BSC~Microsoft Research Centre

Numbers Representaion
Floating point numbers

Dr. Tassadaq Hussain

Riphah International University

Real Numbers

- Two's complement representation deal with signed integer values only.
- Without modification, these formats are not useful in scientific or business applications that deal with real number values.
- Floating-point representation solves this problem.

Floating-Point Representation

- If we are clever programmers, we can perform floating-point calculations using any integer format.
- This is called *floating-point emulation*, because floating point values aren't stored as such; we just create programs that make it seem as if floating-point values are being used.
- Most of today's computers are equipped with specialized hardware that performs floating-point arithmetic with no special programming required.
  - —Not embedded processors!

Floating-Point Representation

- Floating-point numbers allow an arbitrary number of decimal places to the right of the decimal point.

For example: $0.5 \times 0.25 = 0.125$

- They are often expressed in scientific notation.

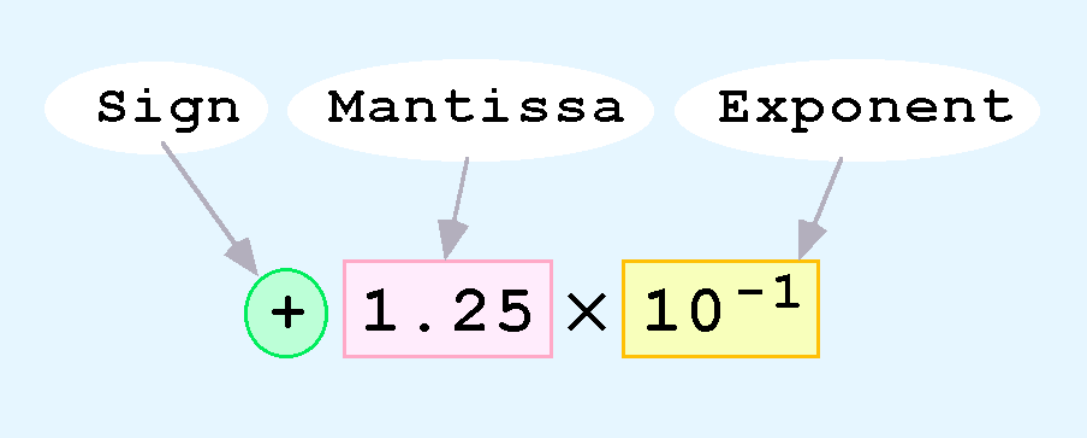
For example:

$$0.125 = 1.25 \times 10^{-1}$$

$$5{,}000{,}000 = 5.0 \times 10^{6}$$

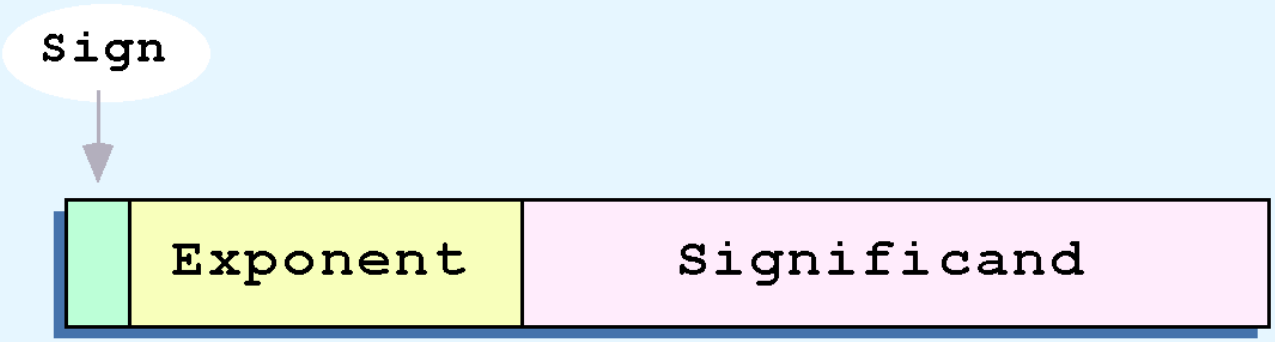
Floating-Point Representation

- Computers use a form of scientific notation for floating-point representation
- Numbers written in scientific notation have three components:



Floating-Point Representation

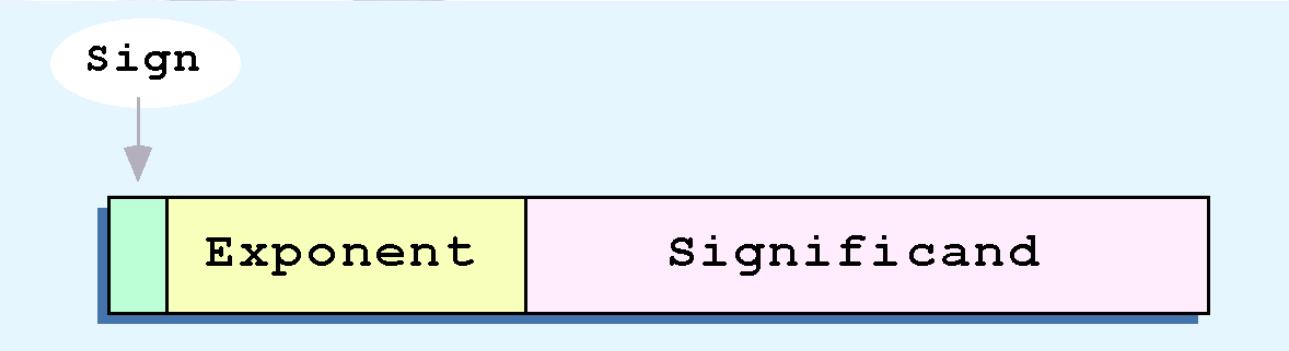
- Computer representation of a floating-point number consists of three fixed-size fields:



- This is the standard arrangement of these fields.

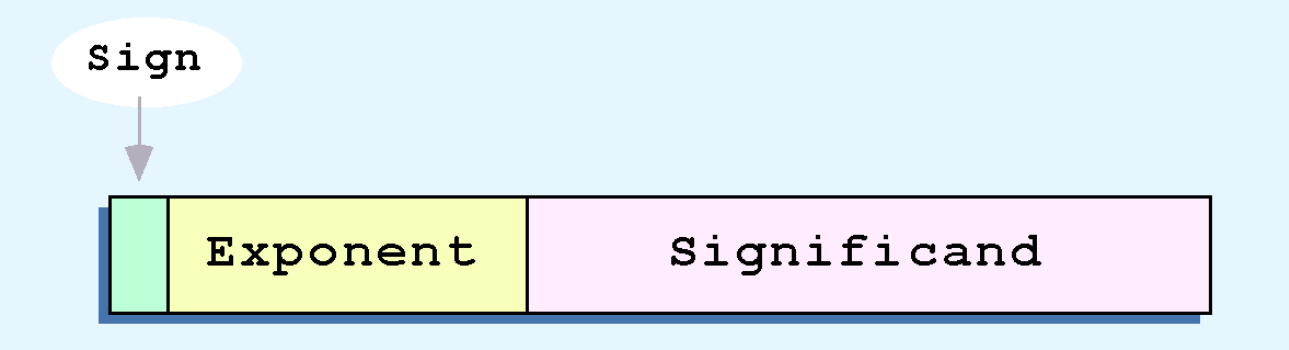
Note: Although "significand" and "mantissa" do not technically mean the same thing, many people use these terms interchangeably. We use the term "significand" to refer to the fractional part of a floating point number.

Floating-Point Representation



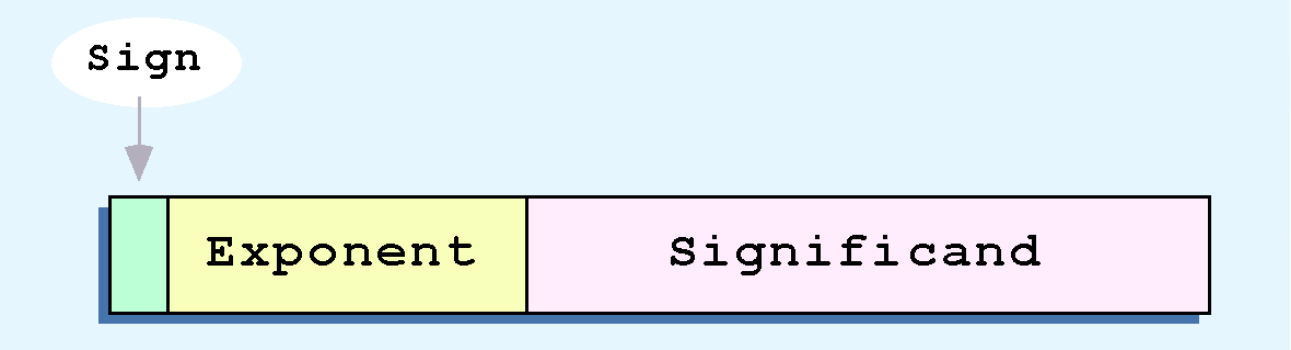
- The one-bit sign field is the sign of the stored value.
- The size of the exponent field determines the range of values that can be represented.
- The size of the significand determines the precision of the representation.

Floating-Point Representation



- We introduce a hypothetical “Simple Model” to explain the concepts
- In this model:
  - A floating-point number is 14 bits in length
  - The exponent field is 5 bits
  - The significand field is 8 bits

Floating-Point Representation



- The significand is always preceded by an implied binary point.
- Thus, the significand always contains a fractional binary value.
- The exponent indicates the power of 2 by which the significand is multiplied.

Floating-Point Representation

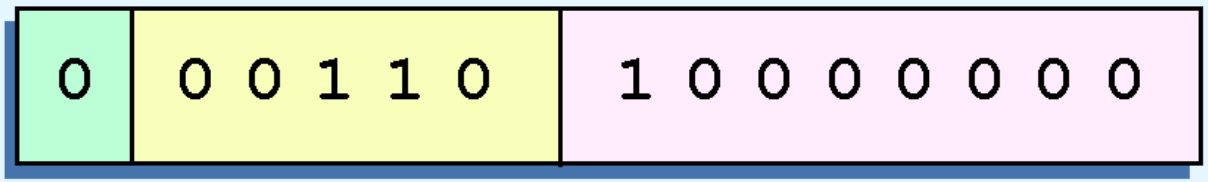
- Example:

Express 32_{10} in the simplified 14-bit floating-point model.

- We know that 32 is 2^5. So in (binary) scientific notation $32 = 1.0 \times 2^5 = 0.1 \times 2^6$.

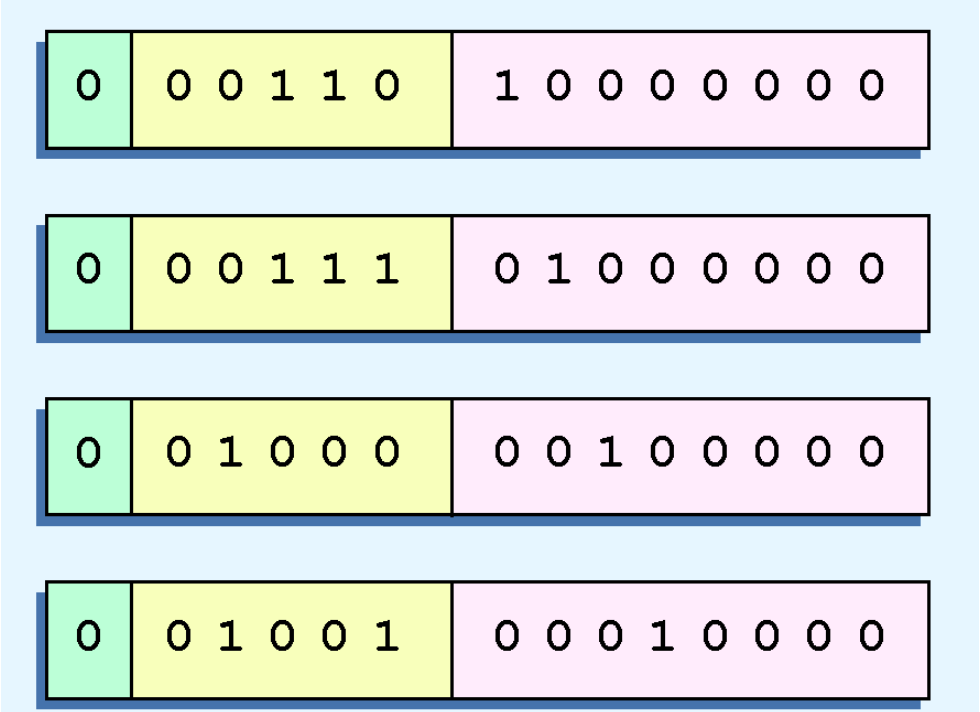
In a moment, we'll explain why we prefer the second notation versus the first.

- Using this information, we put 110 (= 6_{10}) in the exponent field and 1 in the significand as shown.

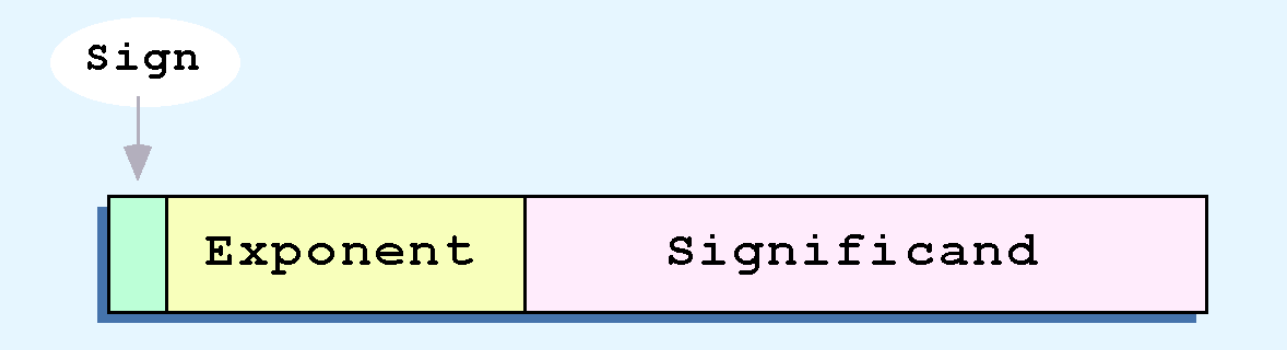


2.5 Floating-Point Representation

- The illustrations shown at the right are *all* equivalent representations for 32 using our simplified model.
- Not only do these synonymous representations waste space, but they can also cause confusion.



Floating-Point Representation



- Another problem with our system is that we have made no allowances for negative exponents. We have no way to express 0.5 ($=2^{-1}$)! (Notice that there is no sign in the exponent field.)

All of these problems can be fixed with no changes to our basic model.

Floating-Point Representation

- To resolve the problem of synonymous forms, we establish a rule that the first digit of the significand must be 1, with no ones to the left of the radix point.
- This process, called *normalization*, results in a unique pattern for each floating-point number.

In our simple model, all significands must have the form 0.1xxxxxxxx

For example, $4.5 = 100.1 \times 2^0 = 1.001 \times 2^2 = 0.1001 \times 2^3$. The last expression is correctly normalized.

In our simple instructional model, we use no implied bits.

Floating-Point Representation

- To provide for negative exponents, we will use a *biased exponent*.
- A bias is a number that is approximately midway in the range of values expressible by the exponent. We subtract the bias from the value in the exponent to determine its true value.

In our case, we have a 5-bit exponent. We will use 16 for our bias. This is called *excess-16* representation.

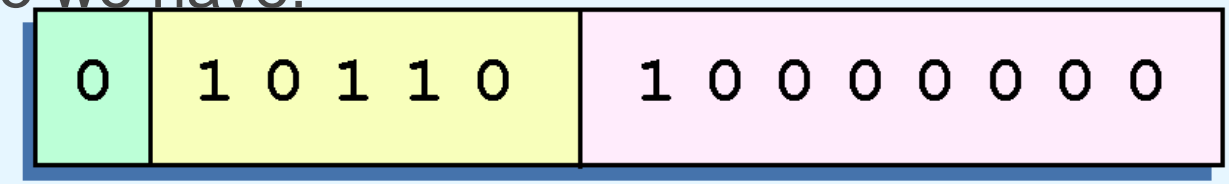
- In our model, exponent values less than 16 are negative, representing fractional numbers.

Example 1

- Example:

Express 32_{10} in the revised 14-bit floating-point model.

- We know that $32 = 1.0 \times 2^5 = 0.1 \times 2^6$.
- To use our excess 16 biased exponent, we add 16 to 6, giving 22_{10} ($=10110_2$).
- So we have:

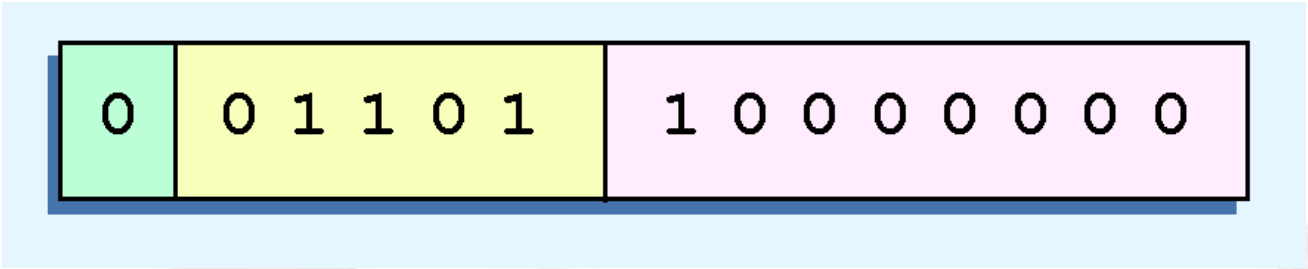


Example 2

- Example:

Express 0.0625_{10} in the revised 14-bit floating-point model.

- We know that 0.0625 is 2^{-4}. So in (binary) scientific notation $0.0625 = 1.0 \times 2^{-4} = 0.1 \times 2^{-3}$.
- To use our excess 16 biased exponent, we add 16 to -3, giving 13_{10} (=01101_2).

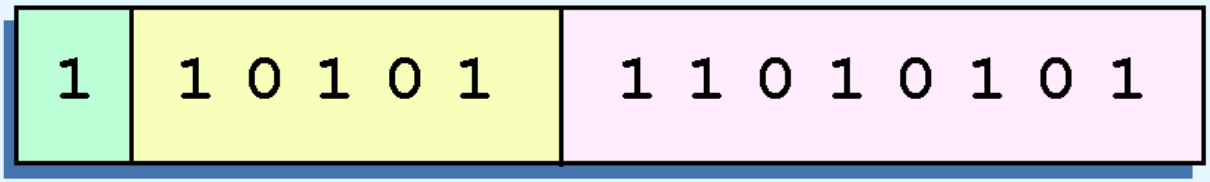


Example 3

- Example:

Express -26.625_{10} in the revised 14-bit floating-point model.

- We find $26.625_{10} = 11010.101_2$. Normalizing, we have: $26.625_{10} = 0.11010101 \times 2^5$.
- To use our excess 16 biased exponent, we add 16 to 5, giving 21_{10} ($=10101_2$). We also need a 1 in the sign bit.



Floating-Point Standards

- The IEEE has established a standard for floating-point numbers
- The IEEE-754 *single precision* floating point standard uses an 8-bit exponent (with a bias of 127) and a 23-bit significand.
- The IEEE-754 *double precision* standard uses an 11-bit exponent (with a bias of 1023) and a 52-bit significand.

Floating-Point Standards

- The IEEE has established a standard for floating-point numbers
- The IEEE-754 *single precision* floating point standard uses an 8-bit exponent (with a bias of 127) and a 23-bit significand.
- The IEEE-754 *double precision* standard uses an 11-bit exponent (with a bias of 1023) and a 52-bit significand.

FP Ranges

For a 32 bit number

8 bit exponent

+/- $2^{256} \approx 1.5 \times 10^{77}$

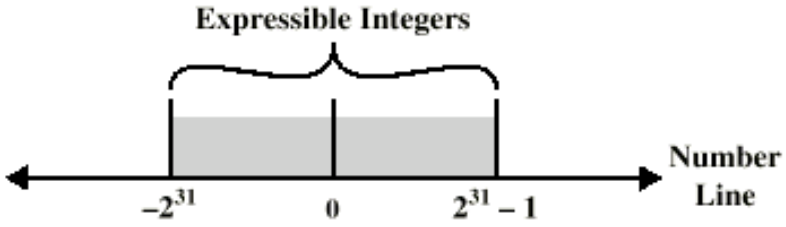
Accuracy

The effect of changing lsb of significand

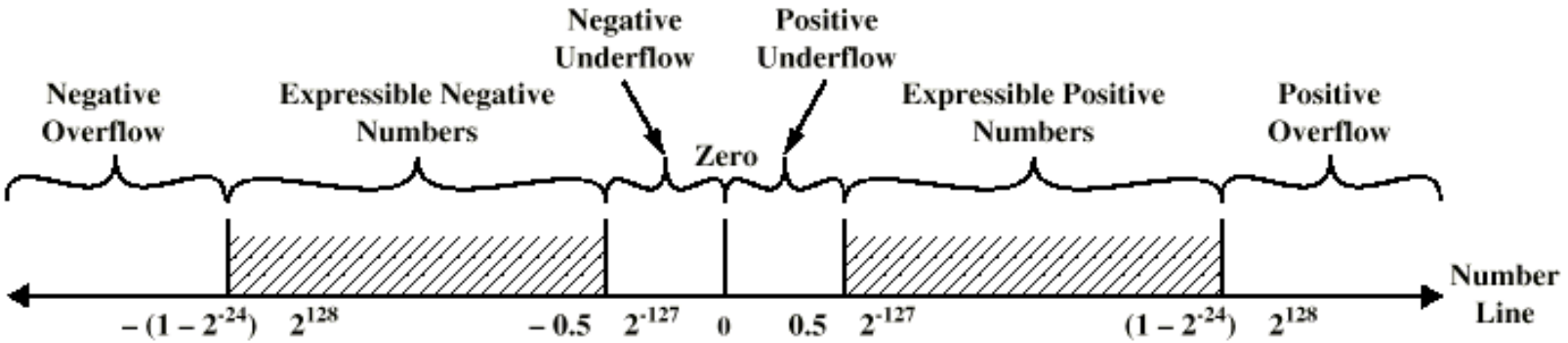
23 bit significand $2^{-23} \approx 1.2 \times 10^{-7}$

About 6 decimal places

Expressible Numbers



(a) Twos Complement Integers



(b) Floating-Point Numbers