FLOATING POINT VARIABLES

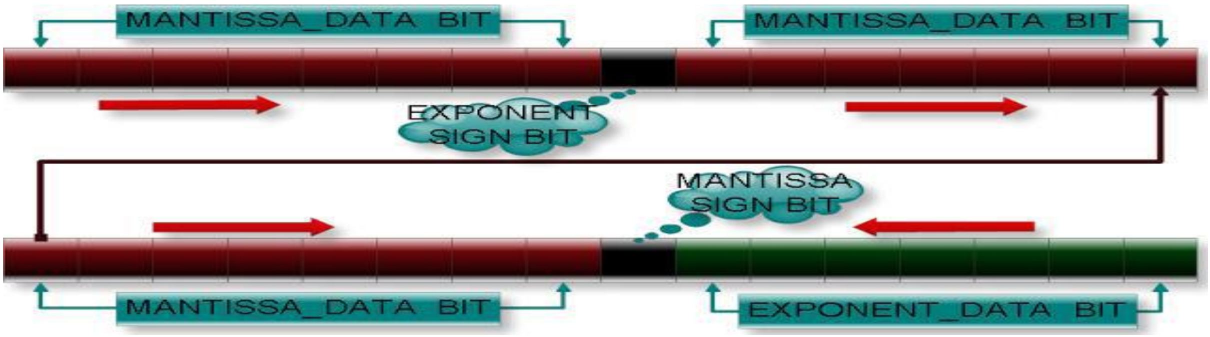
In computing, floating point is the formulaic representation that approximates a real number so as to support a trade-off between range and precision. A number is, in general , represented approximately to a fixed number of significant digits and scaled using an exponent in some fixed base; the base for the scaling is normally two, ten, or sixteen. A number that can be represented exactly is of following form : M x b^e ,

Where m is the mantissa (an integer number), b is base and e is exponent.

- C supports two floating types: FLOAT (4 bytes), DOUBLE(8 bytes) and LONG DOUBLE(10 bytes).

The float and double are represented using 32-bit single precision and 64-bit double precision.

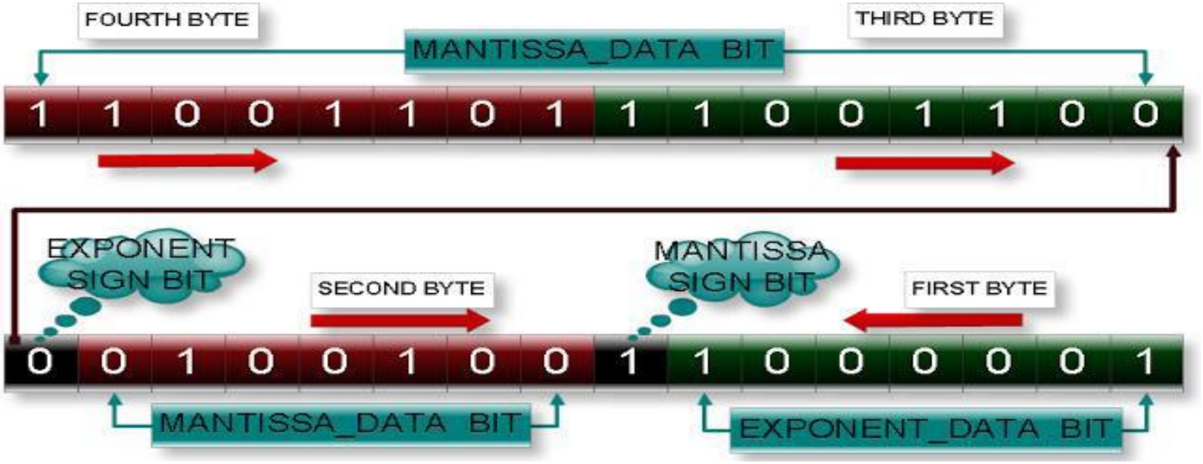
- For single precision floating point we have: 1 sign bit, 8 exponent bits, 23 mantissa bits
- For double precision floating points we have: 1 sign bit, 11 exponent bits, 52 mantissa bits. Following figure illustrate how floating point number is stored in memory:



Five important rules to be followed:

- **Rule 1**: To find the mantissa and exponent, we convert data into scientific form.
- **Rule 2**: Before the storing of exponent, 127 is added to the exponent.
- **Rule 3**: Exponent is stored in memory in first byte from right to left side.
- **Rule 4**: If exponent is negative number it will be stored in 2's complement form.
- **Rule 5**: Mantissa is stored in the memory in second byte from right to left side.

Example: Memory representation of float a= -10.3f


FOURTH BYTE
THIRD BYTE
MANTISSA_DATA BIT
1 1 0 0 1 1 0 1 1 1 0 0 1 1 0 0
EXPONENT SIGN BIT
SECOND BYTE
MANTISSA SIGN BIT
FIRST BYTE
0 0 1 0 0 1 0 0 1 1 0 0 0 0 0 1
MANTISSA_DATA BIT
EXPONENT_DATA BIT