What Is IEEE Standard Arithmetic?

Nicholas J. Higham[*]

May 7, 2020

The IEEE Standard 754, published in 1985 and revised in 2008 and 2019, is a standard for binary and decimal floating-point arithmetic. The standard for decimal arithmetic (IEEE Standard 854) was separate when it was first published in 1987, but it was included with the binary standard from 2008. We focus here on the binary part of the standard.

The standard specifies floating-point number formats, the results of the basic floating-point operations and comparisons, rounding modes, floating-point exceptions and their handling, and conversion between different arithmetic formats.

A binary floating-point number is represented as

$$y = \pm m \times 2^{e-t},$$

where t is the precision and $e \in [e_{\min}, e_{\max}]$ is the exponent. The significand m is an integer satisfying $m \le 2^t - 1$. Numbers with $m \ge 2^{t-1}$ are called normalized. Subnormal numbers, for which $0 < m < 2^{t-1}$ and $e = e_{\min}$, are supported.

Four formats are defined, whose key parameters are summarized in the following table. The second column shows the number of bits allocated to store the significand and the exponent. I use the prefix "fp" instead of the prefix "binary" used in the standard. The unit roundoff is $u = 2^{-t}$.

	Bits (significand, exponent)	Unit roundoff	Min positive subnormal	Min positive normalized	Max finite
fp16	(11, 5)	4.88×10^{-4}	5.96×10^{-8}	6.10×10^{-5}	6.55×10^{4}
fp32	(24, 8)	5.96×10^{-8}	1.40×10^{-45}	1.18×10^{-38}	3.40×10^{38}
fp64	(53, 11)	1.11×10^{-16}	4.94×10^{-324}	2.22×10^{-308}	1.80×10^{308}
fp128	(113, 15)	9.63×10^{-35}	6.48×10^{-4966}	3.36×10^{-4932}	1.19×10^{4932}

Fp32 (single precision) and fp64 (double precision) were in the 1985 standard; fp16 (half precision) and fp128 (quadruple precision) were introduced in 2008. Fp16 is defined only as a storage format, though it is widely used for computation.

The size of these different number systems varies greatly. The next table shows the number of normalized and subnormal numbers in each system.

	Normalized	Subnormal
fp16	6.1×10^{4}	2.0×10^{3}
fp32	4.3×10^{9}	1.7×10^{7}
fp64	1.8×10^{19}	9.0×10^{15}
fp128	3.4×10^{38}	1.0×10^{34}

[*]Department of Mathematics, University of Manchester, Manchester, M13 9PL, UK (`nick.higham@manchester.ac.uk`).

We see that while one can easily carry out a computation on every fp16 number (to check that the square root function is correctly computed, for example), it is impractical to do so for every double precision number.

A key feature of the standard is that it is a closed system, thanks to the inclusion of NaN (Not a Number) and ∞ (usually written as inf in programing languages) as floating-point numbers: every arithmetic operation produces a number in the system. A NaN is generated by operations such as

$$0/0, \quad 0 \times \infty, \quad \infty/\infty, \quad (+\infty) + (-\infty), \quad \sqrt{-1}.$$

Arithmetic operations involving a NaN return a NaN as the answer. The number ∞ obeys the usual mathematical conventions regarding infinity, such as

$$\infty + \infty = \infty, \quad (-1) \times \infty = -\infty, \quad (\text{finite})/\infty = 0.$$

This means, for example, that $1 + 1/x$ evaluates as 1 when $x = \infty$.

The standard specifies that all arithmetic operations (including square root) are to be performed as if they were first calculated to infinite precision and then rounded to the target format. A number is rounded to the next larger or next smaller floating-point number according to one of four rounding modes:

- round to the nearest floating-point number, with rounding to even (rounding to the number with a zero least significant bit) in the case of a tie;
- round towards plus infinity and round towards minus infinity (used in interval arithmetic); and
- round towards zero (truncation, or chopping).

For round to nearest it follows that

$$\mathrm{fl}(x \mathbin{\mathrm{op}} y) = (x \mathbin{\mathrm{op}} y)(1 + \delta), \quad |\delta| \le u, \quad \mathrm{op} \in \{+, -, *, /, \surd\},$$

where fl denotes the computed result. The standard also includes a *fused multiply-add operation* (FMA), $x * y + z$. The definition requires it to be computed with just one rounding error, so that $\mathrm{fl}(x*y+z)$ is the rounded version of $x*y+z$, and hence satisfies

$$\mathrm{fl}(x * y + z) = (x * y + z)(1 + \delta), \quad |\delta| \le u.$$

FMAs are supported in some hardware and are usually executed at the same speed as a single addition or multiplication.

The standard recommends the provision of correctly rounded exponentiation (x^y) and transcendental functions (exp, log, sin, acos, etc.) and defines domains and special values for them, but these functions are not required.

A new feature of the 2019 standard is *augmented arithmetic operations*, which compute $\mathrm{fl}(x \mathbin{\mathrm{op}} y)$ along with the error $x \mathbin{\mathrm{op}} y - \mathrm{fl}(x \mathbin{\mathrm{op}} y)$, for $\mathrm{op} = +, -, *$. These operations are useful for implementing compensated summation and other special high accuracy algorithms.

William ("Velvel") Kahan of the University of California at Berkeley received the 1989 ACM Turing Award for his contributions to computer architecture and numerical analysis, and in particular for his work on IEEE floating-point arithmetic standards 754 and 854.

References

This is a minimal set of references, which contain further useful references within.

- D. Goldberg, What Every Computer Scientist Should Know About Floating-Point Arithmetic, ACM Computing Surveys 23, 5–48, 1991.
- Nicholas J. Higham, Accuracy and Stability of Numerical Algorithms, second edition, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002.
- IEEE Standard for Floating-Point Arithmetic, IEEE Std 754-2019 (Revision of IEEE 754-2008), The Institute of Electrical and Electronics Engineers, New York, 2019.
- Jean-Michel Muller, Nicolas Brunie, Florent de Dinechin, Claude-Pierre Jeannerod, Mioara Joldes, Vincent Lefèvre, Guillaume Melquiond, Nathalie Revol, and Serge Torres, Handbook of Floating-Point Arithmetic, second edition, Birkhäuser, Boston, MA, 2018.

Related Blog Posts

- A Multiprecision World (2017)
- Book Review Revisited: Overton's Numerical Computing with IEEE Floating Point Arithmetic (2014)
- Floating Point Numbers by Cleve Moler (2014)
- Half Precision Arithmetic: fp16 Versus bfloat16 (2018)
- The Rise of Mixed Precision Arithmetic (2015)
- What Is Rounding? (2020)
- What Is Floating-Point Arithmetic? (2020)

This article is part of the "What Is" series, available from `https://nhigham.com/category/what-is` and in PDF form from the GitHub repository `https://github.com/higham/what-is`.