



Licenciatura Ciências de Computação  
Licenciatura Eng<sup>a</sup>. Física

2021/22

*A.J.Proença*

**Tema**

**Introdução aos Sistemas de Computação**



## Estrutura do tema ISC

1. Representação de informação num computador
2. Organização e estrutura interna dum computador
3. Execução de programas num computador
4. Análise das instruções de um processador
5. Evolução da tecnologia e da eficiência



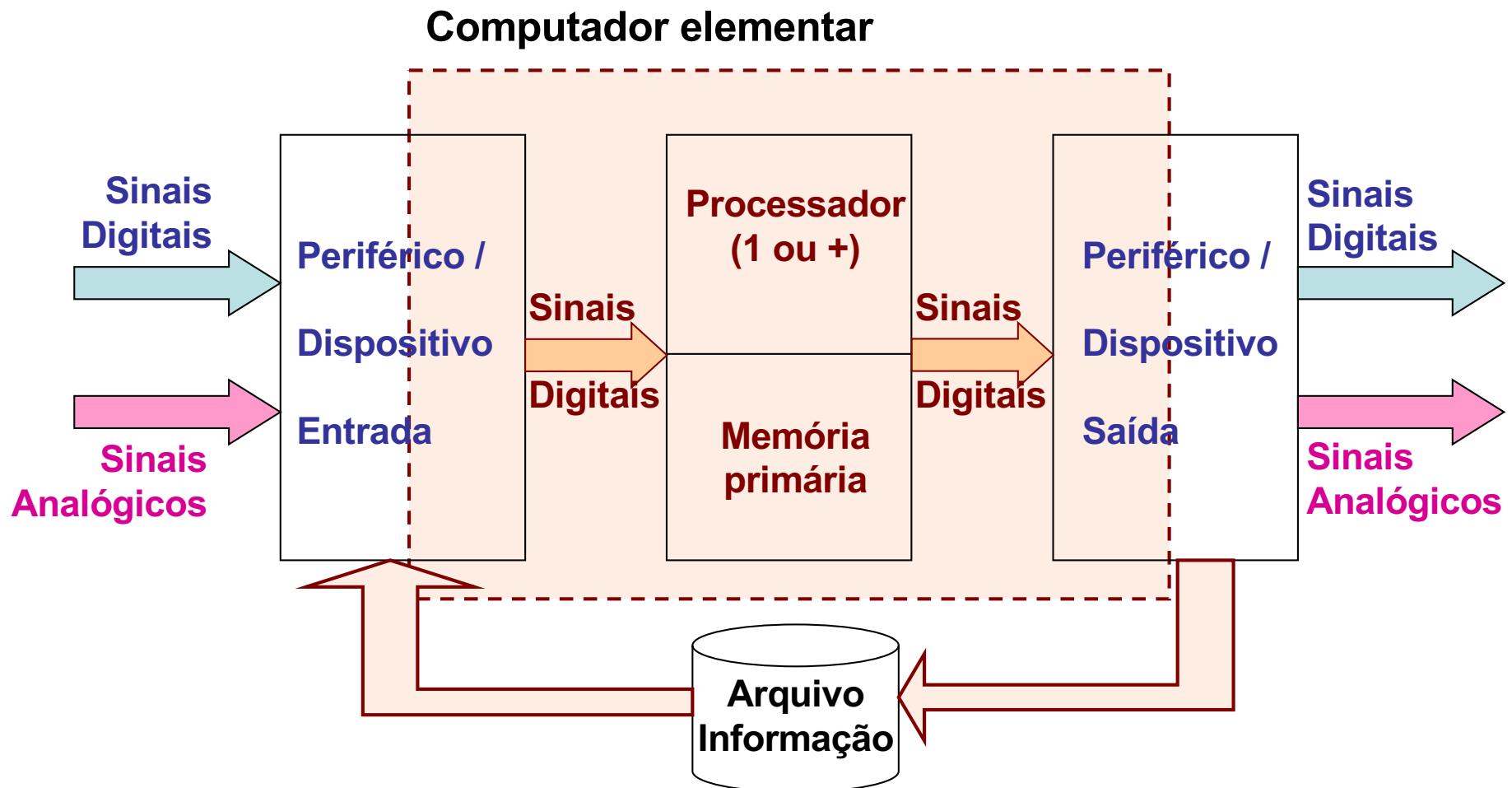
## Um computador é um sistema físico que:

- recebe informação,  
processa / arquiva informação,  
transmite informação, e ...
- é programável  
i.e., a funcionalidade do sistema pode ser modificada,  
sem alterar fisicamente o sistema

Quando a funcionalidade é fixada no fabrico do sistema onde o computador se integra, diz-se que o computador existente nesse sistema está “embebido”: ex. *smart phone*, máq. fotográfica, automóvel, ...

**Como se representa a informação num computador ?**  
**Como se processa a informação num computador ?**

## Noção de computador (2)





- **Como se representa a informação num computador ?**
  - representação da informação num computador ->
- **Como se processa a informação num computador ?**
  - organização e funcionamento de um computador ->

# *Representação da informação num computador (1)*



## Como se representa a informação? – com *binary digits*!

The screenshot shows the top portion of a Wikipedia page. On the left is the Wikipedia logo and the text "WIKIPÉDIA A encyclopédia livre". To the right, there are two blue buttons: "Artigo" and "Discussão". Below these buttons, the title "Algarismo" is displayed in a large, bold, black font.

Um **algarismo** ou **dígito**, é um tipo de representação (um símbolo numérico, como "2" ou "5") usado em combinações (como "25") para representar **números** (como o número 25) em **sistemas de numeração posicionais**. O nome "dígito" vem do facto de os 9 dígitos (do latim *digitem*, "dedo") das mãos corresponderem aos 10 símbolos do sistema de numeração comum de **base 10**, isto é, o decimal (digestivo do latim antigo *decoração* . que significa nove) dígitos.

A palavra "algarismo" tem sua origem no nome do famoso matemático **Al-Khwarizmi**.

Mais:

- Cada um dos elementos de um numeral é um algarismo ou dígito:
  - Numeral com 3 dígitos: 426.
  - Numeral com 10 algarismos: 1.234.567.890
- • Dígitos Binários: podem ser apenas dois, o 0 (zero) e o 1 (um)

# *Representação da informação num computador (2)*



## Como se representa a informação?

- com *binary digits!*

## Tipos de informação a representar:

- números (para cálculo)
  - » bases de numeração, inteiros (positivos e negativos)
  - » reais (*fp*), norma IEEE 754
- textos (caracteres alfanuméricos)
- conteúdos multimédia
- código para execução no computador

## *Sistemas de numeração: quanto vale na base 10 um nº representado numa base qualquer*



**1532.54<sub>10</sub>** (base 10) ; quanto vale cada algarismo?

$$1 \cdot 10^3 + 5 \cdot 10^2 + 3 \cdot 10^1 + 2 \cdot 10^0 + 5 \cdot 10^{-1} + 4 \cdot 10^{-2} = 1532.54_{10}$$

**Nota:** a potência de 10 dá-nos a ordem do algarismo no número...

**1532<sub>6</sub>** (base 6) ; quanto vale cada algarismo na base 10?

$$1 \cdot 6^3 + 5 \cdot 6^2 + 3 \cdot 6^1 + 2 \cdot 6^0 = 416_{10}$$

**1532<sub>13</sub>** (base 13) ; quanto vale cada algarismo na base 10?

$$1 \cdot 13^3 + 5 \cdot 13^2 + 3 \cdot 13^1 + 2 \cdot 13^0 = 3083_{10}$$

**110110.011<sub>2</sub>** (base 2) ; quanto vale cada algarismo na base 10?

$$1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} = \\ 54.375_{10}$$

## *Sistemas de numeração: como se passa um nº na base 10 para uma outra base*



**1532.54<sub>10</sub>** (base 10) ; algoritmo para extrair os algarismos?

- parte inteira: divisão sucessiva pela base e...
- parte decimal: multiplicação sucessiva pela base e...

**416<sub>10</sub>** ; quanto vale cada algarismo na base 6?

- parte inteira ... parte decimal ...

**3083<sub>10</sub>** ; quanto vale cada algarismo na base 13?

- parte inteira ... parte decimal ...

**54.375<sub>10</sub>**; quanto vale cada algarismo na base 2?

- parte inteira ... parte decimal ...

## *Sistemas de numeração: como se passa um nº na base 10 para uma outra base*



**1532.54<sub>10</sub>** (base 10) ; algoritmo para extrair os algarismos?

- parte inteira: divisão sucessiva pela base e...
- parte decimal: multiplicação sucessiva pela base e...

The diagram illustrates the division algorithm for extracting digits from the number 1532.54<sub>10</sub>. It shows two successive divisions by 10:

- The first division: 1532 ÷ 10. The quotient is 153 and the remainder is 2. The digit 2 is circled in red.
- The second division: 153 ÷ 10. The quotient is 15 and the remainder is 3.
- The third division: 15 ÷ 10. The quotient is 1 and the remainder is 5.

The remainders 2, 3, and 5 represent the digits extracted from the integer part of the number.

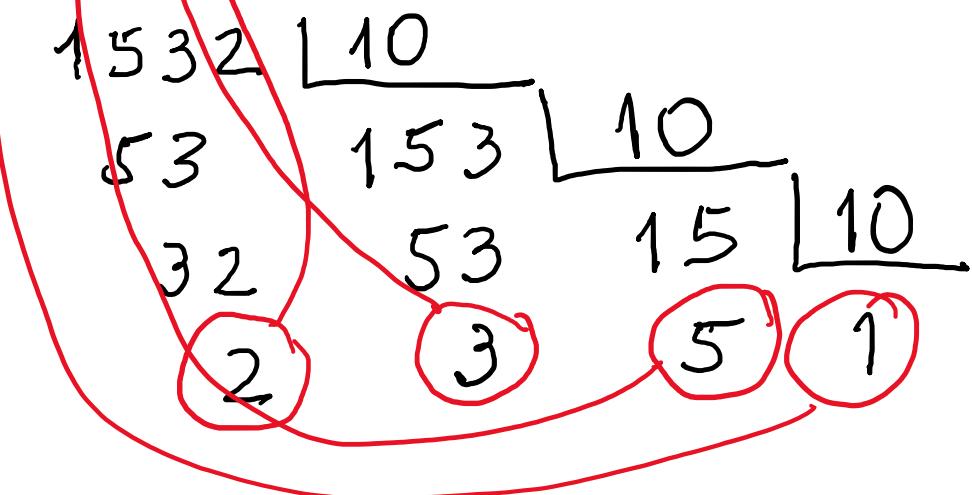
## *Sistemas de numeração: como se passa um nº na base 10 para uma outra base*



**1532.54<sub>10</sub>** (base 10) ; algoritmo para extrair os algarismos?

• parte inteira: divisão sucessiva pela base e...

• parte decimal: multiplicação sucessiva pela base e...



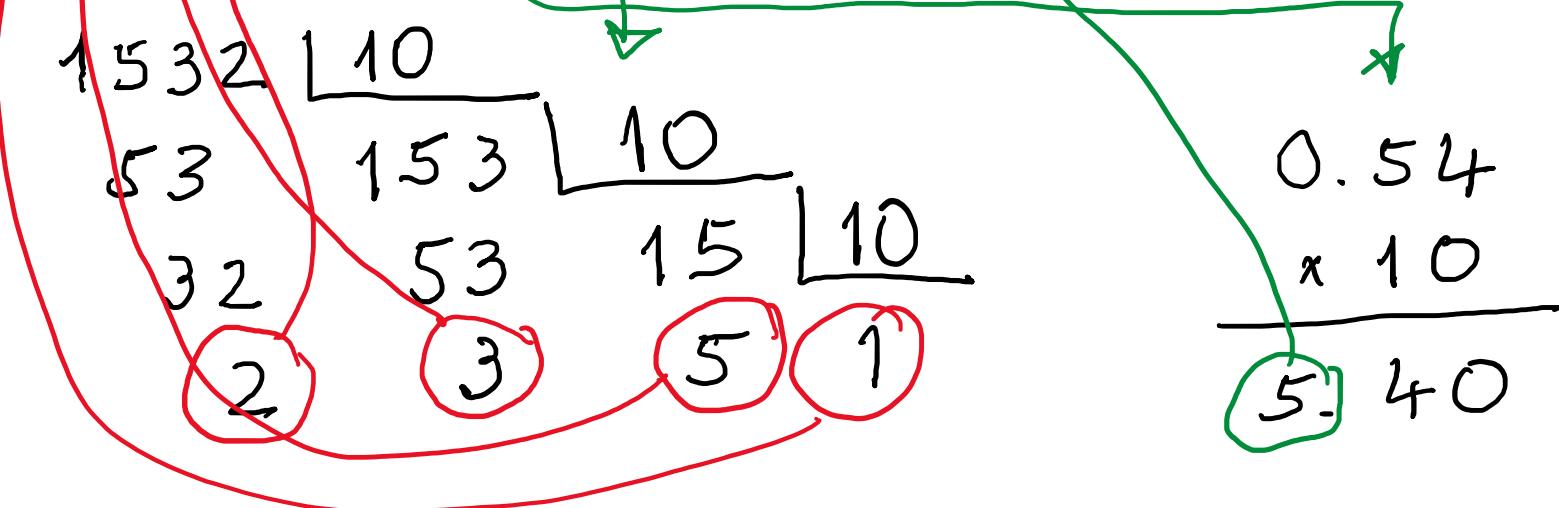
## Sistemas de numeração: como se passa um nº na base 10 para uma outra base



$1532.54_{10}$  (base 10) ; algoritmo para extrair os algarismos?

parte inteira: divisão sucessiva pela base e...

• parte decimal: multiplicação sucessiva pela base e...



## Sistemas de numeração : como se passa um nº na base 10 para uma outra base



$1532.54_{10}$  (base 10) ; algoritmo para extrair os algarismos?

parte inteira: divisão sucessiva pela base e...

• parte decimal: multiplicação sucessiva pela base e...

$$\begin{array}{r} 1532 \\ \times 10 \hline 1532 \\ 53 \quad | \\ 32 \quad 53 \\ \times 10 \hline 5 \quad 3 \\ \times 10 \hline 15 \quad 1 \\ \times 10 \hline 1 \end{array}$$

$$\begin{array}{r} 0.54 \\ \times 10 \hline 5.40 \\ \times 10 \hline 4.00 \end{array}$$

## *Sistemas de numeração: como se passa um nº na **base 10** para uma outra base*



**1532<sub>6</sub>** (base 6) ; quanto vale cada algarismo na base 10?

$$1 \cdot 6^3 + 5 \cdot 6^2 + 3 \cdot 6^1 + 2 \cdot 6^0 = 416_{10}$$

**416<sub>10</sub>** ; quanto vale cada algarismo na **base 6**?

- parte inteira ... parte decimal ...

4 1 6 1 6

## Sistemas de numeração: como se passa um nº na **base 10** para uma outra base

1532<sub>6</sub> (base 6) ; quanto vale cada algarismo na base 10?  
 $1 \cdot 6^3 + 5 \cdot 6^2 + 3 \cdot 6^1 + 2 \cdot 6^0 = 416_{10}$

416<sub>10</sub> ; quanto vale cada algarismo na **base 6**?  
• parte inteira ... parte decimal ...

The diagram shows the conversion of the decimal number 416 into base 6. It uses successive division by 6, with the remainders circled in red and labeled 2, 3, 5, and 1 respectively. The quotient 69 is also circled in red.

416<sub>10</sub>

69

09

11

6

2

3

5

1

## Sistemas de numeração: como se passa um nº na base 10 para uma outra base

110110.011<sub>2</sub> (base 2) ; quanto vale cada algarismo na base 10?  
 $1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} = 54.375_{10}$

54.375<sub>10</sub>; quanto vale cada algarismo na base 2?  
• parte inteira ... parte decimal ...

The diagram illustrates the conversion of the decimal number 54.375 to binary. It shows the division of the integer part (54) and the decimal part (0.375) by 2, with remainders circled in red.

For the integer part:

- 54 divided by 2 gives 27 remainder 0 (circled in red).
- 27 divided by 2 gives 13 remainder 1 (circled in red).
- 13 divided by 2 gives 6 remainder 1 (circled in red).
- 6 divided by 2 gives 3 remainder 0 (circled in red).
- 3 divided by 2 gives 1 remainder 1 (circled in red).
- 1 divided by 2 gives 0 remainder 1 (circled in red).

For the decimal part:

- 0.375 multiplied by 2 gives 0.75 remainder 1 (circled in red).
- 0.75 multiplied by 2 gives 1.5 remainder 0 (circled in red).
- 0.5 multiplied by 2 gives 1 remainder 1 (circled in red).

The remainders from the integer part divisions are read from bottom to top to form the binary integer part: 0110110. The remainders from the decimal part multiplication are read from top to bottom to form the binary decimal part: 011. Therefore, the binary representation of 54.375 is 0110110.011.

## Sistemas de numeração: como se passa um nº na base 10 para uma outra base

110110.011<sub>2</sub> (base 2) ; quanto vale cada algarismo na base 10?  
 $1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} =$   
 $54.375_{10}$

$54.375_{10}$ ; quanto vale cada algarismo na base 2?  
• parte inteira ... parte decimal ...

$$\begin{array}{r} 0.375 \\ \times 2 \\ \hline 0.750 \\ \times 2 \\ \hline 1.50 \\ \times 2 \\ \hline 1.00 \end{array}$$

$$\begin{array}{r} 54 \quad 1 \quad 2 \\ 14 \quad 27 \quad | 2 \\ 0 \quad 07 \quad 13 \quad | 2 \\ 1 \quad 16 \quad 6 \quad | 2 \\ 0 \quad 3 \quad 1 \quad | 2 \\ 1 \quad 1 \end{array}$$

## *Sistemas de numeração: caso particular da base 2*



**110110.011<sub>2</sub>** (base 2) ; quanto vale cada algarismo na base 10?

$$1*2^5 + 1*2^4 + 0*2^3 + 1*2^2 + 1*2^1 + 0*2^0 + 0*2^{-1} + 1*2^{-2} + 1*2^{-3} = \dots$$

Para simplificar:

• eliminar os produtos, ignorar parcelas com produtos por 0

$$\bullet 1*2^5 + 1*2^4 + 0*2^3 + 1*2^2 + 1*2^1 + 0*2^0 + 0*2^{-1} + 1*2^{-2} + 1*2^{-3} = \dots$$

$$\Rightarrow 2^5 + 2^4 + 2^2 + 2^1 + 1/2^2 + 1/2^3 = \dots$$

### Recomendações:

• decorar a tabuada das potências de 2 (**2<sup>0</sup> + 2<sup>10</sup>**) →

• compreender as potências de 2 múltiplas de 10 →

## *Numeração de base 2: dicas para uma rápida conversão de potências de 2 para a base 10*



|            |      |
|------------|------|
| $2^0 =$    | 1    |
| $2^1 =$    | 2    |
| $2^2 =$    | 4    |
| $2^3 =$    | 8    |
| $2^4 =$    | 16   |
| $2^5 =$    | 32   |
| $2^6 =$    | 64   |
| $2^7 =$    | 128  |
| $2^8 =$    | 256  |
| $2^9 =$    | 512  |
| $2^{10} =$ | 1024 |

$$\begin{aligned}2^{10} &= 1024 = 1 \text{ Ki(bi)} \approx 1000 = 10^3 = 1 \text{ K(ilo)} \\ \dots & \\ 2^{12} &= 2^2 * 2^{10} = 4 \text{ Ki(bi)} \approx 4000 = 4 * 10^3 = 4 \text{ K} \\ \dots & \\ 2^{16} &= 2^6 * 2^{10} = 64 \text{ Ki(bi)} \approx 64 * 10^3 = 64 \text{ K} \\ 2^{20} &= 1 \text{ Me(bi)} \approx 1000000 = 10^6 = 1 \text{ M(ega)} \\ 2^{30} &= 1 \text{ Gi(bi)} \approx 1000000000 = 10^9 = 1 \text{ G(iga)} \\ 2^{40} &= 1 \text{ Te(bi)} \approx 10^{12} = 1 \text{ T(era)} \\ 2^{50} &= 1 \text{ Pe(bi)} \approx 10^{15} = 1 \text{ P(eta)}\end{aligned}$$

## *Vantagens destas dicas*



- Na conversão de binário para base 10:

**110110.011<sub>2</sub>** (base 2)

$$\begin{aligned} \Rightarrow & 2^5 + 2^4 + 2^2 + 2^1 + 1/2^2 + 1/2^3 = \\ = & 32 + 16 + 4 + 2 + 1/4 + 1/8 = 54 + 0.375 \end{aligned}$$

- Na conversão de base 10 para binário:

$$\begin{aligned} 290 &= 256 + \\ 256 - \cancel{512} &\quad 2^8 \end{aligned}$$

## Vantagens destas dicas



- Na conversão de binário para base 10:

**110110.011<sub>2</sub>** (base 2)

$$\begin{aligned} \Rightarrow & 2^5 + 2^4 + 2^2 + 2^1 + 1/2^2 + 1/2^3 = \\ = & 32 + 16 + 4 + 2 + 1/4 + 1/8 = 54 + 0.375 \end{aligned}$$

- Na conversão de base 10 para binário:

$$\begin{array}{r} 290 = 256 + 32 + 2 \\ \quad \quad \quad 2^8 \quad 2^5 \quad 2^1 \\ -256 \\ \hline 34 \\ -32 \\ \hline 2 \end{array}$$

## Vantagens destas dicas



- Na conversão de binário para base 10:

**110110.011<sub>2</sub>** (base 2)

$$\begin{aligned} \Rightarrow & 2^5 + 2^4 + 2^2 + 2^1 + 1/2^2 + 1/2^3 = \\ = & 32 + 16 + 4 + 2 + 1/4 + 1/8 = 54 + 0.375 \end{aligned}$$

- Na conversão de base 10 para binário:

$$290 = 256 + 32 + 2$$

$\overline{-256}$        $2^8$        $2^5$        $2^1$   
 $\overline{34}$       ↕      ↕      ↓  
 $\overline{-32}$       1 0 0 1 0 0 0 1 0<sub>2</sub>  
 $\overline{2}$

## *Sistemas de numeração: caso particular da base 16 (hexadecimal)*



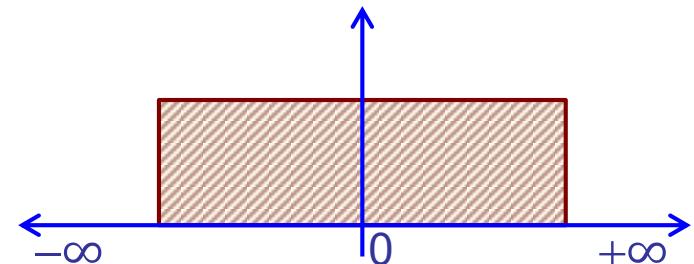
- Dígitos na base 16:  $0, 1, 2, \dots, 9,$        $10, 11, 12, 13, 14, 15$   
 $a, b, c, d, e, f$
- Vantagens sobre um valor de 32 bits:  
 $10100110100001110110010111010100_2$  **vs.**  $a68765d4_{16}$
- Facilidade de conversão:  
 $1010 \quad 0110 \quad 1000 \quad 0111 \quad 0110 \quad 0101 \quad 1101 \quad 0100_2$   
 $a \qquad 6 \qquad 8 \qquad 7 \qquad 6 \qquad 5 \qquad d \qquad 4_{16}$  ←
- Mesmo com ponto decimal: ← →  
 $1010011010000111011001011101.01_2$   
 $1010 \quad 0110 \quad 1000 \quad 0111 \quad 0110 \quad 0101 \quad 1101.0100_2$   
 $a \qquad 6 \qquad 8 \qquad 7 \qquad 6 \qquad 5 \qquad d \qquad . \quad 4_{16}$  ← →

# *Representação de inteiros*



## Gama de valores representáveis

- ideal: todos os valores e simetria em relação ao 0
- mas ...
- e quantos bits para representar um inteiro?



## Representação de positivos & negativos

- estratégias
- análise dum exemplo com todos os valores possíveis
  - S+M: Sinal + Magnitude/amplitude
  - Complemento para 1
  - Complemento para 2
  - Notação por excesso

## *Inteiros positivos e negativos: o universo com 3 bits (1)*



| Base 10 | Base 2 | S+M | Comp 1 | Comp 2 | Excesso |
|---------|--------|-----|--------|--------|---------|
| 0       | 000    |     |        |        |         |
| 1       | 001    |     |        |        |         |
| 2       | 010    |     |        |        |         |
| 3       | 011    |     |        |        |         |
| 4       | 100    |     |        |        |         |
| 5       | 101    |     |        |        |         |
| 6       | 110    |     |        |        |         |
| 7       | 111    |     |        |        |         |

## *Inteiros positivos e negativos: o universo com 3 bits (2)*



| Base 10 | Base 2 | S+M | Comp 1 | Comp 2 | Excesso |
|---------|--------|-----|--------|--------|---------|
| 0       | 000    | +   |        |        |         |
| 1       | 001    | +   |        |        |         |
| 2       | 010    | +   |        |        |         |
| 3       | 011    | +   |        |        |         |
| 4       | 100    | -   |        |        |         |
| 5       | 101    | -   |        |        |         |
| 6       | 110    | -   |        |        |         |
| 7       | 111    | -   |        |        |         |

## *Inteiros positivos e negativos: o universo com 3 bits (3)*



| Base 10 | Base 2 | S+M | Comp 1 | Comp 2 | Excesso |
|---------|--------|-----|--------|--------|---------|
| 0       | 000    | +0  |        |        |         |
| 1       | 001    | +1  |        |        |         |
| 2       | 010    | +2  |        |        |         |
| 3       | 011    | +3  |        |        |         |
| 4       | 100    | -0  |        |        |         |
| 5       | 101    | -1  |        |        |         |
| 6       | 110    | -2  |        |        |         |
| 7       | 111    | -3  |        |        |         |

## Inteiros positivos e negativos: o universo com 3 bits (4)



| Base 10 | Base 2 | S+M | Comp 1           | Comp 2 | Excesso |
|---------|--------|-----|------------------|--------|---------|
| 0       | 000    |     | +0               |        |         |
| 1       | 001    |     | +1               |        |         |
| 2       | 010    |     | +2               |        |         |
| 3       | 011    |     | +3               |        |         |
| 4       | 100    |     | -11 <sub>2</sub> | -3     |         |
| 5       | 101    |     |                  |        |         |
| 6       | 110    |     |                  |        |         |
| 7       | 111    |     |                  |        |         |

The table illustrates the 3-bit binary universe. It shows integers from 0 to 7 in Base 10, their corresponding Base 2 representations, and their values in S+M, Comp 1, and Comp 2 formats. The number 4 is highlighted with a blue oval and a blue arrow pointing to its two's complement representation -11<sub>2</sub>, which is equivalent to -3. The numbers 4, 5, 6, and 7 are also circled in blue.

## Inteiros positivos e negativos: o universo com 3 bits (5)



| Base 10 | Base 2 | S+M | Comp 1                 | Comp 2 | Excesso |
|---------|--------|-----|------------------------|--------|---------|
| 0       | 000    |     | +0                     |        |         |
| 1       | 001    |     | +1                     |        |         |
| 2       | 010    |     | +2                     |        |         |
| 3       | 011    |     | +3                     |        |         |
| 4       | 100    |     | $-11_2 \rightarrow -3$ |        |         |
| 5       | 101    |     | $-10_2 \rightarrow -2$ |        |         |
| 6       | 110    |     | $-01_2 \rightarrow -1$ |        |         |
| 7       | 111    |     | $-00_2 \rightarrow -0$ |        |         |

## Inteiros positivos e negativos: o universo com 3 bits (6)



| Base 10 | Base 2 | S+M | Comp 1                 | Comp 2                     | Excesso |
|---------|--------|-----|------------------------|----------------------------|---------|
| 0       | 000    |     | +0                     | +0                         |         |
| 1       | 001    |     | +1                     | +1                         |         |
| 2       | 010    |     | +2                     | +2                         |         |
| 3       | 011    |     | +3                     | +3                         |         |
| 4       | 100    |     | $-11_2 \rightarrow -3$ | $-(11+1)_2 \rightarrow -4$ |         |
| 5       | 101    |     | $-10_2 \rightarrow -2$ |                            |         |
| 6       | 110    |     | $-01_2 \rightarrow -1$ |                            |         |
| 7       | 111    |     | $-00_2 \rightarrow -0$ |                            |         |

## *Inteiros positivos e negativos: o universo com 3 bits (7)*



| Base 10 | Base 2 | S+M | Comp 1                 | Comp 2                     | Excesso |
|---------|--------|-----|------------------------|----------------------------|---------|
| 0       | 000    |     | +0                     | +0                         |         |
| 1       | 001    |     | +1                     | +1                         |         |
| 2       | 010    |     | +2                     | +2                         |         |
| 3       | 011    |     | +3                     | +3                         |         |
| 4       | 100    |     | $-11_2 \rightarrow -3$ | $-(11+1)_2 \rightarrow -4$ |         |
| 5       | 101    |     | $-10_2 \rightarrow -2$ | $-(10+1)_2 \rightarrow -3$ |         |
| 6       | 110    |     | $-01_2 \rightarrow -1$ | $-(01+1)_2 \rightarrow -2$ |         |
| 7       | 111    |     | $-00_2 \rightarrow -0$ | $-(00+1)_2 \rightarrow -1$ |         |

## Vantagem da representação em complemento para 2: mesmo hardware para soma de inteiros com ou sem sinal (1)

Base 2

The diagram illustrates the conversion of base 10 numbers to base 2 and their addition using binary complements. It shows two tables: one for base 10 to base 2 conversion and another for binary complements.

| Base 2     | Base 10  |
|------------|----------|
| 000        | 0        |
| 001        | 1        |
| 010        | 2        |
| 011        | 3        |
| 100        | 4        |
| 101        | 5        |
| <b>110</b> | <b>6</b> |
| 111        | 7        |

| Comp 2                     | Base 2 |
|----------------------------|--------|
| +0                         | 000    |
| +1                         | 001    |
| +2                         | 010    |
| +3                         | 011    |
| $-(11+1)_2 \rightarrow -4$ | 100    |
| $-(10+1)_2 \rightarrow -3$ | 101    |
| $-(01+1)_2 \rightarrow -2$ | 110    |
| $-(00+1)_2 \rightarrow -1$ | 111    |

$001 + 101 = 110$

The diagram shows the conversion of base 10 numbers to base 2 and their addition using binary complements. The first table shows the conversion of base 10 numbers 0 to 7 to their binary equivalents. The second table shows the binary complements for numbers from -1 to -4. The addition of 001 and 101 is shown as  $001 + 101 = 110$ . The circled row in the first table corresponds to the circled row in the second table, indicating that the same hardware can be used for both addition and subtraction.

## Vantagem da representação em complemento para 2: mesmo hardware para soma de inteiros com ou sem sinal (2)

The diagram illustrates the equivalence between three representations of integers from 0 to 7:

- Base 2:** A column of binary numbers: 000, 001, 010, 011, 100, 101, 110, 111.
- Base 10:** A column of decimal numbers: 0, 1, 2, 3, 4, 5, 6, 7.
- Comp 2:** A column of binary numbers representing the two's complements of the base 10 values. The values are: +0 (000), +1 (001), +2 (010), +3 (011), -(11+1)<sub>2</sub> (-4) (100), -(10+1)<sub>2</sub> (-3) (101), -(01+1)<sub>2</sub> (-2) (110), and -(00+1)<sub>2</sub> (-1) (111).

Arrows indicate the mapping between the columns:

- A green arrow points from the "Base 2" column to the "Base 10" column.
- A red arrow points from the "Base 2" column to the "Comp 2" column.
- A red arrow labeled "+(-3)" points from the "Base 2" column to the "-2" entry in the "Comp 2" column.
- A red arrow labeled "A" points from the "-2" entry in the "Comp 2" column back to the "-2" entry in the "Base 2" column.

## Vantagem da representação em complemento para 2: mesmo hardware para soma de inteiros com ou sem sinal (3)

The diagram illustrates the equivalence between three number representations: Base 2, Base 10, and Complement 2 (Base 2). The numbers 0 through 7 are shown in each format.

| Base 2 | Base 10 | Base 2 | Comp 2                     | Base 2 |
|--------|---------|--------|----------------------------|--------|
| 000    | 0       | 1      | +0                         | 000    |
| 001    | 1       |        | +1                         | 001    |
| 010    | 2       |        | +2                         | 010    |
| 011    | 3       |        | +3                         | 011    |
| 100    | 4       |        | $-(11+1)_2 \rightarrow -4$ | 100    |
| 101    | 5       |        | $-(10+1)_2 \rightarrow -3$ | 101    |
| 110    | 6       |        | $-(01+1)_2 \rightarrow -2$ | 110    |
| 111    | 7       |        | $-(00+1)_2 \rightarrow -1$ | 111    |

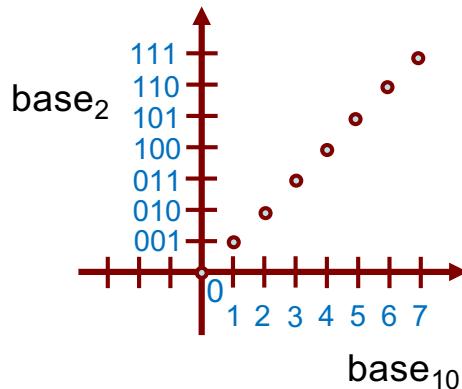
Handwritten annotations include:

- A green oval highlights "Base 10" in the first column.
- A red oval highlights "Comp 2" in the fourth column.
- A green arrow points from the "Base 10" column to the "Base 2" column.
- A red arrow points from the "Comp 2" column to the "Base 2" column.
- A green bracket underlines the row for 6 (110).
- A blue bracket underlines the row for -2 (110).
- A handwritten note "mesmos somadores em binário!" is written below the table.
- A handwritten note "1" is written near the bottom left of the table area.

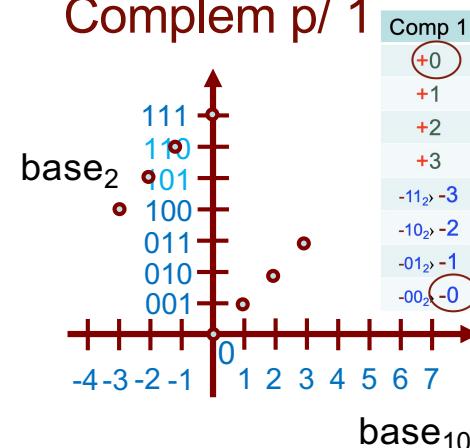
# Inteiros positivos e negativos: porquê mais uma maneira de codificação?



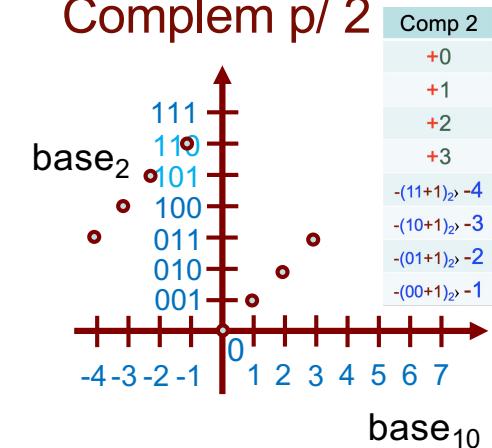
Sem sinal



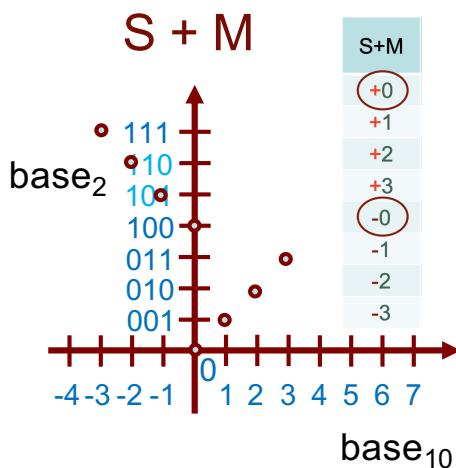
Complem p/ 1



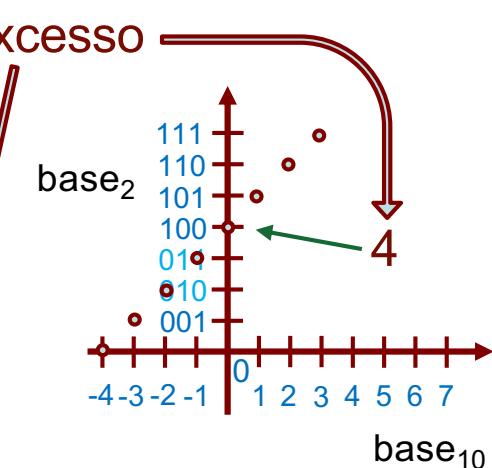
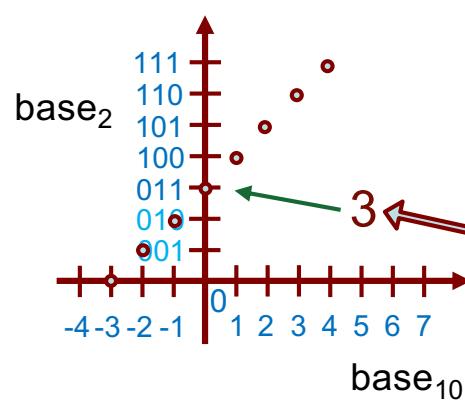
Complem p/ 2



S + M



Notação por excesso



## Inteiros positivos e negativos: o universo com 3 bits (8)



| Base<br>10 | Base 2 | S+M | Comp 1 | Comp 2 | Excesso $2^{n-1}-1$ |
|------------|--------|-----|--------|--------|---------------------|
| 0          | 000    |     |        |        | 0 - 3 → -3          |
| 1          | 001    |     |        |        | 1 - 3 → -2          |
| 2          | 010    |     |        |        | 2 - 3 → -1          |
| 3          | 011    |     |        |        | 3 - 3 → 0           |
| 4          | 100    |     |        |        | 4 - 3 → +1          |
| 5          | 101    |     |        |        | 5 - 3 → +2          |
| 6          | 110    |     |        |        | 6 - 3 → +3          |
| 7          | 111    |     |        |        | 7 - 3 → +4          |

Nota:  $n = \# \text{bits}$ ,  $2^{n-1}-1 = 2^{3-1}-1 = 2^2-1 = 3$

## Inteiros positivos e negativos: o universo com 3 bits (9)



| Base<br>10 | Base<br>2 | S+M | Comp<br>1 | Comp<br>2 | Excesso<br>$2^{n-1}$ | Excesso<br>$2^{n-1}-1$ |
|------------|-----------|-----|-----------|-----------|----------------------|------------------------|
| 0          | 000       |     |           |           | 0 - 4 > -4           | 0 - 3 > -3             |
| 1          | 001       |     |           |           | 1 - 4 > -3           | 1 - 3 > -2             |
| 2          | 010       |     |           |           | 2 - 4 > -2           | 2 - 3 > -1             |
| 3          | 011       |     |           |           | 3 - 4 > -1           | 3 - 3 > 0              |
| 4          | 100       |     |           |           | 4 - 4 > 0            | 4 - 3 > +1             |
| 5          | 101       |     |           |           | 5 - 4 > +1           | 5 - 3 > +2             |
| 6          | 110       |     |           |           | 6 - 4 > +2           | 6 - 3 > +3             |
| 7          | 111       |     |           |           | 7 - 4 > +3           | 7 - 3 > +4             |

Nota:  $n = \# \text{bits}$ ,  $2^{n-1} = 2^{3-1} = 2^2 = 4$ ,  $2^{n-1}-1 = 2^{3-1}-1 = 2^2-1 = 3$

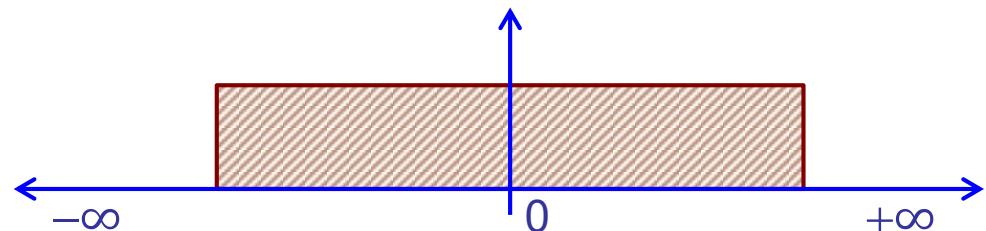
## *Representação de reais em vírgula flutuante (1)*



- **Gama de valores**

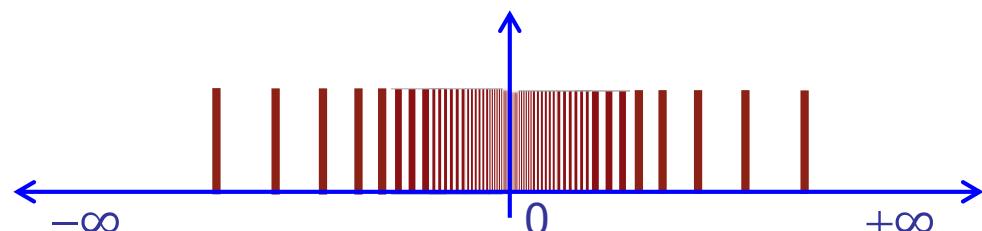
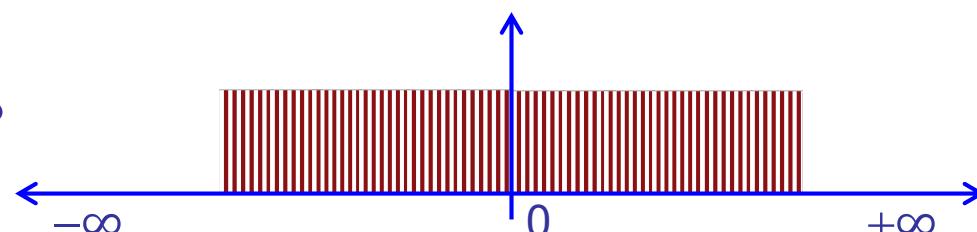
- esta gama é viável?
  - não, porque...  
**entre quaisquer 2 nºs reais há um conjunto infinito de valores**
  - então faz sentido a  
seguinte representação?

Notação vírgula fixa



- ou faria mais sentido a  
seguinte representação?

Notação científica



## *Representação de reais em vírgula flutuante (2)*

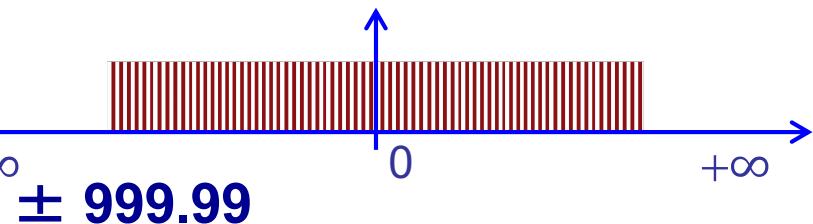


- Notação em vírgula fixa vs. notação científica

- Exemplos na base 10 usando apenas 5 dígitos

- Vírgula fixa:  $\pm X \ X \ X . X \ X$

maior valor representável :



$\pm 999.99$

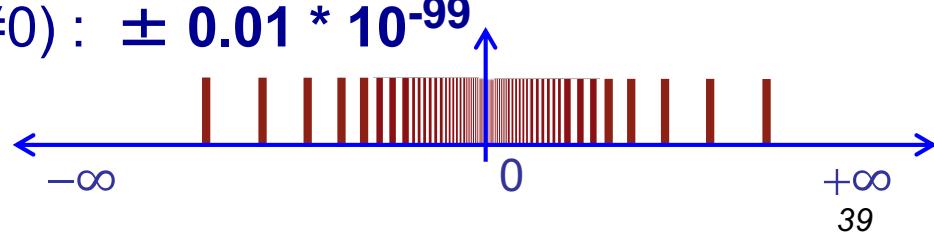
menor valor representável ( $\neq 0$ ) :  $\pm 0.01$

**mantissa**      **expoente**  
 $\underbrace{\quad\quad\quad}_{\text{mantissa}} \quad \underbrace{\quad\quad\quad}_{\text{expoente}}$

- Notação científica ou vírgula flutuante:  $\pm X . X \ X * 10^{\pm X \ X}$

maior valor representável :  $\pm 9.99 * 10^{99}$

menor valor representável ( $\neq 0$ ) :  $\pm 0.01 * 10^{-99}$



39

# *Representação de reais em vírgula flutuante (3)*

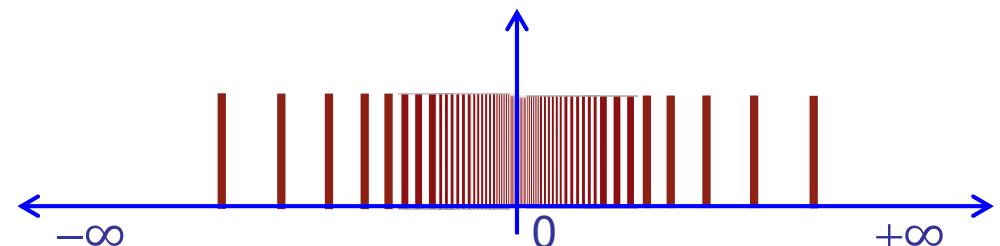


- **Normalização na representação**
  - Para garantir que cada nº a representar usa apenas uma única sequência de dígitos, há que fixar o local para o ponto decimal
  - Nº normalizado: à esquerda do ponto decimal apenas 1 dígito ≠ 0
$$\pm y \cdot x \times \dots \times \text{Radix}^{\text{Exp}} \quad 0 < y \leq \text{maior\_dígito}$$
  - Menor valor normalizado representável:
$$\pm 1.000 \dots 0 * 10^{-9\dots 9}$$
  - Nota\_1: a normalização não permite representar o zero  
Nota\_2: a normalização desperdiça muitos dígitos → subnormais
  - Nº subnormal: expoente fixo (menor normalizado), 0 à esq do ponto de  $\pm 0.000 \dots 0 * 10^{-9\dots 9}$  a  $\pm 0.999 \dots 9 * 10^{-9\dots 9}$

## *Representação binária de reais em vírgula flutuante (4)*



- **Formato binário dum valor em fp**
  - formato inclui mantissa e expoente; radix é implícito (=2)
  - mantissa e expoente podem ser valores positivos ou negativos
  - formato tem nº de bits fixo, variável:
    - **reais com precisão simples:** 32 bits
    - **reais com precisão dupla:** 64 bits
    - **reais com meia precisão :** 16 bits
  - **mantissa vs. expoente  $\Leftrightarrow$  precisão vs. intervalo valores representáveis**



# A norma IEEE 754-2008 para valores em fp (1)



- Representação do sinal na mantissa e no expoente
  - Mantissa:  $S + M$
  - Expoente: Excesso  $2^{n-1} - 1$
- Valor decimal de um fp em binário (normalizado)
  - Precisão simples:  $V_{Norm} = (-1)^S * (1.F) * 2^{E-127}$
  - Bit escondido
- Exceções (zero, subnormais, ...):  $E = 0$  ou  $E = 1111\dots1_2$
- Representação do zero:  $E = 0$  e  $F = 0$
- Representação e valor decimal de um fp (subnormal)
  - Representação:  $E = 0$  e  $F \neq 0$
  - Precisão simples:  $V_{SubN} = (-1)^S * (0.F) * 2^{-126}$
- Representação de  $\pm\infty$ :  $E = 1111\dots1_2$  e  $F = 0$
- Representação de n.º não real:  $E = 1111\dots1_2$  e  $F \neq 0 \rightarrow NaN$

## *A norma IEEE 754-2008 para valores em fp (2)*



### Format of Floating points IEEE754

64bit = double, double precision



32bit = float, single precision

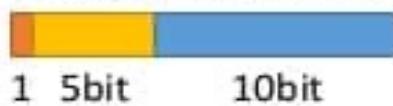


Signal

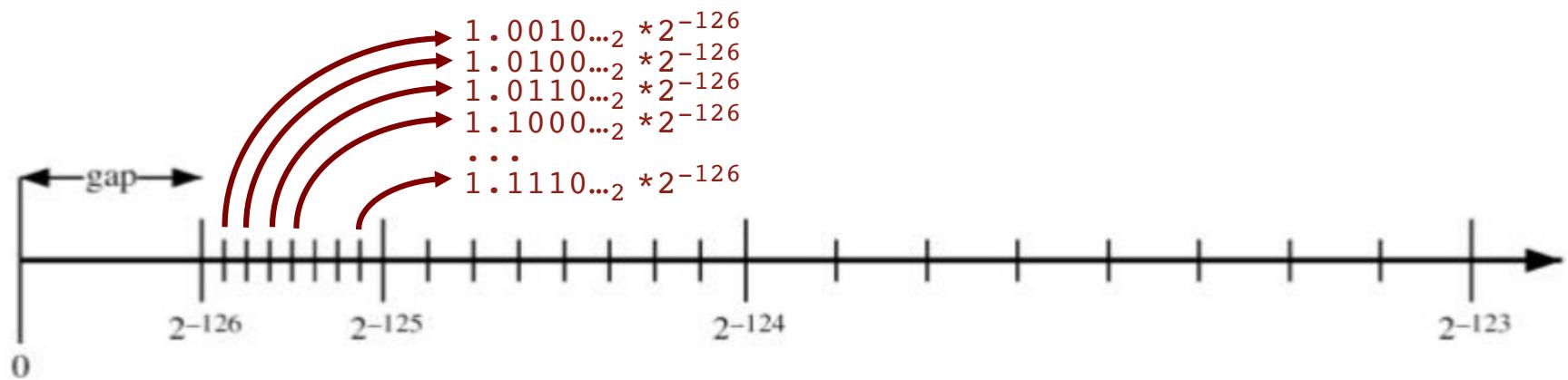
Expoent

Mantissa

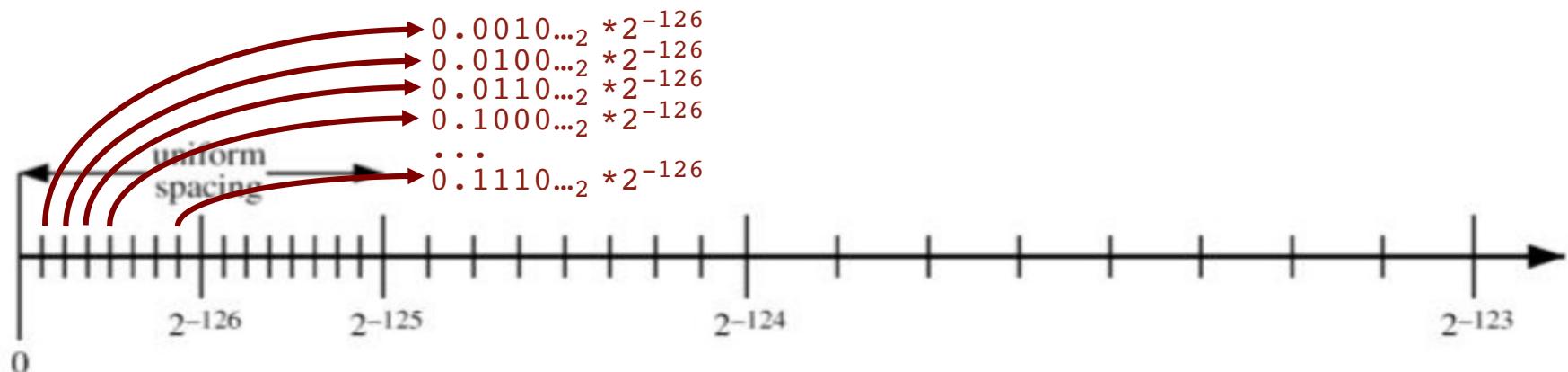
16bit = half, half precision



## O papel dos subnormais na norma IEEE 754



(a) 32-bit format without denormalized numbers



(b) 32-bit format with denormalized numbers



## Como se representa a informação?

- com *binary digits!*

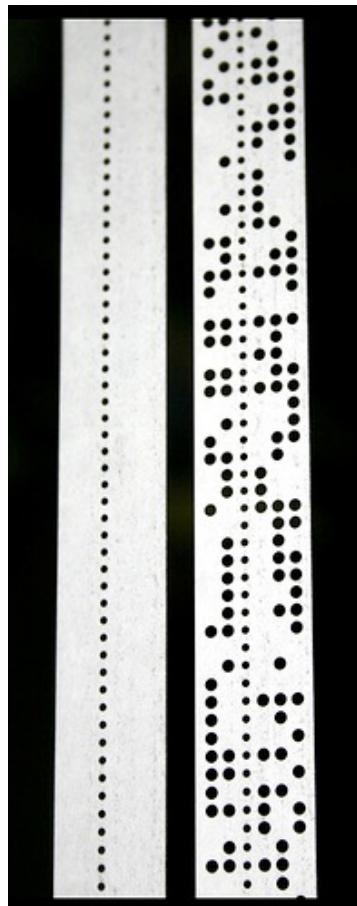
## Tipos de informação a representar:

- números (para cálculo)
  - » bases de numeração, inteiros (positivos e negativos)
  - » reais (*fp*), norma IEEE 754
- textos (caracteres alfanuméricos)
  - » Baudot, Braille, ASCII, Unicode, ...
- conteúdos multimédia
- código para execução no computador

## *Ex.: codificação telegráfica de texto, código de Baudot, 5-bits*



- Baudot,



| V | IV | I  | II | III | V | IV |    | I            | II | III |
|---|----|----|----|-----|---|----|----|--------------|----|-----|
| A | /  | ●  |    |     | ● | ●  | P. | %            | ●  | ●   |
| ● | B  | 8  |    |     | ● | ●  | Q  | /            | ●  | ●   |
| ● | C  | S  |    |     | ● | ●  | R  | -            | ●  | ●   |
| ● | D  | O  | ●  | ●   | ● | ●  | S  | ;            | ●  | ●   |
| E | 2  |    | ●  | ●   |   | T  | !  | ●            | ●  | ●   |
| E | &  |    | ●  | ●   |   | U  | 4  | ●            | ●  | ●   |
| ● | F  | E  | ●  | ●   |   | V  | '  | ●            | ●  | ●   |
| ● | G  | 7  |    |     | ● | ●  | W  | ?            | ●  | ●   |
| ● | H  | "  | ●  | ●   |   | X  | ,  | ●            | ●  | ●   |
| I | ?  |    | ●  | ●   |   | Y  | 3  | ●            | ●  | ●   |
| ● | J  | 6  | ●  |     |   | Z  | :  | ●            | ●  | ●   |
| ● | K  | (  |    |     |   | ≡  | .  | ●            | ●  | ●   |
| ● | L  | =  | ●  | ●   |   | ≡  | ≡  | Erasure      |    |     |
| ● | M  | )  |    |     |   |    |    | Figure Blank |    |     |
| ● | N  | Nº | ●  | ●   | ● |    |    | Letter Blank |    |     |
|   | O  | S  | ●  | ●   |   |    |    |              |    |     |

Fig 1. The Baudot code

| Letters | Figures      | V | IV | I | II | III | Letters      | Figures | V | IV | I | II | III |
|---------|--------------|---|----|---|----|-----|--------------|---------|---|----|---|----|-----|
| A       | 1            |   |    | ● |    |     | -            | .       | ● |    |   | ●  |     |
| E       | 2            |   |    |   | ●  |     | X            | 9       | ● |    |   | ●  |     |
| Y       | 3            |   |    |   |    | ●   | S            | 7       | ● |    |   | ●  |     |
| /       | Y            |   |    |   | ●  | ●   | Z            | :       | ● |    |   | ●  |     |
| I       | 3/           |   |    |   | ●  | ●   | W            | ?       | ● |    |   | ●  |     |
| U       | 4            |   |    |   | ●  | ●   | T            | 2       | ● |    |   | ●  |     |
| O       | 5            |   |    |   | ●  | ●   | V            | f       | ● |    |   | ●  |     |
|         |              |   |    |   |    |     | Letter Blank |         |   |    |   |    |     |
| J       | 6            |   |    | ● | ●  |     | K            | (       | ● | ●  | ● |    |     |
| G       | 7            |   |    | ● | ●  |     | M            | )       | ● | ●  | ● |    |     |
| B       | 8            |   |    |   |    |     |              |         |   |    |   |    |     |
| H       | 1            |   |    |   |    |     |              |         |   |    |   |    |     |
| F       | 5/           |   |    |   |    |     |              |         |   |    |   |    |     |
| C       | 9            |   |    |   |    |     |              |         |   |    |   |    |     |
| D       | O            |   |    |   |    |     |              |         |   |    |   |    |     |
|         | Figure Blank |   |    |   |    |     |              |         |   |    |   |    |     |



## *Ex.: codificação de texto em relevo, código Braille com 6-bits*



- Baudot, Braille,  
Alfabeto Braille

1      4  
2      5  
3      6

|    |    |    |                |    |     |     |     |     |                            |
|----|----|----|----------------|----|-----|-----|-----|-----|----------------------------|
| a  | b  | c  | d              | e  | f   | g   | h   | i   | j                          |
| .  | :  | .. | ..:            | .. | ..: | ..: | ..  | ..  | ..                         |
| k  | l  | m  | n              | o  | p   | q   | r   | s   | t                          |
| .  | :  | .. | ..:            | .. | ..: | ..: | ..  | ..  | ..                         |
| u  | v  | x  | y              | z  | ç   | é   | á   | è   | ú                          |
| .  | .. | .. | ..             | .. | ..  | ..  | ..  | ..  | ..                         |
| â  | ê  | í  | ô              | û  | à   | í   | ü   | ò   | w                          |
| .  | .. | .. | ..             | .. | ..  | ..  | ..  | ..  | ..                         |
| í  | ó  | ã  | sinal numérico |    | -   | ,   | —   | ... | grifo maiúscula caixa alta |
| .  | .. | .. | ..             | .. | ..  | ..  | ..  | ..  | ..                         |
| ,  | ;  | :  | .              | \$ | ?   | !   | ( ) | "   | *                          |
| .. | .. | .. | ..             | .. | ..  | ..  | ..  | ..  | ..                         |
| 1  | 2  | 3  | 4              | 5  | 6   | 7   | 8   | 9   | 0                          |
| .. | .. | .. | ..             | .. | ..  | ..  | ..  | ..  | ..                         |

## *Ex.: representação de texto a norma americana ASCII (7 bits)*



**ASCII: American Standard Code for Information Interchange**

**Tabela ASCII 7 bits**

|   | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9  | A   | B   | C  | D  | E  | F   |
|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|----|-----|-----|----|----|----|-----|
| 0 | NUL | SOH | STX | ETX | EOT | ENQ | ACK | BEL | BS  | HT | LF  | VT  | FF | CR | SO | SI  |
| 1 | DLE | DC1 | DC2 | DC3 | DC4 | NAK | SYN | ETB | CAN | EM | SUB | ESC | FS | GS | RS | US  |
| 2 | SP  | !   | "   | #   | \$  | %   | &   | '   | (   | )  | *   | +   | ,  | -  | .  | /   |
| 3 | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9  | :   | ;   | <  | =  | >  | ?   |
| 4 | @   | A   | B   | C   | D   | E   | F   | G   | H   | I  | J   | K   | L  | M  | N  | O   |
| 5 | P   | Q   | R   | S   | T   | U   | V   | W   | X   | Y  | Z   | [   | \  | ]  | ^  | _   |
| 6 | `   | a   | b   | c   | d   | e   | f   | g   | h   | i  | j   | k   | l  | m  | n  | o   |
| 7 | p   | q   | r   | s   | t   | u   | v   | w   | x   | y  | z   | {   |    | }  | ~  | DEL |

|    |    |    |    |    |    |    |    |    |    |    |    |  |  |  |  |  |
|----|----|----|----|----|----|----|----|----|----|----|----|--|--|--|--|--|
| H  | e  | l  | l  | o  |    | w  | o  | r  | l  | d  | !  |  |  |  |  |  |
| 48 | 65 | 6c | 6c | 6f | 20 | 77 | 6f | 72 | 6c | 64 | 21 |  |  |  |  |  |

## *Ex.: codificação universal de texto, UTF-8 no Unicode*



- Baudot, Braille, ASCII, Unicode, (UTF-8)

Unicode Transformation Format

| binary            | hex   | decimal | notes   |
|-------------------|-------|---------|---|
| 00000000-01111111 | 00-7F | 0-127   | US-ASCII (single byte)  |
| 10000000-10111111 | 80-BF | 128-191 | Second, third, or fourth byte of a multi-byte sequence                      |
| 11000000-11000001 | C0-C1 | 192-193 | Overlong encoding: start of a 2-byte sequence, but code point ≤ 127         |
| 11000010-11011111 | C2-DF | 194-223 | Start of 2-byte sequence  |
| 11100000-11101111 | E0-EF | 224-239 | Start of 3-byte sequence  |
| 11110000-11110100 | F0-F4 | 240-244 | Start of 4-byte sequence  |
| 11110101-11110111 | F5-F7 | 245-247 | Restricted by RFC 3629: start of 4-byte sequence for codepoint above 10FFFF |
| 11111000-11111011 | F8-FB | 248-251 | Restricted by RFC 3629: start of 5-byte sequence                            |
| 11111100-11111101 | FC-FD | 252-253 | Restricted by RFC 3629: start of 6-byte sequence                            |
| 11111110-11111111 | FE-FF | 254-255 | Invalid: not defined by original UTF-8 specification                        |



## Como se representa a informação?

- com *binary digits!*

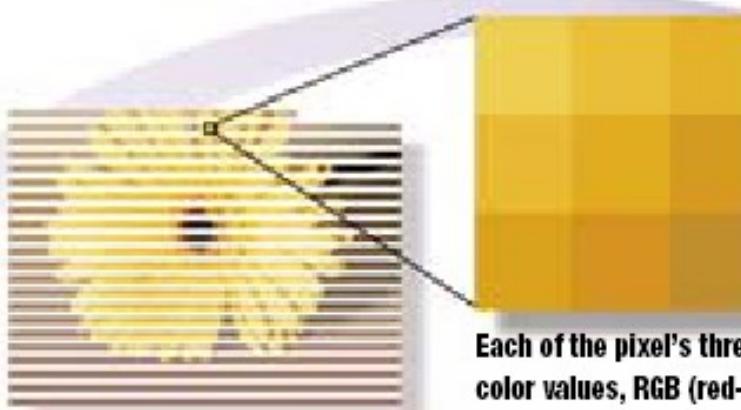
## Tipos de informação a representar:

- números (para cálculo)
  - » inteiros: S+M, Compl. p/ 1, Compl. p/ 2, Excesso
  - » reais (*fp*): norma IEEE 754
- textos (caracteres alfanuméricos)
  - » Baudot, Braille, ASCII, Unicode, ...
- conteúdos multimédia
  - » imagens fixas: BMP, JPEG, GIF, PNG, ...
  - » audio-visuais: AVI, MPEG/MP3, ...
- código para execução no computador

## *Ex.: representação de uma imagem em bitmap*



You can create a 24-bit image in a graphics program such as Paint.



A graphics program saves the image line by line, from the bottom to the top.

Each of the pixel's three-color values, RGB (red-green-blue), are read from left to right.

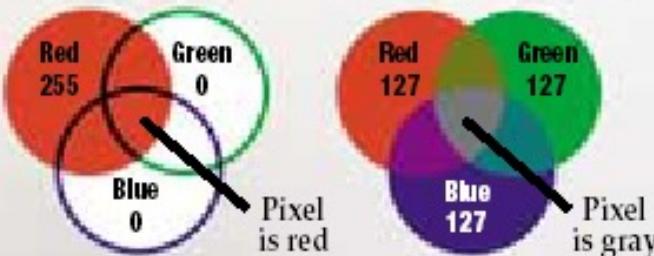
|       |       |       |
|-------|-------|-------|
| R 250 | R 244 | R 238 |
| G 210 | G 196 | G 182 |
| B 94  | B 69  | B 51  |
| R 242 | R 235 | R 222 |
| G 190 | G 176 | G 160 |
| B 60  | B 42  | B 26  |
| R 226 | R 218 | R 201 |
| G 187 | G 153 | G 148 |
| B 27  | B 17  | B 53  |

A graphics program translates the RGB values into palette values. The palette values are a software-specific decision; each program's values are different.

Each palette value, a hexadecimal value in this case, is stored in the same order as displayed in the image.

|        |        |        |
|--------|--------|--------|
| FAD25E | F4C345 | E1EB63 |
| F2BE3C | EBB02A | DEA01A |
| E4A71B | DA9911 | C99435 |

The pixel values are stored in the bit-mapped file in the same width and depth as the original image.



# *Representação da informação num computador (4)*



## Como se representa a informação?

- com *binary digits!*

## Tipos de informação a representar:

- números (para cálculo)
  - » inteiros: S+M, Compl. p/ 1, Compl. p/ 2, Excesso
  - » reais (*fp*): norma IEEE 754
- textos (caracteres alfanuméricos)
  - » Baudot, Braille, ASCII, Unicode, ...
- conteúdos multimédia
  - » imagens fixas: BMP, JPEG, GIF, PNG, ...
  - » audio-visuais: AVI, MPEG/MP3, ...
- código para execução no computador
  - » noção de *instruction set*

## *Ex.: representação de código para execução num PC*



```
int x = x+y;
```

```
addl 8(%ebp),%eax
```

**Idêntico à expressão**  
 $x = x + y$

```
0x401046: 03 45 08
```

- Código numa linguagem de programação
  - somar 2 inteiros
- Código numa linguagem mais próxima do processador
  - somar 2 inteiros (de 4-bytes)
  - operandos:
    - x: no registo eax
    - y: na memória em [ (ebp) +8 ]
- Código “objecto” (em hexadecimal)
  - instrução com 3-bytes
  - na memória em 0x401046