Atividade Prática Intermediária 1

Maria Luiza Fontes Dantas
Estudante de Engenharia Mecatrônica - UFSC
Matrícula: 19103637

I - Introdução

O código exemplificado abaixo foi desenvolvido como atividade da disciplina de Microcontroladores com o objetivo de praticar a implementação de mecanismos aprendidos em aula e observar sua utilização.

O objetivo da atividade é gerar um sinal periódico (Pulso Rede) com uma frequência de 120Hz, seu início de contagem deve ser ativado por um dos botões. Um outro sinal (Pulso Triac) também deve ser gerado, esse é acionado juntamente com o Rede com o pressionamento do botão e pode ser controlado por um segundo botão. Uma variável de nome tempo_triac deve armazenar o tempo decorrido entre o Pulso Rede e o Pulso Triac. Toda vez que o segundo botão for pressionado, o Pulso Triac deve ser atrasado em 10% do tempo máximo que a variável tempo_triac pode armazenar, com no máximo 100% de atraso.

II - Implementação

Em suma, para atender o objetivo proposto pela atividade, foram utilizados 5 Timers e Interrupções em ambos os Botões do Tiva. Um diagrama de blocos foi montado para representar a solução produzida.

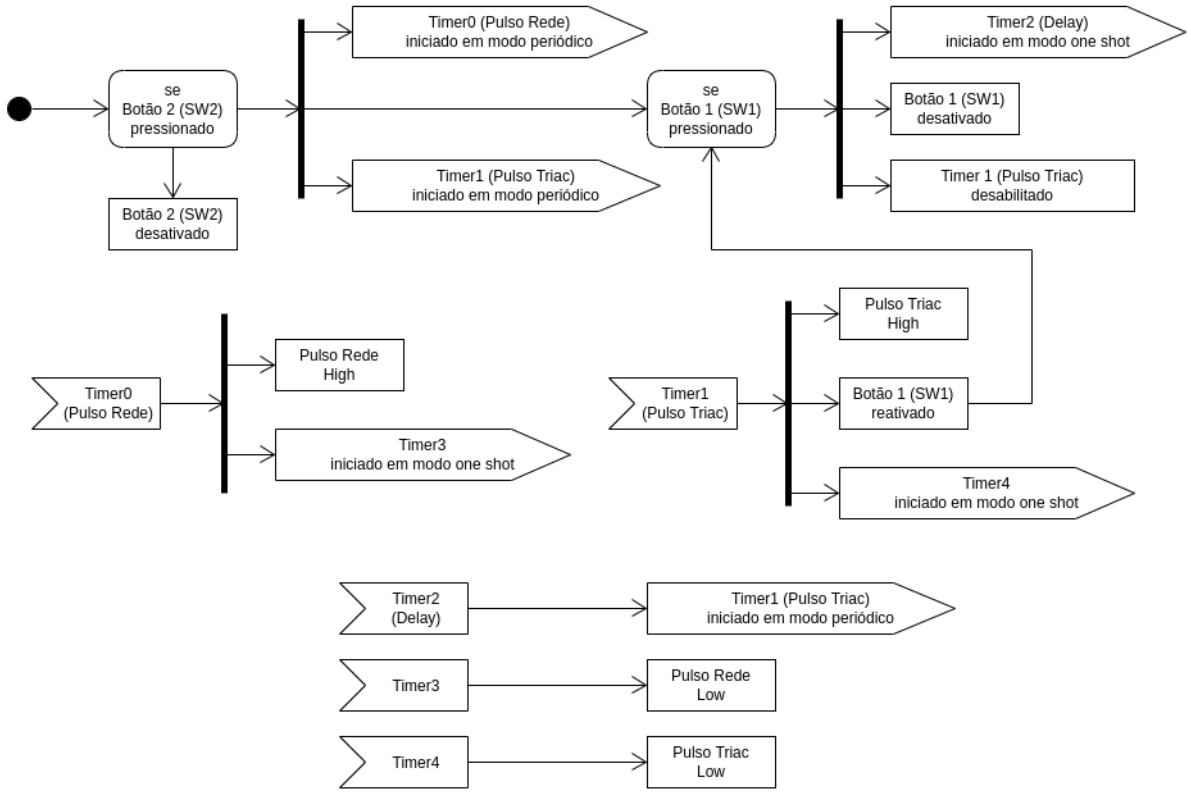

Diagrama de blocos

Como é possível ver no diagrama de blocos, o programa conta com Timers para gerar o acionamento e descida dos Pulsos Rede e TRIAC. Os Timers 0 e 1 são periódicos e controlam o acionamento dos pulsos em uma frequência de 120Hz, enquanto os Timers 3 e 4 são one shot e garantem a descida dos Pulsos após decorrido o tempo de 100us. Além desses, o Timer 2 é configurado como one shot e serve para gerar gerar delay no Pulso TRIAC.

Dentro do programa, a variável tempo_triac armazena o tempo decorrido entre o acionamento do Pulso Rede e o acionamento do Pulso TRIAC. Quando o atraso estiver em 100% do tempo_triac máximo, este será igual ao período do Pulso Rede e que quanto esse atraso ocorrer os Pulsos estarão síncronos, o Pulso TRIAC com uma ocorrência a menos que o Pulso Rede.

Também é possível perceber no diagrama a utilização de 2 botões para controlar a sincronia e assincronia dos pulsos. O botão 2 (SW2) habilita e aciona os Timers 0 e 1 e consequentemente os Pulsos Rede e TRIAC, inicialmente em sincronia e com frequência de 120Hz, não é necessário pressionar o botão SW2 mais de uma vez na execução do programa. Já o botão 1 (SW1) promove a assincronia dos pulsos através da geração de delay no Pulso TRIAC, cada vez que ele é pressionado, o Timer 1 (Pulso TRIAC) é desabilitado e o Timer 2 é acionado com duração de 10% do tempo_triac máximo, depois de sua execução, o segundo timer habilita novamente o Pulso Triac, garantindo um atraso dele em relação ao Pulso Rede. Quando o botão é pressionado 10 vezes, um delay de 100% do tempo_triac máximo é gerado, promovendo novamente a sincronia dos pulsos, se pressionado mais uma vez, todo o processo é reiniciado com os pulsos em sua sincronia inicial.

Cada vez que o botão 1 é acionado, a sua interrupção é desabilitada, sendo reabilitada apenas no Timer que controla o Pulso TRIAC, de maneira a gerar um debouncer no SW1, garantido que a contagem de pressionamentos seja precisa.

III - Testes

Com o objetivo de visualizar os pulsos propostos na atividade, os leds vermelho e azul do próprio Tiva foram adicionados ao programa. O led azul fica aceso enquanto o Pulso Rede está em High e o vermelho enquanto o Pulso TRIAC está em High. Dessa maneira é possível distinguir visualmente quando os pulso estão em sincronia ou assincronia.

Além disso, a frequência dos pulsos foi diminuída para 1 Hz para facilitar a análise do código em vídeo, já que ficaria difícil acompanhar 120 acionamentos do led por segundo sem ferramentas para desacelerar a gravação. Para facilitar ainda mais a análise, o tempo que os pulsos ficam em High foram aumentados para 100ms, facilitando a visualização dos leds.

Em adição, um visualizador da variável contador (que armazena quantas vezes o botão 1 foi pressionado) foi adicionado ao Code Composer Studio para melhor controlar a quantidade de delays que foram aplicados no Pulso TRIAC.

Feitas as mudanças, o código foi carregado no Tiva e testado algumas vezes. Ao pressionar o botão 2, o sistema realmente iniciou em sincronia e saiu desse estado com o primeiro pressionamento do botão 1. A medida que o SW1 foi acionado mais vezes, foi possível perceber que o tempo decorrido entre o acionamento do Pulso Rede (led azul) e do Pulso TRIAC (led vermelho) ia aumentando, até que ficaram bem próximos quando o atraso de 100% se aproximou, quando ocorreu o 10º acionamento os leds voltaram a sincronia. No 11º pressionamento o processo foi reiniciado e o 12º teve o mesmo efeito do primeiro e assim por diante.

IV - Conclusão

Após implementação e teste foi possível perceber que os Timers são muito úteis para controlar ações periódicas, como os Pulsos Rede e TRIAC, e ações com tempo definido, como o delay e a descida dos pulsos, permitindo que tudo isso seja feito sem precisar prender o programa em alguma função com o mesmo objetivo.

As interrupções implementadas para detectar o acionamento dos 2 botões também são muito úteis para que os pressionamentos sejam detectados corretamente, de maneira a evitar que dados sejam perdidos por conta do código não estar verificando o acionamento no momento em que ele ocorre.

Além disso, a utilização do debouncer se mostrou bastante eficiente em não confundir 1 acionamento com 2 ou não detectá-lo, diferentemente de código escritos no início do semestre sem essa ferramenta. Através do uso do debouncer, todos os sinais enviados pelos botões foram perfeitamente capturados, garantindo o funcionamento correto do código.

Em suma, a atividade possibilitou o estudo da implementação dos conceitos aprendidos em aula e a visualização dos seus benefícios para o código.

V - Vídeo

O vídeo com a explicação da implementação foi postado no youtube:
https://youtu.be/bYRIoU9UXw8