Projeto 2- PSI (Documento 2 / 2021)

1. Instruções Básicas

Repetimos aqui a série de tarefas previstas no Projeto-2, já apresentada no Doc_Semana_1.

Até a finalização do projeto pelo(a) aluno(a), cada uma das tarefas do seguinte conjunto deverá estar devidamente concluída (as partes em cinza já foram cumpridas, de acordo com o calendário):

Parte A: Analisar o projeto VHDL do **Base_Circuit_Extended**, fornecido pronto, e testá-lo. Identificar nele os blocos fornecidos em aulas anteriores (poderão conter pequenos ajustes) e relembrar as suas funcionalidades: counter (aula 3); num_gen (aula 4); alu (aula 5); fs_main e fsm_init (aula 6); rand_num (aula 7); mem e reg_bank (aula 8); base_control (aulas 9 e 10).

Parte B: Analisar o projeto VHDL do bloco **FSM_Guru**, componente do **Base_Circuit_Extended**. Verificar particularmente a sua interação com o **Base Datapath** e o circuito de memória **mem**.

Projeto do bloco **Disciple Circuit** que passará pelas seguintes etapas:

Etapa 1: Estudar a especificação dada para o **Disciple Circuit** (interfaces e funcionalidade. Modelar os blocos de controle (a FSM do discípulo) e de datapath para atender a especificação.

Parte C: Etapa 2: Baseando-se no **Base Circuit**, desenvolver o código VHDL da FSM do **Disciple Circuit**, seguindo-se o modelo e formulação de fsm vistos na aula 6.

Etapa 3: Baseando-se no **Base Circuit**, desenvolver o código VHDL do datapath do **Disciple Circuit**.

Etapa 4: Integração dos blocos acima no circuito topo **Disciple Circuit**. Testar em testbench.

Parte D: Integração dos blocos **Disciple Circuit** e **Base_Circuit_Extended** no ciruito top **Wisdom Ciruit**. Verificação da correção com o testbench dado.

Haverá orientações para cada etapa nas aulas 11 a 14.

2. Parte B (Etapa 1) - Especificação do Disciple Circuit

O objetivo principal desta aula da Semana 2 é iniciar o projeto do **Disciple Circuit**. Serão dadas as suas especificações detalhadas, que devem ser seguidas à risca, para que o circuito possa ser integrado corretamente com **Base Circuit-Extended** dentro do **Wisdom Circuit**.

2.1. Descrição da Funcionalidade do Disciple Circuit

O **Disciple Circuit** controla os passos do discípulo no jogo **Wisdom** de forma quase autônoma, isto é, de forma independente de outros blocos do sistema. Apenas alguns sinais de habilitação e de condição serão necessários para o controle de seus passos.

O discípulo percorre uma parte do tabuleiro (posições (x.y)), como é o caso de 8x8 da Figura 1. Entretanto, na codificação VHDL, os alunos poderão optar por usar um parâmetro configurável "WIDTH" para indicar o tamanho do tabuleiro. Estas posições correspondem aos endereços da memória onde será armazenado o código representativo do **discípulo**.

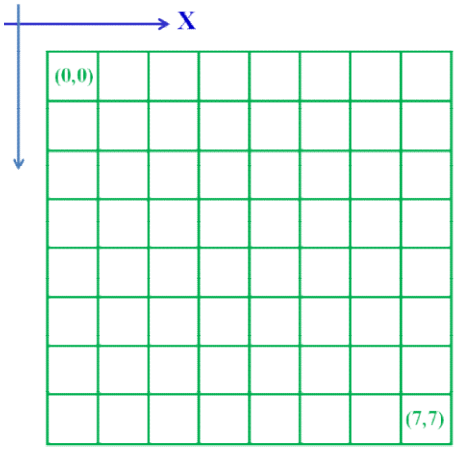

Figura 1. O tabuleiro (memória)

Tão logo o controlador da **FSM Main** (das aulas passadas) ative um sinal *start_step* (é o mesmo que habilita a caminhada do **guru**), indicando que o **discípulo** deve começar com os seus passos, um endereço aleatório deverá ser gerado dentre as posições da Figura 2. Este será o ponto inicial da caminhada do discípulo. É tarefa do circuito enviar à memória o endereço escolhido (6 bits), e o dado a ser escrito nesta posição da memória, no caso, "**0000 0100**" (código **DISCIPLE**).

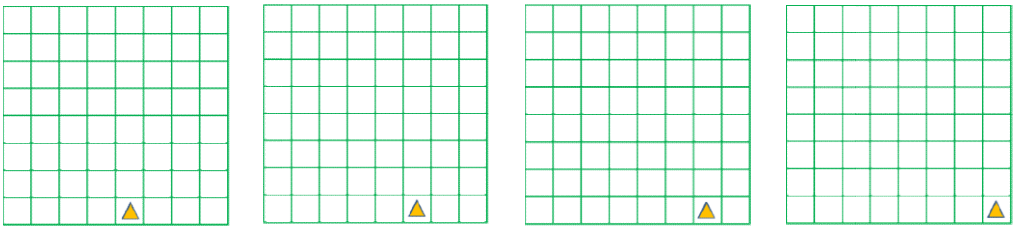

Figura 2. Posição inicial do discípulo

A cada ativação do sinal de contagem *cnt_disc_rdy*, gerado no módulo **step counter** (seguindo a velocidade definida pelo **jogador**), o **discípulo** dará um passo na direção vertical do tabuleiro, para cima. No exemplo da Figura 3(a), o discípulo inicia a sua caminhada na posição (7,6). Durante a caminhada, deve ocorrer o seguinte:

1) a cada passo, a máquina do discípulo deverá providenciar a escrita na memória do código do discípulo, "**0000 0100**" (código **DISCIPLE**) na nova posição, como é o caso da célula (6,6) da Figura 3(b), assim como remover o dado da casa anterior ((7,6) de onde está saindo), escrevendo o valor "**0000 0001**" (código **BLANK** do vazio).

2) uma exceção ocorre quando a máquina reconhece que o discípulo ultrapassou a borda e deve, portanto, apenas escrever o valor "**0000 0001**" na casa da qual está saindo, a célula (0,6) da Figura 3(c). Adicionalmente, deve-se indicar a finalização da caminhada enviando um sinal ao **Base Circuit-Extendend** através da ativação do sinal *end_of_disciple*.

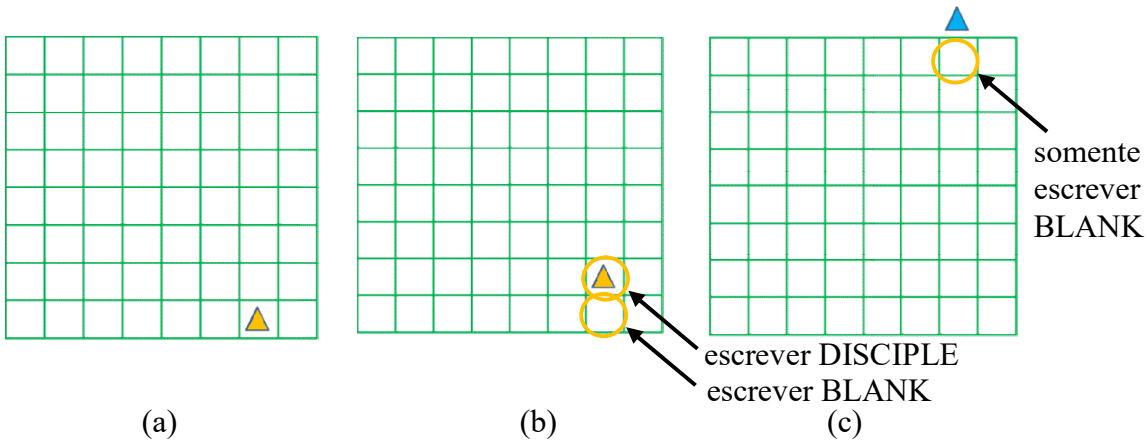

Figura 3. Posição inicial do **discípulo**

3) A caminhada do **discípulo** ocorre de maneira independente de outros circuitos com duas condições de exceção a serem reconhecidas:

4) A primeira exceção ocorre quando o **discípulo** encontra o **guru** (sempre na linha 0); tal situação é alertada pela ativação do sinal externo *duo_formed*. Neste caso, o **discípulo** deve proceder como no caso 1) porém, escrevendo "**0000 1000**" (código **DUO** do duo) ao invés do código do DISCIPLE; é o caso da Figura 4 com a célula (0,6). Deve-se também, como no caso 1) apagar os dados da casa anterior (1,6), de onde o **discípulo** está saindo, escrevendo o valor "**0000 0001**" (código **BLANK** do vazio).

5) A segunda exceção ocorre quando o **discípulo** termina a sua caminhada, mas o **guru** está entrando na sua posição anterior, como indicado na Figura 5. O **discípulo** não deve escrever BLANK nesta posição anterior, diferente do estabelecido no caso 2), para não ter perigo de apagar o código GURU que será escrito pela **FSM guru**. Esta situação é detectada pela ativação do sinal externo de entrada *guru_right_behind*. Neste caso, o **discípulo** deve simplesmente terminar a caminhada e não escrever nada- e enviar a ativação do sinal *end_of_disciple*.

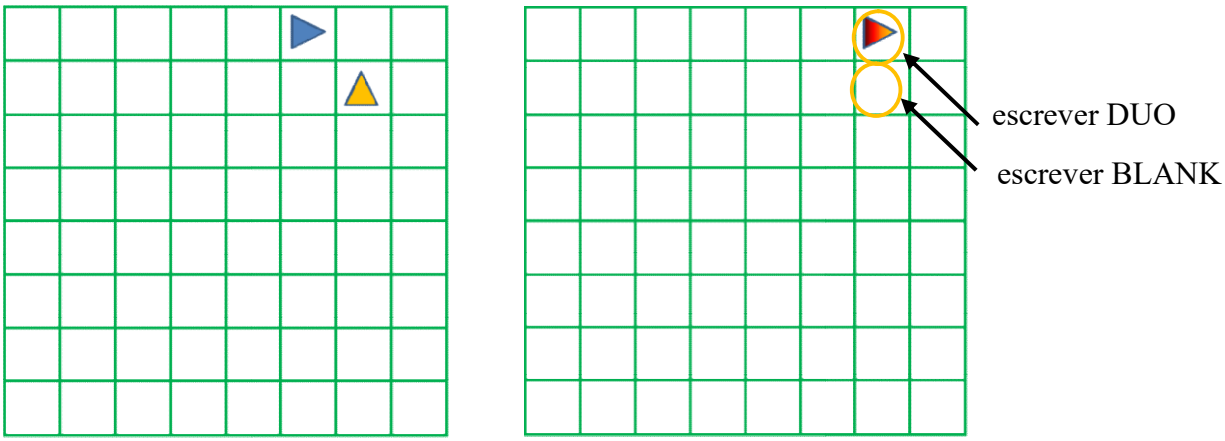

Figura 4. À esquerda, o **discípulo** (amarelo) e o **guru** (azul escuro) fazem o seu próximo passo; à direita, o **duo** (vermelho) é formado e BLANK escrito na célula anterior

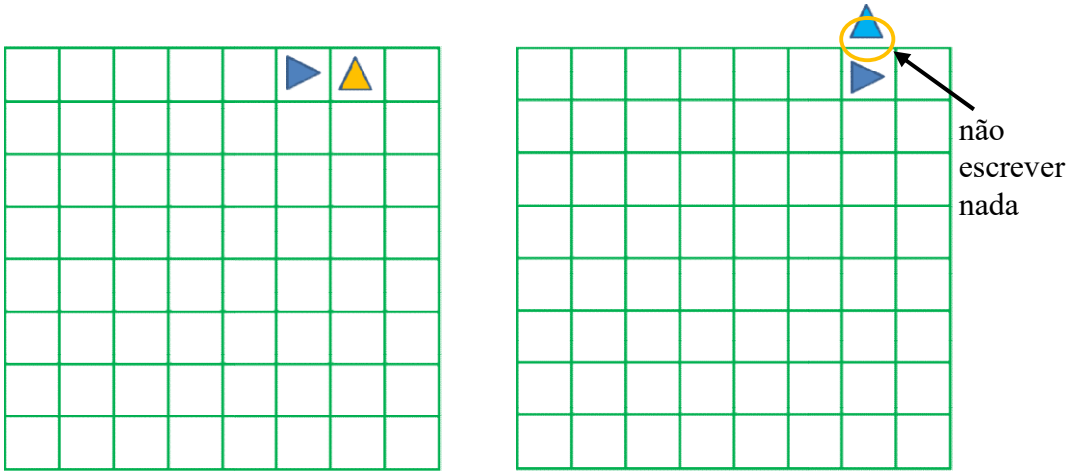

Figura 5. À esquerda, o **discípulo** (amarelo) e o **guru** (azul escuro) fazem o seu próximo passo; à direita, o **discípulo** (azul claro) ao ultrapassar a borda e o **guru** ocupando a sua célula

Ao atingir a borda, o discípulo deve aguardar novamente a sinalização de *start_step* para começar uma nova iteração, que recomeça na condição indicada na Figura 2.

2.2. Descrição das Interfaces do Disciple Circuit

A Figura 6 apresenta o Disciple Circuit com todos os seus portos e sinais previstos, com as direções indicadas. A seguir descrevemos a funcionalidade e cada sinal (Observação: no código VHDL, todos os sinais deverão ser *sdt_logic* ou *std_logic_vector*):

- **clock**: sinal de relógio do sistema
- **reset:** sinal de inicialização do jogo (não para início da iteração/rodada). Em reset='1´, a máquina do discípulo deve ir ao estado inicial.

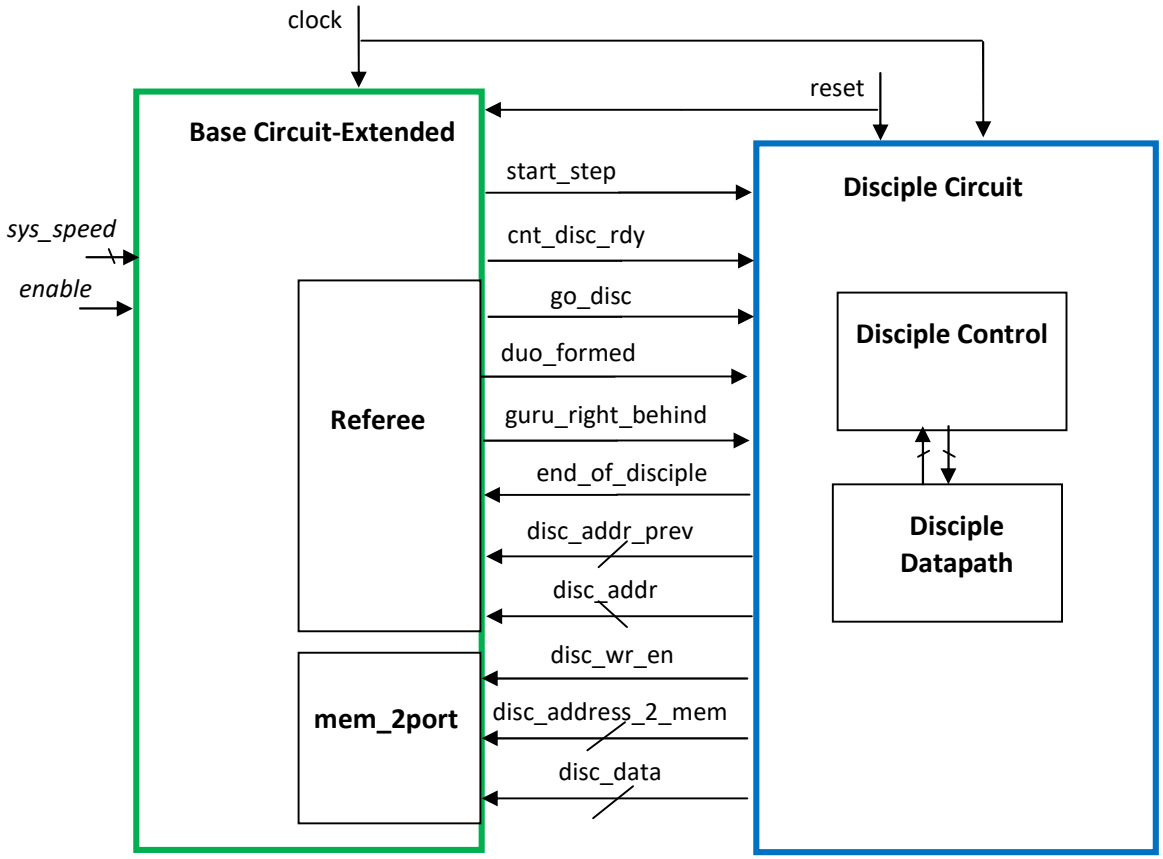

Figura 6. Interfaces do **Disciple Circuit**

- **start_step:** sinal oriundo da **FSM Main** indicando que o discípulo (assim como o guru) deve começar a sua caminhada. É um pulso *active-high* de 1 ciclo de relógio
- **cnt_disc_rdy:** sinal oriundo do **step counter** indicando os disparos para o próximo passo. O esquema de pulsos é apresentado na Figura 7. A distância entre os pulsos indica a velocidade dos passos do **discípulo**, portanto a máquina do **discípulo** não precisa lidar com isto. O **Disciple Circuit** aguarda estes pulsos como uma sequência assíncrona e, uma vez recebido o primeiro pulso, assume-se que, pelo menos, 7 outros pulsos ocorrerão até que o **discípulo** ultrapasse a borda superior.

Figura 7. Pulsos *cnt_disc_rdy*

- **go_disc:** sinal oriundo do **referee**- a cada passo do **discípulo**, antes da gravação na memória, a máquina deve aguardar por este sinal de liberação para prosseguir com a tarefa. Trata-se de um pulso de 1 ciclo de relógio.

Enquanto tal pulso não for verificado no porto correspondente, a máquina de estadoso **Disciple Circuit** deve aguardar e nada realizar.

- **duo_formed:** sinal oriundo do **referee**- este sinal terá valor '1' quando é detectada a condição de que tanto o **guru** e o **discípulo** vão ocupar a mesma célula do tabuleiro (para formar o **duo**). Neste caso, a escrita deve ser do código DUO. Ver figura 4.
- **guru_right_behind:** sinal oriundo do **referee**- este sinal terá valor '1' quando se detecta que o **guru** está entrando na célula da qual o **discípulo** está saindo. É o alerta de que não se deve escrever nada. Ver figura 5.
- **end_of_disciple:** sinal de saída, indicando ao **referee** que o **discípulo** ultrapassou a borda superior. Pulso de, no mínimo, 1 ciclo de relógio.
- **disc_address:** o **Disciple Circuit** deve disponibilizar para o **referee** o endereço (6 bits) da nova posição da memória para onde o **discípulo** está indo. Este endereço deve ser passado tão logo seja registrado o valor obtido com a ULA do datapath do **Disciple Circuit**.
- **disc_address_prev:** o **Disciple Circuit** deve disponibilizar para o **referee** o endereço (6 bits) da posição atual (antiga) da memória de onde o **discípulo** está saindo, escrevendo o valor adequado. Este endereço deve ser passado tão logo seja registrado o valor obtido com a ULA do datapath do **Disciple Circuit**.
- **disc_wr_en:** Sinal para o porto a de **mem_2port**, de habilitação de escrita dos códigos DISCIPLE ou BLANK ou DUO (de acordo com as condições vistas na Seção 2.1).
- **disc_address_2_mem**: Endereço para o porto a de **mem_2port**, com o endereço (6 bits), ou do *disc_address* ou o *disc_address_prev* , dependendo do código a ser escrito (de acordo com as condições vistas na Seção 2.1). Sincronizado com o sinal *disc_wr_en*.
- **disc_data:** Dado para o porto a de **mem_2port**, com dado (8 bits) correspondente a um dos 3 códigos: DISCIPLE, BLANK ou DUO. (de acordo com as condições vistas na Seção 2.1). Sincronizado com o sinal *disc_wr_en*.