Universidade Federal de Minas Gerais
Escola de Engenharia – Departamento de Engenharia Eletrônica
Laboratório de Arquitetura e Organização de Computadores
Assunto: Projeto de um módulo de contadores e *timers* programáveis
Autoria: Prof. Ricardo de Oliveira Duarte

Objetivos:

Explicar o funcionamento de um módulo de contadores e *timers* programáveis.
Apresentar os passos necessários ao projeto de um módulo de contadores e *timers* programáveis.

Seção 1: Funcionamento de um módulo de contadores e *timers* programáveis:

O componente básico desse módulo é um contador binário síncrono, que chamaremos de COUNTx, onde x se refere ao número de identificação do componente, pois esse módulo pode conter um ou mais desses componentes programáveis. A programação desse componente consistirá em definir o valor da condição de parada da contagem que deverá ser guardada em um registrador específico e programável pelo usuário, a esse registrador chamaremos STOPx, além da origem do sinal de *clock* de COUNTx, que poderá vir de um pino de entrada externo sensível a transição de subida ou sensível a transição de descida, ou que poderá vir de um *bit* qualquer de um divisor de frequências construído a partir de um outro registrador contador binário síncrono, que chamaremos de DIVIx, cujo o sinal de *clock* será o de mesma frequência usado pelo FPGA do kit DE10-Lite, ou seja f=50 MHz. A configuração da origem do *clock* para o registrador COUNTx deverá ser guardada no registrador CONFIGx. Quando a origem do *clock* de COUNTx vier de qualquer *bit* de DIVx o componente funcionará como temporizador, do contrário funcionará como um contador de eventos externos (por exemplo: quantidade de transições de subida ou quantidade de transições de descida). Para cada componente COUNTx deverá haver uma saída própria reservada unicamente para tal sinalização da condição de fim de contagem. Essa saída deverá ser ativada toda vez que a condição de igualdade entre os registradores COUNTx e STOPx for alcançada. Cada componente desse módulo pode ser configurado para gerar ou não uma interrupção. Portanto, você deverá associar a cada componente uma posição (*bit*) nos registradores IER e IFS do módulo da controladora de interrupções que você já projetou, assim como deverá destinar um endereço para a ISR de cada periférico desse módulo. Toda vez que for necessário configurar, ou zerar (*reset* síncrono) qualquer registrador de qualquer componente desse módulo, o *bit* reservado para habilitar a atenção à interrupção desse componente deverá ficar desabilitado. Após qualquer leitura do registrador COUNTx, o componente deverá ser zerado (*reset*).

Seção 2: Projeto de um módulo de contadores e *timers* programáveis:

Passo 1: Defina quantos componentes você necessitará para esse módulo.

Passo 2: Faça um desenho completo do módulo com todos os componentes interligados.

Passo 3: Adapte o módulo controlador de interrupções para atender a demanda de interrupções necessária esse seu novo módulo.

Passo 4: Projete em VHDL o módulo desenhado no passo 2.

Passo 5: Compile o seu módulo. Corrija eventuais erros de compilação.

Passo 6: Crie um *testbench* para testar o seu módulo. Simule o seu sistema com o *testbench*.

Passo 7: Escreva um programa em linguagem C que configure esse novo módulo.

Passo 8: Converta o programa que você escreveu no passo anterior para a linguagem *assembly* da sua CPU.

Passo 9: Converta o programa que você escreveu no passo anterior para a linguagem de máquina.

Passo 10: Compile a sua CPU com o novo módulo e as modificações realizadas na controladora de interrupções. Corrija eventuais erros de compilação.

Passo 11: Teste o seu sistema no kit DE10-Lite.

Passo 12: Meça no osciloscópio o tempo exato medido pelo timer na aplicação executada pela sua CPU.