CENG 3420
Computer Organization & Design
Lecture 07: Floating Point

Textbook: Chapter 3.5

Zhengrong Wang

CSE Department, CUHK

zhengrongwang@cuhk.edu.hk

Float Num

Scientific Notation

- To represent 123.456 as 1.23456×10^2
  - A normalized number of certain accuracy (1.23456 is called mantissa).
  - Scale factors to determine the position of the decimal point (10^2 indicates position of decimal point and is called the exponent; the base is implied).
  - Sign bit.

Normalized Form

- Scientific notation can have more than one way to write:

$$123.456 = 1.23456 \times 10^2 = 12.3456 \times 10^1 = 0.123456 \times 10^3$$

- The decimal point is moving – Floating point number.
- We prefer normalized form: mantissa within range $[1, Base)$
  - For decimal, $[1, 10)$
  - For binary, $[1, 2)$

IEEE Standard 754 Single Precision

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

S	E (8-bit)	M (23-bit)

  - 1-bit sign S.
  - 8-bit signed exponent E – 127.
  - 23-bit mantissa M.

$$(-1)^S(1.M) \times 2^{E-127}$$

- Example: $-3 = -1.5 \times 2^1 = -1.1_2 \times 2^1$
  - S = 1.
  - E = 128.
  - M = 1.

1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IEEE Standard 754 Double Precision

- 64-bit, double in C/C++/Java
  - 1-bit Sign.
  - 11-bit Exponent: E – 1023.
  - 52-bit Mantissa M.

$$(-1)^S(1.M) \times 2^{E-1023}$$

Exercise

- How to represent $40C00000_{16}$ in float?

- How to represent -0.5_{10} in float?

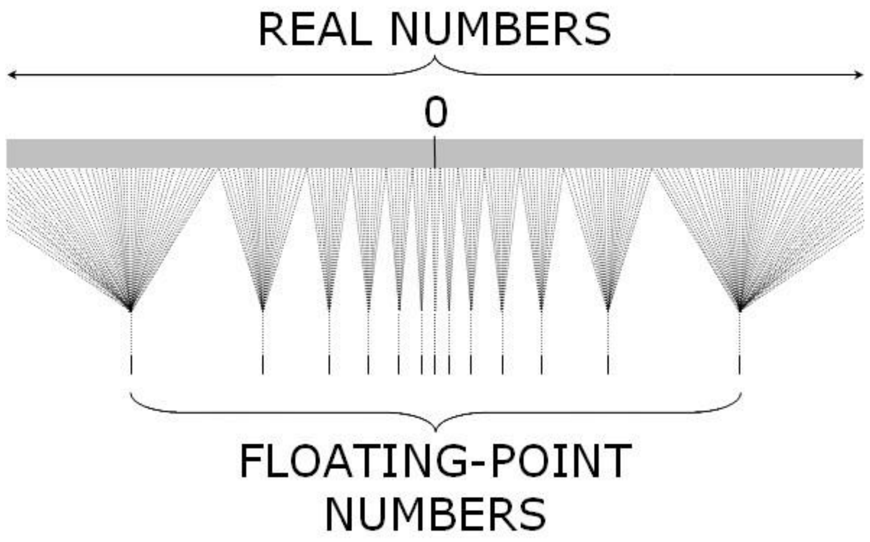
Special Values

- 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 $\pm(0.M) \times 2^{-126}$ (smaller than the smallest normalized number).
  - $E = All\ 1s, M = 0$ represents $\pm$Infinity, depending on Sign.
  - $E = All\ 1s, M \neq 0$ represents NaN (Not a Number).

Approximation of Floating Point Number

- Float point number approximates real number.
  - We only have 32-bit/64-bit.

- This causes precision loss for computation!
- E.g., 0.3 can not be precisely represent in binary.




```
# Python terminal:
>>> format(0.3, ".55f")
'0.2999999999999999888977697537484345957636833190917968750'
>>> format(0.5, ".55f")
'0.5000000000000000000000000000000000000000000000000000000'
```

Rounding

- IEEE standard 754 defined 4 rounding modes:
  - Round up to +infinity.
  - Round down to –infinity.
  - Truncate (get rid of extra mantissa bits → round to zero).
  - Round to closet even (default mode for C/C++).
- Round to closet even to break the tie of 0.5.
  - E.g., we only have 3 digit of mantissa, 3.555 → 3.56, 3.445 → 3.44.

- Fun fact: U.S. Internal Revenue Service (IRS) always round up 0.5 to 1.
  - To collect more tax!
- Hong Kong Inland Revenue Department (IRD) always round down to HK$1.
  - Even if you have HK$1.99 → 1 :)
- This is not financial advice...

Floating Point in AI

- Today large language models (LLM) have billions of parameters.
  - E.g., DeepSeek V3 has 671 billion parameters. ChatGPT-5 is similar.
- Huge cost to perform double/single precision on big models:
  - Higher memory storage to store the parameters.
  - Longer latency to load parameters from memory to registers.
  - Longer computation time.
  - High energy cost → Big AI companies are building in-house power station.
- AI does not need high precision floating point! → Quantization
  - From 32-bit to 16-bit, 2x speedup without performance loss.
  - DeepSeek-V3 is first to deploy 8-bit training and inference.

Quantization in AI

- Many quantization formats for AI.

Format	Bit Width	Structure (S/E/M)	Use Cases
FP64	64 bits	1/11/52	Scientific computing, simulations
FP32	32 bits	1/8/23	General-purpose computing, DL training
TF32	32 bits	1/8/10 (stored in 32 bits)	NVIDIA AI training (Ampere Tensor Cores)
BF16	16 bits	1/8/7	TPU/GPU training and inference
FP16	16 bits	1/5/10	Mobile inference, memory-efficient DL
FP8	8 bits	E4M3 or E5M2	Quantized neural nets, edge inference
MXFP4	4 bits	Vendor-specific (~1/2/1)	Ultra-low-power experimental AI inference

  - Try to ask ChatGPT what is its bit width :)
- Choose the precision based on application.
  - E.g., high precision for chemical simulation, low precision for AI inference.