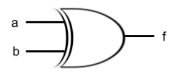
12.6 XOR / XNOR gates

XOR

A two-input ***XOR*** gate (for "exclusive OR") outputs 1 if the input values differ. Thus, y = a XOR b is equivalent to y = ab' + a'b. Digital designers often use the symbol ⊕ for XOR, as in: y = a ⊕ b.

Figure 12.6.1: XOR truth table and gate.

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



PARTICIPATION ACTIVITY 12.6.1: XOR.

1) 1 XOR 0 = ?
- 1
- 0

2) 1 XOR 1 = ?
- 1
- 0

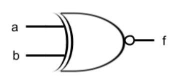
3) 0 XOR 0 = ?
- 1
- 0

XNOR

A two-input ***XNOR*** gate outputs 1 if the input values are the same. XNOR is the opposite (NOT) of an XOR gate, hence the "N". y = a XNOR b is equivalent to y = a'b' + ab.

Figure 12.6.2: XNOR truth table and gate.

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



PARTICIPATION ACTIVITY 12.6.2: XNOR.

1) 1 XNOR 0 = ?
- 1
- 0

2) 1 XNOR 1 = ?
- 1
- 0

3) 0 XNOR 0 = ?
- 1
- 0

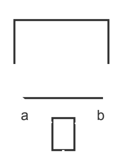
4) In contrast to an XOR gate, an XNOR gate's drawing has a _____ drawn at the output.
- square
- bubble

Basic XOR and XNOR examples

PARTICIPATION ACTIVITY 12.6.3: XOR example: Factory doors.

Start ☐ 2x speed

Goal: For fire safety, a factory's two doors must both be locked (a = 1, b = 1) when empty, or both be unlocked when people are present. If only one door is unlocked, a warning sounds (w = 1).



w = a XOR b   *w = 1 if a, b differ*

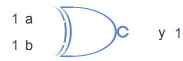
1 a  w 1

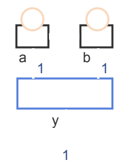
PARTICIPATION ACTIVITY 12.6.4: XNOR example: Park ride.

Start 2x speed

For balance, a two-seat amusement park ride is activated (output y = 1) only when there are two riders (a = 1, b = 1), or no riders.

y = a XNOR b *y = 1 if a, b are the same*

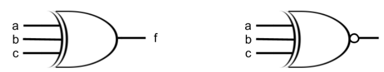




Multi-input XOR / XNOR

If XOR has more than two inputs, the output is 1 if the number of input 1's is odd. XNOR's output is 1 if the number of input 1's is even.

Figure 12.6.3: Truth table and gate for a 3-input XOR and and 3-input XNOR.



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

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

PARTICIPATION ACTIVITY 12.6.5: Multi-input XOR and XNOR.

1) 0 XOR 1 = ?
- 1
- 0

2) 1 XOR 1 = ?
- 1
- 0

3) 0 XOR 1 XOR 0 = ?
- 1
- 0

4) 1 XOR 1 XOR 0 = ?
- 1
- 0

5) 1 XOR 1 XOR 1 = ?
- 1
- 0

6) 1 XOR 0 XOR 1 XOR 1 = ?
- 1
- 0

7) 1 XNOR 1 XNOR 1 = ?
- 1
- 0

8) 1 XNOR 1 XNOR 1 XNOR 1 = ?
- 1
- 0

Example: Parity bit during data transmission

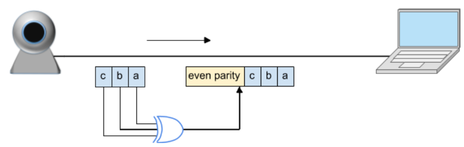
Digital devices commonly communicate bits, such as via a USB cable or via Bluetooth. Ex: A webcam may communicate 010 to a computer. Electrical noise can change a bit from 0 to 1 (or vice-versa), such as 010 changing to 110.

To help a receiver detect an erroneous communication, the sender sends an extra bit, called an ***even parity*** bit, such that the total number of 1's is even. So 010 is sent as 1 010, making the number of 1's even (2). 011 would be sent as 0 011.

An XOR gate quickly computes the desired parity bit.

On the receiving end, another XOR gate detects if the received bits have an odd number of 1's. If odd, the receiver rejects the data.

Figure 12.6.4: Parity during data transmission.



PARTICIPATION ACTIVITY 12.6.6: Parity bits.

1) a = 0, b = 0, c = 0, even parity = ?
- 1
- 0

2) a = 0, b = 1, c = 0, even parity = ?
- 1
- 0

3) a = 1, b = 0, c = 1, even parity = ?
- 1
- 0

Deriving XNOR's expression using DeMorgan's Law

If you've studied DeMorgan's Law, the following shows how XNOR's ab + a'b' can be derived by complementing XOR's a'b + ab'.

(a XOR b)'
(a'b + ab')'
(a'b)' · (ab')' DeMorgan's Law
(a'' + b')(a' + b'') DeMorgan's Law (again)
(a + b')(a' + b)
aa' + ab + b'a' + b'b
0 + ab + a'b' + 0
ab + a'b'
a XNOR b

PARTICIPATION ACTIVITY 12.6.7: DeMorgan's Law and XOR/XNOR.

1) XNOR is the NOT of XOR, so a XNOR b means (a'b + ab')'.
- True
- False

2) The expression for XNOR can be derived by applying DeMorgan's Law as follows: (a'b + ab')' = ab' + a'b.
- True
- False

Provide feedback on this section