1. Explain the Structure

IEEE 754 defines a structure and behaviors of floating-point number, which is a way of representing real number in computer.

In single precision (32 bit) format, a floating-point number is divided in 3 parts: 1 bit sign, 8 bit exponent, 23-bit mantissa.

Sign bit indicates the positive or negative of the value. 0 for positive, 1 for negative.

Exponent represent the power of 2 used to scale the number, ranging from value 0 to 255. The exponent is biased by 127. The actual exponent value is calculated by subtracting the bias from the stored exponent value.

Mantissa is the precision bit of the number, representing the fractional part of the number, effectively representing the digits of the number.

1. Normalization

Normalized floating point number provide maximum precision because the leading 1 doesn’t need to be stored, and ensuring a unique representation for every non zero number.

1. Zero and Special Values

| **Value** | **Sign Bit** | **Exponent (8 bits)** | **Mantissa (23 bits)** | **Description** |
| --- | --- | --- | --- | --- |
| **+0 / -0** | 0 or 1 | All 0s (00000000) | All 0s | Zero (positive or negative) |
| **Infinity** | 0 or 1 | All 1s (11111111) | All 0s | Positive/Negative Infinity |
| **NaN** | 0 or 1 | All 1s (11111111) | Non-zero | Not a Number (e.g. 0/0, ∞ - ∞) |
| **Subnormal** | 0 or 1 | All 0s | Non-zero | Denormalized numbers near zero |

1. Bias in Exponent

The exponent field uses a bias to represent both positive and negative exponents using only positive bit patterns. The bias is calculated using the formula 2^(k - 1) - 1, where k is the number of bits in the exponent field. For 32-bit single precision (k = 8), the bias is 2^7 - 1 = 127. For 64-bit double precision, which uses 11 exponent bits, the bias is 2^10 - 1 = 1023. This bias allows the stored exponent values to be adjusted back to their actual form by subtracting the bias