Number Representation

Narendiran S

20-05-2021

1 Binary Numbers

1.1 Negative Numbers

If $x < 0$, then the number is $2^N + x$.

eg) N = 8, for x = -10, then the number is $2^8 + (-10) = 246$

1.2 Fractional numbers/ Real numbers

If x is a Fractional number, for M bit Fractional part x can be represented as $x * 2^-4$.

For example the number x = '01101.101 = 13.625' is represented as $01101101 * 2^-3 = 109/8 = 13.625$ For example the number x = '101.01101 = 5.40625' is represented as $10101101 * 2^-5 = 173/32 = 5.40625$

Hence, a scaling factor is used.

1.3 Range of Numbers

For N-bit 2's complement integer.

$$-2^{N-1} \text{ to } 2^{N-1} - 1$$
$$(\text{for example N=8}) - 2^{8-1} \text{ to } 2^{8-1} - 1 \implies -128 \text{ to } +127$$

For M.L representation of 2's complement real number - M bit for integer and L bit for Fractional.

$$-2^{M-1} \text{ to } 2^{M-1} - 2^{-L}$$
$$(\text{for example 7.3}) - 2^{7-1} \text{ to } 2^{7-1} - 2^{-3} \implies -64 \text{ to } +63.875$$

1.4 Dynamic Range

for an 8-bit Integer: smallest positve number = 1 and largest positve number is +127.

Then the Dynamic range = $\frac{127}{1} = 2^7 - 1 = 2^{8-1} - 1$

for an 8-bit 4.4 Number: smallest positve number = 2^{-4} ane largest positive number is $7.875 = 2^3 - 2^-4 = 2^{M-1} - 2^{-L}$

Then the Dynamic range is $= \frac{2^3 - 2^-4}{2^{-4}} = 2^7 - 1 = 2^{M+L} - 1$.

From this, the Dynamic range of M.L format depeneds on M+L and not no where the decimal point is.

Therefore, for a 32-bit fixed point number the Dynamic range is 2^{31}

1.5 Fixed Point Arithmetic Operations

We can represented Fixed point as

$$F = I.S$$

where F is the fixed point, I is the integer and S is the scaling factor.

An example woule be to represent F = 12.5 in 6.2 format, then the I would be 50 and scaling factor S wiould 2^-2. So we get $50 * 2^-2 = 12.5$

1.5.1 Addition

$$\begin{aligned} F_1 + F_2 &= I_1.S + I_2.S \\ &= (I_1 + I_2).S \end{aligned}$$

For same scaling factor, from the above, we can use normal adders used for integers to perform Addition.

For differnt scaling factor alignment must be done.

1.5.2 Subtraction

$$\begin{aligned} F_1 - F_2 &= I_1.S - I_2.S \\ &= (I_1 - I_2).S \end{aligned}$$

For same scaling factor, from the above, we can use normal subtractors used for integers to perform Subtraction.

For differnt scaling factor alignment must be done.

1.5.3 Multiplication

$$\begin{aligned} F_1 \times F_2 &= I_1.S \times I_2.S \\ &= (I_1 \times I_2).(S \times S) \end{aligned}$$

FOr example, for 4.4 number $\times$ 4.4 number, we get a 16-bit result - which is a 8.8 (4+4.4+4) number. FOr example, for 2.6 number $\times$ 5.3 number, we get a 16-bit result - which is a 7.9 (2+5.6+3) number.

Multiplication causes more precision in the Fractional part. Now, we can drop either of the integer part or ractional part depending on our applicaion and requireemnt.

1.6 Disadvantage of Fixed-Point

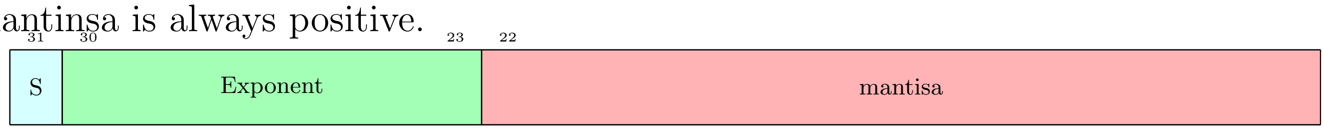
- Dynamic range is poor.

2 Single-Point Precision Floating numbers

Sources: Steve Hollasch Blog

In fixed-point, we explicitly say the scaling factor. **But in Floating point, we have the scaling factor embeeded along with the number to store.**

For a 32-bit number we have a sign bit, 8-bit Exponent and 23-bit mantinsa. 23-bit mantinsa is always positive.



- Mantisa is the actual value.
- Mantisa decides the precision.
- Exponent provides the Dynamic range.
- We use normalized notation. ie.) Matisa has always 1 followed by 23 bits of number but we wont' store the one - it is implicitly assumed.
  - mantisa is 24-bit $1.m_{23}m_{22}\dots m_2m_1m_0$
  - But we store only the 23-bit $m_{23}m_{22}\dots m_2m_1m_0$
- the Exponent e is an unsigned 8-bit integer. (0-255)
- Minimum value of mantinsa = $[1\ .\ 0\ 0\ 0\dots 0\ 0] = 1$
- Maximum value of mantinsa = $[1\ .\ 1\ 1\ 1\dots 1\ 1] = 2 - 2^{23} \approx 2$
- 23-bit Mantisa actually holds 24 bit of implied 'one'.

Hence, the number can be:

$$x = [1.m_{23}m_{22}\dots m_2m_1m_0] \times 2^{e-127}$$

The smallest positive value is $\implies S = 0, e = 1\ (0 \text{ reserved}), m = 0 \implies [1.000\dots 0]2^{1-127} = 1.0 \times 2^{-126}$

The largest positive value is $\implies S = 0, e = 254\ (255 \text{ reserved}), m = 2 \implies [1.111\dots 1]2^{254-127} = (2-2) \times 2^{127}$

Hence, the Dynamic range is $\frac{2^{128}}{2^{-126}} = 2^{254}$

2.1 Normalized number

Assume leading 1 before the binary point. There will be an 1 before the mantisa

Sign (S)	Exponent (e)	Mantisa (m)	Value
0	00000001	$000\ldots00$	Smallest Positive normalized number $=(-1)^0 \times [1.000\ldots00] \times 2^{1-127}$ $=+1.0 \times 2^{1-127}$ $=+2^{-126}$
1	00000001	$000\ldots00$	Smallest Negative normalized number $=(-1)^1 \times [1.000\ldots00] \times 2^{1-127}$ $=-1.0 \times 2^{-126}$ $=-2^{-126}$
0	11111110	$111\ldots11$	Largest Positive normalized number $=(-1)^0 \times [1.111\ldots11] \times 2^{254-127}$ $=+(2-2^{-23}) \times 2^{127}$
1	11111110	$111\ldots11$	Largest Negative normalized number $=(-1)^1 \times [1.111\ldots11] \times 2^{254-127}$ $=-(2-2^{-23}) \times 2^{127}$
0	$00000001\ldots11111110$	$XXX\ldots XX$	Positive normalized Range $=+[1.m_{23}m_{22}\ldots m_2m_1m_0]2^{e-127}$
1	$00000001\ldots11111110$	$XXX\ldots XX$	Negative normalized Range $=-[1.m_{23}m_{22}\ldots m_2m_1m_0]2^{e-127}$

2.2 Denormalized Numbers

Special Case occurs when all the bits of exponent are 0s. Assume leading 0 before the binary point. There will be an 0 before the mantisa. The value can be written as $(-1)^S \times [0.m_{23}m_{22}\ldots m_2m_1m_] \times 2^{-126}$. The exponent is -126 and not $0-127=-127$, because of the implict one assumed in this special case.

Hence, the smallest posivie number can now go beyound 2^{-126} when exponent is all zeros and the mantinsa can provide with values. So the Denormalized number can be represented as shown below:

Sign (S)	Exponent (e)	Mantisa (m)	Value
0	00000000	$000\ldots01$	Smallest Positive Denormalized number $=(-1)^0 \times [0.000\ldots01] \times 2^{-126}$ $=+2^{-23}2^{-126}$ $=+2^{-149}$
1	00000000	$000\ldots01$	Smallest Negative Denormalized number $=(-1)^1 \times [0.000\ldots01] \times 2^{-126}$ $=-2^{-23}2^{-126}$ $=-2^{-149}$
0	00000000	$111\ldots11$	Largest Positive Denormalized number $=(-1)^0 \times [0.111\ldots11] \times 2^{-126}$ $=(1-2^{-23}) \times 2^{-126}$
1	00000000	$111\ldots11$	Largest Negative Denormalized number $=(-1)^1 \times [0.111\ldots11] \times 2^{-126}$ $=-(1-2^{-23}) \times 2^{-126}$
0	00000000	$000\ldots01$ $\vdots$ $111\ldots11$	Positive denormalized Range $=+[0.m_{23}m_{22}\ldots m_2m_1m_0]2^{-126}$
1	00000000	$000\ldots01$ $\vdots$ $111\ldots11$	Negative denormalized Range $=-[0.m_{23}m_{22}\ldots m_2m_1m_0]2^{-126}$

2.3 Special Cases

NaN - Not A Numbers

Sign (S)	Exponent (e)	Mantisa (m)	Value
0	00000000	$000\dots00$	+0
1	00000000	$000\dots00$	-0
0	11111111	$000\dots00$	$+\infty$
1	11111111	$000\dots00$	$-\infty$
0	11111111	$0XX\dots XX$	SNaN
1	11111111	$0XX\dots XX$	SNaN
0	11111111	$1XX\dots XX$	QNaN
1	11111111	$1XX\dots XX$	QNaN

2.4 Numbers which can't be represented

- Positive number greater than $(2-2^{-23})\times 2^{127}$ (positive overflow)
- Negative number less than $-(2-2^{-23})\times 2^{127}$ (negative overflow)
- Zero
- Positive numbers less than 2^{-149} (positive underflow)
- Negative numbers greater than -2^{-149} (neagative underflow)

2.5 Conversion base-10 to Floating point

Value	Calculation	Sign (S)	Exponent (e)	Mantisa (m)	Hex Rep
+0		0	00000000	$000\dots00$	0x00000000
-0		0	00000000	$000\dots00$	0x80000000
$+\infty$		0	11111111	$000\dots00$	0x7F800000
$-\infty$		1	11111111	$000\dots00$	0xFF800000
0.2	$0.2_{10} = 0.00110011001100110011001100_2$ $= 1.10011001100110011001100 \times 2^3$	0	$127-3=124$	10011001100110011001100	0x3E4CCCCC
1.0	$1.0_{10} = 1.00000000000000000000000_2$ $= 1.00000000000000000000000 \times 2^0$	0	$127-0=127$	00000000000000000000000	0x3F800000
1.99999988	$1.99999988_{10} = 1.11111111111111111111110_2$ $= 1.11111111111111111111110 \times 2^0$	0	$127-0=127$	11111111111111111111110	0x3FFFFFFE
16,777,215	$16777215_{10} = 111111111111111111111111_2$ $1.11111111111111111111111 \times 2^{-23}$	0	$127-(-23)=150$	11111111111111111111111	0x4B7FFFFF
3.40282347e+38	I DONT' KNOW	0	254	11111111111111111111111	0x7F7FFFFF