Brakefield Research = *Optimal Architecture* 10/17/25
13423 Blanco Rd. #144, San Antonio, Texas 78216

The Case for a non IEEE-754 Floating-Point
U.S. Patent 5,892,697

The current floating-point standard, IEEE-754, is the basis for most computer floating-point arithmetic. It offers a variety of numerical properties (some advocates want a simpler standard, others want better software availability of the less used features). It offers 32-bit and 64-bit memory values. Cost or performance sensitive applications typically have variances or incomplete implementations. Some implementations use exception handlers in place of non-cost-effective hardware.

The floating-point advocated in this paper, herein called **alt-754**, was motivated by certain hardware and software complexities of IEEE-754. The process of finding solutions to IEEE-754 complexity illuminates the choices faced in floating-point architecture.

Choice #1:
IEEE-754 uses exact numbers to stand for both approximate values and for exact values. The impact is that the rounding mechanism is forced to be speed or complexity sub-optimal. Alt-754 has separate and distinct sets of approximate and exact values.

Choice #2:
Values near zero, termed herein as the "low corner", must deal with decreased resolution. The representatives chosen and their encoding affect hardware and software complexity. Alt-754 has a "high corner" symmetric to the "low corner". The alt-754 encoding is chosen for efficient implementation in high performance, i.e. pipelined, processors.

Choice #3:
Wrapping the choices in a floating-point architecture into a model provides the clearest presentation of the floating-point design. Alt-754 offers a relatively clean model with some useful mathematical properties:

3.1 The model illuminates implementation as the process of choosing a subset of the model.
3.2 Symmetry of lower accuracy values near zero, "denorms", with a similar set of values at the high end of the magnitude scale.
3.3 Rounding is defined so as to clarify the issue of rounding bias.
3.4 A powerful model provides generality in the form of defined variations.
3.5 Alt-754 offers the option to track accuracy.

We begin by having exact and inexact values. The exact values are integers scaled by a power of two. The inexact values are a range of values, e.g. a range of real values, encoded by a central "scaled integer" value. A specific range is encoded by an alt-754 floating-point bit pattern. As the two kinds of values are treated distinctly, the center point of the inexact value ranges can be chosen to facilitate hardware implementation.

The model of an inexact range is:

A floating-point number is a sign bit and a bit string unbounded on both ends. Between two specific bits of the string is a decimal point. The number of positions between the decimal point and the leading one bit is the exponent. The number of positions between the leading one bit and the trailing one bit is the accuracy. The bits less significant than and including the trailing one bit are indeterminate; e.g. a specific sign bit and bit string represent a range of values.

If there is no leading one bit, the value is infinite (positive or negative depending on the sign bit). If there is no trailing one bit, the value is exact. If all bits are zeros, the value is mathematical zero. There is also a single value called NaN ("not a number").

The model provides a mathematical set of floating point values. For a given implementation some subset is chosen: The exponent and accuracy limits are specified. The exponent size is chosen as so many bits. Maximal accuracy is chosen to match the arithmetic unit size.

The model of rounding is:

The sequence of number ranges (e.g. for a specific exponent and accuracy) consists of alternating open and closed intervals. Values at the boundary between open and closed intervals are "moved" to the closed interval. (this rule is the same as IEEE-754).

Since the interval center for approximate numbers is a design parameter, and exact and inexact numbers are not co-mingled, the choice for alt-754 rounding is that of "von Neumann" rounding: chop the excess least significant bits and replace by a single one bit. For unbiased rounding, chose the open and closed intervals so as to avoid carry propagation.

Corner treatment:

IEEE-754 uses denorms to represent values close to zero. The format chosen requires a normalization to occur in the memory read path and a denormalization to occur in the memory write path. IEEE-854 does not force a particular representation of denorms.

For alt-754 the number range is symmetric: loss of accuracy is supported at both ends of the number scale. One set of exponents is allocated to the low end and another set to the high end. The low end is used to represent zero and NaN, and the high end to represent the two infinities.

The alt-754 encoding scheme for denorms is that the mantissas remain normalized (with the usual leading hidden most significant one bit), but that the least significant part of the number contain an exponent adjustment consisting of (from left to right) a one bit followed by zero or more zero bits. The number of zero bits is the exponent adjustment: At the low end (near zero) the adjustment is subtracted and at the high end the adjustment is added to the exponent field to get the true exponent.

With symmetric corners the zero becomes a number range encompassing both positive and negative values near zero. The infinities also become extended ranges encompassing everything larger in magnitude than the highest high end corner values.

Variations:

The number of exponents allocated to the corners (underflow and overflow) is a design parameter. The choice of no corners corresponds to not implementing denorms in IEEE-754. The choice of all corners (all values are denorms) corresponds to accuracy gradually decreasing as values occur further from the center (e.g. +/- 1.0). The "all corners" choice provides a greatly expanded exponent range at the cost of the accuracy taper, e.g. as the values are farther from +/- 1.0 the mantissa accuracy decreases.

The number of bits allocated to the exponent and mantissa does not affect the model. Instead, shortening the mantissa or exponent or varying the corners only affects the subset of numerical values (e.g. set of specific ranges) implemented. E.g. a less accurate implementation lacks the finer resolution or the exponent range of the more accurate implementation. Formal statement: The model contains a superset of all possible implementations.

When more than one exponent code is allocated to each underflow and overflow corner there are unused codes which can be used to provide several NaNs and two zeros.

Accuracy tracking:

Two modes of operation of the arithmetic unit are possible: round to arithmetic unit accuracy and round to arithmetic result accuracy. Either way the arithmetic unit will use non-mantissa register bits to encode result accuracy. Then when the result is stored to memory, a trailing one bit code can be inserted to indicate remaining accuracy. The trailing one bit code matches the representation used for denorms and thus does not require additional hardware (unless both denorm and accuracy codes are used simultaneously).

Hardware issues:

The difficult parts of IEEE-754 are:

denorm processing,
rounding causing carries and possible overflow,
the intricacies of –0, exception handling, and four rounding modes.

Limitations of IEEE-754 are:

no accuracy tracking,
no gradual overflow,
asymmetric exponent range,
no provision for varying exponent range,
no provision for varying mantissa accuracy,
no provision for varying denorm "corners"

Denorm processing: IEEE-754 requires a normalization step for memory reads and a denormalization step for memory writes. Alt-754 requires examining the trailing one bit code

and adjusting the exponent for memory reads and the insertion of the trailing one bit code for memory writes.

Rounding for IEEE-754 requires (50% of the time) adding one to the mantissa. Occasionally the add one causes exponent overflow. Alt-754 chops the mantissa. The exponent never changes when rounding.

In retrospect IEEE-754 was developed in the era of micro-coded processors and non-pipelined arithmetic units. Rare, e.g. infrequent, cases do not have much effect on overall runtime or micro-code size. The current situation is that PCs use pipelining for add/subtract/multiply. Rare cases cause increased logic and increased delay in pipelined arithmetic units. Thus IEEE-754 is technologically dated.

Some features of IEEE-754 reflect historical biases:
Most early floating-point formats had a bit ordering which allowed an integer compare to be used on a pair of floats. This saved an op-code and allowed fast floating-point compares.
Mathematicians wanted features to make numerical analysis easier.
Embedded applications with their emphasis on low cost were not considered.
The trade-off between hardware and software (what was put into the instruction set versus what was put into the programming language) was slanted towards lowest cost hardware.

Summary:
A more through examination of the issues in floating-point design shows that some features of IEEE-754 have viable (and superior) alternatives. One set of design decisions is shown by alt-754. Alt-754 has a simple mathematical model to motivate the design and implementation decisions.

References:

1) Berkeley Design Technology, Inc. (1997). DSP on General-Purpose Processors: http://www.bdti.com/products/gpp604.htm. (floating-point exception processing on PPC 604)
2) Cavanagh, J.J.F. (1984) Von Neumann Rounding, pg 429-430 of Digital Computer Arithmetic: Design and Implementation, McGraw-Hill.
3) Cody, W.J. (1984) A proposed radix- and word-length-independent standard for floating-point arithmetic, pg 86-100 of IEEE Micro 4:4 (IEEE-854).
4) Coonen, J.T. (1980) An Implementation Guide to a Proposed Standard for Floating-Point Arithmetic, pg 68-79 of IEEE Computer, Jan 1980.
5) Goldberg, David (1991). What Every Computer Scientist Should Know About Floating-Point Arithmetic. ACM Computing Surveys, 23:1:5-48. or
Goldberg, David (1990) Appendix A: Computer Arithmetic, pg A1-A62 of Computer Architecture A Quantitative Approach by Hennessy & Patterson, Morgan-Kaufmann.)
6) Von Neumann, John (1945?) First Draft of a Report on the EDVAC, pg 50-51 of IEEE Annals of the History of Computing, 15:4, 1993. (Von Neumann rounding)
7) Woehr, Jack. (1997) A Conversation with William Kahan: How important is numerical accuracy? pg 18-32 of Dr. Dobbs Journal, #271 Nov. 1997.