Trabalho prático 7

Neste trabalho prático deverá implementar um contador horário mm.ss (*countdown timer*), com contagem decrescente, com um comportamento do ponto de vista do utilizador semelhante ao do trabalho prático 6, em que:

- o controlo, a atualização temporal e a interface com o utilizador continuam a ser realizadas por software a executar sobre um SoC baseado no processador MicroBlaze (tal como no trabalho prático 6).

- mas a interface com o *display* passa a ser realizada através de um periférico de hardware especializado, o qual controla a atualização/refrescamento periódico dos dígitos, tendo o software que interagir com esse periférico apenas para alteração dos valores mostrados (pelo que o GPIO associado ao *display* tem de ser removido do projeto).

Desta forma, pretende-se libertar o software da tarefa de refrescamento periódico do *display*, passando esta operação para hardware dedicado.

O periférico especializado deverá integrar o módulo "Nexys4DisplayDriver" desenvolvido no trabalho prático 3.

O seu modelo de programação deverá consistir em diversos registos mapeados no espaço de endereçamento do processador, que disponibilizem ao software os seguintes campos:

- *Digit Enables* - 8 bits para (des)ativação individual de cada dígito;

- *Decimal Point Enables* - 8 bits para (des)ativação individual de cada ponto decimal;

- *Digit Values* - 32 bits (8 x 4 bits) para configuração, em binário, do valor afixado em cada dígito;

Parte 1 - Adaptação e extensão da plataforma de hardware

1. Defina o modelo de programação do periférico especializado (registos necessários, respetivos offsets e organização dos campos).

2. Crie uma cópia da plataforma base, para o desenvolvimento de sistemas baseados em MicroBlaze (para tal deverá, num novo projeto, importar o *Block Design* da plataforma base).

3. Adicione um periférico especializado com interface AXI4-Lite (*memory mapped*).

4. Integre o módulo "Nexys4DisplayDriver" no periférico especializado de acordo com o modelo de programação definido, assim como outro(s) módulos(s) de suporte necessários (e.g. gerador de *clock enable*).

No final o Block Design deverá ser semelhante ao seguinte (incluindo o nome dos módulos): block design no Vivado.

Parte 2 - Desenvolvimento do software de controlo baseado em *polling* ao registo de estado do *timer* de hardware (*AXI Timer*)

1. Desenvolva o software de acordo com o esqueleto fornecido (Trab7_part2_Skel.c).

2. Teste adequadamente o sistema.

3. Conceba uma forma de avaliar qualitativamente / constatar empiricamente que o periférico especializado contribuiu para aliviar o peso computacional do software.

Parte 3 - Desenvolvimento do software de controlo baseado em interrupções do *timer* de hardware

1. Repita a parte 2, mas usando agora interrupções do *timer* de hardware (AXI Timer ou FIT) - de notar que neste caso não é fornecido nenhum esqueleto do software, pelo que terá de usar como ponto de partida o código C da parte 2 do trabalho prático 6.

Parte 4 - Extensão das funcionalidades do periférico especializado

1. Adicione ao módulo "Nexys4DisplayDriver" a lógica necessária para suportar as seguintes entradas e correspondentes funcionalidades (opções de configuração):

- *Refresh Rate* - 3 bits para configuração da taxa de atualização do display (50 Hz, 100 Hz, 200 Hz, 400 Hz, 800 Hz, 1600 Hz, 3200 Hz, 6400 Hz) - para este efeito, recomenda-se que o "Nexys4DisplayDriver" deixe de ter a entrada de "*enable*", passando esta a um sinal interno que deve ser ativado em função da taxa de atualização do display

- *Display Brightness* - 3 bits para controlo da luminosidade do display (8 níveis).

Nota: como sempre, é fundamental que todo o sistema utilize um único sinal de clock (neste caso o sinal de clock da interface AXI).

2. Altere o modelo de programação do periférico de forma a suportar as novas funcionalidades.

3. Adapte o software de forma a tirar partido das novas funcionalidades implementadas no hardware.

4. Teste adequadamente o sistema.