

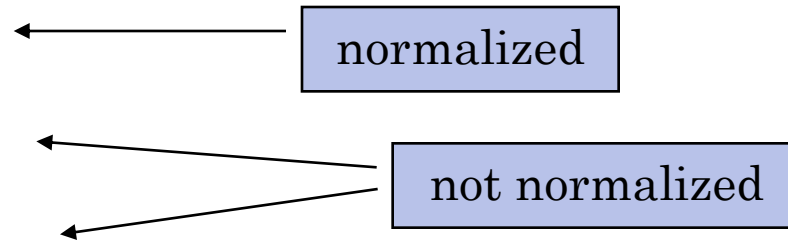


# **IEEE 754 FLOATING POINT REPRESENTATION**

**Alark Joshi**

# FLOATING POINT

- Representation for non-integral numbers
  - Including very small and very large numbers
- Like scientific notation
  - $-2.34 \times 10^{56}$
  - $+0.002 \times 10^{-4}$
  - $+987.02 \times 10^9$
- In binary
  - $\pm 1.xxxxxxx_2 \times 2^{yyyy}$
- Types `float` and `double` in C



# FLOATING POINT STANDARD

- Defined by IEEE Std 754-1985
- Developed in response to divergence of representations
  - Portability issues for scientific code
- Now almost universally adopted
- Two representations
  - Single precision (32-bit)
  - Double precision (64-bit)



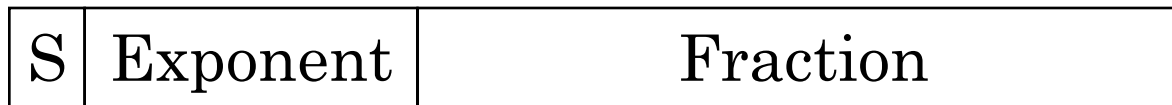
# IEEE FLOATING-POINT FORMAT

single: 8 bits

double: 11 bits

single: 23 bits

double: 52 bits



$$x = (-1)^S \times (1 + \text{Fraction}) \times 2^{(\text{Exponent} - \text{Bias})}$$

- S: sign bit (0  $\Rightarrow$  non-negative, 1  $\Rightarrow$  negative)
- Normalize significand:  $1.0 \leq |\text{significand}| < 2.0$ 
  - Significand is Fraction with the “1.” restored
  - Always has a leading pre-binary-point 1 bit, so no need to represent it explicitly (hidden bit)



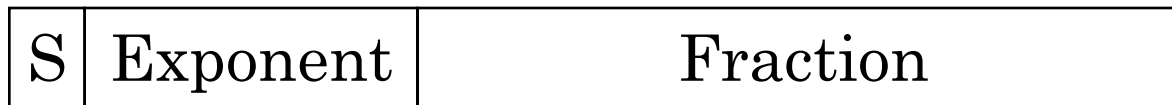
# IEEE FLOATING-POINT FORMAT

single: 8 bits

double: 11 bits

single: 23 bits

double: 52 bits



$$x = (-1)^S \times (1 + \text{Fraction}) \times 2^{(\text{Exponent} - \text{Bias})}$$

- Exponent: excess representation: actual exponent + Bias
  - Ensures exponent is unsigned
  - Single precision: Bias = 127;
  - Double precision: Bias = 1203



single: 8 bits  
double: 11 bits

single: 23 bits  
double: 52 bits

S	Exponent	Fraction
---	----------	----------

## SINGLE-PRECISION RANGE

- Exponents 00000000 and 11111111 are reserved
- Smallest value
  - Exponent: 00000001  
 $\Rightarrow$  actual exponent =  $1 - 127 = -126$
  - Fraction: 000...00  $\Rightarrow$  significand = 1.0
  - $\pm 1.0 \times 2^{-126} \approx \pm 1.2 \times 10^{-38}$
- Largest value
  - exponent: 11111110  
 $\Rightarrow$  actual exponent =  $254 - 127 = +127$
  - Fraction: 111...11  $\Rightarrow$  significand  $\approx 2.0$
  - $\pm 2.0 \times 2^{+127} \approx \pm 3.4 \times 10^{+38}$



single: 8 bits  
double: 11 bits

single: 23 bits  
double: 52 bits

S	Exponent	Fraction
---	----------	----------

## DOUBLE-PRECISION RANGE

- Exponents 0000...00 and 1111...11 are reserved
- Smallest value
  - Exponent: 000000000001  
 $\Rightarrow$  actual exponent =  $1 - 1023 = -1022$
  - Fraction: 000...00  $\Rightarrow$  significand = 1.0
  - $\pm 1.0 \times 2^{-1022} \approx \pm 2.2 \times 10^{-308}$
- Largest value
  - Exponent: 111111111110  
 $\Rightarrow$  actual exponent =  $2046 - 1023 = +1023$
  - Fraction: 111...11  $\Rightarrow$  significand  $\approx 2.0$
  - $\pm 2.0 \times 2^{+1023} \approx \pm 1.8 \times 10^{+308}$



single: 8 bits  
double: 11 bits

single: 23 bits  
double: 52 bits

S	Exponent	Fraction
---	----------	----------

# FLOATING-POINT PRECISION

## ○ Relative precision

- all fraction bits are significant
- Single: approx  $2^{-23}$ 
  - Equivalent to  $23 \times \log_{10} 2 \approx 23 \times 0.3 \approx 6$   
decimal digits of precision
- Double: approx  $2^{-52}$ 
  - Equivalent to  $52 \times \log_{10} 2 \approx 52 \times 0.3 \approx 16$   
decimal digits of precision





single: 8 bits  
double: 11 bits

single: 23 bits  
double: 52 bits

S	Exponent	Fraction
---	----------	----------

## FLOATING-POINT EXAMPLE

### ○ Represent $-0.75$

- $-0.75 = (-1)^1 \times 1.1_2 \times 2^{-1}$   
 $= -1 \times 1. \frac{1}{2} \times \frac{1}{2}$   
 $= -1.5 * .5 = -0.75$

- $S = 1$

- Fraction =  $1000\dots00_2$

- Exponent =  $-1 + \text{Bias}$

- Single:  $-1 + 127 = 126 = 01111110_2$

- Double:  $-1 + 1023 = 1022 = 01111111110_2$

- Single:  $10111111101000\dots00$

- Double:  $101111111111101000\dots00$



# FLOATING-POINT EXAMPLE

- What number is represented by the single-precision float

11000000101000...00

- $S = 1$
  - Fraction =  $01000...00_2$
  - Exponent =  $10000001_2 = 129$
- $x = (-1)^1 \times (1 + 01_2) \times 2^{(129 - 127)}$   
=  $(-1) \times 1.25 \times 2^2$   
=  $-5.0$



# EXAMPLE

- Number to IEEE 754 conversion
- <http://www.h-schmidt.net/FloatConverter/IEEE754.html>
- 127.0 – **0 10000101 111111000000000000000000**
- 128.0 – **0 10000110 000000000000000000000000**
- Check IEEE 754 representation for
  - 2.0, -2.0
  - 127.99
  - 127.99999 (five 9's)
  - What happens with 127.999999 (six 9's) and 3.999999 (six 9's)

