Computer Organization

COMP2120

Qi Zhao

January 24, 2024

Number Representation and arithmetic Part II

Floating-point representation

- Fixed-point representation. A fractional number is essentially an integer that is to be implicitly multiplied by a fixed scaling factor.
- Cannot represent large numbers or very small fractions.
- Scientific notation:

$$\text{Value} = \pm\text{Significand} \times 10^{\pm\text{Exponent}}$$

- Similar to a decimal floating point number, a binary floating point number can be represented as

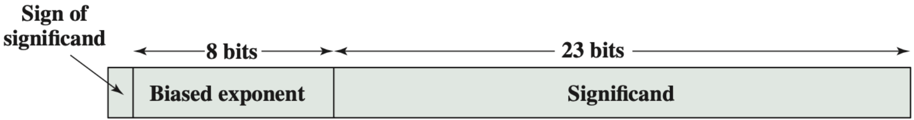
$$\pm(1.\text{xxxxx}) \times 2^{\text{Exponent}} \text{ or } \pm(0.1\text{xxxxx}) \times 2^{\text{Exponent}}$$

Floating-point representation

How to represent a floating-point number?

- a sign bit $\pm$
- **Significant**: normalized to $1.\mathrm{xxxxx}$ or $0.1\mathrm{xxxxx}$. The first digit is always 1, which can be omitted.
- **Exponent**: some signed number representation format (biased representation or excess $2^{m-1} - 1$)

Floating-point representation



(a) Format

$$
\begin{array}{rcl}
1.1010001 \times 2^{10100} & = 0\ 10010011\ 10100010000000000000000 = & 1.6328125 \times 2^{20} \\
-1.1010001 \times 2^{10100} & = 1\ 10010011\ 10100010000000000000000 = & -1.6328125 \times 2^{20} \\
1.1010001 \times 2^{-10100} & = 0\ 01101011\ 10100010000000000000000 = & 1.6328125 \times 2^{-20} \\
-1.1010001 \times 2^{-10100} & = 1\ 01101011\ 10100010000000000000000 = & -1.6328125 \times 2^{-20}
\end{array}
$$

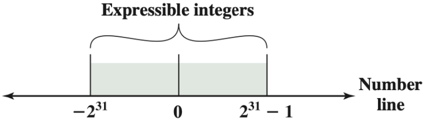
(b) Examples

Figure 10.18 Typical 32-Bit Floating-Point Format

- Biased exponent, excess 127 representation. $10010011 = 2^7 + 2^4 + 2 + 1 = 147$, $147 - 127 = 20$

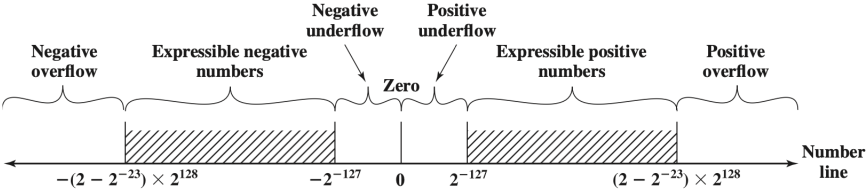
Expressible Numbers

- Two complement integers



(a) Twos complement integers

- Floating-point numbers
  Positive number between 2^{-127} and
  $(1 + 2^{-1} + \cdots + 2^{-23}) \times 2^{128} = (2 - 2^{-23}) \times 2^{128}$



(b) Floating-point numbers

Expressible Numbers

- Compared to two's complement integer representation, does the floating-point method represent more values? No, 2^{32}.
- Do the represented numbers spaced evenly along the number line? No

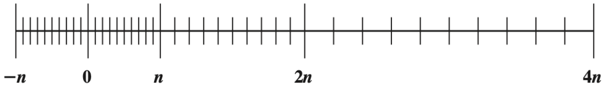


Figure 10.20 Density of Floating-Point Numbers

- Trade-off between range and precision.

IEEE Floating Point Format

Single-, double-, and quadruple-precision floating point format

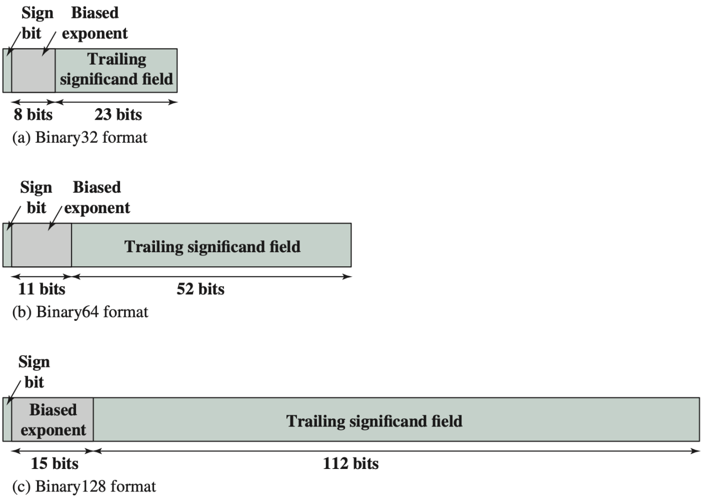


Figure 10.21 IEEE 754 Formats

IEEE Floating Point Format

Table 10.3 IEEE 754 Format Parameters

Parameter	Format		
	Binary32	**Binary64**	**Binary128**
Storage width (bits)	32	64	128
Exponent width (bits)	8	11	15
Exponent bias	127	1023	16383
Maximum exponent	127	1023	16383
Minimum exponent	-126	-1022	-16382
Approx normal number range (base 10)	$10^{-38}, 10^{+38}$	$10^{-308}, 10^{+308}$	$10^{-4932}, 10^{+4932}$
Trailing significand width (bits)*	23	52	112
Number of exponents	254	2046	32766
Number of fractions	2^{23}	2^{52}	2^{112}
Number of values	1.98×2^{31}	1.99×2^{63}	1.99×2^{128}
Smallest positive normal number	2^{-126}	2^{-1022}	2^{-16362}
Largest positive normal number	$2^{128} - 2^{104}$	$2^{1024} - 2^{971}$	$2^{16384} - 2^{16271}$
Smallest subnormal magnitude	2^{-149}	2^{-1074}	2^{-16494}

Note: * Not including implied bit and not including sign bit.

IEEE Floating Point Format

- Exponent uses excess-127 (1023). Actual value= $E - 127, (1023)$.
- S Sign bit, E biased exponent, M significant.
- After normalization, the actual value of significand is $1.M$.
- Actual value is $(-1)^S \times 1.M \times 2^{E-127}$ (Single precision).
- Special Values:
  The pattern of all 0 ($000\ldots00$) and all 1 $(111\ldots11)$, or $\mathrm{FF}\ldots\mathrm{F}$ in the exponent field has special meaning.
  $0\mathrm{x}00$: not $0 - 127 = -127$
  $0\mathrm{xFF}$: not $255 - 127 = 128$
  $0\mathrm{x}$ means hexadecimal.
- Valid values in the exponent field is from -126 to 127 for single precision.

IEEE Floating Point Format

Exponent= $000\ldots0$

- Fraction $= 000\ldots0$ represents values of 0
  we have distinct $+0$ and -0.
- Fraction $f \neq 000\ldots0$, subnormalized

$$(-1)^S \times 0.f \times 2^{-126}$$

It is -126 not -127, why?
Smallest positive normalized is

$$1.000\ldots0 \times 2^{1-127} = 1.000\ldots0 \times 2^{-126}$$

If we use -127,

$$0.111\ldots1 \times 2^{-127}$$

use -126, closer to normalized number, largest positive subnormalized is

$$0.111\ldots1 \times 2^{-126}$$

IEEE Floating Point Format

Exponent= $111\dots1$

- Fraction $= 000\dots0$
  Represents value of infinity ∞
  Result returned for operations that overflow
  Sign indicates positive or negative, $+\infty$, $-\infty$
  E.g., $1.0/0.0 = -1.0/-0.0 = +\infty$, $1.0/-0.0 = -\infty$.
- Fraction $\neq 000\dots0$
  Not-a-Number (NaN)
  Represents the case when no numeric value can be determined
  E.g., $\sqrt{-1}$, $\infty - \infty$

Example

Binary 32-bit format:
1 bit sign, 8 bits for exponent, 23 bits for significand

- Write down the bit pattern corresponding to the value 13.375
- Write down the value corresponding to the bit pattern $3C05D00$

Example

Binary 32-bit format:
1 bit sign, 8 bits for the exponent, 23 bits for significand

- $13 = 1101_2$, $0.375 = 0.011_2$, $13.375 = 1101.011_2$
- normalize it, $1101.011 = 1.101011 \times 2^3$
- Significand= 101 0110 0000 0000 0000 0000
- Exponent $= 3 + 127 = 130 = 1000\,0010_2$.
- Bit pattern =

$$0100\;0001\;0101\;0110\;0000\;0000\;0000\;0000$$

$$=41560000_{16}$$

Example

Binary 32-bit format:
1 bit sign, 8 bits for the exponent, 23 bits for significand

- c0f21000

$$
\begin{aligned}
&= 1100\ \ 0000\ 1111\ 0010\ 0001\ 0000\ 0000\ 0000 \\
&= \underbrace{1}\ \ \underbrace{100\ \ 0000\ 1}\ \ \underbrace{111\ 0010\ 0001\ 0000\ 0000\ 0000}
\end{aligned}
$$

- Sign bit$=$ 1, Negative
- Exponent $100\ \ 0000\ 1_2 - 127 = 2$
- Significand $= 1.111\ 0010\ 0001\ 0000\ 0000\ 0000$
- Value$= -1.111\ 0010\ 0001 \times 2^2 = -7.5645$