**ARM ARCHITECTURE**

**Assignment 3**

**IEEE 754 Floating Point Standard:-**

**1) Precision:-**

The smallest change that can be represented in floating point representation is called as precision. Precision defines how close the value is to the original value after representing in either any of the precision formats.

A floating-point format is specified by:

1. Sign :- indicates whether the no. is +ve or -ve
2. Exponent: - Range will be usually defined by exponent.
3. Significand or fraction: - This part in the format is mainly responsible for the precision of the floating point number.

To increase the precision of the significand, the IEEE 754 Standard uses a normalized significand which implies that its most significant bit is always 1.

Ex: - Assume the no. is 0.6742= 6.742\* 10^-1

And if we represent it as 0.674=6.74 \* 10^-1 or as 0.6743 = 6.743\*10^-1

Then the error will be 0.6742-0.674=0.0002 or 0.6743-0.6742=0.0001

So as we take more and more bits in fractional part value will be closer to the actual one.

**2) Normal and Subnormal values:-**

Normal: - The normalized significand is 1.f (binary dot). Normal floating number doesn’t have leading zeros before the actual value. The value of a normalized number is

**(– 1) ^s × (1.f) ×2^ (exponent – 127)**

Subnormal: - When all the exponent bits are 0 and the leading hidden bit of the significand is 0, then the floating point number is called a subnormal number.

The interpretation of a subnormal number is different. The content of the exponent part ( e ) is zero and the significand part ( f) is non-zero. The value of a subnormal number is

**(–1)^s × (0.f) ×2^ (–127) (all 0s for the exponent)**

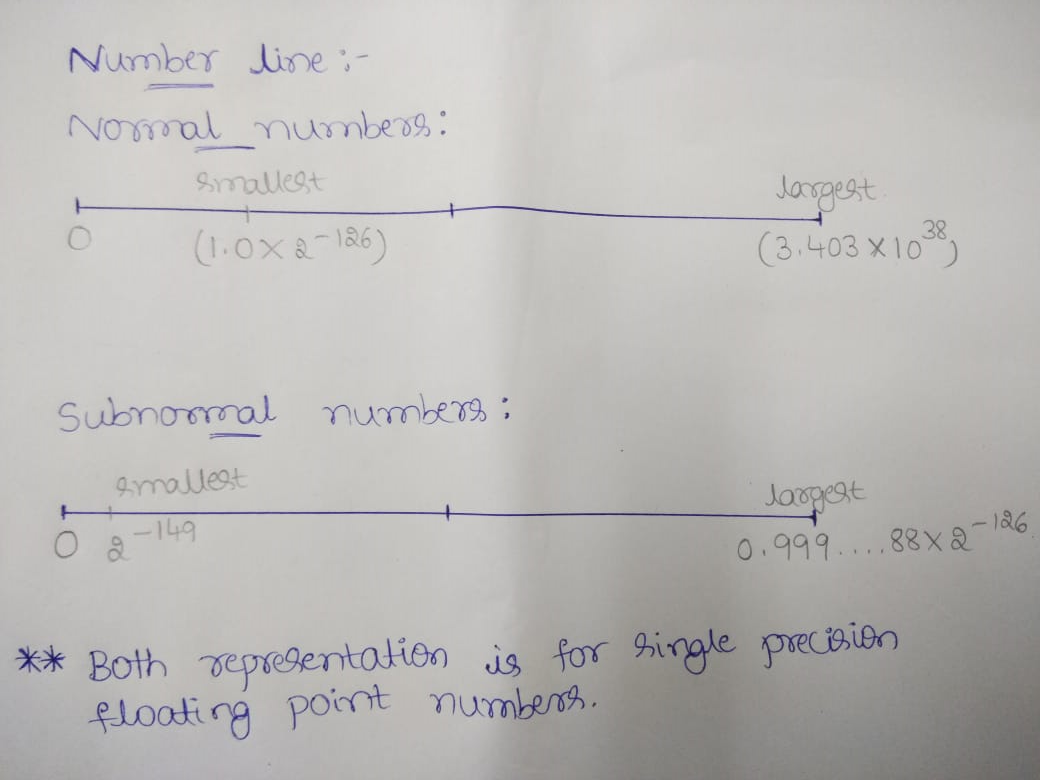
There is no implicit one in the significand.

**\*\***Both representation is for single precision floating point numbers.

Ex: - 0.625 can be represented using normal and subnormal representation:

Normal: 1.01\*2^ (-1)

Subnormal: 0.00101\*2^ (-2)



**3) 5 methods of rounding off floating number:-**

The first two rules round to a nearest value. The others are called directed roundings.

In round to nearest there are two types:

1. [**Round to nearest, ties to even**](https://en.wikipedia.org/wiki/Rounding#Round_half_to_even) – rounds to the nearest value. If the number falls midway, it is rounded to the nearest value with an even least significant digit. This is the default for binary floating point and the recommended default for decimal.
2. [**Round to nearest, ties away from zero**](https://en.wikipedia.org/wiki/Rounding#Round_half_away_from_zero) – rounds to the nearest value. If the number falls midway, it is rounded to the nearest value above (for positive numbers) or below (for negative numbers). This is intended as an option for decimal floating point.

In direct roundings there are 3 types:

1. **Round toward 0**  –> directed rounding towards zero (also known as truncation).
2. **Round toward +∞**  –> directed rounding towards positive infinity (also known as rounding up or ceiling).
3. **Round toward −∞**  –> directed rounding towards negative infinity (also known as rounding down or floor).

Table shows example of rounding to integers using the IEEE 754 rules:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Mode/Example Value** | **15.5** | **16.5** | **-15.5** | **-16.5** |
| to nearest, ties to even | 16.0 | 16.0 | -16.0 | -16.0 |
| to nearest, ties away from zero | 16.0 | 17.0 | -16.0 | -17.0 |
| toward 0 | 14.0 | 16.0 | -15.0 | -16.0 |
| toward +∞ | 16.0 | 17.0 | -15.0 | -16.0 |
| toward -∞ | 14.0 | 16.0 | -16.0 | -17.0 |