

**Instituição:** Instituto Federal de Educação, Ciência e Tecnologia - Paraíba (IFPB).

**Disciplina:** Microprocessadores e Microcontroladores.

**Professor:** Fagner de Araujo Pereira.

Alunos: \_\_\_\_\_

## Exercícios - Programação de ADC, DAC e Timers no STM32F407

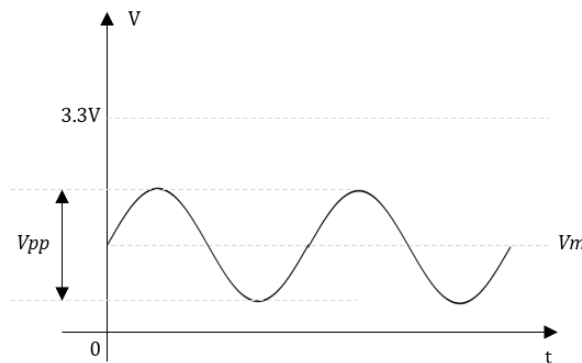
**Realize a programação do STM32F407 para implementar as seguintes tarefas:**

*(\*Alguns valores serão fornecidos individualmente para cada dupla durante a execução da atividade)*

*Em algumas questões, será necessário consultar o Manual de Referência do STM32F407 para verificar detalhes dos registradores para configurar algumas funções.*

☐ 1. Acender um LED conectado ao pino **PA4** com diferentes intensidades de brilho, alterando a intensidade em tempo de execução, utilizando o conversor DAC com resolução de 12 bits. O buffer do conversor deve estar habilitado para garantir corrente suficiente para acender o LED, que deve iniciar apagado, aumentar gradativamente o brilho até o máximo e em seguida reduzir gradativamente o brilho até se apagar, repetindo o processo indefinidamente (*utilize um resistor de  $470\Omega$  para limitar a corrente no LED*).

☐ 2. Gerar no pino **PA5** um sinal analógico senoidal, conforme mostrado na figura abaixo, com resolução de 12 bits, com uma frequência de \_\_\_\_ Hz, amplitude de \_\_\_\_ V de pico-a-pico ( $V_{pp}$ ) e valor médio ( $V_m$ ) de \_\_\_\_ V. Utilize **500** amostras de 12 bits para representar um ciclo do sinal senoidal. Utilize uma interrupção de timer para controlar os instantes de atualização do conversor DAC.



☐ 3. O conversor ADC do STM32 possui um recurso chamado de Watchdog Analógico. Esse recurso compara automaticamente, por meio do hardware, o valor resultante de uma conversão ADC com dois valores limiares configuráveis (limiar inferior e limiar superior). A condição quando o valor resultante da conversão for menor que o limiar inferior ou maior que o limiar superior é sinalizada pelo bit AWD (Analog WatchDog) do registrador SR (Status Register) do conversor ADC. Escreva um programa para testar o recurso Watchdog Analógico no conversor ADC do STM32F407, printando o valor da tensão lida pelo conversor ADC com uma precisão de duas casas decimais e uma frase de alerta quando o valor convertido for menor que \_\_\_\_ V ou maior que \_\_\_\_ V. Utilize um potenciômetro para fornecer uma tensão ajustável ao ADC e testar o programa.

☐ 4. Reproduzir, usando o conversor DAC e um amplificador de áudio, as vozes dos integrantes da equipe dizendo a seguinte frase: "Meu nome é...", completando a frase com os seus próprios nomes. Utilize uma taxa de amostragem de 48 kHz e uma resolução de 8 bits. Utilize uma interrupção de timer para controlar os instantes de atualização do conversor DAC.

☐ 5. Controlar a posição do eixo de dois servomotores usando um joystick analógico, da mesma forma como foi feito nesse vídeo: <https://www.youtube.com/watch?v=3HWUVLiHDDA>.

☐ 6. Desenvolva um sistema de monitoramento de temperatura, usando o sensor de temperatura interno do STM32, que verifica três faixas de temperatura baseado nas seguintes premissas:

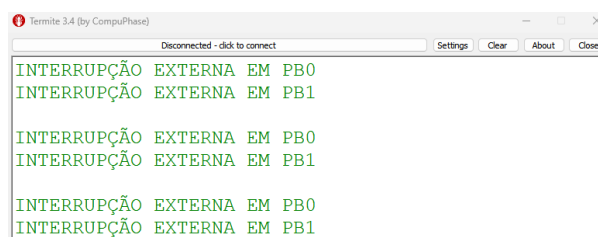
- I. O sistema deve imprimir, a cada segundo, o valor da temperatura em um terminal serial no computador;
- II. Quando a temperatura for menor ou igual a 50 °C, o sistema deve imprimir, além do valor da temperatura, uma mensagem informando que a temperatura está dentro da faixa segura de operação;
- III. Quando a temperatura for maior que 50 °C e menor ou igual a 60 °C, o sistema deve imprimir, além do valor da temperatura, uma mensagem informando que a temperatura está dentro da faixa de atenção. Além disso, o LED D2 deve acender para indicar essa condição;
- IV. Quando a temperatura for maior que 60 °C, o sistema deve imprimir, além do valor da temperatura, uma mensagem informando que a temperatura está dentro da faixa de superaquecimento. Além disso, o LED D3 deve acender e um buzzer deve emitir beeps rápidos para indicar essa condição;

Para evitar flutuações indesejadas na medição da temperatura, utilize o valor médio obtido a partir de **50 leituras** do sensor, homogeneamente distribuídas no intervalo entre as impressões, usando a interrupção de um timer para fazer a aquisição das amostras. Para testar o sistema, jogue ar quente sobre o chip com um soprador térmico ou outra fonte de calor até o limite de temperatura indicado na questão.

☐ 7. Utilizar o pino PA2 para geração de um sinal PWM de 1kHz produzido pelo TIMER5. O sinal será utilizado para controlar a velocidade de um motor DC. O ajuste de velocidade deve ser feito por meio de um joystick analógico. Quando o joystick estiver na sua posição central, o motor deve estar parado. Ao se movimentar o joystick para a esquerda ou direita, o motor deve girar no sentido horário ou anti-horário. Para evitar acelerações bruscas do motor, garanta que, mesmo que o usuário movimente o joystick rapidamente, a aceleração ou desaceleração seja de no máximo 25% da velocidade máxima por segundo.

☐ 8. A função alternativa 2 do pino **PA1** é a saída de comparação do canal 2 do *timer TIM5*. Configure este *timer* e o pino PA1 para geração de um sinal PWM com frequência de **1 Hz** e largura de pulso de **500ms**. Para confirmar o funcionamento, conecte um LED no pino PA1 e observe seu funcionamento (*utilize um resistor de 470Ω para limitar a corrente no LED*).

☐ 9. Conecte o pino **PA1** aos pinos **PB0** e **PB1**, para que estes últimos recebam, simultaneamente, o sinal gerado na questão 1. Configure os pinos PB0 e PB1 como **entradas digitais**, habilite a **interrupção externa pela borda de subida** e configure ambas as interrupções com o maior nível de prioridade possível. Serão disparadas, simultaneamente, duas interrupções a cada segundo, sendo uma correspondente a PB0 e outra correspondente a PB1. Para indicar a execução das rotinas de atendimento, faça com que a ISR de cada interrupção imprima o seguinte texto em um terminal: **"INTERRUPÇÃO EXTERNA EM PBx"**, onde o 'x' deve ser substituído pelo número correspondente ao pino (0 ou 1), conforme está apresentado na figura abaixo.

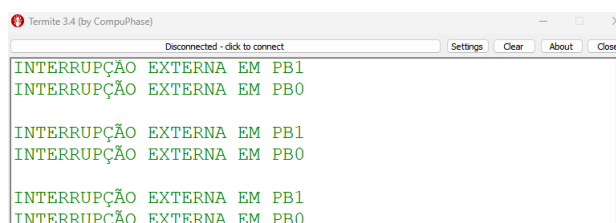


```
INTERRUPÇÃO EXTERNA EM PB0
INTERRUPÇÃO EXTERNA EM PB1

INTERRUPÇÃO EXTERNA EM PB0
INTERRUPÇÃO EXTERNA EM PB1

INTERRUPÇÃO EXTERNA EM PB0
INTERRUPÇÃO EXTERNA EM PB1
```

Em seguida, modifique o **nível de prioridade** das interrupções da questão anterior de forma que as impressões ocorram conforme mostrado na figura abaixo:



```
INTERRUPÇÃO EXTERNA EM PB1
INTERRUPÇÃO EXTERNA EM PB0

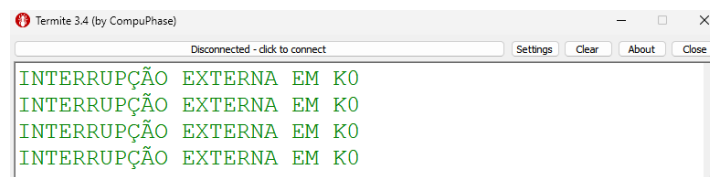
INTERRUPÇÃO EXTERNA EM PB1
INTERRUPÇÃO EXTERNA EM PB0

INTERRUPÇÃO EXTERNA EM PB1
INTERRUPÇÃO EXTERNA EM PB0
```

□ **10.** Escreva um programa que, ao pressionar o botão **K0**, o **LED D2** da placa seja ligado e permaneça ligado por **2 segundos** e, ao pressionar o botão **K1**, o **LED D3** da placa seja ligado e permaneça ligado por **4 segundos**. Ao final dos respectivos períodos, os LEDs devem ser desligados e aguardar um novo pressionamento dos botões para repetir o processo.

Premissas:

I. Os botões devem ser monitorados pelas interrupções externas dos seus respectivos pinos: PE4 para o botão K0 e PE3 para o botão K1, não sendo necessária nem permitida a leitura direta dos pinos. Quando um botão for pressionado, o seguinte texto deve ser impresso em um terminal: “**INTERRUPÇÃO EXTERNA EM Kx**”, onde o ‘x’ deve ser substituído pelo número correspondente ao botão (0 ou 1), conforme está apresentado na figura abaixo.



II. Os períodos durante os quais os LEDs são acionados não devem travar/bloquear o programa.

III. Os períodos de acionamento dos LEDs devem ser controlados por meio de *timers*.

IV. Se o LED D2 já estiver ligado e um novo pressionamento em K0 for feito, um novo intervalo de 2 segundos deve ser iniciado a partir do pressionamento do botão. Por exemplo, se o LED D2 já estiver ligado há 1,5 segundos desde o último pressionamento do botão K0 e o usuário pressionou o botão mais uma vez, o LED só será desligado após 2 segundos deste último pressionamento, ficando ligado por um tempo total de 3 segundos e meio. Em resumo, a cada pressionamento do botão K0, o LED só poderá ser desligado após 2 segundos, independentemente de quando ocorreu o pressionamento, se quando o LED já estava ligado ou se quando o LED estava apagado.

V. O mesmo não deve ocorrer com o LED D3 e o botão K1. Isto é, mesmo que o usuário pressione o botão K1 enquanto o LED D3 estiver ligado, primeiro deverá ser concluído o período de 3 segundos para, em seguida, poder atender novos pressionamentos do botão K1.

VI. Para enfatizar a relevância dos recursos de *timers* e interrupções no paralelismo de ações nos microcontroladores, **AS QUESTÕES 8, 9 e 10**, dessa atividade prática **DEVEM SER EXECUTADAS** simultaneamente em um único programa. ■