Exclusive or and logic circuits

Exclusive or is true when two statements differ. What laws does it follow, how does a parity bit use it to catch errors, and how do logic gates build any statement out of wires?

WORDS TO KNOW

Exclusive or $P \oplus Q$: "P or Q, but not both". True exactly when P and Q differ.

Even parity bit: one extra bit added to a message so the whole word has an even number of 1s.

Circuit: gates joined by wires, each output feeding a later gate, with no loops.

Bit: a truth value written as a digit: 1 for T and 0 for F.

Logic gate: a small electronic part whose output is one connective of its inputs: AND, OR, NOT or XOR.

KEY IDEAS

1. The laws of exclusive or

$P \oplus Q$ reads as "P, flipped if Q is true". Its column (rows TT, TF, FT, FF) is F, T, T, F.

$$P \oplus Q \equiv Q \oplus P \qquad (P \oplus Q) \oplus R \equiv P \oplus (Q \oplus R)$$
$$P \oplus \mathrm{F} \equiv P \qquad P \oplus \mathrm{T} \equiv \neg P \qquad P \oplus P \equiv \mathrm{F}$$

So a chain of exclusive ors can be regrouped and reordered freely, like a sum. Pairs of the same letter cancel.

2. Parity: count the true parts

Each true part flips the chain once, starting from F. An even number of flips lands on F, an odd number on T.

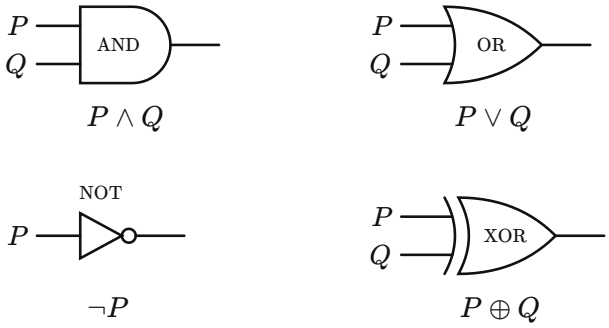
$P_1 \oplus P_2 \oplus \cdots \oplus P_n$ is true exactly when an **odd** number of the parts are true.

The even parity bit of a message $b_1 b_2 \ldots b_n$ is $p = b_1 \oplus b_2 \oplus \cdots \oplus b_n$. Changing any one bit of the word makes its count of 1s odd, so the check catches it.

3. Gates compute statements

Work from the inputs toward the output: each gate computes one column, as in a truth table. Every table has a circuit: for each row with T, take the "and" of the letters (negated where the row has F), then join these with "or".

Half adder. Two bits A and B add to the two-bit number CS, with sum bit $S = A \oplus B$ and carry bit $C = A \wedge B$.



Free sample from math.wiki for classroom use. Do not post. math.wiki

WATCH OUT

- **A chain is not "exactly one".** $P \oplus Q \oplus R$ is true when all three parts are true, since three is odd.
- **A parity bit misses an even number of errors.** Two changed bits flip the chain twice, so the word passes. It also never says which bit changed.
- **"And" distributes over $\oplus$; "or" does not.** $P \wedge (Q \oplus R) \equiv (P \wedge Q) \oplus (P \wedge R)$, but the "or" version fails when P is T and Q, R are F.
- **Crossing wires are joined only at a dot.**

WORKED EXAMPLES

Example 1. Simplify $\neg(P \oplus Q) \oplus (P \oplus \mathrm{T})$ to a single letter or its negation.

1. A negation is an exclusive or with T, since $X \oplus \mathrm{T} \equiv \neg X$. So $\neg(P \oplus Q) \equiv P \oplus Q \oplus \mathrm{T}$, and the statement is $P \oplus Q \oplus \mathrm{T} \oplus P \oplus \mathrm{T}$.
2. Reorder to put equal parts together: $(P \oplus P) \oplus (\mathrm{T} \oplus \mathrm{T}) \oplus Q$.
3. $P \oplus P \equiv \mathrm{F}$ and $\mathrm{T} \oplus \mathrm{T} \equiv \mathrm{F}$, and F changes nothing.

Answer: Q. **Check:** at P T, Q F: $P \oplus Q$ is T, so $\neg(P \oplus Q)$ is F, and $P \oplus \mathrm{T}$ is F. Then F $\oplus$ F is F, the value of Q.

Example 2. (a) Find the word sent for the message 1100101 with an even parity bit. (b) The word arrives as 11011010. What does the check show? (c) What if it arrives as 01101010?

1. (a) The message has four 1s, an even number, so the parity bit is 0. The word is 11001010.
2. (b) 11011010 has five 1s, an odd number. No correct word looks like this, so a bit was changed (here, the fourth).
3. (c) 01101010 has four 1s, so it passes. But compared with 11001010, the first and third bits both changed. Two flips cancel.

Answer: (a) 11001010. (b) The check fails: an error is found. (c) The check passes, yet two bits are wrong.

Example 3. A sprinkler runs when the timer is on and it is not raining, or when the manual switch is pressed. Write the rule as a statement, describe its circuit, and decide whether it runs when the timer is on, it is raining and the switch is not pressed.

1. Let T be "the timer is on", R be "it is raining" and M be "the switch is pressed". The rule is $(T \wedge \neg R) \vee M$.
2. The last connective is $\vee$, so the last gate is an OR. Its inputs are M and an AND gate fed by T and by a NOT gate on R. That is three gates.
3. With T true, R true and M false: $\neg R$ is F, so $T \wedge \neg R$ is F, and F or F is F.

Answer: $(T \wedge \neg R) \vee M$, built from a NOT, an AND and an OR gate. In this case the sprinkler does not run.

Free sample from math.wiki for classroom use. Do not post. math.wiki