IEEE Floating-Point Representation

There is some variation in the schemes used for storing real numbers in computer memory, but one floating-point representation was standardized in 1985 by the Institute for Electrical and Electronic Engineers (IEEE) and has become almost universal. This **IEEE Floating-Point Format** specifies how *single precision* real numbers are to be represented using 32 bits and *double precision* using 64 bits. The double precision format is simply a wider version of the single precision format, so we will describe only single precision.

We begin by writing the binary representation of the number in **floating-point form**, which is like scientific notation except that the base is two rather than ten,

$$b_1.b_2b_3 \ldots \times 2^k$$

where each b_i is 0 or 1, and $b_1 = 1$ (unless the number is 0). $b_1.b_2b_3$... is called the **mantissa** (or **fractional part** or **significand**) and k is the **exponent** (or **characteristic**). To illustrate, consider the real number 22.625, which can be represented in binary as

$$10110.101_2$$

Rewriting this in floating-point form,

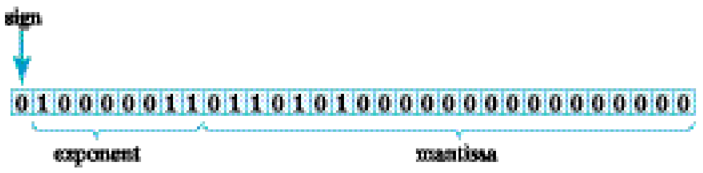
$$1.0110101_2 \times 2^4$$

is easy since multiplying (dividing) a base-two number by 2 is the same as moving the binary point to the right (left). 1.0110101_2 is the mantissa and 4 is the exponent.

In the IEEE format for single-precision real values,

- the leftmost bit stores the sign of the mantissa, 0 for positive, 1 for negative
- the next 8 bits store the binary representation of the exponent + 127 (called a *bias*)
- the rightmost 23 bits store the bits to the right of the binary point in the mantissa (the bit to the left need not be stored since it is always 1, unless the number is 0)

For 22.625, the stored exponent would be 4 + 127 = 10000011_2 and the stored mantissa would be $01101010000000000000000_2$:



The IEEE representation for double precision uses an 11-bit exponent with a bias of 1023 and 53 bits for the signed mantissa.