Boolean Algebra

CS 2130: Computer Systems and Organization 1

Xinyao Yi Ph.D.
Assistant Professor

UNIVERSITY of VIRGINIA | ENGINEERING

Announcement

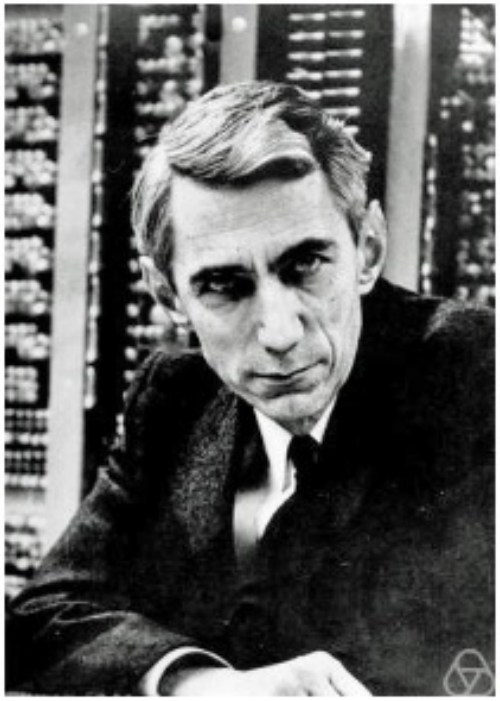
If you need to switch labs:

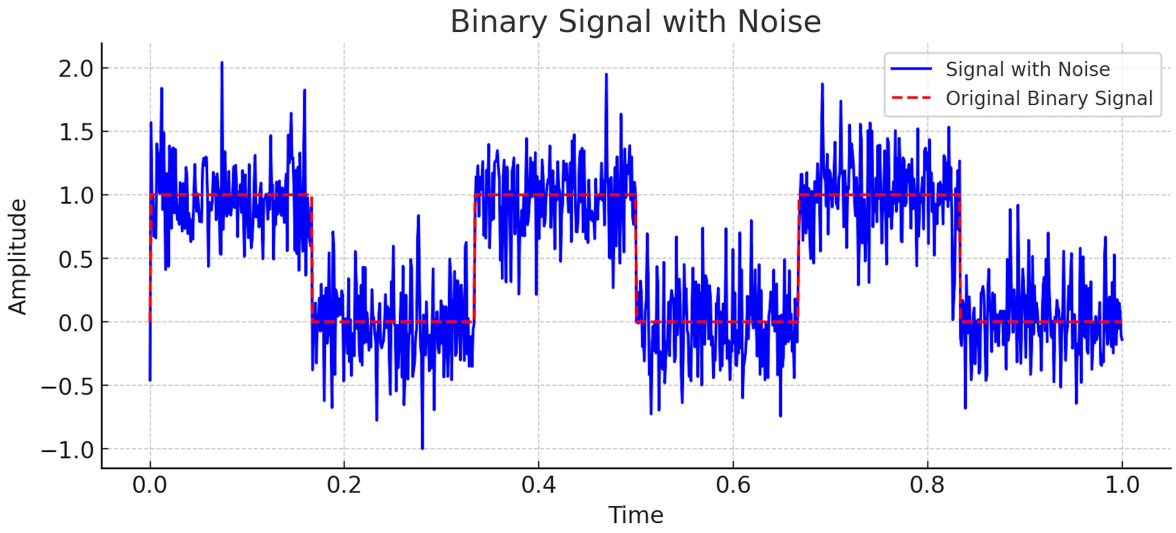
- Most lab sections still have available spots.
- Send me an email if you are unable to swap

Why only 0 and 1?

Shannon's key insight was this:
The biggest enemy of communication is noise.

Claude Shannon (the father of information theory).





Why only 0 and 1?

So yes, more levels give more information per symbol, but fewer symbols per second.

Shannon did the math and proved:
👉 *No matter how many levels you use, the total information rate stays the same.*

So the smartest choice is the simplest one: only two levels—0 and 1.
With only two, we can switch very fast between them, and even with noise, we always know which one we're at."

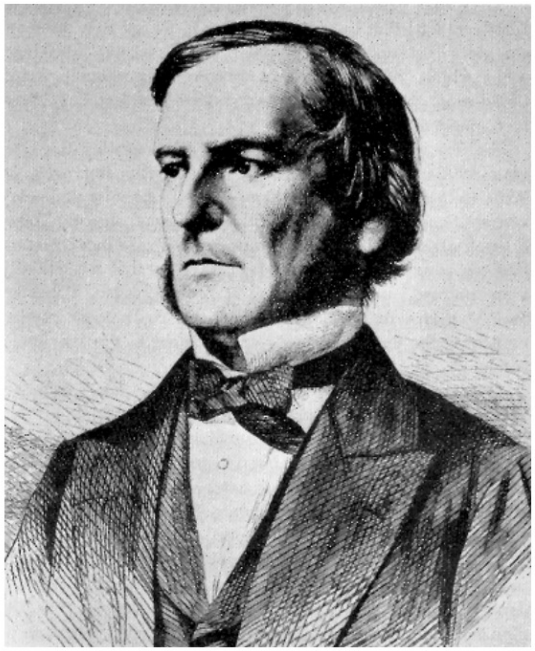
Vocabulary

- bit – either a 0 or 1

- binary - a system that has only two positions
- trinary - a system that has only three positions
- quadrinary - a system that has only four positions
- ...
- decinary - ...
- decimal - system that has ten positions

Boolean Algebra

George Boole



In Boolean Algebra, we live in a world with only two values:

- **True or False**
- **Yes or No**
- **1 or 0**

Boole showed that you could build an entire algebraic system using only these two values.

And that system uses three basic operations: **AND, OR, and NOT**."

Putting Them Together

Overall idea:

- Only need two things (Shannon)
- We can do math with two things (Boole)

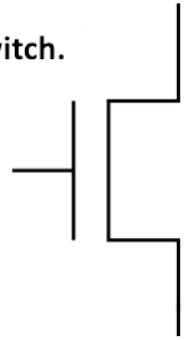
Now we need a physical device that deals in two levels

Transistors

Electricity (conceptually) - involves flow of electrons or other charged carriers through a conductive material

- current - rate of flow
- voltage - pressure of flow

You can think of a transistor as an electronically controlled switch.
If there's no voltage applied → no current flows → that's a 0.
If we apply voltage → current flows → that's a 1.
So voltage is the 'trigger,' current is the 'flow.'

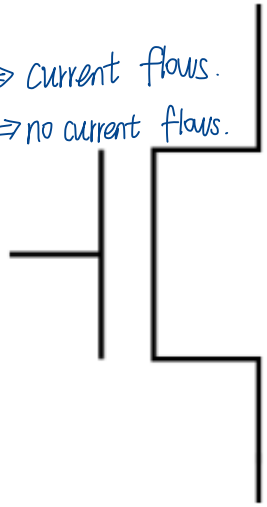


Transistors act like an electrically-triggered switch

- No voltage, no current
- Apply voltage to allow current to flow
- The amount of voltage needed to open the gate is boundary between 0 and 1
- Central technique for how we are going to build binary computers

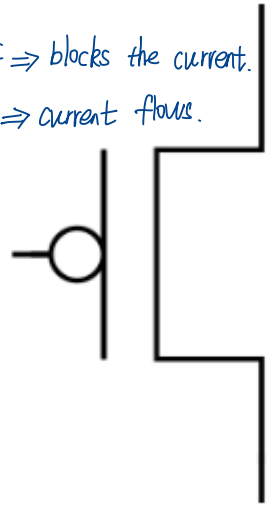
Transistors

n-type transistor
voltage ⇒ switches on ⇒ current flows.
no voltage ⇒ switches off ⇒ no current flows.



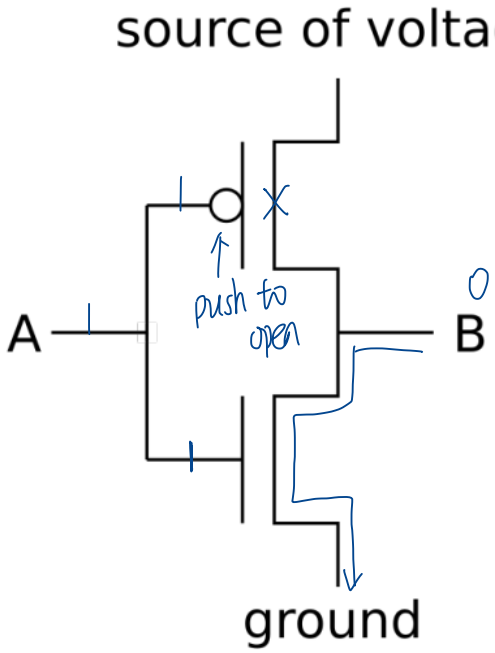
Push to close

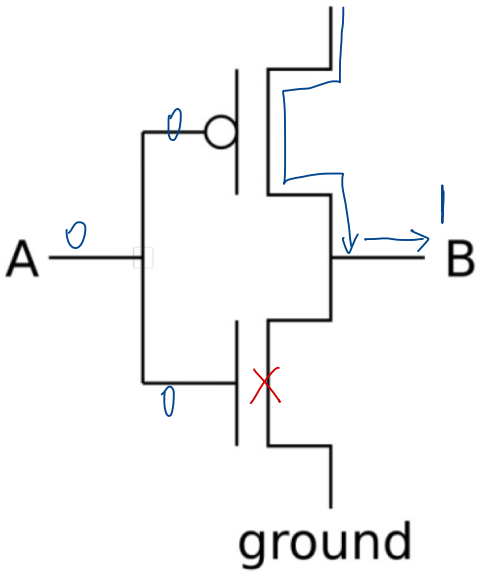
p-type transistor
voltage ⇒ switches off ⇒ blocks the current.
no voltage ⇒ stays on ⇒ current flows.



Push to open

Circuit Diagram





Truth Table

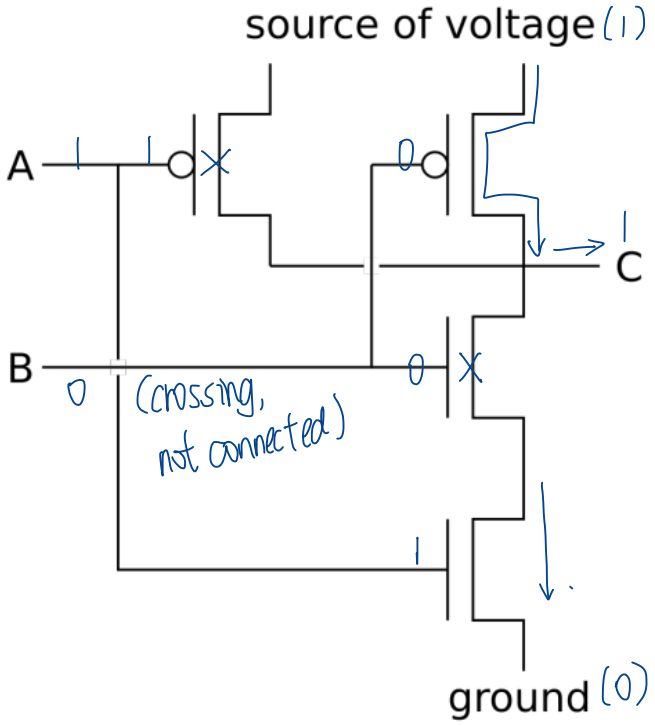
A	B
0	1
1	0

No voltage on it. ←

What is this truth table for?

Not gate.

Circuit Diagram



Given values of A and B, what is C?

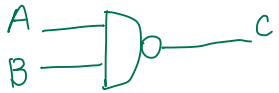
A	B	C
0	0	1
0	1	1
1	0	1
1	1	0

A&B
0
0
0
1

like A&B, but the opposite.

NAND gate

take "AND", then opposite



Other Gates

Reading: https://uva-cs.github.io/cso1-s26/readings/bool.html

Building Up

Where we are now

- World with only 2 states: 0 and 1
- Re-developed Boolean logic: and, or, not

Gives us everything Boole talked about

- We can do a lot of interesting things!
- Next: build higher level ideas: the trinary operator

Trinary Operator

General idea

```
if ( ... ) {
    ...
} else {
    ...
}
```

Trinary operator (expression if)

```
if(a) {
    x = b;
} else {
    x = c;
}
```

Python:

```
x=b if a else c
```

Java:

```
x=a?b:c
```

Multiplexer (mux)

How can we build a mux out of what we have learned so far?

x=a ? b : c

Multiplexer (mux)

How can we build a mux out of what we have learned so far?

x=a ? b : c

Can be built from and, or, and not

- Can be built using transistors
- Can physically put it in silicon!

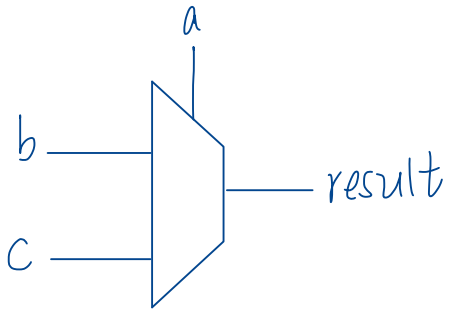
Mux will be the key when constructing a computer out of gates and circuits!

Multiplexer (mux)

ternary operator

x=a ? b : c

A multiplexer (mux) is commonly drawn as a trapezoid in circuit diagrams.




```
if(a){
    b
}else{
    c
}
```

a	b	c	result
0			c (depends on c)
1			b (depends on b)

a, b, c could be anything, they could be strings, numbers, functions.....