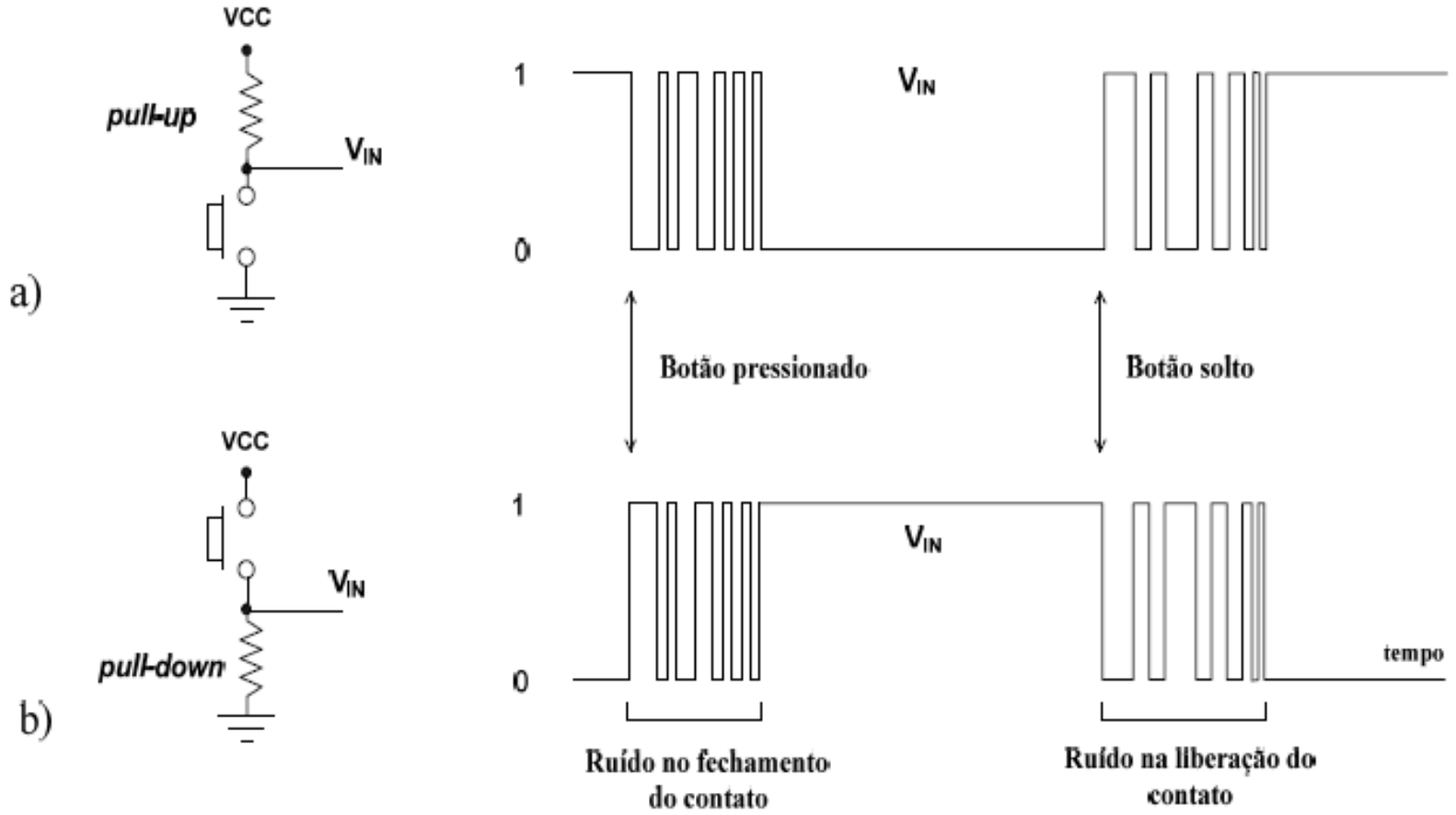
MCO60408 Microcontroladores

Aula 05 – Utilização do Botão e Display de 7 Seguimentos

Sumário

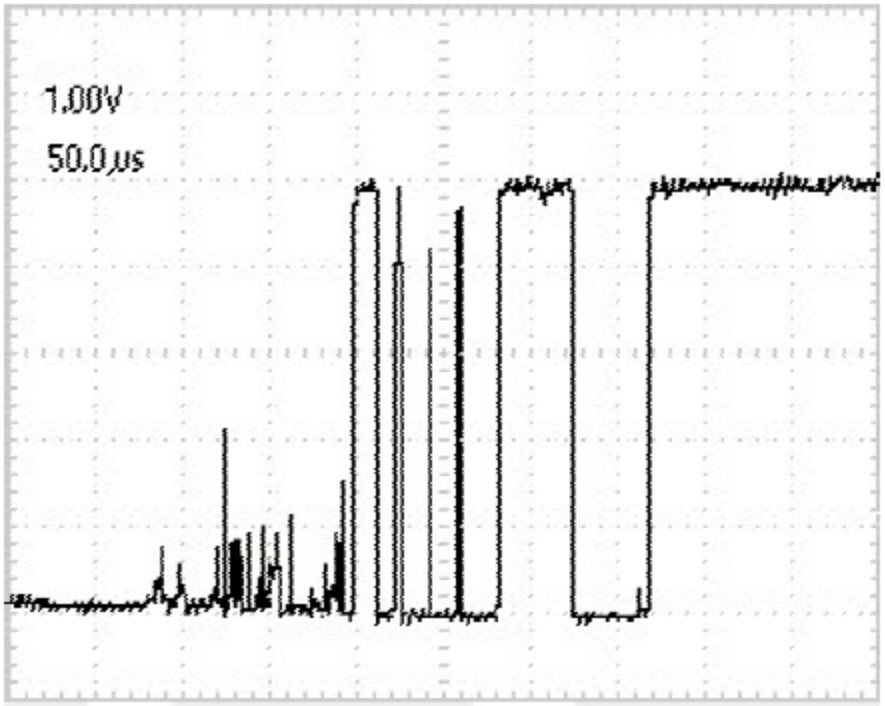
- Ruído ao pressionar o Botão;
- *Display* de & Seguimentos;
- Leituras Obrigatória e Recomendada.

Exemplo do ruído que pode ser gerado ao se pressionar e soltar um botão

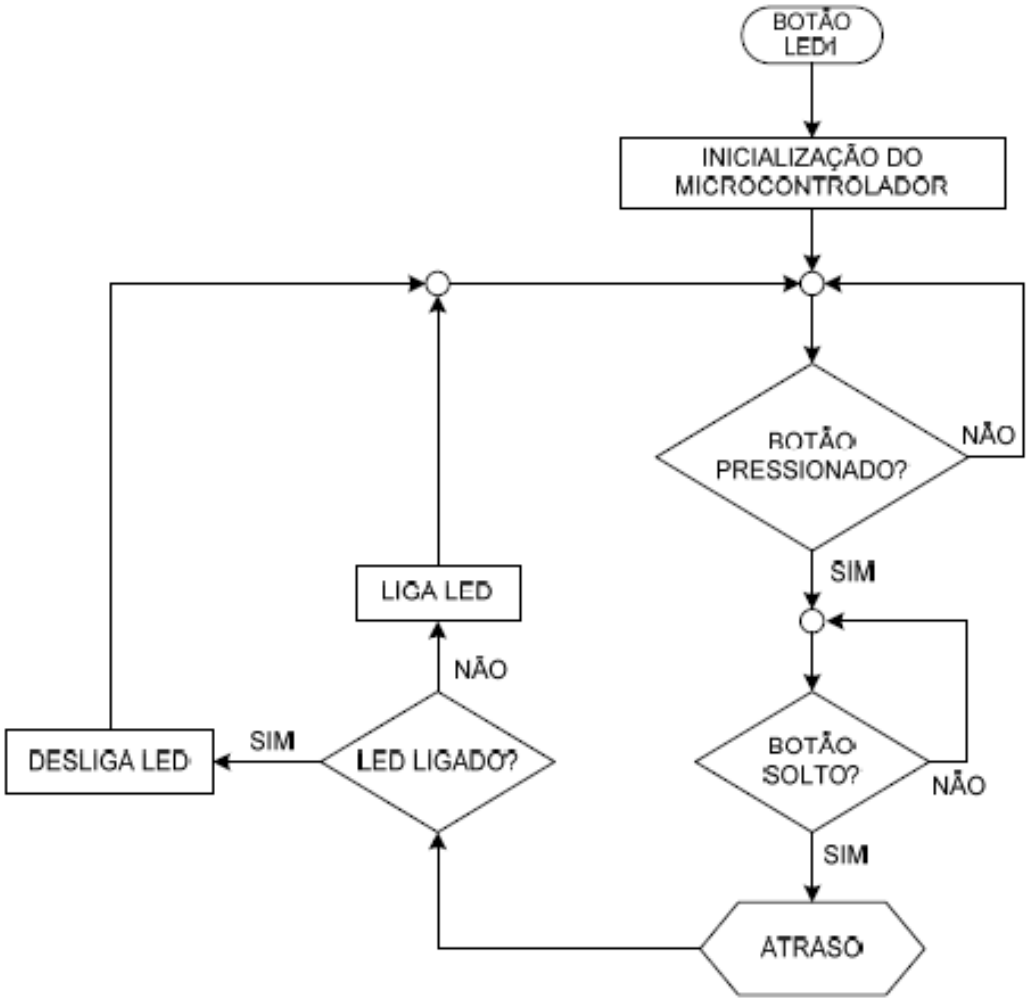


a) usando um resistor de pull-up e b) usando um resistor de pull-down.

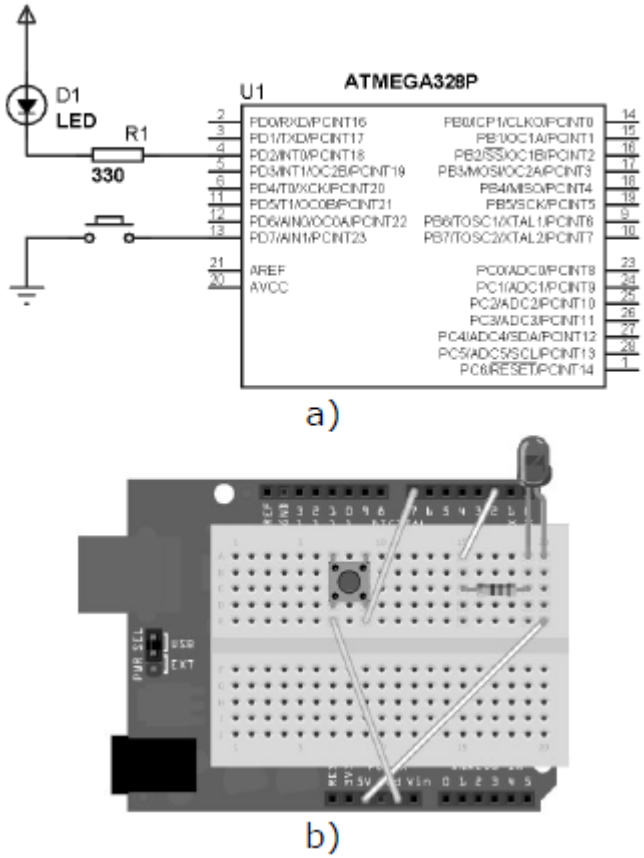
Ruído real gerado ao se soltar um botão com *PULL-UP*.



Fluxograma do programa para ligar e apagar um LED com um botão



Circuito para ligar e apagar um LED com um botão

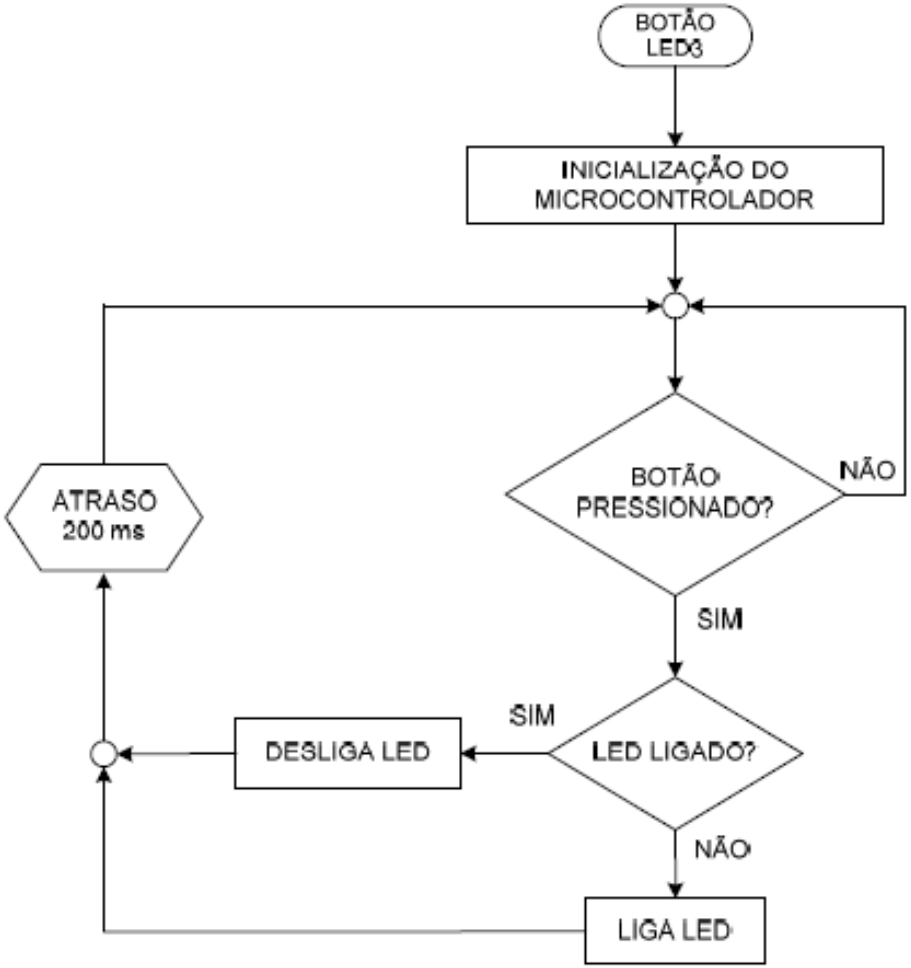


a) esquematico e b) montagem para o Arduino.

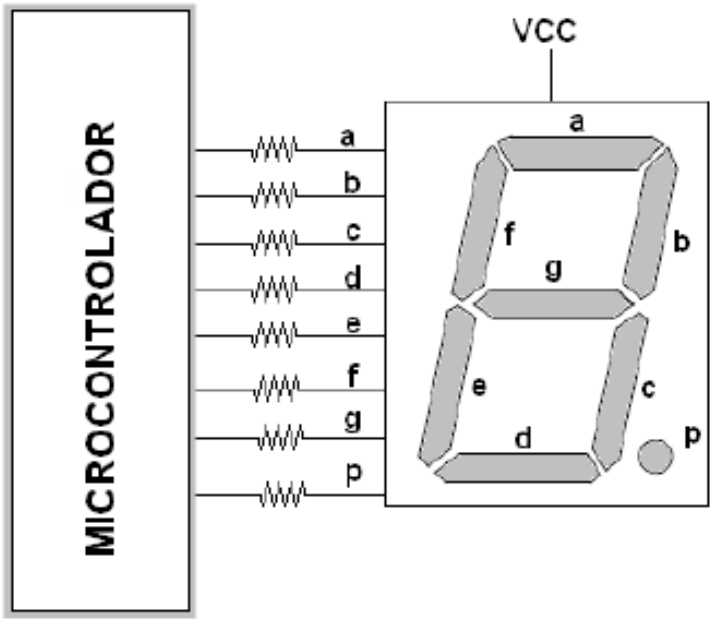
CÓDIGO

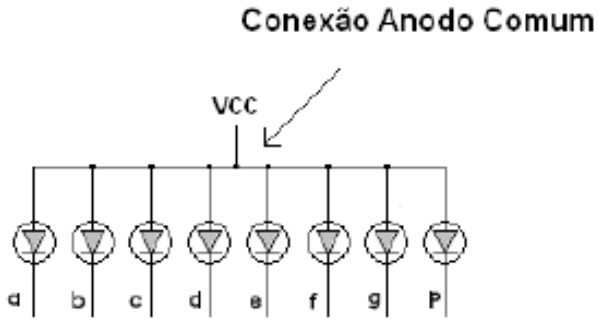
```
7   #define F_CPU 16000000UL    /*define a frequência do microcontrolador 16MHz (necessário
8                               para usar as rotinas de atraso)*/
9   #include <avr/io.h>         //definições do componente especificado
10  #include <util/delay.h>     //biblioteca para as rotinas de
11
12  //Definições de macros - para o trabalho com os bits de uma variável
13  #define set_bit(Y,bit_x) (Y|=(1<<bit_x))        //ativa o bit x da variável Y (coloca em 1)
14  #define clr_bit(Y,bit_x) (Y&=~(1<<bit_x))       //limpa o bit x da variável Y (coloca em 0)
15  #define cpl_bit(Y,bit_x) (Y^=(1<<bit_x))        //troca o estado do bit x da variável Y
16  #define tst_bit(Y,bit_x) (Y&(1<<bit_x))         //testa o bit x da variável Y (retorna 0 ou 1)
17
18  #define LED   PD2   //LED é o substituto de PD2 na programação
19  #define BOTAO PD7   //BOTAO é o substituto de PD7 na programação
20
21  int main(void)
22  {
23      DDRD = 0b00000100;  //configura o PORTD, PD2 saída, os demais pinos entradas
24      PORTD= 0b11111111;  /*habilita o pull-up para o botão e apaga o LED (todas as entradas com pull-ups habilitad
25
26      while(1)                                //laço infinito
27      {
28          if(!tst_bit(PIND,BOTAO))            //se o botão for pressionado executa o if
29          {
30              while(!tst_bit(PIND,BOTAO));    //fica preso até soltar o botão
31              _delay_ms(10);                  //atraso de 10 ms para eliminar o ruído do botão
32              if(tst_bit(PORTD,LED))          //se o LED estiver apagado, liga o LED
33                  clr_bit(PORTD,LED);
34              else                            //se não apaga o LED
35                  set_bit(PORTD,LED);
36          }
37      }
38  }
```

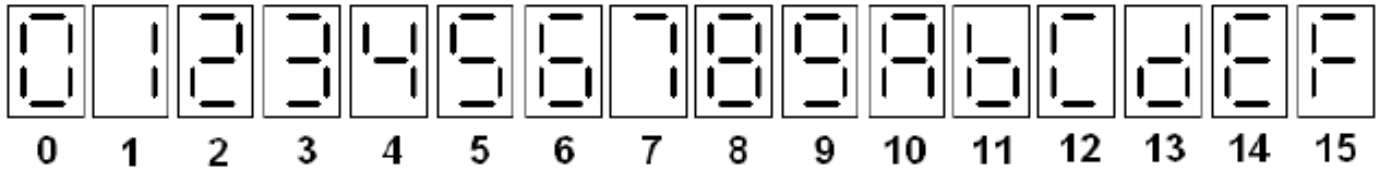
Fluxograma para piscar um LED enquanto um botão e mantido pressionado



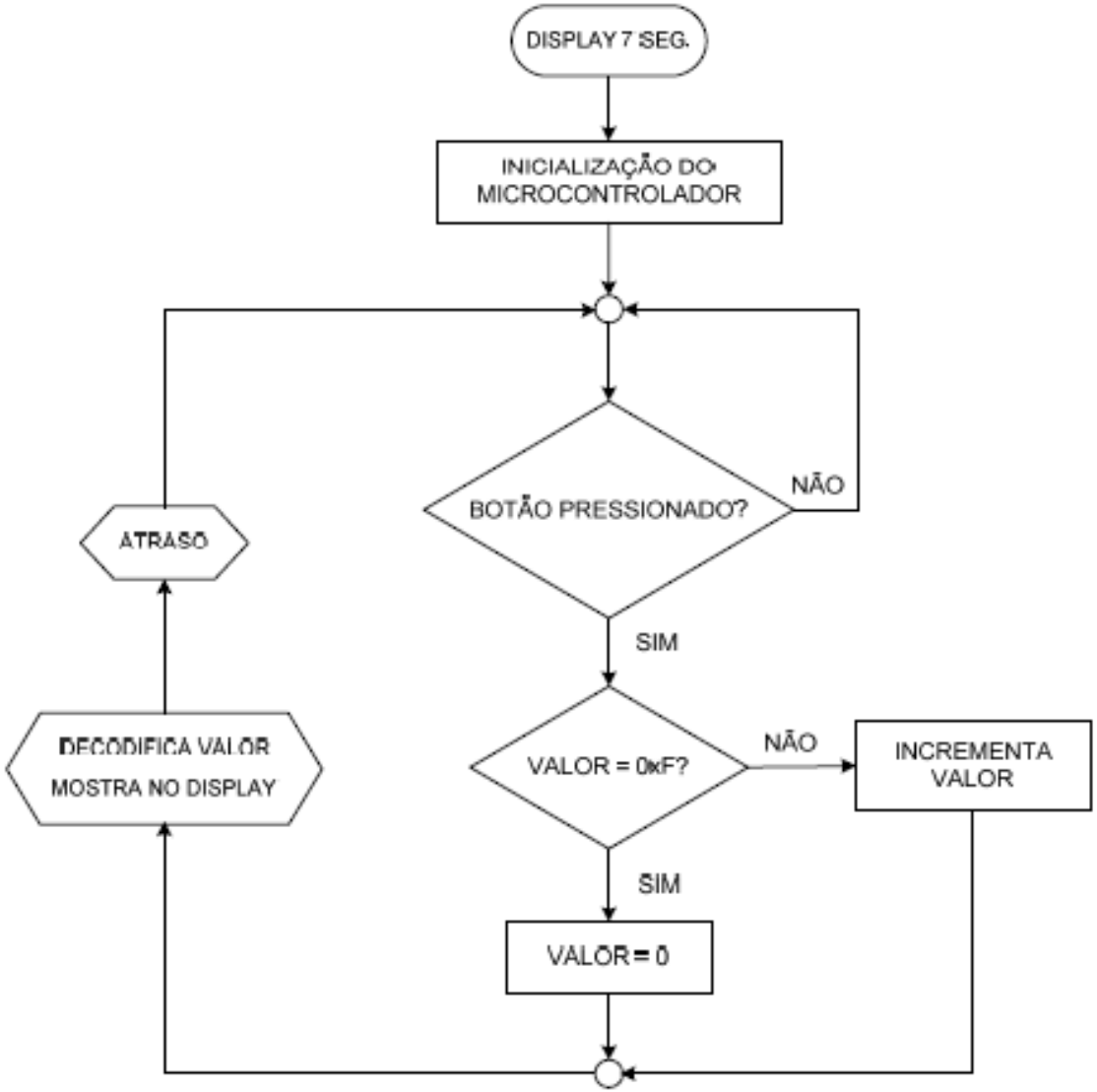
Display de 7 segmentos anodo comum







Fluxograma para apresentar um número hexadecimal de 0 ate F quando um botão é pressionado.



Valores para a decodificação de display de 7 segmentos

Dígito	Anodo comum		Catodo comum	
	gfedcba		gfedcba	
0	1000000b	0x40h	0111111b	0x3Fh
1	1111001b	0x79h	0000110b	0x06h
2	0100100b	0x24h	1011011b	0x5Bh
3	0110000b	0x30h	1001111b	0x4Fh
4	0011001b	0x19h	1100110b	0x66h
5	0010010b	0x12h	1101101b	0x6Dh
6	0000010b	0x02h	1111101b	0x7Dh
7	1111000b	0x78h	0000111b	0x07h
8	0000000b	0x00h	1111111b	0x7Fh
9	0011000b	0x18h	1100111b	0x67h
A	0001000b	0x08h	1110111b	0x77h
B	0000011b	0x03h	1111100b	0x7Ch
C	1000110b	0x46h	0111001b	0x39h
D	0100001b	0x21h	1011110b	0x5Eh
E	0000110b	0x06h	1111001b	0x79h
F	0001110b	0x0Eh	1110001b	0x71h

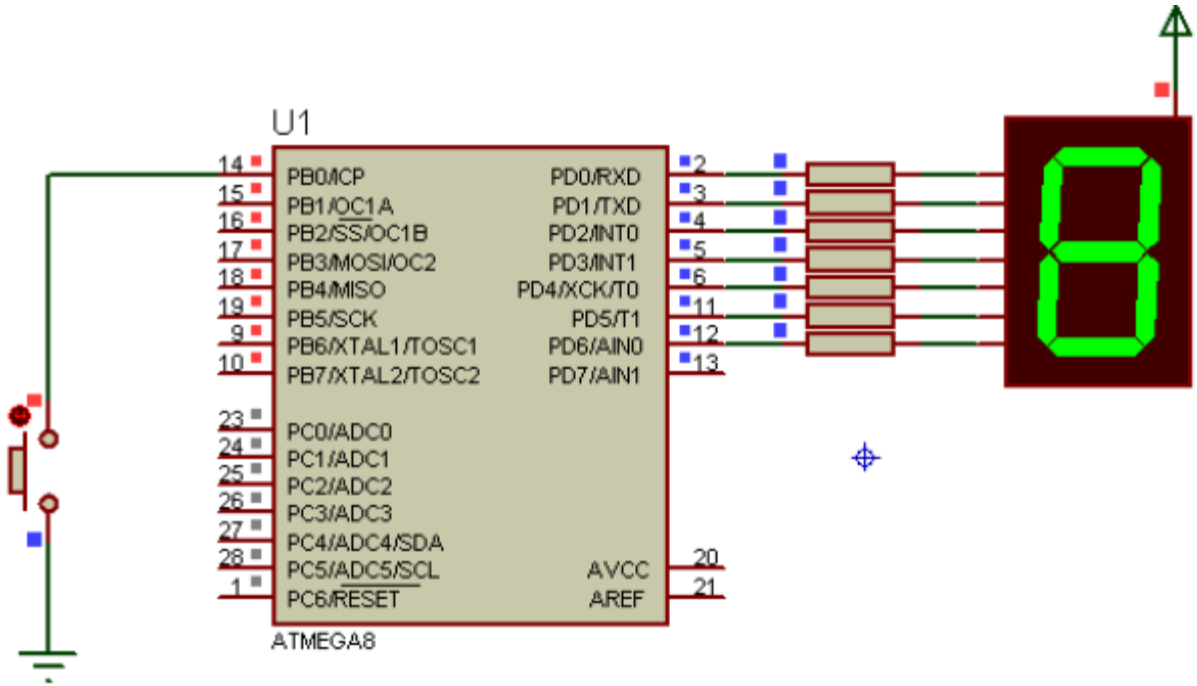
Código

```
#define F_CPU 16000000UL    //define a frequência do microcontrolador em 16MHz

#include <avr/io.h>         //definições do componente especificado
#include <util/delay.h>     //biblioteca para o uso das rotinas de _delay_
#include <avr/pgmspace.h>   //biblioteca para poder gravar dados na memória flash

//Definições de macros - para o trabalho com os bits de uma variável
#define tst_bit(Y,bit_x) (Y&(1<<bit_x)) //testa o bit x da variável Y (retorna 0 ou 1)

#define DISPLAY  PORTD      //define um nome auxiliar para o display
#define BOTAO    PB0        //define PB0 com o nome de BOTAO

//variável gravada na memória flash
const unsigned char Tabela[] PROGMEM = {0x40, 0x79, 0x24, 0x30, 0x19, 0x12, 0x02, 0x78,
                                        0x00, 0x18, 0x08, 0x03, 0x46, 0x21, 0x06, 0x0E};

int main()
{
    unsigned char valor = 0;    //declara variável local
    DDRB = 0b11111110;          //PB0 como pino de entrada, os demais pinos como saída
    PORTB= 0x01;                //habilita o pull-up do PB0
    DDRD = 0xFF;                //PORTD como saída (display)
    PORTD= 0xFF;                //desliga o display
    UCSR0B = 0x00;              //PD0 e PD1 como I/O genérico, para uso no Arduino
```

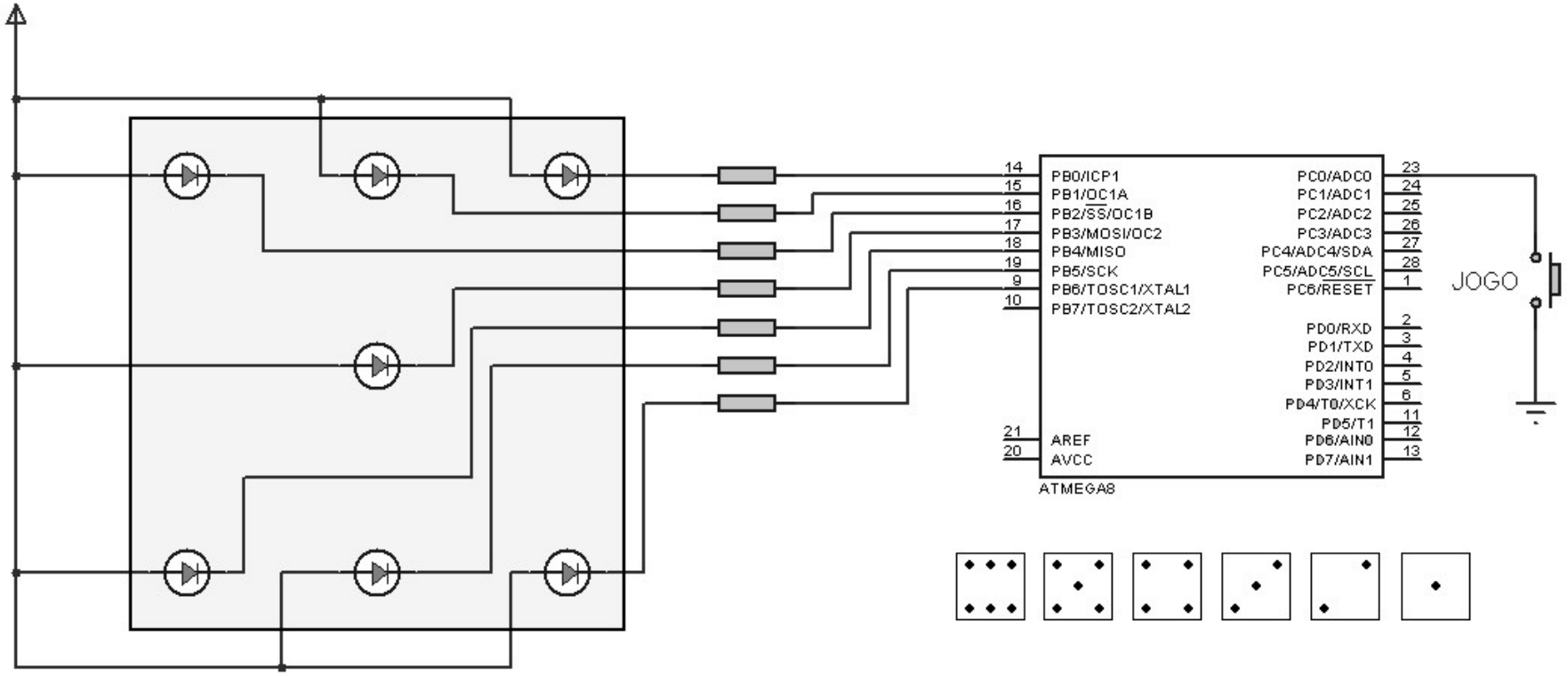
Código

```
31      while(1)                    //laço infinito
32      {
33          if(!tst_bit(PINB,BOTAO))//se o botão for pressionado executa
34          {
35              if(valor==0x0F)     //se o valor for igual a 0xF, zera o valor,
36              valor=0;
37              else                //se não o incrementa
38              valor++;
39              //decodifica o valor e mostra no display, busca o valor na Tabela.
40              DISPLAY = pgm_read_byte(&Tabela[valor]);
41              _delay_ms(200);     //atraso para incremento automático do nr. no display
42          }
43      }
44  }
45
```

Circuito para acionamento de um display de 7 segmentos anodo comum



Exercício


14
15
16
17
18
19
9
10
PB0/ICP1
PB1/OC1A
PB2/SS/OC1B
PB3/MOSI/OC2
PB4/MISO
PB5/SCK
PB6/TOSC1/XTAL1
PB7/TOSC2/XTAL2
21
20
AREF
AVCC
ATMEGA8
PC0/ADC0
PC1/ADC1
PC2/ADC2
PC3/ADC3
PC4/ADC4/SDA
PC5/ADC5/SCL
PC6/RESET
PD0/RXD
PD1/TXD
PD2/INT0
PD3/INT1
PD4/T0/XCK
PD5/T1
PD6/AIN0
PD7/AIN1
23
24
25
26
27
28
1
2
3
4
5
6
11
12
13
JOGO

Leituras Obrigatória e Recomendada

Leituras Obrigatória e Recomendada

- Leitura obrigatória:
  - LIMA, VILLAÇA – Cap 5;

MCO60408 Microcontroladores

Aula 04 – Programação C para AVRs