# Boolean Algebra and Logic Gates

## Part 1: Boolean Algebra

Boolean algebra originates from George Boole, 19th Century British mathematician and philosopher — see excerpt from his Laws of Thought on resources page.

Boolean algebra deals with 2 states:

true , false
on , off
1 , 0

Boolean variable : variable that can take one of two values (typically 0 or 1).

Boolean function : - inputs are one or more Boolean variables.
- output is a single Boolean variable.

[why are we interested? Because this is all computers do, all day every day!!]

Can define a Boolean function by a truth table

e.g. $f(a,b)$ defined by

| $a$ | $b$ | $f(a,b)$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

e.g. $g(a,b,c)$ defined by

| $a$ | $b$ | $c$ | $g(a,b,c)$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

Elementary Boolean functions include:

AND

| $a$ | $b$ | $a$ AND $b$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

OR

| $a$ | $b$ | $a$ OR $b$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

NOT

| $a$ | NOT $a$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

Notation varies quite a bit:

$$a \text{ AND } b \equiv ab \equiv a \cdot b \equiv a \wedge b \equiv a \,\&\&\, b$$
$$a \text{ OR } b \equiv a + b \equiv a \vee b \equiv a \,||\, b$$
$$\text{NOT } a \equiv \bar{a} \equiv a' \equiv \sim a \equiv \neg a \equiv\, !a$$

architecture text book

AI textbook

Java

Some logic books even use $a+b$ for $a$ AND $b$ !
BEWARE!!

We can combine elementary Boolean functions to get <u>any</u> Boolean function

e.g. $\bar{a} b$ :

| $a$ | $b$ | $\bar{a}$ | $\bar{a} b$ |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 |

looking back, we see $f(a,b) = \bar{a} b$.

[Demo TruthTable.java]

**Activity**: Consider the Boolean function $c\bar{a}\bar{b} + a\bar{b}\bar{c}$

1. work out the truth table by hand
2. check using TruthTable.java
3. look back - is this function the same as one we've seen already?

optional:

4. try $\bar{b}(c\bar{a} + a\bar{c})$

   is this the same as something we've seen?

$\longrightarrow$ important point: same function can be represented by different formulas

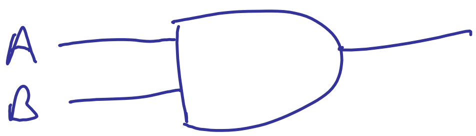
e.g. $\bar{b}(c\bar{a} + a\bar{c}) = ca\bar{b} + a\bar{b}\bar{c}$

# Part 2: Logic Gates

- Computer circuits are built out of <u>gates</u> that implement simple Boolean functions.
- A gate is built from a few <u>transistors</u>.

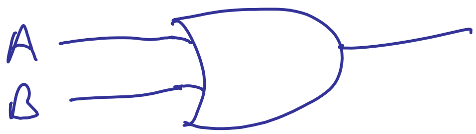
- Common gates and their circuit symbols are:

AND



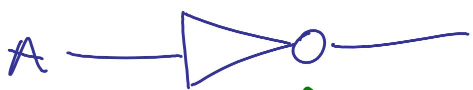

$AB$

OR




$A + B$

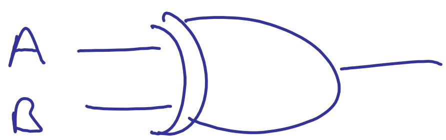
NOT




$\bar{A}$

sometimes this "o" represents a NOT attached to another gate

XOR



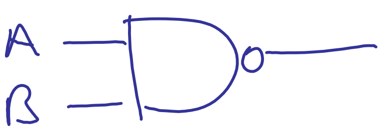

$A\bar{B} + \bar{A}B$
$\equiv \underbrace{A \oplus B}$

symbol for XOR

i.e.

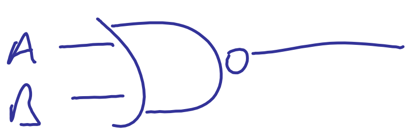
| A | B | A XOR B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

NAND




$\overline{AB}$

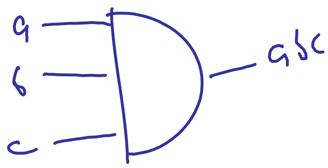
NOR

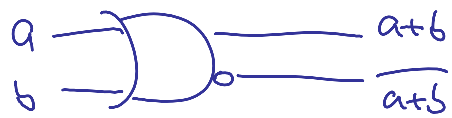



$\overline{(A+B)}$

- NAND is a <u>universal</u> gate i.e. you can build all others from it
- NOR is also universal.
- gates can have multiple inputs or outputs

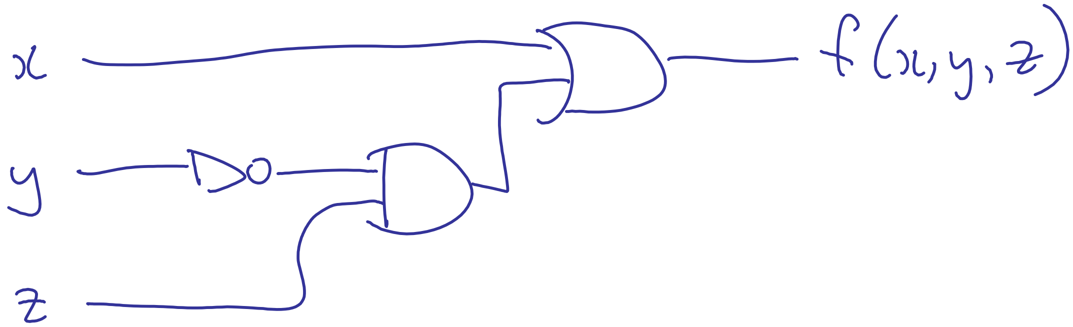
e.g.




- Any Boolean function can be implemented using a combination of gates

e.g. $f(x, y, z) = x + \bar{y}z$




ACTIVITY: build f using SimCir.