UNM SCHOOL of ENGINEERING
Department of Computer Science

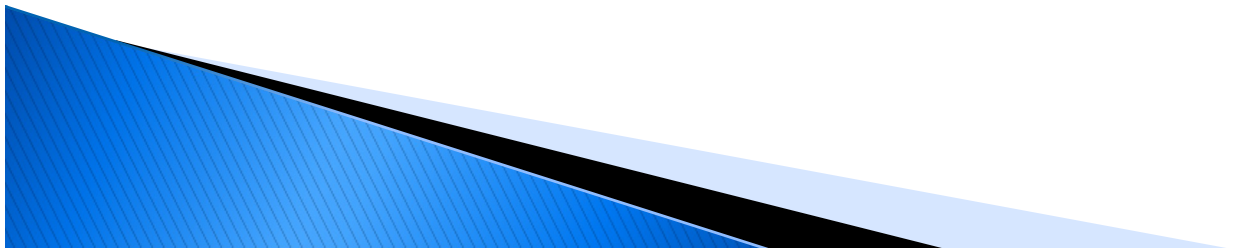
CDR(4): Floating Point representation Lecture #5 – part 1

(Section 2.4 continued)

Prof. Soraya Abad-Mota, PhD

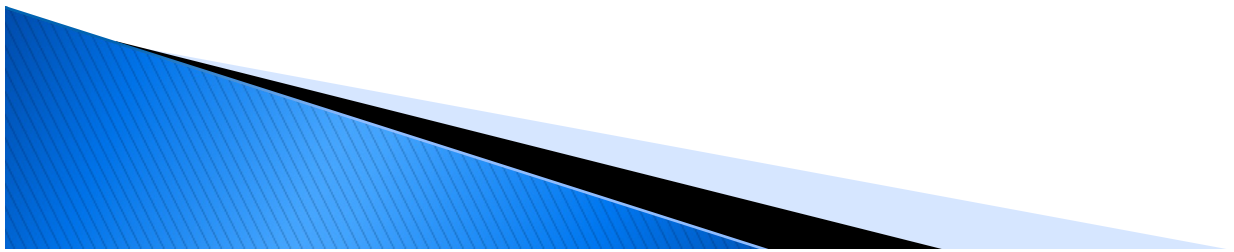
Announcements

- Changes to the calendar are coming up (check canvas later)
  1. The midterm is not on October 9, 2024 anymore, instead it will be on Monday October 14, 2024 (new date for midterm) i.e. the Monday after the fall break.
  2. DATALAB starts tomorrow (will be published late tonight) it is a reduced version from past semesters, but you need to explain more of how you got the puzzles to work and it has less weight on the total percentage given to the projects.
- Respond to the survey linked from canvas (it is worth 3 points!) deadline is next Monday 9/9/24



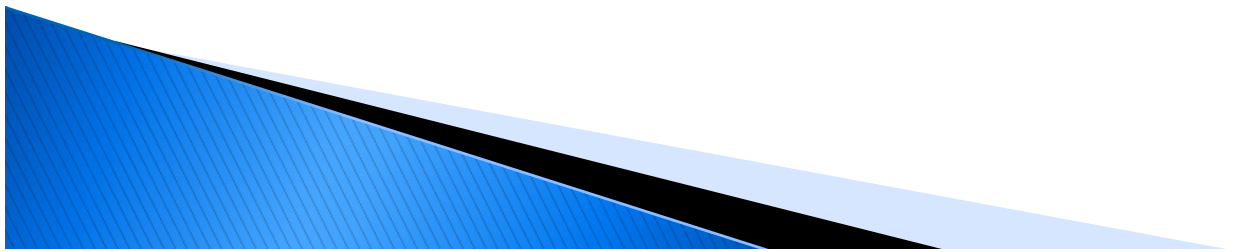
Announcements (2)

- Lab sessions weeks 3 and 4
  - Section 001 this week FP exercises
  - For section 002, the FP exercises will be next Tuesday together with work on the Datalab.
  - Lab session of week 4 for section 001 will be to work on the Datalab



Index cards

- I only accept index cards (not paper cut) of size 3” x 5”
- Write your name and the date at the top
- Empty card -> does not count
- What to write on the cards. Some prompts are: muddy points (questions), answer to some exercises done in class, a comment on some topic covered in the lecture, a reflection on something you learned with this lecture in particular.



Previous lecture (Wed. 8/28/24)

- Started covering floating point, what are binary fractions and how to represent them
- IEEE FP notation (a standard)
  - Normalized values
- Posted two exercises for you to represent with normalized values in the single precision IEEE on 32 bits.
  - 12.0
  - 100.0

Recall IEEE Floating Point format and interpretation

- Numerical Form: (interpretation of bit pattern)

$$(-1)^s \, M \times 2^E$$

  - Sign bit s determines whether number is negative or positive
  - Significand M normally a fractional value in range [1.0,2.0) or [0,1)
  - Exponent E weights value by power of two
- Encoding (how we represent it in the computer)
  - MSB s is sign bit s (1 bit)
  - exp field encodes E (but is not equal to E) (k bits)
  - frac field encodes M (but is not equal to M) (n bits)

Recall 1. "Normalized" case

$v = (-1)^s M 2^E$

- When: exp ≠ 000…0 and exp ≠ 111…1
- Exponent coded as a biased value: E = Exp – Bias
  - Exp: unsigned value of exp field
  - Bias = $2^{k-1} - 1$, where k is number of exponent bits
    - Single precision: 127 (Exp: 1…254, E: -126…127)
    - Double precision: 1023 (Exp: 1…2046, E: -1022…1023)
- Significand coded with implied leading 1: M = $1.xxx…x_2$ (1.0 >= M < 2.0)
  - xxx…x: bits of frac field
  - Minimum when frac = 000…0 (M = 1.0)
  - Maximum when frac = 111…1 (M = 2.0 – ε)
  - Get extra leading bit (= 1) for "free"

How to represent as IEEE normalized values?

- 12.0

- 100.0

2. Denormalized Values

$$v = (-1)^s M 2^E$$
$$E = 1 - \text{Bias}$$

- Condition: exp = 000…0
- Exponent value: E = 1 – Bias (instead of E = 0 – Bias)
- Significand coded with implied leading 0: M = 0.xxx…x_2
  - xxx…x: bits of frac
- Cases
  - exp = **000…0**, frac = **000…0**
    - Represents zero value
    - Note distinct values: +0 and –0 (why?)
  - exp = **000…0**, frac ≠ **000…0**
    - Numbers closest to 0.0
    - Equispaced

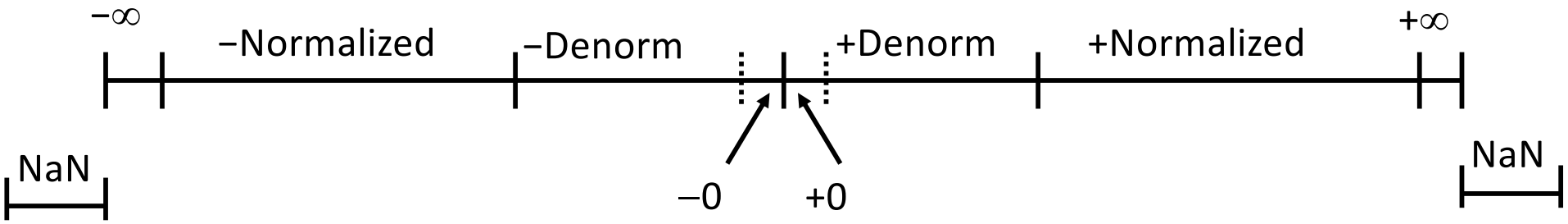
3. Special Values

Condition: **`exp`** = **`111…1`**

- Case: **`exp`** = **`111…1`**, **`frac`** = **`000…0`**
  - Represents value ∞ (infinity)
  - Operation that overflows
  - Both positive and negative
  - E.g., 1.0/0.0 = −1.0/−0.0 = +∞, 1.0/−0.0 = −∞

- Case: **`exp`** = **`111…1`**, **`frac`** ≠ **`000…0`**
  - Not-a-Number (NaN)
  - Represents case when no numeric value can be determined
  - E.g., sqrt(–1), ∞ − ∞, ∞ × 0

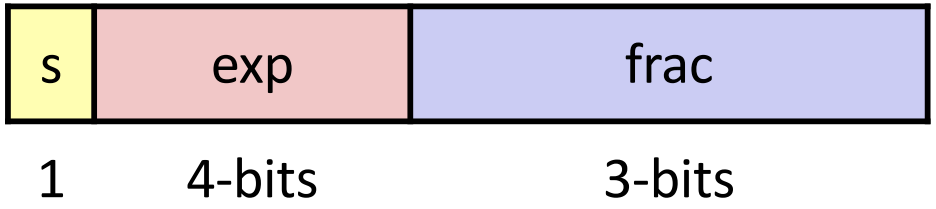
Visualization: Floating Point Encodings



Floating Point

- Background: Fractional binary numbers
- IEEE floating point standard: Definition
- Example
- Distribution of values and properties
- Rounding, addition, multiplication
- Floating point in C
- Properties of FP and integers in their computer representation
- Summary

Tiny Floating Point Example



- 8-bit Floating Point Representation
  - the sign bit is in the most significant bit
  - the next four bits are the exponent, with a bias of 7
  - the last three bits are the **`frac`**

- Same general form as IEEE Format
  - normalized, denormalized
  - representation of 0, NaN, infinity

Dynamic Range (Positive Only)

$v = (-1)^s M 2^E$

n: E = Exp – Bias

d: E = 1 – Bias

	s	exp	frac	E	Value	
Denormalized numbers	0	0000	000	-6	0	closest to zero
	0	0000	001	-6	1/8*1/64 = 1/512	
	0	0000	010	-6	2/8*1/64 = 2/512	
	…					
	0	0000	110	-6	6/8*1/64 = 6/512	
	0	0000	111	-6	7/8*1/64 = 7/512	largest denorm
Normalized numbers	0	0001	000	-6	8/8*1/64 = 8/512	smallest norm
	0	0001	001	-6	9/8*1/64 = 9/512	
	…					
	0	0110	110	-1	14/8*1/2 = 14/16	
	0	0110	111	-1	15/8*1/2 = 15/16	closest to 1 below
	0	0111	000	0	8/8*1 = 1	
	0	0111	001	0	9/8*1 = 9/8	closest to 1 above
	0	0111	010	0	10/8*1 = 10/8	
	…					
	0	1110	110	7	14/8*128 = 224	
	0	1110	111	7	15/8*128 = 240	largest norm
	0	1111	000	n/a	inf	

Denormalized Encoding Example

- Value: `float F = 1/512 = 1 x 2^-9;`
  - = 0.000000001_2
    = 0.001×2^{-6}

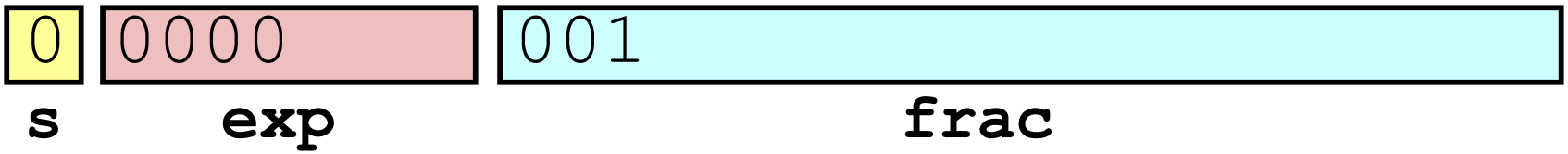
$v = (-1)^s M 2^E$
$E = 1 - \text{Bias}$

in tiny FP example

- Significand
  M = $0.\underline{001}_2$
  frac= $\underline{001}_2$ 3 bits for frac

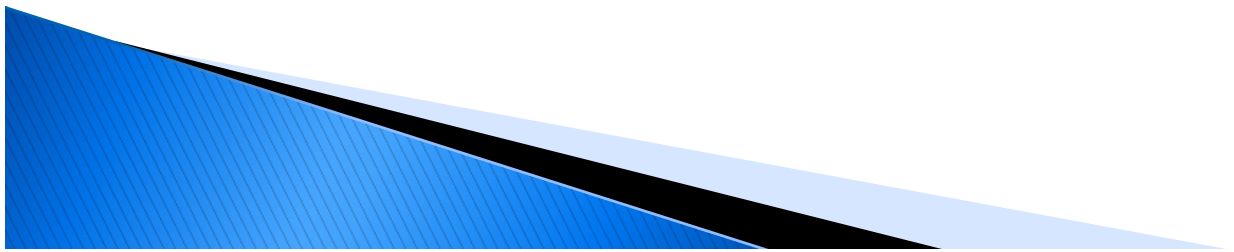
- Exponent
  E = 1 - 7 = -6
  Bias = 7 (k = 4, $2^{k-1} - 1 = 8 - 1$)
  Exp = 0 =

- Result:



How to represent 0.25?

- In the tiny 8-bit IEEE or in 32-bit IEEE
- Normalized or denormalized?
  - 8-bit? (4 bits exp, 3 bits frac)
  - 32 bit? (8 bits exp, 23 bits frac)

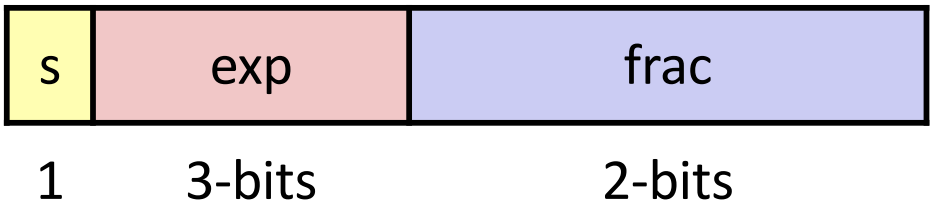


Floating Point

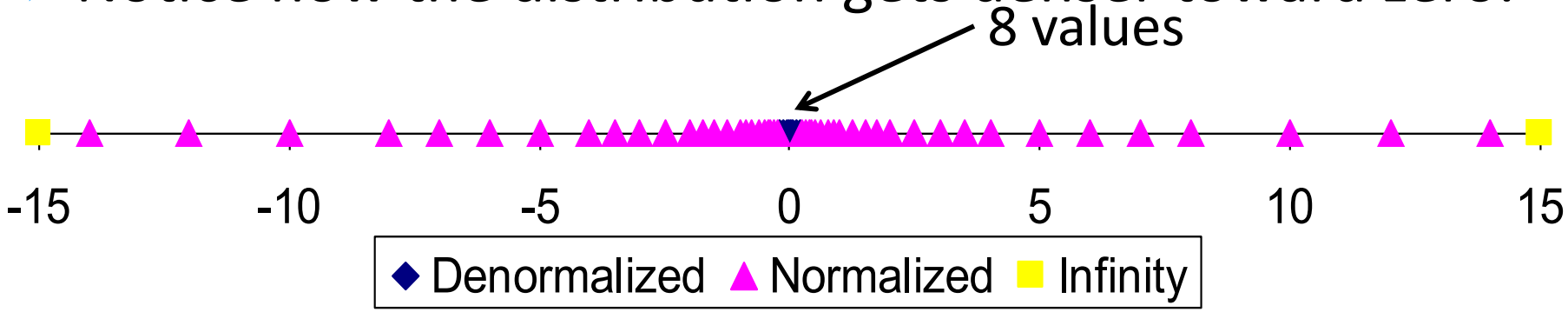
- Background: Fractional binary numbers
- IEEE floating point standard: Definition
- Example
- Distribution of values and properties
- Rounding, addition, multiplication
- Floating point in C
- Summary

Distribution of Values

- 6-bit IEEE-like format
  - e = 3 exponent bits
  - f = 2 fraction bits
  - Bias is $2^{3-1}-1 = 3$

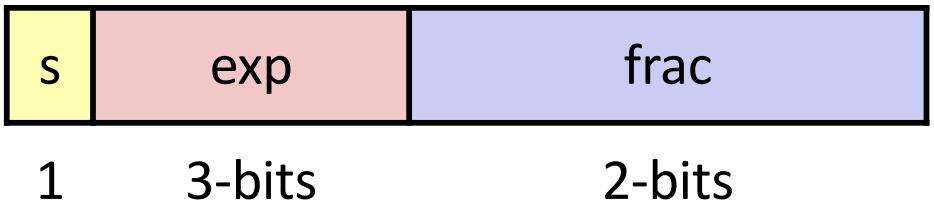


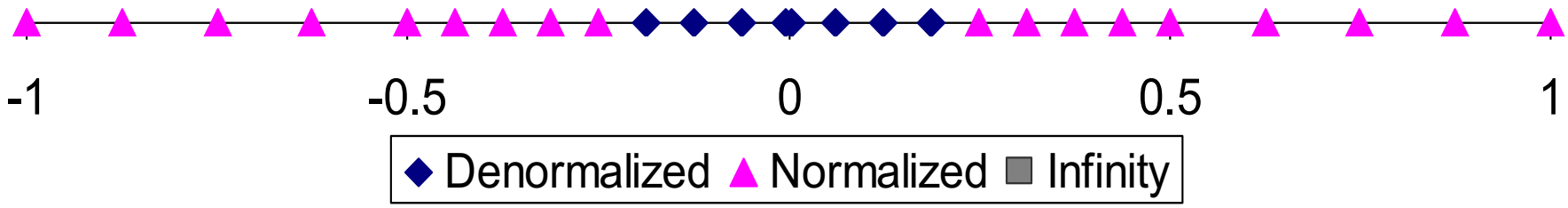
- Notice how the distribution gets denser toward zero.



Distribution of Values (close-up view: numbers between -1.0,1.0)

- 6-bit IEEE-like format
  - e = 3 exponent bits
  - f = 2 fraction bits
  - Bias is 3





Understanding FP representation

Week 3 lab sessions (Tuesday week 4 for section 002):

1. Study Fig. 2.35 p. 116 or slide 31
2. Do problem 2.47 p. 117 (5-bit FP repr.) (slowly, seeing what happens as you fill-in the table)
3. Follow descriptions of the full table on pp. 118-119

- + problem 2.88 (after trying the three steps above)

Special Properties of the IEEE Encoding

- FP Zero Same as Integer Zero
  - All bits = 0
- Can (Almost) Use Unsigned Integer Comparison (p. 119)
  - Must first compare sign bits
  - Must consider –0 = 0
  - NaNs problematic
    - Will be greater than any other values
    - What should comparison yield?
  - Otherwise OK
    - Denorm vs. normalized
    - Normalized vs. infinity