EA871 – LAB. DE PROGRAMAÇÃO BÁSICA DE SISTEMAS DIGITAIS

EXPERIMENTO 5 – Primeira Biblioteca em C

Profa. Wu Shin-Ting

OBJETIVO: Apresentação de uma forma de organização modular e documentação de códigos em linguagem C.

ASSUNTOS: Construção de uma biblioteca estática e documentação dos códigos.

O que você deve ser capaz ao final deste experimento?

Construir uma biblioteca estática no ambiente IDE CodeWarrior.

Construir um projeto usando uma biblioteca no ambiente IDE CodeWarrior.

Documentar a interface do seu código com uso de Doxygen.

INTRODUÇÃO

Duas características desejáveis de um *software* são **modularidade** e **reusabilidade**. Modularidade consiste em divisão de um programa em trechos de códigos com funções bem definidas e a mais independente possível em relação ao resto dos códigos. Isso aumenta a confiabilidade, a legibilidade e a manutenção e a flexibilidade de uso dos códigos. Reusabilidade diz respeito à reutilização de códigos desenvolvidos em diferentes contextos, de forma a reduzir os custos de manutenção e de produção de *software*. Os dois conceitos, modularidade e reusabilidade, são intimamente relacionados. Quanto mais modulares são os aplicativos, mais fácil é a identificação de pontos comuns entre eles, mesmo que eles sejam destinados a finalidades bastante distintas.

Ao invés de implementar uma mesma função em diferentes versões, um desenvolvedor pode se beneficiar do reuso dos códigos amplamente testados, dedicando-se o seu tempo à implementação e aos testes de novas funções. Para facilitar a reusabilidade, é comum organizar os módulos de códigos de reuso em bibliotecas. As **bibliotecas** podem ser **estáticas**, as ligadas estaticamente para gerar um arquivo executável que inclui todos os códigos de reuso, ou **dinâmicas**, as ligadas dinamicamente para gerar um arquivo executável que contém somente referências às funções de reuso. Neste caso, embora os códigos de reuso sejam compartilhados entre os códigos executáveis, eles precisam ser carregados e ligados no tempo de execução – uma tarefa executada usualmente pelo sistema operacional [1]. Como não temos um sistema operacional em tempo real (*Real Time Operating System*, RTOS) [2] instalado nos nossos nossos microcontroladores, implementar funções de carga e ligação dinâmicas é trabalhoso. Portanto, usaremos nesta disciplina somente bibliotecas estáticas de extensão .a. Faz parte dos utilitários binários (Binutils) do GNU a ferramenta **ar** que constrói, a partir de um conjunto de códigos de máquina, de extensão .o, uma biblioteca estática[3]. Vale ressaltar que, para construir um código executável reusando as funções de uma biblioteca, é necessário que os códigos de máquina contidos nas bibliotecas sejam compatíveis com a arquitetura do processador em que o arquivo-alvo vai ser executado.

É importante frisar que disponibilizar os módulos de códigos em formato de bibliotecas sem documentação não assegura a sua reusabilidade, porque para usar as funções de forma correta precisamos saber não só o quê elas fazem como também a sua interface com outros programas. Em

linguagem C, esta interface, denominada o **protótipo**, ou **declaração**, de **função**, compreende o nome de uma função, os tipos e a ordem dos seus argumentos, e o tipo de seu retorno. Uma documentação da interface das funções de uma biblioteca é, portanto, imprescindível [4].

Documentar códigos não é uma tarefa trivial [4]. Felizmente, é disponível uma série de ferramentas que dão suporte a uma documentação de qualidade profissional, como Doxygen [5]. Pela sua popularidade, Doxygen é um padrão-de-fato usado para geração de documentos de interface dos principais aplicativos livres. Ele suporta uma grande gama de linguagens de programação e gera documentos de diferentes formatos, inclusive em formato html que pode ser aberto por um navegador. A lista de palavras-chave interpretadas pelo aplicativo estão listadas em [6]. Todas as palavras-chave devem ser precedidas por uma barra invertida (\) ou por arroba (@). Todos os comentários adicionais que não seguem a sintaxe de Doxygen são ignorados por ele. Como a ênfase de Doxygen é na documentação da interface das funções de um aplicativo, é comum inserir os códigos de documentação nos arquivos-cabeçalho, de extensão .h.

EXPERIMENTO

Neste experimento vamos construir um projeto modular, e adequar a sua versão do projeto hello_FRDMKL25 implementada no experimento [4] para uma versão modeular e documentada.

1. Complete a implementação de todas as funções disponíveis nos arquivos GPIO_ledRGB.*.
2. Adicione o arquivo GPIO_ledRGB.h na sub-pasta Project_Headers e o arquivo GPIO_ledRGB.c na sub-pasta Sources, como no projeto **mod_FRDMKL25Z**. Utilize a função “Add Files” do ambiente IDE (selecione a pasta, clique no botão direito do *mouse* e no *pop-up menu* selecione o item *Add Files*…) para que o IDE entenda que são os arquivo do projeto e processá-los como parte do projeto.
3. Adicione três funções documentadas aos arquivos GPIO_ledRGB.h e GPIO_ledRGB.c: GPIO_ledRG(estado) (ligue e desligue *led* R e *led* G), GPIO_ledGB(estado) (ligue e desligue led G e led B), e GPIO_ledBR (estado) (ligue e desligue *led* B e *led* R).
4. Adapte as chamadas das funções da sua versão do projeto hello_FRDMKL25 para as novas funções, de forma que as configurações dos registradores do microcontrolador fiquem transparentes na função **main** do projeto hello_FRDMKL25. Não é mais necessário incluir “derivative.h” no arquivo main.c. Somente a inclusão de “GPIO_ledRGB.h” no arquivo main.c é suficiente para ter acessos aos registradores.
5. Documente todas as funções no arquivo GPIO_ledRGB.h.

RELATÓRIO

Para este experimento, submeta a versão mudular e documentada (os arquivos que você criou) do projeto hello_FRDMKL25, exportada no ambiente IDE CodeWarrior, ao sistema *Moodle*. **Não se esqueçam de limpar o projeto (*Clean* …) antes e apagar a pasta html gerada pelo Doxygen**.

REFERÊNCIAS

[1] Introdução às Bibliotecas em C
http://mindbending.org/pt/introducao-bibliotecas-em-c
[2] Usando FreeRTOS para aplicações com a FRDM KL25Z
https://www.embarcados.com.br/freertos-nxp-frdm-kl25z/
[3] GNU ar
https://sourceware.org/binutils/docs/binutils/ar.html
[4] A beginner's guide to writing documentation
https://www.writethedocs.org/guide/writing/beginners-guide-to-docs/
[5] Doxygen
https://www.doxygen.nl/index.html
[6] Doxygen: Special Commands
https://www.doxygen.nl/manual/commands.html
[7] Experimento 4
http://www.dca.fee.unicamp.br/cursos/EA871/2s2020/ST/roteiros/roteiro4.pdf

Setembro de 2020