

## Apoio 2

### Representação de Números em Ponto Flutuante

A representação de números inteiros (como 0, 1, 35, -617, 313, 25.987, -13.500) pode ser feita facilmente utilizando um conjunto de bits, organizados em bytes. Números ditos *reais*, ou seja, números que possuem uma parte inteira e uma parte fracionária, ambas separadas por vírgulas (como -1, 25 e 38, 313) requerem uma forma alternativa de representação. Esta forma alternativa de representação considera diferentes componentes do número. Embora seja comum expressar os números reais apenas com apoio da vírgula para separar a parte inteira da parte real, outras visões podem ser consideradas.

Exemplos de representação de números reais

$$\begin{aligned} -1,25 &= -1,25 * 10^0 = -0,125 * 10^{-1} = -0,0125 * 10^{-2} \\ 38,313 &= 38,313 * 10^0 = 0,38313 * 10^2 = 0,038313 * 10^3 \\ 3,1416 &= 3,1416 * 10^0 = 0,31416 * 10^1 = 0,031416 * 10^2 \\ 82,0 &= 82,0 * 10^0 = 0,82 * 10^2 = 0,082 * 10^3 \end{aligned}$$

*Atenção à notação.* Na notação brasileira a vírgula é utilizada para separar a parte inteira da parte fracionária de um número real. O uso do ponto é um facilitador introduzido para auxiliar na visualização do valor apresentado. A notação americana também utiliza ponto e vírgula. No entanto, a notação americana faz uso inverso: o ponto separa a parte inteira da fracionária e a vírgula é utilizada para auxiliar na visualização do número.

Observe nos exemplos acima que mesmo valores numéricos inteiros podem ser representados como números reais. Observe também que é possível criar uma representação genérica para tais números na forma:

$$\text{+/- número} * \text{base}^{\text{+/- expoente}}$$

A questão que se coloca é como criar uma representação deste valor numérico interna ao computador (utilizando, portanto, uma seqüência de bits/bytes) que seja ao mesmo tempo eficiente em termos de consumo de recursos e portátil. Uma forma comum é utilizar um conjunto de N bits (tipicamente N = 32 ou 64, para representar números em ponto flutuante com precisão simples ou dupla) para armazenar valores em ponto flutuante em uma forma *normalizada*.

*Normalizar.* Normalizar neste caso significa apresentar o valor assumindo um conjunto de regras padrão, de forma que todos os valores tenham a mesma estrutura.

Esta forma normalizada é a seguinte:

$$\text{+/- } 0,\text{mantissa} * \text{base}^{\text{+/- expoente}}$$

onde o número é sempre representado com “0,”, ou seja, a mantissa representa os dígitos significantes do número e o expoente regula a precisão da representação.

Exemplos de representação normalizada de números reais

$$\begin{aligned} -1,25 &= -0,125 * 10^{-1} \\ 38,313 &= 0,38313 * 10^2 \\ 3,1416 &= 0,31416 * 10^1 \\ 82,0 &= 0,82 * 10^2 \end{aligned}$$

Como os valores devem ser representados em uma seqüência de N bits, a configuração se dá da seguinte forma:

SN	SE	Expoente	Mantissa
N bits: x bits para o Expoente, y bits para a Mantissa			

onde:

- SN é o sinal do número (+/-): 1 bit
- SE é o sinal do expoente (+/-): 1 bit
- Expoente é o valor numérico do expoente do número normalizado: x bits
- Mantissa é o valor numérico dos dígitos significativos do número normalizado: y bits
- $N = 1 + 1 + x + y$

Deve-se observar que nesta representação a base não é apresentada. Isto é evidente uma vez que toda a representação interna se dá utilizando a base numérica binária. Portanto, a base é 2. Desta forma, pode-se afirmar que, considerando-se a representação acima, existem os seguintes limites de valores:

- O maior expoente possível é:  $2^x - 1$
- A maior mantissa possível é  $2^y - 1$
- O maior número real representável é  $+(0.111...1 \times 2^E)$ , sendo  $E = 2^x - 1$
- O menor número real é  $-(0.111...1 \times 2^E)$ , sendo  $E = 2^x - 1$
- O menor real positivo é  $+(0.100...0 \times 2^{-E})$ , sendo  $E = 2^x - 1$
- O maior real negativo é  $-(0.100...0 \times 2^{-E})$ , sendo  $E = 2^x - 1$

Esta faixa de valores representa também o alcance e a precisão da representação. O alcance diz respeito ao número de bits ocupados para armazenar o expoente: quanto maior for o número de bits para o expoente, maior espectro de alcance do número. A precisão está relacionada ao número de bits alocados para representar a mantissa: quanto maior o número de bits utilizado, maior o número de dígitos significativos que podem ser armazenados. Dependendo do número de bits nos dois casos, podem ocorrer situações de *overflow* e *underflow*.

Overflow. Termo corrente na computação para designar que um valor não pode ser representado pois seu valor excede a capacidade de armazenamento disponível.

Underflow. Termo corrente na computação para designar que um valor não pode ser representado pois está contido entre 0 e o menor valor real normalizado representável.

*Overflow* ocorre quando o valor do expoente não pode ser armazenado no número de bits reservados ao expoente. Não importa se este valor for positivo ou negativo. Já a situação de *underflow* é mais pitoresca: ela ocorre quando o número a ser representado é maior que 0 (zero) mas é menor que o menor número representável. Assim, por maior que seja o expoente, ocorre uma descontinuidade na representação dos números na faixa de números próximas a 0 (zero).

Note que imprecisão é diferente de *underflow*. Enquanto valores numéricos muito próximos a 0 (zero) não podem ser representados, a limitação do número de bits para mantissa incorre em perda de precisão do valor efetivo, sendo armazenado um valor aproximado (portanto impreciso) do valor correto.

Assim, pode-se considerar que a retirada de um bit da representação do expoente para representação da mantissa aumenta a precisão de representação, pois diminui o “passo” de representação do número. No entanto, a perda de um bit no expoente implica na diminuição da faixa de valores representável.

## Representação IEEE 754

O padrão de número 754 definido pelo Institute of Electrical and Electronics Engineers, também conhecido como norma IEEE 754, é empregado para representação de números em ponto flutuante em computadores. Este padrão foi definido em 1985 e é atualmente empregado na maioria dos processadores. A norma define como devem ser representados números em ponto flutuante com precisão simples (utilizando 32 bits) e com precisão dupla (64 bits). Em ambos os casos, a representação requer uma normalização do número com parte



