Estándar IEEE 754

Organización de computadoras 2014

Universidad Nacional de Quilmes

El estándar IEEE 754 define representaciones para números de coma flotante con diferentes tipos de presición: simple y doble, utilizando anchos de palabra de 32 y 34 bits respectivamente. Estas representaciones son las que utilizan los procesadores de la familia x86, entre otros.
Estos sistemas, a diferencia de los anteriores, permiten representar también valores especiales, los cuales serán tratados posteriormente.

Precisión simple

En la representación de 32 bits, el exponente se representa en exceso de 8 bits, con un desplazamiento de 127, y la mantisa está representada en un sistema SM(24+1,23), es decir que:

- Se tienen 24 bits explicitos y uno implícito
- 23 bits son fraccionarios

Como es un sistema signo-magnitud, se tiene 1 bit de signo y 24 bits de magnitud. De los bits de la magnitud, 1 está implícwito y los otros 23 son los que se usan explícitamente. De aqui que estos 23 bits son fraccionarios y el bit implícito es entero.

Además, el total de 32 bits se escriben con el suiguiente formato:

S	Exponente: 8b	Magnitud: 23b

Precisión soble

De manera similar, en la representaciín IEEE de doble precisión, el bit mas significativo es utilizado para almacenar el signo de la mantisa, los siguientes 11 bits representan el exponente y los restantes 52 bits representan la mantisa. El exponente se representa en exceso de 11 bits, con un desplazamiento de 1023.

S	Exponente: 11b	Magnitud: 52b

Como en el caso de precisión simple, también se tiene una mantisa normalizada con un bit entero y los restantes fraccionarios, es decir que tiene la forma "1,X", donde X es el valor de los bits fraccionarios. Además, como se tiene un bit implícito, el dígito 1 (entero) está oculto y por lo tanto no es almacenado en la representación, permitiendo así ganar precisión.

Sin embargo, los parámetros usados en las representaciones de simple y doble precisión son los que se describen en la siguiente tabla:

	P. simple	P. doble
Cant. total de bits	32	64
Cant. de bits de la mantisa (*)	24	53
Cant. de bits del exponente	8	11
Mínimo exponente (emin) (**)	-126	-1022
Máximo exponente (emax) (**)	127	1023

(* incluyendo el bit implícito)
(** emin es -126 en lugar de -127, que corresponde al minimo valor del exceso(8,127), ver siguiente sección)

Nota:

Representación de valores especiales

Una cuestión de interés para los sistemas de numeración usados en las computadoras, es analizar qué sucede cuando una operación arroja como resultado un número indeterminado o un complejo. En estos casos el resultado constituye un valor especial para el sistema y se almacena como NaN (Not a Number) tal como ocurre al hacer, por ejemplo $\frac{\infty}{\infty}$ ó $\sqrt{-4}$.

A veces sucede que el resultado de una operación es muy pequeño y menor que el mínimo valor representable, en este caso se almacenará como +0 ó -0, dependiendo del signo del resultado. También se observa que al existir un 1 implícito en la mantisa no se puede represental el

valor cero como un número normal, por lo que éste es considerado un valor especial.

Por otro lado, ante una operación que arroje un resultado excesivamente grande (en valor absoluto), este se almacenará como $+\infty$ ó $-\infty$.

De las situaciones mencionadas, surge la necesidad de una representación para los valores especiales.

El exponente lo dice todo

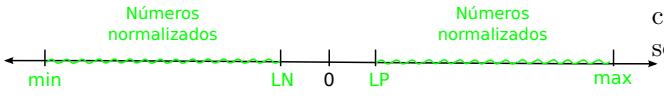
Es importante detenerse en la representación del exponente, que como se ha visto, utiliza el sistema Exceso con frontera no equilibrada (127 o 1023), lo que permite almacenar exponentes comprendidos en el rango [-127,128] en el sistema de precisión simple o [-1023,1024] en el sistema de precisión doble. Pues, puede verse en la tabla de la sección anterior que el rango entre emin y emax no cubre todo el rango disponible, y esto se debe a que se reservan las representaciones de emin-1 y emax+1 en ambas precisiones para representar valores especiales. Nótese que esta elección no es arbitraria: la cadena que representa emin-1 está compuesta de ceros y la cadena que representa el valor emax+1 está compuesta por unos, ambos fácilmente reconocibles.

Adicionalmente pueden representarse valores subnormales o denormalizados, es decir números **no normalizados**, de la forma $\pm 0, X * 2^{\delta}$, que se extienden en el rango comprendido entre el mayor número normal negativo y el menor número normal positivo. Dicho exponente especial δ tiene el valor -126.

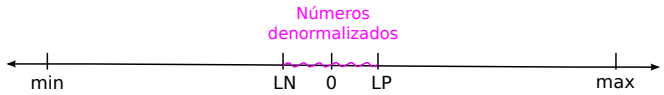
Nota: estos números desnormalizados no tienen bit implícito (ó es cero).

De esta manera, se definieron las siguientes clases de representaciones:

Números normalizados Las cadenas de esta clase se distinguen con un exponente que no sea nulo (cadena compuesta por 0s) ni saturado (cadena compuesta por 1s). Gráficamente, la clase de números normalizados se distribuye como sigue (LP=Límite positivo/ LN=Límite negativo):



Números denormalizados Las cadenas de esta clase tienen exponente nulo pero mantisa no nula. Gráficamente, la clase de números denormalizados se distribuye como sigue:



Ceros Esta clase incluye sólo dos cadenas: aquella compuesta con exponente y mantisa nulos, con ambos signos posibles. Esto permite representar el valor 0 positivo o negativo. Es importante notar que ninguna de las clases anteriores (normalizados o desnormalizados) permite construir por si misma el valor 0.

Infinitos Esta clase se identifica con exponente y mantisa saturados (`1..1`). Permite representar la situación en que el resultado está fuera del rango representable por los normalizados

Not a number (NaN) Esta clase se identifica con exponente saturado y es utilizada para representar los casos de error descriptos antes

La siguiente tabla resume cómo se distinguen las cadenas de las diferentes clases.

Exponente	Mantisa	Clase de número
0..0	0..0	± 0
0..0	$\neq$ 0.,0	Denormalizados: $\pm 0, X * 2^{emin}$
1..1	1..1	$\pm\infty$
1..1	$\neq$ 1.,1	NaN
[emin,emax]	cualquiera	Normalizados: $\pm 1, X * 2^{e}$

Ejemplos de interpretación

Cadena normalizada

Por ejemplo, se quiere interpretar la cadena en formato de precisión simple:

`1100 0010 0110 1011 1000 0000 0000 0000`

Para esto, es necesario separar los diferentes campos de la cadena:

1	10000100	110 1011 1000 0000 0000 0000
S	exponente:8b	Mantisa: 23b t

Dado que el exponente no es la cadena `00000000` ni la cadena `11111111`, se entiende que se lo debe interpretar como **un número en la clase normalizada**, por lo que se debe interpretar separadamente:

- Exponente: interpretar en Exc(8,127)

  $e = I_{ex}(10000100) = I_{bss}(10000100) - 127 = 2^7 + 2^2 - 127 = 5$

- Mantisa: Dado que hay un bit implícito cuyo peso es $2^0 = 1$.

  $$m = -(1 + I_{bss(23,23)}(11010111000000000000000)) =$$

  $$= -(1 + 2^{-1} + 2^{-2} + 2^{-4} + 2^{-6} + 2^{-7} + 2^{-8})$$

Cadena desnormalizada

El siguiente es un ejemplo de una cadena (en formato de precisión simple) cuyo exponente indica que es un número desnormalizado:

`0000 0000 0010 0000 1011 1000 0000 0000`

Separando los campos se obtiene:

0	00000000	010 0000 1011 1000 0000 0000
S	Exp:Exc(8,127)	Mant: BSS(23,23)

- Exponente: En este caso no se interpreta el exponente, sino que se usa el exponente especial

  $$e = -126$$

- Mantisa: Dado que **no hay bit implícito**

  $$m = I_{bss(23,23)}(01000001011100000000000) =$$

  $$= 2^{-2} + 2^{-8} + 2^{-10} + 2^{-11} + 2^{-12}$$