Decimal to Floating-Point Conversions

Floating-Point Conversion Examples Binary/Boolean Main Index

[*Decimal to Floating-Point Conversions*] [*Float to Decimal Conversion*]

The Conversion Procedure

The rules for converting a decimal number into floating point are as follows:

A. Convert the absolute value of the number to binary, perhaps with a fractional part after the binary point. This can be done by converting the integral and fractional parts separately. The integral part is converted with the techniques examined previously. The fractional part can be converted by multiplication. This is basically the inverse of the division method: we repeatedly multiply by 2, and harvest each one bit as it appears left of the decimal.

B. Append $\times 2^0$ to the end of the binary number (which does not change its value).

C. Normalize the number. Move the binary point so that it is one bit from the left. Adjust the exponent of two so that the value does not change.

D. Place the mantissa into the mantissa field of the number. Omit the leading one, and fill with zeros on the right.

E. Add the bias to the exponent of two, and place it in the exponent field. The bias is $2^{k-1} - 1$, where k is the number of bits in the exponent field. For the eight-bit format, $k = 3$, so the bias is $2^{3-1} - 1 = 3$. For IEEE 32-bit, $k = 8$, so the bias is $2^{8-1} - 1 = 127$.

F. Set the sign bit, 1 for negative, 0 for positive, according to the sign of the original number.

Using The Conversion Procedure

- Convert 2.625 to our 8-bit floating point format.

  A. The integral part is easy, $2_{10} = 10_2$. For the fractional part:

  $0.625 \times 2 = 1.25$ $\boxed{1}$ *Generate 1 and continue with the rest.*

  $0.25 \times 2 = 0.5$ $\boxed{0}$ *Generate 0 and continue.*

  $0.5 \times 2 = 1.0$ $\boxed{1}$ *Generate 1 and nothing remains.*

  So $0.625_{10} = 0.101_2$, and $2.625_{10} = 10.101_2$.

  B. Add an exponent part: $10.101_2 = 10.101_2 \times 2^0$.

  C. Normalize: $10.101_2 \times 2^0 = 1.0101_2 \times 2^1$.

  D. Mantissa: 0101

  E. Exponent: $1 + 3 = 4 = 100_2$.

  F. Sign bit is 0.

  The result is $\boxed{0}\boxed{100}\boxed{0101}$. Represented as hex, that is 45_{16}.

- Convert -4.75 to our 8-bit floating point format.

  a. The integral part is $4_{10} = 100_2$. The fractional:

  $0.75 \times 2 = 1.5$ $\boxed{1}$ *Generate 1 and continue with the rest.*

  $0.5 \times 2 = 1.0$ $\boxed{1}$ *Generate 1 and nothing remains.*

  So $4.75_{10} = 100.11_2$.

  b. Normalize: $100.11_2 = 1.0011_2 \times 2^2$.

  c. Mantissa is 0011, exponent is $2 + 3 = 5 = 101_2$, sign bit is 1.

  So -4.75 is $\boxed{1}\boxed{101}\boxed{0011}$ = $d3_{16}$

- Convert 0.40625 to our 8-bit floating point format.
  a. Converting:

  $0.40625 \times 2 = 0.8125$ $\boxed{0}$ *Generate 0 and continue.*
  $0.8125 \times 2 = 1.625$ $\boxed{1}$ *Generate 1 and continue with the rest.*
  $0.625 \times 2 = 1.25$ $\boxed{1}$ *Generate 1 and continue with the rest.*
  $0.25 \times 2 = 0.5$ $\boxed{0}$ *Generate 0 and continue.*
  $0.5 \times 2 = 1.0$ $\boxed{1}$ *Generate 1 and nothing remains.*

  So $0.40625_{10} = 0.01101_2$.

  b. Normalize: $0.01101_2 = 1.101_2 \times 2^{-2}$.

  c. Mantissa is 1010, exponent is $-2 + 3 = 1 = 001_2$, sign bit is 0.

  So 0.40625 is $\boxed{0}\boxed{001}\boxed{1010}$ = $1a_{16}$

- Convert -12.0 to our 8-bit floating point format.
  a. $12_{10} = 1100_2$.

  b. Normalize: $1100.0_2 = 1.1_2 \times 2^3$.

  c. Mantissa is 1000, exponent is $3 + 3 = 6 = 110_2$, sign bit is 1.

  So -12.0 is $\boxed{1}\boxed{110}\boxed{1000}$ = $e8_{16}$

- Convert decimal 1.7 to our 8-bit floating point format.
  a. The integral part is easy, $1_{10} = 1_2$. For the fractional part:

  $0.7 \times 2 = 1.4$ $\boxed{1}$ *Generate 1 and continue with the rest.*
  $0.4 \times 2 = 0.8$ $\boxed{0}$ *Generate 0 and continue.*
  $0.8 \times 2 = 1.6$ $\boxed{1}$ *Generate 1 and continue with the rest.*
  $0.6 \times 2 = 1.2$ $\boxed{1}$ *Generate 1 and continue with the rest.*
  $0.2 \times 2 = 0.4$ $\boxed{0}$ *Generate 0 and continue.*
  $0.4 \times 2 = 0.8$ $\boxed{0}$ *Generate 0 and continue.*
  $0.8 \times 2 = 1.6$ $\boxed{1}$ *Generate 1 and continue with the rest.*
  $0.6 \times 2 = 1.2$ $\boxed{1}$ *Generate 1 and continue with the rest.*
  …

  The reason why the process seems to continue endlessly is that it does. The number 7/10, which makes a perfectly reasonable decimal fraction, is a repeating fraction in binary, just as the faction 1/3 is a repeating fraction in decimal. (It repeats in binary as well.) We cannot represent this exactly as a floating point number. The closest we can come in four bits is .1011. Since we already have a leading 1, the best eight-bit number we can make is 1.1011.

  b. Already normalized: $1.1011_2 = 1.1011_2 \times 2^0$.

  c. Mantissa is 1011, exponent is $0 + 3 = 3 = 011_2$, sign bit is 0.

  The result is $\boxed{0}\boxed{011}\boxed{1011}$ = $3b_{16}$. This is not exact, of course. If you convert it back to decimal, you get 1.6875.

- Convert -1313.3125 to IEEE 32-bit floating point format.
  a. The integral part is $1313_{10} = 10100100001_2$. The fractional:

  $0.3125 \times 2 = 0.625$ $\boxed{0}$ *Generate 0 and continue.*
  $0.625 \times 2 = 1.25$ $\boxed{1}$ *Generate 1 and continue with the rest.*

$0.25 \times 2 = 0.5$ $\boxed{0}$ *Generate 0 and continue.*

$0.5 \times 2 = 1.0$ $\boxed{1}$ *Generate 1 and nothing remains.*

So $1313.3125_{10} = 10100100001.0101_2$.

b. Normalize: $10100100001.0101_2 = 1.01001000010101_2 \times 2^{10}$.

c. Mantissa is 01001000010101000000000, exponent is 10 + 127 = 137 = 10001001_2, sign bit is 1.

So -1313.3125 is $\boxed{1}\boxed{10001001}\boxed{01001000010101000000000}$ = $c4a42a00_{16}$

- Convert 0.1015625 to IEEE 32-bit floating point format.

a. Converting:

$0.1015625 \times 2 = 0.203125$ $\boxed{0}$ *Generate 0 and continue.*

$0.203125 \times 2 = 0.40625$ $\boxed{0}$ *Generate 0 and continue.*

$0.40625 \times 2 = 0.8125$ $\boxed{0}$ *Generate 0 and continue.*

$0.8125 \times 2 = 1.625$ $\boxed{1}$ *Generate 1 and continue with the rest.*

$0.625 \times 2 = 1.25$ $\boxed{1}$ *Generate 1 and continue with the rest.*

$0.25 \times 2 = 0.5$ $\boxed{0}$ *Generate 0 and continue.*

$0.5 \times 2 = 1.0$ $\boxed{1}$ *Generate 1 and nothing remains.*

So $0.1015625_{10} = 0.0001101_2$.

b. Normalize: $0.0001101_2 = 1.101_2 \times 2^{-4}$.

c. Mantissa is 10100000000000000000000, exponent is -4 + 127 = 123 = 01111011_2, sign bit is 0.

So 0.1015625 is $\boxed{0}\boxed{01111011}\boxed{10100000000000000000000}$ = $3dd00000_{16}$

- Convert 39887.5625 to IEEE 32-bit floating point format.

a. The integral part is $39887_{10} = 1001101111001111_2$. The fractional:

$0.5625 \times 2 = 1.125$ $\boxed{1}$ *Generate 1 and continue with the rest.*

$0.125 \times 2 = 0.25$ $\boxed{0}$ *Generate 0 and continue.*

$0.25 \times 2 = 0.5$ $\boxed{0}$ *Generate 0 and continue.*

$0.5 \times 2 = 1.0$ $\boxed{1}$ *Generate 1 and nothing remains.*

So $39887.5625_{10} = 1001101111001111.1001_2$.

b. Normalize: $1001101111001111.1001_2 = 1.0011011110011111001_2 \times 2^{15}$.

c. Mantissa is 00110111100111110010000, exponent is 15 + 127 = 142 = 10001110_2, sign bit is 0.

So 39887.5625 is $\boxed{0}\boxed{10001110}\boxed{00110111100111110010000}$ = $471bcf90_{16}$