Atividade_07 - Livro AVR e Arduino – Técnicas de Projeto
Capítulo: 9 (TEMPORIZADORES/CONTADORES)

Título: Usando timers/contadores para controlar servo-motores

Objetivos: Aprender a usar os timers dos microcontroladores da Atmel. Aprender o que são e para que servem servo-motores.

Nesta prática utilizaremos o Tinkercad para simular um circuito simples usando o microcontrolador Atmega328, utilizado nas placas Arduino UNO. Desta vez, programaremos usando um código C para controlar servo-motores.

1. Procedimentos:

1. Acesse sua conta no Tinkercad (tinkercad.com) e vá para a aba circuits (https://www.tinkercad.com/circuits). Você pode utilizar o projeto disponível em https://www.tinkercad.com/things/5vBJtpVQd5l como exemplo de como controlar os servo-motores. Você pode usar este também: https://wokwi.com/projects/332108726738616915.

2. Mas afinal, **o que são servo-motores?** Pesquise e coloque a sua resposta aqui. Foque nos pequenos servos de até 90 gramas. **Como determinamos a posição do eixo deles?**
Servo-motores são dispositivos eletrônicos e atuadores rotativos ou lineares que giram e empurram peças de uma máquina com precisão. Os servos são usados principalmente em posição angular ou linear e para velocidade específica e aceleração.

3. Crie um projeto adicionando 3 displays de 7 segmentos (3 dígitos), dois botões e um servo-motor.

4. Elaborar um programa para controlar o ângulo de giro do servo motor de acordo com os dois botões. Um botão é responsável pelo incremento da largura do pulso e o outro pelo decremento. Comece com um pulso de 0,5 ms até 2,5 ms, com passo de incremento de 0,1 ms. **Qual a relação angular de giro com a largura de pulso?** Escreva o ângulo do servo-motor nos displays de 7 segmentos (0 a 180 graus). **Atenção: você deve utilizar um dos timers para gerar o evento (interrupção) para o chaveamento entre os displays de 7 segmentos.**

Para a variação de 0,1 ms começando 0,5 ms e indo até 2,5 ms, temos que o delocamento é 200 e o grau de deslocamento é 9, usando a regra de três.

5. Cole o código fonte do microcontrolador ao final deste arquivo e inclua a imagem de seu design. Importante: Deixe seu circuito público no Tinkercad e cole o link para ele aqui:

TINKERCAD: **https://www.tinkercad.com/things/fPHLGsitGds-atividade-07/editel?sharecode=bs2Oa7t3ckTRPr3_LJJkS1wxc4kKkuCFmhrUU94932U**

CÓDIGO:

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>
#include <avr/pgmspace.h>

//Definições de macros
#define set_bit(Y,bit_x) (Y|=(1<<bit_x))
#define clr_bit(Y,bit_x) (Y&=~(1<<bit_x))
#define tst_bit(Y,bit_x) (Y&(1<<bit_x))
#define cpl_bit(Y,bit_x) (Y^=(1<<bit_x))

#define TOP 39999
#define BOTAO1 PB0
#define BOTAO2 PB2
```

```
#define DISPLAY   PORTD
#define D1 PB3
#define D2 PB4
#define D3 PB5

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

int deslocamento = 200;
float grausDeslocamento = 9;

ISR(PCINT0_vect);
void escreveDisplay();

int main()
{
  Serial.begin(300);

  DDRB  = 0b11111010;
  PORTB = 0b00000101;
  DDRD = 0xFF;
  PORTD= 0xFF;
  UCSR0B = 0x00;


  ICR1 = TOP;

  TCCR1A = (1<<WGM11);


  TCCR1B = (1<<WGM13) | (1<<WGM12) | (1 << CS11);
  set_bit(TCCR1A,COM1A1);
  OCR1A = 1000;


  PCICR = 1<<PCIE0;
  PCMSK0 = (1<<PCINT0)|(1<<PCINT2);

  sei();
  while(1){
    unsigned char valorGrau = ((OCR1A-1000)/deslocamento)*grausDeslocamento;
    char d1 = 0, d2 = 0, d3 = 0;
    if (valorGrau<10) {
       d1 = valorGrau;
       d2 = 0;
       d3 = 0;
    }else if(valorGrau<100) {
       d3 = 0;
       d2 = valorGrau/10;
       d1 = valorGrau%10;
    }else{
       d3 = valorGrau/100;
       d2 = (valorGrau/10)%10;
       d1 = valorGrau%10;
    }

    if (d3 >= 0) {
      clr_bit(PORTB,D2);
      clr_bit(PORTB,D1);
      DISPLAY = pgm_read_byte(&Tabela[d3]);
      set_bit(PORTB,D3);
      _delay_ms(2);
    }
    if (d2 >= 0) {
      clr_bit(PORTB,D3);
      clr_bit(PORTB,D1);
      DISPLAY = pgm_read_byte(&Tabela[d2]);
      set_bit(PORTB,D2);
      _delay_ms(2);
    }
          clr_bit(PORTB,D2);
          clr_bit(PORTB,D3);
          DISPLAY = pgm_read_byte(&Tabela[d1]);
```

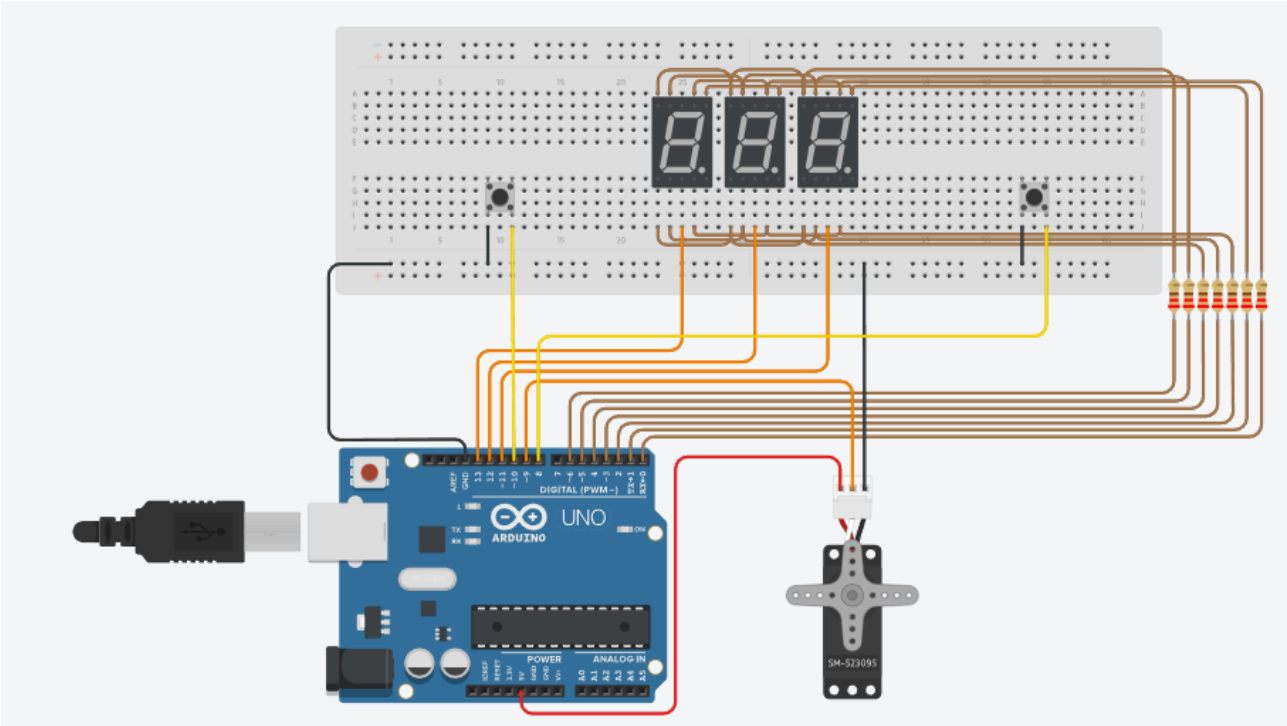
```
            set_bit(PORTB,D1);
            _delay_ms(2);
    }
}

ISR(PCINT0_vect){
            if(!tst_bit(PINB, BOTAO1)){
      if(OCR1A < 5000){
            OCR1A +=deslocamento;
      }

    }if(!tst_bit(PINB, BOTAO2)){
      if(OCR1A > 1000){
            OCR1A -=deslocamento;
      }

    }
}
```

ARDUINO:



ATENÇÃO: Documente seu código. Cada linha/bloco deve deixar explícito o seu papel.
ATENÇÃO: Na versão final do seu projeto, as funções pinMode(), digitalWrite() e digitalRead() são proibidas. O uso delas fará a nota atribuída ser zero. Utilizar PWM do timer 0, 1 ou 2 é obrigatório para a atividade ser avaliada.

RÚBRICA:
Pergunta teórica: 10%
Circuito: 10%
Botões funcionando: 10%
Display funcionando com chaveamento entre displays usando interrupção do timer: 30%
Passo 4 funcionando: 40%
Valor desta atividade na média: 0.6