

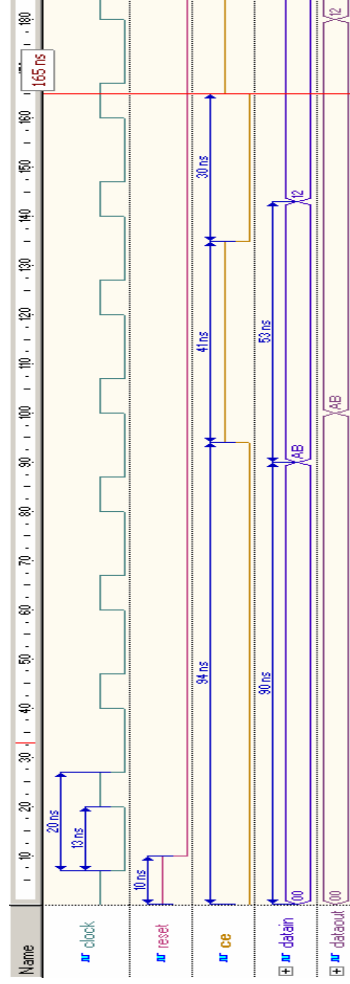
1. [2,5 pontos] Jogo dos 5 erros em VHDL. O código VHDL abaixo contém exatamente 5 falhas, que você deve identificar, deixando claro qual o erro. Os erros são de diversas naturezas: sintaxe, contexto, semântica. Se um mesmo erro acontecer múltiplas vezes, conte apenas um erro para todos os iguais.

```

1. library IEEE;
2.
3. entity p1_oa1_061 is
4.     port(
5.         A1, B2, C3 : in STD_LOGIC;
6.         D1, E2, F3, G4 : out STD_LOGIC
7.     );
8. end p1_oa1_061;
9.
10. architecture p1_oa1_061 of p1_oa1_061 is
11.     signal E : std_logic_vector (2 downto 0);
12.     signal int_E2, int_F3, int_G4 : std_logic;
13.     signal S : out std_logic_vector (2 downto 0);
14. begin
15.     E <= C3 & B2 & D1;
16.     E2 <= int_E2; F3 <= int_F3; G4 <= int_G4;
17.     S <= int_E2 & int_F3 & int_G4 & int_E2;
18.     D1 <= A1 xor (B2 and C3);
19.     with E select
20.         S <= "101" when "111",
21.             "001" when "110" | "100" | "000",
22.             "010" when "011",
23.             "110" when "010",
24.             "011" when others;
25. end p1_oa1_061;

```

2. [3,5 pontos] A Figura representa uma forma de onda gerada pelo testbench de um circuito que possui como entradas os sinais `clock`, `reset` e `ce` e de um bit, um sinal `datain` de 8 bits e como saída um sinal `dataout` de 8 bits. Gere o testbench que produz esta forma de onda. Assuma que nenhum sinal de entrada (exceto o clock) muda após o instante 165ns. Abaixo é dado o início da arquitetura do testbench. Apenas complete o VHDL.



```

1. architecture TB_ARCHITECTURE of reg_ck_rst_ce_tb is
2.     ... -- falta código aqui ...
3.     signal clock : std_logic;
4.     signal reset : std_logic;
5.     signal ce : std_logic;
6.     signal datain : std_logic_vector(7 downto 0);
7.     signal dataout : std_logic_vector(7 downto 0);
8. begin
9.
10.     -- Unit Under Test port map
11.     UUT : reg_ck_rst_ce
12.         port map (
13.             clock => clock,
14.             reset => reset,
15.             ce => ce,
16.             datain => datain,
17.             dataout => dataout
18.         );
19.     -- seu código vai aqui!!
20. end TB_ARCHITECTURE;

```

3. [4 pontos] Implemente em VHDL um circuito que conta o número de entradas consecutivas diferentes em uma linha de dados `DadosIn`, de 8 bits. A cada sinal de relógio o circuito compara a entrada atual (em `DadosIn`) com o valor da entrada anterior neste sinal. Caso estas sejam diferentes, incrementa-se o valor de um contador e coloca-se o valor deste numa saída `Dados_Dif` do circuito, de 12 bits. Senão a saída permanece como está. Adicionalmente, cada vez que o valor anterior for igual a `x"FF"`, a saída `Detectou_FF` vai para '1'. Esta saída vai para '0' apenas quando um novo dado, diferente de `x"FF"` for recebido. Para facilitar, a entidade do circuito é dada abaixo. Assuma o seguinte:

- Existe uma entrada (`x"00"`) que nunca ocorrerá;
- O sistema é sensível à borda de subida do sinal clock e o reset é assíncrono.

```

1. library IEEE;
2. use IEEE.STD_LOGIC_1164.all;
3. use IEEE.STD_LOGIC_unsigned.all;
4.
5. entity Q3_P1_oa1_061 is
6.     port(
7.         clock : in STD_LOGIC;
8.         reset : in STD_LOGIC;
9.         DadosIn : in STD_LOGIC_VECTOR(7 downto 0);
10.        Detectou_FF : out STD_LOGIC;
11.        Dados_Dif : out STD_LOGIC_VECTOR(11 downto 0)
12.    );
13. end Q3_P1_oa1_061;

```

GABARITO

1 [2,5 pontos]

Jogo dos 5 erros em VHDL. O código VHDL abaixo contém exatamente 5 falhas, que você deve identificar, deixando claro qual o erro. Os erros são de diversas naturezas, sintaxe, contexto, semântica.

1. library IEEE;
2. use IEEE.STD_LOGIC_1164.all; -- 1) esta linha está faltando, para declarar a existência dos tipos std_logic e std_logic_vector.
- 3.
4. entity p1_oal_061 is
5. port(
6. A1, B2, C3 : in STD_LOGIC;
7. D1, E2, F3, G4 : out STD_LOGIC
8.);
9. end p1_oal_061;
- 10.
11. architecture p1_oal_061 of p1_oal_061 is
12. signal E : std_logic_vector (2 downto 0);
13. signal int_E2, int_F3, int_G4 : std_logic;
14. signal S : out std_logic_vector (2 downto 0); -- 2) sinais internos não podem ser declarados como de entrada ou saída
15. begin
16. E <= C3 & B2 & D1; -- 3) D1 é saída, não pode ser lida
17. E2 <= int_E2; F3 <= int_F3; G4 <= int_G4;
18. S <= int_E2 & int_F3 & int_G4 & int_E2; -- 4) S é de 3 bits, não se pode atribuir um valor de 4 bits a ele.
19. D1 <= A1 xor (B2 and C3);
20. with E select
21. S <= "101" when "111", -- 5) há duas atribuições distintas em paralelo para o sinal S, aqui e na linha 18.
"001" when "110" | "100" | "000",
"010" when "011",
"110" when "010",
"011" when others;
- 22.
- 23.
- 24.
- 25.
26. end p1_oal_061;

2 [3,5 pontos]

A Figura representa uma forma de onda gerada pelo testbench de um circuito que possui como entradas os sinais `clock`, `reset` e `ce` de um bit, um sinal `datain` de 8 bits e como saída um sinal `dataout` de 8 bits. Gere o testbench que produz esta forma de onda. Assuma que nenhum sinal de entrada (exceto o clock) muda após o instante 165ns. Abaixo é dado o início da arquitetura do testbench. Apenas complete o VHDL.

1. architecture TB_ARCHITECTURE of reg_ck_rst_ce_tb is
2. ... -- falta código aqui ...
3. signal clock : std_logic;
4. signal reset : std_logic;
5. signal ce : std_logic;
6. signal datain : std_logic_vector(7 downto 0);
7. signal dataout : std_logic_vector(7 downto 0);
8. begin
- 9.
- 10.
11. UUT : reg_ck_rst_ce
12. port map (
13. clock => clock,
14. reset => reset,
15. ce => ce,
16. datain => datain,
17. dataout => dataout
18.);

19. -- seu código vai aqui!!
20. reset <= '1', '0' after 10ns;
- 21.
22. process
23. begin
24. Clock <= '1', '0' after 7ns;
25. wait for 20ns;
26. end process;
- 27.
28. ce <= '0', '1' after 94ns, '0' after 135ns, '1' after 165ns;
- 29.
30. datain <= x"00", x"AB" after 90ns, x"12" after 143ns;
- 31.
32. end TB_ARCHITECTURE;

3 [4 pontos]

Implemente em VHDL um circuito que conta o número de entradas consecutivas diferentes em uma linha de dados `DadosIn`, de 8 bits. A cada sinal de relógio o circuito compara a entrada atual (em `DadosIn`) com o valor da entrada anterior neste sinal. Caso estas sejam diferentes, incrementa-se o valor de um contador e coloca-se o valor deste numa saída `Dados_Dif` do circuito, de 12 bits. Senão a saída permanece como está. Adicionalmente, cada vez que o valor anterior for igual a x"FF", a saída `Detectou_FF` vai para '1'. Caso contrário, esta saída fica em '0'. Para facilitar, a entidade do circuito é dada abaixo.

1. library IEEE;
2. use IEEE.STD_LOGIC_1164.all;
3. use IEEE.STD_LOGIC_unsigned.all;
- 4.
5. entity Q3_P1_oal_061 is
6. port(
7. clock : in STD_LOGIC;
8. reset : in STD_LOGIC;
9. DadosIn : in STD_LOGIC_VECTOR(7 downto 0);
10. Detectou_FF : out STD_LOGIC;
11. Dados_Dif : out STD_LOGIC_VECTOR(11 downto 0)
12.);
13.
14. end Q3_P1_oal_061;
15. architecture Q3_P1_oal_061 of Q3_P1_oal_061 is
16. signal Dado_lido : std_logic_vector (7 downto 0);
17. signal Dados_Dif_int : std_logic_vector (11 downto 0);
18. begin
19. Dados_Dif <= Dados_Dif_int;
20. process (reset,clock)
21. begin
22. if reset='1' then
23. Dado_lido <= (others=>'0');
24. Dados_Dif_int <= (others=>'0');
25. Detectou_FF <= '0';
26. elsif clock'event and clock='1' then
27. Dado_lido <= DadosIn;
28. if Dado_lido/=DadosIn then
29. Dados_Dif_int <= Dados_Dif_int +1;
30. end if;
31. if Dado_lido=x"FF" then
32. Detectou_FF <= '1';
33. else Detectou_FF <= '0';
34. end if;
35. end if;
36. end process;
37. end Q3_P1_oal_061;