Sunmin Jeong, Injae Kang

(snucsl.ta@gmail.com)

Systems Software &
Architecture Lab.

Seoul National University

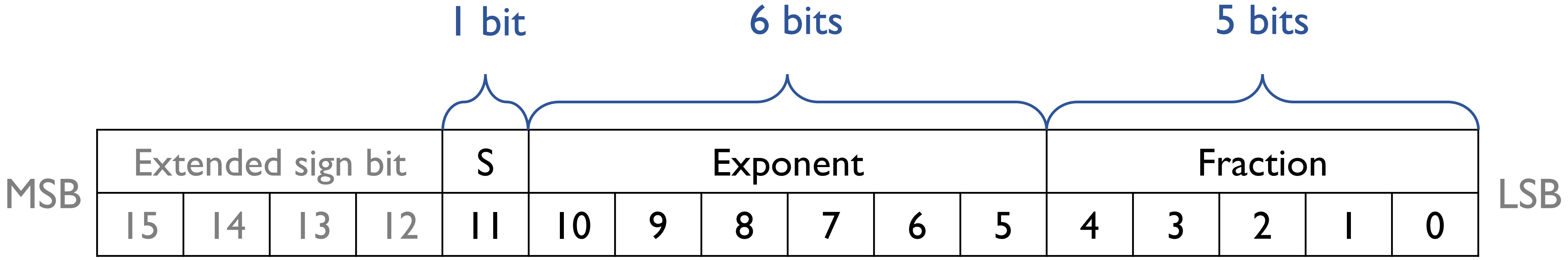
Fall 2020

4190.308: Computer Architecture Lab. 2

FP12 Representation

FP12 (12-bit floating point)

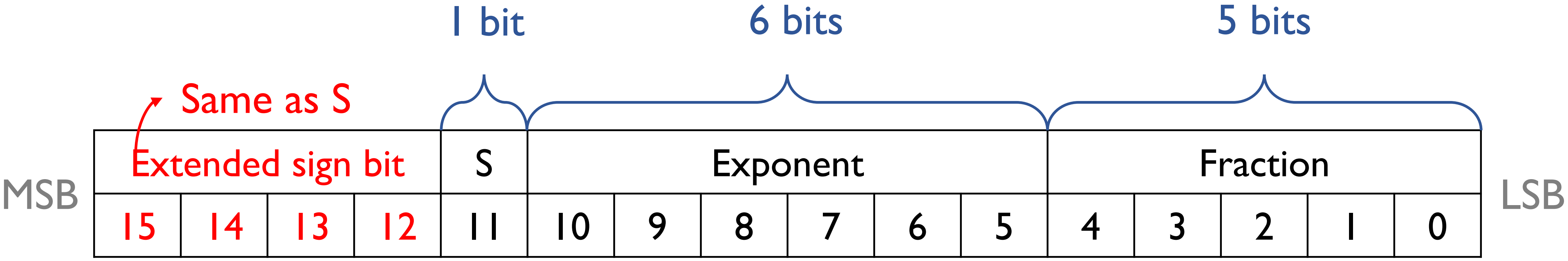
- 12-bit floating point representation that follows the IEEE 754 standard for floating point arithmetic
- It consists of 1-bit Sign bit, 6-bit Exponent and 5-bit fraction



					1 bit	6 bits						5 bits					
	Extended sign bit				S	Exponent						Fraction					
MSB	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	LSB

FP12 (12-bit floating point)

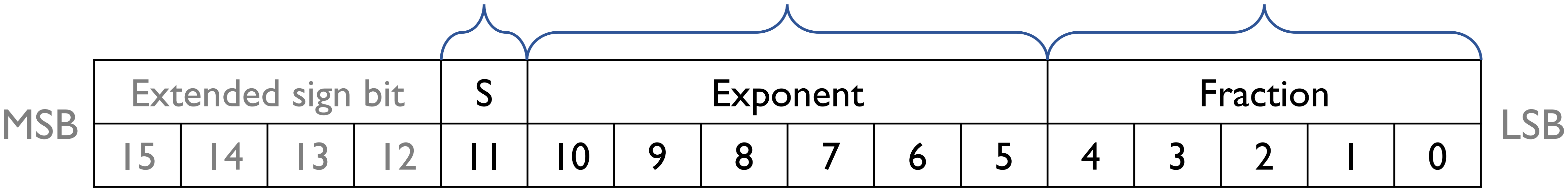
- In C, we use a 16-bit short integer to store FP12
- Upper 4 bits have the same value as the sign bit
  - 1111 when sign bit is 1
  - 0000 when sign bit is 0



Extended sign bit				S	Exponent						Fraction				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

FP12 (12-bit floating point)

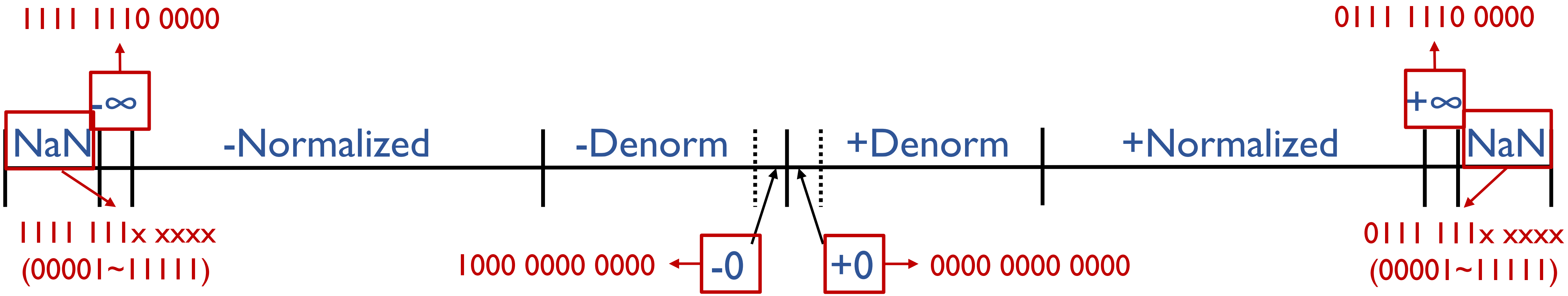
- 6-bit Exponent → Bias = 2^{6-1} -1 = 31
  - Smallest positive number: 0.00001 x 2^{-30}
  - Largest positive number: 1.11111 x 2^{31}



Extended sign bit				S	Exponent						Fraction				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

FP12 (12-bit floating point)

- You should follow the IEEE standard rules and representations
  - Normalized values: exp ≠ 000000 and exp ≠ 111111
  - Denormalized values: exp = 000000



FP12 (12-bit floating point)

For rounding, you should use the **round-to-even** scheme.

For normalized values:

$$1.\underbrace{BBBBL}_{\text{fraction}}RXXX$$

For denormalized values:

$$0.\underbrace{BBBBL}_{\text{fraction}}RXXX$$

Round up conditions

- R = 1, X > 0
- L = 1, R = 1, X = 0 (round to even)

Specification

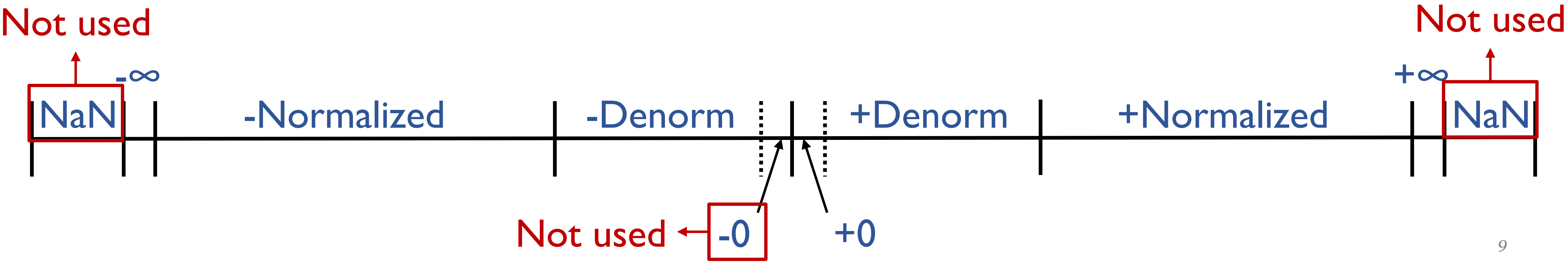
You should implement 4 functions given in pa2.c

- fp12 int_fp12(int n);
- int fp12_int(fp12 x);
- fp12 float_fp12(float f);
- float fp12_float(fp12 x);

Specification

fp12 int_fp12(int n);

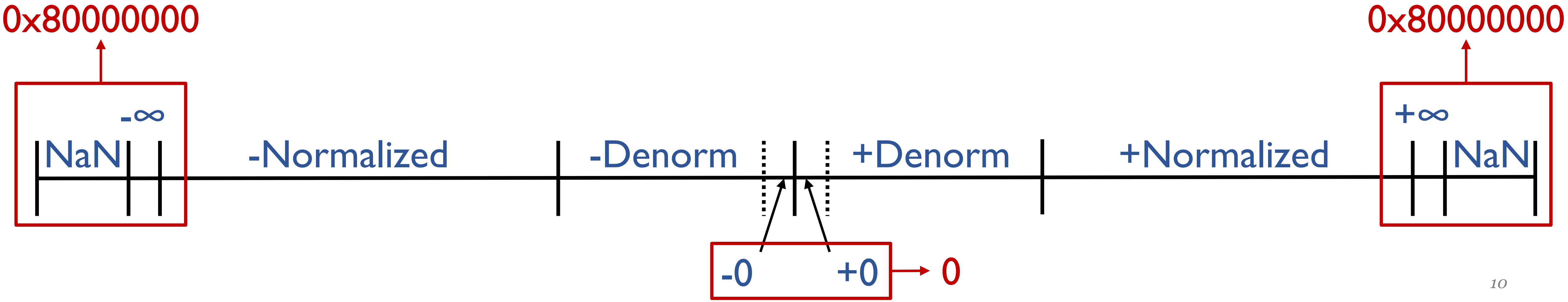
- Convert int to fp12
- Not all int-type values can be represented in the fp12 format
  → Use **round-to-even** rounding mode
- Convert 0 to +0 in fp12
- Values that exceed the range of fp12 → +∞ or -∞



Specification

int fp12_int(fp12 x);

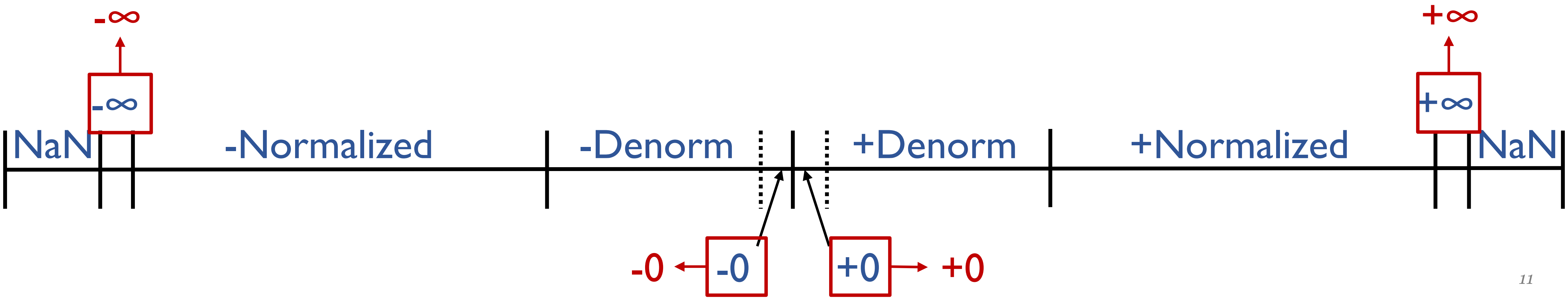
- Convert fp12 to int
- **Round-toward-zero**: any fractional part (xx.yy) is dropped (xx~~.00~~→xx)
- Convert +0.0 and -0.0 to 0
- Convert +∞, -∞, +NaN and -NaN to the smallest number in int



Specification

fp12 float_fp12(float f);

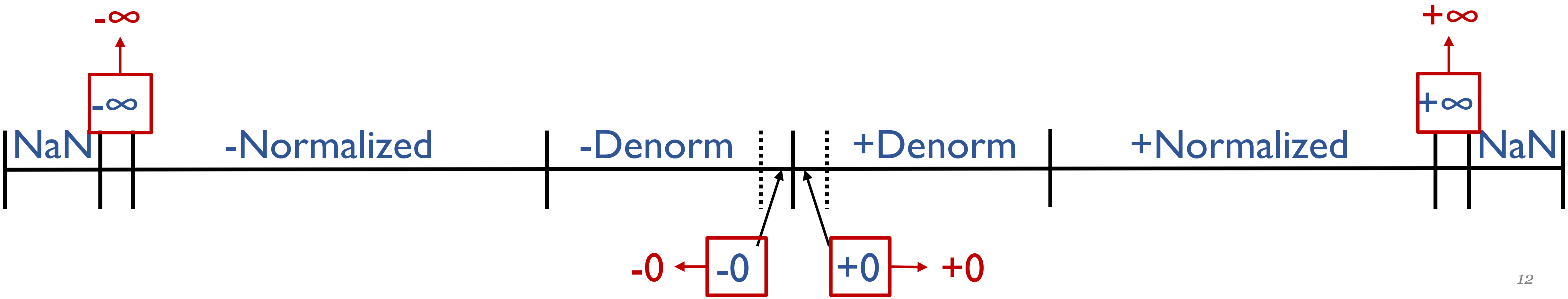
- Convert float to fp12
- Not all float-type values can be represented in the fp12 format
  → Use **round-to-even** rounding mode
- For +NaN/-NaN, any representation for +NaN/-NaN is OK



Specification

float fp12_float(fp12 x);

- Convert fp12 to float
- For +NaN/-NaN, any representation for +NaN/-NaN is OK



Examples

(1) fp12 int_fp12(int n);

$$1 = 1_{(2)} = 1.0 \times 2^0 \rightarrow 0000\ 0\underbrace{011\ 111}_{\substack{\text{exponent}\\(31\ +\ 0)}}\underbrace{0\ 0000}_{\text{fraction}}$$

$$254 = 11111110_{(2)} = 1.111111\overset{\substack{L\ =\ 1,\ R\ =\ 1\\ \text{round-to-even}}}{10} \times 2^7 = 10.00000 \times 2^7 = 1.00000 \times 2^8$$

$$\rightarrow 0000\ 0\underbrace{100\ 111}_{\substack{\text{exponent}\\(31\ +\ 8)}}\underbrace{0\ 0000}_{\text{fraction}}$$

Examples

(2) int fp12_int(fp12 x);

$$0000\ 0\underbrace{100\ 110}_{\text{exponent}}\underbrace{1\ 0110}_{\text{fraction}} \rightarrow e = 38 - 31 = 7$$

$$\rightarrow 1.10110 \times 2^{7} = 11011000_{(2)} = 216$$

Examples

(2) int fp12_int(fp12 x);

$$0000\ 0\underbrace{100\ 000}_{\text{exponent}}\underbrace{1\ 0110}_{\text{fraction}} \rightarrow e = 32 - 31 = 1$$

$$\rightarrow 1.10110 \times 2^{1} = 11.\underset{\text{dropped}}{\cancel{011000}}{}_{(2)} = 11_{(2)} = 3$$

Examples

(3) fp12 float_fp12(float f);

$$0\underbrace{011\ 1111\ 1}_{\text{exponent}}\underbrace{110\ 0000\ 0000\ 0000\ 0000\ 0000}_{\text{fraction}} \rightarrow e = 127 - 127 = 0$$

$$\rightarrow 1.11000 \times 2^{0}$$

$$\rightarrow 0000\ 0\underbrace{011\ 111}_{\substack{\text{exponent} \\ (31\ +\ 0)}}\underbrace{1\ 1000}_{\text{fraction}}$$

Examples

(3) fp12 float_fp12(float f);

$0\underbrace{011\ 0000\ 0}_{\text{exponent}}\underbrace{111\ 1110\ 0000\ 0000\ 0000\ 0000}_{\text{fraction}} \rightarrow e = 96 - 127 = -31$

$\rightarrow \underbrace{1.1111111 \times 2^{-31}}_{\text{denormalized value}} = 0.111111\overset{\text{round up}}{11} \times 2^{-30} = \underbrace{1.00000 \times 2^{-30}}_{\text{minimum normalized value}}$

$\rightarrow 0000\ 0\underbrace{000\ 001}_{\substack{\text{exponent} \\ (31\ -\ 30)}}\underbrace{0\ 0000}_{\text{fraction}}$

Examples

(3) fp12 float_fp12(float f);

$$0\underbrace{100\ 1111\ 0}_{\text{exponent}}\underbrace{111\ 1101\ 1000\ 0000\ 0000\ 0000}_{\text{fraction}} \rightarrow e = 158 - 127 = 31$$

$$\rightarrow 1.111111\overset{\text{no round up}}{011} \times 2^{31} = \underbrace{1.111111}_{\text{maximum value}} \times 2^{31}$$

$$\rightarrow 0000\ 0\underbrace{111\ 110}_{\substack{\text{exponent}\\(31\ +\ 31)}}\underbrace{1\ 1111}_{\text{fraction}}$$

Examples

(4) float fp12_float(fp12 x);

1111 1000 0000 0001 → e = 1 - 31 = -30

denormalized value
in fp12

→ $-0.00001 \times 2^{-30} = -1.0 \times 2^{-35}$

normalized value
in float

$$\rightarrow\ 1\underbrace{010\ 1110\ 0}_{\substack{\text{exponent}\\(127\ -\ 35)}}\underbrace{000\ 0000\ 0000\ 0000\ 0000\ 0000}_{\text{fraction}}$$

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

Restrictions

- You can't use any libraries in pa2.c, including <stdio.h>
  - If you used libraries for debugging, please make sure to erase it before submission.

- Your solution should finish within 5 seconds.

- The top 10 fastest float_fp12() implementations will receive a 10% extra bonus

Submission

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

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

Thank you!

- If you have any question about the assignment, feel free to ask us in KakaoTalk.

- This file will be uploaded after the lab session☺