

# Sistemas Digitais

ET46B

---

Prof. Eduardo Vinicius Kuhn

*[kuhn@utfpr.edu.br](mailto:kuhn@utfpr.edu.br)*

Curso de Engenharia Eletrônica

Universidade Tecnológica Federal do Paraná



# Capítulo 2

## Sistemas de numeração e códigos

# Conteúdo

2.1 Conversões de binário para decimal

2.2 Conversões de decimal para binário

2.3 Sistema de numeração hexadecimal

2.4 Código BCD

2.5 Código Gray

2.6 Relações entre as representações numéricas

2.7 Bytes, nibbles e palavras

2.8 Códigos alfanuméricos

# Objetivos

- Converter um número de um **sistema de numeração** (e.g., **decimal, binário ou hexadecimal**) para outro.
- Abordar o **sistema de numeração hexadecimal** e destacar as suas vantagens.
- Tratar da representação de números decimais usando o **código BCD** e explicitar a diferença para a representação binária.
- Introduzir o **código Gray**, código Johnson, codificação *one-hot* e um exemplo de código alfanumérico (i.e., o **ASCII**).
- Definir o conceito de **bytes, nibbles e palavras**.

# Sistemas de numeração

Considere **um número qualquer**

$$x = (d_n \dots d_0, d_{-1} \dots d_{-m})_B, \quad n, m \in \mathbb{Z}$$

representado em **um sistema de numeração de base  $B$**  (e.g., 2, 8, 10, 16), com  $D$  dígitos  $d_n$ .

A partir disso, é possível mostrar que  $x$  pode ser expresso como

$$x = d_n B^n + \dots + d_0 B^0 + d_{-1} B^{-1} + \dots + d_{-m} B^{-m}$$

Vale destacar que essa representação é também válida  
para outros sistemas de numeração posicionais.

## Revisitando o sistema de numeração decimal

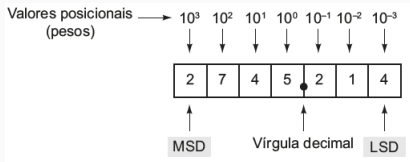
- O sistema **decimal** é composto por 10 símbolos/dígitos.
- **Base 10** por ter dez dígitos, i.e., 0-9.
- **É um sistema posicional.**
- **MSD** (dígito mais a esquerda) e **LSD** (dígito mais a direita).
- A palavra dígito é derivada da palavra 'dedo' em latim.

**Exemplo:** Como representar o número  $2745,214_{10}$ ?

# Revisitando o sistema de numeração decimal

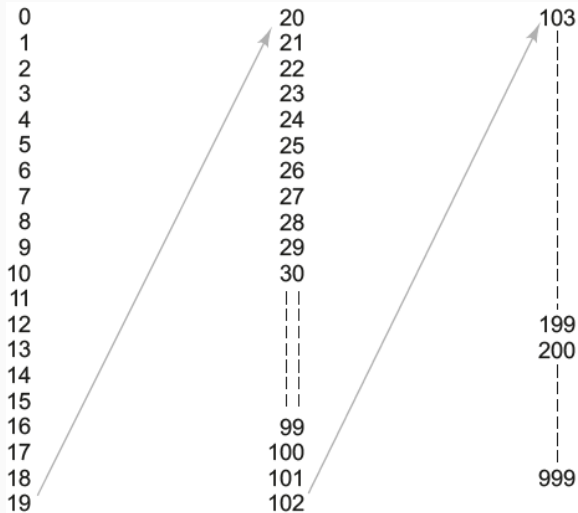
- O sistema **decimal** é composto por 10 símbolos/dígitos.
- **Base 10** por ter dez dígitos, i.e., 0-9.
- **É um sistema posicional.**
- **MSD** (dígito mais a esquerda) e **LSD** (dígito mais a direita).
- A palavra dígito é derivada da palavra 'dedo' em latim.

**Exemplo:** Como representar o número  $2745,214_{10}$ ?



$$(2 \times 10^3) + (7 \times 10^2) + (4 \times 10^1) + (5 \times 10^0) + (2 \times 10^{-1}) + (1 \times 10^{-2}) + (4 \times 10^{-3})$$

# Contagem em um sistema de numeração decimal





Com  $D$  dígitos decimais, quantos números diferentes podem ser representados?

Com  $D$  dígitos decimais, quantos números diferentes podem ser representados?

$10^D$  números diferentes

$$\{0, \dots, 10^D - 1\}$$

# Sistema de numeração binário

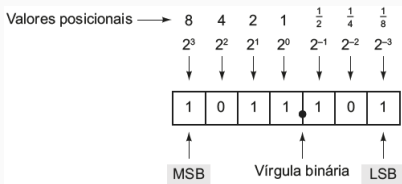
- O sistema **binário** é composto por 2 símbolos/dígitos.
- **Base 2** por ter 2 dígitos, i.e., 0 e 1.
- É um **sistema posicional**.
- **MSB** (bit mais a esquerda) e **LSB** (bit mais a direita).
- O termo dígito binário (*binary digit*) é abreviado para 'bit'.

**Exemplo:** Como representar o número  $1011,101_2$ ?

# Sistema de numeração binário

- O sistema **binário** é composto por 2 símbolos/dígitos.
- **Base 2** por ter 2 dígitos, i.e., 0 e 1.
- É um **sistema posicional**.
- **MSB** (bit mais a esquerda) e **LSB** (bit mais a direita).
- O termo dígito binário (*binary digit*) é abreviado para 'bit'.

**Exemplo:** Como representar o número  $1011,101_2$ ?



$$(1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3})$$

## Contagem em um sistema de numeração binário

Pesos →	$2^3 = 8$	$2^2 = 4$	$2^1 = 2$	$2^0 = 1$		Número decimal equivalente
	0	0	0	0	→	0
	0	0	0	1	→	1
	0	0	1	0	⋮	2
	0	0	1	1		3
	0	1	0	0		4
	0	1	0	1		5
	0	1	1	0		6
	0	1	1	1		7
	1	0	0	0		8
	1	0	0	1	9	
	1	0	1	0	10	
	1	0	1	1	11	
	1	1	0	0	12	
	1	1	0	1	13	
	1	1	1	0	→	14
	1	1	1	1	→	15

↑  
LSB

Com  $D$  bits, quantos números diferentes  
podem ser representados?

Com  $D$  bits, quantos números diferentes podem ser representados?

$2^D$  números diferentes

$$\{0, \dots, 2^D - 1\}$$

# Sistema de numeração binário

**Exemplo:** Considerando a representação binária, determine:

- a) Qual é o número decimal equivalente a  $1101011_2$ ?
- b) Qual é o número binário seguinte a  $10111_2$  em uma sequência de contagem?
- c) Qual o maior número decimal que pode ser representado usando 8 bits?
- d) Qual é o menor número decimal que pode ser representado usando 12 bits?



# Sistema de numeração binário

**Exemplo:** Considerando a representação binária, determine:

a) Qual é o número decimal equivalente a  $1101011_2$ ?

**R:**  $107_{10}$

b) Qual é o número binário seguinte a  $10111_2$  em uma sequência de contagem?

**R:**  $11000_2$

c) Qual o maior número decimal que pode ser representado usando 8 bits?

**R:**  $2^8 - 1 = 255_{10}$

d) Qual é o menor número decimal que pode ser representado usando 12 bits?

**R:**  $0_{10}$

# Conversão de decimal para binário

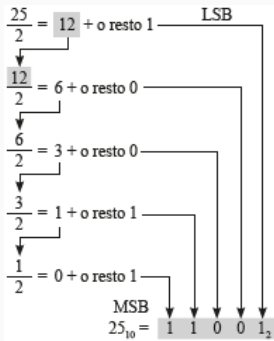
Existem 2 maneiras de converter um número decimal inteiro em seu equivalente binário, a saber:

- 1) **Expressar o número decimal como uma soma de potências de 2**; então, 1s e 0s são colocados nas posições apropriadas, e.g.,

$$\begin{aligned}45_{10} &= 32 + 8 + 4 + 1 \\&= 2^5 + 0 + 2^3 + 2^2 + 0 + 2^0 \\&= 101101_2\end{aligned}$$

# Conversão de decimal para binário

- 2) **Utilizar divisões sucessivas por 2;** então, o resto de cada divisão constitui a representação binária (do LSB ao MSB) **até que um quociente 0 seja obtido**, e.g.,



# Conversão de decimal para binário

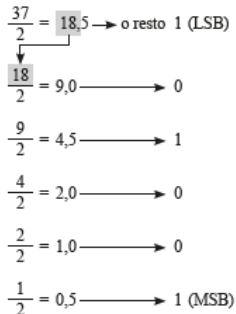
**Exemplo:** Converta  $37_{10}$  para a representação binária.

# Conversão de decimal para binário

**Exemplo:** Converta  $37_{10}$  para a representação binária.

**R:**

$$\begin{aligned} 37_{10} &= 32 + 4 + 1 \\ &= 2^5 + 0 + 0 + 2^2 + 0 + 2^0 \\ &= 100101_2. \end{aligned}$$


$$\begin{array}{l} \frac{37}{2} = 18,5 \rightarrow \text{o resto } 1 \text{ (LSB)} \\ \downarrow \\ \frac{18}{2} = 9,0 \rightarrow 0 \\ \frac{9}{2} = 4,5 \rightarrow 1 \\ \frac{4}{2} = 2,0 \rightarrow 0 \\ \frac{2}{2} = 1,0 \rightarrow 0 \\ \frac{1}{2} = 0,5 \rightarrow 1 \text{ (MSB)} \end{array}$$

## Conversão de decimal para binário

E, como lidar com números “não inteiros” (fracionários)?

[PI], [PF]

- [PI]  $\Rightarrow$  Usando divisões sucessivas por 2.
- [PF]  $\Rightarrow$  Multiplica-se as **partes fracionárias** sucessivamente por 2, pegando as partes inteiras como resultados.
- O separador “,” se mantém.

# Conversão de decimal para binário

**Exemplo:** Converta  $3,25_{10}$  para a representação binária.

# Conversão de decimal para binário

**Exemplo:** Converta  $3,25_{10}$  para a representação binária.

**R:** A conversão da parte inteira segue por divisões sucessivas, i.e.,

$$3_{10} = 11_2$$

Por sua vez, a parte fracionária, é determinada multiplicando sucessivamente por 2 e pegando as partes inteiras, i.e.,

$$0,25_{10} \times 2 \Rightarrow 0 + \text{resto } 0,5$$

$$0,5_{10} \times 2 \Rightarrow 1 + \text{resto } 0$$

o que implica

$$0,25_{10} = 0,01_2$$

Portanto,

$$3,25_{10} = 11,01_2$$



# Sistemas de numeração hexadecimal

- O sistema **hexadecimal** contém 16 símbolos/dígitos.
- **Base 16**, i.e., 0–9 e A–F.
- É um **sistema posicional**.
- Os dígitos A–F equivalem aos valores decimais de 10–15.
- 1 dígito hexa corresponde a 4 bits...

Hexadecimal	Decimal	Binário
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111

Hexadecimal	Decimal	Binário
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

# Sistemas de numeração hexadecimal

**Exemplo:** Converta  $2AF_{16}$  para decimal.

**Exemplo:** Converta  $214_{10}$  para hexadecimal.

# Sistemas de numeração hexadecimal

**Exemplo:** Converta  $2AF_{16}$  para decimal.

**R:** Usando agora potências de 16,

$$\begin{aligned} 2AF_{16} &= 2 \times 16^2 + 10 \times 16^1 + 15 \times 16^0 \\ &= 512 + 160 + 15 \\ &= 687_{10} \end{aligned}$$

**Exemplo:** Converta  $214_{10}$  para hexadecimal.

**R:** Realizando divisões sucessivas,

$$214_{10} = D6_{16}$$

$$\begin{array}{l} \frac{214}{16} = 13 + \text{o resto } 6 \\ \downarrow \\ \frac{13}{16} = 0 + \text{o resto } 13 \\ \downarrow \quad \downarrow \\ 214_{10} = D6_{16} \end{array}$$

(Os restos das divisões sucessivas formam os dígitos do número hexa, sendo restos maiores do que 9 representados pelas letras de A–F.)

# Sistemas de numeração hexadecimal

**Exemplo:** Converta  $423_{10}$  para hexadecimal.

Hexadecimal	Decimal	Binário
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111

Hexadecimal	Decimal	Binário
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

# Sistemas de numeração hexadecimal

**Exemplo:** Converta  $423_{10}$  para hexadecimal.

Hexadecimal	Decimal	Binário	Hexadecimal	Decimal	Binário
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	A	10	1010
3	3	0011	B	11	1011
4	4	0100	C	12	1100
5	5	0101	D	13	1101
6	6	0110	E	14	1110
7	7	0111	F	15	1111

**R:** Realizando divisões sucessivas,

$$\begin{array}{l} \frac{423}{16} = 26 + \text{o resto } 7 \\ \downarrow \\ \frac{26}{16} = 1 + \text{o resto } 10 \\ \downarrow \\ \frac{1}{16} = 0 + \text{o resto } 1 \end{array}$$

$423_{10} = 1A7_{16}$

# Sistemas de numeração hexadecimal

**Conversão de binário em hexa:** consiste, simplesmente, em dispor o número binário em grupos de quatro bits e converter cada grupo no dígito hexa equivalente, e.g.,

$$\begin{array}{r} 1110100110_2 = \overbrace{0011}^3 \overbrace{1010}^A \overbrace{0110}^6 \\ = 3A6_{16} \end{array}$$

(Zeros podem ser acrescentados para completar um grupo de 4 bits.)

**Conversão de hexa em binário:** consiste em converter cada dígito hexa no equivalente binário de 4 bits.

A conversão de binário em hexa (e vice-versa) é simples, o que justifica a utilização do sistema hexadecimal.

# Contagem em um sistema de numeração hexadecimal

Quando contamos em hexa, cada dígito pode ser incrementado (acrescido de 1) de 0–F, e.g.,

- 38, 39, 3A, 3B, 3C, 3D, 3E, 3F, 40, 41, 42
- 6F8, 6F9, 6FA, 6FB, 6FC, 6FD, 6FE, 6FF, 700

(Note que, um dígito 9 quando incrementado, torna-se A.)

Com  $D$  dígitos hexa, quantos números diferentes podem ser representados?



Com  $D$  dígitos hexa, quantos números diferentes podem ser representados?

$16^D$  números diferentes

$$\{0, \dots, 16^D - 1\}$$

# Vantagens do sistema de numeração hexadecimal

- O sistema hexadecimal é usado como uma representação compacta de sequências binárias.
- Sequências binárias longas (de até 64 bits) são comuns (em computadores).
- A escrita em hexadecimal é **mais conveniente e menos propensa a erros** do que a representação binária.

---

Humanos operam com números decimais, sistemas digitais com números binários, e a numeração hexadecimal facilita aos humanos lidarem com números binários.

# Utilização do sistema de numeração hexadecimal

**Exemplo:** Converta  $378_{10}$  em um número binário de 16 bits por meio do sistema de numeração hexadecimal.

# Utilização do sistema de numeração hexadecimal

**Exemplo:** Converta  $378_{10}$  em um número binário de 16 bits por meio do sistema de numeração hexadecimal.

**R:** Realizando a divisão sucessiva por 16 e coletando o resto, obtém-se

$$378_{10} = 17A_{16}.$$

Então, levando em conta que cada dígito hexa pode ser representado por 4 bits,

$$17A_{16} = 0001\ 0111\ 1010_2.$$

Portanto,

$$378_{10} = 0001\ 0111\ 1010_2.$$

# Resumo sobre as conversões

- **Binário/Hexa  $\Rightarrow$  Decimal:** Utilize a soma dos pesos de cada dígito.
  - **Decimal  $\Rightarrow$  Binário/Hexa:** Realize divisões sucessivas por 2 (binário) ou 16 (hexa), reunindo os restos da divisão.
- 
- **Binário  $\Rightarrow$  Hexa:** Agrupe os bits em grupos de 4 e converta cada grupo no dígito hexa equivalente.
  - **Hexa  $\Rightarrow$  Binário:** Represente cada dígito hexa em 4 bits equivalentes.

# Diferença entre sistemas de numeração e código

Um **sistema de numeração** possui uma base numérica e uma estrutura posicional que permite representar quantidades numéricas de forma única, e.g.,

- Decimal (base 10)
  - Binário (base 2)
  - Hexadecimal (base 16)
- 

Um **código** é uma atribuição (mapeamento) de símbolos para representar informações sem necessitar de estrutura posicional ou base numérica, e.g.,

- **BCD**
- **ASCII**
- Unicode

## Código BCD (decimal codificado em binário)

- Quando **cada dígito de um número decimal é representado por seu equivalente binário**, tem-se como resultado um **código decimal codificado em binário** (*binary-coded-decimal*).
- Um dígito decimal tem valor máximo igual a 9; logo, 4 bits são necessários para **codificar cada dígito** (e.g.,  $9_{10} = 1001_2$ ).
- Pouco eficiente já que requer mais bits para representar um dado número decimal.

**Exemplo:** Converta  $874_{10}$  para BCD.

# Código BCD (decimal codificado em binário)

- Quando **cada dígito de um número decimal é representado por seu equivalente binário**, tem-se como resultado um **código decimal codificado em binário** (*binary-coded-decimal*).
- Um dígito decimal tem valor máximo igual a 9; logo, 4 bits são necessários para **codificar cada dígito** (e.g.,  $9_{10} = 1001_2$ ).
- Pouco eficiente já que requer mais bits para representar um dado número decimal.

**Exemplo:** Converta  $874_{10}$  para BCD.

**R:**

8	7	4	(decimal)
↓	↓	↓	
1000	0111	0100	(BCD)



BCD é um código em que cada dígito é **codificado** em seu equivalente binário (**não é um sistema de numeração**).

## Código BCD (decimal codificado em binário)

**Exemplo:** Converta  $110\ 1000\ 0011\ 1001_{(\text{BCD})}$  em seu equivalente decimal.

# Código BCD (decimal codificado em binário)

**Exemplo:** Converta  $110\ 1000\ 0011\ 1001_{(\text{BCD})}$  em seu equivalente decimal.

**R:** Separando em grupos de 4 bits,

$$0110\ 1000\ 0011\ 1001_{(\text{BCD})}$$

e convertendo cada grupo, obtém-se

$$6839_{10}.$$

Portanto,

$$0110100000111001_{(\text{BCD})} = 6839_{10}.$$

(O código BCD não usa os números 1010 até 1111; logo, o aparecimento desses números 'proibidos' é indicativo de erro.)

# Código Gray

- **Desenvolvido para reduzir erros**, dado que somente um bit muda entre números sucessivos.
- No sistema binário, vários bits podem mudar de estado simultaneamente (e.g., de  $3_{10} = 011_2$  para  $4_{10} = 100_2$ ).
- Aplicações envolvem, e.g., i) **sistemas de comunicação** (16-QAM); ii) codificadores de posição (encoders); iii) algoritmos genéticos (codificação de cromossomos)...

$B_2$	$B_1$	$B_0$	$G_2$	$G_1$	$G_0$
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	1	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	1	1	0	0

## Comparativo entre as representações

Decimal	Binário	Hexadecimal	BCD	Gray
0	0000	0	0000	0000
1	0001	1	0001	0001
2	0010	2	0010	0011
3	0011	3	0011	0010
4	0100	4	0100	0110
5	0101	5	0101	0111
6	0110	6	0110	0101
7	0111	7	0111	0100
8	1000	8	1000	1100
9	1001	9	1001	1101
10	1010	A	0001 0000	1111
11	1011	B	0001 0001	1110
12	1100	C	0001 0010	1010
13	1101	D	0001 0011	1011
14	1110	E	0001 0100	1001
15	1111	F	0001 0101	1000

# Código Johnson

- Frequentemente usado em máquinas de estado finito.
- **Fornecer uma sequência de estados com padrões únicos.**
- Pode ser útil em circuitos lógicos sequenciais simples.

Decimal	Código Johnson	Codificação one-hot
0	00000	0000000001
1	00001	0000000010
2	00011	0000000100
3	00111	0000001000
4	01111	0000010000
5	11111	0000100000
6	11110	0001000000
7	11100	0010000000
8	11000	0100000000
9	10000	1000000000

# Codificação one-hot

- Representa variáveis como vetores binários.
- **Apenas um bit está ativo (hot) por vez.**
- Comumente usada em aprendizado de máquina.

Decimal	Código Johnson	Codificação one-hot
0	00000	0000000001
1	00001	0000000010
2	00011	0000000100
3	00111	0000001000
4	01111	0000010000
5	11111	0000100000
6	11110	0001000000
7	11100	0010000000
8	11000	0100000000
9	10000	1000000000

# Códigos alfanuméricos

- **Códigos alfanuméricos:** Incluem **letras, dígitos numéricos, sinais de pontuação e caracteres especiais**, simbolizando todos os caracteres de um teclado.
  - 26 letras minúsculas, 26 maiúsculas
  - 10 dígitos numéricos
  - 7 sinais de pontuação
  - 20 a 40 caracteres especiais (como +, /, #, %, \*, etc.)
- **ASCII** (*American Standard Code for Information Interchange*):
  - Código de 7 bits, 128 representações codificadas
  - **Inclui caracteres de um teclado padrão e funções como (RETURN) e (LINEFEED)**
  - **Usado para a transferência de informação alfanumérica entre um computador e dispositivos externos**
  - Basta realizar a conversão do valor hexadecimal em binário para obter o código binário de 7 bits



# Código ASCII

Caractere	HEX	Decimal	Caractere	HEX	Decimal	Caractere	HEX	Decimal	Caractere	HEX	Decimal
NUL (null)	0	0	Space	20	32	@	40	64	.	60	96
Start Heading	1	1	!	21	33	A	41	65	a	61	97
Start Text	2	2	"	22	34	B	42	66	b	62	98
End Text	3	3	#	23	35	C	43	67	c	63	99
End Transmit	4	4	\$	24	36	D	44	68	d	64	100
Enquiry	5	5	%	25	37	E	45	69	e	65	101
Acknowledge	6	6	&	26	38	F	46	70	f	66	102
Bell	7	7	`	27	39	G	47	71	g	67	103
Backspace	8	8	(	28	40	H	48	72	h	68	104
Horiz.Tab	9	9	)	29	41	I	49	73	i	69	105
Line Feed	A	10	*	2A	42	J	4A	74	j	6A	106
Vert.Tab	B	11	+	2B	43	K	4B	75	k	6B	107
Form Feed	C	12	,	2C	44	L	4C	76	l	6C	108
Carriage Return	D	13	-	2D	45	M	4D	77	m	6D	109
Shift Out	E	14	.	2E	46	N	4E	78	n	6E	110
Shift In	F	15	/	2F	47	O	4F	79	o	6F	111
Data Link Esc	10	16	0	30	48	P	50	80	p	70	112
Direct Control 1	11	17	1	31	49	Q	51	81	q	71	113
Direct Control 2	12	18	2	32	50	R	52	82	r	72	114
Direct Control 3	13	19	3	33	51	S	53	83	s	73	115
Direct Control 4	14	20	4	34	52	T	54	84	t	74	116
Negative ACK	15	21	5	35	53	U	55	85	u	75	117
Synch Idle	16	22	6	36	54	V	56	86	v	76	118
End Trans Block	17	23	7	37	55	W	57	87	w	77	119
Cancel	18	24	8	38	56	X	58	88	x	78	120
End of Medium	19	25	9	39	57	Y	59	89	y	79	121
Substitute	1A	26	:	3A	58	Z	5A	90	z	7A	122
Escape	1B	27	;	3B	59	[	5B	91	{	7B	123
Form separator	1C	28	<	3C	60	\	5C	92		7C	124
Group separator	1D	29	=	3D	61	]	5D	93	}	7D	125
Record Separator	1E	30	>	3E	62	^	5E	94	~	7E	126
Unit Separator	1F	31	?	3F	63	_	5F	95	Delete	7F	127

# Código ASCII

**Exemplo:** Considere que a seguinte mensagem, codificada em ASCII, é armazenada em posições sucessivas de memória:

01010011 01010100 01001111 01010000

Qual é a mensagem?

(O acréscimo de um bit extra é denominado preenchimento com 0s.)

# Código ASCII

**Exemplo:** Considere que a seguinte mensagem, codificada em ASCII, é armazenada em posições sucessivas de memória:

01010011 01010100 01001111 01010000

Qual é a mensagem?

**R:** Usando a tabela,

0101 0011	⇒	53	⇒	S
0101 0100	⇒	54	⇒	T
0100 1111	⇒	4F	⇒	O
0101 0000	⇒	50	⇒	P

(O acréscimo de um bit extra é denominado preenchimento com 0s.)

# Bytes, nibbles e palavras

- **Bytes:** Grupo de 8 bits; geralmente, microcontroladores manipulam e armazenam informações em grupos de 8 bits.
- **Nibbles:** Usado para nomear **grupos de 4 bits** (e.g., em códigos BCD ou sistema de numeração hexadecimal).
- **Palavras:** Uma palavra (*word*) é um grupo de bits que representa uma unidade de informação, cujo comprimento depende da largura do barramento de dados (*pathway*).
  - **Exemplo:** um  $\mu\text{C}$  pode lidar com palavras de 8 bits enquanto um PC com 8 bytes (64 bits) de cada vez.

O termo 'nibble' ('mordiscar') é usado para representar a metade do tamanho de um 'byte' (soa como 'bite', 'mordida').

## Bytes, nibbles e palavras

**Exemplo:** Com respeito à bytes, nibbles e palavras, responda:

- a) Quantos bytes são necessários para representar  $235_{10}$  em binário?
- b) Qual é o maior valor decimal que pode ser representado em BCD, usando 2 bytes?
- c) Um nibble representa quantos dígitos hexadecimais?
- d) Quantos nibbles existem em um dígito BCD?

# Bytes, nibbles e palavras

**Exemplo:** Com respeito à bytes, nibbles e palavras, responda:

- a) Quantos bytes são necessários para representar  $235_{10}$  em binário?

**R:** Como  $2^8 = 256$ , é necessário apenas 1 byte.

- b) Qual é o maior valor decimal que pode ser representado em BCD, usando 2 bytes?

**R:** Com dois bytes (16 bits), o maior valor em BCD é  $9999_{10}$ .

- c) Um nibble representa quantos dígitos hexadecimais?

**R:** Um nibble (4 bits) pode representar 1 dígito hexadecimal.

- d) Quantos nibbles existem em um dígito BCD?

**R:** Um dígito BCD é representado por 4 bits, i.e., 1 nibble.

- A conversão de números decimais em binários ou hexadecimais pode ser realizado por divisões sucessivas.
- Com  $D$  dígitos em um sistema de numeração de base  $B$ , pode-se representar valores decimais de  $\{0, \dots, B^D - 1\}$ .
- Hexa versus binário: 1 dígito hexa corresponde a 4 bits.
- Código BCD para um número decimal é formado convertendo cada dígito decimal no equivalente binário de 4 bits.
- Código Gray define uma sequência de padrões de bits em que apenas um bit varia entre padrões de sequência sucessivos.
- Um código alfanumérico usa grupos de bits para representar caracteres e funções de um teclado típico.
- **Byte: 8 bits, Nibble: 4 bits** e Palavra: depende do sistema.

# Considerações finais

## Exercícios sugeridos:

2.1-2.7, 2.10, 2.15, 2.16, 2.19-2.22, 2.37 e 2.40

de R.J. Tocci, N.S. Widmer, G.L. Moss, *Sistemas digitais: princípios e aplicações*, 11a ed., São Paulo: Pearson, 2011. → (Capítulo 2)

## Para a próxima aula:

R.J. Tocci, N.S. Widmer, G.L. Moss, *Sistemas digitais: princípios e aplicações*, 11a ed., São Paulo: Pearson, 2011. → (Capítulo 3)

Até a próxima aula... =)