4190.308: Computer Architecture Lab. 2

Jaehoon Shim
Ikjoon Son
Seongyeop Jeong
(snucsl.ta@gmail.com)

Systems Software & Architecture Lab.

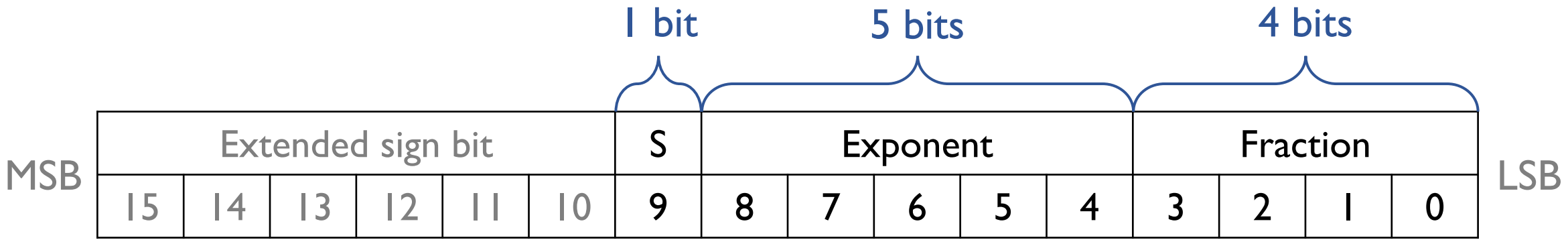
Seoul National University

Fall 2021

FP10 Representation

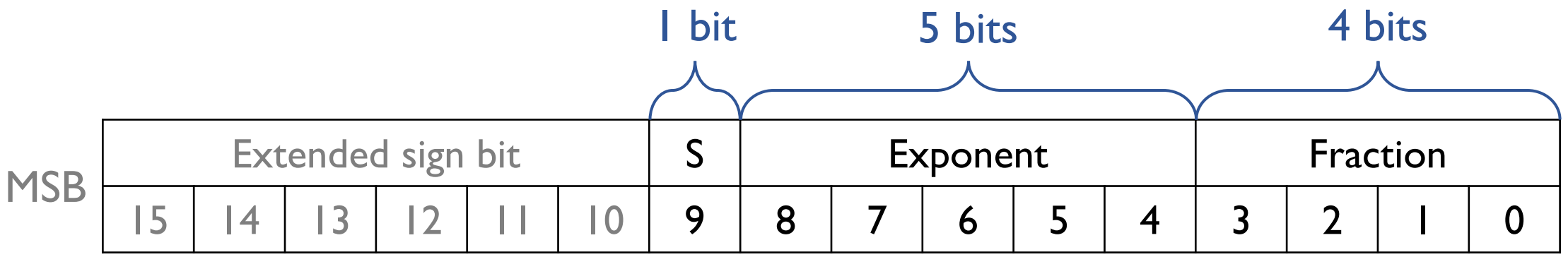
FP10 (10-bit floating Point)

- 10-bit floating point representation that follows the IEEE 754 standard for floating point arithmetic
- It consists of 1-bit sign bit, 5-bit exponent, and 4-bit fraction



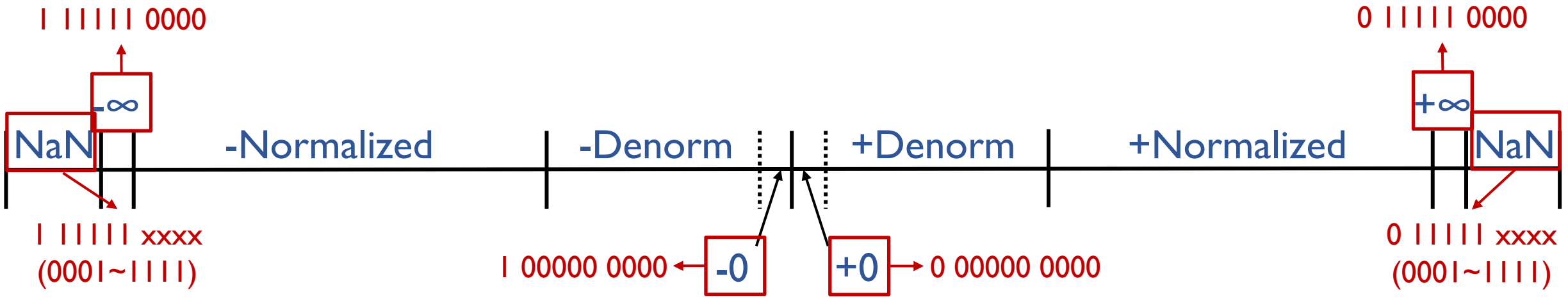
FP10 Bias

- Bias for 5-bit Exponent: $2^{5-1} - 1 = 15$
  - Smallest positive number: 0 00000 0001 → 0.0001×2^{-14}
  - Largest positive number: 0 11110 1111 → 1.1111×2^{15}



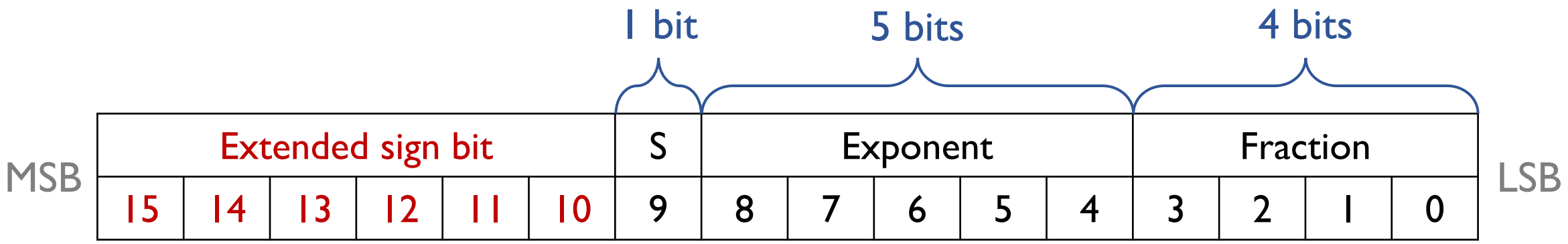
FP10 Values

- You should follow the IEEE standard rules and representations
  - Normalized values: exp ≠ 00000 and exp ≠ 11111
  - Denormalized values: exp = 00000
- For rounding, you should use the **round-to-even** scheme



FP10 in C

- In C, we use a 16-bit unsigned short integer to store FP10
- Upper 6 bits have the same value as the sign bit
  - 111111 when S = 1
  - 000000 when S = 0

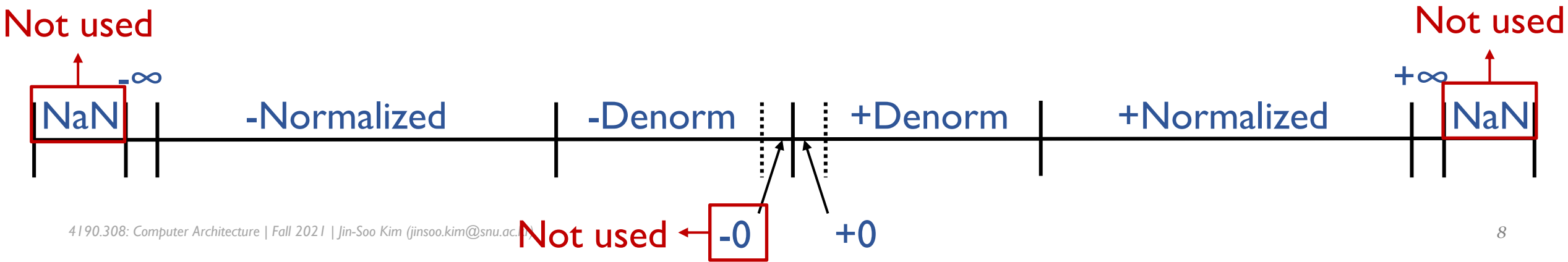


Specification

- You should implement four functions given in pa2.c
- fp10 int_fp10 (int n);
- int fp10_int (fp10 x);
- fp10 float_fp10 (float f);
- float fp10_float (fp10 x);

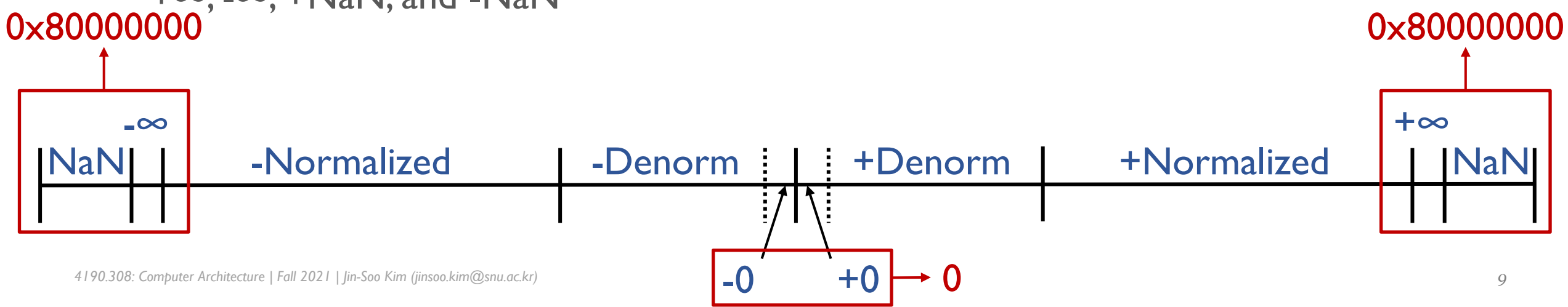
Specification – int_fp10()

- Convert int to fp10
- Use **round-to-even** rounding mode when necessary
- Convert 0 to the plus zero (+0.0) in fp10
- Values that exceed the range of fp10 → convert to +∞ or -∞



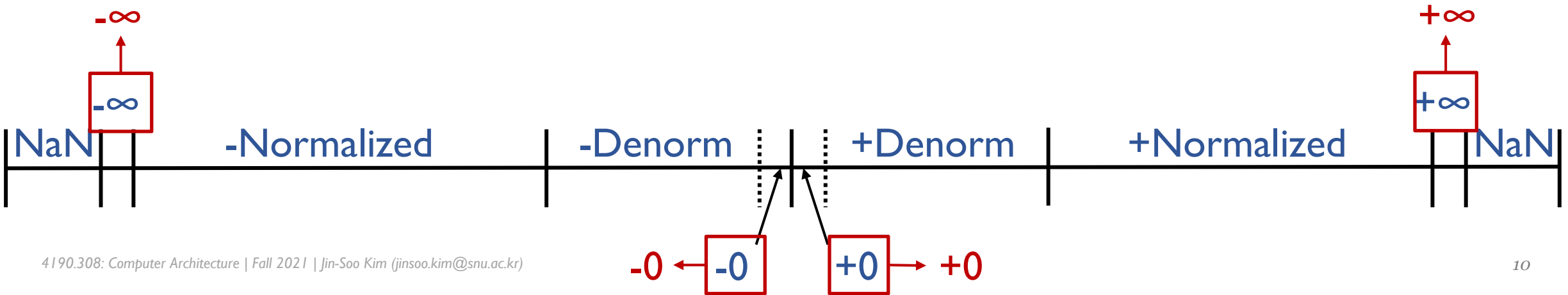
Specification – fp10_int()

- Convert fp10 to int
- Same as using **round-toward-zero** rounding
  - Any fractional part is dropped (xx.yy → xx)
- Covert +0.0 and -0.0 to 0
- Convert to the smallest number in int when,
  - Value is too large to represent as integer
  - +∞, -∞, +NaN, and -NaN



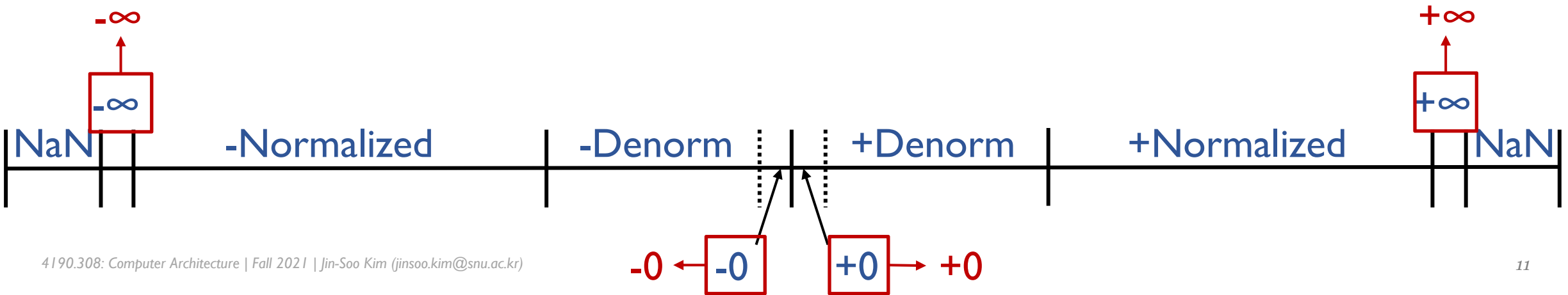
Specification – float_fp10()

- Convert float to fp10
- Use **round-to-even** rounding mode when necessary
- Values that exceed the range of fp10 → convert to +∞ or -∞
- Covert +∞ and -∞ to corresponding +∞ and -∞ in fp10
- For +NaN/-NaN, any representation for +NaN/-NaN is OK



Specification – fp10_float()

- Convert fp10 to float
- Covert +∞ and -∞ to corresponding +∞ and -∞ in float
- For +NaN/-NaN, any representation for +NaN/-NaN is OK



Example: int_fp10()

- (1) fp10 int_fp10 (int n);

$1 = 1_{(2)} = 1.0 \times 2^0 \rightarrow$ 0000 0000 0 1111 0000

exponent (15 + 0) fraction

L = 1, R = 1, S = 1 round up

$254 = 11111110_{(2)} = 1.1111110 \times 2^7 = 10.0000 \times 2^7 = 1.0000 \times 2^8$

$\rightarrow$ 0000 0001 0111 0000

exponent (15 + 8) fraction

Example: int_fp10()

- (1) fp10 int_fp10 (int n);

> 63488 (1.1111×2^{15})

L = 1, R = 0, S = 1

$63554 = 1111100001000010_{(2)} = 1.1111000010000010 \times 2^{15}$

round down

→ 1.1111×2^{15}

→ 0000 0001 1110 1111

exponent (15 + 15)

fraction

Example: fp10_int()

- (2) int fp10_int (fp10 x);

0000 000 1 0011 1101 → e = 19 – 15 = 4

exponent fraction

→ $1.1101 \times 2^4 = 11101_{(2)} = 29$

Example: float_fp10()

- (3) fp10 float_fp10(float f);

0011 1111 1 | 110 0000 0000 0000 0000 0000 → e = 127 – 127 = 0

exponent | fraction

→ 1.1100 x 2^0

→ 0000 0000 0 1111 | 1100

exponent (15 + 0) | fraction

Example: float_fp10()

- (3) fp10 float_fp10(float f);

0011 1000 0111 1110 0000 0000 0000 0000 → e = 112 - 127 = -15

exponent: 0011 1000 0 — fraction: 1111 1110 0000 0000 0000 0000

round up

→ $1.111111 \times 2^{-15} = 0.11111111 \times 2^{-14} = 1.0000 \times 2^{-14}$

smallest normalized value: 1.0×2^{-14}

minimum normalized value

→ 0000 0000 0001 0000

exponent (15 - 14): 0 0001 — fraction: 0000

Example: float_fp10()

- (3) fp10 float_fp10(float f);

0101 0000 0111 1001 1000 0000 0000 0000 → e = 160 – 127 = 33

largest positive value: 1.1111 x 2^{15}

→ 1.11110011 x 2^{33} = 1.1111 x 2^{33}

Exceed range of fp10

→ 0000 0001 1111 0000 (+∞)

exponent fraction

Example: fp10_float()

- (4) float fp10_float(fp10 x);

1111 1110 0000 0001 → e = 1 - 15 = -14

denormalized value in fp10

→ $-0.0001 \times 2^{-14} = -1.0 \times 2^{-18}$

normalized value in float

→ 1011 0110 1000 0000 0000 0000 0000 0000

exponent (127 - 18)

fraction

Restrictions

- You should not use any array even in the comment lines
  - Symbol '[' and ']' will be rejected
- You are not allowed to use long or double data type
  - Word 'long' and 'double' will be rejected
- Do not include any header files other than pa2.h in the pa2.c file

Restrictions

- You can't use any libraries in pa2.c, including <stdio.h>
  - Please make sure to remove it before submission

- Your solution should finish within a reasonable time

- Top15 fastest float_fp10() implementations will receive a 10% extra bonus

Submission

- Due: 11:59PM, October 17 (Sunday)
  - 25% of the credit will be deducted for every single day delay

- Only submit the pa2.c file to the submission server
  - You don't have to write a report in this assignment

Thank You!

- Don't forget to read the detailed description before you start your assignment

- If you have any questions about the assignment,
  feel free to ask via KakaoTalk

- This file will be uploaded after the lab session☺