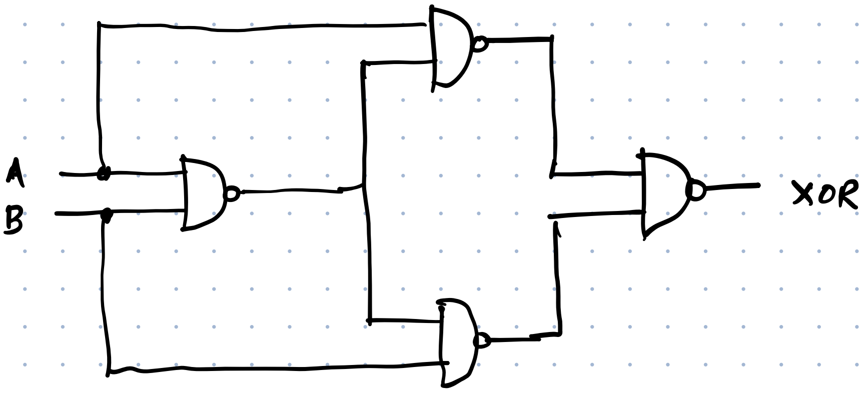
Jan 30, 2026

XOR gate using NAND gates

A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0



Add to +ve numbers ✓

Subtract B from A ? ⇝ $A - B = \underbrace{\overset{+ve}{\overset{\downarrow}{A}} + (\overset{-ve}{\overset{\downarrow}{-B}})}_{\text{addition}}$

We need a new way to represent −ve numbers that would allow us to carry out computation of the form: A − B.

1's Compliment

(i) +ve numbers ⟶ left most bit is 0. $\overset{\text{LMB}}{\boxed{0}}\ \boxed{x}\ \boxed{x}\ \boxed{x}$

(ii) −ve numbers ⟶ left most bit is 1 and the rest of the bits are flipped. $\boxed{1}\ \underbrace{\boxed{\ }\ \boxed{\ }\ \boxed{\ }}_{\text{flipped.}}$

+7 ⇝ $\boxed{0}\ \boxed{1}\ \boxed{1}\ \boxed{1}$

−6 ⇝ (a) 6 → 110

(b) 001 ↙ Flip

(c) $\boxed{1}\ \boxed{0}\ \boxed{0}\ \boxed{1}$ ←

Example: $\boxed{1}\ \underline{\boxed{0}\ \boxed{1}\ \boxed{1}\ \boxed{1}}$

(i) −ve

(ii) 0 1 1 1 $\xrightarrow{\text{Flip}}$ 1 0 0 0 ⟶ 8

(iii) −8

Example : $\boxed{0}\ \underbrace{\boxed{1}\ \boxed{0}\ \boxed{1}\ \boxed{0}}$

(i) +ve

(ii) 1010 ⟶ 10

(iii) +10

Example : $\boxed{0}\ \boxed{0}\ \boxed{0}\ \boxed{0}\ \boxed{0}\ \boxed{0}\ \boxed{0}\ \boxed{0}$

(i) +ve

(ii) 0000000

(iii) +0

Example : $\boxed{1}\ \boxed{1}\ \boxed{1}\ \boxed{1}\ \boxed{1}\ \boxed{1}\ \boxed{1}\ \boxed{1}$

(i) -ve

(ii) 1111111 $\xrightarrow{\text{Flip}}$ 000 0000

(iii) -0

* Solve 28 - 21

(i) Store 28 in 1's compliment

28 ⟶ 11100

$\boxed{0\ 0\ 0\ 1\ 1\ 1\ 0\ 0}$ ⟶ +28

(ii) Store -21 in 1's compliment

21 ⟶ 10101 $\xrightarrow[\text{Flip}]{}$ 01010

$\boxed{1\ 1\ 1\ 0\ 1\ 0\ 1\ 0}$ ⟶ -21

(iii)

$$\begin{array}{rcccccccccl}
① & 1 & & 1 & 1 & 1 & & & & & \\
 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & & (28) \\
 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & & (-21) \\
\hline
 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & & +6
\end{array}$$

no overflow

Wrong answer.
off by 1.

Rough Work

1 0000 +16
1 000 +8
1 00 +4

---

1 0000 +16
1 00 +4
1

10101

(iv) Convert the answer back to decimal

$$\underset{\downarrow \atop +ve}{0}\ \underbrace{0\,0\,0\,0\,1\,1\,0}_{\downarrow \atop 6}$$

$\boxed{+6}$

2's compliment

(i) Take 1's compliment and add 1 → when -v

- standard
- gets rid of double zero
- addition is easier / subtraction is easier

* Store -3 in 2's compliment.

$\boxed{1}\ \boxed{1}\ \boxed{0}\ \boxed{1}$ ← 2's compliment.

(i) -ve

(ii) $3 \rightarrow 0\,1\,1 \xrightarrow{flip} 1\,0\,0$

(iii) 1's compliment: 1 1 0 0

(iv) Add 1:

$$\begin{array}{r} 1100 \\ 1 \\ \hline 1101 \end{array}$$

* How to decode 2's compliment.

1 1 0 1

3 2 1 0 ← positions

$2^3\ 2^2\ 2^1\ 2^0$

↓

$-2^3\ 2^2\ 2^1\ 2^0$

$-8\ 4\ 2\ 1$

$1\times(-8) + 1\times(4) + 0\times(2) + 1\times(1)$

$= -8+4+1$

$= \boxed{-3}$

* Example: Convert 1 1 1 0 0 0 0 1 that is stored in 2's compliment into decimal.

1	1	1	0	0	0	0	1
7	6	5	4	3	2	1	0

✓	✓	✓					✓
−128	64	32	16	8	4	2	1
×1	×1	×1	×0	×0	×0	×0	×1

* Example: 108 − 20

(i) Store +108 in 2's compliment:

(ii) Store −20 in 2's compliment:

(iii) Add $\underline{108 + (-20)}$

88

$$\begin{array}{r} ①\,1\,1\;\;1\,1\;\;\; \\ 0\,1\,1\,0\,1\,1\,0\,0 \\ 1\,1\,1\,0\,1\,1\,0\,0 \\ \hline 0\,1\,0\,1\,1\,0\,0\,0 \\ 64\;32\;16\;8\;4\;2\;1 \end{array}$$

↓

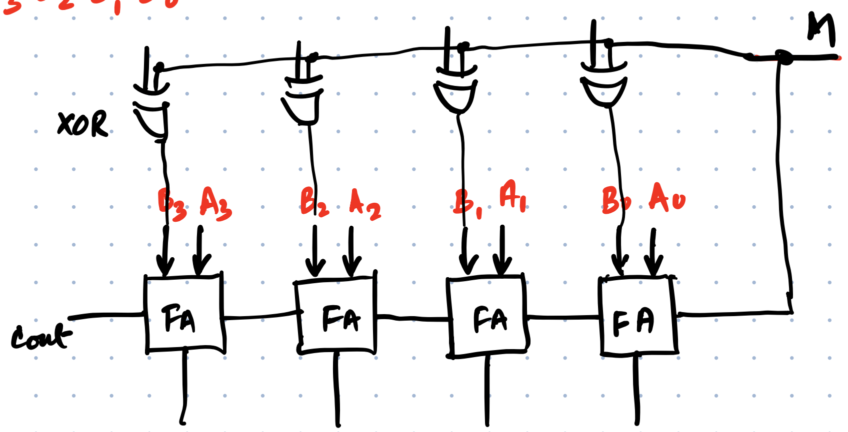
+ve

$64 + 16 + 8 = \underline{\underline{88}}$

* Subtractor Circuit

A: $A_3\ A_2\ A_1\ A_0$

B: $B_3\ B_2\ B_1\ B_0$



0: A+B

1: A−B

* Overflow: result of an arithmetic operation cannot be correctly represented in the allotted bits.

(i) Unsigned integers: recognized by a carry out.

Carry out

$$\begin{array}{r} ①1 \\ 1\,1 \\ 1\,0 \\ \hline 1\,0 \end{array}$$

(ii) Signed integers:

(a) adding a positive and a negative number cannot result in an overflow.

(b) adding two positive or negative numbers can result in an overflow.

↓

sign of the result should match the sign of the two numbers.