CENG 3420
Computer Organization & Design

Lecture 07: Floating Numbers

Bei Yu
CSE Department, CUHK
byu@cse.cuhk.edu.hk

(Textbook: Chapter 3.5)

2024 Spring

Scientific notation: 6.6254×10^{-27}

- A normalized number of certain accuracy (e.g. 6.6254 is called the mantissa)
- Scale factors to determine the position of the decimal point (e.g. 10^{-27} indicates position of decimal point and is called the exponent; the **base** is implied)
- Sign bit

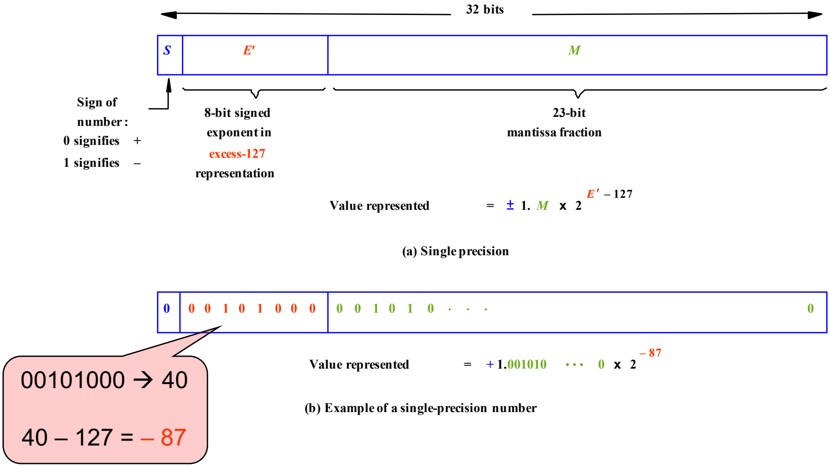
Normalized Form

- Floating Point Numbers can have multiple forms, e.g.

$$
\begin{aligned}
0.232 \times 10^4 &= 2.32 \times 10^3 \\
&= 23.2 \times 10^2 \\
&= 2320. \times 10^0 \\
&= 232000. \times 10^{-2}
\end{aligned}
$$

- It is desirable for each number to have a unique representation => Normalized Form
- We normalize Mantissa's in the Range $[1..R)$, where R is the Base, e.g.:
  - $[1..2)$ for BINARY
  - $[1..10)$ for DECIMAL

32-bit, float in C / C++ / Java

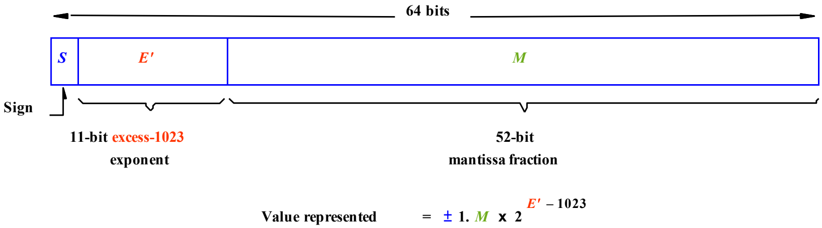


Note:

- minimum exponent: 1 - 127 = -126
- maximum exponent: 254 - 127 = 127
- Why 254? If exponents are all 1, the floating num has special values (please refer to following part)

IEEE Standard 754 Double Precision

64-bit, float in C / C++ / Java



(c) Double precision

Question:

What is the IEEE single precision number $40C0\ 0000_{16}$ in decimal?

Question:

What is the IEEE single precision number 40C0 0000_{16} in decimal?

- Binary: 0100 0000 1100 0000 0000 0000 0000 0000
- Sign: +
- Exponent: 129 - 127 = +2
- Mantissa: 1.100 0000 $..._2 \rightarrow 1.5_{10} \times 2^{+2}$
- $\rightarrow$ +110.0000 $..._2$
- Decimal Answer = $+6.0_{10}$

Question:

What is -0.5_{10} in IEEE single precision binary floating point format?

Question:

What is -0.5_{10} in IEEE single precision binary floating point format?

- Binary: 1.0... $\times 2^{-1}$ (in binary)
- Exponent: 127 + (-1) = 01111110
- Sign bit: 1
- Mantissa: 1.000 0000 0000 0000 0000 0000
- Binary representation: 1011 1111 0000 0000 0000 0000 0000 0000

Special Values

Exponents of all 0's and all 1's have special meaning

- E=0, M=0 represents 0 (sign bit still used so there is ± 0)
- E=0, M$\neq$0 is a denormalized number $\pm 0.\mathrm{M} \times 2^{-126}$ (smaller than the smallest normalized number)
- E=All 1's, M=0 represents $\pm$`Infinity`, depending on Sign
- E=All 1's, M$\neq$0 represents `NaN`

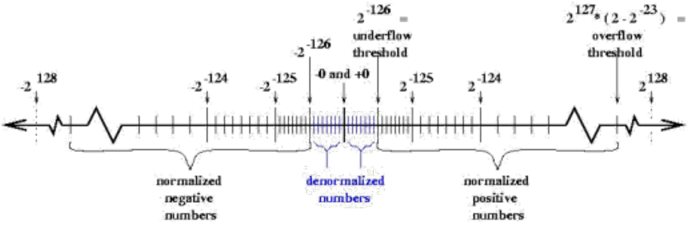
Ref: IEEE Standard 754 Numbers

- Normalized $\pm$ 1.d…d x 2^{exp}
- ***Denormalized*** $\pm$ 0.d…d x 2^{min_exp} → to represent *near-zero* numbers
  e.g. + 0.0000…0000001 x 2^{-126} for Single Precision

Format	# bits	# significant bits	macheps	# exponent bits	exponent range
Single	32	1+23	2^{-24} ($\sim 10^{-7}$)	8	$2^{-126} - 2^{+127}$ ($\sim 10^{\pm 38}$)
Double	64	1+52	2^{-53} ($\sim 10^{-16}$)	11	$2^{-1022} - 2^{+1023}$ ($\sim 10^{\pm 308}$)
Double Extended	>=80	>=64	$\le 2^{-64}$ ($\sim 10^{-19}$)	>=15	$2^{-16382} - 2^{+16383}$ ($\sim 10^{\pm 4932}$)

(Double Extended is *80 bits on all Intel machines*)

macheps = Machine Epsilon = ε_{mach} **=** $2^{-(\text{\# significand bits})}$



Note:

- Smallest normalized: $1.000\ 0000\ \ldots\ 0000_2 \times 2^{-126} = 2^{-126}$
- Largest denormalized: $\mathbf{0}.111\ 1111\ \ldots\ 1111_2 \times 2^{-126} = (1 - 2^{-1/23}) \times 2^{-126}$
- Smallest denormalized: $\mathbf{0}.000\ 0000\ \ldots\ 0001_2 \times 2^{-126} = 2^{-149}$
- Smallest denormalized value is much closer to 0

Inaccurate Floating Point Operations

- E.g. Find 1st root of a quadratic equation
  - r = (–b + sqrt(b*b – 4*a*c)) / (2*a)

Sparc processor, Solaris, gcc 3.3 (ANSI C),

```
Expected Answer   0.00023025562642476431
double            0.00023025562638524986
float             0.00024670246057212353
```

- Problem is that if c is near zero,

$$\mathrm{sqrt}(b*b - 4*a*c) \approx b$$

- Rule of thumb: use the highest precision which does not give up too much speed

Catastrophic Cancellation

- (a – b) is inaccurate when a ≈ b
- Decimal Examples
  - Using 2 significant digits to compute mean of 5.1 and 5.2 using the formula (a+b) / 2:

    a + b = 10 (with 2 sig. digits, 10.3 can only be stored as 10)
    10 / 2 = 5.0 (the computed mean is less than both numbers!!!)
  - Using 8 significant digits to compute sum of three numbers:

    $$(11111113 + (-11111111)) + 7.5111111 = 9.5111111$$

    $$11111113 + ((-11111111) + 7.5111111) = 10.000000$$
- Catastrophic cancellation occurs when

$$\left| \frac{[round(x) \text{"}\bullet\text{"} round(y)] - round(x \bullet y)}{round(x \bullet y)} \right| >> \varepsilon_{mach}$$