

Exercício 1

- Descubra qual nível de privilégio está sendo usado em Modo Thread. Privilegiado ou não privilegiado?

R: Como o BIT 0 (nPriv) do Registrador CONTROL está com o valor 0, o Modo Thread que está sendo utilizado é privilegiado.

- Descubra qual ponteiro de pilha está sendo usado em Modo Thread. Main Stack Pointer ou Process Stack Pointer?

R: Como o BIT 1 (SPSEL) do Registrador CONTROL está com valor 1, a pilha PSP (Process Stack Pointer) está sendo utilizada.

- Descubra quais são os valores numéricos das prioridades das exceções usadas no RTX 5: SVC, PendSV e SysTick.

R:

SVCall utiliza do PRI_11 dentro de SHPR2, que neste exercício tem valor de 0xc, que neste caso tem prioridade 12

PendSV utiliza do PRI_14 dentro de SHPR3, que neste exercício tem valor de 0xe, que neste caso tem prioridade 14

SysTick utiliza do PRI_15 dentro de SHPR3, que neste exercício também tem valor de 0xe, que neste caso tem prioridade 14

Logo neste exercício as exceções PendSV e SysTick, tem prioridade igual e maior que a SVC.

Exercício 2

-Localize o arquivo RTX_Config.h na lista de dependências do arquivo rtx_lib.c e analise-o. Quais são as configurações para:

– OS_TICK_FREQ?

R: Está definido como 1000, o que quer dizer que o tick leva 1ms.

– OS_THREAD_NUM?

R: Está definido como 1, o que quer dizer que só é possível rodar 1 thread ao mesmo tempo.

– OS_THREAD_DEF_STACK_NUM?

R: Está definido como 1, o que quer dizer que apenas 1 thread terá o tamanho padrão se não for passado o tamanho da thread por parâmetro. No caso deste exemplo, como são poucas threads não terá problemas maiores, mas poderia ser a causa de problemas de alocação de memória para as threads. O valor correto deveria ser igual ao total de threads que se tem no sistema.

Altere a configuração OS_TICK_FREQ para 500.

– Qual é o efeito dessa alteração na execução do programa do projeto “tarefas”?

R: Como a frequência ficou menor o tempo de cada tick aumentou, assim os leds demoram mais para acender e ficam mais tempo acesos.

Retorne a configuração OS_TICK_FREQ para o seu valor original.

– Quais são os períodos de ativação das tarefas thread1 e thread2?

R: Para a thread2 o período de ativação é igual a 100, valor que está colocado dentro da função osDelayUntil. Para a thread1 o período de ativação é igual ao tempo de computação mais o tempo de atraso, porém para este exemplo o tempo de computação é desprezível, tendo em vista a complexidade da tarefa a ser executada, logo o período de ativação se torna igual ao tempo de atraso, que neste caso é 100, valor colocado dentro da função osDelay.

Modifique o código-fonte no arquivo tarefas.c para que duas instâncias de uma mesma tarefa sejam usadas para acionar o LED D1 e o LED D2 do kit de desenvolvimento.

R: Respondido no código.

Modifique novamente o código no arquivo tarefas.c para que seja possível informar às instâncias da tarefa o LED a ser acionado e também o seu período de ativação.

R: Respondido no código.

Crie quatro instâncias da nova tarefa, uma para acionar cada LED do kit, com períodos diferentes: LED D1 = 200ms, LED D2 = 300ms, LED D3 = 500ms, LED D4 = 700ms.

R: Respondido no código.