Licenciatura Ciências de Computação
Licenciatura Engª. 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

Noção de computador (1)

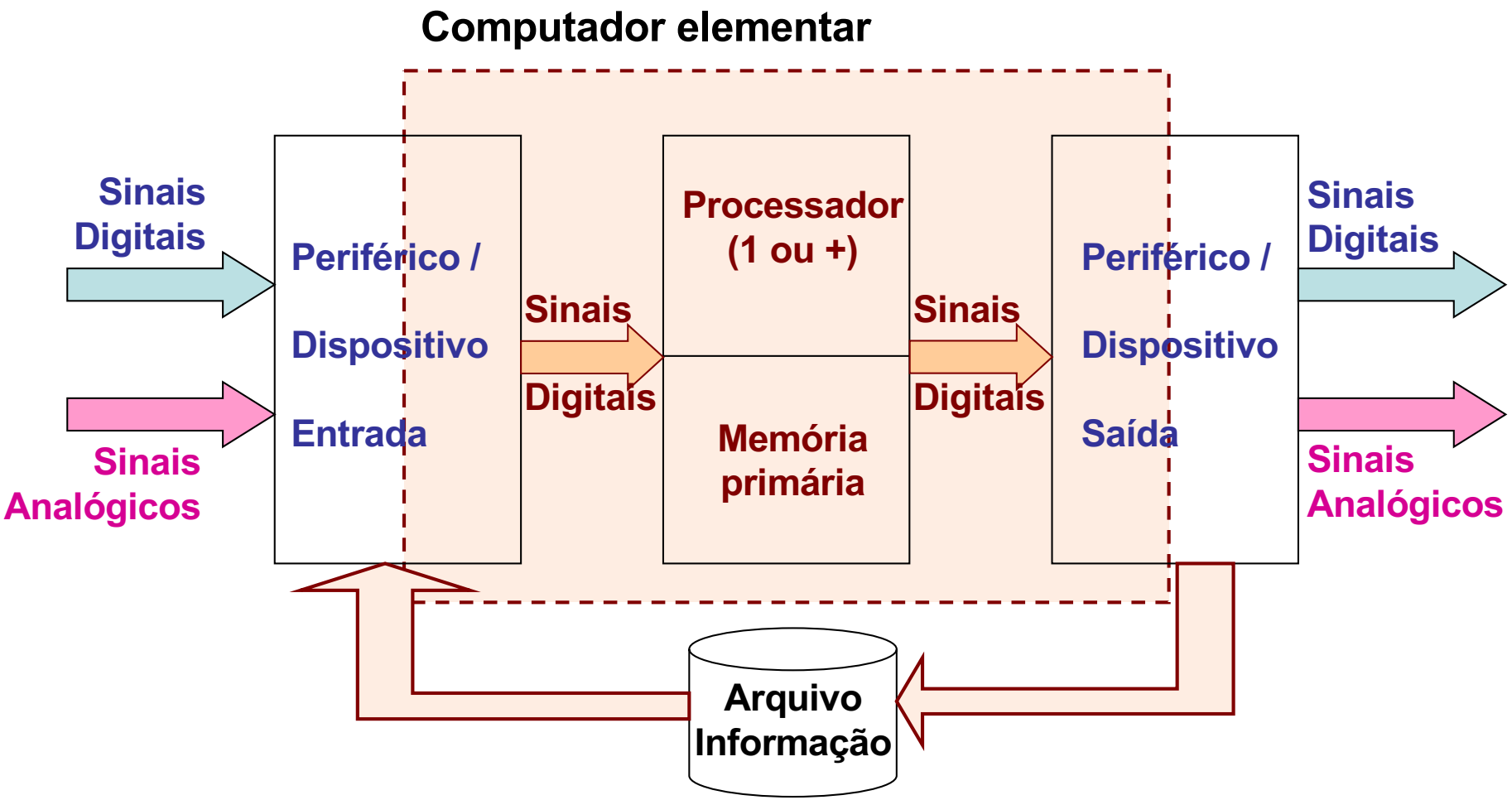
Um computador é um sistema físico que:

- 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)



Noção de computador (3)

- **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 ***b****inary dig****its***!

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 ***b****inary dig****its***!

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$_{10}$ (base 10) ; quanto vale cada algarismo?

1*10^3 + **5***10^2 + **3***10^1 + **2***10^0 + **5***10^{-1} + **4***10^{-2} = 1**5**32.**5**4$_{10}$

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

1532$_6$ (base 6) ; quanto vale cada algarismo na base 10?

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

1532$_{13}$ (base 13) ; quanto vale cada algarismo na base 10?

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

110110.011$_2$ (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} =

54.375$_{10}$

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...

416_{10} ; quanto vale cada algarismo na base 6?

- parte inteira ... parte decimal ...

3083_{10} ; quanto vale cada algarismo na base 13?

- parte inteira ... parte decimal ...

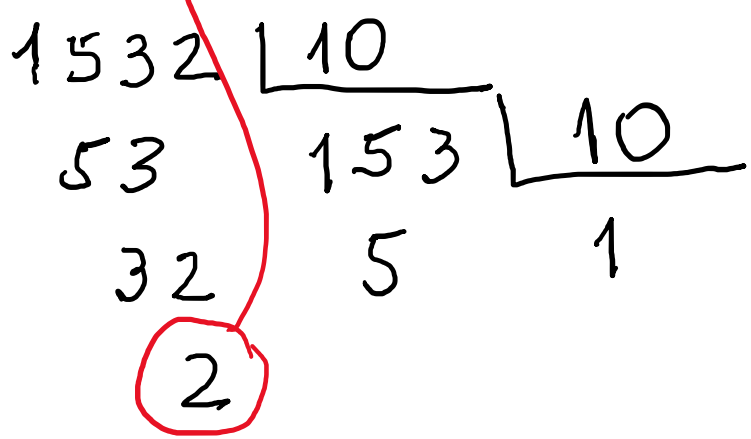
54.375_{10}; quanto vale cada algarismo na base 2?

- parte inteira ... parte decimal ...

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

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

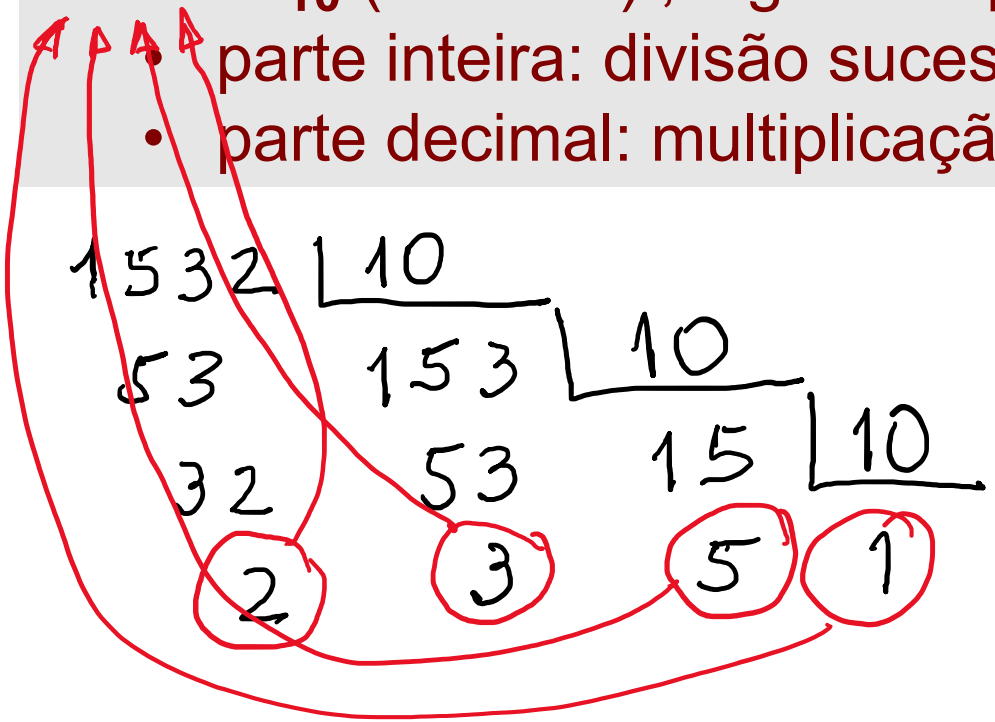
- parte inteira: divisão sucessiva pela base e...
- parte decimal: multiplicação sucessiva pela base e...



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?

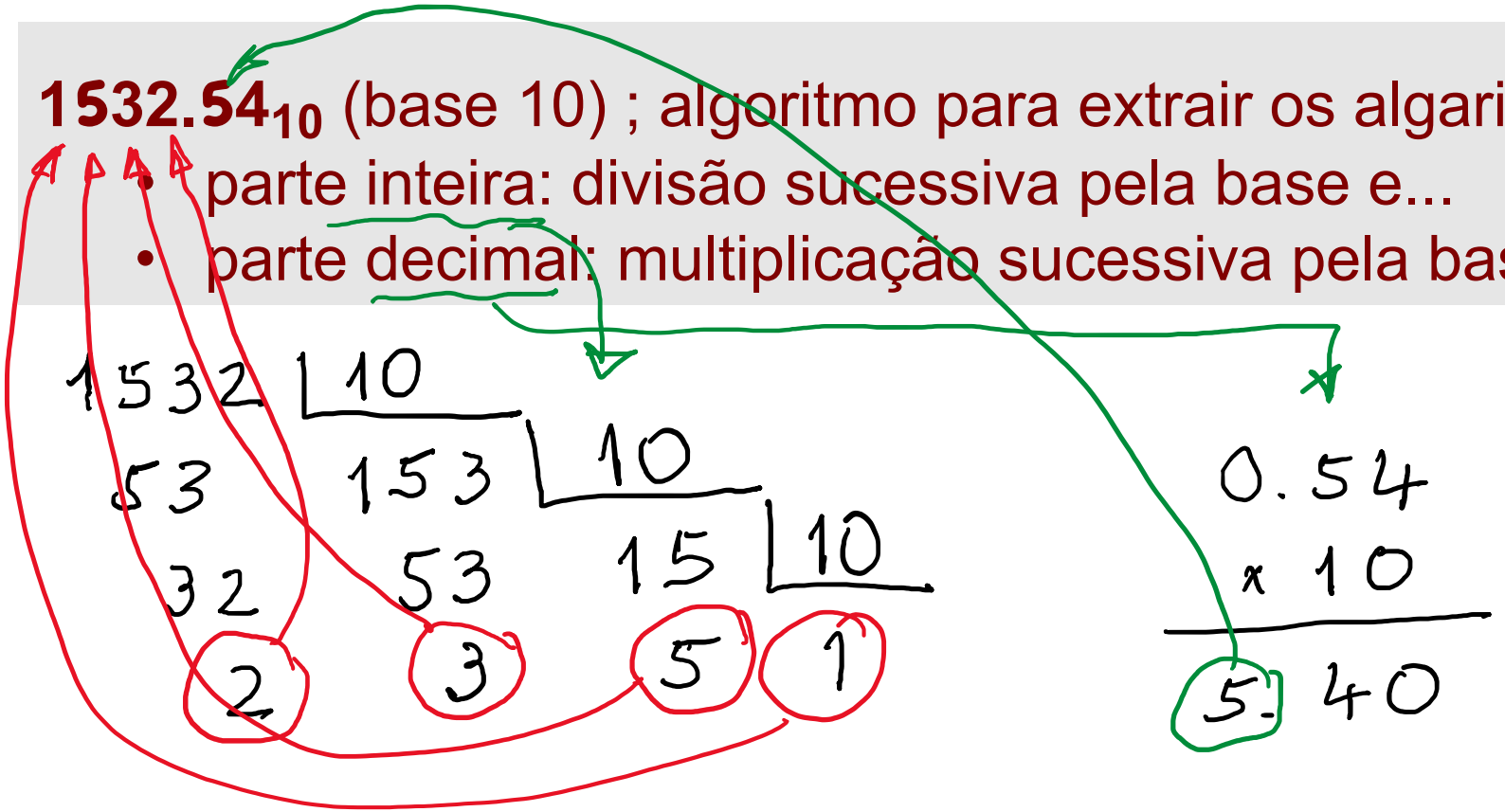
- parte inteira: divisão sucessiva pela base e...
- parte decimal: multiplicação sucessiva pela base e...



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?

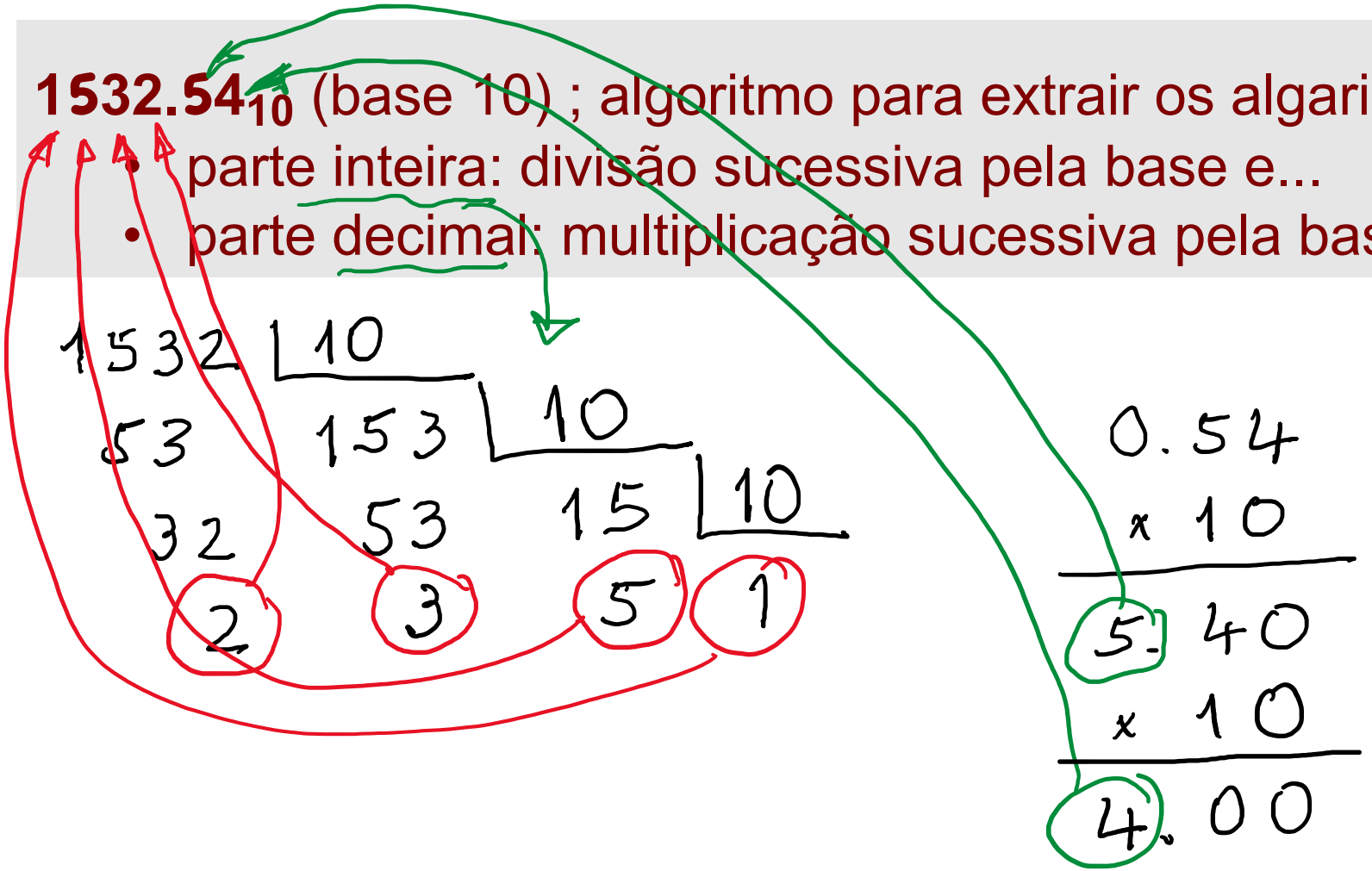
- parte inteira: divisão sucessiva pela base e...
- parte decimal: multiplicação sucessiva pela base e...



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...



Sistemas de numeração:

como se passa um nº na base 10 para uma outra base

1532_6 (base 6) ; quanto vale cada algarismo na base 10?

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

416_{10} ; quanto vale cada algarismo na base 6?

- parte inteira ... parte decimal ...

4 1 6 | 6

Sistemas de numeração:

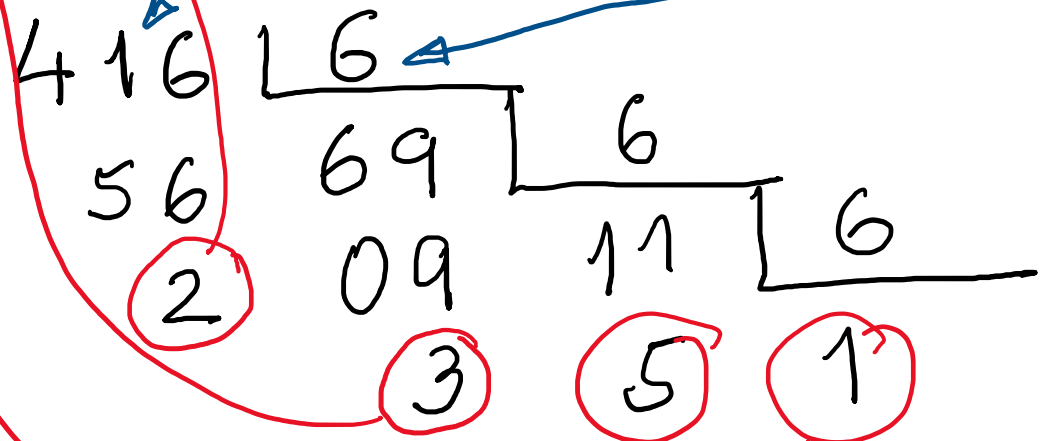
como se passa um nº na base 10 para uma outra base

1532_6 (base 6) ; quanto vale cada algarismo na base 10?

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

416_{10} ; quanto vale cada algarismo na base 6?

- parte inteira ... parte decimal ...



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

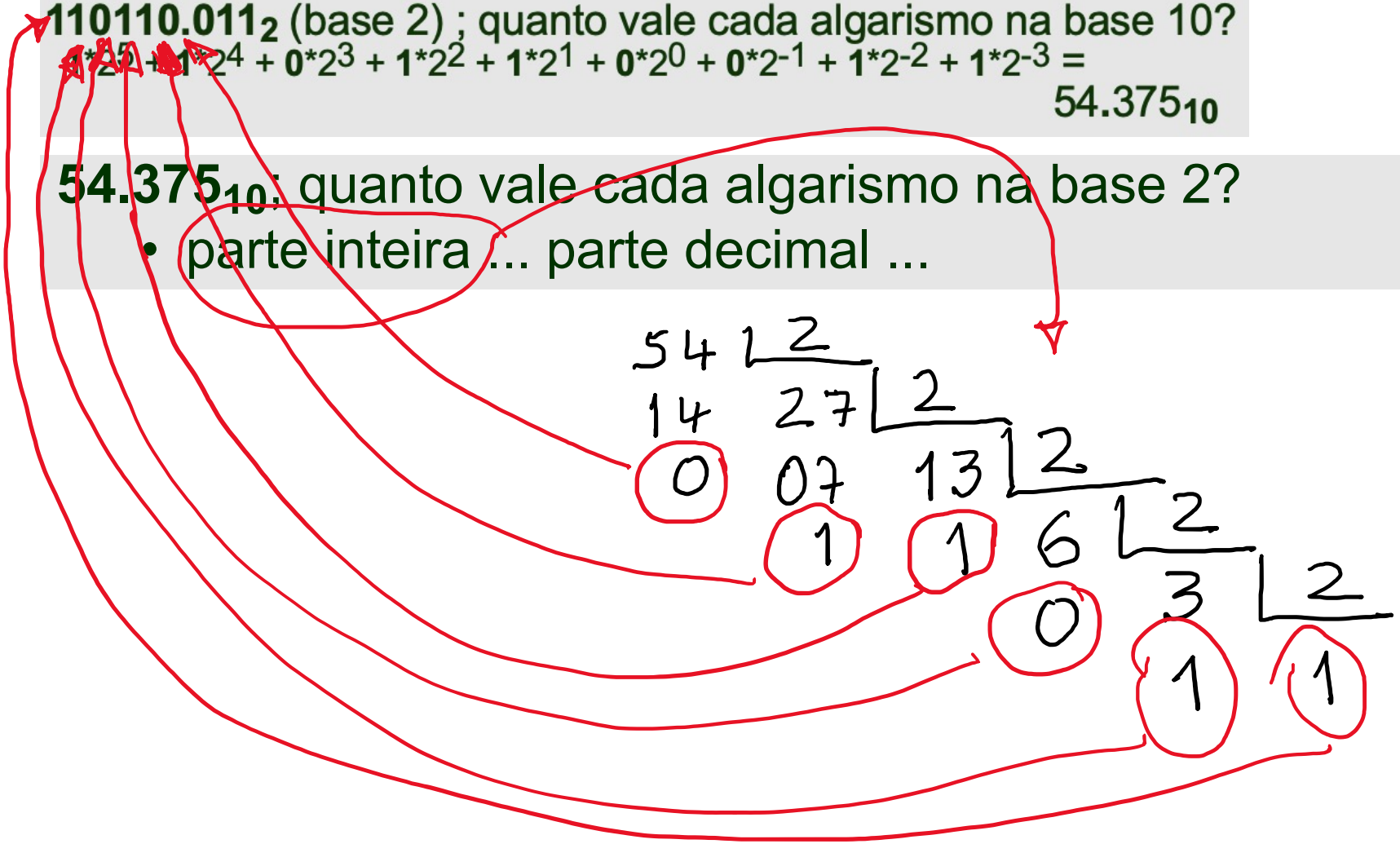
110110.011_2 (base 2) ; quanto vale cada algarismo na base 10?

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

54.375_{10}

54.375_{10}: 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

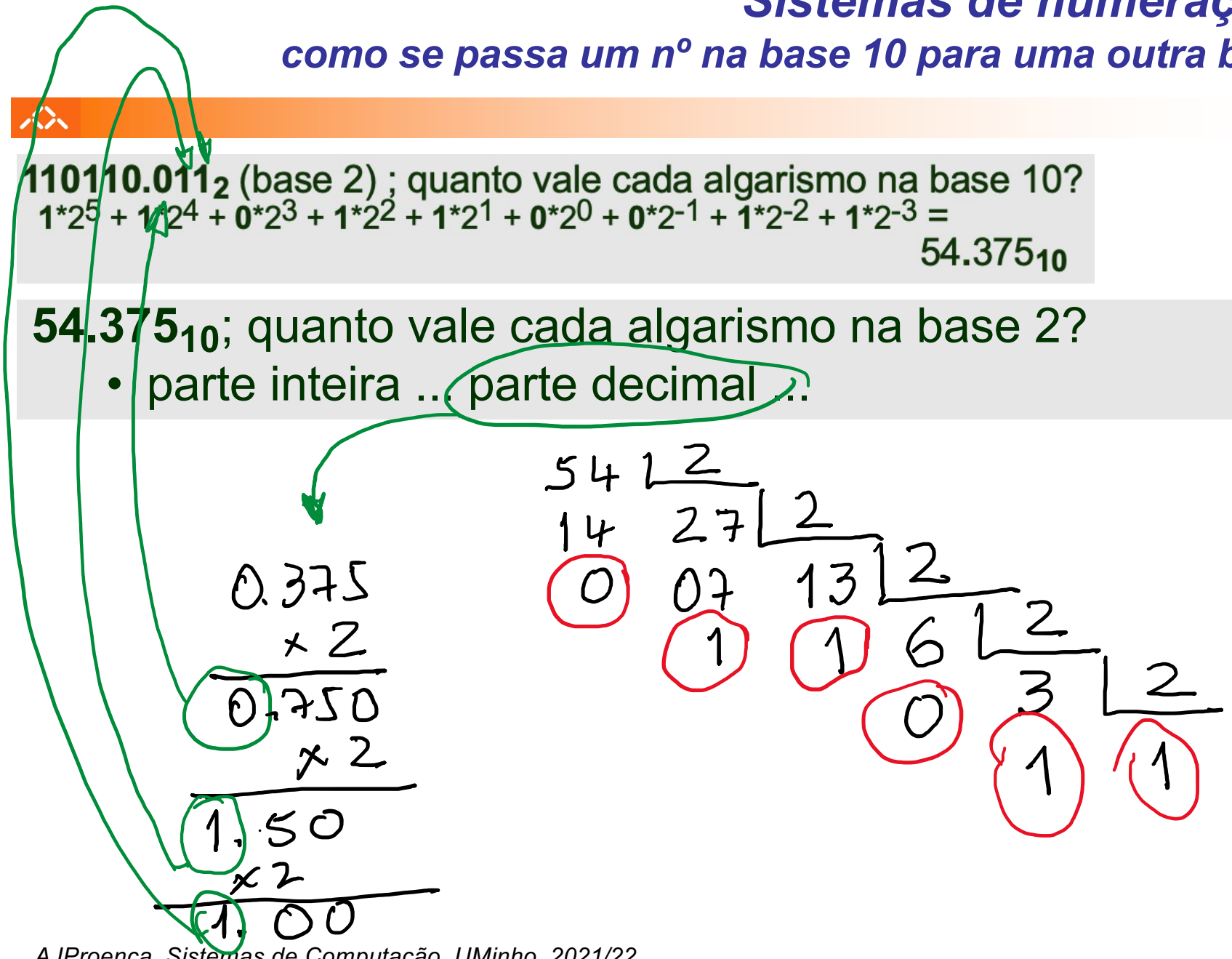
110110.011_2 (base 2) ; quanto vale cada algarismo na base 10?

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

54.375_{10}

54.375_{10}; quanto vale cada algarismo na base 2?

- parte inteira ... parte decimal ...



Sistemas de numeração:
caso particular da base 2

110110.011_2 (base 2) ; quanto vale cada algarismo na base 10?

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

Para simplificar:

- eliminar os produtos, ignorar parcelas com produtos por 0
- $1*2^5 + 1*2^4 + 0*2^3 + 1*2^2 + 1*2^1 + 0*2^0 + 0*2^{-1} + 1*2^{-2} + 1*2^{-3}$ = ...

  => $2^5 + 2^4 + 2^2 + 2^1 + 1/2^2 + 1/2^3$ = ...

Recomendações:

- decorar a tabuada das potências de 2 ($2^0 + 2^{10}$) ⟶
- 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

$2^{10} = 1024 = 1\ Ki(bi) \approx 1000 = 10^3 = 1\ K(ilo)$

...

$2^{12} = 2^2 * 2^{10} = 4\ Ki(bi) \approx 4000 = 4*10^3 = 4\ K$

...

$2^{16} = 2^6 * 2^{10} = 64\ Ki(bi) \approx 64*10^3 = 64\ K$

$2^{20} = 1\ Me(bi) \approx 1000000 = 10^6 = 1\ M(ega)$

$2^{30} = 1\ Gi(bi) \approx 1000000000 = 10^9 = 1\ G(iga)$

$2^{40} = 1\ Te(bi) \approx 10^{12} = 1\ T(era)$

$2^{50} = 1\ Pe(bi) \approx 10^{15} = 1\ P(eta)$

Vantagens destas dicas

- Na conversão de binário para base 10:

110110.011_2 (base 2)
=> $2^5 + 2^4 + 2^2 + 2^1 + 1/2^2 + 1/2^3 =$
= 32 + 16 + 4 + 2 + 1/4 + 1/8 = 54 + 0.375

- Na conversão de base 10 para binário:

290 = 256 +

256 − ~~512~~ 2^8

Vantagens destas dicas

- Na conversão de binário para base 10:

110110.011_2 (base 2)
=> $2^5 + 2^4 + 2^2 + 2^1 + 1/2^2 + 1/2^3 =$
= $32 + 16 + 4 + 2 + 1/4 + 1/8 = 54 + 0.375$

- Na conversão de base 10 para binário:

$290 = 256 + 32 + 2$
$2^8 \quad 2^5 \quad 2^1$

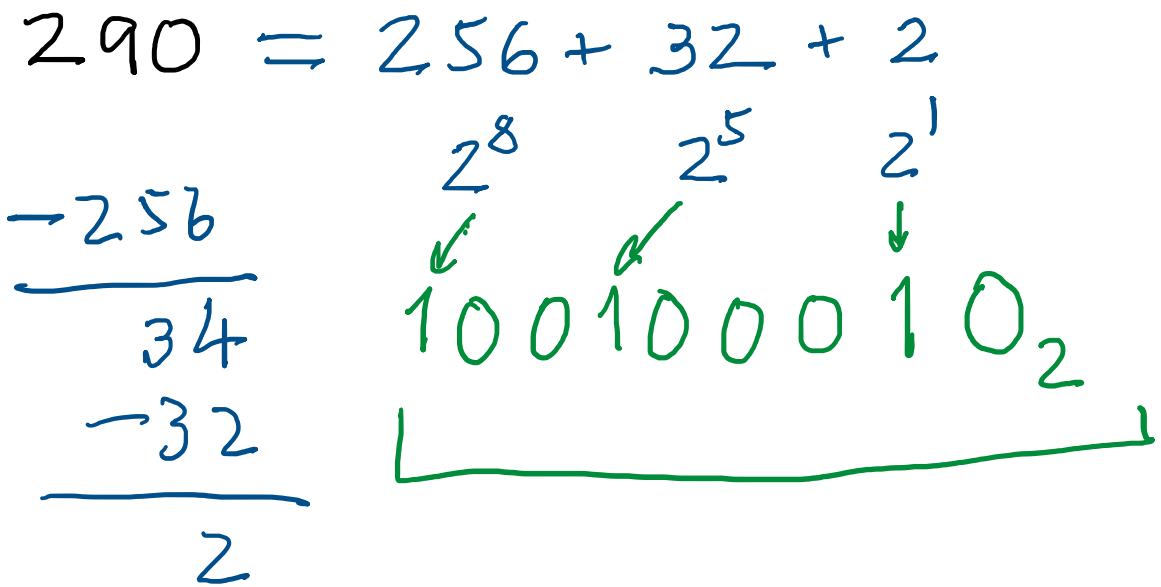
-256
34
-32
2

Vantagens destas dicas

- Na conversão de binário para base 10:

110110.011_2 (base 2)
=> $2^5 + 2^4 + 2^2 + 2^1 + 1/2^2 + 1/2^3 =$
= $32 + 16 + 4 + 2 + 1/4 + 1/8 = 54 + 0.375$

- Na conversão de base 10 para binário:



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

- Dígitos na base 16:

  0, 1, 2, ... 9, *10* a, *11* b, *12* c, *13* d, *14* e, *15* f

- Vantagens sobre um valor de 32 bits:

  $10100110100001110110010111010100_2$ ***vs.*** $a68765d4_{16}$

- Facilidade de conversão:

  1010 0110 1000 0111 0110 0101 1101 0100_2

  a 6 8 7 6 5 d 4_{16}

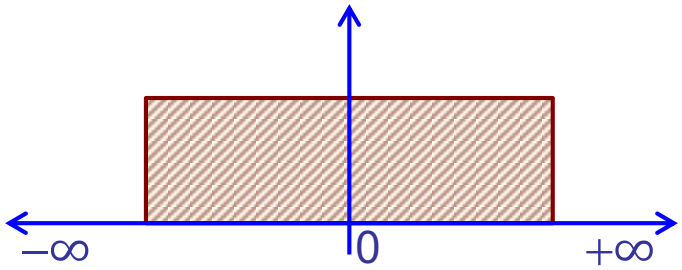
- Mesmo com ponto decimal:

  $10100110100001110110010111101.01_2$

  1010 0110 1000 0111 0110 0101 1101.0100_2

  a 6 8 7 6 5 d . 4_{16}

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_2 › -3		
5	101				
6	110				
7	111				

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 › -3		
5	101		-10_2 › -2		
6	110		-01_2 › -1		
7	111		-00_2 › -0		

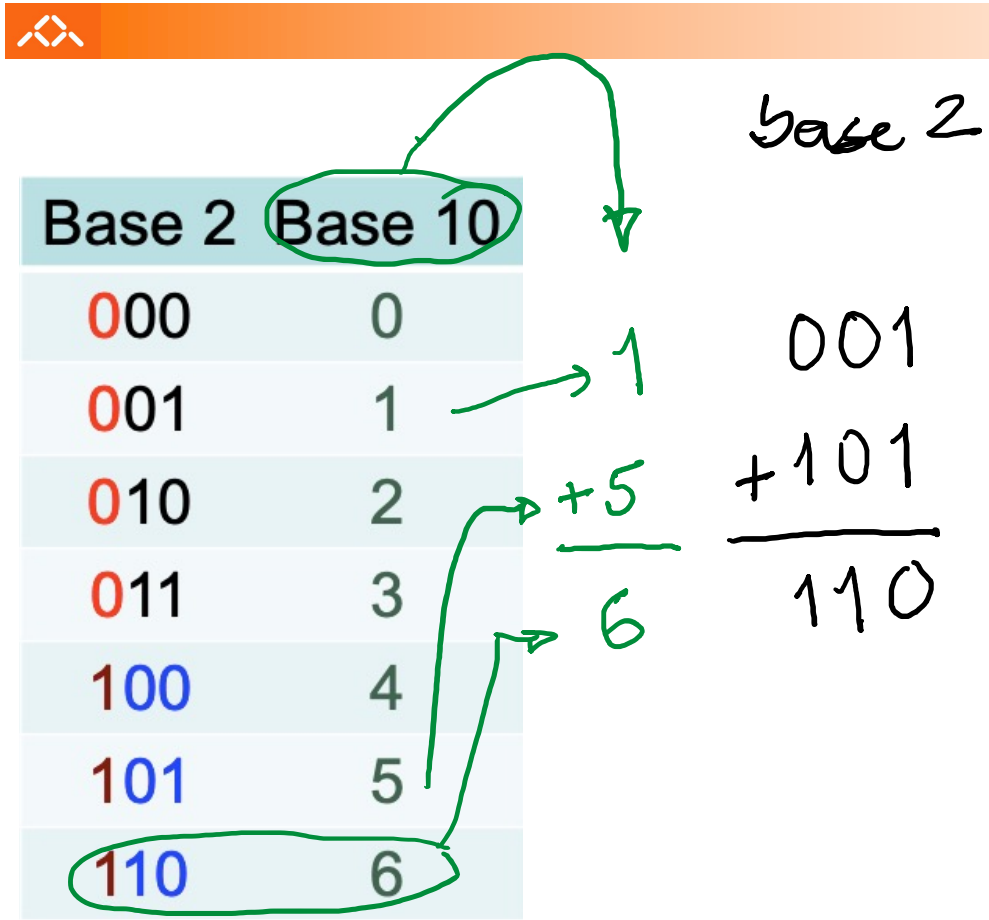
Inteiros positivos e negativos:
o universo com 3 bits (6)

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 › -3	$-(11+1)_2$ › -4	
5	101		-10_2 › -2		
6	110		-01_2 › -1		
7	111		-00_2 › -0		

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

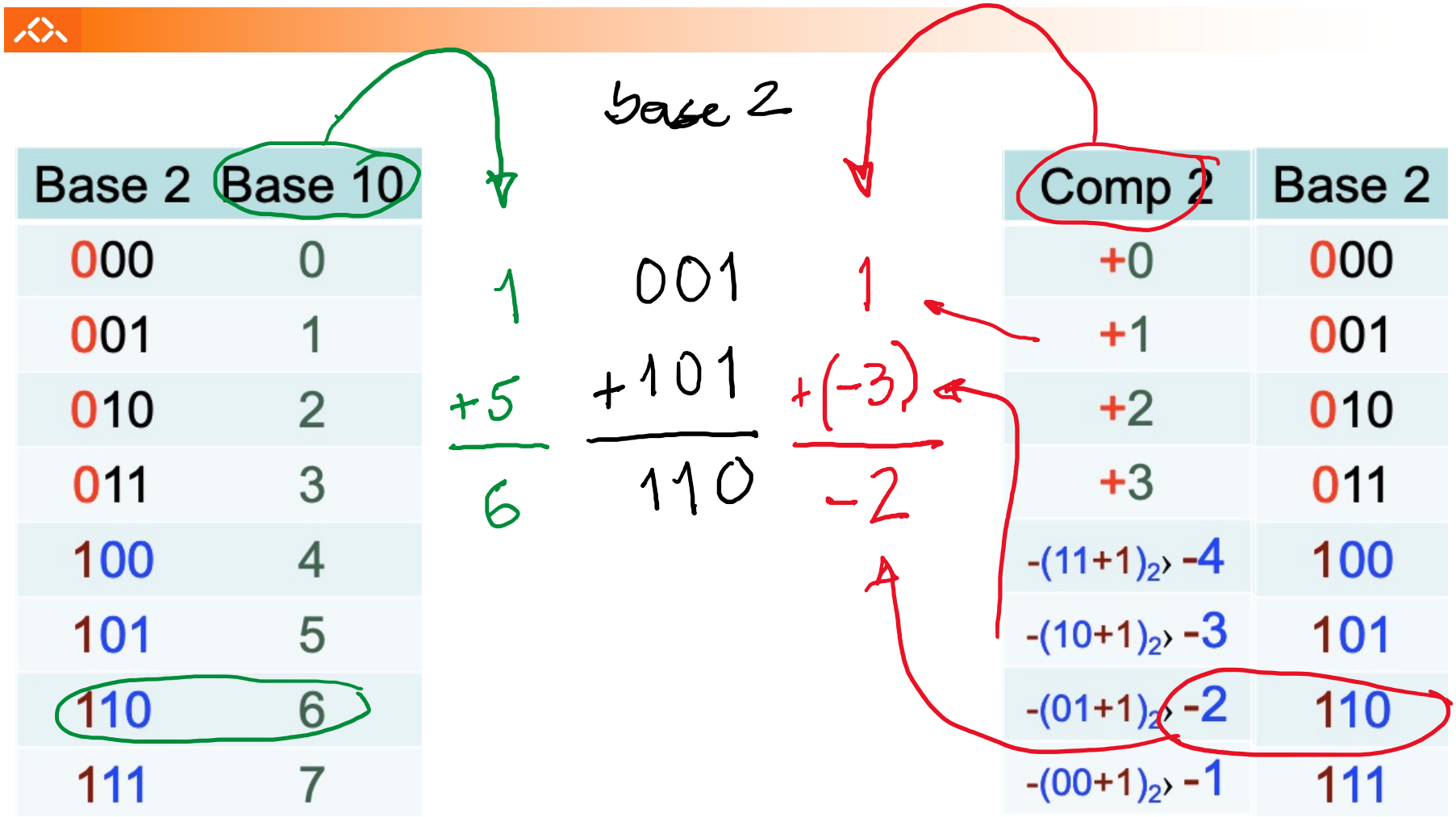
Base 10	Base 2	S+M	Comp 1	Comp 2	Excesso
0	000		+0	+0	
1	001		+1	+1	
2	010		+2	+2	
3	011		+3	+3	
4	100		-11_2› -3	$-(11+1)_2$› -4	
5	101		-10_2› -2	$-(10+1)_2$› -3	
6	110		-01_2› -1	$-(01+1)_2$› -2	
7	111		-00_2› -0	$-(00+1)_2$› -1	

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



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

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



Base 2	Base 10
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Base 2

1 001 +5 +101 6 110 +(-3) -2

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

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

Base 2	Base 10
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Base 2

1 001 1

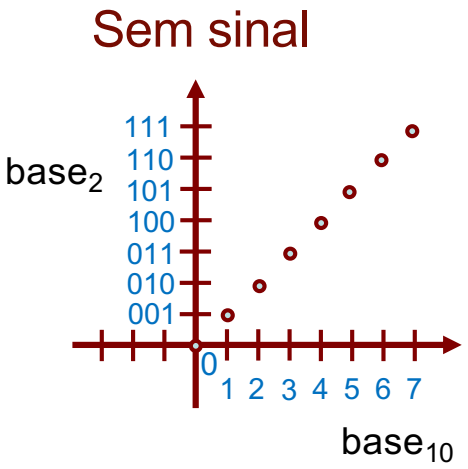
+5 +101 +(-3)

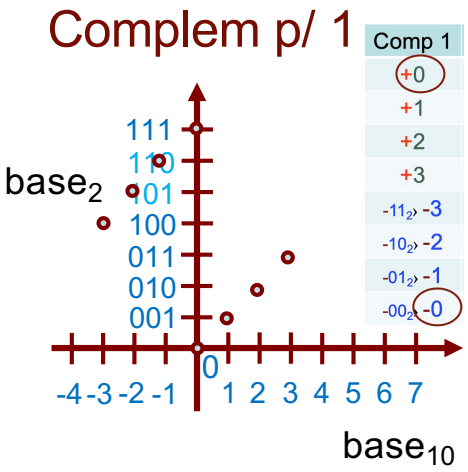
6 110 -2

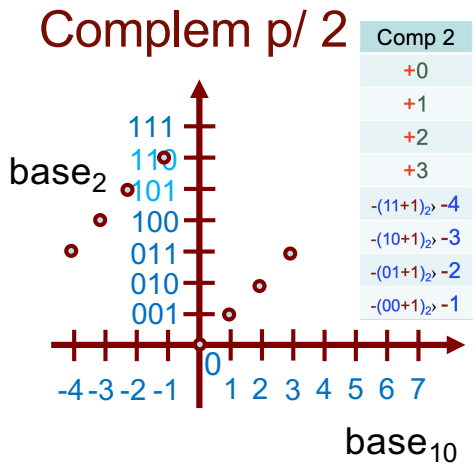
mesmo somador em binário!

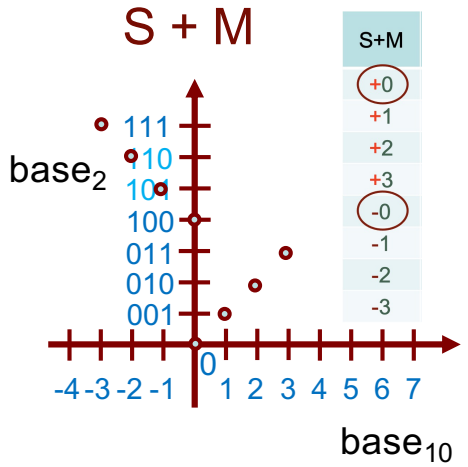
Comp 2	Base 2
+0	000
+1	001
+2	010
+3	011
$-(11+1)_2$ -4	100
$-(10+1)_2$ -3	101
$-(01+1)_2$ -2	110
$-(00+1)_2$ -1	111

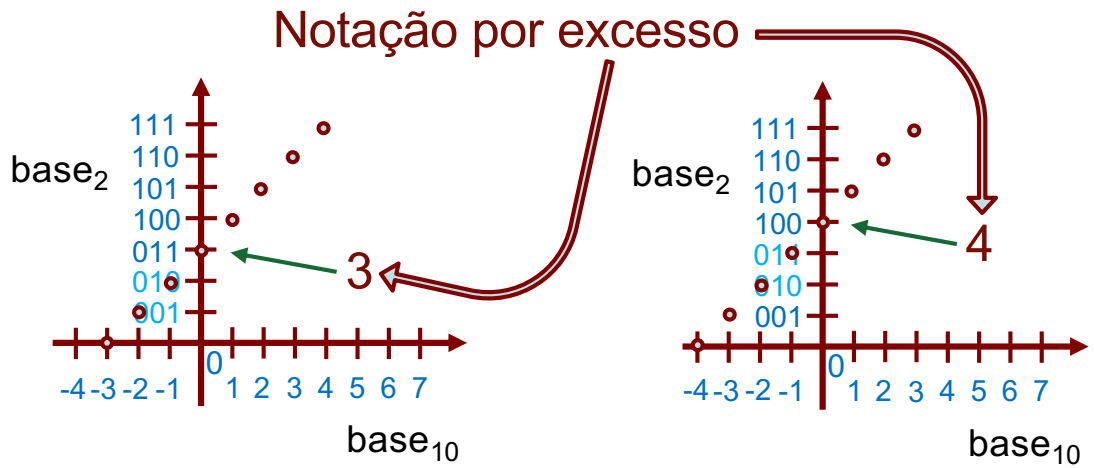
Inteiros positivos e negativos:
porquê mais uma maneira de codificação?











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

Base 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= #bits, $2^{n-1}-1= 2^{3-1}-1= 2^{2}-1= 3$

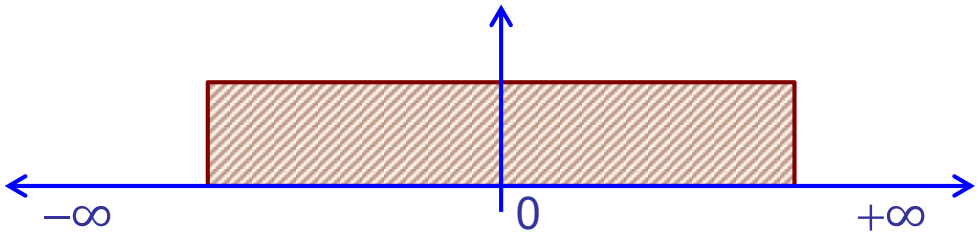
Inteiros positivos e negativos:
o universo com 3 bits (9)

Base 10	Base 2	S+M	Comp 1	Comp 2	Excesso 2^{n-1}	Excesso 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

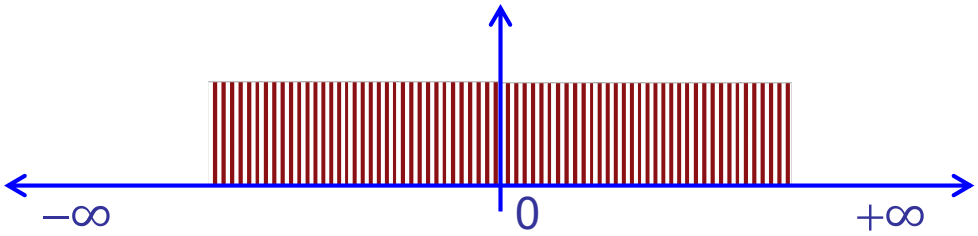
Nota: n= #bits, $2^{n-1}= 2^{3-1}= 2^2=$ **4**, $2^{n-1}-1= 2^{3-1}-1= 2^2-1=$ **3**

Representação de reais em vírgula flutuante (1)

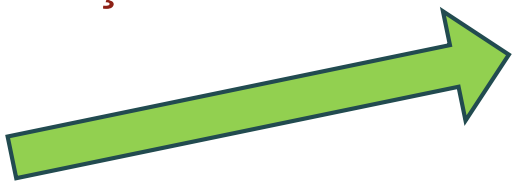
- **Gama de valores**
  - **esta gama é viável?**
  - **não, porque...**
    entre quaisquer 2 nºs reais há um conjunto infinito de valores

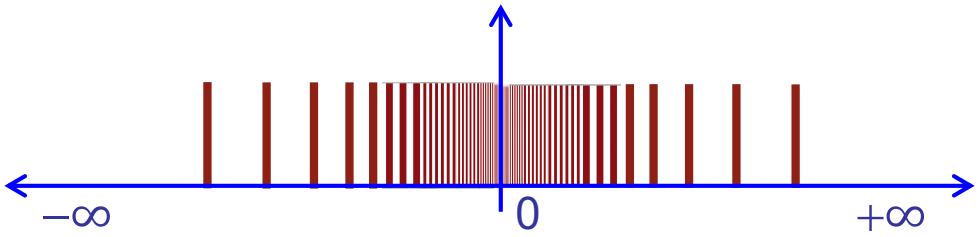


  - **então faz sentido a seguinte representação?**
    Notação vírgula fixa



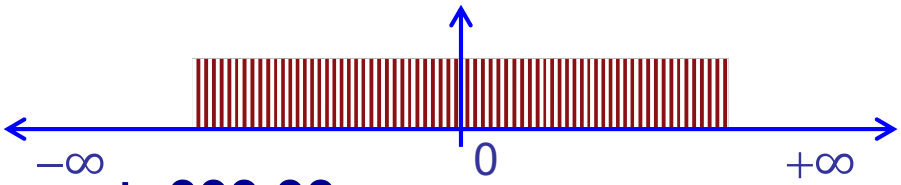
  - **ou faria mais sentido a seguinte representação?**
    Notação científica





Representação de reais em vírgula flutuante (2)

- **Notação em vírgula fixa *vs.* notação científica**
  - **Exemplos na base 10 usando apenas 5 dígitos**
  - **Vírgula fixa:** **± X X X . X X**



maior valor representável : **± 999.99**

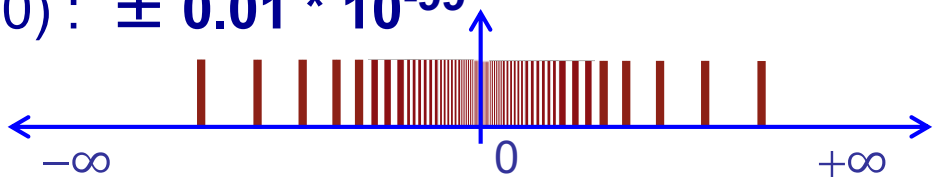
menor valor representável (≠0) : **± 0.01**

mantissa **expoente**

  - **Notação científica ou vírgula flutuante:** **± X . X X * 10^± X X**

maior valor representável : **± 9.99 * 10^+99**

menor valor representável (≠0) : **± 0.01 * 10^-99**



Representação de reais em vírgula flutuante (3)

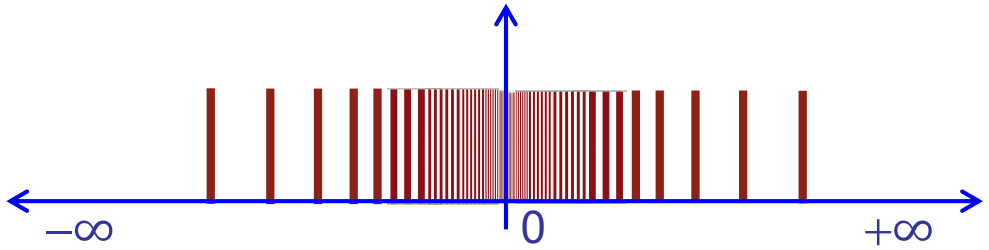
- **Normalização na representação**
  - **Para garantir que cada nº a representar usa apenas uma única sequência de dígitos, há que fixar o local para o ponto decimal**
  - **Nº normalizado: à esquerda do ponto decimal apenas 1 dígito ≠0**

    $\pm$ Y . X X X ... X * Radix^{Exp} 0 < Y ≤ maior_dígito
  - **Menor valor normalizado representável:**

    $\pm$ 1.000 ... 0 * $10^{-9...9}$
  - **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 ... 0 * $10^{-9...9}$ **a** $\pm$ 0.999 ... 9 * $10^{-9...9}$

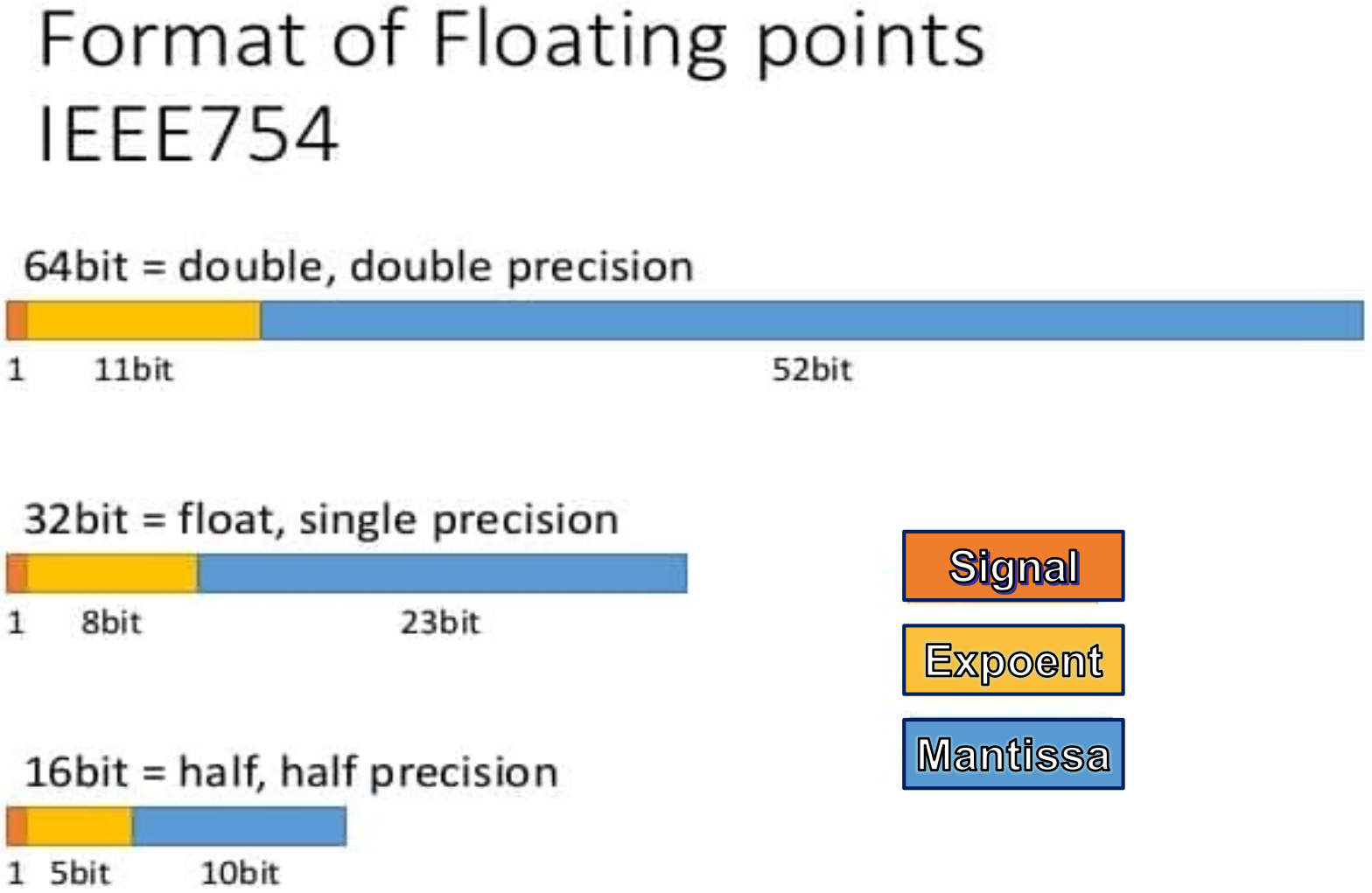
- **Formato binário dum valor em fp**
  - **formato inclui mantissa e expoente; radix é implícito (=2)**
  - **mantissa e expoente podem ser valores positivos ou negativos**
  - **formato tem nº de bits fixo, variável:**
    - **reais com precisão simples: 32 bits**
    - **reais com precisão dupla: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



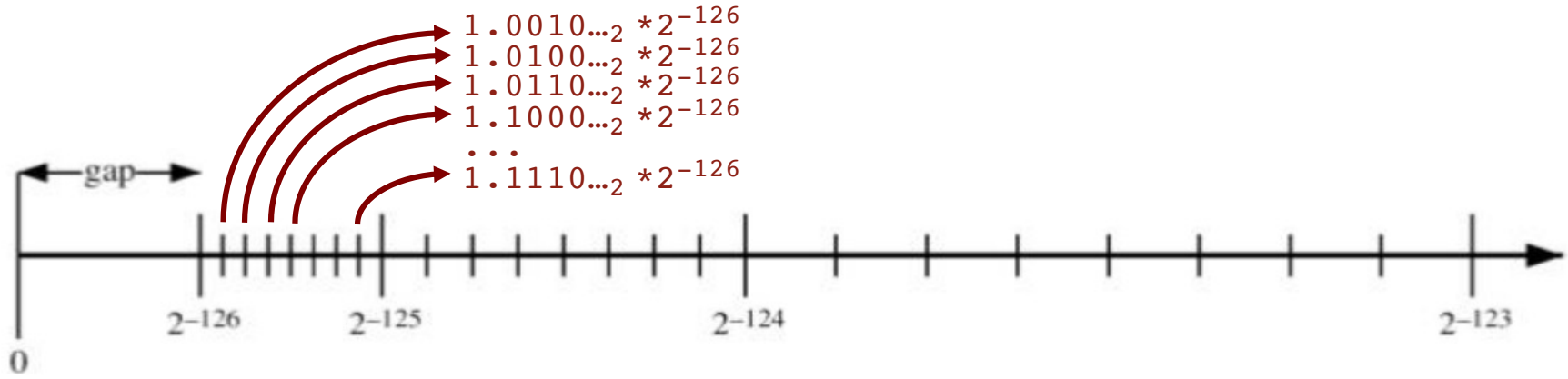
A norma IEEE 754-2008 para valores em fp (1)

- **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**
  - **$1 \leq E \leq 111...10_2$**
- **Exceções** (zero, subnormais, ...): **$E = 0$** ou **$E = 1111...1_2$**
- **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 $\pm\infty$:** **$E = 1111...1_2$** e **$F = 0$**
- **Representação de n.º não real:** **$E = 1111...1_2$** e **$F \neq 0$** $\rightarrow$ **NaN**

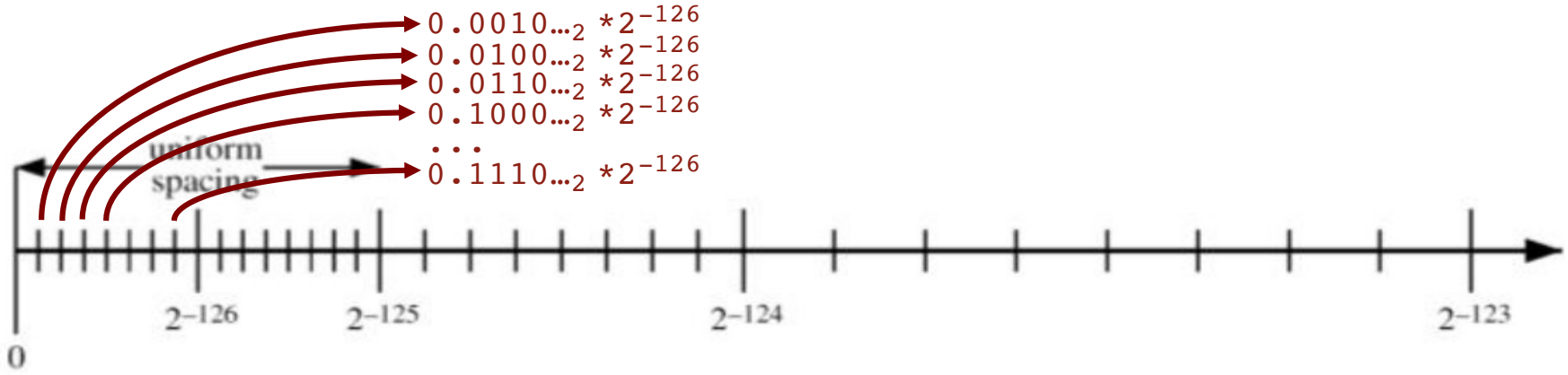
A norma IEEE 754-2008 para valores em fp (2)



O papel dos subnormais na norma IEEE 754



(a) 32-bit format without denormalized numbers



(b) 32-bit format with denormalized numbers

Representação da informação num computador (2)

Como se representa a informação?

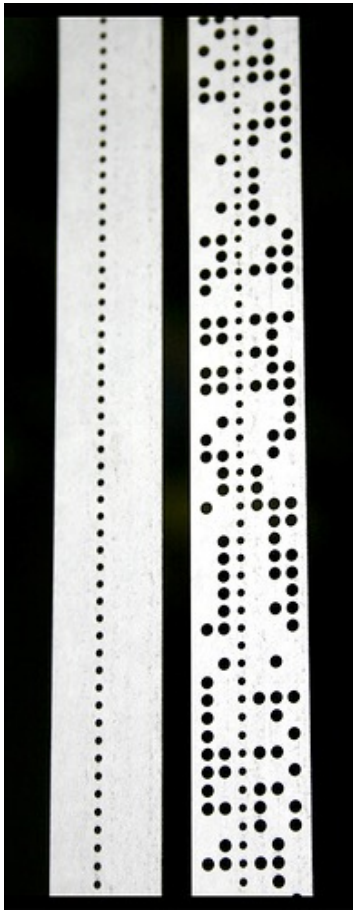
- com ***b****inary dig****its***!

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,



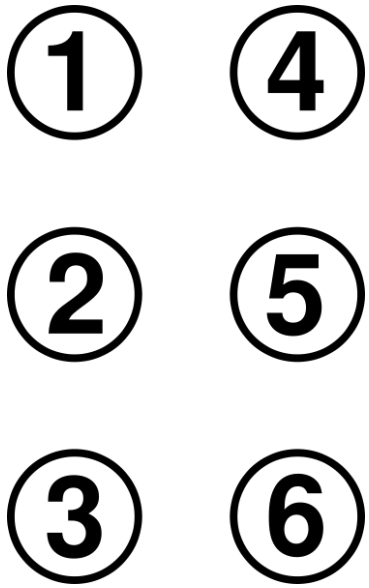
V	IV		I	II	III	V	IV		I	II	III
		A	-	●			●	●	P.	%	●	●	●
	●	B	8			●	●	●	Q	/	●		●
	●	C	9	●		●	●	●	R	-			●
	●	D	0	●	●	●	●		S	;			●
		E	2		●		●		T	!	●		●
		F	&	●	●				U	4	●		●
	●	F	Ⅎ		●	●	●		V	'	●	●	●
	●	G	7		●		●		W	?		●	●
	●	H	ⁿ⁄₌	●	●		●		X	,		●	
		I	º⁄₌		●	●			Y	3			●
	●	J	6	●			●		Z	:	●	●	
●	●	K	(	●			●		⁼⁄₌	.	●		
●	●	L	=	●	●		●	●	Erasure				
●	●	M	)		●			●	Figure Blank				
●	●	N	№		●	●	●		Letter Blank				
		O	5	●	●	●							

Fig 1. The Baudot code

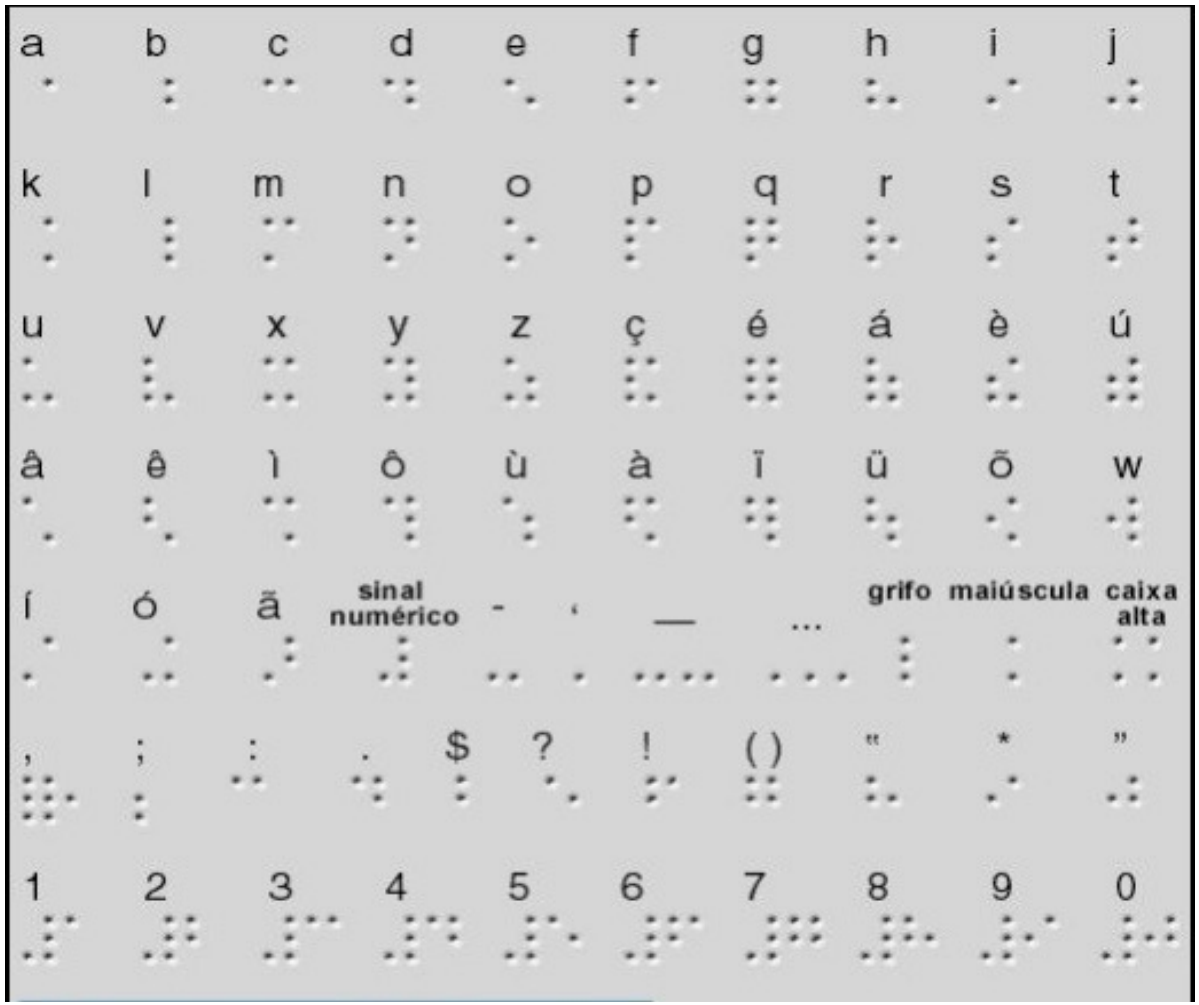
Letters	Figures	V	IV	I	II	III	Letters	Figures	V	IV	I	II	III
A	1			●			←	.	●		●		
E	2				●		X	9/	●			●	
Y	3					●	S	7/	●				●
/	½			●	●		Z	:	●		●	●	
1	¾				●	●	W	?	●			●	●
U	4			●		●	T	2	●		●		●
O	5			●	●	●	V	1	●		●	●	●
							Letter Blank		●				
J	6		●	●			K	(	●	●	●		
G	7		●		●		M	)	●	●		●	
B	8												
H	ⁿ⁄₌												
F	5/												
C	9												
D	0												
Figure Blank													

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

- Baudot, Braille,



Alfabeto Braille



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

***ASCII**: **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange*

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

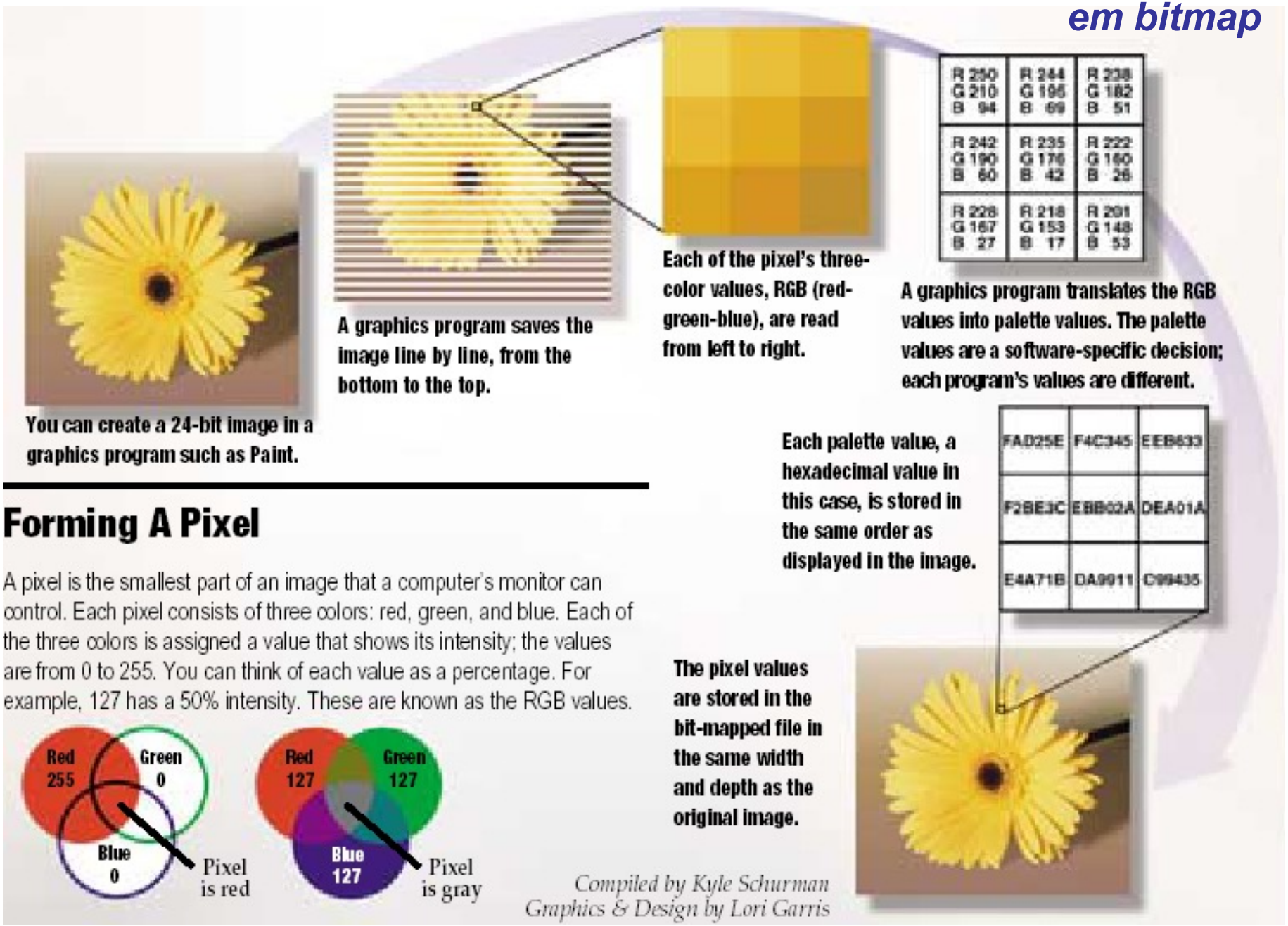
Como se representa a informação?

- com ***b****inary dig****its***!

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



Representação da informação num computador (4)

Como se representa a informação?

- com ***b****inary dig****its***!

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;
```

- Código numa linguagem de programação
  - somar 2 inteiros

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

Idêntico à expressão
`x = x + y`

- Código numa linguagem mais próxima do processador
  - somar 2 inteiros (de 4-*bytes*)
  - operandos:
    - x: no registo `eax`
    - y: na memória em `[(ebp)+8]`

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

- Código "objecto" (em hexadecimal)
  - instrução com 3-*bytes*
  - na memória em `0x401046`