Sistemas Embebidos

Trabalho Prático 5

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

Modulação PWM.

Sumário: **Duração:** 3 horas

1. Compreender o funcionamento da modulação PWM (*Pulse-Width mMdulation*).
2. Utilizar a modulação PWM no Atmega328p para controlo de um LED RGB
3. Compreender o funcionamento de um LED RGB

1. *PWM* - Introdução

Os sinais PWM (*Pulse-Width mMdulation*) são gerados a partir de dispositivos digitais como MCUs com o objetivo de simular sinais analógicos. Comutando rápida e repetidamente um sinal entre '0' e '1' é possível codificar um sinal analógico usando um sinal digital. As principais características de um sinal PWM encontram-se representadas na figura 1:

- Frequência - Número de vezes que o sinal é repetido por segundo.
- *Duty Cycle* - Relação entre o tempo que o sinal permanece no nível lógico '1' e o período do sinal (expresso em percentagem).

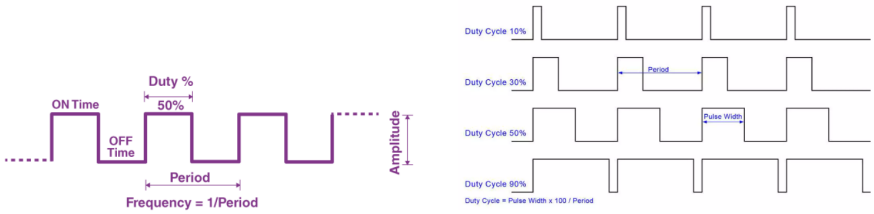

Figura 1

PWM: exemplos e principais características.

Considere um LED controlado por um sinal PWM com as seguintes características:

- Frequência: 2 Hz (500 ms de período).

- Duty-Cycle: 50%

O ser humano consegue processar cerca de 60 imagens por segundo e neste caso vai ver o LED a piscar (250 ms ligado, 250 ms desligado). No entanto, se aumentarmos a frequência para 200 Hz deixamos de ver o LED a piscar e passamos a ter a perceção o LED está ligado com um brilho que corresponde a metade da sua intensidade máxima (2.5ms ligado e 2.5ms desligado).

O Atmega328p disponibiliza 6 canais para gerar sinais PWM. Abra o *datasheet* que se encontra no moodle e identifique os pinos do MCU que podem ser usados para gerar sinais PWM. Ao escolher os pinos a usar como saídas de PWM é necessário ter em atenção que cada par de pinos está associado a um *Timer* pelo que se o *Timer* estiver a ser usado com outra finalidade poderá não estar disponível para gerar o PWM.

As bibliotecas do arduino disponibilizam a função *analogWrite()* que cria uma camada de abstração dos registos do Atmega328 que é necessário configurar para usar o *Timer* no modo PWM. Aceda à documentação da função e identifique a frequência do PWM gerado através da função *analogWrite()* no *arduino UNO*. Consulte a página Basics of PWM (Pulse Width Modulation) para saber como identificar os pinos do *arduino* que permitem gerar saidas PWM.

2. *PWM* - Exercícios

Exercício 1 *Pretende-se implementar um* device driver *para controlar um LED RGB com o seguinte protótipo:*

```
void SET_RGB_COLOR(unsigned int color);
```

*Os LEDs RGB possuem três LEDs encapsulados no mesmo dispositivo e permitem apresentar uma grande quantidade de cores que são originadas a partir de três cores primárias: vermelho (**R**ed), verde (**G**reen) e azul (**B**lue). Cada um destes LEDs pode ser controlado de forma individual ou combinados entre si para gerar diversas cores.*

1.1 *Importe para a sua área de trabalho o circuito disponibilizado neste link e implemente uma função que recebe como parâmetros os valor do duty-cycle entre* 0 *e* 255 *para cada uma das cores do LED RGB e gere o respetivo PWM em cada um dos pinos ligados ao LED RGB. A função deve obedecer ao seguinte protótipo:*

```
void LED_RGB( unsigned char red, unsigned char green, unsigned char blue );
```

1.2 *Implemente uma função que recebe como parâmetro de entrada um valor entre o e 6 e que acenda o LED de acordo com a codificação de cores apresentada na tabela*

Código Cor	LED_RGB()
0	LED_RGB(255, 0, 0)
1	LED_RGB(255, 127, 0)
2	LED_RGB(255, 255, 0)
3	LED_RGB(0, 255, 0)
4	LED_RGB(0, 0, 255)
5	LED_RGB(75, 0, 130)
6	LED_RGB(143, 0, 255)

Tabela 1 **LED RGB - códigos de cores**

1). Tire partido da função implementada no ponto anterior e da utilização de um array bi-dimensional:

```
void SET_RGB_COLOR(unsigned int ColorCode) {

    unsigned char color_map[7][3] = {
         {255, 000, 000}, // [0] - RED
         {255, 127, 000}, // [1] - ORANGE
         ...
         {143, 000, 255} // [6] - VIOLET
    };

    LED_RGB( ... )

}
```

1.3 *Implemente uma solução que permita enviar através da consola um código de cor entre 0 e 6 para configurar a cor pretendida no LED. Sempre que o utilizador inserir um novo código a cor do LED deve ser alterada em conformidade.*

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

Exercício 2 *Pretende-se implementar um sistema utilizando o paradigma de programação orientado a eventos. Do ponto de vista funcional o sistema apresenta os seguintes requisitos:*

- *O sistema arranca com o LED RGB a vermelho com intensidade máxima e vai reduzindo a sua intensidade durante 4 segundos até se apagar;*
- *Após os 4 segundos iniciais, o LED RGB fica verde durante 1 segundo;*

Número de eventos	*Código Cor*
0 *a* 4	0
5 *a* 9	1
10 *a* 14	2
15 *a* 19	3
20 *a* 24	4
25 *a* 29	5
30 *ou mais*	6

Tabela 2 Representação cromática do número de eventos.

- *Depois de* 1 *segundo a verde, o LED RGB fica a piscar em modo intermitente com cor azul e um período de* $500\ ms$ *(*$250\ ms$ *ON,* $250\ ms$ *OFF) até que o botão seja pressionado.*
- *Após o botão ser pressionado:*
  - *O número de vezes que o LED acedeu com a cor azul aparece na consola;*
  - *O LED RGB fica a piscar em modo intermitente com período de* $500\ ms$ *na cor que representa o número de vezes que o LED acendeu (ver tabela 2).*
- *Depois do LED piscar* 3 *vezes com indicação do número de vezes que o LED acendeu a azul, todo o processo é repetido indefinidamente.*

2.1 *Tendo em conta os requisitos da aplicação, quantos estados distintos identifica? Desenhe o respetivo diagrama de estados.*

2.2 *Desenhe os fluxogramas das funções que considerar relevantes e que descrevam a operação lógica recorrendo à implementação de uma máquina de estados.*

2.3 *Importe para a sua área de trabalho o circuito disponibilizado neste link e implemente a solução apresentada nos pontos anteriores.*

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