Sistemas Embebidos
Trabalho Prático 3

Carlos Abreu[1] e João Faria[2]

[1]**cabreu@estg.ipvc.pt**

[2]**joao.pedro.faria@estg.ipvc.pt**

Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Viana do Castelo

Instituto Politécnico de Viana do Castelo
Escola Superior de Tecnologia e Gestão
2024

Carlos Abreu
www.estg.ipvc.pt/~cabreu

Curso:

Licenciatura em Engenharia de Redes e Sistemas de Computadores

Objetivo Pedagógico

Utilização de Timers no Atmega328.

Sumário: **Duração:** 3 horas

1. Compreender, genericamente, o funcionamento de um Timer num MCU.
2. Configurar o TIMER1 do ATmega328 para gerar eventos periódicos recorrendo à técnica de *pooling*.
3. Configurar e utilizar as interrupções internas associadas aos Timers.

1. *Timers* - Introdução

Comece por abrir o datasheet do MCU que se encontra no moodle.

- Consulte a lista de *Features* do MCU que se encontra na primeira página do datasheet. Quantos *Timers* estão disponíveis?
- O modo de operação do *Timer 1* encontra-se descrito na secção 16 (16-bit Timer/Counter1 with PWM), consulte as seguintes secções:
  - 16.2 - Overview
  - 16.4 - Timer/Counter Clock Sources. O pino $clk_{I/O}$ do Arduino usado nas aulas encontra-se ligado a um oscilador de 16MHz.
  - 16.5 - Counter Unit. Qual o registo que pode ser usado para gerar uma interrupção com origem no *Timer 1*?
  - 16.9.1 - Normal Mode. Em que situação é que o TOV1 é ativo?
  - 17 - Timer/Countero and Timer/Counter1 Prescalers. O que é o *Prescaler* e e para que serve?
- Consulte a documentação dos registos usados para configuração do *Timer 1* e determine o valor a configurar em cada um deles para que o *Timer 1* funcione com um período de overflow de 1 segundo.
  - TCCR1A (ver sec. 16.11.1)
  - TCCR1B (ver sec. 16.11.2 - Bit 2:0 – CS12:0: Clock Select)
  - TCNT1 (ver sec. 16.11.4)

- TIMSK1 (ver sec. 16.11.8 - Bit 0 – TOIE1: Timer/Counter1, Overflow Interrupt Enable)
- TIFR1 (ver sec. 16.11.8 - Bit 0 – TOV1: Timer/Counter1, Overflow Flag)

2. *Timers* - Exercícios

Exercício 1 *Pretende-se utilizar o TIMER1 do ATmega328 para implementar uma função em C, com o nome* myDelay(), *que permita definir com precisão um período de espera de 1 segundo de forma a substituir a função* delay() *disponibilizada nativamente em Arduino C e utilizada no trabalho prático n.º 2. A função deverá basear-se numa solução com* POOLING *ao registo TOV1.*

1.1 *Faça uma cópia do circuito com o* display *de 7-segmentos utilizado no trabalho prático n.º 2 e do* device driver *implementado (*(num2display())*).*

1.2 *Desative todas as interrupções invocando a função* noInterrupts() *a partir da função* setup()*:*

```
void setup() { // Função de Inicialização
    ...
    noInterrupts(); // Desactiva todas as Interrupts
    ...
}
```

1.3 *Implemente a função* myDelay() *considerando o seguinte protótipo:*

```
void myDelay(void) { // Initialize timer1

    // 1 - Clear Control registers
    TCCR1A = 0x00; // Normal Operation Mode
    TCCR1B = 0x00; // Set Clock Source (16 MHz)

    // 2 - Define Prescaler
    bitWrite(TCCR1B, CS12, ____); // Define prescaler:
    bitWrite(TCCR1B, CS11, ____); // 256 prescaler (TCCR1B.CS1x = 100)
    bitWrite(TCCR1B, CS10, ____);

    // Init TCNT1 to the correct value for our interrupt period Tovf
    // Initial Value = 2^16 - 16MHz/Prescaler/Freq
    TCNT1 = ____;
    bitWrite(TIMSK1, TOIE1, ____); // Enable timer overflow interrupt
```

```
    while(!bitRead(TIFR1, TOV1)) { // Pooling TOV1 bit
    };

    bitWrite(TIFR1, TOV1, ____); // Clears Overflow Flag
    bitWrite(TIMSK1, TOIE1, ____); // Disable timer overflow interrupt
}
```

1.4 *Teste agora a função implementada com recurso ao seguinte bloco de código:*

```
void loop() { // Loop de Controlo
    num2Display(cnt);
    myDelay();
    cnt++;
}
```

1.5 *Reconfigure o TIMER 1 para operar com diferentes períodos de overflow (2 seg, 500 ms e 250 ms) e teste as diferentes situações. Na submissão do exercício, apresente os valores configurados em cada um dos registos para as várias situações na forma de comentários.*

Exercício 2 *Pretende-se implementar um sistema que apresente no* display *de 7-segmentos um valor que é incrementado a cada segundo e que deverá variar ciclicamente entre 0 e 9 de acordo com os seguintes requisitos:*

- *No loop de controlo é executada apenas a rotina de refrescamento do* display *de 7 segmentos (* num2display() *).*
- *Pretende-se utilizar o TIMER 1 do ATmega328 para gerar interrupções internas periódicas responsáveis pela atualização do contador.*

2.1 *Importe para a sua área de trabalho o circuito disponibilizado neste link.*

2.2 *Implemente (e invoque na função* setup()*) uma função para configuração do Timer 1 de forma a que seja despoletada uma interrupção periódica com a frequência de 1 Hz. Não se esqueça de garantir que as interrupções estão desabilitadas enquanto a sua configuração não estiver concluída.*

```
void initTimer1Int() {    // Configuração da interrupção associada ao timer 1
    noInterrupts();    // Turn off global interrupts
    ...
    ...
    ...
    interrupts();    // Turn on global interrupts
}
```

2.3 *Implemente a Rotina de Serviço à Interrupção despoletada pelo overflow do Timer 1 e responsável por atualizar o valor a apresentar no* display. *O protótipo da RSI é o seguinte:*

```
// Timer 1 Overflow Interrupt Service Routine
ISR(TIMER1_OVF_vect) {

    // (re)Inicialização do Timer 1
    TCNT1 = ____ ;

    // Atualização do valor do display
    ...
}
```

2.4 *Teste o código implementado para diferentes frequências de operação (ex. 2Hz e 0.5Hz) e não se esqueça de o comentar convenientemente.*

Exercício 3 *Pretende-se implementar um semáforo semelhante aos que são usados para circulação alternada, por exemplo, em pontes estreitas para limitar a passagem dos veículos em apenas um sentido:*

- *Durante 4 segundos o semáforo 1 está VERDE e o semáforo 2 VERMELHO.*
- *Durante 1 segundo o semáforo 1 está AMARELO e o semáforo 2 mantém-se VERMELHO.*
- *Durante 4 segundos o semáforo 1 está VERMELHO e o semáforo 2 VERDE.*
- *Durante 1 segundo o semáforo 2 está AMARELO e o semáforo 1 mantém-se VERMELHO.*

3.1 *Importe para a sua área de trabalho o circuito disponibilizado neste link.*

3.2 *Implemente uma solução que cumpra com os requisitos apresentados e cuja arquitetura seja baseada em interrupções internas despoletadas pelo Timer 1.*

3.3 *Teste o código implementado e não se esqueça de o comentar convenientemente.*