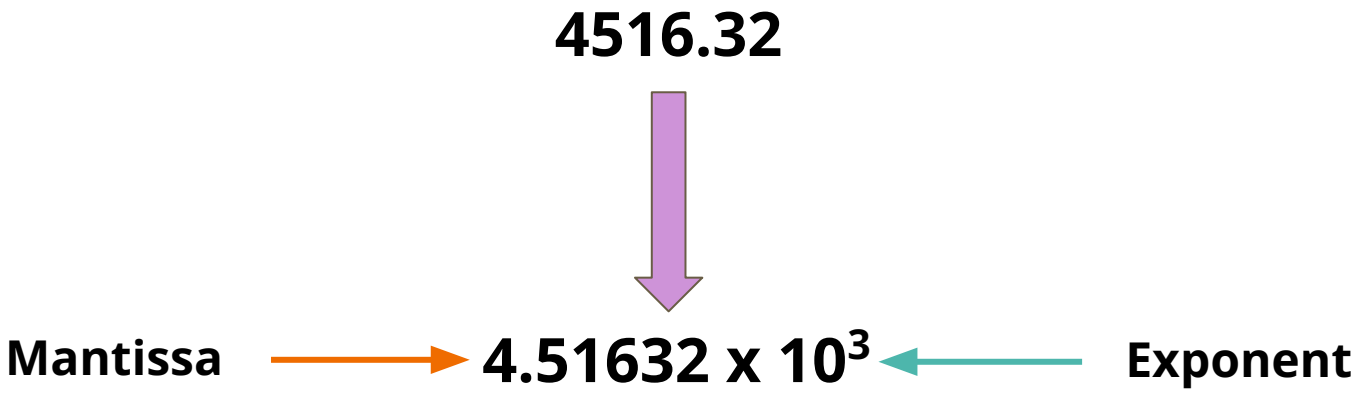
Bits of Architecture

Floating Point Numbers

How Do We Represent Represent Non-Integral Numbers?

Recall Scientific Notation



Floating Point Numbers

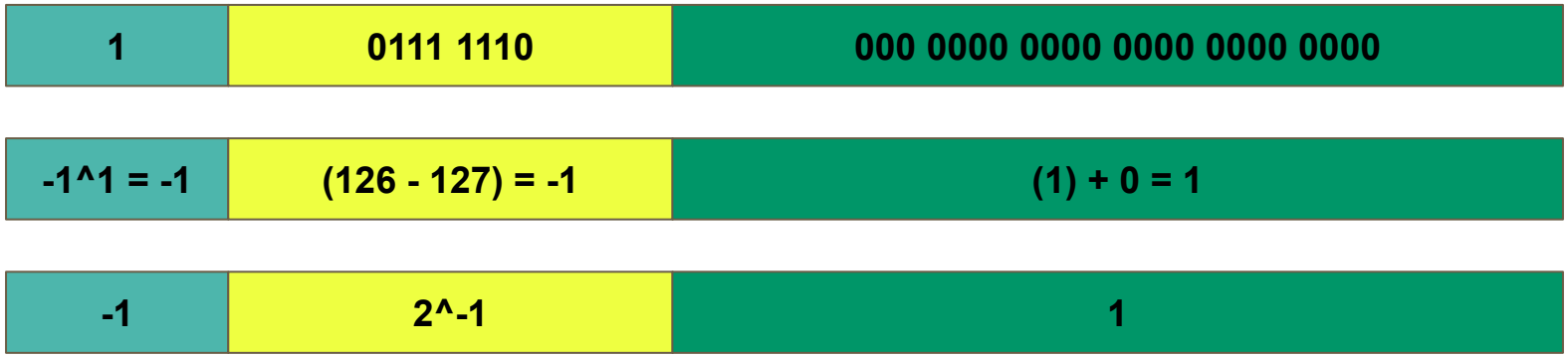
IEEE-754 Floating Point Standard

- Scientific notation in binary
- Encode non-integral numbers in the bits we have
  - 16b = Half Precision
  - 32b = Single Precision (Floats)
  - 64b = Double Precision (Doubles)
- Similar to instructions (break down bits into field)
  - Sign
  - Exponent
  - Mantissa

Fields for Floats (32b)

Example

Fields for Floats (32b)



-0.5_{10}

But Wait…

Fields for Floats (32b)

But why is this biased?

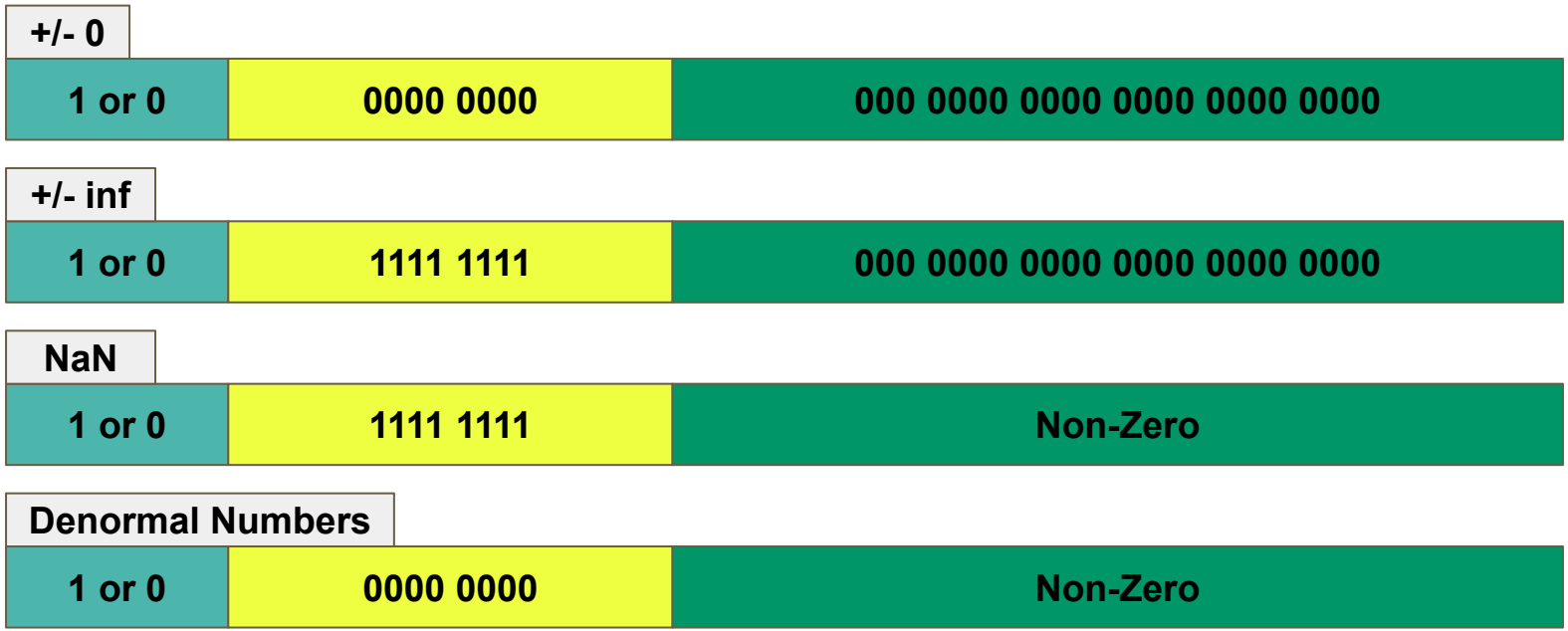
- Makes comparison easier
- Recall, negative numbers look like large unsigned numbers

Why is there a “1 +”?

- Our number is normalized
- No 0 before decimal point (like scientific notation)

Special Representations

Special Representations



Is IEEE-754 Floating Point the Only Way?

No!

- Fixed point
  - Not great, but possible
- Variations on IEEE Floating Point
  - E.g., bfloat
- Posits/Universal Numbers (unums)
  - Interesting research from John Gustafson c. 2015
  - Drop-in replacement for current floating point format