EA871 – Laboratório de Programação Básica de Sistemas Digitais

Atividade 08

Profs. Levy Boccato, Rafael Ferrari e Tiago Tavares – 2º semestre de 2019

1. Objetivos

- Contrastar o funcionamento de soluções para contagem de tempo baseadas em espera ativa e em interrupção de temporizador.

2. Atividade da Aula (faremos em aula, juntos)

Nosso objetivo é piscar o LED incorporado à placa, conectado no pino 13, usando o temporizador 2 para controlar os intervalos de tempo que o LED permanece aceso e apagado.

a) Determine os valores a serem atribuídos aos registradores TCCR2A, TCCR2B e TIMSK para que o temporizador opere no modo normal, ou seja, contagem até o valor máximo (255), *prescaler* = 8 e interrupção por *overflow* ativa. Nessa configuração, qual é o intervalo de tempo entre as interrupções?

b) Utilize o temporizador 2, configurado conforme especificado no item anterior, para piscar o LED do pino 13. O LED deve permanecer 256 ms em cada estado. Construa o programa e valide-o no Arduino.

c) Simule o funcionamento do sistema usando o Tinkercad.

3. Resumo da Atividade (individual, para entrega)

O desafio proposto nesta atividade é desenvolver um programa que faça com que dois LEDs pisquem em diferentes frequências, ao mesmo tempo em que uma mensagem de texto é continuamente transmitida pela UART. Mais especificamente, desejamos que o LED incorporado pisque com uma frequência de 2 Hz (0,5 s aceso, 0,5 s apagado), e que um LED externo, conectado ao pino 12 da placa Arduino, pisque de forma a ficar 0,78 s aceso e 0,78 s apagado.

A solução para a atividade deve necessariamente explorar a interrupção do *timer* 2 associada ao modo *Clear Timer on Compare Match* (CTC). As especificações do temporizador (e.g., *prescaler* e valor máximo da contagem) devem ser projetadas por cada aluno e justificadas nos comentários do código-fonte. Notem que há várias maneiras de se configurar o temporizador para obter o comportamento desejado do sistema.

No tocante à UART, vamos utilizar a mesma configuração da atividade 6. Além disso, assim como feito anteriormente, após a mensagem de texto ser enviada uma vez, deve ser introduzido um atraso de 5 segundos antes de reiniciar as transmissões. Apenas pedimos que todos substituam o vetor de caracteres a ser transmitido pela seguinte mensagem:

```
char msg[] = " Atividade 8 - Interrupcoes temporizadas tratam concorrencia entre tarefas! \n";
```

Observação: uma abordagem baseada somente em espera ativa (i.e., que usa rotinas de atraso para gastar tempo) encontraria algumas dificuldades para manter os vários dispositivos (LEDs e UART) simultaneamente em operação, dada a concorrência existente entre as tarefas.

Instruções para a submissão do trabalho

1) Nos comentários do código-fonte (main.c), justifique as operações e os valores carregados em todos os registradores. **Em especial, como há diferentes estratégias de uso do temporizador para se atingir o intervalo de tempo desejado, é fundamental que a opção feita por cada aluno seja explicada nos comentários logo no início do código-fonte, de modo a justificar as configurações do temporizador.**

2) Crie um projeto chamado ‘atividade8’ (letras minúsculas, sem espaço) no Atmel Studio e, ao final da atividade, salve o diretório completo em um arquivo no formato .zip (Aviso: não use .tar.gz nem .rar), com nome ‘seu_ra.zip’ (Exemplo: 025304.zip).
3) Faça o **upload** da sua solução da atividade 8 no Google Classroom.