Bases Numéricas

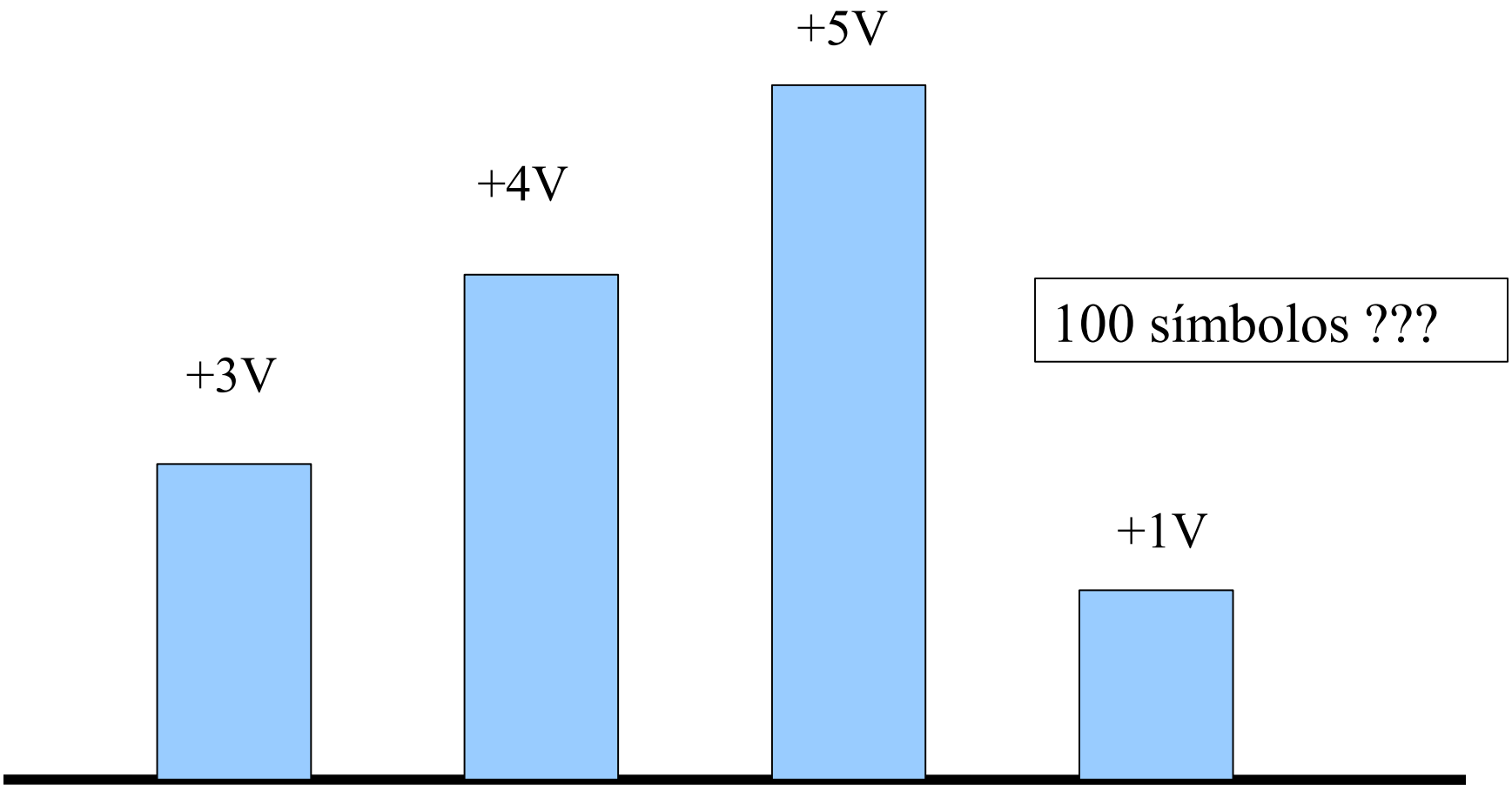
Sistema de Numeração

- Os computadores manipulam dados (sinais brutos e sem significado individual) para produzir informações.
- A conversão de dados em informações, e estas novamente em dados, é uma parte tão fundamental em relação ao que os computadores fazem que é preciso saber como a conversão ocorre para compreender como o computador funciona.
- Aspecto importante: sistema de numeração

Sistema de Numeração

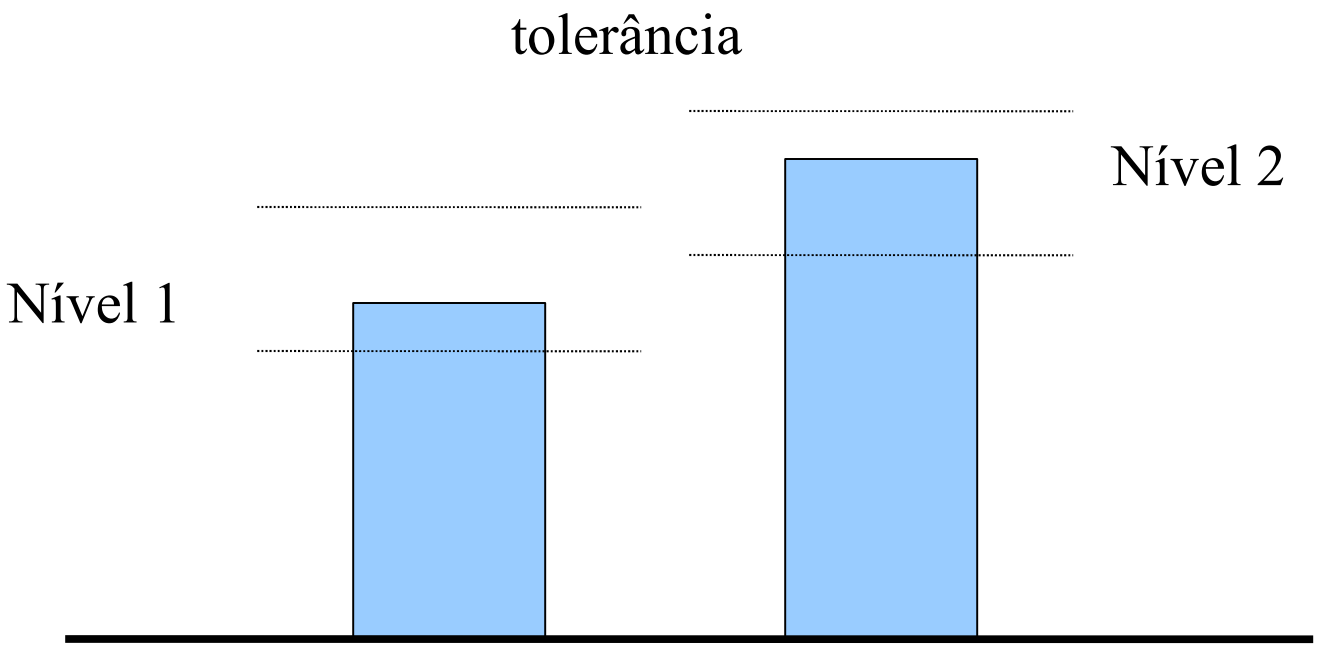
- Conjunto de símbolos utilizados para representação de quantidades e de regras que definem a forma de representação.
- Cada sistema de numeração é apenas um método diferente de representar quantidades.
- As quantidades em si não mudam; mudam apenas os símbolos usados para representá-las.

Símbolos - Níveis de Tensão Elétrica



Símbolos

- ENIAC (1946) – usava o sistema decimal (10 símbolos ou 10 níveis de tensão elétrica)
- Problemas de custo e confiabilidade (consumo de energia, dissipação, ...)



Símbolos

- Solução melhor: utilizar uma quantidade menor de níveis de tensão
- Base binária (2 níveis de tensão)
- O hardware básico (válvulas, relés, chaves, transistores) são binários: passa ou não corrente elétrica, fechada ou aberta, etc.
- Lógica de programação é binária (se/senão)
- Chamamos cada letra de um **dígito binário** ou **bit**

Sistema de Numeração

- A quantidade de algarismos disponíveis em um dado sistema de numeração é chamada de base (b).
- Representação numérica mais empregada: notação posicional
  - Os algarismos componentes dos números assumem valores diferentes, dependendo da posição relativa no número:

123 é diferente de 321

Sistemas de Numeração

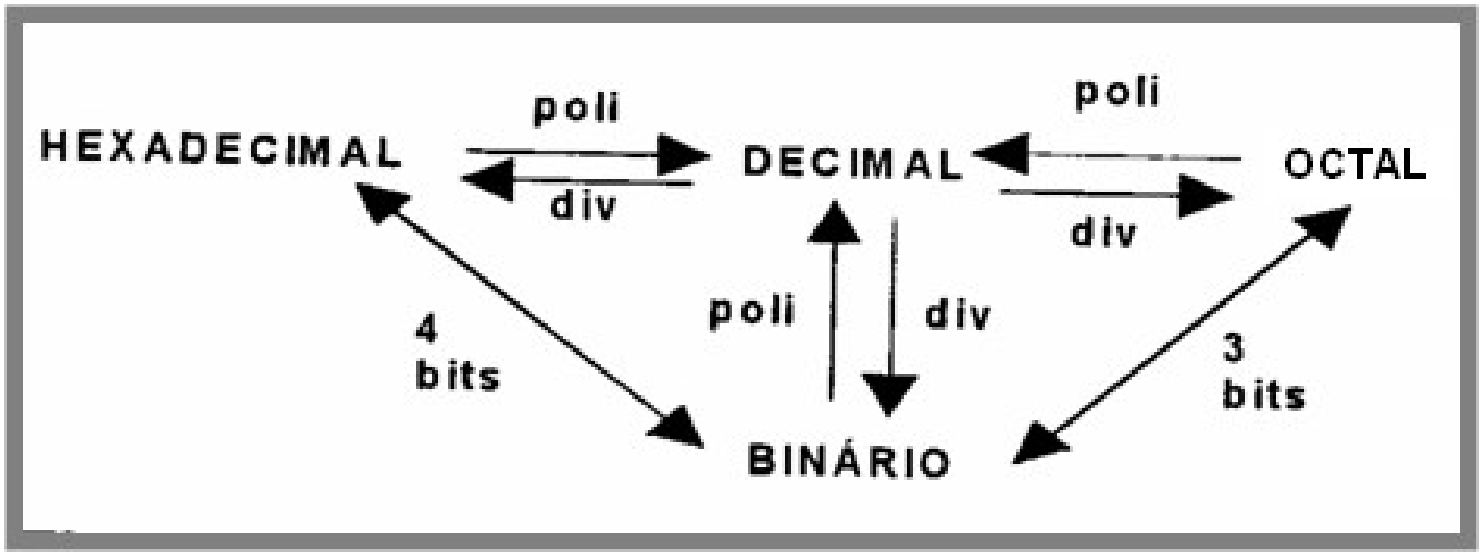
Sistema	Base	Dígitos
Binário	2	0,1
Octal	8	0,1,2,3,4,5,6,7
Decimal	10	0,1,2,3,4,5,6,7,8,9
Hexadecimal	16	0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Números em diferentes bases

Decimal (Base 10)	Binário (Base 2)	Octal (Base 8)	Hexadecimal (Base 16)
00	00000	00	00
01	00001	01	01
02	00010	02	02
03	00011	03	03
04	00100	04	04
05	00101	05	05
06	00110	06	06
07	00111	07	07
08	01000	10	08
09	01001	11	09
10	01010	12	0A
11	01011	13	0B
12	01100	14	0C
13	01101	15	0D
14	01110	16	0E
15	01111	17	0F
16	10000	20	10

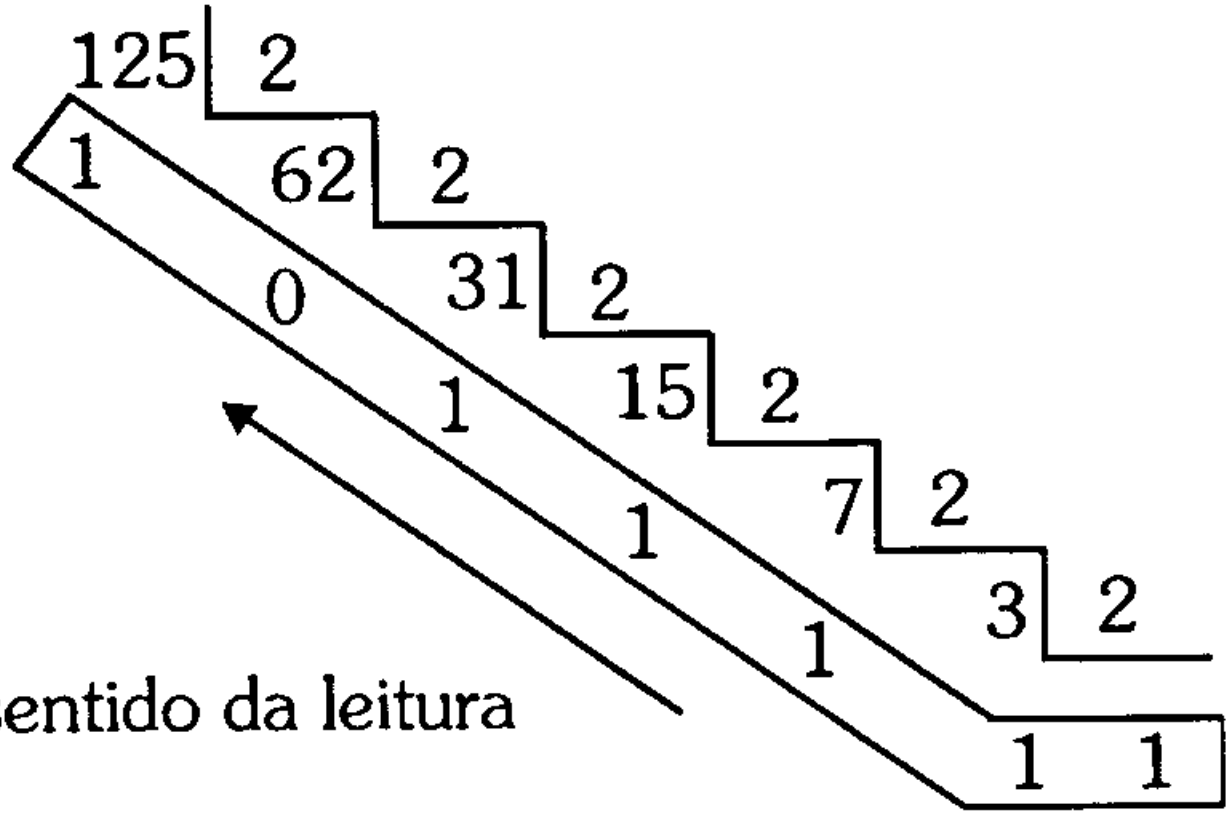
Conversão entre Sistemas de Numeração

- Procedimentos básicos:
  - Divisão (números inteiros)
  - Polinômio
  - Agrupamento de bits



Divisão (decimal → outro sistema)

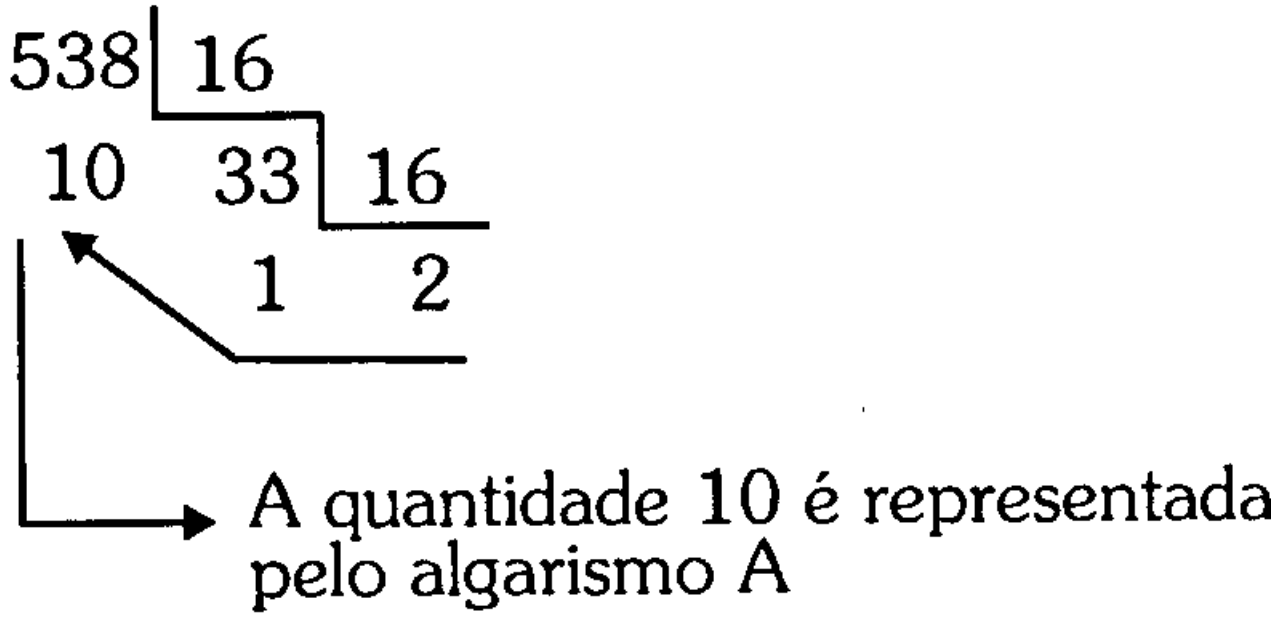
Ex.: $(125)_{10} = (?)_2$



$$(125)_{10} = (1111101)_2$$

Divisão (decimal → outro sistema)

$(538)_{10} = (?)_{16}$



$(538)_{10} = (21A)_{16}$

Notação Polinomial

- Válida para qualquer base numérica.

- LEI DE FORMAÇÃO:

$$a_n b^n + a_{n-1} b^{n-1} + a_{n-2} b^{n-2} + \ldots + a_0 b^0$$

Número =

a_n = algarismo, b = base do número

n = quantidade de algarismo - 1

Notação Polinomial

Ex.:

a) $(1111101)_2 = (?)_{10}$

$(1111101)_2 =$

$1x2^6 + 1x2^5 + 1x2^4 + 1x2^3 + 1x2^2 + 0x2^1 + 1x2^0 = 125_{10}$

b) $(21A)_{16} = (?)_{10}$

$(21A)_{16} = 2x16^2 + 1x16^1 + 10x16^0 = 538_{10}$

Conversão octal→hexadecimal

- Não é realizada diretamente. Não há relação de potências entre as bases oito e dezesseis.

- Semelhante à conversão entre duas bases quaisquer - **base intermediária** (base binária)

Conversão em duas etapas:

1 - número: base octal (hexadecimal) → binária.

2 - resultado intermediário: binária → hexadecimal (octal).

Conversão octal→hexadecimal

Ex.:

a) $(175)_8 = (\ ?\ )_{16}$

$(175)_8 = (1111101)_2 = (7D)_{16}$

b) $(21A)_{16} = (?)_8$

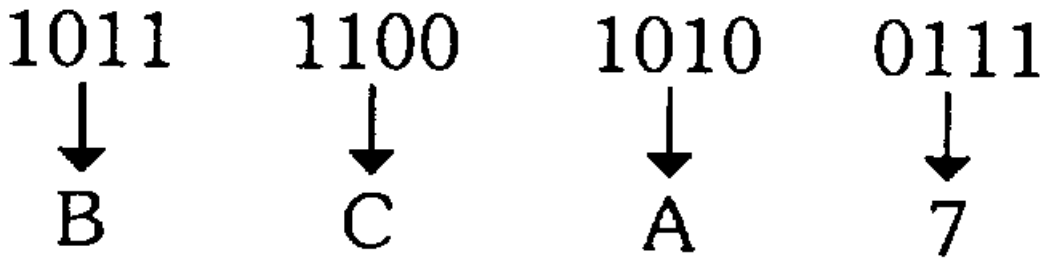
$(21A)_{16} = (001000011010)_2 = (1032)_8$

Agrupamento de Bits

- **Sistemas octal, hexa → binário (e vice-versa)**

- Associando 3 bits ou 4 bits (quando octal ou hexadecimal, respectivamente) e vice-versa.

Ex.:



$$(1011110010100111)_2 = (BCA7)_{16}$$

Agrupamento de Bits

- **Sistemas octal, hexa → binário (e vice versa)**

- Associando 3 bits ou 4 bits (quando octal ou hexadecimal, respectivamente) e vice-versa.

Ex.: **(A**

$$(A79E)_{16} = (1010011110011110)_2$$

Números Fracionários

Lei de Formação ampliada (polinômio):

$$\text{Número} = \underbrace{a_n.b^n + a_{n-1}.b^{n-1} + a_{n-2}.b^{n-2} + \ldots + a_0.b^0}_{\text{parte inteira}} + \underbrace{a_{-1}.b^{-1} + a_{-2}.b^{-2} + \ldots + a_{-m}.b^{-m}}_{\text{parte fracionária}}$$

Exemplo: $(101,110)_2 = (\ ?\ )_{10}$

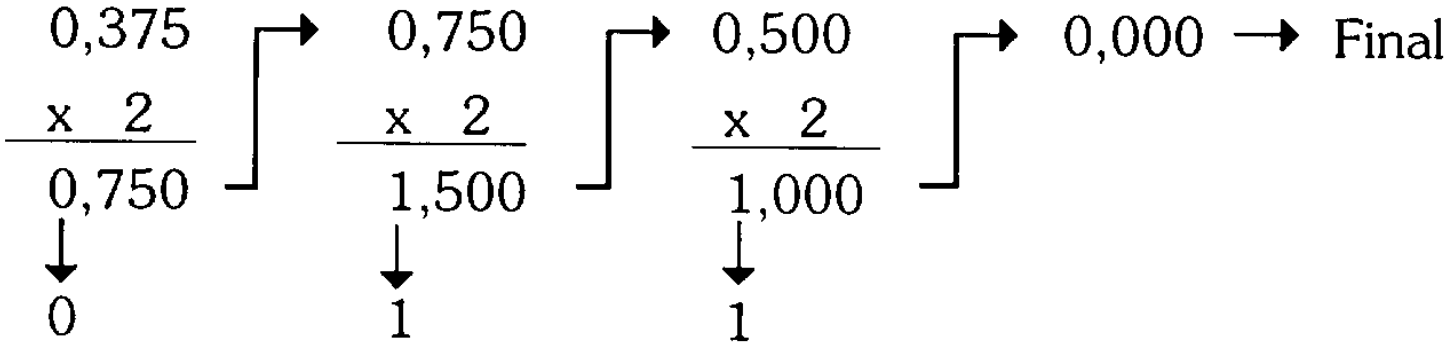
$1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 0 \times 2^{-3} = (5,75)_{10}$

Números Fracionários

Operação inversa: multiplicar a parte fracionária pela base até que a parte fracionária do resultado seja zero.

Exemplo: $(8{,}375)_{10} = (\ ?\ )_2$

- parte inteira: $(8)_{10} = (1000)_2$
- parte fracionária:



$(8{,}375)_{10} = (1000{,}011)_2$

Exercício

Considere o número de alunos presentes na sala de aula. Multiplique-o por 100 e represente-o nos sistemas
-decimal,
-hexadecimal e
-binário.

Some ao número em binário (já multiplicado) o valor $(0,111)_2$ e represente o resultado nos sistemas
-binário,
-decimal.

Sistemas de Numeração

		Geral	Decimal	Binário
Base		r	10	2
Dígitos		0 => r - 1	0 => 9	0 => 1
Potências da base	0	r^0	1	1
	1	r^1	10	2
	2	r^2	100	4
	3	r^3	1000	8
	4	r^4	10.000	16
	5	r^5	100.000	32
	-1	r^{-1}	0,1	0,5
	-2	r^{-2}	0,01	0,25
	-3	r^{-3}	0,01	0,125
	-4	r^{-4}	0,0001	0,0625
	-5	r^{-5}	0,00001	0,03125

Números Binários e Códigos Binários

- **Flexibilidade de representação**

- **Tipos de informação**
  - Numérica
  - Não numérica

Códigos Binários Não Numéricos

- Dados ***n*** dígitos binários, um código binário é um mapeamento de um conjunto de elementos em um subconjunto de 2^n números binários.

Cor	Número Binário
Vermelho	000
Alaranjado	001
Amarelo	010
Verde	011
Azul	101
Azul escuro	110
Violeta	111

Número de bits necessários

- Dados M elementos a serem representados por um código binário, o número mínimo de bits, n, necessários, satisfaz a relação:

$$2^n \geq M > 2^{(n-1)}$$

$$n = \log_2 M$$

- **Ex.:** Quantos bits são necessários para representar dígitos decimais em um código binário?

Códigos Decimais

- Alguns exemplos:

Decimal	8,4,2,1	Excesso de 3	8,4,-2,-1	Gray
0	0000	0011	0000	0000
1	0001	0100	0111	0100
2	0010	0101	0110	0101
3	0011	0110	0101	0111
4	0100	0111	0100	0110
5	0101	1000	1011	0010
6	0110	1001	1010	0011
7	0111	1010	1001	0001
8	1000	1011	1000	1001
9	1001	1100	1111	1000

- Quais as características desses códigos?

Cuidado: Conversão ou Codificação?

- $13_{10} = 1101_2$ (**conversão**)

- $13 \Leftrightarrow 0001|0011$ (**codificação**)

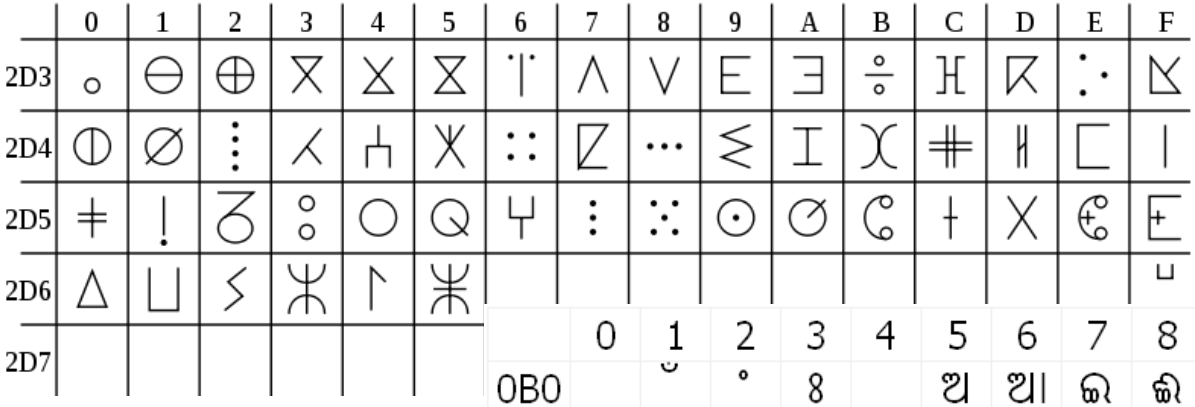
ASCII

- Padrão definido pela organização ANSI (*American National Standards Institute*).
- Código de 7 bits (128 combinações de caracteres).
- ASCII Estendido (utiliza outros 128 códigos para símbolos gráficos, e línguas diferentes do inglês).

- $0\text{-}9_{10} \rightarrow 30 - 39_{16}$
- $A\text{-}Z \rightarrow 41 - 5A_{16}$
- $a\text{-}z \rightarrow 61 - 7A_{16}$

UNICODE

- Oferece 2 bytes para a representação de símbolos (mais de 65.000 símbolos).



	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0B0		◌	◌	ଃ		ଅ	ଆ	ଇ	ଈ	ଉ	ଊ	ଋ	ଌ			ଏ
0B1	ଐ			ଓ	ଔ	କ	ଖ	ଗ	ଘ	ଙ	ଚ	ଛ	ଜ	ଝ	ଞ	ଟ
0B2	ଠ	ଡ	ଢ	ଣ	ତ	ଥ	ଦ	ଧ	ନ		ପ	ଫ	ବ	ଭ	ମ	ଯ
0B3	ର		ଲ	ଳ	ଵ		ଶ	ଷ	ସ	ହ				୍	ଁ	
0B4	1							ଽ	◌ା	◌ି			◌ୀ	◌ୁ		
0B5								◌ୃ					ଡ଼	ଢ଼		ୟ
0B6	ୠ	ୡ					୦	୧	୨	୩	୪	୫	୬	୭	୮	୯
0B7	✓	◌ୢ														

Código para Deteção de Erro: Bit de Paridade

- **Características**
  - Insere redundância
  - Tipos
    - **Paridade Par** → o número de bits iguais a 1 é par
    - **Paridade Ímpar** → o número de bits iguais a 1 é ímpar
  - Pode detectar erro em um bit (ou múltiplos bits)
  - Exemplo de representação de um dado de 3 bits:
    - Palavra-código **"1111"**: **paridade par**
    - Palavra-código **"1110": paridade ímpar**