Representação em Vírgula Flutuante

- A representação por vírgula flutuante utiliza-se quando se pretende aumentar a quantidade de números codificáveis à custa da perda de precisão.
- A representação em vírgula flutuante não é mais do que a notação científica utilizada nas máquinas de calcular.
- Qualquer número X é dado pela expressão geral:

$$X = \pm M \times b^{\pm E},$$

onde b é a base do sistema de numeração considerado (em decimal b=10, em binário b=2)

- Por exemplo, para $22{,}625_{(10)} \equiv 10110{,}101_{(2)}$, viria

$$22{,}625 \equiv 2{,}2625 \times 10^{1}$$

$$10110{,}101 \equiv 1{,}0110101 \times 10^{100}$$ (Note que $10_{(2)} \equiv 2_{(10)}$ e $100_{(2)} \equiv 4_{(10)}$)

Representação em Vírgula Flutuante

- Standard IEEE para aritmética de vírgula flutuante:
  ANSI/IEEE Standard 754-1985, Standard for Binary Floating Point Arithmetic

- Precisão simples
  O standard IEEE para a representação de números em vírgula flutuante utiliza uma palavra de 32 bits:

  S EEEEEEEE MMMMMMMMMMMMMMMMMMMMMMM

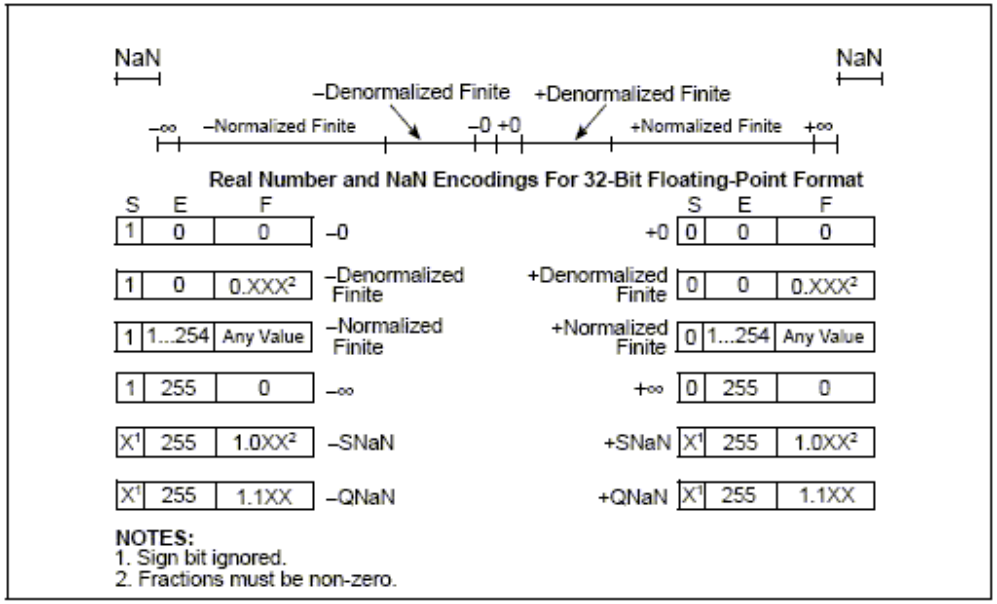
  O primeiro bit é o bit de sinal, S, os oito bits seguintes constituem o campo do expoente, E, e os últimos 23 bits constituem o campo designado por mantissa, M.

  O valor V representado pela palavra pode ser determinado do seguinte modo:

  - Se E=255 e M não é zero, então V = NaN (*"Not a number"*, não é um número)
  - Se E=255 e M é zero e S é 1, então V = -Infinito
  - Se E=255 e M é zero e S é 0, então V = Infinito
  - Se 0<E<255 então V=(-1)**S * 2**(E-127) * (1.M) onde "1.M" pretende representar o número binário criado pelo campo M, precedido por um 1 implícito e uma vírgula.
  - Se E=0 e M não é zero, então V=(-1)**S * 2**(-126) * (0.M)

    Estes são valores não normalizados. São definidos como números sem o 1 "escondido" e com o expoente o mais pequeno possível. Permitem tornar o efeito do *underflow* menos "abrupto". Por exemplo, $(0.0001)_2 \times 2^{-126}$ é um valor não normalizado que não tem representação normalizada em precisão simples. O suporte à representação de valores não normalizados é considerado pelo standard como opcional.
  - Se E=0 e M é zero e S é 1, então V=-0
  - Se E=0 e M é zero e S é 0, então V=0

Representação em Vírgula Flutuante



Real Numbers and NaNs

Em particular,

0 00000000 00000000000000000000000 = 0
1 00000000 00000000000000000000000 = -0

0 11111111 00000000000000000000000 = Infinito
1 11111111 00000000000000000000000 = -Infinito

0 11111111 00000100000000000000000 = NaN
1 11111111 00100010001001010101010 = NaN

0 10000000 00000000000000000000000 = +1 * 2**(128-127) * 1.0 = 2
0 10000001 10100000000000000000000 = +1 * 2**(129-127) * 1.101 = 6.5
1 10000001 10100000000000000000000 = -1 * 2**(129-127) * 1.101 = -6.5

0 00000001 00000000000000000000000 = +1 * 2**(1-127) * 1.0 = 2**(-126)
0 00000000 10000000000000000000000 = +1 * 2**(-126) * 0.1 = 2**(-127)
0 00000000 00000000000000000000001 = +1 * 2**(-126) * 0.00000000000000000000001 =
2**(-149) (menor valor positivo)

Representação em Vírgula Flutuante

- Precisão dupla

  O standard IEEE para a representação de números em vírgula flutuante utiliza uma palavra de 64 bits.

  S EEEEEEEEEEE MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM

  O primeiro bit é o bit de sinal, S, os onze bits seguintes constituem o campo do expoente, E, e os últimos 52 bits constituem o campo designado por mantissa, M.

  O valor V representado pela palavra pode ser determinado do seguinte modo:

  - Se E=2047 e M não é zero, então V = NaN ("Not a number", não é um número)
  - Se E=2047 e M é zero e S é 1, então V = -Infinito
  - Se E=2047 e M é zero e S é 0, então V = Infinito
  - Se 0<E<2047 então V=(-1)**S * 2**(E-1023) * (1.M)

    onde "1.M" pretende representar o número binário criado pelo campo M, precedido por um 1 implícito e uma vírgula.
  - Se E=0 e M não é zero, então V=(-1)**S * 2**(-1022) * (0.M).
  - Estes são valores não normalizados.
  - Se E=0 e M é zero e S é 1, então V=-0
  - Se E=0 e M é zero e S é 0, então V=0

Representação em Vírgula Flutuante

- Resumo

Precisão Simples = 32 bits= 1 / 8 / 23 (Sinal / Expoente / Mantissa)
Precisão Dupla = 64 bits= 1 / 11 / 52 (Sinal / Expoente / Mantissa)

$número_{(10)} = (-1)^{sinal} \times (1 + mantissa_{(10)}) \times 2^{(expoente_{(10)} - desvio)}$ ‡
(desvio=127 para 32 bits e 1023 para 64 bits)

Precisão Simples		Precisão Dupla		Excepção/Número
Expoente	**Mantissa**	**Expoente**	**Mantissa**	
0	0	0	0	0
0	<> 0	0	<> 0	Número não normalizado
1-254	qq. coisa	1-2046	qq. coisa	Número normalizado
255	0	2047	0	Infinito
255	<> 0	2047	<> 0	NaN (Not a Number)

‡ Para números não normalizados a fórmula é: $número_{(10)} = (-1)^{sinal} \times mantissa \times 2^{(-desvio+1)}$