Circuitos Lógicos

Capítulo 3 – Códigos

Prof. Erivelton Geraldo Nepomuceno
http://www.ufsj.edu.br/nepomuceno
nepomuceno@ufsj.edu.br

São João del-Rei, agosto de 2015.

Tópicos da aula – capítulo 3

Recapitulação da aula anterior

- Códigos
  - Código BCD
  - Código Gray
  - Código ASCII
  - Outros códigos
- Detecção de erros

O código BCD (*Binary Coded Decimal*)

- **BCD**
  - **Cada dígito decimal é representado pelo seu equivalente em binário**

O código BCD (*Binary Coded Decimal*)

- **BCD**
  - **Cada dígito decimal é representado pelo seu equivalente em binário**

Ex. Usando a codificação BCD represente o número 637_{10}

6	3	7
0110	0011	0111

UFSJ
UNIVERSIDADE FEDERAL
DE SÃO JOÃO DEL-REI

Representações numéricas

Decimal	Binário	Octal	Hexadecimal	BCD
0	0	0	0	0000
1	1	1	1	0001
2	10	2	2	0010
3	11	3	3	0011
4	100	4	4	0100
5	101	5	5	0101
6	110	6	6	0110
7	111	7	7	0111
8	1000	10	8	1000
9	1001	11	9	1001
10	1010	12	A	0001 0000
11	1011	13	B	0001 0001
12	1100	14	C	0001 0010
13	1101	15	D	0001 0011
14	1110	16	E	0001 0100
15	1111	17	F	0001 0101

O código BCD (*Binary Coded Decimal*)

- **Exercíco: Converta para os números abaixo para o seu equivalente em BCD.**

b) 108_{10}
c) 75_{10}
d) 47_{8}
e) $31F_{16}$
f) 20_{8}
g) $2A_{16}$

O código BCD (*Binary Coded Decimal*)

- **Exercíco: Converta para os números abaixo para o seu equivalente em BCD.**

b) 108_{10} ➔ 000100001000_{BCD}

c) 75_{10} ➔ 01110101_{BCD}

d) 47_{8} ➔ 39_{10} ➔ 00111001_{BCD}

e) $31F_{16}$ ➔ 49_{10} ➔ 01001001_{BCD}

f) 20_{8} ➔ 16_{10} ➔ 00010110_{BCD}

g) $2A_{16}$ ➔ 47_{10} ➔ 01000111_{BCD}

O código BCD (*Binary Coded Decimal*)

- **Exercíco: Conversa o número BCD para o seu equivalente decimal**

b) **00011001**
c) **00010011**
d) **01010111**

O código BCD (*Binary Coded Decimal*)

- **Exercíco: Conversa o número BCD para o seu equivalente decimal**

b) **00011001 ➔ 19_{10}**
c) **00010011 ➔ 13_{10}**
d) **01010111 ➔ 57_{10}**

O código BCD (*Binary Coded Decimal*)

- **Exercíco: Conversa o número BCD para o seu equivalente binário**

b) **01111001**
c) **10010101**
d) **1001**
e) **00110111**
f) **000100100011**

O código BCD (*Binary Coded Decimal*)

- **Exercíco: Conversa o número BCD para o seu equivalente binário**

b) **01111001** ➔ 1001111_2
c) **10010101** ➔ 1011111_2
d) **1001** ➔ 1001_2
e) **00110111** ➔ 100101_2
f) **000100100011** ➔ 1111011_2

O código *Gray*

- **Contagem em que muda apenas um bit de cada vez!**

— Usado em transmissão de informação
— Encoders

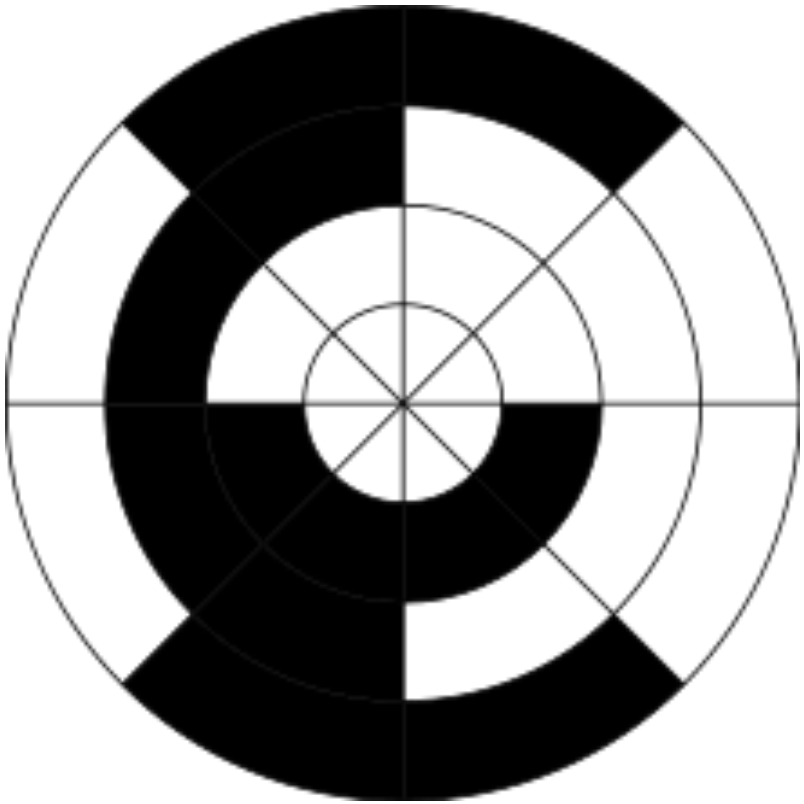
Binário	Gray
000	000
001	001
010	011
011	010
100	110
101	111
110	101
111	100

O código *Gray*

- **Contagem em que muda apenas um bit de cada vez!**

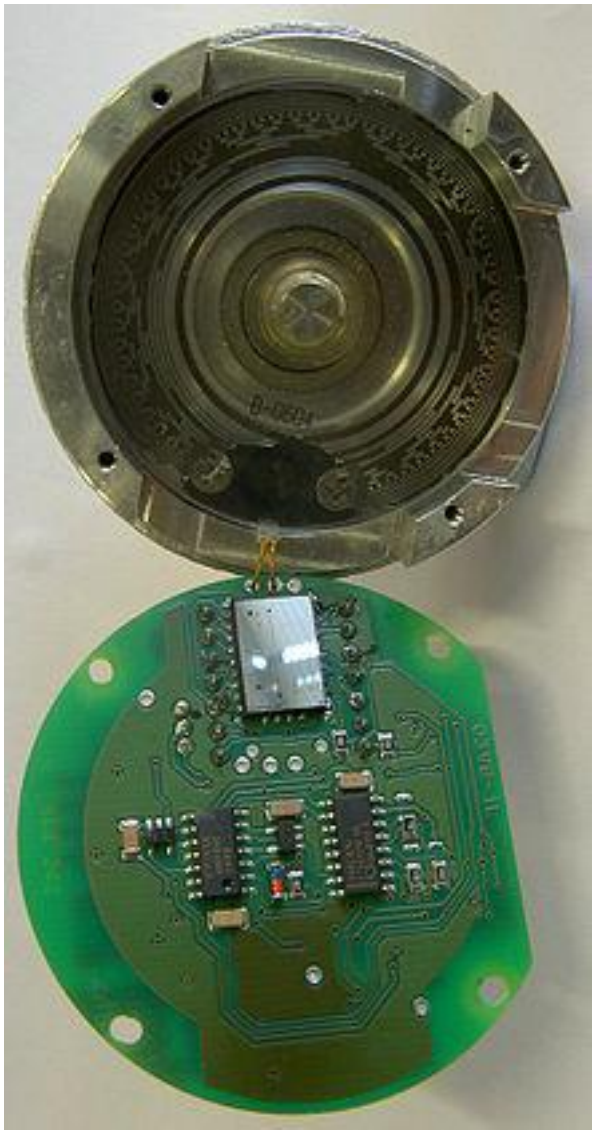
Binário	Gray
000	000
001	001
010	011
011	010
100	110
101	111
110	101
111	100

Código Gray – ilustração



Binário	Gray
000	000
001	001
010	011
011	010
100	110
101	111
110	101
111	100

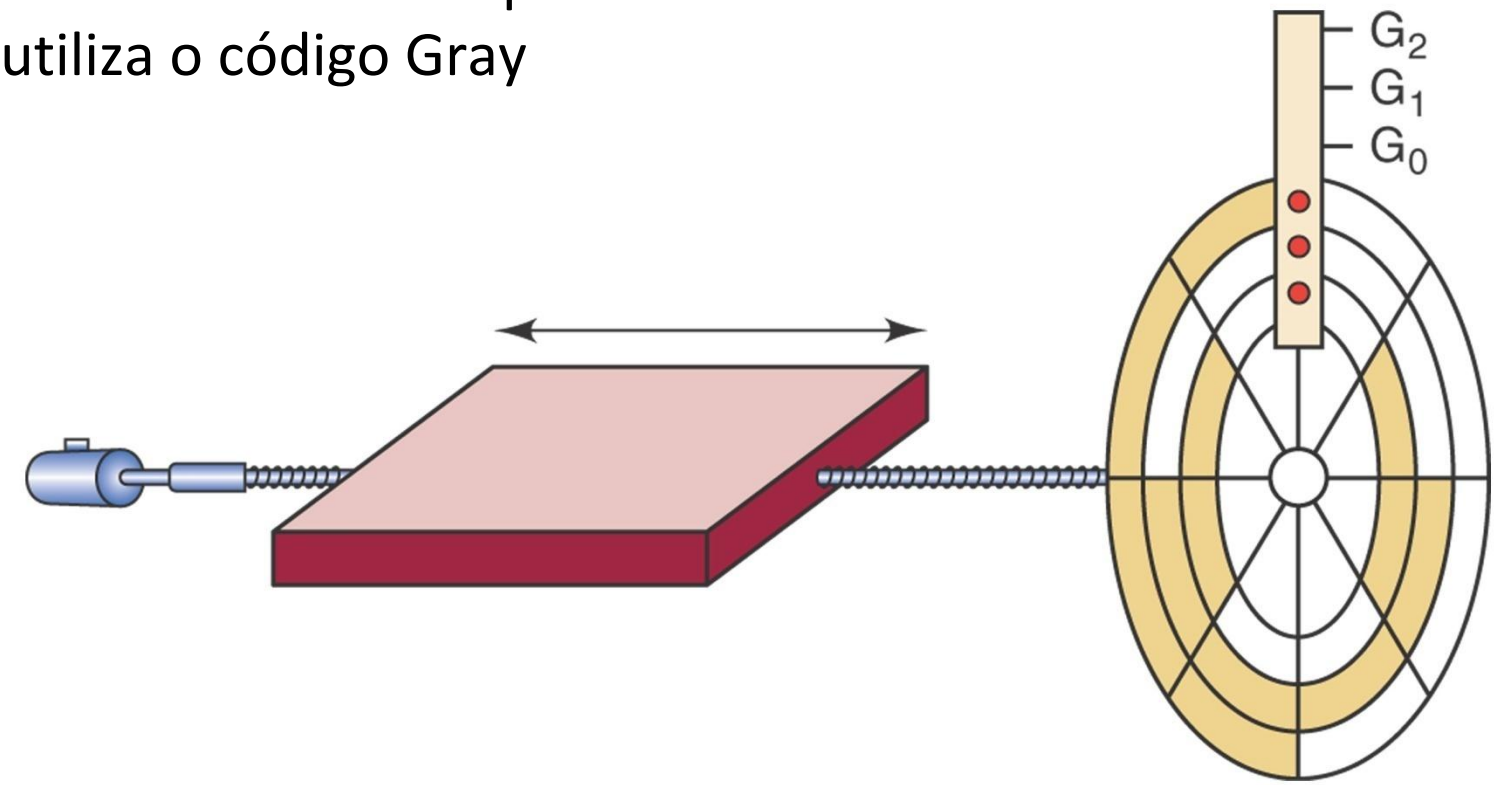
Encoder absoluto que utiliza o código Gray



Binário	Gray
000	000
001	001
010	011
011	010
100	110
101	111
110	101
111	100

Encoder absoluto que utiliza o código Gray

Binário	Gray
000	000
001	001
010	011
011	010
100	110
101	111
110	101
111	100



2.4.2 O código *Gray*

- **Conversão de binário para gray**

Binário			Gray		
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

2.4.2 O código *Gray*

$$g_i = 1, \quad \text{se} \quad b_i \neq b_{i+1},$$

$$g_i = 0, \quad \text{se} \quad b_i = b_{i+1},$$

Binário				Gray		
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

UFSJ
UNIVERSIDADE FEDERAL
DE SÃO JOÃO DEL-REI

O código Gray

$$g_i = 1, \quad \text{se} \quad b_i \neq b_{i+1},$$

$$g_i = 0, \quad \text{se} \quad b_i = b_{i+1},$$

Binário				Gray			
b_3	b_2	b_1	b_0	g_3	g_2	g_1	g_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

2.4.2 O código *Gray*

- **Exercício: Converta para o equivalente em código Gray.**

a) 100101010101
b) 000111010101
c) 111110111101
d) 101
e) 11001100

2.4.3 O código ASCII (*American Standard Code for Information Interchange*)

- **Comunicação**
  - **PC – PC**
  - **PC – Equipamento**
  - **Usuário – PC**

33	21	0010 0001	!	Exclamation point
34	22	0010 0010	"	Quotation mark
35	23	0010 0011	#	Number sign, octothorp, pound
36	24	0010 0100	$	Dollar sign
37	25	0010 0101	%	Percent
38	26	0010 0110	&	Ampersand
39	27	0010 0111	’	Apostrophe, prime
40	28	0010 1000	(	Left parenthesis
41	29	0010 1001	)	Right parenthesis
42	2A	0010 1010	*	Asterisk, ‘star’
43	2B	0010 1011	+	Plus sign
44	2C	0010 1100	,	Comma
45	2D	0010 1101	-	Hyphen, minus sign
46	2E	0010 1110	.	Period, decimal Point, ‘dot’
47	2F	0010 1111	/	Slash, virgule
48	30	0011 0000	0	0
49	31	0011 0001	1	1
50	32	0011 0010	2	2
51	33	0011 0011	3	3
52	34	0011 0100	4	4

52	34	0011 0100	4	4
53	35	0011 0101	5	5
54	36	0011 0110	6	6
55	37	0011 0111	7	7
56	38	0011 1000	8	8
57	39	0011 1001	9	9
58	3A	0011 1010	:	Colon
59	3B	0011 1011	;	Semicolon
60	3C	0011 1100	<	Less-than sign
61	3D	0011 1101	=	Equals sign
62	3E	0011 1110	>	Greater-than sign
63	3F	0011 1111	?	Question mark
64	40	0100 0000	@	At sign
65	41	0100 0001	A	A

66	42	0100 0010	B	B
67	43	0100 0011	C	C
68	44	0100 0100	D	D
69	45	0100 0101	E	E
70	46	0100 0110	F	F
71	47	0100 0111	G	G
72	48	0100 1000	H	H
73	49	0100 1001	I	I
74	4A	0100 1010	J	J
75	4B	0100 1011	K	K
76	4C	0100 1100	L	L
77	4D	0100 1101	M	M
78	4E	0100 1110	N	N
79	4F	0100 1111	O	O
80	50	0101 0000	P	P
81	51	0101 0001	Q	Q
82	52	0101 0010	R	R

83	53	0101 0011	S	S
84	54	0101 0100	T	T
85	55	0101 0101	U	U
86	56	0101 0110	V	V
87	57	0101 0111	W	W
88	58	0101 1000	X	X
89	59	0101 1001	Y	Y
90	5A	0101 1010	Z	Z
91	5B	0101 1011	[	Opening bracket
92	5C	0101 1100	\	Reverse slash
93	5D	0101 1101	]	Closing bracket
94	5E	0101 1110	^	Circumflex, caret
95	5F	0101 1111	_	Underline, underscore
96	60	0110 0000	`	Grave accent

97	61	0110 0001	a	a
98	62	0110 0010	b	b
99	63	0110 0011	c	c
100	64	0110 0100	d	d
101	65	0110 0101	e	e
102	66	0110 0110	f	f
103	67	0110 0111	g	g
104	68	0110 1000	h	h
105	69	0110 1001	i	i
106	6A	0110 1010	j	j
107	6B	0110 1011	k	k
108	6C	0110 1100	l	l
109	6D	0110 1101	m	m
110	6E	0110 1110	n	n
111	6F	0110 1111	o	o
112	70	0111 0000	p	p
113	71	0111 0001	q	q
114	72	0111 0010	r	r

2.4.3 O código ASCII (*American Standard Code for Information Interchange*)

- **Exemplo**
  - **Qual a representação da palavra "ALUNO" em código ASCII?**

2.4.3 O código ASCII (*American Standard Code for Information Interchange*)

- **Exemplo**
  - **Qual a representação da palavra "ALUNO" em código ASCII?**

01000001 01001100 01010101 01001110 01001111

2.4.3 O código ASCII (*American Standard Code for Information Interchange*)

- **Exercício**
  - **Qual é a mensagem?**

01010011 01001111 01000011 01001111 01010010 01010010 01001111 00100001

2.4.4 Outros Códigos

- **ISO-8859-1**
  - **Extensão ASCII**
- **EBCDIC**
  - **Extensão BCD**
- **UNICODE**
- **ISO-10646**
- ***DISPLAY* DE 7 SEGMENTOS**

2.4.4 Outros Códigos

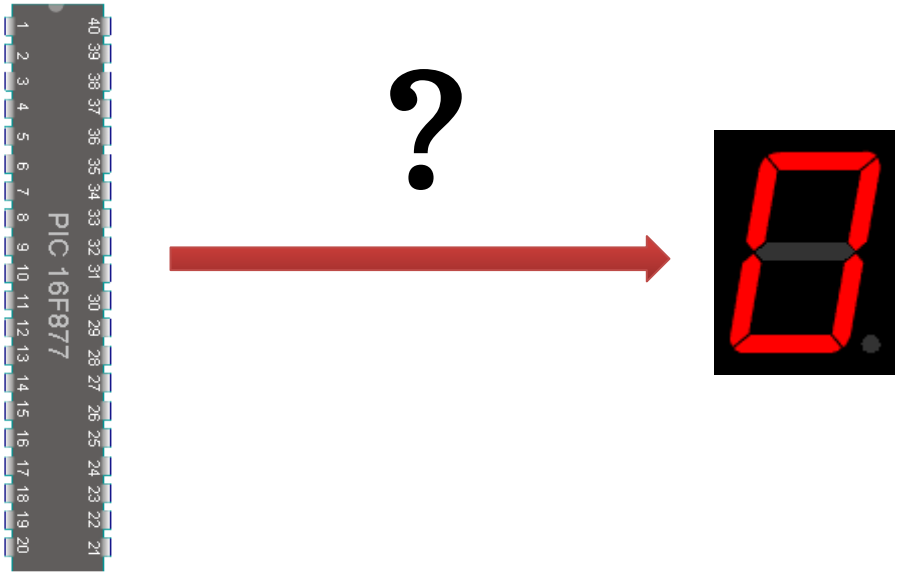
- **ISO-8859-1**
  - **Extensão ASCII**
- **EBCDIC**
  - **Extensão BCD**
- **UNICODE**
- **ISO-10646**

} Vários idiomas

- ***DISPLAY* DE 7 SEGMENTOS**

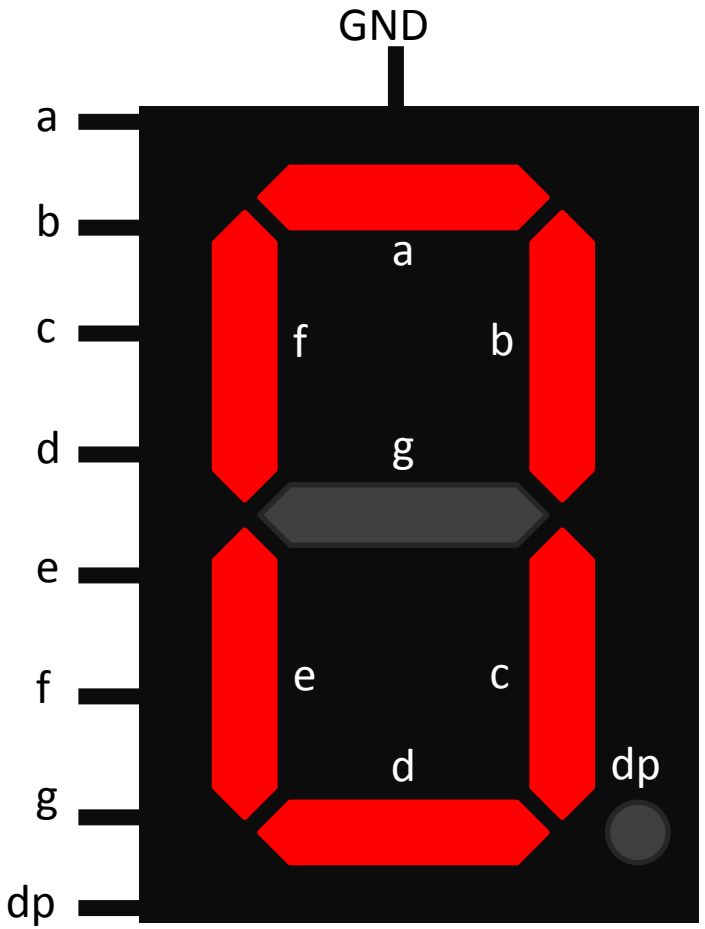
2.4.4 Outros Códigos

- **Display de 7 Segmentos**



2.4.4 Outros Códigos

- **Display de 7 Segmentos**

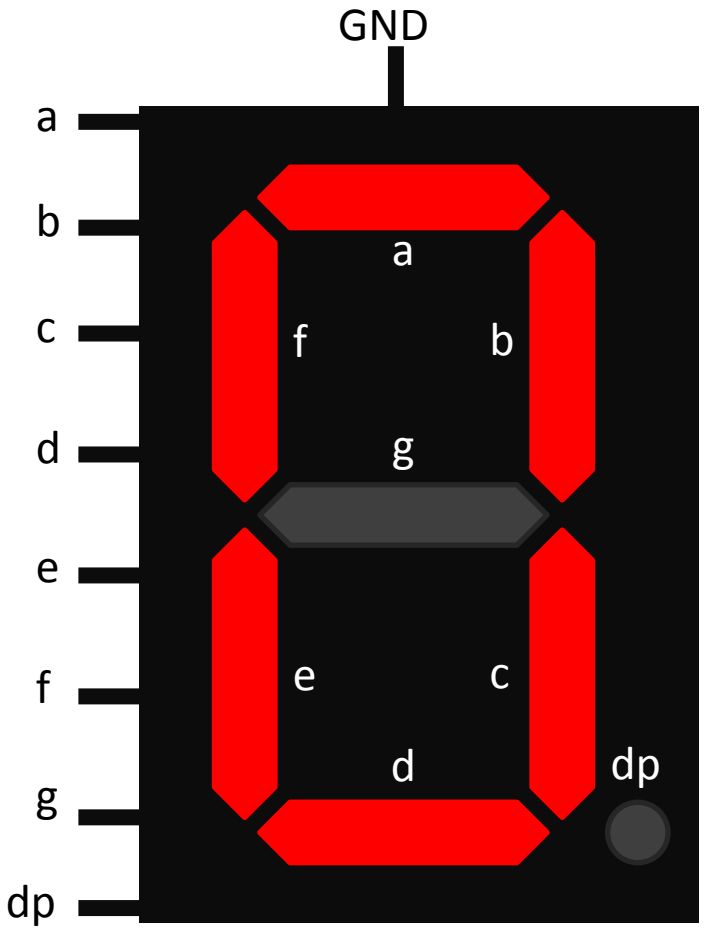


2.4.4 Outros Códigos

- **Display de 7 Segmentos**

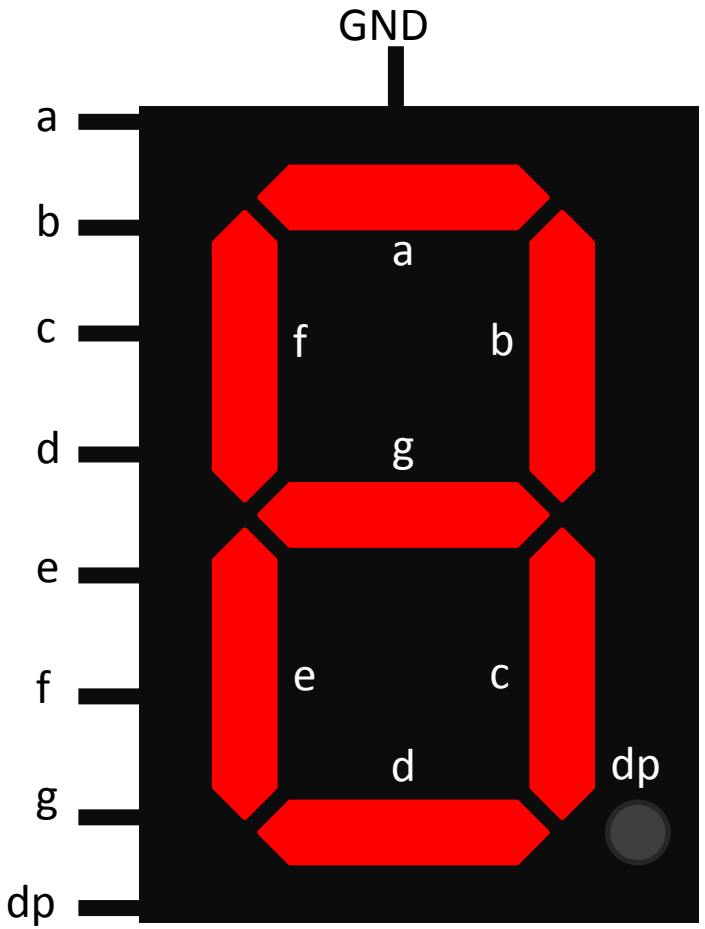
!

Necessidade de 8 bits



2.4.4 Outros Códigos

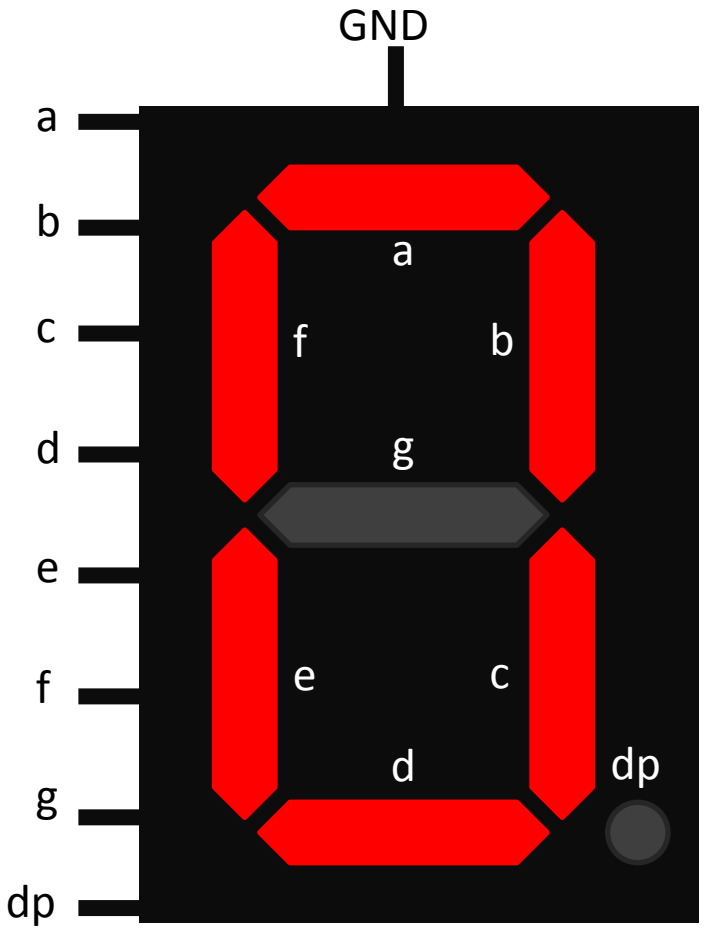
- **Display de 7 Segmentos**



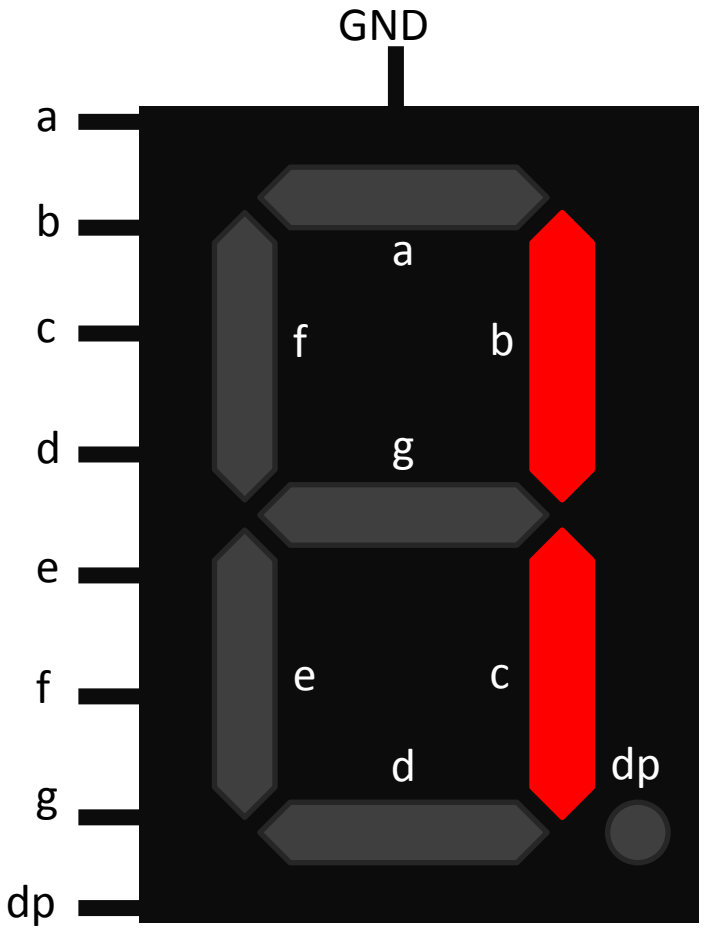
!

Ao conjunto de 8 bits dá-se o nome de ***Byte***

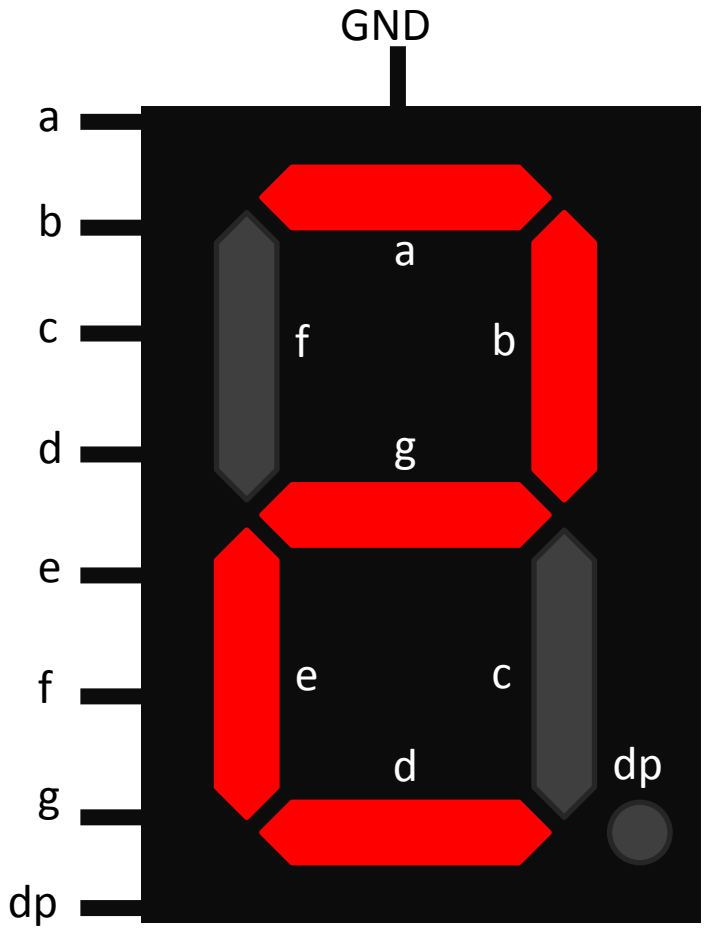
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	



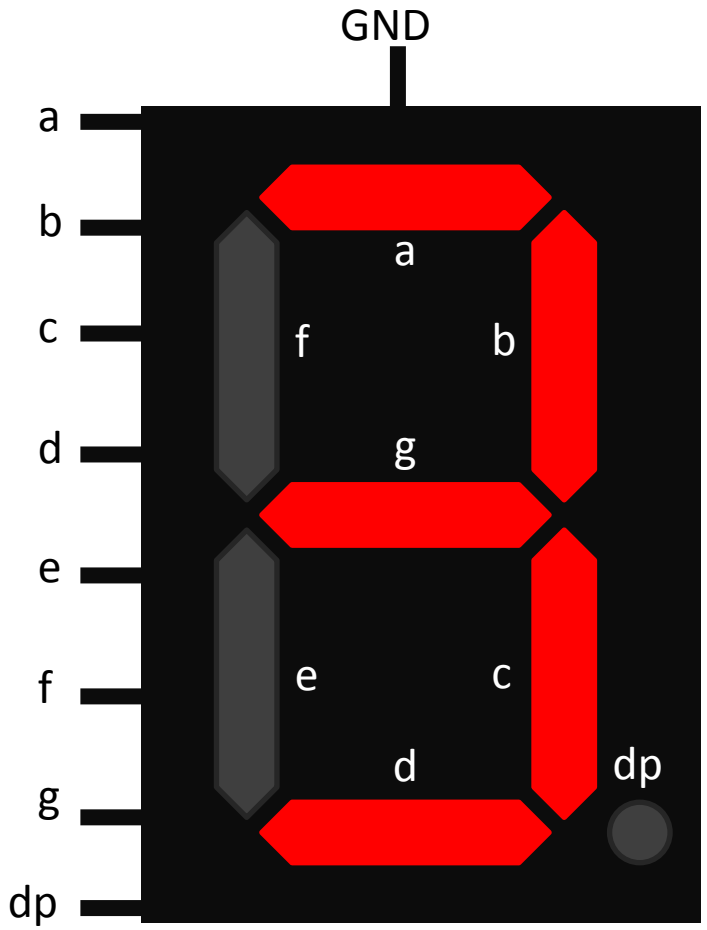
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



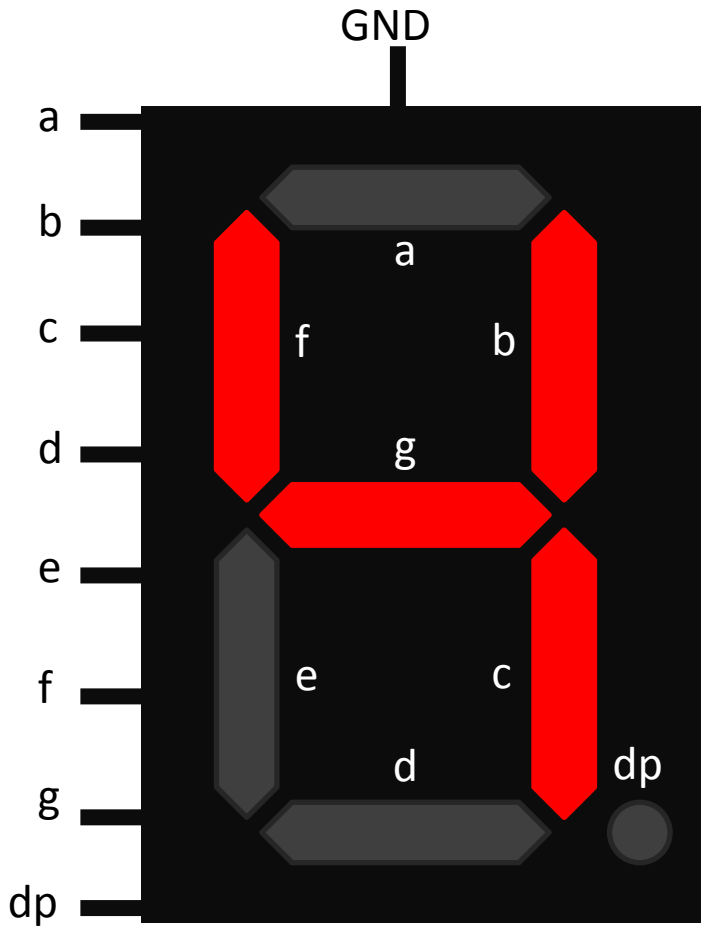
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



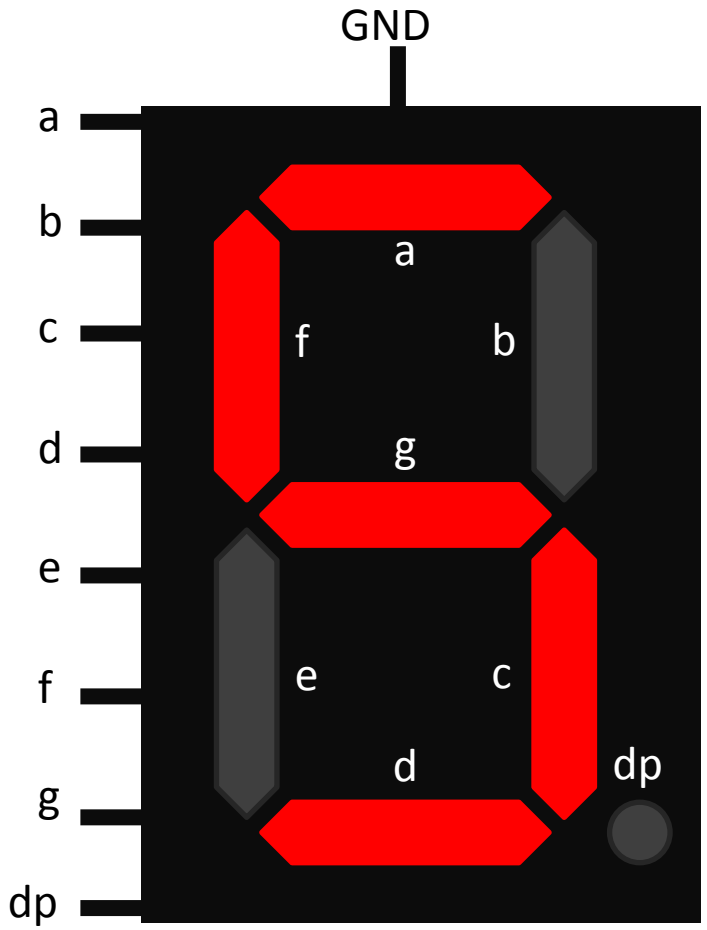
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



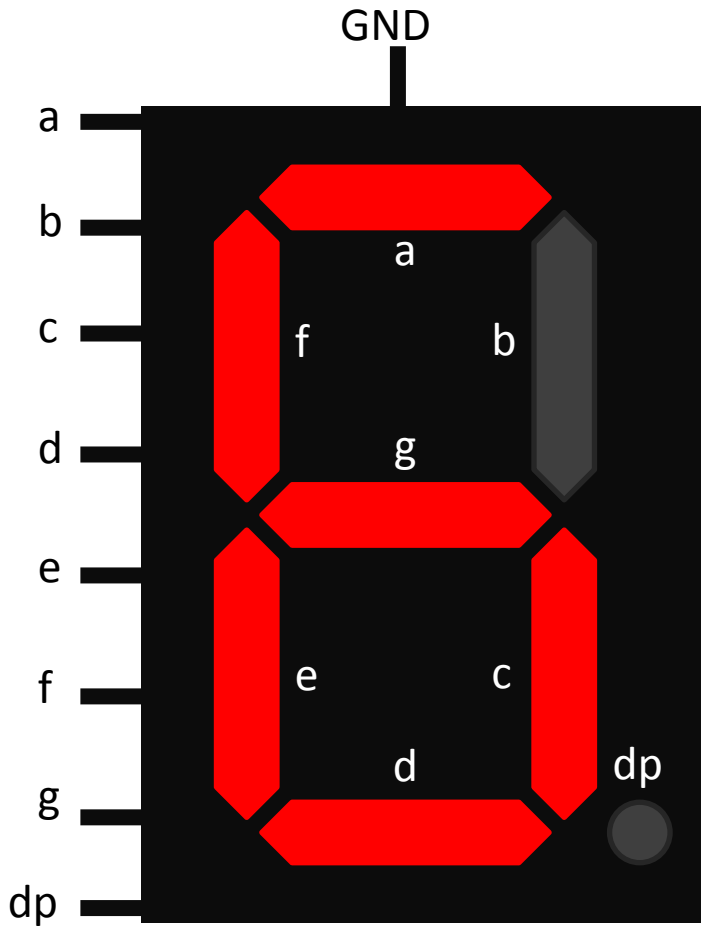
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



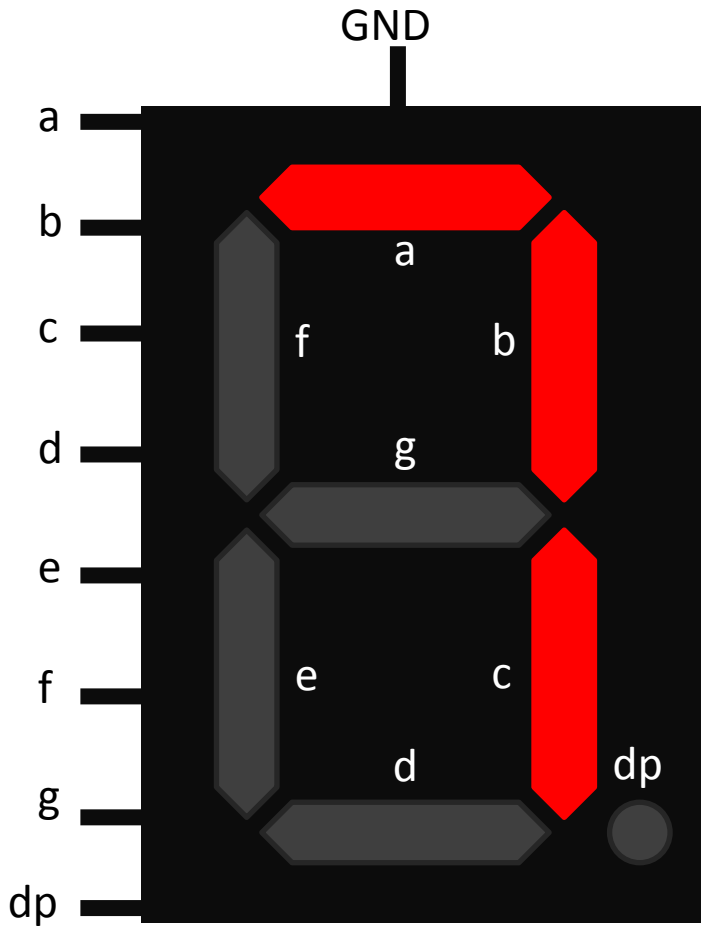
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



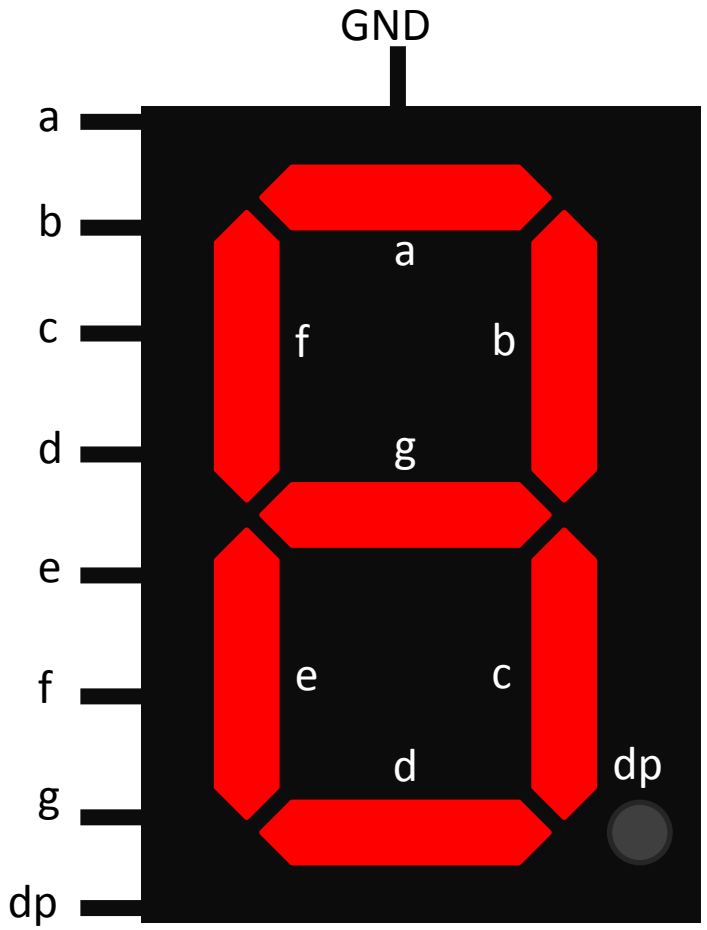
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



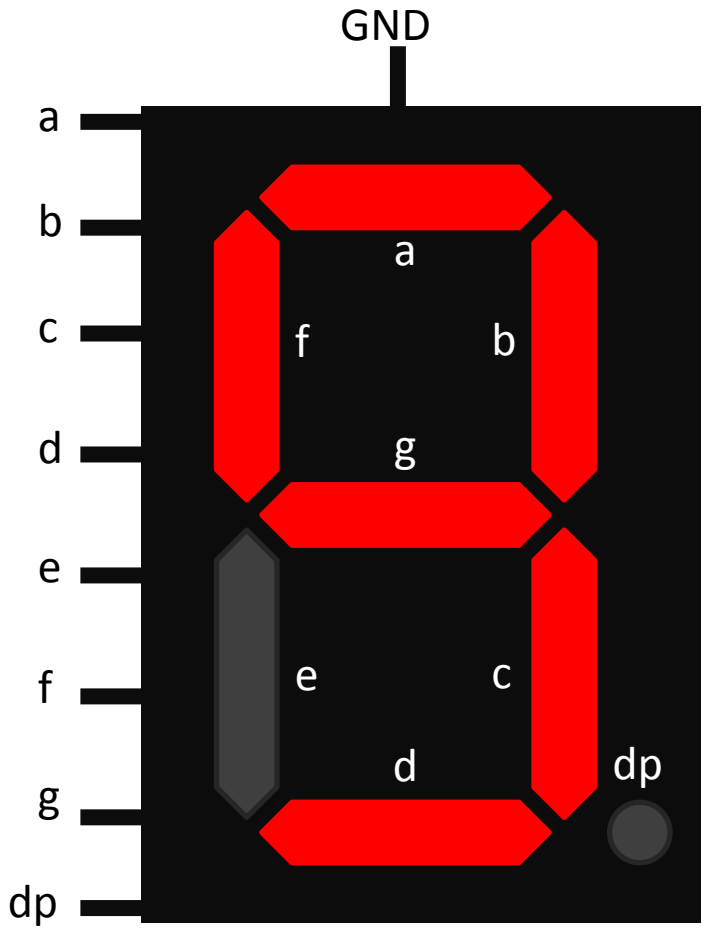
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



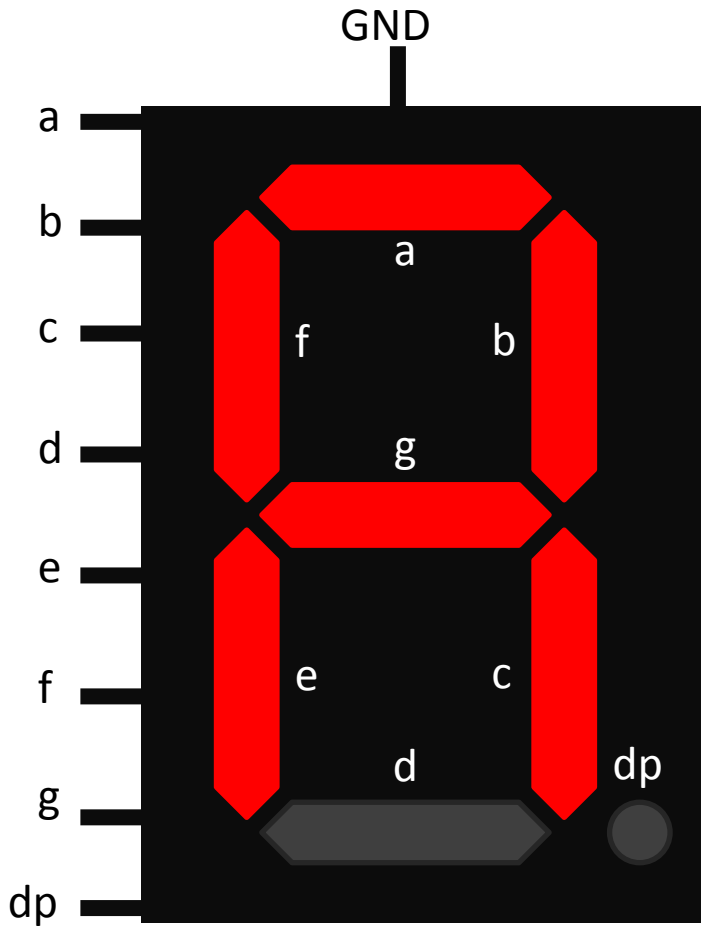
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



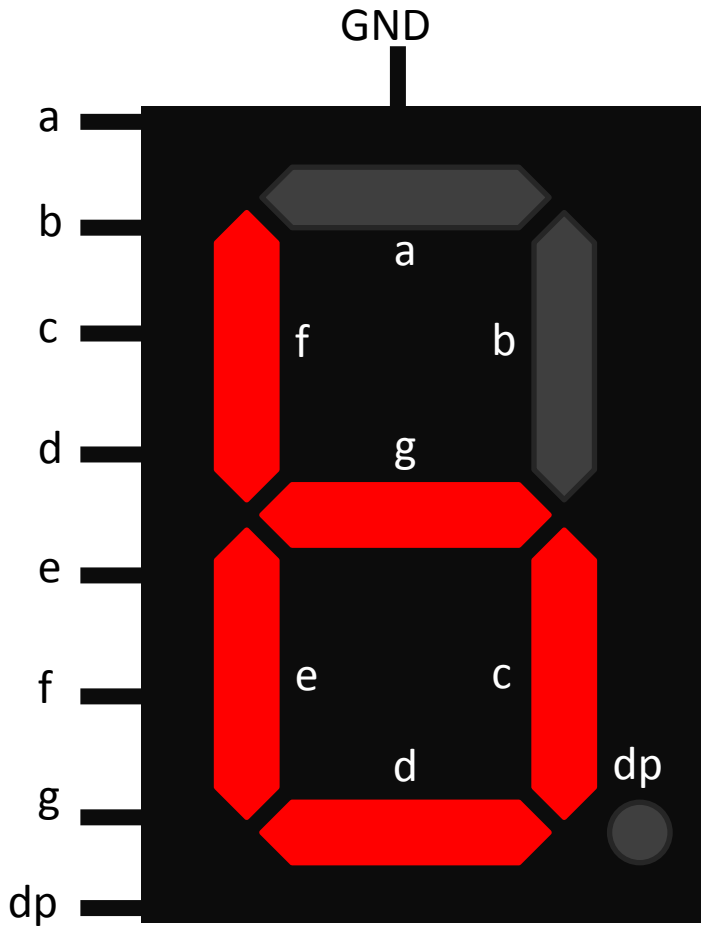
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



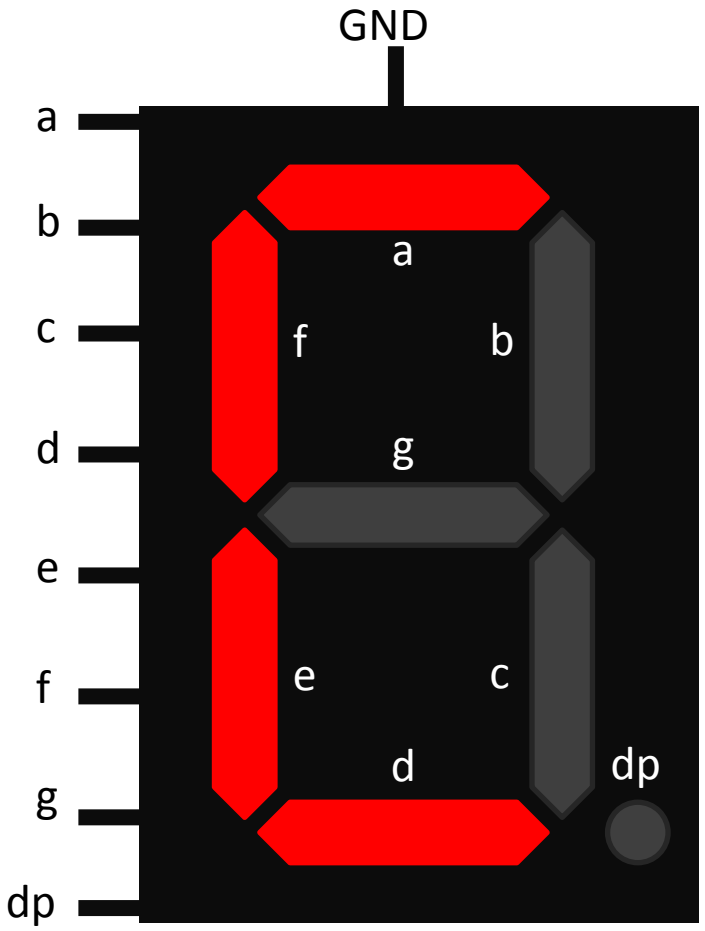
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



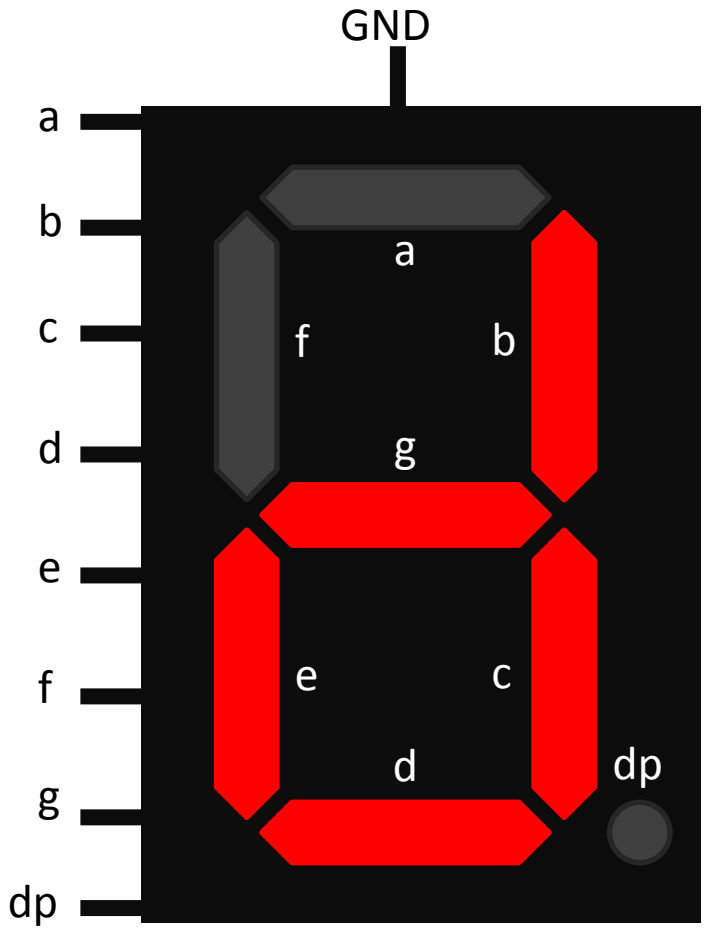
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



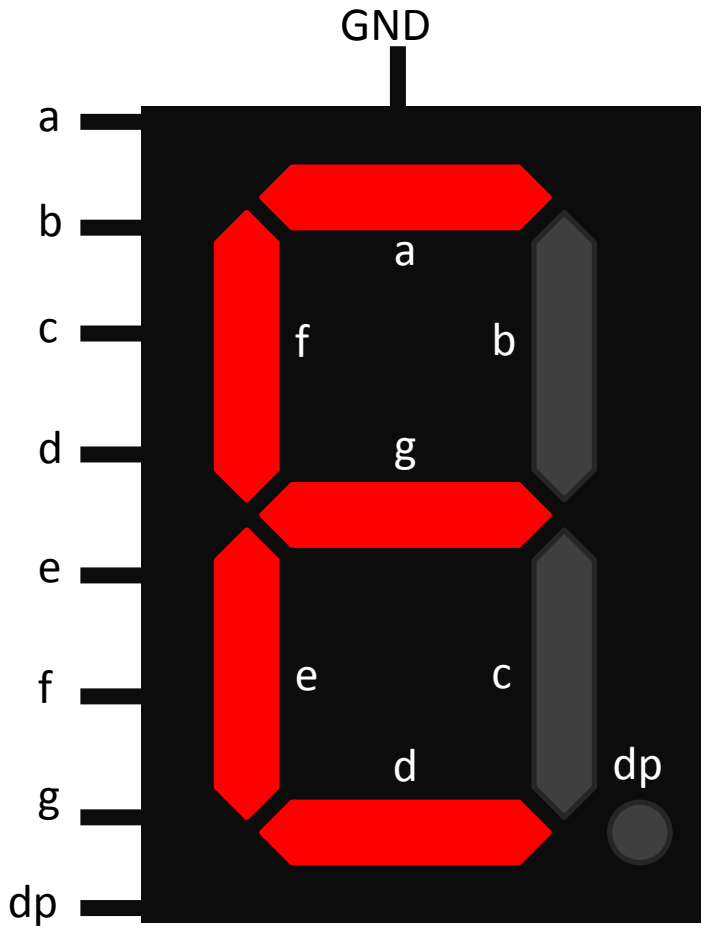
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



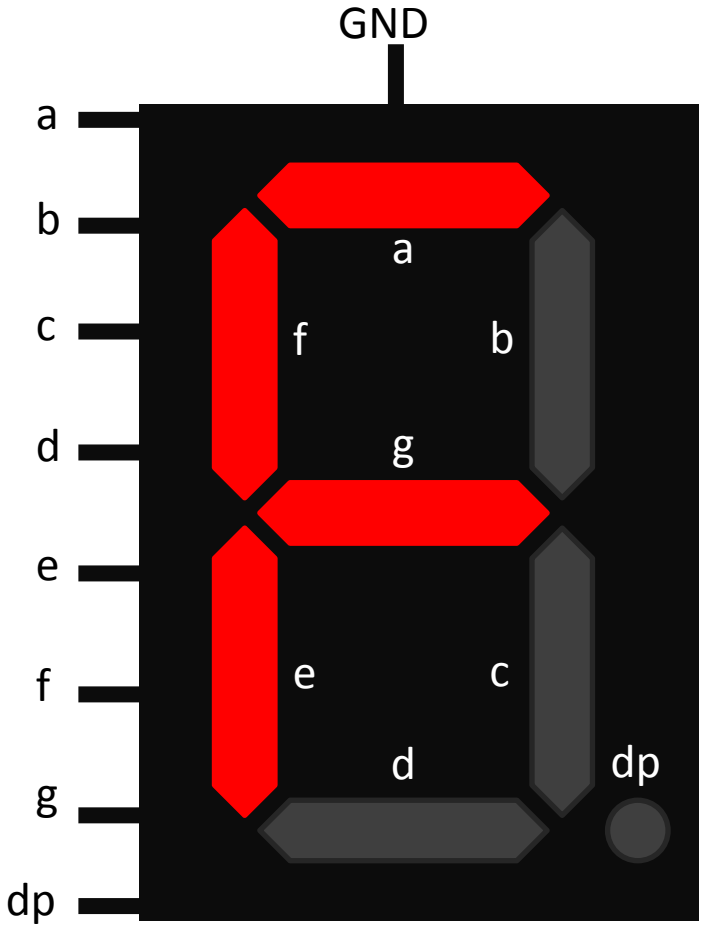
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



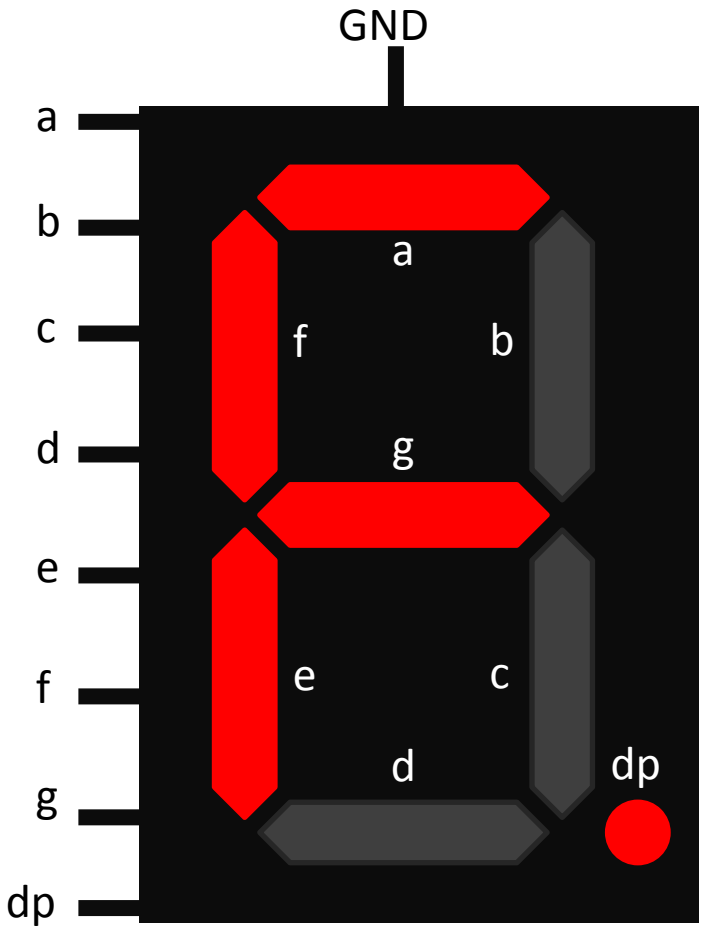
	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0



	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	0
1	0	1	1	0	0	0	0	0
2	1	1	0	1	1	0	1	0
3	1	1	1	1	0	0	1	0
4	0	1	1	0	0	1	1	0
5	1	0	1	1	0	1	1	0
6	1	0	1	1	1	1	1	0
7	1	1	1	0	0	0	0	0
8	1	1	1	1	1	1	1	0
9	1	1	1	1	0	1	1	0
A	1	1	1	0	1	1	1	0
B	0	0	1	1	1	1	1	0
C	1	0	0	1	1	1	0	0
D	1	0	0	1	1	1	1	0
E	1	0	0	1	1	1	1	0
F	1	0	0	0	1	1	1	0

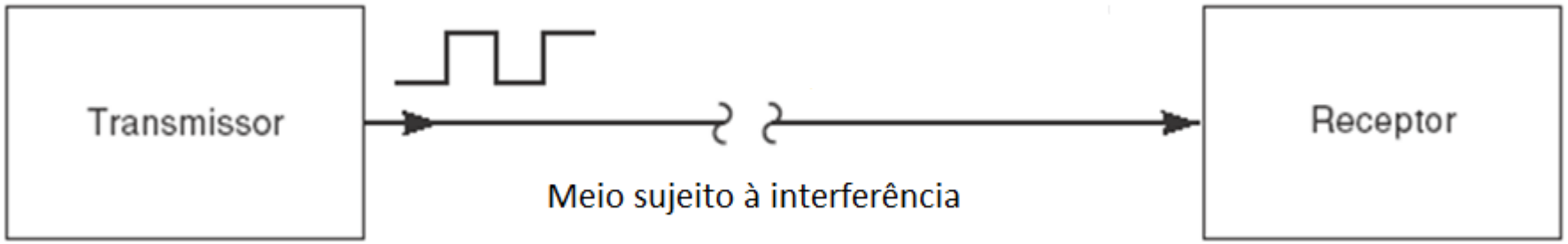


	a	b	c	d	e	f	g	DP
0	1	1	1	1	1	1	0	1
1	0	1	1	0	0	0	0	1
2	1	1	0	1	1	0	1	1
3	1	1	1	1	0	0	1	1
4	0	1	1	0	0	1	1	1
5	1	0	1	1	0	1	1	1
6	1	0	1	1	1	1	1	1
7	1	1	1	0	0	0	0	1
8	1	1	1	1	1	1	1	1
9	1	1	1	1	0	1	1	1
A	1	1	1	0	1	1	1	1
B	0	0	1	1	1	1	1	1
C	1	0	0	1	1	1	0	1
D	1	0	0	1	1	1	1	1
E	1	0	0	1	1	1	1	1
F	1	0	0	0	1	1	1	1



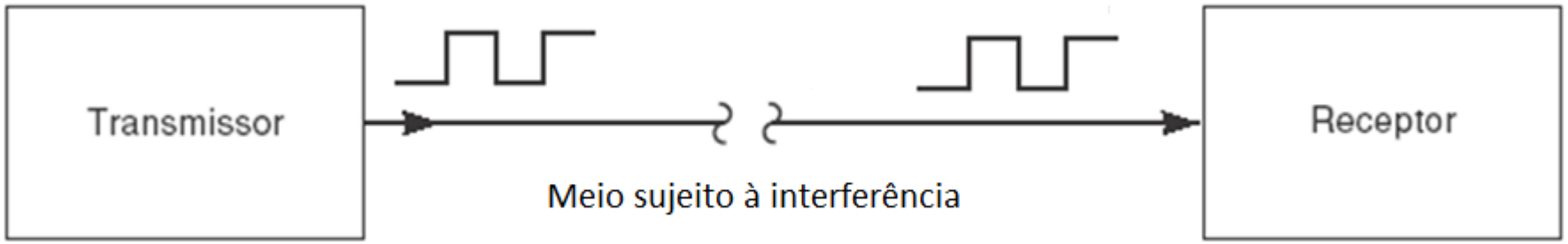
2.5 Detecção de Erros

- **Necessidade de verificação de integridade da informação!**



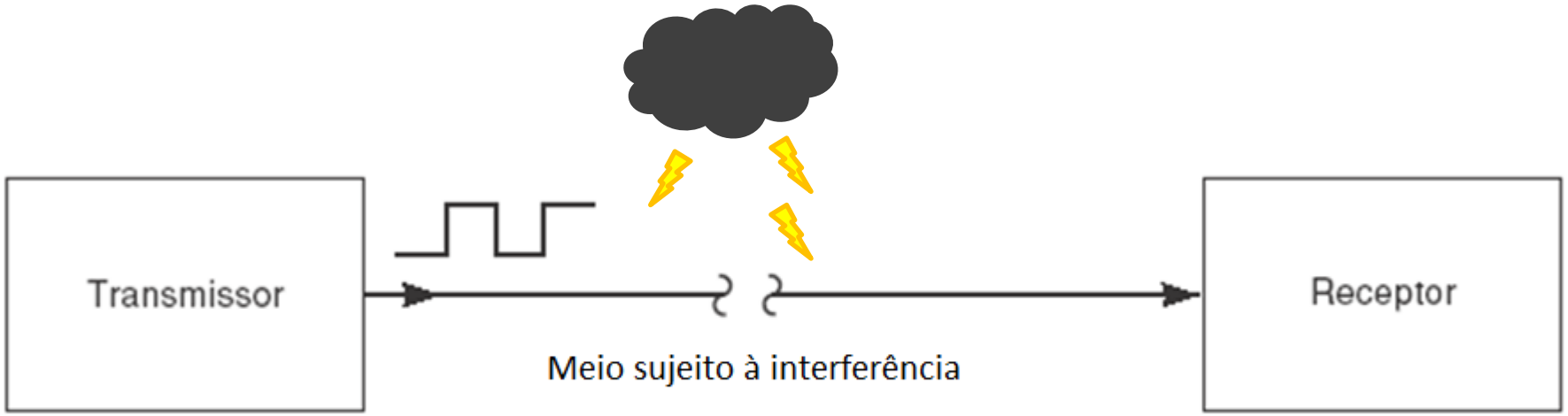
2.5 Detecção de Erros

- **Necessidade de verificação de integridade da informação!**



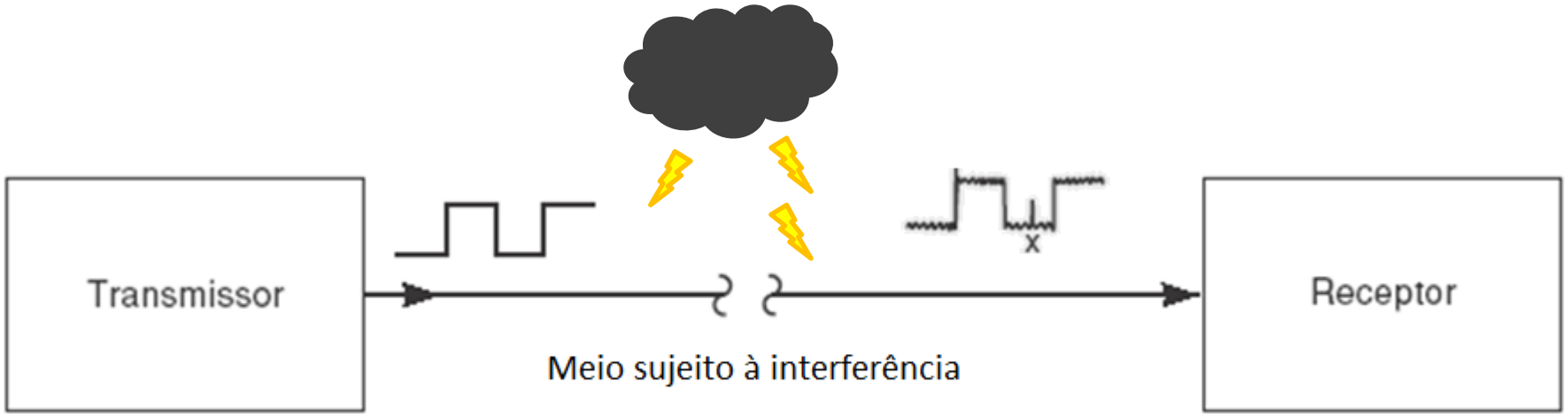
2.5 Detecção de Erros

- **Necessidade de verificação de integridade da informação!**



2.5 Detecção de Erros

- **Necessidade de verificação de integridade da informação!**



2.5 Detecção de Erros

- **Necessidade de verificação de integridade da informação!**
  - **Como diminuir a probabilidade de cometer um erro devido à uma mensagem errada?**

2.5 Detecção de Erros

- **Necessidade de verificação de integridade da informação!**
  - **Como diminuir a probabilidade de cometer um erro devido à uma mensagem errada?**
    - **Adicione redundância!**
    - **“Repita por favor!”**

2.5 Detecção de Erros

2.5.1 Bit de paridade!

- **Consiste em anexar um bit à informação a ser transmitida!**

2.5 Detecção de Erros

2.5.1 Bit de paridade!

- Consiste em anexar um bit à informação a ser transmitida!
- Paridade Par

2.5 Detecção de Erros

2.5.1 Bit de paridade!

- Consiste em anexar um bit à informação a ser transmitida!

- Paridade Par
  - 0 → se o número total de 1's na informação a ser transmitida for par;
  - 1 → se o número total de 1's na informação a ser transmitida for ímpar.

2.5 Detecção de Erros

2.5.1 Bit de paridade!

- **Paridade Par**
  - **0 → se o número total de 1's na informação a ser transmitida for par;**
  - **1 → se o número total de 1's na informação a ser transmitida for ímpar**

Exemplo: Anexar o bit de paridade par aos números:

- **10110110**
- **110101**

2.5 Detecção de Erros

2.5.1 Bit de paridade!

- **Paridade Par**
  - **0 → se o número total de 1's na informação a ser transmitida for par;**
  - **1 → se o número total de 1's na informação a ser transmitida for ímpar**

Exemplo: Anexar o bit de paridade par aos números:

- **10110110_2 → 110110110**
- **110101_2 → 0110101**

2.5 Detecção de Erros

2.5.1 Bit de paridade!

- Consiste em anexar um bit à informação a ser transmitida!
- Paridade Ímpar

2.5 Detecção de Erros

2.5.1 Bit de paridade!

- Consiste em anexar um bit à informação a ser transmitida!

- Paridade Ímpar
  - 1 → se o número total de 1's na informação a ser transmitida for par;
  - 0 → se o número total de 1's na informação a ser transmitida for ímpar.

2.5 Detecção de Erros

2.5.1 Bit de paridade!

- Consiste em anexar um bit à informação a ser transmitida!
- Exemplo: Anexar o bit de paridade ímpar aos números:
  - 10110110_2
  - 110101_2

2.5 Detecção de Erros

2.5.1 Bit de paridade!

- Consiste em anexar um bit à informação a ser transmitida!
- Exemplo: Anexar o bit de paridade ímpar aos números:
  - $10110110_2 \rightarrow 010110110$
  - $110101_2 \rightarrow 1110101$

2.5 Detecção de Erros

- **Exercício**

Adicione o bit de paridade par a cada uma das informações abaixo:

- 10110110_2
- 01100111_2
- 10110001_2
- 01111110 (código ASCII)

2.5 Detecção de Erros

- **Exercício**

  As seguintes informações (ASCII de 8 bits) foram recebidas por um sistema que utiliza paridade ímpar. Qual é a mensagem?

001000111 001001001 001010010 001000011
101010101 001001001 001010100 001001111
101010011

2.5 Detecção de Erros

- **Exercício**

Em um determinado sistema digital, os números decimais de 000 até 999 são representados no código BCD. Um bit de paridade ímpar também é incluído no fim de cada grupo de bits do código. Para cada grupo a seguir determine qual apresenta um erro e, se existir, qual tem definitivamente dois erros! (suponha que não mais do que dois erros ocorreram em cada grupo!)

a) **1001010110000**

b) **0100011101100**

c) **0111110000011**

d) **1000011000101**

2.5 Detecção de Erros

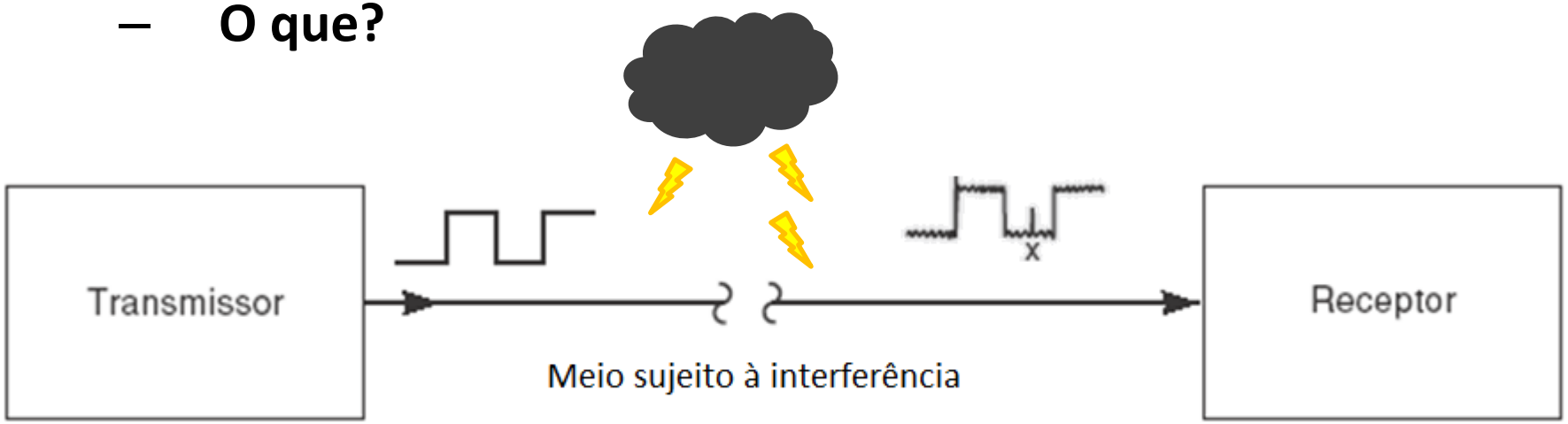
2.5.2 Método da repetição

- O que?

2.5 Detecção de Erros

2.5.2 Método da repetição

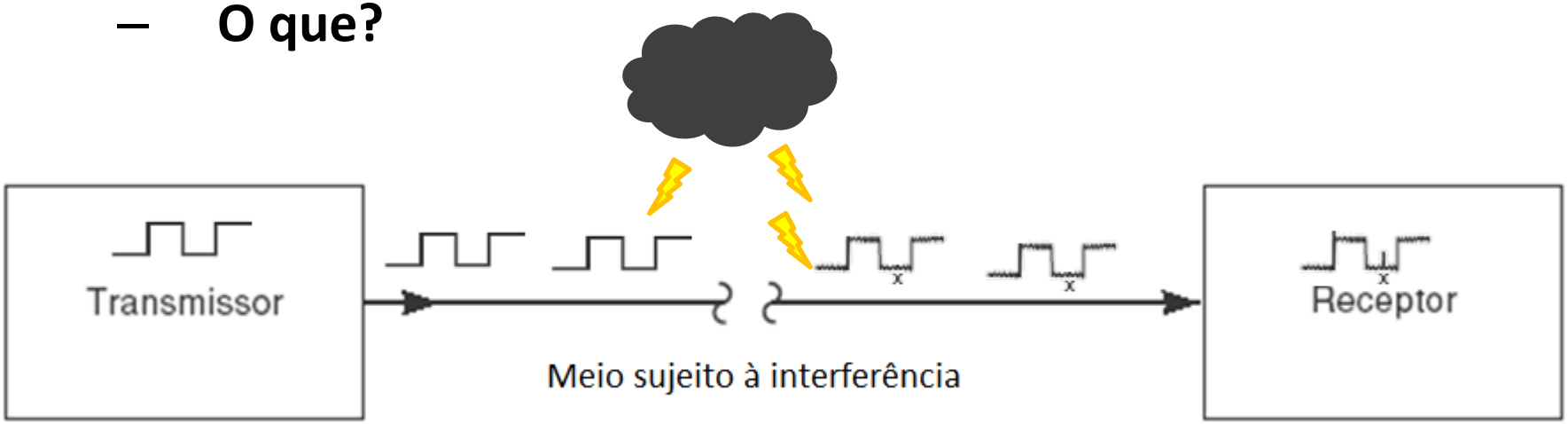
– **O que?**



2.5 Detecção de Erros

2.5.2 Método da repetição

– **O que?**



2.5 Detecção de Erros

2.5.2 Método da repetição

- O que?
- Cada *bit* é repetido durante a transmissão "*n*" vezes
- Exemplo: Transmitir 0 ou 1 usando repetição de 3.
  - 0 →
  - 1 →

2.5 Detecção de Erros

2.5.2 Método da repetição

- O que?
- Cada *bit* é repetido durante a transmissão "*n*" vezes
- Exemplo: Transmitir 0 ou 1 usando repetição de 3.
  - $0 \rightarrow 000$;
  - $1 \rightarrow 111$;

2.5 Detecção de Erros

2.5.2 Método da repetição

- O que?
- Cada *bit* é repetido durante a transmissão "*n*" vezes
- Exemplo: Transmitir 0 ou 1 usando repetição de 3.
  - 0 $\longrightarrow$ 000; 001; 010; 100;
  - 1 $\longrightarrow$ 111; 110; 101; 011;

2.5 Detecção de Erros

2.5.2 Método da repetição

- **Exemplo: transmissão do caractere “N” em ASCII (8 bits)**

N = 01001110

Dado recebido: 010011100100111001001110

Usando quebra em blocos de 4 bits

Dado recebido: 010001000100111011101110

2.5 Detecção de Erros

2.5.2 Método da repetição

- **Vantagens:**
  - **Detecção e possibilidade de correção.**

2.5 Detecção de Erros

2.5.2 Método da repetição

- **Vantagens:**
  - **Detecção e possibilidade de correção.**
- **Desvantagens:**
  - **Ineficiente;**
  - **Resulta em grande número de bits a ser transmitido;**

2.5 Detecção de Erros

2.5.2 Método da repetição

Exercício

Como seria a sequência a ser transmitida para o número 734_{10} utilizando o código BCD e um sistema de transmissão com repetição 3?

2.5 Detecção de Erros

2.5.3 Outros métodos

- Método da verificação cíclica redundante;
- Método do código de Hamming.

Tópicos adicionais

- **Converta para binário**
  - $10{,}75_{10}$
  - $13{,}375_{10}$
  - $17{,}6_{8}$
  - $374{,}26_{8}$
  - $3F{,}D_{16}$
  - $17E{,}F6_{16}$

Tópicos adicionais

- **Converta para octal**
  - $73{,}75_{10}$
  - $1110100{,}0100111_{2}$
- **Converta para hexadecimal**
  - $82{,}25_{10}$
  - $1011001110{,}011011101_{2}$