IEEE coma flotante

De Wikipedia, la enciclopedia libre

El **estándar de la IEEE para aritmética en coma flotante** (**IEEE 754**) es el estándar más extendido para las computaciones en coma flotante, y es seguido por muchas de las mejoras de CPU y FPU. El estándar define formatos para la representación de números en coma flotante (incluyendo el cero) y valores desnormalizados, así como valores especiales como infinito y NaN, con un conjunto de **operaciones en coma flotante** que trabaja sobre estos valores. También especifica cuatro modos de redondeo y cinco excepciones (incluyendo cuándo ocurren dichas excepciones y qué sucede en esos momentos).

IEEE 754 especifica cuatro formatos para la representación de valores en coma flotante: precisión simple (32 bits), precisión doble (64 bits), precisión simple extendida (≥ 43 bits, no usada normalmente) y precisión doble extendida (≥ 79 bits, usualmente implementada con 80 bits). Sólo los valores de 32 bits son requeridos por el estándar, los otros son opcionales. Muchos lenguajes especifican qué formatos y aritmética de la IEEE implementan, a pesar de que a veces son opcionales. Por ejemplo, el lenguaje de programación C, ahora permite pero no requiere la aritmética de la IEEE (el tipo de C `float` es típicamente usado para la precisión simple de la IEEE y el tipo `double` usa la precisión doble del la IEEE).

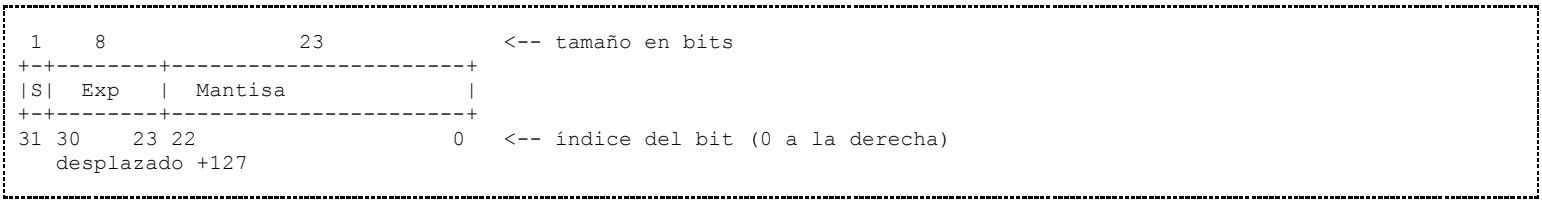
El título completo del estándar es **IEEE Standard for Binary Floating-Point Arithmetic (ANSI/IEEE Std 754-1985)**, y también es conocido por **IEC 60559:1989, Binary floating-point arithmetic for microprocessor systems** (originalmente el número de referencia era IEC 559:1989). [1] (http://www.opengroup.org/onlinepubs/009695399/frontmatter/refdocs.html)

Contenido

Anatomía de un número en coma flotante

Precisión simple 32-bits

Un número en coma flotante de precisión simple se almacena en una palabra de 32 bits.



donde `S` es el bit de signo y `Exp` es el campo exponente. (Para el signo: 0=Positivo ; 1= Negativo).

El exponente es desplazado en el un número en precisión simple, un exponente en el rango −126 a +127 es desplazado mediante la suma de 127 para obtener un valor en el rango 1 a 254 (0 y 255 tienen valores especiales descritos más adelante). Cuando se interpreta el valor en coma flotante, el número es desplazado de nuevo para obtener el exponente real.

El conjunto de valores posibles pueden ser divididos en los siguientes:

- ceros
- números normalizados
- números desnormalizados
- infinitos
- NaN (¬E, no es un número, como por ejemplo, la raíz cuadrada de un número negativo)

Las clases se distinguen principalmente por el valor del campo Exp, siendo modificada ésta por el campo fracción. Considera Exp y Fracción como campos de números binarios sin signo (Exp se encuentra en el rango 0–255):

Clase	Exp	Fracción
Ceros	0	0
Números desnormalizados	0	distinto de 0
Números normalizados	1-254	cualquiera
Infinitos	255	0
NaN (Not a Number)	255	distinto de 0

Para números normalizados, los más comunes, Exp es el exponente desplazado y Fracción es la parte fraccional de la mantisa (o significando). El número tiene valor v:

$v = s \times 2^e \times m$

Donde

s = +1 (números positivos) cuando S es 0

s = −1 (números negativos) cuando S es 1

e = Exp − 127 (en otras palabras, al exponente se le suma 127 y se almacena, a esto también se le llama "biased with 127" en inglés)

m = 1,Fracción en binario (esto es, el significando es el número binario 1 seguido por la coma decimal seguido por los bits de Fracción). Por lo tanto, $1 \leq m < 2$.

Notas:

1. Los números desnormalizados son iguales excepto que e = −126 y m = **0,Fracción**. (e NO es -127 : el significando ha de ser desplazado a la derecha por un bit más, de forma que incluya el bit principal, que no siempre es 1 en este caso. Esto se balancea incrementando el exponente a -126 para el cálculo.)
2. −126 es el menor exponente para un número desnormalizado
3. Hay dos ceros. +0 (S es 0) y −0 (S es 1)
4. Hay dos infinitos +∞ (S es 0) y −∞ (S es 1)
5. Los NaN s pueden tener un signo y un significando, pero estos no tienen otro significado que el que puedan aportar en pruebas de diagnóstico; el primer bit del significando es a menudo utilizado para distinguir *NaN s señalizados* de *NaN s silenciosos*
6. los NaNs y los infinitos tienen todos los bits a 1 en el campo Exp.

Un Ejemplo

Codifiquemos el número decimal -118,625 usando el sistema de la IEEE 754.

Necesitamos obtener el signo, el exponente y la fracción.

Dado que es un número negativo, el signo es "1". Busquemos los demás valores:

Primero, escribimos el número (sin signo) usando notación binaria. Mira el sistema de numeración binario para ver cómo hacer esto. El resultado es 1110110,101.

Ahora, movamos la coma decimal a la izquierda, dejando sólo un 1 a su izquierda.

$1110110{,}101 = 1{,}110110101 \cdot 2^6$ Esto es un número en coma flotante normalizado.

La mantisa es la parte a la derecha de la coma decimal, rellenada con ceros a la derecha hasta que obtengamos todos los 23 bits. Es decir 11011010100000000000000.

El exponente es 6, pero necesitamos convertirlo a binario y desplazarlo (de forma que el exponente más negativo es 0, y todos los exponentes son solamente números binarios no negativos). Para el formato IEEE 754 de 32 bits, el desplazamiento es 127, así es que 6 + 127 = 133. En binario, esto se escribe como 10000101.

Poniendo todo junto:

```
 1     8               23             <-- tamaño en bits
+-+--------+-----------------------+
|S|  Exp   |  Mantisa              |
|1|10000101|11011010100000000000000|
+-+--------+-----------------------+
31 30    23 22                    0   <-- índice del bit (0 a la derecha)
   desplazado +127
```

Enlaces externos

- Referencias IEEE 754 (http://babbage.cs.qc.edu/courses/cs341/IEEE-754references.html)
- Let's Get To The (Floating) Point por Chris Hecker (http://chrishecker.com/images/f/fb/Gdmfp.pdf)
- What Every Computer Scientist Should Know About Floating-Point Arithmetic by David Goldberg (http://docs.sun.com/source/806-3568/ncg_goldberg.html) - una buena introducción y explicación.
- Curso de Representación de los Datos de Carlos Pes (http://www.carlospes.com/curso_representacion_datos/)
- Comparing floating point numbers Bruce Dawson (http://www.cygnus-software.com/papers/comparingfloats/comparingfloats.htm)