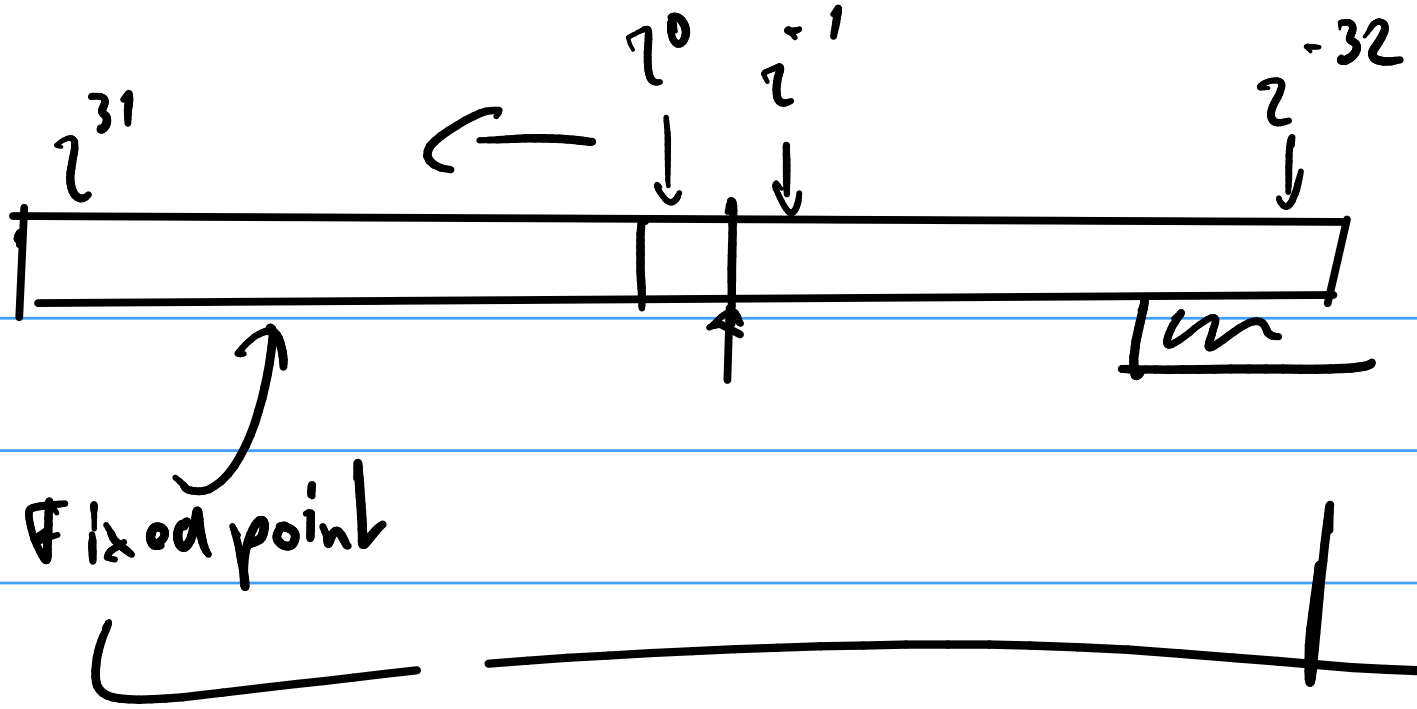
# Overview

- Floating point

→ Example 2

→ IEEE




Flaw: Uneven amount of accuracy between small and large numbers

$1 \leq$ (<2)

(Number) = $1.\underbrace{11111\ldots1}_{\text{significand}} \cdot 2^{\overbrace{\text{exponent}}}$

$2 \cdot 2^{\text{exponent}}$

$= 1 \cdot 2^{\text{exponent}+1}$

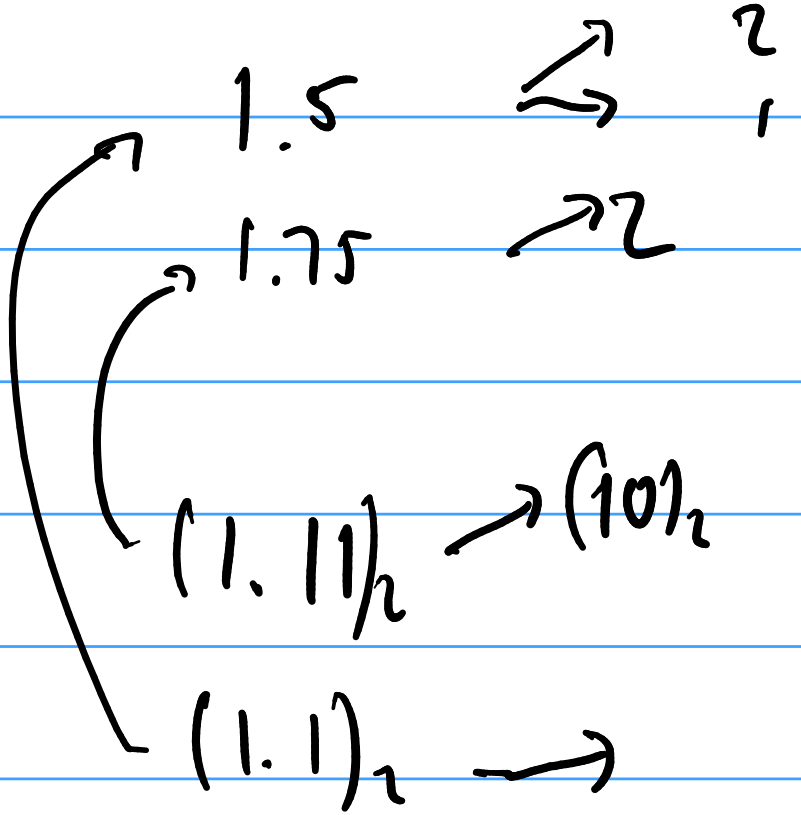
$123.125 = (1\,1\,1\,011.001)_2$

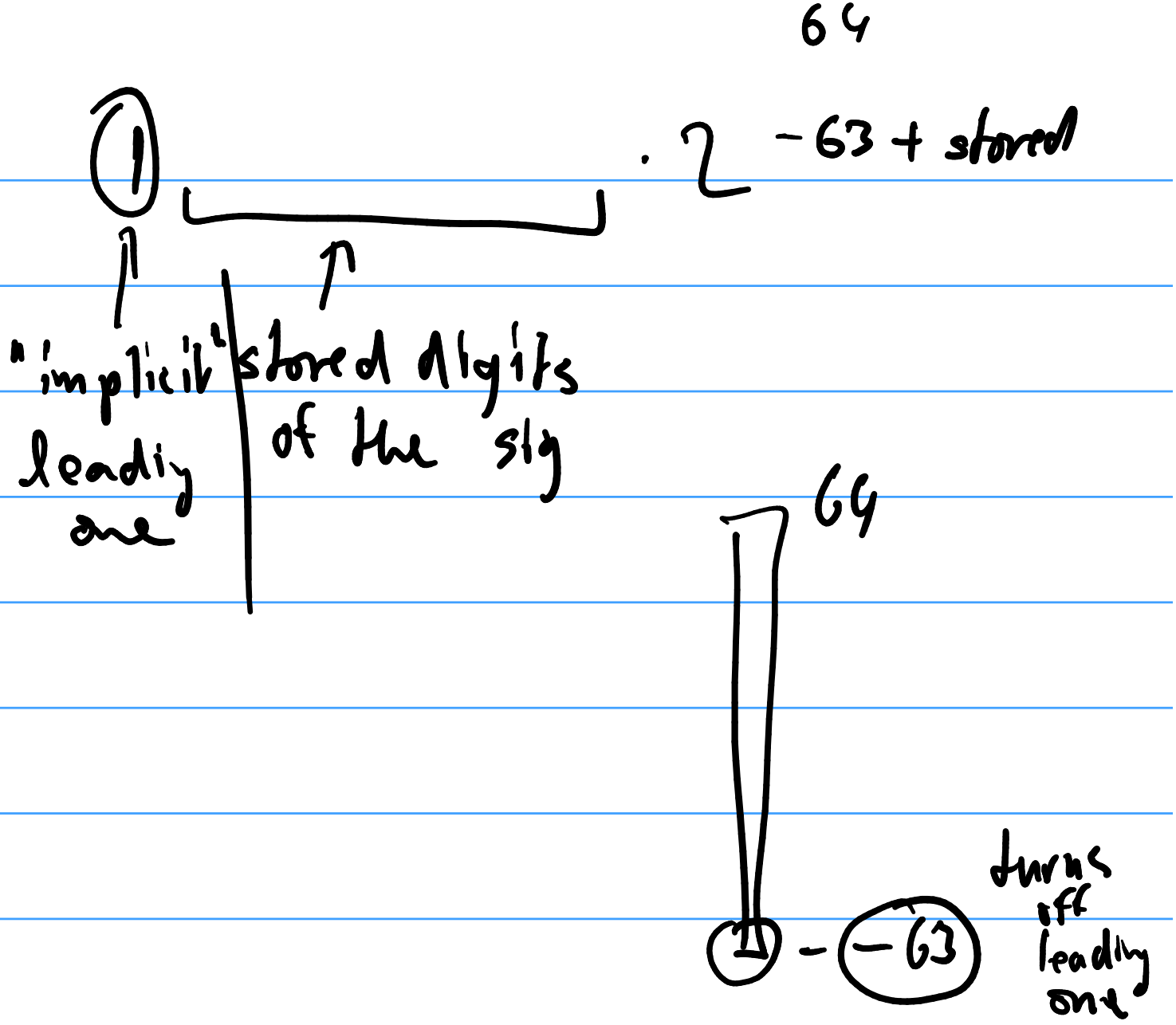
$1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2$

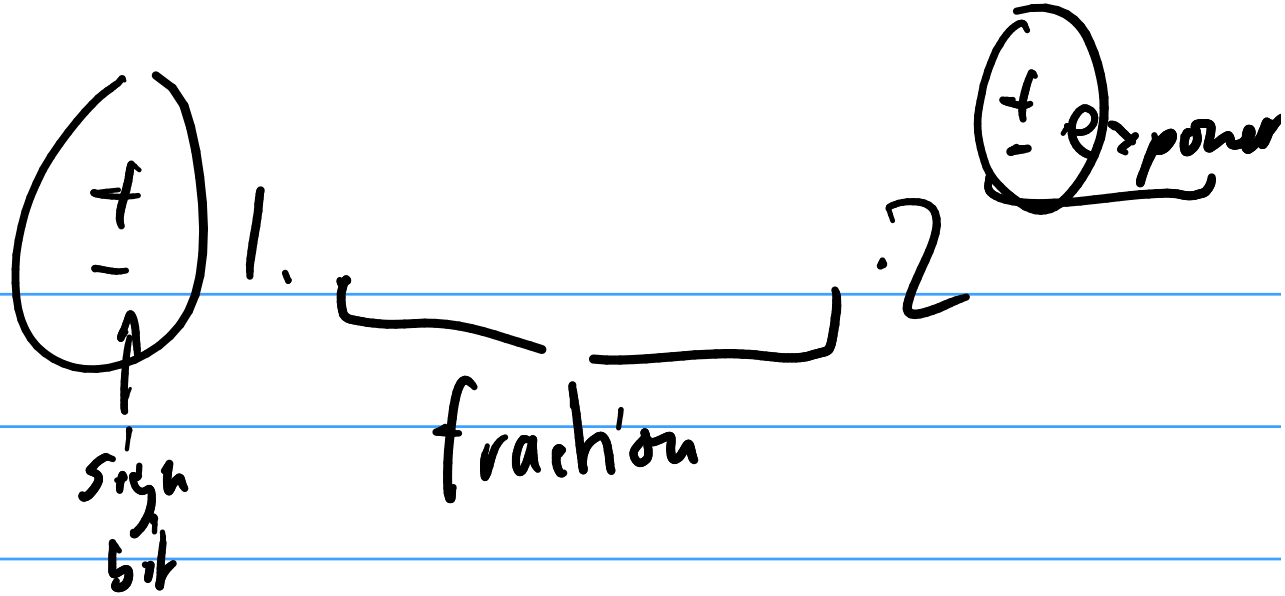
$2^6 = 64$

$$= (1.1110011001) \cdot 2^6$$

rounded




64
1
.2
-63 + stored
"implicit" leading one
stored digits of the sig
64
1
-63
turns off leading one

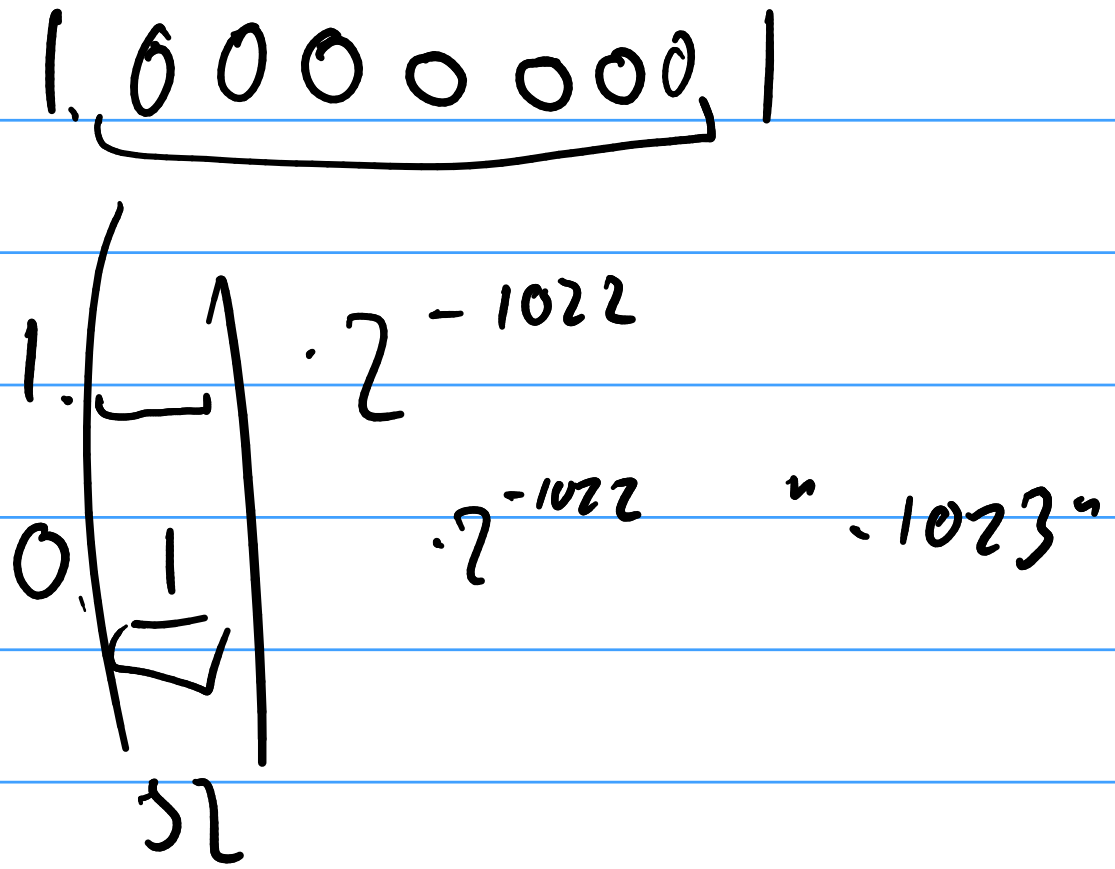



$$2^{-1022} = (1.\_\_\_\_\_)_2 \cdot 2^{-1022}$$

$$2^{-1023} = (0.1000\,0)_2 \cdot 2^{-1022}$$

$2^{-1}$

↑ subnormal numbers


1.0000000 1
1.
0.1
52
.2^-1022
.2^-1022
"-1023"

# Floating Point numbers

Convert $13 = (1101)_2$ into floating point representation.

What pieces do you need to store an FP number?

# **In-class activity:** Floating Point

# Unrepresentable numbers?

Can you think of a somewhat central number that we cannot represent as

$$x = (1.\_\_\_\_\_)_2 \cdot 2^{-p}?$$

# **Demo:** Picking apart a floating point number

# Subnormal Numbers

What is the smallest representable number in an FP system with 4 stored bits in the significand and an exponent range of $[-7, 7]$?

# Subnormal Numbers (II)

Why would you want to know about subnormals? Because computing with them is often slow, because it is implemented using 'FP assist', i.e. not in actual hardware. Many C compilers support options to 'flush subnormals to zero'.

**Demo:** Density of Floating Point Numbers

**Demo:** Floating Point vs. Program Logic

# Floating Point and Rounding Error

What is the relative error produced by working with floating point numbers?

What is smallest floating point number $> 1$? Assume 4 stored bits in the significand.

What's the smallest FP number $> 1024$ in that same system?

Can we give that number a name?