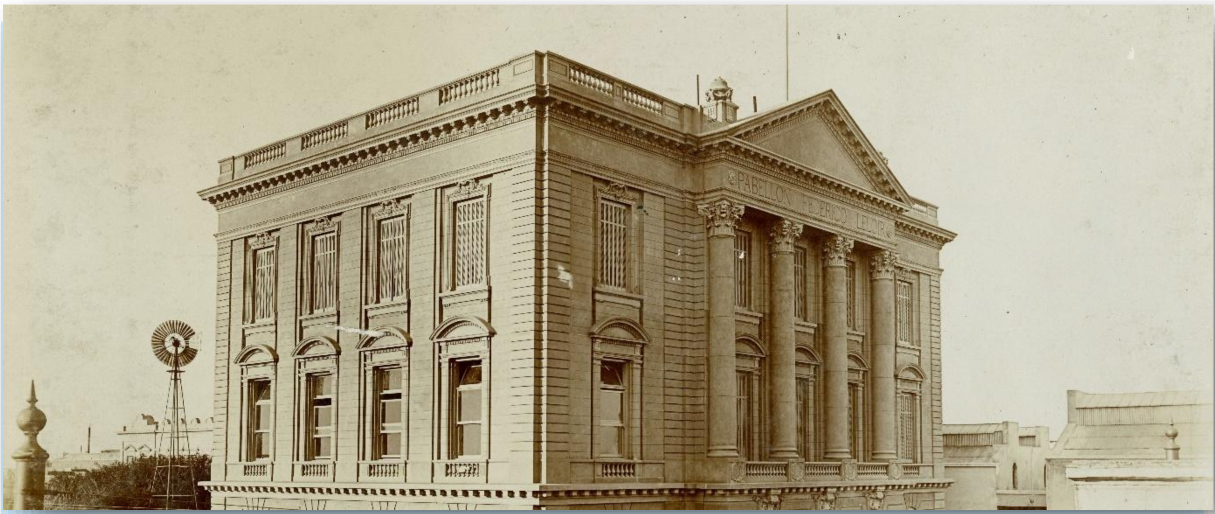
UTN

Tecnicatura Universitaria de Programación

Matemática

UNIVERSIDAD TECNOLÓGICA NACIONAL



CUADERNILLO

Álgebra de Boole – Parte 1

ÍNDICE

ÁLGEBRA DE BOOLE

INTRODUCCIÓN

¿QUE ES LA LÓGICA?

La lógica es el estudio de proposiciones y su valor de verdad. Básicamente, te ayuda a pensar en forma ordenada y a deducir nueva información a partir de datos o conocimientos previos.

En el contexto del estudio de la programación de computadoras la vamos a aplicar de dos maneras:

1) Para analizar enunciados y poder desarrollar programas.
2) Para comprender como “piensa” una computadora y así usarla a nuestro favor.

HISTORIA BREVE DE LA LÓGICA DE BOOLE

La lógica de Boole fue desarrollada en el siglo XIX por el matemático y lógico inglés George Boole (1815-1864). Su trabajo principal estableció las bases de un sistema algebraico que permitía representar el razonamiento lógico mediante operaciones matemáticas.

Boole propuso que el pensamiento lógico podía expresarse con símbolos y reglas similares a las del álgebra, usando valores binarios: 1 (verdadero) y 0 (falso). Introdujo las operaciones básicas que hoy conocemos como AND (∧), OR (∨) y NOT (¬), que permiten combinar proposiciones lógicas.

Aunque en su época su trabajo era puramente teórico, en el siglo XX resultó fundamental para la informática y la electrónica digital. En 1938, Claude Shannon aplicó la lógica booleana a los circuitos eléctricos, sentando las bases de la computación moderna.

Hoy, la lógica de Boole es la base del funcionamiento de computadoras, inteligencia artificial y sistemas digitales en general.

LA COMPUTADORA PENSADA EN NIVELES

NIVEL 5: Del lenguaje orientado hacia los problemas

- Traducción (compilador)

NIVEL 4: Del lenguaje ENSAMBLADOR

- Traducción (ensamblador)

NIVEL 3: Del SISTEMA OPERATIVO

- Interpretación parcial

NIVEL 2: De la arquitrectura de conjunto de INSTRUCCIONES

- Interpretación (microprograma) o ejecución directa

NIVEL 1: De **MICROARQUITECTURA**

- Hardware

NIVEL 0: DE **LÓGICA DIGITAL**

Nosotros sentaremos las bases para el estudio del NIVEL 0 (la lógica digital) y el NIVEL 1 de (microarquitectura) con un álgebra muy especial, la de Boole.

LÓGICA DIGITAL

Este tema está en un lugar entre la electrónica y las ciencias de la computación. Entenderlo nos permite comprender que el funcionamiento de una computadora no es mágico, sigue una lógica y es de esa lógica la que nos aprovecharemos para realizar nuestros programas.

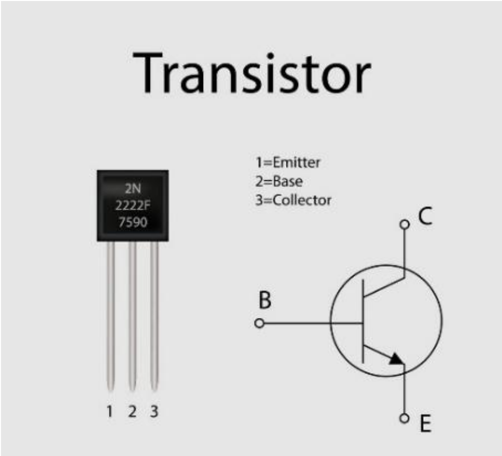
Los elementos que la compone son paradójicamente fáciles de entender, lo que nos asombra es su sencillez.

Básicamente usaremos un álgebra de solo dos valores, unos elementos electrónicos llamados transistores, uno pequeño de grupo de ellos formando compuertas lógicas que se comportan igual simulando los operadores lógicos que conocemos, y un conjunto de compuertas conforman todo lo que ya conocemos: procesadores, memoria…

Dentro de la lógica digital nos concentraremos en desarrollar el ÁLGEBRA DE BOOLE, dejando el resto para el estudio de la arquitectura de computadoras.

PARA QUÉ ESTUDIAMOS ÁLGEBRA DE BOOLE

Para entender el NIVEL 0, el de la electrónica. Para eso brevemente te mostramos a la estrella que permitió hacerlo posible: el transistor, y el elemento base que nos interesa conocer, las compuertas lógicas.



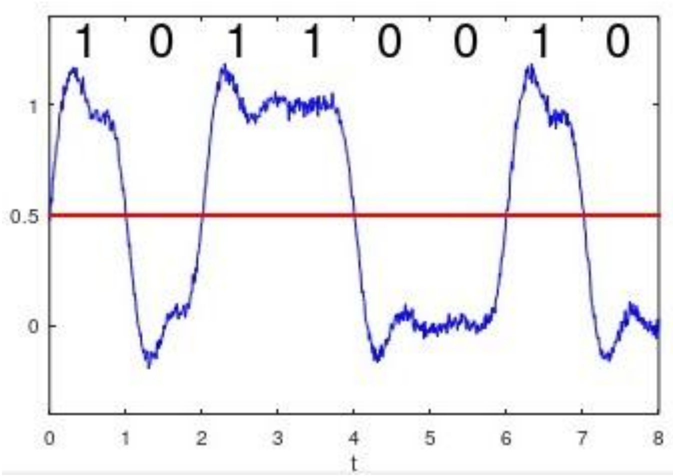
SEÑALES

Seguramente habrás escuchado, o pronto te pasará, que en informática hablamos de señales.

Este es un término ampliamente utilizado en diversos contextos: En Sistemas Operativos donde responde a señales de los programas de usuario o el hardware, en programación a tiempo real que responde a señales de los sensores, pero al nivel de lógica digital decimos:

Una señal es una variación de voltaje que transmite información. Esta puede ser analógica o digital. En nuestro caso hablaremos de SEÑALES DIGITALES.

Estas representan la información mediante valores discretos (generalmente 0 y 1), lo que es fundamental para el procesamiento y transmisión de datos en sistemas digitales.



COMPONENTES BÁSICOS DEL ÁLGEBRA

Para comenzar con el estudio del álgebra de Boole debemos establecer un sistema axiomático. Para ello nos basaremos en las “reglas” o propiedades que ya estudiamos.

Conocimiento previo:

- Teoría de conjuntos
- Teoría de números
- Principios de Lógica

Así que eso haremos. Para jugar este juego te explicaremos sus reglas.

COMPONENTES

ELEMENTOS

Usaremos dos:

S = {0, 1}

OPERADORES

Suma lógica u OR (+)
Multiplicación o AND (.)

OPERACIÓN UNARIA

El complemento de un elemento

En resumen, dos elementos 0 y 1; dos operaciones binarias, suma y multiplicación; y una unaria, el complemento.

CLAUSURA

Para todo elemento A y B ∈ S las operaciones binarias A + B (suma lógica) y A . B (producto lógico) también pertenecen a S.

Dicho en criollo, para todo 1 ó 0, al aplicar las operaciones, obtenemos 1 ó 0. Ni 2, ni -1.

VARIABLES, LITERALES y EXPRESIONES BOOLEANAS

Usamos la expresión A y B, que las operamos con álgebra de Boole.

Variable booleana

A y B son variables booleanas que pueden valer 1 ó 0, pero como no lo sabemos las llamamos así.

Literales

Pronto usaremos las variables booleanas en forma directa (sin modificar) por ejemplo A, o en forma negada o complemento por ejemplo A´o $\overline{A}$.

Ojo, si ponemos A no estamos diciendo que A = 1. Estamos diciendo que el valor que adopte, 1 ó 0 se mantiene así. Si ponemos A´ no estamos diciendo que A = 0 estamos indicando que el valor que adopte se invierte.

Expresión booleana

Cuando combinamos dos o más variables booleanas en su forma directa o negada (literales) con operadores booleanos. Por ejemplo, A . B + C . $\overline{A}$.

De ahora en más, cuando mencionamos variables booleanas, estamos hablando de variables que solo pueden ser 1 ó 0, o TRUE o FALSE.

OPERADORES BOOLEANOS

SUMA LÓGICA, no es lo mismo que la suma binaria

A + B =

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 1

En esto último renglón vemos que es diferente al sumar 1 + 1 en suma binaria.

Equivale al **OR** en lógica.

MULTIPLICACIÓN LÓGICA

A . B =

0 . 0 = 0

0 . 1 = 0

1 . 0 = 0

1 . 1 = 1

Equivale al **AND** de lógica.

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

REPRESENTACIÓN DEL COMPLEMENTO EN EXPRESIONES BOOLEANAS

Lo solemos expresar de varias formas:

- Con la virgulilla: ~A
- Con una comilla simple: A´
- O con una barra horizontal: $\overline{A}$
- O con ¬A

En papel es preferible la barra porque cuando simplificás es más rápida de trabajar y visible.

La comilla simple es más práctica si usas un editor de texto, la virgulilla se ve poco.

PROPIEDADES Y REGLAS BOOLEANAS

PROPIEDADES BÁSICAS o ALGEBRAICAS

Podemos establecer reglas básicas que son las mismas que estudiamos con los números decimales.

Las que nos permite que los elementos que son operados sean movidos o agrupados como queramos.

CONMUTATIVAS	ASOCIATIVAS
A + B = B + A	**(A + B) + C = A + (B + C)**
A . B = B . A	**(A . B) . C = A . (B . C)**

Todas las operaciones pueden “darse vuelta” y si se operan 3 o más, se pueden agrupar como quieras.

Éstas son bastante intuitivas porque son idénticas a las reglas de los números decimales.

Pero no siempre son iguales, así que no te confíes.

DISTRIBUTIVAS

$A \,.\, (B + C) = (A \,.\, B) + (A \,.\, C)$

$A + (B \,.\, C) = (A + B) \,.\, (A + C)$

O las recíprocas

$(A \,.\, B) + (A \,.\, C) = A \,.\, (B + C)$

$(A + B) \,.\, (A + C) = A + (B \,.\, C)$

Si una multiplicación lógica opera sobre un paréntesis con una suma, podés distribuirla, lo mismo al revés.

Acá vemos una regla que no podemos usar en decimales. La segunda Ley. Por ejemplo: Si tenemos 4 + (3 . 2) aplicamos PEMDAS y nos resulta 10. Pero si intentamos distribuir como vemos aquí tendríamos 12 + 8 = 20.

Así que no caigas en la tentación de usar las mismas reglas que estudiamos antes.

REGLAS DERIVADAS DE LA LÓGICA USANDO TABLAS DE VERDAD

ELEMENTO IDENTIDAD

$$A + 0 = A$$
$$A \,.\, 1 = A$$

LEYES DE ANULACIÓN

$$A + 1 = 1$$
$$A \,.\, 0 = 0$$

Si sumo a cualquier elemento un 0 queda tal cual. Lo mismo si multiplico por 1 a cualquier elemento.

Si a A le sumo un 1 queda 1, ¿por qué? Es un OR, no importa el valor de A, si el otro es 1 siempre la suma da 1.

Si multiplico, los dos tienen que ser 1 para resultar 1. En cualquier otro caso, basta que haya un 0 para que el resultado sea igual a 0.

ELEMENTO COMPLEMENTARIO

$$A + \bar{A} = 1$$
$$A \,.\, \bar{A} = 0$$

Más complicado entenderlo. Recordemos conjuntos, si sumamos cualquier conjunto con su complemento (una unión) obtenemos el todo, el universal. Si un conjunto lo queremos interceptar con su complemento, es obvio que resulta en un conjunto vacío. En este caso 0.

LAS OPERACIONES BOOLEANAS LLEVAN A DOS LEYES MÁS

LEYES DE IDEMPOTENCIAS

O sea, mantiene el valor

A + A = A

A . A = A

Si observás las tablas de verdad de OR, 1 + 1 es 1 y 0 + 0 es 0.

Si observás las tablas de verdad de AND, 1 . 1 es 1 y 0 . 0 es 0.

LAS LEYES DISTRIBUTIVAS NOS LLEVAN A MÁS

LEYES DE ABSORCIÓN

$$A + (A . B) = A$$

$$A . (A + B) = A$$

$$A + (\bar{A} . B) = A + B$$

$$A . (\bar{A} + B) = A . B$$

Unas de las leyes que más nos ayuda para simplificar expresiones booleanas muy complejas. Veamos como llegamos a ellas.

Tomemos la primera y busquemos pasar de la expresión de la izquierda a la derecha. Esto se llama simplificar, que usa intensamente las leyes, reglas y propiedades que vimos.

$$A + (A . B) = A . (1 + B) = A . 1 = A$$

¿Y SI NEGAMOS DOS VECES?

En la vida real es difícil ver estructuras semánticas como “No no ...” Sin embargo, si existen expresiones de doble negación. Las usamos todos los días de manera contraria a lo que realmente significa.

“No digas nada”. Decir nada es no decir algo, pero si digo no digas nada estoy haciendo una doble negación. Realmente estoy diciendo que diga algo.

“No pasa nada” es otro ejemplo.

Acá vamos a negar dos veces a una variable booleana:

LEY DE INVOLUCIÓN

$$\overline{\overline{A}} = A$$

También es útil para simplificar, si tenés doble rayita, las podés eliminar.

Y HABLANDO DE NEGACIONES...

LEYES DE DE MORGAN

$$\overline{A + B} = \overline{A} \,.\, \overline{B}$$

$$\overline{A \,.\, B} = \overline{A} + \overline{B}$$

Básicamente si negás una suma es igual al producto de las variables negadas. Si negás una multiplicación es igual a la suma de las variables negadas.

Antes que explote tu cabeza repasemos todas las reglas.

Aquí tienes un resumen:

Ley	Expresión en Álgebra de Boole	Descripción
Conmutativa	$A + B = B + A$ $A \cdot B = B \cdot A$	El orden de los términos no afecta el resultado.
Asociativa	$(A + B) + C = A + (B + C)$ $(A \cdot B) \cdot C = A \cdot (B \cdot C)$	El agrupamiento no afecta el resultado.
Distributiva	$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$ $A + (B \cdot C) = (A + B) \cdot (A + C)$	La multiplicación se distribuye sobre la suma y viceversa.
Identidad	$A + 0 = A$ $A \cdot 1 = A$	El 0 es neutro en la suma, el 1 es neutro en la multiplicación.
Anulación	$A + 1 = 1$ $A \cdot 0 = 0$	Cualquier valor sumado con 1 da 1; cualquier valor multiplicado por 0 da 0.
Complemento	$A + \overline{A} = 1$ $A \cdot \overline{A} = 0$	Un valor OR su negación da 1; un valor AND su negación da 0.
Idempotencia	$A + A = A$ $A \cdot A = A$	Un valor operado consigo mismo no cambia.
Absorción	$A + (A \cdot B) = A$ $A \cdot (A + B) = A$ $A + (\overline{A} \cdot B) = A + B$ $A \cdot (\overline{A} + B) = A . B$	Un término dentro de una operación ya está representado por la expresión. Si dentro se halla un negado del de afuera, se anula.
Involutiva	$\overline{\overline{A}} = A$	La doble negación devuelve el valor original.
De Morgan	$\overline{A + B} = \overline{A} \cdot \overline{B}$ $\overline{A . B} = \overline{A} + \overline{B}$	Expresa la relación entre OR y AND a través de la negación.

Ahora aprenderemos a simplificar expresiones

CIRCUITOS LÓGICOS

Ya estudiamos las variables booleanas, los literales, las expresiones booleanas, las reglas que las gobiernan. Ahora estudiaremos un poco de circuitos y luego relacionaremos el campo de la matemática lógica con la electrónica.

VARIABLE BOOLEANA COMO CIRCUITO

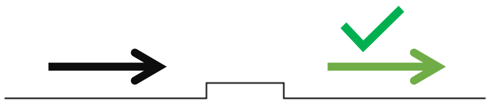
La forma más sencilla.

Consiste en una alimentación, una llave que deja pasar o interrumpe la corriente.

Si tenemos una variable A = 0 se representa con una llave abierta no dejando pasar corriente:

A = 0

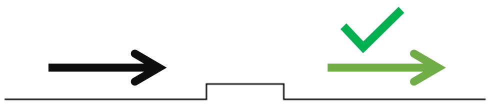
Si tenemos una variable A = 1 se representa con una llave cerrada:



A = 1

Pero no confundas abierta con igual a 0 porque si negamos la variable es al revés:

A’ = 1

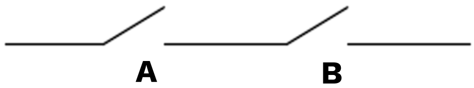


A’ = 0

Al ser la negación A’ = 1, el valor da A = 0.

DOS VARIABLES BOOLEANAS, CIRCUITO EN SERIE

En este caso vemos el siguiente circuito. Dos variables booleanas como interruptores montados en serie, uno a continuación del otro.



Analizamos:

¿Pasa corriente si ambos interruptores están cerrados?

¿Y si uno de ellos o ambos está abierto?

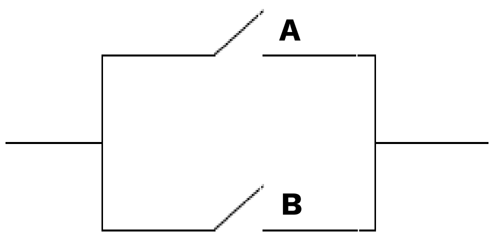
Armemos una tabla de verdad recordando que si está cerrado la variable es 1, abierto es 0:

A	B	A . B
0	0	0
0	1	0
1	0	0
1	1	1

O sea, un CIRCUITO SERIE equivale a una CONJUNCIÓN, AND o MULTIPLICACIÓN BOLEANA.

DOS VARIABLES BOOLEANAS, CIRCUITO EN PARALELO

En este caso vemos el siguiente circuito. Dos variables booleanas como interruptores montados en paralelo.



Analizamos los casos:

¿Pasará corriente si al menos uno de los interruptores está cerrado?

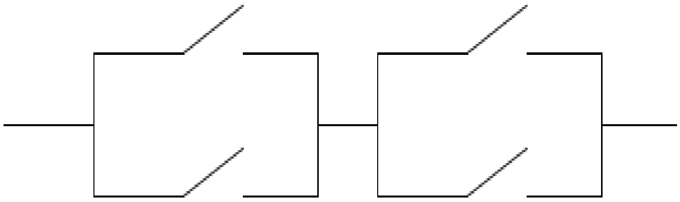
¿Qué tiene que suceder para que no pase?

A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

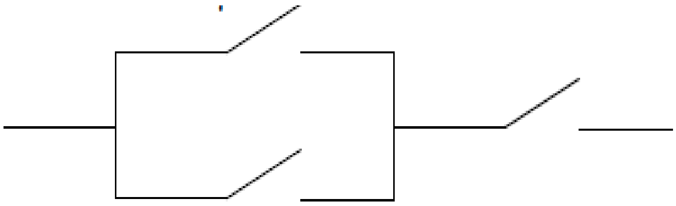
O sea, un CIRCUITO PARALELO equivale a una DISYUNCIÓN, OR o SUMA BOOLEANA.

EXPRESIÓN BOOLEANA Y CIRCUITOS

Si una + (OR) es una conexión paralela y una . (AND) es una conexión en serie, interpretemos la siguiente configuración:



Dos grupos de dos variables son sumas entre sí (paralelo) y estos grupos se multiplican entre sí (serie).



Un grupo que es una suma, multiplicado a otra variable.

Ya que adquiriste la vista veamos un ejemplo:

$$Z = (A + B) . (C + D)$$

¿A cuál circuito lo asociarías?

Y en este caso:

$$Z = (A + B') . (C' + D)$$

Si respondiste igual al anterior ya lo tenés claro. No importa si vez abierto o cerrado el interruptor, es lo de menos. Los interruptores se abren o cierran continuamente.

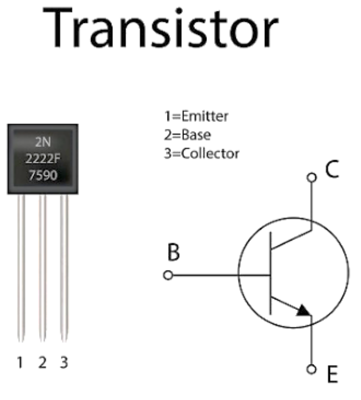
Si ponés variable directa B estás diciendo que cuando B = 1 el interruptor se cierra. Si pones B' estas diciendo que cuando B = 0, o sea B' = 1, el interruptor se cierra.

COMPUERTAS

BASE ELECTRÓNICA

Ya entendimos los circuitos que se comportan igual que los operadores lógicos. Es más, hasta avanzamos mostrándolos como expresiones.

La pregunta, ¿cómo hacemos para introducir interruptores en un chip de computadora?



Al principio de este cuadernillo empezamos con el transistor, también lo terminaremos con él.

El componente lo que hace es actuar como un interruptor. Recibe una señal y decide dejarla pasar o no.

En nuestra casa pulsamos o movemos una palanca, en el transistor es una señal la que le indica que deje o no pasar corriente.

Veamos la figura. Una corriente recorre el transistor desde el COLECTOR (patita 3) al EMISOR (patita 1). Controla si abre o cierra la señal que entra por la BASE (patita del medio).

En esencia, el transistor es el interruptor electrónico que, mediante un estímulo relativamente pequeño, permite o impide el flujo de corriente en un circuito. Lo hace indispensable para la conmutación digital y el procesamiento de señales en dispositivos electrónicos.

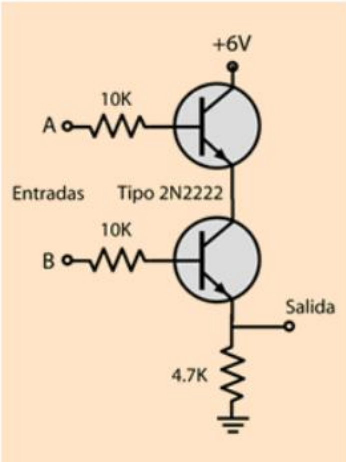
CIRCUITOS ELÉCTRICOS VERSUS COMPUERTAS

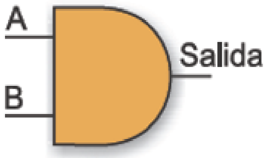
A los transistores pensémoslos como interruptores, entra la señal, sale la señal.

Por ende, podemos aplicar los conceptos de circuitos eléctricos que vimos en la sección anterior.

COMPUERTAS AND

Recordá que la BASE regula la apertura o cierre del flujo de corriente. Así que es por ahí donde entra la señal de la variable booleana. Una señal 1 por el cable A cierra el circuito. Lo mismo con B. Estamos en un circuito en SERIE.





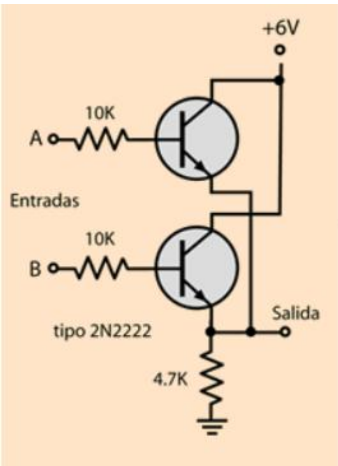
A	B	Salida
0	0	0
0	1	0
1	0	0
1	1	1

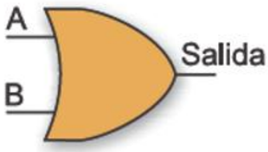
Para que no estés pensando en conexiones, ni ese dibujo raro de la izquierda, lo simplificamos con el ícono del medio, una pancita.

A la derecha, la misma tabla de verdad de una AND o INTERSECCIÓN o MULTIPLICACIÓN BOOLEANA.

COMPUERTAS OR

Observemos bien la conexión. Las entradas de ambos transistores están unidas, lo mismo que las salidas, es un circuito en PARALELO.





A	B	Salida
0	0	0
0	1	1
1	0	1
1	1	1

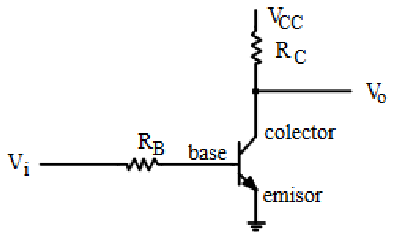
El icono es como un escudo acostado.

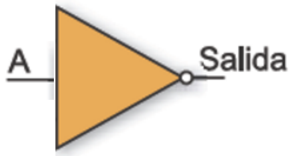
A la derecha la misma tabla de verdad de una OR o UNIÓN o SUMA BOOLEANA.

COMPUERTA NOT

Es un circuito inversor. O sea, buscamos invertir el sentido de la señal, si entra 1 devuelve 0, si entra 0 devuelve 1.

Pensemos un poco. Si conecto el EMISOR a tierra, lo que entre por el COLECTOR se descarga. O sea, si cierra el circuito con una señal 1 (cable V_i), por el cable V_0 no hay señal, mejor dicho, señal 0. Por el contrario, si hay señal 0 no se cierra el circuito y por la salida hay señal 1.

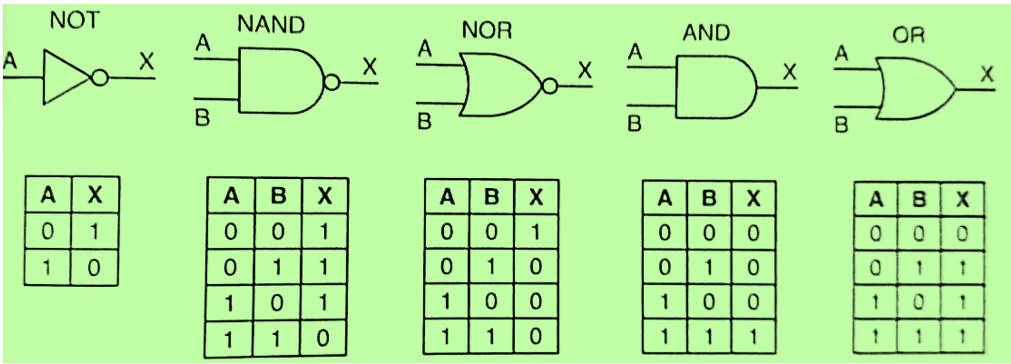




A	Salida
0	1
1	0

OTRAS COMPUERTAS Y TABLAS DE VERDAD

Hay otras. No te vamos a aburrir explicando el circuito de cada una de ellas, no nos interesa. Las compuertas AND, OR y NOT son las que más usaremos.



NOT

A	X
0	1
1	0

NAND

A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

NOR

A	B	X
0	0	1
0	1	0
1	0	0
1	1	0

AND

A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

Fuente: Tenenbaum, "Organización de Computadoras" 4° Edición (Prentice Hall, 2000)

APLICACIONES DE LAS COMPUERTAS

Lo vamos a desarrollar en la próxima clase cuando veamos las simplificaciones de expresiones booleanas.

Por ahora quédate que al mismo modo que con los circuitos eléctricos, podemos “traducir” expresiones booleanas a circuitos con compuertas o viceversa.

En la clase que viene:

Pensaremos un ejercicio, lo traduciremos a condiciones lógicas en una tabla de verdad, extraeremos una expresión algebraica, las simplificaremos con diversas técnicas y con esa expresión simplificada armaremos un circuito electrónico.

Tengo: A’ . B + A

Lo simplifico a: A + B usando la ley de absorción.

Lo traduzco a un circuito: Es una suma, una compuerta OR, cuyas señales son A y B directas: