Real Numbers to Floating Point Number Representation

Dr. PRADUMN KUMAR PANDEY

2022-11-15

Example 1

Convert $(23.75)_{10}$ to 32 Bit Single Precision IEEE 754 Binary Floating Point Standard, From a Base 10 Decimal Number

For this we require:

- 1 bit for sign,
- 8 bits for exponent,
- 23 bits for mantissa

Step 1: First, convert to the binary (base 2) the integer part: 23.

- Divide the number repeatedly by 2.
- Keep track of each remainder.
- Stop when we get a quotient that is equal to zero.

division = quotient + remainder

- $23 \div 2 = 11 + 1$;
- $11 \div 2 = 5 + 1$;
- $5 \div 2 = 2 + 1$;
- $2 \div 2 = 1 + 0$;
- $1 \div 2 = 0 + 1$;

Step 2: Construct the base 2 representation of the integer part of the number.

- Take all the remainders starting from the bottom of the list constructed above.

$$\mathbf{23}_{(10)} = \mathbf{1\ 0111}_{(2)}$$

Step 3: Convert to the binary (base 2) the fractional part: 0.75.

- Multiply it repeatedly by 2.
- Keep track of each integer part of the results.
- Stop when we get a fractional part that is equal to zero.

multiplying = integer + fractional part;

- $0.75 \times 2 = 1 + 0.5;$
- $0.5 \times 2 = 1 + 0;$

Step 4: Construct the base 2 representation of the fractional part of the number.

- Take all the integer parts of the multiplying operations, starting from the top of the constructed list above.

$$\mathbf{0.75}_{(10)} = \mathbf{0.11}_{(2)}$$

contd..

- Positive number before normalization:

$$\mathbf{23.75}_{(10)} = \mathbf{1\ 0111.11}_{(2)}$$

Step 5: Normalize the binary representation of the number.

- Shift the decimal mark 4 positions to the left so that only one non zero digit remains to the left of it

$$\mathbf{1\ 0111.11}_{(2)} \mathbf{=} \mathbf{1.0111\ 11}_{(2)} \times 2^4$$

Step 6

- Up to this moment, there are the following elements that would feed into the 32 bit single precision IEEE 754 binary floating point representation:

  - Sign: **0** (a positive number)
  - Exponent (unadjusted): **4**
  - Mantissa (not normalized): **1.0111 11**

Step 7:Adjust the exponent.

- Use the 8 bit excess/bias notation:

Exponent (adjusted) =

- Exponent (unadjusted) + ($2^{(8-1)}$ - 1)
- $4 + 127_{(10)} = 131_{(10)}$

Step 8: Convert the adjusted exponent from the decimal (base 10) to 8 bit binary.

- Use the same technique of repeatedly dividing by 2:

division = quotient + remainder;

- $131 \div 2 = 65 + 1;$
- $65 \div 2 = 32 + 1;$
- $32 \div 2 = 16 + 0;$
- $16 \div 2 = 8 + 0;$
- $8 \div 2 = 4 + 0;$
- $4 \div 2 = 2 + 0;$
- $2 \div 2 = 1 + 0;$
- $1 \div 2 = 0 + 1;$

Step 9: Construct the base 2 representation of the adjusted exponent.

- Take all the remainders starting from the bottom of the list constructed above:

$$\textbf{Exponent (adjusted)} = \mathbf{131_{(10)}} = \mathbf{1000\ 0011_{(2)}}$$

Step 9: Construct the base 2 representation of the adjusted exponent.

- Take all the remainders starting from the bottom of the list constructed above:

$$\textbf{Exponent (adjusted)} = \mathbf{131}_{(10)} = \mathbf{1000\ 0011}_{(2)}$$

Step 10: Normalize the mantissa.

- Remove the leading (the leftmost) bit, since it's allways 1, and the decimal point, if the case.
- Adjust its length to 23 bits, by adding the necessary number of zeros to the right.

Mantissa (normalized) =

- ~~1~~. 01 1111 0 0000 0000 0000 0000 = 011 1110 0000 0000 0000 0000

Step 11: The three elements that make up the number's 32 bit single precision IEEE 754 binary floating point representation:

- Sign (1 bit) $= 0$ (a positive number)
- Exponent (8 bits) $= 1000\ 0011$
- Mantissa (23 bits) $= 011\ 1110\ 0000\ 0000\ 0000\ 0000$

contd..

Sign (1 bit):

0
31

Exponent (8 bits):

1	0	0	0	0	0	1	1
30	29	28	27	26	25	24	23

Mantissa (23 bits):

0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Result: Number 23.75 converted from decimal system (base 10) to 32 bit single precision IEEE 754 binary floating point:

0 1000 0011 011 1110 0000 0000 0000 0000

Exercise

A) **Convert 69.75 to 32 Bit Single Precision IEEE 754 Binary Floating Point Standard, From a Base 10 Decimal Number**

B) **Convert -51.2 to 32 Bit Single Precision IEEE 754 Binary Floating Point Standard, From a Base 10 Decimal Number**

Solution

A) 0 1000 0101 000 1011 1000 0000 0000 0000

B) 1 1000 0100 100 1100 1100 1100 1100 1100