

**Hardware-Table básico del contenido**

<b>Asunto</b>	<b>Paginación</b>
Índice	1
Trabajo con el sistema de número binario	2
El Sistema De Número binario	2
Dígitos binarios y octetos	3
El convertir binario al decimal	3
Decimal que convierte a binario	4
Adición Binaria	4-5
Substracción Binaria	5
Introducción a las puertas de la lógica	6-7
Puertas Básicas De la Lógica	7
La Puerta AND	7
La Puerta OR	8
La Puerta NOT	8
Otras Puertas De la Lógica	9
La Puerta NAND	9
La Puerta NOR	10
La Puerta XOR	10-11
Puertas Interiores	11
La Puerta NOT	12
La Puerta AND	12
La Puerta OR	13
Serpientes Binarias	13
El Half-Adder Binario	13-14
El Full-Adder Binario	15
Serpiente Binaria Paralela	16
Álgebra Boleana	17
Puertas de la lógica y sus expresiones booleanas	18
Fabricación de un vector de verdad con una ecuación	19

## Trabajo con el sistema de número binario

Somos todos familiares con la sistema decimal. Utilizamos diez dígitos en diversas combinaciones para representar números. Estos números, por supuesto, son 0.1.2.3.4.5.6.7.8 y 9. El ordenador, por otra parte, utiliza el sistema binario. Este sistema es mucho más simple que nuestra sistema decimal familiar. En vez de utilizar diez dígitos, utiliza solamente 0 y 1.

Las cuatro operaciones aritméticas básicas (adición, substracción, multiplicación y división) se pueden reducir a dos - adición y substracción. La multiplicación es adición realmente relanzada, mientras que la división es substracción realmente relanzada. Este hecho permite manejar estas operaciones en forma binaria.

### El Sistema De Número binario

Para convertir un número decimal en un número binario, tenemos que tener algunas cosas presente. Por ejemplo, en la sistema decimal, el número 151 realmente significa esto:

Centenares	Diez	Unidades
1	5	1

Así, el número 150 puede también ser representado como  $(1 * 10^2) + (5 * 10^1) + (1 * 10^0)$ . El 1 está en el lugar de los centenares. Los cinco está en el lugar de los diez, y el cero está en el lugar de las unidades. Cada posición en la sistema decimal está parada para un valor multiplicado por una diversa potencia de diez. Esta es la razón por la cual la sistema decimal se refiere a veces como **los diez bajos** sistema. Los números  $10^2$ ,  $10^1$  y  $10^0$  representan los valores de lugar de los dígitos los números por los cuales los dígitos son multiplicados. Aquí está un vector para mostrarle lo que este los medios, usando el número 150 como ejemplo:

	Centenares	Diez	Unidades
Dígito	1	5	1
Significado	$1 * 10^2$	$5 * 10^1$	$1 * 10^0$
Valor	100	50	1

Semejantemente, el sistema binario se puede representar exactamente de la misma manera, excepto, los valores de lugar es las potencias de dos. Así, el sistema binario se llama a veces **los dos bajos** sistema. Puesto que la lata binaria tiene solamente los dígitos 1 y 0, utilizaremos el número 110101 como ejemplo:

Dígito	1	1	0	1	0	1
Significado	$1 * 2^5$	$1 * 2^4$	$0 * 2^3$	$1 * 2^2$	$0 * 2^1$	$1 * 2^0$

Valor	32	16	0	4	0	1
-------	----	----	---	---	---	---

### Dígitos binarios y octetos

Aunque un dígito binario es la forma más pequeña de datos que un ordenador, allí sea nombres para cantidades más grandes de dígitos binarios. Por ejemplo, 8 dígitos binarios se consideran ser 1 octeto. Con esta información en mente, aquí está un vector que las demostraciones usted los diversos datos clasifican los nombres usados por el ordenador, y cuántos dígitos binarios cada uno es:

Nombre	Abbrivation	Número de octetos (como potencia)	Número de octetos
Octeto	-	$2^0$	1
KiloByte	KB	$2^{10}$	1.024
MegaByte	Mb	$2^{20}$	1.048.576
GigaByte	GB	$2^{30}$	1.073.741.824
TeraByte	TB	$2^{40}$	1.099.511.627.776
PetaByte	PB	$2^{50}$	1.125.899.906.842.624
ExaByte	Eb	$2^{60}$	1.152.921.504.606.846.976
ZettaByte	ZB	$2^{70}$	1.180.591.620.717.411.303.424
YottaByte	YB	$2^{80}$	1.208.925.819.614.629.174.706.176

Como usted puede ver, cada paso de progresión va para arriba por  $2^{10}$  Nota, usted necesita solamente ser conocido de octeto hasta GigaByte, porque las otras tallas no se utilizan todavía (los ordenadores no han avanzado bastantes para poder utilizarlos), aunque hay algunos mecanismos impulsores duros que tienen un almacenaje de un TeraByte o más.

### El convertir binario al decimal

Convertir un número binario en un número decimal es simple. Para calcular fuera para de lo que está parado este número, tenemos que agregar para arriba números en cada valor de lugar del número binario. Utilicemos el número binario 110101 como ejemplo:

1 * 32 =	32
1 * 16 =	16
0 * 8 =	0
1 * 4 =	4
0 * 2 =	0
1 * 1 =	1
<b>Total</b>	<b>53</b>

Respuesta - 110101 en binario es 53 en decimal.

### Decimal que convierte a binario

Convertir un número decimal en binario es también muy simple. Implica una división continua del número decimal y de no perder de vista el resto. Utilicemos el número decimal 220 como ejemplo:

División	Resto
$220 / 2 = 110$	0 Dígitos binarios Lo más menos Posible Significativos (Lsb)
$110 / 2 = 55$	0
$55 / 2 = 27$	1
$27 / 2 = 13$	1
$13 / 2 = 6$	1
$6 / 2 = 3$	0
$3 / 2 = 1$	1
$1 / 2 = 0$	1 - La mayoría Del Dígitos binario Significativo (Msb)

Respuesta - 220 en decimal es 11011100 en binario.

El dígito binario más significativo es exactamente lo que deduce el nombre, el más significativo. Esto significa que tiene el valor más grande fuera de todos los dígitos binarios. Puesto que el MSB es el último en el vector, tenemos que poner el dígito binario pasado del vector primero, y trabajamos nuestra manera para arriba. Por lo tanto el número decimal 220 es 11011100 en decimal, no 00111011.

### Adición Binaria

Ahora que sabemos convertir binario al decimal y viceversa, es hora de aprender cómo agregar dos números binarios juntos. La adición binaria se hace la misma manera que la adición decimal, que está utilizando el método del llevar. Aquí está un ejemplo de usar el método del llevar para agregar para arriba dos números decimales:

$$\begin{array}{r}
 1 \\
 3248 \\
 + 1426 \\
 \hline
 4674
 \end{array}$$

Como usted puede ver, cuando 8 y 6 fueron agregados, 4 fueron registrados como el dígito de las unidades para la respuesta, y un 1 fue transportado a la columna de los diez. Aquí está un ejemplo de cómo se agregan los números binarios

Por el ejemplo binario de la adición, los números binarios 1001 y 0101 serán agregados:

$$\begin{array}{r} 1 \\ 1001 \\ + 0101 \\ \hline 1110 \end{array}$$

Primero agregamos los dos dígitos binarios lo más menos posible significativos de los dos números que se agregarán, de 1 y de 1. Recuerde que la lata binaria solamente tiene 0 y 1  $1 + 1$  sería registrado como 0 en los  $2^0$  columnas, y un 1 tendría que ser llevado a los 2 siguientes  $1$  columna. Entonces agregamos  $0 + 0 +$  el llevar de 1, y conseguimos a una respuesta de 1 en los  $2^1$  columna. Agregamos después el 0 y el 1, y conseguimos una respuesta de 1 en los  $2^2$  Finalmente, agregamos para arriba el 1 y el 0 y conseguimos una respuesta de 1 en los  $2^3$  La respuesta final entonces se revela para ser 1110. Como usted puede ver, la adición binaria es muy fácil para la gente que sabe agregar números decimales más grandes juntos.

### Substracción Binaria

Hay algunas maneras de restar números binarios, pero evitar cualquier confusión, utilizaremos el método del prestado ya familiar a usted. Vamos recordar la substracción decimal con un ejemplo:

$$\begin{array}{r} 21 \\ 3248 \\ - 1426 \\ \hline 1822 \end{array}$$

Cuando restamos los números en la columna de las unidades, tan bien como la columna de los diez, no hay problema. Cuando conseguimos a la columna de los 100, aviso cómo el número en la tapa es más pequeño que el número en el fondo. Para continuar con la substracción, pedimos prestados de la columna 1000's, y la colocamos en la columna de los 100. Ahora, conseguimos  $12 - 4$  que es 8. Finalmente, los 3 se cambia a los dos en la columna 1000's debido a el pedir, y  $2 - 1 = 1$ , dejándonos con la respuesta final de 1822.

Como ejemplo de la substracción binaria, restaremos 0101 a partir del 1011:

$$\begin{array}{r} 011 \\ 1001 \\ - 0101 \\ \hline 0100 \end{array}$$

En los  $2^0$  y  $2^1$  columna, sustracción ocurre la misma manera que en la sustracción decimal. Cuando conseguimos a los  $2^3$  columna, tenemos que pedir prestados. El 0 en la columna  $2^3$  se cambia a dos 1 porque usted necesita pedir prestada la columna  $2^4$  de la forma 2, que vale dos veces tanto como la columna  $2^3$ . Porque usted pidió prestado un 1 de la columna  $2^4$  usted la cambia a un 0. El resto de la sustracción es simple, y hecho la misma manera que sería hecha en la sustracción decimal normal.

### Introducción a las puertas de la lógica

En el lenguaje inglés, el término "puerta" es "una estructura que se puede hacer pivotar, trazar, o bajar para bloquear una entrada o un callejón". Un ejemplo es una puerta en una cerca, que se puede abrir dejó a gente adentro, y cerrado para guardar a la gente hacia fuera. En ordenadores, las puertas de la lógica tienen una aplicación similar. Una puerta de la lógica tiene unas o más entradas de información (1 o 0), y una hecha salir. Las puertas de la lógica se diseñan para permitir el paso de señales binarias a través de la puerta para ciertas combinaciones de entradas de información. Estas puertas pudieron parecerse simples, primitivas, e inútiles. Aunque una puerta de la lógica solamente es casi inútil, una combinación de las puertas que trabajan junta puede responder a muchos propósitos. Los ordenadores de hoy, que hacen el trabajo muy complejo, se componen de millones de estas puertas simples de la lógica. Hay tres puertas básicas de la lógica. Son las AND, OR y NOT puertas. Estas puertas se utilizan en diversas combinaciones para hacer puertas más complejas tales como el XOR. Pues usted descubrirá en un momento, las tres puertas básicas de la lógica son muy fáciles de entender. Pero antes de que aprendamos sobre las puertas de la lógica, hay algunos términos que debemos saber alrededor:

1. **Vector De Verdad:** Un vector de verdad es un vector de combinaciones. Muestra todas las posibilidades de la entrada de información y las salidas qué resultado de estas posibilidades. Por ejemplo, aquí está una demostración del vector de verdad cuál será los productos cuando el número 2 es multiplicado por los números a partir de la 1-5.

Números	Producto
2 * 1	2
2 * 2	4
2 * 3	6
2 * 4	8
2 * 5	10

Esto se llama un vector de verdad porque es un vector que tiene hechos sobre el producto de dos y numera a partir de la 1-5. Un vector de verdad para las puertas de la lógica será utilizado para mostrar le todas las entradas de información posibles para las puertas, y a las salidas que resultan para cada combinación de entradas de

información.

**2. Símbolos De la Puerta:** Los símbolos de la puerta se utilizan en los diagramas para planear antes de que se construya el trazado de circuito del ordenador. Las puertas realmente no parecen los símbolos, los símbolos se utilizan para hacer los diagramas más simples y más comprensivos.

**3. Más cosas a saber sobre binario:** De las lecciones anteriores, sabemos que ese 1 y 0 es las dos posibilidades en binario. En el mundo de ordenador, usted oirá con frecuencia el " COLMO " usado en vez de 1, y el " PUNTO BAJO " que es utilizado en vez de 0.

**4. Entradas de información y salidas:** Las entradas de información y las salidas para las puertas de la lógica son representadas por las cartas. Comúnmente, la salida es representada por la letra Q o la carta Y. We utilizará la letra Y para la salida. Las entradas de información son representadas casi siempre por las primeras cartas del alfabeto. A y B son usado lo más a menudo posible, mientras que se utilizan C, D y otras cartas cuando los diagramas son más complejos.

### Puertas Básicas De la Lógica

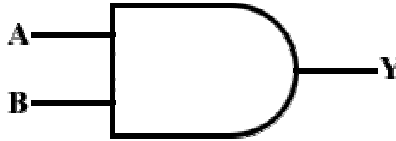
Esto le ayudará a entender cómo la función de tres puertas básica. Usted aprenderá que sus nombres se asemejan al tipo de operación que fueron hechos para hacer. Con esto dicha, hay tres puertas básicas. Son AND la puerta, la puerta OR y la puerta NOT.

#### La Puerta AND

La puerta AND da una ALTA salida solamente cuando todas las entradas de información son ALTAS. Si de las entradas de información es BAJA, la salida es también BAJA. Esta información sobre la puerta se puede representar en un vector de verdad. Aquí está el vector de verdad de AND puertas:

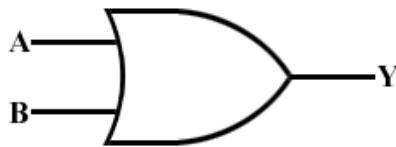
ENTRADA DE INFORMACIÓN		SALIDA
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

La puerta AND puede tener más de dos entradas de información. Un COLMO binario será producido solamente cuando todas las entradas de información son ALTAS. Así como el resto de las puertas, esta puerta tiene un símbolo usado para representarlo en un diagrama:



### La Puerta OR

La puerta OR es una puerta muy simple. Entrega una ALTA salida cuando cualesquiera de las entradas de información son ALTAS. La única manera que una puerta OR produciría una salida BAJA es si todas las entradas de información eran BAJAS. Con esto dicha, aquí está el vector de verdad y el símbolo para la puerta 2-input OR:

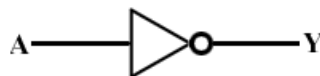


ENTRADA DE INFORMACIÓN		SALIDA
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Justo como la puerta AND, la puerta OR puede tener más de dos entradas de información. La puerta OR la funcionaría es lógica la misma manera, sólo las entradas de información una tendrían que ser ALTAS en la orden para que la salida sea ALTAS.

### La Puerta NOT

El más simple de las tres puertas básicas es la puerta NOT. Comúnmente se llama un inversor. El propósito de la puerta NOT es tomar la entrada de información, la cambia a su binario enfrente de, e hizo salir la entrada de información invertida. Desemejante de los AND o de las OR puertas, la puerta NOT puede solamente tener uno entrado. Aquí está el vector y el símbolo de verdad para la puerta NOT:



ENTRADA DE INFORMACIÓN	SALIDA
A	B
0	1
1	0

### Otras Puertas De la Lógica

Todas las otras puertas se crean usando una combinación de dos o más de las AND, OR y/o NOT puertas.

#### La Puerta NAND

La puerta NAND es justa como la puerta básica AND. La única diferencia es que la puerta NAND invierte la salida final. Aquí está el vector de verdad para la PUERTA NAND:

ENTRADA DE INFORMACIÓN		SALIDA
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Aunque esta puerta se parece bastante simple, es más compleja que cualesquiera de las puertas básicas. Como usted puede ver por el vector de verdad, las salidas son el contrario exacto de la puerta AND. Esto es alcanzada usando una combinación de dos puertas básicas, los AND y los NOT. Puesto que utiliza una combinación de dos puertas básicas, no se considera una puerta básica. Aquí es cómo los componentes de la puerta NAND se arreglan:



Como usted puede ver, las entradas de información entran una puerta AND. La salida de la puerta AND entonces es invertida por la puerta OR para alcanzar el

resultado de la puerta entera NAND. La puerta NAND es realmente una puerta AND con una salida invertida.

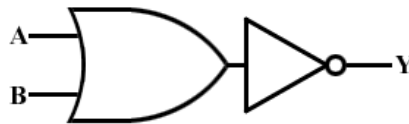
En vez de tener que utilizar la combinación de las dos puertas en el diagrama antedicho como símbolo para la puerta NAND, se utiliza un símbolo simplificado:



Observe que la puerta NOT se ha substituido por un círculo pequeño. Este círculo pequeño se utiliza comúnmente para substituir una puerta entera NAND al diseñar los circuitos de lógica.

### La Puerta NOR

La puerta NOR es el contrario exacto de la puerta OR. La salida es ALTA solamente cuando todas las entradas de información son BAJAS. Cuando cualesquiera de las entradas de información son ALTAS, la salida es siempre ALTA. La puerta NOR también se construye usando dos puertas básicas. Esta vez la puerta OR y NOT se emplea para crear la puerta NOR. Aquí está un diagrama de la puerta NOR junto con el vector de verdad:



ENTRADA DE INFORMACIÓN		SALIDA
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

Como usted puede ver, la puerta NOR es igual que la puerta NAND a menos que utilice una puerta OR en lugar de la puerta AND. Para hacer diagramas más simples, la misma cosa se hace con la puerta NOR como fue hecho con la puerta NAND:

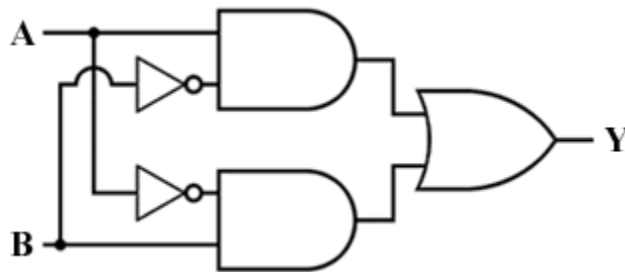


### La Puerta XOR

OR la puerta EXCLUSIVA (XOR se utiliza muy comúnmente en circuitos aritméticos. La puerta XOR compara las entradas de información. Si ambas entradas de información (puede solamente haber dos entradas de información para esta puerta) son iguales, la salida es BAJA. En contraste, si diferencian las entradas de información, entonces la salida es ALTA. Aquí está el vector de verdad para la puerta XOR:

ENTRADA DE INFORMACIÓN		SALIDA
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

La puerta XOR es la más compleja de todas las puertas; requiere dos AND puertas, dos NOT puertas, tan bien como una puerta OR. Aquí está un diagrama que muestra cómo las cinco puertas se arreglan para crear la puerta XOR:



Las entradas de información de A y de B son ambas fractura en dos para permitir que sean las entradas de información para dos puertas en el mismo tiempo. Se invierten las dos entradas de información que acaban de ser creadas de la fractura. Hay dos AND puertas. Las dos entradas de información originales van a las diversas AND puertas. La entrada de información invertida de A se convierte en la segunda entrada de información de la puerta AND en la cual B es la otra entrada de información. Asimismo, la entrada de información invertida de B se convierte en la segunda entrada de información de la puerta AND en la cual A es la otra entrada de información. Las salidas de las AND puertas se convierten en las entradas de información de la puerta OR. Si cualquiera de las entradas de información es ALTA, entonces la salida de la puerta entera XOR es ALTA. Si no, la salida de la puerta XOR es BAJA.

Justo como las dos otras puertas en esta sección, la puerta XOR se simplifica en diagramas. Aquí está el símbolo usado para representar la puerta XOR:

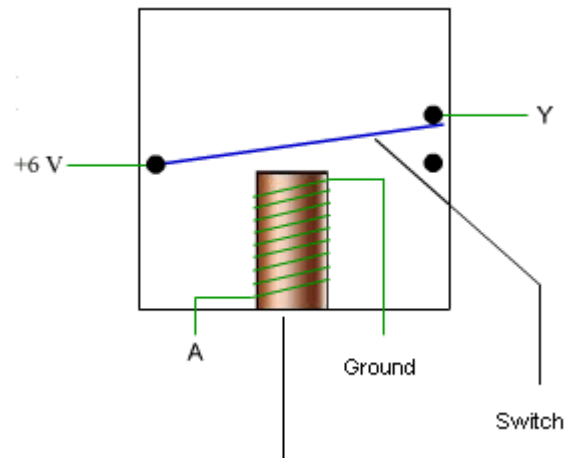


### Puertas Interiores

Esta sección nos tomará el interior las puertas básicas como los AND, los OR, y los NOT para considerar cómo trabajan mecánicamente.

### La Puerta NOT

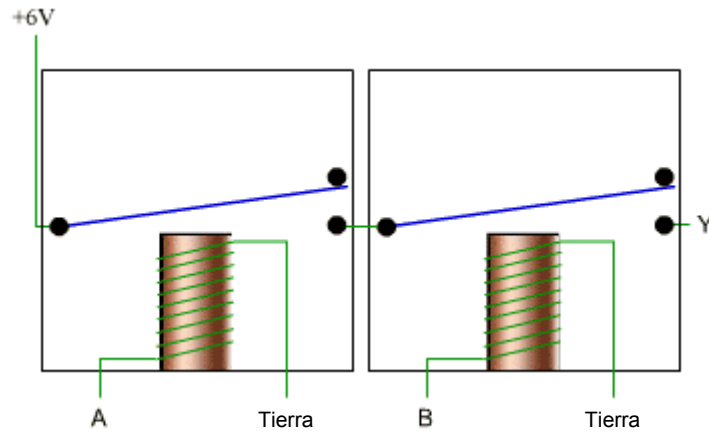
Antes de que nos entendamos cómo las AND y OR puertas funcionan debemos primero entender cómo los NOT trabajos más básicos de la puerta.



Electroimán - Un metal conductor con una bobina envuelta alrededor de ella. Cuando la potencia pasa a través de la bobina, el metal magnetiza y atrae el interruptor.

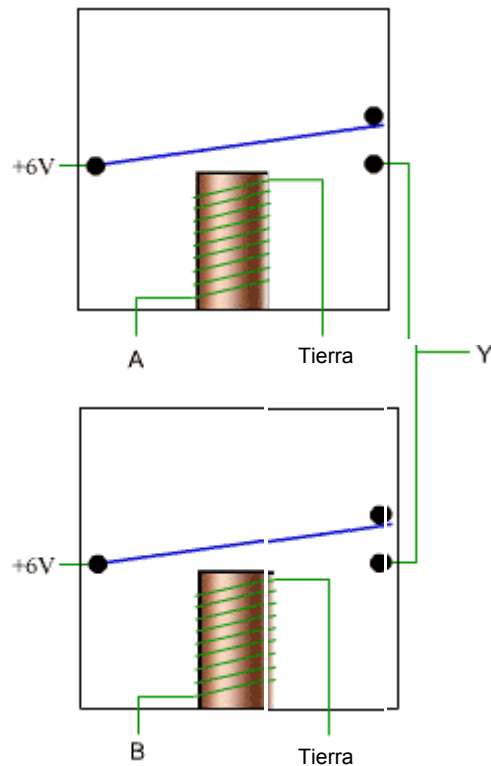
Cuando A está en 0 voltios, el interruptor deja la corriente a través, y la salida en Y es 1. Cuando A está en + 6 voltios, que da una entrada de información de 1, el comienzo del electroimán, tirando del interruptor y rompiendo el flujo del actual, dando una salida de 0.

### La Puerta AND



Cuando la entrada de información de A y de B es BAJA, ni unos ni otros de los electroimanes se accionan, y la salida (y) es BAJA. Cuando B es ALTO, su interruptor se fija para permitir la corriente a través. Pero la corriente debe pasar a través del primer interruptor primero antes de que pase con el segundo. Puesto que no puede Y permanece BAJA. Cuando A es ALTA, su interruptor se fija para permitir la corriente a través. Pero la corriente debe pasar a través del segundo interruptor después de que pase con el primer. Puesto que no puede Y permanece BAJA. Cuando A y B son ALTOS, ambos interruptores dejan la corriente a través, dejando la salida en Y como ALTA.

### La Puerta OR



Cuando ambas entradas de información (A y B) son BAJOS, los electroimanes permanecen apagado y el interruptor no permite que la corriente fluya a la salida, dando a Y un 0 binario hecho salir. Si una de las entradas de información es BAJA y

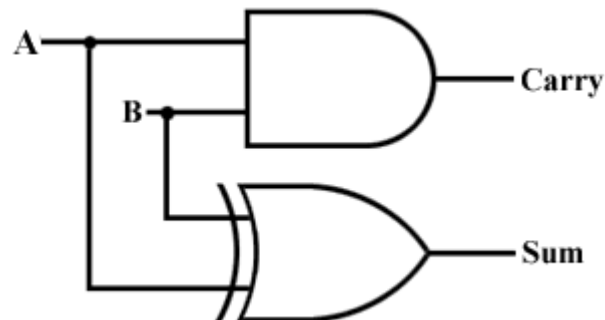
uno es alto, un electroimán continúa, haciendo un interruptor abrirse. Uno es todo que es necesario para que Y tenga un 1 binario hecho salir. Finalmente, cuando ambas entradas de información son ALTAS, Y sigue siendo ALTA porque hay dos fuentes de la potencia y por lo menos una se requiere para una ALTA salida.

### Serpientes Binarias

Ahora, usted ha aprendido cómo agregar números binarios. Usted también ha aprendido cómo las 6 puertas comunes de la lógica funcionan. Ahora es hora de aprender cómo las puertas de la lógica se utilizan para agregar números binarios.

### El Half-Adder Binario

Usted puede ser sorprendido ver cómo es fácil debe crear un circuito de lógica que agregue números binarios. El half-adder binario es perfecto para agregar los números binarios 1-bit. Se hace fuera de una AND puertas y de una puerta XOR. Aquí es cómo se arregla en un diagrama:



A y B son las dos entradas de información. En este caso son los dos dígitos binarios que deben ser agregados. La puerta XOR se utiliza para calcular fuera de la suma, mientras que la puerta AND se utiliza para calcular fuera del llevar. Comencemos con la puerta XOR. Se entran A y B, y si diferencian los dos dígitos binarios, la salida de la puerta XOR es 1, que significa que la suma es 1. Si no, la salida de la puerta XOR es 0. si usted recuerda la adición de dígitos binarios,  $0 + 0 = 0$ ,  $0 + 1 = 1$ , mientras que  $1 + 1 = 0$  con un llevar de 1. Aquí es adonde viene la puerta AND adentro. Si A y B son 1, entonces la salida para la puerta AND, que representa el llevar, es 1. Si no el llevar es 0. Esto se puede resumir en un vector de verdad para el half-adder:

Entrada de información		Salida	
A	B	Suma	Lleve
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

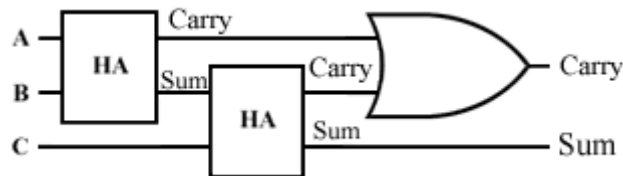
El half-Adder tiene su propio símbolo especial en diagramas para hacerlos más simples. Esto es lo que parece



Ahora que sabemos el ordenador agrega dos dígitos al mismo tiempo, es hora de aprender cómo agrega dos dígitos binarios y un llevar en el mismo tiempo.

### El Full-Adder Binario

La lleno-serpiente binaria es muy útil porque puede agregar tres dígitos al mismo tiempo. Si usted recuerda de nuestra sección binaria de la adición, hay lleva a veces esa necesidad de ser agregado junto con los dos otros dígitos. El requiere la adición de tres dígitos binarios al mismo tiempo, que es exactamente para lo que se utiliza la lleno-serpiente. Aquí está el diagrama de la lógica para la lleno-serpiente más compleja, que emplea dos HÁs (half-adders) así como una puerta OR:



Como usted puede ver, ahora hay tres entradas de información: Utilizan A, B y a C. A y B generalmente para los dos dígitos binarios que serían agregados normalmente, mientras que C se utiliza generalmente como llevar de un valor de lugar más bajo. Como usted puede ver, entra A y B entra un half-adder. La suma del half-adder se envía como una de las entradas de información (junto con la entrada de información C) al segundo half-adder. La suma del segundo half-adder es la suma de las tres entradas de información. Si había un llevar de cualquiera o de ambos half-adders, el llevar hecho salir para la serpiente llena es ALTO. Hay 8 combinaciones posibles de las entradas de información para la lleno-serpiente, registradas todo en el vector de verdad de la lleno-serpiente:

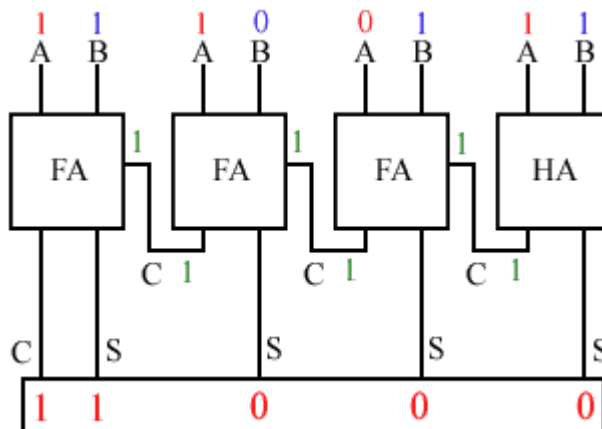
Entrada de información			Salida	
A	B	C	Suma	Lleve

0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Para simplificar este diagrama, un símbolo se utiliza para la lleno-serpiente. El símbolo es idéntico al símbolo del half-adder, a menos que lleve las letras FA en vez de la ha en él.

### Serpiente Binaria Paralela

El uso de un half-adder o de una lleno-serpiente solamente es grande para agregar para arriba dos números binarios con una longitud de un dígito binario cada uno, pero qué sucede cuando el ordenador necesita agregar para arriba dos números binarios con una longitud más larga? Bien, hay varias maneras de hacer esto. La manera más rápida en gran medida es utilizar la serpiente binaria paralela. La serpiente binaria paralela utiliza un half-adder, junto con unas o más serpientes llenas. El número de las serpientes totales necesitadas depende de la longitud del más grande de los dos números binarios que deben ser agregados. Por ejemplo, si agregáramos para arriba los números binarios 1011 y 1, necesitaríamos cuatro serpientes en total, porque la longitud del número más grande es cuatro. Teniendo esto presente, aquí es una demostración de cómo una serpiente binaria paralela four-bit funciona, con 1101 y 1011 como los dos números para agregar:



Apenas como cuando agregamos sin el ordenador, en la serpiente binaria paralela, el ordenador agrega de derecho a la izquierda. Aquí está una lista paso a paso, mostrándole qué sucede en la serpiente binaria paralela:

1. En el único half-adder, las entradas de información de 1 y 1 nos dan 0 con un llevar de 1.
2. En la primera lleno-serpiente (que va de derecho a la izquierda), las entradas de información de 1 y 0 más el llevar de 1 del half-adder nos dan un 0 con un llevar de 1.
3. En la segunda serpiente llena, las entradas de información de 0 y 1 más el llevar de 1 de la lleno-serpiente anterior nos dan un 0 con un llevar de 1.
4. En la tercera y final serpiente llena, las entradas de información de 1 y 1 más el llevar de 1 de la lleno-serpiente anterior nos dan un 1 con un llevar de 1.
5. Puesto que no hay números a agregar para arriba, y todavía hay un llevar de 1, el llevar se convierte en el dígito binario más significativo
6. La suma de 1101 y 1011 es 11000.

## Álgebra Boleana

Pues la mayoría de la gente está bien enterada, las expresiones matemáticas se utilizan para hacer los cálculos y otros problemas de matemáticas más simples y más pequeños. La álgebra booleana se utiliza para hacer la misma cosa, a menos que la haga con los circuitos de lógica en vez de otros problemas matemáticos.

La primera cosa que necesita ser sabida para la álgebra booleana es el significado de las varias muestras. Hay tres de éstos, todo derivando de las tres puertas básicas. Son AND, OR y NOT.



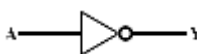
La operación AND también se conoce como conjunción. Da el producto de dos dígitos binarios binarios. Usando A y B como entradas de información, sería escrito como el AB o a veces  $A \cdot B$ . Esto por supuesto significa que A es multiplicada por B, que es exactamente cómo funciones AND de una puerta.





La operación OR también se conoce como separación. Esta operación da la suma de dos dígitos binarios binarios. Una vez más usando A y B como entradas de información, sería escrito como  $A + B$ , que es cómo funciones OR de una puerta. NOTE, " + " significa necesariamente la misma cosa que hace en matemáticas normales que nos todos utilizan a. En álgebra booleana, está parado para OR. Por ejemplo,  $1 + 1$  igualaría a 1, no 10, el equivalente binario de dos. Si cualquiera de las entradas de información es 1, la salida es 1. La operación NOT también se conoce como negación. Esta operación da el contrario de un solo término. Por ejemplo, la negación de A se escribe como  $\bar{A}$  Mientras que sería la negación del AB  $\overline{AB}$  Podemos relacionar las AND, OR y NOT operaciones con las puertas correspondientes:

Vector De Verdad AND	Vector De Verdad OR	Vector De Verdad NOT
$A \cdot B = Y$	$A + B = Y$	$\bar{A} = Y$
$0 \cdot 0 = 0$	$0 + 0 = 0$	0 negado = 1
$0 \cdot 1 = 0$	$0 + 1 = 1$	
$1 \cdot 0 = 0$	$1 + 0 = 1$	1 negado = 0
$1 \cdot 1 = 1$	$1 + 1 = 1$	

### Puertas de la lógica y sus expresiones booleanas

Todas las puertas de la lógica tienen una expresión booleana. Mostrarle cómo la álgebra booleana trabaja más lejos, aquí sea algunos ejemplos de puertas y de sus expresiones booleanas:

Nombre De la Puerta De la Lógica	Símbolo	Expresión Booleana
AND		$Y = A \cdot B$
OR		$Y = A + B$
NOT		$Y = \bar{A}$

NAND		$Y = \overline{AB}$
NOR		$Y = \overline{A+B}$
AND con una de dos entradas de información invertidas		$Y = \overline{A} + B$
OR con una de dos entradas de información invertidas		$Y = \overline{A} \cdot B$

### Fabricación de un vector de verdad con una ecuación

Cuando está dada una ecuación, es fácil crear un vector de verdad para la ecuación. Aquí está un ejemplo de cómo se hace, con la expresión  $Y = \overline{A} \cdot B$ . Enchufamos todos los valores posibles para A y B, y registramos los valores de Y:

1. <b>A = 0, B = 0</b> Y = $\overline{A} \cdot B$ Y = 1 · 0 <b>Y = 0</b>	2. <b>A = 0, B = 1</b> Y = $\overline{A} \cdot B$ Y = 1 · 1 <b>Y = 1</b>
3. <b>A = 1, B = 0</b> Y = $\overline{A} \cdot B$ Y = 0 · 0 <b>Y = 0</b>	4. <b>A = 1, B = 1</b> Y = $\overline{A} \cdot B$ Y = 0 · 1 <b>Y = 0</b>

Una vez que se haga esto, lo ordenamos en un vector de verdad:

Entrada de información	Salida
------------------------	--------

<b>A</b>	<b>B</b>	<b>Y</b>
0	0	0
0	1	1
1	0	0
1	1	0

Esperamos que usted ahora tenga una mayor comprensión del sistema de número binario y de las puertas booleanas de la lógica.