PCS3115 - Sistemas Digitais I
Projeto 1 - Jogo da Velha Eletrônico

Glauber De Bona

Deadline: 24/05/2023

O objetivo deste trabalho é exercitar a descrição de circuitos combinatórios em VHDL, bem como o uso de componentes, com a modelagem estrutural. Como consequência, este trabalho oferece um primeiro contato com a linguagem VHDL, a preparação de testbenchs e simulação.

Introdução

O Jogo da Velha é um clássico mundial de regras simples, onde dois jogadores alternam jogadas num tabuleiro 3 por 3, vencendo que fizer a primeira linha.

Se você não conhece o jogo, veja por exemplo: `https://pt.wikipedia.org/wiki/Jogo_da_velha`

Você trabalha em uma empresa de brinquedos infantis que quer desenvolver um jogo da velha eletrônico. A ideia é usar um circuito combinatório que receba em sua entrada o estado atual do jogo e determine na saída se há algum ganhador, além de oferecer dicas de jogadas para ganhar uma partida.

As 9 casas do tabuleiro serão nomeadas conforme o esquema abaixo:

a3	b3	c3
a2	b2	c2
a1	b1	c1

Sensores no tabuleiro detecta se há algo ('×' ou '○') em cada casa (o *estado* do tabuleiro), gerando 2 bits para cada uma das 9 casas: a casa pode estar vazia (00), pode conter um '×' (10) ou um '○' (01). O resultado de uma partida pode ser codificado usando também 2 bits: 10 para vitória do jogador '×'; 01 para a vitória do jogador '○'; 00 para empate ou jogo em andamento; e 11 para situações em que ambos os jogadores já fizeram uma linha (o que não deveria acontecer de acordo com as regras).

Entrada 11 para uma posição não deve ocorrer (don't care).

Dicas para os jogadores serão dadas através de leds nas casas. Quando um jogador pedir um dica, e houver jogadas que vençam a partida, leds indicarão quais são essas casas. O jogador poderá apertar um botão para pedir dica, e uma chave determina qual é o jogador da vez ('×' ou '○').

Sua missão é projetar este sistema digital!

Atividades

T1A1 (2 pontos) Implemente um componente combinatório em VHDL que determine o vencedor de uma partida de Jogo da Velha, obedecendo a entidade abaixo:

Trabalho 1, Atividade 1

```
entity JogoDaVelha is
port(
  a1, a2, a3: in  bit_vector(1 downto 0);
  b1, b2, b3: in  bit_vector(1 downto 0);
  c1, c2, c3: in  bit_vector(1 downto 0);
  z:          out bit_vector(1 downto 0)
);
end JogoDaVelha;
```

As entradas representam as casas do tabuleiro, e a saída `z` (2 bits), codifica o resultado da disputa, conforme explicado acima.

Embora as regras proibam, seu circuito deve apontar normalmente o resultado da partida em situações em que um jogador jogou mais de uma vez seguida; por exemplo, se há 9 (nove) '×' no tabuleiro, a saída deve ser 10.

T1A2 (5 pontos) Para gerar as dicas, implemente um componente combinatório em VHDL que receba o estado do tabuleiro, além de qual o jogador tem a vez, e, quando habilitado, indique quais casas correspondem a uma jogada vencedora:

Trabalho 1, Atividade 2

```
entity Ajuda is
port(
  dica, jogador: in bit;
  a1, a2, a3: in  bit_vector(1 downto 0);
  b1, b2, b3: in  bit_vector(1 downto 0);
  c1, c2, c3: in  bit_vector(1 downto 0);
  La1, La2, La3: out  bit;
  Lb1, Lb2, Lb3: out  bit;
  Lc1, Lc2, Lc3: out  bit
);
end Ajuda;
```

As entradas `a1` a `c3` representam o estado do tabuleiro, a entrada `dica` (ativa em ALTO) diz quando o jogador está pedindo uma dica, e a entrada `jogador` indica de quem é a vez (1 para '×', 0 para '○'). As saídas `La1` a `Lc3` controlam os leds nas casas do tabuleiro correspondes (1 para aceso, zero para agado). Quando não é solicitada dica, todos os leds devem permanecer apagados. Quando um jogador pede uma dica, e o jogo ainda está em adamento (sem vencedor), o led de uma casa deve acender se uma jogada ali vence a partida (faz a primeira linha). Para isso, o sistema deve levar em conta de quem é a vez, o que será informado pela entrada `jogador`.

Se já há alguma linha feita, o pedido de dica deve ser ignorado.

Por exemplo, considere a situação abaixo:

○	b3	○
×	○	c2
×	×	c1

Quando uma dica é pedida no estado acima, se `jogador=1`, então apenas o led em c1 deve acender; se `jogador=0`, então apenas os leds em b3 e c1 devem acender.

DICA: Use o componente da atividade anterior para detectar quando uma jogada é vencendora. Instanciando 9 componentes (1 para cada casa), você pode inserir uma jogada por casa (ou seja, por componente) para ver se ela leva à vitória. Por exemplo, se um dica é pedida, para um jogo em andamento, na vez de '○', o led de a1 deve acender se a casa a1 estiver vazia (a1=00) e um jogada lá (a1=01) leve à vitória.

Trabalho 1, Atividade 3

T1A3 (3 pontos) Como os sinais que vêm dos sensores apresentam eventuais erros de no máximo 1 bit, decidiu-se usar o Código de Hamming (distância 3, paridade par) para corrigi-los. O código será aplicado separadamente aos sensores de '× e de '○', tendo então 9 bits de dados (1 para cada casa). Implemente um componente em VHDL que receba na entrada 9 bits de dados e os 4 bits de paridade, codificados com Hamming, e forneça na saída nos os 9 bits de dados corrigidos.

Você pode assumir que há no máximo um erro, e que não acontecerão situações que implicam mais erros, como por exemplos quando todos os 4 bits de paridade estão errados.

```
entity hamming is
port(
  entrada: in  bit_vector(13 downto 1);
  dados:   out bit_vector(8 downto 0)
);
end hamming;
```

Os 13 bits na entrada são indexados de 13 a 1, sendo que os bits de paridade estão nas posições 1, 2, 4 e 8. O 9 bits de dados corrigidos na saída estão indexados de 8 a 0, sendo o 0 (zero) o menos significativo, correspondendo a `entrada(3)`.

Instruções para Entrega

Você deve acessar o link específico para cada tarefa (T1A1, T1A2 e T1A3) dentro do tópico "Projetos" no e-Disciplinas, já logado com seu usuário e senha, que levará à página apropriada do juiz eletrônico. Em cada atividade, você pode enviar apenas um único arquivo codificado em UTF-8. O nome do arquivo não importa, mas sim a descrição VHDL que está dentro. As entidades nas suas soluções devem ser idênticas àquelas neste enunciado ou o juiz não irá processar seu arquivo.

Quando acessar o link no e-Disciplinas, o navegador abrirá uma janela para envio do arquivo. Selecione-o e envie para o juiz. Jamais recarregue a página de submissão pois seu navegador pode enviar o arquivo novamente, o que vai ser considerado pelo juiz como um novo envio e pode prejudicar sua nota final. Caso desista do envio,

simplesmente feche a janela. Depois do envio, a página carregará automaticamente o resultado do juiz, quando você poderá fechar a janela. A nota dada pelo juiz é somente para a submissão que acabou de fazer.

O prazo para a submissão das soluções no Juiz é 24 de maio de 2023, quarta-feira, às 23:59. O Juiz aceitará até 5 submissões para cada atividade deste projeto. Sua submissão será corrigida imediatamente e sua nota será apresentada. A maior nota dentre as submissões será considerada. Neste trabalho, os problemas valem no máximo 10 pontos no juiz, porém a nota final deste trabalho será calculada com as ponderações indicadas em cada atividade neste enunciado, totalizando 10 para o trabalho todo. Como boa prática de engenharia, faça seus *test-benches* e utilize o GHDL ou o EDA Playgorund para validar suas soluções antes de postá-las no juiz.

Atenção: Para as três atividades do projeto, está proibido o uso de `process` ou das bibliotecas `std_logic_1164` e `textio`, ou de qualquer biblioteca não padronizada. Se seu arquivo mencionar essas bibliotecas, mesmo em um comentário, sua submissão nem será avaliada pelo juiz e ficará com nota 0 (zero).