Lògica binària

Porta NOT

Realitza una inversió a nivell d'entrada. Si a l'entrada hi ha un nivell alt (1), a la sortida hi haurà un nivell baix (0), i viceversa.

Simbol

Taula de veritat

Entrada	NOT (Sortida)
0	**1**
1	**0**

PORTA AND

La porta AND implementa la multiplicació lògica de les entrades. És a dir, sempre que almenys a una de les entrades hi hagi un 0, a la sortida hi haurà un 0. Només quan a les dues entrades hi hagi un 1 a la sortida hi haurà un 1 també.

Simbol

Taula de veritat

X	Y	AND (SORTIDA)
0	**0**	**0**
1	**0**	**0**
0	**1**	**0**
1	**1**	**1**

Porta OR

La porta OR implementa la suma lògica de les entrades. És a dir, sempre que almenys a una de les entrades hi hagi un 1, a la sortida hi haurà un 1. Només quan a les dues entrades hi hagi un 0 a la sortida hi haurà un 0 també.

Simbol

Taula de veritat

A	B	OR (SORTIDA)
0	**0**	**0**
1	**0**	**1**
0	**1**	**1**
1	**1**	**1**

PORTA XOR

La porta XOR implementa una desigualtat; si totes dues entrades són iguals, tant si a totes dues hi ha un 0 com si a totes dues hi ha un 1, el resultat és FALS (0), però si són diferents, és a dir, si a una entrada hi ha un 1 i a l'altra un 0, el resultat és CERT (1).

Simbol

Taula de veritat

A	B	XOR (SORTIDA)
0	**0**	**0**
1	**0**	**1**
0	**1**	**1**
1	**1**	**0**

Només és cert quan unes de les dues son diferents. Si son iguals és fals.

Son les mateixes però negades.

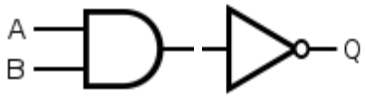
Porta NAND

La porta AND negada forma la porta NAND.

Simbol

Compte amb el símbol de negació que el diferència de la porta AND.

Desglossat



Taula de veritat

X	Y	NAND (SORTIDA)
0	**0**	**1**
1	**0**	**1**
0	**1**	**1**
1	**1**	**0**

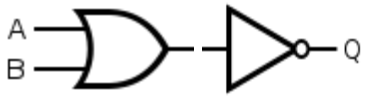
Porta NOR

La porta OR negada forma la porta NOR.

Simbol

Compte amb el símbol de negació que el diferència de la porta OR.

Desglossat



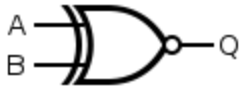
Taula de veritat

X	Y	NOR (SORTIDA)
0	**0**	**1**
1	**0**	**0**
0	**1**	**0**
1	**1**	**0**

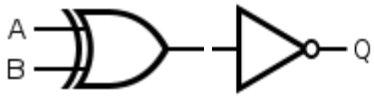
PORTA XNOR

Si la porta XOR implementa una desigualtat, la seva negació implementa la **igualtat** dels bits d'entrada; retornarà CERT (1) quan totes dues entrades són iguals, tant si a totes dues hi ha un 0 com si a totes dues hi ha un 1.

Simbol



Desglossat



Taula de veritat

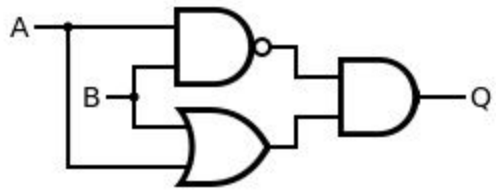
A	B	XNOR (SORTIDA)
0	**0**	**1**
1	**0**	**0**
0	**1**	**0**
1	**1**	**1**

Operacions primitives

Generalment es considera com a operacions primitives **OR**, **AND** i **NOT**. Els llenguatges de programació incorporen aquestes tres operacions de bit. De vegades també se incorpora l'operació XOR, proporcionant més expresivitat.

En els circuits lògics és important la simplicitat de construcció, i sovint es considera la porta NAND com a operació primitiva i les altres es deriven.

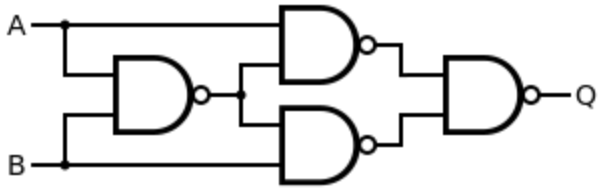
Exemple: Construcció de la porta XOR amb altres més simples.



Podem llegir **(**A **NAND** B**) AND (**A **OR** B**)**
Que és el mateix que **(**A **OR** B**) AND NOT (**A **AND** B**)**.

A	B	X1 = A NAND B	X2 = A OR B	Q = X1 AND X2
0	0	1	0	0
1	0	1	1	1
0	1	1	1	1
1	1	0	1	0

Tot i això en els circuits sovint es fan servir les portes NAND:



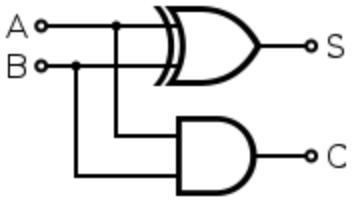
Verifiqueu amb taules de veritat el circuit anterior, i comproveu que efectivament el resultat és **A XOR B**.

Exemple: Suma de bits

De vegades no n'hi ha prou amb un bit de resultat; en l'operació de suma el resultat poden ser dos bits, perquè 1 + 1 = 10.

Llavors ens caldrà calcular tots dos bits.

La porta XOR és útil com a sumador de bits.



A	B	C (ròssec)	S (suma)
0	0	0	0
1	0	0	1
0	1	0	1
1	1	1	0

Exemple amb més entrades

X	Y	Z	P = X AND Y	Q = Y AND Z	P OR Q
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	1	1
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	1	0	1
1	1	1	1	1	1

Propietats de les operacions AND i OR

commutativa i associativa

commutativa	associativa
A OR B = B OR A	A OR (B OR C) = (A OR B) OR C = A OR B OR C
A AND B = B AND A	A AND (B AND C) = (A AND B) AND C = A AND B AND C

Propietat distributiva de AND respecte de OR

Podem veure AND com a un producte i OR com la suma. La propietat distributiva ens permet el que familiarment es diu "***treure factor comú***".

	(A AND B) OR (A AND C)
Distributiva AND respecte OR	**= A AND (B OR C)**

Prioritat de les operacions

En les expressions efectivament podem fer l'analogia d'AND amb el producte i OR amb la suma; així, l'operació AND té prioritat en front de l'operació OR, i podem estalviar parèntesi innecessaris.

	Prioritat d'operador AND més prioritat que OR (no fan falta parèntesi)
(A AND B) OR (A AND C)	**= A AND B OR A AND C**

Element neutre

OR	**0**	**A OR 0 = A**	**0 OR A = A**
AND	**1**	**A AND 1 = A**	**1 AND A = A**

Element invers

L'element invers és la negació

A	**NOT A**	**A AND NOT A = 0**	**A OR NOT A = 1**
0	**1**	**0**	**1**
1	**0**	**0**	**1**

Taula de veritat

La taula de veritat consisteix en una taula amb totes les combinacions possibles d'entrada, reflectint llavors, si és un circuit combinacional, totes les combinacions possibles de sortida.

Quan dues funcions tenen la mateixa taula de veritat vol dir que retornen les mateixes sortides per a totes les entrades possibles; si tenen el mateix comportament, **són circuits equivalents**, i per tant funcionalment iguals.

El nombre de diferents valor d'entrada depèn del nombre d'entrades: 2^N, sent N el nombre d'entrades.

Indicació per fer les taules de veritat: per tenir tots els possibles valors d'entrada, la millor manera és fer servir el binari pel valor de les entrades, considerant cada entrada com a un bit, com si fos significatiu l'ordre en què apareix en la taula; si escrivim en binari tots els números consecutivament, no ens deixarem cap valor.

Circuits combinacionals

Els circuits combinacionals **depenen únicament de les dades d'entrada**; el seu comportament queda reflectit si considerem tots els possibles valors d'entrada, fent servir taules de veritat.

Existeixen altres circuits, els circuits seqüencials, que són circuits amb memòria. En els circuits seqüencials les sortides també poden dependre d'entrades anteriors, perquè tenen memòria. No estudiarem els circuits seqüencials, només els combinacionals, però el comportament dels circuits seqüencials també es pot determinar fent servir taules de veritat, només que cal incorporar els estats interns.

Exemple (exercici)

Sigui la expressió: **(X AND NOT Y) OR (X AND NOT Z)**

- Quantes variables hi té?
- Feu la **taula de veritat**
- Quantes entrades hi té?
- Proveu a fer el **circuit**.
- Comprova que és el mateix resultat si treiem el factor comú:
    - **X AND (NOT Y OR NOT Z)**
    - Com ho comproves?
    - Quina hi fa servir més portes lògiques?
    - Quina hi fa servir més operacions?
    - *Ampliació*: Prova d'aplicar la llei de Morgan del punt següent (la trobaràs en la següent pàgina).
        - Tens ara menys operacions?
        - Comprova que és el mateix resultat.

Per saber-ne més: Lleis de Morgan (ampliació)

A més de les propietats que hem vist, que totes són propietats que coneixem bé des de petits per la suma i el producte, n'hi ha també les lleis de Morgan, que són molt útils per transformar expressions booleanes.

1. **NOT (A OR B) = NOT A AND NOT B**
2. **NOT (A AND B) = NOT A OR NOT B**

Compte! S'han pivotat les taules per motius d'espai (amplada). Es prega al lector que disculpi l'inconvenient. Les diferents opcions de la entrada es troben en filera, en comptes d'estar en una columna.

NOT (A OR B) = NOT A AND NOT B				
A	**0**	**1**	**0**	**1**
B	**0**	**0**	**1**	**1**
A OR B	**0**	**1**	**1**	**1**
NOT (A OR B)	**1**	**0**	**0**	**0**
NOT A	**1**	**0**	**1**	**0**
NOT B	**1**	**1**	**0**	**0**
NOT A AND NOT B	**1**	**0**	**0**	**0**

NOT (A AND B) = NOT A OR NOT B				
A	**0**	**1**	**0**	**1**
B	**0**	**0**	**1**	**1**
A AND B	**0**	**0**	**0**	**1**
NOT (A AND B)	**1**	**1**	**1**	**0**
NOT A	**1**	**0**	**1**	**0**
NOT B	**1**	**1**	**0**	**0**
NOT A OR NOT B	**1**	**1**	**1**	**0**