Trabalho Prático 2 – Familiarização com VHDL
Terça-feira, 03 de Agosto de 2022

Objetivos:

Familiarização com a implementação de circuitos combinacionais em VHDL e seu teste em FPGA. Acesso ao LabRemoto.

Projetos:

Crie os projetos abaixo no Vivado, seguindo o fluxo de projeto visto em aula, e teste no LabRemoto. Use os slides das aulas e os materiais de referência para fazer a codificação em VHDL.

1) Circuito do projeto do estacionamento

$$c0 = \overline{s0} \cdot \overline{s1} \cdot s2 + \overline{s0} \cdot s1 \cdot \overline{s2} + s0 \cdot \overline{s1} \cdot \overline{s2} + s0 \cdot s1 \cdot s2$$
$$c1 = \overline{s0} \cdot s1 \cdot s2 + s0 \cdot \overline{s1} \cdot s2 + s0 \cdot s1 \cdot \overline{s2} + s0 \cdot s1 \cdot s2$$

Inputs			Outputs	
s0	**s1**	**s2**	**c1**	**c0**
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

2) Decodificador do display de 7 segmentos (exercício 4)

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

entity seven_seg_decoder is
  port ( bcd   : in  std_logic_vector(3 downto 0);
         blank : in  std_logic;
         seg   : out std_logic_vector(7 downto 1) );
end entity seven_seg_decoder;


architecture behavior of seven_seg_decoder is
  signal seg_tmp : std_logic_vector(7 downto 1);
begin
  with bcd select
    seg_tmp <= "0111111" when "0000",   -- 0
               "0000110" when "0001",   -- 1
               "1011011" when "0010",   -- 2
               "1001111" when "0011",   -- 3
               "1100110" when "0100",   -- 4
               "1101101" when "0101",   -- 5
               "1111101" when "0110",   -- 6
               "0000111" when "0111",   -- 7
               "1111111" when "1000",   -- 8
               "1101111" when "1001",  -- 9
               "1000000" when others;  -- "-" for invalid code

  seg <= "0000000" when blank = '1' else
         seg_tmp;
end architecture behavior;
```

3) Somador 1 bit

```
----------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Somador_1bit is

    Port ( A    : in  STD_LOGIC;
           B    : in  STD_LOGIC;
           Cin  : in  STD_LOGIC;
           S    : out STD_LOGIC;
           Cout : out STD_LOGIC);

end Somador_1bit;
architecture Behavioral of Somador_1bit is
begin

S    <= A xor B xor Cin;
Cout <= (A and B) or (A and Cin) or (B and Cin);

end Behavioral;
```

4) Somador 3 bits

```
entity Somador_3bits is

 Port ( Ai    : in  STD_LOGIC_VECTOR (2 downto 0);
        Bi    : in  STD_LOGIC_VECTOR (2 downto 0);
        Cin   : in  STD_LOGIC;
        Sum   : out STD_LOGIC_VECTOR (2 downto 0);
        Cout  : out STD_LOGIC);
end Somador_3bits;

architecture Behavioral of Somador_3bits is

signal i : STD_LOGIC_VECTOR (2 downto 0);-----------------------------------


component Somador_1bit
Port ( A    : in  STD_LOGIC;
       B    : in  STD_LOGIC;
       Cin  : in  STD_LOGIC;
       S    : out STD_LOGIC;
       Cout : out STD_LOGIC);

end component;--------------------------------------------------------------

begin--

bit0: Somador_1bit port map (Ai(0), Bi(0), Cin,  Sum(0), i(0));

bit1: Somador_1bit port map (Ai(1), Bi(1), i(0), Sum(1), i(1));

bit2: Somador_1bit port map (Ai(2), Bi(2), i(1), Sum(2), Cout;

end Behavioral;_____________________________________________________________
```

Vistos:

1) Mostrar *testbenches* e simulações de cada um dos projetos (1 ponto para cada).
2) Mostrar o funcionamento de cada um dos projetos na placa (1,5 ponto para cada).

Bom trabalho!