

## Atividade\_06 - Livro AVR e Arduino – Técnicas de Projeto

### Capítulo: 9 (TEMPORIZADORES/CONTADORES)

#### **Título: Usando timers/contadores para gerar interrupções periódicas e PWMs – TC0, TC1 e TC2**

**Objetivos:** Aprender a usar os timers dos microcontroladores da Atmel.

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 acender e apagar LEDs usando interrupções geradas por timers. Também faremos um LED acender progressivamente e apagar progressivamente usando PWM.

#### **1. Procedimentos:**

1. Acesse sua conta no Tinkercad ([tinkercad.com](https://www.tinkercad.com)) e vá para a aba circuits (<https://www.tinkercad.com/circuits>).
2. Comece colocando uma placa do Arduino. Ligue três LEDs, um na porta PB5 (13), um na porta PB4 (12), e um na porta PB3 (11). Baixe o capítulo 9 do livro e mãos à obra.
3. Configure seu timer 0 para que a cada interrupção (escolha que interrupção), alterne o estado do LED no pino 13. O período de tempo entre cada troca de estado do LED deve ser o mais próximo possível de 10ms. **Pergunta teórica: Quais os parâmetros do timer 0 que o fazem se aproximar do valor pedido (valores do prescaler e do comparador, mostre seus cálculos)? Qual o intervalo de tempo exato obtido?**

#### **Solução:**

Calculando a frequência para 10ms (0,01s):

$$f = 1 \div 0.01 \Leftrightarrow 100\text{Hz}$$

Para calcular o valor do registrador OCR0A utilizando prescaler para 1024, temos:

$$(16000000 \div (1024 * 100)) - 1 = 156.25 - 1 \Leftrightarrow 155.25$$

Assim, o intervalo de tempo mais próximo obtido é:

$$156 * (1 \div 16000000) * 1024 = 0.009984\text{ms}$$

4. Agora configure seu timer 1 para que a cada interrupção, alterne o estado do LED no pino 12. O período de tempo entre cada troca de estado do LED deve ser o mais próximo possível de 998ms. **Pergunta teórica: Quais os parâmetros do timer 1 que o fazem se aproximar do valor pedido (valores do prescaler e do comparador, mostre seus cálculos)? Qual o intervalo de tempo exato obtido?**

**Solução:**

Calculando a frequência para 998ms (0,998s):

$$f = 1 \div 0.998 \Leftrightarrow 1.00200401\text{Hz}$$

Para calcular o valor do registrador OCR1A utilizando prescaler para 1024, temos:

$$(16000000 \div (1024 * 1,00200401)) - 1 = 15592,74997$$

Assim, o intervalo de tempo mais próximo obtido é:

$$15593 * (1 \div 16000) * 1024 = 997.952\text{ms}$$

**DICA: Lembre-se de usar main() ao invés de setup() e loop(), pois com setup() o init() é invocado configurando o timer1.**

5. Agora configure seu timer 2 como PWM. Configure o pino 11 (PB3) como saída do comparador do timer 2. Use o modo PWM rápido. Vamos fazer o LED ligado no pino 11 acender e apagar progressivamente. Para isto, use a interrupção do timer 0 para ajustar o valor do PWM. Como ele gera uma interrupção a cada 10ms, seu LED deve levar aproximadamente 2.5s para acender completamente e 2.5s para apagar por completo.

**Atenção: O Tinkercad não simula bem o PWM.**

**Solução:**

Calculando a frequência para 2.5s:

$$f = 1 \div 2.5 \Leftrightarrow 0.4\text{Hz}$$

Para calcular o valor do registrador OCR2A utilizando prescaler para 1024, temos:

$$(16000000 \div (1024 * 0.4)) - 1 = 39062.5 - 1 \Leftrightarrow 39061.5$$

Assim, o intervalo de tempo mais próximo obtido é:

$$39061 * (1 \div 16000000) * 1024 = 2,499904\text{ms}$$

**6. 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:**

**LINK TINKERCAD:**

[https://www.tinkercad.com/things/hM3hgVSurm2-spectacular-jarv/editel?sharecode=sQdZhpvtngREXsJTdZ0Uu\\_LPH\\_2vH\\_OS17fx6nYtJBM](https://www.tinkercad.com/things/hM3hgVSurm2-spectacular-jarv/editel?sharecode=sQdZhpvtngREXsJTdZ0Uu_LPH_2vH_OS17fx6nYtJBM)

## CÓDIGO:

```
#define F_CPU = 16000000UL

#include <avr/io.h>

#include <avr/interrupt.h>

#define cpl_bit(y,bit) (y^=(1<<bit)) //troca o estado lógico do bit x da variável Y

#define LED0 PB5

#define LED1 PB4

#define LED2 PB3

ISR(TIMER0_COMPA_vect) {

    cpl_bit(PORTB,LED0);

}

ISR(TIMER1_COMPA_vect) {

    cpl_bit(PORTB,LED1);

}

ISR(TIMER2_COMPA_vect) {

    cpl_bit(PORTB,LED2);

}

int main() {

    DDRB = 0b00111000; //somente pino do LED como saída

    PORTB = 0b11000111; //apaga LED e habilita pull-ups nos pinos não utilizados
```

```

//Timer 0

TCCR0A = 0b01000000;

OCR0A = 155;

TCCR0B = (1<<CS02) | (1<<CS00);    // TC0 com prescaler de 1024, a 100Hz

TIMSK0 = 1<<OCIE0A;                //habilita a interrupção do TC0


//Timer 1

TCCR1A = 0b01000000;

OCR1A = 15592;

TCCR1B = (1<<CS11) | (1<<CS10);    // TC1 com prescaler de 1024, a 1.00200401Hz

TIMSK1 = 1<<OCIE1A;                //habilita a interrupção do TC1


//Timer 2

TCCR2A = 0b01010011;

OCR2A = 39061;

TCCR2B = (1<<CS22) | (1<<CS21) | (1<<CS20); // TC2 com prescaler de 1024, a 0.4Hz

TIMSK2 = 1<<OCIE2A;                //habilita a interrupção do TC2

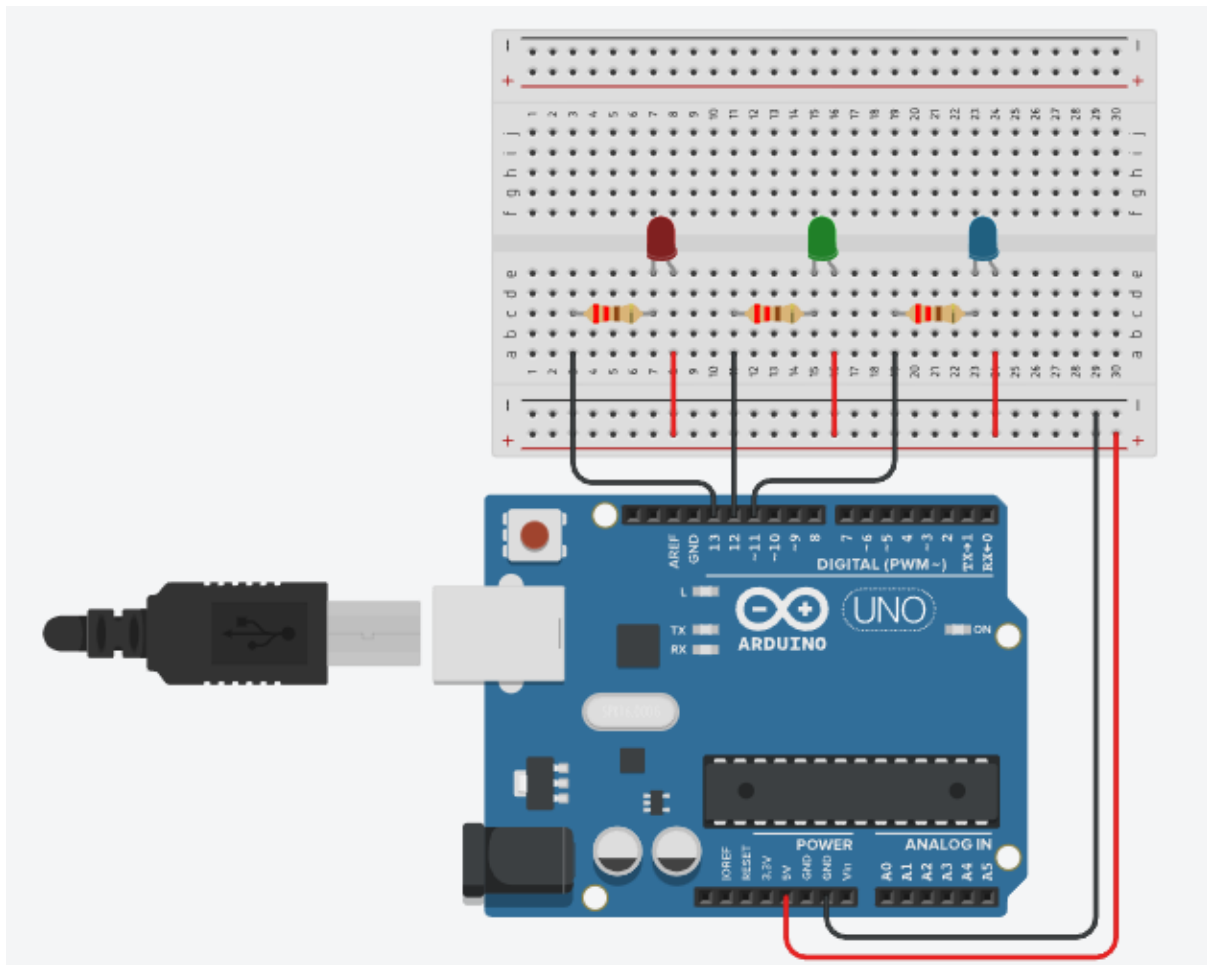

sei();

while(1){};

}

```

## 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.

---

## RÚBRICA:

**TC0:** 30%

**TC1:** 40%

**TC2:** 30%

**Valor desta atividade na média:** 0.6