USP-EP-PCS Agosto de 2020

Testbenches - Exercícios

PCS3225 - Sistemas Digitais II Versão 1

1. A seguinte entitidade descreve em VHDL um decodificador 3 para 8:

```
entity decoder_3to8 is
  port (
    A : in bit_vector (2 downto 0); -- Decoder Input
    Y : out bit_vector (7 downto 0) -- Decoder Output
    );
end entity decoder_3to8;
```

   A descrição completa da entidade está no arquivo decoder_3to8.vhd e o testbench em decoder_3to8_tb.vhd. Execute a simulação e verifique se o resultado corresponde ao da Figura 1.

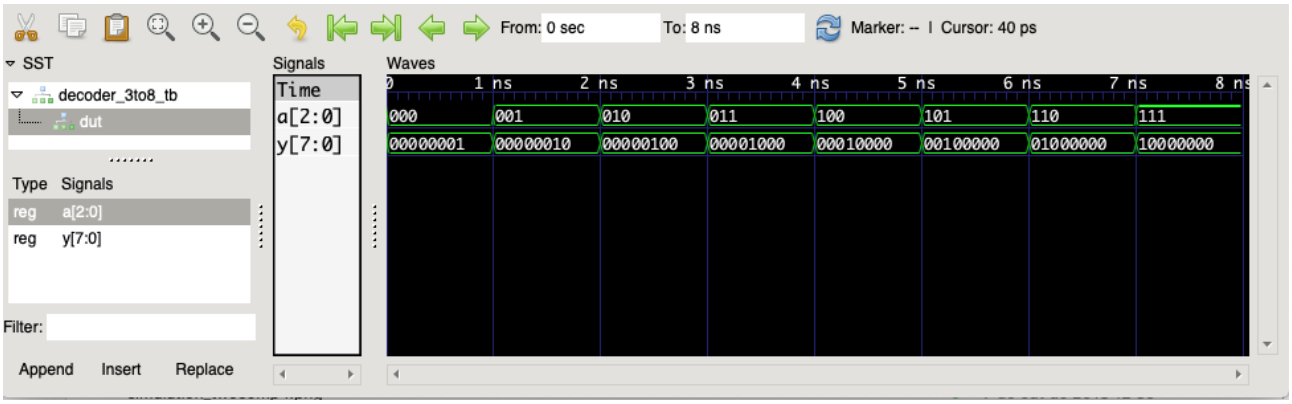

Figura 1: Simulação do decodificador 3 para 8

2. Projete e descreva em VHDL o testbench para o decodificador 2 para 4 com entrada ENABLE e saídas ativo baixo, cuja entidade é:

```
entity decoder_2to4en is
  port (
    EN : in bit; -- Enable
    A : in bit_vector (1 downto 0); -- Decoder Input
    Y_L : out bit_vector (3 downto 0) -- Decoder Output (active low)
    );
end entity decoder_2to4en;
```

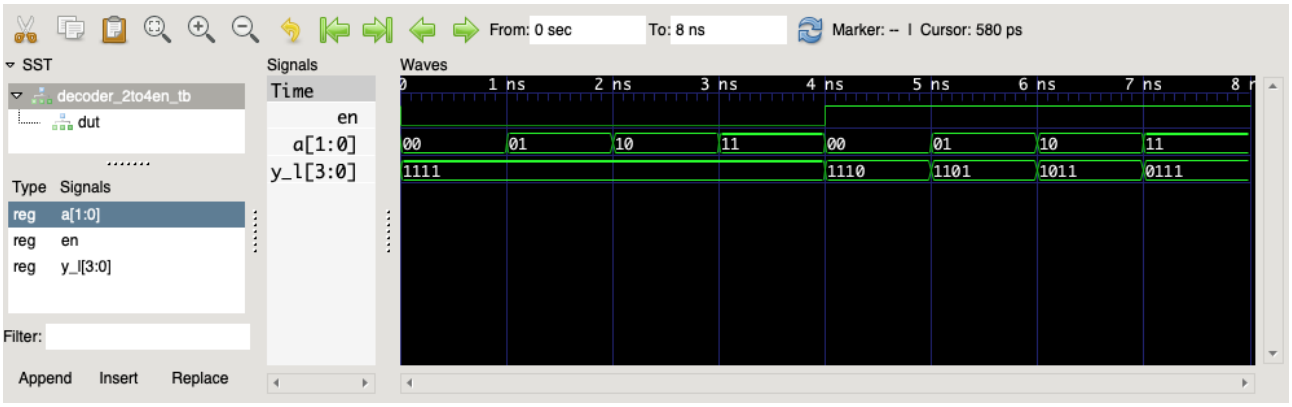

Figura 2: Simulação do decodificador 2 para 4

A descrição completa da entidade está no arquivo decoder_2to4en.vhd. Execute a simulação e verifique se o resultado corresponde ao da Figura 2.

3. Projete e descreva em VHDL um circuito que transforma código 1-out-4 para binário, ou seja, um codificador 4-para-2 com a seguinte tabela de transformação:

Entrada A	Saída Y
0001	00
0010	01
0100	10
1000	11

Utilize a seguinte entidade para descrever sua solução em VHDL:

```
entity encoder_4to2 is
  port (
    A : in bit_vector (3 downto 0); -- Encoder Input
    Y : out bit_vector (1 downto 0) -- Encoder Ouput
    );
end entity encoder_4to2;
```

A descrição completa da entidade está no arquivo encoder_4to2.vhd. Projete o testbench e execute a simulação e verifique se o resultado corresponde ao da Figura 3. Identifique se há erro no projeto do codificador. Se houver, faça a correção do circuito.

4. Projete um jogo de Batalha Naval, onde 2 jogadores usam um tabuleiro com 16 casas (Tabuleiro T com 4×4 posições). Um diagrama do sistema do jogo é apresentado na Figura 4. O Jogador 1 coloca submarinos no tabuleiro usando um teclado composto por 16 botões pressionáveis do tipo *toogle* (ao pressionar o botão ele muda de 0 para 1 e vice-versa). As posições com valor 1 indicam a presença de um submarino. O jogador 2, por sua vez, usa 2 teclados, L e C. O jogador 2 usa o teclado L para escolher a linha do tabuleiro e usa o teclado C para selecionar a coluna. Se o jogador 2 acertar um submarino, então o som "bum" será ouvido no alto-falante (saída S = 1), senão será

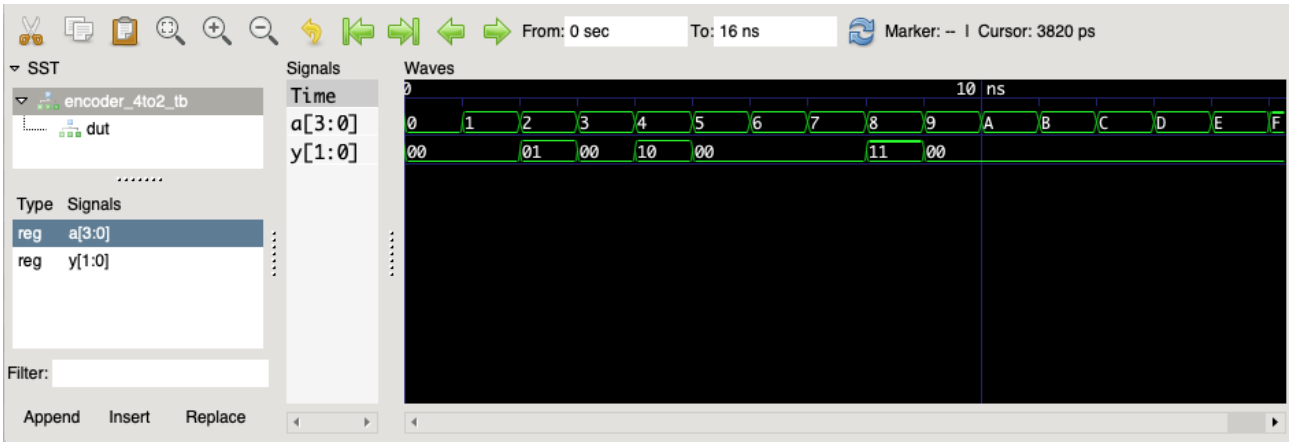

Figura 3: Simulação do codificador 4-para-2

emitido o ruído "chuá" (saída S = 0). Projete o circuito usando apenas multiplexadores, demultiplexadores, codificadores e decoficadores de 2 a 4 bits. Utilize a entidade a seguir para representar o jogo da Batalha Naval:

```
entity batalha_naval is
  port (
    T   : in bit_vector(15 downto 0);   -- Jogador 1
    C,L : in bit_vector(3 downto 0);    -- Jogador 2
    S   : out bit    -- Speaker "chuá" (erro) or "bum" (acerto)
    );
end entity batalha_naval;
```

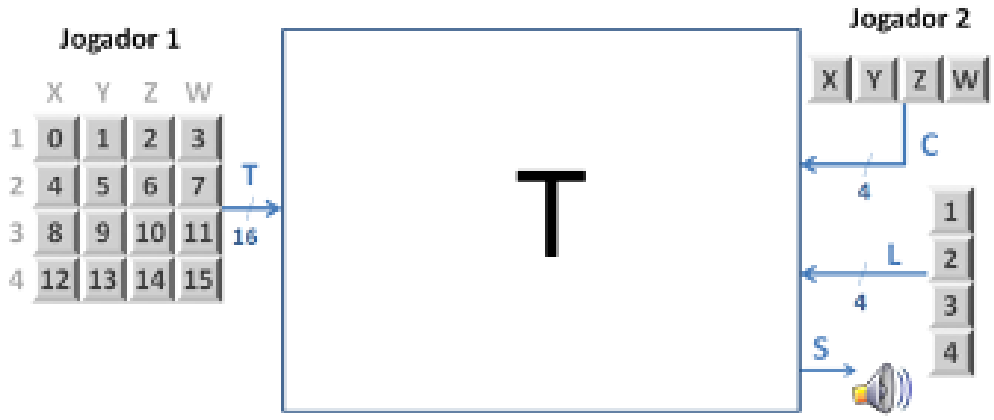

Figura 4: Diagrama do sistema da Batalha Naval

A descrição completa da entidade está no arquivo batalha_naval.vhd. Projete o testbench, execute a simulação e verifique se o resultado corresponde ao da Figura 5.

5. A entidade `twocomp4` recebe A (de 4 bits) e calcula o complemento de 2 de A. Projete a arquitetura de `twocomp4` e apresente o diagrama lógico do circuito. Projete e simule o

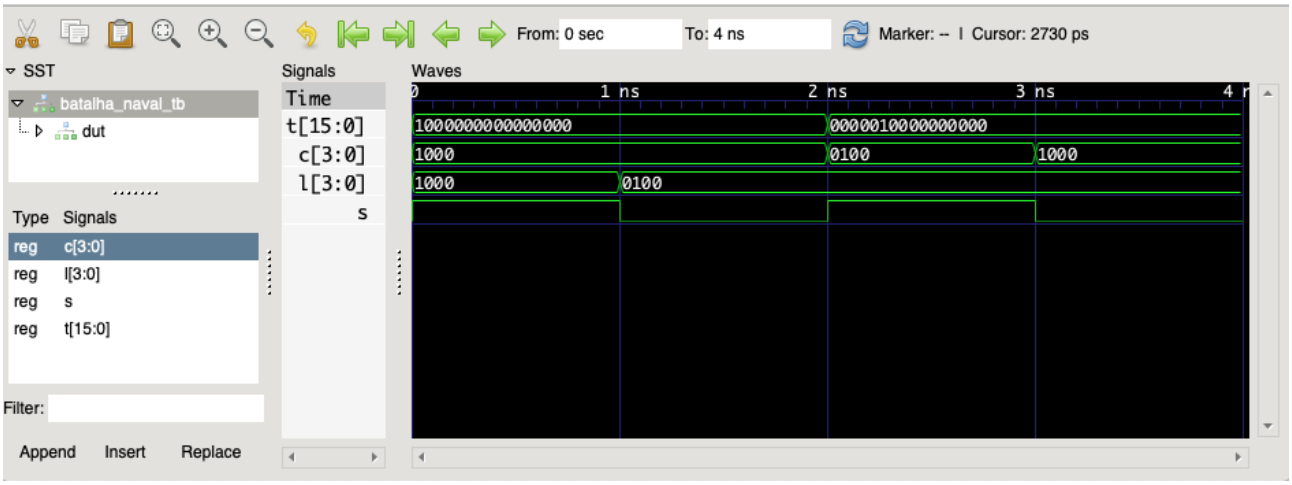

Figura 5: Simulação do circuito da Batalha Naval

circuito usando o testbench `twocomp4_tb.vhd`. Explique os resultados obtidos nos casos de teste apresentados na Figura 6.

```
entity twocomp4 is
  port (
    A : in bit_vector (3 downto 0);
    A2COMP : out bit_vector (3 downto 0)
    );
end entity twocomp4;
```

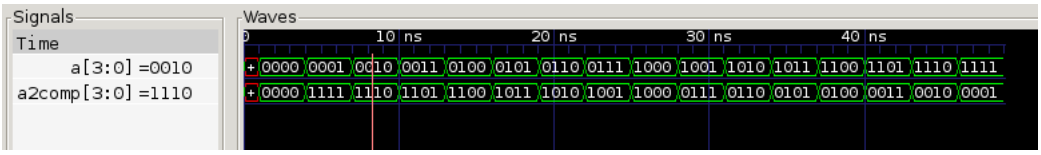

Figura 6: Casos de teste do circuito Complemento de 2 de 4 bits

.

6. Na Figura 7 podemos observar um cruzamento entre uma rua principal e uma rua secundária. O semáforo estará continuamente verde para a rua principal, exceto quando o sensor detectar a passagem de um ou mais carros na rua secundária. Neste momento, o controlador do semáforo iniciará a contagem de tempo de um TIMER. Ao final da contagem, o semáforo voltará a ficar verde para a rua principal. Um circuito de saída (SINAIS DOS SEMÁFOROS) produz os sinais de controle para as luzes dos semáforos das ruas principal e secundária. Assume-se que o tráfego de carros pela rua secundária é muito baixo. Podemos observar na Figura 8 um diagrama de blocos do sistema do controlador de semáforos.

   (a) Projete o controlador do semáforo, que tem as entradas CAR (que é ativo quando o sensor detectar passagem de carro na rua secundária) e TIMED (que é ativo quando o TIMER termina a contagem de tempo) e a saída MJGREEN (que é ativo quando

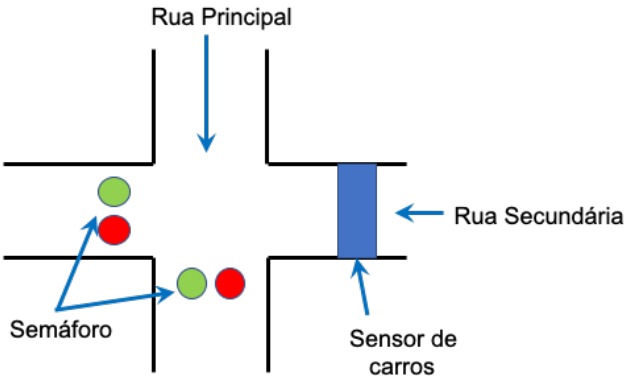

Figura 7: Cruzamento de rua principal com rua secundária

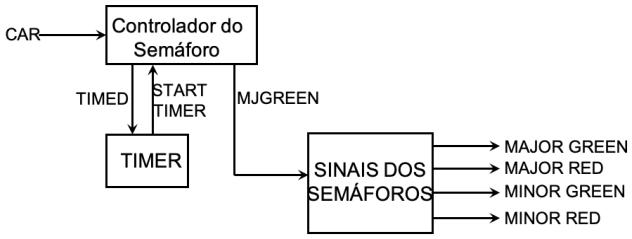

Figura 8: Sistema do controlador de semáforo

o semáforo é verde na rua principal). Construa a Máquina de Estado e projete o circuito sequencial correspondente, no modelo Moore e usando flip-flops do tipo D.

(b) Projete o TIMER, que tem a entrada START_TIMER (que é ativo para iniciar a contagem). O TIMER conta por 2 ciclos de relógio, quando então ativa o sinal de saída TIMED. Construa a Máquina de Estado e projete o circuito sequencial correspondente, no modelo Moore e usando flip-flops do tipo D.

(c) Projeto o circuito combinatório SINAIS DOS SEMÁFOROS, que deriva, a partir do sinal de entrada MJGREEN, os sinais de saida MAJORGREEN, MAJORRED, MINORGREN e MINORRED (MAJOR : principal, MINOR : secundário), que são ativos para acender os semáforos verde e vermelho da rua principal e verde e vermelho da rua secundária, respectivamente.

(d) Descreva o circuito completo, incluindo o controlador do semáforo, o TIMER e o SINAIS DOS SEMÁFOROS, em VHDL. Prepare um testbench e execute a simulação. Analise o resultado para verificar o correto funcionamento do circuito.

7. No mundo Glingon, as palavras da língua dos seus habitantes são formadas apenas por 3 letras, equivalentes às nossas letras K, Z e W. Sendo um brilhante aluno de Sistemas Digitais, você captou e descobriu que na transmissão digital em Glingon, as letras são codificadas em código de Huffman[1], sendo K = 0, Z = 10 e W = 11. A sua missão agora é projetar uma Máquina de Estado Finito, no modelo de Moore, para ouvir as comunicações dos habitantes de Glingon e aprender mais sobre a cultura deles.

[1]Código de Huffman foi mencionado apenas como conhecimento geral, não é necessário saber para resolver este problema. O aluno interessado poderá conhecer o Código de Huffman em `https://en.wikipedia.org/wiki/Huffman_coding`

(a) Projete a arquitetura da entidade glingon, onde SIN é a entrada serial da transmissão Glingon, SOUT é a saída com o símbolos decodificado e ACK é 1 quando um símbolo é reconhecido e 0 quando a máquina está em processo de reconhecimento. Assuma que SOUT = 00 quando a letra da transmissão é K, SOUT = 01 quando a letra é Z, SOUT = 10 quando a letra é W e SOUT = 11 quando não há caracter reconhecido. O sinal de entrada STX fica ativo quando o início de uma transmissão é detectada. Apresente o Diagrama de Transição de Estados da Máquina de Moore, a Tabela de Codificação de Estados, a Tabela de Transição de Estados, a Tabela de Saída, e as funções Próximo Estado e de Saída. Utilize flip-flops D. Quantos flip-flops serão necessários?

```
entity glingon is
  port (
    CLOCK,RESET : in bit;
    STX : in bit; -- Start of transmission
    SIN : in std_logic; -- Input symbol
    SOUT : out bit_vector (1 downto 0); -- Output symbol
    ACK : out bit -- Symbol acknowledge
    );
end entity glingon;
```

(b) Qual será a sequência de bits da saída SOUT quando SIN for "011101011001011"?

(c) Descreva o circuito projetado em VHDL, prepare um testbench e execute a simulação. Analise o resultado para verificar o correto funcionamento do circuito.