# CSARCH Lecture Series: Binary Floating-Point format for Double Precision (special cases)

Sensei RL Uy

College of Computer Studies

De La Salle University

Manila, Philippines

‘2110

# Copyright Notice

This lecture contains copyrighted materials and is use solely for instructional purposes only, and not for redistribution.

Do not edit, alter, transform, republish or distribute the contents without obtaining express written permission from the author.

# Overview

Reflect on the following questions:

- How are zeros and infinity represented in the memory?
- How large should a double-precision floating-point number be to considered an infinity?

# Overview

- This sub-module introduces the IEEE-754 double-precision floating-point format involving special cases
- The objective is as follows:
  - ✓Describe the process of representing special cases such as zero, infinity, denormalized and NaN using IEEE-754 standard

# Special cases (IEEE-754 Double Precision)

- IEEE-754 supports the following special cases:
  - ✓Zero (0)
  - ✓Infinity (very big number)
  - ✓Denormalized or subnormal (very small number)
  - ✓NaN (log(-1), $\sqrt{-1}$)

# Special cases

| Sign bit | E' (11-bit) | Significand (52-bit) | Value |
|---|---|---|---|
| 0 | 000 0000 0000 | 000 0000 0000 0000 0000 0000 … 0 | +0 (Positive Zero) |
| 1 | 000 0000 0000 | 000 0000 0000 0000 0000 0000 … 0 | -0 (Negative Zero) |
| 0/1 | 000 0000 0000 | ≠ 0 | Denomalized |
| 0 | 111 1111 1111 | 000 0000 0000 0000 0000 0000 …0 | + Infinity |
| 1 | 111 1111 1111 | 000 0000 0000 0000 0000 0000 …0 | - Infinity |
| x | 111 1111 1111 | 0xx xxxx xxxx xxxx xxxx xxxx …x | sNaN |
| x | 111 1111 1111 | 1xx xxxx xxxx xxxx xxxx xxxx …x | qNaN |

Special cases use smallest (00000000000) and the largest (11111111111) exponent representation (e')

# Special case (Denormalized)

- Denormalized are numbers so small (approaching 0) that it cannot be represented normally
- What is the smallest positive normal number?

| Sign | Exponent representation | Fraction part of significand |
|---|---|---|
| 0 | 0000 0001 | 0 … 0 |

The smallest possible e’ is 1. Thus e=1-1023 = -1022

The smallest positive normal number is +1.0x2$^{-1022}$ (or 2.23x10$^{-308}$)

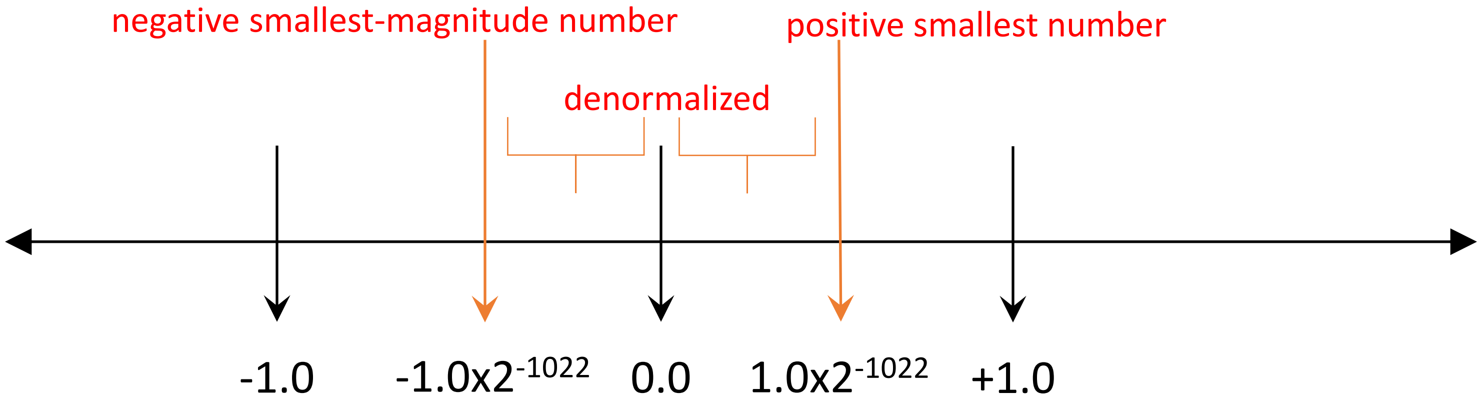
- What is the smallest-magnitude negative normal number?

The smallest-magnitude negative normal number is -1.0x2$^{-1022}$ (or -2.23x10$^{-308}$)

# Special case (Denormalized)

The smallest positive normal number is $+1.0\text{x}2^{-1022}$ (or $2.23\text{x}10^{-308}$)

The smallest-magnitude negative normal number is $-1.0\text{x}2^{-1022}$ (or $-2.23\text{x}10^{-308}$)



To represent denormal number

- peg the exponent to -1022 and denormalized the significand
- e’ = 0
- significand is the denormalized significand

# Special case (Denormalized)

Example: $-1.1110_2\text{x } 2^{-1026}$

normalized format: $-0.0001111_2\text{x } 2^{-1022}$

| | |
|---|---|
| Significand in binary? | Yes |
| Base-2? | Yes |
| Normalized? | Yes. But special case, need to denormalized |
| Sign bit | 1 |
| Exponent representation | special case: 000 0000 0000 |

Answer:

| Sign | Exponent representation | Fraction part of significand |
|---|---|---|
| 1 | 000 0000 0000 | 000 1111 0…0 |

Hex: 0x8001E000000000000

# Special case (infinity)

- Infinity are very big numbers (approaching infinity) that it cannot be represented normally
- What is the largest positive normal number?

| Sign | Exponent representation | Fraction part of significand |
|---|---|---|
| 0 | 111 1111 1110 | 0..0 |

The largest possible e' is 2046. Thus e=2046-1023 = 1023

The largest positive normal number is +1.1…1x2$^{1023}$ (or 1.8x10$^{308}$)

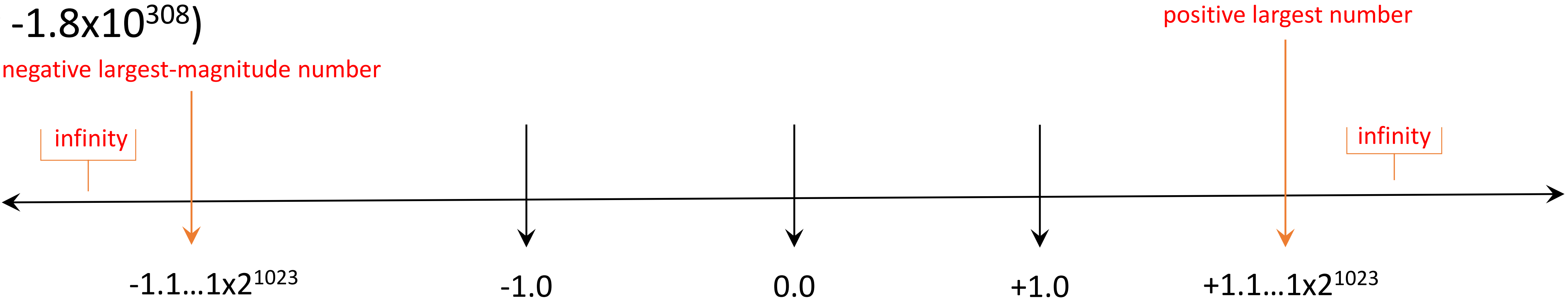
- What is the largest-magnitude negative normal number?

The largest-magnitude negative normal number is -1. 1…1x2$^{1023}$ (or -1.8x10$^{308}$)

# Special case (infinity)

The largest positive normal number is +1.1…1x$2^{1023}$ (or 1.8x$10^{308}$)

The largest-magnitude negative normal number is -1.1…1x$2^{1023}$ (or -1.8x$10^{308}$)



To represent infinity number

- e’ = 11111111111
- significand is 0…0

# Special case (Infinity)

Example: $+1.111_2 x\ 2^{9999}$

normalized format: $+1.111_2 x\ 2^{9999}$ (Same)

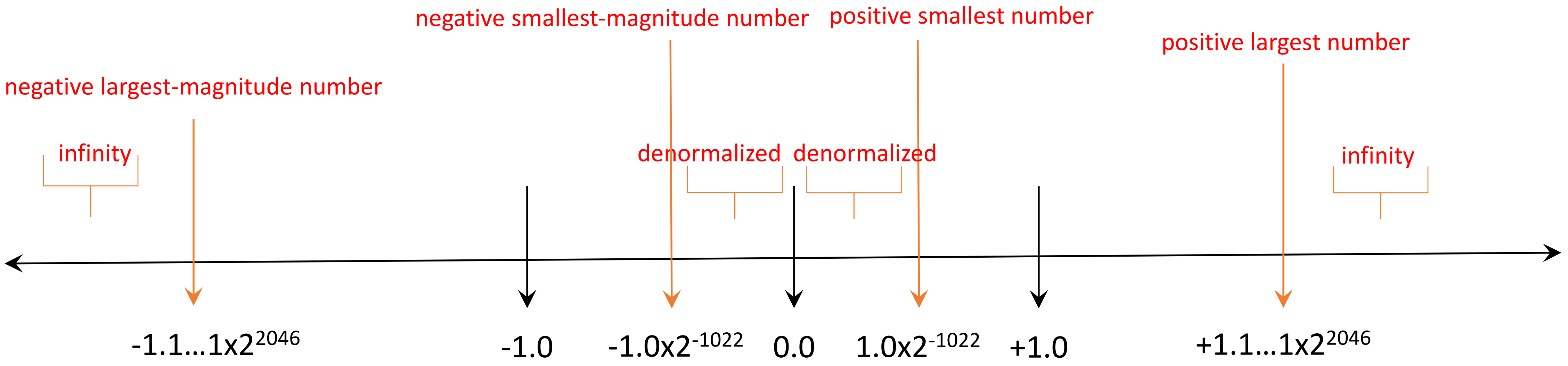
| | |
|---|---|
| Significand in binary? | Yes |
| Base-2? | Yes |
| Normalized? | Yes |
| Sign bit | 0 |
| Exponent representation | special case: 111 1111 1111 |

Answer:

| Sign | Exponent representation | Fraction part of significand |
|---|---|---|
| 0 | 111 1111 1111 | 0…0 |

Hex: 0x7FF0000000000000

# Special case (number line)



# Special case (NaN)

- Indeterminate numbers are example of Not a Number (NaN)
- Sign bit is don’t care
- there are 2 types of NaN representation:
    - Signaling NaN (sNaN)
        - Two most significant bit of the significand is 01
        - floating-point result using sNaN signals the invalid operation exception
    - Quiet NaN (qNaN)
        - most significant bit of the significand is 1
        - floating-point result using qNaN allows the result to be propagated

| Sign Bit | E’ | Significand | Value |
|---|---|---|---|
| x | 1111 1111 | 01x…x | sNaN |
| x | 1111 1111 | 1x…x | qNaN |

# To recall …

- What have we learned:
  - ✓Describe the process of representing special cases such as zero, infinity, denormalized and NaN using IEEE-754 standard