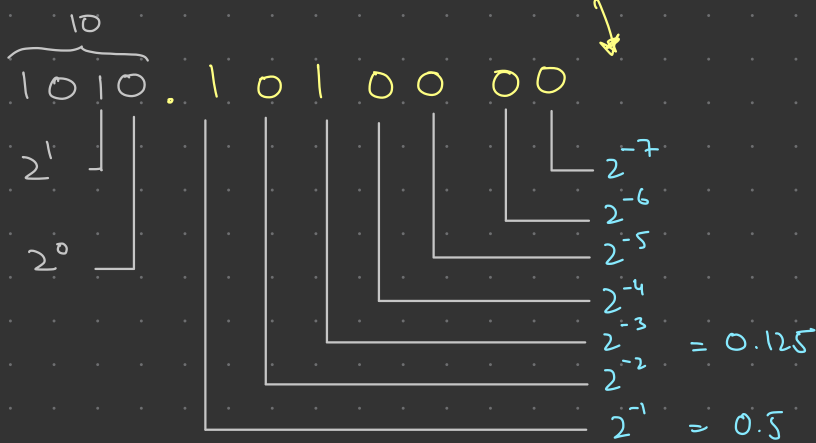
Floating Point Representation

1. Fractions

10.625



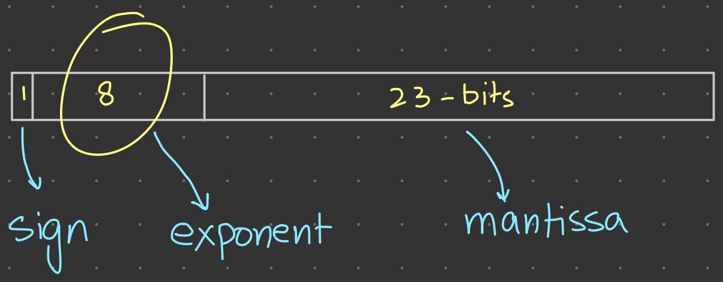
$= 0.5 + 0.125$

$= 0.625$

Issue: fixed "precision" versus "size".

Solution: Use a floating point

32-bits — IEEE 754 Standard



Double precision — (1, 11, 52) = 64-bits

First, "how".

Example: 19.59375

Step 1: Convert to binary

19	/ 2	=	9	rem	1
9	/ 2	=	4	"	1
4	/ 2	=	2	"	0
2	/ 2	=	1	"	0
1	/ 2	=	0	"	1

stopping point

0.59375	x 2	= 1.1875	1
0.1875	x 2	= 0.375	0
0.375	x 2	= 0.75	0
0.75	x 2	= 1.5	1
0.5	x 2	= 1.0	1

Stopping point.

Step 2: Find Exponent (Biased)

left-most 1

1 0 0 1 1 . 1 0 0 1 1

Exponent = 4 , Biased exponent = 4 + 127 ⟶ $(2^8 - 1)$ exp bits

= $(131)_{10}$

= 10000011

Step 3: Find Mantissa

1 0 0 1 1 . 1 0 0 1 1

Mantissa is the "remainder" ⇝

0	10000011	0011.10011 ——————

Example 2: 15.0

Step 1 :

$15/2 = 7$ rem 1
$7/2 = 3$ " 1
$3/2 = 1$ " 1
$1/2 = 0$ " 1

1.111.000000

Step 2: $3 + 127 = (130)_{10} \rightarrow 10000010$

Step 3 : 0 10000010 111.0000 ————

Reverse Operation:

Step 1: Add leading 1 to matissa

1 1 1 1 0 0 0 0 0 0 0 0 ————

Step 2: Find Exponent

10000010 → 130 − 127 = 3

jump 3

1 1 1 1 0 0 0 0 0 0 0 0 ————

1111.000000

15.0

Next Largest Possible Number:

15.0

0 10000010 1110 0000000000000000000000

0 10000010 1110 0000000000000000000001

$\frac{1}{2}$ $\frac{1}{4}$ $\frac{1}{8}$ $\frac{1}{16}$ $\frac{1}{32}$ $\frac{1}{64}$ $\frac{1}{128}$ $\frac{1}{256}$ $\frac{1}{512}$ ...

Exponent = 3

15.000000119209

So: 15.000000119207

=

15.000000119205

=

15.0000000000000

Let's see some code...