Laboratório 1: Programação *Bare Metal* Monotarefa (sem uso de Interrupções)

Objetivo:

Escrever um programa em linguagem C (ou C + Assembly) para o kit EK-TM4C1294XL que realize a função de **decodificador PWM** para um sinal digital externo ao kit (a ser fornecido por um gerador de funções). As especificações para a implementação são:

1) A frequência de operação da CPU do kit de desenvolvimento deverá ser fixada em 24MHz.
2) A medição da largura dos pulsos do sinal externo deverá ser realizada totalmente por software (ou seja, por meio do monitoramento de um terminal de GPIO) e sem o uso de recursos de interrupção;
3) A medição da largura dos pulsos deverá ocorrer tanto para nível alto quanto para nível baixo do sinal externo, permitindo o cálculo do seu ciclo de trabalho, período e frequência;
4) As características medidas (ciclo de trabalho, período e frequência) do sinal externo deverão ser informadas ao usuário por meio da interface serial (UART disponível na porta USB do *debugger* disponível no kit) e um software emulador de terminal (p.ex.: Tera Term) rodando no PC hospedeiro (opcionalmente, pode-se utilizar também o display TFT LCD do Educational BoosterPack MKII ou os displays de sete segmentos do "BoosterPack do Prof. Peron");
5) A programação dos periféricos do kit de desenvolvimento (SCB, PLL, GPIO, UART, display TFT LCD) pode ser realizada utilizando-se a biblioteca de software TivaWare (driverlib, grlib, utils, etc.) ou soluções próprias de cada equipe.

Depois de implementado, o decodificador PWM resultante deve ser avaliado quanto ao seu desempenho. Para tanto, deve-se idealizar um método para determinar os seguintes parâmetros:

- Estatísticas das medidas (média, desvio padrão e erro médio percentual) de sinais de entrada com ciclos de trabalho e frequências conhecidas em 10 amostras (sugestão: ondas retangulares de frequências 10kHz e 10MHz e ciclos de trabalho de 1%, 25%, 50%, 75% e 99%)
- Resolução do tempo nas medidas da solução implementada.

Repetir a avaliação para uma frequência de operação da CPU do kit de desenvolvimento de 120MHz. Qual é o impacto que a frequência de operação tem sobre os parâmetros de desempenho do decodificador PWM implementado dessa forma (software *bare metal* monotarefa)?

Metodologia:

- Estudo/revisão dos conceitos sobre decodificação PWM;
- Elaboração de um diagrama de atividades para planejamento do algoritmo de medição de largura de pulso;
- Implementação do algoritmo planejado;
- Teste e depuração do decodificador PWM, incluindo análises estatísticas dos resultados;
- Apresentação do funcionamento e dos resultados obtidos ao professor.

Cronograma de desenvolvimento (*entregas*):

29/08/2019 (S11) e 30/08/2019 (S12) – Diagrama de atividades da solução planejada.
05/09/2019 (S11) e 06/09/2019 (S12) – Código-fonte parcial desenvolvido até o momento.
12/09/2019 (S11) e 13/09/2019 (S12) – Estatísticas de desempenho (planilha).
19/09/2019 (S11) e 20/09/2019 (S12) – Apresentação e demonstração do funcionamento.

Critérios para as entregas:

Os arquivos das entregas deverão ter os seus nomes codificados da seguinte forma: Sxx_Gyy_Lab1.*, onde Sxx codifica a turma (S11 ou S12) e Gyy codifica a equipe (G01, G02, etc). Entregas deverão ser feitas sempre por e-mail para o endereço hvieir@gmail.com.

1. A entrega do diagrama de atividades da solução planejada deverá ser em formato **pdf** *sem a identificação dos integrantes da equipe* (apenas o nome do arquivo Sxx_Gyy_Lab1.pdf vinculará o trabalho à equipe).
   O diagrama a ser elaborado deverá levar em conta as seguintes diretrizes:
   - O detalhamento do algoritmo deverá ser equilibrado, não sendo genérico demais a ponto de não fornecer nenhum detalhe de implementação, nem específico demais a ponto de praticamente corresponder ao código-fonte a ser implementado.
   - O detalhamento deverá ser maior em aspectos mais relevantes da implementação, aqueles que serão cruciais para o funcionamento desejado, inclusive em termos de desempenho, para a implementação.
   - Deverá ser utilizada a notação UML 2.x (ver na próxima página alguns dos símbolos gráficos disponíveis).

   Sugestão de material de consulta sobre diagramas de atividades:
   - https://www.lucidchart.com/pages/uml-activity-diagram

   Sugestão de ferramentas na nuvem para elaboração dos diagramas:
   - https://www.draw.io/
   - http://www.umlet.com/umletino/umletino.html

2. A entrega do código-fonte da implementação parcial deverá preferencialmente ser em um repositório no GitHub contendo a pasta *completa* do projeto, cujo link deverá ser informado por e-mail. Os nomes dos integrantes devem constar em comentários logo no início dos arquivos de código-fonte de autoria da equipe. Alternativamente, a entrega do código-fonte poderá ser feita em formato **zip** ou **rar** (Sxx_Gyy_Lab1.zip ou Sxx_Gyy_Lab1.rar) – nesse caso, o arquivo compactado também deve conter a pasta *completa* do projeto.

3. A entrega das estatísticas de desempenho (planilha) deverá ser em formato **xslx** ou **ods** *com a identificação dos integrantes da equipe* (nomes dos integrantes devem constar do arquivo Sxx_Gyy_Lab1.xlsx ou Sxx_Gyy_Lab1.ods). **Importante**: a planilha contendo as estatísticas de desempenho deverá seguir o modelo disponível para download no website da disciplina.

4. A entrega do código-fonte da implementação final deverá seguir os mesmos moldes da entrega do código-fonte da implementação parcial previamente entregue. **Importante**: a estrutura do projeto a ser entregue deve seguir as instruções para configuração de novos projetos no ambiente IAR EWARM fornecidas anteriormente – o projeto a ser entregue deve ser elaborado a partir do *zero*, inclusive com nome diferente dos fornecidos pelo professor.

Bônus:

Os integrantes da equipe cuja implementação apresentar melhor desempenho (maior exatidão e estabilidade nas medidas) receberão um bônus de 0,5 ponto na nota final da disciplina (sujeito às condições estabelecidas anteriormente).

Notação (parcial) de um diagrama de atividades: