Floating Point Assignment

1 IEEE 754 Single - precision floating point format

IEEE 754 Single precision floating point format is a 32 bit floating point representation. The bit fields are recognized as listed below.

- **bit 31: Sign bit (0: positive, 1: negative).**
- **bits 30-23: exponent bits**

  The exponent field has 8 bit width. So 256 values are possible. But the values 0 and 255 are used for representing special numbers. The legal values are from 1 to 254.
- **bits 22-0: significant precision.**

  The significant bits are represented in memory using 23 bit fields. But there is an implicit leading bit and the 23 bits are after the binary point. Unless the exponent is zero the leading bit takes value 1. So the total number of significant bits are 24.

The equation below shows how the value is calculated from bit representation.

$$(-1)^{b_{31}} \times (1.b_{22}b_{21}...b_0)_2 \times 2^{(b_{30}b_{29}...b_{23})_2-127} \tag{1}$$

To yeild the decimal value

$$value = (-1)^{sign} \times \left(1 + \sum_{i=1}^{23} b_{23-i}2^{-i}\right) \times 2^{(e-127)} \tag{2}$$

Floating point numbers are used to represent real numbers in the number system. Since computers can store only finite amount of numbers, a trade of has to be made between range and precision. This can be explained by an example.
Consider the numbers:

$$2025.626 \times 10^2$$

and

$$20.25626 \times 10$$

Both numbers have same significant digits. But second number has more places after decimal point, it is more precise but it's exponent value is less so smaller range.

A hypothetical example can be given as follows. Consider a machine that can store 1000 values. It can therefore use only 3 significant digits (in decimal context). In one case consider the division of numbers from 1 to 2 by thousand. The obtained series is

$$1.\{000, 001, ..., 999\}$$

In this case the number 1 to the left of decimal point is implied. In another case with the same resources we can store the sequence

$$1.00, 1.01, ..., 9.99$$

. The second case covers more range while precision is lesser than case 1.

2 What is Normal and Subnormal Values as per IEEE 754 standards

It is mentioned earlier that there is an implied digit 1 to the left of binary pint in floating point representation. Thus there are 24 significant binary digits. The number which is represented with this extra 1 at the left of binary point is called normal value. In case of IEEE 754 single precision floating point format , the smallest (positive) normal value has the bit pattern as in figure 1 It's decimal value is

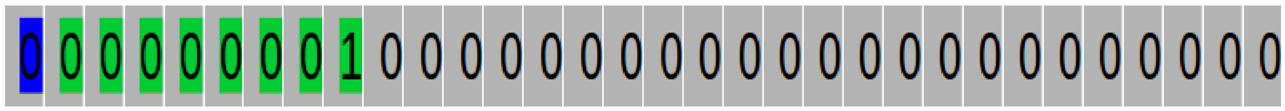

Figure 1: Smallest normal value

$$2^{-126} \cong 1.1755 \times 10^{-38}$$

In IEEE 754 standared the two values of exponent are reserved. $e = 255$ and $e = 0$. The all bit set value (255) is reserved for Not a Number (NaN) warnings. The zero value of exponent is used for subnormal values. In subnormal representation the implied bit is zero. In subnormal representation the floating point value is

$$(-1)^{b_{31}} \times (0.b_{22}b_{21}...b_{02}) \times 2^{-127}$$

This representation is allowed to provide more closeness towards zero. By using subnormal numbers, underflow which may occur in some calculations are gradual.

The bit pattern of smallest (positive) value using subnormal representation is shown in figure 2 The decimal value is 2^{-149} . The largest subnormal value is obtained when all

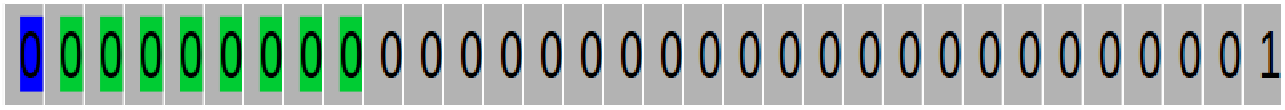

Figure 2: Smallest subnormal value

significant bits are set. The equivalent decimal value $\cong 0.999999988 \times 2^{-126}$. It is close to the smallest normal number 2^{-126}. Figure 3 shows the number line representation of normal and subnormal values.

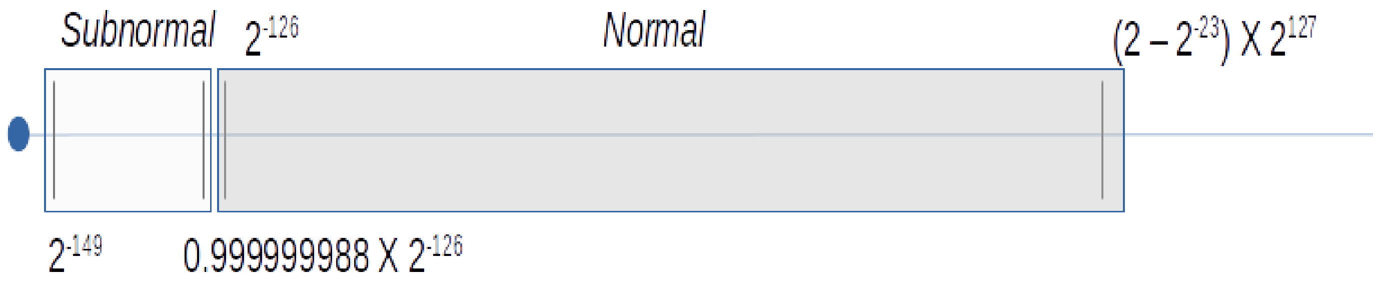

Figure 3: Normal and Subnormal values in number line

3 rounding floating points numbers

When an operation is performed on floating point numbers, the result can overflow the number of significant bits provided for this representation (in case of IEEE 754 - more than 23 bits). In such cases there are two alternatives: Truncation - in which the excess bits are ignored and Rounding towards nearest significant bit. The standard suggests that in an equation

$$z = x < op > y$$

where op is an arithmetic operator, the result z should be as if it was computed exactly and then rounded. This is called correct rounding. IEEE 754 specifies the following rounding modes

1. round to nearest, where ties round to the nearest even digit in the required position
2. round to nearest, where ties round away from zero
3. round up (towards $+\infty$)
4. round down (towards $-\infty$)
5. round toward zero (truncation)

The table 1 shows examples of different rounding modes on floating numbers.

Number	1.20	1.80	1.50	-1.50
Round to nearest, to nearest even digit	1	2	2	-2
Round to nearest, round away from zero	1	2	2	-2
Round up (towards $+\infty$)	2	2	2	-1
Round down (towards $-\infty$)	1	1	1	-2
Round toward zero (truncation)	1	1	1	-1

Table 1: IEEE 754 Rounding Modes Examples