**Lista de Exercícios #2** 

Pado Labs - Microcontroladores

Registradores e IOs

*Tips and Tricks* : Utilizar os documentos do kit para resolver os exercícios. *Requirements* : Resolva pelo menos 7 exercícios. Exercícios com a *tag* **Challenge** valem por dois.

*Requirements* : Exercícios que requerem desenvolvimento de um código deve ser separado e zipado, com título do projeto e nome do arquivo fácil de identificar como *Lista2-Ex2.rar*, *L2-E2.rar*.

**Luan Marçal - programmer of the future.**

**1:** Qual a vantagem de se trabalhar com os tipos da biblioteca *stdint.h*

**R:** A vantagem é poder definir os valores de variáveis do tipo inteiro e poder padronizar os tamanhos alocados.

**2:** Qual a principal característica de uma viariável do tipo *int\_fastX\_t*?

**R:** Essa função define o tamanho alocado de uma variável parte ter a melhor performance possível no microcontrolador.

**3:** No nosso kit NUCLEO-G0B1RE, qual o tamanho da variável, em bytes, do int\_fast8\_t, int\_fast16\_t, int\_fast32\_t e int\_fast64\_t.

**R:** int\_fast8\_t = 4 bytes

int\_fast16\_t = 4 bytes

int\_fast32\_t = 4 bytes

int\_fast64\_t = 8 bytes

**4:** Qual a função dos registradores:

• **GPIOx\_MODER =** Definir o modo de I/O do microcontrolador.

• **GPIOx\_OTYPER =** Define o tipo de saída I/O

• **GPIOx\_OSPEEDR =** Define velocidade de saída I/O

• **GPIOx\_PUPDR =** Define I/O como pull up ou pull down

• **GPIOx\_IDR =** É um registrador de leitura que contém o valor de entrada das portas I/O.

• **GPIOx\_ODR =** Registro de valor de leitura e escrita de dados, mantém o valor de reset reservado.

• **GPIOx\_AFRL =** Configura I/O paralelamente aos outros registradores.

**5:** Como posso fazer para ler diretamente o registrador sem utilizar a implementação da ST? (*tip*: lembre-se dos ponteiros!)

**R:** Pode-se acessar o endereço do registrador e modificar o valor do bit para a função desejada.

**6:** Desenvolva um firmware que pisque o LD4 com uma frequência de 1Hz (500ms aceso, 500ms apagado)

**R:** Anexo.

**7:** Desenvolva um firmware que pisque o LD4 com uma frequência de 100Hz.

**R:** Anexo.

**8:** Faça um programa que pisque o LD4 em 20Hz enquanto o botão USER é pres sionado e pisque com frequêcia de 5Hz ao ser solto.

**R:** Anexo.

**9:** Faça a leitura dos *switches* do tipo DIP de 4 posições utilizando os resistores de *pull-up* ou *pull-down* internos. Armazene em uma variável o valor correspondente, onde a chave 1 corresponde ao bit 0, e o bit 4 corresponde ao bit 3.

**R:** Anexo.

**10:** Aproveite o exercício anterior, monte 4 LEDs e associe cada um deles a uma chave do DIP switch de 4 posições, quando a chave estiver em ON, acenda o LED, e em OFF, apague o LED.

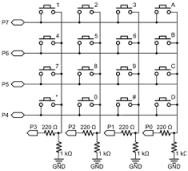
**R:** Anexo.

**11: Challenge**: Aproveite o exercício anterior novamente, mas sem os LEDs, e exiba em um display de 7 segmentos o valor correspondente em hexadecimal (0 à F).

**R:** Anexo.

**12: Challenge**: Existe uma técnica comumente chamada de *Varredura*, esta téc nica consiste em ligar elemento em matriz para otimizar o uso de GPIOs, muito utili zada para acionar LEDs e realizar a leitura de botões e *keypads*. Nisto, como desafio, deve-se montar o circuito da figura 1 e desenvolver um firmware que faça a leitura dessas teclas e armazene em uma variável a linha e coluna da tecla pressionada (a li nha e coluna deve ser numerada de 1 a 4, quando nenhuma tecla estiver pressionada, deve ser exibido o valor 0).

Figura 1: Esquemático de uma matriz de botões 4x4.



**R:** Anexo/Simulador.