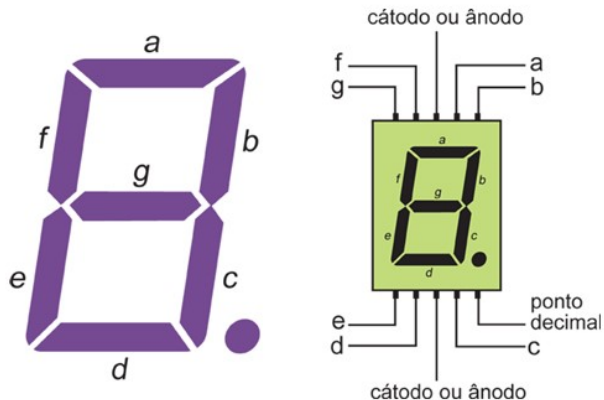


# 7 SEGMENTOS



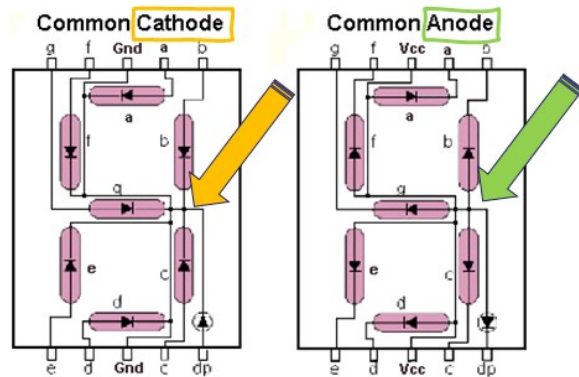
[http://www.dreaminc.com.br/sala\\_de\\_aula/15a-display-7-segmentos/](http://www.dreaminc.com.br/sala_de_aula/15a-display-7-segmentos/)

© Prof. Eng<sup>o</sup> esp Luiz Antonio Vargas Pinto  
[www.vargasp.com](http://www.vargasp.com)

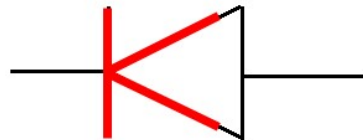
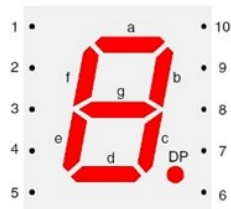
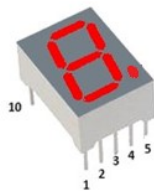
Revisado em março 2025



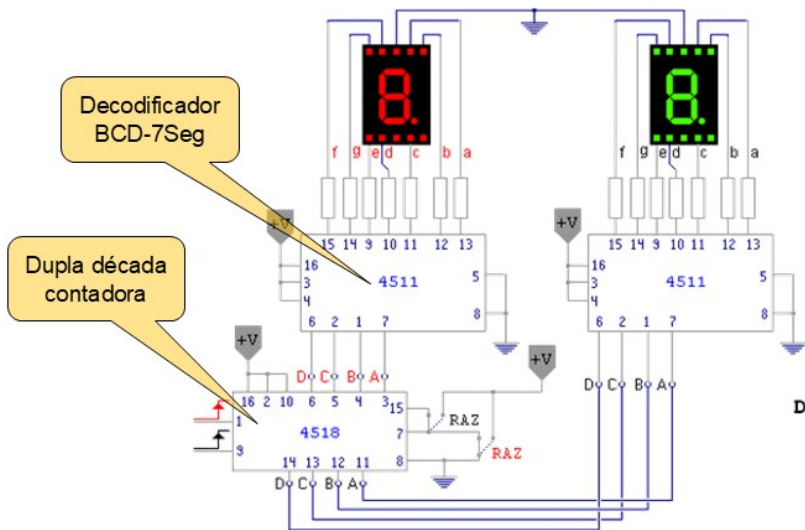
# Modos



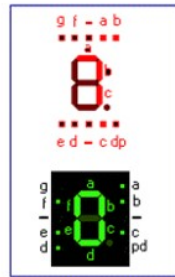
<https://sites.google.com/site/netaula/blog/contadorbcdcomdisplayde7segmentos>



# Exemplo de aplicação

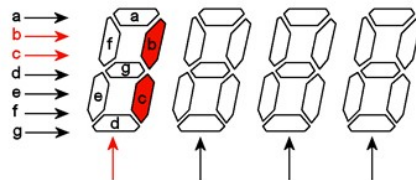


<https://www.electronica-pt.com/contadores-binarios>



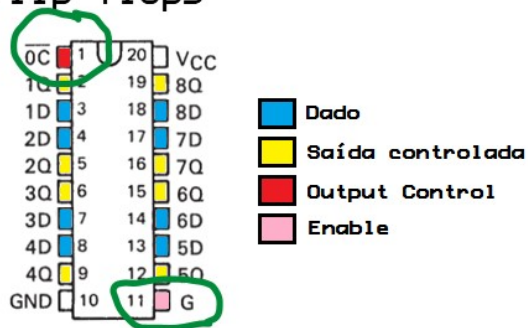
Display 7 segmentos  
cátodo comum

Por exemplo:  
**1000** será  
convertido  
para **8**



# 74LS373

Octal D-type transparent latches and edge-triggered flip-flops



Output Control	Enable G	D	Output
L	H	H	H
L	H	L	L
L	L	X	Q <sub>0</sub>
H	X	X	Z



4 displays  
de 7Seg

# Princípio

Mensagem

T E C H

TECHNOLOGIC -

Vetor  
(software)

0 1 2 3 4 5 6 7 8 9 10 11

Índices

tEch  
Echn  
chno  
hnoL  
noLo  
oLo9  
Lo9,  
o9 ic  
9 ic\_  
ic\_t  
c\_tE  
\_tEc

i	j	k	l
0	1	2	3
1	2	3	4
2	3	4	5
3	4	5	6
4	5	6	7
5	6	7	8
6	7	8	9
7	8	9	10
8	9	10	11
9	10	11	0
10	11	0	1
11	0	1	2
0	1	2	3

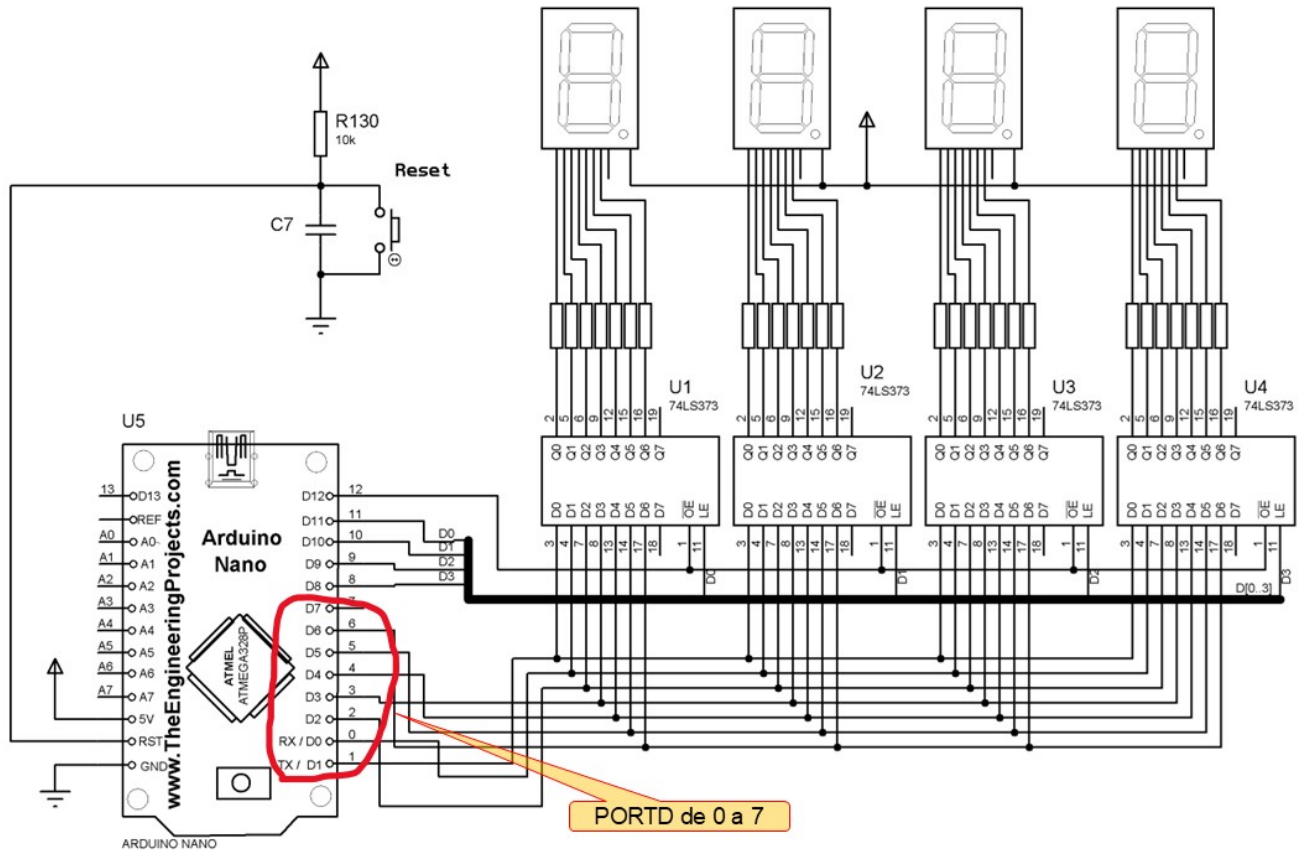
Aqui começa  
a repetir



# Aplicação

---

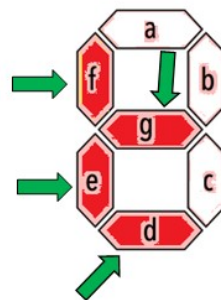




# O código

```
1 1/*-----  
2 Programa: 7Seg.ino  
3 Data      : 19/09/2022  
4 Autor     : Prof. Vargas  
5 Descrição:  
6 Uso do display de 7 Segmento padrão Anôdo Comum para escrever mensagem que  
7 se desloca para a esquerda usando controle de I/O com o CI 74LS373 e não  
8 multiplex. Escreve "technoLogic-" com 7 segmentos correndo o display.  
9 (Lembre-se que anôdo comum acende com 0)  
10      a      a = RD0  
11      ---    b = RD1  
12     f| g |b  c = RD2  
13      ---    d = RD3  
14     e|   |c  e = RD4  
15      ---    f = RD5  
16      d      g = RD6
```

17			0gfe.dcba	
18	t	= d e f g . . . . .	0000.0111	0x07
19	E	= a d e f g . . . . .	0000.0110	0x06
20	c	= g e d . . . . .	0010.0111	0x27
21	h	= f g e c . . . . .	0000.1011	0x0B
22	n	= g e c . . . . .	0010.1011	0x2B
23	o	= c d e g . . . . .	0010.0011	0x23
24	L	= f e g . . . . .	0100.0111	0x47
25	o	= c d e g . . . . .	0010.0011	0x23
27	i	= c . . . . .	0111.1011	0x7B
28	A	= b c d e f g . . . . .	0000.1000	0x27
29	-	= . . . . .	0111.0111	0x3F
30				



🌐 E como 0 acende o segmento de LED, o display é de Catôdo comum

31 Os ponteiros seguirão:

```
32     i   j   k   l
33     0   1   2   3  <---
34     1   2   3   4   |
35     2   3   4   5   |
36     3   4   5   6   |
37     4   5   6   7   |
38     5   6   7   8   |
39     6   7   8   9   |
40     7   8   9  10   |
41     8   9  10  11   |
42     9  10  11  12   |
43    10  11  12   0   |
44    11  12   0   1   |
45    12   0   1   2   ----
```

D0



D1



D2



D3



```
47 Saída Digital 12 controla G - Enable (Alta Impedância)
48 Saída Digital 11 controla I/O de D0 zero congela
49 Saída Digital 10 controla I/O de D1 zero congela
50 Saída Digital 09 controla I/O de D2 zero congela
51 Saída Digital 08 controla I/O de D3 zero congela
```

52

```

53 byte i, j, k, l;
54 byte const A[12] = {0x07, // T
55                     0x06, // E
56                     0x27, // c
57                     0x0B, // h
58                     0x2B, // n
59                     0x23, // o
60                     0x47, // L
61                     0x23, // o
62                     0x10, // g
63                     0x7B, // i
64                     0x27, // c
65                     0x3F // -
66                     };
67
68 const byte LE_D3 = 8;
69 const byte LE_D2 = 9;
70 const byte LE_D1 = 10;
71 const byte LE_D0 = 11;
72 const byte OE = 12;

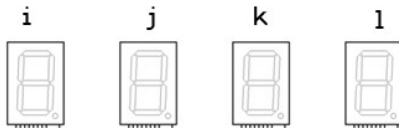
```

Vetor

Pino 11 do CI 74LS373 que controla a passagem de dados

```
73
74 void setup()
75 {
76     PORTD = 0b11111111; //garante displays apagados
77     DDRD = 0xFF; //PORTD é só saída
78     pinMode(LE_D3, OUTPUT); //Pinos de controle
79     pinMode(LE_D2, OUTPUT); ←
80     pinMode(LE_D1, OUTPUT);
81     pinMode(LE_D0, OUTPUT);
82     pinMode(OE, OUTPUT);
83     digitalWrite(OE, LOW); //Tira de alta impedância mas ainda está congelado
84 }
85
```

```
86 void loop()  
87 {  
88     i = 0;           //Ponteiros de caractere  
89     j = 1;  
90     k = 2;  
91     l = 3;  
92     do  
93     {  
  
121 } while (i < 12);  
122 }
```



92  
93

```
do  
{
```

```
PORTD = A[i]; //Carrega caractere de D0  
digitalWrite(LE_D0, HIGH); //Passa D0  
delay(5); //Temporiza transição  
digitalWrite(LE_D0, LOW); //Congela D0  
PORTD = A[j]; //Carrega caractere de D1  
digitalWrite(LE_D1, HIGH); //Passa D1  
delay(5); //Temporiza transição  
digitalWrite(LE_D1, LOW); //Congela D1  
PORTD = A[k]; //Carrega caractere de D2  
digitalWrite(LE_D2, HIGH); //Passa D2  
delay(5); //Temporiza transição  
digitalWrite(LE_D2, LOW); //Congela D2  
PORTD = A[l]; //Carrega caractere de D3  
digitalWrite(LE_D3, HIGH); //Passa D3  
delay(5); //Temporiza transição  
digitalWrite(LE_D3, LOW); //Congela D3  
digitalWrite(LE_D0, LOW); //Congela D0  
digitalWrite(LE_D1, LOW); //Congela D1  
digitalWrite(LE_D2, LOW); //Congela D1  
delay(350); // Retém os 4 caracteres disponíveis  
i++; // Incrementa posicionadores  
j++;  
k++;  
l++;  
if (j > 11) j = 0;  
if (k > 11) k = 0;  
if (l > 11) l = 0;
```

121

```
} while (i < 12);
```

```
94 PORTD = A[i]; //Carrega caractere de D0  
95 digitalWrite(LE_D0, HIGH); //Passa D0  
96 delay(5); //Temporiza transição  
97 digitalWrite(LE_D0, LOW); //Congela D0  
98 PORTD = A[j]; //Carrega caractere de D1  
99 digitalWrite(LE_D1, HIGH); //Passa D1  
100 delay(5); //Temporiza transição  
101 digitalWrite(LE_D1, LOW); //Congela D1  
102 PORTD = A[k]; //Carrega caractere de D2  
103 digitalWrite(LE_D2, HIGH); //Passa D2  
104 delay(5); //Temporiza transição  
105 digitalWrite(LE_D2, LOW); //Congela D2  
106 PORTD = A[l]; //Carrega caractere de D3  
107 digitalWrite(LE_D3, HIGH); //Passa D3  
108 delay(5); //Temporiza transição  
109 digitalWrite(LE_D3, LOW); //Congela D3  
110 digitalWrite(LE_D0, LOW); //Congela D0  
111 digitalWrite(LE_D1, LOW); //Congela D1  
112 digitalWrite(LE_D2, LOW); //Congela D1  
113 delay(350); // Retém os 4 caracteres disp  
114 i++; // Incrementa posicionadores  
115 j++;  
116 k++;  
117 l++;  
118 if (j > 11) j = 0;  
119 if (k > 11) k = 0;  
120 if (l > 11) l = 0;
```

---



*Até a próxima!*