Logic Gates, Mux, Binary Arithmetic

CS 2130: Computer Systems and Organization 1
September 1, 2025

Announcements

- Lab 1 tomorrow!

Putting it together

Overall idea:

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

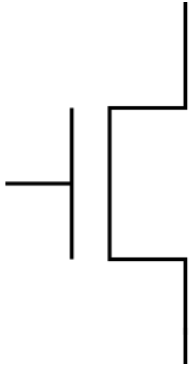
Putting it together

Overall idea:

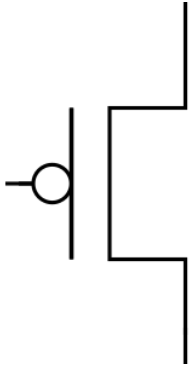
- Only need two things (Shannon)
- We can do math with two things (Boole)

Now we need a physical device that deals in two levels

Transistors

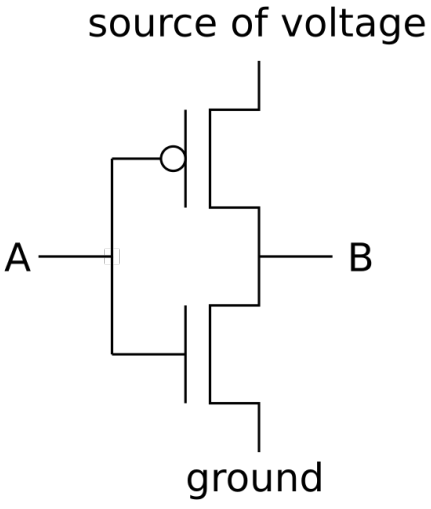


push to close

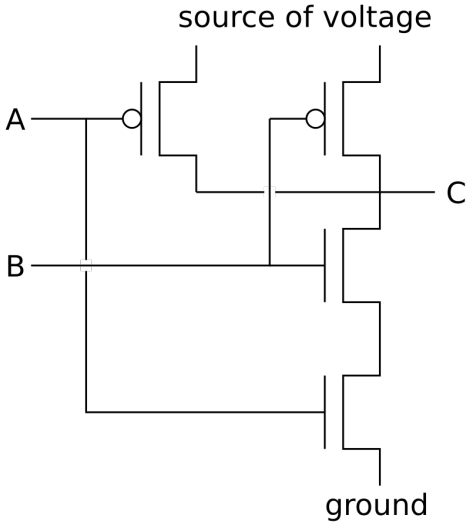


push to open

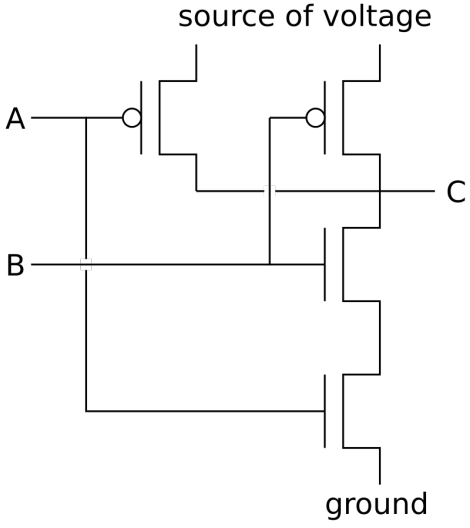
Circuit Diagram



Circuit Diagram



Circuit Diagram



Other Gates (reading)

Building Up

Where we are now

- World with only 2 states: 0 and 1
- Re-developed Boolean logic (gates):
  - `and`, `or`, `not`
  - `nand`, `nor`, `xor`

Gives us everything Boole talked about

- Next: build higher level ideas, something powerful!

Trinary Operator

this will be key when we are constructing our computer out of these gates and circuits

Trinary Operator

- General idea:

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

this will be key when we are constructing our computer out of these gates and circuits

Trinary Operator

- Python: `x = b if a else c`

this will be key when we are constructing our computer out of these gates and circuits

Trinary Operator

- Python: `x = b if a else c`
- Java: `x = a ? b : c`

this will be key when we are constructing our computer out of these gates and circuits

Multiplexer (mux)

```
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)

Can be built from `and`, `or`, and `not`

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

Questions?

More bits!

2-bit Multiplexer (mux)

2-bit values instead of 1-bit values

Multi-bit Values

- So far, only talking about 2 things
- Numbers, strings, objects, ...

Numbers

From our oldest cultures, how do we mark numbers?

Numbers

From our oldest cultures, how do we mark numbers?

- **unary** representation: make marks, one per "thing"

Numbers

From our oldest cultures, how do we mark numbers?

- **unary** representation: make marks, one per "thing"
  - Awkward for large numbers, ex: CS 2130?
  - Hard to tell how many marks there are

Numbers

From our oldest cultures, how do we mark numbers?

- **unary** representation: make marks, one per "thing"
  - Awkward for large numbers, ex: CS 2130?
  - Hard to tell how many marks there are
- Update: group them!

Numbers

From our oldest cultures, how do we mark numbers?

- **unary** representation: make marks, one per "thing"
  - Awkward for large numbers, ex: CS 2130?
  - Hard to tell how many marks there are
- Update: group them!
- Romans used new symbols:

Numbers

From our oldest cultures, how do we mark numbers?

- Arabic numerals
  - Positional numbering system

Numbers

From our oldest cultures, how do we mark numbers?

- Arabic numerals
  - Positional numbering system
  - The 10 is significant:
    - 10 symbols, using 10 as base of exponent

Numbers

From our oldest cultures, how do we mark numbers?

- Arabic numerals
  - Positional numbering system
  - The 10 is significant:
    - 10 symbols, using 10 as base of exponent
  - The 10 is *arbitrary*
  - We can use other bases! $\pi, 2130, 2, \ldots$

Base-8 Example

Try to turn 134_8 into base-10:

Bases

We will discuss a few in this class

- Base-10 (decimal) - talking to humans
- Base-8 (octal) - shows up occasionally
- Base-2 (binary) - most important! (we've been discussing 2 things!)
- Base-16 (hexadecimal) - nice grouping of bits

Binary

2 digits: 0, 1

Try to turn 1100101_2 into base-10:

Binary

Any downsides to binary?

Turn 2130_{10} into base-2:
hint: find largest power of 2 and subtract

Long Numbers

How do we deal with numbers too long to read?

Long Numbers

How do we deal with numbers too long to read?

- Group them by 3 (right to left)

Long Numbers

How do we deal with numbers too long to read?

- Group them by 3 (right to left)
- In decimal, use commas: ,
- Numbers between commas: 000 - 999

Long Numbers

How do we deal with numbers too long to read?

- Group them by 3 (right to left)
- In decimal, use commas: ,
- Numbers between commas: 000 - 999
- Effectively base-1000

Long Numbers in Binary - Readability

- Typical to group by 3 or 4 bits
- No need for commas *Why?*

100001010010

Long Numbers in Binary - Readability

- Typical to group by 3 or 4 bits
- No need for commas *Why?*
- We can use a separate symbol per group
- How many do we need for groups of 3?

100001010010

Long Numbers in Binary - Readability

- Typical to group by 3 or 4 bits
- No need for commas *Why?*
- We can use a separate symbol per group
- How many do we need for groups of 3?
- Turn each group into decimal representation

100001010010

Long Numbers in Binary - Readability

- Typical to group by 3 or 4 bits
- No need for commas *Why?*
- We can use a separate symbol per group
- How many do we need for groups of 3?
- Turn each group into decimal representation
- Converts binary to **octal**

100001010010

Long Numbers in Binary - Readability

- Groups of 4 more common
- How many symbols do we need for groups of 4?

100001010010

Long Numbers in Binary - Readability

- Groups of 4 more common
- How many symbols do we need for groups of 4?
- Converts binary to **hexadecimal**
- Base-16 is very common in computing

100001010010

Hexadecimal

Need more than 10 digits. What next?

$$1110$$

Hexadecimal Exercise

Consider the following hexadecimal number:

852dab1e

Is it even or odd?

Using Different Bases in Code

	Old Languages	New Languages
binary		
octal		
decimal		
hexadecimal		