IEEE Floating Points

Single Format: (32)

$$\pm \mid a_1, a_2, \ldots a_8 \mid b_1, b_2, \ldots b_{23}$$

$\pm$ refers to the sign, 0 for positive, 1 for negative

Double Format: (64)

$$\pm \mid a_1, a_2, \ldots a_{11} \mid b_1, b_2, \ldots b_{52}$$

Hidden bit normalization: don't store b_0, as we know $b_0 = 1$

IEEE Single format

$\pm$	$a_1 a_2 a_3 \ldots a_8$	$b_1 b_2 b_3 \ldots b_{23}$

If exponent $a_1 \ldots a_8$ is	Then value is
$(00000000)_2 = (0)_{10}$	$\pm(0.b_1..b_{23})_2 \times 2^{-126}$
$(00000001)_2 = (1)_{10}$	$\pm(1.b_1..b_{23})_2 \times 2^{-126}$
$(00000010)_2 = (2)_{10}$	$\pm(1.b_1..b_{23})_2 \times 2^{-125}$
$(00000011)_2 = (3)_{10}$	$\pm(1.b_1..b_{23})_2 \times 2^{-124}$
$\downarrow$	$\downarrow$
$(01111111)_2 = (127)_{10}$	$\pm(1.b_1..b_{23})_2 \times 2^{0}$
$(10000000)_2 = (128)_{10}$	$\pm(1.b_1..b_{23})_2 \times 2^{1}$
$\downarrow$	$\downarrow$
$(11111100)_2 = (252)_{10}$	$\pm(1.b_1..b_{23})_2 \times 2^{125}$
$(11111101)_2 = (253)_{10}$	$\pm(1.b_1..b_{23})_2 \times 2^{126}$
$(11111110)_2 = (254)_{10}$	$\pm(1.b_1..b_{23})_2 \times 2^{127}$
$(11111111)_2 = (255)_{10}$	$\pm\infty$ if $b_1, \ldots, b_{23} = 0$; NaN otherwise.

The exponent representation $a_1, a_2, \ldots, a_8$ uses **biased representation**: this bit-string is the binary presentation of $E + 127$. 127 is the **exponent bias**.

$$127 = (111111110)_2/2 = (2^8 - 1 - 1)/2 = 127$$

- **Smallest positive normal number** is

$$(1.00\ldots0)_2 \times 2^{-126}$$

$$0 \mid 00\ldots1 \mid 00000\ldots0$$

- **Largest positive normal number** is

$$(1.11\ldots1)_2 \times 2^{127}$$

$$0 \mid 11\ldots10 \mid 1111\ldots1$$

Subnormal Numbers:

Subnormal Numbers are in the form:

$$0.b_1b_2,\ldots b_{23} \times 2^{-126}$$

Smallest Positive number we can store:

$$0 \mid 000\ldots 0 \mid 000000\ldots 01 = 2^{-23} \times 2^{-126} = 2^{-149}$$

Subnormal numbers can't be normalized as the exponent field won't fit.
Subnormal numbers have less accuracy as the less room for non-zero bits in the fraction.

$\pm\infty$ and NaN:

This shows an exponent bit-string of all ones is a special pattern for $\pm\infty$ or NaN, depending on the value of the fraction.
if $b_1 = b_2 = \ldots = b_{23} = 0 \implies \pm\infty$

a quite NaN (qNaN) if $b_1 = 1$ and a signalling NaN (sNaN) if $b_1 = 0$.

Machine Epsilon:

Definition: The **gap** between the number **1** and **the next larger** floating point number is called the machine epsilon of the floating point system, denoted by ε.

The number of bits in the significant (including the hidden bit) is called the **precision of the floating point system** , denoted by p.

In the **single format** system, the number after 1 is

$$b_0 \;.\; b_1b_2b_3\ldots b_{23} = 1 \;.\; 000\ldots 1$$

So, Machine Epsilon is 2^{-23}

In the **double format** system, the number after 1 is

$$b_0 \;.\; b_1b_2b_3\ldots b_{52} = 1 \;.\; 000\ldots 1$$

So, Machine Epsilon is 2^{-52}

GAP:
Let $x = m \times 2^E$ be a single format number with $1 \leq m < 2$. The gap between x and the next single format number is

$$\varepsilon \times 2^E$$

Rounding:

- Round down: $round(x) = x_-$
- Round up: $round(x) = x_+$
- Round towards zero: $round(x)$ is either x_- or x_+, whichever is between zero and x.
- Round to nearest: $round(x)$ is either x_- or x_+, whichever is nearer to x.
  In the case of a tie, the one with **its least significant bit equal to zero** is chosen

Absolute Rounding Error:

Definition: The absolute rounding error associated with x:

$$|round(x) - x|$$

For all modes, we obviously have $|round(x) - x| < |x_+ - x_-|$

Suppose $N_{min} \leq x \leq N_{max}$,

$$x = (b_0.b_1b_2\ldots b_{22}b_{23}b_{24}b_{25}\ldots)_2 \times 2^E, b_0 = 1$$

.

$$\text{IEEE single } x_- = (b_0.b_1b_2\ldots b_{22}b_{23})_2 \times 2^E, b_0 = 1$$

$$\text{IEEE single } x_+ = x_- + 0.00\ldots001 \times 2^E$$

So for any mode:

$$|round(x) - x| < |x_+ - x_-| = 0.00\ldots001 \times 2^E = 2^{-23} \times 2^E = \epsilon \times 2^E$$

Question: Is this the same for subnormal numbers?

Relative Rounding Error:

Definition: The **relative rounding error** is defined by $|\delta|$, where

$$|\delta| = |\frac{round(x) - x}{x}|$$

$$|\frac{round(x) - x}{x}| \begin{cases} < \varepsilon & \text{Any mode} \\ \leq \dfrac{\varepsilon}{2} & \text{the nearest} \end{cases} \tag{1}$$

Question: How to prove this?
NOTE: condition: x is in the normal range

IEEE for Rounded Arithmetic

$$x \ominus y = round(x - y)$$

According to the relative rounding errors, we have:

$$x \ominus y = round(x - y) = (x - y) \cdot (1 + \delta)$$

Exception Cases:

- $\frac{a}{0} \implies \infty$
- $a \times \infty \implies \infty$
- $a + \infty \implies \infty$
- $a - \infty \implies -\infty$
- $\frac{a}{\infty} \implies 0$
- $\infty + \infty \implies \infty$
- $\infty \times 0 \implies NaN$
- $\frac{0}{0} \implies NaN$
- $\frac{\infty}{\infty} \implies NaN$
- $\infty - \infty \implies NaN$

We stated before that there are two types of NaN: qNaN and sNaN. Their only difference is that sNaN generates interruption while qNaN does not. The application decides if it generates qNaN or sNaN.

Overflow and Underflow:

Overflow is said to occur when

$$N_{max} < |\text{ true result }| < \infty$$

where N_{max} is the largest normal FPN.

Two **pre-IEEE** standard treatments:
(i) Set the result to ($\pm$) N_{max}, or
(ii) Interrupt with an **error message**.
In IEEE arithmetic, the standard response depends on the **rounding mode**:
Suppose that the overflowed value is **positive**. Then

rounding model	result
round up	∞
round down	N_{max}
round towards zero	N_{max}
round to nearest	∞

Round to nearest is the **default** rounding mode and any other choice may lead to very misleading final computational results.

Underflow is said to occur when

$$0 < |\text{ true result }| < N_{min}$$

where N_{min} is the minimum normal FPN.

Historically the response was usually: **replace the result by zero**.

In **IEEE arithmetic**, the result may be a **subnormal** number instead of zero. This allows results **much smaller** than N_{min}. But there may still be a significant loss of accuracy, since subnormal numbers have fewer bits of precision.

IEEE Standard Response to Exceptions

Invalid Opn.	Set result to NaN
Division by 0	Set result to $\pm\infty$
Overflow	Set result to $\pm\infty$ or $\pm N_{max}$
Underflow	Set result to ± 0, $\pm N_{\text{min}}$ or subnormal
Inexact	Set result to correctly rounded value