

# Fixed Point and Floating Point Number Representations

- Digital Computers use Binary number system to represent all types of information inside the computers. Alphabetic characters are represented using binary bits (i.e., 0 and 1). Digital representations are easier to design, storage is easy, accuracy and precision are greater.

Here are various types of number representation techniques for digital number representation, for example:

Binary number system,

octal number system,

decimal number system, and

hexadecimal number system etc.

But Binary number system is most relevant and popular for representing numbers in digital computer system.

# Storing Real Number:

Unsigned integer	Integer
Signed integer	Sign Integer
Unsigned fixed point	Integer Fraction
Signed fixed point	Sign Integer Fraction
Floating point	Sign Exponent Sign Mantissa
Variable length	Sign Size Digits
Unsigned rational	Numerator Denominator
Signed rational	Sign Numerator Denominator

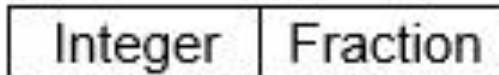
# two major approaches to store real numbers

- (i) Fixed Point Notation and
  - (ii) Floating Point Notation.
- 
- In fixed point notation, there are a fixed number of digits after the decimal point, whereas floating point number allows for a varying number of digits after the decimal point.

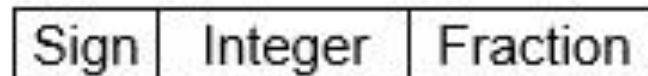
# Fixed-Point Representation:

- This representation has fixed number of bits for integer part and for fractional part.
- For example, if given fixed-point representation is xxxx.yyyy, then you can store minimum value is 0000.0001 and maximum value is 9999.9999.
- There are three parts of a fixed-point number representation: the sign field, integer field, and fractional field.

Unsigned fixed point



Signed fixed point



# Fixed-Point Representation:

We can represent these numbers using:

Signed representation: range from  $-(2^{(k-1)}-1)$  to  $(2^{(k-1)}-1)$ , for  $k$  bits.

1's complement representation: range from  $-(2^{(k)}-1)$  to  $(2^{(k)}-1)$ , for  $k$  bits.

2's complement representation: range from  $-(2^{(k)})$  to  $(2^{(k)}-1)$ , for  $k$  bits.

2's complement representation is preferred in computer system because of unambiguous property and easier for arithmetic operations.

- **Example:** Assume number is using 32-bit format which reserve 1 bit for the sign, 15 bits for the integer part and 16 bits for the fractional part.
- Then, -43.625 is represented as following:

- 

1	000000000101011	1010000000000000
Sign bit	Integer part	Fractional part




# Floating-Point Representation:

- This representation does not reserve a specific number of bits for the integer part or the fractional part. Instead it reserves a certain number of bits for the number (called the mantissa or significand) and a certain number of bits to say where within that number the decimal place sits (called the exponent).

$$2 \times 10^9$$

2.000000000



1 2 3 4 5 6 7 8 9

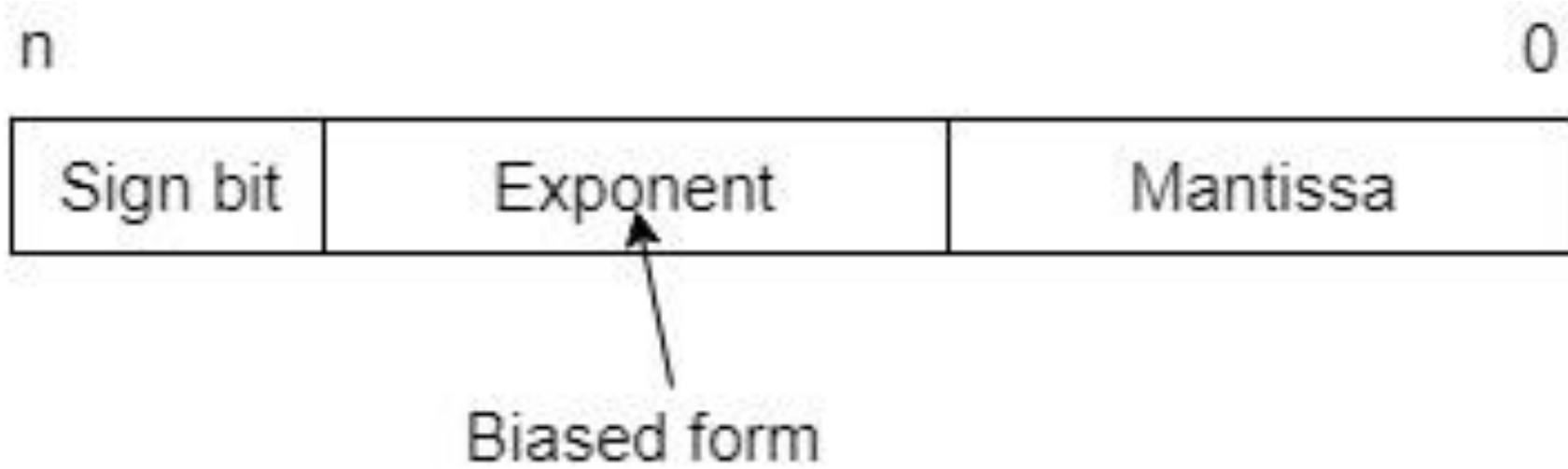
2,000,000,000

# What is Floating in Floating Point Representation

- ?

- The floating number representation of a number has two part: the first part represents a signed fixed point number called mantissa. The second part of designates the position of the decimal (or binary) point and is called the exponent. The fixed point mantissa may be fraction or an integer. Floating -point is always interpreted to represent a number in the following form:  $M \times r^e$ .

- Only the mantissa  $m$  and the exponent  $e$  are physically represented in the register (including their sign). A floating-point binary number is represented in a similar manner except that it uses base 2 for the exponent. A floating-point number is said to be normalized if the most significant digit of the mantissa is 1.



- So, actual number is  $(-1)^s(1+m)x2^{(e-Bias)}$ , where  $s$  is the sign bit,  $m$  is the mantissa,  $e$  is the exponent value, and  $Bias$  is the bias number.
- Note that signed integers and exponent are represented by either sign representation, or one's complement representation, or two's complement representation.
- The floating point representation is more flexible. Any non-zero number can be represented in the normalized form of  $\pm(1.b_1b_2b_3 \dots)_2x2^n$  This is normalized form of a number  $x$ .

- **Example:** Suppose number is using 32-bit format: the 1 bit sign bit, 8 bits for signed exponent, and 23 bits for the fractional part. The leading bit 1 is not stored (as it is always 1 for a normalized number) and is referred to as a *“hidden bit”*.
- Then  $-53.5$  is normalized as  $-53.5 = (-110101.1)_2 = (-1.101011) \times 2^5$ , which is represented as following below,

1	00000101	10101100000000000000000
Sign bit	Exponent part	Mantissa part

- The precision of a floating-point format is the number of positions reserved for binary digits plus one (for the hidden bit). In the examples considered here the precision is  $23+1=24$ .
- The gap between 1 and the next normalized floating-point number is known as machine epsilon. the gap is  $(1+2^{-23})-1=2^{-23}$  for above example, but this is same as the smallest positive floating-point number because of non-uniform spacing unlike in the fixed-point scenario.
- Note that non-terminating binary numbers can not be represented in floating point representation, e.g.,  $1/3 = (0.010101 \dots)_2$  cannot be a floating-point number as its binary representation is non-terminating.



# IEEE Floating point Number Representation:

- IEEE (Institute of Electrical and Electronics Engineers) has standardized Floating-Point Representation as following diagram.

- 



So, actual number is  $(-1)^s(1+m) \times 2^{(e-Bias)}$ , where  $s$  is the sign bit,  $m$  is the mantissa,  $e$  is the exponent value, and  $Bias$  is the bias number. The sign bit is 0 for positive number and 1 for negative number. Exponents are represented by or two's complement representation.

According to IEEE 754 standard, the floating-point number is represented in following ways:

Half Precision (16 bit): 1 sign bit, 5 bit exponent, and 10 bit mantissa

Single Precision (32 bit): 1 sign bit, 8 bit exponent, and 23 bit mantissa

Double Precision (64 bit): 1 sign bit, 11 bit exponent, and 52 bit mantissa

Quadruple Precision (128 bit): 1 sign bit, 15 bit exponent, and 112 bit mantissa

# Special Value Representation:

- There are some special values depended upon different values of the exponent and mantissa in the IEEE 754 standard.
- All the exponent bits 0 with all mantissa bits 0 represents 0. If sign bit is 0, then +0, else -0.
- All the exponent bits 1 with all mantissa bits 0 represents infinity. If sign bit is 0, then  $+\infty$ , else  $-\infty$ .
- All the exponent bits 0 and mantissa bits non-zero represents denormalized number.
- All the exponent bits 1 and mantissa bits non-zero represents error.

# Summary

## FIXED POINT VERSUS FLOATING POINT

FIXED POINT	FLOATING POINT
A representation of real data type for a number that has a fixed number of digits after the radix point	A formulaic representation of real numbers as an approximation so as to support a tradeoff between range and precision
Used to represent a limited range of values	Used to represent a wide range of values
Higher performance	Lower performance
Less flexible	More flexible