Floating Point

How does RISC-V support numbers with ~~fractions~~ ?

Scientific Notation is just a way to represent very large or very small number.

$\Rightarrow 4500000 = 4.5 \times 10^{6}$

Coefficient, Base, Exponent

$\Rightarrow 0.00453 = 4.53 \times 10^{-3}$

Decimal

- $\Rightarrow 5.64 \times 10^{33}$
- $\Rightarrow -2.34 \times 10^{56}$

→ Normalized.

- $\Rightarrow 109.64 \times 10^{33}$
- $\Rightarrow 0.002 \times 10^{-4}$
- $\Rightarrow +987.02 \times 10^{9}$

→ Not Normalized.

In Binary,

$\pm 1.xxxx_2 \times 2^{yyy}$

IEEE-754 → defines floating point standard.

- Single precision. (32-bit)
- Double precision. (64-bit)

In case of double precision, using more bits, you can represent a larger or a smaller number than single precision.

Normalized Number

Decimal Point (representing Decimal Numbers)

⇒ A decimal number is Normalized if :

i) Only one digit before the decimal point.

ii) And that digit must be a non-zero number.

64.8 ✗ not a normalized number.

6.48×10^{1} ✓ a normalized number.

To normalize a number you need to shift the decimal point (.) left or right until you have a single non-zero digit before the decimal point.

⇒ If you shift left, the number of times you left shifted will be added with the exponent. ⇒ 112.54×10^{35}

⇒ $1.1254 \times 10^{35+2}$ ⇒ 1.1254×10^{37}

⇒ If you shift right, the number of times you right shifted will be subtracted from the exponent. ⇒ 0.0065

⇒ 0.0065×10^{0}

⇒ $6.5 \times 10^{0-3} = 6.5 \times 10^{-3}$

Binary Point (representing Binary Numbers)

⇒ A binary number is Normalized if :

i) Only one digit before the binary point.

ii) And that digit must be a non-zero number.

⇒ 11.00101×2^{35} ✗ not a normalized number.

⇒ 1.100101×2^{37} ✓ a normalized number.

To normalize a number you need to shift the binary point (.) left or right until you have a single non-zero digit before the binary point.

=> If you **shift left**, the **number of times** you left shifted will be **added** with the exponent. => 110.111×2^{35}

$\Rightarrow 1.10111 \times 2^{35+2}$

=> If you **shift right**, the **number of times** you right shifted will be **subtracted** from the exponent. => 0.00110

$\Rightarrow 0.00110 \times 2^{0}$

$\Rightarrow 1.10 \times 2^{0-3} = 1.10 \times 2^{-3}$

IEEE Floating-Point Format

	Sign Bit	Exponent	Fraction
Single P.	1 bit	8 bits	23 bits
Double P.	1 "	11 "	52 "

Single Precision (32 Bit)

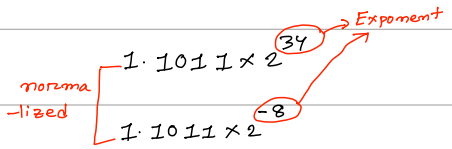
(Biased)

Sign Bit	Exponent	Fraction
1	8	23

Sign Bit = 0 => positive number

1 => negative "

Exponent = It will be represented as unsigned number.



8 bit unsigned binary range = 0 to 2^8-1

= 0 to 255

But 0000 0000 and 1111 1111 are reserved, so the range for biased exponent is 1 to 254

If the size of biased exponent field is n bits, Bias = $2^{(n-1)}-1$

Hence, for 8 bit biased exponent, bias = $2^7-1 = 127$

Biased Exponent = Actual exponent of the binary number + Bias

If you find the actual exponent within (-126 to +127) only then that number can be represented using Single Precision Format.

Ex: 1.1011×2^{34}; Find the Biased Exponent of the given number in IEEE 754 single precision format.

Soln: Check if the number is in normalized format or not.

↳ If no. then normalize it and proceed.

1.1011×2^{34} => normalized number.

(34 = Actual Exponent of the binary number)

Bias = $2^{(n-1)} - 1 = 2^{8-1} - 1 = 127$.

∴ Biased Exponent = 34 + 127 = 161 = 1010 0001 (Ans)

Ex: 11.1011×2^{-8}; Find the Biased Exponent of the given number in IEEE 754 single precision format.

Soln: 11.1011×2^{-8}

$= 1.11011 \times 2^{-8-1}$

$= 1.111011 \times 2^{-9}$ (-9 = actual exponent) } Normalized

Bias = $2^{(n-1)} - 1 = 2^{8-1} - 1 = 127$.

∴ Biased Exponent = -9 + 127 = 118 = 0111 0110 (Ans)

Decimal to Floating Point Conversion:

(i) Convert the decimal number to binary number.

(ii) Normalize the binary number.

(iii) Find the biased exponent.

(iv) Sign Bit

(v) Find the fraction.

Ex: Convert 50.6749 to 32 bit IEE-754 Floating point representation :-

$50 = 11\ 0010$

$.6749 = 10\ 1011\ 0011\ldots$

(i) $50.6749 = (11\ 0010.10\ 1011\ 0011\ldots)$

$= 11\ 0010.10\ 1011\ 0011\ldots \times 2^0$

(ii) $= 1.1001\ 0101\ 0110\ 011\ldots \times 2^5$ ← actual exponent

(iii) Bias $= 2^{8-1} - 1 = 2^7 - 1 = 127$

$\therefore$ Biased exponent $= 5 + 127 = 132 = 1000\ 0100$

(iv) positive number; sign bit = 0

(v) $1.1001\ 0101\ 0110\ 011\ldots \times 2^5$

fraction

Fraction = 1001 0101 0110 011 00000000

rest of the bits will be filled up by 0s.

$.6749 \times 2 = 1.3498$

$.3498 \times 2 = 0.6996$

$.6996 \times 2 = 1.3992$

$.3992 \times 2 = 0.7984$

$.7984 \times 2 = 1.5968$

$.5968 \times 2 = 1.1936$

$.1936 \times 2 = 0.3872$

$.3872 \times 2 = 0.7744$

$.7744 \times 2 = 1.5488$

$.5488 \times 2 = 1.0976$

(Biased)

Sign Bit	Exponent	Fraction
0	1000 0100	1001 0101 0110 011 00000000

$50.6749 = 0100\ 0010\ 0100\ 1010\ 1011\ 0011\ 0000\ 0000$

$= 0x424AB300$

Double Precision (64 Bit)

Sign Bit	Exponent (Biased)	Fraction
1	11	52

Sign Bit = 0 ⇒ positive number

1 ⇒ negative "

Exponent = It will be represented as unsigned number.

11 bit unsigned binary range = 0 to $2^{11}-1$

= 0 to 2047

But 000 0000 0000 and 111 1111 1111 are reserved, so the range for biased exponent is 1 to 2046

If the size of biased exponent field is n bits, Bias = $2^{(n-1)}-1$

Hence, for 11 bit biased exponent, bias = $2^{10}-1 = 1023$

Convert −0.0232 to 12 bit IEEE-754 floating point representation, where biased component is 4 bits.

Soln:

(i) $-0.0232 = -0.00000\,10$

(ii) $-0.00000\,10 = 1.0 \times 2^{-6}$ ← actual exponent; 1.0 ← fraction

(iii) Bias = $2^{4-1}-1 = 7$

∴ Biased Exponent = −6 + 7 = 1 = 0001

(iv) Sign Bit = 1.

(v) Fraction = 000 0000

1	0001	000 0000

−0.000232 = 1000 1000 0000

= 0x880 (Ans)

Floating Point to Decimal:

(i) Hex to Bimary.

(ii) Arrange the bimary according to the format.

(iii) Determine the sign

(iv) Find out the exponent from biased exponent.

(v) Convert fraction to decimal.

(vi) Decimal Number = $(-1)^{\text{sign bit}} \times (1 + \text{Fraction}) \times 2^{(\text{Exponent})}$

Example: 0xF2400120; convert this single precision floating point number to decimal.

Soln: 0xF2400120

(i) 1111 0010 0100 0000 0000 0001 0010 0000

(ii) 1 111 00100 100 0000 0000 0001 0010 0000

1 — Sign Bit; 111 00100 — Biased Exp.; 100 0000 0000 0001 0010 0000 — Fraction

(iii) Sign = −

(iv) Biased exp. = 111 0010 0 = 228

Bias = $2^{8-1} - 1 = 127$

∴ Exponent = 228 − 127 = 101

(v) Fraction = 100 0000 0000 0001 0010 0000

= 0.100 0000 0000 0001 0010 0000

= 0.5000343323

(vi) Decimal Value = $(-1)^1 \times (1 + 0.5000343323) \times 2^{101}$

$= -1.5000343323 \times 2^{101}$

$= -3.80303884 \times 10^{30}$

Extension: Upto 6 decimal points with rounding = -3.803039×10^{30}

" 6 " " without " = -3.803038×10^{30}

Floating Point Addition / Subtraction

A and B both are floating point number.

⇒ A + B (Make sure the number is in binary)

i) Normalize both A and B.

ii) Align the bin point so that the lower exponent match with the higher exponent.

iii) Now add / sub accordingly.

iv) Normalize the result.

iv) Round if neeessary.

Ex: $0.999 \times 10^{1} + 1.610 \times 10^{-1}$

$= 99.99 + 0.1610$

$= 11000011.1111\ 1101\ 01 + 0.0010\ 1001\ 00$

$= 1.1000\ 1111\ 1111\ 0101 \times 2^{6} + 1.0100\ 100 \times 2^{-3}$

$= 1.1000\ 1111\ 1111\ 0101 \times 2^{6} + 0.0000\ 0000\ 1\ 0100\ 100 \times 2^{6}$

$= 1.10010 \times 2^{6}$ (Ans)

Floating Point Multiplication

A and B both are floating point number.

⇒ A + B (Make sure the number is in binary)

i) Normalize both A and B.

ii) Add the exponents.

iii) Now multiply accordingly.

iv) Normalize the result.

iv) Round if neeessary.

v) Determine the sign from the operation.

Ex: $1.110 \times 2^{5} \times 1.11 \times 2^{-5}$

$= 1.110 \times 1.11 \times 2^{5+(-5)}$

$= 11.0001 \times 2^{0}$

$= 1.10001 \times 2^{1}$ (Ans)