



# Floating Point - Examples

These slides adapted from materials provided by the textbook

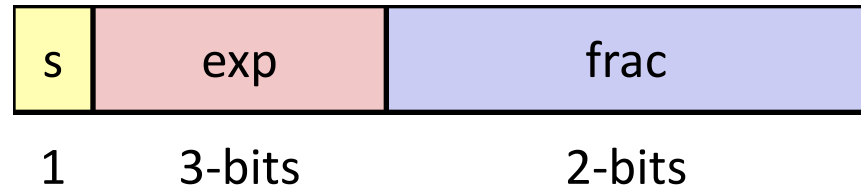
# Floating Point

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

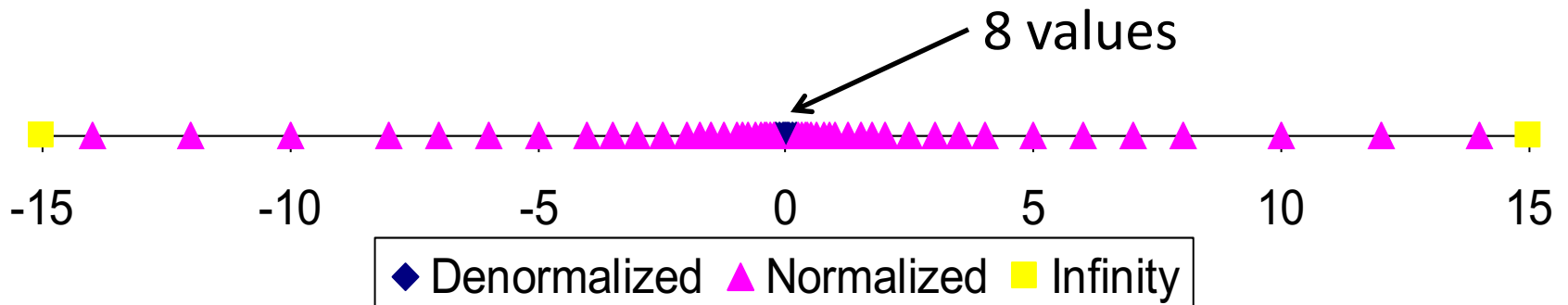
# Distribution of Values

## ■ 6-bit IEEE-like format

- $e = 3$  exponent bits
- $f = 2$  fraction bits
- Bias is  $2^{3-1}-1 = 3$



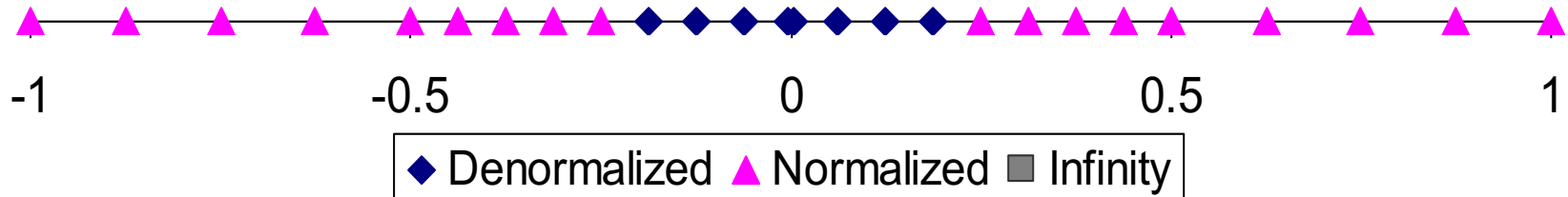
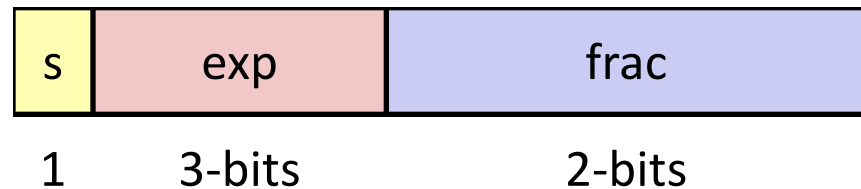
## ■ Notice how the distribution gets denser toward zero.



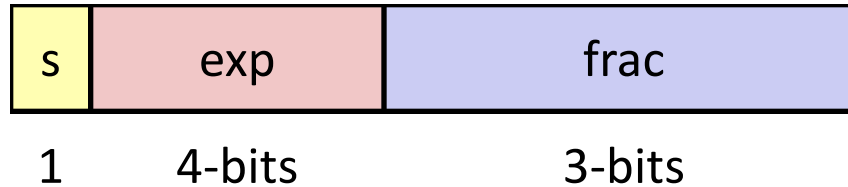
# Distribution of Values (close-up view)

## ■ 6-bit IEEE-like format

- $e = 3$  exponent bits
- $f = 2$  fraction bits
- Bias is 3



# 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:  $E = \text{Exp} - \text{Bias}$   
d:  $E = 1 - \text{Bias}$

closest to zero

largest denorm

smallest norm

closest to 1 below

closest to 1 above

largest norm

Denormalized  
numbers

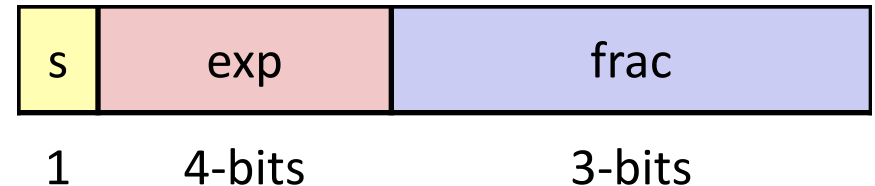
Normalized  
numbers

s	exp	frac	E	Value
0	0000	000	-6	0
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	$7/8 * 1/64 = 7/512$
0	0001	000	-6	$8/8 * 1/64 = 8/512$
0	0001	001	-6	$9/8 * 1/64 = 9/512$
...				
0	0110	110	-1	$14/8 * 1/2 = 14/16$
0	0110	111	-1	$15/8 * 1/2 = 15/16$
0	0111	000	0	$8/8 * 1 = 1$
0	0111	001	0	$9/8 * 1 = 9/8$
0	0111	010	0	$10/8 * 1 = 10/8$
...				
0	1110	110	7	$14/8 * 128 = 224$
0	1110	111	7	$15/8 * 128 = 240$
0	1111	000	n/a	inf

# Creating Floating Point Number

## ■ Steps

- Normalize to have leading 1
- Round to fit within fraction
- Postnormalize to deal with effects of rounding



## ■ Case Study

- Convert 8-bit unsigned numbers to tiny floating point format

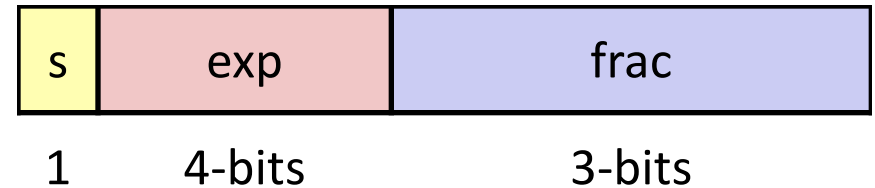
Example Numbers

128                    10000000

# Creating Floating Point Number

## ■ Steps

- Normalize to have leading 1
- Round to fit within fraction
- Postnormalize to deal with effects of rounding



## ■ Case Study

- Convert 8-bit unsigned numbers to tiny floating point format

Example Numbers

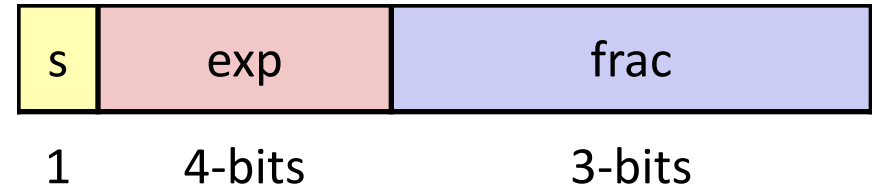
35                      00010011



# Creating Floating Point Number

## ■ Steps

- Normalize to have leading 1
- Round to fit within fraction
- Postnormalize to deal with effects of rounding



## ■ Case Study

- Convert 8-bit unsigned numbers to tiny floating point format

Example Numbers

15                      00001101

# Special Properties of the IEEE Encoding

- **FP Zero Same as Integer Zero**
  - All bits = 0
- **Can (Almost) Use Unsigned Integer Comparison**
  - 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