 SENSOR CNY70

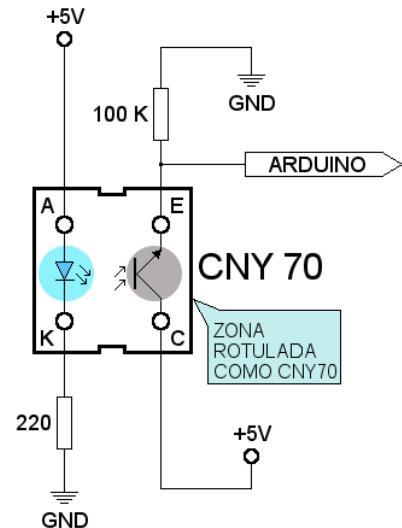
El CNY70 es un sensor óptico infrarrojo de corto alcance. El CNY70 tiene una construcción compacta donde la fuente de emisión de luz y el detector están dispuestos en la misma dirección para detectar la presencia de un objeto utilizando el haz de IR de reflexión en el objeto.

Su uso más común es para construir robots seguidores de Línea. Contiene un emisor de radiación infrarroja (fotodiodo) y un receptor (fototransistor). El fotodiodo emite un haz de radiación infrarroja, el fototransistor recibe ese haz de luz cuando se refleja sobre alguna superficie u objeto.

Para verificar el funcionamiento del diodo IR se puede utilizar una cámara digital como la de cualquier smartphone.

### 1.9.1 ESPECIFICACIONES TÉCNICAS

- Tipo de emisor: Fotodiodo IR
- Tipo de detector: fototransistor
- Dimensiones (L x W x H en mm): 7 x 7 x 6
- Distancia de funcionamiento máximo: <0.5 mm
- Longitud de onda del emisor: 950 nm



## 1.10 Cables Dupont Macho Hembra de 20 cm:

### 1.10.1 Especificaciones:

- Longitud: 20 cm
- Tipo de conector: Macho-Hembra
- Peso: 28.6 g



## 1.11 BATERÍA:

La batería es fundamental para este proyecto, sin batería el coche sigue líneas no funcionaría.

### 1.11.1 Características:

- 7.4V paquete de 2 celdas
- 300 mAh de carga
- 45C velocidad de descarga continua  
90 C velocidad de descarga pico
- Conector de carga JST-XH
- Conector descarga JST
- Peso 19g



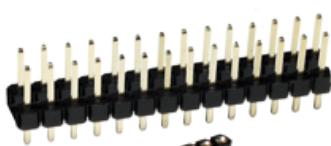
/

### 1.11.1.1 Dimensiones:

- 45mm x 17mm x 12mm.

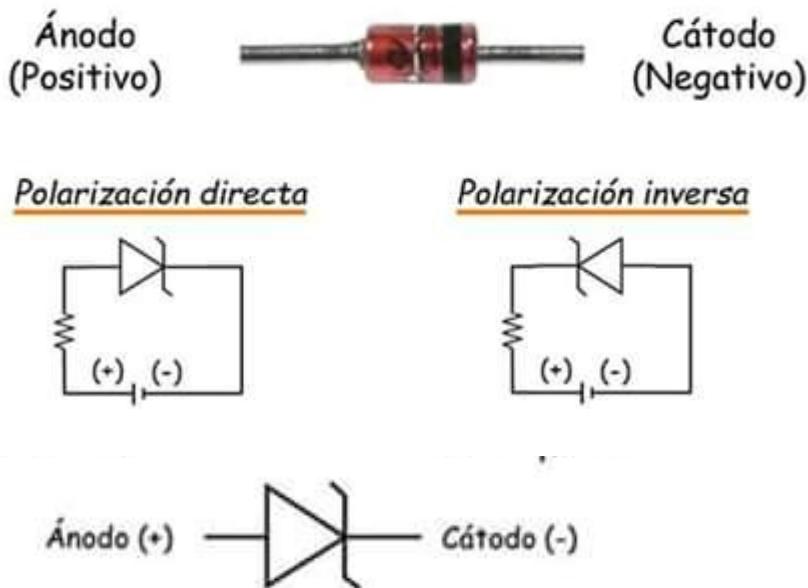
## 1.12 PINES:

Los pines usamos para hacer el conexionado de un elemento conectándolo con otro elemento que queremos.



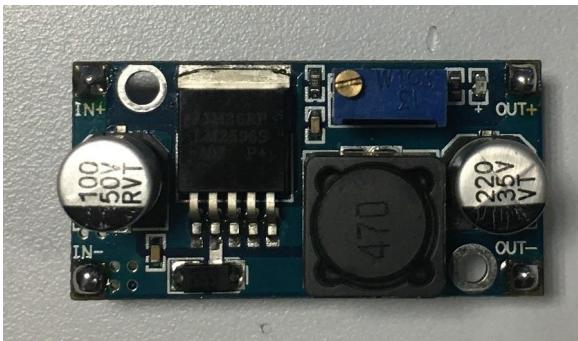
### 1.13 2 DIODOS ZENER:

El diodo ZENER se comporta similar a un diodo rectificador si cumple la polarización directa. Un diodo zener se aplica al cumplir una polarización inversa mantiene una tensión constante y actúa como estabilizador de tensión.



### 1.14 FUENTE CONMUTADA LM2596:

Este elemento es un circuito integrado ideal para el diseño de fuentes de conmutación tipo buck (reductor de Voltaje). Es capaz de suministrar una corriente de hasta 3A (con disipador de calor). Este dispositivo, suministra una tensión de salida ajustable de 1,25v a 35v DC, pero para ello necesita una tensión de entrada mayor a la de salida. La tensión de entrada debe ser por lo menos un par de voltios mayor a la de salida y estar comprendida entre 3,2v y 40v DC. Este módulo reduce al mínimo el uso de componentes externos para simplificar el diseño de fuentes de alimentación. Dispone de un potenciómetro de precisión multivuelta para ajustar el voltaje de salida al nivel deseado. El módulo convertidor LM2596 es una fuente de alimentación conmutada, así que su eficiencia es significativamente mayor en comparación con los populares reguladores lineales de tres terminales como el LM317 o los de la serie L78XX, especialmente con tensiones de entrada superiores, además puede suministrar intensidades de hasta 3A de manera mucho más eficiente.



#### 1.14.1 Características:

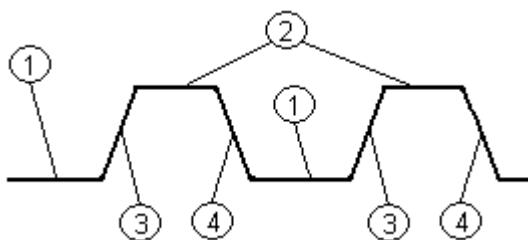
- Voltaje (Tensión) de entrada: 3.2 v ~ 40 v.
- Voltaje (Tensión) de salida: 1,25v -35V.
- Corriente de salida: 2A, 3A si se le añade un disipador adicional (max 12W).
- Eficiencia de conversión: 92%.
- Salida ondulación < 30 mv.
- Frecuencia de conmutación: 65 khz.
- Temperatura de trabajo:-40°C / +85°C.
- Tamaño: 43mm x 21mm x 14 mm (largo x ancho x altura).

**ATENCIÓN:** Una vez sobrepasado el límite (máximo o mínimo) todo el recorrido que sigamos haciendo será recorrido nulo. Para poder volver a modificar el valor de voltaje, tendremos que volver a recorrer todo el recorrido nulo que hayamos realizado, de lo contrario no cambiará el valor.

## 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 analogica 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? Intente 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.

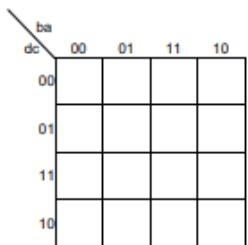
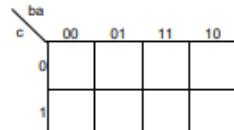
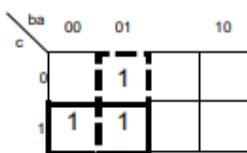


Diagrama de Karnaugh para 4 variables    Diagrama de Karnaugh para 3 variables



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.



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  $\{c,b,a\} = \{\}, 1,0,0 \quad \{1,0,1\}$  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:  $f = c \cdot b + b \cdot \bar{a} = b \cdot (a + c)$ .

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

	ba	dc	00	01	11	10
00						
01			1			
11						
10						

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

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

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

	ba	dc	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  $f = \bar{d}$

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

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

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

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

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

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

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

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

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

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

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

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

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

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

	ba	dc	00	01	11	10
00			1			
01						
11						
10						

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

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

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.

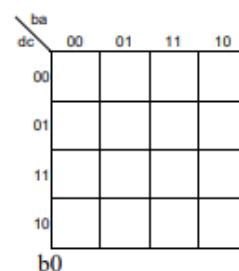
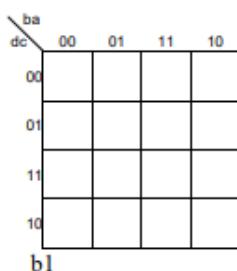
Como ya se sabe, la división entera de un número tiene dos partes: cociente y resto, ambos enteros. Nuestro circuito hará la división de un número BCD (o sea, del 0 al 9) entre 3. El cociente más grande será el obtenido al dividir el número más grande, que en BCD es el 9. Como  $9:3 = 3$  necesitamos 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  $\{(0,1,1,0), (0,0,1,0), (1,0,0,1)\}$  y b0 en las posiciones  $\{(0,1,1,0), (0,0,1,0), (1,0,1,0)\}$

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

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

Ahora buscamos grupos de los más grandes posibles. Vemos que para b1 tendremos dos grupos de dos: uno formado por  $\{(0,1,1,0), (0,1,1,0)\}$  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	b1	1	1		

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

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  $dcb = 011$  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:

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

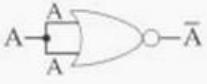
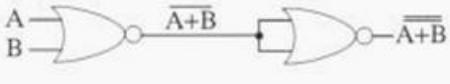
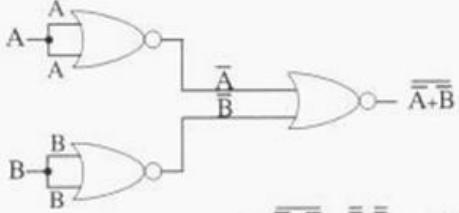
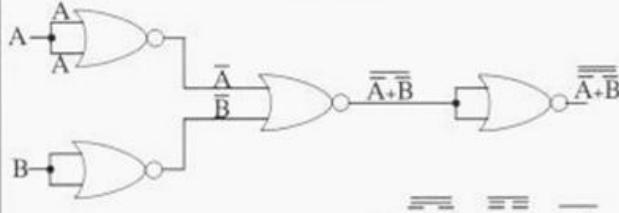
### 3.3 ¿Cómo se puede implementar un circuito que consta de puertas lógicas?

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.

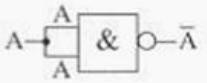
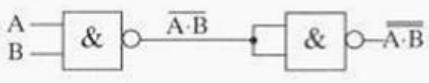
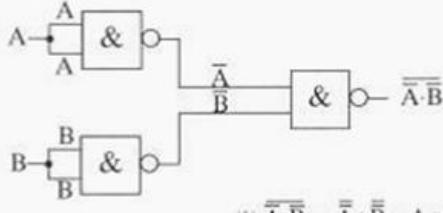
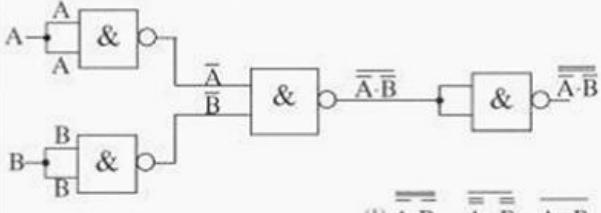
Este método de implementación solo se puede realizar con puertas NAND o NOR, ya que solo estas dos puertas lógicas son universales, es decir se puede realizar cualquier circuito lógico y sustituir cualquier puerta empleando únicamente este tipo de puertas, para ello debemos seguir un cierto protocolo 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.

<p style="text-align: center;">puerta <i>NOT</i></p>  <p style="text-align: center;">Nota: <math>\overline{A+A} = \overline{A}</math></p>	<p style="text-align: center;">puerta <i>OR</i></p>  <p style="text-align: center;"><math>\overline{\overline{A+B}} = A+B</math></p>
<p style="text-align: center;">puerta <i>AND</i></p>  <p style="text-align: center;">(i) <math>\overline{\overline{A+B}} = \overline{\overline{A}\cdot\overline{B}} = A\cdot B</math></p>	<p style="text-align: center;">puerta <i>NAND</i></p>  <p style="text-align: center;">(i) <math>\overline{\overline{A+B}} = \overline{\overline{A}\cdot\overline{B}} = A+B</math></p>

De igual forma podemos utilizar puertas lógicas NAND:

<p style="text-align: center;">puerta <i>NOT</i></p>  <p style="text-align: center;">Nota: <math>\overline{A\cdot\overline{A}} = \overline{A}</math></p>	<p style="text-align: center;">puerta <i>AND</i></p>  <p style="text-align: center;"><math>\overline{A\cdot B} = A\cdot B</math></p>
<p style="text-align: center;">puerta <i>OR</i></p>  <p style="text-align: center;">(i) <math>\overline{\overline{A\cdot B}} = \overline{\overline{A}+\overline{B}} = A+B</math></p>	<p style="text-align: center;">puerta <i>NOR</i></p>  <p style="text-align: center;">(i) <math>\overline{\overline{A\cdot B}} = \overline{\overline{A}+\overline{B}} = A+B</math></p>

### 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

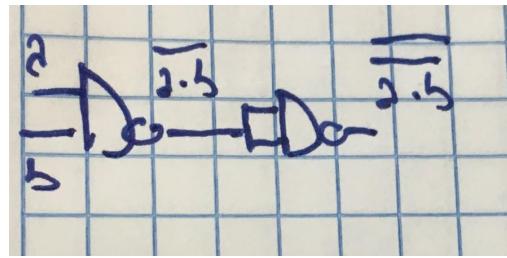
podemos ver que  $f$  es la inversa de la  $x$  es decir si hacemos un NOT de  $f$ , tendremos el resultado de  $x$ , y al saber como hacer una NOT con NAND podemos crear una AND utilizando NAND.

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

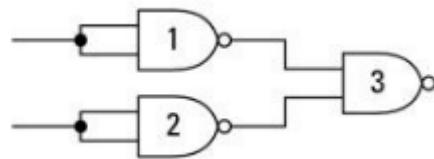
Sabemos que una OR es igual a  $a+b$ , para confundir esto primero debemos negar esto 2 veces para que siga siendo la misma operación, a continuación y para transformar el  $+$  en una multiplicación rompemos una de las negaciones.

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

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

TABLA DE NOR

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

$$\begin{aligned}
 & \text{Or con Nand} = \overline{\overline{a+b}} \\
 & \text{Nor con Nand} = \overline{\overline{\overline{a+b}}} \\
 & \text{Xor con Nand} = \overline{\overline{a+b}} \cdot \overline{\overline{\overline{a+b}}} = \overline{\overline{a+b}} \cdot \overline{a+b} = \overline{a+b} \cdot \overline{a+b} = \overline{a} \cdot \overline{b} + b \cdot a
 \end{aligned}$$

### 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

$$\begin{aligned}
 \text{XOR} &= a \oplus b = \overline{a} \cdot b + \overline{b} \cdot a = \overline{\overline{a} \cdot b + \overline{b} \cdot a} = \overline{\overline{a} \cdot b} \cdot \overline{\overline{b} \cdot a} = \overline{a} \cdot \overline{b} + b \cdot a \\
 & \text{Xor con Nand} \\
 & \text{Diagram: Two NAND gates in series. The first takes inputs } a \text{ and } \bar{b} \text{ and outputs } \overline{a} \cdot \overline{b}. \text{ This is fed into the second NAND gate along with input } b \text{ and its output is the result.}
 \end{aligned}$$

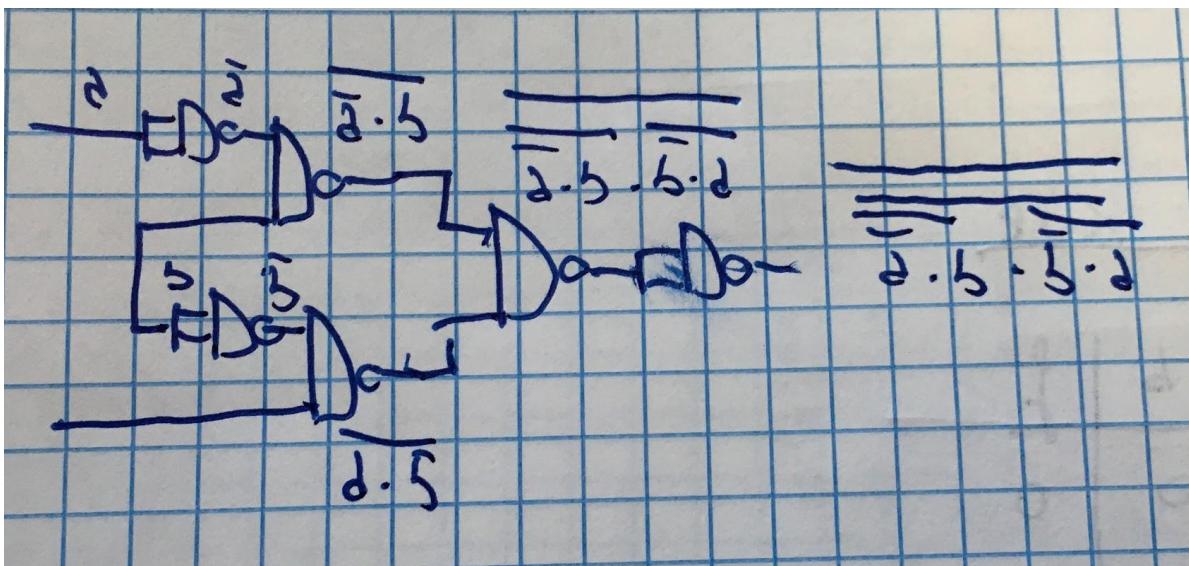
### 3.7 XNOR con NAND

en este caso sabemos que la XNOR es la negación de la XOR, por lo que deberemos negar la fórmula que hemos sacado anteriormente.

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

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

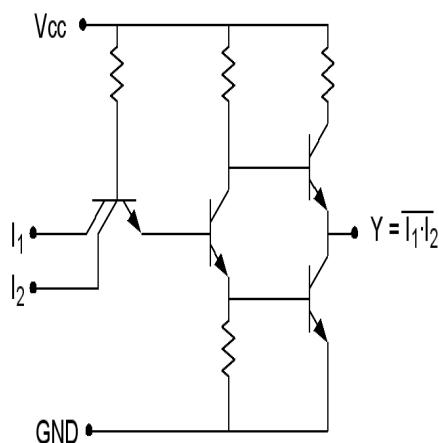
<u>Tabla NAND</u>			<u>Tabla XNOR</u>		
a	b	c	a	b	c
0	0	1	0	0	1
0	1	1	0	1	0
1	0	1	1	0	0
1	1	0	1	1	1
0	1	0	0	1	0
1	0	0	1	0	1
0	0	0	0	0	0



## 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  $V_{cc}$  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.

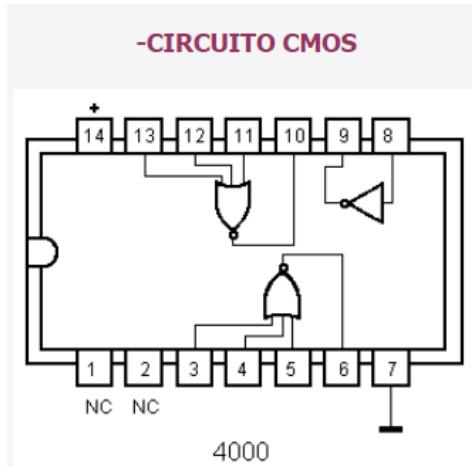
Los transistores CMOS tienen características que los diferencian notablemente de los bipolares:

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](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

$$\begin{array}{r}
 28 \quad | \quad 2 \\
 0 \quad 14 \quad | \quad 2 \\
 0 \quad 7 \quad | \quad 2 \\
 1 \quad 3 \quad | \quad 2 \\
 1 \quad 1
 \end{array}$$

$$28 = 11100_2$$

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

$$\begin{array}{c}
 1 \ 1 \ 0 \ 1 \ 0 \ 1_2 \\
 \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\
 \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \\
 32 + 16 + 0 + 4 + 0 + 1 = 53
 \end{array}$$

$$110101_2 = 53_{10}$$

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

$$\begin{array}{r}
 18541 \quad | \quad 16 \\
 \underline{13} \quad 1158 \quad | \quad 16 \\
 \underline{\underline{6}} \quad 72 \quad | \quad 16 \\
 \underline{\underline{\underline{8}}} \quad 4
 \end{array}$$

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

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

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

1010101001000011110101

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

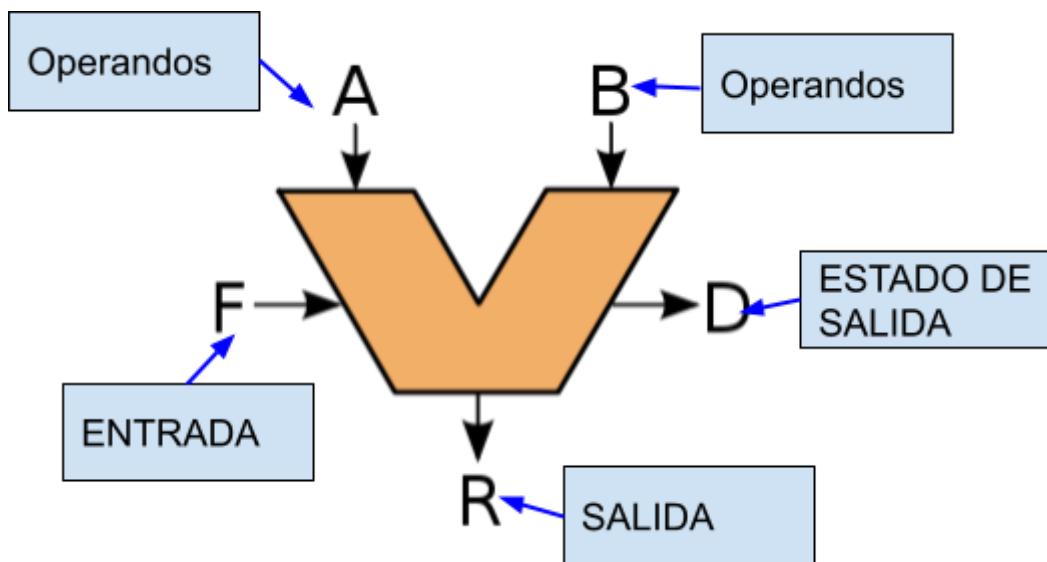
Por último para pasar de hexadecimal a binario debemos escribir cada número hexadecimal en binario, 1 por 1

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:

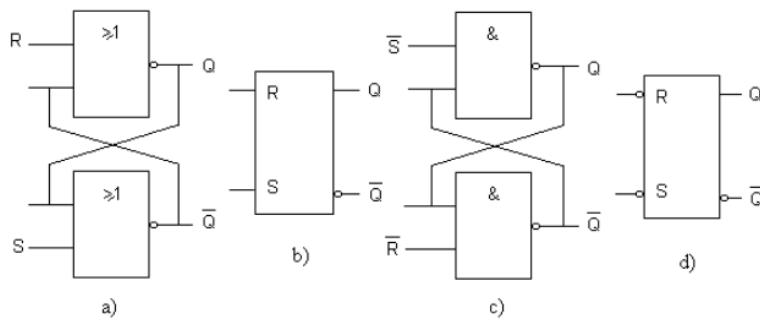


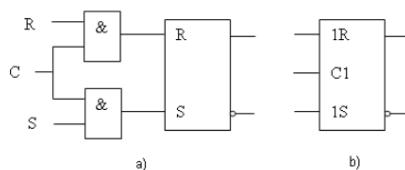
Tabla de verdad biestable RS

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:

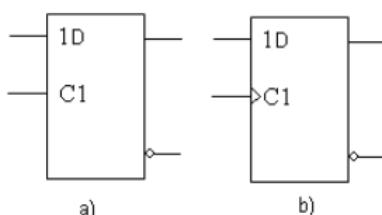
- **Activo por nivel** (alto o bajo), también denominado registro o cerrojo (*latch* en inglés).
- **Activo por flanco** (de subida o de bajada).

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

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

y su [tabla de verdad](#):

D	Q	Q <sub>siguiente</sub>
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

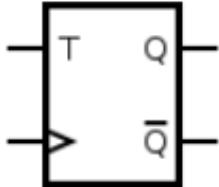
unir las entradas de control de un biestable JK, unión que se corresponde a la entrada T.

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:

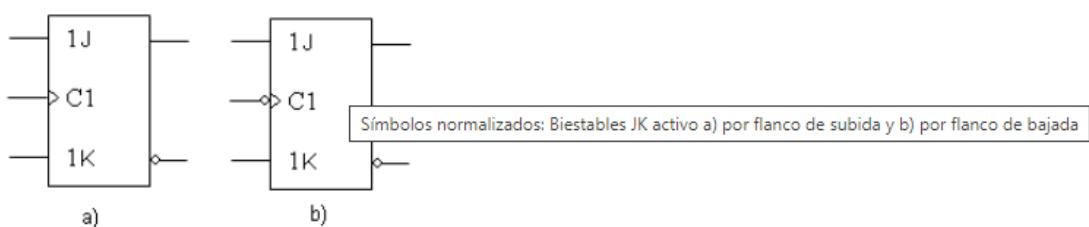
$$Q_{\text{siguiente}} = J\bar{Q} + \bar{K}Q$$

Y su tabla de verdad es:

J	K	Q	<b>Q<sub>siguiente</sub></b>
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 <sub>siguiente</sub>
0	X	0	0
1	X	0	1
X	1	1	0
X	0	1	1
X=no importa			

### LINKS DE WEB:

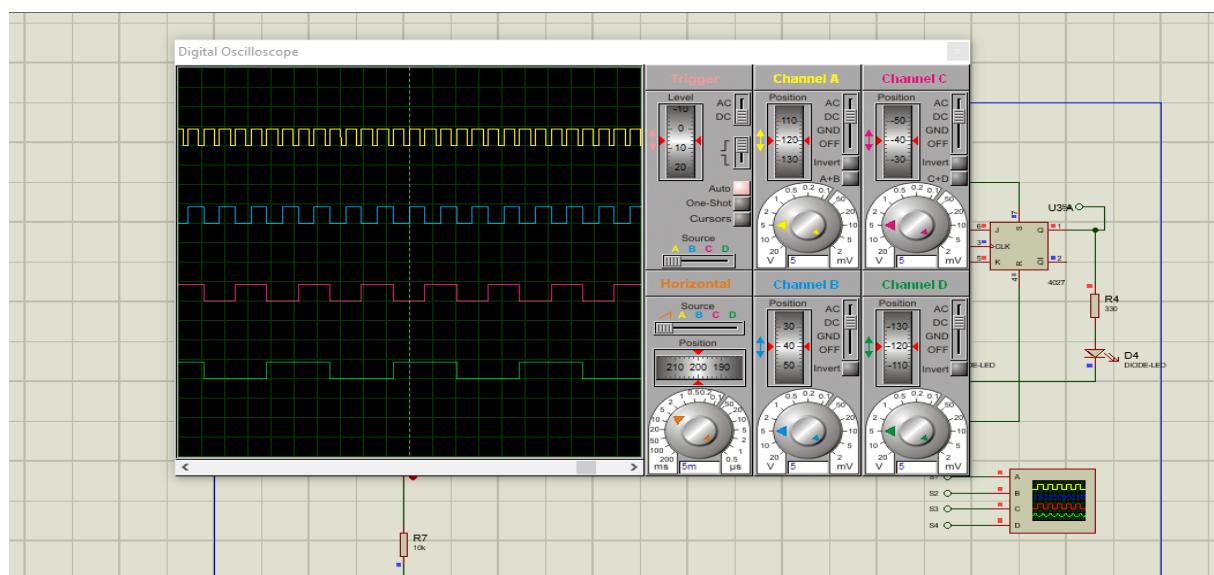
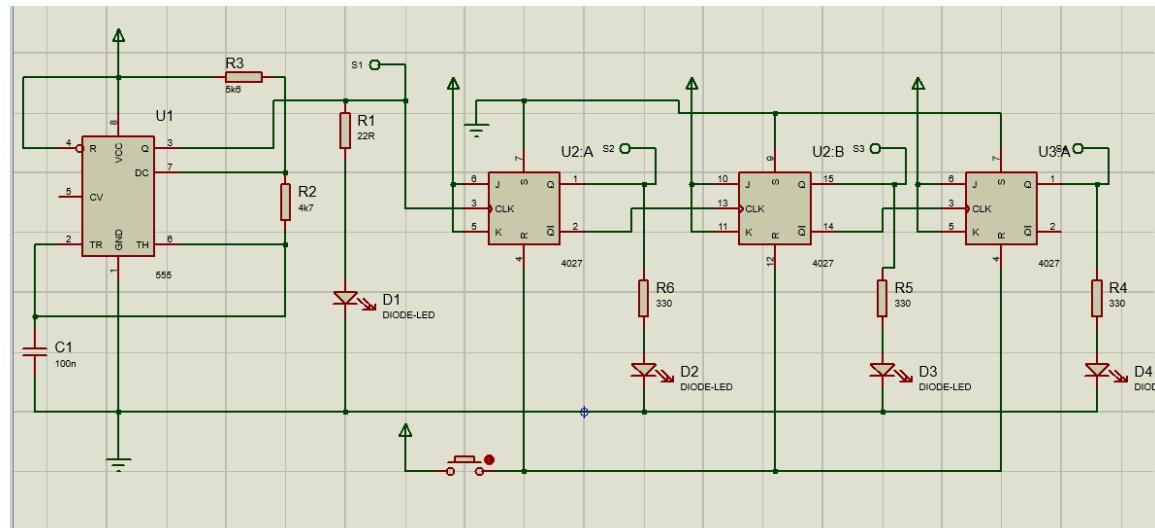
- <https://es.wikipedia.org/wiki/Biestable>
- [https://ocw.ehu.eus/pluginfile.php/42742/mod\\_page/content/1/Tema\\_6/6\\_4.pdf](https://ocw.ehu.eus/pluginfile.php/42742/mod_page/content/1/Tema_6/6_4.pdf)

## 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:

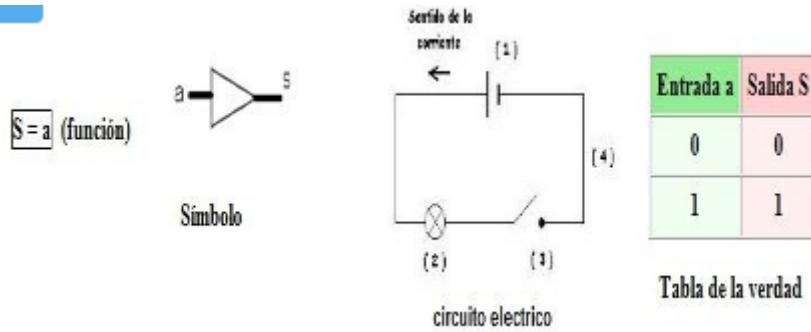
Son componentes electrónicos representados por un símbolo con una o dos entradas y una sola salida que realizan una función y que toman unos valores de salida en función de los que tenga en las entrada.

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

---

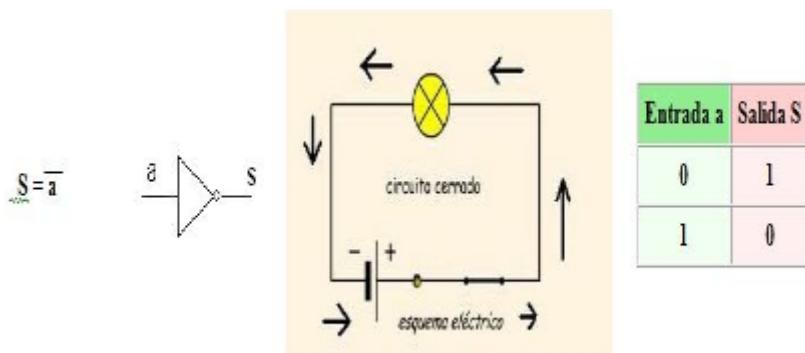


## 9. PUERTAS LÓGICAS

### 9.1 Puerta NO O NOT:

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

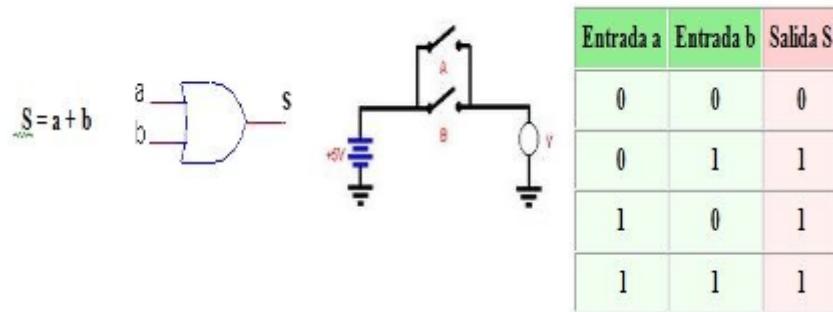
En las funciones, una línea sobre una variable significa que tomará el valor contrario.



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:

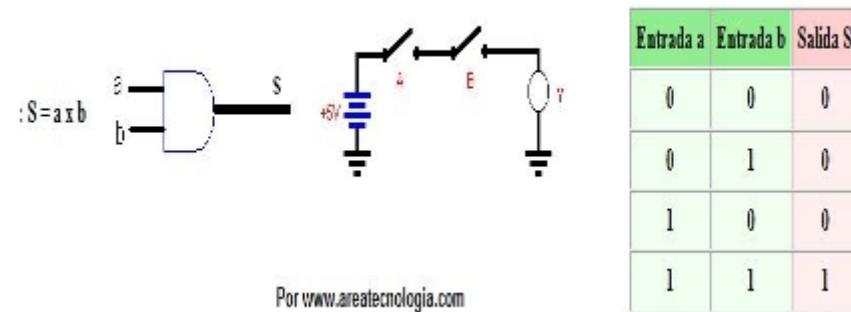
En este caso tenemos dos elementos de entrada, en este caso para que la entrada sea un 1 una de las entradas tendrá que estar pulsada, ya que la suma de las entradas será el valor que tendrá la salida.



## 9.3 Puerta AND:

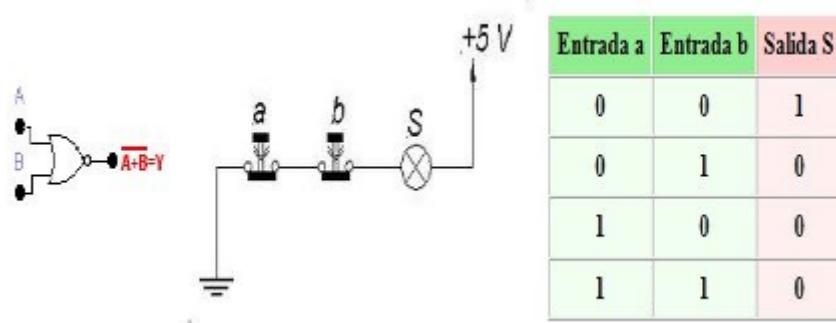
En este caso a caso a diferencia de la anterior debemos tener pulsadas las dos salidas para que la salida esté en 1

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



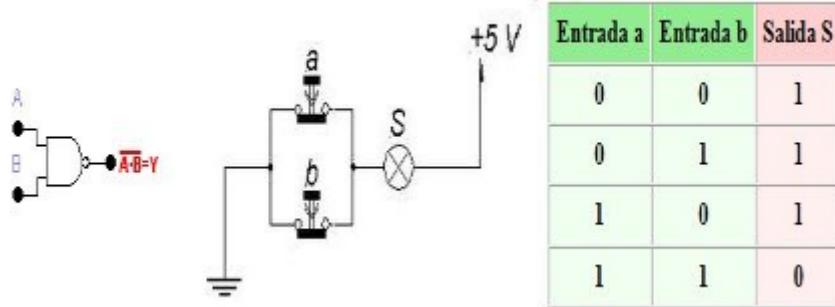
## 9.4 Puerta NOR:

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 entrada estén accionadas nuestra salida estará apagada, en este caso nuestra salida obtendrá el valor inverso de la multiplicación de las entradas.



## 10. TABLA DE LA VERDAD:

A partir de que nos planteen un problema lo primero que deberemos saber es el número de variables de entrada (sensores, pulsadores, interruptores, etc) que vamos a utilizar y a cada variable le asignamos una letra (a, b, c, etc).

Al elemento de salida le llamamos S.

Ahora debemos sacar la tabla de la verdad poniendo los posibles valores de cada una de las variables de entrada (0 o 1) y el valor que tomará la salida S para cada combinación de valores de las variables de entrada.

Esa tabla es lo que se conoce como "**Tabla de la Verdad**" del problema o circuito

Veamos un ejemplo: queremos que una caja fuerte se abra cuando se pulsen dos pulsadores a la vez.

Tenemos dos pulsadores a y b y una salida que será el motor de la caja fuerte. Este motor funcionará para abrir la caja, es decir estará en estado 1, cuando cuando a y b (pulsadores) estén accionados, es decir en estado 1 (si están normalmente abiertos).

Para el resto de combinaciones de a y b el estado de la salida será 0.

Ya sabemos como debe funcionar.

Ahora sacamos la tabla de la verdad. Una tabla con dos variables de entrada a y b y con una salida.

Tendremos una tabla con 4 casos posibles. Para esta tabla vamos pensando para cada caso como será el valor de la salida.

Entrada a	Entrada b	Salida S
0	0	0
0	1	0
1	0	0
1	1	1

[https://docs.google.com/spreadsheets/d/1qUb6M8ckw74evKGVs1ps2eOUTE11Tm  
MY6F2kK-flcwQ/edit#gid=0](https://docs.google.com/spreadsheets/d/1qUb6M8ckw74evKGVs1ps2eOUTE11TmMY6F2kK-flcwQ/edit#gid=0) Aquí tenemos nuestra tabla de la verdad.

## 10.1 FUNCIÓN LÓGICA:

A continuación sacamos la función lógica que representará el problema. Para sacar la función usamos la tabla de la verdad. Cogemos solo las filas que den como salida el valor 1 (en el ejemplo solo hay una y es la última), y multiplicamos las variables de entrada de cada fila que tengan valor 1 (recuerda solo hay una) poniendo invertidas las que tengan valor 0, y en estado normal las que tengan valor 1. En este caso las dos tienen valor 1 luego no habrá ninguna invertida.

La función lógica sería:

$$S = a \times b$$

Así de sencillo. Si tuviéramos dos fila con salida 1 tendríamos dos productos y estos productos se sumarían para sacar la función (no es el caso).

Esta forma se llama "Suma de Productos" en inglés "minterms".

Hay otra forma que es "productos de sumas", en inglés "maxterms" que sería el caso contrario.

Se tomarían los valores que dan salida 0 en la tabla, se haría la suma de ellos pero invirtiendo las variables que tengan valor 1 y no invertidas las que tengan valor 0 (al revés que antes), y estas sumas se multiplicarán.

En nuestro ejemplo tendríamos 3 combinaciones que dan salida 0:

$S = (a + b) \times (a + b') \times (a' + b)$ . Una ecuación mucho más larga en este caso. Normalmente siempre usaremos el primer caso ``suma de productos'' porque la tabla de la verdad suele tener menos salidas 1 que 0.

Nota: el signo "'' en una variable significa invertida (0) y si no lo lleva es no invertida (1).

A	B	C	S
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Suma de Productos-Minterms

$$S = \bar{A} \cdot \bar{B} \cdot \bar{C} + \bar{A} \cdot \bar{B} \cdot C + \bar{A} \cdot B \cdot \bar{C} + A \cdot \bar{B} \cdot \bar{C} + A \cdot B \cdot C$$

Productos de Sumas - Maxterms

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

Otro ejemplo:

### TABLAS DE LA VERDAD

2 <sup>2</sup> = 4 Combinaciones			
	a	b	s
0	0	0	0
1	0	1	0
2	1	0	0
3	1	1	1

2 <sup>3</sup> = 8 Combinaciones				
	a	b	c	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	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	0

2 <sup>4</sup> = 16 Combinaciones					
	a	b	c	d	s
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	1
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	1

#### ■ PUERTAS LÓGICAS SÍMBOLOS

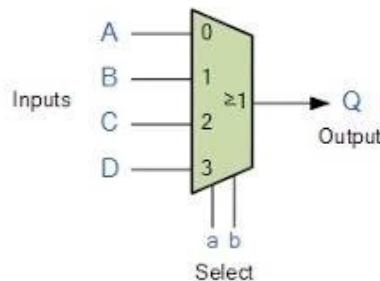
	TRADICIONAL	ANSI (ANSI/IEEE Standard 91-1984)
AND		
OR		
NOT		
NAND		
NOR		
EXCLUSIVE OR		

#### Link de la web de FUNCIÓN LÓGICA:

- <https://www.areatecnologia.com/electronica/electronica-digital.html>

# 11. CIRCUITOS COMBINACIONALES

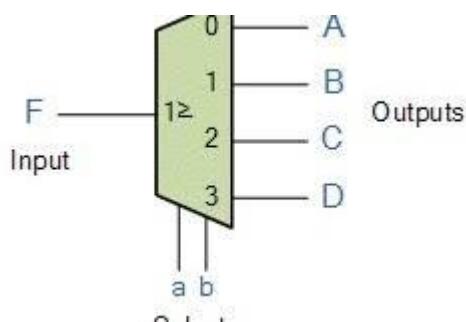
## 11.1 MULTIPLEXORES:



El multiplexor es un elemento que tiene unas entradas, estas entradas son canales y en cada canal podemos poner una orden, con lo cual podemos poner más de un canal. Este caso tiene dos entradas de selección que según el valor que le demos las entradas de selección lo que hace es elegir una entrada y esa entrada que ha elegido la lleva a la salida, es decir, con la tabla de la verdad cada entrada tiene su numero y al poner ese código en la de selección pues seleccionara la que le mandemos y la convertirá en una salida para hacer esa orden.

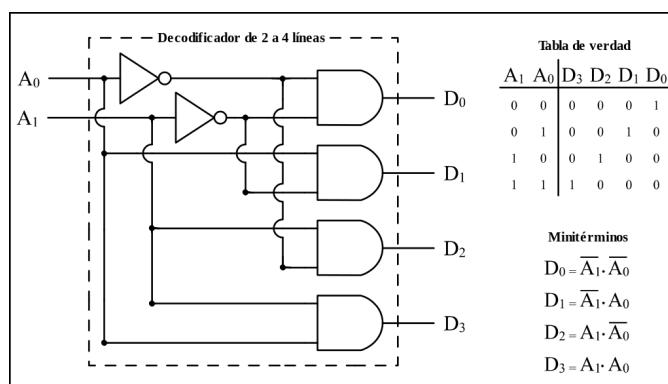
## 11.2 DEMULTIPLEXOR

El demultiplexor es lo mismo que el “multiplexor” pero al revés. Este funciona con una sola entrada junto a las entradas de selección y en este caso tiene más de una salida. Esto nos sirve para tener una orden y dentro de esta orden poder elegir distintas salidas.



## 11.3 DECODIFICADOR

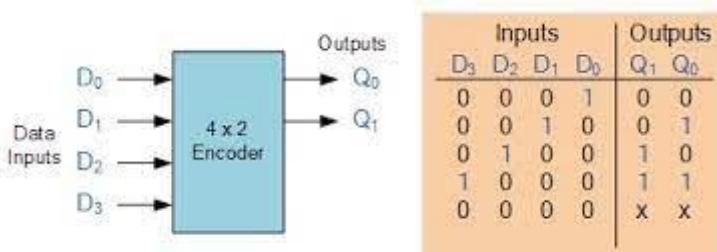
El decodificador es un circuito combinacional, su función es la inversa al codificador, convierte un código binario de  $x$  bits de entrada y  $x$  líneas de salida. Por ejemplo, si tiene 2 bits de entrada, tendrá 4 salidas y de esas 4 salidas dos serán seleccionadas.



## 11.4 CODIFICADOR:

El codificador funciona igual que el decodificador pero al revés, es decir, este elemento analógico tiene unas entradas y según el valor que nos de las entradas nos dará una salida. Esto lo conseguimos con la tabla de la verdad.

Como vemos en la imagen, tenemos cuatro entradas y al tener 2 entradas en la tabla se escribe de esa manera, entonces, depende del valor que tengamos en las entradas tendremos una salida y otra



## 12. ¿Qué es FPGA ?

Una FPGA o matriz de puertas programables (del inglés Field Programmable Gate Array) es un dispositivo programable que contiene bloques de lógica cuya interconexión y funcionalidad puede ser configurada in situ mediante un lenguaje de descripción especializado. La lógica programable puede reproducir desde funciones tan sencillas como las llevadas a cabo por una puerta lógica o un sistema combinacional hasta complejos sistemas en un chip. Internamente están compuestos principalmente de cables, puertas lógicas, biestables, y puertos de entrada y salida. Todo ello sin conectar, como una plantilla en blanco, hasta que se les carga un archivo bitstream (un archivo generado a partir de la descripción del circuito).

La principal diferencia entre las FPGA y los microprocesadores es la complejidad. Aunque ambos varían en complejidad dependiendo de la escala, los microprocesadores tienden a ser más complejos que las FPGA. Esto se debe a los diversos procesos ya implementados en él. Los microprocesadores ya tienen un conjunto fijo de instrucciones, que los programadores deben aprender para crear el programa de trabajo apropiado. Cada una de estas instrucciones tiene su propio bloque correspondiente que ya está cableado en el microprocesador. En cambio, una FPGA no tiene ningún bloque lógico cableado porque eso vencería el aspecto programable del campo del mismo.

Una FPGA se presenta como una red con cada unión que contiene un interruptor que el usuario puede hacer o romper. Esto determina cómo se determina la lógica de cada bloque.

## 12.1 Ze abantaila ditu?

- Tiempo rapido de cálculo.
- Las FPGAs se pueden conectar directamente a las entradas y pueden ofrecer un ancho de banda muy alto.
- La funcionalidad de la FPGA puede cambiar con cada encendido del dispositivo. Así que, cuando un ingeniero de diseño quiere hacer un cambio, puede simplemente descargar un nuevo archivo de configuración en el dispositivo y probar el cambio.
- A menudo, se pueden hacer cambios en la FPGA sin necesidad de hacer costosos cambios en la placa de PC.
- Debido a la flexibilidad de las FPGAs, los fabricantes de equipos originales pueden enviar los sistemas tan pronto como el diseño funciona y se prueba.
- Las FPGAs proporcionan funciones de descarga y aceleración a las CPU, acelerando eficazmente el rendimiento de todo el sistema.
- Los FPGAs de hoy en día incluyen procesadores integrados, E/S de transceptores a 28 Gbps (o más rápido), bloques de RAM, motores DSP y más. Más funciones dentro de la FPGA significan menos dispositivos en la placa de circuito, aumentando la fiabilidad al reducir el número de fallos de los dispositivos.
- Aplicaciones de las FPGAs. HAY MUCHAS APLICACIONES PARA PROGRAMAR LA FPGAs.
- Las FPGAs permiten mayores grados de flexibilidad
- ...

## 13. ¿Qué es Alhambra II-a? ¿Qué características tiene?

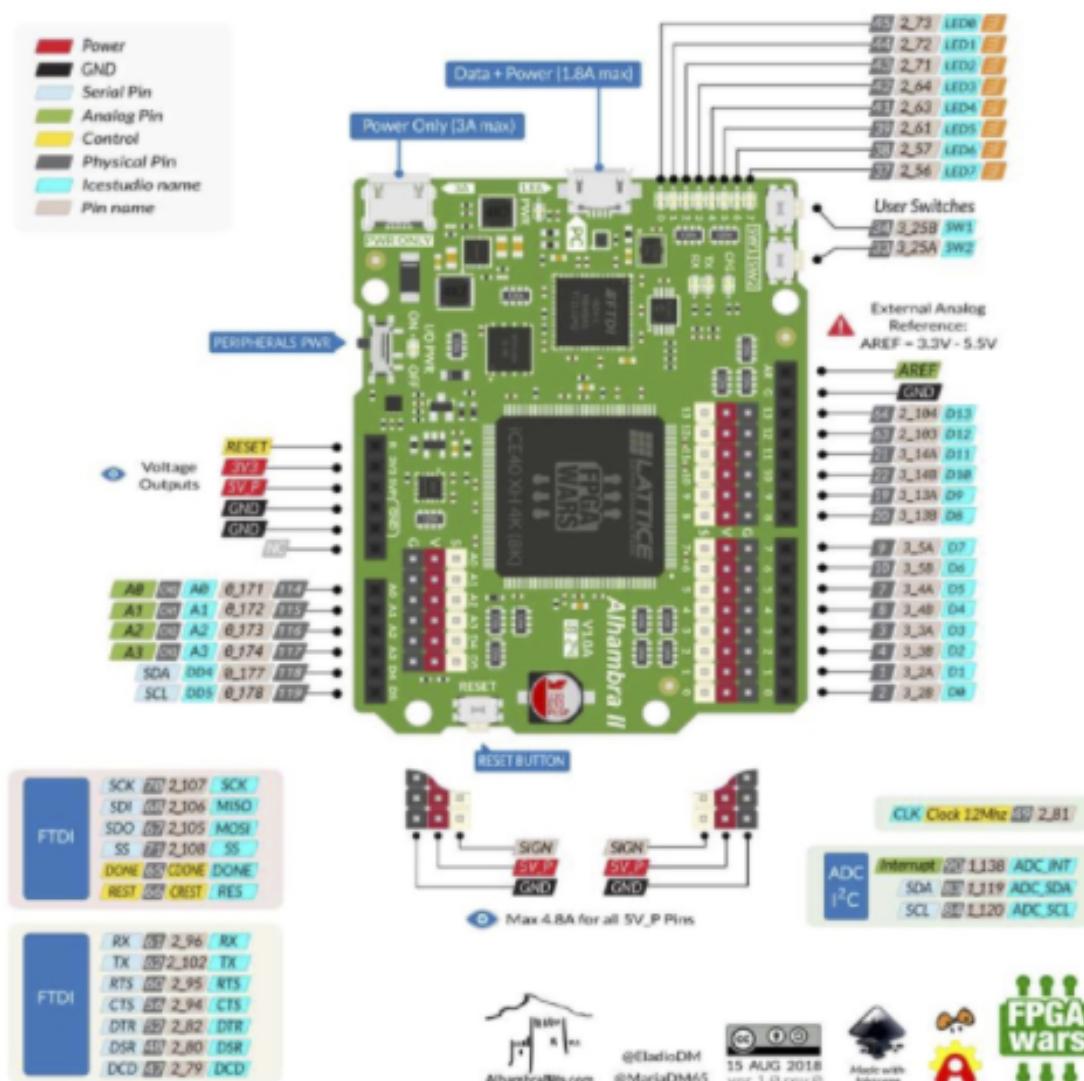
Placa Alhambra II, del mismo tamaño que Arduino, con las FPGAs libres, del fabricante Lattice Semiconductor, cuyo diseño y método de programación fue liberado por Clifford Wolf.

Orientada a makers, sintetiza hardware fácilmente, explorando el lado abierto de FPGAs. Alhambra II es una PCB libre, accesible a todo el mundo, diseño de circuitos digitales mediante herramienta de código abierto.

### 13.1 Características de la placa Alhambra II :

- Placa de desarrollo FPGA iCE40HX4K (Lattice) (8K con cadena de herramientas de código abierto)
- Hardware abierto
- De código abierto: Compatible con el código abierto cadena de herramientas Icestorm e Icestudio
- Similar al de Arduino
- Puedes reutilizar la mayoría de los escudos disponibles
- Controla tus robots / printbots desde un FPGA
- El dispositivo USB FT2232H permite la programación FPGA y la interfaz UART a una PC
- Interruptor electrónico de ENCENDIDO / APAGADO de la placa(apague su robot móvil fácilmente, encenderlo no sobrecarga su puerto USB)
- 8 LED de uso general (LED de usuario)
- 2 botones de propósito general
- Memoria flash de 32Mb para hasta 30 flujos de bits o datos de usuario
- 20 pines de entrada / salida de 3.3V (5V tolerante)
- Todos los pines de E / S incluyen resistencias en serie de 200 ohmios para la activación directa de LED
- Convertidor A / D de 4 canales y 12 bits
- Pines de arranque en frío accesibles desde GPIO
- Reguladores de commutación para procesamiento de alta frecuencia a baja potencia de entrada
- Oscilador MEMS de 12 MHZ
- Botón de reinicio
- Fuente de alimentación USB, dos conectores (hasta 4.8A)

- Los pines de alimentación y los pines de E / S permiten cortocircuito permanente



## 13.2 Zer da PWM-a?

El PWM es un tipo de señal de tensión que usamos en electrónica con muchos objetivos distintos y para muchas tareas distintas.

PWM son siglas en inglés que significan *Pulse Width Modulation* y que lo podemos traducir a español como Modulación de ancho de pulso.

La modulación de ancho de pulso está formada por una señal de onda cuadrada que no siempre tiene la misma relación entre el tiempo que está en alto y el tiempo que está en bajo.

En la siguiente imagen vemos una señal que varía entre 5 voltios y 0 voltios. A lo largo del tiempo la señal varía entre dos valores de tensión. Durante un tiempo determinado la señal se encuentra en el nivel alto ( en este caso 5v ) y durante otro periodo de tiempo se encuentra en el segundo valor de tensión (en este caso 0v).

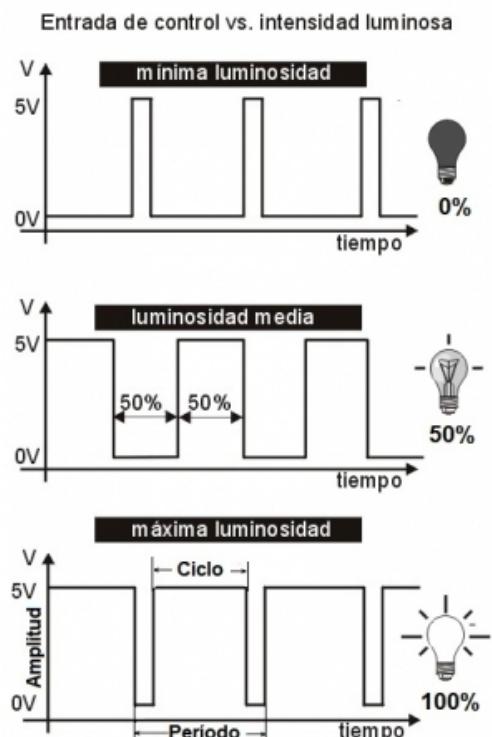
El tiempo que la señal se encuentra en el nivel alto ( 5 voltios ) lo denominamos como tiempo on (  $T_{on}$  ) mientras que el tiempo que está en nivel bajo lo denominamos tiempo off (  $T_{off}$  ). La suma del tiempo on y el tiempo off es el periodo de la señal (  $T$  ).

Y como en toda señal periódica, el inverso de del periodo (  $1 / T$  ) es la frecuencia de la señal.

¿Cómo funciona el PWM? Variando su valor de tensión entre dos valores conocidos, por ejemplo Vcc y GND en períodos concretos de tiempo y con una frecuencia fija. Estos períodos reciben nombres especiales.

## LINK:

- <https://www.rinconingenieril.es/que-es-pwm-y-para-que-sirve/>



## 14. GitHub

### 14.1 ¿QUÉ ES?

GitHub es un servicio basado en la nube que aloja un sistema de control de versiones (VCS) llamado Git. Esto permite a los desarrolladores colaborar y realizar cambios en proyectos compartidos, a la vez que mantienen un seguimiento detallado de su progreso.

## 14.3 ¿Qué es un repositorio de GITHUB?

Un repositorio es un espacio centralizado donde se almacena, organiza, mantiene y difunde información digital, habitualmente archivos informáticos, que pueden contener trabajos científicos, conjuntos de datos o software.

Los datos almacenados en un repositorio pueden distribuirse a través de una red informática, como Internet, o de un medio físico, como un disco compacto. Pueden ser de acceso público o estar protegidos y necesitar de una autentificación previa. Los repositorios más conocidos son los de carácter académico e institucional. Los repositorios suelen contar con sistemas de respaldo y mantenimiento preventivo y correctivo, lo que hace que la información se pueda recuperar en el caso que la máquina quede inutilizable. A esto se lo conoce como preservación digital, y requiere un exhaustivo trabajo de control de calidad e integridad para realizarse correctamente.

Depositar no debe confundirse con publicar. El depósito en los repositorios es una manera de comunicar públicamente los trabajos de los investigadores, aumentando su difusión: los autores ponen disponibles en acceso abierto una versión de los artículos que han publicado en revistas, tradicionales o de acceso abierto. Para ello, los sistemas de repositorios suelen integrarse e interoperar con otros sistemas y aplicaciones web. Asimismo, los repositorios cumplen un rol importante en la formación universitaria.

## 15. ¿QUÉ ES UN WIKI?

El término wiki (palabra que proviene del hawaiano *wiki*, «rápido») alude al nombre que recibe una comunidad virtual, cuyas páginas son editadas directamente desde el navegador, donde los mismos usuarios crean, modifican, corren o eliminan contenidos que, habitualmente, comparten. No tiene por qué ser necesariamente un sitio en la web, puesto que hay wikis instalables para uso en el escritorio de un computador personal o que pueden portarse en un llavero USB que lleven un entorno LAMP como, por ejemplo, XAMPP.

## 16. CONCLUSIONES

A la hora de poner este proyecto en marcha hemos tenido unos cuantos problemas, primero tuvimos que cambiar la batería de nuestro coche al tocarse los cables positivo y negativo dejándolo inservible.

Después hemos tenido que cambiar la forma de pensar y con ello la forma de plantearnos la tabla de la verdad, primero los casos que nunca pasarían colocamos 1 o 0 según lo que nos convenía para simplificar a la hora de sacar las formulas, después al ver que no funcionaba decidimos poner en todos estos casos 0 para que si por casualidad ocurriera uno de los casos no funcionase, y por fin conseguimos que el coche funcionara, pero en la recta cogia demasiada velocidad y se salía del circuito, y por ello decidimos cambiar y en las opciones que tenía que girar le obligamos a rotar, haciendo que así fuera más lento, y aunque mejoró seguía saliendo por lo que por último decidimos que si los detectores en algún momento no detectan nada el coche vaya marcha atrás, de esta manera cuando se salga del circuito el coche irá marcha atrás hasta que consiga detectar y entrar dentro del circuito nuevamente

Otro problema que tuvimos fue con uno de los motores ya que se nos salió uno de los cables y tuvimos que esperar hasta que nos lo cambiaron.

## **BIBLIOGRAFÍA:**

- [FPGA: La placa Alhambra-II](#)
- <https://groups.google.com/g/fpga-wars-explorando-el-lado-libre/c/oCZvixhDu-k>
- [Sensores de Infrarrojos \(IR\)](#)
- <https://www.prometec.net/fuentes-step-down/>
- <https://github.com/Obijuan/digital-electronics-with-open-FPGAs-tutorial/wiki>

**(ICESTUDIO)**