deti universidade de aveiro
departamento de electrónica, telecomunicações e informática

Relatório
Projeto Final

Tiago Sousa Fonseca | Mec. 107266
Tomás Sousa Fonseca | Mec. 107245

Laboratório de Sistemas Digitais

Introdução

Este trabalho tem como objetivo a modelação em ***VHDL*** e a testagem em ***FPGA*** de um marcador de Ténis. Este marcador tem como função contabilizar e registar os pontos inseridos pelo utilizador, os jogos ocorridos e os sets, através dos seus *inputs*, refletindo a ação do mesmo através de *outputs*, para que possa existir uma interação utilizador/máquina.

- *Inputs*

  O registo e contabilização de pontos é feita através da interação do utilizador com a máquina através da ***KEY[3]*** (Ponto do Jogador B) e ***KEY[0]*** (Ponto do Jogador A).
  No final de cada set o utilizador disponibiliza de um botão ***KEY[1]*** (Next Set) que lhe permite avançar para o próximo set, dando por encerrado o presente set.

- *Outputs*

  O utilizador dispõe de **6 displays** de **7 segmentos** para que possa visualizar a sua pontuação e o número de jogos ganhos no set em curso.
  - A pontuação pode ser visualizada através dos displays ***HEX[7]*** e ***HEX[6]*** (Pontuação do Jogador A), e através dos displays ***HEX[5]*** e ***HEX[4]*** (Pontuação do Jogador B);
  - Os jogos ganhos no set em curso são disponibilizados pelos displays ***HEX[1]*** (Jogos Ganhos do Jogador A) e ***HEX[0]*** (Jogos Ganhos do Jogador B).

Arquitetura

A arquitetura do Marcador de Ténis é representada através do seguinte diagrama de blocos (**Figura 1**), onde os principais blocos serão explorados com um maior grau de detalhe no capítulo **Implementação**.

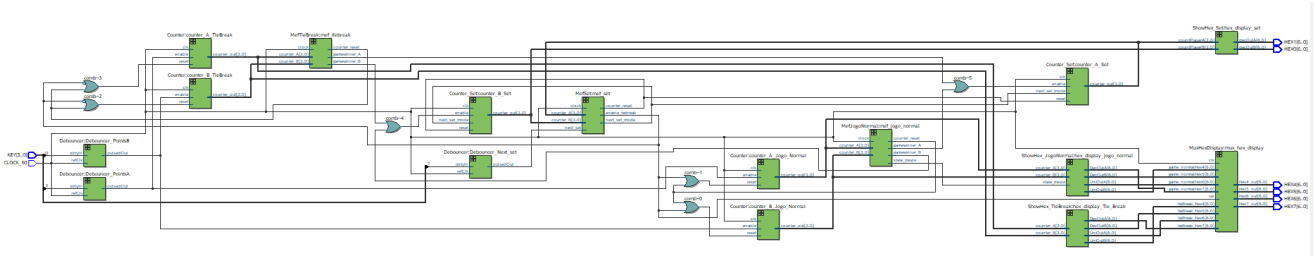

Figura 1

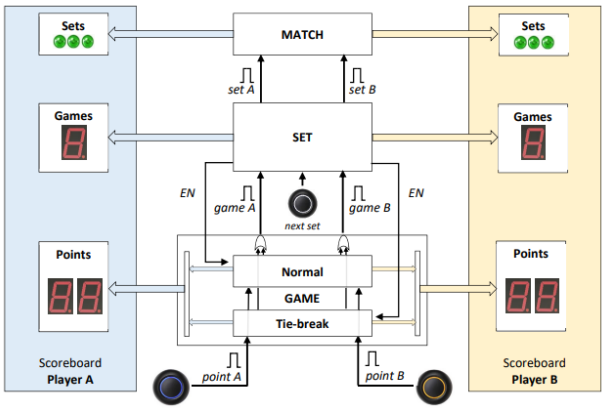

Figura 2

Como pode ser visto no diagrama da **Figura 1**, e na arquitetura da **Figura 2**, os ***inputs*** são dados pela ***KEY[3]***, ***KEY[1]*** e ***KEY[0]***, que se encontram ligados a blocos de ***debounce***, de forma a permitirem um ***input viável***, dentro de um intervalo aceitável. As saídas dos ***debouncers*** para a ***KEY[3]*** e ***KEY[0]*** por sua vez estão conectadas a 4 contadores (2 contadores para o modo de **Jogo Normal**, e 2 contadores para o modo de **Jogo TieBreak**, sendo que em cada modo de jogo, é atribuído 1 contador a cada jogador), para que possam ser contabilizadas as **pontuações** de cada jogador, ou no caso de ***TieBreak***, os **pontos** de cada jogador.

Neste diagrama estão ainda presentes **3 MEFS** (***MefTieBreak***/***MefSet***/***MefJogoNormal***) que irão receber como ***input*** os outputs dos contadores anteriormente mencionados, uma vez recebidos os ***inputs***, as **MEFS** ficam encarregues de determinar a situação de jogo de cada jogador.

Os blocos ***ShowHex_TieBreak***, ***ShowHex_JogoNormal*** e ***ShowHex_Set*** são usados para converter um **input binário**, vindo dos contadores de cada jogador, num **output em BCD**, para que mais tarde possam ser apresentados nos **displays de 7 segmentos** os valores das respetivas **pontuações**, **pontos** e **jogos**.

Ambos os ***outputs*** dos blocos ***ShowHex_TieBreak*** e ***ShowHex_JogoNormal*** encontram-se ligados a um *multiplexer* representado pelo bloco ***MuxHexDisplay***, cujo **SELECT** (**sel**) encontra-se ligado ao ***output enable_tiebreak*** do bloco ***MefSet***, o mesmo ***output*** tem o propósito de servir como **comutador** entre o modo de **Jogo Normal** ou modo de **Jogo TieBreak**, para que seja exibida a informação certa nos **displays de 7 segmentos**.

Manual do Utilizador

Ao ligar o Marcador de Ténis, os **displays de 7 segmentos** deverão apresentar o valor inicial **'0'**.

Para que comece a usufruir da máquina e a registar os pontos dos jogadores, o utilizador deverá pressionar a ***KEY[3]*** para incrementar a pontuação do jogador A ou ***KEY[0]*** para incrementar a pontuação do jogador B.

A pontuação decorre no formato **0/15/30/40** representada nos displays ***HEX[7..4]*** (***HEX[7]*** e ***HEX[6]*** para o jogador A, ***HEX[5]*** e ***HEX[4]*** para o jogador B) onde, o primeiro jogador a atingir a pontuação **40** com uma diferença de **2 pontos do adversário** ganha o jogo, vendo assim a sua pontuação de jogos incrementada através dos *displays* ***HEX[1]*** (jogos ganhos do jogador A) e ***HEX[0]*** (jogos ganhos do jogador B).

Em caso de empate numa pontuação igual a **40**, irá ser mostrada a letra **'d'** nos displays **HEX[6]** e **HEX[4]** que simbolizará o empate (**deuce**). Para sair do empate, os jogadores precisam de ter uma vantagem de 2 pontos em relação ao adversário sendo que, o jogador que registar o primeiro ponto irá receber a mensagem **"ad"** (**advantage**) no seu *display*, representando que se encontra em vantagem por 1 ponto em relação ao seu adversário, sendo apenas necessário mais 1 ponto consecutivo para ganhar o jogo em que se encontra.

Caso o jogador que se encontre com a mensagem **"ad"** falhe em conseguir o ponto consecutivo, irá voltar para o empate, sendo assim demonstrada a letra **'d'** nos displays de ambos os jogadores.

O primeiro jogador a chegar aos 6 jogos ganhos sem que o adversário tenha atingido 5 jogos, ou chegar aos 7 jogos ganhos sem que o seu oponente atinga os 6 jogos, irá automaticamente ser vencedor do *set*.

Caso ambos os jogadores empatem os jogos a 6, irão entrar assim em **TieBreak** que apenas termina quando um jogador (vencedor do jogo) atinge um número de pontos igual ou superior a 7 com diferença igual ou superior a 2 para o seu adversário.

No final de cada *set*, o utilizador irá ter de pressionar a ***KEY[1]*** para dar como encerrado o *set* em que se encontra, dando assim início ao próximo *set*.

Implementação

A implementação da estrutura usada foi feita de forma hierárquica, tendo sido iniciada com a construção do bloco **Jogo Normal**, seguida da construção do bloco do modo de jogo **Tie Break**, e sendo finalizada com a construção do bloco **Set,** sendo efetuada a validação de cada um destes blocos por testes na placa e simulação.

- *Bloco Jogo Normal*

  A máquina de estados do bloco em questão adquiriu a representação gráfica apresentada na **Figura 3**.
  Esta **MEF** possui **5 estados** (***S0***, ***S1***, ***S2_A***, ***S2_B***, ***S3***), que podem ser legendados da seguinte forma:
  - **S0** : ***estado inicial***, usado para detetar uma vitória ou empate;
  - **S1** : **estado de deuce**, usado para detetar uma vantagem para um dos jogadores;
  - **S2_A** : ***estado de vantagem do Jogador A***, usado para detetar se o mesmo marca um ponto sequente, e caso não o faça, a máquina volta para o estado S1;
  - **S2_B** : ***estado de vantagem do Jogador B***, usado para detetar se o mesmo marca um ponto sequente, e caso não o faça, a máquina volta para o estado S1;
  - **S3** : ***estado da vitória***, usado para definir o vencedor do jogo, e para reiniciar o contador e a máquina.

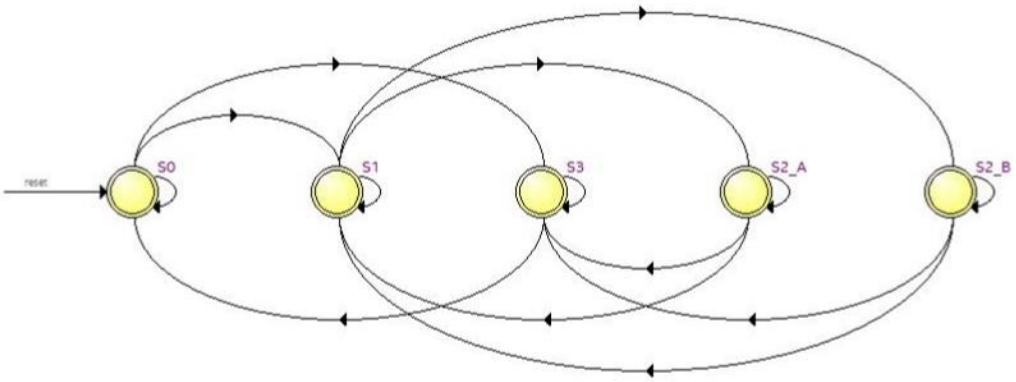

Figura 3

- *Bloco Tie Break*

  Esta **MEF** possui **2 estados** (***S0***, ***S1***), que podem ser legendados da seguinte forma:
  - **S0** : ***estado inicial***, usado para detetar a vitória de um dos jogadores;
  - **S1** : ***estado da vitória***, usado para definir o vencedor do jogo e reiniciar o contador.

- *Bloco Set*

  A máquina de estados do bloco em questão adquiriu a representação gráfica apresentada na **Figura 4**.
  Esta **MEF** possui **3 estados** (***S0***, ***S1***, ***S2***), que podem ser legendados da seguinte forma:
  - **S0** : ***estado inicial***, usado para detetar a vitória do set, ou um empate em jogos, o que levará a um jogo de Tie Break;
  - **S1** : ***estado de tie break***, usado para detetar a vitória de um dos jogadores neste modo de jogo;
  - **S2** : ***estado da vitória***, usado para definir o vencedor do set, e caso o utilizador assim escolha, avançar para o próximo set.

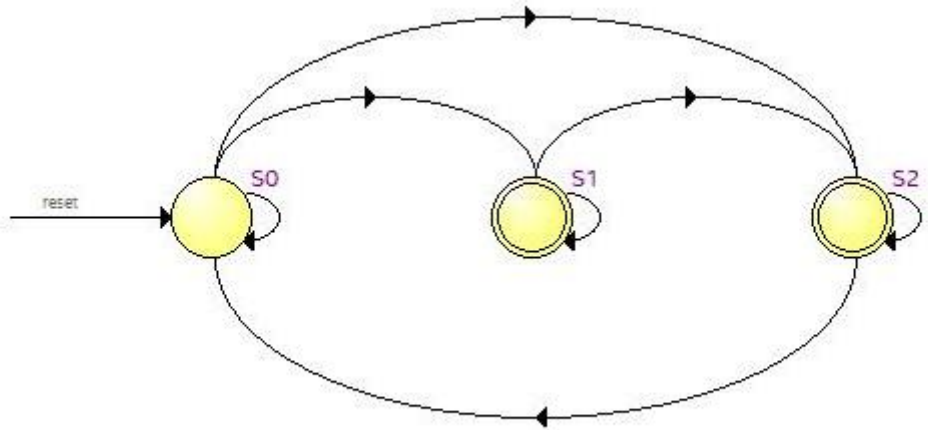

Figura 4

Com isto, a implementação ficou concluída com a adição do bloco **Set** ao sistema, sendo de seguida criados sinais adequados para interagir com os diferentes blocos, e de forma a facilitar a interação com as outras máquinas de estados. Foi ainda criado um bloco de **Mux**, utilizado com o intuito de selecionar o tipo de pontos a serem apresentados ao utilizador, ou seja, caso o bloco **Set** detete um ***Tie Break***, a **Mux** irá alterar o seu output para que os pontos de ***Tie Break*** sejam exibidos a ambos os jogadores.

Conclusão

Em suma, o trabalho realizado acabou por não corresponder na totalidade aos objetivos definidos, tendo ocorrido alguns erros no bloco **Set**, o que levou á ausência do bloco **Match**, bloco este que teria o intuito de assinalar os sets que já teriam ocorrido. O **sistema de gestão de serviço** e o **sistema de deteção de break points** ficaram condicionados às mesmas circunstâncias que levaram à ausência do bloco Match.

No entanto, o sistema encontra-se operacional para registar a pontuação, os pontos e os jogos ganhos, sendo capaz de reconhecer quando um *set* é ganho, parando assim a contabilização dos jogos da partida, e recorrendo ao *input* do utilizador para avançar para o próximo *set*.

Ambos os elementos do projeto realizaram papéis importantes, sendo capazes de trabalhar em conjunto para resolver vários obstáculos apresentados ao longo da realização do projeto.

Assim, apesar de o projeto não ter sido realizado na sua totalidade, e tendo em conta que a implementação da estrutura foi feita de forma faseada, tendo sido realizada uma parte significativa do projeto, o grupo autoavalia-se de forma positiva, atribuindo uma nota de **11**.

Percentagem de trabalho realizada por cada elemento:

- Tomás Sousa Fonseca | Mec. 107245 : **50%**
- Tiago Sousa Fonseca | Mec. 107266 : **50%**

Informação Relevante

Durante a realização do projeto, e predominantemente na implementação do bloco Set, ao testar o projeto na FPGA do macro-grupo, a mesma apresentava valores não compreendidos pelo grupo. No entanto, ao testar o mesmo projeto noutra FPGA, sem qualquer tipo de alteração ao código, esta apresentava valores esperados e compreendidos pelo grupo, valores que concordavam com o bom funcionamento de todo o projeto.

Assim, assume-se que por motivos desconhecidos ao grupo, a FPGA possuída pelo macro-grupo não estaria a funcionar de forma correta, sendo que para FPGAS iguais, programadas de forma igual, com o mesmo código e estrutura, foram obtidos valores totalmente diferentes, sendo que na FPGA que não pertencia ao macro-grupo, os valores eram os esperados.

Ao dialogar com os restantes colegas com projetos diferentes, os mesmos relataram o mesmo problema, não tendo qualquer tipo de conhecimento quanto á origem do mesmo.

Com isto, o nosso projeto viu-se atrasado em vários dias, pois tínhamos em mente que seria um problema de código ou estrutura, tendo ambos os elementos sido induzidos em erro, o que nos levou a várias tentativas de implementação diferentes, mas sem sucesso. Este erro é esperado acontecer na apresentação do projeto (caso não seja apresentado nas placas em que funcionou), sendo que caso apareça, não teremos qualquer tipo de capacidade para justificar o comportamento da FPGA, visto que ao rever o código, não são encontrados erros.