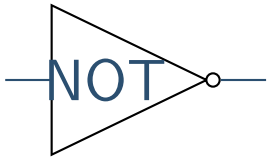
Types of data

The basic unit of data is a bit — 0 or 1.

Bit string — 01101000

- chars
- numbers
- images
- sounds
- truth values
  - 0 – false
  - 1 – true

Gates



$$\neg 0 = 1; \quad \neg 1 = 0;$$

x_1	NOT(x_1)
0	1
1	0

Gates

$$0 \wedge 0 = 0;\ 0 \wedge 1 = 0;\ 1 \wedge 0 = 0;\ 1 \wedge 1 = 1;$$

x_1	x_2	AND(x_1, x_2)
0	0	0
0	1	0
1	0	0
1	1	1

Gates

$$0 \vee 0 = 0;\ 0 \vee 1 = 1;\ 1 \vee 0 = 1;\ 1 \vee 1 = 1;$$

x_1	x_2	$\mathsf{OR}(x_1, x_2)$
0	0	0
0	1	1
1	0	1
1	1	1

Gates

$$0 \oplus 0 = 0;\ 0 \oplus 1 = 1;\ 1 \oplus 0 = 1;\ 1 \oplus 1 = 0;$$

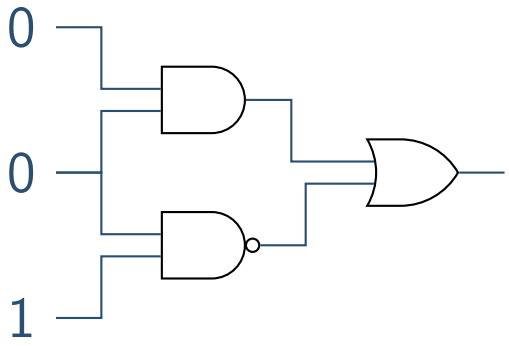
x_1	x_2	XOR(x_1, x_2)
0	0	0
0	1	1
1	0	1
1	1	0

Gates

0 nand $0 = 1$; 0 nand $1 = 1$; 1 nand $0 = 1$; 1 nand $1 = 0$;

x_1	x_2	NAND(x_1, x_2)
0	0	1
0	1	1
1	0	1
1	1	0

Example circuit



What are the top, bottom and rightmost gates?

A. AND, NAND, XOR

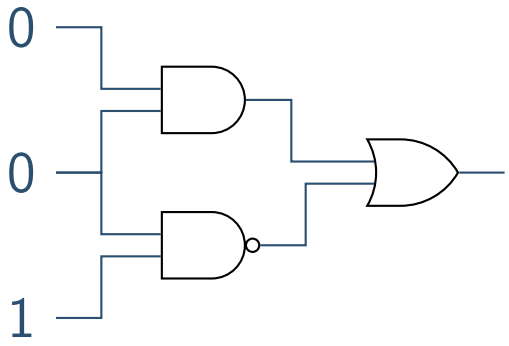
B. OR, NAND, XOR

C. AND, NAND, OR

D. OR, NAND, OR

Vote at m.socrative.com. Room number 415439.

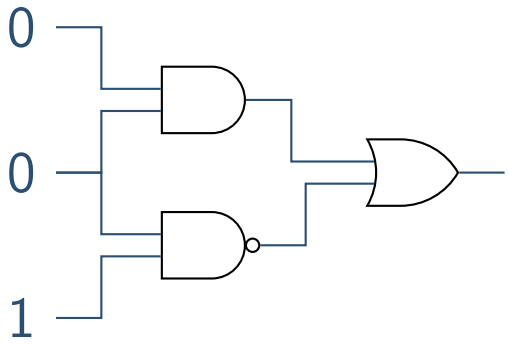
Example circuit



What are the top, bottom and rightmost gates?

C. AND, NAND, OR

Example circuit



What is the output of this circuit?

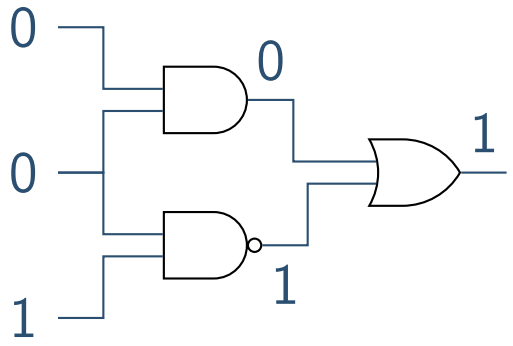
A. 0

B. 1

C. not defined

Vote at m.socrative.com. Room number 415439.

Example circuit

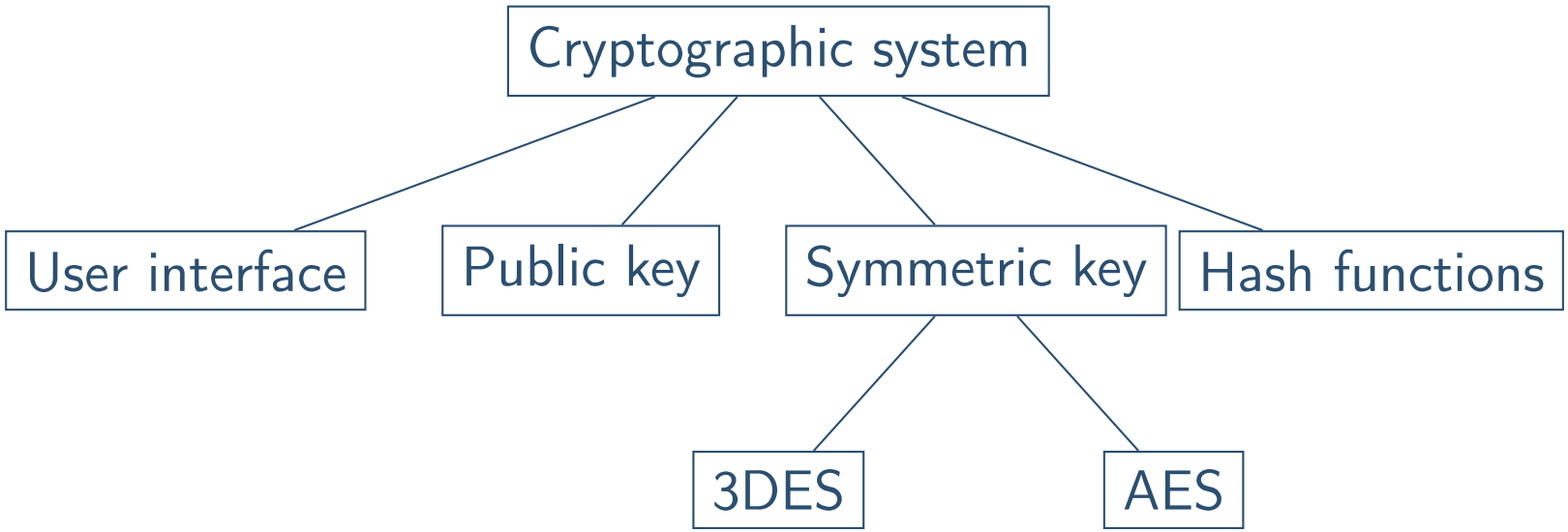


What is the output of this circuit?

B. 1

Abstraction

Example: Top-down design - cryptographic system



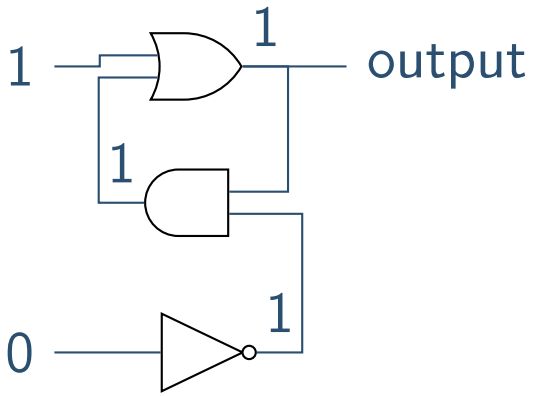
Abstraction

Things at higher levels need not know how things at lower levels function, only how to use them.

Interface, modularity, and modelling give:

- Structure — divide up work
- Independence between modules
  (can re-implement without changing the rest)
- Ability to analyze
- Increased innovation, productivity
  (don't need to re-invent the wheel)

Flip flop

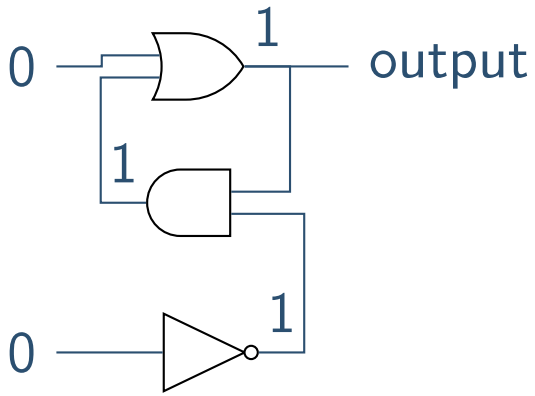


Note that this is stable.

Keeps same output until temporary outside pulse.

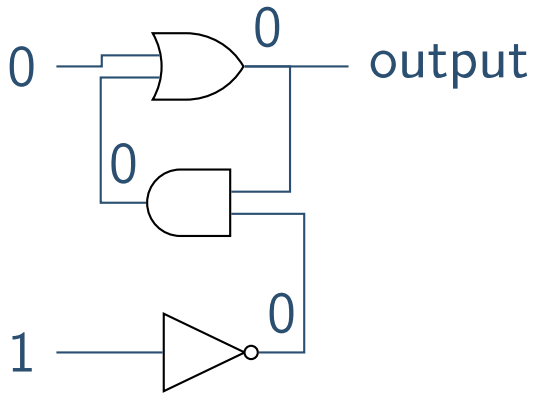
Can store a bit.

Flip flop



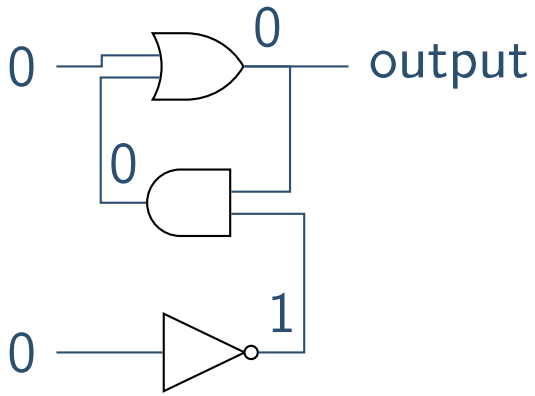
Note that this is stable.

Flip flop



Note that this is stable.

Flip flop



Note that this is stable.
But two different stable outputs are possible with input (0,0).

Flip flops can be implemented differently. Fig. 1.5, p. 38.
Abstraction: know input/output effect —
don't care about implementation.