

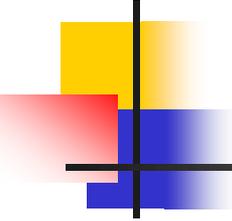
Computación – Curso 2024

Números en punto flotante

Prof. Jorge M. Runco

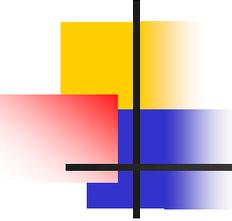
Lic. en Física Medica

Lic. en Física



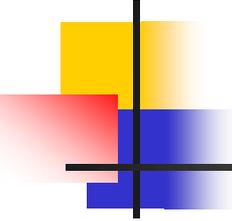
Temas de Clase

- Representación de números en Punto Flotante



Números en punto fijo

- Todos los números a representar tienen exactamente la misma cantidad de dígitos y la coma fraccionaria está siempre ubicada en el mismo lugar.
- La diferencia principal entre la representación en el papel y su almacenamiento en la computadora, es que no se guarda coma alguna, se supone que está en un lugar determinado.

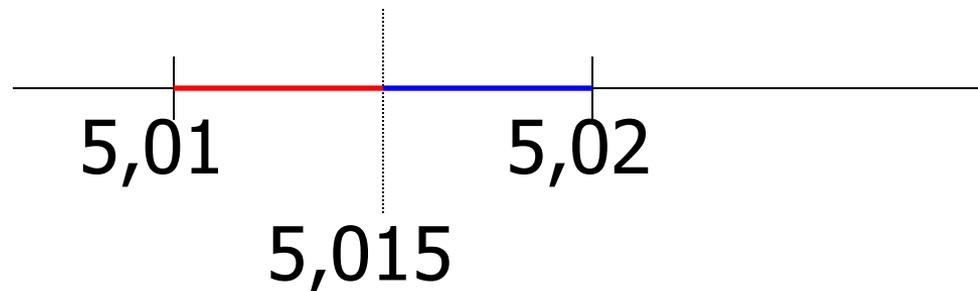


Rango y Resolución

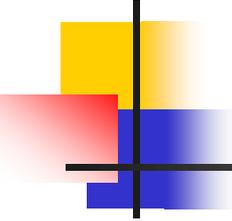
- **Rango**: desde el número mayor al menor que puede representarse
- **Resolución**: diferencia entre dos números consecutivos

Error en punto fijo (1)

- El máximo error cometido en una representación puede considerarse como la mitad de la diferencia (resolución) entre dos números consecutivos



- $5,01 \leq N^{\circ} \leq 5,015$ se representa por 5,01
- $5,015 < N^{\circ} \leq 5,02$ se representa por 5,02



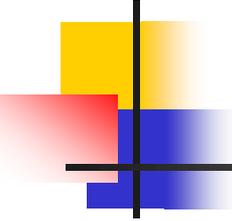
Error en punto fijo (2)

- En cualquiera de los dos casos el Error Absoluto máximo resulta ser:

$$EA \text{ max} = 5,015 - 5,01 = 0,005 \quad \text{ó}$$

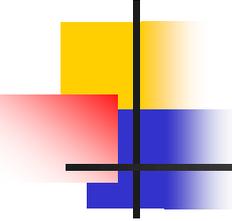
$$(5,02 - 5,01)/2 = 0,005$$

- Que corresponden a los N^o marcados en rojo ó azul.



Números en punto flotante

- En punto fijo (ej. Ca2), es posible representar un rango de enteros positivos y negativos centrados en 0.
- Suponiendo un número con componente fraccionaria, en este formato de punto fijo también se pueden representar números.
- Limitaciones: “números muy grandes y números muy pequeños”.



Números en punto flotante (2)

- Un número decimal “muy grande”:

976.000.000.000.000

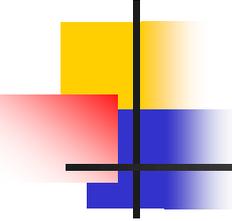
se puede representar como:

$$9,76 \times 10^{14}$$

- Un número decimal “muy pequeño”:

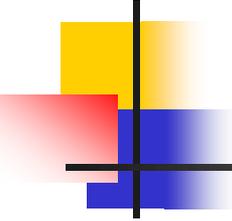
0,000000000000000976

$$9,76 \times 10^{-14}$$



Números en punto flotante (3)

- Lo que hemos hecho es desplazar en forma dinámica la coma decimal a una posición conveniente y usar el exponente de base 10 para mantener la “pista” de la coma.
- Esto permite tener un rango de números desde “muy pequeños” a “muy grandes” y pueden ser representados con pocos dígitos.



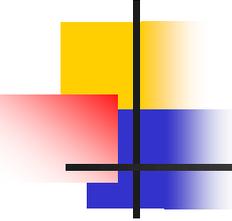
Números en punto flotante (4)

Veamos este mismo enfoque con números binarios:

- Un número se puede representar de la forma:

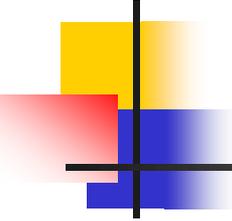
$$\pm M \times B^{\pm E}$$

- Este número se puede almacenar en una palabra binaria con dos campos:
 - Mantisa M
 - Exponente E



Números en punto flotante (5)

- La base B es implícita y no necesita almacenarse ya que es la misma para todos los números. Debemos almacenar M y E .
- Se necesitan menos bits para almacenar M y E , que para almacenar el “número completo” en la base correspondiente.

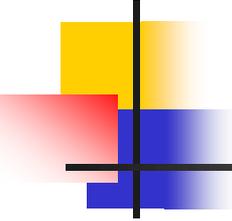


Números en punto flotante (6)

- ✓ M y E están representados en alguno de los sistemas en punto fijo que ya conocíamos como BSS, BCS, Ca2, Ca1, Exceso.

exponente	mantisa
-----------	---------

La figura muestra un formato típico

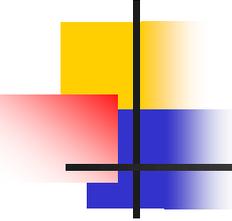


Ejemplo

❖ Supongamos el siguiente formato en punto flotante



Determinar el rango y resolución



Ejemplo 1

✓ Máximo = $1111 \times 2^{1111} = 15 \times 2^{15}$ ←

✓ Mínimo = 0 ←

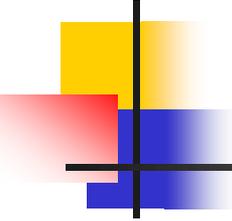
✓ Rango = $[0, \dots, 15 \times 2^{15}] = [0, \dots, 491520]$ ←

✓ Resolución en el extremo superior

$$R = (15 - 14) \times 2^{15} = 1 \times 2^{15} \quad \leftarrow$$

✓ Resolución en el extremo inferior

$$R = (1 - 0) \times 2^0 = 1 \quad \leftarrow$$



Ejemplo 2

Consideremos enteros de 8 bits y en BSS
Calcular el rango y resolución:

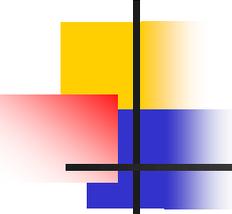
➤ Rango = [0,...,255]

➤ Resolución en el extremo superior

$$R = 255 - 254 = 1 \quad \longleftarrow$$

➤ Resolución en el extremo inferior

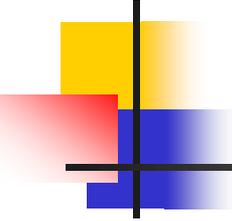
$$R = 1 - 0 = 1 \quad \longleftarrow$$



Comparación

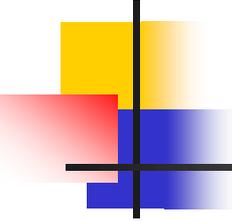
Si comparamos ambos ejemplos vemos:

- ✓ el rango en punto flotante es mayor
- ✓ la cantidad de combinaciones binarias distintas es la misma en ambos sistemas $2^8 = 256$ ←
- ✓ en punto flotante la resolución no es constante a lo largo del intervalo, como lo es en el segundo ejemplo.



Conclusión

- ✓ En el sistema de punto flotante el rango es mayor. Podemos representar números más grandes ó más pequeños que en un sistema de punto fijo (para igual cantidad de bits), pero pagamos el precio que los N^os no están igualmente espaciados, como en punto fijo.

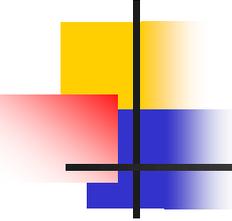


Mantisa y exponente en Ca2

- ❖ Ejemplo: supongamos el siguiente formato en punto flotante

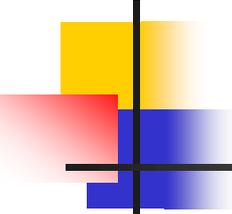


Determinar el rango y resolución



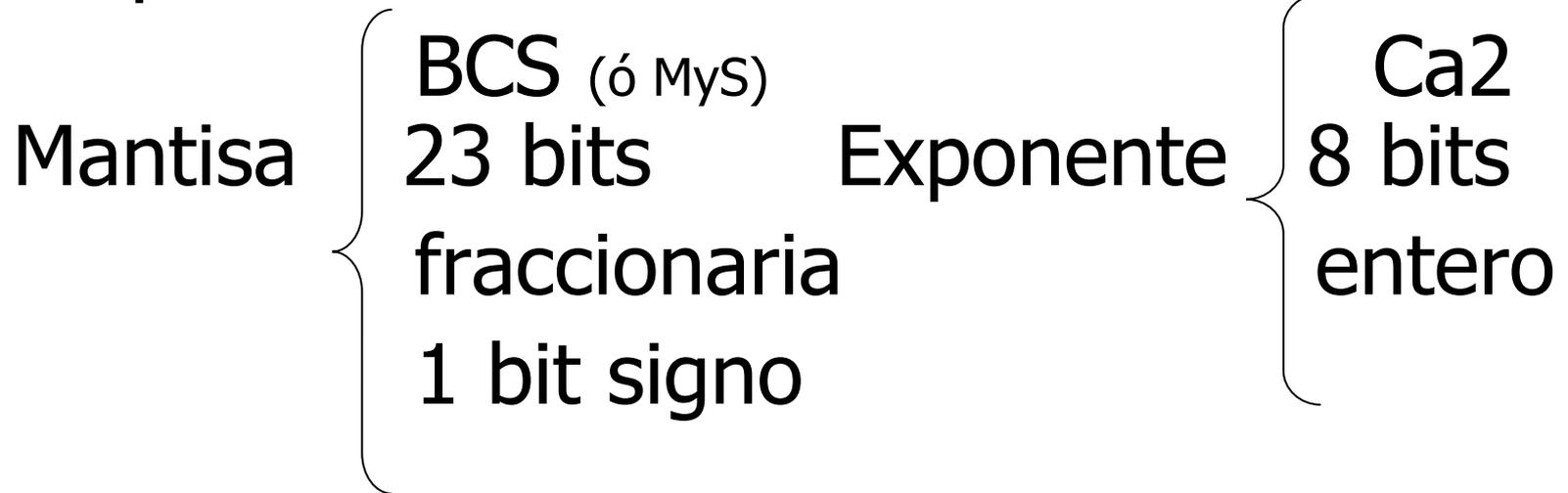
Mantisa y exponente en Ca2

- Máximo = $0111 \times 2^{0111} = +7 \times 2^{+7}$
- Mínimo = $1000 \times 2^{0111} = -8 \times 2^{+7}$
- Rango = $[-8 \times 2^{+7}, \dots, +7 \times 2^{+7}]$
- Resolución en el extremo superior
$$R = (7 - 6) \times 2^7 = 1 \times 2^7$$
- Resolución en el origen
$$R = (1 \times 2^{-8} - 0) = 1 \times 2^{-8}$$

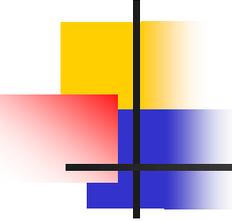


Mantisa fraccionaria

❖ Ejemplo: supongamos el siguiente formato en punto flotante



Determinar el rango y resolución



Mantisa fraccionaria

✓ Máximo positivo

$$0 \quad 0,111\dots111 \times 2^{01111111} = +(1-2^{-23}) \cdot 2^{+127}$$

✓ Mínimo positivo ($\neq 0$)

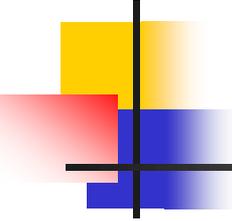
$$0 \quad 0,000\dots001 \times 2^{10000000} = +(2^{-23}) \cdot 2^{-128}$$

✓ Máximo negativo ($\neq 0$)

$$1 \quad 0,000\dots001 \times 2^{10000000} = - (2^{-23}) \cdot 2^{-128}$$

✓ Mínimo negativo

$$1 \quad 0,111\dots111 \times 2^{01111111} = -(1-2^{-23}) \cdot 2^{+127}$$

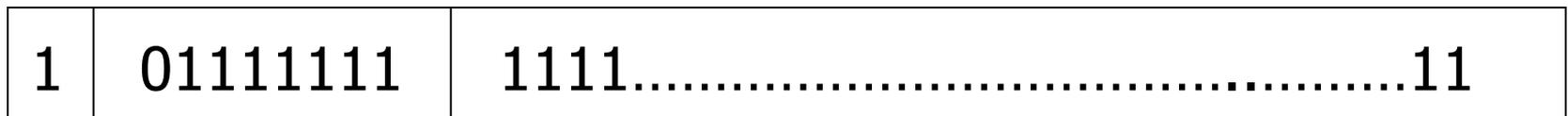


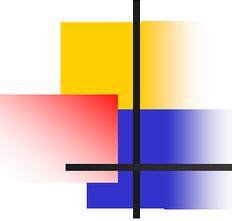
Formato final

- ❖ El formato anterior se puede representar



- ❖ El mínimo negativo es



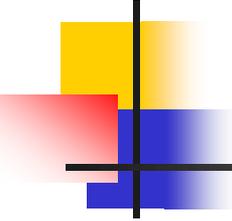


Normalización

Veamos el siguiente ejemplo:

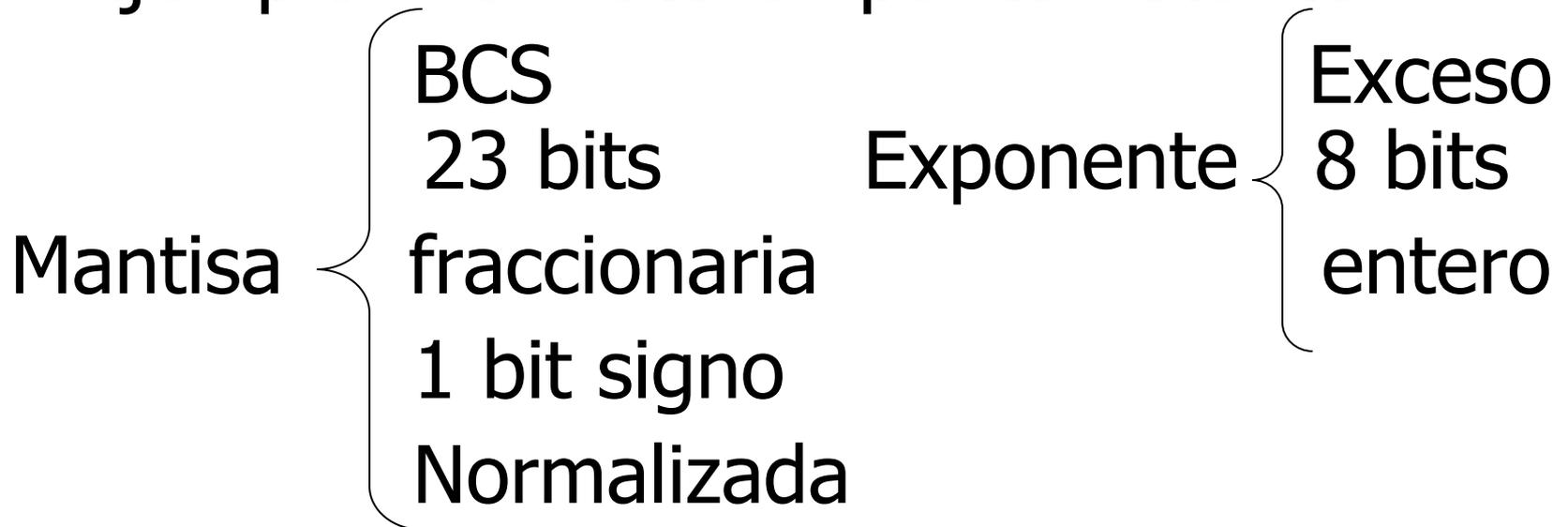
$$40 \times 10^0 = 4 \times 10^1 = 0,4 \times 10^2 = 400 \times 10^{-1}$$

- Existen distintos valores de mantisa y exponente para representar un mismo número.
- Lo mismo sucede en base 2.
- Con el objetivo de tener un único par de valores de mantisa y exponente para un número, se introduce la *normalización*.

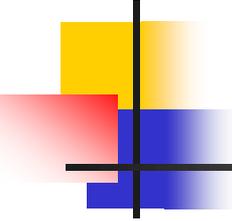


Normalización

- Ejemplo: formato en punto flotante



Determinar el rango y resolución



Normalización

✓ Máximo positivo

$$0 \quad 0,111\dots111 \times 2^{11111111} = +(1-2^{-23}) \cdot 2^{+127}$$

✓ Mínimo positivo ($\neq 0$)

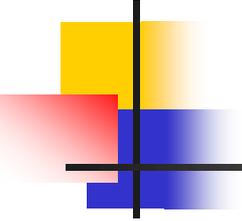
$$0 \quad 0,100\dots000 \times 2^{00000000} = +(0,5) \cdot 2^{-128}$$

✓ Máximo negativo ($\neq 0$)

$$1 \quad 0,100\dots000 \times 2^{00000000} = - (0,5) \cdot 2^{-128}$$

✓ Mínimo negativo

$$1 \quad 0,111\dots111 \times 2^{11111111} = -(1-2^{-23}) \cdot 2^{+127}$$

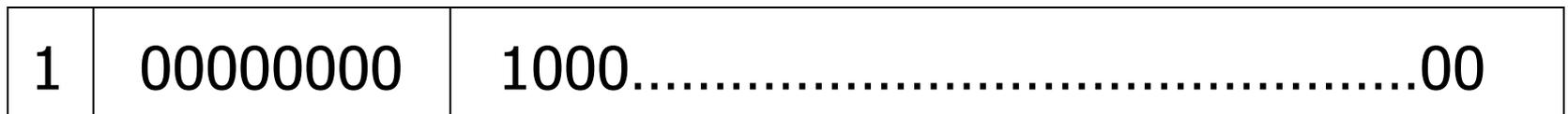


Normalización

- El formato anterior se puede representar



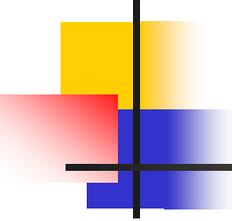
- El máximo negativo ($\neq 0$) es



Bit implícito

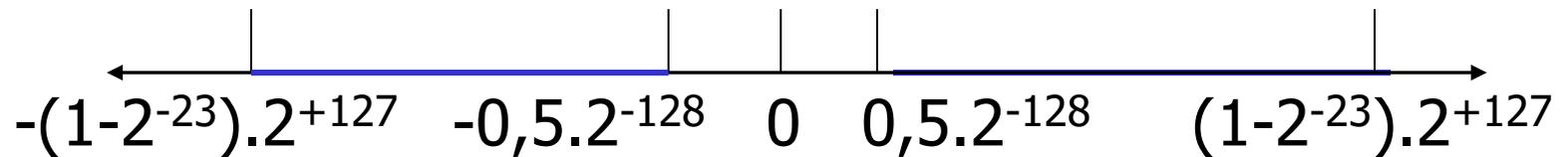
- Como todos los números comienzan con 0,1 ¿es necesario almacenar el 1?
 - siempre está presente !!!
- Si no lo almaceno, puedo “adicionar” un bit más a la mantisa. El bit no almacenado se conoce como *bit implícito*.



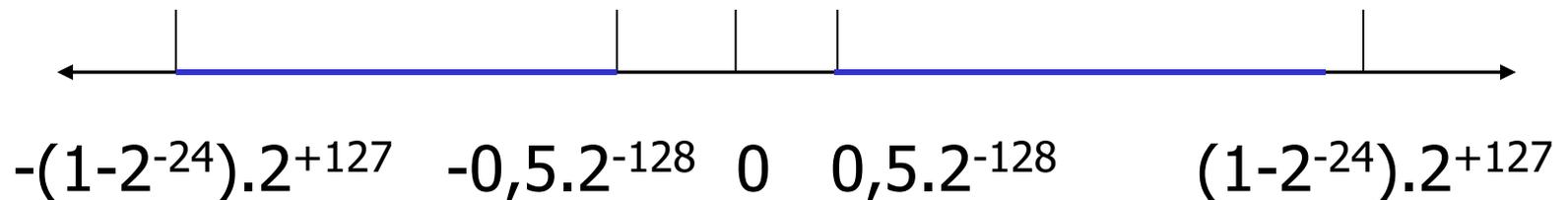


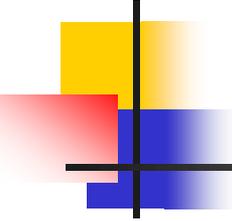
Recta numérica

- Sin bit implícito



- Con bit implícito





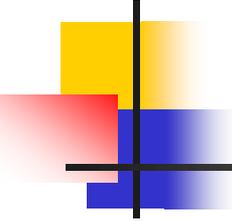
¿Cómo se escribe un N° en punto flotante normalizado?

1. Se escribe el N° en el sistema propuesto para la mantisa.
2. Se desplaza la coma y se cambia el exponente hasta obtener la forma normalizada.
3. Se convierte el exponente al sistema propuesto para él.

¿Cómo.....? (2)

Ej. - 13,5 . Formato anterior

- 1) $1\ 1101,100..0 = 1\ 1101,100..0 \times 2^0$
- 2) $1\ 0,110110..0 \times 2^4$
- 3) 4 en $Ca2 = 00000100$
4 en Exceso = 10000100
- Finalmente 



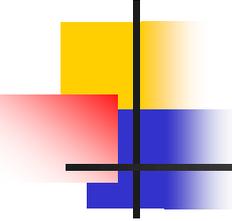
¿Cómo..... ? (3)

- Sin bit implícito

1	10000100	1101100000.....00
---	----------	-------------------

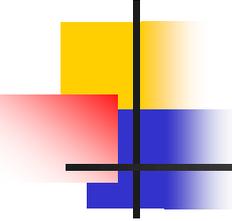
- Con bit implícito

1	10000100	101100000.....00
---	----------	------------------



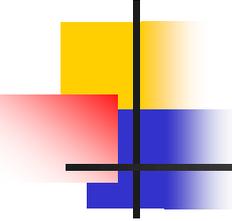
Resolución – Error absoluto

- Resolución: es la diferencia entre dos representaciones sucesivas, y varía a lo largo del rango, no es constante como en el caso de punto fijo.
- Error Absoluto: es la diferencia entre el valor representado y el valor a representar



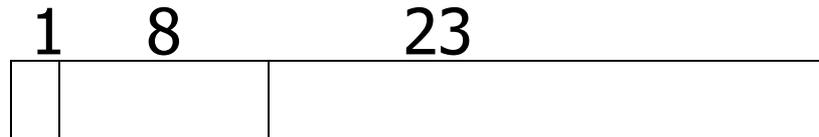
Error absoluto y relativo

- Error Absoluto máximo = Resolución/2
- Error Relativo = $EA/Número$ a representar



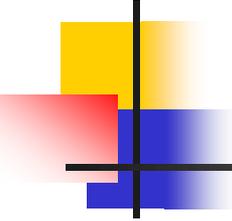
Estándar IEEE 754

➤ Simple precisión



➤ Doble precisión



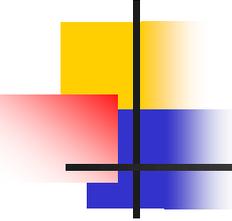


Estándar IEEE 754

➤ Mantisa: fraccionaria normalizada, con la coma después del primer bit que es siempre uno (1,) en M y S.

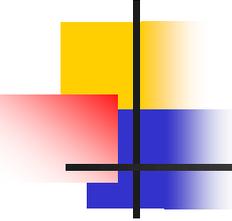
➤ Exponente: representado en exceso

$$2^{n-1} - 1$$



Estándar IEEE 754

	Simple	Doble precisión
Bits en signo	1	1
Bits en exponente	8	11
Bits en fracción	23	52
Total de bits	32	64
Exponente en exceso	127	1023
Rango de exponente	-126 a $+127$	-1022 a $+1023$
Rango de números	2^{-126} a $\sim 2^{128}$	2^{-1022} a $\sim 2^{1024}$



Ejemplo 1 en simple precisión

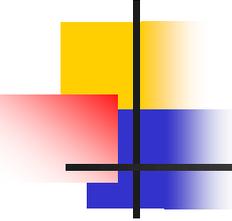
¿Qué valor representa el hexadecimal
3F800000?

0011 1111 1000 0000 0000 0000 0000 0000

01111111=127 en exceso 127 representa 0

000000000000000000000000=0

+ 1,0 x 2⁰ = 1



Ejemplo 2 en simple precisión

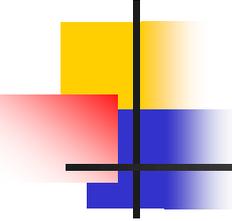
¿Qué valor representa el hexadecimal
C0066666?

1100 0000 0000 0110 0110 0110 0110 0110

10000000=128 en exceso 127 representa 1

00001100110011001100110=0,05

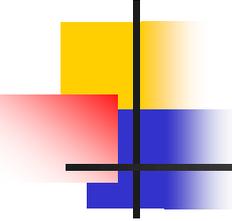
- 1,05 x 2¹ = -2,1



Estándar IEEE 754

Casos especiales:

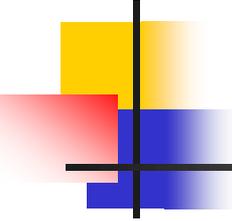
- $E = 255/2047, M \neq 0 \Rightarrow \text{NaN -Not a Number-}$
- $E = 255/2047, M = 0 \Rightarrow \text{Infinito}$
- $E = 0, M = 0 \Rightarrow \text{Cero}$
- $E = 0, M \neq 0 \Rightarrow \text{Denormalizado}$
 - $\pm 0, \text{mantisa}_s\text{-p } 2^{-126}$
 - $\pm 0, \text{mantisa}_d\text{-p } 2^{-1022}$



Operaciones aritméticas en pf

Sumar y restar

- Comprobar valores cero.
- Ajuste de mantisas (ajuste de exponentes).
- Sumar o restar las mantisas.
- Normalizar el resultado.



Operaciones aritméticas... (2)

Multiplicar y dividir

- Comprobar valores cero.
- Sumar y restar exponentes.
- Multiplicar y dividir mantisas
 - tener en cuenta el signo
- Normalizar.
- Redondear.

Todos los resultados intermedios deben doblar su longitud al almacenarse