CS/COE 0447

Implementing Logic:

Teaching the Machine to “think”

wilkie (with content borrowed from:

Jarrett Billingsley

Dr. Bruce Childers)

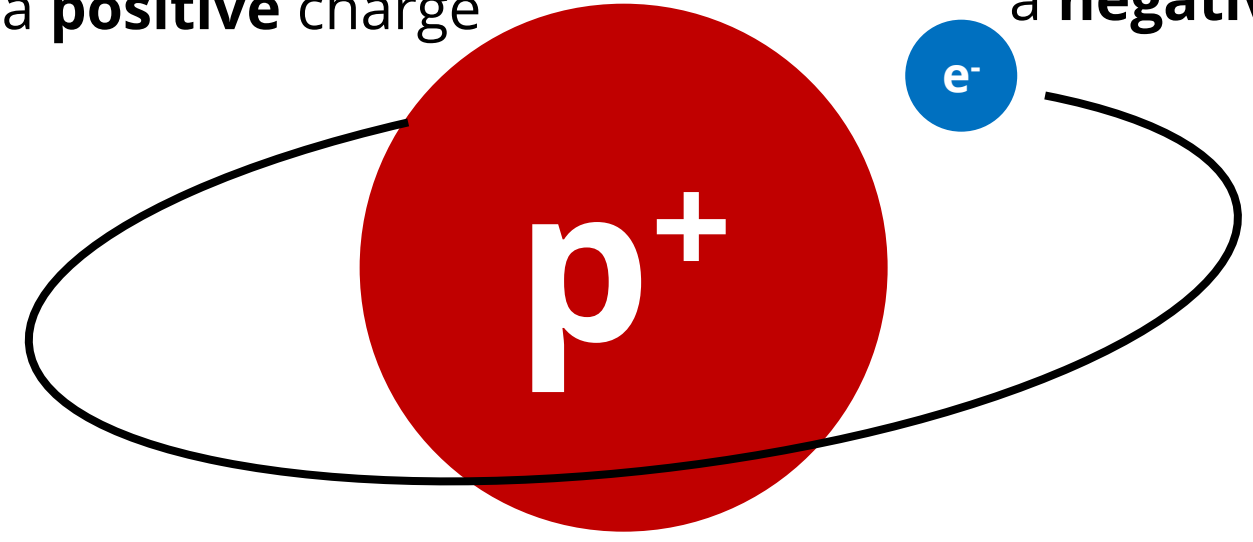
What's electricity?

(for fun section)

In your orbit

here's a proton. it has a **positive** charge

here's an electron. it has a **negative** charge

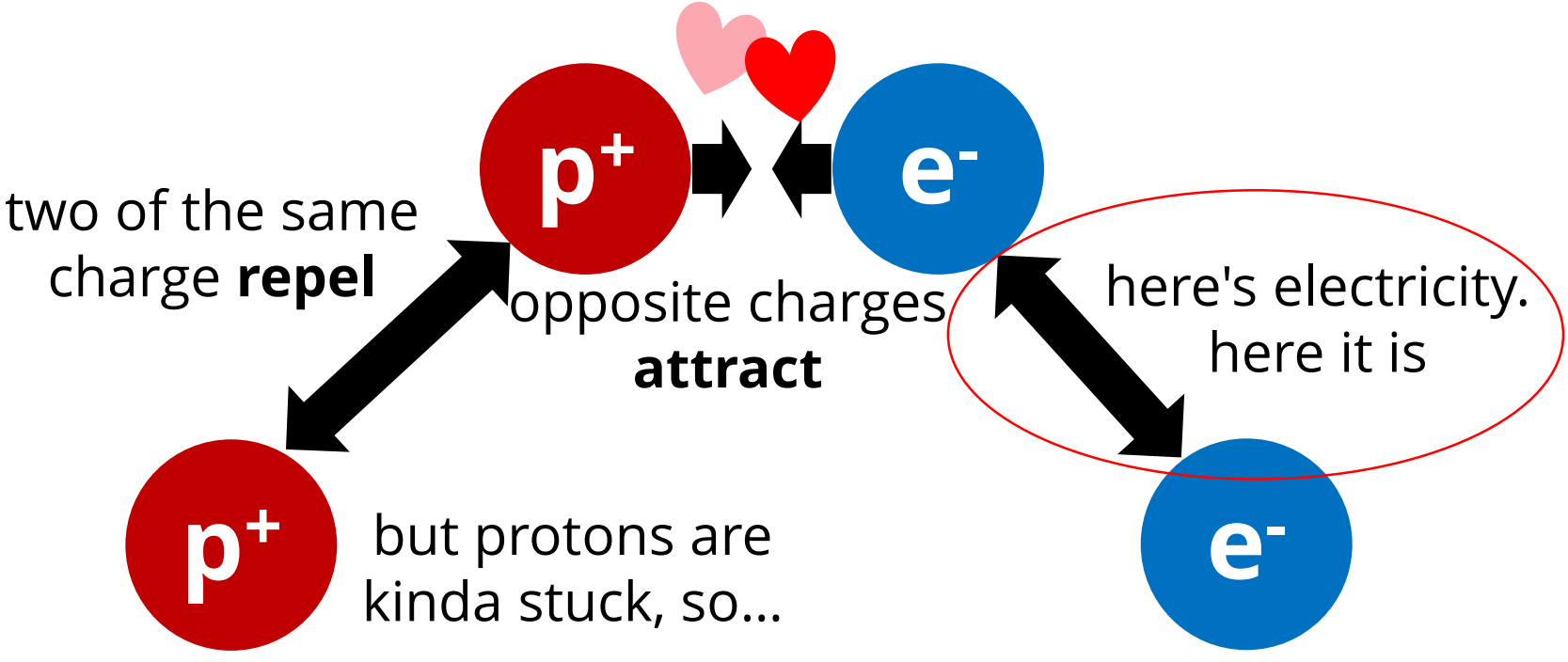


it kinda goes around the proton in an orbit*

protons sit still while electrons can move around

narrator: that was not true.

Opposites attract...

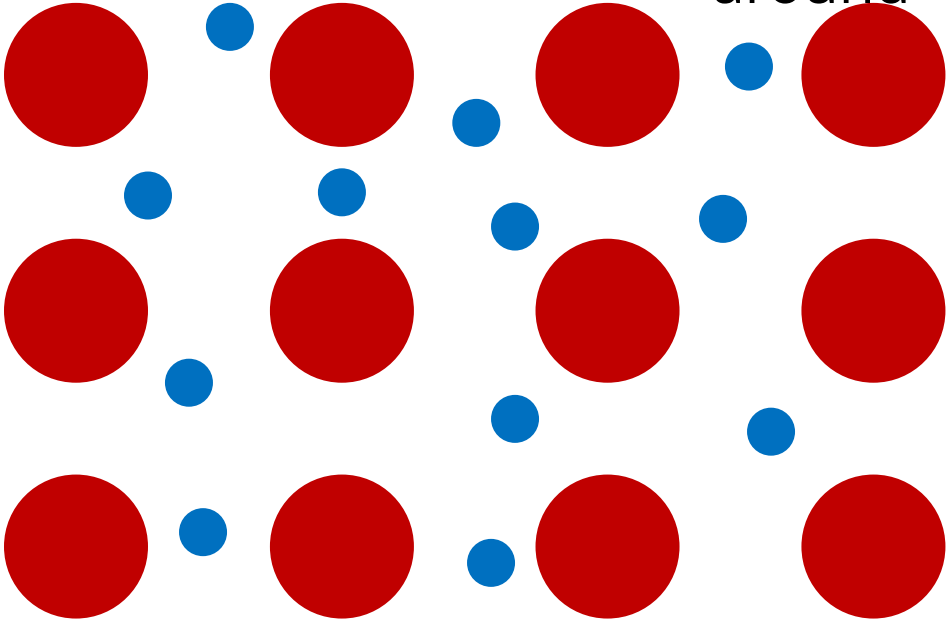


Lots of ~~fish~~ electrons in the sea

here's a solid piece of metal

the atoms are in a fixed structure

but *some* of the electrons are free to move around

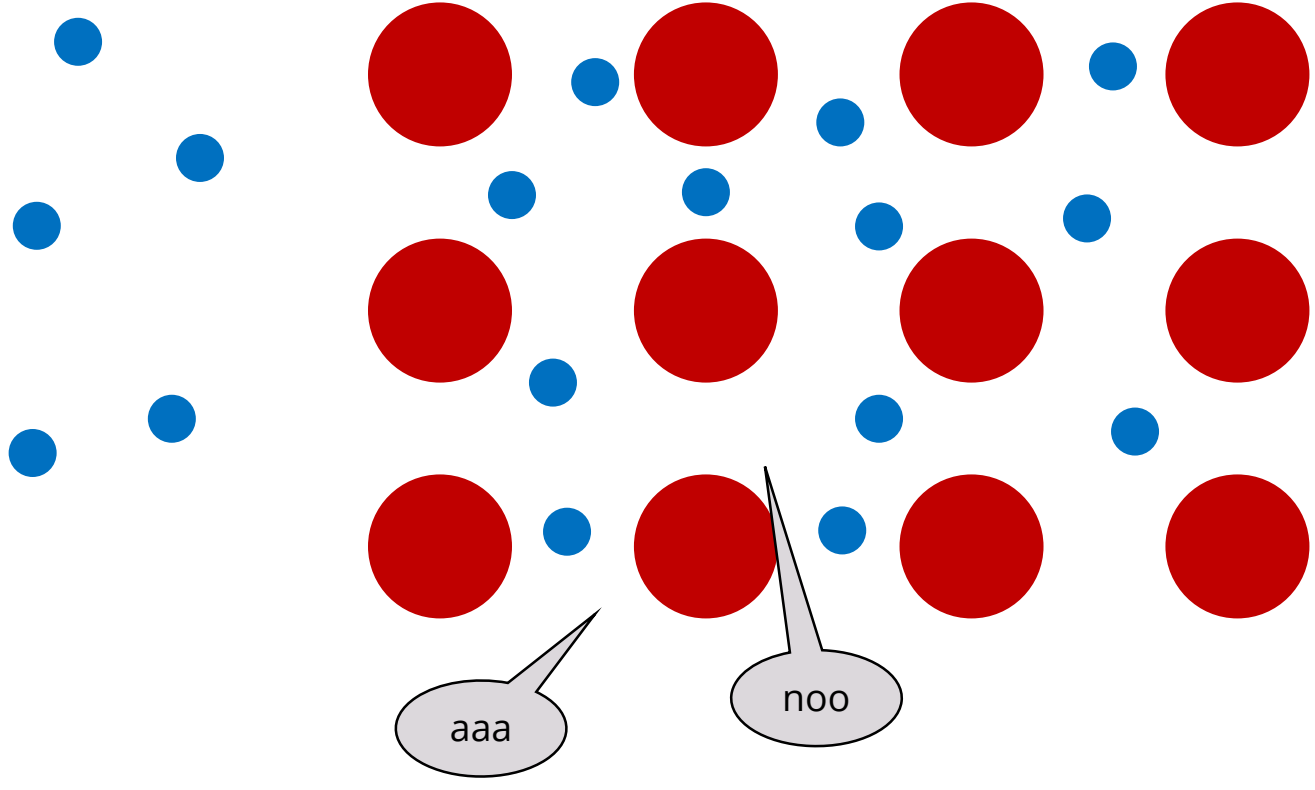


right now, the charges are **balanced:** same number of positive and negative

let's knock it out of whack

Two moles is company, three's a crowd

let's shove more electrons in



packing more electrons in leads to two things:

1. this metal is now **negatively charged**

2. the electrons are now **closer together** meaning they're **less happy**

Carousel of unhappiness

if we shove some down a wire

we'd like to get off
mister bones wild ride

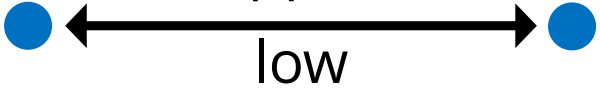
and then give them a
(difficult) way out

and then have something which
will **squeeze them back together**
again

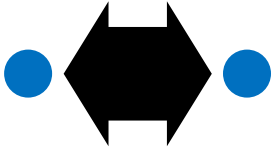
well now you have electricity

Watch em go

voltage measures electron unhappiness



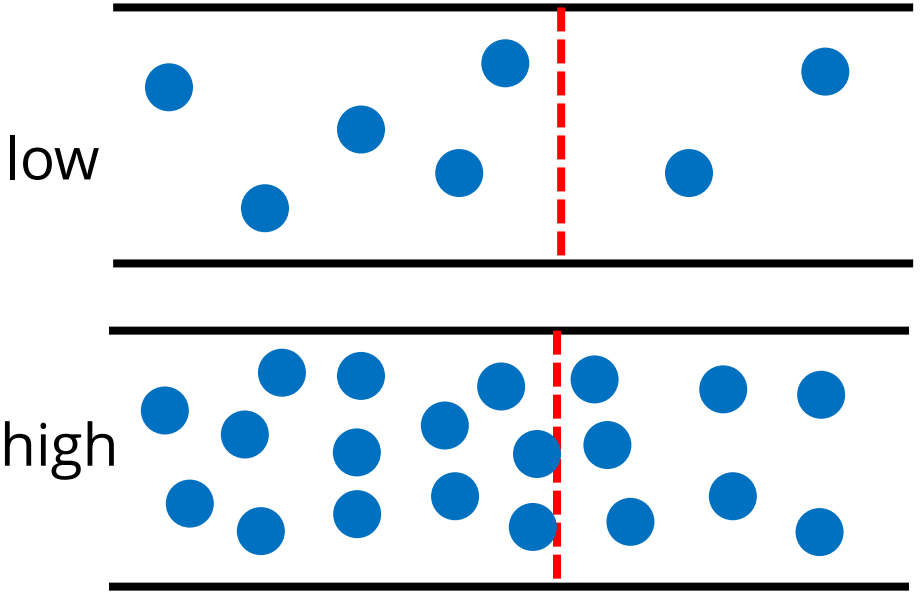
low



high

and that's electricity

current measures how many electrons per second are moving past a point

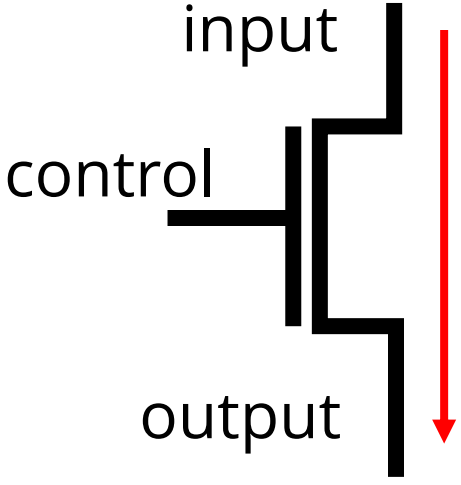


Logic Basics

Transistors n’at

Transistors

- A transistor is like a little valve, or switch
- The input, output, and control are all single bits
- The bits are represented as voltages (maybe 3.3V = **1**, 0V = **0**)

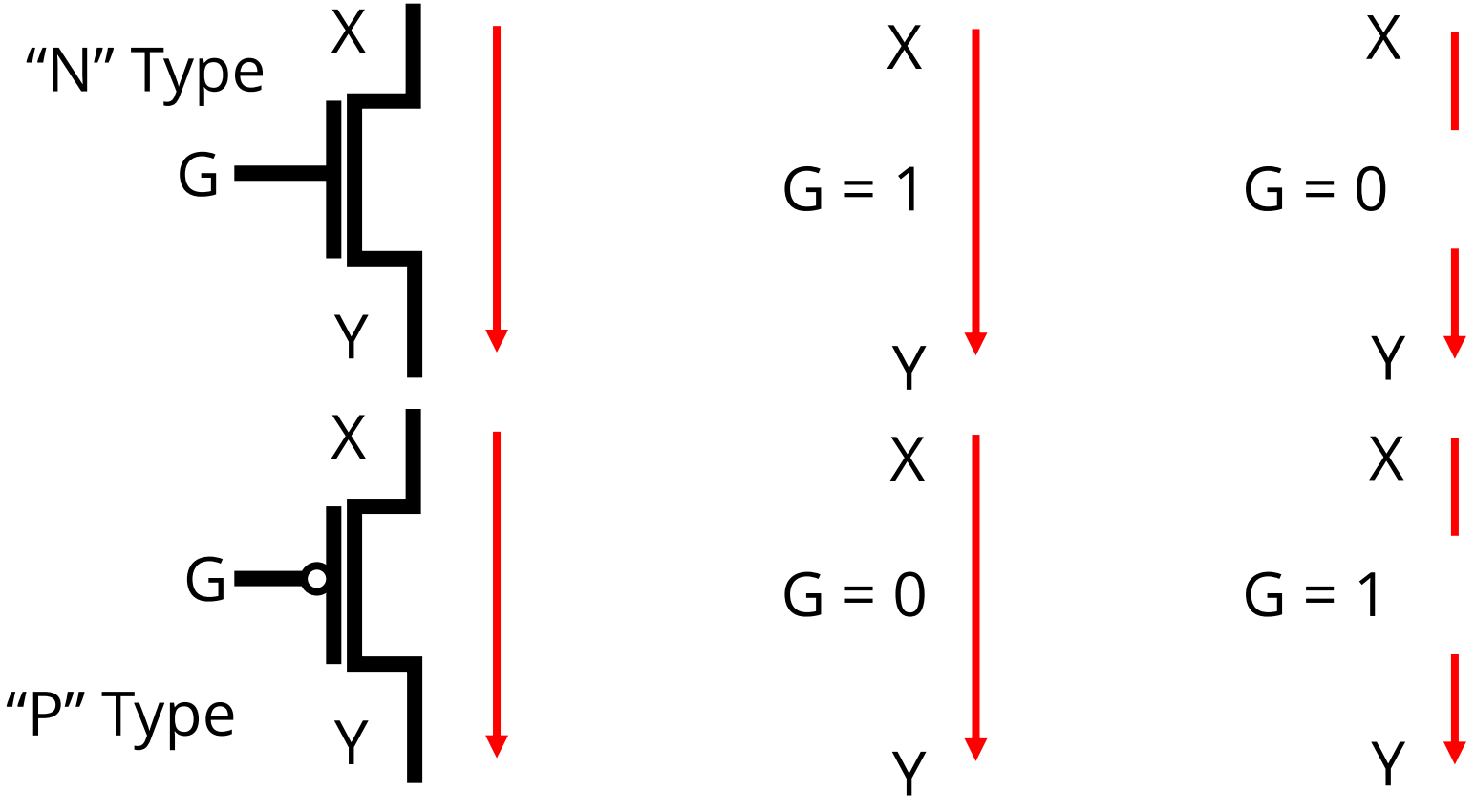


now just put 3 billion of them together! who said EE was hard?

it connects its input to its output if control is a **1**

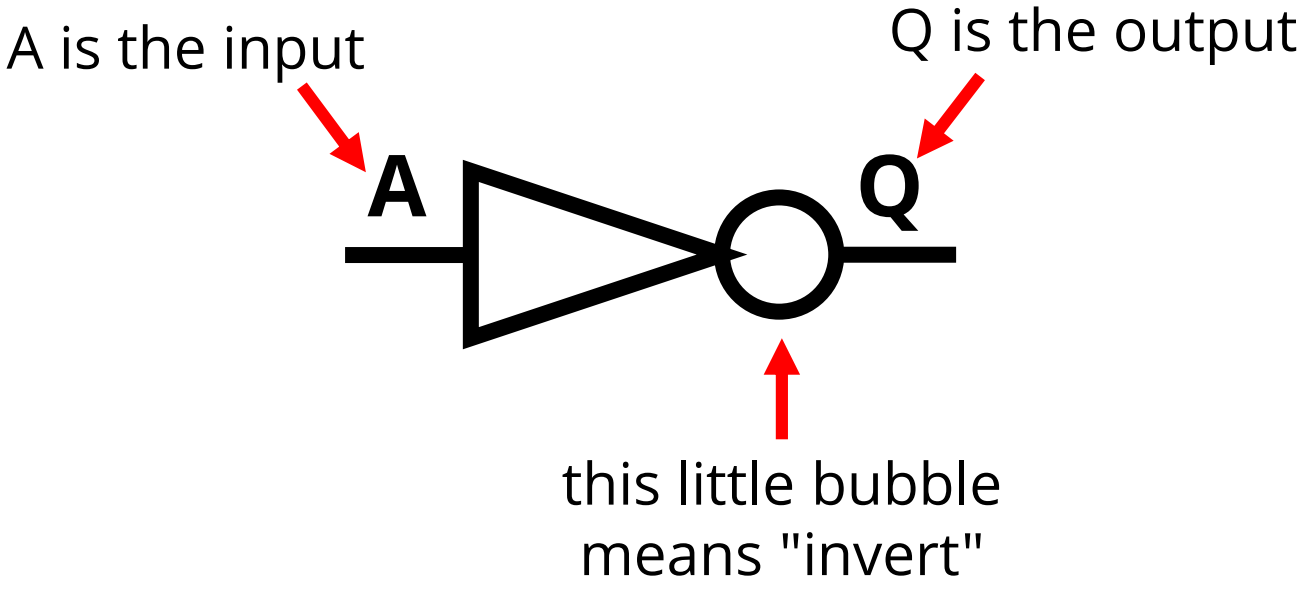
Two Types of MOS Transistor

- Metal Oxide Semiconductor transistors come in two varieties:



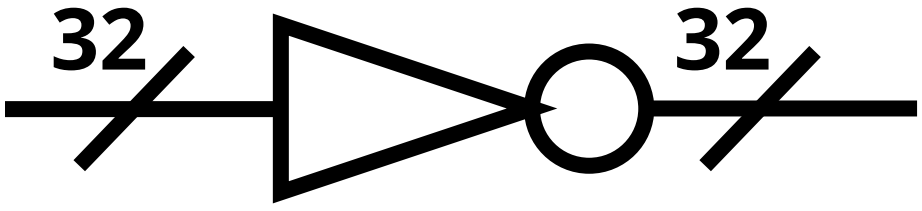
Gates

- We can combine transistors in interesting ways to make **gates**
- A **gate** implements one of the basic **boolean logic** functions
- Let's start with the simplest: the **NOT gate**



Time to bundle up

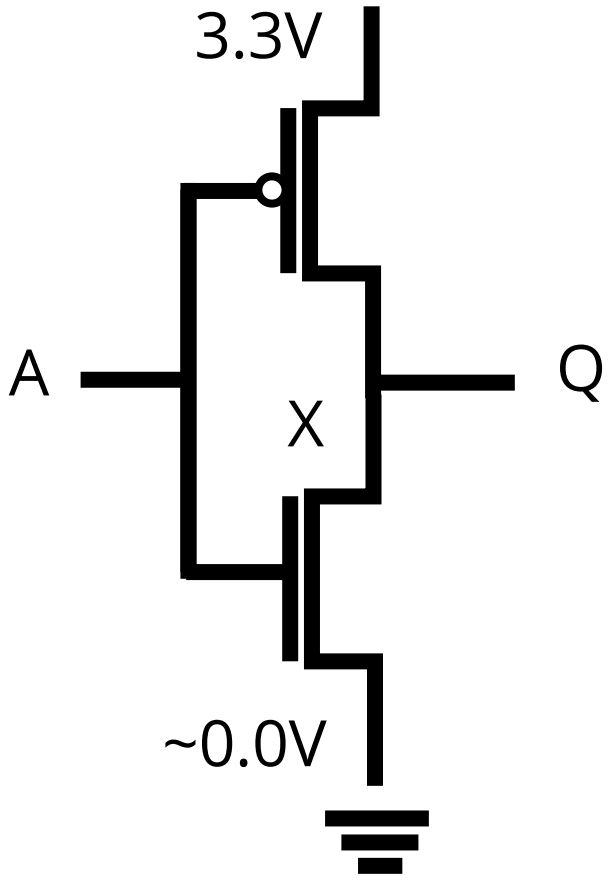
- If we want to, say, NOT a 32-bit value, we can draw it like:



- It's a lot nicer than drawing 32 wires with 32 NOT gates
- Logisim calls these **wire bundles**
  - It doesn't draw the slashes and numbers though...

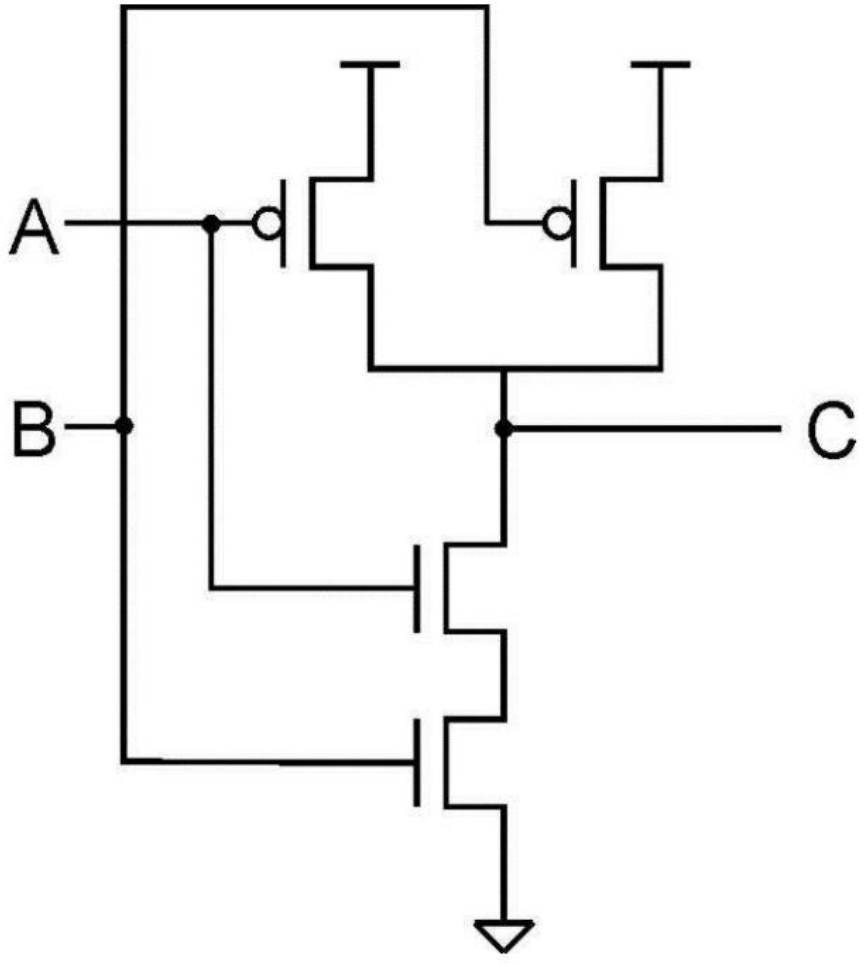
NOT (With MOS Transistors)

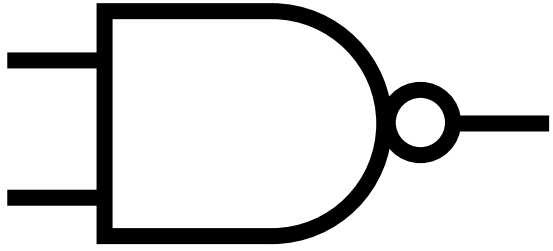
- AKA an “Inverter”
- Input value “A” manipulates transistors and output “Q”
- Voltage goes to ground ( ) unless it goes to Q
- “high” voltage == “1"



A	Q
0	1
1	0

NAND (MOS)



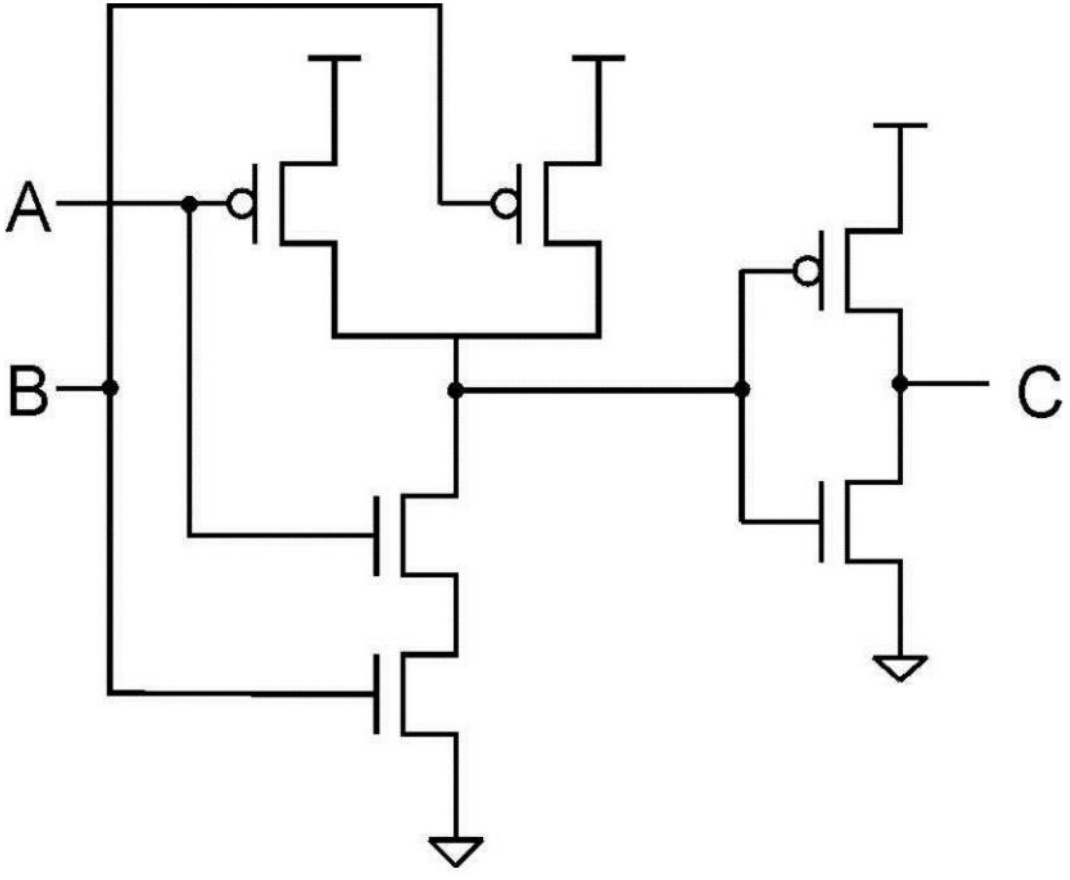


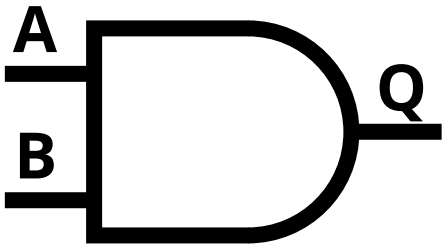
NAND gate

A	B	Q
0	0	1
0	1	1
1	0	1
1	1	0

$\overline{AB}$

AND (MOS)



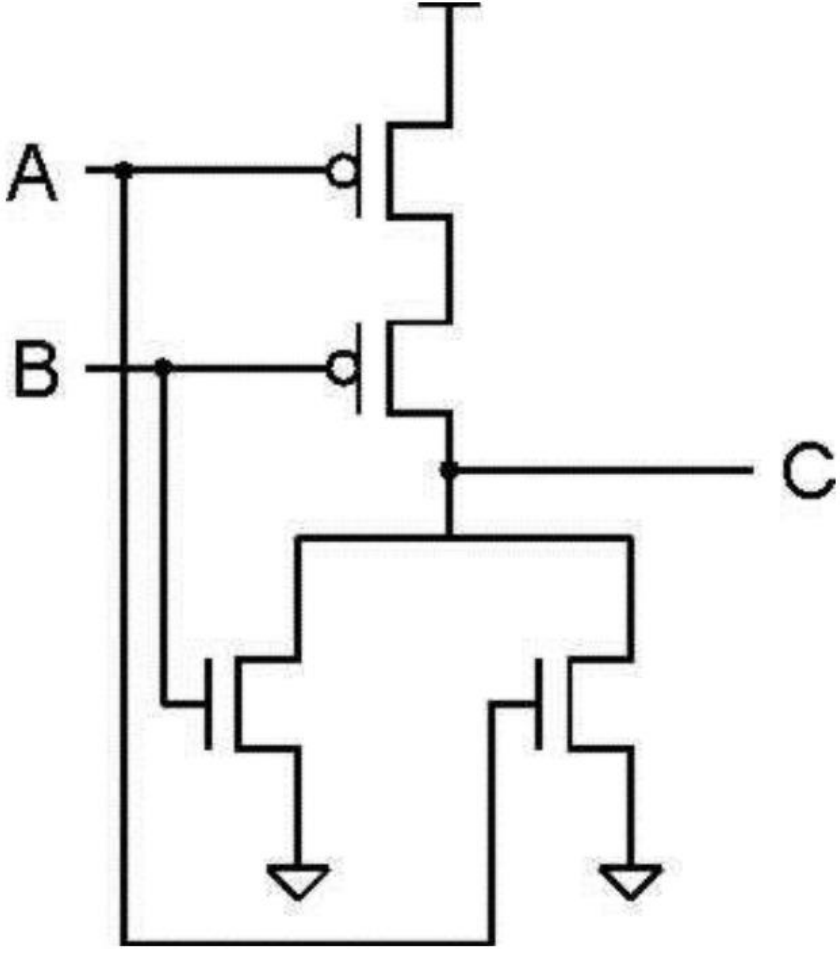


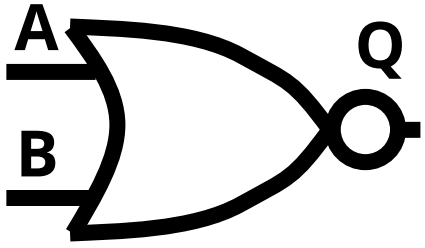
AND gate

A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

AB

NOR (MOS)



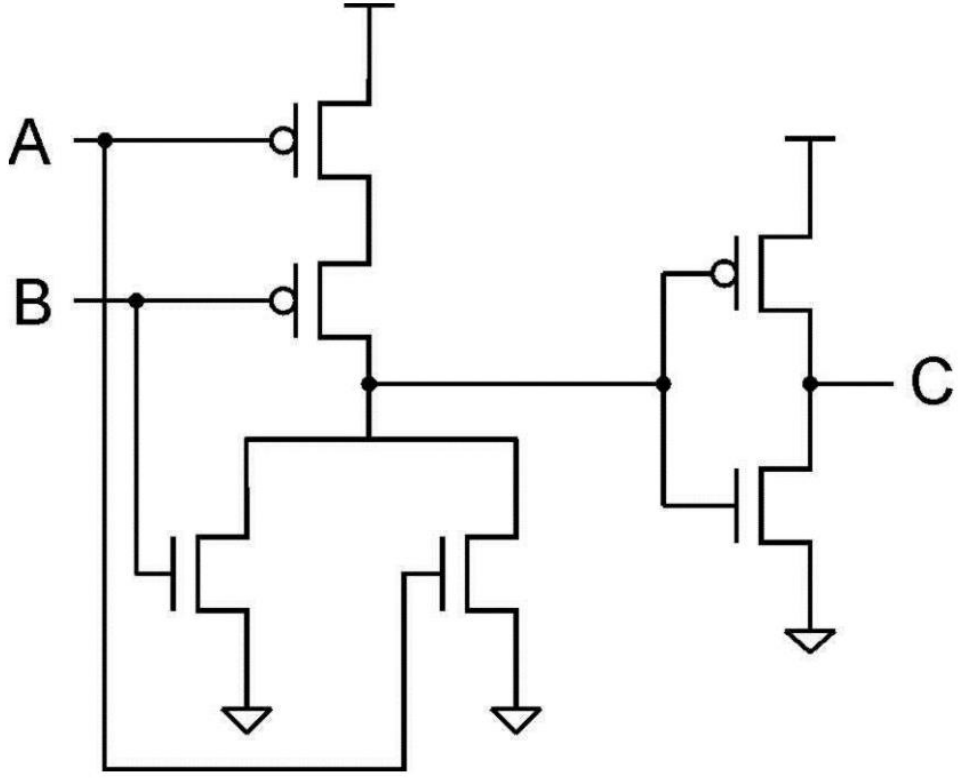


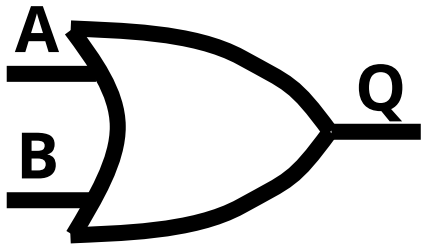
NOR gate

A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

$\overline{A+B}$

OR (MOS)





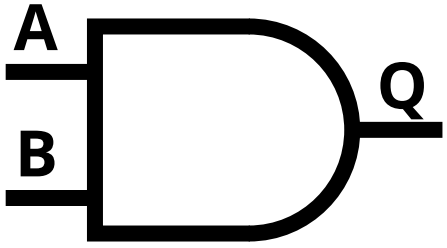
OR gate

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

A+B

AND, OR, and... XOR?

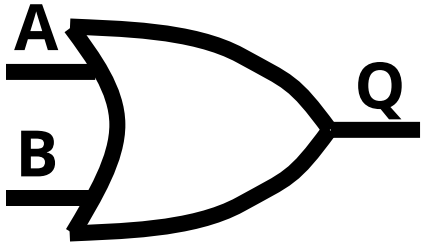
- we know about AND and OR, but what's XOR?



AND gate

A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

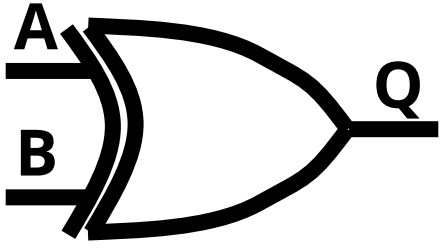
AB



OR gate

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

A+B



XOR gate

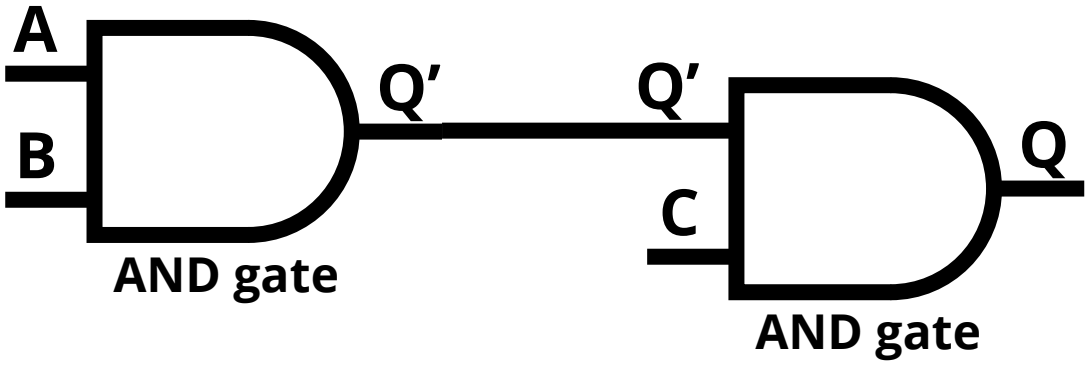
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

AxB

eXclusive **OR** means "one or the other, but NOT BOTH."

AND (multiple inputs)

- If you have more than 2 inputs…
  you can just concatenate:



A	B	C	Q
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	1

AND and/or OR

- Combining logic is just a matter of combining logic gates:

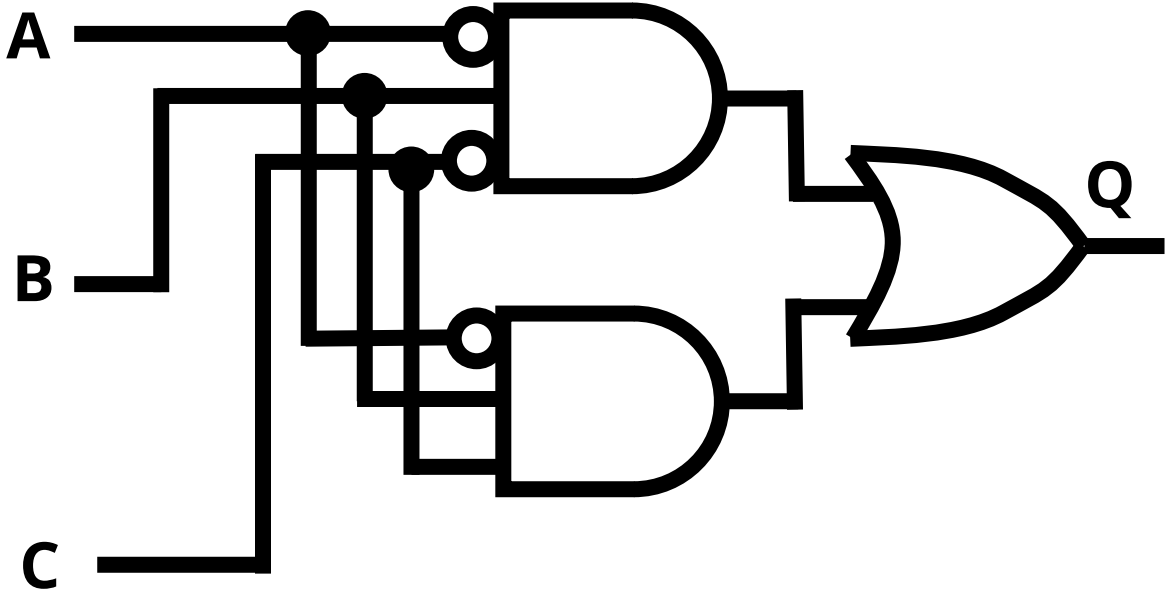
- Look at when Q is “1”
- OR each of those by the AND of each term:

$$\overline{A}B\overline{C} + \overline{A}BC$$

A	B	C	Q
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	1
1	1	1	0

AND and/or OR

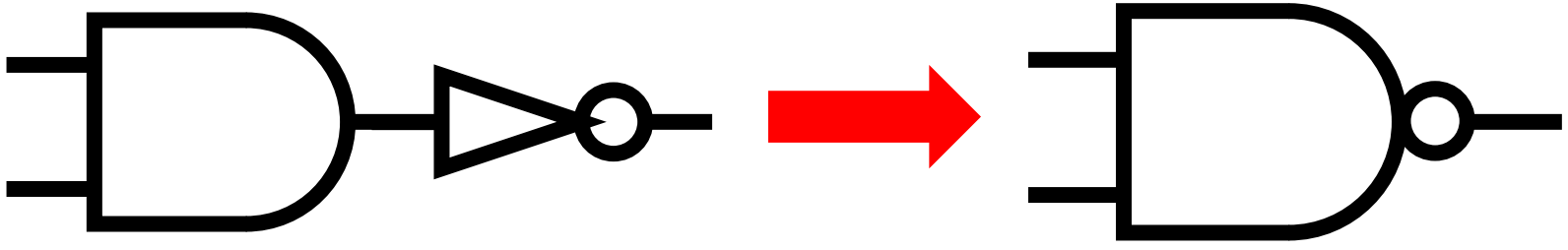
$$\overline{A}B\overline{C} + \overline{A}BC$$



A	B	C	Q
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	1
1	1	1	0

If you give an electrical engineer a NAND gate...

- Sum up: If a NOT gate is after an AND gate, you get a NAND gate:



- This kind of gate has a cool property: it's **universal**
  - In other words, you can build **an entire computer** with NANDs (or NORs)
  - Think how to build a NOT from a NAND (Then build an AND, etc)
- But this isn't how real circuits are designed, at least not anymore
- Digital logic courses use them cause **NAND gate chips are cheap**
- But in Logisim, we have infinite gates for free :D
- **Use the kind of gate you need for the situation at hand**