Laboratório Digital I - PCS3635

Planejamento da Experiência 6:

Projeto Base do Jogo do Desafio da Memória

Marco Aurélio C. O. Prado - NUSP 11257605
Victor Hoefling Padula - NUSP 10770051
Turma 04 - Bancada A1

São Paulo - SP
23/02/2021

1) Objetivo

Os objetivos da aula consistem em aprender sobre:

- Projeto de circuitos usando descrição estrutural VHDL;
- Interface de circuitos digitais com elementos externos de entrada de dados;
- Documentação de projetos (planejamento e relatório);
- Uso de sinais periódicos como clock;

2) Elaboração do Circuito “circuito_exp6.vhd”

2.1) Descrição do funcionamento do circuito

O circuito é composto de uma unidade de controle e um fluxo de dados com 2 contadores, 2 comparadores, 1 memória, 1 registrador, conversores para 7 segmentos e 1 detector de jogadas. O circuito basicamente segue uma lógica de ter um contador limitando a contagem de outro contador que incrementa os endereços da memória. Isso é feito usando um comparador que verifica se o endereço é menor que o limite. Quando o endereço é igual ao limite, ele incrementa 1 ao contador de limite e repete a lógica do circuito.

2.2) Descrição VHDL do circuito “circuito_exp6.vhd”

O código da unidade de controle segue a seguinte lógica:

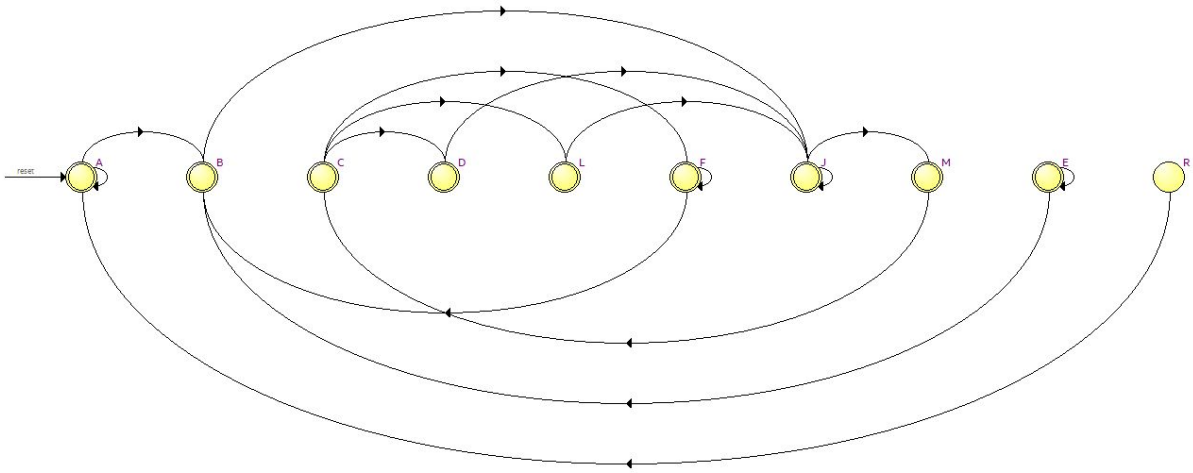
```
library ieee;
use ieee.std_logic_1164.all;

entity unidade_controle is
  port (
    clock:     in  std_logic;
    reset:     in  std_logic;
    iniciar:   in  std_logic;
    fimC:       in  std_logic;
    fimL:       in  std_logic;
    jogada:   in std_logic;
    enderecoIgualLimite: in std_logic;
    zera:      out std_logic;
    conta_end:     out std_logic;
    conta_lim: out std_logic;
    zera_lim: out std_logic;
    pronto:    out std_logic;
    db_estado: out std_logic_vector(3 downto 0);
```

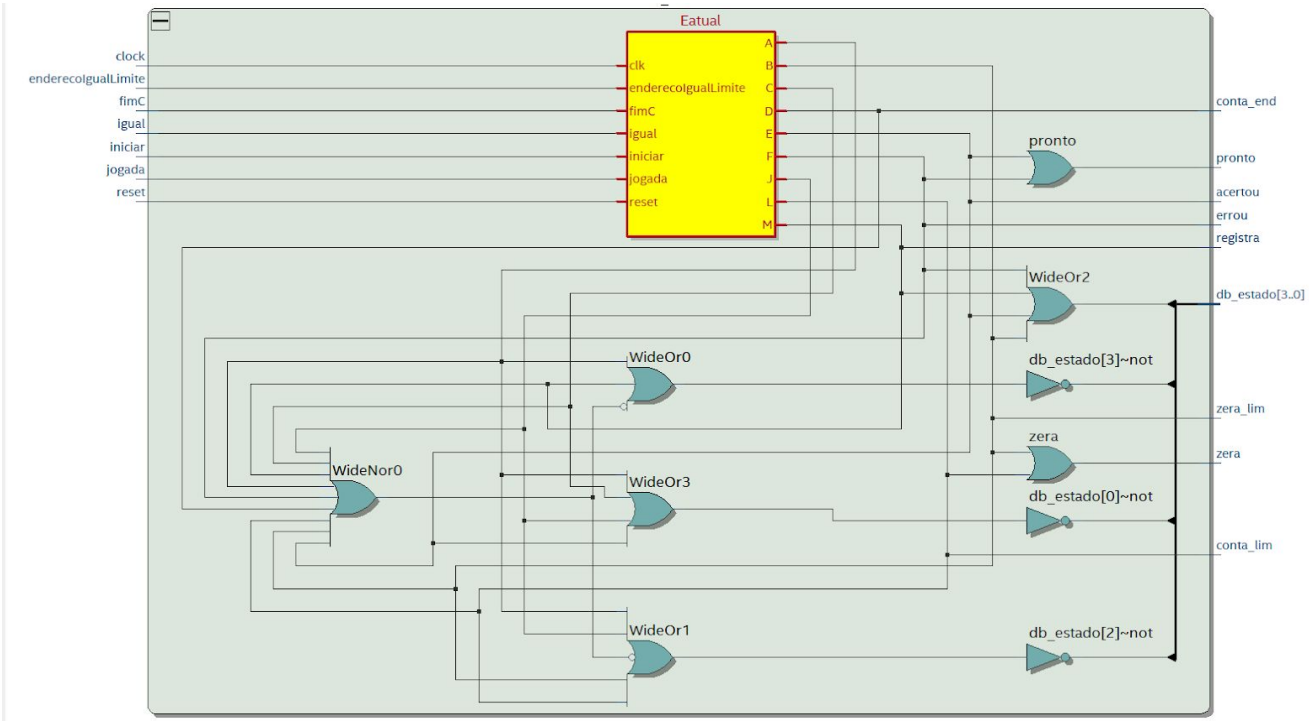
```
    acertou: out std_logic;
    errou: out std_logic;
    registra: out std_logic;
    igual: in std_logic
  );
end entity;

architecture fsm of unidade_controle is
  type t_estado is (A, B, C, D, E, F, M,J,L,R);
  signal Eatual, Eprox: t_estado;
begin
  -- memoria de estado
  process (clock,reset)
  begin
    if reset='1' then
      Eatual <= A;
    elsif clock'event and clock = '1' then
      Eatual <= Eprox;
    end if;
  end process;

  -- logica de proximo estado
  Eprox <=
      A when  Eatual=A and iniciar='0' else
      B when  Eatual=A and iniciar='1' else
      J when Eatual=B else
      J when Eatual=J and jogada='0' else
      M when Eatual=J and jogada='1' else
      C when Eatual=M else
      D when Eatual=C and enderecoIgualLimite='0' and igual='1' else
      F when  Eatual=C and igual='0' else
       L when Eatual=C and enderecoIgualLimite='1'  and igual ='1'
else
      J when  Eatual=D else
      J when Eatual=L else
      E when  Eatual=C and FimC='1' and igual ='1' else
      B when  Eatual=E and iniciar='1' else
      E when  Eatual=E and iniciar='0' else
      B when  Eatual=F and iniciar='1'else
      F when  Eatual=F and iniciar='0' else
```

```
    A;

-- logica de saída (maquina de Moore)
with Eatual select
  registra <= '0' when A | B | C | D | E | F | J,
              '1' when M,
              '0' when others;
with Eatual select

  zera <=   '0' when A | C | D | E | F | M | J,
              '1' when B | L,
              '0' when others;

with Eatual select
  conta_end <=  '0' when A | B | C | E | F | M | J,
              '1' when D,
              '0' when others;

with Eatual select
    conta_lim <=  '0' when A | B | C | E | F | M | J | D,
                '1' when L,
                '0' when others;

with Eatual select
    zera_lim <=  '0' when A | C | E | F | M | J | D | L,
                  '1' when B,
                 '0' when others;

with Eatual select
  pronto <= '0' when A | B | C | D | M | J,
              '1' when E | F ,
              '0' when others;

with Eatual select
  acertou <='0' when A | B | C | D | F | M | J,
              '1' when E,
              '0' when others;

with Eatual select
  errou <=  '0' when A | B | C | D | E | M | J,
```

```
              '1' when F,
              '0' when others;

  -- saida de depuracao (db_estado)
  with Eatual select
    db_estado <= "0000" when A,       -- 0
                 "1011" when B,       -- B
                 "1100" when C,       -- C
                 "1101" when D,       -- D
                 "1110" when E,       -- E
            "0111" when M,-- M
            "1000" when J,  -- J
            "1001" when L,
            "1111" when F,       -- F
            "0001" when others;  -- 1
end fsm;
```

Diagrama:





O fluxo de dados segue a seguinte lógica:

```
library ieee;
use ieee.std_logic_1164.all;

entity fluxo_dados is
    port (
        clock : in std_logic;
        zeraE : in std_logic;
        limpaR: in std_logic;
```

```
        registraR: in std_logic;
        zeraL: in std_logic;
        contaL: in std_logic;
        contaE : in std_logic;
        escreve: in std_logic;
        botoes : in std_logic_vector (3 downto 0);
        fimE: out std_logic;
        fimL: out std_logic;
        db_tem_jogada: out std_logic;
        db_contagem : out std_logic_vector (3 downto 0);
        db_memoria: out std_logic_vector(3 downto 0);
        db_limite: out std_logic_vector (3 downto 0);
        jogada_feita: out std_logic;
        db_jogada: out std_logic_vector (3 downto 0);
        chavesIgualMemoria: out std_logic;
        enderecoMenorOuIgualLimite: out std_logic;
        enderecoIgualLimite: out std_logic
    );
end entity fluxo_dados;

architecture estrutural of fluxo_dados is

    component contador_163 is
        port (
        clock : in  std_logic;
        clr   : in  std_logic;
        ld    : in  std_logic;
        ent   : in  std_logic;
        enp   : in  std_logic;
        D     : in  std_logic_vector (3 downto 0);
        Q     : out std_logic_vector (3 downto 0);
        rco   : out std_logic
   );
    end component;



    component comparador_85 is
        port (  -- entradas
  i_A3   : in  std_logic;
```

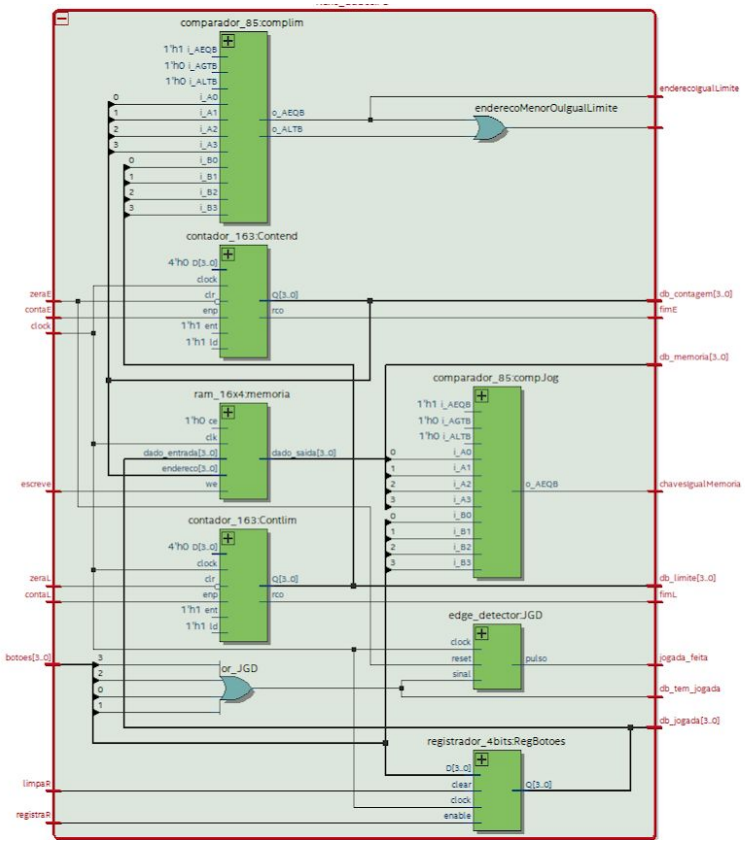
```
i_B3   : in  std_logic;
i_A2   : in  std_logic;
i_B2   : in  std_logic;
i_A1   : in  std_logic;
i_B1   : in  std_logic;
i_A0   : in  std_logic;
i_B0   : in  std_logic;
i_AGTB : in  std_logic;
i_ALTB : in  std_logic;
i_AEQB : in  std_logic;
-- saidas
o_AGTB : out std_logic;
o_ALTB : out std_logic;
o_AEQB : out std_logic
      );
  end component;




  component ram_16x4 is
      port(
        clk          : in  std_logic;
        endereco: in std_logic_vector(3 downto 0);
        dado_entrada: in std_logic_vector(3 downto 0);
        we: in std_logic;
        ce: in std_logic;
        dado_saida: out std_logic_vector(3 downto 0)
      );
  end component;


  component registrador_4bits is
      port (
        clock:  in  std_logic;
        clear:  in  std_logic;
        enable: in  std_logic;
        D:      in  std_logic_vector(3 downto 0);
        Q:      out std_logic_vector(3 downto 0)
      );
```

```
      end component;

    component edge_detector is
        port (
          clock  : in  std_logic;
          reset  : in  std_logic;
          sinal  : in  std_logic;
          pulso  : out std_logic);
        end component;

                                                                signal
great,zeraE_baixo,igual_out,menor,great_o,menor_o,enable_cin,rco_out
c,or_JGD,zeraL_baixo,rco_outL,fim_L: std_logic;
    signal enderecoMenorqueLimite, IgualLimite: std_logic;
    signal s_jogada: std_logic_vector (3 downto 0);
     signal  contador_out,  s_dado,s_endereco,s_lim:  std_logic_vector
(3 downto 0);
    begin


          zeraE_baixo<= not zeraE;
        Contend: contador_163 port map (
            clock => clock,
            clr => zeraE_baixo,
            ld    => '1',
            ent   => '1',
            enp   =>enable_Cin,
            D     => "0000",
            Q     => s_endereco,
            rco   => rco_outC
        );
        fimE<=rco_outC;
        ZeraL_baixo<= not ZeraL;
        Contlim: contador_163 port map (
            clock => clock,
            clr => ZeraL_baixo,
            ld    => '1',
            ent   => '1',
            enp   =>contaL,
            D     => "0000",
```

```
            Q     => s_lim,
            rco   => rco_outL
        );
        fimL<=rco_outL;
        db_limite<=s_lim;
      enable_Cin<=contaE;
      db_contagem<=s_endereco;

      complim: comparador_85 port map (
        i_A3  =>s_endereco(3),
          i_B3   => s_lim(3),
          i_A2   =>s_endereco(2),
          i_B2   => s_lim(2),
          i_A1   =>s_endereco(1),
          i_B1   => s_lim(1),
          i_A0   =>s_endereco(0),
          i_B0   => s_lim(0),
          i_AGTB =>'0',
          i_ALTB => '0',
          i_AEQB => '1',
          -- saidas
          o_ALTB =>enderecoMenorQueLimite,
          o_AEQB =>IgualLimite
    );
    enderecoIgualLimite<=IgualLimite;
           enderecoMenorOuIgualLimite<=   enderecoMenorQueLimite   or
IgualLimite;

        compJog: comparador_85 port map (
            i_A3  =>s_dado(3),
            i_B3   => botoes(3),
            i_A2   =>s_dado(2),
            i_B2   => botoes(2),
            i_A1   =>s_dado(1),
            i_B1   => botoes(1),
            i_A0   =>s_dado(0),
            i_B0   => botoes(0),
            i_AGTB =>'0',
            i_ALTB => '0',
            i_AEQB => '1',
```

```
            -- saidas
            o_AGTB =>great,
            o_ALTB =>menor,
            o_AEQB => igual_out
        );


        memoria: ram_16x4 port map(
          clk=> clock,
            endereco => s_endereco,
            dado_entrada => s_jogada,
            we => escreve,
            ce => '0',
            dado_saida => s_dado
        );
        db_memoria<=s_dado;
        chavesIgualMemoria<=igual_out;

        RegBotoes: registrador_4bits port map(
        clock=>clock,
        clear=>limpaR,
        enable=>registraR,
        D=>botoes,
        Q=>s_jogada);
        db_jogada<=s_jogada;
        or_JGD<=botoes(0) or botoes(1) or botoes(2) or botoes(3);
        JGD: edge_detector port map(
            clock=>clock,
            reset=>zeraE,
            sinal=>or_JGD,
            pulso=>jogada_feita
        );
        db_tem_jogada<=or_JGD;
    end architecture;
```

Diagrama:



E por fim o circuito segue:

```
library ieee;
use ieee.std_logic_1164.all;
entity circuito_exp6 is
    port(
        clock : in std_logic;
        reset : in std_logic;
        iniciar : in std_logic;
        botoes : in std_logic_vector (3 downto 0);
        acertou : out std_logic;
        errou : out std_logic;
        pronto : out std_logic;
        leds: out std_logic_vector (3 downto 0);
        db_limite : out std_logic_vector (6 downto 0);
        db_igual : out std_logic;
        db_contagem : out std_logic_vector (6 downto 0);
        db_memoria : out std_logic_vector (6 downto 0);
        db_estado : out std_logic_vector (6 downto 0);
        db_jogada : out std_logic_vector (6 downto 0);
        db_clock : out std_logic;
```

```
        db_tem_jogada : out std_logic

    );
end entity;

architecture estrutural of circuito_exp6 is
    component fluxo_dados is
        port(
            clock : in std_logic;
        zeraE : in std_logic;
        limpaR: in std_logic;
        registraR: in std_logic;
        zeraL: in std_logic;
        contaL: in std_logic;
        contaE : in std_logic;
        escreve: in std_logic;
        botoes : in std_logic_vector (3 downto 0);
        fimE: out std_logic;
        fimL: out std_logic;
        db_tem_jogada: out std_logic;
        db_contagem : out std_logic_vector (3 downto 0);
        db_memoria: out std_logic_vector(3 downto 0);
        db_limite: out std_logic_vector (3 downto 0);
        jogada_feita: out std_logic;
        db_jogada: out std_logic_vector (3 downto 0);
        chavesIgualMemoria: out std_logic;
        enderecoMenorOuIgualLimite: out std_logic;
        enderecoIgualLimite: out std_logic
        );
    end component;

    component unidade_controle is
        port(
            clock:     in  std_logic;
    reset:     in  std_logic;
    iniciar:   in  std_logic;
    fimC:       in  std_logic;
    fimL:       in  std_logic;
    jogada:   in std_logic;
    enderecoIgualLimite: in std_logic;
```

```
    zera:      out std_logic;
    conta_end:     out std_logic;
    conta_lim: out std_logic;
    zera_lim: out std_logic;
    pronto:    out std_logic;
    db_estado: out std_logic_vector(3 downto 0);
    acertou: out std_logic;
    errou: out std_logic;
    registra: out std_logic;
    igual: in std_logic
        );
    end component;

    component hexa7seg is
        port (
            hexa : in  std_logic_vector(3 downto 0);
            sseg : out std_logic_vector(6 downto 0)
        );
        end component;

                signal    conta4,memo4,estad4,joga4,lim4,botoes_led:
std_logic_vector (3 downto 0);
    signal conta,fim,zeraE,registra,clk,jogada,igual_i: std_logic;
                                                              signal
zeraL,contaL,contaE,fimE,fimL,jogada_feita,chavesIgualMemoria,endere
coIgualLimite: std_logic;
    begin
        leds(0)<=botoes(0);
        leds(1)<=botoes(1);
        leds(2)<=botoes(2);
        leds(3)<=botoes(3);
        clk<=clock;
        FD: fluxo_dados port map(
            clock =>clk,
        zeraE =>zeraE,
        limpaR=> zeraE,
        registraR=>registra,
        zeraL=>zeraL,
        contaL=>contaL,
        contaE =>contaE,
```

```
        escreve=>'1',
        botoes =>botoes,
        fimE=>fimE,
        fimL=>fimL,
        db_tem_jogada=>db_tem_jogada,
        db_contagem =>conta4,
        db_memoria=>memo4,
        db_limite=>lim4,
        jogada_feita=>jogada_feita,
        db_jogada=>joga4,
        chavesIgualMemoria=>chavesIgualMemoria,
        enderecoIgualLimite=>enderecoIgualLimite
        );
        db_igual<=igual_i;
        UC: unidade_controle port map(
            clock=>clock,
        reset=>reset,
        iniciar=>iniciar,
        fimC=>fimE,
        fimL=>fimL,
        jogada=>jogada_feita,
        enderecoIgualLimite=>enderecoIgualLimite,
        zera=>zeraE,
        conta_end=>contaE,
        conta_lim=>contaL,
        zera_lim=>zeraL,
        pronto=>pronto,
        db_estado=>estad4,
        acertou=>acertou,
        errou=>errou,
        registra=>registra,
        igual=>chavesIgualMemoria
        );
        db_clock<=clk;
        HEX1: hexa7seg port map(
            hexa =>conta4,
            sseg =>db_contagem
        );

        HEX2: hexa7seg port map(
```

```
            hexa =>memo4,
            sseg =>db_memoria
        );

        HEX3: hexa7seg port map(
            hexa =>estad4,
            sseg =>db_estado
        );

        HEX4: hexa7seg port map(
            hexa=>joga4,
            sseg=> db_jogada
        );
        HEX5: hexa7seg port map(
            hexa=>lim4,
            sseg=> db_limite
        );
end architecture;
```

2.3) Plano de testes

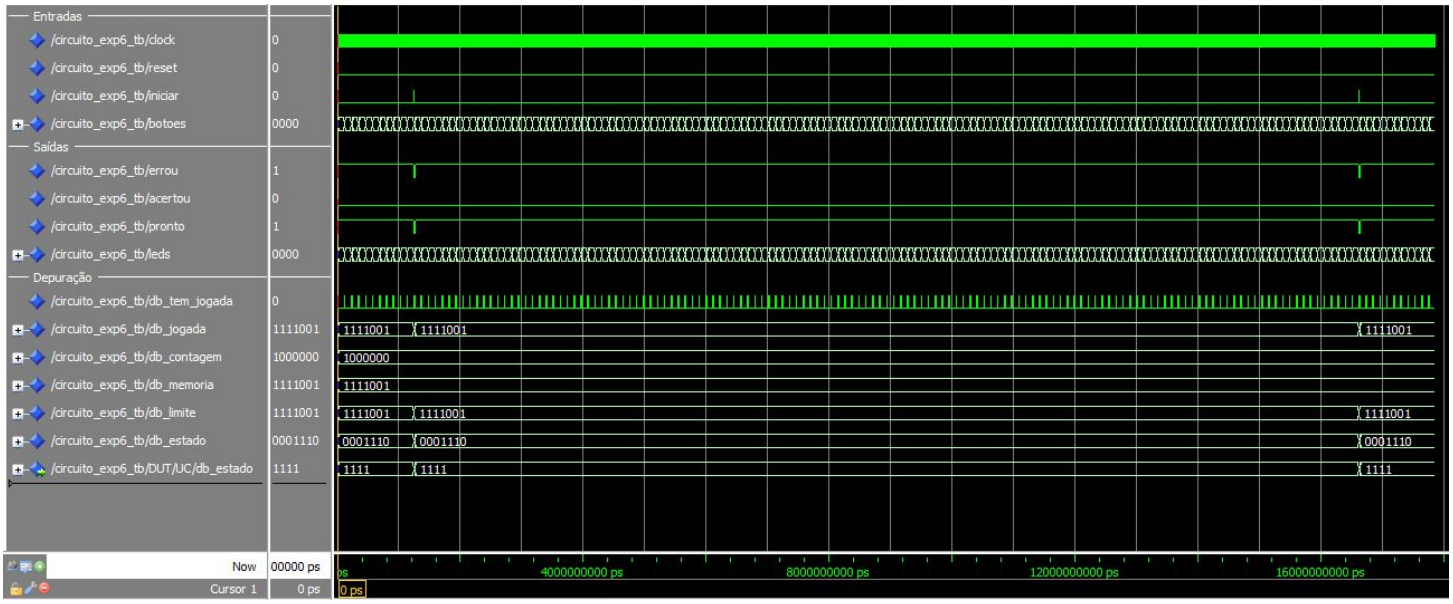
Cenário #1 – Acerto de todas as jogadas				
#	**Operação**	**Sinais de Entrada**	**Resultado Esperado**	**Resultado Observado**
c.i.	**Condições Iniciais**			
1	**Iniciar ciclo**	**iniciar = 1**		
2	**Acertar jogada**	**botoes = 0001**	**igual = 1**	
3	**Acertar jogada**	**botoes = 0001, botoes = 0010**	**igual = 1**	
4	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100**	**igual = 1**	
5	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000**	**igual = 1**	
6	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100**	**igual = 1**	

7	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010**	**igual = 1**	
8	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010, botoes = 0001**	**igual = 1**	
9	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010, botoes = 0001, botoes = 0001**	**igual = 1**	
10	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010, botoes = 0001, botoes = 0001, botoes = 0010**	**igual = 1**	
11	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010, botoes = 0001, botoes = 0001, botoes = 0010, botoes = 0010**	**igual = 1**	
12	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010, botoes = 0001, botoes = 0001, botoes = 0010, botoes = 0010, botoes = 0100**	**igual = 1**	
13	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010, botoes = 0001, botoes = 0001, botoes = 0010, botoes = 0010, botoes = 0100, botoes = 0100**	**igual = 1**	
14	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010, botoes = 0001, botoes = 0001, botoes = 0010, botoes = 0010, botoes = 0100, botoes = 0100, botoes = 1000**	**igual = 1**	
15	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010, botoes = 0001, botoes = 0001, botoes = 0010, botoes = 0010, botoes = 0100, botoes = 0100, botoes = 1000, botoes = 1000**	**igual = 1**	
16	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000,**	**igual = 1**	

		botoes = 0100, botoes = 0010, botoes = 0001, botoes = 0001, botoes = 0010, botoes = 0010, botoes = 0100, botoes = 0100, botoes = 1000, botoes = 1000, botoes = 0001		
17	**Acertar jogada**	**botoes = 0001, botoes = 0010, botoes = 0100, botoes = 1000, botoes = 0100, botoes = 0010, botoes = 0001, botoes = 0001, botoes = 0010, botoes = 0010, botoes = 0100, botoes = 0100, botoes = 1000, botoes = 1000, botoes = 0001, botoes = 0100**	**igual = 1, pronto = 1, acertou = 1**	

Cenário #2 – Erro na 4ª jogada				
#	**Operação**	**Sinais de Entrada**	**Resultado Esperado**	**Resultado Observado**
c.i.	**Condições Iniciais**			
1	**Iniciar Ciclo**	**iniciar = 1**		
2	**Acertar jogadas**	**botoes = 0001**	**igual = 1**	
3		**botoes = 0001, botoes = 0010**	**igual = 1**	
4		**botoes = 0001, botoes = 0010, botoes = 0100**	**igual = 1**	
5	**Errar jogada**	**botoes = 0010**	**pronto = 1, errou =1**	

2.4) Simulação do circuito



2.5) Designação de pinos

Sinal	Pino na Placa DE0-CV	Pino no FPGA	Analog Discovery
CLOCK	**GPIO_0_D13**	**PIN_T22**	**StaticIO – LED – DIO0 Patterns – Clock – 1kHz**
RESET	**GPIO_0_D15**	**PIN_N19**	**StaticIO – Button 0/1 – DIO1**
INICIAR	**GPIO_0_D17**	**PIN_P19**	**StaticIO – Button 0/1 – DIO2**
BOTOES(0)	**GPIO_0_D19**	**PIN_P17**	**StaticIO – Button 0/1 – DIO3**
BOTOES(1)	**GPIO_0_D21**	**PIN_M18**	**StaticIO – Button 0/1 – DIO4**
BOTOES(2)	**GPIO_0_D23**	**PIN_L17**	**StaticIO – Button 0/1 – DIO5**
BOTOES(3)	**GPIO_0_D25**	**PIN_K17**	**StaticIO – Button 0/1 – DIO6**
LEDS(0)	**Led LEDR0**	**PIN_AA2**	**-**
LEDS(1)	**Led LEDR1**	**PIN_AA1**	**-**
LEDS(2)	**Led LEDR2**	**PIN_W2**	**-**
LEDS(3)	**Led LEDR3**	**PIN_Y3**	**-**

PERDEU	**Led LEDR7**	**PIN_U1**	**-**
GANHOU	**Led LEDR8**	**PIN_L2**	**-**
PRONTO	**Led LEDR9**	**PIN_L1**	**-**
db_chavesIgualMemoria	**Led LEDR4**	**PIN_N2**	**-**
db_clock	**Led LEDR5**	**PIN_N1**	**-**
db_tem_jogada	**Led LEDR6**	**PIN_U2**	**-**
db_contagem	**Display HEX0**	**[0] PIN_U21** **[1] PIN_V21** **[2] PIN_W22** **[3] PIN_W21** **[4] PIN_Y22** **[5] PIN_Y21** **[6] PIN_AA22**	**-**
db_memoria	**Display HEX1**	**[0] PIN_AA20** **[1] PIN_AB20** **[2] PIN_AA19** **[3] PIN_AA18** **[4] PIN_AB18** **[5] PIN_AA17** **[6] PIN_U22**	**-**
db_jogada	**Display HEX2**	**[0] PIN_Y19** **[1] PIN_AB17** **[2] PIN_AA10** **[3] PIN_Y14** **[4] PIN_V14** **[5] PIN_AB22** **[6] PIN_AB21**	**-**
db_limite	**Display HEX3**	**[0] PIN_Y16** **[1] PIN_W16** **[2] PIN_Y17** **[3] PIN_V16** **[4] PIN_U17** **[5] PIN_V18** **[6] PIN_V19**	**-**
db_estado	**Display HEX5**	**[0] PIN_N9** **[1] PIN_M8** **[2] PIN_T14** **[3] PIN_P14** **[4] PIN_C1** **[5] PIN_C2** **[6] PIN_W19**	**-**